Bernd-Helge Schäfer was born in Mainz at the river Rhine in 1979. Mr. Schäfer studied informatics at the *University of Kaiserslautern*. Since 2003 he was student in the PhD Programme of the Department of Computer Sciences. In 2005 Mr. Schäfer graduated in the context of the cooperation between the *Robotics Research Lab (University of Kaiserslautern)* and the *Unmanned Vehicle Centre* at the *Royal Military Academy Belgium*. Until 2010, Mr. Schäfer worked on design methodology as a research assistant at the *Robotics Research Lab*.

In this doctoral thesis, a novel design approach is proposed which combines action orientation and perception orientation into one coherent methodology. More precisely, design schemata for representation, translation, and fusion of environmental information are developed which establish thorough abstraction mechanisms between components. Furthermore, a set of design guidelines for the application of the schemata is proposed. The explicit introduction of abstractions particularly supports extensibility and scalability of robot control systems by design. The proposed schemata furthermore allow for the fine-grained reuse of software components. The validity of the methodology is shown in a large-scale application study on the off-road platform RAVON.

ISBN 978-3-86853-865-6

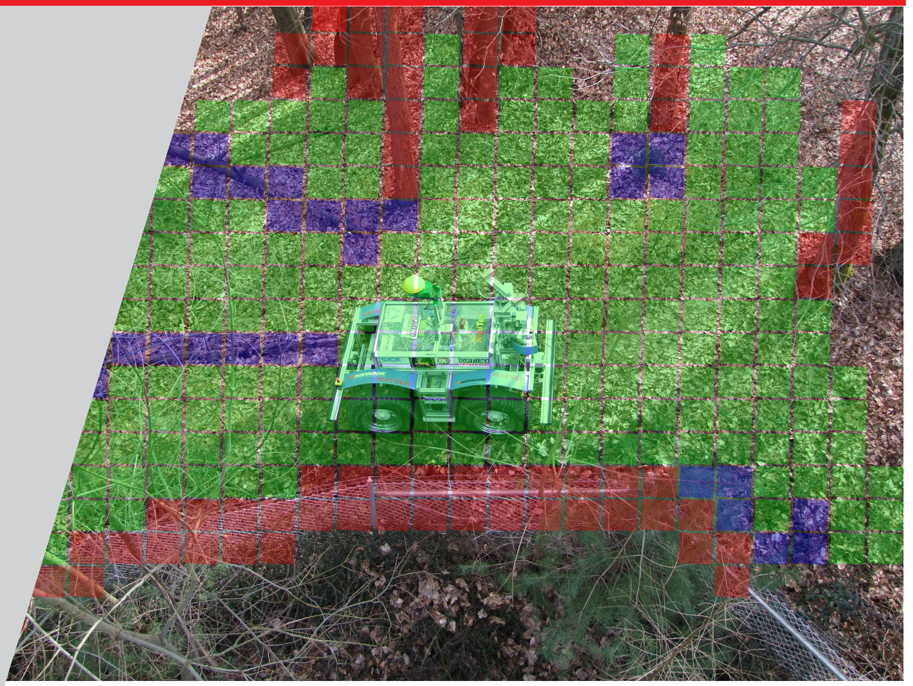
Verlag Dr. Hut

9 783868 538656

B.-H. Schäfer

Robot Control Design Schemata – Applications in Off-road Robotics

Technische Universität Kaiserslautern



Bernd-Helge Schäfer

Robot Control Design Schemata and their Application in Off-road Robotics

RRLAB The Robotics Research Lab

Control System Design Schemata and their Application in Off-road Robotics

Dipl.-Inf. Bernd-Helge Schäfer

Vom Fachbereich Informatik der

Technischen Universität Kaiserslautern

zur Verleihung des akademischen Grades

Doktor der Ingenieurwissenschaften (Dr.-Ing.)

genehmigte Dissertation.

Bibliografische Information der Deutschen Nationalbibliothek
Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über
http://dnb.d-nb.de abrufbar.

ISBN 978-3-86853-865-6

Zur Begutachtung eingereicht am:	24. November 2010
Datum der wissens. Aussprache:	24. Februar 2011
Vorsitzender:	Prof. Dr. Stefan Deßloch
Erster Berichterstatter:	Prof. Dr. Karsten Berns
Zweiter Berichterstatter:	Prof. Dr. Frank Kirchner
Dekan:	Prof. Dr. Arnd Poetzsch-Heffter
Zeichen der TU im Bibiliotheksverkehr:	D 386

Acknowledgements

First of all I would like to thank Prof. Dr. Karsten Berns for giving me the opportunity to study one of the most interesting subjects in mobile robotics. He has always taken his time for talking over my ideas and worries and contributed essentially to my work with inspiring visions of what would be possible next. Furthermore, I want to thank Prof. Dr. Frank Kirchner for reviewing in particular the software-technological aspects of this Doctoral Thesis.

For supporting my application for the PhD Programme, I want to thank Prof. Dr. Zimmermann, Prof. Dr. Wiehagen, and Prof. Dr. Cullum as well as their research assistants Andreas Metzger, Thorsten Michels, and Matthias Redenbach. During the initial phase in the PhD Programme, Prof. Dr. Stefan Deßloch was a reliable contact for many of my questions. Therefore, I was very happy that he also agreed to chair my scientific defence. Thanks for the wide scientific and administrative support in that time. Further thanks go to Prof. Dr. Klaus Madlener for challenging my pride and to Prof. Dr. Jürgen Avenhaus for providing me with the self-confidence I needed. Your theory courses – and in particular some of the side notes – have been a source of inspiration for both my work and my living.

I would like to thank my colleagues at the Robotics Research Lab for the unique amicable working environment, the many discussions, hands given, and parties celebrated throughout my whole time in Kaiserslautern. Special thanks in that context go to Tim Braun and Martin Proetzsch with whom I have been working on the ravon platform from the very beginning. The matchless team work and the impulses whenever I thought to have reached a dead-end in my scientific work always gave me ideas of how to continue. Tim, thanks for sharing your rich knowledge about optics, cameras, and image processing. Martin, thanks for listening to every esoteric idea I had for restructuring the ravon's control system ("Ah, another one of these 'minor' changes, right? ... "). In 2008 our team was significantly enriched by Christopher Armbrust, Tobias Föhst and Alexander Renner who have done a great job learning the ropes and contributing to the development of ravon. Special thanks go to Tobias who spent many afternoons discussing aspect orientation and implementing parts of the reference implementation which is subject to this Doctoral Thesis. The same is for Christopher who helped me to cast into reality my sketchpad-status concept for passage detection.

Furthermore, I want to thank Tobias Luksch for the numerous discussions on software design, control methodology, and various other topics. Thanks as well for the (perhaps mutual) ideational support during the last phase of this work. Special thanks also go to Carsten Hillenbrand who encouraged me to design away many problems by sound mechanical methods. Thank you for the long discussions of construction matters and the access to basic skills and best practices in mechanics and electronics. In that field Alex was also a great help in particular when it was about putting things together. Thanks for taking over several duties for which I was often not patient enough. Talking about mechanics, further thanks go to the workshop of the University of Kaiserslautern for helping me to cast my ideas into reality. From Mr. Dick and Mr. Dunkel I learnt a lot with respect to the production-relevant issues of mechanics which assured that the pieces I designed actually fit together in the end. Thanks for the interest in my work and the effort to get the parts produced as unbureaucratically as possible.

The calm daily life of a researcher is frequently interrupted by administrative challenges. At university booking conference trips and ordering components is related to a lot of paperwork. Fortunately, our secretary Rita Broschart, our technician Lothar Gauß, or the dean's secretaries Marianne Stumpf and Gabriele Sakdapolrak always knew what to do. Thanks for all the help.

In the final phase of writing this thesis, I had a lot of support from the whole workgroup. Thanks to everybody who took the time for reading my work and providing me with vital information on how to improve it. Special thanks in that context go to the early birds on the day of submission - Karsten Berns, Lisa Kiekbusch, and Martin Proetzsch - who quickly proofread my nightly improvements. This time I really needed the night shift to finish before the dead-line and I am grateful for all the last-minute feedback. That way, many embarrassing typos did not make it into the final manuscript.

Apart from the members of the RRLab, I got a lot of support from researchers of other workgroups. Special thanks go to my fellow student Jean-Marie Gaillourdet for reviewing and discussing the software technology sections with me. Further thanks go to my former math tutor Dr. Joseph Tadjuidje Kamgaing for helping me improve the probabilistic issues related to the abstract fusion concept.

Last but not least I want to thank my significant other, Michaela Leroch, my family and friends who have always been there for me whenever I needed them. In particular Michaela had to suffer from my growing nervous during the final phase of this work. Thanks for everything. I would not have made it without you.

This thesis was funded by the PhD Programme of the Department of Computer Sciences at the University of Kaiserslautern. Research in off-road robotics on ravon is furthermore sponsored by many small and medium-size companies. This support is gratefully acknowledged.

Further Credits

The vast amount of technical details related to creating a complex multi-sensor robotic vehicle suitable for off-road environments would have made it difficult to reach the current level of sophistication without additional workforce. Therefore, I would like to thank the students I tutored during my PhD for their dedication. The following project and diploma theses represent a crucial part of the technical basis of this work.

Obstacle Tracking in Rough Outdoor Terrain
Boris Gilsdorf, November 2006

Forward und Reverse Engineering in der Robotik
Carsten Knapwost, May 2007

Vision-based Self-localization for Mobile Robots
Petra Hahnfeld, June 2007

Laser-based Obstacle Detection in Off-road Terrain
Andreas Hach, July 2007

GPU Stereo Vision
Sebastian Prehn, November, 2007

Laser-based Vegetation Discrimination in Rough Terrain
Sebastian Bachtler, December 2007

Water Detection for Mobile Off-road Robotics
Alexander Renner, June 2008

Special thanks go to Ulf Stocker and his tutor Jan Koch who laid the groundwork for the sector maps that became an important building block in my representation scheme.

Datenabstraktion und Integration von Distanz-Sensorsystemen auf mobilen Robotern
Ulf Stocker, September 2007

Abstract

In robotics, information is often merely regarded as a means to an end, which finds its expression in lacking schematisation of information modelling and deployment. The question of how to structure information and how to bridge the semantic gap between different levels of abstraction in a uniform way is still widely regarded as a technical issue. Ignoring these challenges appears to lead robotics into a similar stasis as experienced in the software industry of the late 1960s. From the beginning of the software crisis until today, numerous methods, techniques, and tools for managing the increasing complexity of software systems have evolved. The attempt to transfer several of these ideas towards applications in robotics yielded various control architectures, frameworks, and process models. These attempts mainly provide modularisation schemata which suggest how to decompose a complex system into less and less complex subsystems. The schematisation of representation and information flow however is mostly ignored.

In this work, a set of design schemata is proposed in the attempt to strike a balance between prescription and flexibility. These schemata are embedded into an action/perception-oriented design methodology which promotes thorough abstractions to separate distinct levels of control. Action-oriented design decomposes a control system top-down and sensor data is extracted from the environment as required. This comes with the problem that partial results are often computed several times for slightly different purposes as information is often condensed in a premature fashion. That way, sensor processing is dependent on the control system design resulting in a monolithical system structure with limited options for reusability. In contrast, perception-oriented design constructs control systems bottom-up starting with the extraction of environment information from sensor data. The extracted entities are placed into structures which evolve with the development of the sensor processing algorithms. The control system thus has to pick relevant information from a representation that is tailored towards sensor data evaluation rather than control. In consequence, the control system is strictly dependent on the sensor processing algorithms which again results in a monolithic system. In their particular domain, both design approaches have great advantages but fail to create inherently modular systems.

The design approach proposed in this work combines the strengths of action orientation and perception orientation into one coherent methodology without inheriting their weaknesses. More precisely, design schemata for representation, translation, and fusion of environmental information are developed which establish thorough abstraction mechanisms between components. The explicit introduction of abstractions particularly supports extensibility and scalability of robot control systems by design. The proposed schemata furthermore allow for the fine-grained reuse of software components. En passant, this methodology also solves the challenge of integrating third-party components, as thorough abstractions are an inherent part of the design approach.

Contents

1. Introduction

1.1 Motivation

The world of today is confronted with an increasing ecological, economical, and political complexity with interdependencies which are difficult to follow and to safeguard.

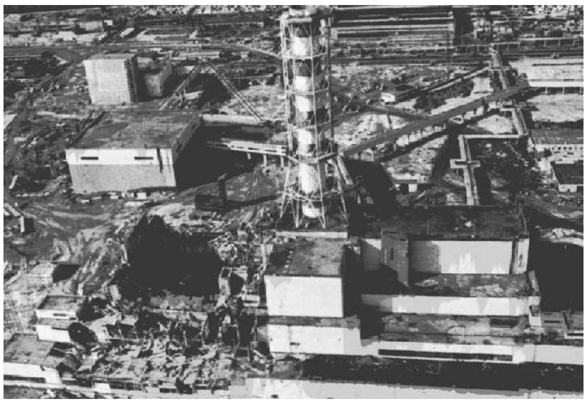
Every year natural catastrophes are registered to cause numerous human casualties and rising economic damage all over the world [Blake 07]. Even years after severe incidents clearing efforts are often still in progress. For instance, many residential quarters destroyed by Hurricane Katrina in 2005 were still uninhabitable two years after the incident. Scientists argue that – through climate change – frequency and intensity of hurricanes will increase significantly [Emanuel 05]. To shorten time to reconstruction, mobile robots could provide logistics support and assist in clearance duties.

Critical accidents in chemical or nuclear plants represent a further threat for both living creatures and the environment. Interwoven procedures, insufficient or inapt maintenance and human failure render such incidents rather a question of when then whether. In 1986 a series of malfunctions in combination with disastrous decisions lead to the largest nuclear accident ever: the explosion of reactor four (see Figure 1.1a) at Chernobyl (former Soviet Union) power plant [CFEG 06]. Over one hundred cases of the ARS[1] have been recorded among the 237 staff members and emergency workers alone. In the attempt to put a teleoperated robot for mapping purposes into the disaster area also required humans to expose themselves to high radiation as no autonomous navigation support was available [Abouaf 98]. In case of emergency, mission operators could deploy mobile platforms to gather valuable information on survivors' locations, infrastructure damage, and contamination distribution. On the basis of such data, mission plans and equipment for task forces can be optimised.

Apart from the direct impact and urgent response casualties, radioactive fall-out contaminated six million hectares of highly fire-prone wood land [Dusha-Gudym 05]. Several hundred wildfires occur every year putting in danger operation forces and affecting life in the proximity through radioactive smoke particles which may travel hundreds of miles

[1]Acute Radiation Syndrome (ARS)

depending on weather conditions. Note that besides dry seasons this problem is further intensified by the effects of radioactivity onto plant life. The radiation of surface deposited materials results in a high quote of dying trees. Mobile robotic platforms might be used to support fire brigades and aid organisations in terms of logistics, routine patrols in endangered areas, and reconnaissance tasks at dangerous spots. Systematic deployment could help operation controllers to prevent large-scale catastrophes, concentrate missions towards hot spots and minimise danger for action forces through enhanced information availability.


(a) Chernobyl disaster 1986 [CFEG 06]

(b) Terror threat with planes (Photo courtesy of Greenpeace)

Figure 1.1: Reactor four of Chernobyl nuclear plant exploded in 1986 (a). Frame captured from the video "Nuclear Power and Terrorism" (`http://www.youtube.com/`) by Greenpeace illustrating the potential threat of a passenger plane crashing into a nuclear power plant (b).

After the 9/11 attacks on the World Trade Center (New York / USA), chemical and nuclear facilities are feared to be future targets of terrorist attacks despite additional security measures [IAEA 07]. In November 2001, Mohammed ElBaradei – at that time director general of the IAEA[2] – stated that "the ruthlessness of such assaults creates a new dimension in the fight against terrorism" [IAEA 02]. In the context of such statements, Greenpeace published a dossier [Greenpeace 06] which deals with the potential impact of a passenger plane crashing into a British nuclear plant (see Figure 1.1b). The consequences are claimed to exceed even those of the Chernobyl disaster.

Further fields of application can be found in the military domain. In particular, tasks like assuring camp security, reconnaissance or surveillance missions as well as logistics are of greatest interest. Warfare as such is today not the main cause of casualties in troops of developed countries. Rather the protection of logistics routes and long lasting peace missions which involve local authorities and personal intensify the immanent threat through suicide bombing. Difficult to anticipate, this type of attack has become a major reason for loss of lives in the fight against terrorism. Therefore, the transformation processes of armed forces world wide focus on the involvement of robotic systems for perilous duties [Dep 04, Fitschen 07]. For several years the U. S. American and the European Forces

[2]International Atomic Energy Agency (IAEA) – `http://www.iaea.org`
Mohammed ElBaradei was Director General of IAEA from 1997 until 2009.

formulate their needs in terms of scenarios for large-scale robot competitions like the *DARPA GC*[3] (see Appendix B.4) and the *M-ELROB*[4] (see Appendix B.5).

The long-term objective for all application fields mentioned above is the development of fully autonomous systems. These promise to prevent long and cost-intensive training of specialised personal and are in principle robust towards loss of communication. In fact, the *ELROB 2008* impressively showed the superiority of fully autonomous systems as many teams banking on teleoperation had to give up due to telemetry problems.

Autonomous navigation in harsh, unknown environments poses many unsolved scientific questions; from obstacle detection over terrain representation towards the selection of adequate actions to master critical driving situations. The diversity of possible hazards appears sheer unlimited, visible range may be obstructed by high vegetation or terrain clutter, dead ends may not be anticipated until the vehicle is actually stuck, and narrow passages can make manoeuvring virtually impossible. Map material may be unavailable or outdated and intensive canopy might limit the feasibility of remote sensing with satellites or drones. This lack of a priori knowledge results in mission plans which are based on vague educated guesses rather than solid information. On the one hand, the crucial requirement for robots in such scenarios is the capability to remain manoeuvrable under all conditions. On the other hand, performance factors like the time to the target or distance to travel are of significant importance as well. To balance between these needs, a thorough design methodology for navigational competences is mandatory. This Doctoral Thesis evolves an extensible approach towards the design of complex robot control systems as required for manoeuvrability in unknown, harsh, vegetated, and narrow environments.

1.2 Scientific Contribution of this Doctoral Thesis

Robotics is an interdisciplinary field which requires competences from mechanical engineering, electrical engineering, as well as informatics. Each of these disciplines has a different perspective on robotic development which is highly dependent on the particular educational background. Therefore, the term *scientific contribution* is highly ambivalent and disputed among roboticists. To rate contributions to a field of research as diverse as robotics, the relationship between the contributing discipline and the scientific field has to be defined.

Informatics in Robotics

The author considers informatics as the discipline which deals with modelling, abstraction, and transformation of information with respect to a given context. In that sense, informatics only gains relevance linked to a field of application and not on its own behalf. Information scientific contributions to robotics should therefore deal with the systematic design of information models, suitable abstractions of these models for particular purposes, and the specification of transformations to deduce the latter from the former.

However, the complexity of each detail in robot development seems to keep robotics from actually accepting these issues as scientific core challenges. Information is often merely regarded as a means to an end, which finds its expression in lacking schematisation of

[3]DARPA Grand Challenge (DARPA GC) – `http://www.darpa.mil/`

[4]Military European Land Robot Trial (M-ELROB) – `http://www.m-elrob.eu/`

information modelling and deployment. The question of how to structure information and how to bridge the semantic gap between different levels of abstraction in a uniform way is still widely regarded as a technical issue. Ignoring these challenges appears to lead robotics into a similar stasis as experienced in the software industry of the late 1960s. From the beginning of the software crisis until today, numerous methods, techniques, and tools for managing the increasing complexity of software systems have evolved. The attempt to transfer several of these ideas towards applications in robotics yielded various control architectures, frameworks, and process models. As their equivalents in software engineering, these attempts mainly provide modularisation schemata which suggest how to decompose a complex system into less and less complex subsystems. Furthermore, solutions for the problem of condensing conflicting control commands into a valid set of actuator inputs (i. e. a fusion scheme for control flow) are a major focus. The schematisation of representation and information flow however is mostly ignored. One reason for this may be the opinion that schemata would impair development flexibility [Matarić 97]. Another, that prescription is difficult to make because applications may be too diverse to define a common basis [Albus 93]. On the other hand, the prescription of schematisation may be the groundwork for the management of complex robotic systems due to increased reusability, extensibility, and maintainability. In contrast to the first claim, schematisation thus actually promotes flexibility as subsystems can more strictly be separated from one another.

Thesis

In this work, a set of design schemata is proposed in the attempt to strike a balance between prescription and flexibility. These schemata are embedded into an action/perception-oriented design methodology which promotes thorough abstractions to separate distinct levels of control. Action orientation (See [Arkin 98] pp. 265) is a concept from robot vision which aims at tailoring detection mechanisms towards the motivations and intentions of a robotic system in contrast to solving the *General Vision Problem*[5]. Action-oriented design decomposes a control system top-down and sensor data is extracted from the environment as required. This comes with the problem that partial results are often computed several times for slightly different purposes as information is often condensed in a premature fashion. That way, sensor processing is dependent on the control system design resulting in a monolithic system structure with limited options for reusability. In contrast, perception-oriented design constructs control systems bottom-up starting with the extraction of environment information from sensor data. The extracted entities are placed into structures which evolve with the development of the sensor processing algorithms. The control system thus has to pick relevant information from a representation that is tailored towards sensor data evaluation rather than control. In consequence, the control system is strictly dependent on the sensor processing algorithms which again results in a monolithic system. In their particular domain, both design approaches have great advantages but fail to create inherently modular systems.

The design approach proposed in this work combines the strengths of action orientation and perception orientation into one coherent methodology without inheriting their weaknesses. More precisely, design schemata for representation, translation, and fusion of environmental

[5]General Vision Problem → The attempt to find and identify *all* objects in an image.

information are developed which establish thorough abstraction mechanisms between components. The explicit introduction of abstractions particularly supports extensibility and scalability of robot control systems by design. The proposed schemata furthermore allow for the fine-grained reuse of software components. En passant, this methodology also solves the challenge of integrating third-party components, as thorough abstractions are an inherent part of the design approach.

1.3 Structure

This Doctoral Thesis is organised in two major parts. The design methodology and the theory of respective design schemata shall be introduced in Part I. In Part II, the validity of the proposed approach will be proven in an application study on a concrete robotic platform.

In Chapter 2, the state of the art in robot architecture is discussed with a strong focus towards applications in mobile off-road robotics. In Chapter 3, challenges in robot control system design are formulated, followed by a brief excursus on architecture in "natural mobile systems". On the basis of the design ideas deduced from nature, an action/perception-oriented design methodology forming the frame of this work is developed. The design schemata are evolved in Chapter 4. First of all, the schematisation of information representation is discussed. In this context, the explicit introduction of thoroughly defined levels of abstraction is promoted. For that purpose, the semantics of required and generated information for each component is specified in a symbolic fashion. That way, information used for a particular task can be handled in natural way. In order to migrate information between different levels of abstraction, the semantics of the source component has to be translated to the semantics of the target component. A configurable procedure for this translation is proposed in terms of an abstraction scheme. The transfer of information between different levels of abstraction allows for the gradual bridging of semantic gaps in robot control systems. Furthermore, this capability establishes a formal way to lift semantically incompatible information from different sources onto a common level of abstraction. Provided a common semantics, information can be combined in a generic fashion using the proposed fusion scheme. The author argues that abstract fusion should be preferred over fusion of raw sensor data as this mechanism is less vulnerable to noisy data. Chapter 5 summarises the key challenges addressed by the proposed design methodology and the related schemata. Several practical facets of the outlined theory are discussed to evolve more concrete ideas of the abstract concepts. A compact reference manual of design guidelines concludes this chapter and Part I of this work.

Part II of this work can be regarded as a large-scale experiment to prove the applicability of the proposed design methodology using the respective schemata on a concrete robotic platform. The application study carried out in this context deals with the development of a modular navigation system for the off-road robot RAVON. In the first chapter of Part II, a brief overview of the experimental platform is provided. This comprises the mechatronics of the vehicle, the deployed control concept, as well as the architectural foundations. In Chapters 8 through 10, the navigation system is step-by-step evolved with a focus on the application of the particular design schemata and guidelines. In Chapter 8, the control system is modularised in an action-oriented fashion yielding components and interface specifications according to the proposed representation scheme. After that, the sensor processing facilities are designed in a perception-oriented manner in Chapter 9. The representation specifications of both design procedures are the input to the third design step, the aspect-oriented configuration of the semantic abstraction. This configuration defines how the semantics on different levels of abstraction are to be translated to bridge semantic gaps in the control system. As already alluded above, the designed navigation system was technically realised on the off-road platform RAVON. In Chapter 11, a series of experiments is presented to prove the applicability of the proposed methodology for the design of complex robotic systems. Finally, Chapter 12 provides a summary of the achievements of this work. The discussion of the scientific results and an outlook on future perspectives conclude this Doctoral Thesis.

Part I

Design in Autonomous Off-road Robotics

2. Design Schemata in Autonomous Off-road Navigation

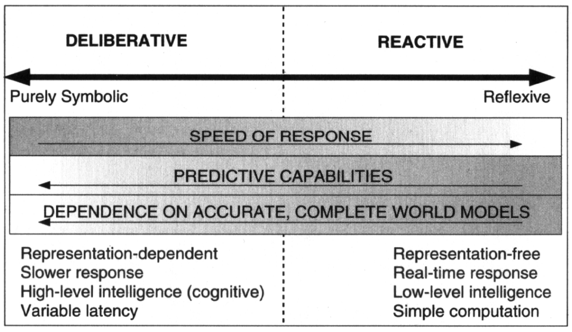
The work at hand spans from design schematisation on a theoretical level towards practical experiments on a complex real-life off-road robot. Therefore, related work shall be presented in a distributed fashion. This chapter is dedicated to the methodic level of related approaches. In particular architectural properties will be highlighted and discussed in the context of how these support control system development through schematisation. Related work concerning particular subproblems to be solved during the design of a robotic system (e. g. obstacle detection, navigation techniques, etc.) will be covered on demand in the respective chapters.

The design of robotic systems is a highly complicated task which requires elaborate methods, techniques, and tools to manage complexity. In recent years, many different architectures have evolved which aim at providing standardised methods for control system design. The core challenge for any of these attempts is to provide suitable schemata (see Definition 1) for particular design tasks.

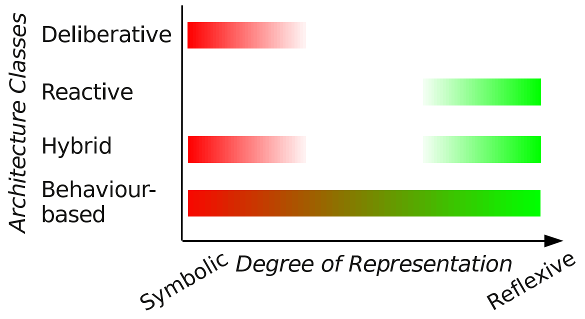
Design Schemata (Definition 1) Design Schemata *are general modelling patterns which aim at solving particular design tasks in a standardised and uniform way.*

These may for instance address component modelling, the modularisation of control systems, the interaction between components, or the representation of information required or generated by components.

Even though design schemata can in the first place be considered independent of a specific architecture, they mostly occur in an embedded form. The reason for that is probably that design schemata often emerge while casting development experience into a design methodology which promotes specific architectural properties. In robotics research, the term "architecture" is very vague and evokes many different connotations depending on the particular context. In this work, *architecture* is referred to as the fundamental methodology to organise a robot's control system adopting a slightly abstracted form of the architecture definition introduced in [Arkin 98] (see Definition 2).



(a) Robot control system spectrum by the criterion *degree of representation* deployed ([Arkin 98] p. 20).



(b) Architecture classification by the criteria *degree of representation* and *structural coherence*.

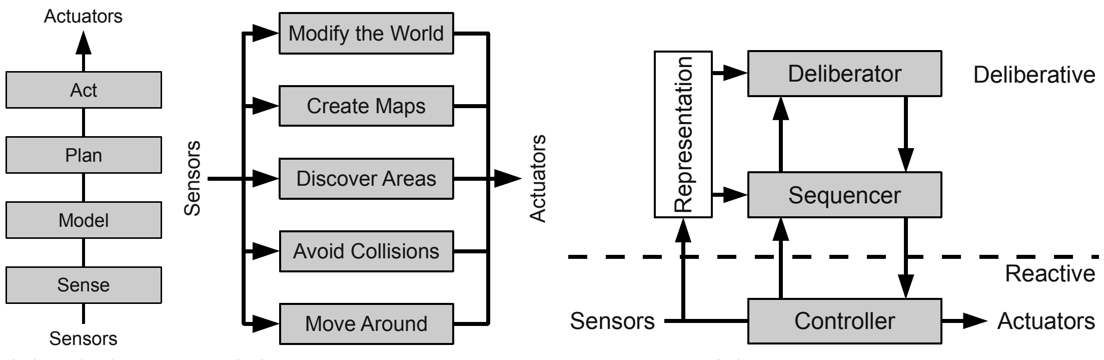
Figure 2.1: Arkin's visualisation of the control system spectrum (a) inspired the architecture classification used in this work (b).

Robotic Architecture (Definition 2) *Robotic architecture is the discipline devoted to the design of highly specific and individual robots from a collection of common software building blocks ([Arkin 98] p. 125).*

On a more abstract level, a robotic architecture can be regarded as a collection of schemata and adjoint guidelines for the design of control systems. Concrete code frames, reference implementations and supporting tools are not seen as part of an architecture and shall rather be referred to as parts of a (software) *framework*.

As for the definition of the term architecture there exist numerous models to classify architectures according to various properties. Many of these classifications illuminate only a fraction of the full range of architectures and are not very precise in distinguishing different approaches. Furthermore, the properties considered are often not linearly independent which renders the comparison among classifications impossible. The classification scheme used in this work is similar to [Matarić 97], where classes of architectures are defined according to the fundamental position towards representation. Representation in that context comprises any kind of information that is retained over a certain period of time. The term representation thus subsumes the range of simple values (like the current state of a finite state machine) towards arbitrarily complex structures (like environment maps or knowledge data bases). In [Arkin 98] p. 20, the spectrum of control systems is visualised according to the criterion *degree of representation* (see Figure 2.1a). Inspired by this visualisation, Figure 2.1b illustrates the classification scheme by [Matarić 97]. The diagram shows the primary criterion *degree of representation* on the horizontal axis and the architecture classes on the vertical axis. The secondary criterion, namely *structural coherence*, can be derived from the nature of the bars representing the structure of the particular classes. In the following, this classification scheme shall be used to group similar approaches.

2.1 From Sense-Plan-Act towards Reactive Systems



(a) S(M)PA **(b)** Reactive Control **(c)** Hybrid Control

Figure 2.2: Structure of robot control systems according to different architecture classes.

In the mid 1980s, common approaches to robot control followed the S(M)PA[1] paradigm which was initially promoted by the artificial intelligence community (see Figure 2.2a) [Chatila 85, Albus 87, Moravec 89, Office 99]. The core of *deliberative* systems is a central world model into which as much data about the environment as possible is aggregated. On the basis of this internal symbolic representation of the real world, the robot computes plans on how to proceed. The plan is then step-wise delegated to an execution layer (`Act`) on which classic control theory is employed. For applications where tasks are clearly defined and uncertainty is well-predictable at design time, deliberative approaches are a good choice.

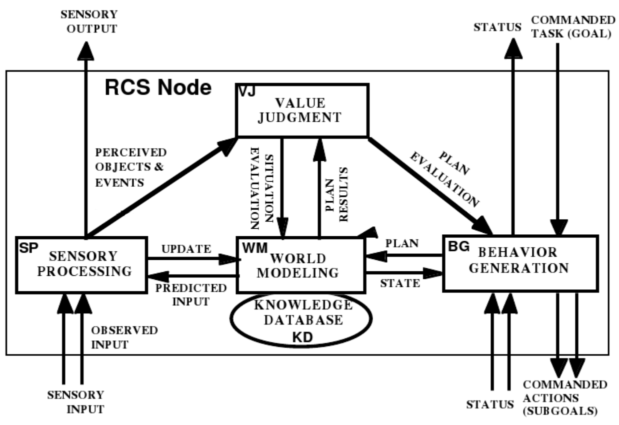
With regard to standardisation, the most popular and influential deliberative architecture is probably NASREM[2] [Albus 87] which was designed according to the NIST[3] standard Real-time Control System (RCS) [Barbera 84]. RCS is an application independent template for control system architectures and is not limited to the control of (mobile) robots. Different reference models tailor the general patterns to specific application domains [Albus 95]. Later revisions of RCS attempt to compensate the conceptual lack of reactive components by introducing multi-resolution world models which are organised in a hierarchical fashion [Albus 02]. Planning on lower levels is carried out on spatially limited high resolution maps and planning on higher levels operates on larger scale low resolution maps. That way, replanning in the vicinity of the robot may achieve fast cycle times. Note that the fundamental problems of data registration and delayed action (open / delayed loop control) remain unaddressed by multi-resolution approaches. Therefore, RCS is regarded as a deliberative hierarchical architecture in the context of this work.

Figure 2.3 illustrates the RCS reference model developed in the context of the Demo programme (see Appendix B.3) conducted from the late 1980s until today. The fundamental building block of the RCS architecture is the RCS node (see Figure 2.3a). Each RCS node is composed of four functional elements:

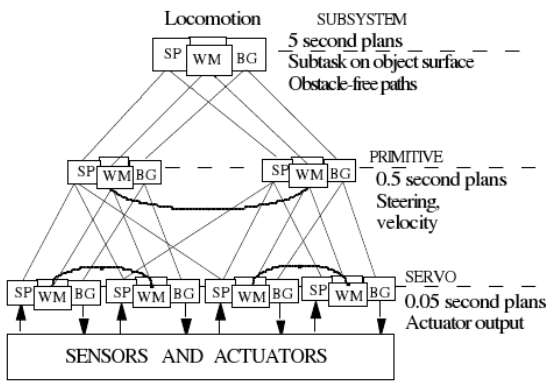
[1]Sense-(Model)-Plan-Act (S(M)PA)

[2]*NASA/NBS Standard REference Model* for Telerobot Control System Architecture (NASREM)

[3]The National Institute of Standards and Technology (NIST) was named National Bureau of Standards (NBS) before 1988.

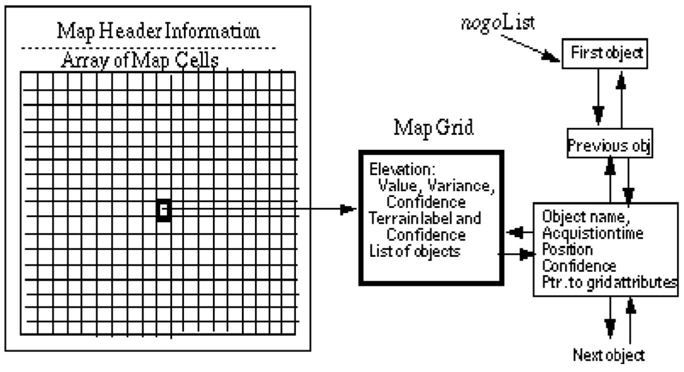


(a) RCS node



(b) RCS locomotion subsystem

(c) Demo III XUV [Lacaze 02]



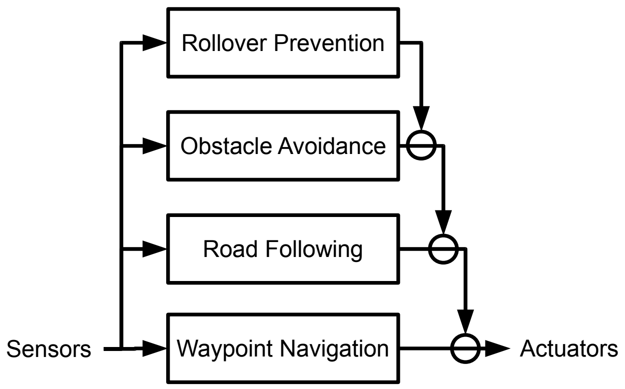
(d) Central world model developed in the context of the Demo programme [Hong 02]

Figure 2.3: 4D/RCS reference model for the Demo III Experimental Unmanned Vehicles (XUV) [Albus 98, Albus 02].

- `Behaviour Generation`: receives goals and generates plans to break these down into subgoals.
- `World Modelling`: maintains the internal representation of the world which is stored in the central knowledge database.
- `Sensory Processing`: uses a priori knowledge to retrieve useful information about the environment which is used to update the world model and to assess the current situation.
- `Value Judgement`: evaluates *perceived objects* and *plan results* to assess the situation. This information is used to influence `World Modelling` and `Behaviour Generation`.

The RCS node can be regarded as a component model which is used to build up a hierarchical network of tasks (see Figure 2.3b). The root RCS node represents the highest task level which is gradually broken down into subtasks. This top-down decomposition is conducted until the control level of the actuators is reached. RCS thus promotes a strict modularisation scheme which does not leave room for other than hierarchical deliberative solutions. Even though RCS promotes a certain kind of modularity, the resulting systems are inherently monolithic due to the way tasks are realised by subtasks. The particular reference models furthermore contain stringent representation systems (see Figure 2.3d).

(a) Cliff by Virginia Tech [Leedy 06b]



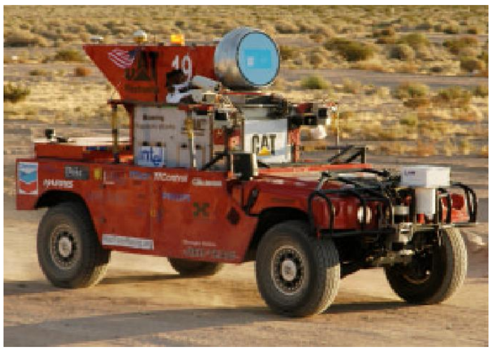
(b) Subsumption diagram of the navigation system structure derived from [Leedy 06a]

Figure 2.4: Cliff follows a strictly reactive control approach [Putney 06, Leedy 06b]. Similar to [Brooks 86], higher level components subsume lower level components.

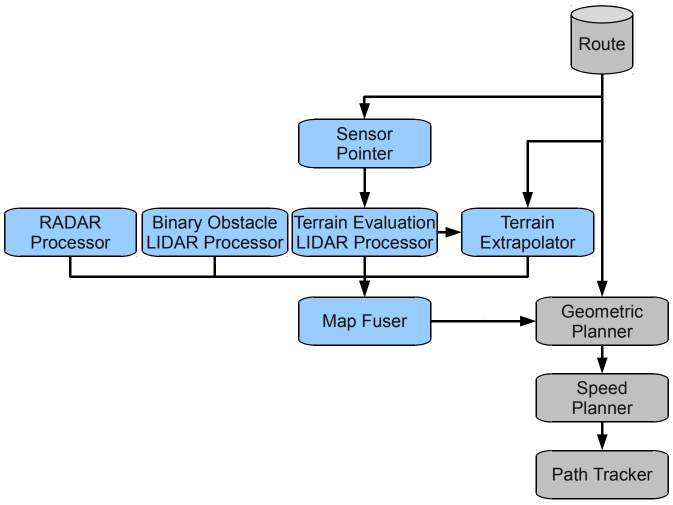
These representation systems specify the internal representation of each RCS node and are therefore highly application dependent. A generic representation scheme is not provided by RCS.

In the early days of robotics research, computational limitations often resulted in robotic systems which spent most of the time frozen and reasoning on what to do next. For that reason, planning approaches were highly vulnerable to changes in the environment as registered data got out-dated during planning such that severe problems occurred at execution time. The will to overcome the lack of responsiveness and the inability to cope with dynamic environments resulted in an increasing popularity of the *reactive* paradigm which was inspired by reflex circuits in biological systems. The most prominent representative of this trend is the subsumption architecture [Brooks 86]. The novel approach promotes the tight coupling of sensor stimuli to actions of the robot on the basis of sets of simple rules. Robot control systems are organised in levels of competences which support the incremental implementation of more and more complex capabilities (see Figure 2.2b). In contrast to hierarchical deliberative systems, reactive architectures are thus inherently extensible which addresses a fundamental issue in robot design. On the downside, reactive approaches almost completely lack the support of design schemata. Subsumption for instance does not even provide a component model for the so called *behaviours*. The guideline for task decomposition is in consequence also rather vague. The major focus is on the interaction between the different levels of competence to arbitrate control handover which can be regarded as a rudimentary fusion scheme for the control flow. Reactive systems are responsive, real-time capable, and robust but lack the capability to master complex situations in which current sensor data alone is insufficient for taking the right decisions. For applications where a limited view on the world is sufficient these approaches provide a robust and computationally efficient alternative to deliberative architectures.

In the DARPA Grand Challenges (see Appendix B.4), Virginia Tech made the attempt to compare the performance of a purely reactive approach with a purely deliberative approach by sending two robots onto the course. This comparative case study yielded some



(a) Sandstorm 2005 is a modified 1986 Model 998 HMMWV [Urmson 06]



(b) Navigation system structure and data flow (reproduced according to [Urmson 06])

Figure 2.5: The CMU vehicle Sandstorm follows a strictly deliberative control architecture with a central sensor fusion [Urmson 06].

qualitative confirmation of architectural properties attributed to both control approaches. [Leedy 06b] rates the reactive control approach of Cliff (see Figure 2.4) as more successful for the 2005 DARPA Grand Challenge scenario than the deliberative control approach realised on their robot Rocky. Theoretical work on deriving rules for reactive systems from planners showed that both paradigms are equally expressive [Agre 87]. Yet, the high dimensionality leads to state explosion in complex systems which makes such "universal plan" approaches [Schoppers 87] unhandy in practice. For reactive systems, the lack of representation is the origin of both their strengths and their weaknesses. The very same holds for the rigorous world modelling in deliberative systems. The central world model is the source of information which lets the planning facility cope with complex situations. Furthermore, the hunter-gatherer mentality reflected in this approach makes sensor fusion an inherent part of such architectures. Data from complementing sensor systems are combined to yield more reliable information about the environment which is quite useful in noisy scenarios as common outdoors. On the downside, the strong dependence on the world model makes such systems vulnerable to representation corruption. Aggregation of data over time is particularly problematic when state estimation is error-prone or sensors are not carefully calibrated among each other. As a matter of fact, a plan is only as good as the data basis on which it was computed.

In particular in off-road robotics where sensor noise introduces a tremendous amount of uncertainty, planning approaches regained a lot of attention. Over the years, planning theory had reached a high level of maturity yielding correct and complete solutions for many real-world problems at acceptable computational expenses. Several vehicles participating in the DARPA Grand Challenge used deliberative approaches [Urmson 06, Leedy 06b, Braid 06, Chen 06]. The most successful contenders were the CMU[4] vehicles Sandstorm

[4]Carnegie Mellon University (CMU) – `http://www.cmu.edu/`

Figure 2.6: Stanley by Stanford University [Thrun 06]

(see Figure 2.5a) and H1ghlander [Urmson 06]. Both vehicles used the same monolithic deliberative control approach which is illustrated in Figure 2.5b. The core element of the architecture is the central `Map Fuser` which integrates the information from all sensor systems. The fused grid map is passed to the `Geometric Planner` which computes a feasible path following the provided `Route`. Apart from the high-level partitioning, none of the mentioned platforms seems to be developed according to a defined design process which comes with the drawback that the control system probably becomes difficult to maintain over time.

2.2 The Emergence of Mix-Architectures

The retro trend towards traditional deliberative systems due to the dead-end reached in reactive robotics could not deceive unnoticed the fundamental weaknesses of S(M)PA. In particular the lack of scalability with growing task complexity, the unsatisfactory extensibility as well as the inherent deficiencies in dynamic environments led to the insight that neither purely reactive nor deliberative systems represent the silver bullet in robot control. Instead of deciding for one or the other position, researchers started to develop architectures which attempted to strike a balance between both extremes by combining the robustness of reactive approaches with the foresightedness of deliberation.

The resulting *hybrid* architectures integrate low-level reactive facilities with high-level deliberative planners in a layered fashion. Over the years, a design featuring three layers (see Figure 2.2c on page 11) was developed by many independent groups [Firby 89, Connell 92, Gat 92, Saffiotti 95, Konolige 97, Arkin 97, Alami 98, Joyeux 10]. The integration of two radically different control approaches poses many problems which are far from being solved. In particular, the definition of slim yet powerful interfaces between the layers has proven to be hard such that resulting systems are difficult to extend and maintain. As indicated in Figure 2.1b on page 10, the semantic gap between reactive and deliberative parts is difficult to close because of the sharp break in methodology. [Gat 98] provides an interesting overview over further work and the historic context of the emergence of hybrid architectures to the de facto standard in mobile robot control. In three-layer architectures, the lowest layer – the `Controller` – encapsulates the robot hardware and realises tight

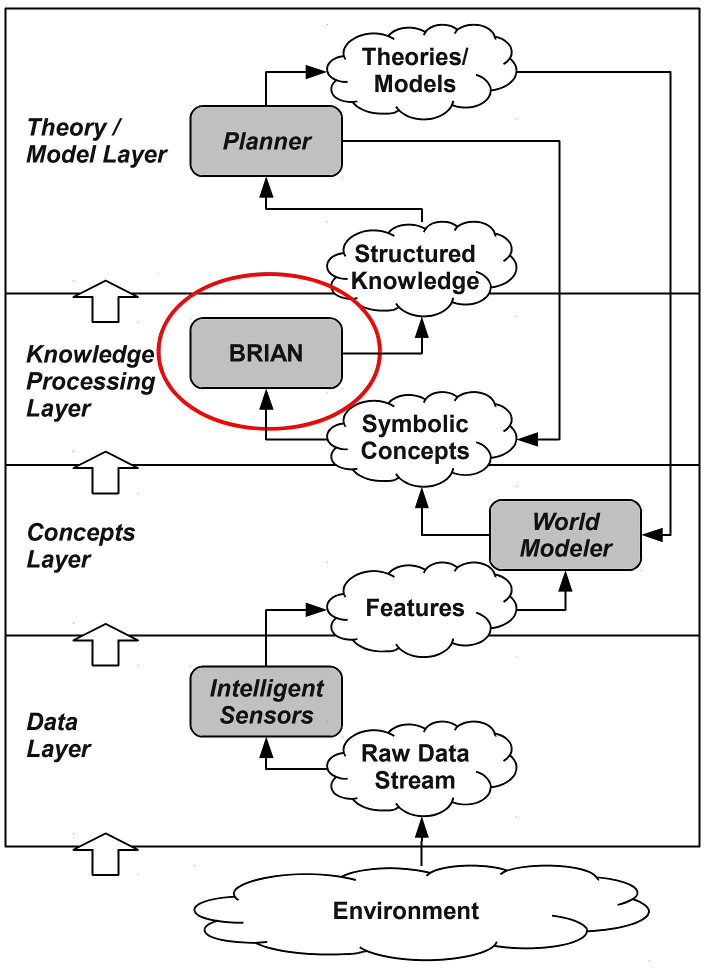


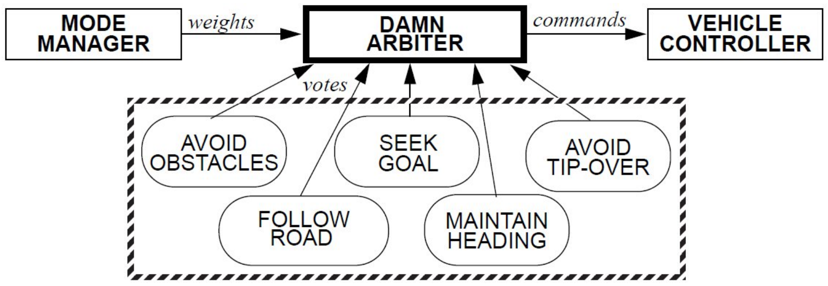
Figure 2.7: The behaviour-based agent *BRIAN* is embedded into a deliberative cognitive architecture. Reproduced according to [Bonarini 03].

sensor-actor reflexes for fundamental safeguarding. The `Deliberator` aggregates a world model from sensor data and makes plans as in traditional S(M)PA architectures. To mediate between the two, a third layer – the `Sequencer` – is introduced which selects appropriate low-level strategies to execute the high-level plan.

A prominent off-road robot following a hybrid control approach is Stanley developed by the Stanford University (see Figure 2.6) which won the DARPA Grand Challenge 2005. Apart from the trajectory computed by the deliberative path planner on the basis of a fused grid map, several reactive components directly couple into the steering control to centre the vehicle on the road or to stop the robot in case of emergency [Thrun 06]. However, there is no explicit notion on the deployment of a specific architecture for this robot.

In general, hybrid architectures tend to provide only a high-level partitioning which splits the control system into reactive and deliberative layers (see Figure 2.7). Adjoint guidelines furthermore give hints what tasks to assign to what layer. Due to the heterogeneous nature of these architectures, it is difficult to define components in a uniform way. For that reason, hybrid architectures usually do not provide any component model.

In parallel to the development of hybrid architectures, another approach towards integrating



(a) Overall structure of DAMN

(b) Navlab vehicle controlled with DAMN

Figure 2.8: The behaviour-based architecture DAMN (a) was deployed on several robots including the Navlab vehicle at Carnegie Mellon University (b) [Rosenblatt 97b].

higher-level intelligence with reactive base functionality was put forth. These novel *behaviour-based*[5] methods can be regarded as extensions to the initial reactive approach proposed by Brooks. The fundamental way of thinking about the decomposition of a robot's tasks into elementary units called basic *behaviours* remained a central idea [Schöner 92, Jäger 95, Rosenblatt 97a, Nicolescu 02]. In contrast to the reactive paradigm, behaviour-based approaches explicitly promote the usage of suitable representations [Matarić 97]. The idea is to model as much as necessary yet as little as possible for a specific task. That way, the benefits of representation on the one hand and the robustness of tight sensor-actor coupling on the other were cast into one coherent framework. Furthermore, this approach provides the possibility to gradually introduce representation on the different levels of competence yielding a distributed, not necessarily hierarchical world model. The particular portions of information about the environment are to a certain degree independent of one another. In consequence, the resulting control systems are more robust against representation corruption than approaches using a central world model. Furthermore, the natural bottle neck of deliberative components – the central sensor fusion facilities – can be avoided by deferring fusion from the sensor to the control level.

The probably most well-known behaviour-based architecture is DAMN[6] [Rosenblatt 97a]. DAMN is a direct answer to the shortcomings of Brook's subsumption architecture and was developed in the context of the CMU Navlab project (see Figure 2.8b). Several complex robot control systems covering the complete spectrum from reactive towards deliberative components have been realised with DAMN [Langer 94, Rosenblatt 95]. Figure 2.8a shows the overall structure of a common control system realised with DAMN. Each task is realised as a separate behaviour which are executed in parallel. Particular behaviours, such as `Obstacle Avoidance` or `Road Following` compute weights which are passed to the `DAMN arbiter`. The weights are combined under moderation of the `Mode Manager` which uses a

[5]Note that many authors in the 1990s – e. g. [Arkin 98] – use the terms *reactive* and *behaviour-based* synonymously. From this point, the former shall only be employed for strictly *stateless* systems while the latter shall be used for architectures which allow internal state to vary from none at all to arbitrarily complex representations. Following this terminology, *behaviour-based* architectures can thus be used to model *reactive* systems but not vice versa.

[6]Distributed Architecture for Mobile Navigation (DAMN)

priori knowledge on which behaviours are most suitable under the current conditions. The resulting command is then sent to the `Vehicle Controller`.

As multiple behaviours are executed in parallel, command arbitration is the core challenge in designing such architectures. Behaviour-based approaches virtually always provide some sort of component model which specifies a common interface for behaviours. This is required in order to couple the behaviours to standardised command arbitration mechanisms which decide which behaviours gain relevance [Proetzsch 10a]. Command arbitration can be thought of as a fusion scheme which is limited to the organisation of the control flow. Sensor data flow and representation are mostly neglected as the behaviour-based robotics community rates schematisation at this point as a loss of flexibility [Matarić 97]. Only very general constraints are formulated to guide the design process.

2.3 Discussion

Control system design in robotics is often reduced to the problem of finding a suitable task decomposition for a given problem. Resulting architectures usually propose (if at all) a more or less formalised component model, a modularisation scheme, and guidelines for the interaction between components. Interaction schemata in that context are mostly limited to the control data flow. One major focus in reactive and behaviour-based robotics is for instance the fusion of control commands from competing components into a consistent set of control values for the actuators. Generally speaking, these design methodologies leave a lot of room for interpretation and often fail to really guide the development of large systems.

The other extreme on the scale is represented by rigorous reference models which make prescriptions on almost every detail of the control system to be developed. These highly domain-specific solutions barely leave room for other than the intended purpose and fail to provide the flexibility required in applications where tasks cannot fully be specified at design time. This kind of rigid systematisation is particularly popular in the control engineering community which promotes hierarchical deliberative system architectures. The resulting monolithic control systems are furthermore difficult to extend which makes the application of incremental iterative design procedures inefficient. However, iterative process models are required to adapt and maintain the functionality of a robotic system over its complete life cycle.

In between both extremes, hybrid control architectures tend to provide a rather rigid skeleton of partitions which represents the preferred top-level modularisation of control systems. Furthermore, a set of guidelines specifies how to sort functionality into the particular partitions. To a certain degree, these approaches assist at structuring complex control systems featuring reactive and deliberative strategies but fail to provide a common component model for functional units. This lacking prescription results in labour-intensive procedures to define communication interfaces which also makes extensibility questionable as interfaces quickly grow quite complex.

Recapitulatory, contemporary architectures fail to provide appropriate modelling techniques for many facets of control system design. Either rigorous prescription impairs development flexibility or lax prescription results in lacking guidance of the design process. For these reasons, the author argues that generic design schemata have to be defined which account for both, design standardisation and flexibility.

3. Control Design Methodology in Autonomous Off-road Robotics

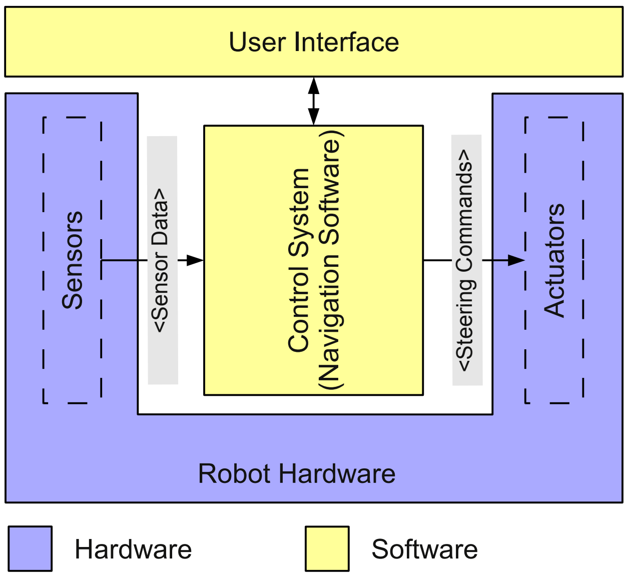


Figure 3.1: General structure of mobile robotic systems.

Mobile robotic systems today are compounds consisting of hardware and software components roughly working together as depicted in Figure 3.1. As this work is basic research in the domain of autonomous navigation, the following will be limited to navigation-relevant components of mobile platforms. The `Control System` in terms of the `Navigation Software` is embedded into the `Robot Hardware` which comprises `Sensors` on the one hand and `Actuators` on the other. In order to be able to provide commands to the robot and to get status information, some sort of `User Interface` needs to be provided as well. The `Navigation Software` has to perceive environmental conditions through the sensor systems, assess the current driving situation, and generate steering commands for the actuators. On the way from one location to another, the robot will encounter numerous

Table 3.1: Sensor data volume on RAVON.

Sensor Type	# Samples / Vol.	Freq.	# Sensors	Volume
Wheel Encoder	1 / 2 byte	50 Hz	4	0.4 $\frac{kB}{s}$
Steering Encoder	1 / 2 byte	50 Hz	2	0.2 $\frac{kB}{s}$
Digital Compass (Magnetic field in 3 planes)	3 / 6 byte	100 Hz	1	0.6 $\frac{kB}{s}$
Inertial Measurement Unit (3×ang. velocity 3×acceleration)	6 / 12 byte	100 Hz	1	1.2 $\frac{kB}{s}$
GPS (Longitude and latitude)	2 / 8 byte	1 Hz	2	0.008 $\frac{kB}{s}$
Bumper Deflexion Monitor	1 / 2 byte	100 Hz	2	0.4 $\frac{kB}{s}$
2D LRF[1] SICK LMS2XX	361 / 0.722 kB	37.5 Hz	2	54.15 $\frac{kB}{s}$
2D LRF[1] SICK LMS1XX	541 / 1.082 kB	50 Hz	2	108.3 $\frac{kB}{s}$
3D LRF[1] SICK S300X	541 / 1.082 kB	25 Hz	1	27.05 $\frac{kB}{s}$
Camera System (320×240)[2]	76.8 k / 230 kB	15 Hz	2	6.9 $\frac{MB}{s}$
Camera System (320×240)[3]	76.8 k / 230 kB	15 Hz	2	6.9 $\frac{MB}{s}$
Camera System (1280×1024)[4]	1.3 m / 3.9 MB	5 Hz	2	39 $\frac{MB}{s}$
		Total Data Volume:		$\sim 53\frac{MB}{s}$

challenges which have to be met by the control system in order to prevent mission failure. Besides issues related to particular sensor systems, accordant evaluation algorithms, or the realisation of concrete manoeuvring strategies, many architectural challenges arise from the required system flexibility. The design of robust and versatile navigation systems therefore has to follow a thorough methodology, which is supported by a suitable control architecture. In the following, crucial architectural challenges addressed in this work shall be introduced in more detail. After that, Section 3.3 will introduce the proposed design methodology itself. Further challenges dealing with particular sensors and algorithms shall be deferred to the application study in Part II of this work.

3.1 Architectural Challenges in Off-road Navigation

In complex environments – which feature a wide variety of different terrain types and obstacle structures – data from a single sensor system does not suffice to yield reliable information about the environment. For that reason, multiple sensors have to be evaluated and information has to be fused in order to mutually confirm or discard what was detected. This fact results in a flood of raw sensor data that has to be filtered, evaluated, and stored for further use. To give the reader a slight idea of the dimensions talked about, the amount of sensor data per second for each sensor system on the robot RAVON, which is subject to the application study presented in Part II, is specified in Table 3.1. The sensor type is given in the first column followed by the number of samples per reading and the data

[1]Laser Range Finder (LRF)

[2]Active low-resolution stereo system used for obstacle detection.

[3]Specialised low-resolution camera system used for water detection.

[4]Active high-resolution stereo system used for terrain classification

volume received per reading. The third column holds the update frequency of each sensor in *readings per second*. In the right-most column the *data volume* for each type of sensor is listed as the product of the *volume per reading*, the *frequency* and the *number of deployed sensors* given in column four.

Besides the sheer dimensions of raw sensor data, the table illustrates that there is a significant gradient in the update frequencies of the various sensor systems. Sensors yielding larger portions of data at once tend to have lower update rates than sensors which deliver only a few values per reading. Furthermore, the delay resulting from accumulating, transporting, and reading the data into the computer is very different as well. The severe differences in size and timeliness lead to significant problems when synchronising and fusing information. In that context, the delay introduced by preprocessing and filtering steps is a further crucial dimension.

Multi-sensor-fusion is a traditional field of research in many disciplines. On mobile robotic systems, limited computational power and motion-induced inaccuracy in localisation adds further complexity to data acquisition. Approaches exploiting several sensors on the raw sensor level tend to be highly dependent on the actual sensor configuration (e. g. sensor type, number of data sources, relative position between sensors, etc.). This results in limited reusability and portability of software components among different platforms. Even slight differences may result in labour-intensive configuration procedures to calibrate and model the multi-sensor-system (see e. g. [Hong 02, Urmson 06, Unnikrishnan 09]).

Data Integration (Challenge 1) *Data Integration refers to the problem of integrating several data sources with different timeliness on the raw sensor data level across time and space. In the literature, the term* Data Registration Problem *is often used synonymously. Careful calibration of the data sources among each other and accurate localisation at all times is a crucial requirement for precise data registration which is difficult to meet in off-road applications.*

Configuration Dependence (Challenge 2) *This issue is related to the* Data Integration Challenge *as sensor fusion on the raw sensor data results in specialised algorithms which are tailored to a particular sensor configuration. This configuration dependency complicates the migration of algorithms between different platforms decreasing reuse potentials.*

Another aspect related to perception is the coverage of sensor systems deployed for environment assessment. The more complex a robot's working place gets, the more sophisticated these facilities need to be designed. In particular, sophisticated sensor systems tend to have limited fields of vision and are therefore unable to cover the complete area around the robot. Even if there was space for mounting an arbitrary number of sensors, the individual systems will still have dead angles. Apart from the higher weight and costs connected to such decisions, the data volume will soon exceed the limited computational capacities on a mobile platform. Furthermore, the omnipresence of sensor systems will probably be a problem for many application scenarios. In essence, the robot control software has to account for dead angles and explicitly deal with the consequences resulting from these. Especially agile platforms – the robot subject to this Doctoral Thesis for example features twin and crab steering (see Chapter 7 for further details) – need to compensate for the sensors' narrow fields of vision as abrupt changes in direction are feasible.

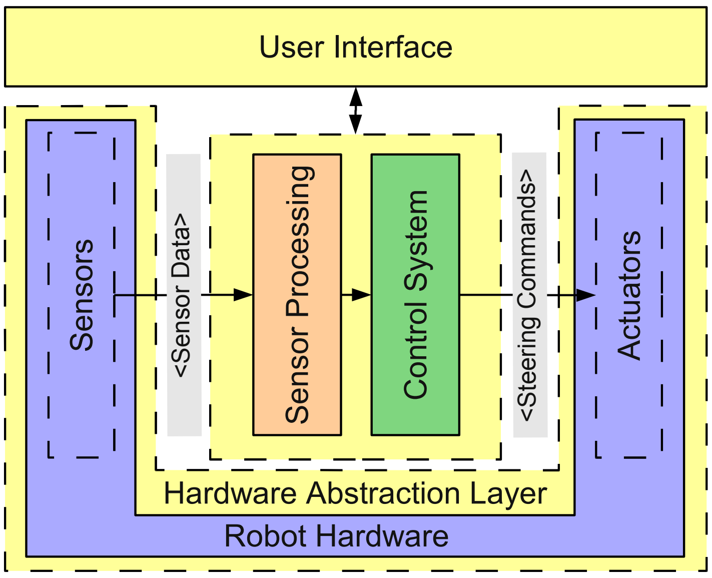


Figure 3.2: `Sensor Processing` is often separated from the `Control System`.

Limited Field of Vision (Challenge 3) *Monitoring the vicinity of a mobile platform in an exhaustive manner is not feasible due to the limited fields of vision of sensor systems. Therefore, dead angles represent an important issue when dealing with complex navigational manoeuvres in narrow environments.*

The development of complex robotic systems is a time-consuming endeavour. Therefore, the coordination of several specialised teams working on different parts of the robot hard- and software is a crucial point. A prerequisite for managing multiple teams is modularisation. In off-road robotics, a common top-level division separates the navigation software into a `Sensor Processing` component and a `Control System` as illustrated in Figure 3.2. Furthermore, `Hardware Abstraction Layers` are often introduced to minimise the dependence on concrete hardware. This layer usually encapsulates low-level controllers, basic data filtering, and sensor protocols. The separation of `Sensor Processing` from `Control` is motivated by the complexity of both fields. The evaluation of sensor data requires other skills than the integration of tasks into a versatile control system as the algorithms deployed vary a lot. Roboticists specialised on each field work on a different level of abstraction, use a different language, and interpret environmental properties on a different semantic level. Furthermore, the deployed software design methods tend to run in opposite directions. Sensor processing facilities are usually designed bottom up from the raw sensor data over features towards objects (see *Perception-oriented Design* (Definition 4 on page 32)). In contrast , control system design tends to follow a top-down approach starting with high-level tasks which are iteratively broken down into lower-level tasks which finally yield set values for the actuators (see *Action-oriented Design* (Definition 5 on page 33)). This fact results in a semantic discrepancy between both groups of developers which has to be handled with care in order to assure an effective working environment for both sides.

A developer working on a particular obstacle detection algorithm which is possibly tailored to a highly specific sensor system is for example interested in the various types of hazards that can be detected in the data available. In contrast, a developer designing the navigation system does not care for concrete obstacle characteristics, different nuances and reliabilities of measurements. On the control level it is not important whether the path is blocked by a tree trunk, a river or whatever entity else that means harm to the robot. On this level, the way more abstract concept of traversability is of interest. Furthermore, the reader should note that traversability strongly depends on the concrete target platform. For an amphibian vehicle the obstacle situation used in the example above has to be interpreted in a more differentiated way as water bodies may be well-traversable for this type of robot.

Where the designer of sensor processing algorithms requires a lot of flexibility for the representation of different entities, the control strategy designer seeks the opposite: standardised data structures and interfaces which represent exactly the portion of environmental information that is required for the task at hand. This results in a disparity depending on point of view and priority of developers. In essence, the challenge is to support prospering ideas on each level through suitable abstractions without ending up in endless interface definition discussions.

The reader should note that the integration of third-party components belongs to the very same problem class. Today many isolated solutions for specific problems in robot control exist on the market. The most successful ones represent self-contained units which tend to export preprocessed data in a proprietary format (e. g. SICK[1] laser range finders, Point Grey[2] Bumblebee stereo cameras, PMDTec[3] CamCube 3D cameras, or IBEO[4] detection systems). This format tends to reflect the supplier's business field, the education and profession of the involved developers and in the end the organisational culture of the division responsible for the component. The probability that this background matches the concrete requirements of a robot development project is virtually zero. That way, the integration of third-party components usually introduces even deeper semantic gaps into the system than the integration of components realised by different subdivisions of the same team. In order to close these gaps, information from one semantic level has to be lifted to another with the constraint that one level is completely beyond the control of the integrator. The key to accelerated development of powerful new robotic applications is the capability to incorporate components from various suppliers (in-house or third-party) deploying one coherent methodology.

Semantic Discrepancy (Challenge 4) *Developers working on different levels of control do not share a common point of view. The semantic gap between the various levels results in a semantic discrepancy which has to be bridged in order to integrate components from different teams into a coherent system. The integration of third-party components raises similar problems with the additional constraint that internals of the component are beyond the control of the robot developer. Both issues might be solved in a uniform way.*

Besides the "horizontal" semantic gaps identified above, conceptual breaks are a further source of semantic discrepancy. In order to manage the complexity of navigational tasks in difficult terrain, a robot's control system has to be split up into well-defined components.

[1]SICK – `http://www.sick.com`
[2]Point Grey Research – `http://www.ptgrey.com/`
[3]PMDTec – `http://www.pmdtec.com`
[4]IBEO Automotive Systems – `http://www.ibeo-as.com`

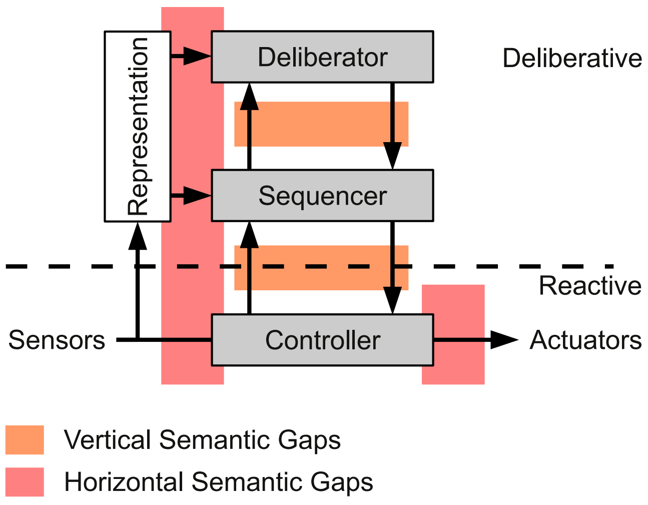


Figure 3.3: Semantic gaps in hybrid control architectures.

As already stated in Section 2.2, a logical partitioning into three components reflecting qualitatively different fundamental capabilities was established over the years [Gat 98]. Robot control systems following this partitioning feature a `Deliberator`, a `Sequencer`, and a `Controller`. The `Deliberator` generates plans on the basis of environmental information. The `Sequencer` breaks these high-level plans down into smaller control steps which can be executed by the reactive `Controller`. In common hybrid architectures these three components are arranged in a layered fashion. A strict separation of capabilities into these three components results in conceptual breaks at the component interfaces. These conceptual breaks lead to "vertical" semantic gaps between the layers which leave room for system failure. The semantic discrepancies identified in robot control are illustrated in Figure 3.3.

Besides challenges in sensor processing, robot control in complex environments with high uncertainty also unveils many unsolved scientific problems. In particular, local manoeuvring in unknown narrow and vegetated environments has not yet been addressed by many researchers so far. In addition to the perceptional and representational complexity required for controlling a vehicle under such conditions, the graceful control handover represents a further crucial issue. The wide variety of different terrain types (open field, forest areas, dirt tracks, underwoods, etc.) – each featuring different soil conditions and obstacle structures – requires multiple strategies to master various driving situations and to decide what action to take. *Action Selection* in that context does not only comprise the selection of the strategy appropriate in the current situation but also the selection of the perceptional basis to use for this strategy. To support this process, meta classification of terrain type and environmental conditions can be called into service. On the basis of this meta information, traditional *Action Selection* switches between available strategies on a single level of competence in an abrupt fashion. This has the disadvantage of scaling poorly with terrain types which were not anticipated at design time. This problem concerning control handover between strategies also occurs when integrating complementing strategies on different levels of competence. The lower-level competence may not have the farsightedness to take the right decision but the appropriate information granularity (i. e. the strategy might run into a local minimum). The higher-level competence on the other hand might have the farsightedness but may lack the required granularity (i. e. the strategy might not *see* a solution due to aliasing effects).

Action Selection / Control Handover (Challenge 5) *When operating in highly unpredictable terrain, the robot needs to choose from several strategies in order to adapt to the terrain type at hand. Classical action selection switches strategies in an abrupt fashion which may result in a lack of granularity or farsightedness depending on the chosen strategy.*

3.2 Architecture in "Natural Mobile Systems"

Until today "natural mobile system" are way better at performing navigational tasks than mobile *robotic* systems. In millions of years nature has apparently evolved outstanding solutions for navigational and structural challenges. This includes the challenges mentioned above and many more which will emerge as soon as less and less structured environments is targeted. The less structured the environment, the more evident the superiority of natural mobile systems becomes and the more it leads to the desire to better understand biological methods for perception, decision, and control. At all times, nature has been a driving inspiration for scientists and engineers to advance in technology and the statement above virtually advises to take nature as a guide.

Today the magnitude of available sensor data and the processing capabilities of modern machines has almost reached the dimensions of human capabilities. Though nowhere near fully understood, the brain is certainly very different from computers in terms of data representation and computation which makes any comparison of capabilities probably invalid. Nonetheless, this should not lead to the conclusion that computer systems are simply not apt for the task of navigation through difficult environments. No more is the author of the opinion that it is sensible to imitate the brain at any price. In this case the strengths of computer systems like accuracy, memory persistence, and the possibility to carry out millions of computations per second would melt away very quickly.

In spite of the differences between the brain and computer systems, the author of this Doctoral Thesis believes that there is still a lot to be learnt from biological systems on a logical level. In essence, not the concrete imitation of nature – which is not feasible at the moment – but the adoption of concepts in modularisation, filtering, and processing are of interest. The strengths of the technical means at hand shall be exploited while the current level of understanding concerning the architecture of natural perception, decision, and control serves as a creative director for the endeavour of this work.

3.2.1 Conquer System Complexity through Modularisation

On a logical level, the brain is a well-organised structure which is partitioned into different well-identifiable processing regions. Already at the dawn of the 20^{th} century, the German neurologist Korbinian Brodmann defined a partitioning of the brain into areas – today known under the term Brodmann map – according to structural difference. With the years, the Brodmann map (see Figure 3.4) was annotated and consolidated as the functions

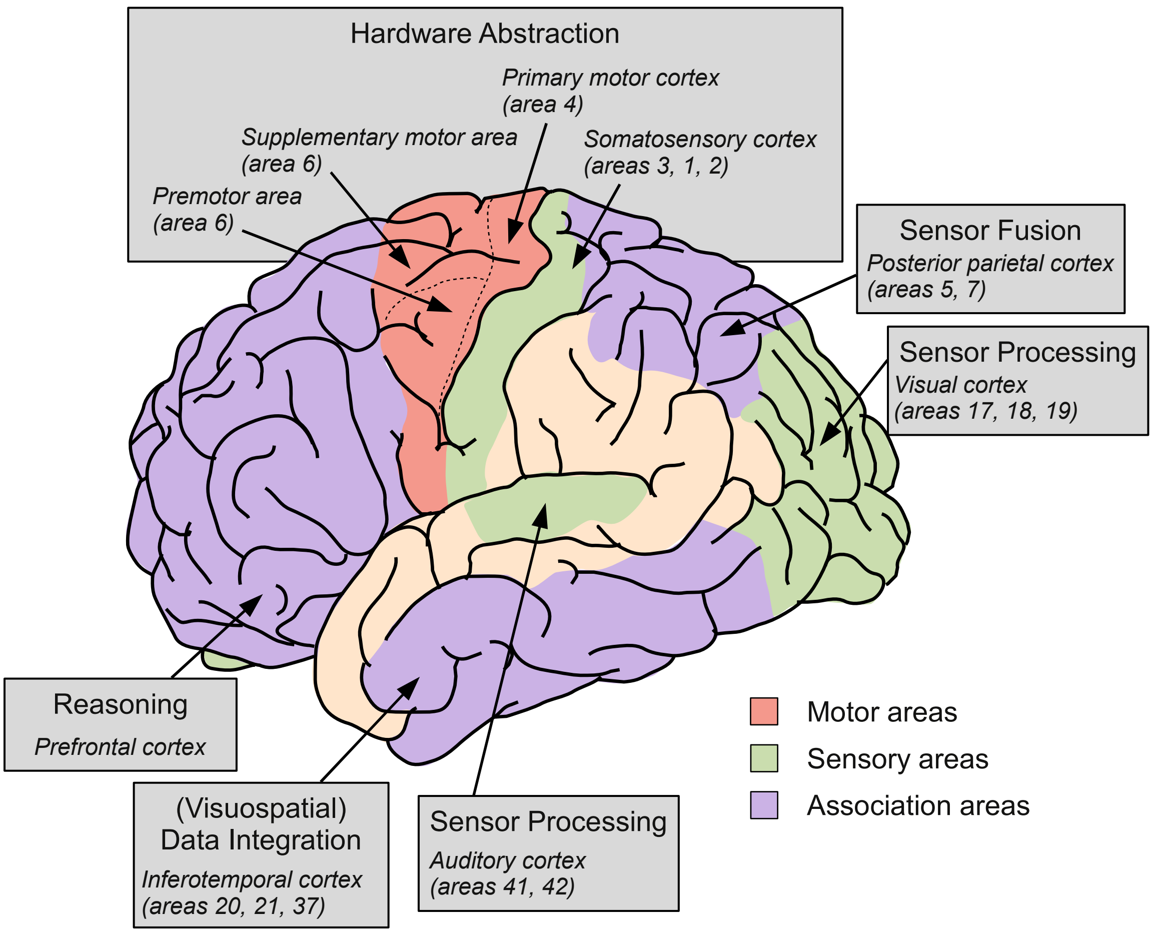

Hardware Abstraction
Primary motor cortex (area 4)
Supplementary motor area (area 6)
Somatosensory cortex (areas 3, 1, 2)
Premotor area (area 6)
Sensor Fusion
Posterior parietal cortex (areas 5, 7)
Sensor Processing
Visual cortex (areas 17, 18, 19)
Reasoning
Prefrontal cortex
(Visuospatial) Data Integration
Inferotemporal cortex (areas 20, 21, 37)
Sensor Processing
Auditory cortex (areas 41, 42)
🟥 Motor areas
🟩 Sensory areas
🟪 Association areas

Figure 3.4: Brain map by Brodmann annotated with neurobiological names and classification of function in robotic terms.

of regions were discovered. In that context, Brodmann's initial thought that structural difference is an indicator for functional difference was confirmed. Interestingly enough, certain brain areas can be mapped to the arrangement of common structures in robotic systems. Analogies of `Sensory Cortices` and `Motor Cortices` can be found in many control systems in terms of `Hardware Abstraction Layers` (HAL). Furthermore, the separation into functional components dealing with perception, safety reflexes, reasoning, and planning has widely been adopted to manage system complexity. In the nervous system these functional units can be found in the form of sensory cortices like the `Visual` or `Auditory Cortex`, reflex circuits on multiple levels, and the `Prefrontal Cortex` for higher-level reasoning. There is strong evidence that the function of the `Parietal Cortex` is involved with integrating information from several senses for spatial awareness which is crucial in locomotion. Partitioning a robot control system as motivated above is not a new idea. Many insights can be derived from engineering experience and need no biological justification. Nonetheless, the similarities identified in both structures raise the question whether there are further structural conditions in the nervous system which are worth porting into the technical world. In particular methodologies to manage complexity, assure extensibility, and support maintainability of multi-sensor systems may benefit from biological models.

3.2.2 Sensor Centres minimise “Configuration Dependence”

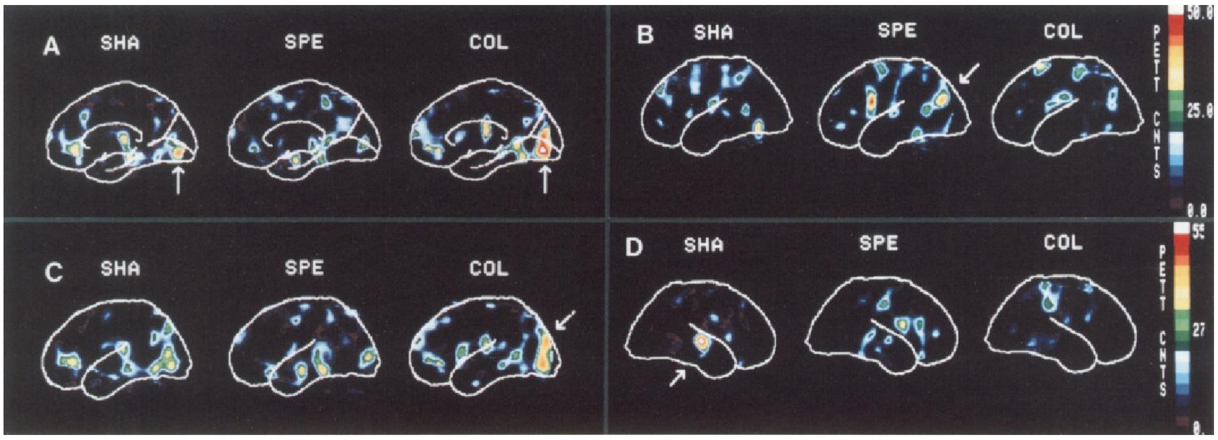


Figure 3.5: PET images in the sagittal plane[5] on attention to speed (B), colour (C), shape (D), and combined attention to shape and colour (A). Brain activity minus activity during divided attention (i. e. no focus on any specific feature) is plotted on outlines of the brain (taken from [Corbetta 91]). In each experiment, different stimuli for shape (SHA), speed (SPE), and colour (COL) were shown to the test persons. The results are presented in the three columns SHA, SPE, COL where centres of significant activity are indicated by white arrows.

(A) Left collateral sulcus activation for combined attention to shape and colour (slice taken 27 mm left of the midline)

(B) Left intraparietal and sulcus activation for attention to speed (slice taken 44 mm left of the midline)

(C) Left dorsolateral occipital cortex activation for attention to colour (slice taken 29 mm left of the midline)

(D) Right superior temporal sulcus activation for attention to shape (slice taken 54 mm right of the midline)

Neuroscience has proven that specific nerve centres exist in the brain for processing data from different senses. The partitioning of the Brodmann map already indicates this presumption. Furthermore, different aspects of one and the same sense also appear to be processed in different specialised regions of the brain which provide a subdivision of the sensor centres. In the 1990s, PET[6] imaging allowed for the non-invasive monitoring of brain activity which disclosed qualitatively new neuroscientific possibilities to localise brain functions. In several experimental series Corbetta et al. asked test persons to pay attention to either shape, speed, or colour of presented objects (combinations of the three criteria were also investigated) [Corbetta 91]. Figure 3.5 shows the PET-based results which indicate that specific sensor centres are activated on similar stimuli according to the current focus of attention demanded from the investigated person. These experiments proved that there is a strict separation of distinct concerns in the processing of sensory

[5]The sagittal plane is a standard plane in medical imagery which divides bodies vertically into left and right halves.

[6]Positron Emission Tomography (PET) visualises a radioactive marker which is introduced into the blood of the investigated subject. Today PET is often combined with Computer Tomography (CT) or Magnetic Resonance Tomography (MRT) to register metabolic and anatomic information at a time.

stimuli. The information of multiple senses appears to be integrated after the extraction of semantic features in different areas of the brain.

Evolving lines of robotic systems may vary a lot as to their sensor and actuator equipment which makes configuration dependence an important issue with respect to reusability. In contrast living creatures come with a rather fixed "configuration". Nonetheless, nature appears to avoid interdependencies between different stimuli processing strands – even when operating on identical sensor data. The author argues that this model represents an adequate approach for minimising *Configuration Dependence* (Challenge 2 on page 21) in mobile robotic systems by strictly separating distinct concerns in sensor processing and defer data integration and fusion to later processing steps. That way, information extraction algorithms do not depend on complex multi-sensor-systems and can be designed more independent of concrete sensor/actuator configurations.

Perceptional Separation (Design Idea 1) *Strictly separate perception into self-contained sensor centres to minimise* Configuration Dependence *(Challenge 2 on page 21). Perceptional units should be regarded as filters which generate semantic entities based on data streams from individual senses.* Semantic Discrepancy *(Challenge 4 on page 23) shall be treated explicitly such that particular subsystems can work in a natural way using their own semantics and granularity.*

Deferred Fusion (Design Idea 2) *Fuse perceptional information on the semantic level (i. e. subsequent to filtering through the percepts) to support traceability and reusability. This approach further addresses* Data Integration *(Challenge 1 on page 21) to a certain degree. Preprocessed information on an abstract level tends to be significantly filtered compared to raw sensor data. The author therefore argues that fusion on an abstract semantic level somewhat defuses problems related to data registration (e. g. laborious calibration procedures) due to an improved robustness towards slight localisation inaccuracies.*

3.2.3 Data Integration and Semantic Discrepancy

The design idea to minimise configuration dependence by separating sensor processing into fine-grained strands and defer data integration to the semantic level leads to the question: Where and how is data from different senses combined to yield more reliable and complete information about the environment. As invasive experiments with the human brain are not possible, scientific research in that domain is difficult and insights on this subject remain on a very high level of abstraction. The cognition scientist Baddeley and his colleagues analysed the human capabilities for parallel processing of auditory and visual stimuli to develop a brain model that would answer such questions. The neurological deficits identified in experiments with people who suffered serious brain injuries [Baddeley 74, Baddeley 96] support the PET-based results from [Corbetta 91] that data from different senses is processed in different areas of the brain. As illustrated in Figure 3.6, the model introduces distinct components for audio processing (`Phonological Loop`) and visual processing (`Visuospatial Sketchpad`). In later publications [Baddeley 00], a third component – the `Episodic Buffer` – is introduced to model information integration from several sources (i. e. senses) across time and space. This structure is described as a temporary storage with

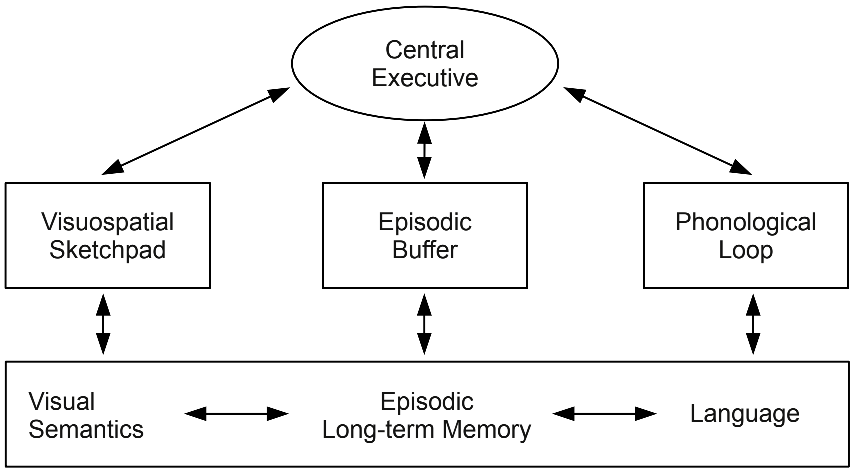
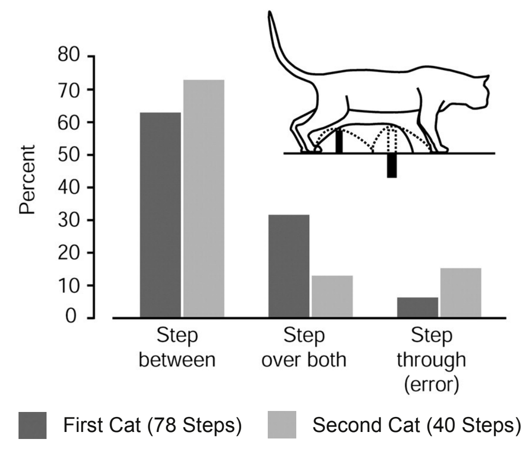


Figure 3.6: Model for Working Memory by the Neuroscientist A. Baddeley. Reproduced according to [Baddeley 00].

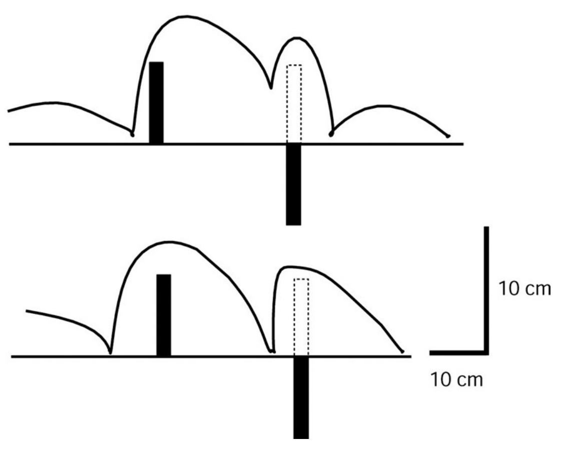
limited capacity which is believed to “serve as an interface between a range of systems, each involving a different set of codes”. In this brain model fusion of sensory data is apparently carried out on an abstract level and well after the initial extraction of semantic features. Experience stored in the `Episodic Long-term Memory` provides the semantic translation between the various codes used. This comprises the recognition and production of visual and acoustic features (i. e. `Visual Semantics` and `Language`), as well as the data integration which is regarded as a further feature of the `Episodic Buffer`. In Baddeley’s model, a `Central Executive` arbitrates the processing in the brain. This entity is a vehicle to model concurrent and sequential processing capabilities. For further reading on memory and learning the reader shall be referred to the text book [Baddeley 09] which gives a broad overview of the field.

At first glance, there is not much analogy between the development of the brain and the design of a robot control system. Anyhow, Baddeley’s model of working memory provides ideas how to tackle *Data Integration* (Challenge 1 on page 21) and *Semantic Discrepancy* (Challenge 4 on page 23) by deferring both to later processing stages and regarding both as separate problems. Different processing subsystems are allowed to use different sets of codes to model internal state in a natural fashion. Therefore, a configurable translation scheme should be designed to allow for the transformation of these codes into a form that is semantically compatible. This furthermore allows data integration on an abstract level which can be handled in a uniform way. Note that this approach also provides the possibility to design and realise sensor processing independent of the control strategy and vice versa.

Sense-Control-Duality (Design Idea 3) *Minimise interdependencies between sensor processing and control to support large teams of developers, layer-independent reuse of components, and portability between target platforms. As before,* Semantic Discrepancy *(Challenge 4 on page 23) shall be treated explicitly to allow for natural semantics (codes) in the particular subsystems.*



(a) Experimental setup and statistics of the results.



(b) Common trajectories of the hind legs: Stepping over both obstacles in one step (upper) and stepping between the obstacles (lower)

Figure 3.7: Cats remember obstacles which are out of their field of vision [McVea 06].

Semantic Coupling (Design Idea 4) *Provide standardised yet generic mechanisms for the translation of information between different semantic levels (e. g. sensor processing on the one and control on the other hand) to bridge* Semantic Discrepancy *(Challenge 4 on page 23). That way information can be provided in a purpose-driven fashion which allows for the integration of action-oriented and perception-oriented design approaches into one coherent methodology (i. e. without endless interface definition discussions).*

Abstract Fusion (Design Idea 5) *Besides furnishing information in a tailored fashion, semantic abstraction also provides a tool for lifting information from different sources onto one coherent level of abstraction. Semantically compatible data can be integrated in a generic fashion allowing for extensible multi-sensor-systems (*Data Integration *(Challenge 1 on page 21)).*

3.2.4 Solving the Limited Field of Vision Problem

Writing about working memory highlights another feature in natural mobile systems: the capability to keep in mind situation and context during the execution of a task. The short-term memory allows creatures to compensate their usually narrow field of vision with information already perceived some time ago. That way, decisions can be taken on a more complete view of the world. The hind legs of a cat for example avoid obstacles and step on tree branches which are no longer in the animal's field of vision. Figure 3.7 illustrates experiments carried out by McVea et al. [McVea 06]. The cats had to step with their forelegs over two obstacles. While being held back by offered food the right obstacle was lowered (dotted lines). After a certain period of time the cat was released and the hind legs' trajectories were analysed as to avoiding the second obstacle or not. Even if the cat is held back for several minutes it apparently remembers position and extent of entities nearby with amazing precision. Apart from this insight another interesting fact is that the cats seem to favour stepping between the two obstacles (lower trajectory in Figure 3.7b) rather than trying to step over both in one step (upper trajectory in Figure 3.7b). Apparently, the cat decides for the safer gait on the basis of potentially outdated information.

Note that information is not held in a central repository of the brain. Knowledge is implicitly represented in distributed neural patterns. That way, natural mobile systems are very robust against world model corruption. The author therefore promotes distributed representations for mobile robots.

Short-term Memory (Design Idea 6) *Aggregate local terrain information into fine-grained representations of limited spatial and temporal extent to compensate for the sensors'* limited fields of vision *(Challenge 3 on page 22). In the style of natural mobile systems, a distributed aggregation is promoted to minimise the risk of world model corruption. [Schäfer 07b] presents the initial concept of a short-term memory for mobile robotic systems.*

3.3 Lessons Learnt from Nature

To solve the challenges in mobile off-road robotics outlined above, the design ideas derived from nature can be applied at two different stages of development. General solution patterns can be cast into a tool box or framework in order to provide standardised mechanisms for the development process. Furthermore, principles and guidelines need to be defined to assist developers in solving problems in concrete applications at design time. Both issues are addressed in this Doctoral Thesis where the former to a certain degree provides the frame for the latter. This reflects the fact that some problems can be solved efficiently in a uniform fashion and others need to be tailored to particular applications. Note that the decisions what problem to solve when – at framework design time or at application design time – mutually influences the conceptual layout of the framework and the design methodology. In the following section the general ideas underlying the framework design shall be highlighted together with the concepts that can be deployed to design a control system.

To lay the conceptual basis for the design methodology proposed in the Doctoral Thesis at hand, the previous section introduced the design ideas which guided the schemata developed in this context. The decision what problems to solve at this point and what problems to defer to the application design time shall be motivated here.

The aim of schemata is to provide a generic infrastructure to facilitate the design of a certain class of large systems. This usually comprises a structural frame and a number of base components which can be tailored to meet requirements of a concrete application. Base components in that context may be algorithms which solve core problems in a uniform way or information models which assist at designing standardised interfaces. The central requirement to solve a challenge in principle – i. e. at scheme design time – is thus the degree of generalisability and the benefit from such a generalisation. The author decided to focus in particular on the sensors' *Limited Fields of Vision* (Challenge 3 on page 22) and *Semantic Discrepancy* (Challenge 4 on page 23). While the former falls into the domain of representation design, the latter requires a generic translation scheme which transforms representations between different semantic levels.

3.3.1 An Integrative Design Methodology for Robot Control Systems

Experience shows that features, properties, and capabilities of robotic systems are commonly inaccurate, incorrect, incoherent or frankly incomplete at design time. The reason for this inconvenience is that interaction of the system has to be validated in its target environment. This is only possible with an operational demonstrator which is not yet existent. Developers are thus confronted with a chicken-and-egg question which is difficult to resolve. This fact conflicts with principles from software engineering which require the opposite for sound design methodologies. In this Doctoral Thesis a novel design approach to address this problem is proposed as the four C's, namely *Clarity, Completeness, Consistency, and Correctness* [Cooper 05], cannot be met in a requirements document for a complex robotic system. Furthermore, an architecture has to be used that allows for incremental implementation, posterior extension, and partial redesign during the complete life cycle of the system.

In the previous sections, the architectural challenges in off-road navigation have been highlighted and solutions in natural mobile systems have been discussed. The insights obtained from these considerations have yielded design ideas which form the basis for the design methodology developed in the context of this Doctoral Thesis. In the following, the proposed methodology and the architectural foundations will be introduced in more detail.

3.3.2 Action/Perception-oriented Design Methodology

The proposed design methodology attempts to solve design challenges by thorough abstractions. In particular the *Semantic Discrepancy* (Challenge 4 on page 23) in large control systems which is further intensified by *Perceptual Separation* (Design Idea 1 on page 28) and *Sense-Control-Duality* (Design Idea 3 on page 29) shall be bridged in a conceptually explicit way. The central concept underlying the proposed design methodology is to achieve *separation of concerns* via generic representations, gradual abstraction of semantics, and abstract fusion of information flowing through the control system. The design methodology proposed in this work combines the strengths of action-orientation in control level design and perception orientation in sensor processing design into one coherent methodology.

Separation of Concerns (Definition 3) *Separation of concerns is a key principle in software engineering which aims at the design of robust, adaptable, maintainable, and reusable software components. This principle can be traced back to Dijkstra's essay on scientific thought [Dijkstra 82] which laid the basis for the quality factors named above.*

The proposed design methodology attempts to systematically port this idea to component and system development in the robotics domain.

Perception-oriented Design (Definition 4) *Perception orientation constructs robot software bottom-up starting from the problem of how to extract and represent required information from the environment.*

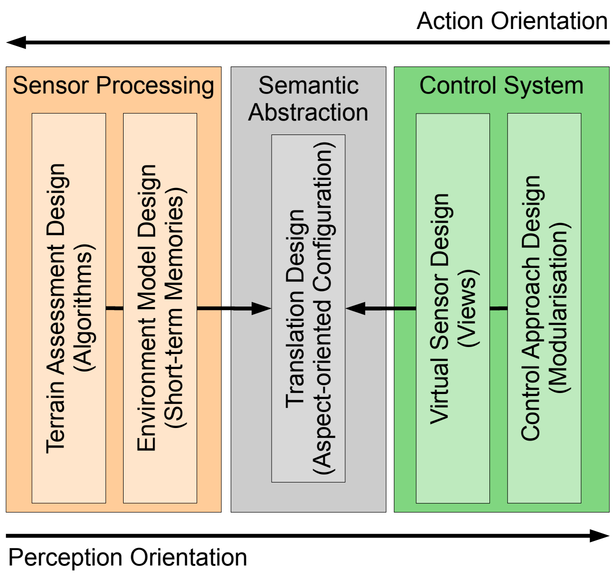

Action Orientation
Sensor Processing
Terrain Assessment Design (Algorithms)
Environment Model Design (Short-term Memories)
Semantic Abstraction
Translation Design (Aspect-oriented Configuration)
Control System
Virtual Sensor Design (Views)
Control Approach Design (Modularisation)
Perception Orientation

Figure 3.8: Overview of the proposed design methodology.

Action-oriented Design (Definition 5) *Action-oriented design approaches take the top-down perspective and start with the modularisation of tasks into subtasks. For each subtask specific portions of the environment are then extracted from sensor data on an as-needed basis*[7].

Figure 3.8 gives an overview of how the proposed design methodology integrates *Action-oriented Design* with *Perception-oriented Design* [Schäfer 08b]. Bottom-up perception-oriented principles are deployed for the design of `Terrain Assessment` facilities and environment models. In the style of natural mobile systems, these models of the environment have been named `Short-term Memories` (Design Idea 6 on page 31 – remember the cats from Section 3.2.4). Top-down the control strategy is developed in an action oriented fashion leading to a modularisation which is suitable for the tasks at hand. Based on this modularisation, required sensor information is specified in terms of *Virtual Sensors* which provide tailored `Views` on the environment which represent exactly the portion of information that is apt for a particular task. The specifications are interpreted by an additional `Semantic Abstraction` layer which retrieves environment information from the `Short-term Memories` and casts this information into the requested `Views`. `Views` thus represent an abstraction from semantics on the sensor processing layer towards semantics on the control layer. That way, sensor processing design is decoupled from control system design such that effects of changes applied to one layer are prevented from affecting the other. The configuration of the `Semantic Abstraction` is based on the `Short-term Memory` specifications from the sensor processing design step and the `View` specifications from the control system design step. Each specification in principle refers to a different aspect of the environment and defines appropriate semantics and coverage for this aspect. The transfer of information between aspects is specified in the `Aspect-oriented Configuration`. *Aspect-oriented* in this context denotes that the

[7]The term *Action-oriented Perception* is often used synonymously [Arkin 98].

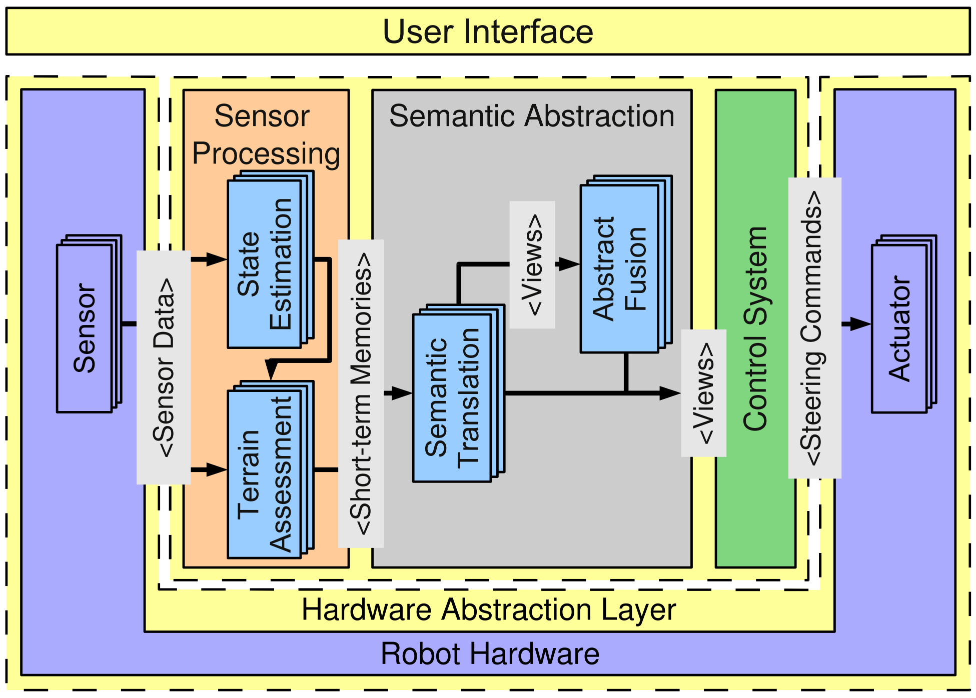


Figure 3.9: To mediate between `Sensor Processing` and `Control System`, `Semantic Abstraction` is introduced as a third layer.

translation from source aspects to each target aspect is regarded individually, allowing for a fine-grained *Translation Design*.

Aspect-oriented Design (Definition 6) *Aspect-oriented design considers the translation of source aspects to each target aspect individually. The resulting* ***`Aspect-oriented Configuration`*** *exhaustively specifies the generic transfer of information between aspects.*

On the logical level the proposed design approach results in a further subdivision of the architectural layout presented above. Figure 3.9 illustrates the refined logical overview. A mediating component, namely the `Semantic Abstraction`, is introduced between the `Sensor Processing` and the `Control System`. Besides the translation of semantics on the sensor processing level to semantics on the control level (`Semantic Translation`), the `Semantic Abstraction` layer is responsible for the fusion of abstract terrain information (`Abstract Fusion`). In general, the author promotes the separate evaluation of particular sensor systems and the deferred fusion on the control level. This reflects the idea that sensor processing algorithms should not rely on the specific arrangement of several sensors in order to minimise system dependence and algorithm complexity. *Deferred Fusion* (Design Idea 2 on page 28) therefore improves both portability and testability. Furthermore, the integration of third-party components is simplified as clear abstractions are an integral part of the design approach.

Note that deferred fusion rather represents a recommendation than a strict principle. In particular higher planning behaviours often require a condensed representation of the environment to yield sensible results. In that context fusion on the control level may be difficult and inefficient. For that reason a generic sensor fusion mechanism on abstracted terrain information, i. e. `Views`, is provided by the `Semantic Abstraction` layer. Terrain information yielded from different `Terrain Assessment` facilities are not necessarily semantically compatible. Depending on the deployed algorithm, the content of the resulting `Short-term Memories` may be incomparable. After the translation to control level semantics, information to be fused is available in a compatible form which allows for the uniform treatment of sensor fusion. *Abstract Fusion* (Design Idea 5 on page 30) thus represents the consequent interpretation of the *Deferred Fusion* (Design Idea 2 on page 28) recommendation towards the requirements on the control level. The individual flexibility of representations on the sensor processing layer remains untouched by this procedure.

4. Representation, Abstraction, and Fusion of Environmental Information

In this section, the design schemata supporting the methodology outlined in the previous section shall be introduced. As already stated in Design Idea 6 (Short-term Memory), adequate representations of the world are – irrespective of the underlying control approach – vital to succeed in complex scenarios. However, behavioural robotics is traditionally focussed on the standardisation of the control flow neglecting sensor data flow and representation design. The reason for this attitude becomes apparent when taking into account the major ambition of behavioural robotics to fuse competing commands into a single set of commands that is executable by the actuators. Furthermore, it is widely promoted that the set of possible environment representations used in *Behaviours* must not be constrained [Matarić 97]. This tenor in the community may lead to a defensive position towards representation schemata as these might imply restrictions on expressiveness. Domain-specific reference models, for instance derived from standard RCS [Albus 95], represent the other extreme on the scale. These approaches tend to overregulate the design process such that the range of possible solutions is quite limited. In this work, a representation-centred approach towards schematisation was chosen which shows that sound prescription may lead to increased expressiveness and standardisation alike.

4.1 Representation Scheme Design

First of all, it is important to note the qualitative difference between designing representation for information retrieval (i. e. storage) and designing representation for data exchange (i. e. communication). In particular, the flexibility required for the former calls for a generic and extensible representation scheme while the latter tends to demand as much standardisation as possible in order to support reusability of control structures. *Sense-Control-Duality* (Design Idea 3 on page 29) further states that interdependencies between sensor processing and control should be minimised. From the control point of view, stored context knowledge should be provided on an as-needed basis in order to keep functional units as simple as possible. In essence, perspective, resolution, and semantic content should reflect the functional units' needs. From the sensor processing point of view, a wide variety of different properties should be storable in order to represent the environment as effective and natural as possible. Perspective, resolution, and semantic content should therefore reflect the specifics of the sensor system in question.

Despite these partially conflicting requirements and a defensive position towards representation schemata, a design methodology is meant to guide the overall design process. As representation design is a crucial point in this context, the author proposes a representation scheme which attempts to strike a balance between standardisation and expressiveness by adhering to the following design points:

Structure-Content-Duality (Design Point 1) *Representation structure and representation content should strictly be separated. Each structural entity should provide a standardised interface for accessing generic content elements.*

Content Standardisation and Genericity (Design Point 2) *All concrete content elements should provide and maintain a simple standardised interface. Further standards should be deployable on demand.*

Handler Standardisation and Genericity (Design Point 3) *By commitment to certain representational standards, generic algorithms can be implemented to handle structure and part of the content in a transparent fashion. In particular the transformations between different structures and the translation between various semantic levels of abstraction are of crucial importance to realise* Semantic Coupling *(Design Idea 4) and* Abstract Fusion *(Design Idea 5).*

As already stated above, representation design for data storage sets other priorities than representation design for communication. In order to balance between expressiveness on the one hand and uniformity on the other, the author proposes to modularise representations into `Structure` and `Content`. While `Structure` reflects the general arrangement (i. e. (logical) location[1]) of information, `Content` models the information itself. The interfaces of `Structures` represent one element of standardisation introduced by this scheme. The second element of standardisation is related to the modality of modelling `Content` with property sets. The standard `Structures` with accordant default content constitute the common communication interface which should preferably be used to transfer data between

[1]Locations may not be measurable in a metric sense (e. g. nodes in a topological graph may only have a logical ordering). Locality in that sense shall be called *logical*.

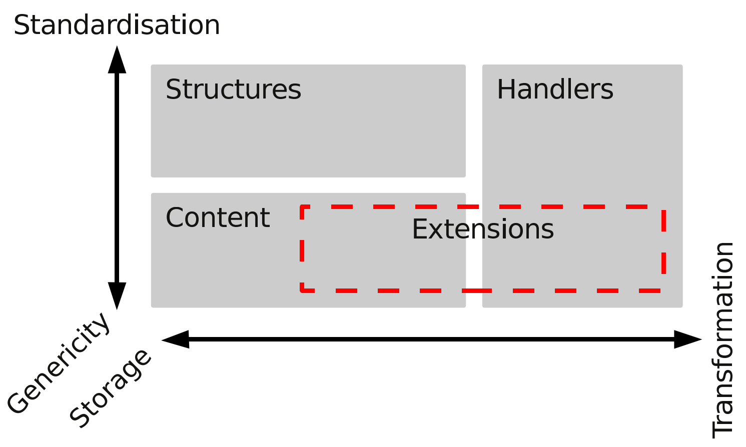


Figure 4.1: Top-level partitioning of the proposed representation scheme.

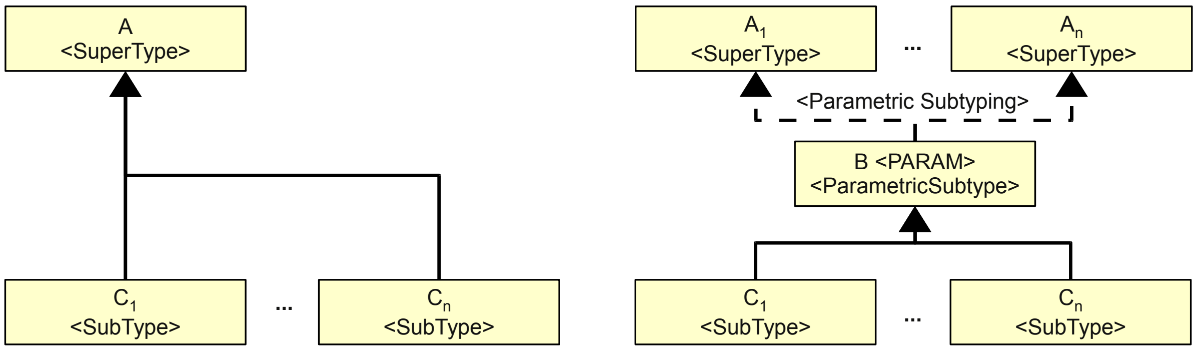
functional units. On the basis of these commitments, a uniform interface is provided which further allows the deployment of several standard transformation algorithms which are wrapped into standardised functional units called `Handlers`. Generic `Extensions` for `Content` and `Handlers` provide additional expressiveness for the realisation of flexible representations with appropriate base functionality as for example required by sensor processing or mapping facilities (both have a strong primary focus on data storage). Figure 4.1 illustrates the top-level partitioning of the proposed representation scheme. The vertical axis of the diagram indicates the gradient from standardisation towards flexibility which ranges from rather standardised `Structures` to rather generic `Content`. The horizontal axis highlights the role of entities in the representation scheme. Where `Structures` and `Content` are used for storing information `Handlers` manipulate and transform information.

For the design of a representation scheme, suitable constructs and notations are required which form the foundations to model data, arrangements, and transformations. Before going into detail with each partition of the representation scheme, constructs and according notations from software technology shall be introduced.

4.1.1 Software Technological Foundations

This section aims at providing the reader with the software technological modelling concepts which represent the basis for the schemata proposed in the context of this work. Note that an exhaustive discussion on software theory is beyond the scope of this work. This section will make use of notations which are based on the UML[2] and may have an object-oriented connotation even though the general concepts are independent of concrete programming paradigms. This decision origins from the lack of standardised more general notations and the desire not to introduce a proprietary notation. The reader should abstract from a possible implementation recommendation implied by the notations and focus on the ideas behind the presented.

[2]Unified Modeling Language (UML) – `http://www.omg.org/spec/UML/2.0/`



(a) Notation for *non-parametric subtyping.* **(b)** Notation for *parametric subtyping.*

Figure 4.2: Non-parametric subtyping allows functionality transfer top-down from a *super type* `A` to its subtypes $\texttt{C}_1$ through $\texttt{C}_n$ (a). Parametric subtyping shares functionality of a *parametric type* `B` among concrete subtypes ($\texttt{C}_1$ through $\texttt{C}_n$) of unrelated types $\texttt{A}_1$ through $\texttt{A}_n$. Each $\texttt{C}_i$ is a concrete type which results from `B` inheriting $\texttt{A}_i$.

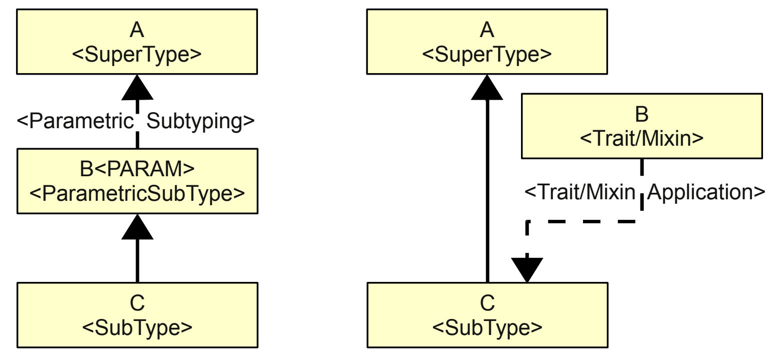
To support reusability in a generic and fine granular way, the author proposes to make intensive use of *parametric polymorphism*[3] [Cardelli 85, Strachey 00]. In the first place, *parametric polymorphism* – as introduced by Strachey in 1967 – allows routine definitions which abstract from concrete types such that algorithms can be applied to different types which share a common structure. A straightforward extension of this concept towards modelling state is achieved by using parameters in type definitions yielding *parametric types* (e. g. containers which abstract from their content). In object-oriented paradigms, *parametric polymorphism* can transparently be applied to *classes* – i. e. types which encapsulate state and associated routines – as both state and routines of the *class* in question may be parametrised. *Parametric types* can further be designed to derive from a parameter resulting in a construction which is often called *parametric subtyping*. In that case the *parametric type* abstracts from its *super type* and may thus be combined with any type that matches the required interface. That way, functionality can be shared between types which are otherwise unrelated. Figure 4.2 illustrates the conceptual difference to non-parametric subtyping which only allows functionality transfer from the *super type* to its *subtypes*[4].

Parametric Polymorphism (Definition 7) *Parametric polymorphism allows the definition of functions which work uniformly on a range of types [Strachey 00].*

Parametric Type (Definition 8) *Type definitions which use parameters to capture state in a generic fashion shall be called* parametric types *in the context of this work.*

[3]In the context of programming languages *parametric polymorphism* is often referred to as *generic programming* [Garcia 03, Garcia 07]. Examples of according language means are for example C++ templates or Java Generics.

[4]E. g.: *Inheritance* in object-oriented paradigms only allows code reuse among “family members” in a top-down fashion



(a) *Parametric subtyping* **(b)** *Trait/mixin application*

Figure 4.3: In *parametric subtyping* a concrete type `C` results from deriving a *parametric type* `B` from a *super type* `A`. Type `C` – which is equivalent to `B<A>` – is thus a subtype of `A`. In *trait/mixin application* `B` is unrelated to `A`.

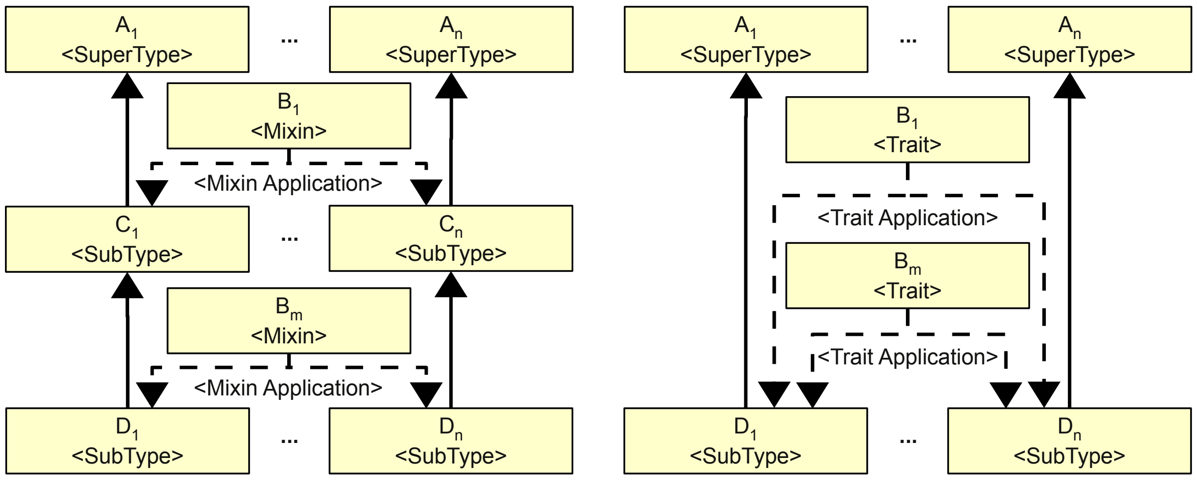
Parametric Subtype (Definition 9) *Parametric type which is defined to take a parameter as its* super type.

In recent years, research on parametric polymorphism has introduced many further nuances like *mixins* [Flatt 98, Smaragdakis 01] or (stateful) *traits* [Schaerli 05, Bergel 06] which offer conceptually novel approaches towards modularisation. Conceptually, *traits* and *mixins* are quite similar. Both constructs encapsulate self-contained functionality which is applicable to a variety of types. The difference lies in how several of these are integrated into one concrete type.

In the general case *parametric subtyping* is an appropriate vehicle for explaining *mixins* and *traits* and realisations often actually use the very same mechanisms for both. Despite the technical analogy to subtyping, *mixins* and *traits* are not meant to have an *is-a*-relationship with any concrete *super type*. These constructs are rather to be regarded as units of add-on functionality which may be applied to a type on demand. In order to clarify the conceptual difference, graphical notations therefore often distinguish *mixin/trait application* from *parametric subtyping* as illustrated in Figure 4.3. In Figure 4.3a the conceptual idea is that `C` results from `B` by deriving from `A`. In contrast the conceptual idea reflected in Figure 4.3b is that `C` results from `A` by applying *mixin/trait* `B`. Note that in both cases type `C` may be of logical nature and needs not be an explicit realisation[5].

In some paradigms, *mixins* may only be applied one after another yielding an unhandy linearisation of types while several *traits* may be applied to a type at a time. In current literature on *mixins* and *traits* in object-oriented paradigms for instance, the two are often distinguished as to their integration modality. In [Schaerli 05] *mixins* are technically coupled with the inheritance mechanism while *traits* represent a more general concept which is not restricted regarding the realisation. As modern object-oriented languages often ban multiple inheritance, *mixins* have to be applied one after another yielding a linearisation of types as illustrated in Figure 4.4a. Inheritance linearisation may result in

[5]In C++ type `C` may for example be a typedef on a (nested) template cascade.



(a) Notation for *mixin application* **(b)** Notation for *trait application*

Figure 4.4: In object-oriented paradigms which ban multiple inheritance *mixins* may only be applied one after another (a). *Traits* do not have this restriction (b).

an unhandy member shadowing, which has to be resolved at the lowest inheritance level. In contrast, multiple *traits* may be applied to one concrete type at a time (see Figure 4.4b). In single inheritance languages, additional language means have to be integrated to support *traits* while *mixins* can be realised with on-board means (e. g. *parametric subtyping*[6]).

Mixin (Definition 10) Mixins *are self-contained collections of functionality which may be deployed with various types. In some paradigms,* Mixins *may only be applied in a linearised fashion.*

Trait (Definition 11) Traits *are self-contained collections of functionality which may be deployed with various types. Multiple* traits *may be pulled into one concrete type at a time.*

Stateful Trait (Definition 12) Stateful traits *are* traits *which incorporate state that is required for the encapsulated functionality.*

[Bergel 06] further extends *traits* from pure function collections to structures which encapsulate state and associated routines in a similar manner as classes. These *stateful traits* provide the conceptual basis for the extensible representation scheme subject to this Doctoral Thesis. In the following, the term *trait* shall be used to subsume the concepts *mixin*, *trait*, and *stateful trait*.

Now that the software technological foundations have been introduced, more details on each partition of the proposed representation scheme shall be given in the order of the design points outlined at the beginning of this section (see page 37 ff).

[6]In that context, the term *mixin inheritance* is often used in spite of the lacking *is-a*-relationship between *super type* and *mixin*.

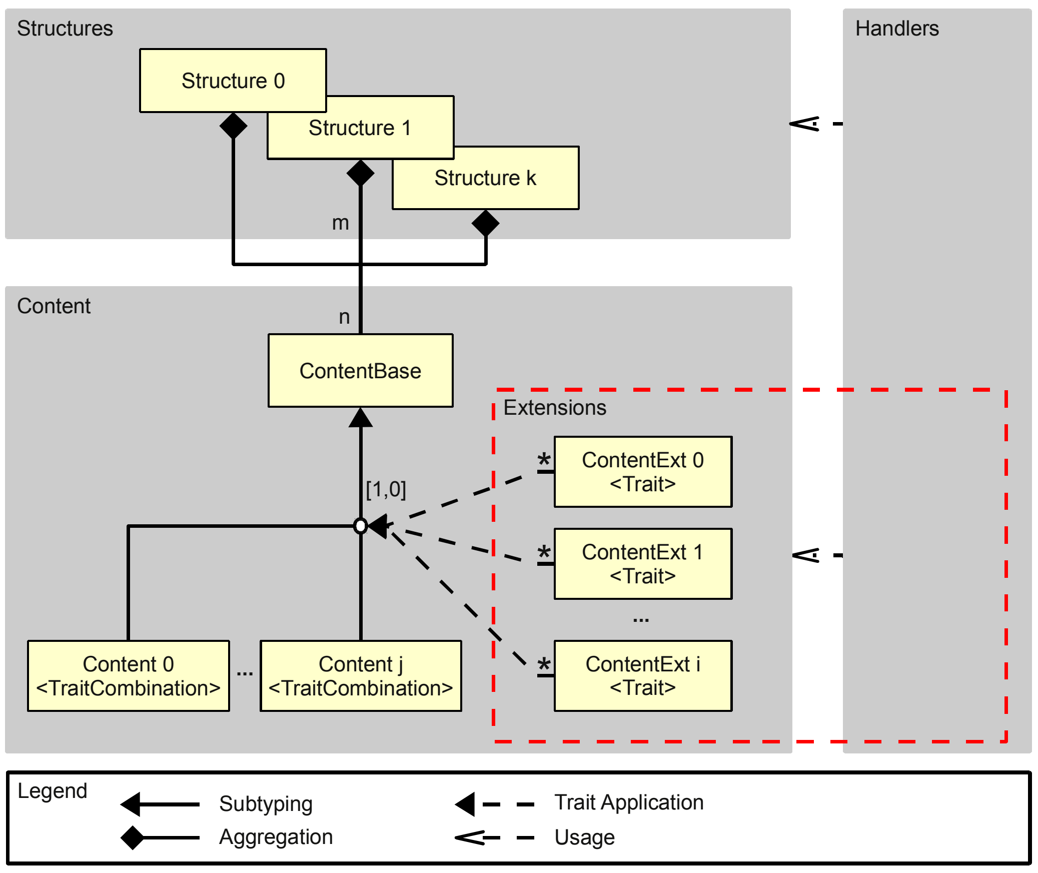


Figure 4.5: Refined representation scheme.

4.1.2 Structure-Content-Duality (Design Point 1)

As a first modularisation step, the separation of locality and semantics shall be highlighted. For that purpose, Figure 4.5 illustrates the refinement of the top-level partitions `Structures` and `Content`. As already mentioned above, a preferably small number of structural arrangements form the basis of the representation scheme (see box `Structures` in Figure 4.5). As this work is focussed on navigation, straightforward structures would for example be topological, grid-based, and sector-based arrangements. Structural entities contain generic content elements (see box `Content` in Figure 4.5) which can be accessed using the standardised structural interfaces (e. g. getContent[AtPosition|InSector|OfNode|OfEdge| . . .]). Structural entities possess full information on content locality which allows for the uniform treatment of coordinate transformations between different structural entities. Note that such transformations – due to the strict separation of structure and content – can be applied independently of content translation on the semantic level.

The choice of base structures is not restricted in any way and further suitable arrangements should be added if (but only if) required. The initial set of arrangements has been chosen to clarify the ideas behind the representation scheme. Some less straightforward representations shall be introduced later on and can be regarded as examples of how suitable arrangements can be integrated into the scheme on demand (see Section 8.3.2 on page 112).

4.1.3 Content Standardisation and Genericity (Design Point 2)

While `structure` holds information on locality, `content` is to represent the semantics tied to entities at a given location. To meet the flexibility requirement, content elements have to be tailorable to represent arbitrary sets of features and context. Nonetheless, further elements of standardisation shall be introduced to allow for generic translation between different semantic levels. That way, interdependencies between layers (e. g. sensor processing and control) can be avoided. The standardised part of content elements is encapsulated into type `ContentBase` while the generic part is distributed over a set of well-defined `ContentExtensions` (see Figure 4.5).

To yield a common interface for all content elements, semantics can as a start be modelled in terms of an n-tuple $PROP = (prop_1, prop_2, ..., prop_n)$ of distinct abstract properties which may be applicable for represented entities or not. For each content element i in a given structure, the property function tuple $PROP(i)$ is defined as:

$$PROP(i) = \big(prop_1(i), prop_2(i), \ldots, prop_n(i)\big) \in \{true, false\}^n$$

where $prop_k(i)$ are Boolean functions that indicate whether a property applies to an entity represented in content element i or not. Note that the $prop_k(i)$ are only restricted as to their value range. Internally the property functions may evaluate the full representation of the provided content element i. The property function tuple $PROP(i)$ can be made persistent in a vector of truth values which can be indexed using the $prop_k \in PROP$. This vector of truth values has to be maintained when updating a content element and can therefore be regarded as a cache for the property functions.

Example 1: Occupancy Grid *To model the content elements for a binary occupancy grid similar to [Borenstein 91, Moravec 96], one single property occupied – telling whether an object was observed within the scope of a content element or not – can be defined:*

$$PROP = (occupied)$$

with

$$PROP(i) = (occupied(i))$$

with

$$occupied\,(i) = \begin{cases} true, & \text{an object was observed within content element } i \\ false & \text{otherwise} \end{cases}$$

However, the author encourages developers to model properties more selectively and naturally to support semantic clarity on all levels of abstraction. *Rich Representations* as shown in Example 2 represent the environment with explicit semantics of what was detected. To a certain degree, such elaborate representations furthermore allow for independent a posteriori interpretation of detected entities for different applications on varying platforms.

Example 2: Rich Representation *A more selective property set designed for an obstacle detection facility might for example look as follows:*

$$PROP = (\mathit{visibleGround},\ \mathit{positiveObstacle},\ \mathit{negativeObstacle},\ \mathit{vegetation},\ \mathit{water})$$

where

$$\mathit{visibleGround}(i) = \begin{cases} \mathit{true}, & \text{ground reference detected within content element } i \\ \mathit{false} & \text{otherwise} \end{cases}$$

$$\mathit{positiveObstacle}(i) = \begin{cases} \mathit{true}, & \text{objects protruding from the ground} \\ & \text{(e. g. a rock or a tree) within content element } i \\ \mathit{false} & \text{otherwise} \end{cases}$$

$$\mathit{negativeObstacle}(i) = \begin{cases} \mathit{true}, & \text{a hole or trench within content element } i \\ \mathit{false} & \text{otherwise} \end{cases}$$

$$\mathit{vegetation}(i) = \begin{cases} \mathit{true}, & \text{vegetation within content element } i \\ \mathit{false} & \text{otherwise} \end{cases}$$

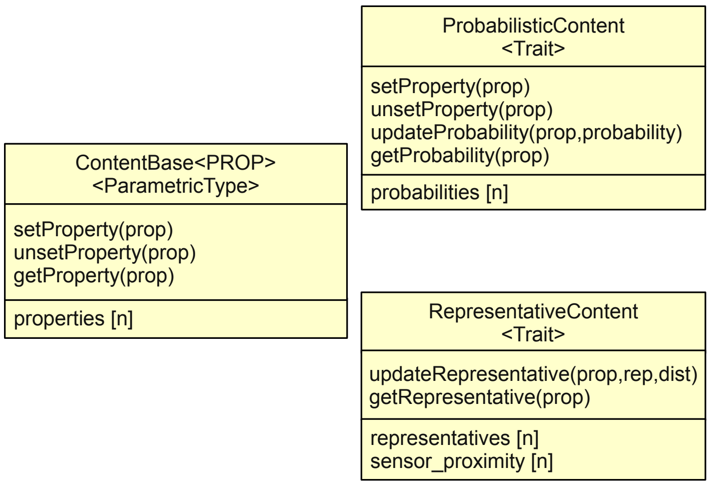
$$\mathit{water}(i) = \begin{cases} \mathit{true}, & \text{water within content element } i \\ \mathit{false} & \text{otherwise} \end{cases}$$

Note that properties in this representation scheme may but do not have to be semantically disjoint. That way, information can be stored at a very fine granularity which allows further interpretation in later processing steps.

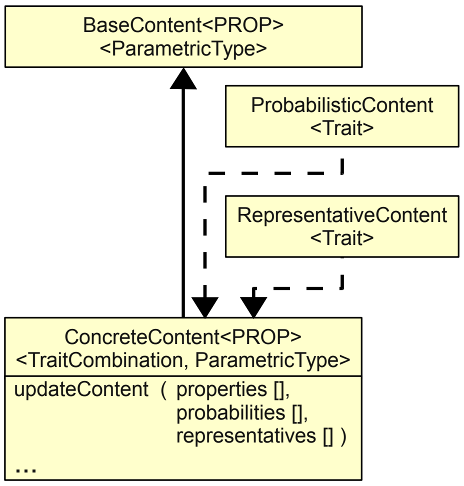
On the basis of abstract properties, the translation between semantics can be modelled as a mapping from one set of properties to another. Such mappings may vary from simple grouping operations to complex logic expressions. To support usability and maintainability, the configuration scheme should be kept as simple as possible. Furthermore, a pool of standard property sets shared by several representations may be introduced to reduce configuration overhead for translating between semantics.

The standardised `ContentBase` is a *parametric type* which takes a property tuple *PROP* as a parameter. That way arbitrary content element types can freely be configured with regard to representable properties without code duplication. All content element types share a common interface which allows for the configurable translation between content elements on different semantic levels. Furthermore, type safety between semantically incompatible content elements (i. e. such content elements which do not share a common property tuple) can already be assured at compile time which eliminates a whole class of potential errors by design.

Self-contained sets of information and associated functionality can be realised in separate independent *traits*. These *traits* can be regarded as fundamental units of a generic building



(a) Trait-based building set where
$PROP = (prop_1, prop_2, ..., prop_n)$
$prop \in PROP$
$n = |PROP|$


BaseContent<PROP> <ParametricType>
ProbabilisticContent <Trait>
RepresentativeContent <Trait>
ConcreteContent<PROP> <TraitCombination, ParametricType>
updateContent (properties [], probabilities [], representatives [])
...

(b) *Trait*-based combination.

Figure 4.6: Sample building set of traits (a) and a *trait*-based combination integrating probabilistic functionality and storage of representatives for each property of a content element (b).

set which may be used to assemble a whole family of representations and related algorithms.

Example 3: Trait-based Content Design *In Figure 4.6a the coarse signature of* ***`ContentBase`*** *and two example* traits *have been depicted to illustrate the proposed separation modality.*

ProbabilisticContent

To model the probability $P(prop) \in [0...1)$ of a content element's property a trait *called* ***`ProbabilisticContent`*** *could be designed. It would provide storage for the probability values together with strategies to compute these probabilities from relative frequency, and to check boundary conditions. In further processing steps the probabilities can be used to fuse several sources of information on an abstract level (see Section 4.3 for further details). The PROP interface provided by class* `ContentBase` *is maintained by threshold evaluation of the respective probability values.*

RepresentativeContent

By default, content elements do not hold any information on locality. It is the structural entities which maintain that kind of information at an accuracy that can be selected freely. In order to increase accuracy, the developer might want to store representatives in terms of concrete absolute coordinates from real sensor readings. Trait ***`RepresentativeContent`*** *provides exactly that portion of data storage and appropriate update*

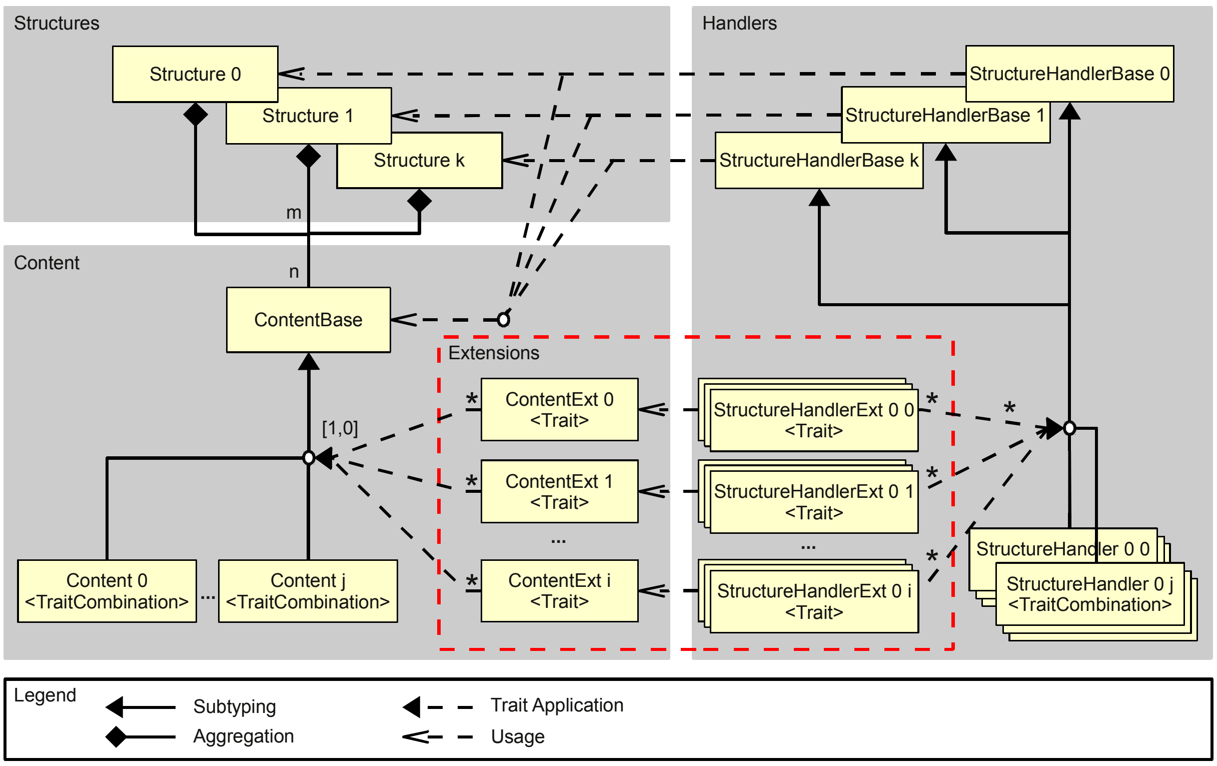


Figure 4.7: Representation scheme featuring handler constructs.

routines. Under the assumption that sensor readings become more accurate the closer the sensor gets to an object, the distance between representatives and sensor could be provided to the update routine. This information can be used to update only those representatives that origin from readings that are closer than previous ones.

Trait application yields a family of types

The sample traits may now be combined with the `ContentBase` as illustrated in Figure 4.6b to yield a new (parametric) type by trait application*. Note that the resulting type is still parametric and may be configured by defining arbitrary property sets for parameter PROP. In that sense, the* trait*-based combination yields a family of content element types rather than one concrete type. In Part II of this work some more complex examples for content* traits *and* trait*-based combinations will be discussed in more detail.*

4.1.4 Handler Standardisation and Genericity (Design Point 3)

Handlers are realised in a similar fashion as content elements. The standardised part is implemented in terms of parametric `HandlerBase` types which take a representation configuration as parameter. The generic part is modularised into `HandlerExtension` traits which may be pulled into a concrete handler to deal with their corresponding

`ContentExtension` *trait*. Figure 4.7 indicates the combination of the `HandlerBase` with suitable `Extensions` to yield concrete handler types. Furthermore, the relationship between `ContentExtensions` and `HandlerExtensions` is highlighted.

Note that a concrete handler type may only provide the basis for handling the content. In sensor processing for example further algorithms are required to extract information from sensor data and to actually fill the representation. In some cases, however, standard combinations of `Handler/ContentBase` and accordant `Content/HandlerExtensions` provide fully functional algorithmic units. One evident class of such functional units are display facilities. The common top-level interface in terms of standardised structure and content may be used for rudimentary drawing. Further details can gradually be added by introducing functionality by deploying further *traits* as outlined above. The two most prominent examples attack *Semantic Abstraction* (Design Idea 4) and *Abstract Fusion* (Design Idea 5). Gradual semantic abstraction and abstract fusion can be regarded as the core design aims of this Doctoral Thesis. The following sections will deal with translating information from one semantic level to information on another semantic level and how semantically compatible information can be fused in a generic fashion.

4.2 Abstraction Scheme Design and Semantic Translation

The fundamental design idea of the representation scheme at hand is that representation should be as natural as possible for a given purpose. This results in tailored property sets *PROP* for particular components as these work on different semantic levels. Property sets should be derived from their particular purpose in a straightforward fashion to contribute to traceability in complex control systems.

Note that different property sets are semantically incompatible and may further be enriched with additional context which may be incomparable, as well. For that reason, a mechanism to translate between different semantic levels is required. Location does not play any role in this context as coordinate transformations can be handled independently from content as mentioned before.

The translation from a source content element to a more abstract destination content element is first of all reduced to a transition from the source property set *srcPROP* to the destination property set *dstPROP*. Formally this transition can be described as:

$$\mathit{srcPROP} \rightarrow \mathit{dstPROP}$$

where

$$\begin{aligned} \mathit{srcPROP} &= (\mathit{srcProp}_1, \ldots, \mathit{srcProp}_n) \qquad n \in \mathbb{N} \\ \mathit{dstPROP} &= (\mathit{dstProp}_1, \ldots, \mathit{dstProp}_m) \qquad m \in \mathbb{N} \end{aligned}$$

For each destination property $\mathit{dstProp}_k$, the property function $\mathit{dstProp}_k(j)$ – of a given destination content element j – is defined as the translation function $\mathit{translate}_k(\mathit{srcPROP})$.

$$\mathit{dstProp}_k = \mathit{translate}_k(\mathit{srcPROP}) \qquad \forall k \in [1, m]$$

The translation functions are configured according to a configuration scheme which will be highlighted in the following. In order to illustrate the concepts of semantic abstraction, an intuitive series of escorting examples shall be given along the development of the theory (see Example 4).

Example 4: Sense-Control-Duality *As already alluded above, representation plays a different role in sensor processing and control (see Design Idea 3). To stick with the example from Section 4.1.3, sensor processing designers think in terms of obstacle classes like:*

$$PROP_{sensor} = (ground,\ positive,\ negative,\ vegetation,\ water)^{7}$$

Control system designers however tend to be rather interested in traversability, which might lead to the following property set:

$$PROP_{control} = (traversable,\ non\text{-}traversable)$$

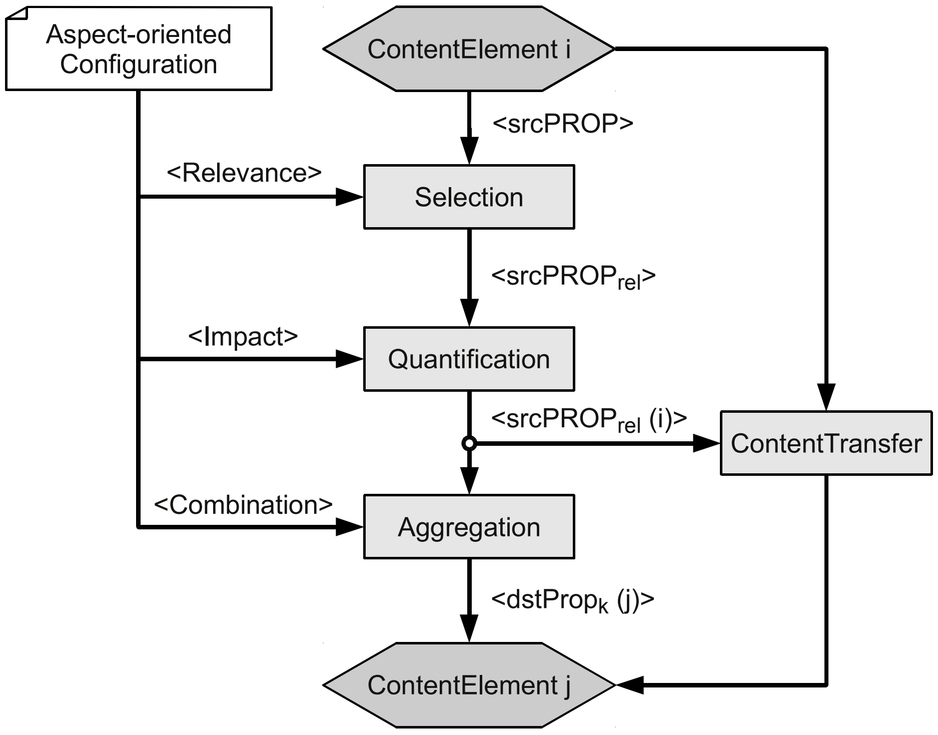


Figure 4.8: Semantic translation selects, quantifies, and aggregates property values of a source content element i into a destination element j. Additional content is transferred between content elements in a parallel operation.

[7]Straightforward abbreviations for properties with long names shall be used from here forth for readability reasons.

Figure 4.8 illustrates the proposed translation process. The translation is organised into the three major steps `Selection`, `Quantification`, and `Aggregation` which are tailored using the central `Aspect-oriented Configuration`. Furthermore, step `ContentTransfer` migrates possible extended content in parallel to the main translation process. In the first place basic capabilities for translating the semantics of a content element i to the semantics of a content element j are realised on the basis of the *PROP* interface. For each destination property $dstProp_k \in dstPROP$ of content element j, the property function $dstProp_k(j)$ is constructed from content element i as follows. In the `Selection` step, relevant properties are masked from the source property set *srcPROP* resulting in the filtered source property set $srcPROP_{rel}$. `Quantification` then determines which of the relevant properties actually apply to content element i resulting in a vector of truth values which can be computed from the property function subset $srcPROP_{rel}(i)$. Finally, the output of the $srcPROP_{rel}(i)$ is aggregated according to a logical term specified in the `Aspect-oriented Configuration` to yield the output of $dstProp_k(j)$.

The procedure outlined above can be regarded as the standard behaviour of the translation mechanism. Note that here as well each step may be extended via *traits* for the translation facility. Nesting several *traits* results in a tailored translation mechanism which reflects the full spectrum of the deployed representation. For the `ProbabilisticContent` trait introduced above, a corresponding `SemanticTranslation` *trait* might implement a weighted update as an overlay for step `Aggregation` to extend the standard procedure. In that context, the aggregated probabilities for the destination properties $dstProp_k$ could also be transferred into the destination content j. Such an extension also allows for the realisation of an abstract fusion mechanism which shall be discussed in the next section.

Example 4 (continued): Sense-Control-Duality *Continuing the example from above, the configuration for an autonomous land vehicle could be designed to reflect the following logical expressions:*

$$\begin{aligned} traversable &= ground \wedge \neg positive \wedge \neg negative \wedge \neg water \\ non\text{-}traversable &= positive \vee negative \vee water \end{aligned}$$

Note that source properties which have not yet been configured are implicitly ignored. As the control system is designed to take its decisions on the basis of the abstract traversability properties, extending the source property set does not have any direct impact on the control layer. This fact allows for the gradual refinement of representation with evolving algorithms without permanent consultations for negotiating interfaces.

Example 4 (continued): Sense-Control-Duality *If the hypothetical obstacle detection facility is improved such that tree branches hanging into the path can be detected this novel obstacle conformation could be mapped to a new property named overhanging. As soon as this additional feature extraction capability is declared "stable", sensor and* *control designers agree on the integration of the property which might yield the following translation configuration:*

$$\begin{aligned} \textit{traversable} &= \textit{ground} \wedge \neg \textit{positive} \wedge \neg \textit{negative} \wedge \neg \mathbf{overhanging} \\ \textit{non-traversable} &= \textit{positive} \vee \textit{negative} \vee \textit{water} \vee \mathbf{overhanging} \end{aligned}$$

That way the point in time when the robot actually uses new extraction mechanisms can clearly be defined. This allows for separate integration testing in a parallel development process supporting large teams of developers.

Configurable abstraction through semantic translation further facilitates control (sub)system sharing among platforms. As already alluded above, concretely detectable features are sensor and algorithm dependent. Abstract concepts like traversability which are required on the control level are platform dependent. While an unmanned ground vehicle should avoid larger water bodies at any price an amphibian vehicle may simply swim through them without being harmed. In the example from above, large parts of the control system may be reusable only by changing the classification of property `water` in the translation configuration as follows:

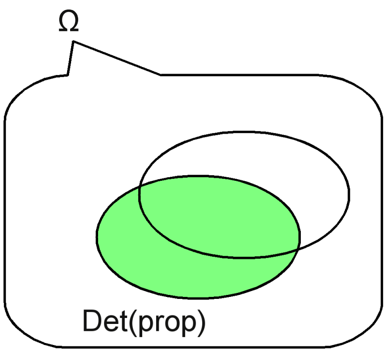
Example 4 (continued): Sense-Control-Duality

$$\begin{aligned} \textit{traversable} &= (\textit{ground} \vee \mathbf{water}) \wedge \neg \textit{positive} \wedge \neg \textit{negative} \wedge \neg \textit{overhanging} \\ \textit{non-traversable} &= \textit{positive} \vee \textit{negative} \vee \textit{overhanging} \end{aligned}$$

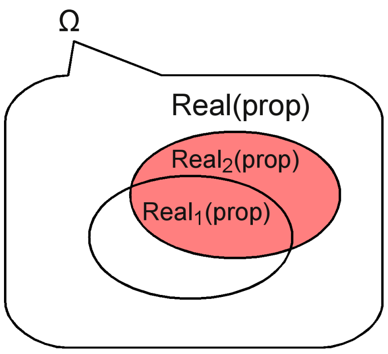
As water bodies do not represent a hindrance for an amphibian vehicle, *water* plays a similar role as visible *ground*. Both properties may be regarded as a negotiable "medium" which may be populated with obstacles. Hence, *water* has to be removed from the logical expression for *non-traversable* and added to the expression for *traversable* at the same level as *ground*. Informally speaking, ground and water alike are traversable in case no obstacles are present.

4.3 Fusion Scheme Design and Abstract Fusion

Semantic abstraction allows for gradually adapting the semantics of different information sources. That way, information from different representations are rendered semantically compatible which is a core requirement for generic data fusion. In addition to the translation scheme and the probabilistic representation scheme, a generic fusion scheme is required in order to make data integration configurable. This section introduces the abstract fusion mechanism which realises central concepts – namely *Deferred Fusion* (Design Idea 2 on page 28) and *Abstract Fusion* (Design Idea 5 on page 30) – required in the design methodology proposed in this work.



(a) let *Det*(*prop*) the event that property *prop* is detected



(b) let *Real*(*prop*) the event that property *prop* really applies

Figure 4.9: $Real(prop)$ consists of two disjoint sets $Real(prop)_1$ and $Real(prop)_2$ where the former is directly observable while the other has to be derived indirectly. Ω demotes the set of all possible outcomes.

The foundation of any data fusion method is a probabilistic framework which allows to judge data reliability. That way, different information sources can be related to one another such that a combined information basis can be built up. For that purpose, input data in terms of property sets have to be enriched with probabilities. These probabilities have to be modelled, represented, and merged. In the following, the formal basis of the probabilistic extension mentioned in Section 4.1.3 shall be introduced. At first, probabilistic modelling of detection facility characteristics will be derived. After that, the update mechanism of the representation will be illuminated before going into detail with merging several sources of information.

4.3.1 Probabilistic Models for Detection Facilities

In order to model the uncertainty of unstructured environments, properties have to be enriched with probability values. For that purpose, the probabilistic extension briefly introduced above was proposed to intercept calls to `setProperty (prop)` and `unsetProperty (prop)` for integrating relative frequency computations. From the relative frequencies, probabilities can be computed in a straightforward fashion. In practice, probabilities enriching properties computed by different detection facilities (developed in-house or third party) tend to result from heuristics and arbitrarily chosen metrics. These probabilities therefore rather represent confidence values than true probabilities and are thus difficult to compare.

Before going into detail with merging probabilistic data over time and over multiple sources, probabilistic modelling itself shall be discussed. To improve detection selectivity and comparability sensor characteristics can be modelled. A detection facility is a component that encapsulates sensor hardware and detection routines which compute probabilities $P(prop_i)$ that properties $prop_i$ apply based on data from the sensor hardware in question. These compounds of sensor hardware and algorithms can be evaluated statistically with respect to observed properties such that the observed probabilities more accurately reflect real conditions.

let $Det(prop)$ the event that property *prop* is detected by a detection facility

let $Real(prop)$ the event that property *prop* actually applies in reality

$Det(prop)$ represents what is observable from sensor data while $Real(prop)$ is what actually should be observed as it represents reality (see Figure 4.9). Probability theory provides the tools to model the environment on the basis of realistic uncertainties.

As illustrated in Figure 4.9b, $Real(prop)$ is composed of two disjoint sets:

$$Real(prop) = Real(prop)_1 \,\dot{\cup}\, Real(prop)_2 \tag{4.1}$$

$Real(prop)_1$ and $Real(prop)_2$ can be traced back to observations from sensor data represented through $Det(prop)$ as follows:

$$\begin{aligned} Real(prop)_1 &= Real(prop) \cap Det(prop) \\ Real(prop)_2 &= \overline{Det(prop)} \setminus \big(\overline{Real(prop)} \cap \overline{Det(prop)}\big) \end{aligned}$$

yielding

$$Real(prop) = \big[Real(prop) \cap Det(prop)\big] \,\dot{\cup}\, \Big[\overline{Det(prop)} \setminus \big(\overline{Real(prop)} \cap \overline{Det(prop)}\big)\Big] \tag{4.2}$$

As already stated above, $Real(prop)_1$ and $Real(prop)_2$ are disjoint sets. Hence, the probability $P\big(Real(prop)\big)$ is the sum of the individual probabilities. Furthermore, $\overline{Det(prop)}$ is a strict super set of $\big(\overline{Real(prop)} \cap \overline{Det(prop)}\big)$ which allows the computation of $P(Real(prop)_2)$ by subtraction of the individual probabilities.

$$\begin{aligned} P\big(Real(prop)\big) &= P\big(Real(prop)_1\big) + P\big(Real(prop)_2\big) \\ P\big(Real(prop)_2\big) &= P\big(\overline{Det(prop)}\big) - P\big(\overline{Real(prop)} \cap \overline{Det(prop)}\big) \end{aligned}$$

yielding

$$\begin{aligned} P\big(Real(prop)\big) = {} & P\big(Real(prop) \cap Det(prop)\big) + \\ & P\big(\overline{Det(prop)}\big) - \\ & P\big(\overline{Real(prop)} \cap \overline{Det(prop)}\big) \end{aligned} \tag{4.3}$$

So far the derivation for estimating $P\big(Real(prop)\big)$ still contains unknown compounds which depend on $Real(prop)$. From theory on conditional probability, the following rule applies:

$$\begin{aligned} P(B|A) &= \frac{P(B \cap A)}{P(A)} \\ P(B \cap A) &= P(B|A) \cdot P(A) \end{aligned}$$

For the cases in question this yields substitutions which are appropriate for solving this problem.

$$\begin{aligned} P\big(Real(prop) \cap Det(prop)\big) &= P\big(Real(prop)|Det(prop)\big) \cdot P\big(Det(prop)\big) \\ P\big(\overline{Real(prop)} \cap \overline{Det(prop)}\big) &= P\big(\overline{Real(prop)}|\overline{Det(prop)}\big) \cdot P\big(\overline{Det(prop)}\big) \end{aligned}$$

In order to prevent false positives and false negatives, the characteristics of detection facilities have to be modelled.

Note that $P\big(Real(prop)|Det(prop)\big)$ denotes the probability that a property actually applies in reality under the condition that the property was detected. From the point of view of statistics $P\big(Real(prop)|Det(prop)\big)$ is the probability for *true positives* (positive detection rate) which can be determined in an empirical fashion. The same holds for $P\big(\overline{Real(prop)}|\overline{Real(prop)}\big)$ which is basically the probability for *true negatives* (negative detection rate). Both detection rates can be computed from statistical measures as follows:

> let $|positives|$ the number of positive detection events for a given property independently of whether the property was actually present in reality or not
>
> let $|true\ positives|$ the number of positive detection events for a given property if and only if the property in question was actually present in reality
>
> let $|negatives|$ the number of negative detection events for a given property independently of whether the property was actually present in reality or not
>
> let $|true\ negatives|$ the number of negative detection events for a given property if and only if the property in question was actually not present in reality

$$P\big(Real(prop)|Det(prop)\big) = \frac{|true\ positives|}{|positives|} \tag{4.4}$$

$$P\big(\overline{Real(prop)}|\overline{Det(prop)}\big) = \frac{|true\ negatives|}{|negatives|} \tag{4.5}$$

Note that the detection rates need not be simple values. In order to create more sophisticated sensor models, further parameters can be taken into account (e. g. distance to sensor, detection angle, or any other parameter which may have an influence on detection rates). In experimental series, a performance map is built up which serves as a data base for a lookup function which determines the appropriate detection rate on the fly using the current parameter configuration.

$$\begin{aligned} P\big(Real(prop)\big) = {} & P\big(Real(prop)|Det(prop)\big) \cdot P\big(Det(prop)\big) + \\ & \big(1 - Det(prop)\big) - \\ & P\big(\overline{Real(prop)}|\overline{Det(prop)}\big) \cdot \big(1 - Det(prop)\big) \end{aligned} \tag{4.6}$$

Returning to the formulae the detection rates are thus empirically determined factors which moderate the detection probabilities such that comparability is established.

4.3.2 Accumulation of Sensor Information over Time

Probability enhanced *Content* as introduced above can be fused to yield a combined data basis. For that purpose, the overload methods `setProperty (prop)` and `unset-Property (prop)` intercept calls for integrating relative frequency computations. In case probabilities are available through metrics from sensor models (e. g. [Miura 02, Thrun 03]), method `updateProbability (prop, probability)` can be used to update the probability of a particular property `prop`.

Probability enhanced *Content* as introduced above can be fused to yield a combined data basis. Besides the provided current `probability` that property `prop` applies to the content element in question, the update function depends on the internal accumulated probability represented in the content element. An update function must assure the following three criteria:

1. **Accumulation Criterion**: Accumulated probability increases with repeated detection of the property in question.
2. **Decay Criterion**: Accumulated probability decreases when the property in question is repeatedly not detected.
3. **Saturation Criterion**: Accumulated probability remains a valid probability which is usually in range $[0, 1]$.

The functionality of the probabilistic extensions proposed here are inspired by early works on probabilistic occupancy grids. In [Elfes 87], the occupancy of each cell is regarded independently breaking the high-dimensional mapping problem down to a set of one-dimensional problems. Each cell has two antagonistic properties *occupied* and *empty* with accordant accumulated probabilities $P_{cum}(occupied)$ and $P_{cum}(empty)$ with:

$$P_{cum}(empty) = P_{prev}(empty) + P_{cur}(empty) - P_{prev}(empty) \cdot P_{cur}(empty) \tag{4.7}$$

$$\begin{aligned} P_{cum}(occupied) = P_{prev}(occupied) &+ P_{cur}(occupied) \cdot \big(1 - P_{cum}(empty)\big) - \\ &P_{prev}(occupied) \cdot P_{cur}(occupied) \cdot \big(1 - P_{cum}(empty)\big) \end{aligned} \tag{4.8}$$

where P_{prev} denotes the previous probability and P_{cur} the current probability of the cell being empty or occupied respectively.

Mathematically the basic update function can be derived as follows:

> Let $Det_{cur}(prop)$ the event that property *prop* was detected on the basis of the current sensor reading.
> Let $Det_{prev}(prop)$ the event that property *prop* was detected in previous sensor readings.

Then the cumulated probability $P_{cum}(prop)$ of having detected property *prop* at a given location can be denoted as:

$$\begin{aligned} P_{cum}(prop) &= P\big(Det_{prev}(prop) \cup Det_{cur}(prop)\big) \\ &= P\big(Det_{prev}(prop)\big) + P\big(Det_{cur}(prop)\big) - P\big(Det_{prev}(prop) \cap Det_{cur}(prop)\big) \end{aligned} \tag{4.9}$$

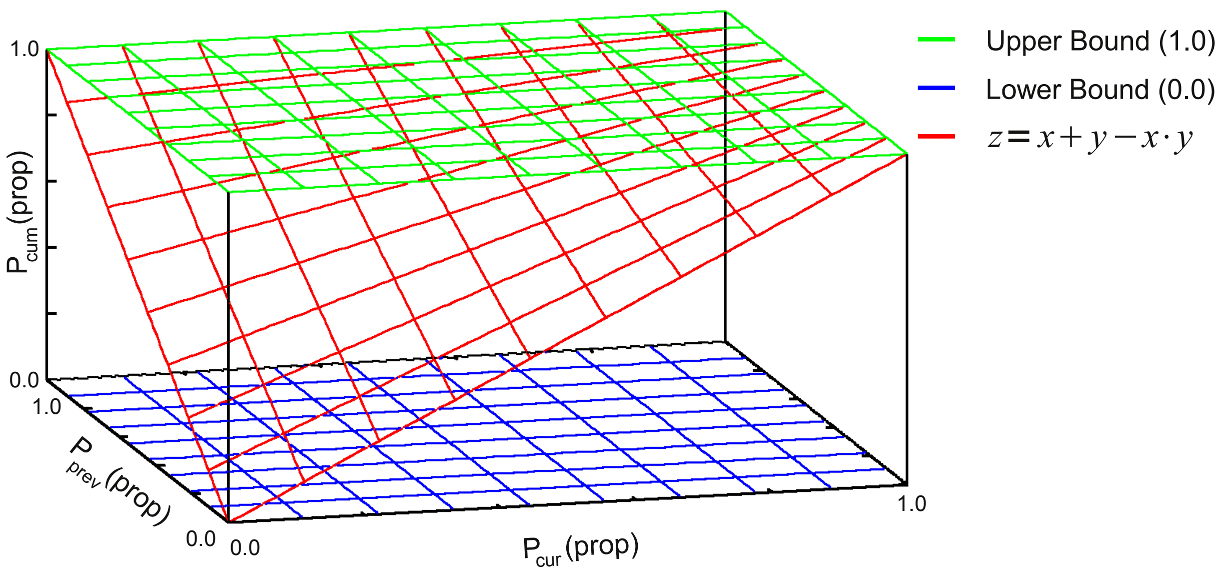


Figure 4.10: Probabilistic update function with two parameters x and $y \in [0,1[$ where $x = P_{cur}(prop)$ and $y = P_{prev}(prop)$ as proposed in [Elfes 87].

$Det_{prev}(prop)$ and $Det_{cur}(prop)$ are semantically dependent as both are related to the very same property *prop*. Therefore, the addition of both, the cumulated and the current probability for property *prop* may yield values outside interval $]-1,1[$ as there is a certain probability that an object detected at a certain place remains there over time. This is the case if the object in question is stationary. $P\big(Det_{prev}(prop)\big)$ and $P\big(Det_{cur}(prop)\big)$ thus overlap to a certain degree and this overlap has to be removed with a correction term which reflects the combined probability of a property over time: $P\big(Det_{prev}(prop) \cap Det_{cur}(prop)\big)$.

Though semantically dependent, $Det_{prev}(prop)$ and $Det_{cur}(prop)$ can be regarded statistically independent events – in the sense that every new reading provides new information about objects at a given location. Under this assumption, the correction term representing the combined probability of a property over time can be computed as the product of the particular probabilities. For readability reasons the particular probabilities

$P\big(Det_{prev}(prop)\big)$ will be written as $P_{prev}(prop)$ and
$P\big(Det_{cur}(prop)\big)$ as $P_{cur}(prop)$.

In the discrete world of computer systems the cumulated probability is recomputed at well-defined time steps t when new sensor data is available. At a given point in time t the previous probability $P_{prev}(prop)_t$ is thus the cumulated probability $P_{cum}(prop)_{t-1}$ of the time step before, yielding the following update function which is visualised in Figure 4.10:

$$P_{cum}(prop)_t = P_{cur}(prop)_t + P_{cum}(prop)_{t-1} - P_{cur}(prop)_t \cdot P_{cum}(prop)_{t-1} \qquad (4.10)$$

In [Elfes 87] the computation of $P(occupied)$ is dependent on $P(empty)$ (see Equation 4.7 and 4.8). This dependence adds a factor to relate the antagonistic properties with one another in an ad hoc fashion. In the proposed representation scheme, interdependencies

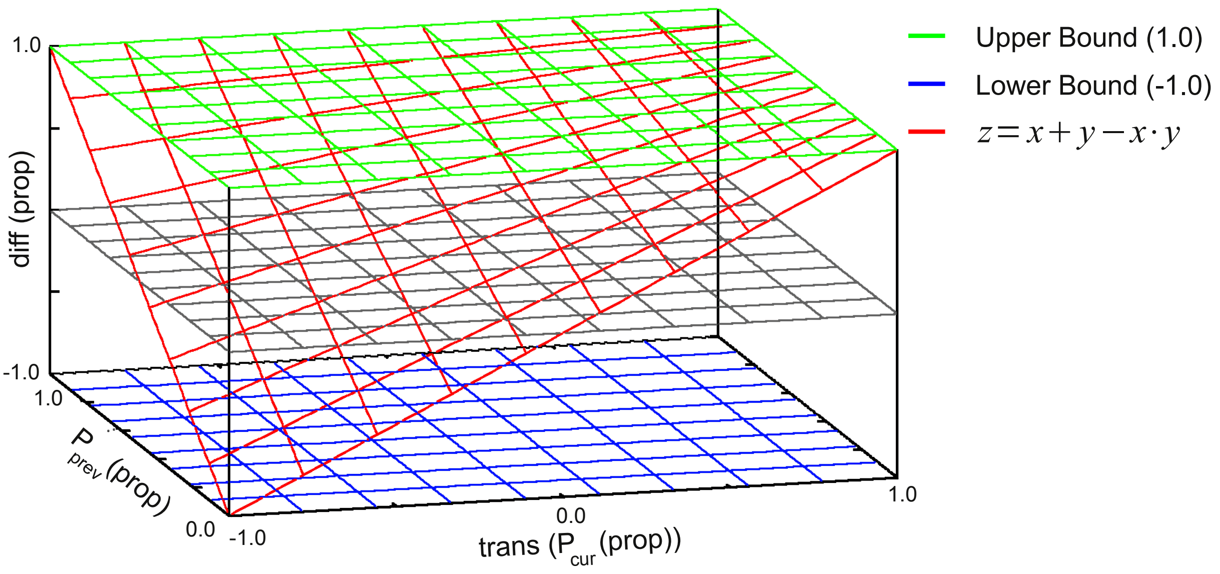


Figure 4.11: Probabilistic update function with parameter $x \in \,]-1,1[$ and $y \in \, [0,1[$ where $x = P_{cur}(prop)$ and $y = P_{prev}(prop)$.

between properties are difficult to model as arbitrary property sets are allowed. By design interaction between properties is deferred to the semantic translation step to prevent premature assumptions on property interpretation. Furthermore, the absence of abstract properties is difficult to model. For the determination of mere occupancy from range data, sensor characteristics can be called into service. The distance from the sensor to an object can be regarded as free space and at the distance in question the probability for occupancy has to be increased. For more complex detection mechanisms the negative case is difficult to judge. A further short-coming of the approach proposed by Elfes et al. is the static world assumption. As illustrated in Figure 4.10, this update mechanism meets the criteria accumulation (Criterion 1) and saturation (Criterion 3) but lacks the possibility to model changes in the environment by decaying belief (Criterion 2). In case of non-stationary environments, the probabilities of both antagonistic properties saturate over time and lose significance.

For these reasons, Elfes' cell update mechanism was revised with some constraints to meet all criteria. As already alluded above, interdependencies between properties shall be avoided for design reasons. Therefore, antagonistic property pairs shall be integrated into single properties. This approach further allows the extension towards multiple properties as intended in the proposed representation scheme. In order to account for dynamic environments the input value range of the update function shall be extended to interval $]-1,1[$. That way the absence of a certain property can intuitively be modelled with negative values such that input values to the update function can be regarded as probabilistic differences or quasi probabilities to be applied to a cumulated probability.

Input probabilities yielded from detection algorithms, which are usually in interval $[0,1]$, can be cast into the appropriate value range by configurable offsetting, scaling or more complex functions to model sensor characteristics. These transformations should be separated from the actual update function for software technological reasons. This shall be indicated by applying a configurable transformation function $trans(P_{cur}(prop))$ to input probabilities. While the update function is a general tool to aggregate probabilities over time, semantic borders, and representational borders, the preparative transformations carried out in function $trans(P)$ are dependent on particular sensor systems and evaluation algorithms. As common probabilistic theory does not support negative probabilities[8], the configurable preparative transformations are regarded as plug-ins to the update function which are actually part of the update mechanism. That way, probabilistic theory is not violated although intermediate values $trans(P_{cur}(prop)_i)$ are not actually valid probabilities. The proposed update function is a slight modification of Elfes' approach which bounds the cumulated probability $P_{cum}(prop) \in [0,1[$ as follows:

$$P_{cum}(prop)_i = max\big(0, diff(prop)_i\big) \tag{4.11}$$

where

$$diff(prop)_i = trans\big(P_{cur}(prop)_i\big) + P_{cum}(prop)_{i-1} - trans\big(P_{cur}(prop)_i\big) \cdot P_{cum}(prop)_{i-1}$$

As illustrated in Figure 4.11 the proposed modifications yield an update function which meets all three criteria. Note that $diff(prop)$ is in interval $]-1,1[$ which allows for the hierarchical application of the content element update function for multiple purposes. The intermediate cumulation result $diff(prop)$ can be interpreted as a probabilistic difference – just like $trans\big(P_{cur}(prop)\big)$ – which can be accumulated on the next higher level of abstraction.

4.3.3 Abstract Fusion of Information from Multiple Sources

Besides the cumulation of information – on a given property *prop* – from a single sensor system over time, fusion of data from multiple sources is often required to create a more complete view on the world. As already mentioned above, the fundamental approach in this work is to fuse information on an abstract level rather than on the level of raw sensor data. Besides being less vulnerable against data registration and localisation inaccuracies, *Abstract Fusion* can further be modelled with the very same update mechanism as data accumulation over time.

Abstract Fusion is carried out in two stages. In the first stage, information from the sources to fuse is rendered semantically comparable by semantic translation (remember Section 4.2). After the translation to a common semantic level, readings from different sensor systems can be regarded as readings from a common perception system. The probabilities of semantically comparable entities may therefore be combined in a similar way as if the data was yielded from one and the same source. In order to rank the data sources according to general perception capabilities, the proposed fusion scheme provides a set of three general **Impact Modes**. Each data source is assigned **Impact Modes** for each destination property, which specifies how the property in question contributes to the fused property:

[8][Dirac 42] introduces the concept of negative energy and negative probability which are still intensely discussed concepts in the quantum mechanics community today.

1. **No Impact**: The property from the source in question shall be ignored and does thus not contribute to the fused property.
2. **Full Impact**: The property from the source in question contributes with its full probability to the fused property.
3. **Overrule All Impact**: The property from the source in question overrules all previous readings (i. e. the cumulated probability stored in the target content element is ignored).

The **Impact Modes** allow for a high-level distinction of the capabilities of different detection facilities. Some sensors for instance in principle cannot contribute to all properties of a fused data basis. Imagine a planar laser range finder mounted horizontally to measure distances to objects. To build up a 3D representation of the world, destination property *traversable* cannot be decided as only a single plane is measured. In contrast, property *non-traversable* can be decided from this data. Therefore, the 2D-laser-based data source should be configured to have **No Impact** on property *traversable* and **Full Impact** on property *non-traversable*. The third mode is designed for obstacle detection facilities that „cannot fail". If a robot passes over an area, this area is apparently traversable; no matter what other sensor systems may have detected beforehand. This information source is highly reliable and should therefore be configured to have **Overrule All Impact** on property *traversable*. For a more elaborate example of *Abstract Fusion* the reader shall be referred to Section 10.2 in Part II.

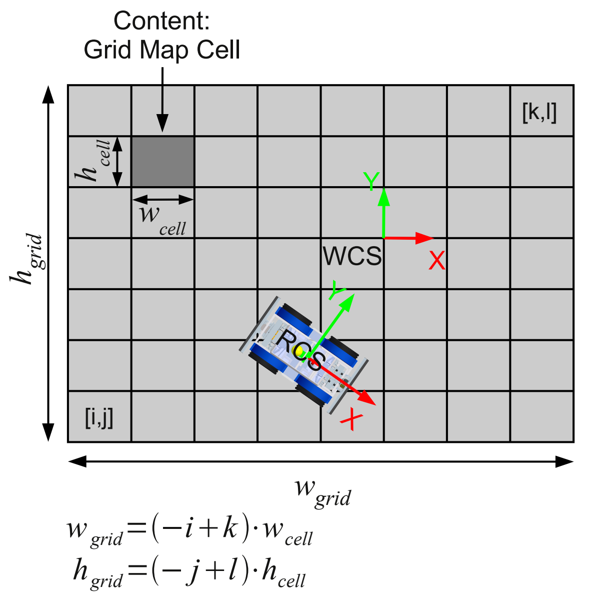
4.4 Specification of Structures

The representation scheme introduced in Section 4.1 proposes a strict separation of `Structure` and `Content` in order to distinguish clearly between locality and semantics of represented entities (see *Structure-Content-Duality* (Design Idea 1)). Modelling, abstraction, and fusion of semantic `Content` was the focus of the preceding sections. In this section the formal definition and common usage of `Structures` relevant for this work shall be specified to complete the design schemata.

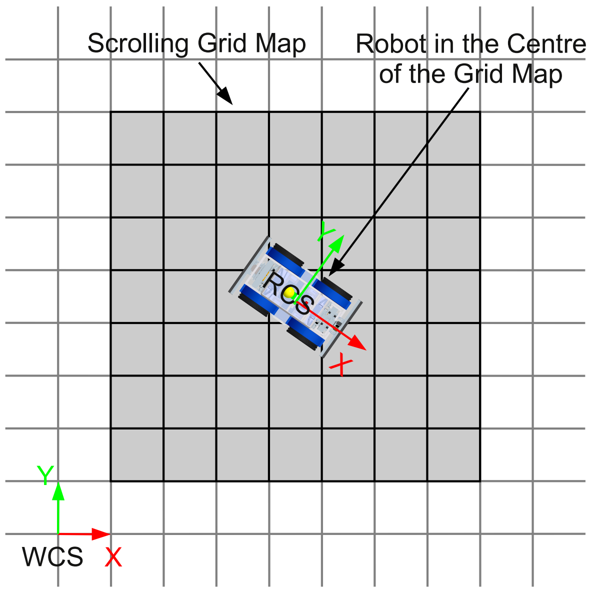
As already alluded above, this work concentrates on environment perception and local navigation aspects. Navigation problems on a global scale, like a priori selecting a sensible route are not part of this Doctoral Thesis. The core `Structures` relevant for this work are grid-based or sector-based. Details on the definition of topological `Structures` in the context of the proposed representation scheme can be found in [Braun 09a].

4.4.1 Grid-based Structures

A grid can be defined by the cell coordinates of two corners, p_{min} and p_{max}, yielding a rectangular structure. Furthermore, the cells are assigned a width w_{cell} and height h_{cell} which defines the resolution of the *Structure*. In this work, grid maps are always orientation-aligned with the robot's Working Coordinate System (WCS). Grid maps can either be defined as static global structures in which the robot moves (Figure 4.12a) or as dynamic local structures which move with the robot in a scrolling fashion (Figure 4.12b). When the robot moves, the scrolling mechanism adjusts the grid map by clipping row- and



(a) Global grid map: Absolute WCS orientation-aligned grid-based structure in which the robot moves.



(b) Local scrolling grid map: Absolute WCS orientation-aligned grid-based structure moving with the robot.

Figure 4.12: Grid-based *Structures* can be configured to represent global context (a) or local context (b). The reference frame is always given by the robot's working coordinate system (WCS).

column-wise in direction of the WCS axes such that the robot remains in the centre of the map (see Figure 4.12b). This behaviour can be achieved with simple modulo operations and selective clearing of clipped cells which yields a computationally efficient data structure. That way, the representation's memory footprint is constant and access is computationally feasible as only the vicinity of the robot is monitored. In large environments, scrolling local maps therefore provide a memory efficient way to represent the vicinity of the robot in a fine-granular and rich way. To distinguish between *global* and *local* maps, an additional *mode* flag is provided by the grid map specification.

In the proposed representation scheme, grid-based *Structures* can thus completely be defined by the formal specification given below:

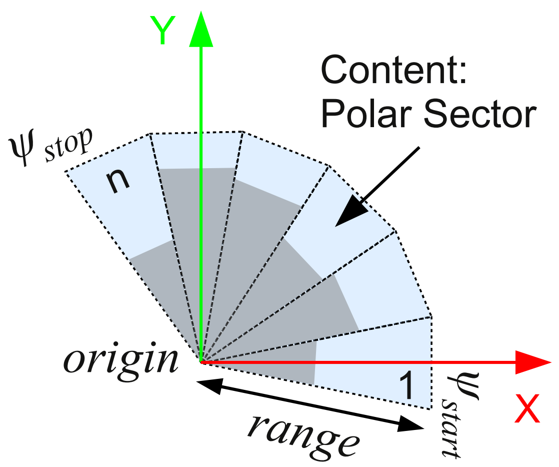
$$GridMap := (p_{min}, p_{max}, d_{cell}, mode) \tag{4.12}$$

where

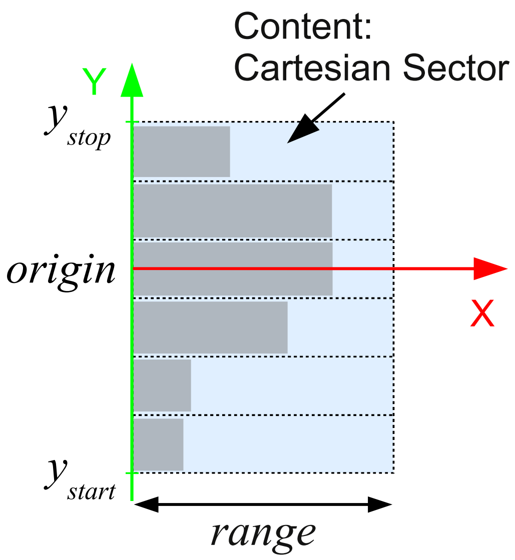
$p_{min} = (i, j) \in \mathbb{N}^2$	*specifies the lower left corner of the grid*
$p_{max} = (k, l) \in \mathbb{N}^2$	*specifies the upper right corner of the grid*
$d_{cell} = (w_{cell}, h_{cell}) \in \mathbb{R}^2$	*width and height covered by one grid content element*
$mode \in \{global, local\}$	*indicates whether the grid is globally fixed or moves locally with the robot*

4.4.2 Sector-based Structures

In this work, sector-based *Structures* occur in two different arrangements, namely as sector maps in polar format and sector maps in Cartesian format as illustrated in Figure 4.13. These configurations have been regarded as most suitable for describing relevant regions of the environment in a simplified and formal way.



(a) Sector-based structure with polar sector arrangement.



(b) Sector-based structure with Cartesian sector arrangement.

Figure 4.13: Polar (a) and Cartesian (b) sector-based representations.

Polar maps are defined by start and stop angles (ψ_{start} and ψ_{stop}) whereas Cartesian maps have an extent in positive and negative y-direction (y_{min} and y_{min}). Both structures allow for the specification of the number of sectors n (i. e. the sampling) and the *range* of the map. Each sector is represented by generic *Content* which can be tailored to the requirements of the particular application. By default, *Content* fulfils the *PROP* interface and for sector maps each sector holds the most prominent representative in the form of a polar or Cartesian coordinate. Further details may be added with the extension mechanism explained in Section 4.1. That way, sector maps might for example also be configured to monitor a certain height range, such that 3D coverage in terms of cuboids (Cartesian) or wedges (polar) can be achieved (see Section 9.3.4 in Part II). As for grid-based *Structures*, sector-based *Structures* may also be defined to capture local or global context by specifying a *mode* flag accordingly. The anchor pose specifying the *origin* of the sector map is interpreted relative to the Robot Coordinate System (RCS) in local *mode* and relative to the WCS in global *mode*.

In the proposed representation scheme, sector-based *Structures* can be defined by the formal specifications given below:

$$PolarSectorMap := (origin, \psi_{start}, \psi_{stop}, \Delta\psi, range, mode) \tag{4.13}$$

$$CartesianSectorMap := (origin, y_{start}, y_{stop}, \Delta y, range, mode) \tag{4.14}$$

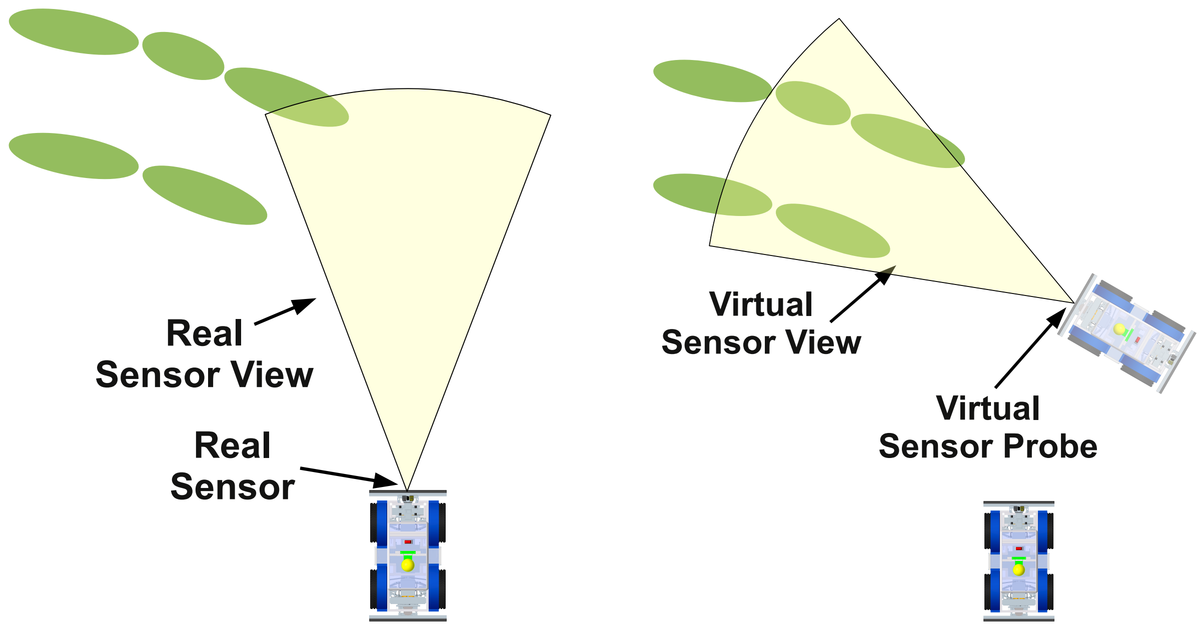
where

$origin = (x, y, \gamma) \in \mathbb{R}^3$	*pose specifying where the structure is anchored*
$range \in \mathbb{R}$	*specifies the range of the sectors in distance to the origin*
$mode \in \{global, local\}$	*indicates whether the sector map is globally fixed or moves locally with the robot*
$\psi_{start} \in \mathbb{R}$	*specifies the start angle of the angular range*
$\psi_{stop} \in \mathbb{R}$	*specifies the stop angle of the angular range*
$\Delta\psi \in \mathbb{R}$	*specifies the angular resolution*
$y_{start} \in \mathbb{R}$	*specifies the extent of the map in negative y direction*
$y_{stop} \in \mathbb{R}$	*specifies the extent of the map in positive y direction*
$\Delta y \in \mathbb{R}$	*specifies the width of the sectors*

4.4.3 Sector-based Structures as Virtual Sensors

Sector maps are particularly useful for the specification of abstract views on the environment in terms of *Virtual Sensors*. *Virtual* in that context means that perspective, range, and resolution of a sensor is mimicked as if it was mounted at a certain place. That way, (virtual) sensor characteristics can be defined to suit various particular applications without actually having a real sensor system which covers the region in question. These *Virtual Sensors* are defined relative to the robot and are usually fixedly mounted (see Section 8.2 in Part II for example configurations). For this purpose, the sector maps are usually configured in local *mode* and the origin of the map in question is static relative to the RCS. In collision avoidance for example, a protective belt of *Virtual Sensors* could be defined which approximates the robot shape with sector-based *Structures* (see Figure 5.1 on page 66).

However, some applications require the gathering of information about structures in the environment from a specific perspective which is not necessarily connected to the robot's current position. In that case, the *origin* of a *Virtual Sensor* can be manipulated such that a robot-perspective independent *View* on the environment is yielded. This special type of *Virtual Sensor* shall be called *Virtual Sensor Probe (VSP)*. VSP are usually defined in global *mode* as the VSP perspectives remain stable even if the robot moves. Otherwise, the VSP would also move along with the robot, which may be less intuitive depending on the application. As for ordinary *Virtual Sensors* both *modes* are supported nonetheless. Semantic translation and coordinate transformation (reference coordinate



(a) Real sensor **(b)** Virtual sensor probe

Figure 4.14: The views of a real sensor (a) and a virtual sensor probe (b).

system to SCS) are carried out by the *Semantic Abstraction* layer in a transparent fashion. Therefore, higher components do not have to deal with these technical details, but can simply provide poses in the reference coordinate system (WCS or RCS depending on the VSP configuration) for the VSP in question. The *Views* generated by the VSP will then be filled with the specified information as required.

Using VSP, the robot can thus "look" at places in the environment from a different point of view (see Figure 4.14). While this naturally does not yield new information that the robot cannot gather from its actual pose, it structures the existing sensor data in an abstract way, allowing for the development of algorithms in a more straightforward manner.

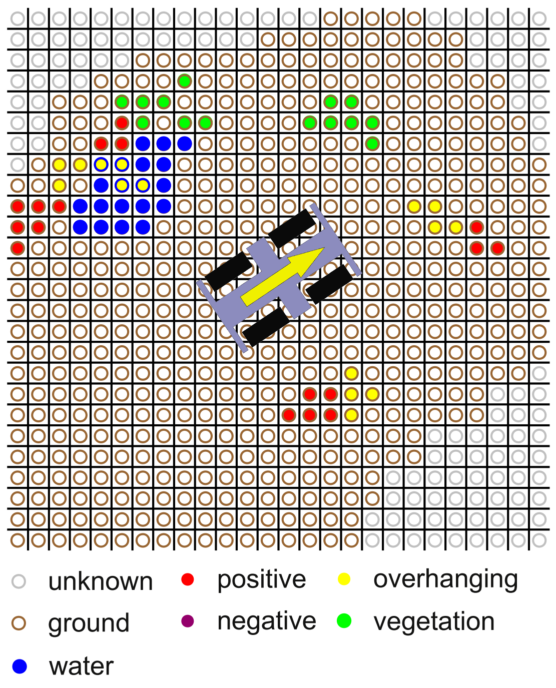
5. Discussion

In the theoretical part of this work, a novel design methodology for creating complex robotic control systems was introduced. Furthermore, an expressive representation scheme with adjoined basic functionality for translating semantics and fusing abstract information was proposed. To conclude the excursus on design schemata forming the theoretical design foundations of this work, the reader should take a step back and arrange the details into the context of the design methodology outlined in Section 3.3.1. The proposed methodology follows a combined action-/perception-oriented approach which strictly separates sensor processing design and control system design. *Sense-Control-Duality* (Design Idea 3) is explicitly supported via thorough abstractions which are an inherent part of the design approach. Besides improved reusability of components (in-house as well as third-party), this approach conceptually supports parallel design and development of robotic systems with large teams of developers (see Figure 3.8 on page 33). Components from both design directions are brought together by the configurable `Semantic Abstraction` layer. This mediating layer translates `Short-term Memories` which represent concrete environmental information on the sensor processing level into abstract `Views` containing tailored portions of information required on the control level (see Figure 3.9 on page 34).

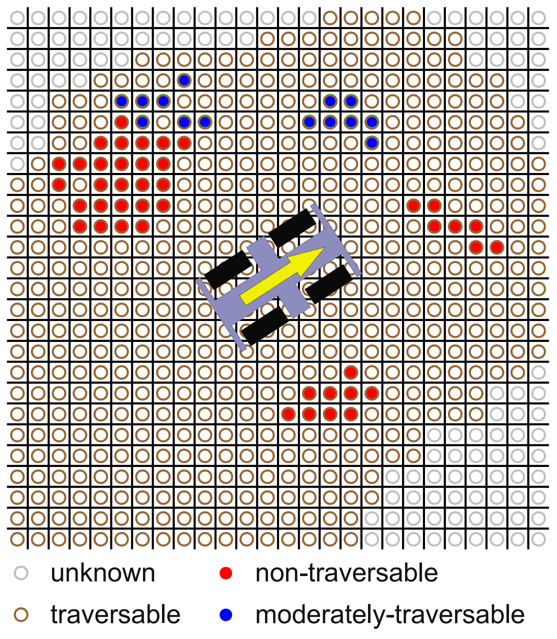
5.1 Short-term Memories versus Views

The representation scheme outlined above does not explicitly provide different structures or content elements for `Short-term Memories` and `Views`. The differentiation of both types of structures rather lies in the semantic complexity of the represented information. In general, `Short-term Memories` store more concrete information on entities of the real world (e. g. obstacle types, exact dimensions in 3D, direction of motion, velocity, and many more are thinkable), while `Views` hold abstract information which have been tailored to a specific task (e. g. slowing down, evading obstacles, or planning a path).

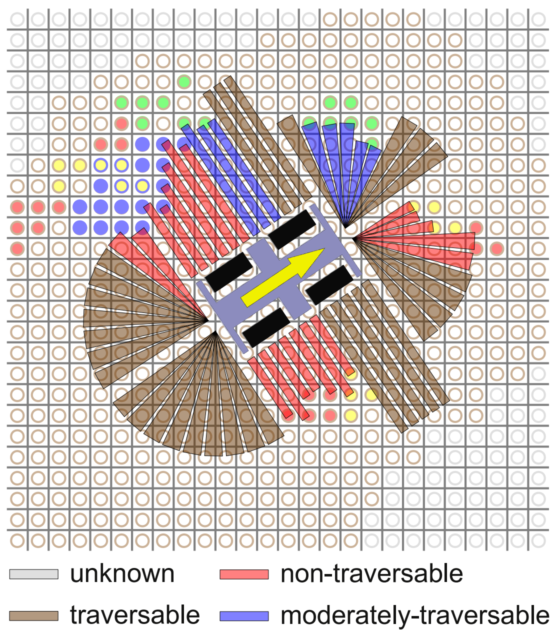
`Short-term Memories` thus aggregate a rich information basis which is cast into various compact `Views` to communicate effectively between different components. Figure 5.1 illustrates a `Short-term Memory` which classifies a robot's environment into detailed obstacle classes (a), a grid-based `View` holding traversability information (b) and a set of



(a) Grid-based Short-term Memory containing obstacle information.



(b) Grid-based View containing traversability information.



(c) Sector-based Views containing distances to non-traversable entities.

Figure 5.1: Short-term Memory (a) and generated Views (b) and (c).

sector-based `Views` monitoring the distance to obstacles (c). Note that the compactness of `Views` refers to the semantics (i. e. the number of properties represented and the additional annotations stored), not to terrain coverage or structure type. For planning activities, abstracted terrain information may be fused and further aggregated into larger `Views` to allow for reasoning on a more complete model of the environment.

5.2 A compact Reference Manual for Design

Thorough abstractions allow for the design of self-contained components with well-defined interfaces which can be shared among several platforms. The schemata outlined above provide the theoretical framework for the proposed action/perception-oriented design methodology. With the practical comments from the section above, this section summarises the design guidelines for the application of the schemata in the context of the proposed design methodology. The diversity of the mobile robotics domain makes the specification of stringent rules virtually impossible. Therefore, the following guidelines attempt to provide a way of thinking about a problem rather than a concrete step-by-step recipe. This overview can be regarded as a reference manual according to which the application study will be carried out in Part II of this work.

5.2.1 General Guidelines for Representation Design

The employment of representation is a central idea of the proposed design methodology. In order to unfold the full expressiveness of the design schemata, some general rules should be followed.

Structure / Content / Handler Reuse (Design Guideline 1) *Prefer existing base structures, contents, and handlers over creating new ones. This comprises the reuse of complete* Trait Combinations *as well as the partial reuse by combining* extension suites[1] *in a different way.*

This guideline follows the general idea to ***Reuse components*** *([Proetzsch 10a] p. 104) on a fine-grained scale which is supported by defining extension suite which adhere to* Extension Independence *(Guideline 14) and guideline* Minimalistic Extensions *(Guideline 13).*

Representation Deployment (Design Guideline 2) *Embrace representation to compensate for the sensors'* Limited Fields of Vision *(Challenge 3 on page 22).*

If information may be valuable over a certain period of time, standardised representations should be designed to retain the data in a way that is suitable for subsequent deployment.

Representation Locality (Design Guideline 3) *Use the representation scheme to design terrain representations of limited scope in the style of* Short-term Memories *(Design Idea 6) in natural mobile systems.*

Collision avoidance for example requires an accurate representation of structures in the proximity of the robot while path planning needs information on a larger scale but probably with less details.

[1] An *extension suite* is a content element extension with accompanying handler extensions for manipulating the data encapsulated by the content element extension in question.

Representational Separation (Design Guideline 4) *Avoid the combination of information which is not required in a condensed form.*

This guideline aims at the design of distributed world models which consist of tailored representations for individual function units. The idea is to decrease the vulnerability of the overall system to world model corruption by increasing the integrity of individual components through stringent abstractions. Different algorithms for example should store information in separate representations to minimise Configuration Dependence *(Challenge 2 on page 21) and to avoid bottle necks during data access.*

Algorithmic Separation (Design Guideline 5) *Similar to sensor centres in natural mobile systems (see* Perceptional Separation *(Design Idea 1)), evaluation algorithms should be separated to rely on a minimal suite of sensor data.*

Separate sensor processing algorithms strictly to minimise Configuration Dependence *(Challenge 2 on page 21) concerning the sensor suite deployed on a particular robot to support fine-grained reuse of components.*

Deferred Fusion (Design Guideline 6) *Fuse information from different sources as late as possible and as early as necessary. This guideline prefers control-level fusion over representation-level fusion (abstract fusion) over sensor-level fusion (fusion of raw sensor data).*

Note that this guideline represents a generic extension of [Rosenblatt 97a][2] *with an additional fusion level. Deferred Fusion supports component reuse on the sensor processing side and the control system side via thorough abstractions. Besides* Configuration Dependence *(Challenge 2 on page 21), Deferred Fusion also addresses the challenge of* Data Integration *(Challenge 1 on page 21). Condensing data on the most abstract level possible tends to require less accuracy in data registration and robot localisation.*

Competence Overlap (Design Guideline 7) *Design levels of competences to complement and overlap one another to handle the* Control Handover Problem *related to* Action Selection *in a graceful manner.*

This guideline represents a more stringent variant of guideline ***Prefer redundancy*** *proposed in [Proetzsch 10a] (see p. 104).*

5.2.2 Guidelines for Structure Design

The strict separation of structure and content is a core concept of the proposed representation scheme. Structures hold information about the locality of stored entities and should be standardised in order to benefit from generic base handlers. For the integration of supplementary base structures into the representation scheme, the following guidelines should be followed.

[2]Rosenblatt prefers command arbitration over sensor fusion.

Minimal Set of Base Structures (Design Guideline 8) *Prefer a minimal set of base structures. This guideline is related to* Structure Reuse *(Guideline 1).*

Structural Partitioning (Design Guideline 9) *Use simple structures to approximate more complex forms in a natural way.*

This guideline adds at maintaining a Minimal Set of Base Structures *(Guideline 8).*

Structure Simplicity (Design Guideline 10) *Prefer the deployment of simple tailored structures over the distribution of information bases which are more powerful than required to support* Handler Simplicity *(Guideline 15).*

That way, the prerequisites of functional units can be satisfied more easily when migrating to a different platform.

5.2.3 Guidelines for Content and Handler Design

The suitability of representations is strongly dependent on the deployment context and its semantics. Therefore, the proposed set of schemata provides mechanisms for fine-grained specification of semantics and the translation between different levels of abstraction. In order to exploit the flexibility of the schemata, contents and handlers should be designed according to the following guidelines.

Natural Naming (Design Guideline 11) *Find natural names for properties to be represented in the context of the purpose at hand to support traceability.*

Property Selectivity (Design Guideline 12) *Define properties selectively in order to separate distinct concerns precisely. Avoid premature condensation of different aspects into one property.*

Minimalistic Extensions (Design Guideline 13) *Define the scope of content element and handler extensions as minimal as possible to support fine-grained reuse.*

Extension Independence (Design Guideline 14) *Define* extension suites *independent from one another to support individual reuse.*

Handler Simplicity (Design Guideline 15) *Prefer simple generic functional units which operate on a minimal set of input structures over specialised monolithic components. This guideline represents an extension of statement* Favour Simple Behaviours *(see [Proetzsch 10a] p. 92) towards representation.*

Functional Partitioning (Design Guideline 16) *Try to split functional units if the set of deployed structures grows large to improve* Handler Simplicity *(Guideline 15).*

Handler Interaction (Design Guideline 17) *Use generic handler interaction to support* Functional Partitioning *(Guideline 16) avoid duplicate computation of partial results.*

Prefer interaction of generic functional units operating on simple structures over monolithic solutions operating on complex structures.

Note that generic handler interaction requires support by the deployed architecture. Generally speaking reactive and behaviour-based approaches are more suitable for Handler Interaction *than deliberative control architectures which promote more complex monolithic components.*

This compact reference of guidelines concludes the theoretical part of this work. In the second part of this Doctoral Thesis, a large-scale experiment in terms of an exemplary design of a complex robot navigation system will be presented. This application study summarises the development efforts of several years on the off-road platform RAVON and aims at providing a practical view on the theory presented in this work. A copy of this section may serve as a handy lookup for reading Part II.

Part II

Application Study – Design of a Modular Navigation System for RAVON

6. Overview

In Part I a design methodology, as well as underlying schemata for representation, abstraction, and fusion of information have been introduced. Furthermore, accompanying design principles and guidelines have been motivated and exemplified in tiny show cases. At this point a concrete design for a complex robotic system shall be evolved step-by-step in order to demonstrate the applicability of the abstract concepts and design means outlined above.

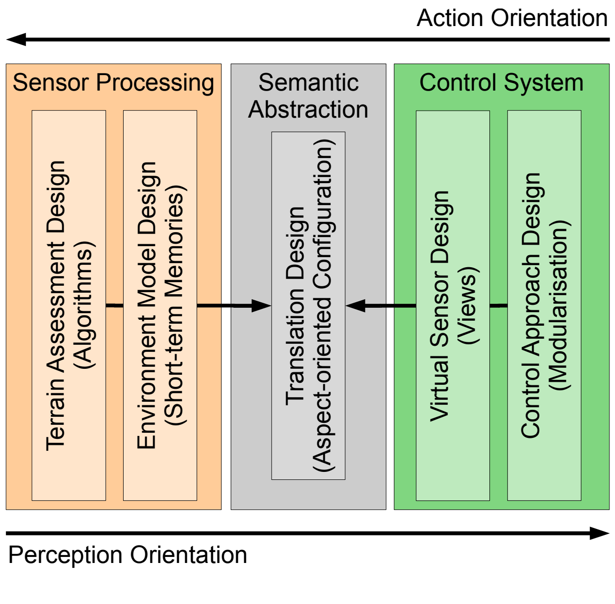

Action Orientation
Sensor Processing
Terrain Assessment Design (Algorithms)
Environment Model Design (Short-term Memories)
Semantic Abstraction
Translation Design (Aspect-oriented Configuration)
Control System
Virtual Sensor Design (Views)
Control Approach Design (Modularisation)
Perception Orientation

Figure 6.1: Overview of the proposed design approach.

The proposed action-/perception-oriented design methodology features the possibility to work on several components in parallel and to integrate functionality in an incremental fashion (remember Section 3.3.2). `Control System` and `Sensor Processing` can be treated as independent design tasks which both ask for different design competences and approaches. `Semantic Abstraction` generates control-level *Views* from sensor-level *Short-term Memories* on the basis of an *Aspect-oriented Configuration*. This configuration requires the structural specifications of *Short-term Memories* and *Views* together with the accordant semantics in terms of property sets. Owing the nature of printed media, a suitable linearisation of the particular design activities has to be found in order to describe the process step-by-step.

The author decided to start with the action-oriented design of the `Control System` in Chapter 8. This design step yields a modularisation of the `Control System` consisting of functional units which require particular *Views* on the environment to fulfil their tasks. The structural specifications in terms of virtual sensors specifying terrain coverage (see Section 4.4 on page 59) and the destination property sets specifying the semantics of the control aspect to be reflected in the *Views* (see Section 4.1 on page 37), are the control level input to the *Aspect-oriented Configuration* of the `Semantic Abstraction`.

Subsequent to the control system design, `Terrain Assessment` strategies for the various sensor systems installed on RAVON will be introduced in Chapter 9. At this point, the capabilities of representation schemata shall be highlighted by developing several *Short-term Memories* of varying complexity for the particular sensor evaluation mechanisms. This chapter will present algorithms which range from the processing of simple bumper events towards detailed 3D scene analysis and the suitable representation of the extracted information. The structural specifications and the semantics of the *Short-term Memories* in terms of the source property sets are the sensor-level input for the aspect-oriented configuration of the `Semantic Abstraction`. Finally, the mappings of source property sets to destination property sets are specified in terms of the *Aspect-oriented Configuration* which will be discussed in Chapter 10. At the beginning of each chapter, the progress of the design procedure will be highlighted as an overlay to Figure 6.1.

Before going into detail with design, the experimental off-road platform RAVON and the architectural foundations subject to this application study shall be introduced in the next chapter.

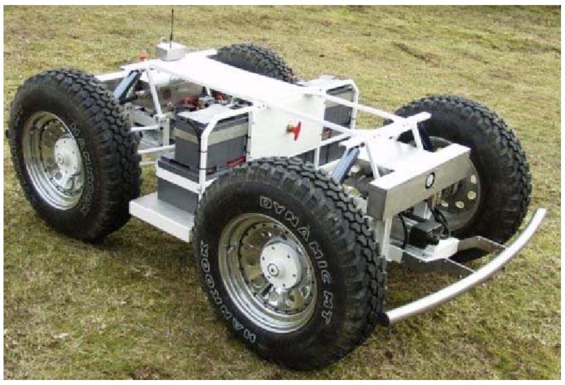
7. The Off-road Robot RAVON

The robotic platform RAVON serves as a testbed for the investigation of behaviour-based strategies on motion adaptation, localisation, and navigation in rough vegetated off-road terrain. The name is an acronym which is related to these research goals and stands for *R*obust *A*utnomous *V*ehicle for *O*ff-road *N*avigation. For an overview of the development platform over the years, the reader shall be referred to [Armbrust 10a, Armbrust 09a, Schäfer 06].

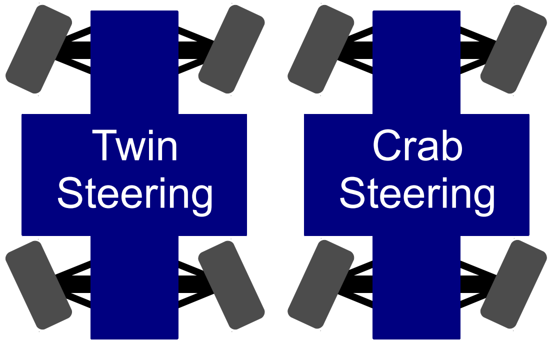
RAVON is based on the commercial *RobuCar TT*[1] platform by the french robot maker RoboSoft[2]. The base system comes as a naked chassis as illustrated in Figure 7.1a. The four-wheel-drive features four separate motors for traction control. Front and rear axis can be controlled independently allowing for advanced manoeuvres like twin steering for tight radii or crab steering for sideward movements (see Figure 7.1b). Factory capabilities are

[1]RobuCar TT → The *TT* in the 2004 version is for french *Tout Terrain*. Since 2005 the series was renamed *RobuCar AT* (All Terrain).

[2]RoboSoft → http://www.robosoft.com

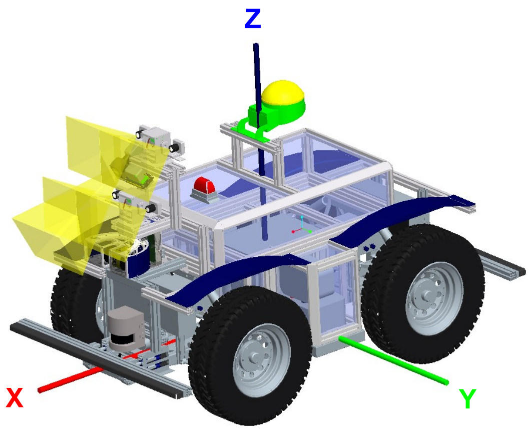


(a) The basic 2004 RobuCar TT platform.

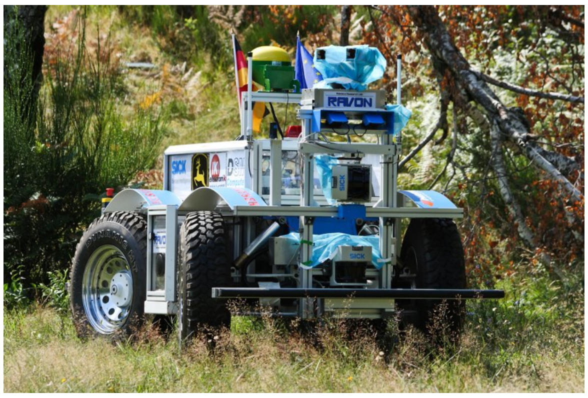


(b) Advanced manoeuvrability.

Figure 7.1: The RobuCar TT platform by RoboSoft.



(a) CAD model of RAVON.



(b) RAVON in harsh terrain.

Figure 7.2: The off-road platform RAVON.

limited to manual driving using the wire-connected safety control stick. Control computers can be connected to the robot using the serial connection of the integrated electronics.

In its base configuration the RobuCar TT is not yet suitable for research in off-road robotics. The harsh environmental conditions call for a protective hull to minimise the risk of damage to the electronics. The chassis was cased in a robust and extensible cover based on the Minitec profile system which renders the integration and modification of the sensor setup very convenient. Figure 7.2a shows the CAD[3] model of the robot with the protective casing[4].

In order to augment the robot's climbing capabilities, the 1 kW motors delivered with the system were replaced by 2 kW motors designed and built by Hübner Giessen[5]. In that context, the base electronics was also abandoned in favour of the DSP[6] board line developed at the ROBOTICS RESEARCH LAB [Hillenbrand 09]. Furthermore, a new set of SpiralCell batteries by YellowTop was installed to assure the operation time of about four hours at the extended energy requirements. The final RAVON robot has the dimensions of a city car, weighs about 750 kg and can ascend and descend slopes of 45° at a maximal velocity of 7 km/h. Technical data is summarised in Table 7.1.

The off-road platform RAVON features a variety of sensor systems for obstacle detection. Figure 7.3 shows the front sensor phalanx of the robot. Horizontally mounted 2D LRF[7] monitor close range safety zones at the front and rear side of the robot. Two stereo camera systems are mounted for mid-range [Schäfer 07a, Schäfer 05a, Schäfer 05c] (up to 20 m at coarse resolution) obstacle detection and far-range terrain classification (up to 40 m). To complement for the weaknesses of camera-based systems, an additional 3D laser scanner is mounted at the front which provides terrain information to a distance of about 15 m. Furthermore, a spring-mounted bumper system is used to detect hindrances on a tactile basis to support navigation through high vegetation where visual approaches fail.

[3]Computer Aided Design (CAD)
[4]For the mechanical extension, the CAD model of the base construct was kindly provided by RoboSoft.
[5]The motors and the costs for the integration were sponsored by Hübner Giessen.
[6]Digital Signal Processor (DSP)
[7]2D Laser Range Finder (2D LRF)

Length	2.35 m
Width	1.4 m
Height	1.7 m (highest spot: StarFire ITC GPS aerial)
Weight	750 kg
Ground Clearance	0.3 m
Power Supply	8 Spiral Cell batteries (12 V, 55 Ah each)
Operation Time	about 4 hours
Drive	4 WD with independent electric motors (4 × 1.9 kW)
Velocity	10 km/h max.
Max. Slope	45° (at 7 km/h)
Controllers	4 Unitek BAMOBIL-D3 for separate motor control 6 Motorola 56F803 DSPs • 2 for separate steering of front and rear axles • 2 for camera pan-til-units, 1 for panning LRF • 1 for inertial navigation system
Computers	3 On-board Industrial PCs

Table 7.1: Technical specification of RAVON.

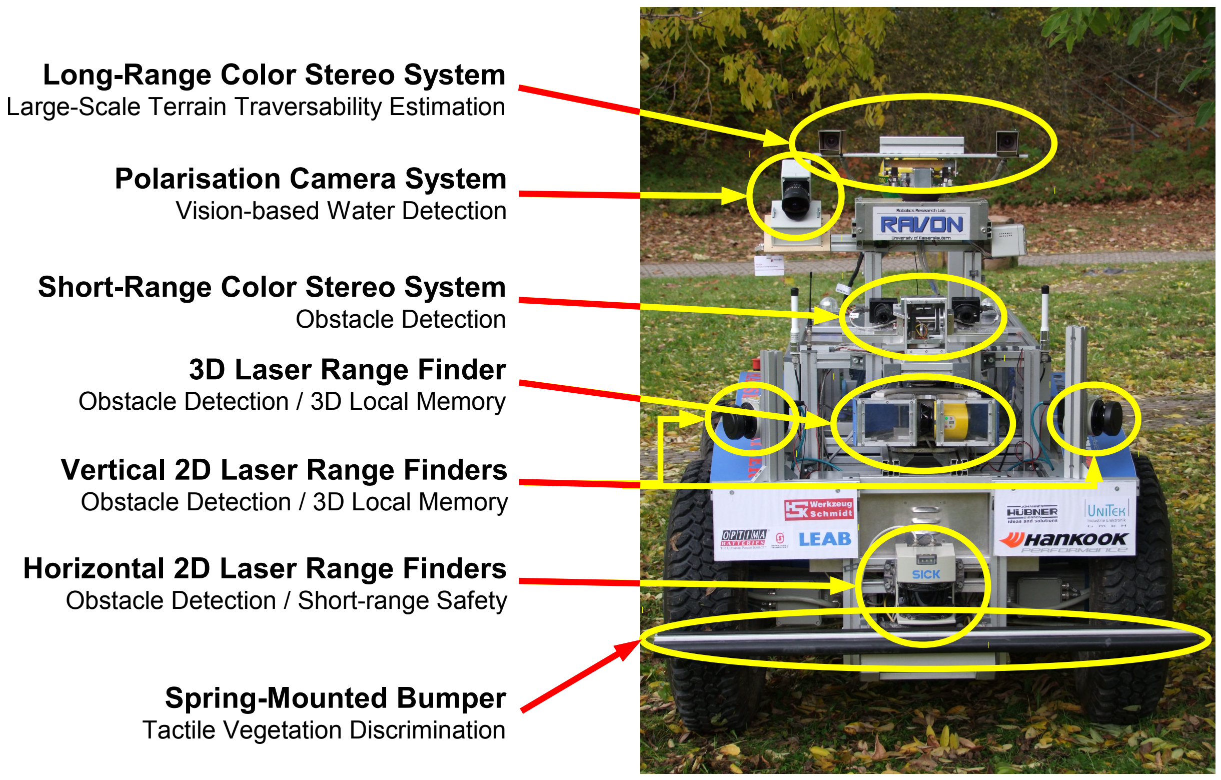

Long-Range Color Stereo System
Large-Scale Terrain Traversability Estimation
Polarisation Camera System
Vision-based Water Detection
Short-Range Color Stereo System
Obstacle Detection
3D Laser Range Finder
Obstacle Detection / 3D Local Memory
Vertical 2D Laser Range Finders
Obstacle Detection / 3D Local Memory
Horizontal 2D Laser Range Finders
Obstacle Detection / Short-range Safety
Spring-Mounted Bumper
Tactile Vegetation Discrimination
RAVON
SICK

Figure 7.3: Sensor systems mounted on RAVON.

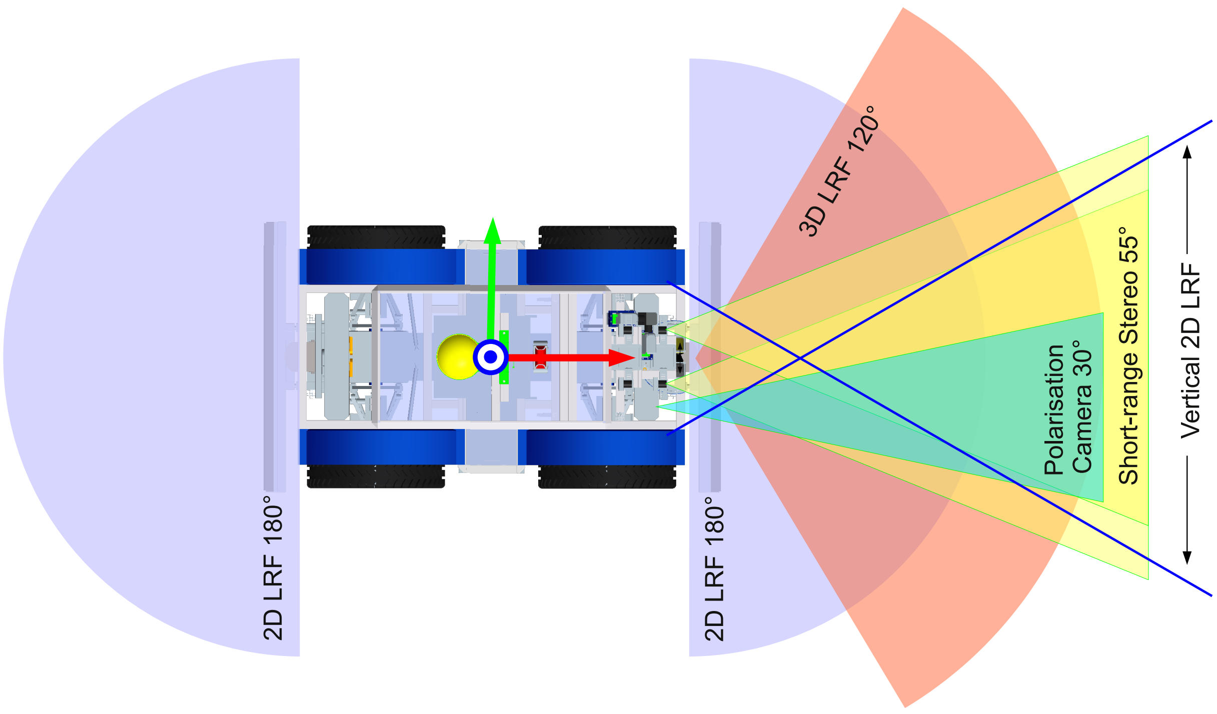


Figure 7.4: Regions monitored by the particular sensors mounted on RAVON.

In order to give the reader an idea of what areas are covered by what sensor system, Figure 7.4 presents a top view of the vehicle with the angle of vision for each visual sensor. Details on the sensor hardware used and the actuation mechanics developed in the context of this work can be found in Appendix A.1 and A.2.

For off-road navigation, localisation and orientation estimation are crucial factors. In order to assess its current pose, RAVON disposes of odometers on all four wheels and both steering motors. This data is fused with the three axis orientation information from a custom-built Inertial Measurement Unit (IMU) [Koch 05] with integrated digital compass [Renner 06] and two GPS[8] devices using a Kalman filter [Schmitz 05]. This *globally stable pose*[9] is used for far-range navigation. For the aggregation of short-term memories – which is robust against global drift but requires local pose coherence – a *locally stable pose*[10] is computed from odometry, the IMU, and the compass. For high-precision navigation under clear sky a StarFire ITC with the GreenStar correction signal yielding spade-width localisation precision is used. In the deep forest this device usually is not sensitive enough to yield position information. Therefore, a consumer-grade U-blox GPS mouse is used as backup.

[8]Global Positioning System (GPS)

[9]A *globally stable pose* shall be defined here as pose information that may have local leaps in position but does not drift on a global scale.

[10]The *locally stable pose* does not have local leaps in position but is bound to drift on a global scale.

8. Action-oriented Control Design

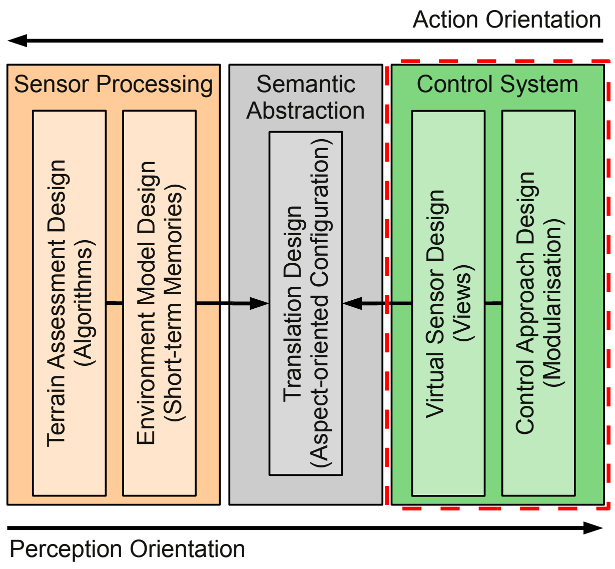

Action Orientation
Sensor Processing
Terrain Assessment Design (Algorithms)
Environment Model Design (Short-term Memories)
Semantic Abstraction
Translation Design (Aspect-oriented Configuration)
Control System
Virtual Sensor Design (Views)
Control Approach Design (Modularisation)
Perception Orientation

Figure 8.1: The first step in the design process is the action-oriented design of RAVON's control system.

Figure 8.1 illustrates the progress of the design carried out in the context of this application study. The first step in the design approach conducted in this application study is the action-oriented design of the `Control System`. This design step aims at breaking the robot's control system into a hierarchy of straightforward subunits to manage system complexity (divide and conquer). The resulting modularisation and the specification of inputs and outputs declare the information bases required for accomplishing particular subtasks. These declarations are the control-level input for the *Aspect-oriented Configuration* of the `Semantic Abstraction`.

RAVON's control system is organised in multiple levels of competences. In the style of three layer architectures, these competences have coarsely been assigned to three logical components which shall be named `Navigator`, `Mediator`, and `Pilot`. As already alluded

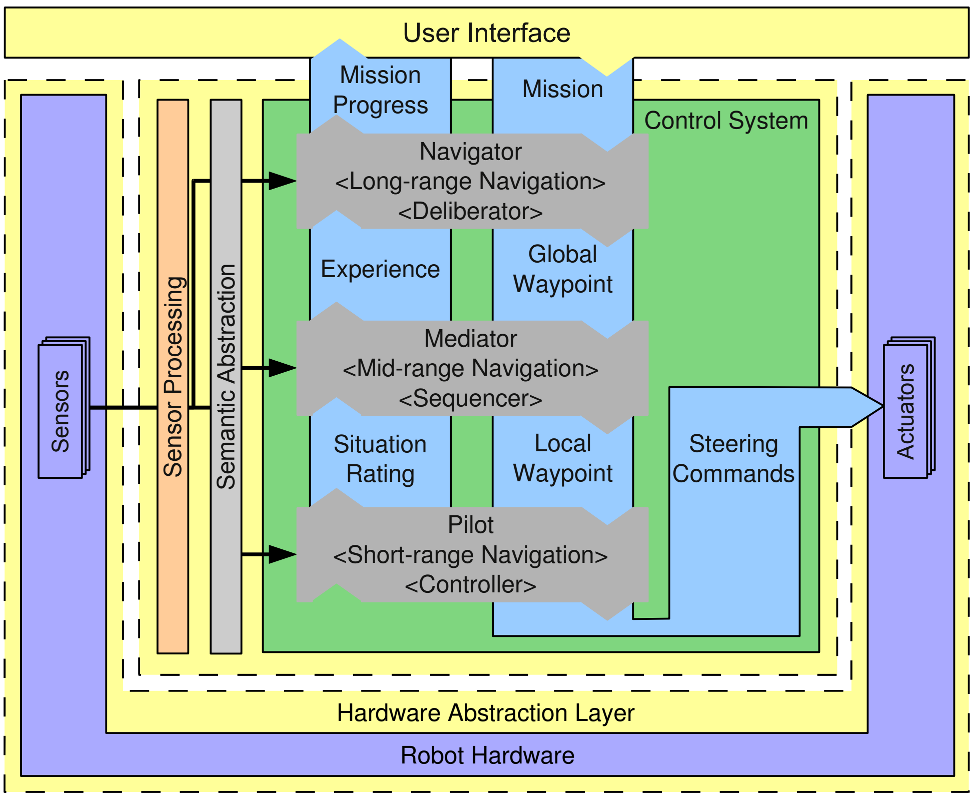


Figure 8.2: Overview over the control system design.

above, the major goal at this stage of the design is the assignment of tasks to the particular components and the definition of slim yet powerful interfaces between them. On the one hand, tasks should be grouped such that one component can be regarded as a self-contained entity with well-defined interfaces. On the other hand, synergies between the layers shall not be hindered by artificial barriers built up for non-technical reasons. As already mentioned above, the interaction between the capabilities on different layers is the key for robust control handover in critical situations. The qualitatively different aims pursued by each component and the level of navigational granularity operated on, as well as the interaction between the components shall be highlighted in the following. Figure 8.2 gives a refined view onto the overall control concept.

As the host of the highest level of navigational competence, the `Navigator` is responsible for control on the mission level. Missions in that context might for example be the navigation to a target location or the request for exploring an area given in GPS coordinates. The `Navigator` is thus responsible for the far-range navigation capabilities of the robot. For RAVON this comprises missions of several kilometres travelling distance. The decisions of the `Navigator` rely on topological map material that has either been introduced manually by the operator or extracted from GIS[1] data. On the basis of this data mission goals are broken down into coarse (in the range of the next hundred metres) subgoals and passed to the `Mediator` layer in terms of subsequent global waypoints which are to be reached. Note that the pool of information the far-range navigation capabilities rely on

[1]Geographic Information System (GIS)

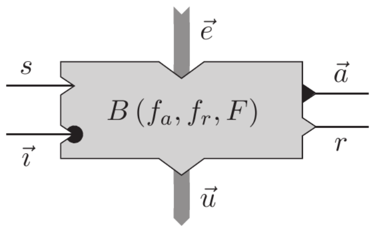
is potentially inaccurate and/or outdated. Therefore, global waypoints may be beyond reach for the robot. If mission progress is unsatisfactory, the `Navigator` may abandon waypoints in favour of an alternative route. For this purpose, the `Navigator` may have to extend the topological map provided by the operator on the basis of sensor information and experiences made on the lower layers. Experience encapsulates abstract information on how difficult the terrain at hand was actually negotiable. On the basis of this data, future operator commands can be cast into more efficient routes. The `Navigator` is thus the central authority to assure mission progress and goal achievement on a global scale [Braun 09a].

The `Mediator` is concerned with negotiating mid-range navigation situations in order to reach the *next* global waypoint issued by the *navigator*. The aim of the `Mediator` is to keep the vehicle out of trouble by developing a refined plan for the submission in a foresighted fashion. Decisions on the mid-range navigation capabilities are based on terrain information gathered from the on-board sensors. Global waypoints provided by the `Navigator` are broken down into sequences of local waypoints which are sent to the next lower layer.

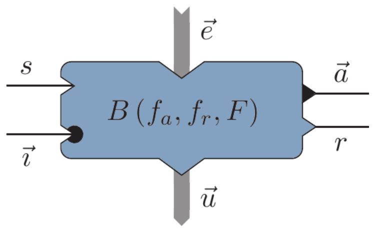
The lowest levels of competences reside in the `Pilot` which takes responsibility for short-range navigation. This component deals with acute threats to the robot or its environment in the dimension of several metres ahead. First of all, the local waypoints provided by the `Mediator` are cast into steering commands that draw the robot towards the waypoint in question. These steering commands are further moderated in tight sensor-actor loops to assure vehicle safety at all times. If the local waypoint from the `Mediator` conflicts with vehicle safety such that the `Pilot` cannot take responsibility for execution, control is handed back to the `Mediator` which carries out ranking manoeuvres on a local scale to bring the robot into an operational state again. Furthermore, situation ratings yielded from direct interaction with obstacles and terrain structures are passed from the `Pilot` to the `Mediator` in order to prevent oscillations.

With this coarse overview in mind, the following sections will illuminate the desired competences of the three components in greater detail. Through further partitioning the design of the particular components will gradually be refined in a top-down fashion. For the work at hand, the key issue in control design is to derive data requirements for control components in order to obtain the configuration of the *Semantic Abstraction Layer* from the control side. The concrete implementation of the various *Behaviours* is not important at this point. A detailed discussion of transfer functions or interactions between the *Behaviours* is therefore beyond the scope of this work. Nonetheless, the author is of the opinion that a certain understanding of the control strategy will focus the reader on how the proposed design approach allows to bridge semantic gaps in robot control. The following sections will thus only provide the conceptual ideas about control approach followed on the different levels of competence. For a subset of *Behaviours*, the required environmental information will be specified in terms of appropriate *Virtual Sensor* specifications. These *Virtual Sensor* definitions represent the control side input for the *Aspect-oriented Configuration* of the *Semantic Abstraction Layer*. Before going into detail with the control design, the foundations of the behaviour-based architecture iB2C which was used in the context of this application study shall be illuminated.

8.1 Architectural Foundations – The Integrated Behaviour-based Control iB2C



(a) Behaviour Module



(b) Behaviour Fusion Module

Figure 8.3: Behaviour-based Modules in iB2C.

In this section, a brief introduction to the architectural foundations relevant for the application study carried out in this work shall be given. The navigation software deployed on the off-road platform RAVON is based on the behaviour-based architecture iB2C, which is developed at the ROBOTICS RESEARCH LAB. iB2C provides a modularisation scheme together with design principles and best practices for the design of complex control systems. The reference implementation comes with numerous software building blocks and tools for the design and analysis of robot software. The discussion of the full expressiveness of iB2C is therefore beyond the scope of this work. For more details on iB2C, the reader shall be referred to [Proetzsch 10b].

The fundamental building block in iB2C is the *behaviour-based module* which defines a well-defined control flow interface as depicted in Figure 8.3a. To distinguish *behaviour-based modules* from general system behaviour, *behaviour-based modules* shall be referred to as *Behaviours* in the rest of this work.

Stimulation s, Inhibition i, and Activation ι (Definition 13) *Each Behaviour B can be stimulated externally via the stimulation input $s \in [0,1]$ representing the intended relevance of B ($s = 1$ indicates full stimulation). The inverse effect can be achieved via the inhibition inputs $\vec{\imath} \in [0,1]^p$ (with $p = |\vec{\imath}|$, the number of values in vector $\vec{\imath}$). The total inhibition $i \in [0,1]$ where $i = \max_{j=1,\dots,p}(i_j)$ of a Behaviour B reduces the relevance of B ($i = 1$ indicates full inhibition). The combination of stimulation and inhibition is the activation $\iota = s \cdot (1 - i)$.*

Each *Behaviour* generates two classes of *Behaviour signals* that allow for deducing information about its state and its assessment of the current situation.

Activity $\vec{a}$ (Definition 14) *Activity $\vec{a}$ represents the amount of influence a particular Behaviour wants to exert in the Behaviour network. For $a_i \in \vec{a}$, $a_i = 1$ indicates full activity and $a_i = 0$ indicates inactivity.*

The activity vector $\vec{a} = (a, \underline{\vec{a}})^T \in [0,1]^{|\vec{a}|}$ consists of the main activity a of Behaviour B and – where applicable – the a vector of derived activities $\underline{\vec{a}}$. The derived activities allow a Behaviour to transfer only a part of its activity to other Behaviours for more selective interactions. Therefore, it is essential that $a \geq \underline{a}_i$ holds $\forall \underline{a}_i \in \underline{\vec{a}}$.

Target Rating r (Definition 15) *The target rating $r \in [0,1]$ serves as an indicator for the contentment of the Behaviour. $r = 1$ indicates full dissatisfaction, while $r = 0$ indicates satisfaction with the current situation. In case of full satisfaction, a particular Behaviour has reached its goal and usually ceases its influence in the Behaviour network. As a rule of thumb $r \geq a$ holds in most Behaviours*[2].

On the basis of these definitions, a *Behaviour B* can formally be specified as a triple of functions $B = (f_a, f_r, F)$ where f_a is the *activity function*, f_r the *target rating function*, and F the *transfer function* of B.

$$\begin{aligned} f_a(\vec{e}, \iota) &= \vec{a} \\ f_r(\vec{e}) &= r \\ F(\vec{e}, \iota) &= \vec{u} \end{aligned}$$

Note that the activity vector $\vec{a}$ is moderated by the activation ι while the target rating r is not. A *Behaviour* may thus be discontent with the current situation ($r > 0$) but not capable of reacting on this discontentment because it is not stimulated ($s = 0$) or externally inhibited ($i > 0$) by other *Behaviours*. That way, the impact of particular *Behaviours* may be manipulated by external factors, allowing for *Behaviour* interaction which yields an emerging system behaviour.

The intelligence of a *Behaviour* is provided by its transfer function F which determines the output vector $\vec{u}$ using the input vector $\vec{e}$. Transfer functions may range from simple stateless mappings between value ranges towards arbitrarily complex stateful computations. The activity function f_a and the target rating function f_r of a *Behaviour* are typically simple differentiable functions, like sigmoids. Sigmoid functions are often deployed as these have favourable properties for behaviour-based fusion and robot control.

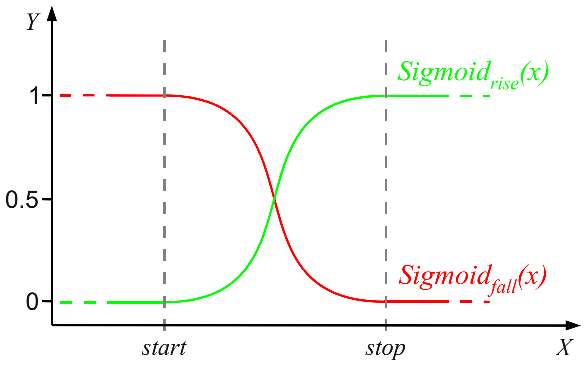
Sigmoid Function (Definition 16) *Sigmoids are differentiable functions sigmoid(x) which feature a non-negative or a non-positive first derivative and have exactly one inflection point. Sigmoids are furthermore horizontally asymptotic for $x \to \pm\infty$.*

The iB2C reference implementation for instance provides two section-wise defined sigmoid approximations $\mathit{sigmoid}_{\mathit{flank}} : \mathbb{R} \to [0,1]$, $\mathit{flank} \in \{\mathit{rise}, \mathit{fall}\}$ which realise continuous normalised flanks over the input value range $[\mathit{start}, \mathit{stop}]$ (see also Figure 8.4a):

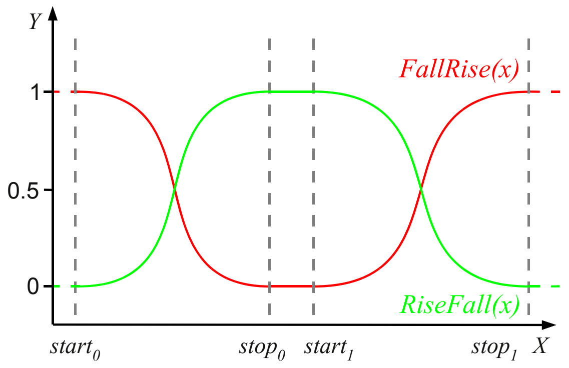
$$\mathit{sigmoid}_{\mathit{rise}}(x) = \begin{cases} 0 & x \in [-\infty, \mathit{start}] \\ \frac{1}{2} + \frac{1}{2} \cdot \mathit{sin}(\pi \cdot (\frac{x - \mathit{start}}{\mathit{stop} - \mathit{start}} - \frac{1}{2})) & x \in [\mathit{start}, \mathit{stop}] \\ 1 & x \in [\mathit{stop}, \infty] \end{cases}$$

$$\mathit{sigmoid}_{\mathit{fall}}(x) = \begin{cases} 1 & x \in [-\infty, \mathit{start}] \\ \frac{1}{2} - \frac{1}{2} \cdot \mathit{sin}(\pi \cdot (\frac{x - \mathit{start}}{\mathit{stop} - \mathit{start}} - \frac{1}{2})) & x \in [\mathit{start}, \mathit{stop}] \\ 0 & x \in [\mathit{stop}, \infty] \end{cases}$$

[2]Exceptions are *Behaviours* which execute low-level control tasks which require activity to maintain the goal state (e. g. a position controller working against a disturber).



(a) Basic functions for building up *Behaviour* functions in iB2C.



(b) Combined example *Behaviour* functions *RiseFall*(x) and *FallRise*(x) with two flanks.

Figure 8.4: Common transfer functions in iB2C.

Sigmoids feature a smooth gradual change between the extreme values of the function, which is favourable for many control tasks. Furthermore, the piece-wise approximation proposed by the reference implementation has the advantage that the input range for minimum and maximum can be specified precisely. For asymptotic functions this is not possible, as the function value never really reaches the limits. More complex *Behaviour* functions f_a, f_r, and F can be built up using these basic functions as illustrated in Figure 8.4b.

Behaviour-based Fusion

Competing or collaborating *Behaviours* are coordinated by standardised fusion *Behaviours* (Figure 8.3b on page 82) which provide the very same interface as basic *Behaviour* modules. iB2C organises levels of competence in layers of *Behaviours* which are linked by layers of fusion *Behaviours* (see Figure 8.8b on page 90 for an example of the general arrangement). The input vector $\vec{e}_f$ of fusion nodes is composed of the activities a_c, the target ratings r_c, and the output vectors $\vec{u}_c$ of p competing or collaborating *Behaviours* B_c $(c \in [1, p])$:

$$\vec{e}_f = (a_1, r_1, \vec{u}_1, \ldots, a_p, r_p, \vec{u}_p)^T$$

The transfer function F is the fusion function (weighted average or maximum[3] according to the activities a_c) processing the output vectors $\vec{u}_c$ of the particular *Behaviours* to be fused value-wise to a fused output vector. The maximum fusion node $Fusion_{max} = (f_a, f_r, F)$ is defined as:

$$\begin{aligned} f_a(\vec{e}_f, \iota) &= a_s \\ f_r(\vec{e}_f) &= r_s \\ F(\vec{e}_f, \iota) &= \vec{u}_s \qquad \text{where} \qquad s = \underset{c}{\operatorname{argmax}}(a_c) \end{aligned}$$

[3]The third standard fusion mechanism computes the weighted sum of the input values. As it is not of relevance for this work, there will be no further details on this.

In analogy, the weighted average fusion $Fusion_{avg} = (f_a, f_r, F)$ is defined in a straightforward fashion:

$$f_a(\vec{e}_f, \iota) = \frac{\sum_{c=1}^{p} a_c^2}{\sum_{c=1}^{p} a_c} \cdot \iota \qquad f_r(\vec{e}_f) = \frac{\sum_{c=1}^{p} a_c \cdot r_c}{\sum_{c=1}^{p} a_c} \qquad F(\vec{e}_f, \iota) = \frac{\sum_{c=1}^{p} a_c \cdot \vec{u}_c}{\sum_{c=1}^{p} a_c}$$

This way, a subtle gradation of coordinating *Behaviour* control outputs regarding their activity is achieved. A structural example of *Behaviour* fusion with three *Behaviours* collaborating on slowing down the robot in the proximity of obstacles is depicted in Figure 8.5. Each of the three *Slow Down Behaviours* – (a) through (c) – monitors a different region of the terrain around the robot. In order to determine the combined deceleration, the output from the three *Behaviours* is fused using a generic fusion *Behaviour* (d) in maximum fusion mode.

Behaviour-based Groups

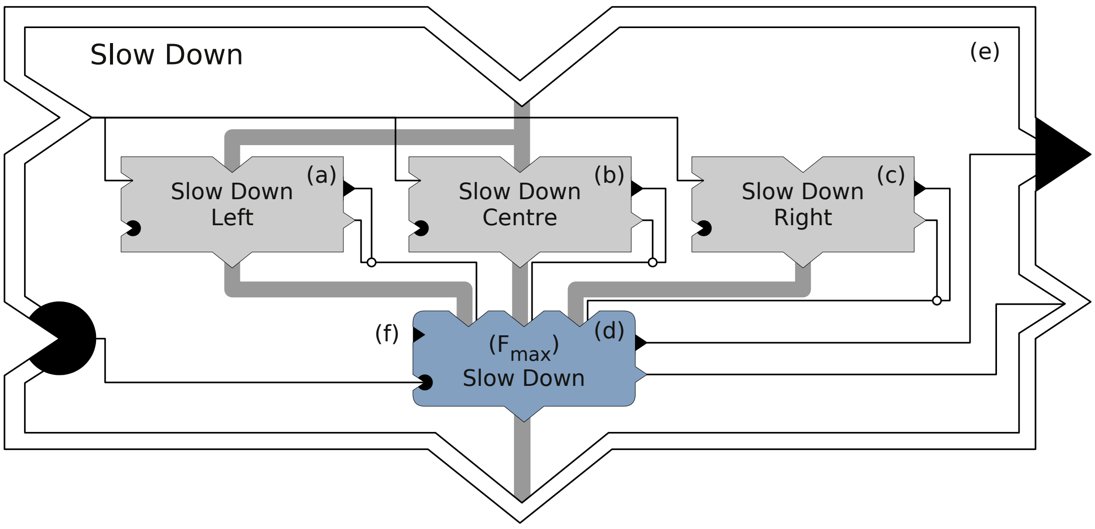


Figure 8.5: Behaviour-based Groups may be used to encapsulate related *Behaviours*.

The power of behaviour-based systems lies in the interaction of many simple units which yield an emerging overall system behaviour. In particular, the capability to merge the commands of multiple independent units pursuant different strategies to achieve potentially conflicting goals into one set of control commands makes behaviour-based systems inherently reusable and extensible. On the downside, behaviour-based networks grow large easily, such that structural mechanisms are a crucial concept for the applicability to complex systems. In iB2C, hierarchical structures can be built up by grouping related *Behaviours* into groups which are themselves *Behaviours* again. These behaviour-based groups may consist of an arbitrarily complex *Behaviour* subnetwork, which is preferably loosely coupled to the remaining behaviour network. The three collaborating *Slow Down Behaviours* in the example above are predestined to form a self-contained behaviour-based group (see Figure 8.5 (e)).

The interface of the behaviour-based group is fulfilled by the fusion node, which exports its activity and target rating. It is common practice to have the fusion *Behaviour* fully stimulated at all times. That is, the stimulation input of the fusion node is not connected to the group stimulation but directly set to $s = 1$. This *permanent stimulation mode* is graphically indicated by a filled triangle blocking the stimulation input of the *Behaviour* in question (f). In this example, the inhibition inputs from the group are also connected to the fusion node, such that only the combined impact of the group is inhibited. Selective inhibition or stimulation of specific *Behaviours* in the network can be achieved with additional inhibition and stimulation links. The same holds for selective stimulation of external behaviours which can be modelled by exporting derived activities from particular *Behaviours*.

Further examples of hierarchical *Behaviour* networks will be evolved during the control design carried out in the following sections. For this design step, the author decided to start with the rather reactive *Behaviours* of the `Pilot`. Gradually more complex *Behaviours* encapsulating more and more state reside in the `Mediator` and shall be discussed afterwards. The `Navigator` component is beyond the scope of this work. A short summary and a collection of further reading will be provided in Section 8.4.

8.2 Pilot: Short-Range Navigation

As indicated above, the `Pilot` is responsible for short-range navigation, i. e. for negotiating terrain on a scale of a few metres. The mission of this component is keeping the robot out of trouble in case the input from the higher navigation layers is unfavourable. This task in particular requires strategies for avoiding obstacles in order to keep the vehicle manoeuvrable at all times.

RAVON's `Pilot` follows two general strategies for achieving collision-free navigation towards a given waypoint by means of rather reactive *Behaviours*. The lowest level of navigational competence is realised by *Behaviours* which try to steer the robot away from hindrances in a repulsive fashion (Definition 17). Furthermore, higher-level competences are realised by *Behaviours* which attempt to keep the robot out of trouble by guiding the vehicle towards favourable places in an attraction-based manner (Definition 18).

Behaviours following the former approach are integrated into a behaviour-based group called `Guardian` which reflects the idea that these competences prevent the robot and its environment from danger. The second class of *Behaviours* are combined into the behaviour-based group `Guide` which is supposed to underline the leadership of the `Pilot`. Figure 8.6 illustrates the partitioning of the `Pilot` into `Guardian` and `Guide` with annotated data flow. The `Guide` casts the provided *Local Way Point* into steering commands which are moderated by the `Guardian` to avoid collisions. The *Behaviour* signals *Activity* and *Target Rating* of both behaviour-based groups are combined using a *Fusion Behaviour* in maximum fusion mode to determine the `Pilot`'s *Activity* and *Target Rating.* The following definitions serve as guideline for assigning *Behaviours* to the two piloting components:

Repulsion-based Navigation (Definition 17) *This strategy merely reacts towards obstacles in the proximity of the vehicle. In case that nothing was detected in the current heading no measures are taken. Repulsion-based Navigation can never be designed to reach a target location. It is solely appropriate to guarantee collision-free navigation in a rather reactive fashion.*

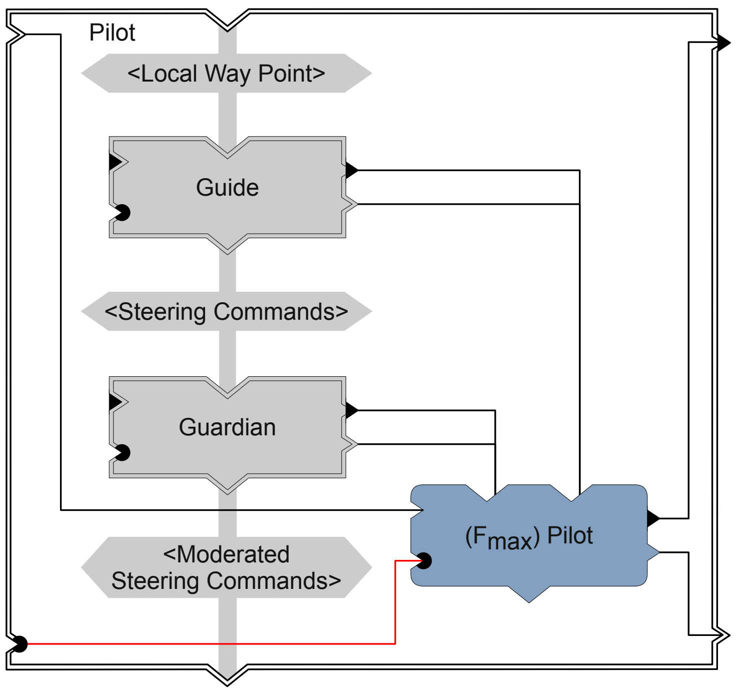


Figure 8.6: The `Pilot` encapsulates two behaviour-based groups, the `Guardian` and the `Guide`.

Attraction-based Navigation (Definition 18) *This strategy draws the robot towards* favourable *locations and is inherently goal-driven.* Favourable *in this context may mean towards a waypoint to reach, towards easy terrain to navigate, or towards a vehicle or person leading the robot.*

Note that attraction-based *Behaviours* in themselves need not take into account local terrain information as their task may not require this. Repulsion-based *Behaviours* further need not worry about goals as these may conflict with vehicle safety. Therefore, the repulsive `Guardian` and the attraction-based `Guide` have to be designed to complement one another to yield a robust short-range navigation performance.

8.2.1 Guardian: Repulsion-based Hull Protection

The `Guardian` is the fundamental safety component of RAVON's control system and is responsible for preventing damage from the robot and its environment. In order to guarantee safety at all times, this layer shall host rather reactive *Behaviours* which realise tight sensor-actor loops to slow the robot down in the proximity of obstacles and to steer it away from objects in a repulsive fashion.

Interface and structure of this layer are designed according to the iB2C **DOF access pattern** (see [Proetzsch 10b] p. 99) which recommends the logical separation of the control system according to the most abstract set of Degrees of Freedom (DOF) relevant on the particular level of competence. This supports separation of concerns and a fine-grained access to individual motion capabilities of the locomotion system. For structuring the `Guardian`, the robot's Degrees of Manoeuvrability (DOM) have been chosen. For double-ackerman kinematics these are velocity (in the directions forward and backward), rotation

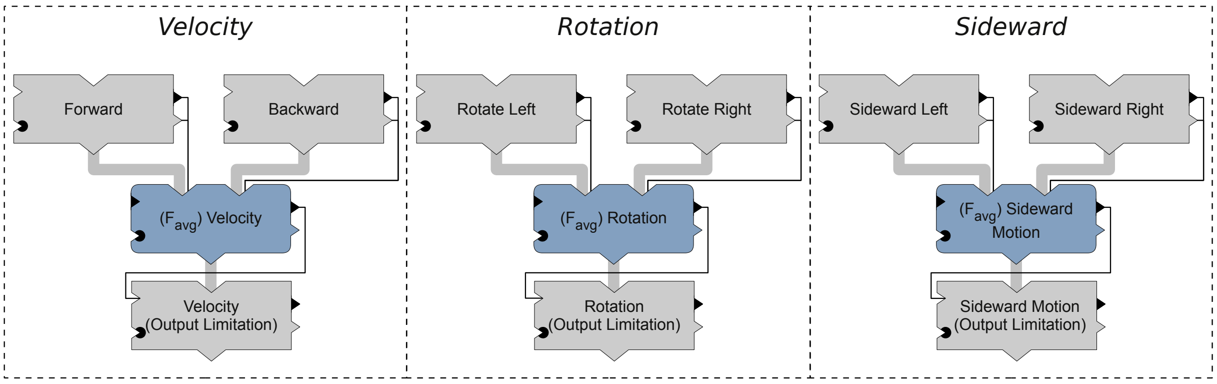


Figure 8.7: The three fundamental control chains of the `Guardian` component: Velocity, Rotation and Sideward Motion

(left/right), and sideward (left/right) resulting in three more or less independent control chains, each featuring two competing strands. Figure 8.7 illustrates the *Behaviour* network terminating the three fundamental control chains. The competing two strands of each control chain are combined using a fusion *Behaviour* in weighted average mode. The output limitation *Behaviour* represents the interface between the *Behaviour* network and the hardware abstraction layer. After this point, the *Behaviour* signals *Activity* and *Target Rating* are no longer considered because the controllers encapsulated by the hardware abstraction layer work on the raw control values. In order to control the motors as intended, the *Output Limitation Behaviours* draw the control values to 0 in case that the activation ι is 0.

Abstracting from the concrete vehicle with its DOM improves the portability of the control system between platforms which feature different locomotion systems. Even systems with incompatible DOM can be supported to a certain degree. On the one hand physically inexistent DOM can easily be deactivated in the control system. On the other hand, physically available DOM which have no counterpart in the control system can be integrated in terms of a new control chain. As far as no extension is made, the control system will gracefully compensate for missing DOM and ignore additional DOM. The proposed separation is also reflected in the control vector passed from the `Pilot` over the `Guardian` towards the vehicle abstraction which comprises normed values for velocity, rotation, and sideward motion. The vehicle abstraction uses the parameters and constraints of the real platform to compute and control concrete velocity and steering angles on the basis of the kinematic model for the double ackerman steering.

For fundamental safety of an agile platform as the one at hand, the entire vicinity of the robot (i. e. 360°) has to be monitored. The proposed approach approximates the robot's shape by piece-wise defined safety zones of similar shape all around the robot. That way, basic *Behaviours* remain simple and generic which supports testability and reusability (*Handler Simplicity* (Guideline 15 on page 69)). Furthermore, the complete vicinity of the robot can be taken into account for manoeuvring to yield a robust overall system behaviour. Note that *Representation Deployment* (Guideline 2 on page 67) may be required to provide the information basis for that purpose (see Chapter 9 for details on the sensor processing design).

Slow Down *Behaviours* Moderate the Velocity Control Chain

The velocity control chain is moderated by several *Slow Down Behaviours* monitoring the vicinity of the robot for obstacle proximity and the presence of ground clutter. For that purpose, a "protective field" of monitored regions approximating the rectangular shape of the robot is defined in terms of *Virtual Sensors*. Figure 8.8a illustrates the general idea of the approach and the regions to be monitored. Structurally, this is achieved by partitioning the "protective field" into four *Virtual Sensors* providing Cartesian *Views* at either side of the vehicle and four *Virtual Sensors* providing polar *Views* at the corners of the robot. This *Structural Partitioning* (Guideline 9) yields a set of simple structures which can be processed in a straightforward fashion. The *Behaviours* operating on the Cartesian *Views* simply react anti proportional to the proximity of obstacles in the monitored region. For the polar *Views*, an additional angle dependency is integrated in order to yield a continuous overall repulsive behaviour on the basis of the shape indicated by the light-blue overlay in Figure 8.8a. Each *Behaviour* monitors exactly one well-defined *View* supporting *Handler Simplicity* (Guideline 15). Note that only two classes of *Slow Down Behaviours* are required to realise the proposed concept. The complete coverage of the area around the vehicle can be achieved by multiple instances of these simple generic *Behaviour* classes.

Formally, the *Virtual Sensors* are defined using the sector-map specifications introduced in Section 4.4. The specifications of the Cartesian sector maps only differ in their (virtual) mount point on the robot and the extents in y-direction.

Using the acceleration formula from physics outlined in Equation 8.1, the maximum *range* of the virtual sensors can be determined from the maximum velocity v_{max} and the desired maximum deceleration a_{max}.

$$range = \frac{1}{2} \cdot \frac{{v_{max}}^2}{a_{max}} \tag{8.1}$$

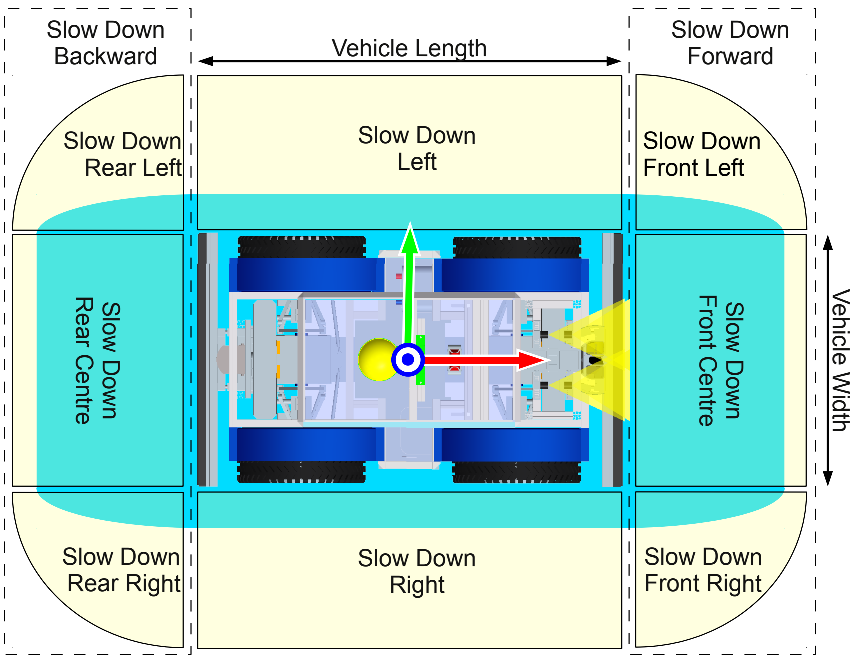
For the *Behaviour Slow Down Front Centre*, the *Virtual Sensor* would for example be specified as:

$$VirtualSensor_{SlowDownFrontCentre} = (origin, y_{start}, y_{stop}, \Delta y, range, mode)$$

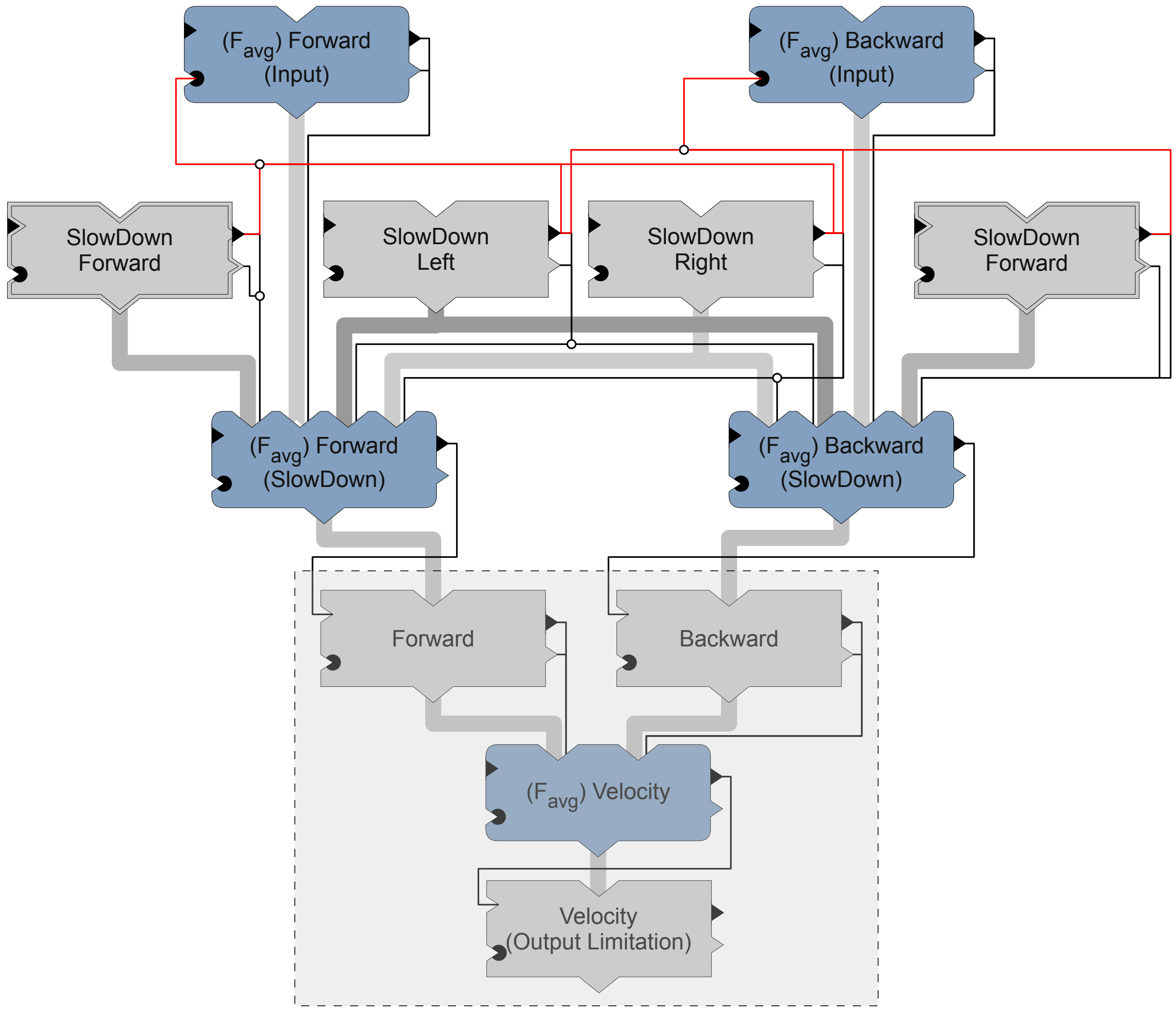
where

$$\begin{aligned} origin &= \left(\frac{VehicleLength}{2}, 0, 0^\circ\right) \\ y_{start} &= -\frac{VehicleWidth}{2} \\ y_{stop} &= \frac{VehicleWidth}{2} \\ \Delta y &= 20cm \\ range &= \frac{1}{2} \cdot \frac{{v_{max}}^2}{a_{max}} \quad \text{(see Equation 8.1)} \\ mode &= local \end{aligned}$$

Each sector in the specified sector map holds the distance to relevant objects (i. e. obstacles) in the covered portion of the terrain. The resulting vector of distance values represents the input vector $\vec{e}$ of the *SlowDown Behaviour*.



(a) Concept for *Slow Down*



(F$_{avg}$) Forward (Input)
(F$_{avg}$) Backward (Input)
SlowDown Forward
SlowDown Left
SlowDown Right
SlowDown Forward
(F$_{avg}$) Forward (SlowDown)
(F$_{avg}$) Backward (SlowDown)
Forward
Backward
(F$_{avg}$) Velocity
Velocity (Output Limitation)

(b) *Slow Down Behaviours*

Figure 8.8: Integration of the *Slow Down Behaviours* into the control system. The existing stub already introduced above (see Figure 8.7) is drawn in lighter colours.

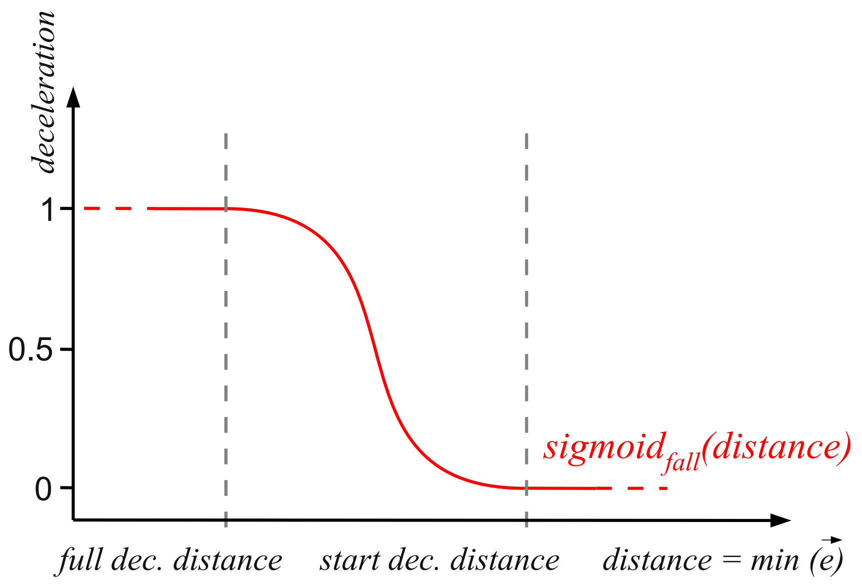


Figure 8.9: Sigmoid function as configured in *Behaviour SlowDown*$_{Cartesian}$.

let $n = \frac{-y_{start}+y_{stop}}{\Delta y}$ the number of sectors s_i with $i \in 1, \ldots, n$ in a sector map

let d_i the distance to obstacles in sector s_i

let $\vec{e} = (d_1, \ldots, d_n)^T$ the input vector of *Behaviour SlowDown*$_{Cartesian}$

let $\min_i(\vec{e}) = \min(d_i)$ the component-wise minimum of vector $\vec{e}$

then $\mathit{SlowDown}_{Cartesian} = (f_a, f_r, F)$ can be implemented using the iB2C building block $\mathit{sigmoid}_{fall}$ introduced in Section 8.1 as follows:

$$\begin{aligned}
\mathit{deceleration} &= \mathit{sigmoid}_{fall}(\min(\vec{e}))\\
\mathit{velocity} &= 0\\
a &= \mathit{deceleration} \cdot \iota\\
f_a &= (a)\\
f_r &= \mathit{deceleration}\\
F(\vec{e}, \iota) &= (\mathit{velocity})
\end{aligned}$$

The fundamental idea of *Slow Down Behaviours* is to guarantee that the vehicle does not move closer to objects than a specified distance. These *Behaviours* are thus designed to bring the robot to a halt and therefore, the command $\mathit{velocity} = 0$ is set through the output vector $\vec{u}$. The propagation of this command is modulated by the *Behaviour*'s activity function f_a. To reduce velocity in a graceful fashion, a sigmoid building block is configured with the parameters *start deceleration distance* and *full deceleration distance* resulting in a function mapping *distance* to interval $[0, 1]$ as illustrated in Figure 8.9. The closer the robot gets to hindrances, the higher is the normalised *deceleration* computed by the sigmoid. This potential deceleration desire is expressed by the target rating r

(remember that $r = 1$ indicates full dissatisfaction). Furthermore, the *Behaviour* tries to exert its goal to stop the vehicle (i. e. $\mathit{velocity} = 0$) by propagating the *deceleration* under moderation of the activation ι via activity a ($a = 1$ indicates full activity). In essence: the closer the hindrance, the more active and the less satisfied the *Behaviour*, and the more stringent the command to stop the vehicle. Note that this approach can be proven to guarantee that the vehicle comes to a halt at the specified distance to obstacles. Therefore, the proposed solution is suitable for fundamental vehicle safeguarding.

The *Virtual Sensors* mounted at the corners of the vehicle are specified in terms of four polar sector maps. As for the Cartesian sector maps, the specifications of all four are mostly identical. Only the position information specified in the specification entry *origin* of the maps differ in sign.

The *Virtual Sensor* for the *Slow Down Front Left Behaviour* would for example be specified as follows:

$$\mathit{VirtualSensor}_{\mathit{SlowDownFrontLeft}} = (\mathit{origin}, \psi_{start}, \psi_{stop}, \Delta\psi, \mathit{range}, \mathit{mode})$$

where

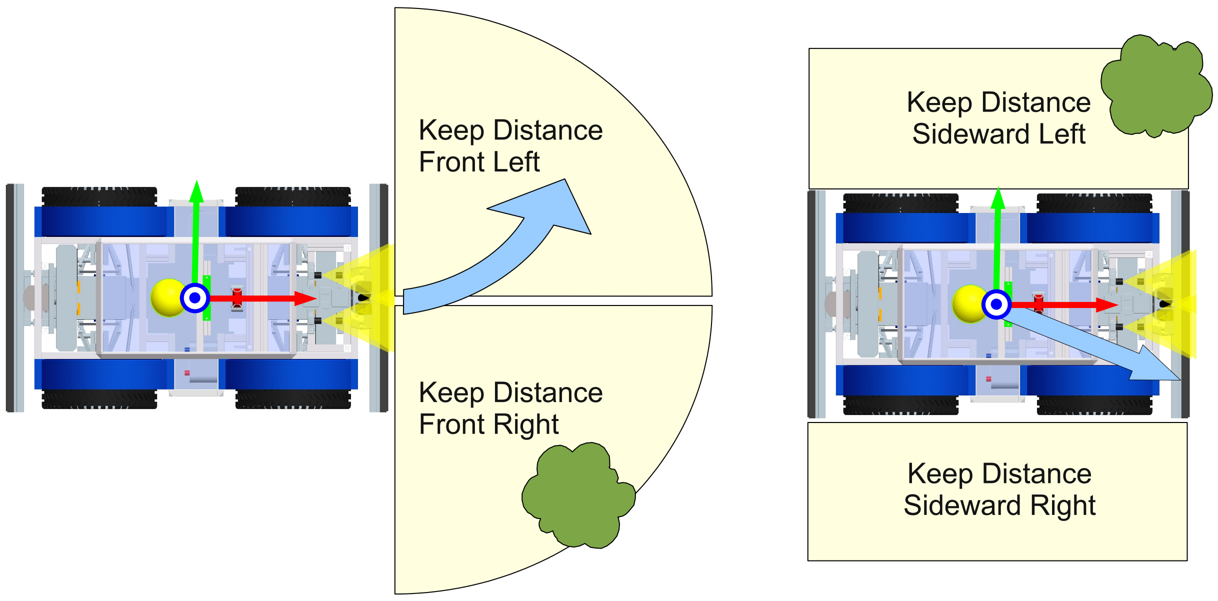
$$\begin{aligned}
\mathit{origin} &= \left(\frac{\mathit{VehicleLength}}{2}, \frac{\mathit{VehicleWidth}}{2}, 45^\circ\right)\\
\psi_{start} &= -45^\circ\\
\psi_{stop} &= 45^\circ\\
\Delta\psi &= 5^\circ\\
\mathit{range} &= \frac{1}{2} \cdot \frac{{v_{max}}^2}{a_{max}} \qquad \text{(see Equation 8.1)}\\
\mathit{mode} &= \mathit{local}
\end{aligned}$$

The remaining polar sector maps only differ in signs of the position values of *origin* and can be derived from the specification above in a straightforward fashion. All *Slow Down* sector maps are supposed to move with the robot and have therefore been set to local *mode*. The *Virtual Sensor* specifications represent the structural part of the control-level input to the *Aspect-oriented Configuration* (see Section 6).

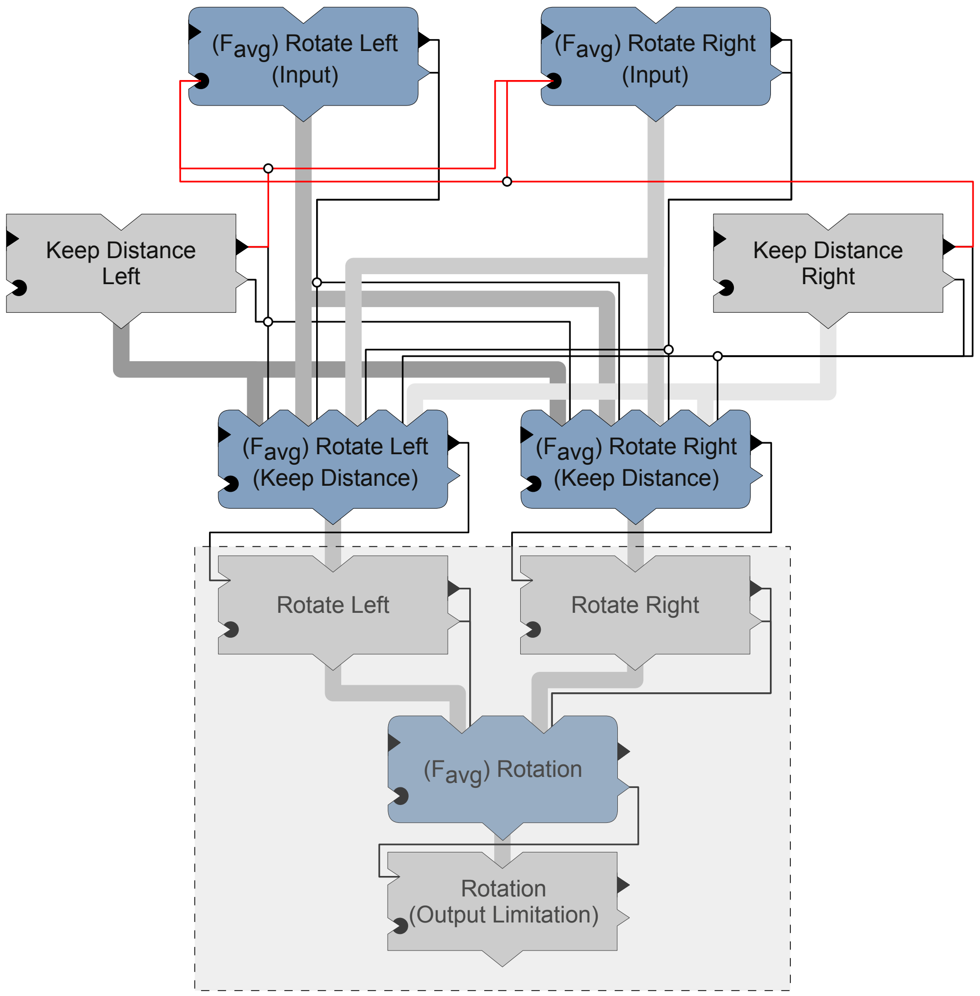
The integration of the *Slow Down Behaviours* into the control network is depicted in Figure 8.8b. On the logical control level, the affected motion direction is more interesting than the concrete nature of the particular *Behaviour* slowing down the vehicle (remember the **DOF access pattern**). For that reason, *Behaviours* monitoring the front side of the robot have been grouped into the behaviour-based group *Slow Down Forward* according to the iB2C **Behavioural group pattern** (see [Proetzsch 10a] p. 100). The same pattern can be applied to combine the *Behaviours* monitoring the rear side. Note that the structural similarity of both groups allows for the reuse of one generic *Slow Down* group, which has already been introduced in Figure 8.5 on page 85 (**Reuse components** [Proetzsch 10a] p. 104).

Evasive *Behaviours* Keep the Robot away from Threats

The rotational and the sideward control chains are influenced on a similar information basis to steer the robot away from obstacles and unfavourable ground conformations. Both chains



(a) Concept for *Keep Distance*



(b) *Keep Distance Behaviours*

Figure 8.10: Integration of the *Keep Distance Behaviours* into the control system. The existing stub already introduced above (see Figure 8.7) is drawn in lighter colours.

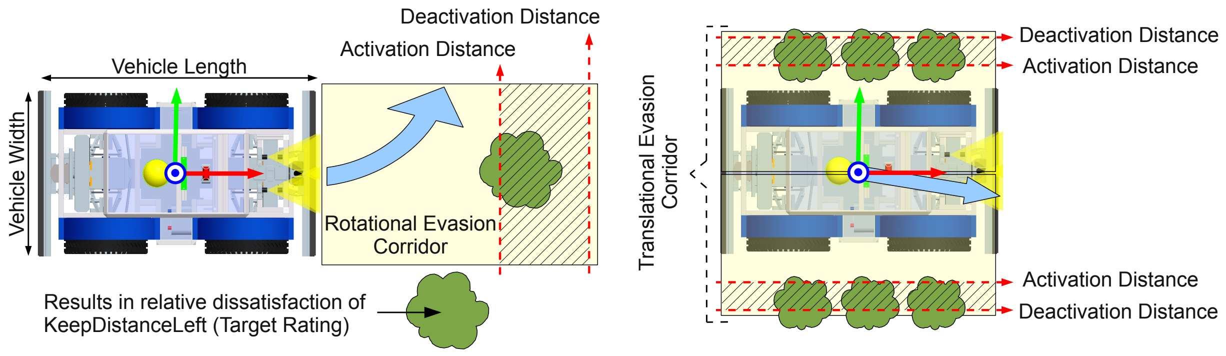
have an impact on the steering of the vehicle and have therefore been designed to share the same fundamental classes of evasive *Behaviour* modules. *Keep Distance Behaviours* attempt to guide the robot through openings between obstacles in an antagonistic fashion by steering the robot into the opposite direction when obstacles get close to one side of the robot. Figure 8.10a illustrates the idea of keeping the robot away from obstacles by rotational and sideward motion.

For the rotational control chain, regions in front of the robot are monitored. For the sideward control chain, regions to both sides of the robot provide the required environment information. The *Behaviour* network resulting from the integration of the *Keep Distance Behaviours* is illustrated in Figure 8.10b by example of the rotational control chain. In case *Keep Distance Left* becomes active, the steering commands from above are weakened via inhibition of the input fusion *Behaviours Rotate Left (Input)* and *Rotate Right (Input)*. At the same time, commands steering the robot to the right are propagated to the fusion nodes *Rotate Left (Keep Distance)* and *Rotate Right (Keep Distance)*. *Keep Distance Right* reacts identically on objects to the right of the robot, yielding the desired competitive overall behaviour. Note that the network for the sideward chain is structurally identical to the network for the rotational chain. Therefore, a generalised *Behaviour* network can be instantiated for rotational and for sideward motion according to the iB2C guideline **Reuse components** ([Proetzsch 10a] p. 104).

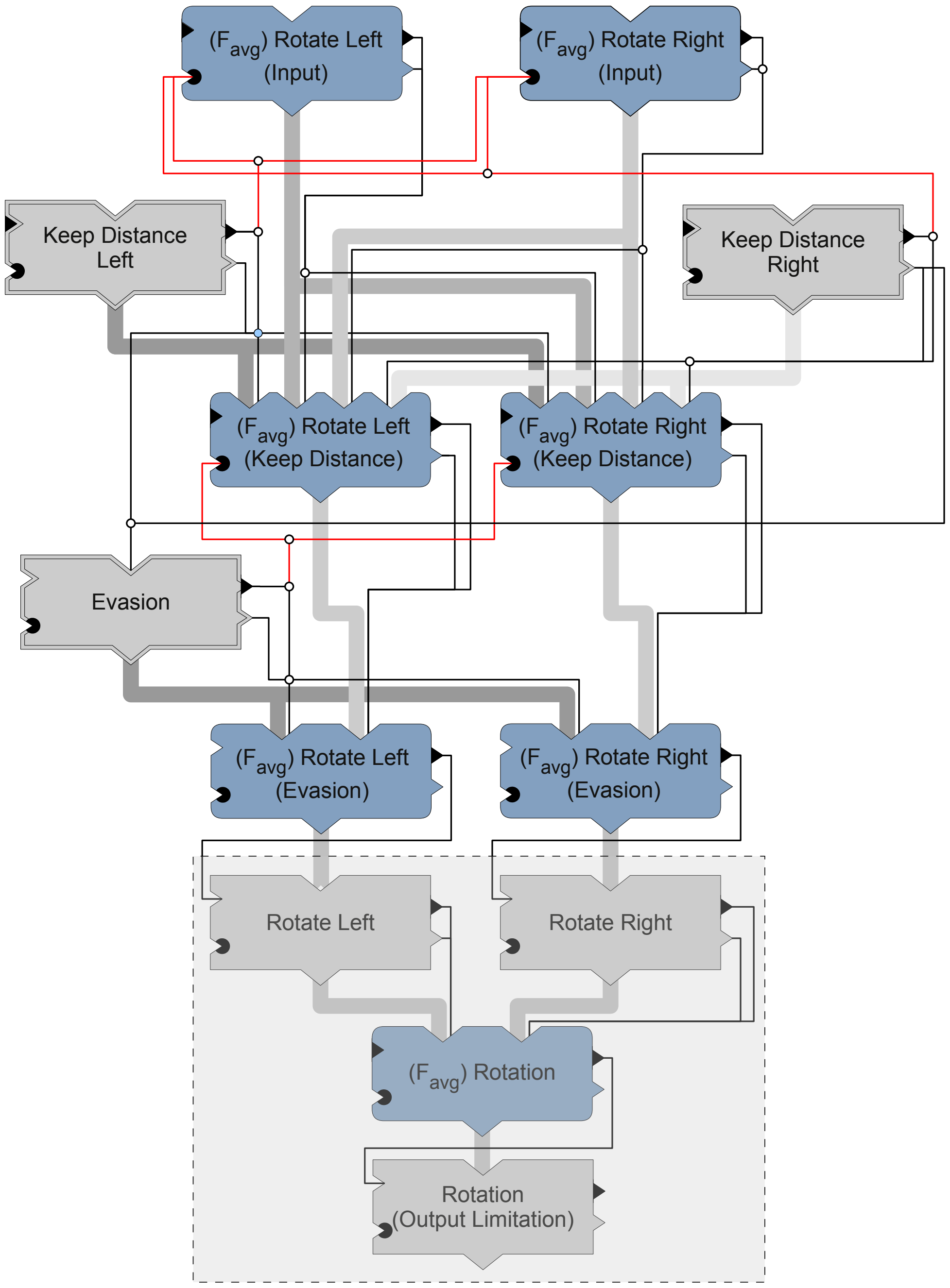
As long as the obstacle density is low and passages exist that are wide enough for the robot to pass through, the approach outlined above will succeed. In case no passage exists or the robot is too large to pass through, a local arbitration is proposed. For that purpose a further level of competence realised with the *Behaviour* class called *Evasion* shall be introduced. Note that the rotational and sideward control chain require different arbitration. The rotational chain *Evasion* has to evade obstacles in the centre of the corridor while the sideward chain *Evasion* has to centre up the robot in narrow passages when both lateral *Keep Distance Behaviours* are already fully active.

Nonetheless, the general arbitration structure of both *Behaviours* is similar in the sense that corner cases of the purely reactive *Keep Distance Behaviour* network are solved. *Evasion Behaviours* monitor the central corridor of the vehicle's path. In case that objects reach extreme proximity (that is proximity which goes beyond the distance where the accordant *Keep Distance Behaviour* unfolds full activity), the *Evasion Behaviours* arbitrate between the competing *Keep Distance Behaviours* by supporting the steering direction of the less satisfied *Keep Distance* behaviour. This effect is firstly achieved explicitly by providing a more strict steering angle and implicitly by inhibiting the *Keep Distance* behaviours.

The evasion corridors are monitored with Cartesian *Views* as illustrated in Figure 8.11a. The rotational variant operates on a single *View* which reflects objects of relevance in front of the vehicle while the translational variant requires two *Views* which reflect the distances to relevant objects to either side of the robot. Besides the environment information on the corridor in question, *Evasion Behaviours* therefore require the *Target Rating* signals (which model *Behaviour* satisfaction) of the *Keep Distance Behaviours* to arbitrate (see Figure 8.11b). Since the abstract situational assessment in terms of the *Target Ratings* already holds all necessary information, no additional terrain data has to be processed according to guideline **Take advantage of behaviour signals** (see [Proetzsch 10a] p. 104). This adds to behaviour simplicity (i. e. *Handler Simplicity* (Guideline 15)) and



(a) Concept for *Evasion*


(F$_{avg}$) Rotate Left (Input)
(F$_{avg}$) Rotate Right (Input)
Keep Distance Left
Keep Distance Right
(F$_{avg}$) Rotate Left (Keep Distance)
(F$_{avg}$) Rotate Right (Keep Distance)
Evasion
(F$_{avg}$) Rotate Left (Evasion)
(F$_{avg}$) Rotate Right (Evasion)
Rotate Left
Rotate Right
(F$_{avg}$) Rotation
Rotation (Output Limitation)

(b) *Evasion Behaviours*

Figure 8.11: Integration of the *Evasion Behaviour* Group into the control system. The existing stubs already introduced above are drawn in lighter colours.

reduces the computational load, as preprocessed information is reused on a higher level of abstraction.

This *Behaviour* interaction is a concrete example of how neighbouring levels of competences can be designed to overlap each other (*Competence Overlap* (Guideline 7 on page 68)) in order to guarantee graceful control handover and to yield a robust overall performance. Note that the *Evasion Behaviours* – in contrast to all other *Behaviours* introduced so far – require an internal state. In order to prevent oscillations, the *Evasion Behaviours* realise a hysteresis-based state switching which activates evasive manoeuvres when objects are closer than the *Activation Distance* and deactivates the *Behaviour* when the corridor is clear up to the *Deactivation Distance*.

Structuring the Guardian with Behaviour-based Groups

With the proposed sets of *Behaviours* and according *Views*, RAVON can avoid obstacles by slowing down and steering away. So far, the three control chains velocity, rotation, and sideward have been regarded in isolation. Before adding more capabilities to RAVON's control system, some further substructuring shall be carried out to highlight how hierarchical control systems can be designed in an incremental fashion.

Figure 8.12 shows the structuring of the `Guardian`. At the bottom of the diagram, the *Behaviour* subnetworks for fusing the two strands of each control chain into a single control value have been wrapped into separate groups. These behaviour-based groups have been named *Velocity*, *Rotation*, and *Sideward* according to the degree of manoeuvrability influenced by the control chain in question. The obstacle avoidance facilities in each control chain have also been cast into individual behaviour-based groups. The *Slow Down Behaviours* have been grouped under the name *Deceleration Obstacle Avoidance*. In the rotational and the sideward control chain, the according *Keep Distance* and *Evasion Behaviours* have also been combined into separate groups, yielding the two behaviour-based groups *Rotational Obstacle Avoidance* and *Sideward Obstacle Avoidance*.

The particular behaviour-based *ObstacleAvoidance* groups are coupled into the respective control chain using the iB2C **Inhibition layer pattern** (see [Proetzsch 10a] p. 98). According to this pattern, the *Behaviour* to be coupled in is framed by layers of fusion nodes in weighted fusion mode. In case the *Behaviour* wants to contribute to the control chain, the influence of control values from above are weakened by inhibiting the upper layer of fusion *Behaviours* (Inhibition Layer). In parallel, the control values computed by the *Behaviour* in question are coupled into the control chain via the lower fusion layer (Coupling Layer). If the *Behaviour* remains inactive because no interference is required, the control values from above are propagated through both fusion layers without change. For more details on iB2C patterns see [Proetzsch 10b] p. 95ff.

Seamless Drive Mode Selection

For robust navigation, off-road robots require a variety of sensor systems which cover a wide range of detection techniques. The suitability of sensor systems and obstacle detection approaches is highly dependent on the terrain type at hand. In open land where hindrances can easily be distinguished from traversable terrain, far-ranging visual techniques with high update rates can be used to navigate at high speeds. In environments where vegetation obstructs the load-bearing surface or narrow passages have to be negotiated in order

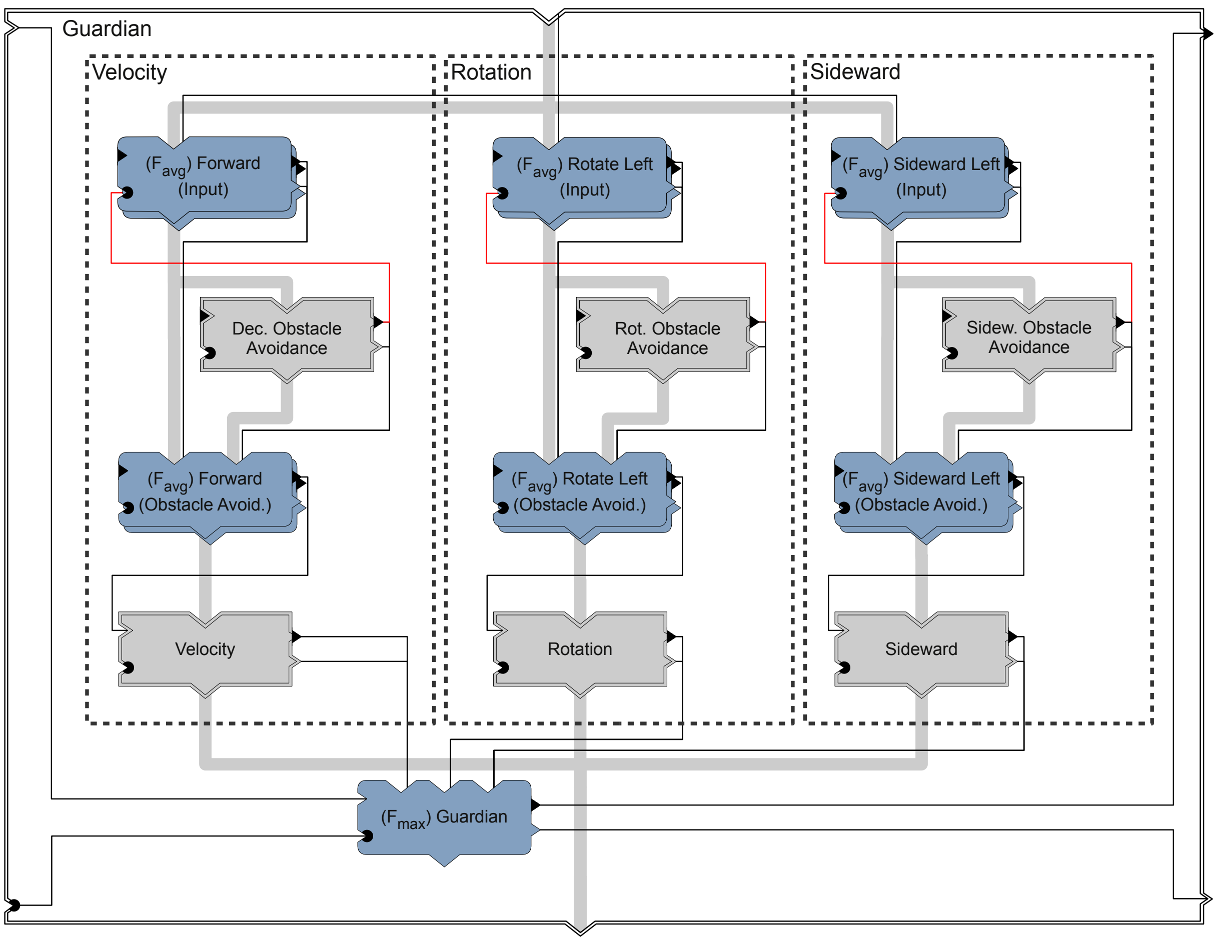

Guardian
Velocity
Rotation
Sideward
(F_avg) Forward (Input)
(F_avg) Rotate Left (Input)
(F_avg) Sideward Left (Input)
Dec. Obstacle Avoidance
Rot. Obstacle Avoidance
Sidew. Obstacle Avoidance
(F_avg) Forward (Obstacle Avoid.)
(F_avg) Rotate Left (Obstacle Avoid.)
(F_avg) Sideward Left (Obstacle Avoid.)
Velocity
Rotation
Sideward
(F_max) Guardian

Figure 8.12: Structure of the `Guardian`.

to reach goals, higher resolutions at lower ranges are of interest. For densely vegetated environments visual systems may not provide a solution at all, as obstacles may completely be hidden in vegetation. In these situations, alternative sensing systems, such as tactile sensors, have to be called into service. Without going into detail with various detection mechanisms, the key questions to be solved at design time are:

1. "Which obstacle detection mechanisms are suitable for which terrain type?"

2. "Which velocity is suitable for which terrain type?"

3. "How can the robot find out which terrain type it is currently operating in?"

While the first two questions can be solved empirically by thorough statistical analysis of obstacle detection facilities and the underlying physical principles (sensor range, sensor reliability on a given terrain, etc.), the latter is a central scientific question in robot control. Research in off-road robotics tends to treat this question as a sensor problem which led to the development of elaborated terrain classification schemata [Hebert 03, Jansen 05, Lalonde 06]. Terrain classification can be regarded as a kind of meta obstacle detection which is designed to determine the terrain type at hand explicitly in order to choose the suitable information basis and actions accordingly. This approach yielded many powerful – mostly visual – classification techniques which allowed more robust navigation in changing environments. The problem with terrain classification is that the selection of suitable actions is only as good as the classification itself and therefore the very same problems as before arise in terrain that was not anticipated at design time. Terrain classification thus does not actually provide a scalable solution for question 3 as the control side problem is not solved.

In this work, question 3 is regarded as a problem in the *Action Selection* domain which shall be treated at control design time. The fundamental idea here is to deploy an action selection mechanism which implicitly switches between different overlapping *Drive Modes*. *Drive Modes* in that context refer to questions 1 and 2 by defining the maximum velocity of the robot and the data basis to be used in a given situation. In order to answer question 3, the velocity of *Dec. ObstacleAvoidance Behaviour* groups computed on the different obstacle data bases is used to determine which configuration allows for the quickest negotiation of the current driving situation.

In order to illustrate the *Drive Mode Selection* mechanism, three major terrain types shall be distinguished:

1. *Open Land*: tracks and clearings with a low density of obstacles which can be negotiated at *Maximum Velocity*.
2. *Underwood*: narrow passages and areas beyond forest tracks which feature a high density of obstacles which have to be handled with *Moderate Velocity*.
3. *Grassland*: areas with intense high vegetation which can only be traversed at *Tactile Creep* velocity.

For each terrain type one combination of obstacle information and according maximum velocity for negotiating terrain on this data basis is configured. Figure 8.13 illustrates the extensions introduced into the `Guardian`. The *Drive Mode Selection* is coupled into the velocity control chain to explicitly moderate the robot's maximum velocity. The *Obstacle Detection* groups in all three control chains are replicated for the three *Drive Modes* – *Maximum Velocity*, *Moderate Velocity*, and *Tactile Creep* – to operate on suitable environment information. The velocity of each *Drive Mode* is fed back to the *Drive Mode Selection* to support the decision which *Drive Mode* to activate. To choose a particular *Drive Mode*, the *Drive Mode Selection* stimulates the corresponding set of *Obstacle Avoidance* groups such that the according control parameters for velocity, rotation, and sideward motion are propagated. The *Drive Mode Selection* mechanism thus arbitrates between the *Drive Modes* on suggested velocity values from the *Dec. Obstacle Avoidance* behaviours.

In order to allow the robot to slowly "creep" through high grass, the *Dec. Obstacle Avoidance* behaviours annul the impact of visual sensor systems in *Drive Mode Tactile Creep*. For that reason an additional group of *Emergency Stop Behaviours* is coupled in further down the velocity chain. These *Behaviours* assure static vehicle safety at all times by monitoring the robot's pitch and roll angles, as well as bumper events. In case any of these emergency situations arises, the robot will be stopped at once and will further be inhibited from proceeding in the former driving direction. Besides these basic safety

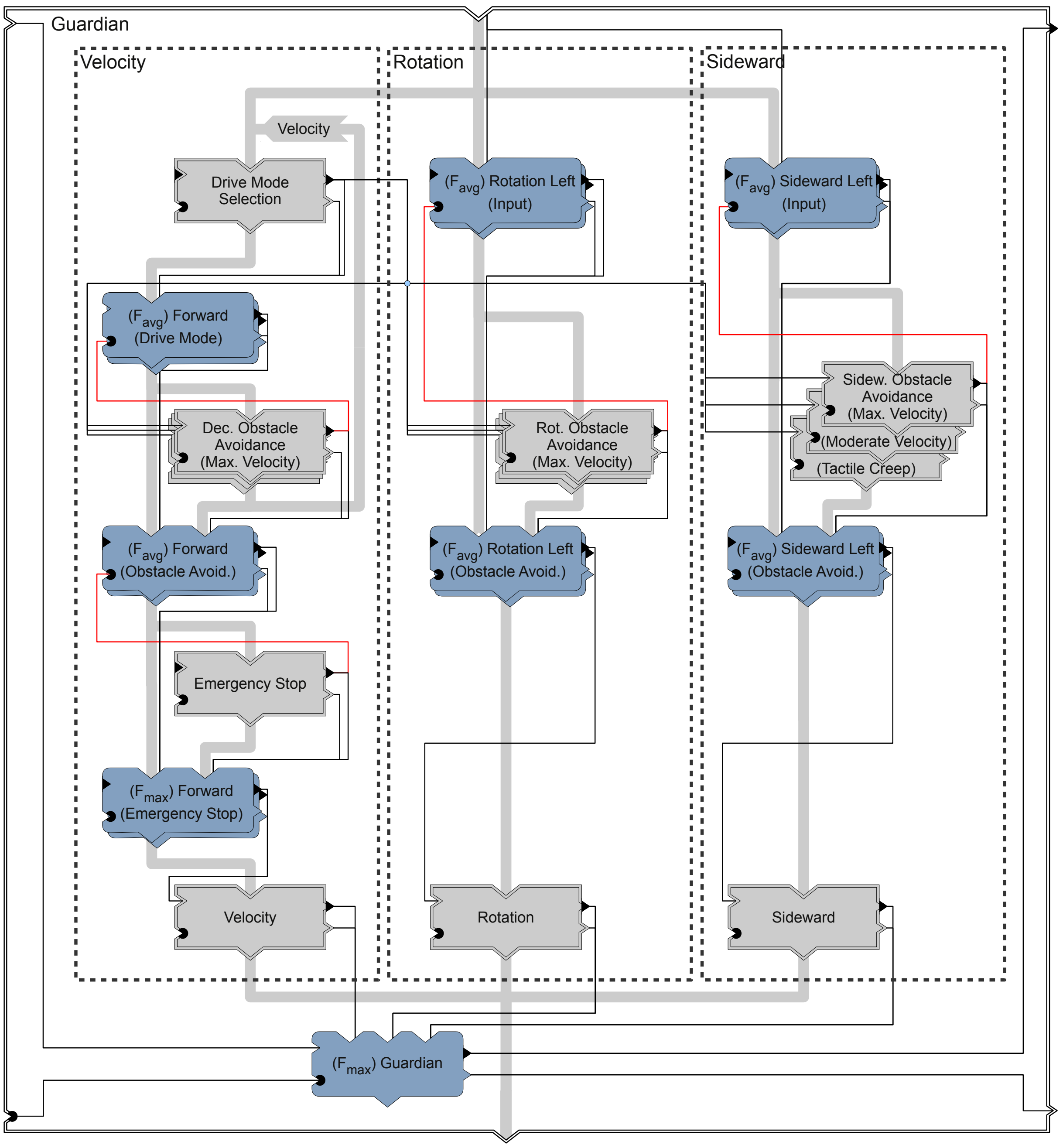

Guardian
Velocity
Rotation
Sideward
Velocity
Drive Mode Selection
(F$_{avg}$) Rotation Left (Input)
(F$_{avg}$) Sideward Left (Input)
(F$_{avg}$) Forward (Drive Mode)
Sidew. Obstacle Avoidance (Max. Velocity)
(Moderate Velocity)
(Tactile Creep)
Dec. Obstacle Avoidance (Max. Velocity)
Rot. Obstacle Avoidance (Max. Velocity)
(F$_{avg}$) Forward (Obstacle Avoid.)
(F$_{avg}$) Rotation Left (Obstacle Avoid.)
(F$_{avg}$) Sideward Left (Obstacle Avoid.)
Emergency Stop
(F$_{max}$) Forward (Emergency Stop)
Velocity
Rotation
Sideward
(F$_{max}$) Guardian

Figure 8.13: The `Guardian` is arranged in three rather independent control chains, namely *Velocity, Rotation, Sideward*, which reflect the degrees of manoeuvrability (DOM) of the vehicle. All three DOM are monitored by *Obstacle Avoidance Behaviours* which keep the robot away from hazards in a repulsive fashion. The *Velocity* chain further features an *Emergency Stop Behaviour* which brings the robot to an abrupt halt in extreme situations. In addition to that, a separate *Drive Mode Selection Behaviour* monitors the suitability of obstacle detection data and chooses the *Drive Modes* accordingly. A *Drive Mode* in that context means an appropriate combination of maximum velocity and interpretation of obstacle information. The *Drive Mode Selection* thus chooses a *Drive Mode* by limiting the velocity command and stimulating the appropriate *Obstacle Avoidance Behaviours* for deceleration, rotation, and sideward. Besides the *Drive Mode Selection* all *Behaviours* couple into the control chains using the **Inhibition layer pattern** which results in alternating layers of *Behaviour* fusion nodes and *Behaviour* nodes. The fusion nodes are responsible for passing the command values on if the *Behaviour* coupled in is inactive. For the *Drive Mode Selection*, a bypass in case of inactivity is not desired due to the design of this component.

systems, particular aspects of visual obstacle detection systems may have to be rated critical, as well. Therefore, an additional set of *Slow Down Behaviours* is integrated into the *EmergencyStop* group which is configured to operate on fine-grained environment information. This information basis is supposed to comprise only critical obstacle classes (for instance water hazards, negative, and overhanging obstacles) which may never be neglected. In contrast to the *Slow Down Behaviours* in the *Dec. Obstacle Avoidance* group, the *Slow Down Behaviours* in the *Emergency Stop* group always stop the vehicle completely with maximum deceleration.

In total, four distinct data bases are required for the behaviour-based network of the `Guardian`. One for each of the three *Drive Modes* and one more for the *Emergency Stop*. Each data basis can be modelled as a separate control-level aspect. For that purpose, a semantic placeholder is declared in terms of a property set which has to be configured later in the *Aspect-oriented Configuration* (see Chapter 10). The safety *Behaviours* encapsulated by the `Guardian` are kept as simple and robust as possible. Therefore, each of the aspects only contains a single property.

$$\begin{aligned} \mathit{driveModeMaximumVelocityPROP} &= (\mathit{obstaclesMaximumVelocity}) \\ \mathit{driveModeModerateVelocityPROP} &= (\mathit{obstaclesModerateVelocity}) \\ \mathit{driveModeTactileCreepVelocityPROP} &= (\mathit{obstaclesTactileCreepVelocity}) \\ \mathit{emergencyStopPROP} &= (\mathit{obstaclesEmergencyStop}) \end{aligned} \tag{8.2}$$

These property sets represent the semantic part of the control-level input to the *Aspect-oriented Configuration*.

The *Drive Mode Selection* is realised as a behaviour-based group. Each *Drive Mode* is instantiated in terms of a *Drive Mode Behaviour*. As depicted in Figure 8.14, these *Behaviours* are arranged in a priority-based cascade with ascending velocity limit. The *Behaviour* with the lowest velocity limit – here *Drive Mode Tactile Creep* – is the default *Drive Mode* which operates as the ultimate fall back. Each *Behaviour* locally decides whether the *Drive Mode* it represents is more appropriate than the next lower-priority *Drive Mode*.

The velocity ranges of the *Drive Modes* are overlapping such that this decision can be reduced to checking the proposed velocities from the corresponding *Dec. Obstacle Avoidance* groups. In case the proposed velocity of the higher-priority *Drive Mode* is higher, the environment may apparently be negotiated at higher velocities.

For that purpose, the lower-priority *Drive Mode Behaviour* is inhibited and the *Limited Velocity* as proposed by the *Behaviour* in question is propagated to the next higher-priority *Drive Mode*. In the opposite case that the lower-priority *Drive Mode* proposes the higher velocity, the terrain at hand is evidently too obstructed for the *Dec. Obstacle Avoidance Behaviours* of the higher-priority *Drive Mode*. Thus switching to a more moderate *Drive Mode* on the basis of filtered environment information is carried out. Similar to a duplex clutch, the control handover between adjacent *Drive Modes* takes place in a seamless and gradual fashion.

Note that this concept for *Drive Mode Selection* – by creating an explicit connection between environment information and the suitable maximum velocity at which that kind of terrain is negotiable – to implicitly answer the following essential questions from the control side:

- "Can the terrain at hand be negotiated faster?"
- "Must the terrain at hand be negotiated slower?"

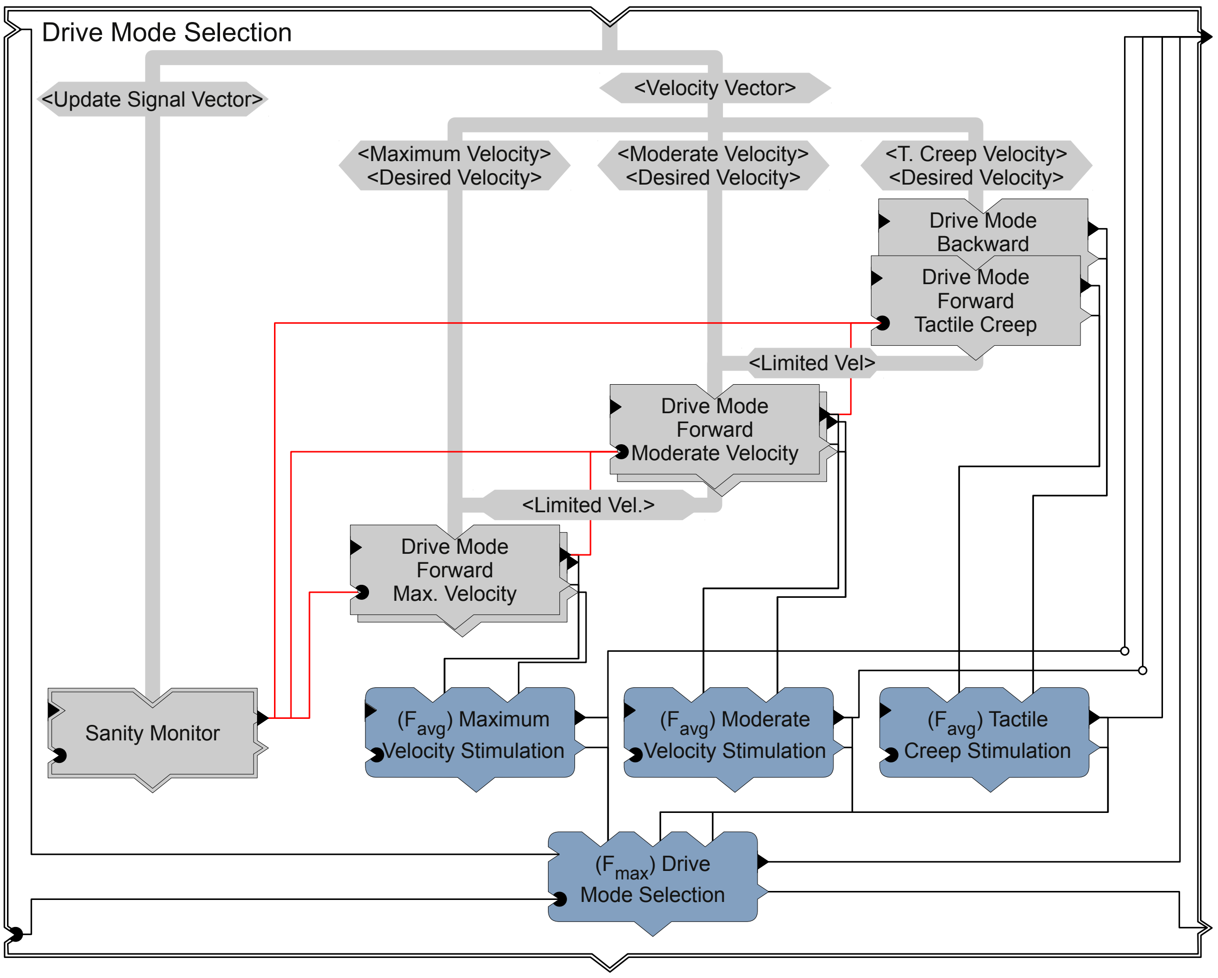

Drive Mode Selection
<Update Signal Vector>
<Velocity Vector>
<Maximum Velocity>
<Desired Velocity>
<Moderate Velocity>
<Desired Velocity>
<T. Creep Velocity>
<Desired Velocity>
Drive Mode Backward
Drive Mode Forward Tactile Creep
<Limited Vel>
Drive Mode Forward Moderate Velocity
<Limited Vel.>
Drive Mode Forward Max. Velocity
Sanity Monitor
(F_avg) Maximum Velocity Stimulation
(F_avg) Moderate Velocity Stimulation
(F_avg) Tactile Creep Stimulation
(F_max) Drive Mode Selection

Figure 8.14: The *Drive Mode Selection* is realised as a priority-based cascade of *Drive Mode Behaviours* which limit the maximum velocity according to the proposed velocity of the according *Dec. Obstacle Avoidance Behaviours*.

That way, a generic gradual switching mechanism is realised that allows the robot to autonomously decide what navigation strategy is apt for the terrain at hand. This decision is supported by identical *Behaviour* networks (guideline **Reuse components** [Proetzsch 10a] p. 104) operating in parallel on different views of the world making an explicit meta classification obsolete. As each *Drive Mode* only interacts with its neighbouring *Drive Modes*, this mechanism is inherently extensible from the control design side due to *Behaviour* locality (guideline **Prefer locality** [Proetzsch 10a] p. 103). The navigational competences outlined above can be complemented with further nuances by specification of further semantic translations-velocity pairs. The schematisation of representation and translation (see Chapter 4) however is the key for providing the tailored information bases to solve the control side problem of action selection in varying terrain by *Overlapping Competences* (Guideline 7 on page 68). Note that *Property Selectivity* (Guideline 12 on page 69) of the representations deployed in the sensor processing facilities is a further building block for tailored views on the environment. Chapters 9 and 10 will provide deeper insights into the fundamental way of thinking in abstract representations.

Drive Mode Selection and Sanity Monitoring

The sensor systems are the most fragile parts of a mobile robot and may therefore become inoperable during mission runtime. In order to prevent the robot from crashing into obstacles due to missing sensor data, watch dog systems are a common means to realise fault-tolerant control systems [Zhang 08]. Furthermore, critical missions do not allow for human intervention and therefore a graceful degradation of robot performance would be appreciated. For that reason, the *Drive Mode Selection* represents an ideal coupling point for watch dog functionality. The behaviour-based group *Sanity Monitor* watches over the health status of all obstacle detection facilities on the basis of abstract update signals. These signals are generated by arbitrarily complex plausibility checks concerning sensor data and algorithm output. *Sanity Monitor Behaviours* compute a main-loop-independent update frequency for each plausibility input, which is used to determine whether an obstacle detection facility is sane or not using thresholds. In case a certain group of sensor systems fails, the maximum velocity can be reduced by inhibiting particular *Drive Mode Behaviours*. This concept can be regarded as a gracefully degrading software watch dog system located on a higher level of control. Note that this facility also allows for gradual integration testing of particular (groups of) obstacle detection algorithms.

The nature of all the *Behaviours* introduced so far allows for regarding data from different sources independent of each other. For each additional source of information, the *Behaviours* are replicated and fusion is carried out on the control level as proposed in [Schäfer 05b]. At this point *Deferred Fusion* (Guideline 6 on page 68) results in improved extensibility as new information sources can seamlessly be integrated without changing the structure of the control system. Note that this design further adds to the traceability of integration tests for sensor processing facilities as the deactivation of algorithms results in a graceful degradation of navigational performance but never in an entirely different overall behaviour. Since the `Guardian` *Behaviours* work on the individual sources of information, tight sensor-actor-loops are achieved which improve the reactivity and robustness of this safeguarding facility. Decisions taken on potentially outdated information by higher levels of control are compensated by design.

8.2.2 Guide: Attraction-based Hull Protection

As already mentioned above, attraction-based *Behaviours* are grouped into behaviour-based group `Guide`. The *Behaviours* on this level of competence primarily realise the target approach and low-level ranking capabilities. Furthermore, the cooperative control, which allows for the seamless integration of tele-operation into the autonomous navigation system, is located in this group. Cooperative control allows the operator to assist the robot by providing guidance in terms of coarse velocity and heading commands, while the robot's safety *Behaviours* assure collision-free navigation. This is particularly handy for low-bandwidth tele-operation. The behaviour-based cooperative control approach allows for gradual operator interference during autonomous navigation ranging from pure tele-operation over assisted tele-operation towards full autonomy. Tele-operation is not part of this work and is only mentioned to give the reader an idea of the control system flexibility. For further details on these issues see [Armbrust 10c].

Target Approach

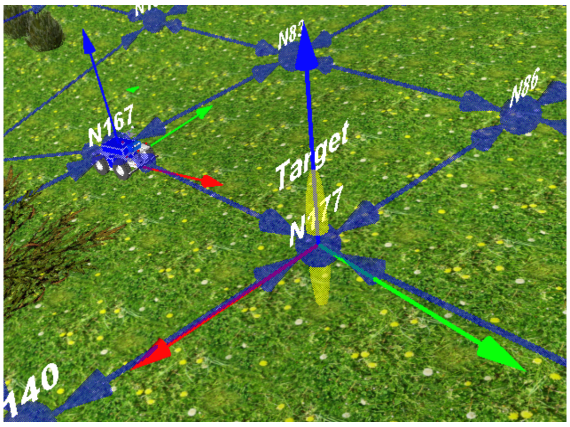
The *Target Approach* is responsible for drawing the robot towards a *Local Way Point* provided in terms of WCS coordinates[4]. This functionality is realised as a rather simple pose controller, which tries to reach a designated target pose by controlling the distance to the target (*Behaviour Point Attractor Forward*) and the angular deviation (*Behaviour Point Attractor Rotation*) from the target pose. To account for the non-holonomy of the vehicle, an intermediate target is computed which resides on a circular path reflecting the robot's feasible steering radius. The designated orientation at the intermediate target is set tangential to the circular path. That way, the desired orientation can be met at the target point with a simple controller. Figure 8.15 illustrates the idea of the *Target Approach.* To keep the *Target Approach* as simple as possible (*Handler Simplicity* (Guideline 15)), no environmental information besides the current pose of the vehicle is taken into account for controlling the approach. The safety *Behaviours* in the `Guardian` implicitly interact with the *Target Approach* through the environment by altering the vehicle's path in case obstacles have to be evaded. The intermediate target is continuously recomputed such that changes in course, resulting from evasive manoeuvres, are incorporated into the target approach.

Open Terrain Attraction

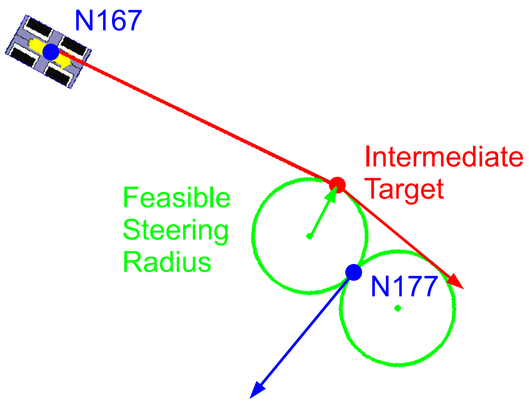
In addition to the repulsive safety *Behaviours*, the *Target Approach* is further supported by the *Open Terrain Attractor Behaviour* which draws the robot towards open terrain. The task of this *Behaviour* is to evaluate environmental information in heading direction to determine the most promising course towards obstacle-free terrain. That way, this *Behaviour* complements the safety *Behaviours* by guiding the robot away from trouble in a proactive fashion if there is enough space (*Competence Overlap* (Guideline 7)).

Figure 8.16a illustrates the concept of the open terrain attraction supported target approach. The structure of the *Behaviour* network designed for this purpose is depicted in Figure 8.16b. The *Target Approach* is split into three separate *Behaviours*. *Behaviour Target Approach with Orientation* continuously computes the coordinates for the oriented approach towards the `Local Way Point` provided by the next-higher level of competence. This *Behaviour* further coordinates the target approach by stimulating the position controller *Point Attractor Forward* and the orientation controller *Point Attractor Rotation* when a new navigation target is available. In order to determine whether the (intermediate) target point has been reached, the target ratings of the two pose controlling *Behaviours* are monitored

[4]The target approach is provided with a rather dense sequence of way points (in the dimension of a few metres) by higher planning facilities. In order to avoid problems with GPS-induced leaps in localisation, the *locally stable pose* is used for short-range navigation (remember Chapter 7 on page 75)

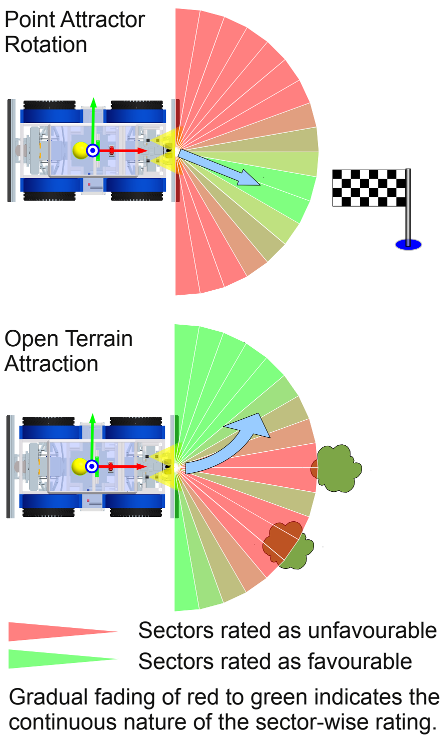


(a) Short-range navigation command from node N167 to N177 with orientation −90° in simulation.

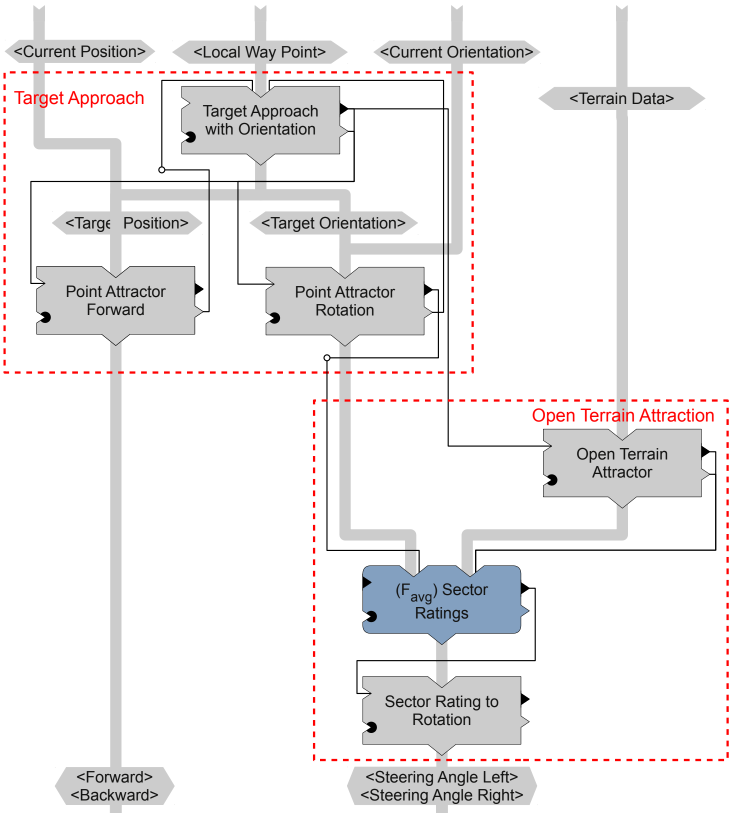


(b) During *Target Approach*, an Intermediate Target residing on a circular path with tangential orientation is computed to reach N177 with the designated orientation.

Figure 8.15: The *Target Approach* consists of a simple pose controller with a straightforward strategy to compute intermediate targets which accounts for the kinematic specifics of non-holonomous vehicles.


Point Attractor
Rotation
Open Terrain
Attraction
Sectors rated as unfavourable
Sectors rated as favourable
Gradual fading of red to green indicates the
continuous nature of the sector-wise rating.

(a) Concept for the *Open Terrain Attraction* support.



(b) *Target Approach* with supporting *Behaviour Open Terrain Attractor*.

Figure 8.16: The *Target Approach Behaviour* network is supported by *Behaviour Open Terrain Attractor*.

(guideline **Take advantage of behaviour signals**). If both *Behaviours* are satisfied, the target was reached with the desired orientation. To take environment information into account during the target approach, *Behaviour Open Terrain Attractor* is provided with a polar *Virtual Sensor* which is mounted at the central front of the vehicle. Obstacles and terrain clutter are rated sector-wise to yield a score for each individual sector. In a second pass, these scores are correlated with the neighbouring scores to yield a filtering effect. *Behaviour Point Attractor Rotation* generates a compatible sector-wise rating for the heading towards the designated target. As no terrain data is considered in this *Behaviour*, the scores are sampled from a Gaussian distribution with the peak oriented in the direction of the target. In Figure 8.16a, sectors rated as unfavourable are highlighted with red colour, sectors rated as favourable are marked green. The continuous nature of the sector-wise rating is indicated by gradual fading between the two colours.

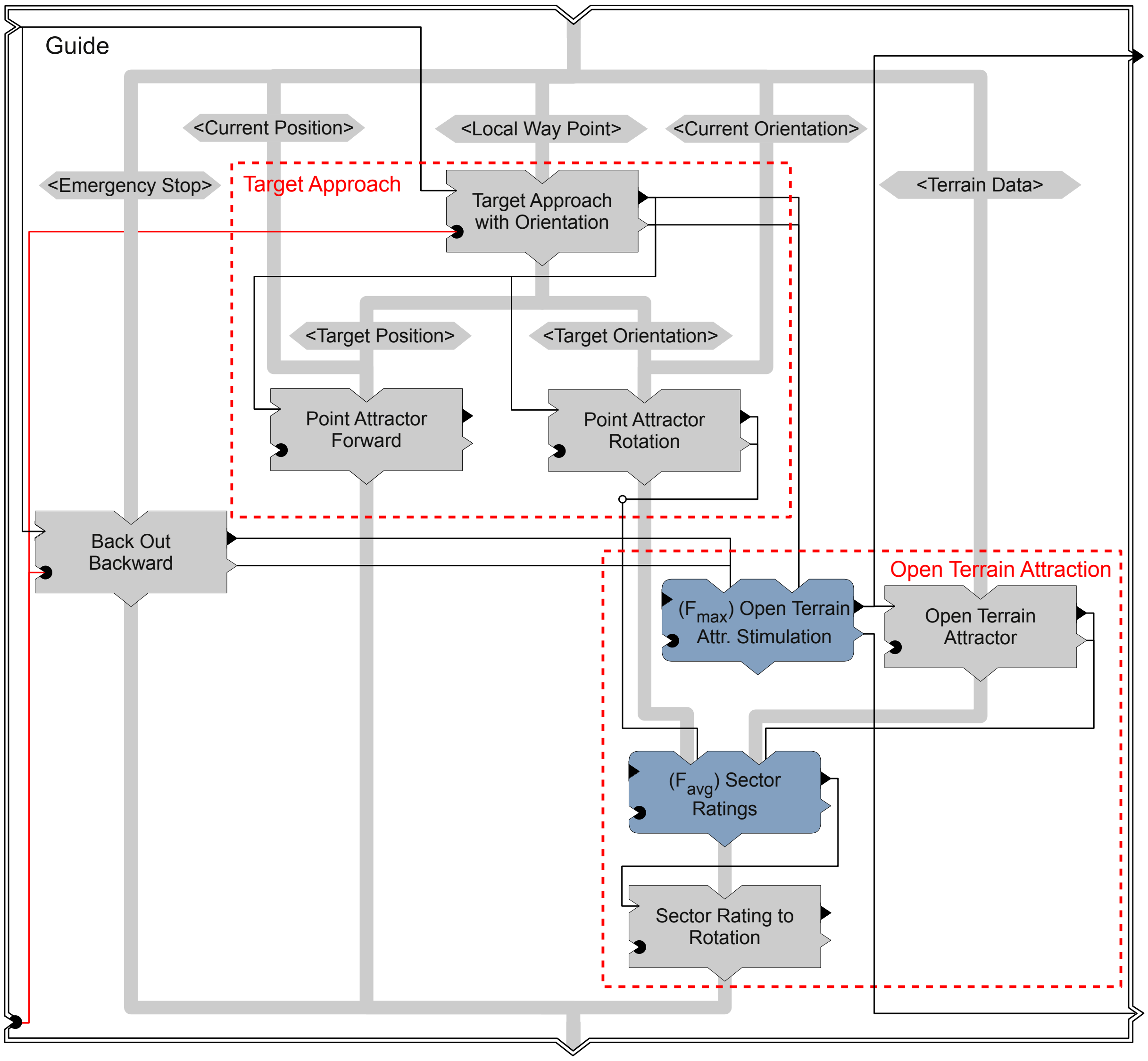


Figure 8.17: The behaviour-based group *Guide*.

Both ratings are then transparently combined with a fusion *Behaviour* in weighted fusion mode. Finally, *Behaviour Sector Rating to Rotation* selects the sector with the best score and propagates the accordant heading to the control system. The property set specifying the placeholders for the control-level aspect of preferring open terrain is defined as follows:

$$openTerrainAttractionPROP = (openTerrainAttraction)$$

Ranking Capabilities

Like most robots, RAVON has a main heading direction as most sensors are mounted to monitor the terrain in front of the robot. The *Target Approach* is therefore also limited to driving forward, as new terrain has to be analysed before the robot passes an area. One of the major objectives for RAVON's navigation system is the goal-directed negotiation of entirely unknown terrain. Navigation in that context, is not limited to dirt roads and open fields. If required, the robot shall also try to push through intense vegetation and narrow passages in the underwood. The limit of what is negotiable is defined by the all-terrain capabilities of the vehicle. When seeking a path through unknown terrain, the robot may get stuck in dead ends or may require to manoeuvre in order to negotiate difficult driving situations. Therefore, ranking capabilities like backing out represent a vital part of the navigation software. The *Behaviour* network realising the mentioned parts of the *Guide* is illustrated in Figure 8.17. As already alluded above, several components (cooperative control, person following, etc.) have been left away as these are not of relevance for this Doctoral Thesis.

8.3 Mediator: Mid-range Navigation

The concept of separating a robot navigation system into two layers called `Pilot` and `Navigator` is widespread in the literature. On the basis of metrically precise local terrain information, the `Pilot` tries to reach the target location indicated by the `Navigator`. Particularly in large natural environments, these loosely coupled navigation techniques will limit the robot's operational range to dirt roads and rather open terrain with a low density of obstacles. As already mentioned above (see *Action Selection / Control Handover* (Challenge 5 on page 25)), the `Pilot` uses a detailed information basis but may not have the farsightedness to take the right decision, while the `Navigator` has the farsightedness but may not have the required information granularity available.

In order to close the gap between the deliberative `Navigator` and the rather reactive `Pilot`, mid-range navigation techniques have to be called into service which employ environment information of appropriate range and precision. These levels of competence mediate steering commands between `Navigator` and `Pilot` and shall therefore be grouped into a third component named `Mediator`. As illustrated in Figure 8.18a, the `Mediator` breaks the sequence of sparse `Global Waypoints` provided by the `Navigator` into a dense sequence of `Local Waypoints` which are passed one after another to the `Pilot`. In that sense, the `Mediator` can be regarded as a *Sequencer* in terminological analogy to hybrid architectures (see Section 2.2). The term `Mediator` was chosen because this component, as well as the *Sequencer* as discussed for instance in [Gat 98], do a lot more than what was briefly described above. This connection was already indicated in the top-level partitioning at the beginning of this chapter (see Figure 8.2 on page 80). In contrast to hybrid architectures, conceptual breaks resulting in semantic gaps are prevented by the homogeneous nature of the behaviour-based control approach followed in this application study. Besides top-down task decomposition, the Mediator also communicates knowledge bottom up to notify the Navigator about possible alternatives to the highlighted route. That way, the Navigator may decide to extend its topological map on the basis of detailed information from lower levels of competence to shorten the distance to the target. The mediating layer of competence on RAVON combines classical planning approaches with structure-based navigation which banks on the concept of *Passages*. In contrast to the rather reactive *Behaviours* of the short-range navigation, the planning facilities deployed in the mid-range navigation require environment information in a condensed form as control-level fusion is not feasible. Following the concept of *Deferred Fusion* (Guideline 6 on page 68), representation-level fusion using the *Abstraction Scheme* outlined in Section 4.3 is used to satisfy this need in a transparent and robust fashion.

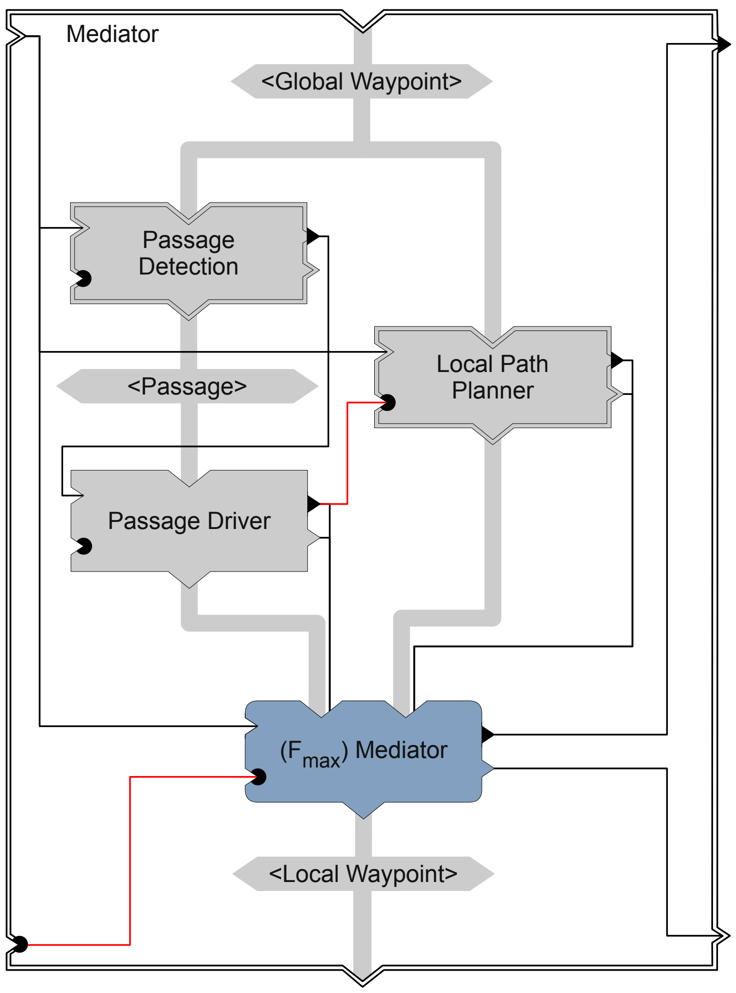
The Local Path Planner uses a grid-based dense local representation of the environment to compute sequences of waypoints using A*-based planning[5]. While path planners always produce a dense sequence of local waypoints between two global waypoints, the Passage Detection component aims at evaluating characteristic structures in the vicinity of the robot to propose alternative routes towards the target waypoint via the Passage Driver. The Passage Driver receives a set of suitable Passages and negotiates with the Navigator whether or not following one of these passages is appropriate. If the passage entry is granted, the Passage Driver inhibits the Local Path Planner and forwards a sequence of local waypoints to the Pilot, which leads the robot into the passage in question. That way, narrow passage ways which may have been overlooked by the Local Path Planner can be used to shorten the distance to target.

The term Passage refers to a general concept which means any sort of favourable terrain which is flanked by less favourable terrain. In the context of this work, this concept is cast into a standardised Structure, which extends the representation scheme outlined in Section 4.1 as a tradeoff between *Structure Simplicity* (Guideline 10) and a *Minimal Set of Base Structures* (Guideline 8). As indicated in Figure 8.18b, the Passage Detection component consists of several detectors which identify Passages using a variety of different approaches. The output of each algorithm is a set of Passages that are suitable with respect to navigation-relevant criteria like passage width, orientation, or heading towards the global waypoint. In the next section, the Local Path Planner will briefly be introduced with a strong focus on the required data basis. After that, the passage detection facility working on virtual sensor probes (VSP) shall be presented in detail to give the reader an idea on how to model Passages in the context of the proposed design schemata and how this concept complements traditional planning (*Competence Overlap* (Guideline 7)).

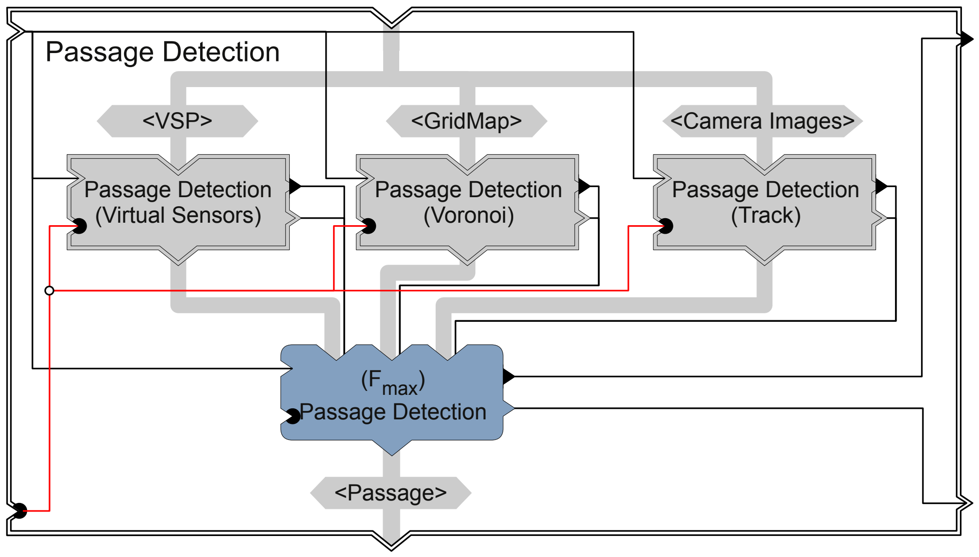
8.3.1 A*-based Path Planning

A*-based path planners require a grid map with occupancy information to compute the cheapest path towards a given target. Common approaches abstract from concrete vehicle kinematics and use region growing to allow for considering the vehicle as a single point at

[5]A*-based in that context denotes that besides A* [Hart 68] also A*-derived planning strategies (e. g. D* algorithm [Stentz 94]) can be deployed.



(a) Top-level partitioning of the `Mediator`.


Passage Detection
<VSP>
<GridMap>
<Camera Images>
Passage Detection (Virtual Sensors)
Passage Detection (Voronoi)
Passage Detection (Track)
(F_max) Passage Detection
<Passage>

(b) *Behaviour Group Passage Detection* subsumes multiple detection algorithms.

Figure 8.18: The `Mediator` combines traditional planning approaches with structure-based navigation techniques.

planning time. That way, the planning procedure becomes computationally quite efficient and independent of the kinematic constraints of the concrete platform. This approach suits the development philosophy for RAVON's control system to prefer simple generalised strategies over specialised algorithms (*Handler Simplicity* (Guideline 15 on page 69)). In fact the `Local Path Planner` deployed on RAVON is identical with the path planner initially developed for the institute's indoor platforms ARTOS and MARVIN [Armbrust 07]. The path planner works on two separate congruent grid representations. One grid map serves as an input to the path planning facility and holds traversability information about the robot's environment. The second grid map is used during the planning procedure to store ratings and states of particular locations.

In natural terrain, binary decisions on traversability do not provide suitable selectivity to distinguish expensive paths which may be passable at low velocities from blocked paths which are definitely impassable. The appropriate granularity can transparently be achieved by translating information into the input grid map with suitable selectivity. To profit from *Property Selectivity* (Guideline 12 on page 69), the path planner is designed to distinguish different classes of traversability per input grid map cell. Furthermore, uncertainty is accounted for by assigning each traversability class a probability value that the property in question actually applies. As already alluded above, the input data is provided in terms of an abstract view on a larger portion of terrain and is generated by the `Semantic Abstraction` via abstract fusion of traversability information from multiple concrete terrain data sources (see Section 4.3 for the theory and Section 10.2 for the aspect-oriented configuration of the information base in question). Larger is to be regarded in the context of mid-range navigation and means that information is aggregated over a certain distance to allow the planner to negotiate difficult structures on a local scale, i. e. without having to employ sophisticated mapping approaches. On RAVON, mid-range navigation deals with an operational range of about 50 m. Structures of larger extents are handled by the topological long-range navigation facilities in cooperation with dead-end detection mechanisms. This reflects the idea of *Representation Locality* (Guideline 3 on page 67) which aims at providing scalable and robust solutions towards navigation. On RAVON, the concrete property set specifying the contents of this structure is defined as follows:

$$fusionPROP = (nonTraversable, moderatelyTraversable, traversable)$$

where

$$nonTraversable(i) = \begin{cases} true, & \text{content element } i \text{ is definitely impassable} \\ false & \text{otherwise} \end{cases}$$

$$moderatelyTraversable(i) = \begin{cases} true, & \text{content element } i \text{ is traversable at lower velocities} \\ false & \text{otherwise} \end{cases}$$

$$traversable(i) = \begin{cases} true, & \text{content element } i \text{ was analysed} \\ & \text{and is passable without risk} \\ false & \text{otherwise} \end{cases}$$

For each cell i, a cost function maps each property p to a configurable value range scaled by the probability $P(p)$. The sum of the resulting values represents the costs for passing over this cell. That way, further nuances of traversability can seamlessly be added by introducing further properties with adjoined cost configurations.

let $n = |\mathit{fusionPROP}|$ the number of properties

let $\mathit{range}(p) = [0, \mathit{max}_p]$ the value range for p

then the cost function for content element i is defined as:

$$\begin{aligned} \mathit{costs}_p(i) &= \mathit{max}_p \cdot P(p) \\ \mathit{costs}(i) &= \sum_{p=0}^{n} \mathit{costs}_p(i) \end{aligned} \tag{8.3}$$

The cost value and further internal assessments are stored in a custom *ContentExtension* named *PlannerContent*. This extension is combined with the common *ContentBase* which manages further flags. During region growing, the neighbours of *nonTraversable* content elements are tagged as *nearObstacle*. These cells are assigned additional costs in dependence of $\mathit{costs}(i)$. On the basis of the individual costs computed for each cell, the cheapest path is determined using an A*-based algorithm. Content elements considered in the planning procedure are tagged as *processed*. This allows for the comparison of different planning algorithms regarding the planning scope. The computed path is highlighted using property *pathPoint*. The turning points of the path represent the sequence of local waypoints which characterise the route of the robot. Content elements containing these waypoints are marked as *relevantPathPoint*. In summary, the property set of the internal path planner grid map is defined as follows:

$$\mathit{plannerInternalPROP} = (\mathit{nearObstacle}, \mathit{processed}, \mathit{pathPoint}, \mathit{relevantPathPoint})$$

where

$$\mathit{nearObstacle}(i) = \begin{cases} \mathit{true}, & \text{content element } i \text{ is a neighbour} \\ & \text{of a non-traversable content element} \\ \mathit{false} & \text{otherwise} \end{cases}$$

$$\mathit{processed}(i) = \begin{cases} \mathit{true}, & \text{content element } i \text{ was considered during path planning} \\ \mathit{false} & \text{otherwise} \end{cases}$$

$$\mathit{pathPoint}(i) = \begin{cases} \mathit{true}, & \text{content element } i \text{ is part of the computed path} \\ \mathit{false} & \text{otherwise} \end{cases}$$

$$\mathit{relevantPathPoint}(i) = \begin{cases} \mathit{true}, & \text{content element } i \text{ is a local waypoint} \\ & \text{in the computed path} \\ \mathit{false} & \text{otherwise} \end{cases}$$

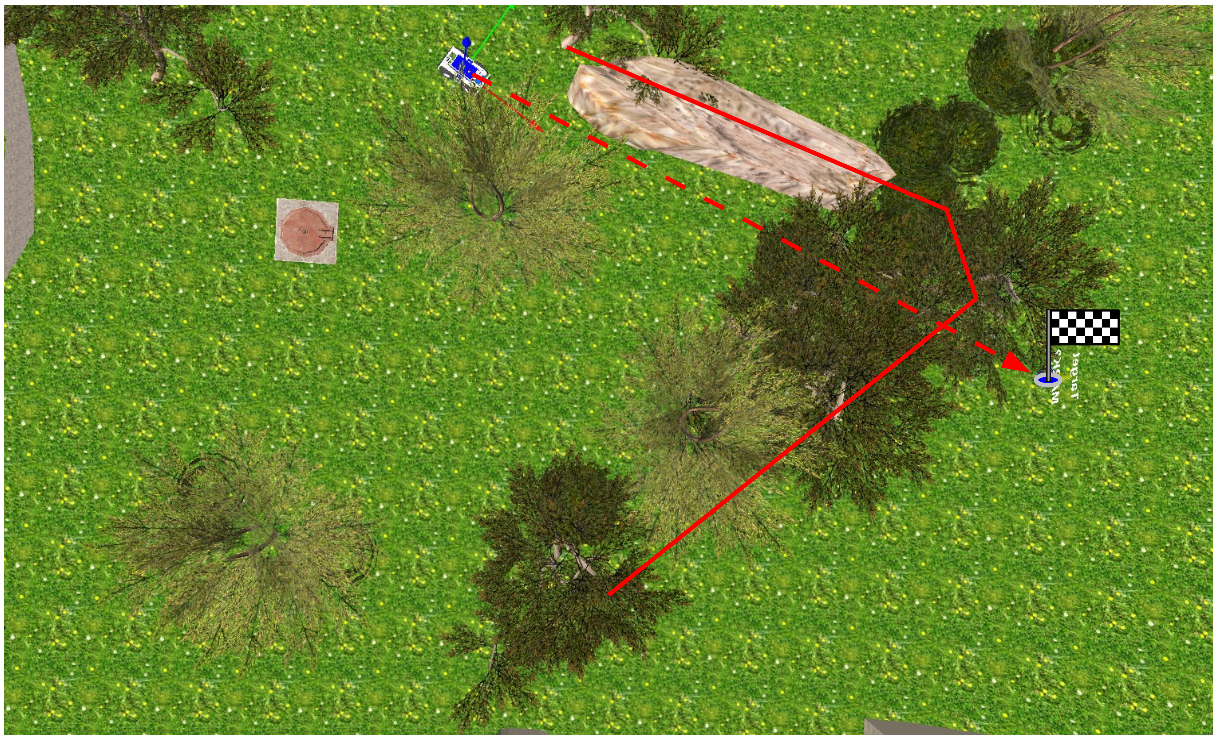


Figure 8.19: Dead-end scenario which requires the robot to temporarily move away from the target position in order to finally reach its destination.

An elastic bands approach similar to [Quinlan 93] is deployed to smoothen the path in a further processing step. The relevant path points are connected by a virtual elastic band which is optimised in an iterative fashion. Contraction forces simulating the tension in the band remove indentations from the path while repelling forces push the band away from obstacles. According to *Handler Simplicity* (Guideline 15 on page 69), path planning, elastic bands computation, and following the computed trajectory are realised as separate *Behaviours* which are encapsulated by the behaviour-based group `Local Path Planner`.

Figure 8.19 shows a simulated scenario in which the robot has to negotiate a dead-end in order to reach the target location. At the beginning of the experiment, the robot only knows its own pose and the location of the target. No additional knowledge about the environment is provided such that the vehicle has to explore the terrain in order to find a path to the destination. On the move, the aggregated short-term memories are cast into an abstract view, namely the fused traversability grid map introduced above, which the path planner uses to determine feasible paths. As the input grid map and the internal path planner grid map are congruent, the common display facilities can be used to superimpose the latter onto the former to keep track of planning process and results. The optimised path computed by the elastic bands algorithm is highlighted by the centres of blue circles which indicate the corridor width at the locations in question. After several attempts to approach the target more or less directly, enough information about the environment has been collected such that the path planner is enabled to compute a suitable path. The final path that leads to the target area is illustrated in Figure 8.20. The way covered by the robot until it has reached this point is indicated by the dashed blue line. Figure 8.21 shows the complete simultaneous exploration and path planning procedure at different

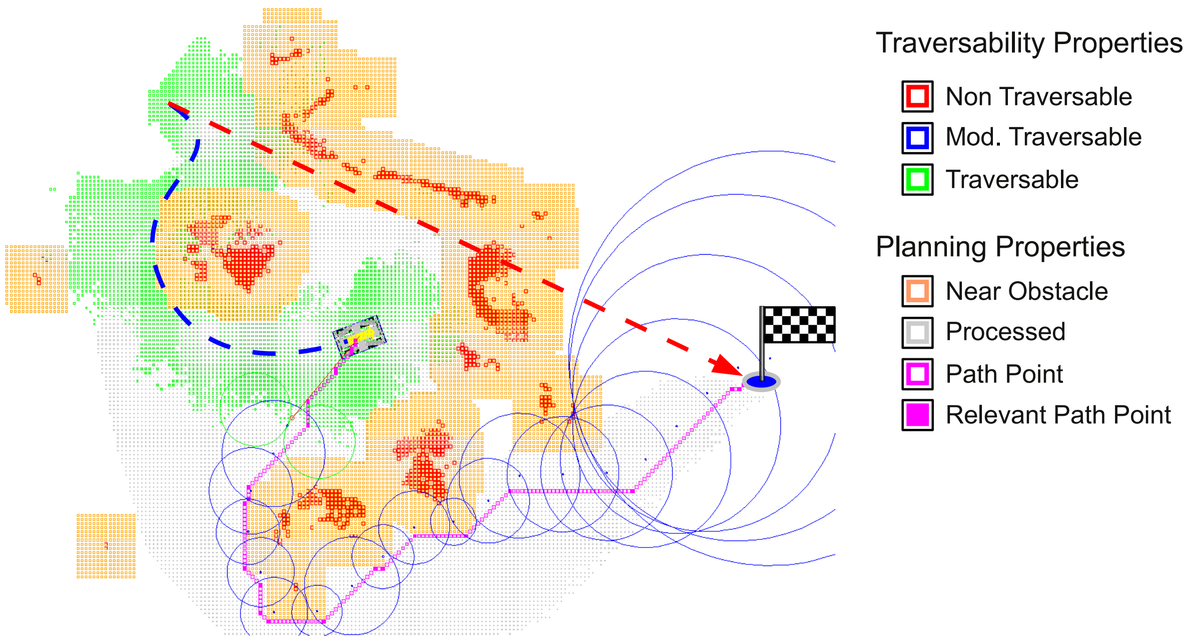


Figure 8.20: The path planner successfully resolves the dead-end scenario via exploration.

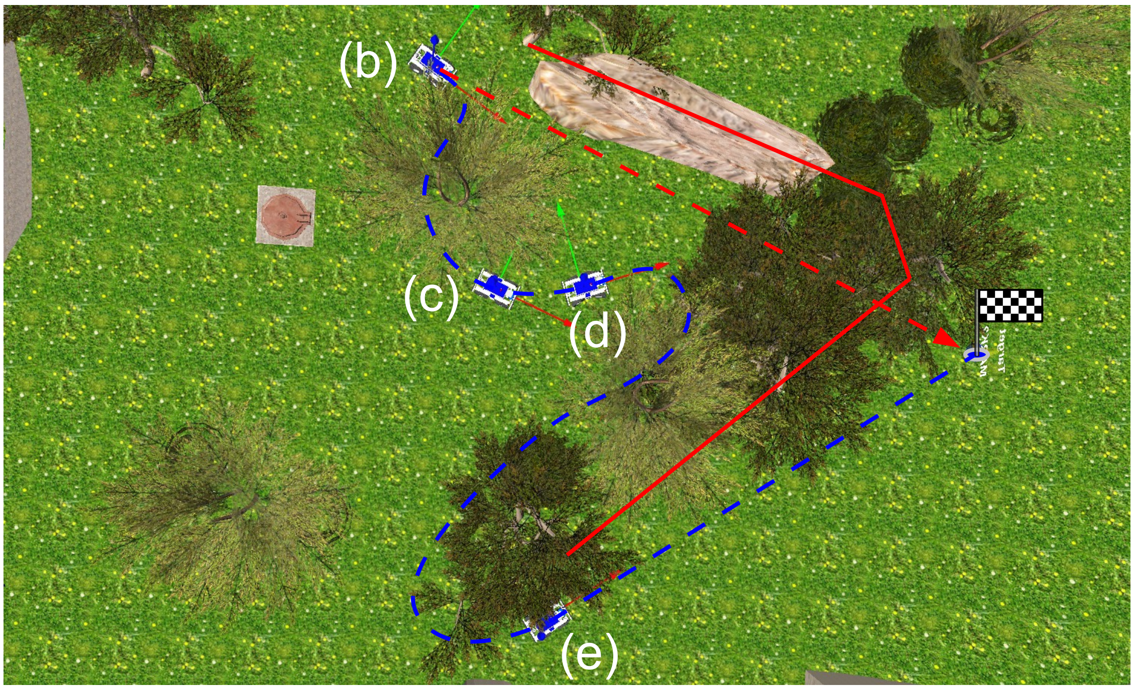
stages of the experiment.

The deliberative A*-based planner allows RAVON to master complex situations which are beyond the capabilities of the rather reactive approaches introduced before. In particular dead-end structures and winding tracks on the way towards the target area can be negotiated more successfully. The simplification to use region growing and regard the robot as a single point comes with the disadvantage that the kinematics of the vehicle is not considered by the planner. The region growing must therefore be configured rather pessimistic to leave enough room for the vehicle to pass through. For that reason, narrow passages may not be considered by the planner at all or they may not be negotiable because the passage entry is approached in an unsuitable way. In order to compensate for this weakness and to extend RAVON's off-road capabilities, structure-based navigation techniques which are based on the concept of *Passages* have been developed. In the following, a passage detection approach using *Virtual Sensor Probes* shall be discussed in more detail.

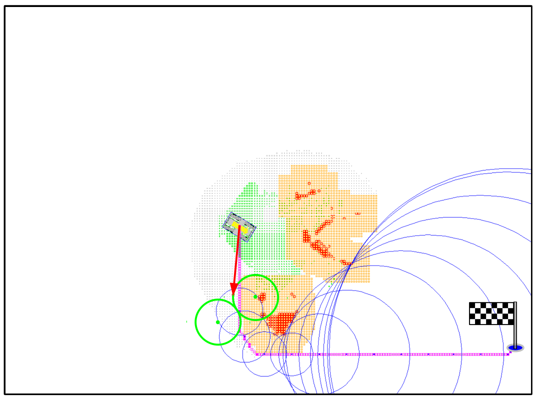
8.3.2 Using Passages to Improve Mid-range Navigation

With the navigation competences introduced so far, RAVON is drawn towards a target location by the long-range navigation while avoiding obstacles with the rather reactive *Behaviours* of the short-range navigation. The `Local Path Planner` acts as a mediating component between these partially conflicting strategies to keep the robot away from indentations, dead-ends, and narrow openings between obstacles.

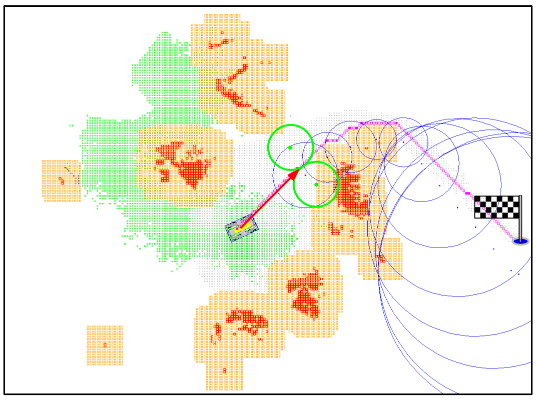
As already mentioned above, the `Local Path Planner` keeps the robot out of trouble but may fail to approach openings in a suitable way to pass through. The dead-end in the scenario introduced above may actually be escaped through a group of loosely standing



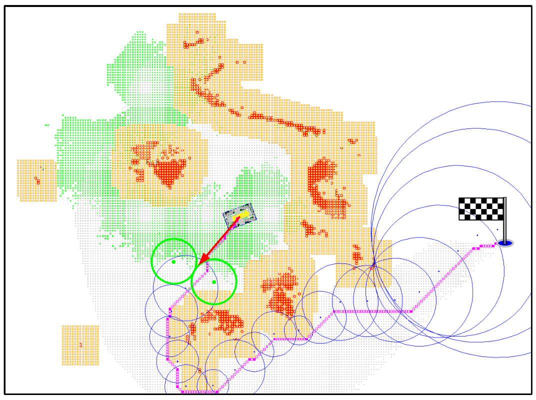
(a) Overview of the dead-end scenario. The dead-end is highlighted in red, the target vector in dashed red, and the route taken by the robot in dashed blue.



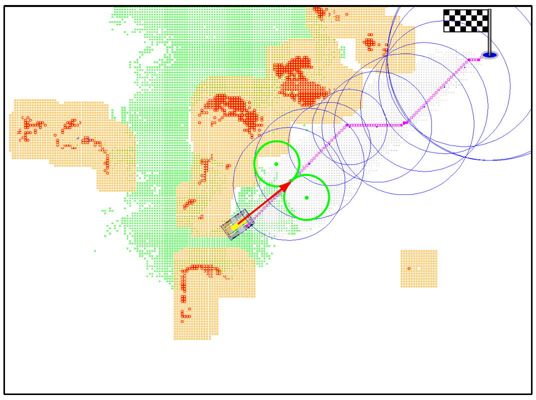
(b) Between the tree and the rock is not enough space, so a first detour is planned.



(c) Initially the far side of the dead-end is out of reach of the sensors.



(d) As the exploration unveils the dead-end structure, an alternative route is planned.



(e) Finally, the robot has negotiated the dead-end.

Figure 8.21: In this dead-end scenario (a), the robot has to retreat from the target position in order to reach it. Along the route, the planning state in four different stages are shown (Illustrations (b) through (e)).

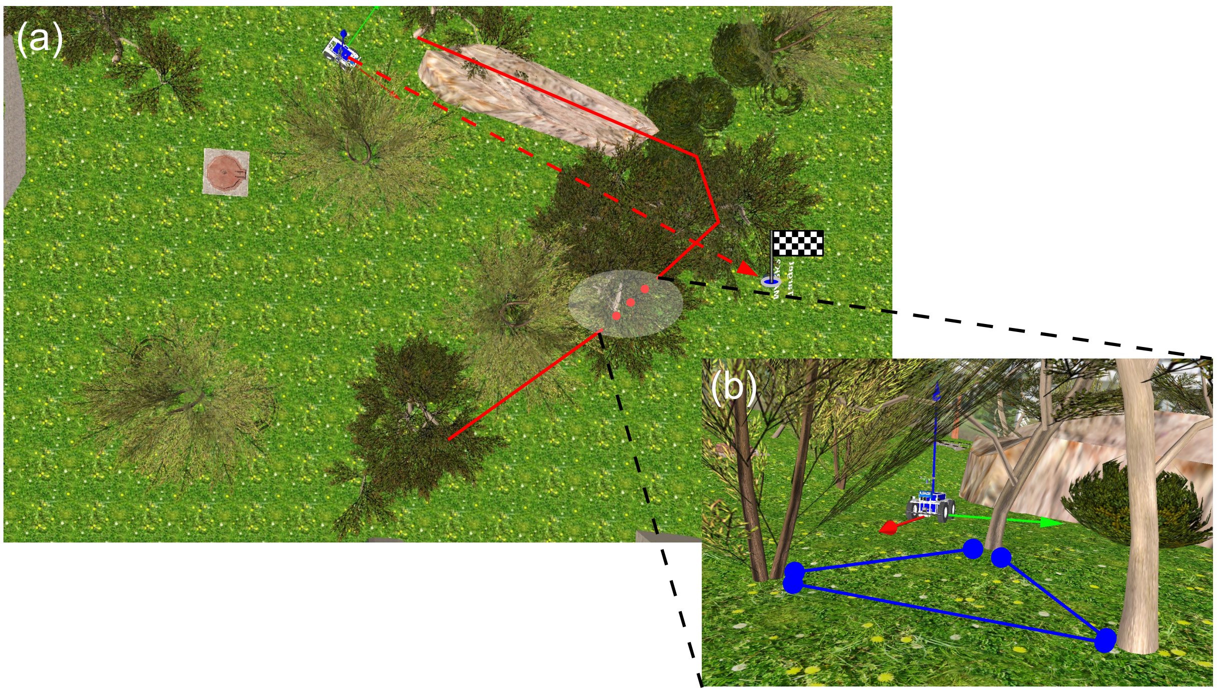


Figure 8.22: In the scenario introduced in the previous section (a), the path planner successfully negotiated a dead-end. At one side however, loosely standing trees would allow the robot to escape the dead-end on a shorter path (b). The openings that would be large enough for the robot to pass through have been marked with blue lines.

trees (see Figure 8.22). In stage (c) illustrated in Figure 8.21, the region growing closes the opening between the trees and the `Local Path Planner` computes a route around the group of trees. The robot furthermore approaches the opening in a sharp angle such that the vehicle would probably fail to enter the narrow track anyhow. This may result in detours or mission failure, even in situations where the simple combination of far-range navigation and short-range navigation may have succeeded. The fine line between what is negotiable and what conformations are too risky to pass through is very difficult to decide. In order to improve mid-range navigation, additional strategies shall be called into service to close the gap between classical path planning and reactive obstacle avoidance.

In the literature, different approaches have been published which propose advanced mid-range navigation techniques [Wooden 07, Ranganathan 03]. An approach for detecting narrow passages in indoor environments is described in [Schröter 05]. The author used the polar data of a laser scanner to identify and assess passages and combined it with 3D rectangular objects reconstructed from the images of a stereo vision system to identify doors. The algorithm used for processing the laser data resembles the one presented in the following. However, it only processes the data of one polar sensor and not of several (virtual) sensors. Evidently, using a stereo system in such a way to detect passages will not work in unstructured off-road environments. Approaches for keeping the robot from leaving the road or path include detecting curbs using a light-stripe scanner [Thorpe 03], detecting lanes using edge extraction from images, or road detection using a combination of LADAR data and colour information [Hong 02]. The work described in [Lieb 05] uses the assumption that the vehicle is situated on the road to form templates of the road's appearance and from these and current images calculates an estimate of the road's curvature. [Alon 06] describes a system that uses two different path-finding algorithms in parallel and uses the output of the one with the highest confidence.

All of the approaches briefly discussed above are tailored to detecting a path in rather structured environments which renders the transfer to the domain of this application complicated. The schematisation of abstraction and representation can be deployed to reduce the complexity of natural environments to the problem-relevant properties, such that structure detection becomes feasible. The aim of this approach is to exploit *Passages* which lead towards the designated target location to shorten the distance to travel and time to target [Armbrust 09b]. The assessment procedure features two major steps, namely the actual detection of *Passages* and the evaluation of navigation-relevant *Passage* parameters (e. g. width, length, orientation).

Passage Detection

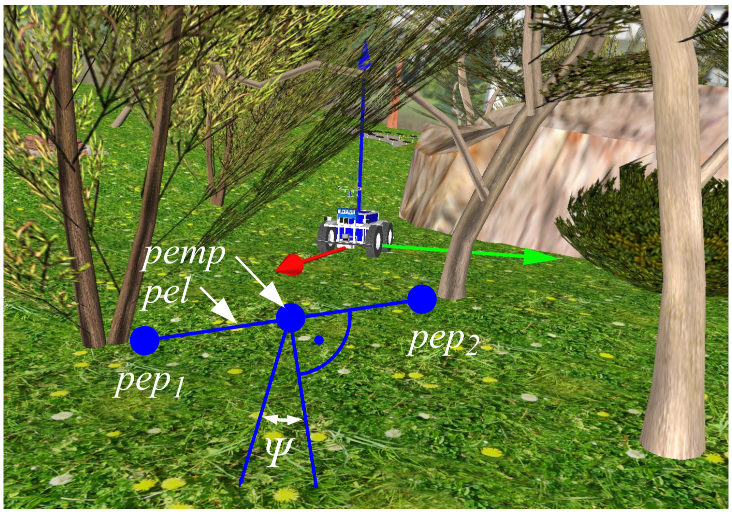


Figure 8.23: A *Passage* is defined by three points, the *Passage Entry MidPoint* (*pemp*) and the flanking *Passage Entry Points* (*pep*$_1$ and *pep*$_2$). All three points reside on the *Passage Entry Line* (*pel*). The *Passage Orientation* ψ is given relative to the normal of the *pel*.

In this section, a novel approach towards passage detection in unstructured environments will be presented. Before going into detail with the algorithm itself, the representational foundations shall be illuminated. In the context of this work, the term *Passage* shall refer to a patch of favourable terrain that is flanked by less favourable terrain. The set of base structures specified in the proposed representation scheme does not feature an optimal structure for modelling *Passages* (see Section 4.1). Therefore, the tradeoff between a *Minimal Set of Base Structures* (Guideline 8 on page 69) and *Structure Simplicity* (Guideline 10 on page 69) has to be rated. As already indicated at the beginning of this section, *Passages* shall be used by different detection strategies as a common basis to communicate results. This requires a compact structure which is tailored to the needs of the detection algorithms. *Structure Simplicity* (Guideline 10 on page 69) therefore has to be rated as the more important factor. In consequence, the abstract concept of a *Passage* shall be modelled in terms of an additional base structure in the representation scheme. Formally, a *Passage* is defined by the *Passage Entry MidPoint* (*pemp*), which can be regarded as the origin of an opening, and the flanking *Passage Entry Points* (*pep*$_1$ and *pep*$_2$) which mark the boundaries of the opening. All three points reside on the *Passage Entry Line* (*pel*) defining the width of the passage. The passage orientation angle ψ is represented relative to the normal of the *pel*. In Figure 8.23, one of the passages from the example scenario in Figure 8.22 (a) is annotated respectively.

$$Passage := (pemp, pep_1, pep_2, \psi) \tag{8.4}$$

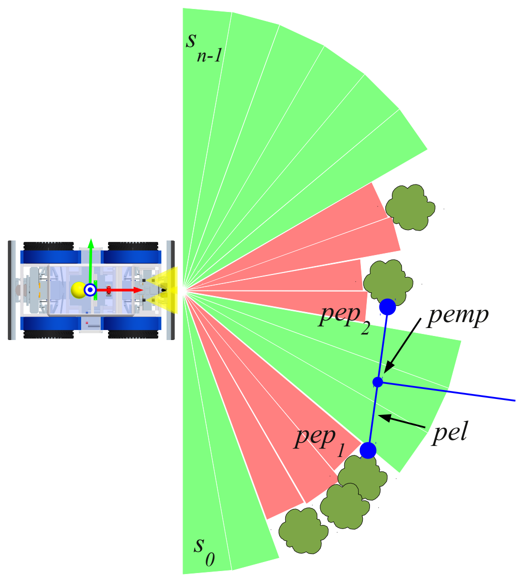
where

$pemp \in \mathbb{R}^2$	the passage entry midpoint
$pep_1 \in \mathbb{R}^2$	the first passage entry point
$pep_2 \in \mathbb{R}^2$	the second passage entry point
$\psi \in \mathbb{R}$	the passage orientation angle

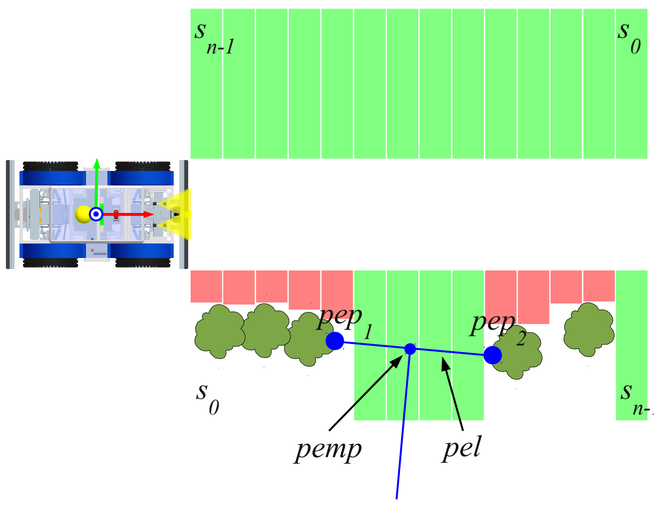
Furthermore, *Passages* contain one content element of type *ContentBase* which defines several properties for annotation and assessment purposes. These properties will be introduced step-by-step as the assessment procedures are evolved. The particular subunits of the passage management system communicate solely on the basis of the *Passage* structure. The `Passage Detection` passes a set of *Passages* to the `Passage Evaluation`, which filters the *Passages* according to several suitability criteria before passing relevant *Passages* to the `Passage Driver` (see Figure 8.18a on page 108).

In addition to the representation for communication, the *Passage Detection* requires information about the environment. As already alluded above, the abstraction and representation scheme proposed in this work shall be used to create tailored *Views* on the world to simplify the detection and evaluation of passages. As a mid-range navigation facility, *Passage Detection* requires a preferably complete view of the robot's environment. Therefore, the fused traversability grid map already deployed for path planning may serve as initial source of information. In fact, the detection algorithm presented in this work does not work on the traversability grid map directly. In order to simplify detection and evaluation procedures, several *Virtual Sensors* are specified which render suitable *Views* from the fused data basis. That way, the passage detection and evaluation algorithms do not have to access the complex grid map, but may operate on the way simpler *Views* supporting *Handler Simplicity* (Guideline 15 on page 69) via *Structure Simplicity* (Guideline 10 on page 69).

As illustrated in Figure 8.24a, one polar *Virtual Sensor* is deployed for the detection of *Passages* in front of the robot. The radial nature of this *View* allows for the detection of *Passages* of varying orientation. For the detection of lateral *Passages*, two further Cartesian *Views* are specified to monitor the terrain to either side of the robot's current heading (Figure 8.24b). In each of these *Views*, the passage detection algorithm traverses all sectors s_i and compares the distances d_i to non-traversable structures (with $i \in [0, n)$ and $i \in \mathbb{N}$). If d_{j+1} is greater than d_j by at least the threshold Δd, then sector j is considered as candidate for containing the first *Passage Entry Point* pep_1. Let sector k with $k \geq j+1$ the first sector, where d_k is smaller than d_{j+1} by at least the threshold Δd. Then a *Passage* was detected with sector j containing the first *Passage Entry Point* pep_1 and sector k containing the second *Passage Entry Point* pep_2. The sector representatives (see Section 3) can directly be assigned the *Passage Entry Points*: $pep_1 = s_j.\text{getRepresentative}(nonTraversable)$ and $pep_2 = s_k.\text{getRepresentative}(nonTraversable)$. All sectors between the *Passage Entry*



(a)



(b)

Figure 8.24: *Virtual Sensors* deployed for passage detection.

Points are considered to lie within the passage and are subject to further evaluation. The new passage is added to a list and the search goes on with the remaining sectors. Algorithm 8.1 presents the procedure in pseudo code. For each passage, the *Passage Entry MidPoint pemp*, which is defined as the point lying in the middle between pep_1 and pep_2, is calculated. The *pemp* is the characteristic point or origin of a *Passage* and, as will be described below, is important for approaching the *Passage*. Note that the slim interface to environment information via *Views* allows to formulate a straightforward algorithm for passage detection. The semantic and structural filtering applied by the `Semantic Abstraction` layer (see Section 3.3.1 on page 32) is the key to this reduction of complexity. The proposed passage detection algorithm has a very local scope such that each *View* can be treated separately allowing for *Functional Partitioning* (Guideline 16 on page 70).

In order to keep track of *Passages* over time, the *ProbabilisticContent* extension is deployed (*Content Reuse* (Guideline 1 on page 67)). The correlation of *Passages* is achieved by matching the *pemp* of newly detected *Passages* with those in the list of *Passages*. A pair of passages are regarded as similar if the distance between the two *pemp* is below a certain threshold. If a similar *Passage* is found in the list, it is updated with data from the *Passage* just detected. In case no similar *Passage* is found in the list, the new passage is added to the list. Passages that have not been detected again for a certain amount of time are discarded. A *Passage* update comprises the three characteristic points (*pemp*, pep_1, and pep_2) as well as the probability that this *Passage* really exists. For that purpose, a property *persistent* is defined, which is managed by the *ContentBase* element stored in the *Passage* structure.

Passage Evaluation

The *Passage Evaluation* procedure operates on the *persistent Passages* only. In a first step, the *basic requirements* passage width, length, and orientation suitability with respect

Algorithm 8.1: Detecting Passages

```
searching_for = cFIRST_PEP; // the status of the search process
index_first_pep = 0; // the index of $pep_1$
index_point_in_passage = 0; // the index of a point within the passage
for (i = 0; i < $n-1$; i++) do
  if (searching_for = cFIRST_PEP) then
    if ($d[i] - d$[index_first_pep] $< \Delta d$) then
      index_first_pep = i;
    else
      searching_for = cSECOND_PEP; index_point_in_passage = i;
  else // currently searching for $pep_2$
    if ($d$[index_point_in_passage] $- d[i] \geq \Delta d$) then
      passages.add(new Passage ($r_{\text{index\_first\_pep}}, r_i$));
      index_first_pep = i;
      searching_for = cFIRST_PEP;
    else
      // nothing to be done here
```

to the current global waypoint are checked. To model these attributes, the property set for passage detection is extended with respective properties. A passage which fulfils all requirements is further assigned the attribute *suitable* which is also modelled as a property yielding the following specification:

$$passagePROP = (persistent, suitableWidth, suitableLength, suitableOrientation, suitable)$$

where

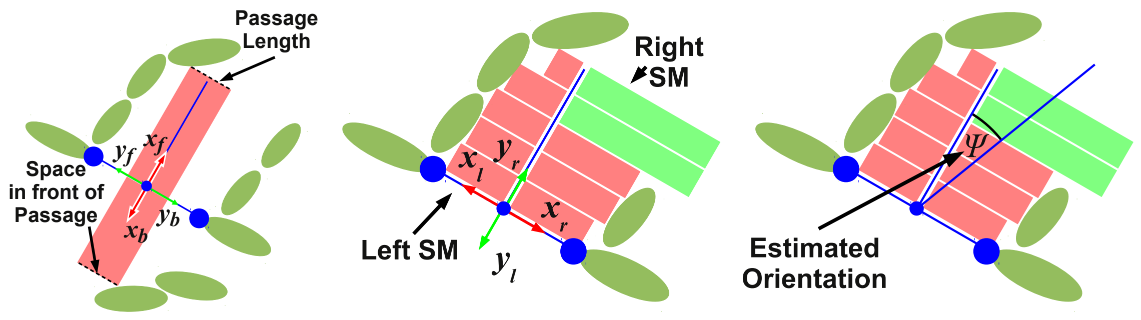
$$persistent(i) = \begin{cases} true, & \text{if } Passage\ i \text{ is constantly detected} \\ & \text{over several sensor readings} \\ false & \text{otherwise} \end{cases}$$

$$suitableWidth(i) = \begin{cases} true, & \text{if } Passage\ i \text{ is wide enough} \\ & \text{for the robot to pass through} \\ false & \text{otherwise} \end{cases}$$

$$suitableLength(i) = \begin{cases} true, & \text{if } Passage\ i \text{ has a certain length} \\ & \text{such that traversability is likely} \\ false & \text{otherwise} \end{cases}$$

$$suitableOrientation(i) = \begin{cases} true, & \text{if } Passage\ i \text{ is oriented} \\ & \text{towards the current global waypoint} \\ false & \text{otherwise} \end{cases}$$

$$suitable(i) = \begin{cases} true, & persistent\ (i) \wedge suitableWidth\ (i) \wedge \\ & suitableLength\ (i) \wedge suitableOrientation\ (i) \\ false & \text{otherwise} \end{cases}$$



(a) Length estimation **(b)** Orientation evaluation **(c)** Estimated orientation

Figure 8.25: For length (a) and orientation (b), (c) estimation, several VSP are specified to obtain specific information about the passage.

The determination of the three requirements outlined above is straightforward. The width of a passage is computed as the distance between pep_1 and pep_2. If a *Passage* entry is too narrow for the robot, it is apparently irrelevant for navigation.

As *Passages* shall be used to assist the navigation system in driving the robot to the target location, a suitable passage should lie in the direction of the target with respect to the robot. This is checked by comparing the orientation of two lines—one going from the robot to the target and one going from the robot to the *pemp*. Furthermore, a suitable passage has to point approximately into the direction of the target. This is checked by first calculating the orientation of a line that starts at the *pemp* and is perpendicular to the *pel*. This orientation is then compared to the one of the line defined by the *pemp* and the target. If the difference between the two orientations is below a certain threshold, the passage is considered to be well-oriented.

In the next processing step, more precise passage length and orientation estimates are computed using *Virtual Sensor Probes* (VSP – see Section 4.4.3 on page 62 for details). One VSP represented by a Cartesian *View* (defined by axes (x_f, y_f)) consisting of only one sector is used to "look" forward into the *Passage* and measure the distance to the closest obstacle. As illustrated in Figure 8.25a, this value is used to estimate the *Passage* length. A second VSP monitors the opposite direction in order to determine whether there is enough free space in front of the passage for the robot to enter (defined by axes (x_b, y_b)). Using VSP instead of accessing the grid map directly facilitates data access and spares the evaluation component more complex data processing. In a addition to *Structure Simplicity* (Guideline 10 on page 69), *Structural Partitioning* (Guideline 9 on page 69) is applied at this point to support *Handler Simplicity* (Guideline 15 on page 69).

It is advisable to navigate the robot in a way that it reaches the *pemp* with approximately the orientation of the *Passage* as this facilitates turning into the opening. The orientation estimation described above does not take into account information about the presence of obstacles behind the *Passage* entry. A more precise estimate is calculated for the passage that best satisfies the above criteria, the *relevant passage*. Two Cartesian *Views* are used to monitor both sides of the *Passage* (see Figure 8.25b). The origin of both VSP is the *pemp* with the x-axes (x_l, x_r) residing on the *pel*. Each sector stores information about the closest obstacle in the covered area, i. e. it contains a local estimate of how far away the border of the *Passage* is. For each sector, the angle between the *pel* normal starting

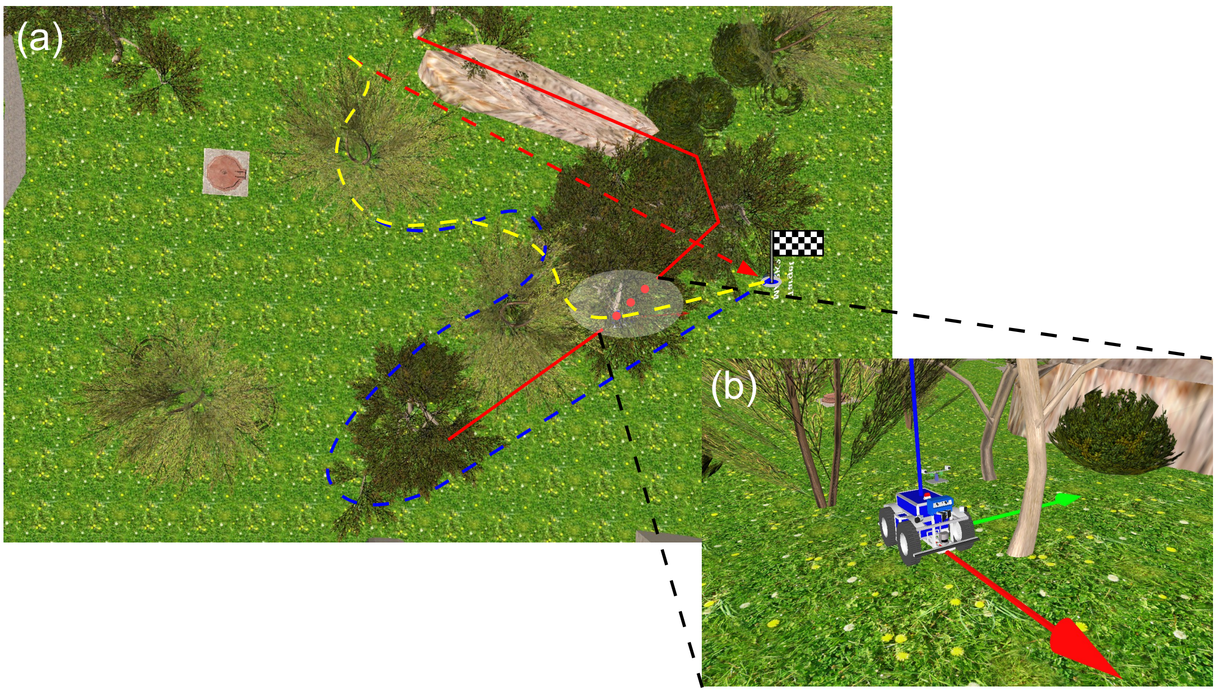


Figure 8.26: The robot chooses the shorter path towards the target (a) by using the passage detected between the trees (b). The path travelled with *Passage Detection* support (yellow dashed line) is way shorter than the one chosen by the *Local Path Planner* (blue dashed line).

in the *pemp* and the line segment connecting the *pemp* with the sector representative is computed. The *Passage* orientation is defined as the arithmetic mean of these angles, such that its axis is pushed away from obstacles (see Figure 8.25c). Note that this is only a rough estimate, but it can be calculated using the existing mechanisms, while other methods need more complex, specialised algorithms.

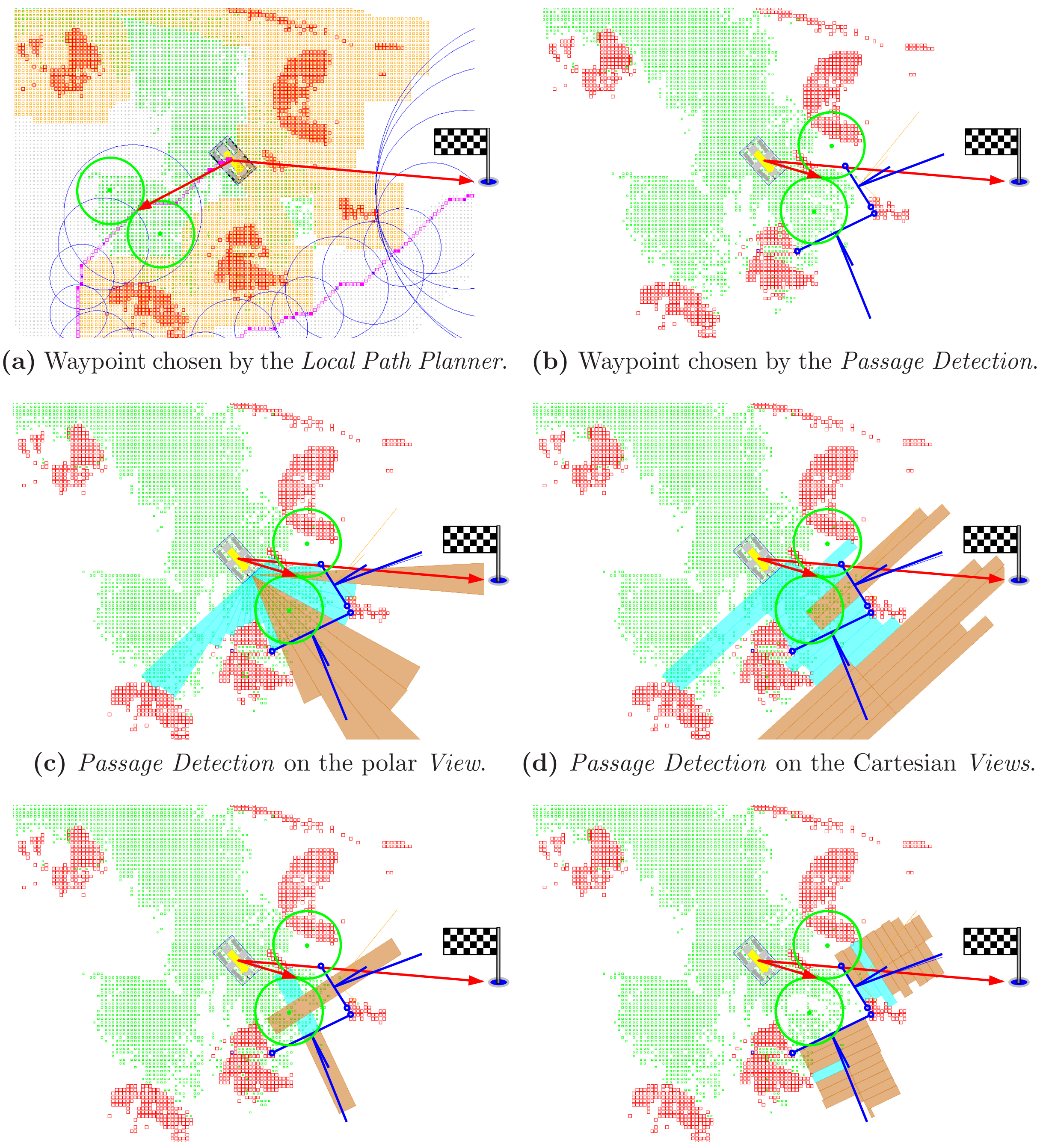
Whenever a relevant passage is detected, the *Passage Driver* negotiates with the `Navigator` whether to follow this passage or not (*Competence Overlap* (Guideline 7 on page 68)). In case the use of the *Passage* is granted by the `Navigator`, the *Passage Driver* gets active and sends the coordinates of the *pemp* to the `Pilot`, where the *Target Approach Behaviours* take over in order to drive the robot to the given target coordinates. Note that the *Target Approach Behaviours* are also used by the long-range navigation system and the *Local Path Planner*. When a *Passage* is detected, the *Passage Driver* overwrites the target pose provided by higher-level navigation facilities via inhibition of the respective *Behaviours*.

The *Obstacle Avoidance Behaviours* within the short-range navigation usually try to keep the robot far away from obstacles (see Section 8.2.1). For negotiating narrow passages, the rotational *Keep Distance* and *Evasion Behaviours* are therefore inhibited as well. The sideward and deceleration control strands are not influenced to guarantee fundamental safety of the vehicle. Furthermore, the *Emergency Stop Behaviours* which are never inhibited bring the robot to an abrupt halt in case the vehicle gets too close to obstacles.

Figure 8.27 resumes the scenario described in Figure 8.22 on page 114 at the beginning of this section. The documented scene is taken from a repetition of the dead-end experiment carried out for evaluating the *Local Path Planner* in Section 8.3.1. This time, the *Local*

Path Planner is complemented with *Passage Detection.* Figure 8.27 shows about the same state as stage (d) illustrated in Figure 8.21. To verify this, Figure 8.27a shows the local waypoint that would be chosen by the *Local Path Planner.* At the same time, the *Passage Detection* found *Passages* in the polar and the Cartesian *Views* (see Figures 8.27c and 8.27d). Evaluation of *Passage* length (Figure 8.27e) and orientation (Figure 8.27f) yields two *suitable Passages* towards the target location, which are highlighted in blue (Figure 8.27b).

In the experiment at hand, the robot took the smaller passage to its left leading directly to the target and completed its mission shortly after this decision. Figure 8.26 shows the different paths chosen by both configurations. The path travelled with activated *Passage Detection* is indicated by the yellow dashed line. In contrast to the path chosen in the experiment where only the *Local Path Planner* was active (blue dashed line), the distance travelled and time to target were tremendously reduced. At this point this qualitative result shall suffice for motivating the applicability of the proposed approach. A profound statistical analysis of the performance gain will be provided in Section 11.1.



(a) Waypoint chosen by the *Local Path Planner.* **(b)** Waypoint chosen by the *Passage Detection.*

(c) *Passage Detection* on the polar *View.* **(d)** *Passage Detection* on the Cartesian *Views.*

(e) *Passage* length estimation. **(f)** *Passage* orientation estimation.

Figure 8.27: In the dead-end scenario of Section 8.3.1 (a), the *Passage Detection* actually finds a traversable path through a group of loosely standing trees (b). The detection of the *Passages* (c, d) as well as the evaluation of *Passage* length (e) and orientation (f) are illustrated in the respective subfigures. The local waypoint passed to the `Pilot` is visualised by the *Target Approach* facility (see Figure 8.15 on page 104).

8.4 Navigator: Long-range Navigation

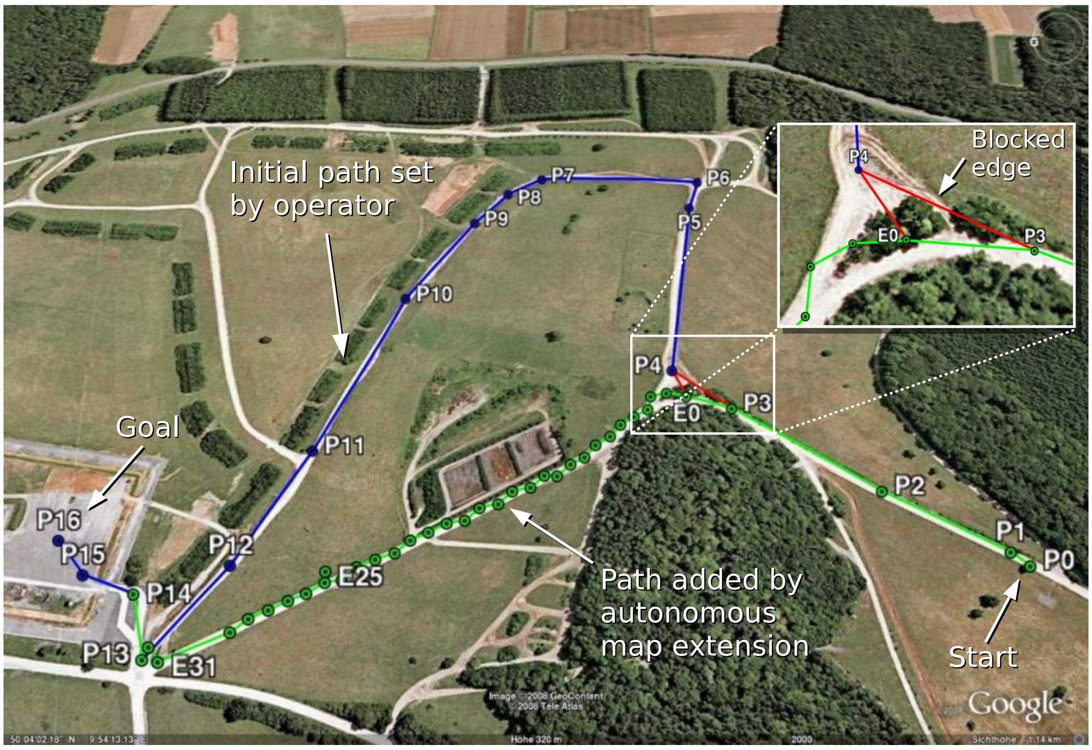


Figure 8.28: Autonomous map extension during the European Land Robot Trial 2008 [Braun 09b]. This figure was created using GoogleEarth®.

Localisation quality in forest areas cannot be assured to be accurate enough for established robot navigation approaches based on dense global maps and an A* or related planning algorithms. Therefore, the mid-range navigation capabilities introduced in the previous section shall be complemented with further competences which are suitable for long-range navigation. Long-range navigation in this context addresses mission extents in the dimension of kilometres. Terrain information in this scope has to be assumed imprecise, incomplete, or even wrong, which are unconsidered issues in classical approaches. Furthermore, mission planning should be as simple as possible to minimise training efforts for operators. These requirements lead to a long-range navigation strategy on the basis of coarse topological maps which can conveniently be introduced by the operator using a Geographic Information System (GIS). Note that for this approach, the global localisation precision limits the applicability to distances larger than 20 m to virtually unlimited distances. Therefore, the proposed combination with the mid-range navigation facilities represents a powerful example for *Competence Overlapping* (Guideline 7 on page 68).

Once provided with a map, the robot can be commanded to approach a target location following a sequence of critical waypoints. Paths between critical waypoints are computed using the Dijkstra algorithm [Dijkstra 59] and a multi-dimensional cost measure which is learnt on the move. The basis for the learning procedure are abstract situation assessments in terms of the *Behaviour* signals provided by the obstacle avoidance *Behaviours* of the short-range navigation (**Take advantage of behaviour signals** [Proetzsch 10a] p. 104). In case a critical waypoint is unreachable for the robot because all existing paths are blocked (this may happen due to outdated or inaccurate remote imagery of the GIS), the topological map is automatically extended with alternative routes on the basis of local terrain information. The costs for traversed paths is annotated to the edges and waypoints are enriched with local terrain information to optimise the planning in future missions in the same area. The topological map is designed according to the representation scheme outlined in in Section 4.1. That way, structure and contents can be tailored using the generic semantic abstraction facilities (see Section 4.2).

Figure 8.28 illustrates the long-range navigation capabilities of RAVON by example of the European Land Robot Trial (see Appendix B.5) scenario *Reconnaissance and Surveillance* [Braun 09b]. The path initially set by the operator was followed until deep lane grooves, not visible in the satellite image, prevented the robot to proceed on the predefined route. On the basis of experiences made up to this point, RAVON incrementally planned an alternative path using local terrain information and finally reached the target area. Further details on the long-range navigation facilities can be found in [Braun 09a].

8.5 Navigational Competences in Summary

In this section, the navigational competences presented above shall be arranged into the big picture of the complete control system. Figure 8.29 depicts a high-level overview of RAVON's control concept, which is roughly be divided into four layers. The lowest layer subsumes the mechatronics of the platform described above and the software that serves as an interface for the control system. In this overview, localisation and sensor data preprocessing are also regarded as part of this layer. On top of this interfacing layer, the *Short-range Navigation* introduced in Section 8.2 realises the collision-free target approach employing rather reactive repulsion-based and attraction-based *Behaviours*. Abstract views on the environment which are directly rendered from the particular short-term memories are the primary information source at this point to reduce latency. Besides these rather rich representations, further sensor information – e. g. bumper events, pitch and roll angles – are directly transferred to particular components for fundamental safeguarding. The next higher level of competence contains the mid-range navigation facilities which have been introduced in Section 8.3. These planning facilities operate on views of the world which are generated from a condensed information basis, which is yielded by abstract fusion of the short-term memories. The resulting *Fusion Map* further serves as input for an A*-based local path planner. The highest level of navigational competence contains the global navigation system which uses a topological representation of the environment annotated with navigation-relevant information from the lower layers.

Each layer interacts with its neighbouring layers in a cooperative fashion yielding a control system with complementing and overlapping competences. The proposed schemata and adjoined guidelines have furthermore been applied to design a powerful and modular control system. The information bases required for the particular *Behaviours* have been specified in terms of *Virtual Sensors* and control-level aspects. These specifications can be regarded as stubs which have to be filled with information by the *Semantic Abstraction*. In the following chapters, the information sources, as well as the *Aspect-oriented Configuration* of the *Semantic Abstraction* will be explained.

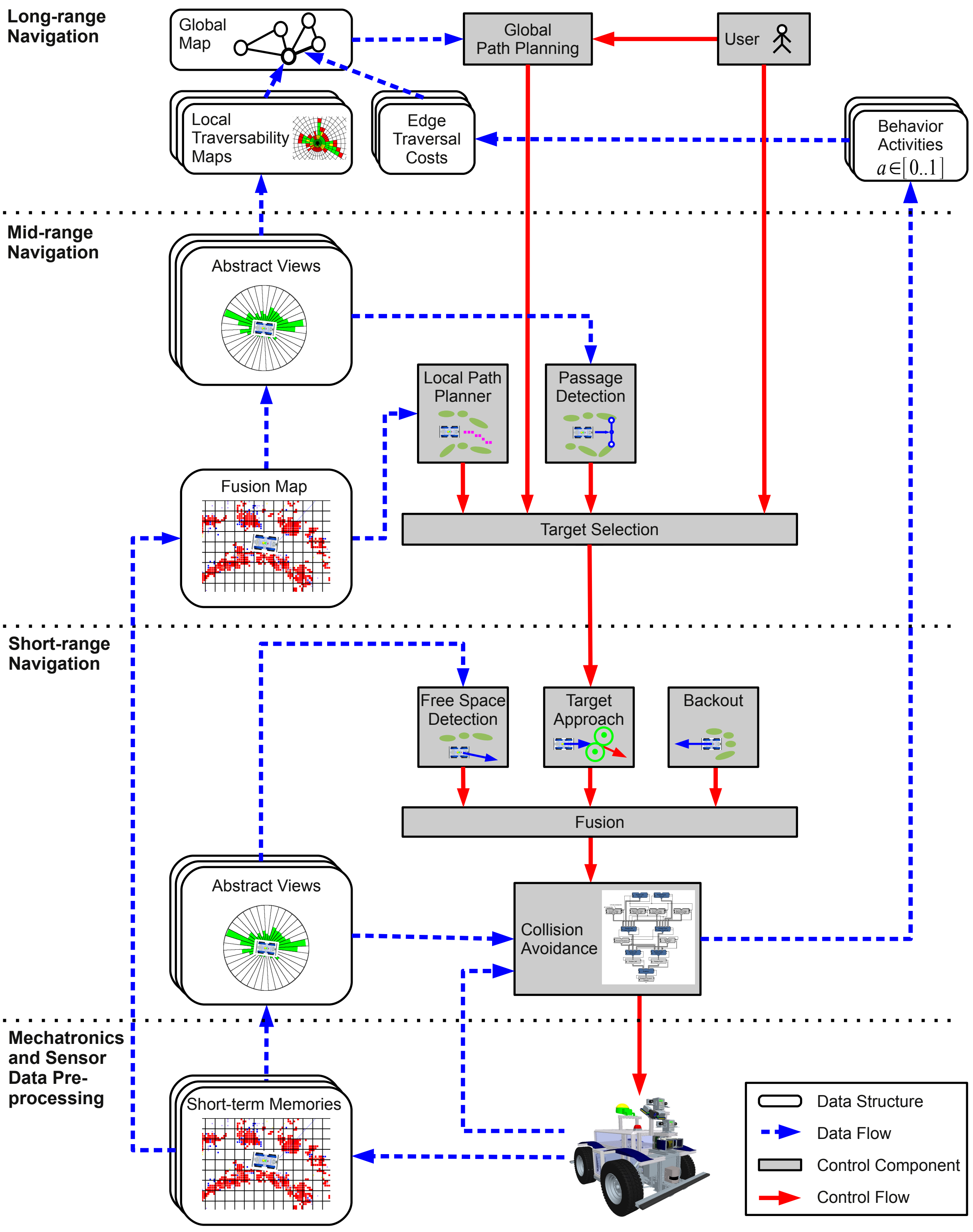


Figure 8.29: High-level overview of the control system components

9. Perception-oriented Sensor Processing Design

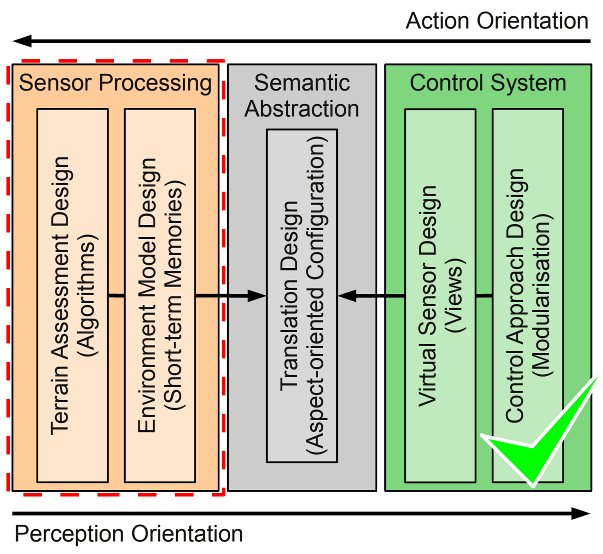

Action Orientation
Sensor Processing
Terrain Assessment Design (Algorithms)
Environment Model Design (Short-term Memories)
Semantic Abstraction
Translation Design (Aspect-oriented Configuration)
Control System
Virtual Sensor Design (Views)
Control Approach Design (Modularisation)
Perception Orientation

Figure 9.1: The second step in the design process is the perception-oriented design of RAVON's sensor processing facilities.

Figure 9.1 resumes the illustration of the design progress. In the previous design step, RAVON's control system was divided into straightforward subunits to manage the overall complexity of the system. For each subunit, the required environment information was declared in terms of *Virtual Sensor* specifications. The *Virtual Sensors* render the requested *Views* from short-term memories filled with information extracted from sensor data.

In this chapter, the design of the obstacle detection facilities developed in the context of this Doctoral Thesis shall be illuminated. In order to give the reader a comprehensive overview of the wide applicability range of the proposed methodology, the theoretical concepts shall be exemplified with concrete evaluation algorithms. In particular the design decisions taken will be discussed with a focus on how the sensor systems and algorithms chosen complement each other. The design procedure will be carried out bottom-up in a perception-oriented way. That way, specific sensor characteristics can be exploited to extract and represent as much information about the environment as possible. In contrast to the control level, required data structures will be very diverse and as a consequence partially incompatible among different algorithms. To give the reader a preferably straightforward understanding of algorithm and representation design, the author decided to introduce the sensor systems in order of increasing data complexity.

The following section will be dealing with the evaluation and integration of the instrumented bumper system. After that, the close-range monitoring using planar laser scanners shall be introduced in Section 9.2. In Section 9.3 the field of vision of the robot will further be increased by calling the 3D laser scanner into service. Finally, Section 9.4 will deal with the detection of water bodies using the 3D laser scanner and a specialised multi-camera system.

9.1 Hull Protection using Tactile Sensors

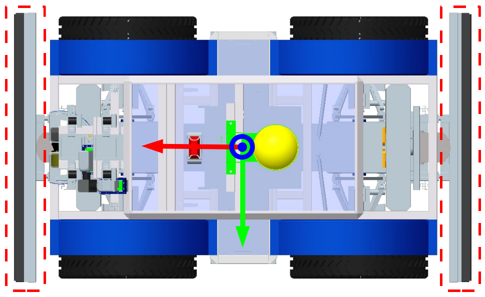
For robots operating in indoor and structured outdoor environments, the German *Technical Inspection Authority*[1] has specified several minimum safety requirements. One guideline states that safety bumpers must be placed at less than 5 cm over ground to bring the vehicle to an immediate halt in case the vehicle hits something. This emergency stop function has to be hardwired to the electronics of the vehicle and the trigger itself may not involve any software.

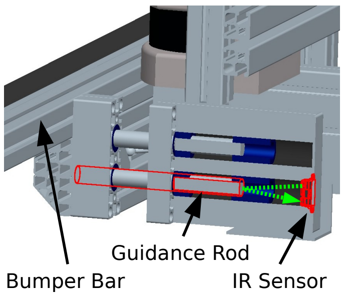
For off-road applications there are no clear guidelines so far which is certainly owed to the fact that ultimate safety in these scenarios is way more difficult to assure. The mounting height of a bumper system for example has to be tailored to the vehicle's climbing abilities as these facilities limit the system's ground clearance. On the other hand it is absolutely sensible to have an ultimate way to stop the robot from destroying itself or its environment using standardised industrial means. In consideration of these issues, the safety concept [Hillenbrand 07] initially developed for the RRLab's indoor platform marvin was transferred to ravon. Roughly speaking, this concept requires safety-critical hardware components to be coupled into a chain of hardwired electronic monitoring devices. At the terminal of this so called *safety chain*, the robot's critical actuation units (e. g. the propulsion system) are connected. In case one of the safety-critical components fails, the safety chain is opened and the connected actuation units are switched off.

On the basis of this concept, industrial safety bumpers were installed on ravon at about 30 cm over ground which reflects the ground clearance of the robot. That way, the all-terrain capabilities of the robot are not impaired and safety can be assured to a certain degree. Further problems arise from the fact that the wheels of most non-holonomous robots sheer significantly in curves constantly changing the shape of the robot while driving. To cover the entire area around the robot, a bumper system would either have to adapt to the changes in shape or enclose the complete kinematic hull of the vehicle. The former would certainly be connected with tremendous mechanical efforts which would probably never work reliably. The latter would significantly reduce the robots manoeuvrability as the minimum width of negotiable passages would increase due to a protruding bumper design.

[1]Translation from German: Technischer Überwachungsverein (TÜV).

(a) Instrumented bumper used for tactile terrain negotiation.


(b) Bumpers are mounted at the front and the rear side of the robot.



(c) Contact-less deflexion measurement mechanism.

Figure 9.2: Instrumented bumpers are mounted 30 cm over ground such that safety requirements can be met without impairing the platform's all-terrain capabilities.

As a compromise between safety and agility industrial safety bumpers were mounted at the front and the rear side of the robot as indicated in Figure 9.2b. The monitoring circuits of both bumpers are directly coupled into the safety chain of the robot stopping the vehicle without delay in case of emergency.

9.1.1 Bumpers that Go beyond Fundamental Safeguarding

Until now the vehicle does not feature any visual perception system and may only detect obstacles in a tactile fashion. For one moment, the author would like to assume that this remains like that and think about possibilities to exploit the facilities at hand as good as possible. Apart from the fundamental safety functions (introduced with certain compromises), tactile sensors can be deployed for obstacle detection and in consequence for obstacle avoidance. The problem with safety guidelines in that context is that if the safety chain is open it is not possible to benefit from the knowledge that the robot has hit an obstacle in order to try another path because the vehicle has already been deactivated. Reactivating the robot is electronically not possible as long as the safety chain is not closed again. For that reason a deflexion measurement system was integrated into the mounting construction, resulting in a two-staged instrumented bumper system which enables the robot to detect obstacles in a tactile fashion at very low speeds. The safety bumper is installed on a guidance mechanism consisting of four steel rods mounted into smooth running bearings. That way, the bumper gives way for about 5 cm (Stage 1) before the safety chain is opened by the industrial bumper bar (Stage 2).

For damping the construction and to rebound the bumper after a deflecting entity is gone, a gas pressure spring is placed between the chassis and the bumper bar. At higher speeds, hard impulses may act on the bumper bar when an obstacle is hit. The deflexion measurement system thus has to cope with strong forces without being damaged. For that reason, a contact-less mechanism on the basis of off-the-shelf infrared sensors was integrated which measures the deflexion distance of the guiding steel rods as illustrated in Figure 9.2c). By coating the inner side of the guidance tubes and the end of the rods in matt black colour, satisfactory measurement precision can be achieved with these simple means.

9.1.2 Seeing Touch through Representation

With the bumper construct at hand, certain safety properties can be met while further allowing for tactile obstacle detection well before the vehicle is shutdown by the bumpers coupled into the safety chain. To exploit this capability, the bumper has to be integrated into the robot's control system. Common approaches react to bumper events by recoiling the robot with a random steering angle in order to evade the obstacle at the next try. While this approach works well for simple situations, it will not stand a chance under more complex conditions like for example in dead ends. Furthermore, specific control strategies would be necessary for both the recoiling and the evasion of detected hazards. According to the control design outlined in Chapter 8, the behaviour combination *KeepDistance* and *Evasion* represents the interface desired for evasive manoeuvres. Furthermore, the reader should remember the Design Idea *Short-term Memory* from Section 3.2, which encourages the usage of representation to locally model the environment according to guidelines *Representation Deployment* (Guideline 2 on page 67) and *Representation Locality* (Guideline 3 on page 67).

In the case of the bumper system, a simple occupancy grid suffices for modelling the terrain as the source data does not allow for further interpretation. On the basis of the representation scheme introduced in Section 4.1, a suitable data structure can be configured without any extension of the available `ContentBase` element. Formally, the structural and semantic specification (see Section 4.1.3 on page 44 and Section 4.4 on page 59 for the theory on content and structure) can be chosen as follows:

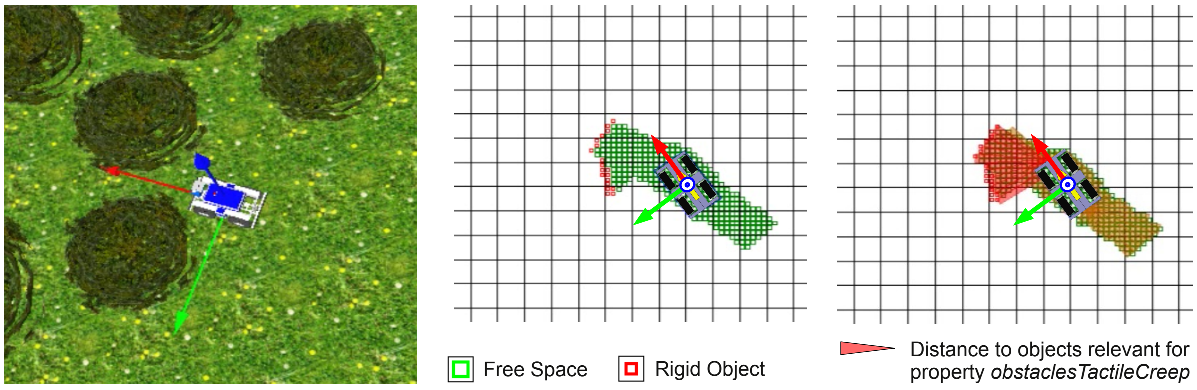
$$ShortTermMemory_{tactile} := (p_{min}, p_{max}, d_{cell}, mode)$$

where

$$\begin{aligned}
p_{min} &= (-r_{LocalPoseStability}, -r_{LocalPoseStability}) \\
p_{max} &= (r_{LocalPoseStability}, r_{LocalPoseStability}) \\
d_{cell} &= (width_{wheel}, width_{wheel}) \\
mode &= local
\end{aligned}$$

The dimensions of short-term memories should be chosen according to the properties of the sensor system in question like accuracy of measurements and correlation techniques used to aggregate the representation. In this application study, the range $r_{LocalPoseStability}$ in which the locally stable pose estimation (see Chapter 7) has an acceptable drift is chosen. The measurement accuracy of the instrumented bumper is difficult to rate as the sensor only provides information whether the bumper hit a rigid object or not. A sensible configuration for the cell dimensions would for instance be the width of a wheel $width_{wheel}$.

Furthermore, the property set for the short-term memory has to be specified. The *PROP* interface allows for an efficient and intuitive definition of relevant entities. The design methodology proposed in this work states, that property sets should be designed to feature *Natural Naming* (Guideline 11) and *Property Selectivity* (Guideline 12). While the former supports traceability at sensor processing design time, the latter is an important issue to prevent premature condensation of information that may be of use on higher levels of abstraction. From the perception-oriented point of view, the instrumented bumper can distinguish rigid objects from free space. Most ad hoc designs would probably result


🟩 Free Space
🟥 Rigid Object
Distance to objects relevant for property *obstaclesTactileCreep*

(a) Experimental setup. **(b)** Short-term memory. **(c)** Virtual Sensors.

Figure 9.3: Simulated dead-end situation (a) with the corresponding short-term memory (b) representing tactile events (green: free space, red: rigid objects) and views of virtual sensors (c) used for seeing touch (brown: free space sectors, red: obstructed sectors).

in a simple occupancy grid map which merely stores occurring bumper events. At first sight this modelling appears appropriate and seems to capture all relevant information. *Property Selectivity* (Guideline 12 on page 69) however states, that information should not be condensed in a premature fashion. At sensor processing design time, the potential usage of extracted sensor information may not yet be settled. Apart from simple occupancy, the knowledge whether a patch of terrain was already traversed before may also be of vital importance for certain control strategies. In Section 10.2, the configuration of the planning aspects unveils that this information may for instance be used to overrule data from less reliable sensors in the fused data basis (see page 194). For these reasons, the property set *tactilePROP* is specified in a more selective fashion:

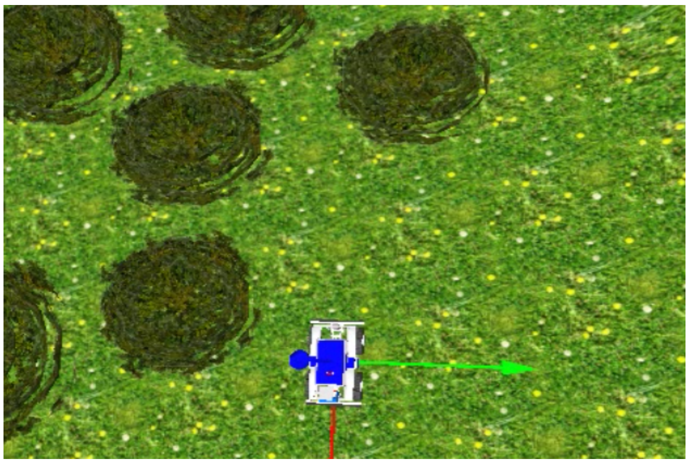
$$tactilePROP = (rigidObject, freeSpace)$$

where

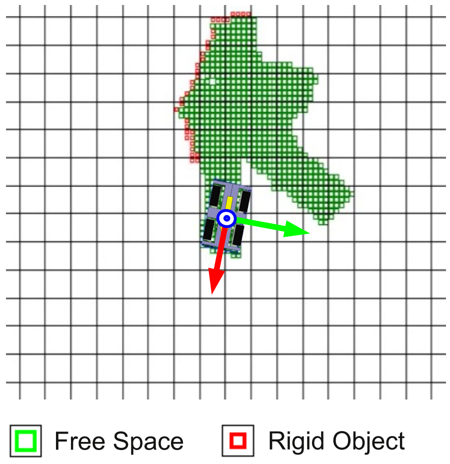
$$rigidObject(i) = \begin{cases} true, & \text{the bumper was activated} \\ & \text{while passing over content element } i \\ false & \text{otherwise} \end{cases}$$

$$freeSpace(i) = \begin{cases} true, & \text{the bumper was not activated} \\ & \text{while passing over content element } i \\ false & \text{otherwise} \end{cases}$$

Whenever a bumper event occurs, the current location of the bumper bar is marked as *rigidObject*. As long as the bumper is not deflected, the covered terrain patches are marked as *freeSpace* resulting in a local terrain representation as illustrated in Figure 9.3b. This terrain representation can now be cast into views using virtual sensors (Figure 9.3c)



(a) Experimental setup.



(b) Short-term memory.

Figure 9.4: The dead-end situation (a) was negotiated on the basis of tactile data with assistance of quasi-visual behaviours using views rendered from an internal representation (b).

suitable for the control strategy based on *Evasion* and *Keep Distance* behaviours in a transparent fashion. That way the robot practically uses a visual projection of tactile information in order to cope with more difficult situations like dead ends.

In essence, the vehicle sees what it touches as a "natural" mobile system would do for example in case of complete darkness. Furthermore, the complete kinematic capabilities can be exploited in order to negotiate very narrow situations. Figure 9.4 shows a dead end situation in a simulated environment (a) and the short-term memory (b) built up on the basis of the tactile facilities only.

In the experiment at hand the robot was merely given the order to drive forwards. In case that only the tactile sensing systems are activated, drive mode *Tactile Creep* is selected by the *Sanity Monitor* (see Section 8.2.1 on page 102). The maximum velocity is thus limited to $0.1\frac{\mathrm{m}}{\mathrm{s}}$ which allows for safe operation of the robot. Necessary backward manoeuvring is achieved by a *Back Out Behaviour* which is guided towards free space using a forward-looking virtual sensor and an *Open Terrain Attractor Behaviour* (see Section 8.2.2). As soon as the path is obstacle-free, the robot resumes forward motion using the evasive behaviours on the basis of the short-term memory to avoid obstacles which have already been detected in the past.

With the tactile sensor equipment and the control strategy outlined above, the robot has the competence to navigate terrain featuring vertical obstacles that are large enough for the bumper to detect. The crucial key to extended evasive capabilities is the integration of a short-term memory for locally remembering obstacles that have already been detected before. By abstracting the short-term memory using standardised virtual sensors, the generic evasive behaviours introduced in Section 8.2.1 on page 87 can be deployed to steer the robot around obstacles on a quasi-visual basis.

The semantic basis for these *Behaviours* is configured by specifying how the control-level aspect *obstaclesTactileCreep* introduced in Equation 8.2 on page 100 has to be interpreted from the tactile short-term memory. For obstacle avoidance, objects of relevance are clearly the rigid entities. Free space in contrast is not critical as it represents traversable terrain. In this case, the formal specification for the semantic translation (see Section 4.2 on page 48) can be derived in a straightforward fashion:

$$\mathit{obstaclesTactileCreep}_{\mathit{tactile}} = \mathit{rigidObject} \wedge \neg \mathit{freeSpace}$$

Note that no special control strategy has to be implemented to realise the functionality illustrated in the experiment. The tactile information integrates seamlessly into the control system via the *Virtual Sensor* interface.

9.2 Close-range Safety using Planar Laser Scanners

So far the robot can navigate at very low speeds and react to rigid obstacles which it encounters on its way. In order to increase travelling velocity, further perception systems shall be introduced in this section. At higher speeds the tactile safety facilities are of little use, as the deflexion range of bumper stage 1 (see Section 9.1) cannot compensate for longer breaking distances. In a higher-speed collision emergency, the strong impulse on the bumper bar will immediately stop the vehicle by opening the safety chain.

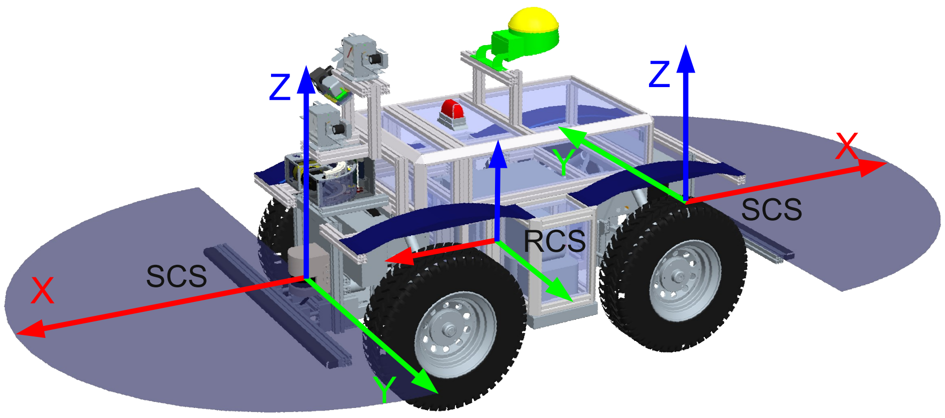
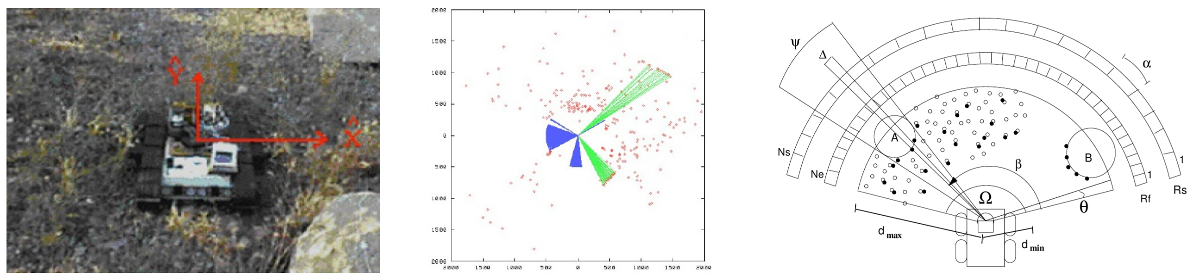


Figure 9.5: Close-range safety zones are monitored by planar laser range finders mounted at the front side and the rear side of the robot.

In order to prevent injuries or severe damage when the robot collides with humans or other entities, the bumper bar is mounted on a flexible spring system which gives a few centimetres way in cases of abrupt impacts. The flexible spring system will prevent serious injuries and severe damage when the robot collides with humans or other entities at speeds up to $0.5\,\frac{m}{s}$ but the deflexion measurement mechanism will not have any chance to gracefully stop the robot and initiate positioning manoeuvres. Therefore, further close-range safety facilities are required to assure collision-free operation at higher speeds.

9.2.1 Obstacle Detection in 2D Laser Data with Noise

As illustrated in Figure 9.5, two planar laser range finders (2D LRF) have been mounted right above the bumper bars at the front and rear side of the vehicle. In the following, the Sensor Coordinate System (SCS) of a 2D LRF shall be defined as illustrated in Figure 9.5. That way, ground clearance is not impaired while again trying to capture obstacles as low over the ground as possible. This laser scanner configuration is common for driverless transport systems in industrial environments. Resolution and update frequency of the sensors can be proven suitable for the detection of vertical objects a few centimetres wide at distances that allow for bringing the vehicle to a graceful halt in time. For the platform at hand, this comprises in particular human beings of whom the legs are detectable and thin obstacles like small trees and posts. This property of laser scanners can to a certain degree be assumed in off-road terrain, as well. Yet, further filtering steps will be necessary



(a) Robot in vegetated scenario. **(b)** Scan classification. **(c)** Classification concept.

Figure 9.6: Vegetation discrimination according to [Castano 03].

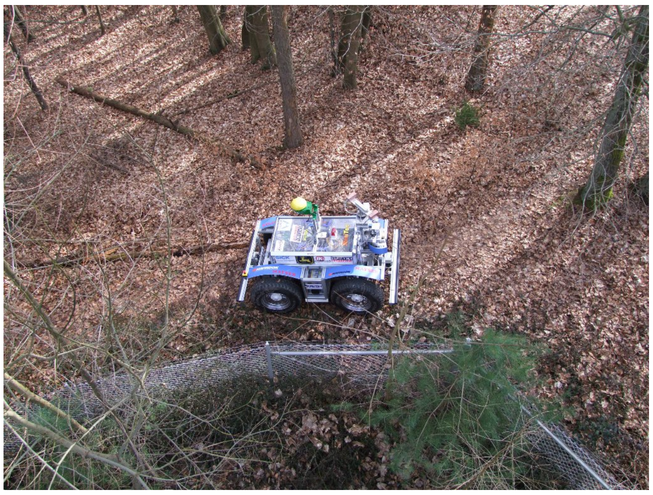
to compensate for environmental conditions like rain, dust, and vegetation which are omnipresent in natural environments. State of the art approaches analyse the distribution of range values in a statistical fashion to cope with noisy input data. In [Macedo 01], a theoretical model for vegetation is used to simulate typical situations of a robot in vegetated environment. Furthermore, a classification process is derived from the theoretical model which is based on the statistical analysis of range histograms. Experiments on real sensor data show the validity of the approach despite certain simplifications in the statistical model. Additional experiments [Matthies 03, Manduchi 04] yielded statistical results on laser penetration depth for different types of vegetation.

The statistical models were combined with a refined version of the approach which further exploits spatial locality of laser range data and temporal locality of the scenery. In [Castano 03], methods from the signal processing domain are used to filter range data according to the frequency of change in distance (see Figure 9.6). The scan is interpreted as polar coordinates which are partitioned into non-overlapping angle ranges. A maximum filter is applied to each angle range. Obstacles in contrast to vegetation tend to feature locally flat surfaces. Therefore, the subsequent selection step thresholds low values of the second derivative of the filtered signal (i. e. areas of low frequency in distance change) as obstacles and high values as vegetation. Simple temporal filtering is used to reduce false classifications.

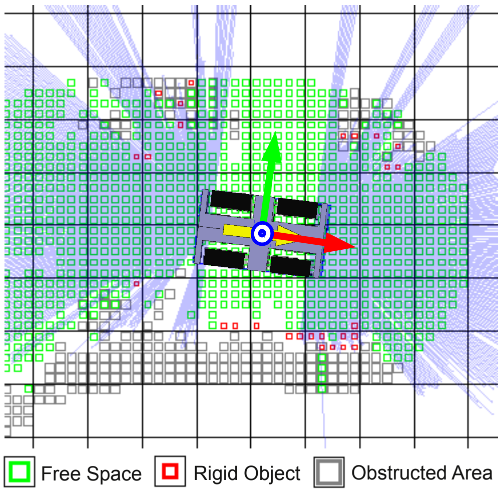
9.2.2 Density Measure based on Laser Penetration Statistics

In the context of this work, a statistical density analysis is carried out on the laser range data, which is based on a regular grid and ray tracing techniques. Reflecting the *Representation Deployment* (Guideline 2 on page 67) in this context is straightforward as the density data can be computed on the basis of a scrolling grid map, which may as well serve as short-term memory. Relative frequencies of hits and intersections are computed using the probabilistic content element extension introduced in Section 4.1.3 to account for spatial locality. The probabilistic update mechanism of the representation furthermore provides temporal filtering in a transparent fashion (see Section 4.3.2).

Laser beams are followed from the sensor centre until an object is hit. For the cell that was hit, a counter reflecting object rigidity is incremented. For all cells between that cell and the sensor origin, a counter reflecting free space is incremented. For all cells behind the cell in question, a counter reflecting an obstruction is incremented. After the evaluation of



(a) Experimental Setup.



(b) Short-term memory.

Figure 9.7: RAVON in a boundary fence patrol scenario.

all beams in one scan, the relative occurrences of hits, intersections, and obstructions are computed into probabilities for each cell for the following properties:

$$laser2dPROP = (rigidObject, obstructedArea, freeSpace)$$

where

$$rigidObject(i) = \begin{cases} true, & \text{a statistically significant number of} \\ & \text{laser beams hit content element } i \\ false & \text{otherwise} \end{cases}$$

$$obstructedArea(i) = \begin{cases} true, & \text{a statistically significant number of} \\ & \text{laser beams were absorbed before content element } i \\ false & \text{otherwise} \end{cases}$$

$$freeSpace(i) = \begin{cases} true, & \text{a statistically significant number of} \\ & \text{laser beams passed through content element } i \\ false & \text{otherwise} \end{cases}$$

Figure 9.7a shows RAVON in a boundary fence patrol scenario. The short-term memory resulting from the statistical analysis of the 2D LRF is illustrated in Figure 9.7b.

Off-road terrain cannot be assumed flat and it is therefore possible that the scanning plane intersects the ground resulting in phantom obstacles. For this reason, the ray tracing algorithm further distinguishes low range and far range objects and obstructions in order to have a further criterion for controlling the robot. Far range entities may either represent real objects or phantom objects but there is no possibility to decide this from the distance. While real objects always require evasive action, phantom objects do not, as these mean no harm to the robot. Yet, phantom objects are an indication for rugged terrain or a severe change in slope. In both cases it is therefore a good idea to reduce the velocity in closer proximity in order to have time to react if the detected entities turn out to pose a threat to the robot. Low range entities fall into a distance where the robot is most certainly dealing with a rigid object which requires evasive action. Furthermore, it is sensible to reduce the velocity nearby obstacles to assure safe navigation. The control strategy outlined in Chapter 8 provides means for both slowing down the vehicle and initiating evasive manoeuvres. *Semantic Translation* (see Section 4.2) allows for the extraction of different aspects from the environment information to cast relevant data into abstract *Views* which separate different concerns. At this point, two sets of *Views* can be generated from the short-term memory for the *Guardian* aspects (see Section 8.2.1). While one set contains only low range entities, the other considers both, low and far range entities. The former set of *Views* is then integrated into the steering control chains[2] – namely the *Rotation* and *Sideward* chain – such that evasive manoeuvres are carried out on the basis of this information. The latter is only integrated into the *Velocity* control chain such that the vehicle slows down in proximity of potential obstacles but does not change direction on premature data. This is a good example how *Semantic Abstraction* can be configured to tailor views for a particular purpose in a transparent fashion. In that context, selective attention is directed towards different types of entities in order to separate velocity control from steering control without breaking the control paradigm (*Property Selectivity* (Guideline 12 on page 69)).

9.3 Scene Analysis using a 3D Laser Range Finder

The obstacle detection mechanism introduced in the previous section is robust for avoiding collisions with vertical structures that reside on a more or less flat ground. For closed industrial facilities this in combination with a safety bumper which shuts the system down on collision is the state of the art [Pradalier 08]. Yet, at the base of slopes the planar laser range finders will not be able to distinguish obstacles from the ground. Even if the orientation of the vehicle is known, only rough estimates can be made. In addition, off-road environments are highly unpredictable and not necessarily under the control of humans – i. e. it is not possible to simply put away hindrances the robot cannot detect. In order to cope with rugged terrain, spatial information of the scenery is mandatory. For that reason RAVON was equipped with a 3D LRF[3]. Before going into detail with the detection mechanism, the mechanical setup shall be motivated.

9.3.1 3D Laser Ranging

In off-road robotics, 3D LRF are very popular as they provide accurate and reliable geometric information about the environment. Most 3D LRF are built up from commercial planar LRF[4] in combination with actuation units which yield 3D data by panning, tilting, or rotating the sensor (e. g. [Brenneke 03, Singh 02, Patel 05, Lamon 06]). Besides the

[2]Control chains reflect the robot's degrees of manoeuvrability.
[3]3D Laser Range Finder (3D LRF)
[4]e. g. SICK → `http://www.sick.com/`

basic actuation mechanisms, several alignments of the scan plane are possible for creating a 3D representation of the environment [Wulf 03]. One complete 3D scan thus consists of a set of subsequent individual scans. Timing issues, sample distribution, and sensor coverage have to be considered with regard to the application in question.

Y-Axis actuated 2D LRF

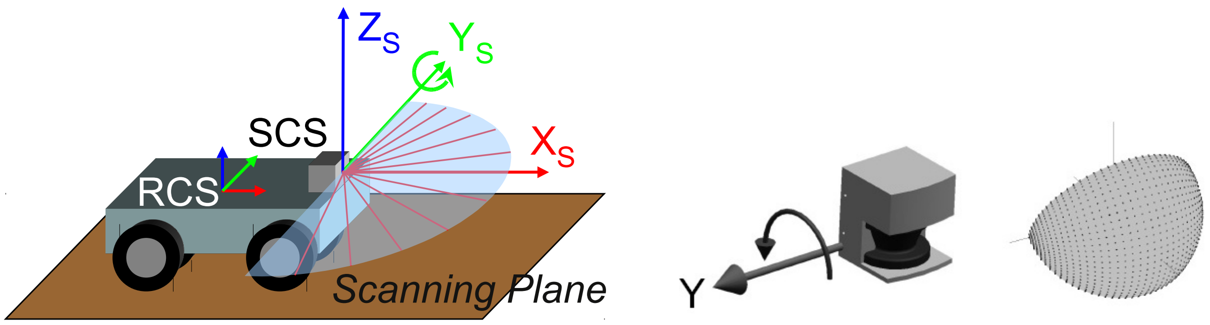


Figure 9.8: 3D LRF built up from an y-axis actuated 2D LRF with horizontal y-axis (a) and the resulting point cloud (b) according to [Wulf 03].

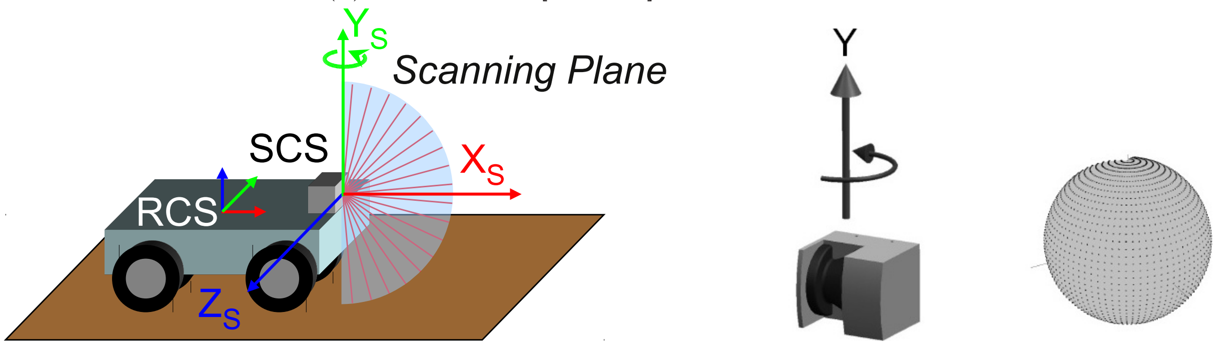
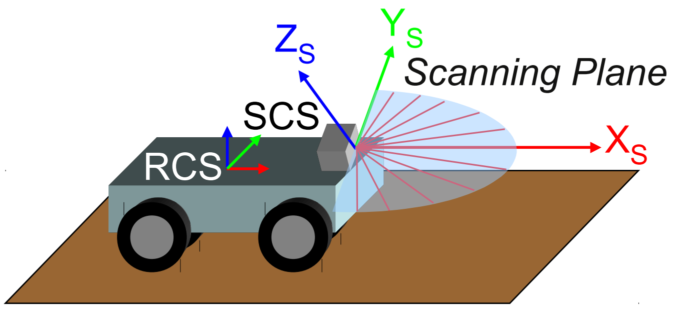


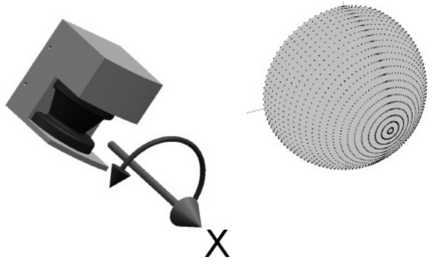
Figure 9.9: 3D LRF built up from an y-axis actuated 2D LRF with vertical y-axis (b) and the resulting point cloud (b) according to [Wulf 03].

In case the 2D LRF is rotated around its y-axis, the scenery is captured either in horizontal sections (Figure 9.8) or vertical sections (Figure 9.9). Both approaches allow for the straightforward evaluation of the range data. In particular if scans shall be treated independently of one another the latter has further advantages as vertical terrain sections always provide a ground reference. The resolution of scanned objects highly depends on their distance to the sensor. If the x-axis is pointed towards the driving direction, resolution decreases with the distance to the vehicle. Besides continuous rotation yielding 360° scans, pan or tilt modes can be handy to focus on a particular angular range. The horizontal alignment of the y-axis, as depicted in Figure 9.8, yields a horizontal sampling of the terrain. In tilting mode the scanner can be concentrated towards ground-near parts of the terrain such that larger obstacles are virtually present in all scans. When the y-axis is aligned vertically (see Figure 9.9), the scanning plane is orthogonal to the x-y-plane of the robot coordinate system (RCS). The directions of this plane can be regarded as an approximation of the ground at the robot's current location. On the basis of this assumption, each individual scan captures the terrain profile in the current sensing direction. In essence, every scan provides a ground reference which allows the separate evaluation of individual scans. This has the advantage that inaccuracies in robot pose estimation do not have any impact on the detection performance. However, the area in front of the robot is not covered completely until a complete panning pass has been carried out. This makes this approach vulnerable in dynamic environments. The panning period is thus a central point when deploying this mechanism.

X-Axis actuated 2D LRF

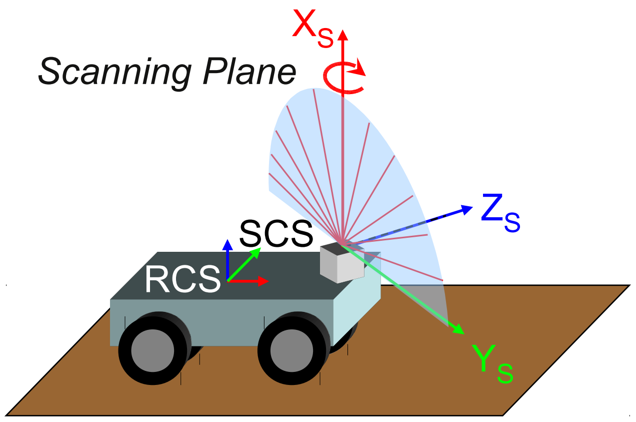


(a) 3D LRF mounted on a robot.

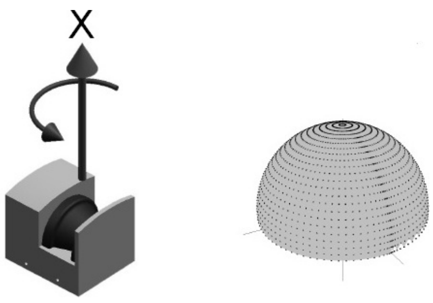


(b) Point cloud yielded from the 3D LRF [Wulf 03].

Figure 9.10: 3D LRF built up from an x-axis actuated 2D LRF with horizontal x-axis.



(a) 3D LRF mounted on a robot.



(b) Point cloud yielded from the 3D LRF [Wulf 03].

Figure 9.11: 3D LRF built up from an x-axis actuated 2D LRF with vertical x-axis.

In contrast to y-axis actuation, which may yield full 360° range data, x-axis actuated 2D LRF can in principle only generate 180° scans. In exchange a complete scan is already available after one half a turn. Figures 9.10a and 9.11a illustrate configurations with x-axis actuated 2D LRF. Evidently the laser beams close to the x-axis are only slightly displaced in subsequent scans while the displacement increases with angular distance to the x-axis. Therefore, the resolution of a 3D scan is highest at the centre of rotation and decreases outwards. To yield a high resolution ahead of the robot, the x-axis should be aligned to the driving direction of the vehicle (Figure 9.10a). In the centre of rotation, the distance of objects to the sensor does not have much impact on the resolution. Furthermore, the continuous rotary motion is mechanically more efficient than panning or tilting as the effort

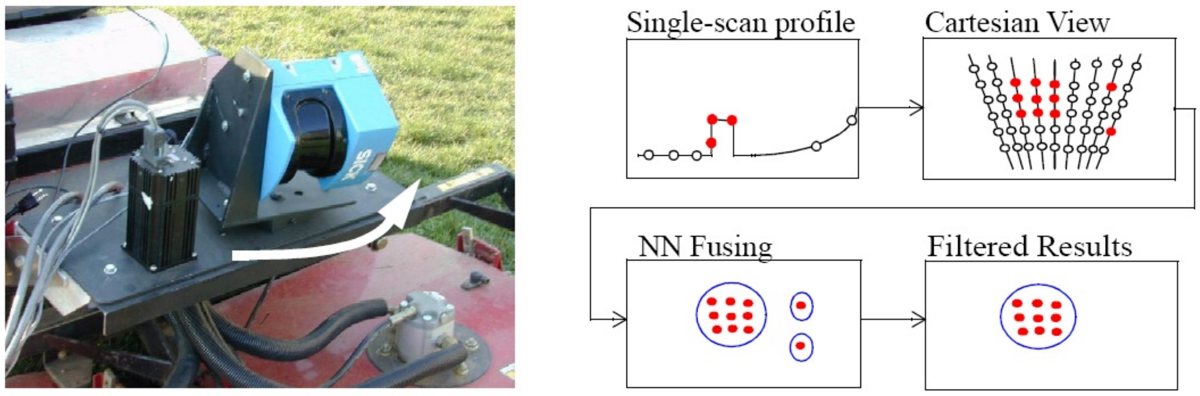


Figure 9.12: In [Singh 02] scans are analysed independent of one another. The preprocessed information is filtered and accumulated using a neural network.

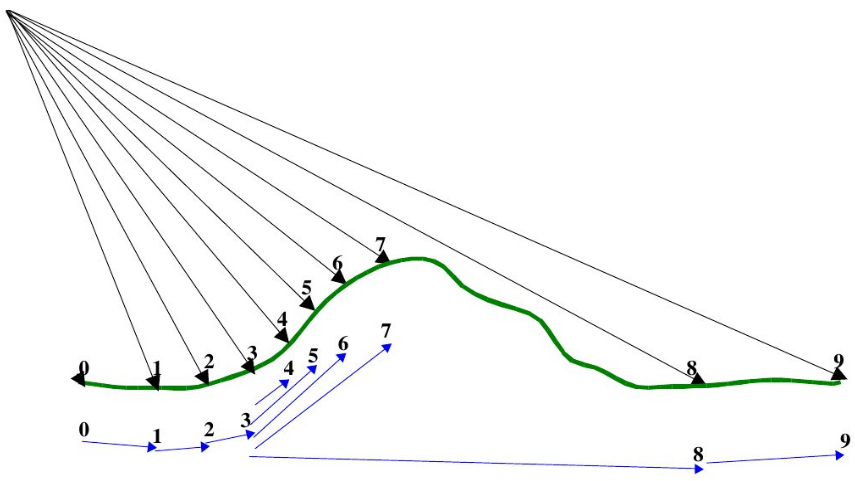
for inversion of the direction falls away. That way, a steady sampling of the environment at high rotary velocity is feasible. On the downside, this configuration only allows for the evaluation of point clouds resulting from a complete scan (half a turn). Individual scans are difficult to judge independently of each other as no assumptions on the connection between ground and objects can be made. Figure 9.11a shows the second configuration of an x-axis rotated 2D LRF where the x-axis is aligned orthogonal to the x-y-plane of the RCS. The angular range is in this case limited to the upper hemisphere. Indoors or in caves this arrangement may find its niche but due to the missing ground reference this configuration is rather inapt for obstacle detection in off-road robotics.

9.3.2 Evaluation of 3D Laser Range Data

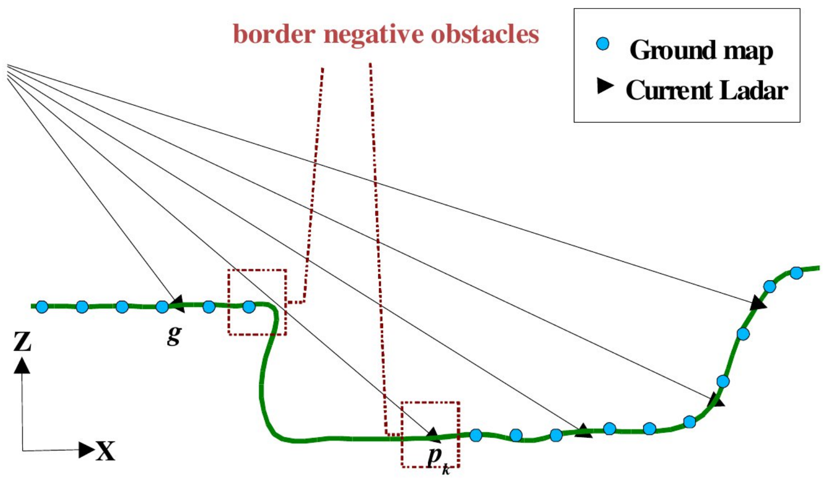
Approaches for the evaluation of 3D laser range data yielded from 2D scans can be assigned to two distinct classes: The individual scans are either evaluated independent of one another or accumulated over time to generate a 3D point cloud which is then analysed as a whole. For example [Talukder 02, Lalonde 06] present approaches working on accumulated 3D point clouds. A central advantage of using 3D point clouds of complete sceneries is that ground and obstacle structures can be analysed more thoroughly as more context information is available than in approaches based on individual terrain sections. On the downside, the evaluation of large point clouds is computationally very expensive due to the high data volume in comparison to individual scans.

Furthermore, the registration of several subsequent scans to yield a complete scan is highly error-prone (see *Data Integration* (Challenge 1) on page 21). In off-road applications, vehicles have to cover rugged terrain preferably at rather high velocities. With 3D LRF which consist of actuated 2D LRF, the sampling of the terrain may take up to several seconds. During this period of time the robot may have moved several metres between the first and the last scan of a complete pass. Errors in position and orientation thus accumulate until the point in time where the evaluation of the point cloud is carried out.

The second class of approaches evaluates individual scans or a small number of subsequent scans. The preprocessed information can later be combined into a representation that is suitable for the application at hand. Depending on the field of application and the purpose of the target platform, the definitions of obstacle classes to be extracted from



(a) Detection of positive obstacles on the basis of slope and height difference (indicated by blue arrows).



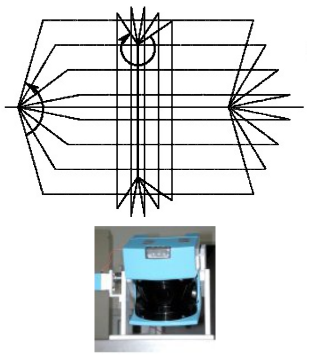
(b) Detection of negative obstacles.

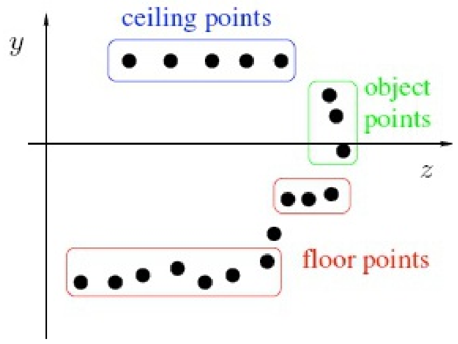
Figure 9.13: Obstacle detection in 3D LADAR data which is analysed column-wise [Hong 00].

sensor information vary tremendously. 3D scene analysis ranges from obstacle detection over land mark extraction towards localisation and mapping applications. A further crucial criterion is the maximal velocity that is allowed during data registration. Some approaches generate detailed 3D views during which the robot may not be moved at all. In other works the focus is laid on gathering information at preferably high velocities.

[Singh 02] presents an algorithm for obstacle detection in park or golf course scenarios. The ground may to a certain degree be covered by vegetation but hindrances are always discrete well-identifiable entities. The 3D LRF used here is a 2D LRF panned periodically around the sensor's y-axis yielding vertical terrain sections as illustrated in Figure 9.12. The proposed algorithm consists of two steps. In the first step, scan points of individual scans are classified on the basis of terrain slope as representing free space or obstacles. After that, obstacle points of several subsequent scans are clustered and filtered using learning techniques. Data evaluation is carried out in a continuous fashion at a velocity of $1.5 - 2\frac{\text{m}}{\text{s}}$. The work presented in this paper is limited to the detection of hindrances.

(a) Outdoor robot Kurt3D.


(b) Scan point registration.



(c) Scan point classification.

Figure 9.14: Scan point registration (b) and classification (c) on Kurt3D (a) according to [Nüchter 06].

The extracted information is neither aggregated to a terrain representation nor used for avoiding obstacles.

[Hong 00] evaluates the range image from a 3D LADAR[5] column-wise which is similar to analysing individual scans. Starting from the last point that was classified as ground, critical slopes and height differences are identified using thresholds. In case one of the two criteria is met, the point in question is classified as belonging to a positive obstacle and the next point is analysed relative to the current starting point. Otherwise the point is classified as belonging to the ground plane and the algorithm restarts at the new point. Figure 9.13a illustrates the procedure for the detection of positive obstacles. According to the algorithm outlined above, points 1, 2, 3 as well as 8, 9 are classified as belonging to the ground plane while the remaining points are classified as a positive obstacle. The ground points are stored in a robot-local grid map, which moves with the robot in its absolute working coordinate system in an orientation-aligned fashion. Among other things this map is used to classify negative obstacles. The density of ground points in the grid map is a lot higher than in a single scan. New ground points p_k are related to the ground point in the grid map which is closest in negative x-direction (see Figure 9.13b). In case that x- and z-distance of the two ground points are larger than thresholds which reflect the vehicle's dimensions and climbing capabilities, a critical hole or trench was detected. In consequence, these points are marked as the border of a negative obstacle.

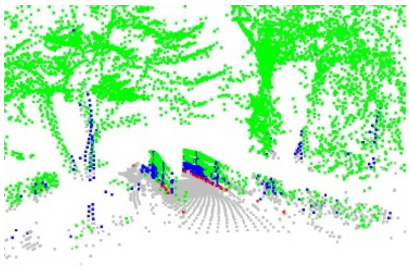
[Nüchter 06] describes the extraction of drivable planes on the basis of 3D LRF data. The approach is evaluated using the outdoor robot *Kurt3D*[6] (see Figure 9.14a) which is suitable for parks and urban areas. The vehicle is equipped with two cameras and a tiltable 2D LRF which yields horizontal sections of the terrain in front of the robot. The scans of one pass are aggregated to create a 3D point cloud which is then analysed column-wise. The focus of research in this work is on localisation and mapping (6D-SLAM[7]). In order

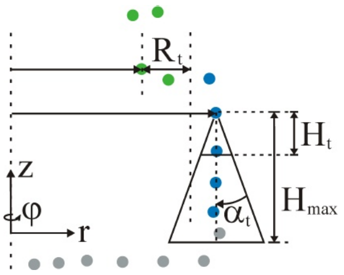
[5]Laser Detection and Ranging (LADAR)

[6]Kurt3D is based on the Kurt2 platform developed at the Fraunhofer Institute for Autonomous intelligent Systems AiS [Worst 02].

[7]Simultaneous Localisation and Mapping (SLAM)

(a) Outdoor robot RTS Dora.


(b) Scan points are classified as ground (grey), overhanging (green), or landmark (blue).



(c) Classification scheme.

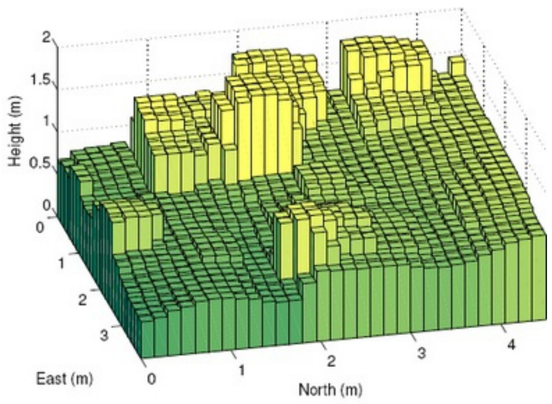
Figure 9.15: Scan point classification (c) and classified point cloud (b) on RTS Dora (a) according to [Brenneke 03].

to obtain preferably accurate local 3D point clouds, data registration is not carried out on the move. From the point of view of the latest 3D scan, a promising novel position for taking the next scan is computed (best next point of view approach). The robot approaches the designated location and registers a new point cloud which is classified and then integrated with already registered data. In Figure 9.14b, the actuated LRF and the point cloud assembly from individual scanning planes is depicted. For the extraction of drivable planes, the 3D point cloud is transformed into a cylindrical coordinate system yielding virtual scanning planes which are perpendicular to the ground plane. The slope between subsequent scan points in these virtual scanning planes are classified into *ground points*, *object points*, and *ceiling points* using thresholds.

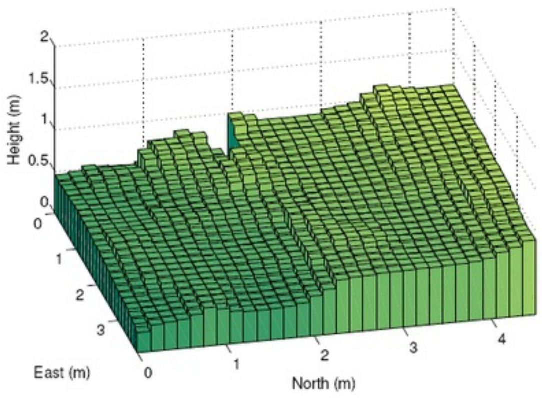
For navigation at high velocities (about $10\frac{\mathrm{m}}{\mathrm{s}}$), [Patel 05] presents an approach for optimising sensor coverage in curves using vertical terrain sections from a 3D LRF based on a panning mechanism. Scans are registered to an absolute 2D grid map with the attempt to focus on grid cells that lie on the trajectory of the vehicle. The route is given as a sequence of 2D waypoints. The major objective of the proposed approach is to pan a vertically mounted 2D LRF such that the number of unscanned grid cells to traverse are minimised. Depending on the current and the planned heading as well as the regions already covered by the sensor, the optimal pan angle is determined. The height difference of neighbouring samples in each scan is computed to classify the scan points as traversable or non-traversable.

[Brenneke 03] introduces a SLAM approach for parks and urban environments which is based on 3D LRF data. The robot (see Figure 9.15a) is equipped with a rotating mechanism that yields a 360° scan of the environment that is composed of vertical terrain sections. The scans are continuously registered on the move. The samples of one complete revolution are aggregated into a 3D point cloud. The coordinate system of each point cloud is axis-aligned to an absolute coordinate frame and has its origin at the robot position at the point in time the first scan of this very point cloud was registered. The points of the vertical scans are classified as ground, landmark, or overhanging according to the classification scheme outlined in Figure 9.15c. Ground points are extracted straightforward using the slope as criterion. A scan point is labelled overhanging if the distance of a lower angle point is larger than that of the point in question with R_t being the minimum distance difference. Landmarks are non-overhanging vertical objects of a minimal height H_t in a conical tolerance area of maximal angular misalignment of $\pm\alpha_t$. The landmark points are projected into a 2D grid map which is used for data alignment in the SLAM process.

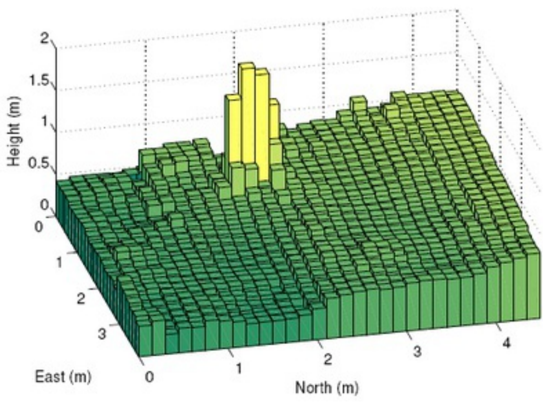
(a) Man partially hidden in high grass.



(b) Highest samples in the voxel map.



(c) Lowest samples in the voxel map.



(d) Load-bearing surface prediction learnt online.

Figure 9.16: In scenarios with intense vegetation (a) a load-bearing surface prediction (d) is learnt online from differences of highest (b) and lowest (c) samples in a voxel map [Wellington 04].

A further challenge in off-road navigation is terrain covered with intense vegetation. [Wellington 04] presents a machine learning approach for extracting the load-bearing surface and obstacles hidden under high vegetation. The approach uses a voxel-based representation of the environment to capture statistical density values computed from laser beam hits and penetrations (see Figure 9.16). Furthermore, features like object reflectivity and the statistics of height values per voxel column are determined. By means of this statistical data and the real ground level determined while passing over terrain previously registered in the voxel-map, the prediction of the load-bearing surface is learnt online. In order to provide the learning algorithm with a good starting point, non-traversable locations are manually labelled during a training phase. More details on this approach and extensive experiments can be found in [Wellington 06].

9.3.3 Obstacle Detection and Traversability Analysis for Off-road Environments

Many of the presented approaches are limited to outdoor environments like parks which feature a certain structure that allows for straightforward obstacle detection mechanisms. Furthermore, several types of hindrances crucial for off-road navigation have not been treated. Overhanging objects are often neglected or not taken into account for traversability analysis. In general, obstacles are mostly regarded as discrete and isolated objects which reside on a rather well traversable ground plane. A detailed analysis of the ground structures is not carried out in any of the works outlined above. Minor jaggedness or obstacles near the ground (e. g. trunks of fallen trees that may be traversable at low speeds) are ignored as well as the current roll and pitch angles of the robot. RAVON for example is in principle capable of climbing slopes of up to 100% and its ground clearance of 30 cm allows for the negotiation of pretty jagged terrain. Yet in order to exploit the agility of the platform, a precise evaluation of ground structures is of great importance to prevent the robot from getting stuck. In off-road scenarios, traversability analysis has to capture as many details on the terrain as possible and the mechanisms have to be very stringent to allow the control system to evolve navigation strategies for harsh and narrow driving situations.

Obstacle Structures in Off-road Environments

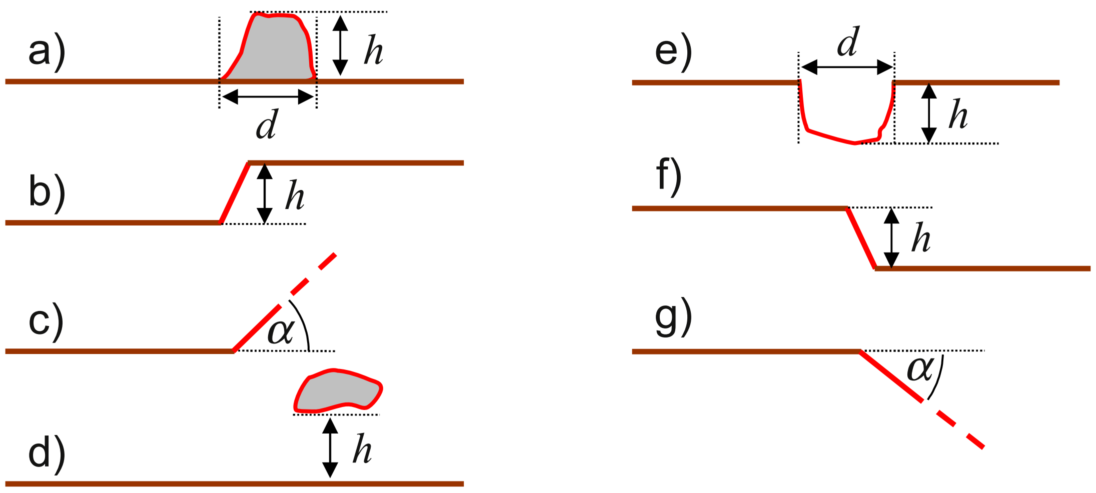


Figure 9.17: Common obstacle structures in harsh terrain: a) positive obstacle, b) positive step, c) ascent, d) overhanging object e) negative obstacle, d) negative step, c) descent

Before going into detail with the proposed obstacle detection mechanism, conformations which may represent a threat to a robot in natural terrain shall be discussed. Figure 9.17 illustrates several common obstacle classes which occur in off-road terrain.

a) *Positive Obstacles* may be stones, tree stumps, or higher vegetation. These objects have ground contact and surpass a determined height which exceeds the mounting abilities of the vehicle at hand. In contrast to stones and tree stumps, flexible vegetation may be negotiable with certain effort which is related to the platform size and weight.

b) *Positive Steps* protrude from the ground in an abrupt fashion and are only traversable if the height difference h is smaller than the mounting abilities of the robot. Otherwise, such structures represent a threat to the robot. In case h is very large, the more specific term cliff may be used.

c) *Ascents* in contrast to *Positive Steps* ascend smoothly over a longer distance. In the case of an extreme slope angle α, ascents may also represent a non-traversable hindrance for the robot.

d) *Overhanging Objects* are partially without contact to the ground. Branches of trees and bridges are representatives of this class of objects. If the distance to the ground h is larger than the vehicle height, the object does not mean a threat to the robot. Otherwise, the object in question has to be considered as an obstacle.

e) *Negative Obstacles* are structures like holes and trenches which back into the ground in contrast to the obstacle classes introduced so far. For the classification of such indentations, the depth h and the length d are important parameters. Trenches and holes which are of minor extent or feature only a small height difference can be negotiated. In that context, wheel diameter and climbing capability determine the limits of the vehicle. In case either h or d is too large, the terrain structure represents a negative obstacle.

f) *Negative Steps* represent the same natural entities as *Positive Steps* from the other viewing direction. The same holds for g) *Descents* and c) *Ascents*.

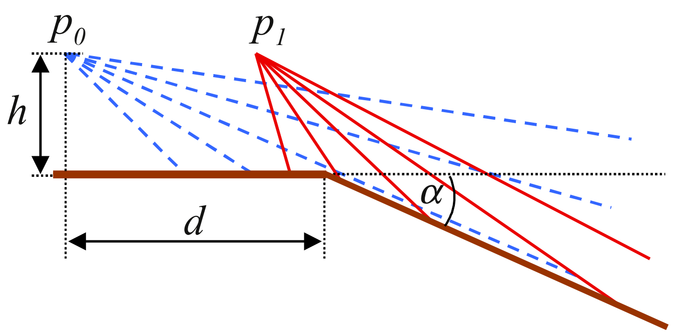


Figure 9.18: From larger distances (p_0), descents may be misinterpreted as cliffs. Only in closer proximity (p_1) the steepness can reliably be determined.

Hindrances which feature nagative slope – i. e. back into the ground – are more difficult to detect than structures protruding from the ground, as their characteristics can only fully be determined when the sensor is in close proximity to the structure in question. Figure 9.18 illustrates how descents are captured by a sensor from two locations of different distance. At location p_0 the sensor is too far away such that the summit of the descent obstructs the characteristics of the terrain further ahead. Therefore, the conformation may be (mis-)classified as a negative step or cliff. The minimal distance from which the terrain profile can be evaluated is given in Equation 9.1. For robots which can negotiate severe slopes, the distance at which the true nature of a negative structure becomes apparent may be too short in order to turn away from the hindrance without manoeuvring. This is in particular a problem for non-holonomous platforms which are often deployed for off-road applications.

$$d = \frac{h}{\tan(\alpha)} \tag{9.1}$$

Ground profiles which feature high-centring threats represent a further class of obstacles (see Figure 9.19). Except for obstacle class d), all types of hindrances discussed so far may be an evidence for a high-centring threat. Even though the primary obstacle parameters indicate that the structure is traversable, unfavourable combinations of certain properties result in a non-traversable ground structure.

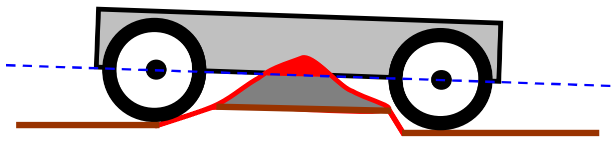


Figure 9.19: Ground bumps which may be traversable regarding height and slope can nonetheless represent a threat to the robot as the vehicle might get high-centred.

The considerations from above indicate that harsh natural environments feature smooth transitions between open terrain and obstacle structures. Whether an object represents a non-traversable hindrance or not does not only depend on the properties of objects, but also on the vehicle parameters. This fact renders a clear traversability decision impossible at the sensor processing level. In case certain object parameters – e. g. object height – surpass vehicle-dependent limits, the object can definitely be classified as an obstacle. Otherwise, the object may be traversable at low velocity or a tactile manoeuvre attempt may be conducted depending on the current driving situation and the capabilities of the vehicle at hand. To prevent premature decisions, the obstacle detection mechanism presented here represents object properties in a natural way and defers the evaluation of the detected properties to the semantic abstraction layer (see Section 3.3.2). During semantic abstraction, the terrain properties are analysed for the platform at hand in a tailored fashion. On the semantic level, the obstacle detection algorithm presented here can thus be regarded independent of the target platform.

Analysis of individual Vertical Terrain Sections

As already alluded above, the ground cannot be assumed flat in off-road environments. Severe variation of slope and ground-near obstacle structures have to be anticipated. The determination of the load-bearing surface in terms of a ground reference relative to which obstacles can be detected is therefore most crucial. For the assessment of hindrances – and hence the decision whether certain regions are traversable or not – height and distance to ground of such objects play a central role. Bumpy ground results in shocks and permanent change in 3D orientation of the vehicle. These have a direct impact on a 3D LRF and render pose estimation imprecise. As the environment is to be monitored by the scanner in a continuous fashion, the precise registration of raw sensor data over time is rather unfavourable. The computational power of mobile systems is furthermore strictly limited, such that the evaluation of large point clouds appears too expensive. Balancing the reasons mentioned above, the deployment of a 3D LRF which is built up from a y-axis actuated

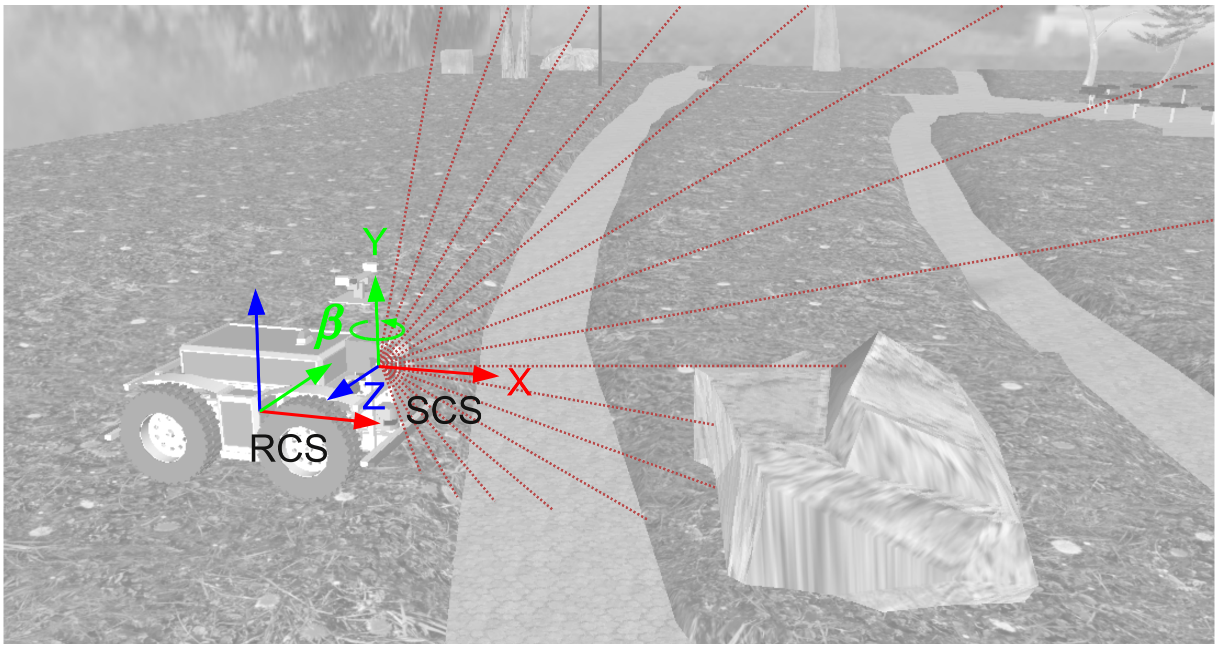


Figure 9.20: A 2D LRF is panned around the y-axis with pan angle β covering an angle of vision of $\pm 60°$.

2D LRF operated in pan mode was chosen as a sound tradeoff (see Figure 9.20). The scanning plane is thus orthogonal to the x-y-plane of the robot coordinate system. As long as the vehicle is not in extreme pitch or roll conditions (due to severe ground bumps or surpassing small objects like rocks or fallen tree trunks), this plane captures a longitudinal ground reference. Each individual 2D scan thus yields a vertical section of the terrain which contains samples representing ground and potential obstacle structures.

As already stated above, the evaluation of vertical terrain sections is a common approach towards 3D obstacle detection in the literature (see Section 9.3.2). This configuration allows to determine ground slope, degree of jaggedness, and the extent and distance of objects to the ground reference. The proposed terrain analysis algorithm consists of four major steps:

1. *Coordinate transformation, reduction, and sorting*: The polar coordinates yielded from the 2D LRF are transformed into Cartesian coordinates which reflect the sensor pan angle, as well as the vehicle's current pitch and yaw angle. In this step, the data volume is reduced by filtering irrelevant scan points. For further processing, the scan points are sorted with ascending distance to the vehicle.

2. *Ground profile estimation*: The raw ground profile is extracted from the scan by analysing the load-bearing surface.

3. *Scan point preclassification*: The scan points are classified according to their distance relative to the estimated ground profile.

4. *Determination of ground and obstacle structures*: Based on the preclassification, ground and obstacle structures are further evaluated to yield a detailed description of the environment.

(a) *Estimation of ground characteristics*: Statistical measures are computed to analyse terrain roughness and steepness.

(b) *Detection of non-traversable ground conformations*: In this step, ground-related obstacles like steps, negative obstacles, and high-centring threats are detected.

(c) *Detection of obstacle structures*: In this step, ground profile-independent hindrances like positive and overhanging obstacles are detected on the basis of clusters of scan points.

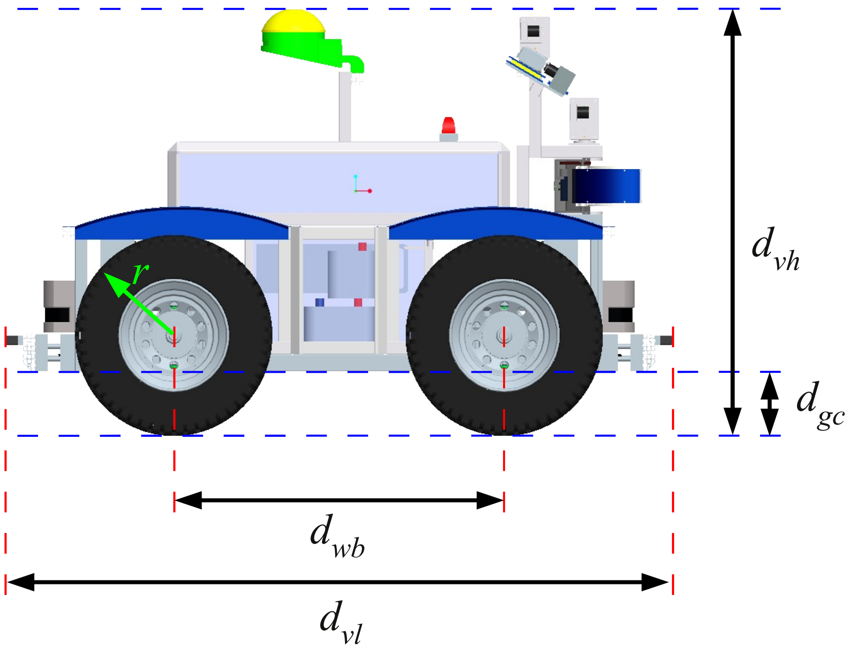


Figure 9.21: Vehicle parameters relevant for the meaningful evaluation of 3D LRF data.

As already indicated above, the environment representation created by the proposed algorithm is semantically independent of the target platform. This is a prerequisite for reuse of the detection mechanism on different robotic systems. Nonetheless, the stringent classification as required for off-road navigation cannot be made without close dependence to certain parameters of the target platform. The following vehicle parameters have to be defined in order to yield satisfactory information about the environment (see Figure 9.21):

- Vehicle length d_{vl}
- Vehicle height d_{vh}
- Ground clearance d_{gc}
- Wheel base d_{wb}
- Wheel radius r
- Maximal climbing ability α_{max}

Step 1: Coordinate Transformation, Reduction, and Sorting

As a preparative step for the obstacle detection algorithm, the polar coordinates yielded from the 2D LRF are transformed into Cartesian coordinates. In vertical terrain sections yielded from a 2D LRF, slope is in the first place relative to the scanner's x-axis. In rugged environments however, the vehicle orientation is a further crucial factor to be considered for terrain analysis. Therefore, the sensor's pan angle, as well as vehicle roll and pitch have to be taken into account such that absolute terrain steepness can be determined. For that purpose, the transformation matrix T(RCS to SCS) of the sensor coordinate system (SCS) to the robot coordinate system (RCS) and the rotation matrix R(Roll and Pitch) of the vehicle's current roll and pitch are computed. Multiplication of each coordinate $(x_{SCS}, y_{SCS})^T$ with these matrices yields the transformed coordinates $(x_{RCS}, y_{RCS}, z_{RCS})^T$ in the RCS.

$$\begin{pmatrix} x_{RCS} \\ y_{RCS} \\ z_{RCS} \end{pmatrix} = R(\text{Roll and Pitch}) \cdot T(\text{RCS to SCS}) \cdot \begin{pmatrix} x_{SCS} \\ y_{SCS} \\ 0 \end{pmatrix} \tag{9.2}$$

Note that vehicle roll is taken into account in order to make more accurate estimates about object heights. The third dimension introduced in this step is neglected in the further evaluation as extreme roll and pitch angles may violate some of the assumptions of the presented detection mechanisms. Despite this weakness, the algorithm has proven very robust towards variability in rugged terrain. In order to aggregate the environment information yielded by this algorithm into a short-term memory, the coordinates will later be transformed into the robot's working coordinate system (WCS). For simplicity reasons, the following steps will be explained in the basic coordinate directions of the SCS. Actually the point $p = (x, y)$ in the algorithms are composed as follows: $(x, y) = (x_{RCS}, z_{RCS})$.

Sensor processing is computationally quite expensive. For that reason, the input data is filtered to limit the following evaluation steps to relevant scan points. Common LRF have a defined maximum sensor range which is also returned in case no object is within this range. These samples are only of use for obstacle detection in the sense that they represent void regions in the scan. This property is documented with the special label *infinity*. At greater distances the resolution of common LRF is too coarse for a detailed terrain analysis. Therefore, the maximum evaluation range can be limited tremendously. Samples which lie beyond a certain distance are marked as *irrelevant* and are only considered to figure out the tendency after the last ground representative (e. g. flat terrain, positive, or negative obstacle). Most of the subsequent processing steps expect the scan points to be sorted ascending with distance to the vehicle. Nonetheless, the original order of the samples is also required at some points. Therefore, a sorted view of the scan is created before proceeding to the next step.

Step 2: Ground Profile Estimation

In harsh terrain, the ground is mostly jagged and covered with vegetation. There may be indentations, trenches, or cliffs which abruptly terminate the drivable terrain. Protruding objects and overhanging entities may block passages. In order to estimate a region's degree of traversability, the load-bearing surface has to be determined. After that, obstacle structures can be judged relative to the load-bearing surface.

The part of the load-bearing surface with which the wheels of the robot may interact – in essence those parts which are in principle traversable – shall be named *ground profile* in the following. Areas featuring extreme slopes or obstacles are thus excluded by definition.

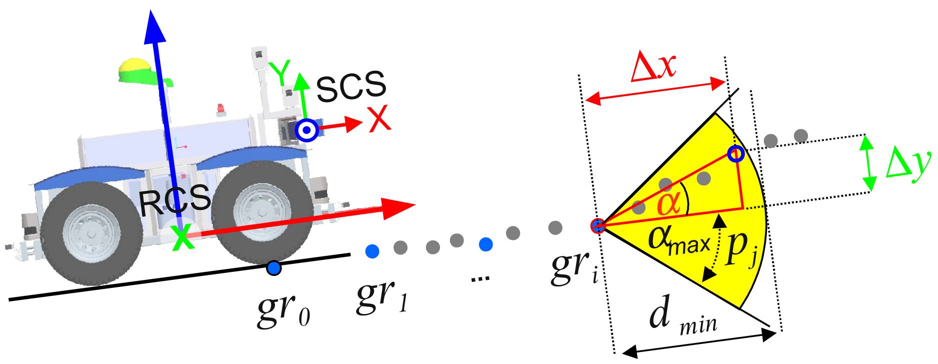


Figure 9.22: Load-bearing surface analysis used to determine the true ground profile in an iterative fashion. In the depicted processing step i, candidate point p_j is analysed whether it fulfils the properties to become the next ground representative gr_{i+1}.

In vertical terrain sections, subsequent scan points with minor slope variation characterise the ground profile. As long as the vegetation is not too dense, samples originating from penetrating laser beams permit the approximation of the ground profile by iterative determination of *ground representatives* gr_i as described in Equation Set 9.3 with support of Figure 9.22.

let SP the set of scan points representing a vertical terrain section (9.3)

let $p = (p_x, p_y), \quad q = (q_x, q_y), \quad gr_i = (gr_{ix}, gr_{iy})$

let $\alpha(q,p) = \tan\left(\frac{p_y - q_y}{p_x - q_x}\right)$

the slope from point q towards q

let $x_{\min}(M) = p$ with $p_x \leq b_x \ \forall\ b \in M$

the point p with the smallest x-value in M

let $\alpha_{\min}(M,q) = p$ with $\alpha(q,p) \leq \alpha(q,p) \ \forall\ b \in M$

the point p with the smallest slope from q

let $y_{\min}(M) = p$ with $p_y \leq b_y \ \forall\ b \in M$ the point p with the smallest y-value in M

let $x_{\max}(p,q) = \begin{cases} p, & \text{if } p_x > q_x \\ q, & \text{otherwise} \end{cases}$ the point with the larger x value

let $M_i = \{p \in SP \mid gr_{ix} < p_x \wedge |\alpha(p, gr_i)| < \alpha_{\max}\}$

the set of points following the last ground point gr_i within the slope range $\pm\alpha_{max}$

let $C_i = \{p \in M_i \mid p_x \leq \max(gr_{ix} + d_{\min}, x_{\min}(M_i))\}$

the set of points in the distance range d_{min}

then the subsequent ground point gr_{i+1} is defined as the point with the smallest y-value or the smallest slope value which is most distant from gr_i:

$$gr_{i+1} = x_{\max}\left(\alpha_{\min}(C_i, gr_i), y_{\min}(C_i)\right)$$

As a ground reference, the point gr_0 residing beneath the front wheels of the robot – which are assumed to have ground contact – is introduced. Starting from the predetermined

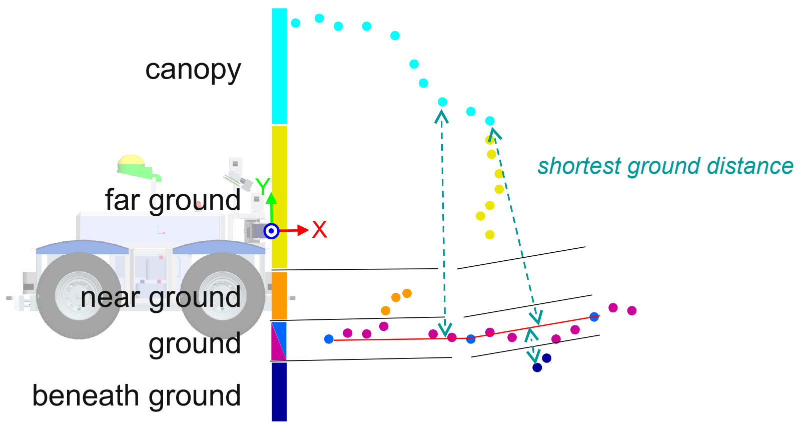


Figure 9.23: Ground distance as classification criterion. Associated ground points are depicted in purple to contrast the ground representatives which are highlighted in light blue colour.

ground representative gr_0, all following representatives are identified in relation to their respective predecessor. Based on the current representative gr_i, the points with greater x-distance within the configurable climbing limit $\alpha_{\max}$ are considered (M_i). The candidate set C_i is filled with points from M_i within search distance $d_{\min}$. If no candidate is in this interval, the search continues until a suitable point is found or no points remain in M_i. From the set of candidates C_i, the point with the least slope and the point with the smallest y-value are identified. The point with the largest x-distance then becomes the following ground representative. That way a uniform distribution of ground representatives is achieved.

Step 3: Scan Point Preclassification

The detectability of ground representatives is the necessary condition for traversability of the terrain segment in question. For sufficiency, the absence of hindrances has to be assured. In order to obtain further information on ground and obstacle structures, the distance of the remaining scan points to the estimated ground profile is used as major criterion. In case this distance is small, the scan point represents the ground profile. Otherwise, the scan point is potentially part of an obstacle. Under the assumption that the local terrain profile lies approximately in the plane that is defined by the contact points of the robot's wheels, deviations from this plane can be regarded as a measure for terrain jaggedness. Transferred to an individual terrain section yielded from a particular scan, the local terrain profile corresponds to the line segments connecting subsequent ground representatives. Based on the distance to these lines, the remaining points are assigned to five classes. Figure 9.23 illustrates the classification by example of a typical vertical terrain section.

For that purpose, the linear equation for each pair of subsequent ground representatives is computed as a segment-wise approximation of the ground profile. All scan points in the interval determined by the orthogonal lines intersecting the x-values of the ground representatives are then analysed as to their distance to the ground profile approximation line. In case the absolute distance to the line is smaller than a threshold defining common terrain roughness, the scan point is classified as *(associated) ground* point (purple). Points labelled as *ground* (i. e. *ground representatives* (light blue) and *associated ground* (purple)) mean no harm to the vehicle but are used to estimate terrain roughness and steepness in the next evaluation step. Larger deviations in negative y-direction result in a classification as *beneath ground* (dark blue). Representatives labelled as *beneath ground* potentially belong to negative obstacles. If the scan point resides high above the ground profile line, the sample is labelled as *canopy* (turquoise). In general, *canopy* points can be ignored as the robot can pass below the structure in question. Scan points with a deviation in positive y-direction which is no larger than the ground clearance of the vehicle model are classified as *near ground* (orange). All remaining points between the ground profile and class *canopy* are classified as *far ground* (yellow). Members of class *near ground* represent structures which are traversable at low speeds, whereas members of class *far ground* mark insurmountable entities.

Step 4: Determination of Ground and Obstacle Structures

The coarse ground profile estimated in Step 2 represents the reference relative to which further analysis concerning terrain characteristics and obstacle structures can be carried out. Besides the coarse preclassification of scan points on the basis of the distance to the ground profile made in Step 3, several more specific criteria for obstacle conformations are proposed in this section.

Step 4a: Estimation of Ground Characteristics

At first, the *ground* points are analysed as to ground characteristics like terrain roughness and steepness using statistical measures. For that purpose, the partial regression lines rl_i (see Figure 9.24 and Equation 9.4) of ground representatives and associated ground points are computed according to the least squares method (see Equation Set 9.5). The standard deviation of the associated ground points to the partial regression line normalised to the maximum deviation is used as a measure for jaggedness of the terrain segment in question. Furthermore, the slope of the regression lines is an estimate of the steepness of the terrain segments in scanning direction.

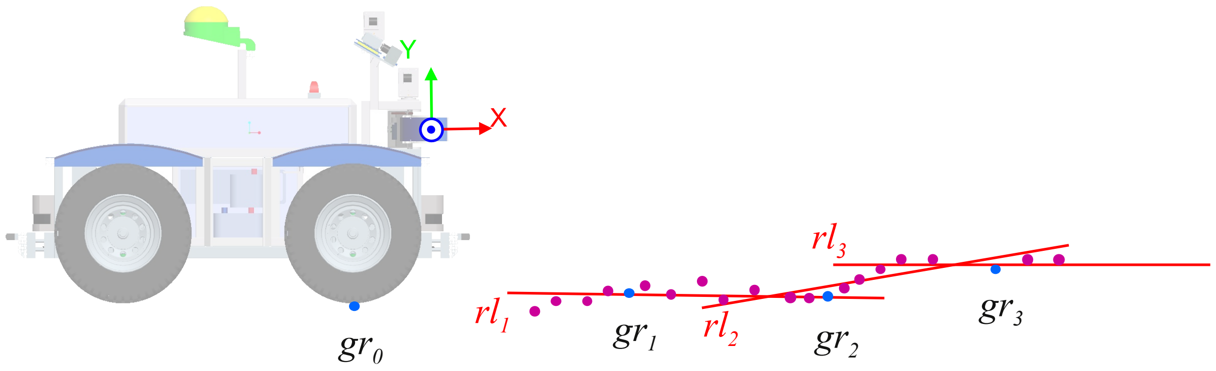
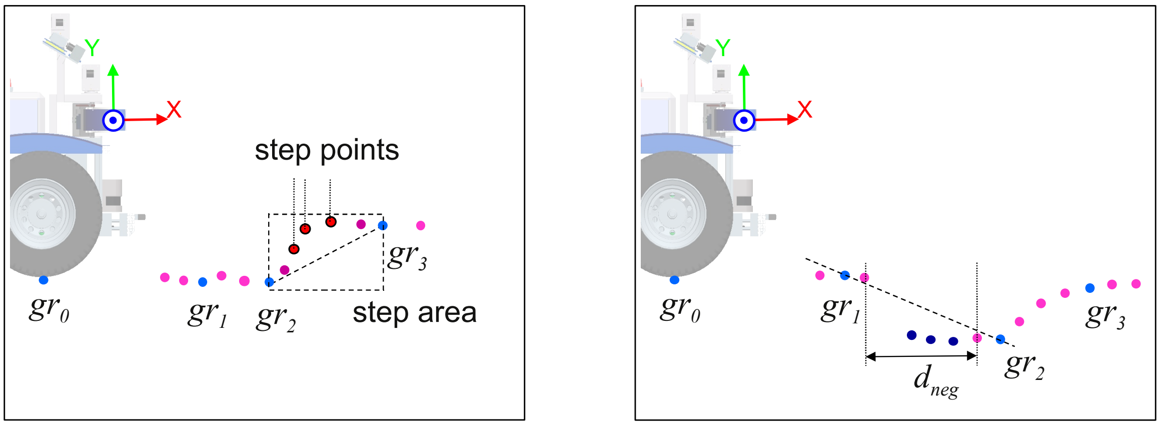


Figure 9.24: The mean steepness of the ground profile is computed section-wise as the slope of the partial regression lines rl_i defined by ground representatives gr_i and associated ground points (highlighted with purple).



(a) Step on a steep terrain segment. **(b)** Non-traversable negative obstacle.

Figure 9.25: In off-road environments, the ground profile has to be analysed carefully as to non-traversable ground conformations like steps (a) and negative obstacles (b).

$$y(x) = y_0 + m \cdot x \tag{9.4}$$

where

$$m = \frac{\sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n} (x_i - \bar{x})^2} \tag{9.5}$$

$$y_0 = \bar{y} - m \cdot \bar{x}$$

$$\bar{x} = \frac{1}{n} \sum_{i=1}^{n} x_i \text{ and } \bar{y} = \frac{1}{n} \sum_{i=1}^{n} y_i$$

Step 4b: Detection of Non-traversable Ground Conformations

After the determination of general terrain parameters like roughness and steepness, ground conformations which may represent a threat to the robot shall be detected. This comprises in particular *steps*, *negative obstacles*, and *high-centring threats*.

Steps can occur in steep regions between subsequent ground representatives. By definition, the line connecting two subsequent ground representatives can never exceed the specified maximum climbing ability parameter. Yet the distant representative may have been chosen such that the steepness is close to the maximum climbing ability and an additional ground bump resides between the representative pair as depicted in Figure 9.25a. All points residing in the triangle above the line segment defined by the two representatives which have not been classified as *ground* or *near ground* are relabelled as *step*. The close *ground representative* is further annotated to indicate that it is followed by a step conformation.

Terrain segments which feature points *beneath ground* can only be crossed in case these regions are small enough to be bridged by the wheels of the vehicle. Regions of larger extents represent insurmountable negative obstacles (see Figure 9.25b). As already mentioned above, negative obstacles and uncritical summits are difficult to distinguish from the distance. Besides the dimensions of potential negative obstacles, the length of regions without valid ground points are also measured. Larger void regions in the proximity of the vehicle have to be approached with care, since further obstacles may be hidden behind a summit.

For the detection of high-centring threats it is assumed that *ground* points represent the load-bearing surface and do not give way in noteworthy dimensions. The proposed procedure furthermore acts as if the vehicle traverses the captured terrain section in x-direction of the scan with the wheels of one side residing on the extracted ground profile.

This simplification is owed to the decision to examine individual scans in a self-contained fashion. On the basis of these assumptions, a simple vehicle model can be used to "simulate" how the robot passes through the environment ahead (see Figure 9.26). The model consists of two parallel line segments, namely the wheel base l_{wb} and the vehicle base l_{vb}. The length d_{wb} of the line segment representing the wheel base is limited by the two ground contact points of the wheels. The line segment representing the vehicle base has the length of the vehicle d_{vl}. The distance between the two line segments is determined by the vehicle's ground clearance d_{gc}. This vehicle model is now moved along the point sequence defining the ground profile such that the wheel contact points touch the estimated ground level. That way, a kind of "simulated" pass through the terrain is carried out.

In the case that ground points are found to reside above the line segment l_{vb} representing the vehicle base, a high-centring threat is registered. The vehicle may further be high-centred if the line g_h connecting the first ground point of the interval in question and the next ground point outside the interval intersects the vehicle base l_{vb}.

In vegetated terrain, flexible ground vegetation may invalidate the rigidity assumption made in the proposed approach. In order to improve the load-bearing surface estimates, the difference of a priori estimated and true vehicle orientation while passing over the terrain can be used.

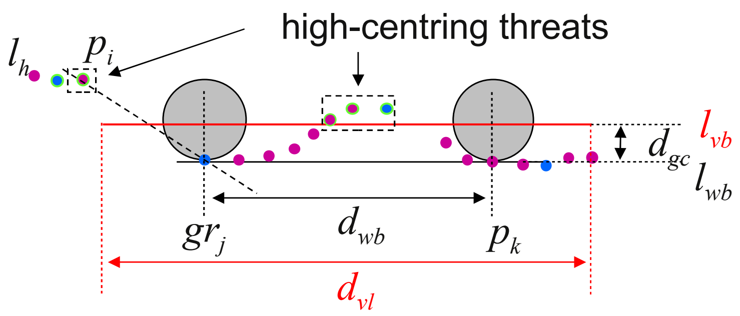


Figure 9.26: Ground conformations which represent a high-centring threat are detected in a "simulated" traversability analysis.

Step 4c: Clustering of Obstacle Structures

In the previous evaluation steps, individual points have been classified according to distance to ground and certain characteristic obstacle conformations. Whether these points actually belong to a non-traversable hindrance or not and if so what extents and properties this obstacle structure has is not yet determined. This information shall be derived in this step by selectively clustering neighbouring representatives in dependence of the individual point classifications made so far. A cluster in that context is an aggregation of individual scan points which represent one object or conformation in the environment. Each cluster is assigned a number of attributes which are derived from its members and characterise the represented natural entity. This condensed information is later used for building up the short-term memory of the 3D LRF obstacle detection facility. The requirements for the clustering step are summarised in the following:

Cluster regions of high point density (Requirement 1) *Regions with a high density of points shall be clustered. Regions with a low density of points ought to be ignored, as these certainly represent outliers resulting from sparse vegetation.*

Account for resolution decreasing with distance (Requirement 2) *Sample density decreases with distance to the scanner due to the measurement principle. Therefore, the region to be taken into account for clustering should increase dynamically with increasing distance.*

Use ground points for annotations (Requirement 3) *Ground points do not represent hindrances but they provide valuable information on whether an object has ground contact or not. These annotations shall be used to distinguish between positive and overhanging entities.*

Prevent non-obstacle cluster connections (Requirement 4) *Regions of* far ground *points which are only connected via* ground *or* near ground *points must not be clustered as they belong to different objects. This requirement is essential because otherwise virtually all obstacle points would be merged into one cluster in case of intensely rugged or vegetated scenarios.*

Split clusters of incompatible classes (Requirement 5) *Depending on the involved classes, neighbouring regions of points belonging to different classes should be split selectively into multiple clusters. Clusters consisting of* near ground *and* far ground *points have to be split as the* near ground *part is traversable while the* far ground *part represents an obstacle. Similarly, clusters consisting of* canopy *points do not mean any harm. Clustered together with* far ground *points this information would get lost. Contrariwise, the information whether* far ground *points have contact to* canopy *or* ground *may be of interest to characterise the objects more precisely. Therefore, mixed clusters containing the mentioned class combinations are split into two clusters which are annotated with the contact information.*

In order to meet the requirements outlined above, a modified variant of the density-based clustering algorithm *DBSCAN*[8] [Ester 96] is used. Given a point cloud D, *DBSCAN* examines the set of neighbouring points $N_\varepsilon(p)$ of each point $p \in D$ within the clustering radius ε (see Equation 9.6).

$$N_\varepsilon(p) = \{q \in D | dist(p,q) \leq \varepsilon\} \tag{9.6}$$

In case the cardinality of $N_\varepsilon(p)$ is greater than the minimal cluster size MinPts, p is labelled as *Core Point* or otherwise as *Border Point*. Point q is further defined as being *directly density-reachable* from a second point p, if p is a *Core Point* and q is in the neighbourhood of p (see Equations 9.7 and 9.8). If a pair of points meets both conditions, these points belong to the same cluster.

$$q \in N_\varepsilon(p) \tag{9.7}$$

$$|N_\varepsilon(p)| > \text{MinPts} \tag{9.8}$$

Two points are defined *density-reachable*, if a connecting path of pairwise *directly density-reachable* points exists between both points. A cluster thus consists of a number of *Core Points* which are *density-reachable* among each other (transitive hull of *density-reachable* points) and a set of *Border Points* which are *directly density-reachable* from one of the core points. The *DBSCAN* clustering procedure is exemplified in Figure 9.27.

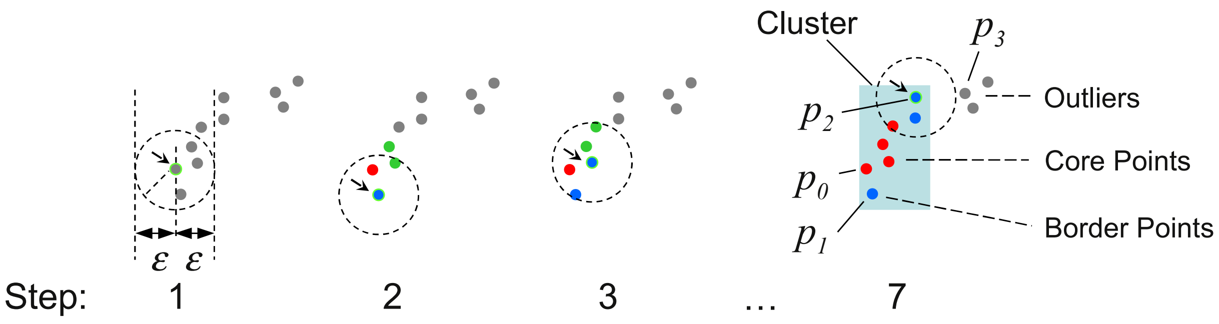


Figure 9.27: *DBSCAN* example (let MinPts = 3): Starting from a particular point p_i (indicated by the arrows), the neighbourhood $N_\varepsilon(p_i)$ is examined. If $|N_\varepsilon(p_i)| > \text{MinPts}$ holds, p_i is labelled a *Core Point* (e. g. Step 1 → 2). Otherwise the point is labelled as *Border Point* (e. g. Step 2 → 3). In Step 7 the clustering is complete yielding one cluster and a set of three outliers: p_1 is *directly density-reachable* from p_0 but not vice versa. Furthermore, p_0 and p_2 are *density-reachable* but p_3 is not.

DBSCAN meets Requirement 1 (Cluster regions of high point density) out of the box, as resulting clusters contain at least MinPts elements and regions of high point density are clustered together.

In order to meet Requirement 2 (Account for resolution decreasing with distance) the neighbourhood range ε is dynamically modified proportional to the distance between the sample in question and the neighbouring laser beam. The neighbourhood range ε is further limited by an upper and a lower boundary to prevent under and over segmentation.

[8]Density Based Spatial Clustering of Applications with Noise (DBSCAN)

let $p = (\psi, r)$ the sample in question in polar notation

let $\Delta\psi$ the angular resolution of the scanner

then the distance d between p and the neighbouring laser beam is defined as

$$d = \tan(\Delta\psi) \cdot r$$

as depicted in Figure 9.28.

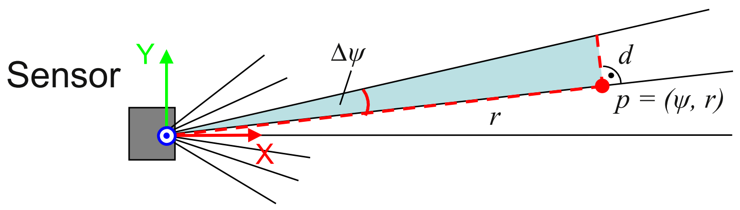


Figure 9.28: Distance d between two laser beams in dependence of distance r to the sensor.

Scan points which have been labelled *infinity* or *irrelevant* in Step 1 of the obstacle detection algorithm are not considered for clustering. Furthermore, *ground* points which do not represent an obstacle may be added to a cluster but are prevented from becoming a *Core Point*. That way, the extension of a cluster starting from a *ground* point is excluded but the connection of the cluster to the load-bearing surface is documented. This extension fulfils Requirement 3 (Use ground points for annotations).

Requirement 4 (Prevent non-obstacle cluster connections) further states that *far ground* points must not be clustered if they are only connected via *near ground* points. This can be avoided by clustering the scan in two iterations. In the first pass the *near ground* points are not allowed to become *core points*. These points can thus be associated to clusters but there is no way that a connection of *near ground* points exists over which members of otherwise independent clusters become *density-reachable*. In a second pass the remaining *near ground* points are assigned to clusters. This procedure assures that *near ground* points, which may be traversable under special precautions, are not mixed up with obstacle representatives.

The distinction of neighbouring points of different classes is also relevant for combinations of *far ground* and *canopy* points. Since *canopy* points do not represent a threat to the robot, it is not sensible to combine them with a cluster of *far ground* points which really represents an obstacle. The cluster would be perceived a lot larger than it actually is with respect to its impact on traversability. On the other hand, information on whether the *far ground* obstacle has contact to *canopy* might be of interest for further classification. Therefore, clusters consisting purely of *canopy* points may not be tainted with points of other classes. In contrast, clusters of *far ground* points may contain *canopy* points but only in the role of *border points* to prevent the extension of the cluster towards *canopy*.

The result of the classification algorithm of one scan is outlined in Figure 9.29. One cluster of *near-ground* points (a) was detected in the proximity of the robot. This area should be traversed with care to prevent damage to the robot. The cluster of *far ground* points (b) with contact to *ground* and *canopy*, identified several metres before the robot, represents

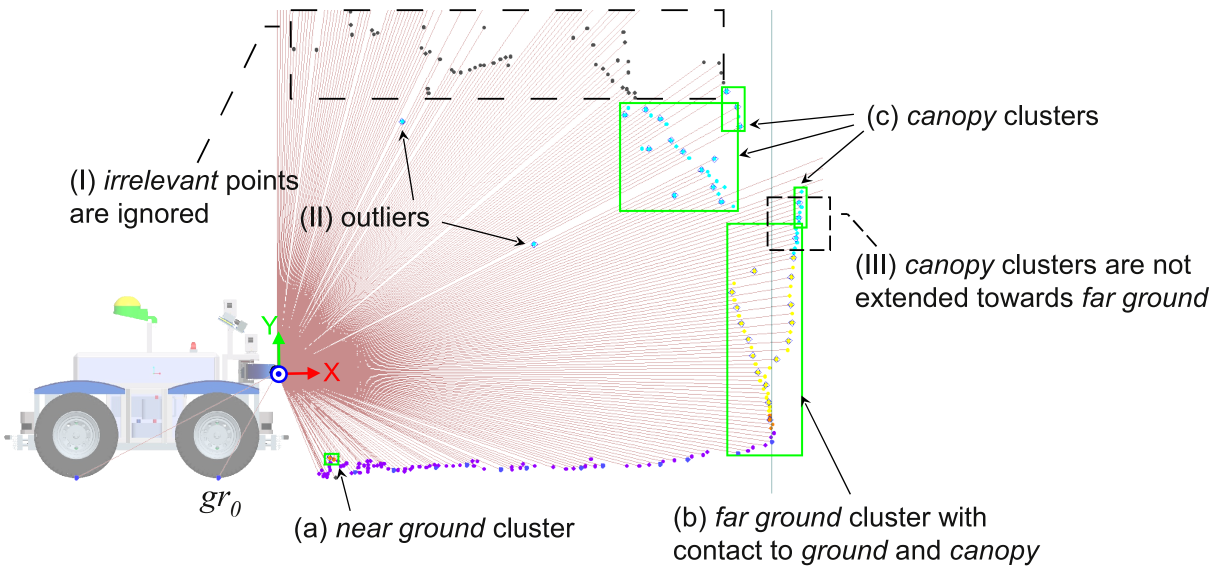

(I) *irrelevant* points are ignored
(II) outliers
(c) *canopy* clusters
(III) *canopy* clusters are not extended towards *far ground*
Y
X
gr_0
(a) *near ground* cluster
(b) *far ground* cluster with contact to *ground* and *canopy*

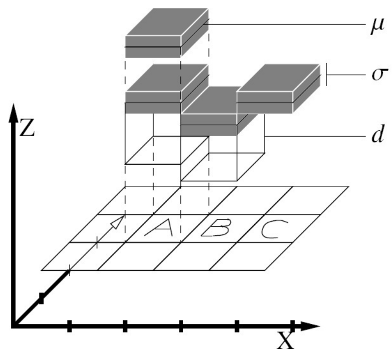
Figure 9.29: Classification output of the proposed algorithm. The colour coding is the same as introduced in Figure 9.23. Clusters are further highlighted with green bounding boxes.

an insurmountable *positive obstacle*. In contrast, the three *canopy* clusters (c) do not mean any harm to the vehicle as the robot can pass beneath them. As *canopy* points in clusters of other classes are always labelled *Border Points*, the *far ground* cluster is not extended towards *canopy* and vice versa (III). Points classified as *irrelevant* were not considered during the clustering process (I). The two isolated *canopy* points (II) were not clustered as the point density in the neighbourhood was too low. These points can thus be regarded as outliers which probably result from tiny objects like thin branches or foliage.

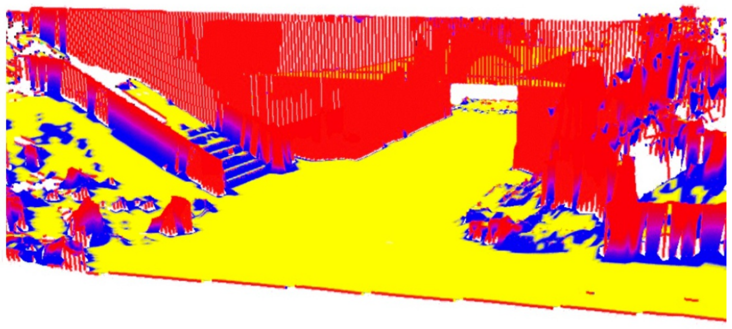
The proposed scene analysis approach has proven suitable for navigation in harsh environments [Schäfer 08a]. In numerous experiments and during participation in several competitions (e. g. SICK Robot Day 2007, European Land Robot Trial 2007 through 2010), one major short-coming of the evaluation system was identified: when the robot approaches a ramp in a straight fashion, the ramp and the area besides the ramp are considered as ground. Therefore, RAVON might pass over the ramp border resulting in an emergency stop due to an extreme roll angle. This is not fatal, yet undesirable. This problem was tackled by mounting two additional vertical LRF to either side of the robot with crossing fields of vision (see Figure 7.4 on page 78 in Chapter 7). These LRF are fixedly mounted and are evaluated with the same algorithms as the data from the 3D LRF. That way, ramps can reliably be detected in whatever orientation the vehicle approaches the hindrance. All obstacle information extracted from these three LRF is stored in the same representation which shall be introduced in the following.

9.3.4 Representation of 3D Environment Information

The 3D LRF continuously captures 2D sections of the terrain in front of the robot and analyses the data as to ground and obstacle structures. Instantaneous information yielded from a single scan without context is of little value for the control system. In order to profit from the 3D coverage of the sensor, environment information has to be aggregated over time to yield a representation of the vicinity around the robot. This representation can then be used to render suitable *Views* for the *Behaviours* of the control layers.



(a) Schematic of the Multi-level Surface Map.



(b) Multi-level Surface Map annotated with traversability information.

Figure 9.30: Multi-level Surface Map as proposed in [Triebel 06].

As for other detection facilities, the short-term memory for the 3D LRF evaluation shall be realised as a spatially limited, scrolling grid map with tailored content. In contrast to the tactile sensors and the planar LRF discussed in earlier sections, the 3D LRF yields precise height information which cannot be captured by the representations introduced so far. Therefore, suitable extensions have to be created, which store and manipulate additional information.

State of the Art in 3D Representation

In outdoor robotics, probabilistic elevation maps which merely store height and confidence values at a given location are widespread [Fong 03, Yu 06]. These have the disadvantage that overhanging objects are difficult to represent. In natural environments, this limitation requires premature simplifications of the world model which may impair the utility of the representation in later applications. On the other hand, full-featured 3D representations – as the 3D evidence grids proposed in [Moravec 96] – require a lot of storage and have a rather inefficient accessability. From a local point of view, the vehicle can only move in two dimensions. The third dimension is given by the ground profile. Therefore, a $2\frac{1}{2}$ D map appears more appropriate as it can capture all required information and is more efficient for data retrieval than a full 3D representation. In [Hong 00] a complex grid-based world model is developed which is designed to strike a balance between storage requirements and expressiveness. Each cell of the grid map stores general terrain parameters and manages a linked list of objects which characterise the represented location.

Similarly, [Triebel 06] proposes a specialised grid map, which may store an arbitrary number of surface patches per cell. The so called multi-level surface map is designed for mapping structured outdoor environments. As illustrated in the schematic in Figure 9.30a, each surface patch is defined by the mean μ and the variance σ of the measured height d. Measurements of similar height are combined to surface patches which are stored in the corresponding grid map cells. In case the height difference of the measurement samples is small, the patch is classified as a horizontal surface with height $d = 0$ (see cell C Figure 9.30a). If the samples are more distant from one an other, the patch is classified as a vertical object. The mean height μ is in this case computed from the highest points which represent the top surface of the element. The element height d is the distance between the lowest and the highest point belonging to the surface patch (see cell B Figure 9.30a). This type of representation has a compact memory footprint while allowing for the registration of overhanging objects on multiple levels (see cell A in Figure 9.30a). Figure 9.30b illustrates a multi-level surface map yielded from the mapping algorithm presented in [Triebel 06]. Vertical objects represent obstacles and have been highlighted with red colour. Horizontal surface patches are classified as traversable (yellow), if a minimum number of neighbouring cells feature surface patches of similar height. Those surface patches around which there is not enough horizontal space of similar height are classified as non-traversable (blue).

Extension of the Representation Scheme

The representation scheme proposed in this work in principle supports (scrolling) grid maps in 3D. Yet, as already mentioned above, data retrieval becomes computationally a lot more expensive with the additional dimension. Furthermore, accurate height estimation is crucial for full 3D representations which is difficult to achieve on the move in cluttered terrain. For these reasons, the short-term memory for the 3D laser obstacle detection shall be designed as a scrolling $2\frac{1}{2}$D grid map. In each content element, ground characteristics and objects at the location in question have to be represented. In consequence, several objects of different type and extent are to be stored. Even though ground structures do not represent hindrances for the vehicle, they provide vital information on whether a load-bearing surface was detected and what conditions are to be expected when traversing a patch of terrain. As a basic configuration, the combination of `ContentBase`

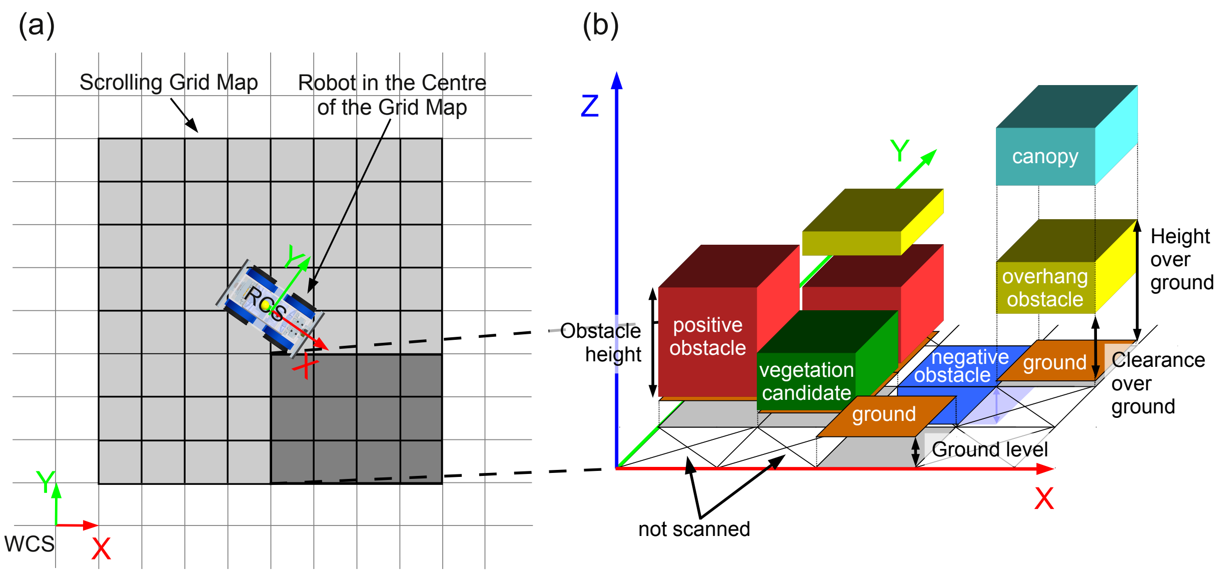


Figure 9.31: The $2\frac{1}{2}$D grid map is inspired by the multi-layer surface maps introduced in [Triebel 06].

and `ProbabilisticContent` – already used to model the short-term memories above – shall be deployed (*Content Reuse* (Guideline 1 on page 67)). The *PROP* interface of the `ContentBase` accounts for the immense semantic bandwidth of off-road environments and guarantees extensibility. Furthermore, the generic semantic translation and the probabilistic infrastructure can be reused (*Handler Reuse* (Guideline 1 on page 67)). In order to benefit from the 3D information yielded by the evaluation algorithm, an independent and minimalistic set of extensions to the schemata has to be designed (*Extension Independence* (Guideline 14 on page 69), *Minimalistic Extensions* (Guideline 13 on page 69)). Inspired by the multi-level surface map, a `Content Extension` named `LeveledContent` is designed to model object heights. In contrast to [Triebel 06], level information shall be linked to generic properties rather than to specialised object types. Figure 9.31 illustrates the extension of the representation scheme. As a ground reference, the `LeveledContent` extension stores the `Ground level` with respect to the X-Y-plane defined by the WCS. In order to represent `Object height`, `Clearance over ground` and `Height over ground` relative to this `Ground level` are stored for each object of property $prop \in PROP$. That way, a constant memory footprint and constant access times to environment information can be assured. As no list management is required, the representation also remains compact in memory which has certain advantageous properties for the communication between components. Note that this design decision implies that only one object per property can be stored. This minor limitation can be compensated by a selective definition of the *PROP* interface according to *Property Selectivity* (Guideline 12 on page 69):

$$\begin{aligned} laser3dPROP = (&negative,\ overhanging,\ highCentring,\ positive,\ step,\\ &nearGround,\ ground,\ canopy,\ vegetation) \end{aligned}$$

where

$$negative(i) = \begin{cases} true, & \text{holes or trenches} \\ false & \text{otherwise} \end{cases}$$

$$overhanging(i) = \begin{cases} true, & \text{objects without ground contact} \\ false & \text{otherwise} \end{cases}$$

$$highCentring(i) = \begin{cases} true, & \text{objects which represent a high-centring threat} \\ false & \text{otherwise} \end{cases}$$

$$positive(i) = \begin{cases} true, & \text{protruding objects (e. g. a rock or a tree)} \\ false & \text{otherwise} \end{cases}$$

$$step(i) = \begin{cases} true, & \text{positive or negative step} \\ & \text{on which the robot might get high-centred} \\ false & \text{otherwise} \end{cases}$$

$$groundClutter(i) = \begin{cases} true, & \text{objects close to the ground} \\ & \text{which may be negotiable at low velocities} \\ false & \text{otherwise} \end{cases}$$

$$ground(i) = \begin{cases} true, & \text{ground reference was detected} \\ false & \text{otherwise} \end{cases}$$

$$canopy(i) = \begin{cases} true, & \text{overhanging objects which reside} \\ & \text{high above the ground reference} \\ false & \text{otherwise} \end{cases}$$

$$vegetation(i) = \begin{cases} true, & \text{objects detected in content element } i \\ & \text{may represent flexible vegetation} \\ false & \text{otherwise} \end{cases}$$

Note that in particular large property sets as the one defined here should adhere to *Natural Naming* (Guideline 11 on page 69) to support semantic clarity. The semantic specification of the short-term memory in terms of the property set *laser3dPROP* barely leaves room for multiple objects per class. The only property for which multiple objects at one location are possible is class *overhanging*. For navigational purposes however it is sufficient to know the outer boundaries of overhanging entities. In case more detailed information is required, additional properties may be defined. The representation scheme in principle also allows for the realisation of a solution deploying lists with all the implicated advantages and disadvantages of lists.

Figure 9.32a shows RAVON in the boundary fence patrol scenario from Section 9.2. The supplementary height-level information provided by the `LeveledContent` extension is processed by a handler extension for the display facilities to generate a 3D visualisation of the short-term memory as illustrated in Figure 9.32b. Note that the standard visualisation used for all other grid-based short-term memories so far can be deployed to create a view from above the robot in a transparent fashion (see Figure 9.32c).

Extension of the Translation Scheme

In order to benefit from the supplementary height level information, semantic translation is extended by handler `LeveledSemanticTranslation` which couples into step `Quantification` of the translation procedure (see Figure 9.33). The `Aspect-oriented Configuration` is extended with height levels which are specified in terms of height intervals:

let $Levels = (level_1, \ldots, level_n)$ a tuple of n height levels defined by the designer

let $[lowerBound_{level}, upperBound_{level}]$ the interval of relevance of $level \in Levels$

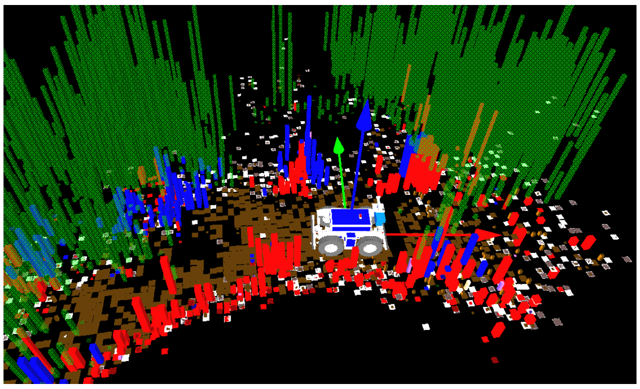
Let further for each content element i

$clearance(i, prop)$ the clearance over ground of property $prop \in srcPROP$ and

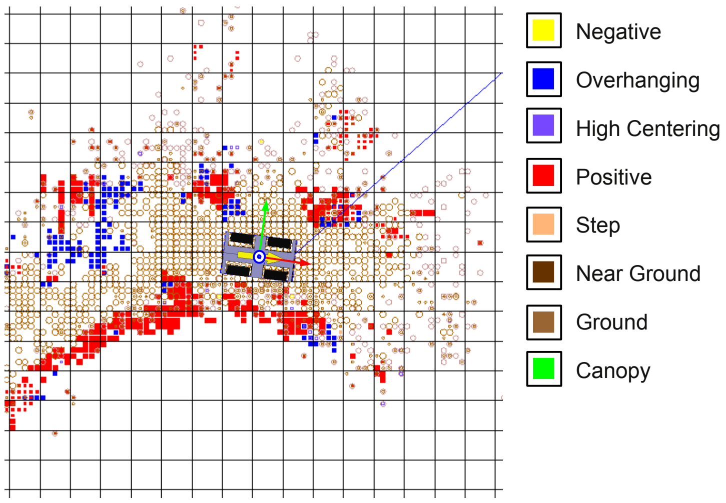
$height(i, prop)$ the height over ground of property $prop \in srcPROP$

The property function tuple $srcPROP_{rel}(i, level)$ of content element i can be defined for each height level as:

(a) Experimental Setup.



(b) 3D visualisation of the Short-term memory.



(c) 2D visualisation of the Short-term memory.

Figure 9.32: RAVON in a boundary fence patrol scenario (a) and the $2\frac{1}{2}$D map resulting from 3D LRF evaluation (b, c). For the screen shot of the 2D visualisation, property *Canopy* was blanked out to improve clarity.

$$srcPROP_{rel}(i, level) = (srcProp_{rel}(i, level)_1, \ldots, srcProp_{rel}(i, level)_k)$$

where

$$srcProp_{rel}(i, level)_k = \begin{cases} true, & srcProp_{rel}(i)_k \wedge \\ & (clearance(i, srcProp_k) \in [lowerBound_{level}, upperBound_{level}] \vee \\ & height(i, srcProp_k) \in [lowerBound_{level}, upperBound_{level}] \vee \\ & lowerBound_{level} \in [clearance(i, prop), height(i, srcProp_k)] \vee \\ & upperBound_{level} \in [clearance(i, prop), height(i, srcProp_k)]) \\ false, & otherwise \end{cases}$$

Note that this extension is inherently independent of the target application and the target platform. The height level predicates act as a filter which check whether the stored

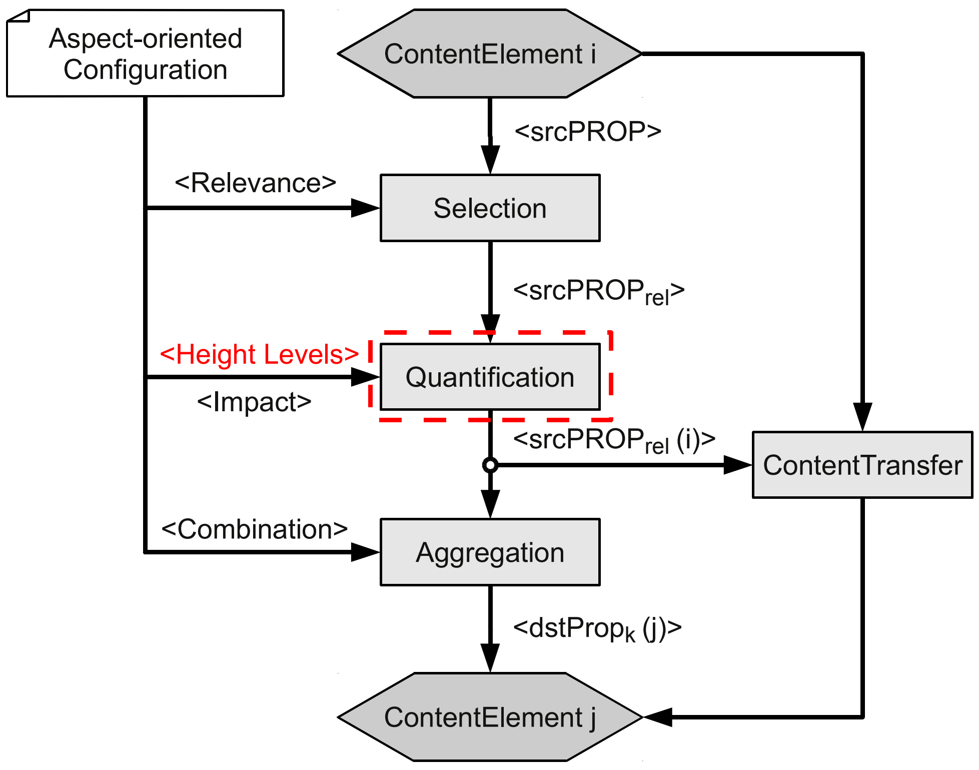


Figure 9.33: The `LeveledSemanticTranslation` handler extension couples into step `Quantification` of the translation procedure in order to determine the properties of relevance $srcPROP_{rel}$.

height interval $[clearance(i, srcProp_k), height(i, srcProp_k)]$ overlap with the height level of relevance $[lowerBound_{level}, upperBound_{level}]$. Height levels can be configured to reflect for example vehicle dimensions or any other parameter. It is even possible to specify overlapping height intervals which change over time. For the representation scheme and the semantic translation scheme this does not make any difference.

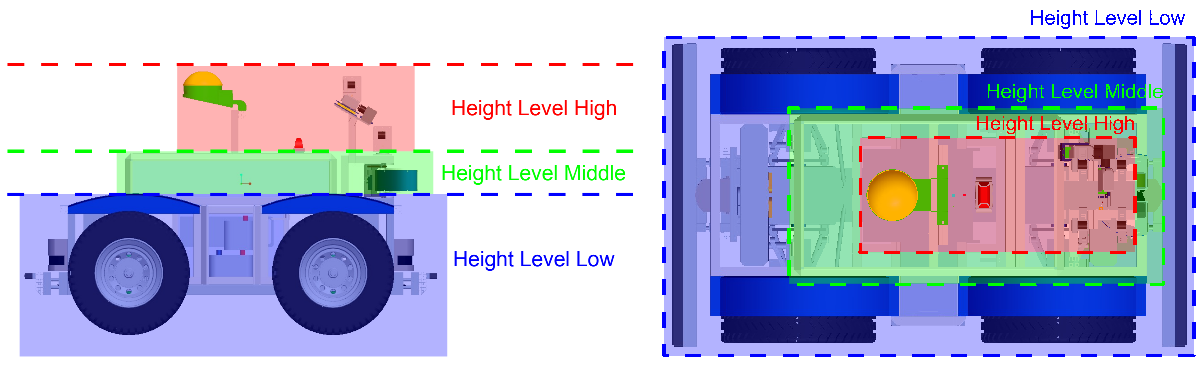
The probably most evident application of height level information in off-road navigation is to determine whether the robot may pass below overhanging obstacles or not. On RAVON, the height-dependent diminution of the chassis is further considered to approximate the robot's shape in 3D as illustrated in Figure 9.34. This approximation formally results in the following set of height levels:

$$Levels = \{low, middle, high\} \tag{9.9}$$

where

$$\begin{aligned} low &= [-\infty\,\mathrm{m}, 0.7\,\mathrm{m}] \\ middle &= [0.7\,\mathrm{m}, 1.0\,\mathrm{m}] \\ high &= [1.0\,\mathrm{m}, 1.8\,\mathrm{m}] \end{aligned}$$

To incorporate this approximation into the control system, protective *Virtual Sensors* are rendered on each individual height level. The respective generic *Behaviours* are replicated for the relevant height levels in a straightforward fashion [Schäfer 05b].



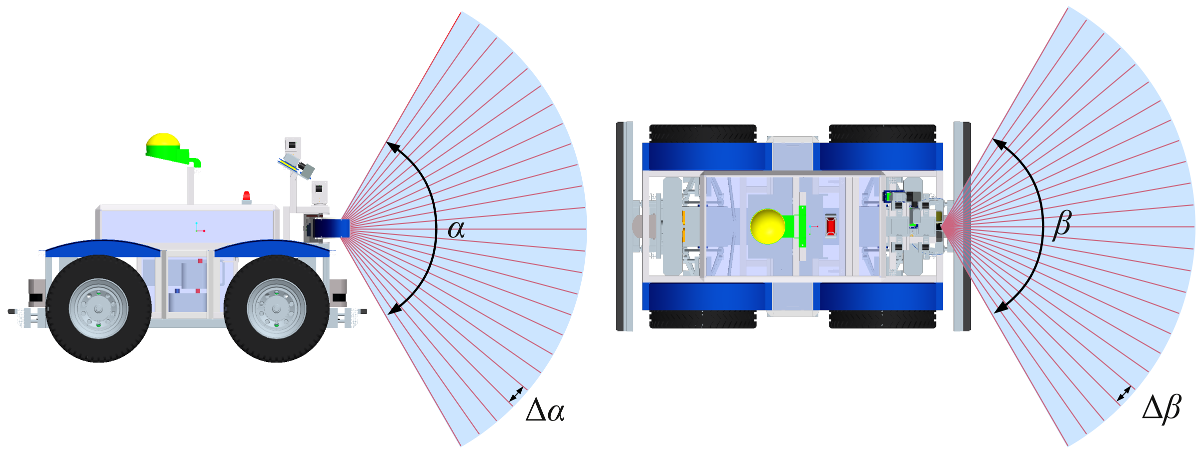
(a) Side view of RAVON. **(b)** Top view of RAVON.

Figure 9.34: Height levels can be used to approximate the robot's shape in 3D for high-precision navigation.

It is even possible to configure different approximations for different *Drive Modes*. High-precision navigation is actually limited to the *Drive Modes Moderate Velocity* and *Tactile Creep*. In *Drive Mode Maximum Velocity*, the high velocities do not allow for the negotiation of very tight situations, anyway. Therefore, a precise approximation of the robot's shape does not yield better navigational results but consumes more computational resources and may lead to additional latency. By introduction of the additional height-level $all = [-\infty\,\text{m}, 1.8\,\text{m}]$ which subsumes all heights, a specific configuration for *Drive Mode Maximum Velocity* can be achieved.

In *Drive Mode Tactile Creep*, the height information is used to selectively evade obstacles which may endanger sensitive sensor systems. On RAVON, the sensors to be protected are the 3D LRF and the stereo cameras, which are mounted at exposed positions of the robot's sensor tower, i. e. height levels *middle* and *high*. Any object detected on these height levels has to be circumnavigated in order to protect the sensor systems. Therefore, these height levels are accounted for by the *Behaviours* in the groups *Obstacle Avoidance (Tactile Creep)*, while height level *low* is widely ignored. Only highly critical obstacles like water hazards, overhanging entities, and negative obstacles are accounted for on height level *low* when in *Drive Mode Tactile Creep*. These types of obstacles are further considered by the *Emergency Stop Behaviour* group in order to guarantee safety in tight situations and to trigger the *Back Out Behaviours*.

The selective height-level-dependent configuration of the data bases for different *Drive Modes* allows for high-precision navigation in tight situations as well as fast operation in open terrain. Furthermore, the tactile negotiation of dense ground vegetation of up to 0.7 m height can be integrated in a gradual fashion (see Chapter 10 for details on the *Aspect-oriented Configuration*). In order to achieve even higher precision for the negotiation of narrow spots, the `RepresentativeContent` extension from Page 46 is pulled into the content element. *Extension Independence* (Guideline 14 on page 69) assures that this is possible without side effects.



(a) Side view of RAVON. **(b)** Top view of RAVON.

Figure 9.35: The angular resolution of the 3D LRF is dependent on the deployed 2D LRF and the actuation mechanism.

Updating the Short-term Memory

As opposed to the planar LRF, which yield new terrain data at high resolution and fast update rates, the 3D sampling of the environment takes much more time. For the 3D LRF built up in the context of this Doctoral Thesis, the vertical angular resolution $\Delta\alpha$ is identical with the angular resolution of the deployed 2D LRF. The horizontal angular resolution $\Delta\beta$ depends on the update frequency f_{2D} of the 2D LRF, as well as the panning frequency f_{3D} and the horizontal angle of vision β (see Figure 9.35):

$$\Delta\beta = \frac{\beta \cdot f_{3D}}{f_{2D}}$$

Deploying a common commercial 2D LRF in the proposed construction results in a dense horizontal angular resolution $\Delta\alpha = 0.5°$. In comparison the vertical sampling is rather sparse. Provided that the update rate of the 2D LRF is $f_{2D} = 25\,\text{Hz}$[9] and one pass over an angle of vision $\beta = 120°$ is completed after 1 s ($f_{3D} = 1\,\text{Hz}$), the horizontal angular resolution is $\Delta\beta = 4.8°$. The resulting vertical sampling gaps and the low update frequency f_{3D} thus have to be considered when updating the grid map.

During one pass of the 3D LRF, the robot may have moved several metres such that new objects may be present in the scanning area. Furthermore, dynamic objects may have moved into or out of the scanning area of the sensor. On the one hand, the dynamic parts of a scene require the update strategy to account for the quick removal of outdated objects. On the other hand, the sparse vertical angular resolution calls for a rather conservative position towards the discarding of observations, as thin vertical objects may not be detected in every pass. At this point, the decision to process vertical terrain sections individually

[9]In recent months, SICK has presented several compact LRF with update frequencies of 50 Hz and 100 Hz. The general problem described here persists even though the vertical resolution can be reduced tremendously with the new LRF generations.

for more robust operation in cluttered terrain unveils the disadvantage that no information on object membership can be obtained.

To strike a balance between both requirements, the update strategy has to account for data decay over time as well as the radial nature of the 3D range data. The closer an object to the sensor, the more likely it is detected again. Using the theory on probabilistic models from Section 4.3.1, characteristic data can be determined for the detectability of particular structures. This information can then be used to configure the decaying mechanism of the update strategy.

The representation scheme thus has to be extended with temporal information, as the update routine needs to know about the “age” of a content element. For that purpose a *Content Extension* named *AgeingContent* is designed to store a timestamp and the pass number of the panning mechanism. On every update cycle of the grid map, the cells falling between the two sampling planes are analysed as to their age and discarded if applicable. Note that the *AgeingContent* – as well as the *Extensions* introduced before – strictly adheres to guideline *Extension Independence* (Guideline 14 on page 69) to assure that combinations are side-effect-free. Apart from that, the minimalistic nature of the extensions (*Minimalistic Extensions* (Guideline 13 on page 69)) further supports *Content and Handler Reuse* (Guideline 1 on page 67).

9.4 Detection of Water Hazards

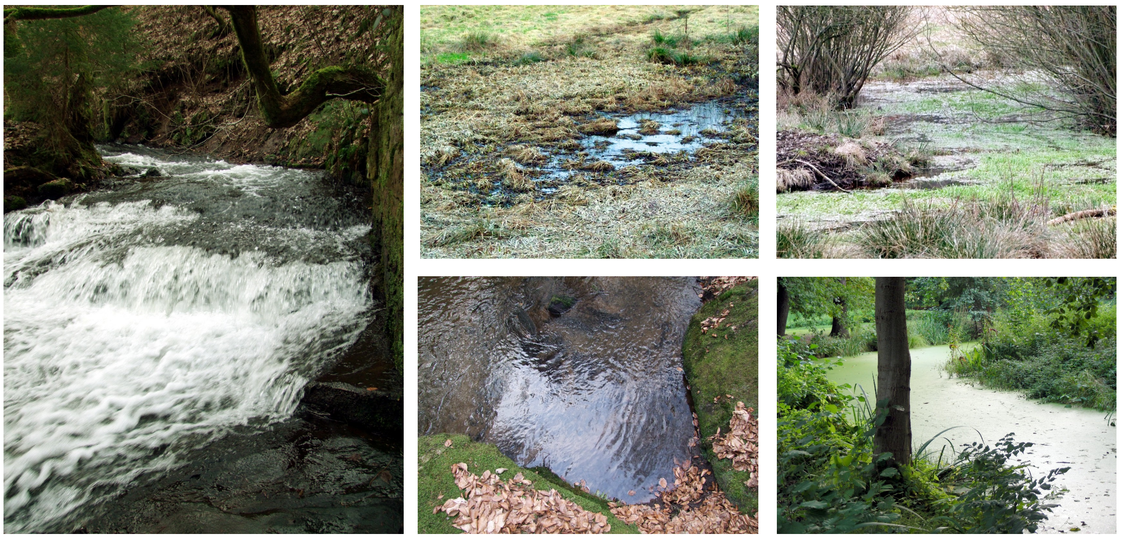


Figure 9.36: The occurrence of water bodies is highly variable and depends on lighting conditions, flow velocity, and perspective of the observer.

So far RAVON is equipped with a sound suite of sensors and obstacle detection facilities which allow for safe navigation through harsh environments. Even high vegetation can be negotiated to a certain degree (see Section 9.1).

Nonetheless, natural terrain features a wide variety of critical obstacle conformations which require further detection capabilities. One of these critical obstacle classes is water hazards. Even if a vehicle is waterproof (which is not the case for the experimental platform subject to this work), this does not mean that water does not represent a threat to the system. The robot may get stuck in mud adjacent to water surfaces or loose ground contact in water hazards of larger extents, etc. In any case, the navigational strategy has to be adapted according to the current conditions. Amphibian vehicles for example will at least have to switch the propulsion method during the transit between ground and water. Therefore, information on the location of water bodies is a crucial information for autonomous navigation in natural environments. The problem with water is that it interacts with light in an ambivalent fashion due to its reflective properties (see Figure 9.36). Depending on lighting conditions, flow velocity, and perspective, even humans sometimes may have problems to reliably recognise water hazards on the basis of camera images. The occurrence of water bodies is highly variable and the observable effects are often discarded as artifacts by common obstacle detection mechanisms. For that reason, specific water detection algorithms are required to obtain robust information on water in the vicinity of the robot. In particular visual approaches towards water detection are of interest because already the proximity of water may represent a threat to the vehicle as mentioned above. In the following section, a brief overview of visual water detection mechanisms shall be given before introducing the evaluation facilities developed in the context of this application study.

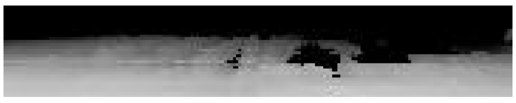
9.4.1 Approaches towards Visual Water Detection

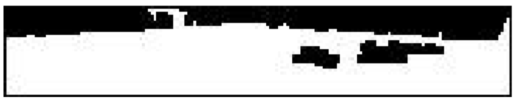
The visual detection of water bodies can roughly be divided into two major classes, namely laser-based and camera-based approaches. Laser-based approaches exploit the reflective properties of water surfaces. For camera-based water detection, many different features can be deployed. The undisputed advantage of laser-based approaches is that these in principle work at daytime as well as at night. The results may even be better at night, as lesser noise is induced by ambient light. Camera-based approaches on the other hand allow to exploit a wide variety of different cues and are inherently passive which makes them interesting for military applications.

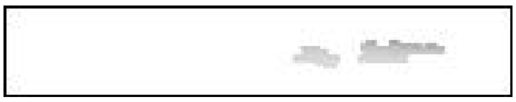
[Hong 02] presents a water detection algorithm on the basis of 3D laser range data. In case that laser beams hit the water at a certain angle, the light is totally reflected resulting in void (i. e. out-of-range) readings in the scan. To decide whether void readings are caused by a water body or just represent a part of the sky, the borders of void areas are analysed. If the void readings are flanked by non-void range data, a water surface is detected. Otherwise, the void readings are declared as sky. In order to augment reliability, the detected water bodies are validated with a camera-based approach which analyses the colour of the regions in question. Figure 9.37 illustrates the results of the approach.

In [Matties 03] the capabilities of laser-based water detection are thoroughly analysed. The interaction of laser beams with air and water is examined in more detail and the presented approach even allows to determine water depth under certain conditions. These conditions include angle of incidence, wavelength of the laser, and attenuation in the water.

Similar to [Hong 02], colour features are used to backup the laser-based detection mechanism. Regardless of the water and weather conditions (e. g. waves on the water, flow velocity, cloudy or bright sky), water can be discriminated from other terrain by colour and brightness. According to the data presented in [Matties 03], the brightness of sky is two and a half times higher than the mean brightness. Furthermore, the brightness of sky


(a) Water bodies in raw laser data


(b) Sky and water extracted


(c) Water bodies extracted

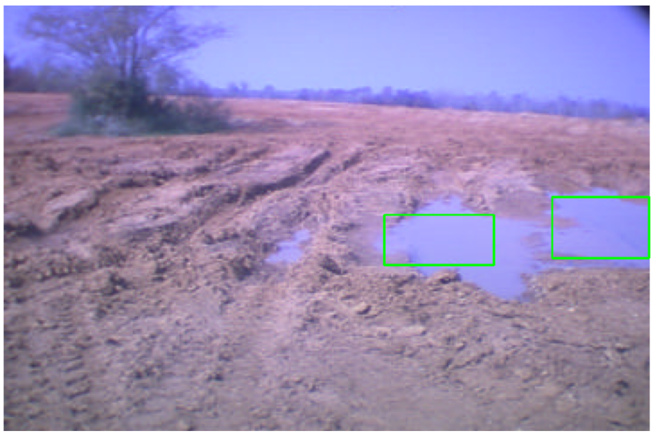
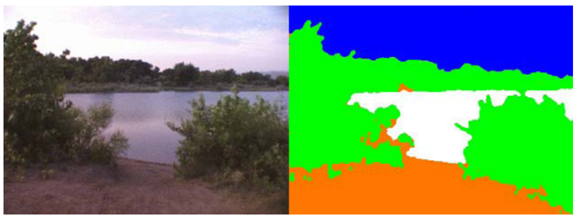

(d) Colour image with projected water bodies.

Figure 9.37: Results from the water detection system using a combination of features extracted from laser and camera data according to [Hong 02]. Detected water hazards have been marked with green bounding boxes (d).

reflected by water surfaces is between the brightness of sky and terrain. Characteristic values of saturation and brightness are initially determined using training images. These characteristic values are then used during classification. Results of this approach are illustrated in Figure 9.38.


(a) Source camera image (left) and classification result (right).

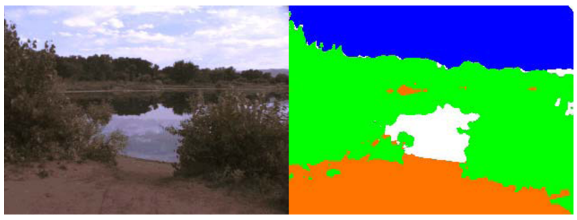

(b) Part of the water body is labelled as vegetation due to trees reflected by the water surface.

Figure 9.38: Results using colour classification according to [Matties 03].
Legend: white → water, brown → soil, green → vegetation, blue → other

The recognition of blue areas in an image and the discrimination of water and sky is a straightforward approach towards water detection. Yet, the colour constancy problem makes these algorithms vulnerable to changing lighting conditions. In particular changes after the initial training phase may lead to unsatisfactory classification results. Furthermore, water surfaces not reflecting the sky cannot be detected as illustrated in Figure 9.38b. In forest areas and under dark cloudy sky this represents a major drawback.

Besides the colour classification mechanism, [Matties 03] presents two further approaches which are based on infrared imaging. The infrared spectrum can be divided into four wavelength ranges: Near Infrared (NIR), Short-wave Infrared (SWIR), Mid-wave / Thermal Infrared (MWIR), and Long-wave Infrared (LWIR). Figure 9.39 illustrates the wavelength spectrum of infrared light in comparison to visible light.

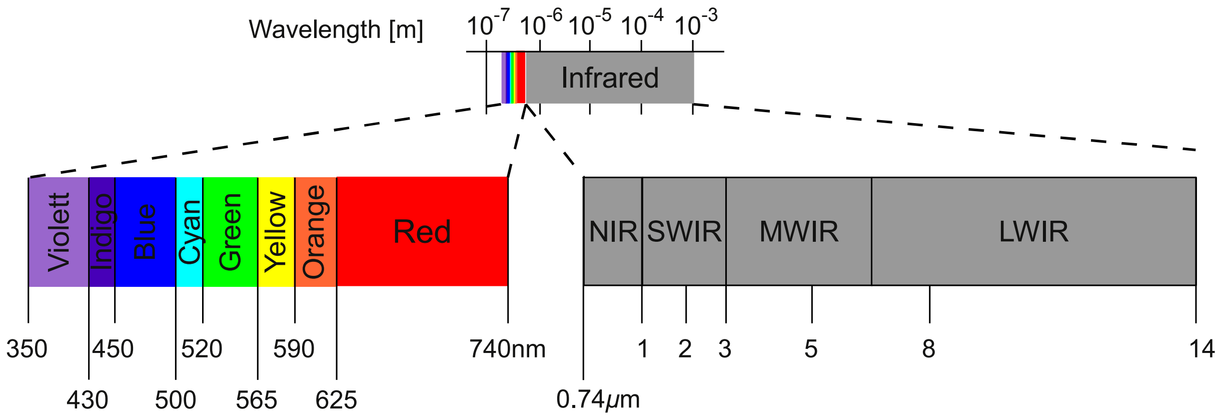
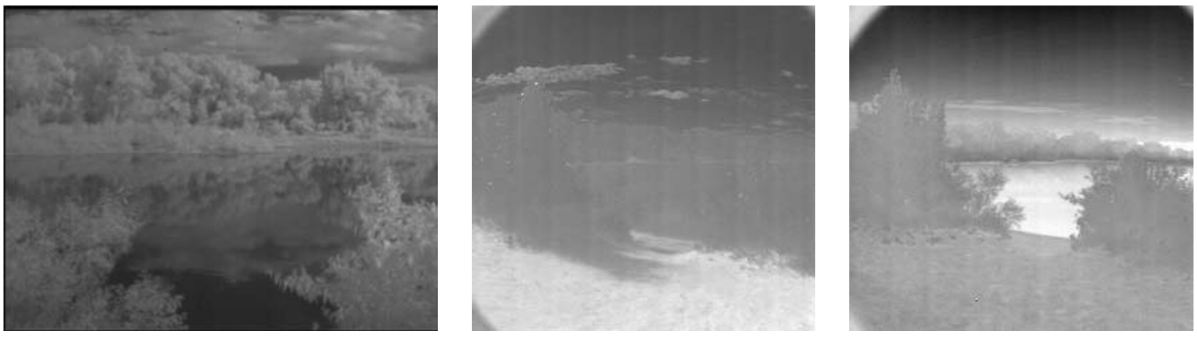


Figure 9.39: Wavelength spectrum of visible light and infrared light.

In SWIR camera images, water has a very low intensity because of the high absorption coefficient of water. Water bodies thus result in regions of dark pixels which can be reliably detected (see Figure 9.40a). Liquid water and ice can also be discriminated due to the characteristic absorption rates [Green 95]. This approach has proven very robust but is limited to daytime applications.



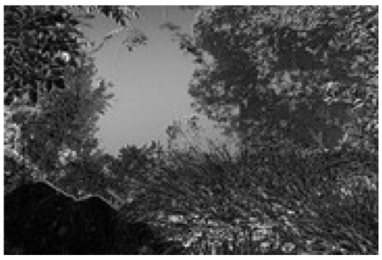
(a) SWIR camera image. **(b)** MWIR camera image at daytime. **(c)** MWIR camera image at nighttime.

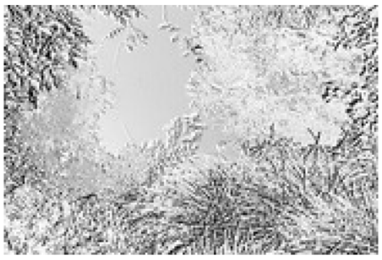
Figure 9.40: Water detection on the basis of infrared imaging according to [Matties 03].

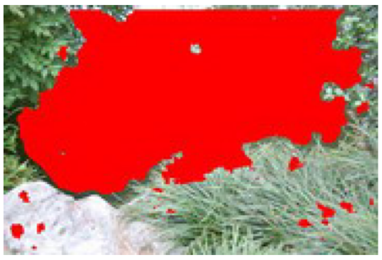
For passive water detection at day and night, MWIR camera images can be used. The fundamental idea of this approach is that at daytime, water is colder then the surrounding terrain (see Figure 9.40b). At nighttime, this relationship is reversed as water retains the heat over a longer period of time than soil (see Figure 9.40b). These characteristic thermal signatures can be identified in MWIR camera data as long as the water body is not too small. In [Haas 06], a similar approach is used for the analysis of satellite images to create a map of temporary water bodies in Western Africa.

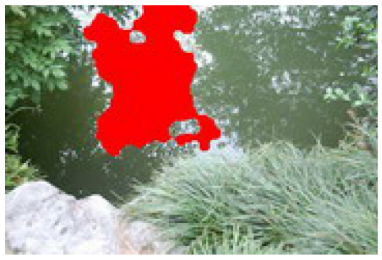
As temperature does not change abruptly, the results of infrared-based approaches are in general temporary stable. Another advantage, in particular of the MWIR method, is the support of nighttime operation. Unfortunately, temperature-based detection mechanisms are quite error-prone. Small water bodies assimilate the ambient temperature too easily, such that these cannot be detected reliably. Furthermore, shadowed areas may falsely be detected as water because of the lower temperature of soil in shadows (see Figure 9.40b). The major drawback of thermal approaches is that the characteristic signatures gradually change during the day in dependence of the solar irradiation. Therefore, precise temperature models have to be generated and adapted over time. A further weak spot of this approach is the point in time when the temperature curves of water and surrounding terrain cross each other during the transition from day to night. For a certain period of time, no temperature difference can be measured.

(a) Original scenery


(b) Intensity image reflecting the degree of polarisation.


(c) Intensity image reflecting the polarisation phase.


(d) Segmentation by polarisation phase similarity


(e) Segmentation by degree of polarisation

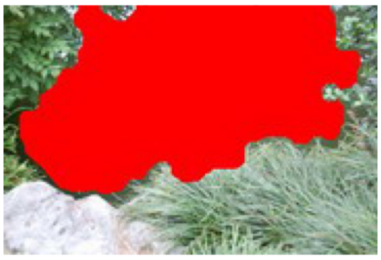

(f) Combined result

Figure 9.41: Water detection on the basis of polarisation degree and phase according to [Xie 07].

[Xie 07] describes a polarisation-based approach which uses either one camera and a rotating polarisation filter or three separate cameras with three fixed polarisation filters. Three images of the same scenery are taken with polarisation angles of 0°, 45°, and 90°. Puddles are detected by comparing the three images and searching for regions with differences in intensity and phase. If the sky is reflected on the water surface, the degree of polarisation is sufficient for reliable water detection. In case vegetation is reflected, the degree of polarisation falls below a critical level such that water detection may become impossible. This problem can be solved by considering the polarisation phase which can be determined on the basis of the image with 45° polarisation. The results of this approach are presented in Figure 9.41.

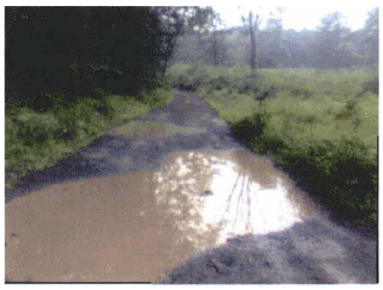
Similar approaches with different camera setups were already presented in earlier publications. In [Wolff 95], twisted nematic (TN) liquid crystals are used as polarisation filters. [Sarwal 04] uses the three-way beam-splitting camera system *Triscene* developed by *Equinox Corporation*[10]. The *Triscene* system splits the incoming light such that the

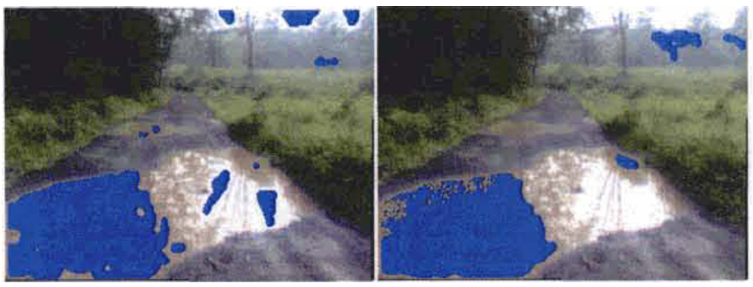
[10]Equinox Corporation → http://equinoxsensors.com/

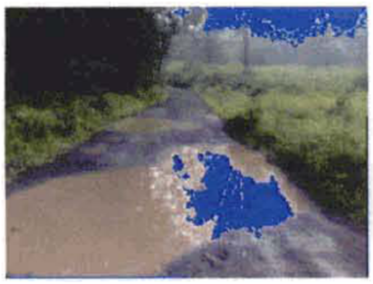
three internal camera sensors observe the same scene with different polarisation directions.

In [Rankin 04] a multi-cue approach towards water detection is described which is based on stereo camera images. Three classification criteria, namely colour, texture, and stereo range, are deployed to determine water hazards.

The colour classification is carried out on images in the HSV[11] colour space. Sky areas feature a characteristic distribution of low saturation and high brightness values. The same holds for sky reflected on water surfaces which is the target of this algorithm. The upper ten rows of each image are analysed with respect to the mentioned characteristics. The characteristic information determined in this step is used for the classification of the rest of the image. Experimental results of this approach are depicted in Figure 9.42c.


(a) Source image containing two water bodies.


(b) Results of the texture analysis: green channel of the RGB image (left) and saturation channel of the HSV image (right).


(c) Result of the colour-based water detection algorithm.

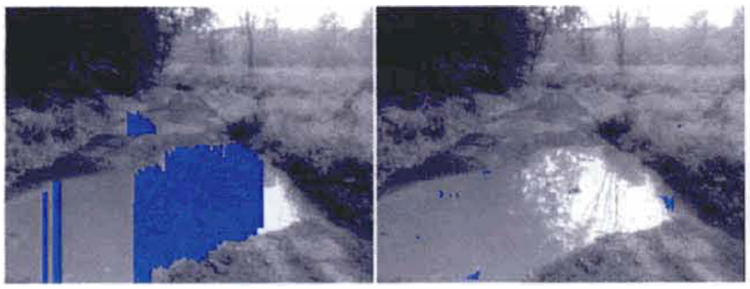

(d) Results of the stereo-based water detection: range reflexion detection (left) and zero-disparity pixels (right).

Figure 9.42: Water detection results yielded by the individual classifiers as proposed in [Rankin 04].

The texture-based classification searches the source image for weakly-textured regions – i. e. regions with uniform colour distributions. An intensity variance filter is passed over the green channel of the RGB source image and the saturation channel of the HSV image also used by the colour classifier (see Figure 9.42b). Regions of uniform colour like sky, vegetation, and overexposed area result in false detections which have to be eliminated by the fusion algorithm.

The last cue for the water detection is performed on the 3D reconstruction yielded from the stereo camera system. Objects reflected on water surfaces result in wrong range information. Instead of the actual distance to the water body, the distance to the reflected

[11]Hue, Saturation and Value (=brightness)

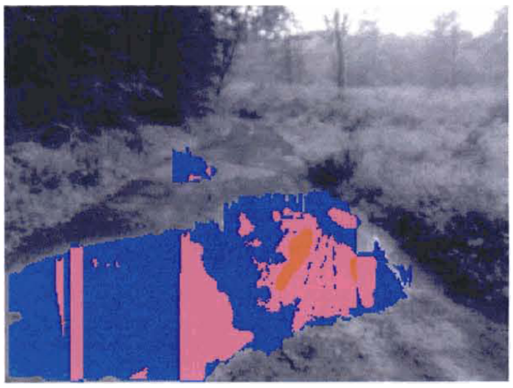


Figure 9.43: Result yielded from the fusion of colour, texture, and disparity cues according to [Rankin 04]. Blue colour indicates support by only one single cue, magenta support by two cues, and red support by all three cues.

objects is measured. The algorithm analyses the range image column-wise top-down, searching for inconsistent range values. The start pixel of a water body is found if the range data resides below the ground surface which has to be determined in a previous processing step. If the range data is consistent with the ground surface again, the end pixel of the range reflexion is detected. The bottom of the column always terminates an open area of range reflexion. Pixels of zero disparity usually indicate large distances to a stereo system. In case regions of zero disparity occur in the lower half of the image, a reflective surface on the ground may have been detected; for instance a water body. Figure 9.42d shows the results of the stereo-based classifiers.

The fusion algorithm integrates the results of the individual classifiers in a selective and weighted fashion. First of all, detections above the horizon and in regions framed by far distances are ignored as these tend to represent sky rather than water. Furthermore, water bodies that have been detected to reside above the vehicle are discarded. After the fusion step, further filtering is carried out to remove small water bodies which do not represent a threat to the robot. The result of the fusion step is illustrated in Figure 9.43. Blue regions are supported only by a single classifier, while magenta indicates that two classifiers have voted for water in the area in question. Finally, the small red regions are supported by all three classifiers. The colour code shows that the particular classifiers complement one another very well and yield an overall satisfactory result.

Further improvements of the proposed set of algorithms concerning the ground surface detection and priorities during the fusion step are described in [Rankin 06]. An extension towards mud detection is described in [Rankin 08].

Despite the ambivalent appearance of water, certain optical effects can be exploited to detect water bodies in natural terrain. Since RAVON already features a suitable 3D LRF, the realisation of a laser-based mechanisms is a straightforward choice. In order to obtain more robust results and to support passive operation, a polarisation-based approach was evolved to complement the laser-based evaluation. In the next section, the laser-based algorithm developed in the context of this application study shall be illuminated. Section 9.4.3 will deal with the camera-based detection facility.

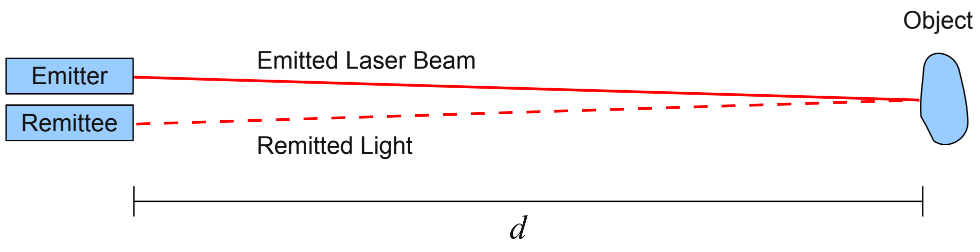


Figure 9.44: Simplified measurement principle of a single point laser range finders used in common 2D LRF constructions.

9.4.2 Laser-based Water Detection

The reliability of laser-based water detection at day and night time, as well as the availability of the required hardware on the target platform RAVON in terms of a 3D LRF, imply the development of according algorithms. As mentioned before, water hazards can be identified via the analysis of void readings in laser range data. The physical background shall briefly be illuminated to clarify the properties of laser data yielded from observations of water surfaces. After that, the detection mechanism will be explained in more detail before discussing required extensions of the representation scheme and how water cues are embedded into the abstraction scheme.

The Physical Background of Laser-based Water Detection

Put simple, off-the-shelf laser range finders commonly deployed in robotics emit laser beams and measure the runtime Δt of the light returning to a photo sensitive device (see Figure 9.44). Mirror systems are used to deflect the laser beams of high-frequency point lasers to build up 2D LRF (e. g. SICK[12]) and 3D LRF [Bergh 00]. As the propagation velocity of light in air c_{air} is known, the distance d to objects can be computed from the runtime measurement as follows:

$$d = \frac{\Delta t}{2} \cdot c_{air}$$

The intensity of the remitted light is highly dependent on the reflective properties of the surface the laser beam is reflected from. Dark surfaces for example may absorb enough light to make the object in question invisible for the LRF. Reflective surfaces on the other hand may reflect the incident laser beam such that no light returns to the sensor. In either case the intensity of the remitted light is insufficient for a reliable range measurement such that the sensor registers a void reading.

As water is only partially reflective, the optical effects on water surfaces differ from the effects of reflective materials. In the following, water is assumed to be an ideal dielectric medium, which means that absorption can be ignored. A ray of light falling onto a partially reflective surface is split into a reflected part and a refracted part as illustrated in Figure 9.45.

[12]SICK – `http://www.sick.com/`

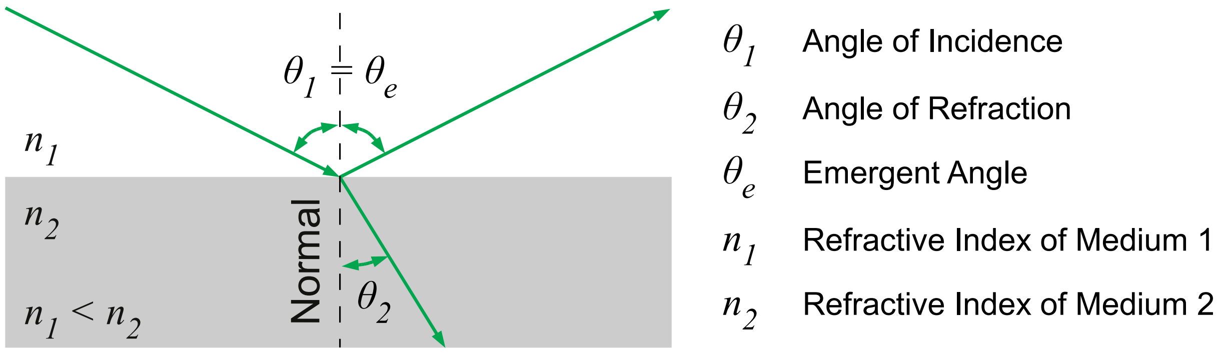

$\theta_1 = \theta_e$
n_1
n_2
$n_1 < n_2$
Normal
θ_2
θ_1 Angle of Incidence
θ_2 Angle of Refraction
θ_e Emergent Angle
n_1 Refractive Index of Medium 1
n_2 Refractive Index of Medium 2

Figure 9.45: Reflexion and refraction of light at the surface of optically denser media.

According to Snel's law of refraction, the product of the refraction index n_1 of the source medium and the angle of incidence θ_1 is equal to the product of the refraction index n_2 of the refractive medium and the angle of refraction θ_2:

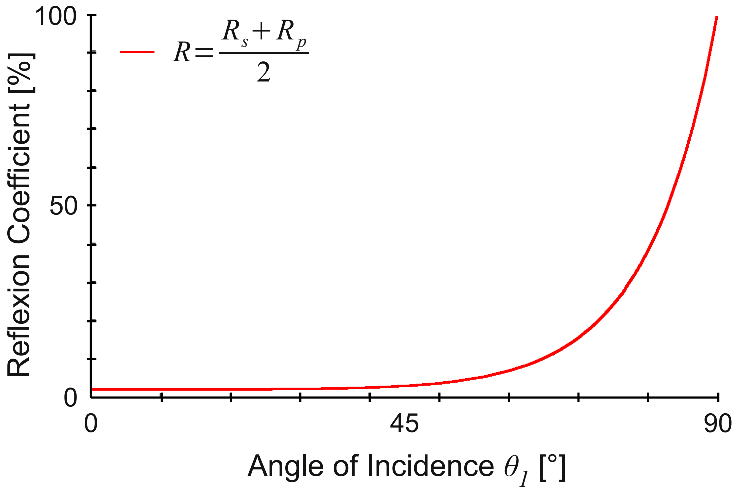
$$n_1 \cdot \sin(\theta_1) = n_2 \cdot \sin(\theta_2) \tag{9.10}$$

All angles are measured relative to the surface normal at point of incidence. On transition into an optically denser medium – e. g. from air to water – light is thus refracted towards the normal of the interface between the two media. Figure 9.46 illustrates the relation of incident angle of light on the water surface and the intensity of the reflexion. Due to the polarisation properties of water surfaces, the reflexion coefficient R has to be computed in two components, namely the p-polarised[13] part R_p and the s-polarised[14] part R_s. The assumption of ideal dielectrics allows to simplify the computation of R_s and R_p according to the Fresnel equations [Gerthsen 89] as outlined in Equations 9.11 (Figure 9.46). For non-polarised light, the arithmetic mean of the two components is the overall reflexion coefficient (see Equation 9.12). The flatter the angle of incidence θ_1, the larger the reflected part of the incident light.

Certain natural conditions including depth, cleanness, and flow velocity of the water add to the reflectivity of water surfaces and result in absorption of incident light. Depending on these conditions and the perspective of the observer, no light may return to the sensor, resulting in void samples. In the experiment setup illustrated in Figure 9.47a, a commercial LRF was placed at the side of a lake such that the scanner captures vertical sections of the water body ahead. The angle of vision of the deployed SICK LMS111 (see Appendix A.1 for technical details.) was restricted to 180° at an angular resolution of 0.25°. The resulting laser data is depicted in Figure 9.47b in terms of red lines indicating the measured range. In this illustration out of range or void samples are indicated by zero-length lines. The critical angle of incidence θ_c in this configuration is about 52°. Apparently angles larger θ_c yield void data points.

[13]p-polarised (short for *parallel polarised*) light waves lying in the plane defined by the incident ray of light and the surface normal.

[14]s-polarised (short for German *senkrecht/perpendicular polarised*) light waves are perpendicular to the plane defined by the incident ray of light and the surface normal.

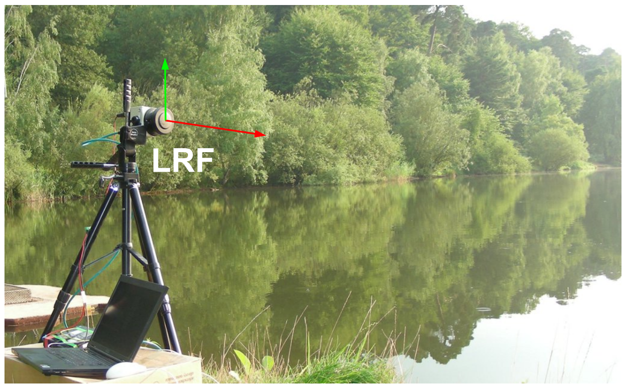


$$R_s = \left(\frac{\sin(\theta_1 - \theta_2)}{\sin(\theta_1 + \theta_2)}\right)^2 \tag{9.11}$$

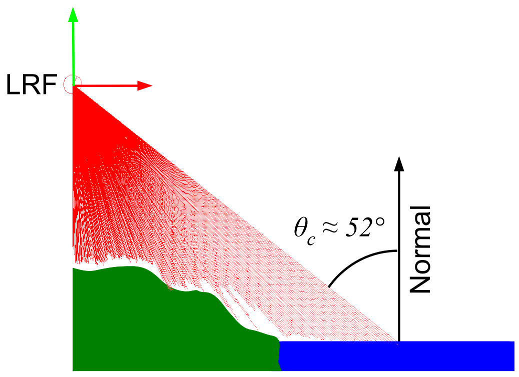
$$R_p = \left(\frac{\tan(\theta_1 - \theta_2)}{\tan(\theta_1 + \theta_2)}\right)^2$$

$$R = \frac{R_s + R_p}{2} \tag{9.12}$$

Figure 9.46: Reflexion coefficient in dependence of the angle of incidence of a ray of light falling onto a water surface.



(a) Experimental setting



(b) Laser range data of a large water body.

Figure 9.47: In this experiment, a SICK LMS111 laser range finder captures a vertical section of a large water body (a). This setup allows to determine the critical angle of incidence θ_c at which void readings occur (b).

Detecting Water in Laser Range Data

The water detection algorithm to be developed for the 3D LRF deployed on RAVON relies on the identification of characteristic void regions.

According to guideline *Algorithmic Separation* (Guideline 5 on page 68), the water detection shall be considered independent of the evaluation procedures presented in Section 9.3.3. That way, both facilities can be treated independent of each other. The sensor provides vertical sections of the terrain like in the experiment of the previous section.

Since the detectability of water bodies is highly dependent on the angle of incidence, the configuration at hand is appropriate. The laser beams are emitted in a radial fashion such that the angle of incidence gradually changes in each scan.

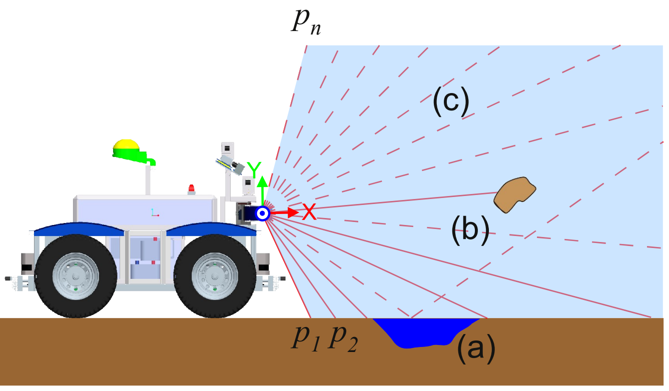


Figure 9.48: Void samples caused by water and regular out-of-range values.

Unfortunately the optical effects of the interaction between laser light and water are not the only source of void readings in laser data. Laser beams which do not hit any object also yield out-of-range values as illustrated in Figure 9.48. Analysing the scanner data bottom up starting with sample p_1, the first void region (a) origins from a water hazard. The void region is flanked by valid data points which represent the embankment of the water hazard. In contrast the second void region (b) is caused by a regular out-of-range point which just did not hit any object. Due to an overhanging object, this void region is surrounded by valid data points. A plausibility check takes into account height and distances to filter such effects. The last void region (c) is sky and is not closed by a valid data point. Such regions are discarded even though this prevents the detection of larger water bodies.

The detection of large water bodies on the basis of laser range data is very difficult. The characteristics of the data resembles the signature resulting from the crossing of a small summit (remember Figure 9.18 on page 146). On observing the summit / large water body from a certain distance, the data indicates a potential cliff. When approaching the location in question, the abrupt end of the load-bearing surface vanishes. While this is wanted for summits, it is not favourable in the case of water hazards. Similar to [Hong 02], the water detection algorithm presented here will focus on smaller water bodies which can reliably be detected. The detection of larger water bodies is carried out with a camera-based approach which shall be explained later.

In a first step, the set of void regions VR are extracted from the set of scan points SP in terms of the flanking valid sample pairs (p_i, p_j) as defined below:

$$\text{let } SP = (p_1, \ldots, p_n) \text{ the set of scan points of a vertical terrain section } n \in \mathbb{N}$$

$$\text{let } void(p_i) = \begin{cases} true & \text{if } p_i \text{ is out of range} \\ false & otherwise \end{cases}$$

$$VR = \{(p_i, p_j) | \neg void(p_i) \wedge \neg void(p_j) \wedge void(q) \quad \forall q \in \{p_{i+1} \ldots p_{j-1}\}\} \qquad (9.13)$$

The set VR contains in the first place all void regions in a scan. Those originating from water surfaces, as well as those indicating overhanging objects. In order to preserve the selectivity of the algorithm, further plausibility checks are carried out to distinguish water bodies from other obstacle types. The surface of water hazards can be assumed horizontal.

Therefore, the far embankment of the water body can be assumed to reside at level with the near embankment. Furthermore, the near embankment has to be closer to the sensor than the far embankment. Both constraints are deployed to define the set of water bodies WB as formalised in Equation 9.14:

$$\begin{aligned} &\text{let } p_i = (p_{ix}, p_{iy}),\ p_j = (p_{jx}, p_{jy}) \\ &\quad x \sim y \text{ denotes that x and y are approximately equal} \\ &WB = \{(p_i, p_j) \in VR | (p_{ix} < p_{jx}) \wedge p_{iy} \sim p_{jy}\} \end{aligned} \tag{9.14}$$

The (p_i, p_j) are assumed to be Cartesian coordinates which have been pitch corrected using orientation information from an inertial measurement unit. Orientation information can further be used to limit the void region extraction to samples which reside below the laser beam which is parallel to the ground. This optimisation reflects the idea that void regions above the horizon rather represent sky than water. Even for water falls this assumption is valid as these always end in a water body which meets the outlined criteria.

9.4.3 Polarisation-based Water Detection

As already discussed in Section 9.4.1, laser-based water detection is often complemented with camera-based mechanisms. Particularly approaches exploiting the polarisation properties of water bodies represent an independent physical basis for increased performance. In contrast to (systems of) colour and texture classifiers, polarisation-based water detection is not vulnerable to colour constancy[15] and may work at variable weather conditions. For these reasons, the obstacle detection facilities of RAVON shall be extended with a water detection system that analyses polarisation effects on water surfaces. After a brief discussion of the physical background, a novel polarisation camera developed in the context of this work will be presented.

The Physical Background of Polarisation-based Water Detection

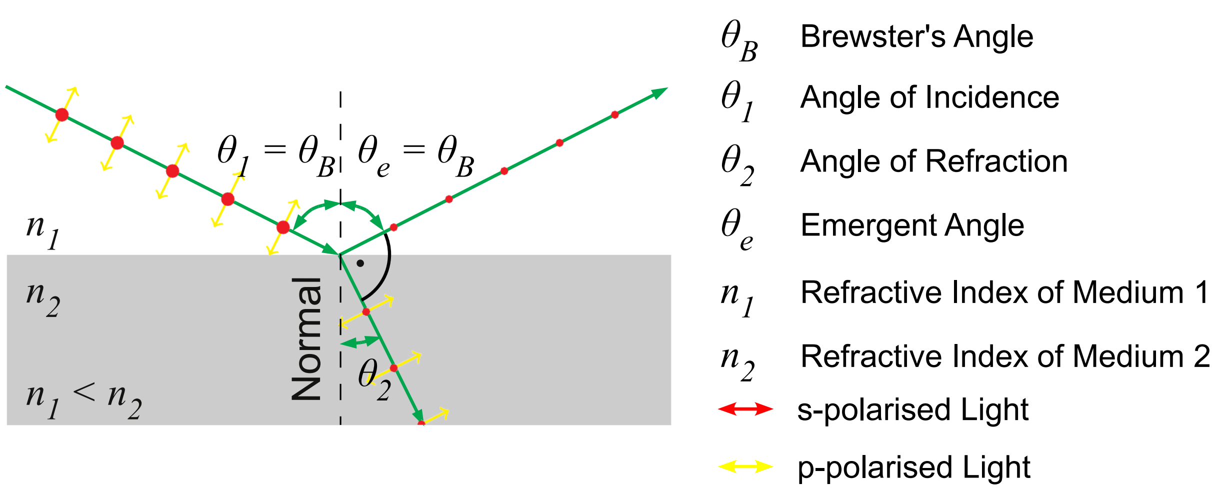
Light falling onto the interface between two transparent media with different optical densities is split into a reflected part and a refracted part (remember Figure 9.45 on page 176). In case the angle of refraction θ_2 has an offset of 90° from the emergent angle θ_e, p-polarised light is refracted completely while s-polarised light is partially reflected as illustrated in Figure 9.49. The respective angle of incidence θ_B $(= \theta_1 = \theta_e)$ is called Brewster's angle. From the considerations in the context of the physical background of laser-based water detection (Section 9.4.2), Snel's law of refraction is already known:

$$n_1 \cdot \sin(\theta_1) = n_2 \cdot \sin(\theta_2)$$

With $\theta_2 = 90° - \theta_1$, Brewster's law for computing θ_B can be derived in a straightforward fashion:

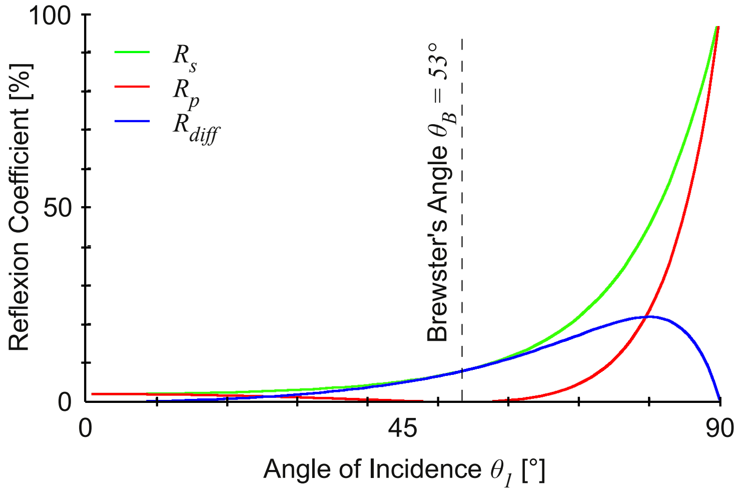
$$\theta_B = arctan(\frac{n_2}{n_1})$$

[15]Colour constancy refers to the problem that colours in machine vision are highly dependent on lighting conditions. Changing illumination in natural environments represent a tremendous difficulty for vision-based approaches.



$\theta_1 = \theta_B$
$\theta_e = \theta_B$
n_1
n_2
$n_1 < n_2$
Normal
θ_2
θ_B Brewster's Angle
θ_1 Angle of Incidence
θ_2 Angle of Refraction
θ_e Emergent Angle
n_1 Refractive Index of Medium 1
n_2 Refractive Index of Medium 2
s-polarised Light
p-polarised Light

Figure 9.49: The p-polarised part of the light incident in the Brewster angle $\theta_B = \theta_1 = \theta_e$ onto a partially reflective surface is completely refracted with $\theta_2 = 90° - \theta_B$.



$$R_s = \left(\frac{\sin(\theta_1 - \theta_2)}{\sin(\theta_1 + \theta_2)}\right)^2 \tag{9.15}$$

$$R_p = \left(\frac{\tan(\theta_1 - \theta_2)}{\tan(\theta_1 + \theta_2)}\right)^2 \tag{9.16}$$

$$R_{diff} = R_s - R_p \tag{9.17}$$

Figure 9.50: Comparison of the reflexion coefficients of s-polarised and p-polarised light in dependence of the angle of incidence of a ray of light falling onto a water surface.

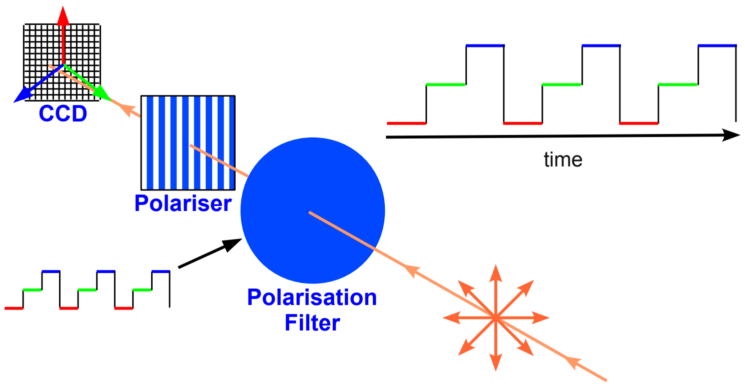
In the vicinity of θ_B, the reflection intensity of s-polarised light and p-polarised light is apparently significant for polarisation effects on water surfaces. The intensity of the reflected light depends on the angle of incidence θ_1, the refractive indices n_1, n_2 of the interfacing media, and the polarisation direction of the incident light.

According to Fresnel's equations, the intensity of the r-polarised part and the p-polarised part of the reflected light are given by Equations 9.15 and 9.16. In Section 9.4.2, the overall reflexion coefficient $R = \frac{R_s + R_p}{2}$ was consulted to explain the reflective properties of water surfaces (see Figure 9.46 on page 177).

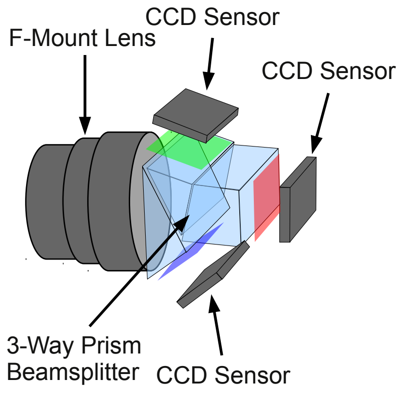
Figure 9.50 now shows the separate reflexion coefficients for R_s and R_p. The curves unveil a significent intensity difference between s-polarised and p-polarised light illustrated in terms of the intensity difference curve R_{diff} (see Equation 9.17). This intensity difference is the main feature of polarisation-based water detection. As already mentioned before, this feature does not rely on highly ambivalent colour or texture cues. The nature of polarisation effects makes this approach in principle very robust towards the variability of water body appearance in natural environments.

Multi-spectral Camera Systems for Water Detection

In order to measure the intensity difference R_{diff}, a multi-camera system has to capture the same scenery with different polarisation angles. In Section 9.4.1 several settings using three polarisation directions have been discussed. For applications on mobile platforms, the setup with rotating polarisation filters or the side-by-side arrangement of three cameras with fixedly mounted polarisation filters [Xie 07] are not appropriate. The former is subject to tremendous motion-induced image disparity while the latter introduces distance-dependent disparity. Both types of disparity lead to a missalignment of the image data which is difficult to compensate on the move. The commercial *Salsa* camera by *Bossa Nova Tech*[16] uses a fast switching liquid crystal polarisation filter to obtain images in three polarisation directions at almost identical points in time (see Figure 9.51a). The manufacturer claims that the disparity effects can be neglected. Configurations using



(a) Schematic of the Bossa Nova Tech Salsa camera.



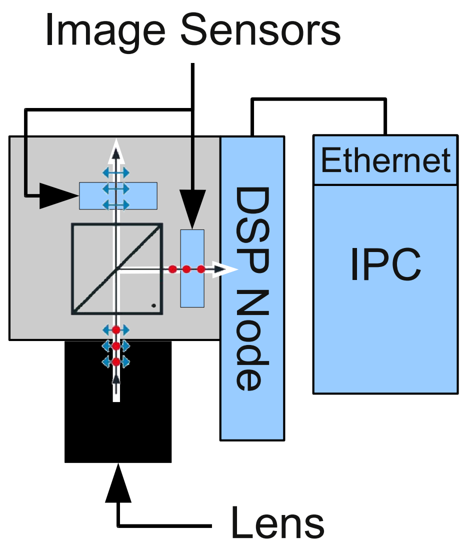
(b) Schematic of the FluxData 3-CCD camera system.

Figure 9.51: Schematic drawings of off-the-shelf polarisation cameras. The drawings have been reproduced according to the online documentation of the respective manufacturer.

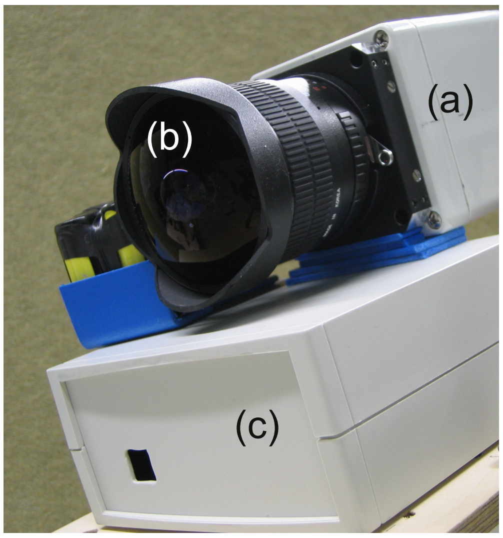
beam-splitting devices with synchronised cameras are the best solution as these assure synchronous image registration of the same scenery. For this configuration two commercial cameras are available off-the-shelf. Besides *Equinox Corporation*'s *Triscene* deployed in [Sarwal 04], the *FluxData*[17] 3-CCD multi-spectral camera is available (see Figure 9.51b). Both manufacturers state to use proprietary multi-way prism technology to split the incident light into three separate channels which are captured by individual image sensors. Figure 9.51b illustrates the arrangement of the *FluxData* camera. The problem with this approach is that the light yield at each individual sensor is very poor. Provided that the absorption rate of the prism system can be neglected, the light yield behind the three-way prism can be at most one third of the incident light. The incident light is unpolarised and the prism system does not feature any prepolarisation. Therefore, the polarisation

[16]Bossa Nova Tech → `http://www.bossanovatech.com/`
[17]FluxData → `http://www.fluxdata.com/`

filters that are placed between the light outlets and image sensors are a further absorption factors. The poor light efficiency renders the usage of this configuration under difficult lighting conditions impossible. In order to get rid of this constrait, a novel polarisation camera system was designed in the context of this work.



(a) Schematic of the polarisation camera system.



(b) The sensor prototype *AquaVisor F*.

Figure 9.52: Schematic of the polarisation camera (a) and the prototype *AquaVisor F* manufactured by H&S Robotic Solutions.

From the physical background investigated above, it is known that orthogonal polarisation directions are sufficient for the detection of water hazards. The third polarisation direction is required to detect phase shifts. In favour of improved light efficiency, this possibility was abandoned. Inspired by the 3-CCD multi-spectral camera by *FluxData*, a sensor system with two cameras and a polarising beam-splitter cube was developed [Renner 08]. Polarising beam splitters have the advantage that the light is split directly according to the desired polarisation directions. That way, no additional polarisation filters which absorp light intensity are required.

In the context of a study on water detection funded by the German Ministry of Defence[18], the integrated prototype sensor *AquaVisor F*[19] was manufactured by H&S Robotic Solutions (see Figure 9.52). The results yielded in the context of this study have been summarised in technical reports [Schäfer 09a, Schäfer 09b]. As depicted in Figure 9.52b, the camera consists of a commercial F-mount lens (a), the optical arrangement mounted into a compact casing (b), and an embedded DSP node for onboard data evaluation. The optical arrangement of the polarising beam splitter cube and the two image sensors is illustrated in Figure 9.52a. In order to assure that images are captured in a synchronous fashion, the image sensors are directly connected to the onboard DSP node. Colour skew

[18]The study „Wasserkerkenner" was kindly supported by the „Wehrtechnische Dienststelle für Informationstechnologie und Elektronik (WTD-81) GF 340 – Aufklärungstechnik, Robotik, Security"

[19]Fundamental technology for multi-camera systems is in patent pending status – patent issue 10 2010 020 537.0

and image missalignment are removed with simple calibration procedures to assure that both cameras show the same scenery in similar colours.

Detecting Water and other Reflections

With the polarisation camera setup outlined above, synchronous registration of identical image sections can be achieved. Thorough calibration removes problematic colour shift and image disparity. That way, reflection-free surfaces are assured to yield sufficiently similar colour values in both camera images. Under these conditions the water detection algorithm itself can be kept rather simple.

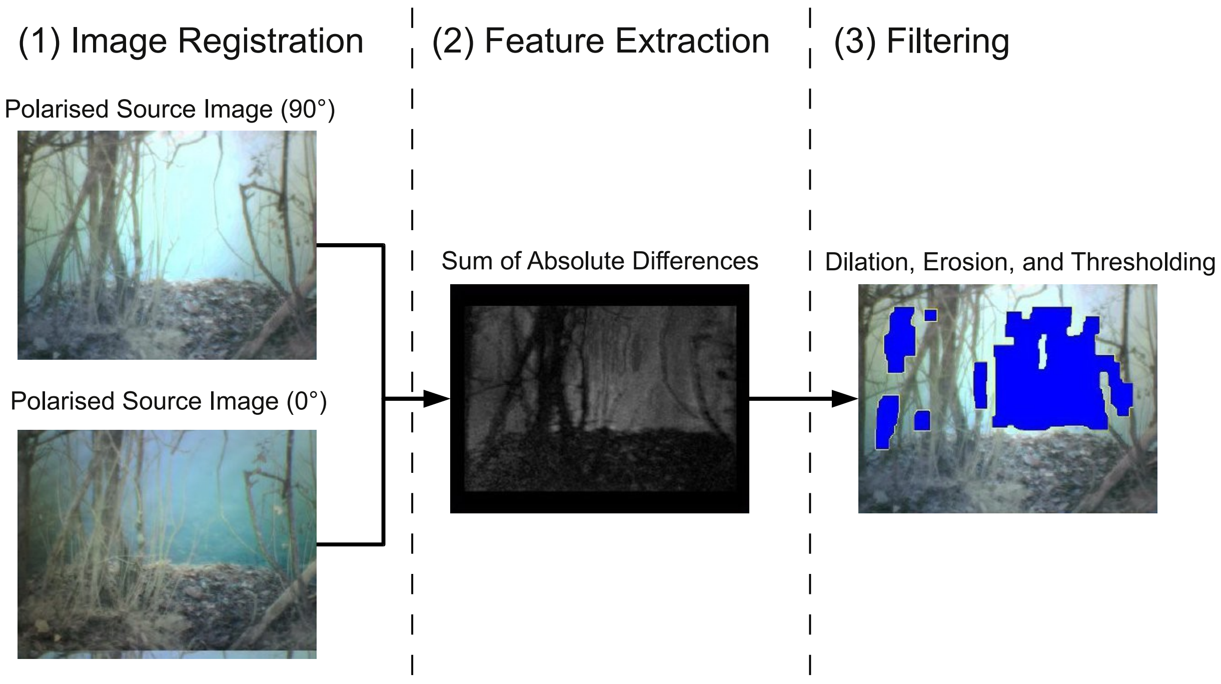

(1) Image Registration
(2) Feature Extraction
(3) Filtering
Polarised Source Image (90°)
Polarised Source Image (0°)
Sum of Absolute Differences
Dilation, Erosion, and Thresholding

Figure 9.53: Polarisation-based water detection procedure in three steps.

The fundamental idea of the algorithm is to detect intensity differences in two source images which are taken with perpendicular polarisation directions. Figure 9.53 illustrates the three-step evaluation procedure. In the first step, the source images are registered in a synchronous fashion. After that, pixel-wise intensity deviations are computed in terms of the sum of absolute differences (SAD). This step yields a grey value image in which more reflective surfaces are indicated by lighter grey shades (i. e. greater difference between the source images) while less reflective surfaces appear darker. In a third step, the difference image is analysed as to larger connected regions of light grey which represent water bodies of relevance. This step consists of common image processing filters to group neighbouring regions of similar colour (Dilation), to discard small patches (Erosion), and to identify blobs with significant difference values (Thresholding). Finally, a contour detection algorithm is deployed to group the particular patches of significant intensity representing the water bodies. In the experiment depicted in Figure 9.53, a lake shore featuring sparse vegetation was captured. The increased intensity in the upper source image, resulting from the reflection of the s-polarised part of the incident light, is clearly visible in comparison to the lower source image. In the difference image which documents this fact, the lake shore and the vegetation appear as dark regions while the patches of the water surface are rather light grey. The result of the detection algorithm is highlighted with a blue overlay in step three.

The laser-based water detector introduced in Section 9.4.2 for instance used the geometric embarkment structure of water bodies to distinguish true water hazards from water region candidates resulting from overhanging objects. Polarisation-based water detection does not rely on such structural constraints to eliminate false positives. For that reason, larger water bodies which were undetectable with the proposed laser-based approach can be identified with the polarisation cameras at hand. Extensive experiments carried out in the context of [Schäfer 09b], the arrangement with two cameras and a polarising beam splitter allowed for relyable water detection under a wide variety of environmental conditions. In particular unfavourable illumination conditions during dusk and dawn have been proven to be a minor problem with this configuration. An improved prototype with high dynamic range cameras and revised optics shall be evaluated in the second phase of study „Wasserkerkenner[20]".

9.4.4 Representation of Water Hazards

The detection of water bodies is a crucial requirement for autonomous operation in natural environments. Two complementing approaches realised in the context of this application study have been outlined above. What remains to be done is the integration of the yielded data into the information basis of the target platform. For that purpose, a suitable representation has to be designed.

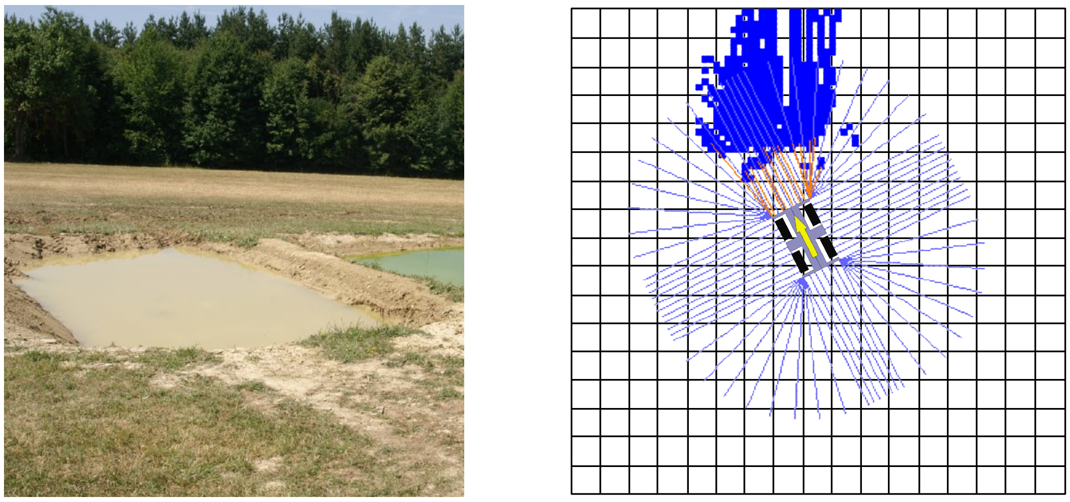
The water detection approaches are in themselves rather defensive and unfold their performance in combination with each other. Following guideline *Representational Separation* (Guideline 4 on page 68), each approach will be storing its data in an individual representation. A fused information basis will be generated at a later processing stage as proposed by *Deferred Fusion* (Guideline 6 on page 68).

As before, the combination of `ContentBase` and `ProbabilisticContent` seems to be a good starting point for modelling a representation for water hazards (*Content / Handler Reuse* (Guideline 1 on page 67)).

Due to the reflective properties of water surfaces, detection mechanisms are highly dependent on the perspective of the observer. When approaching a water body, the exploited effects occur and disappear over time.

In contrast to other obstacle types, the disappearance of water cues does not mean that the hazard has disappeared. It is rather an indicator that the perspective has changed. The decaying mechanism of the `ProbabilisticContent` (see Section 4.3.2) would thus lead to the unintentional removal of water hazards from the representation. For that reason, the `ProbabilisticContent` is extended with an option to deactivate information decay. Furthermore, the absence of water cannot be definitely decided for covered patches of terrain.

[20]Wassererkenner → German for "water detector".



(a) Water hazard from the robot's point of view. **(b)** Short-term memory of the laser-based water detection facility.

Figure 9.54: Water detection experiment in the context of the European Land Robot Trials.

The property set for the water detection reflects this fact by modelling water and terrain coverage as follows:

$$waterPROP = (water, scannedArea)$$

where

$$water(i) = \begin{cases} true, & \text{water was detected several times} \\ & \text{in content element } i \\ false & \text{otherwise} \end{cases}$$

$$scannedArea(i) = \begin{cases} true, & \text{content element } i \text{ was scanned} \\ & \text{but no water was detected so far} \\ false & \text{otherwise} \end{cases}$$

Figure 9.54 shows the integration of the laser-based water detection system in an experiment carried out in the context of the European Land Robot Trials 2008.

9.5 Sensor Processing Capabilities in Summary

In this chapter, several sensor processing algorithms have been presented. In order to demonstrate the applicability of the proposed design schemata to real-world challenges in sensor processing, a wide variety of different data sources have been considered. The design of the terrain assessment facilities was carried out in a perception-oriented fashion with a strong focus on representation design. This procedure yielded a set of short-term memories, which are tailored to the specifics of the particular sensor system and the evaluation algorithm in question. In that context, the flexibility of the proposed representation scheme was shown by introducing several sets of *Extensions* under consideration of the proposed Design Guidelines. In the following chapter, the generic transfer of information from the sensor processing level to the control level shall be defined in terms of the *Aspect-oriented Configuration* of the *Semantic Abstraction*.

10. Aspect-oriented Configuration of the Semantic Abstraction

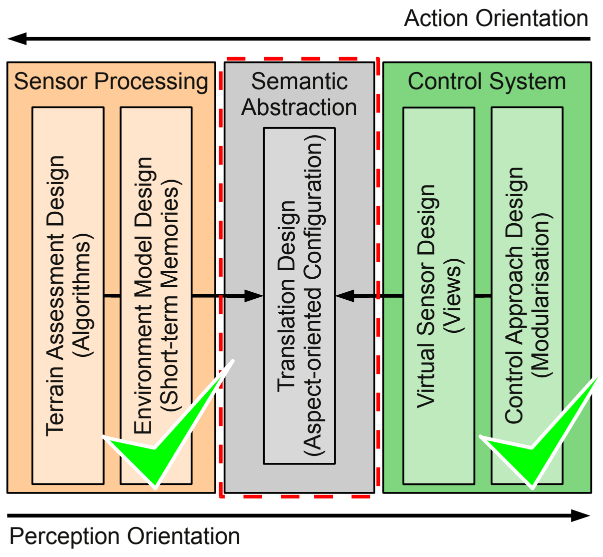

Action Orientation
Sensor Processing
Terrain Assessment Design (Algorithms)
Environment Model Design (Short-term Memories)
Semantic Abstraction
Translation Design (Aspect-oriented Configuration)
Control System
Virtual Sensor Design (Views)
Control Approach Design (Modularisation)
Perception Orientation

Figure 10.1: Overview of the proposed design approach.

In the preceding chapters, sensor processing and control system were exemplary designed for the off-road robot RAVON. Following the proposed design methodology (see Figure 10.1), the control system was designed in an action-oriented fashion yielding a behaviour-based modularisation according to the deployed architecture iB2C. For each *Behaviour*, *Virtual Sensors* were specified which are to provide required information about the environment in terms of abstract *Views*. The sensor processing facilities were designed according to perception-oriented principles resulting in a sensor-centric modularisation. Each sensor processing facility represents a self-contained component dealing with the detection of specific features from one sensor system. Information extracted from the environment is accumulated into tailored *Short-term Memories*.

Virtual Sensor specifications as well as *Short-term Memory* specifications contain spatial definitions (mount point, range, resolution, etc.) and semantic definitions. The semantic specification is defined in terms of a property set which reflects the particular control-level or sensor-level aspect dealt with. As already stated before, these aspects need not be semantically compatible in the first place. In the remaining design step subject to this chapter, the semantic compatibility has to be established. For that purpose, the semantics of the particular aspects are mapped to yield the configuration for the `Semantic Abstraction` facilities which translate the sensor-level information into control-level information.

The reader should note that design activities may very well be conducted in parallel and in an iterative fashion. It is perfectly possible to specify the configuration of the `Semantic Abstraction` in part and to extend the specification whenever novel sensor processing facilities or control approaches are integrated. The well-defined interface specifications according to the proposed schemata allow to develop sensor processing and control components independently of each other which provides further options for parallelisation. In the rest of this chapter, the configuration of the `Semantic Abstraction` for RAVON will be evolved step by step following the same outline as the action-oriented control design in Chapter 8. First of all, safety-relevant aspects shall be covered to assure collision-free navigation. After that, the planning facilities will be supplied with information to activate the mid-range navigation capabilities of the robot.

10.1 Configuration of Safety-critical Aspects

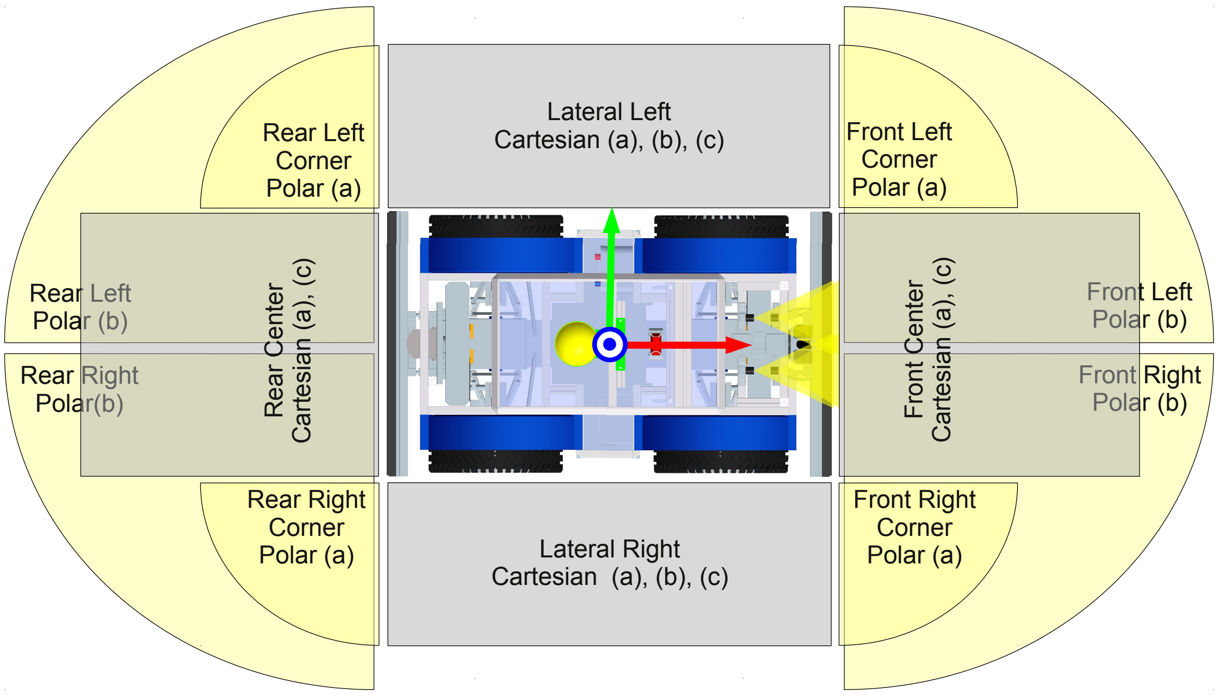

Rear Left Corner Polar (a)
Lateral Left Cartesian (a), (b), (c)
Front Left Corner Polar (a)
Rear Left Polar (b)
Rear Center Cartesian (a), (c)
Front Center Cartesian (a), (c)
Front Left Polar (b)
Rear Right Polar(b)
Front Right Polar (b)
Rear Right Corner Polar (a)
Lateral Right Cartesian (a), (b), (c)
Front Right Corner Polar (a)

Figure 10.2: Summary of the safety-relevant *Virtual Sensors* defined on RAVON (see Section 8.2.1). *Virtual Sensors* are specified by three major classes of behaviours: the *Slow Down Behaviours* (a), the *Keep Distance Behaviours* (b), and the *Evasion Behaviours* (c).

In the control design step outlined in Chapter 8, the safety-relevant *Behaviours* are located in the *Behaviour* group `Guardian` (see Section 8.2.1 on page 87). As a reminder, Figure 10.2 summarises the *Virtual Sensor* specifications for the safety *Behaviours* in a structural overview. The safety level of competence on RAVON is composed of three major classes of *Behaviours*:

(a) *Slow Down*
(b) *Keep Distance*
(c) *Evasion*

In Figure 10.2, the *Virtual Sensors* required by the particular *Behaviour* class are highlighted with letters (a) through (c).

Safety-critical *Behaviours* always require most current information in order to achieve the smallest possible delay from data acquisition to control value setting. Therefore, the required *Views* are directly rendered from the short-term memory of the particular sensor evaluation algorithms. For each algorithm, the *Behaviours* in question are replicated and information fusion is carried out on the *Behaviour* level following *Deferred Fusion* (Guideline 6). That way, tight sensor-actor loops can be realised on a distributed information basis.

Having refreshed the structural configuration of the *Virtual Sensors* and the control strategies followed on RAVON, the semantics of the control-level aspects declared for the safety *Behaviours* shall be configured. In order to minimise latency and to keep safety-relevant *Behaviours* as simple and robust as possible, all related *Virtual Sensors* have been specified to render *Views* containing only a single property. For the safety *Behaviours*, the following aspects have been declared at design time:

$$\begin{aligned} \mathit{driveModeMaximumVelocityPROP} &= (\mathit{obstaclesMaximumVelocity}) \\ \mathit{driveModeModerateVelocityPROP} &= (\mathit{obstaclesModerateVelocity}) \\ \mathit{driveModeTactileCreepVelocityPROP} &= (\mathit{obstaclesTactileCreepVelocity}) \\ \mathit{emergencyStopPROP} &= (\mathit{emergencyStop}) \end{aligned}$$

Since the safety-relevant *Views* are to be rendered directly from the individual *Short-term Memories*, the aspects have to be configured separately for each detection facility. The algorithm in question is indicated in subscript appended to the property names. Note that these appendages are just a syntactical means to distinguish the different specifications. The resulting compounds do not represent new semantical entities and therefore remain semantically compatible. That way, the definitions can be grouped for one aspect without loss of clarity.

Aspect *driveModeMaximumVelocityPROP*

In drive mode *Maximum Velocity*, the robot is supposed to travel at high velocities such that any kind of obstacle, even minor terrain jaggedness, may represent a threat to the robot. In order to reflect this, aspect *driveModeMaximumVelocityPROP* has to be configured as defensively as possible, i. e. anything that looks like an obstacle or a bump in the ground is regarded as a threat:

$$\begin{aligned} \mathit{driveModeMaximumVelocityPROP} \leftarrow (&\mathit{tactilePROP},\\ &\mathit{laser2dPROP},\\ &\mathit{laser3dPROP},\\ &\mathit{waterPROP}) \end{aligned}$$

$$\begin{aligned} \mathit{obstaclesMaximumVelocity}_{\mathit{tactile}} &= \mathit{rigidObject} \land \neg \mathit{freeSpace}\\ \mathit{obstaclesMaximumVelocity}_{\mathit{laser2d}} &= \mathit{rigidObject} \land \mathit{obstructedArea} \land \neg \mathit{freeSpace}\\ \mathit{obstaclesMaximumVelocity}_{\mathit{laser3d}} &= \mathit{positive} \lor \mathit{negative} \lor \mathit{highCentring} \lor\\ &\quad \mathit{step} \lor \mathit{overhanging} \lor \mathit{groundClutter}\\ \mathit{obstaclesMaximumVelocity}_{\mathit{water}} &= \mathit{water} \end{aligned}$$

Aspect ***driveModeModerateVelocityPROP***

In drive mode *Moderate Velocity*, the vehicle is throttled such that minor clutter can be passed without risk. The *PROP* interface and the abstraction scheme allow to configure this nuance in a straighforward fashion. Taking the specification of control aspect *driveModeMaximumVelocityPROP* as a basis, most sensor-level properties are mapped to control aspect *driveModeModerateVelocityPROP* in an identical fashion. By adapting the specification for the 3D LRF data, ground clutter can be ignored when in drive mode *Moderate Velocity*:

$$\mathit{driveModeModerateVelocityPROP} \leftarrow (\mathit{tactilePROP}, \mathit{laser2dPROP}, \mathit{laser3dPROP}, \mathit{water})$$

$$\begin{aligned} \mathit{obstaclesModerateVelocity}_{\mathit{tactile}} &= \mathit{rigidObject} \land \neg \mathit{freeSpace}\\ \mathit{obstaclesModerateVelocity}_{\mathit{laser2d}} &= \mathit{rigidObject} \land \mathit{obstructedArea} \land \neg \mathit{freeSpace}\\ \mathit{obstaclesModerateVelocity}_{\mathit{laser3d}} &= \mathit{positive} \lor \mathit{negative} \lor \mathit{highCentring} \lor\\ &\quad \mathit{step} \lor \mathit{overhanging}\ \cancel{\lor \mathit{groundClutter}}\\ \mathit{obstaclesModerateVelocity}_{\mathit{water}} &= \mathit{water} \end{aligned}$$

Using indirections, control aspects featuring identical configurations can be linked to one another to simplify the configuration of the `Semantic Abstraction`. Furthermore, *Views* with identical structural and semantical specifications need not be rendered multiple times reducing computational effort.

The control aspect specification for the drive mode *Moderate Velocity* can thus be reduced as follows:

$$driveModeModerateVelocityPROP \leftarrow (tactilePROP, laser2dPROP, laser3dPROP, water)$$

$$\begin{aligned}
obstaclesModerateVelocity_{tactile} &= \cancel{rigidObject \wedge \neg freeSpace} \\
&\quad obstaclesMaximumVelocity_{tactile} \\
obstaclesModerateVelocity_{laser2d} &= \cancel{rigidObject \wedge obstructedArea \wedge \neg freeSpace} \\
&\quad obstaclesMaximumVelocity_{laser2d} \\
obstaclesModerateVelocity_{laser3d} &= positive \vee negative \vee highCentring \vee \\
&\quad step \vee overhanging \cancel{\vee groundClutter} \\
obstaclesModerateVelocity_{water} &= \cancel{water} \\
&\quad obstaclesMaximumVelocity_{water}
\end{aligned}$$

Aspect ***driveModeTactileCreepPROP***

Drive mode *Maximum Velocity* and drive mode *Moderate Velocity* can be derived from one another in a straightforward fashion. In comparison, the data basis for drive mode *Tactile Creep* has to be designed with care as the vehicle is supposed to negotiate terrain with the tactile facilities. This procedure requires the system to annul the impact of the visual sensor evaluation facilities to a certain degree. In order to achieve this, *Behaviour group Deceleration Obstacle Avoidance (Tactile Creep)* (see Section 8.2.1 on page 96) is configured to keep the vehicle at a velocity of $10\frac{\text{cm}}{\text{s}}$ while reacting to obstacles with the evasive *Behaviours*. Vehicle safety is ultimately achieved via the *Emergency Stop Behaviours* which bring the robot to a halt in front of critical obstacles. That way, a clear distinction between vehicle performance and vehicle safety can be achieved via selective configuration of both data bases.

For improved selectivity, the height level set introduced in Section 9.3.4 can be deployed to moderate the property predicates:

$$Levels = \{low, middle, high\} \quad \text{(see Equation 9.9)}$$

let $L \subseteq Levels$ the set of levels relevant for a given aspect

let $prop(L) \in \{true, false\}$ the height-level-filtered property information[1].

where

$$prop(L) = \begin{cases} true & \text{if property } prop \text{ applies on any of the height levels } level_i \in L \\ false & \text{otherwise} \end{cases}$$

[1]The schemata proposed in this work support general filters which are specified in a hyper-dimensional configuration space. Predicate $prop(L)$ was chosen as a simple notation for the application of the height level filter exemplifying the filtering concept in the context of this application study.

Then aspect *driveModeTactileCreepPROP* shall be specified for RAVON as follows:

$$driveModeTactileCreepPROP \leftarrow (tactilePROP, laser2dPROP, laser3dPROP, water)$$

$$\begin{aligned}
obstaclesTactileCreep_{tactile} &= obstaclesMaximumVelocity_{tactile} \\
obstaclesTactileCreep_{laser2d} &= \text{impact mode } \textbf{No Impact} \\
obstaclesTactileCreep_{laser3d} &= negative \lor overhanging \lor positive(\{middle,\ high\}) \lor \\
&\quad (\neg vegetation(\{low\}) \land (step \lor highCentring)) \\
obstaclesTactileCreep_{water} &= obstaclesMaximumVelocity_{water}
\end{aligned}$$

For fundamental safety, the bumper configuration remains untouched as for the aspects discussed before. The same holds for water hazards which may be detected beneath sparse ground vegetation. As the 2D LRF only yield information on a horizontal plane close to the ground, the data basis cannot contribute any valuable information during tactile manoeuvring. Therefore, this data basis is configured to have no impact on aspect *driveModeTactileCreepPROP*.

The 3D LRF provides a rich information basis on the environment that is composed of critical structures (e. g. negative obstacles and overhanging obstacles) which may never be ignored and structures which may be pushed aside by the instrumented bumper (e. g. positive obstacles). Overhanging objects do not have ground contact and are thus not detectable for the bumper construction deployed on RAVON. The same holds for negative obstacles and therefore both structures are considered as obstacles in drive mode *Tactile Creep*. Positive objects in contrast have ground contact and may be negotiable on a tactile basis. In order to prevent damage from the more sensitive hardware (e. g. the 3D LRF and the camera systems) mounted on RAVON's sensor tower, positive obstacles are blanked out selectively on height level *low*. Objects on the height levels *middle* and *high* are always regarded as obstacles. If there is an indicator for vegetation at height level *low*, ground conformations like steps or high centring threats may be ignored as these can be assumed to stem from sparse ground vegetation.

Aspect *driveModeEmergencyStopPROP*

The line between what terrain is negotiable and what structures really represent a threat to a robot is nowhere as fine as in off-road robotics. The "leaks" that the drive mode *Tactile Creep* introduces into vehicle safety in order to allow for tactile negotiation attempts have to be sanctioned by a further safety instance. This independent safety instance is provided by the control system design in terms of the behaviour group *Emergency Stop*. For static vehicle safety, several simple *Behaviours* monitor critical vehicle parameters like roll and pitch angle. Bumper events are also monitored directly by separate *Behaviours* in order to minimise latency. In addition to that, critical conformations detected by the visual sensor systems are subsumed to aspect *driveModeEmergencyStopPROP* which is evaluated by a *Slow Down Behaviour* group.

Structurally this group is identical to the *Deceleration Obstacle Avoidance* behaviour groups (see Section 8.2.1 on page 96) but the parameters are chosen more restrictively. While the *Deceleration Obstacle Avoidance* groups are designed to gradually slow down the vehicle in tighter situations, the *Emergency Stop* group is intended to stop the robot immediately when obstacles come closer than a critical distance threshold. Therefore, the former *Slow Down Behaviours* are configured to decelerate at moderate rates starting when obstacles are still quite distant. In contrast, the latter are configured to decelerate at maximal rate just before the vehicle crashes into critical obstacles.

Semantically, aspect *emergencyStopPROP* is quite similar to aspect *obstaclesTactileCreep* since both reflect critical obstacles which have to be avoided at any stake:

$$emergencyStopPROP \leftarrow (tactilePROP, laser2dPROP, laser3dPROP, water)$$

$$\begin{aligned}
obstaclesEmergencyStop_{tactile} &= \text{impact mode } \textbf{No Impact}\\
obstaclesEmergencyStop_{laser2d} &= \text{impact mode } \textbf{No Impact}\\
obstaclesEmergencyStop_{laser3d} &= obstaclesTactileCreep_{laser3d}\\
obstaclesEmergencyStop_{water} &= obstaclesTactileCreep_{water}
\end{aligned}$$

In order to allow creeping through high vegetation, the 2D LRF has to be ignored for the same reason as in aspect *driveModeTactileCreepPROP*. The information bases from the 3D LRF and the water detection facilities can also be configured as in aspect *driveModeTactileCreepPROP* as these already reflect the critical entities of the environment. As bumper events are monitored directly, the short-term memory of the tactile facility may be ignored as it does not provide additional information. On the contrary, the imprecision of the deployed bumper system might even block paths if the short-term memory was considered for aspect *emergencyStopPROP*. Whenever a rigid entity is touched, the instrumented bumper reports this impact, but there is no information on where the bumper bar was hit. In the short-term memory, property *rigidObject* is therefore set over the complete width of the bumper bar (see Section 9.1 on page 128). This approach is most pessimistic in the sense that it potentially assumes a hindrance larger than it is in reality. While this is appropriate for vehicle safety, availability may suffer if there was no possibility to test traversability at tight locations again. The fine-grained configuration of aspect *driveModeTactileCreepPROP* which is used for evasive manoeuvres and aspect *emergencyStopPROP* which is used for stopping the vehicle provides exactly this selectivity.

10.2 Configuration of Planning Aspects

The safety-critical *Behaviours* of the short-range navigation facilities realise tight sensor-actor loops which operate on local terrain information. Each individual *Behaviour* therefore only has a very limited impact on the overall behaviour. These limited operating ranges allows to consider the distinct short-term memories separately and to carry out fusion on the control level (see [Schäfer 05b] for details) using the behaviour-based fusion mechanisms of the control architecture [Proetzsch 10b]. In contrast, the planning components located on the mid-range navigation layer require a condensed and more complete view on the world which rules out control-level fusion. Following *Deferred Fusion* (Guideline 6), the next best option is representation-level fusion. Using the abstraction scheme and the fusion scheme outlined in Sections 4.2 on page 48 and 4.3 on page 51, the various data sources can be integrated to yield one coherent representation.

As a reminder, the property set *fusionPROP* designed for the local path planner was declared as follows (see Section 8.3.1):

$$\begin{aligned} \mathit{fusionPROP} = (&\mathit{traversable}, \\ &\mathit{nonTraversable}, \\ &\mathit{moderatelyTraversable}) \end{aligned}$$

Places marked as *traversable* should not contain any hindrances while *nonTraversable* is supposed to indicate locations which are evidently occupied by obstacles that are not negotiable by any means. For locations for which this cannot be decided from the distance, tag *moderatelyTraversable* is reserved. For the concrete platform RAVON this means that the status of these places is unclear but that a negotiation attempt using the robot's advanced tactile capabilities may be carried out (see Section 9.1). The configurable cost function (see Equation 8.3 on page 110) introduced in Section 8.3.1 allows to rate *moderatelyTraversable* patches of terrain more expensive than *traversable* locations. That way, the tradeoff between detours and the traversal of more difficult terrain can be modelled. Additional nuances can be integrated in a transparent fashion to model traversability more selectively.

Figure 10.3 resumes the boundary fence patrol scenario introduced in Sections 9.2.2 and 9.3.4 (see Figure 9.7a on page 136 for the scenario overview). The short-term memories from the particular sensor evaluation units are combined following the two-staged procedure outlined in Section 3.3.2 on page 32 (see Figure 3.8 on page 33 for a general overview). In the first stage, the semantics from the distinct data sources have to be rendered compatible. This is achieved by means of the generic *Semantic Translation* facilities (see Section 4.2 on page 48) which transfer the obstacle information from the short-term memories into abstract traversability information. In stage two, the resulting semantically compatible information bases are combined via *Abstract Fusion* which follows the fusion scheme introduced in Section 4.3 on page 51. Note that the key to the fine-grained modeling of traversability is the *Property Selectivity* (Guideline 12 on page 69) of the information sources. To demonstrate this, the configuration of the *Semantic Translation* for each source shall be derived in the following.

fusionPROP* ← *tactilePROP

$$\begin{aligned} \mathit{traversable} &= \mathit{freeSpace} \land \neg \mathit{rigidObject} \\ &\quad \text{with impact mode } \textbf{Overrule All} \\ \mathit{nonTraversable} &= \mathit{rigidObject} \\ &\quad \text{with impact mode } \textbf{Overrule All} \\ \mathit{moderatelyTraversable} &= \text{impact mode } \textbf{No Impact} \end{aligned} \tag{10.1}$$

The instrumented bumper on RAVON is an ultimate indicator for traversability. In case a *rigidObject* is detected during the attempt of negotiating a patch of terrain, the location in question seems to be *nonTraversable* for the robot. In contrast, if no bumper event occurs, the respective area is apparently *traversable*. Both traversability decisions are

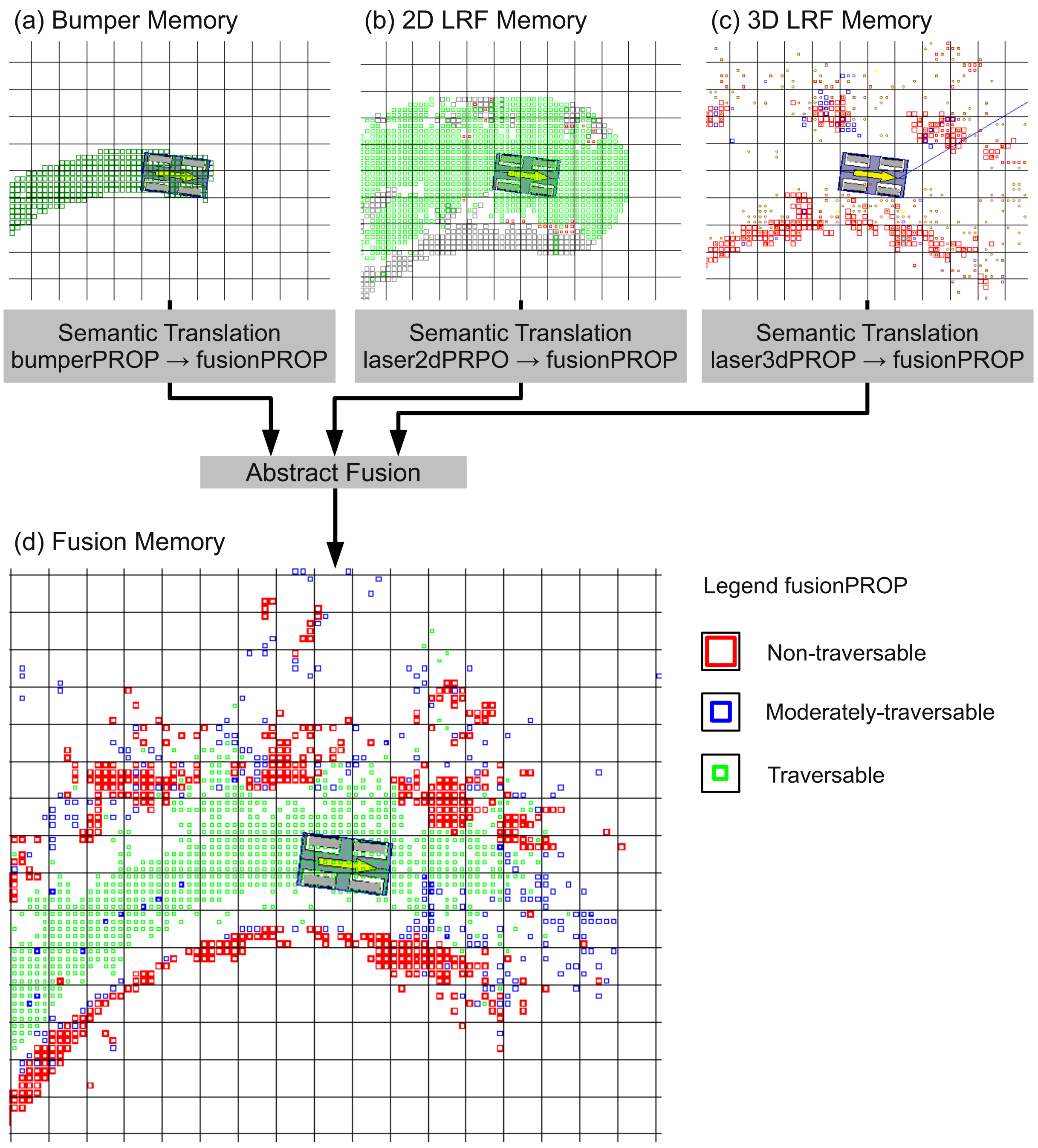


Figure 10.3: The source short-term memories (a), (b), and (c) are translated to a common semantic. After that, fusion is carried out on this more abstract semantic level resulting in a short-term memory containing condensed environment information (d).

highly reliable and therefore, *Impact Mode* **Overrule All** shall be used to stress this fact and overrule the information from other detection mechanisms. As the bumper system yields definite traversability information, property *moderatelyTraversable* is not influenced by this data basis. To model this, *Impact Mode* **No Impact** is specified.

fusionPROP* ← *laser2dPROP

$$\begin{aligned}
traversable &= \text{impact mode } \textbf{No Impact} \\
nonTraversable &= rigidObject \vee obstructedArea \\
&\quad \text{with impact mode } \textbf{Full Impact} \\
moderatelyTraversable &= vegetation \wedge \neg(rigidObject \vee obstructedArea) \\
&\quad \text{with impact mode } \textbf{Full Impact}
\end{aligned} \tag{10.2}$$

In comparison to the tactile detection mechanism discussed above, the 2D LRF yields less reliable traversability information. The 2D LRF on RAVON are mounted close to the ground with horizontal orientation. As data is only captured in a single plane, traversability cannot be decided in natural environments. Assuming locally flat terrain, property *non-traversable* can be decided from this data in the vicinity of the robot. Therefore, the 2D-laser-based data source should be configured to have **No Impact** on property *traversable* and **Full Impact** on property *non-traversable*.

fusionPROP* ← *laser3dPROP

$$\begin{aligned}
traversable &= ground \wedge \neg(overhanging \vee negative \vee positive \vee \\
&\quad highCentring \vee step \vee groundClutter) \\
&\quad \text{with impact mode } \textbf{Full Impact} \\
nonTraversable &= negative \vee overhanging \vee positive(middle \vee high) \vee \\
&\quad ((positive \vee highCentring \vee step) \wedge \neg vegetation) \\
&\quad \text{with impact mode } \textbf{Full Impact} \\
moderatelyTraversable &= \neg(negative \vee overhanging \vee positive(middle \vee high)) \wedge \\
&\quad (\\
&\quad\quad groundClutter \vee \\
&\quad\quad ((positive \vee highCentring \vee step) \wedge vegetation) \\
&\quad) \\
&\quad \text{with impact mode } \textbf{Full Impact}
\end{aligned} \tag{10.3}$$

The 3D LRF is the major source of environment information on RAVON as it captures the vicinity of the robot in three dimensions and yields a very selective description of the world. The configuration via the *PROP* interface allows for a fine-grained and straightforward configuration of the traversability classes required by the local path planner. Patches of terrain are *traversable* if a ground reference was detected and no obstacles are present.

Definitely *nonTraversable* are locations where critical obstacles like negative or overhanging entities are detected. Furthermore, positive obstacles which may cause damage to the sensitive mechanics and sensor systems on height levels *middle* and *high* have to be considered. Finally, ground structures like steps, high centring threats, and positive obstacles are critical if there is no indication for potential misclassifications induced by ground vegetation.

Traversability class *moderatelyTraversable* is designed to model the tradeoff between making a detour but travel on open terrain or using a shorter path while accepting to pass through more difficult structures. Patches with critical structures cannot be negotiated by any means, so the first part of the specification for *moderatelyTraversable* is the inverse of the first line of the *nonTraversable* specification. In contrast, clutter and further ground structures like steps, high centring threats, or steps which may be negotiable if accompanied by vegetation, are to be regarded as *moderatelyTraversable*.

Note that the short-term memory containing the fused data basis needs not share scope or resolution of the source short-term memories. For sensor processing, a smaller scope with high resolution may be appropriate while for planning purposes a larger scope at lower resolution might be preferred. All short-term memories can be configured according to the requirements of the application at hand.

Conceptually, abstraction and fusion scheme are semantically self-contained such that hierarchical applications are feasible.

On RAVON, aspect *openTerrainAttractionPROP* and aspect *passageDetectionPROP* are for instance configured from the abstraction level of aspect *fusionPROP*. Passage detection and open terrain attraction require a condensed information basis just like the planning facilities. Semantic abstraction may therefore be applied in a hierarchical fashion. The *Views* specified for passage detection and open terrain attraction are directly rendered from the fused short-term memory according to the following definitions:

$$(\mathit{openTerrainAttractionPROP}, \mathit{passageDetectionPROP}) \leftarrow \mathit{fusionPROP}$$

$$\begin{aligned} \mathit{openTerrainAttraction} &= \mathit{nonTraversable} \vee \mathit{moderatelyTraversable} \\ \mathit{passageDetection} &= \mathit{nonTraversable} \vee \mathit{moderatelyTraversable} \end{aligned}$$

This concludes the aspect-oriented configuration of the `Semantic Abstraction`. Note that even though some configurations are very similar (or even identical), the semantical distinction makes sense from the point of view of the particular designer declaring the property sets. On the configuration level, emergency stop and tactile creep for instance may require similar information bases but on the control level, they are two completely different aspects. The same could be said for instance about negative and overhanging obstacles. In the end, both are critical obstacles but from the designer of a sensor evaluation algorithm they are completely different concerns. Furthermore, premature simplifications would imply that the deployment platform is known a priori which limits the reusability of components.

11. Experiments and Results

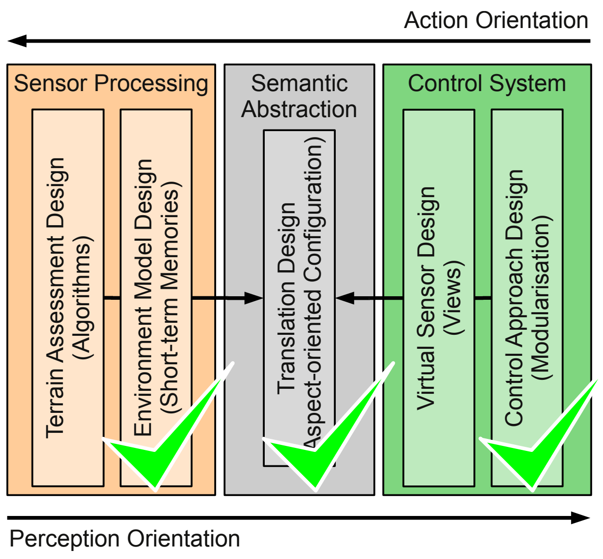

Action Orientation
Sensor Processing
Terrain Assessment Design (Algorithms)
Environment Model Design (Short-term Memories)
Semantic Abstraction
Translation Design (Aspect-oriented Configuration)
Control System
Virtual Sensor Design (Views)
Control Approach Design (Modularisation)
Perception Orientation

Figure 11.1: Having finished the aspect-oriented configuration in the third design step, the quality of the control system shall be evaluated in a series of experiments.

To conclude the application study illustrating the usage of the proposed design schemata in the context of the design methodology outlined in this work, RAVON shall be evaluated in a series of experiments.

The experiments are split into a qualitative part which was carried out in the real world and a quantitative evaluation in simulation. Quantitative results are difficult to obtain in real-world scenarios as the conditions cannot be held constant over a longer a period of time. To compensate for this fact, an elaborate series of experiments was carried out in the 3D simulation environment *SimVis3d* that is developed at the ROBOTICS RESEARCH LAB [Braun 07][1].

[1] *SimVis3d* is freely available under `http://rrlib.cs.uni-kl.de/software/simvis3d/`.

11.1 Experiments in Simulation

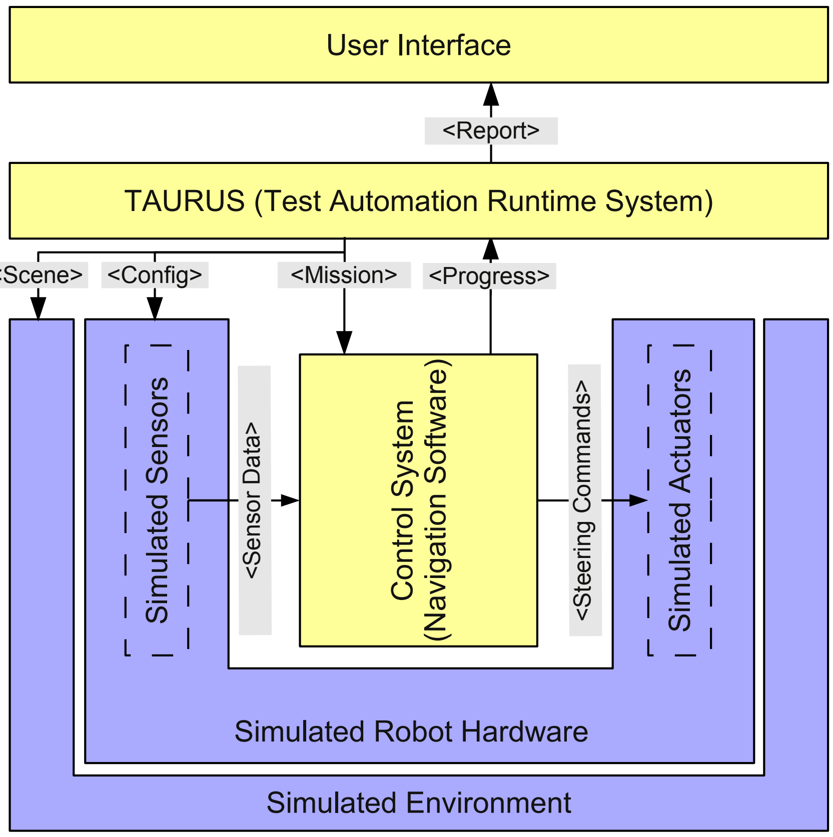


Figure 11.2: The test run automation facility TAURUS seamlessly docks in between the robot control system and the user interface.

In this section, the quality of the robot control system designed and implemented in the context of this application study shall be evaluated statistically. For that purpose, the Test Automation Runtime System (TAURUS) developed in [Proetzsch 10c] is used to carry out a suite of characteristic test cases. `TAURUS` is a generic test run automation framework which allows for the specification of complex *experiments* for the simulation environment *SimVis3d*. `TAURUS` *experiments* consist of a *scenario description* containing navigational challenges and a *mission*, the robot is supposed to accomplish. For the evaluation of navigational capabilities, *missions* may contain several *paths*, each consisting of a sequence of waypoints. Particular parameters of the simulated robot and its environment are scheduled for monitoring. In case a monitoring invariant is violated (e. g. by timeout, collision, etc.), respective data is journalised and cast into a test report after each test run.

The emergent behaviour of behaviour-based control systems tends to shadow the performance of particular competences by mutual interference. While this kind of robustness can generally be regarded as a major advantage of such architectures, it is difficult to evaluate the performance of individual (groups of) *Behaviours*. In order to illustrate the capabilities of the various layers of competence, the test run automation framework was extended with support for integration tests. In addition to the parameters indicated above, the *experiment* specification may further define different *configurations* of *Behaviours*

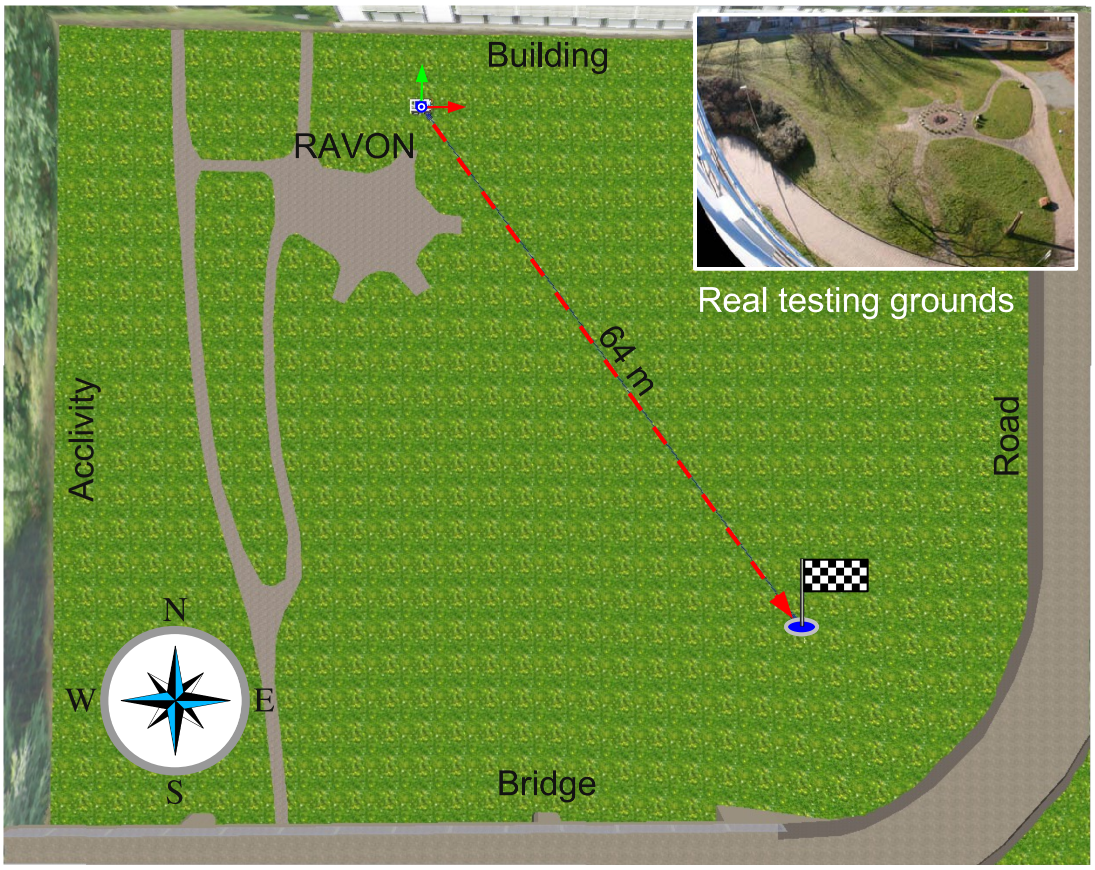


Figure 11.3: The *Open Terrain* scenario is a 3D model of the testing grounds near the ROBOTICS RESEARCH LAB without any obstacles.

to be activated during a test run. The *experiments* are repeatedly executed for each *configuration* to allow for the statistical assessment of the control system performance at each stage of extension.

As illustrated in Figure 11.2, `TAURUS` seamlessly docks into the live system between the `Control System` and the `User Interface`. `TAURUS` automatically switches *experiments*, *missions*, and *configurations* while monitoring progress and journalising the state of the control system. In this mode of operation, the `User Interface` merely serves for observing the execution of the tests. Active intervention is blocked by `TAURUS` to prevent accidental influence of the statistical test results.

`TAURUS` is also used to continuously monitor the progress of the control system development on RAVON and to detect faults accidentally introduced during the integration of new features. Nightly execution of the tests and automatic notification of the responsible developers allows to spot problems early in each phase of the implementation. A subset of these test cases was chosen for the documentation of the results of this Doctoral Thesis. In the following, the scenarios, configurations, and missions shall be introduced before going into detail with the experiments.

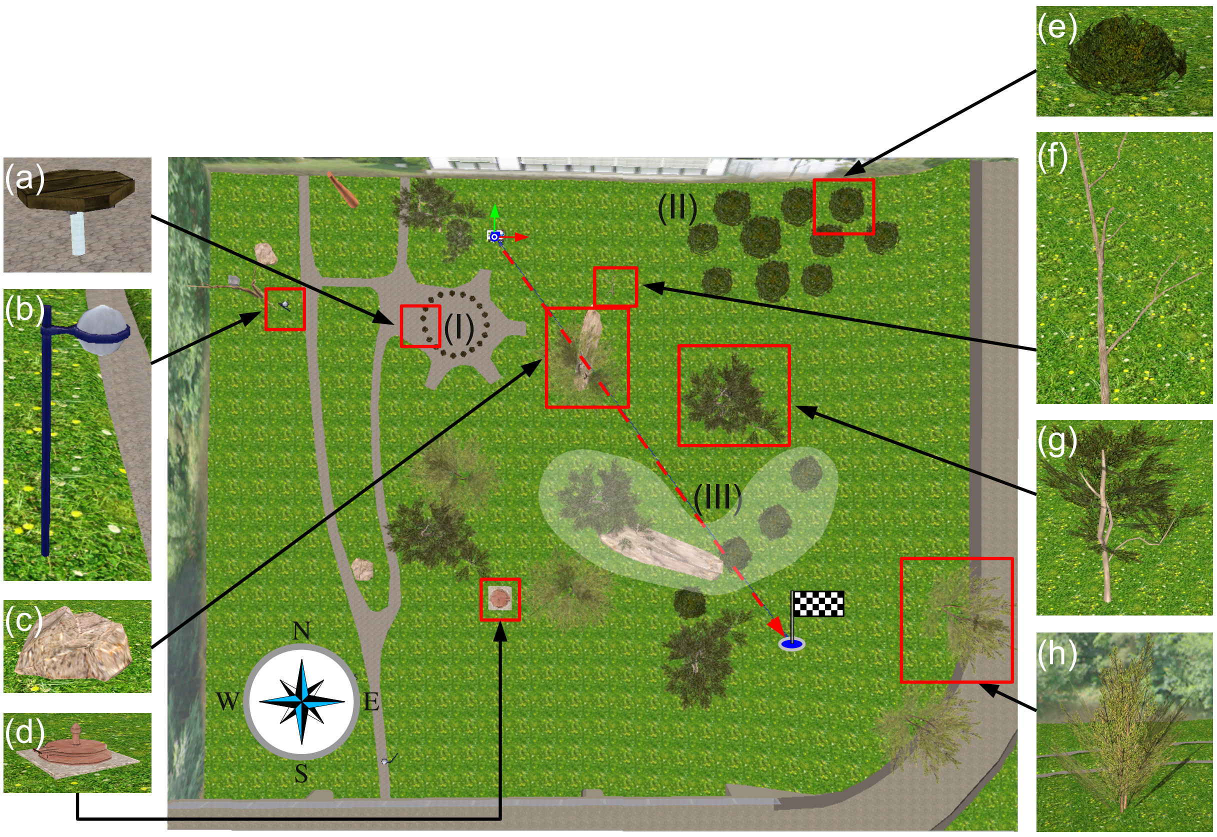


Figure 11.4: Scenario *Sparse Obstacles* features several isolated obstacles like trees, bushes, and stones which block the direct connection between the two waypoints.

11.1.1 Experimental Setup – Scenario and Mission Descriptions

The base scenario used for the experiments carried out in the context of this work is inspired by the real-world testing grounds nearby the Robotics Research Lab. Figure 11.3 shows the 3D model of the location in bird's eye view. The scene is a rectangular world of roughly 100 m × 80 m which is bordered by the building of the RRLab at the northern side, a bridge at the southern side, an acclivity to the west and a road in the east. A photograph of the real-world testing grounds taken from the roof of the building in the north is depicted in the upper right corner of Figure 11.3. In all scenarios, the mission of the robot is to travel from the starting point at the institute building in the north-west towards a target location in the south-east. The air-line distance between the two waypoints is about 64 m.

Scenario Open Terrain

The base scenario features undulating open terrain without any obstacles but the borders of the simulated world and minor bumps (see Figure 11.3). For the experiments, this scenario is used as ground truth and shall be referred to as scenario *Open Terrain*.

Scenario Sparse Obstacles

The second scenario is designed to test basic obstacle avoidance capabilities in a world populated with isolated objects. As illustrated in Figure 11.4, the direct connection between the two waypoints is mainly blocked by trees (f, g, h), bushes (e), and rocks (c). In particular the thin structures of the dead tree (f) are very hard to detect as there is no leafage increasing its signature in sensor data. Further objects like stools (a), lanterns (b), and the outlet of a water tank (d) complete the scene. The obstacle models are based on objects in the real world and were designed to challenge the obstacle detection facilities. In particular, overhanging tree branches and thin vertical structures are the main features. In this scenario, only three larger conformations have been placed, a circular stool arrangement (I), a group of bushes (II), and a small indentation (III) composed of trees, bushes, and a larger rock. While the former two will only play a peripheral role in this scenario, the indentation represents *the* benchmark for the short-range navigation facilities. For this mostly reactive component, this indentation is at the edge of what is possible with this approach.

Scenario Maze

The last scenario is a maze that covers a large portion of the simulated world. In addition to the isolated obstacles of scenario *Sparse Obstacles*, several larger conformations block the direct path to the target location. In Figure 11.5, the main topology of the maze is highlighted by red lines indicating blockades.

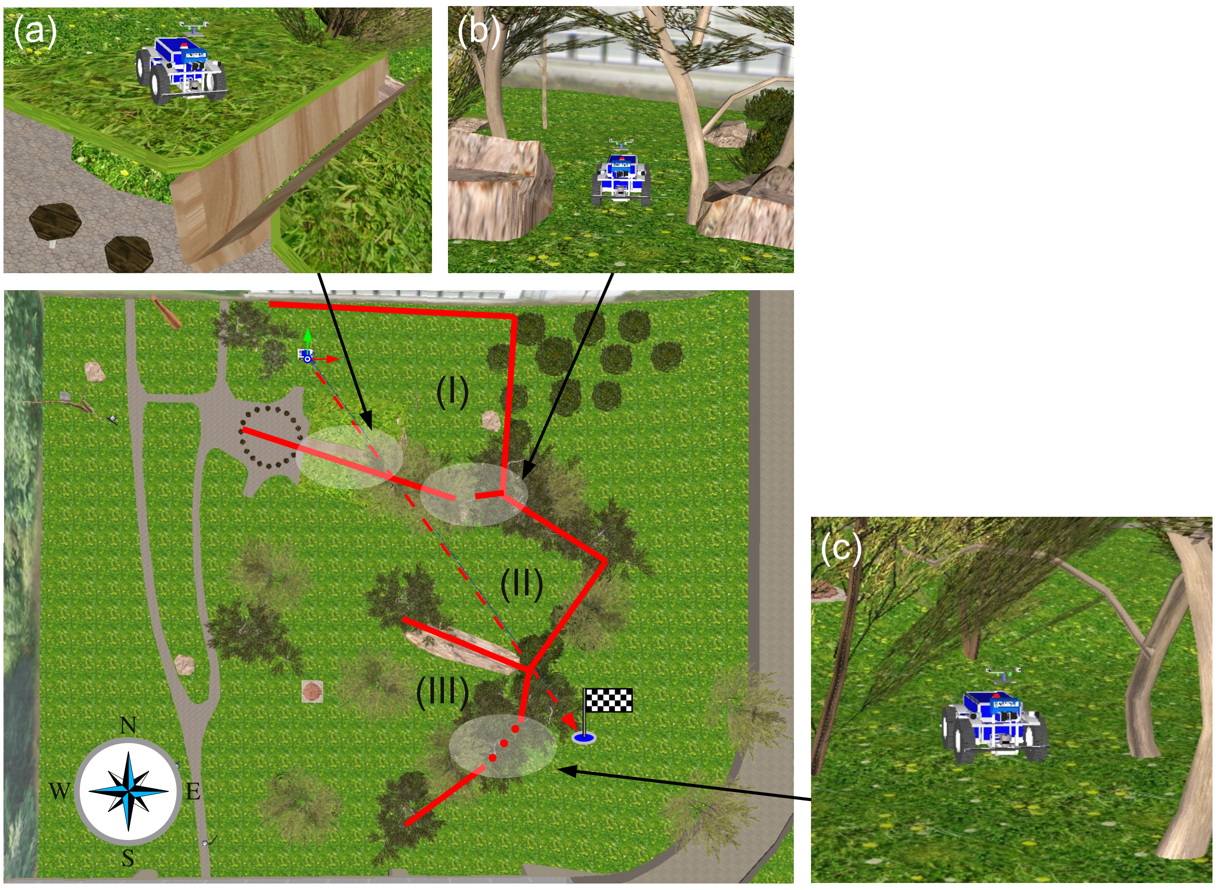


Figure 11.5: The *Maze* scenario features isolated obstacles and several larger conformations which block the path between the waypoints of the robot's mission. At several places, narrow passages provide an opportunity to shorten the distance to target.

The maze consists of three baggy conformations which have been marked with roman numbers (I) through (III). The robot starts in conformation (I), which only features a narrow passageway to conformation (II). This small exit is about 2.5 m wide and is flanked by several overhanging objects which render the approach quite difficult (see Figure 11.5 (b)). The second option to escape from conformation (I) is to accept a detour and use the exit between the stool circle and the institute. Conformation (II) is a dead-end which does not feature any exit towards the target. The only possibility to reach the goal location is to retreat from the direct course and to put up with the detour. Once trapped in this conformation, the reactive short-range navigation alone will not be capable of escaping from this structure again. The third conformation (III) is similar to conformation (I) in the sense that the direct way towards the target is not completely blocked by obstacles. In this case, a group of more loosely standing trees open up several narrow passages to shorten the path (see Figure 11.5 (c)).

The blockades are mostly composed of the obstacle types that have already been introduced in the second scenario. The sole exception is a trench (a) that occupies the space between the stool circle in the west and a group of trees in the east. This trench is based on drainage ditches which represent a tremendous challenge for the obstacle detection facilities. As a negative obstacle, this trench is difficult to detect from the distance. The narrow shape makes this even more complicated as the vehicle may anticipate the opposite side of the trench as continuing ground. The access to the trench is a ramp. The sides of ramps are a further difficult challenge, as these smoothly change from a negotiable bump to a non-traversable step structure. Since the trench is on the direct path to the target location, the robot is bound to be confronted with this benchmark obstacle.

11.1.2 Selection of Quality Parameters

For the evaluation of the control system quality, several parameters are scheduled for monitoring. Besides the logging of system parameters, `TAURUS` allows for the specification of invariants which provide information whether a specific parameter was inside acceptable bounds or not. These invariants may be terminal or non-terminal depending on how severe a violation has to be rated. Invariant violations are interpreted online and the test run supervision component terminates the current run in case any terminal invariant is affected.

In quality assessment, a common approach is to distinguish functional and non-functional requirements. For the evaluation of a navigation system, the functional requirement would be the *availability* of the system, i. e. whether the robot actually reaches the target location or gets stuck on the way. Non-functional requirements are vehicle *safety* and navigational *performance*. While vehicle safety clearly means the ability of the robot not to harm its environment or itself, performance refers to parameters like the time and the distance travelled until the target is reached. For each of these three quality classes *availability*, *safety*, and *performance*, two characteristic parameters are scheduled for monitoring during the experiment:

1. Availability
    - *Deadlocks*: If the robot does not move for a certain period of time, it is assumed to be stuck. Criterion *Deadlock* is modelled as a terminal invariant which results in mission failure via test run abort.

- *Livelocks*: In case the vehicle is not immobilised but fails to reach the designated target location in a time frame specified for the scenario at hand, another terminal invariant is violated.

2. Safety

   - *Number of collisions*: The major characteristic for vehicle safety is the obstacle avoidance performance. On RAVON, this parameter has to be evaluated with care, since the robot may by design get in contact with objects when in drive mode *Tactile Creep* (see Section 8.2.1 on page 96). Nonetheless, collisions have to be accounted for when occurring at velocities higher than creep velocity and in case the object in question is not hit by the bumper bar but another part of the chassis. In order to allow for a statistical evaluation of this parameter, collisions with objects in the environment are specified as a non-terminal invariant.
   - *Risk*: The second safety-related parameter, is the risk assessment of the `Navigator` component. In [Braun 09a] (see pages 60ff), a *Risk* measure for the traversal of a path between two waypoints is defined on the basis of the *Obstacle Avoidance Behaviours*' target ratings. Data for this measure is gathered on the move and reflects the evasive actions required on the way from one waypoint to another. Among others, [Braun 09a] deploys this *Risk* measure to model experience which is used to optimise the topological path planning. In order to remove time dependency from the *Risk* measure, a spatial integration was developed, which yields an absolute risk value for each path once traversed.

3. Performance

   - *Time to Target*: Navigational performance of a robotic vehicle can be measured by the time required to accomplish a well-defined mission. The faster the robot reaches the designated goals, the better. *Time to Target* is therefore registered for monitoring as a simple incremental value.
   - *Distance Travelled*: Another performance measure is the distance that had to be covered for reaching the destination. This value is often correlated with the *Time to Target* but may provide additional information. In some cases for instance, shortcuts do no reduce the *Time to Target* as the robot has to negotiate difficult hindrances on the way and must turn back in the end as the shortcut leads into a dead-end. *Distance Travelled* is also registered as a simple incremental value.

11.1.3 Configurations

The aim of the experiments carried out in the context of this application study is to prove the applicability of the proposed design methodology for the design of complex robotic systems. One particular point in this argumentation is that the methodology and the adjoined design schemata allow for incremental closing of gaps in the control system without introducing conceptual breaks. The central means in this context is the gradual and tailored abstraction of environment information and their representation in a uniform way. In order to show the gradual closing of gaps in the robot control system, automated integration tests shall be carried out.

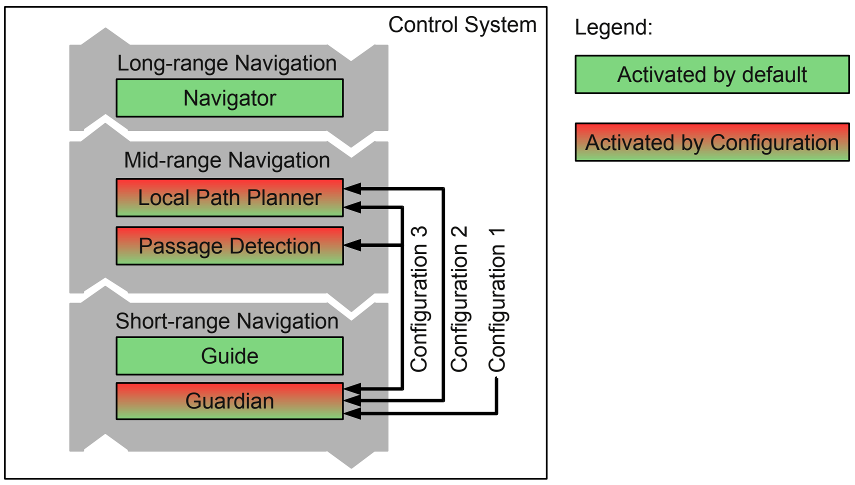


Figure 11.6: The control system is evaluated in three *configurations*. `Navigator` and `Mediator` draw the robot towards the target location.

For that purpose, three different *configurations* shall be evaluated in all scenarios outlined above. Figure 11.6 coarsely illustrates the composition of these *configurations*. In all scenarios introduced above, the mission of travelling from a given location to a target waypoint in about 64 m air distance shall be executed. Due to the simplicity of this mission, the `Navigator` only serves as target waypoint provider and does not have any impact on the results. On the basis of the global target waypoint provided by the *Navigator*, the `Mid-range Navigation` computes local waypoints which are passed to the `Guide`. The robot is drawn towards these intermediate waypoints by *Behaviour Target Approach* with support of the *Open Terrain Attraction* (see Section 8.2.2 on page 102). All other *Behaviours* of the *Guide* remain unstimulated during the experiments. Apart from these basic *Behaviours* which are always required for approaching a target, the remaining competences of the *Short-range* and *Mid-range Navigation* shall be considered in the integration tests using the following *configurations*:

1. Obstacle Avoidance (OA): In the first *configuration*, only the *Obstacle Avoidance Behaviours* are stimulated. In combination with the *Target Approach*, this rather reactive stage of extension can be assumed to master the two simple scenarios, while the *Maze* scenario is certainly beyond its capabilities.

2. Obstacle Avoidance + Local Path Planner (OA+PP): In order to provide the robot with more farsightedness, the second *configuration* makes use of the *Local Path Planner* in addition to the *Obstacle Avoidance*. This *configuration* represents a powerful combination which will succeed in all three scenarios. However, the path planner will certainly fail to pass through narrow passages such that detours have to be expected in scenario *Maze*.

3. Obstacle Avoidance + Local Path Planner + Passage Detection (OA+PP+PD): The third *configuration* represents RAVON's control system at the current stage of extension. In addition to *Obstacle Avoidance* and *Local Path Planner*, the *Passage Detection* is activated. In this *configuration*, the robot will master all scenarios and possibly with shorter time and distance to target than with *configuration* OA+PP.

11.1.4 Results of the TAURUS Experiments

As already indicated above, each scenario was executed multiple times with each *configuration*. To allow for the statistical evaluation of the experiments, 120 repeats were carried out for each combination of scenario and *configuration*. In total, 1080 (3 scenarios × 3 configurations × 120 repeats) test runs have been executed with an overall duration of about 90 hours. Statistical differences of data were checked by unpaired, two-tailed t tests, and labelled as follows: n.s.: Not significant; * p,0.05; ** p,0.01; *** p,0.001. In the graphs, standard deviations are indicated.

Scenario Open Terrain

Configuration	Deadlocks (%)	Livelocks (%)	#Collisions	Abs. Risk
OA	0	0	0	0
OA+PP	0	0	0	0
OA+PP+PD	0	0	0	0

Table 11.1: Summary of safety and availability parameters yielded in scenario *Open Terrain*.

Scenario *Open Terrain* serves as a ground truth to show that all configurations do comparably well in case no obstacles are present. Besides minor bumps in the ground, the robot did not encounter any navigational challenge and therefore no collisions, deadlocks or livelocks were registered for all configurations (see Table 11.1). Furthermore, the robot approaches the target location without any detour in all three configurations. In consequence, the distance travelled and the time elapsed until reaching the target location are very similar as illustrated in Figure 11.7. As no obstacles are blocking the path, risk is exactly zero.

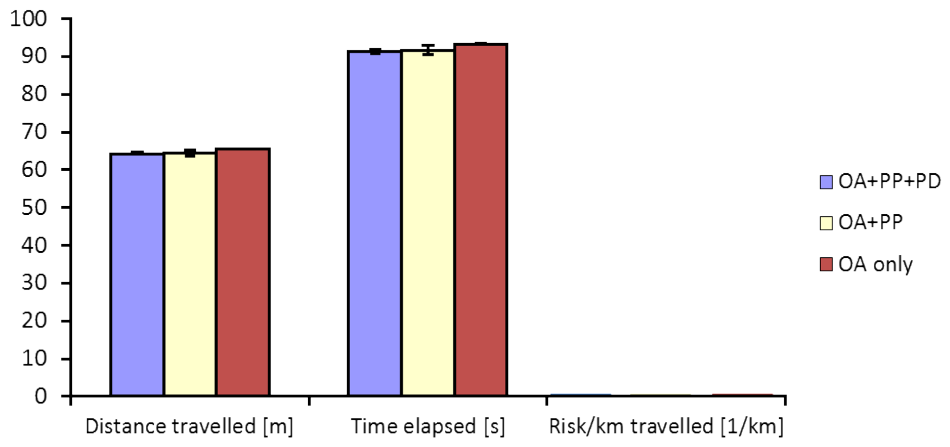


Figure 11.7: In the ground truth scenario *Open Terrain*, all configurations yield a similar navigation quality.
OA+PP+PD: Obstacle Avoidance, Path Planner, and Passage Detection are stimulated.
OA+PP: Obstacle Avoidance and Path Planner are stimulated.
OA: Only the Obstacle Avoidance *Behaviours* are stimulated.

Scenario Sparse Obstacles

In scenario *Sparse Obstacles*, configuration **OA** yields significantly worse results than the other configurations. Table 11.2 summarises the availability and safety parameters for all three configurations.

Configuration	Deadlocks (%)	Livelocks (%)	#Collisions	Abs. Risk
OA	0	18.3	0	66.46
OA+PP	0	0	0	9.83
OA+PP+PD	1.67	0	0	16.57

Table 11.2: Summary of safety and availability parameters yielded in scenario *Sparse Obstacles*.

In more than 10% of the test runs, configuration **OA** did not manage to reach the target location in time. As for the significance of this result, the availability of configuration **OA** can be rated as limited even for scenarios with sparse obstacles. On the upside, dead locks were not reported for configuration **OA**. Furthermore, the absolute *Risk* value is several times higher than with the other two configurations. This means that the robot got significantly more often very close to obstacles with configuration **OA** than with those incorporating planning activities. Nonetheless, all configurations managed to complete the scenario without any collision. The deadlock that was reported for configuration **OA+PP+PD** could be traced back to a situation in which the *Passage Detection* neutralised the *Target Approach* in an attempt to enter a *Passage* detected to one side of the robot. In several experiments, similar situations have been reproduced and in all cases the deadlock was dissolved after some time such that availability is not impaired. Nonetheless, this finding indicates a bug in the interaction of *Passage Detection* and *Target Approach*.

Figure 11.8 compares the performance parameters *Time to Target* and *Distance Travelled*, as well as the safety parameter *Risk*. In contrast to the table above, *Risk* is not regarded as an absolute value but relative to the distance covered. That way, the spatial dependence introduced by design to this measure is removed. Just like the safety and availability parameters, the performance parameters attest a relatively bad navigation quality for configuration **OA**. Even though failed missions were removed from the statistics, *Distance Travelled* and *Time to Target* are way worse in comparison to the other configurations. This proves that scenario *Sparse Obstacles* is really at the verge of what is possible with the rather reactive approach realised by the *Obstacle Avoidance Behaviours* of the `Guardian`.

While *Time to Target* and *Distance Travelled* are quite similar for configurations **OA+PP** and **OA+PP+PD**, parameter *Risk* unveils a significant discrepancy. The mid-range navigation techniques thus already unfold a certain effect in simple scenarios. This is not unexpected since the passage detection tries to find shortcuts towards the target which may bring the robot closer to obstacles than conservative planning. Figure 11.9 further underpins the findings that can be interpreted from the statistical data. This illustration is the basis for the spatial integration of the risk assessment facility of the `Navigator` component. In order to remove time dependency from the risk measure, the locations which cause an *Obstacle Avoidance Behaviour* to become active are projected into a coarse grid ([Braun 09a] pages 61ff). For each grid cell, the maximal *Target Rating Obstacle*

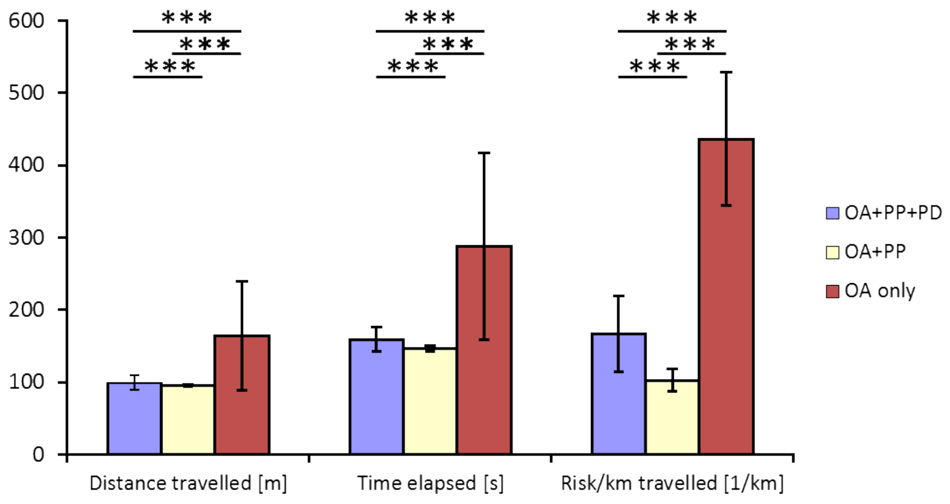
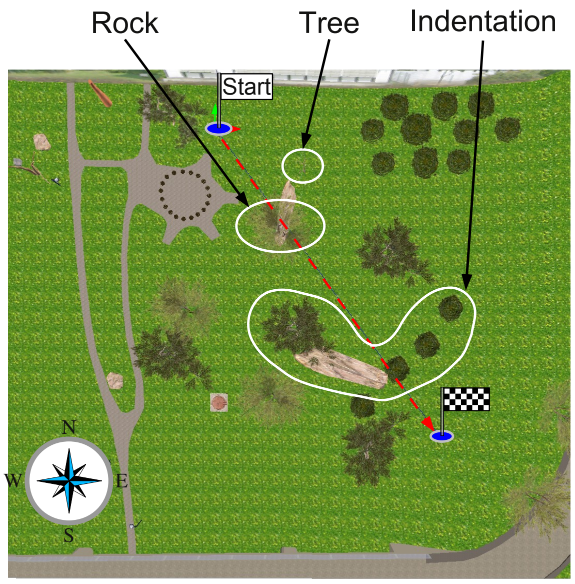


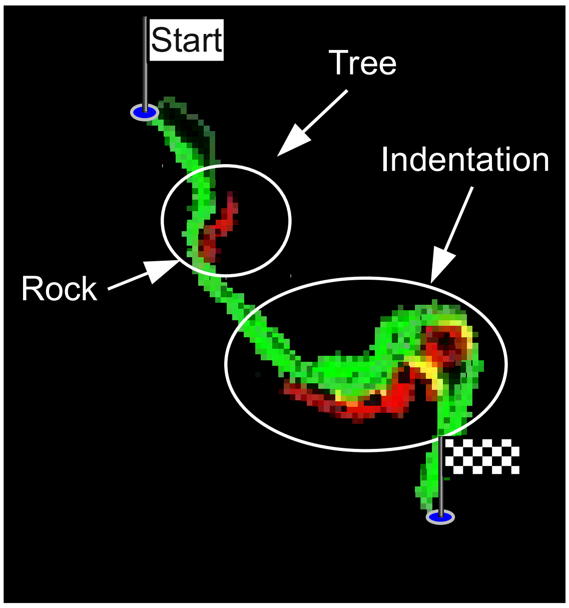
Figure 11.8: In scenario *Sparse Obstacles* configurations **OA+PP+PD** and **OA+PP** do comparably well while configuration **OA** is far behind.
OA+PP+PD: Obstacle Avoidance, Path Planner, and Passage Detection are stimulated.
OA+PP: Obstacle Avoidance and Path Planner are stimulated.
OA: Only the Obstacle Avoidance *Behaviours* are stimulated.
Levels of significance: n.s.: Not significant; * p,0.05; ** p,0.01; *** p,0.001

Avoidance Behaviour in question is stored. Furthermore, the cells the robot traversed on its journey are marked green. The illustrations show the average over all test runs carried out in a particular scenario with one particular configuration. *Risk*-inducing locations are highlighted with red colour while traversed terrain is indicated by green patches.

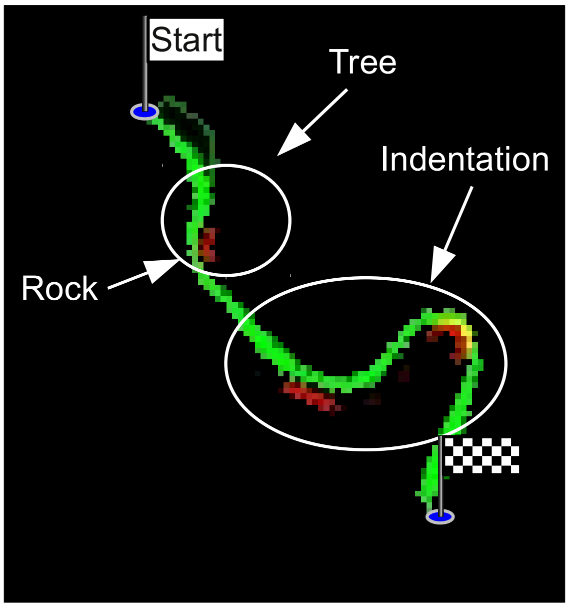
Figure 11.9b shows that the navigation strategy of configuration **OA** often gets trapped in the small indentation nearby the target location. After several attempts, the robot usually finds a feasible path but the short-sightedness of the *Behaviours* do not allow for a systematic approach of this challenge. In consequence, the robot accumulates a high *Risk* value. In contrast, the behaviour of configuration **OA+PP** is highly reproducible and keeps the robot away from obstacles (see Figure 11.9c). As a marginal remark it can be stated that configuration **OA+PP** fails to traverse the small passage between two bushes of the indentation which was sometimes achieved by configuration **OA**. As illustrated in Figure 11.9d, configuration **OA+PP+PD** yields a lot less reproducible trajectories than configuration **OA+PP**. The passage detection mechanism is more sensitive to timing issues and sensor noise than A*-based planning. The emergent behaviour of the robot therefore appears a lot more creative in finding a way to the target than strict path planning. The increased *Risk* introduced by the passage detection can be traced back to several tight openings in the indentation close to the target location. These openings are detected as wide enough from the distance but when actually trying to pass through, they turn out to be too narrow for the vehicle. The passage that was intensely used by configuration **OA** was also found by the passage detection several times. In this scenario, detours are not a high penalty and therefore the advantages of the passage detection do not take effect. The more remarkable however, that this strategy yields comparable statistical results for *Time to Target* and *Distance Travelled*.



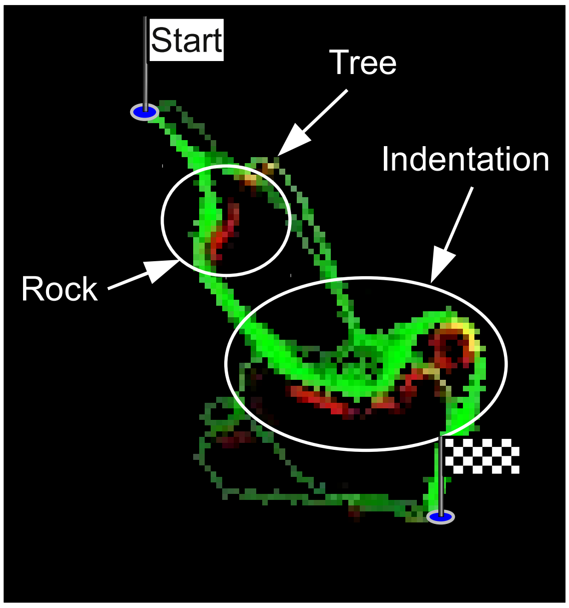
(a) Scenario *Sparse Obstacles*



(b) OA



(c) OA+PP



(d) OA+PP+PD

Figure 11.9: Average trajectory maps of the risk assessment facility of the `Navigator` component for scenario *Sparse Obstacles*. The trajectories travelled are highlighted in green while objects contributing to the *Risk* assessment are marked red.

Scenario Maze

Scenario *Maze* is the most complex environment in these experiments. The dead-end structures placed in this scenario require the robot to call its farsighted planning facilities into service. Furthermore, several tight situations provide opportunities to shorten the path to the target location. These conditions have a significant impact on the statistical data collected in the test runs.

Configuration	Deadlocks (%)	Livelocks (%)	#Collisions	Abs. Risk
OA	16.67	83.33	0.23	–
OA+PP	0	0	0.091	87.38
OA+PP+PD	0	0	0.1	67.79

Table 11.3: Summary of safety and availability parameters yielded in scenario *Maze*.

Table 11.3 gives a brief overview of the safety and availability parameters monitored. For configuration **OA**, the baggy structures are apparently an insurmountable hindrance. This rather reactive strategy never really got close to the target location. The remaining distance to the target measured after mission abort averages to more than 25 m. As the risk assessment is only carried out by the `Navigator` when the path between the two waypoints was actually completed, no *Risk* values are available for configuration **OA** in this scenario. The analysis of the partial trajectories showed that already conformation (I) (see Figure 11.5) could not be escaped in several runs. In all other runs, conformation (II) was the ultimate destination for the reactive navigation strategy. For the other configurations, the availability parameters do not show any significant impairment.

As in the scenarios before, collisions were only reported on rare occasions and all of these occurred at the bumper bar with only slightly elevated velocities. *Risk* however shows a highly significant deviation for strategies **OA+PP** and **OA+PP+PD**. Most interestingly, configuration **OA+PP+PD** appears to endanger the robot less than configuration **OA+PP** as indicated by the *Absolute Risk* values. This is somewhat unexpected at first sight because the *Passage Detection* tends to draw the robot into more tricky situations.

Figure 11.10 resolves this astonishing fact. As already mentioned above, *Absolute Risk* is a measure that is spatially integrated. Therefore, the distance covered until the destination is reached, plays an important role. The performance parameters *Distance Travelled* and *Time to Target* unveil, that configuration **OA+PP** took significantly longer than configuration **OA+PP+PD** to approach the target location. While the mean distance travelled was about 170 m for the latter, the former needed more than 240 m. The *Risk per km travelled* shows, that configuration **OA+PP** still keeps the vehicle better out of trouble than configuration **OA+PP+PD**. Yet, by keeping the robot away from obstacles, the path planner has to accept detours which may come with a high *Risk* penalty. Configuration **OA+PP+PD** may thus compensate the higher *Risk* per distance by actually finding shorter paths to the target. Configuration **OA+PP+PD** can therefore be attributed a better performance than configuration **OA+PP**. Furthermore, vehicle safety is affected in the sense that the former strategy is more willing to take a risk but may finally turn out to endanger the vehicle and its environment less.

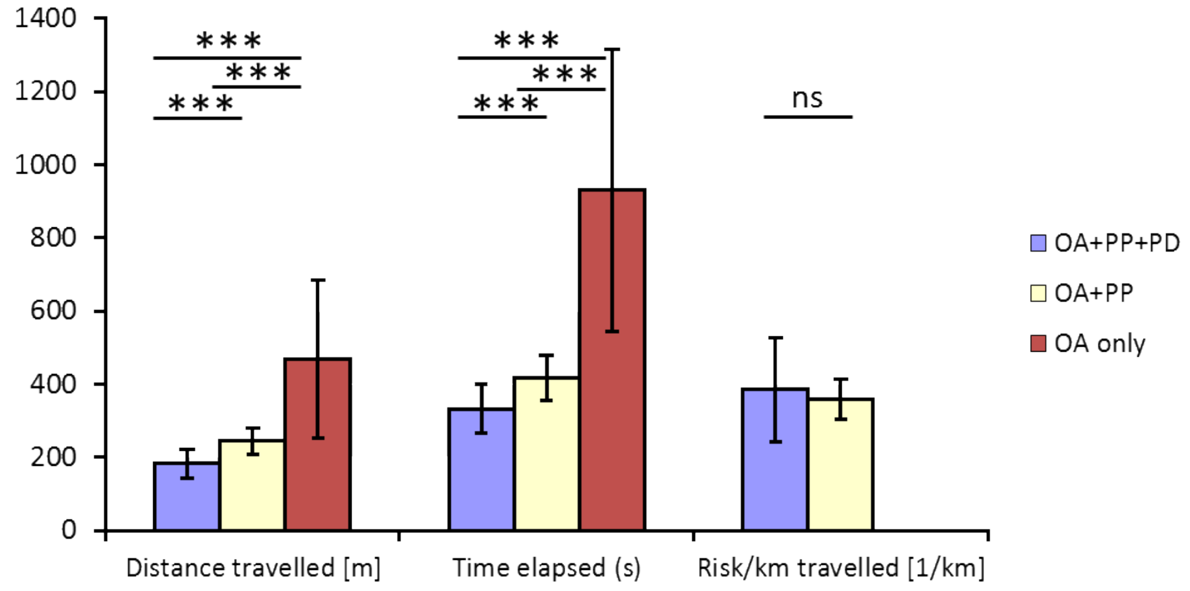
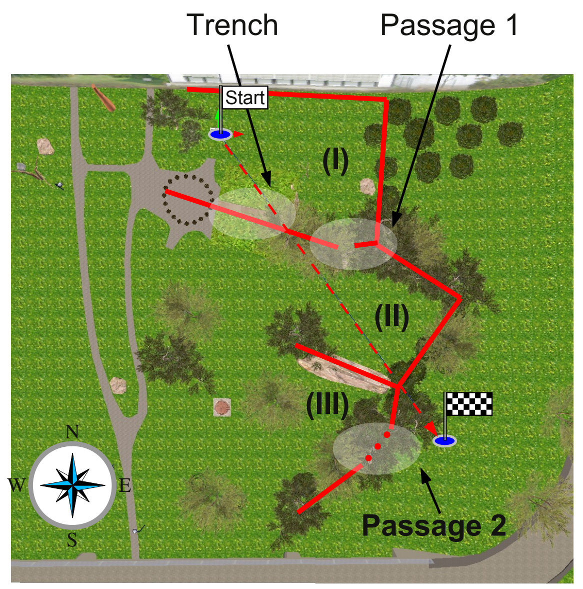


Figure 11.10: In scenario *Maze*, configuration **OA** does not complete any run successfully. Configurations **OA+PP** and **OA+PP+PD** both master the scenario but the former takes significantly longer and travels larger distances. The inverse is true for the normalised *Risk*.
OA+PP+PD: Obstacle Avoidance, Path Planner, and Passage Detection are stimulated.
OA+PP: Obstacle Avoidance and Path Planner are stimulated.
OA: Only the Obstacle Avoidance *Behaviours* are stimulated.
Levels of significance: n.s.: Not significant; * p,0.05; ** p,0.01; *** p,0.001

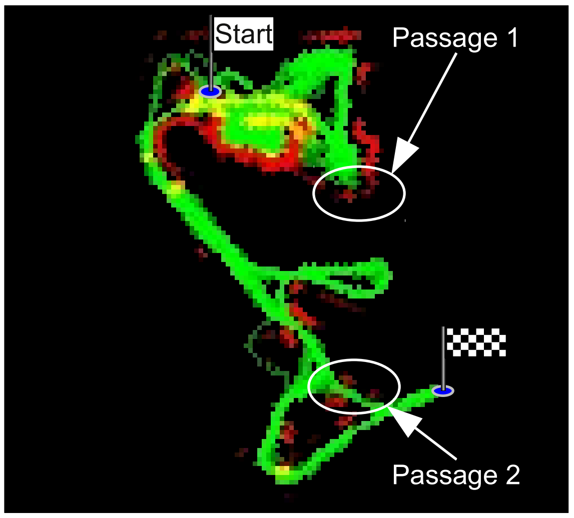
The trajectory maps of the `Navigator`'s risk assessment facility underpin the statistical results. Figure 11.11 shows the scenario overview (a) and the averaged spatial *Risk* data for configuration **OA+PP** (b) and configuration **OA+PP+PD** (c). As in scenario *Sparse Obstacles*, the routes chosen by configuration **OA+PP+PD** are more variable than those of configuration **OA+PP**. This results in a larger number of different intensively used routes towards the target. In the following, the major routes of both approaches shall be discussed step-by-step.

The conservative planning strategy fails to exit baggy structure (I) via the narrow *Passage 1* at the eastern side of the structure. Therefore, a clear main route can be identified in the trajectory map for this part of the mission. Having circumnavigated the stool circle, the robot approaches baggy structure (II). At this point, two major routes can be identified. Either the robot is sidetracked into the dead-end by the draw to the target, or structure (II) is not entered. In the former case, the path planner easily figures out that there is no exit and proceeds to structure (III) after a short detour. In structure (III), the vehicle also finds two options to approach the target location. The most intensively used route leads south-west around the group of trees which represents a larger detour. As a second option, *Passage 2* is used which is the shortest path to the target.

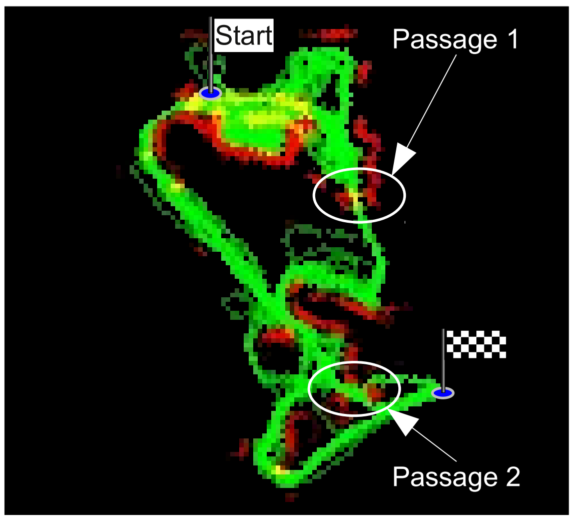
OA+PP+PD manages to negotiate the tight opening of *Passage 1* in about 50% of the test runs saving the detour around the stool circle. However, this shortcut leads directly into baggy structure (II) which only features an exit in the west. This leads the robot away from the target again but the dead-end is negotiated without greater



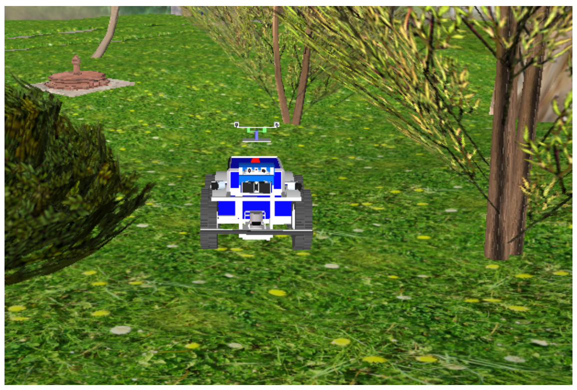
(a) Scenario *Maze* features three baggy structures of which two may be escaped via *Passages*.



(b) OA+PP: This strategy never manages to find *Passage 1* but in some test runs, *Passage 2* was successfully negotiated.



(c) OA+PP+PD: This strategy manages to negotiate *Passage 1* in about 50% of the test runs. Apart from *Passage 2*, the *Passage Detection* found several more openings between the loosely standing trees.



(d) An additional *Passage* leading out of structure (III) was found in several test runs with configuration **OA+PP+PD**.

Figure 11.11: Average trajectory maps of the risk assessment facility of the `Navigator` component for scenario *Maze*. The trajectories travelled are highlighted in green while objects contributing to the *Risk* assessment are marked red.

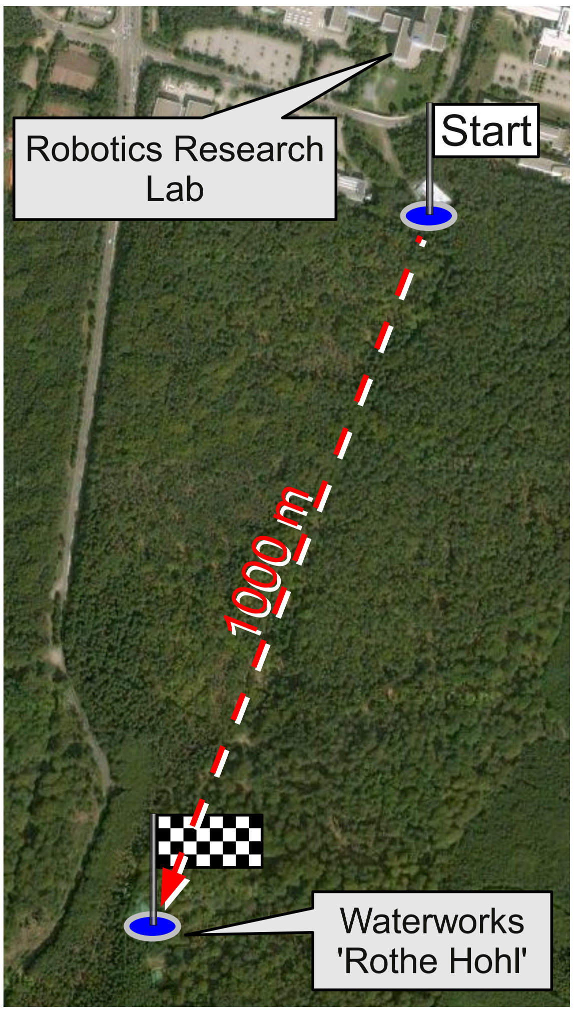
problems. Even more interesting about the second part of the mission is the fact that configuration **OA+PP+PD** apparently never enters structure (II) when taking the detour around the stool circle. The opening between the large rock forming the northern border of structure (III) and the tree further south is detected as a *Passage* leading towards the target. Therefore, the robot is drawn into structure (III) saving the detour over structure (II). For the *Passage Detection*, structure (III) appears like a leaky cauldron. Apart from the detour around the group of trees, two further intensively used paths leading through *Passage 2* can be identified in Figure 11.11c. In some cases however, a fourth option is followed which leads through another opening between a bush and a tree (see Figure 11.11d). While the position of the tree trunk makes this passage appear relatively wide, the overhanging branches make this situation difficult to negotiate for the robot.

In summary, the *Passage Detection* is a suitable building block in bridging the gap between purely reactive approaches towards autonomous navigation and classical planning. The flexibility introduced by this mechanism clearly supports the conservative planner and improves the overall performance of the system. The integration tests further prove the applicability of the proposed design methodology with the adjoined schemata and guidelines. The ability to seamlessly combine various levels of competences spanning the whole bandwidth of reactive towards deliberative control approaches shows the potential of unification of representation, abstraction, and fusion of environmental information. The homogeneity of the control approach furthermore facilitates the integration of extensions.

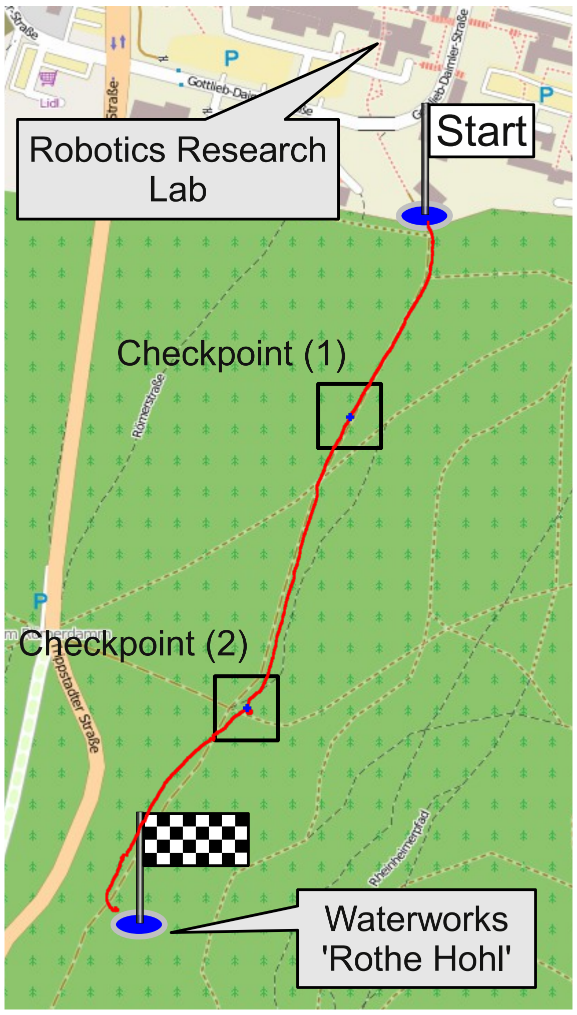
11.2 Experiments in the Real World

In this section, the results yielded in simulation shall further be discussed on the basis of experiments in the real world. As already alluded above, the effort for carrying out equivalent tests in reality as in simulation is immense. A sound statistical evaluation as conducted in simulation is therefore beyond the scope of this work. Numerous individual experiments in varying scenarios have been carried out with RAVON in recent years. In order to document that the statistical results can be transferred to the vehicle's behaviour in the real world, individual test runs carried out in comparable environments will be presented. As in the previous section, the focus will be on demonstrating the capability of the proposed design methodology and schemata to bridge the gap in robot control.

11.2.1 Mission *Waterworks*

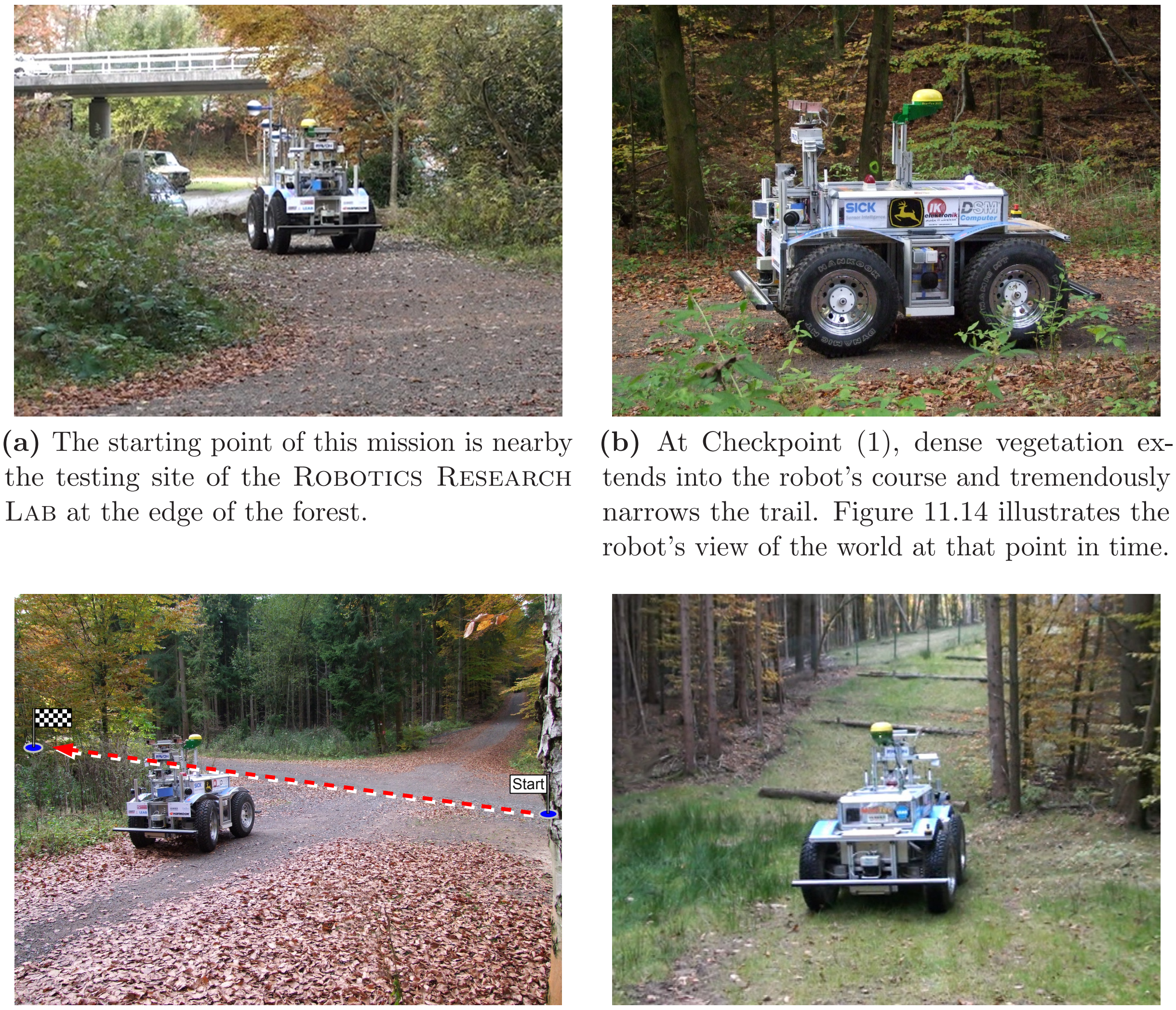


(a) Satellite view of the mission area indicating the airline distance. This figure was produced using Google Maps[1].



(b) Map view of the mission area with overlaid pose trace and highlighted checkpoints. This figure was produced using Open Street Map[2].

Figure 11.12: Mission *Waterworks* is quite similar to scenario *Maze* in the previous section.



(a) The starting point of this mission is nearby the testing site of the Robotics Research Lab at the edge of the forest.

(b) At Checkpoint (1), dense vegetation extends into the robot's course and tremendously narrows the trail. Figure 11.14 illustrates the robot's view of the world at that point in time.

(c) At Checkpoint (2), ravon enters a promising side road which turns out to lead away from the target. Figure 11.15b illustrates the robot's view of the world at that point in time.

(d) The target of this mission is the waterworks "Rothe Hohl". On this photo, the fence of this facility is already visible.

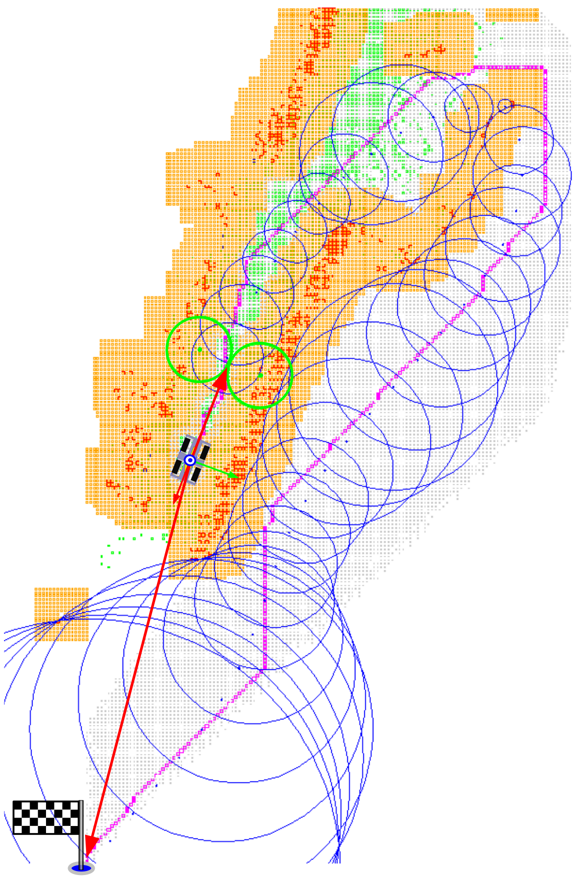
Figure 11.13: Mission *Waterworks* features several interesting challenges for autonomous systems. The two checkpoints illustrated in (b) and (c) shall be discussed in more detail.

In this mission, ravon is supposed to navigate from the edge of the forest nearby the RRLab testing site to the waterworks "Rothe Hohl". As in the simulated experiments, the robot is merely provided with the coordinates of the target. Apart from the on-board sensors, no further information of the environment is available to the robot. As illustrated in the satellite image in Figure 11.12a, the waterworks is located at about 1 km airline distance from the starting point. Figure 11.12b shows the mission setup which is similar to scenario *Maze* introduced in the previous section. The route to the waterworks leads through a forest area and can be reached via hiking trails. On the way, several crossings

[1]Google Maps^{TM} is a registered trademark of Google Inc. – `http://maps.google.de/`
[2]Open Street Map – `http://www.openstreetmap.de/`

provide options for changing the heading. Furthermore, several tighter passages have to be negotiated to reach the target location. The actual route taken by RAVON in this experiment is indicated by the red pose trace overlay. Two characteristic locations which shall be discussed in the following have been marked as Checkpoint (1) and Checkpoint (2). In order to give the reader an idea about the mission environment, Figure 11.13 shows the course at different locations. Apart from the starting point (a) and the target location (d), Figure 11.13 illustrates the terrain at the two checkpoints (b), (c).

Checkpoint (1)


(a) The route planned by the `Local Path Planner` would have caused the robot to turn back.

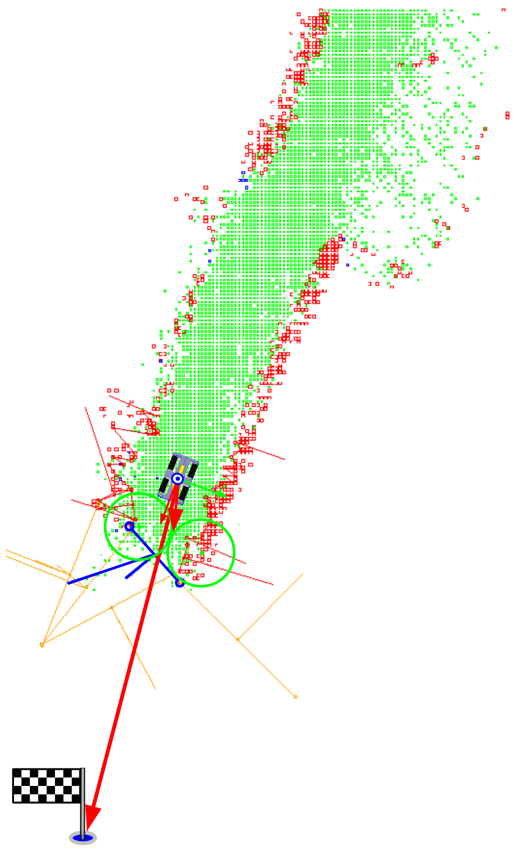

(b) The `Passage Detection` identifies the narrow passage as suitable and leads the robot through the tight location.

Figure 11.14: Checkpoint (1) is a tight spot where vegetation extends into the trail. The views of the `Local Path Planner` (a) and the `Passage Detection` (b) show that the latter takes control in order to guide the robot towards the goal.

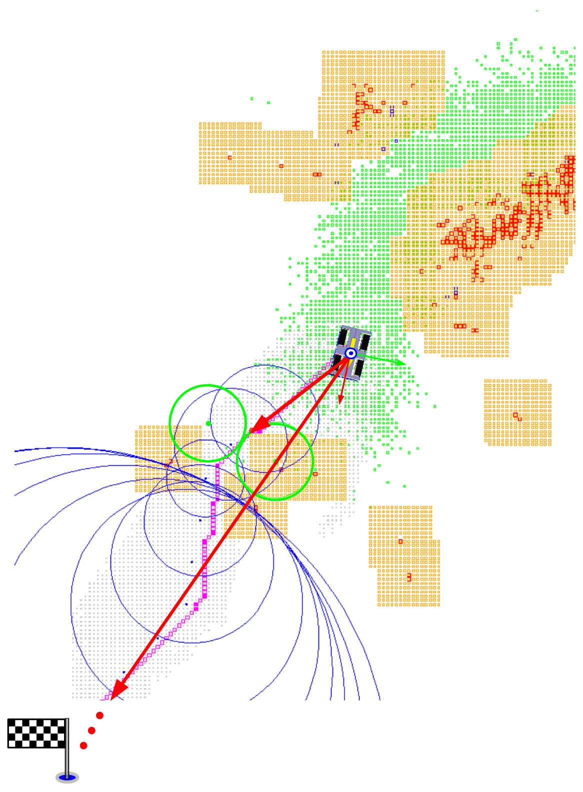
At Checkpoint (1), the robot has to negotiate a tight part of the trail which is bordered with dense vegetation extending into the robot's way (see Figure 11.13b). As illustrated in Figure 11.14a, the trail appears to be too narrow and the `Local Path Planner` proposes a detour which leads the robot into the opposite direction. However, the `Passage Detection` identifies a narrow passage which is traversable and leads towards the goal. The orientation estimation is pretty poor because the robot's current point of view does not provide much information on the characteristics inside the passage. The bottleneck representing the passage entry shadows most of the terrain ahead. Nonetheless, entering the passage is granted as it appears to be a good opportunity to reach the target location. In consequence, the `Local Path Planner` is inhibited and the `Passage Driver` leads the robot through the passage.

Note that the route proposed by the `Local Path Planner` in this situation would have lead the robot into a dead-end. The opening on the planned detour only exists because this place was not scanned yet. Having a look at the map depicted in Figure 11.12b it is obvious that the robot would have to return almost to the very beginning of the mission and take the long way round to reach the waterworks. This checkpoint shows that the interaction between both facilities that was observed in simulation also yields superior results in real-world scenarios.

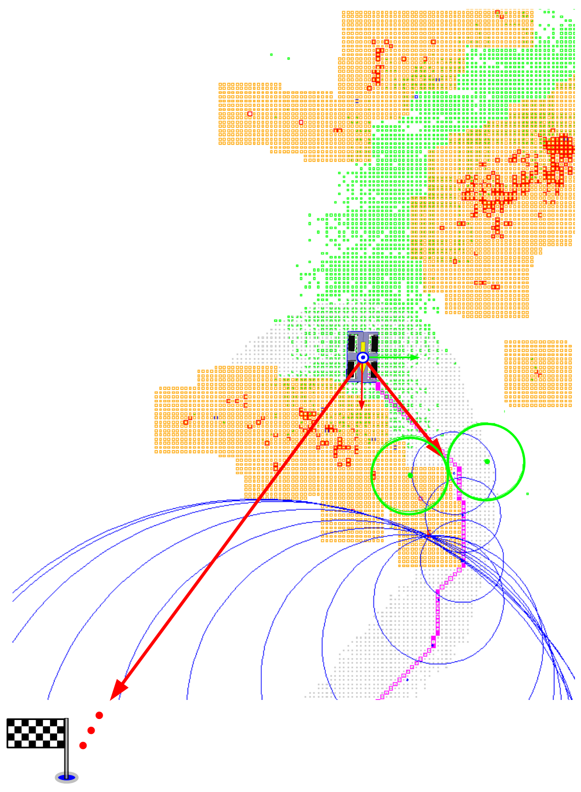
Checkpoint (2)

At Checkpoint (2), RAVON reaches a clearing which offers four options to continue (see Figure 11.12b). Figure 11.13c shows the robot's situation at that point in time. As no map material is used in this experiment, the robot initially continues straight towards the target location which draws the vehicle to the left of the clearing. Figure 11.15a shows RAVON at the junction entry to its left. The clearing represents a large patch of open terrain such that the robot initially perceives the bordering structures only in part. As the robot gets closer to the edge of the clearing, the initially detected opening leading straight to the waterworks turns out to be non-traversable (see Figure 11.15b). From the current point of view, the left turn appears to be more promising to the `Local Path Planner` such that RAVON follows this side trail. The vehicle continues to explore the chosen trail and gathers more and more information about its environment to update the decision taken (see Figure 11.15c). Note that in this phase, the `Local Path Planner` already starts to consider returning to the clearing as is indicated by property *processed* (cells marked grey). Further down the trail it becomes apparent, that this route leads the robot away from the target location and the `Local Path Planner` starts an attempt via the second option. The robot detects the trail leading south and continues towards the waterworks. Note that this situation would not have been negotiated without the `Local Path Planner`. Neither the *Obstacle Avoidance Behaviours* nor the *Passage Detection* has the farsightedness to realise that the chosen trail represents a tremendous detour. At the next crossing would have been a second chance to resume towards the waterworks.

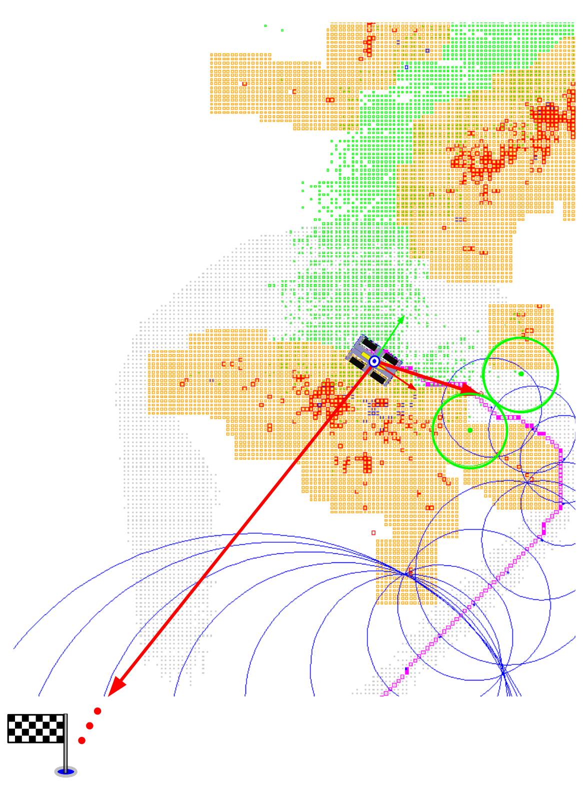
Earlier attempts to reach the waterworks without the *Mid-range Navigation* capabilities failed due to the semantic gap in the robot's control system. The rather reactive *Obstacle Avoidance Behaviours* of the *Short-range Navigation* got trapped in obstacle conformations at an early stage of the mission. The attempt to follow a rather dense series of global waypoints failed due to localisation inaccuracies in the *globally stable pose*. The dense foliage of the high trees in this forest area results in large GPS errors, which is a tremendous problem if waypoints are close to one another. In that case, the *Long-range Navigation* chases moving targets and the robot does not reach its destination. This shows that complementing *and* overlapping *Behaviours* are required to build a robust control system (*Competence Overlap* (Guideline 7)).



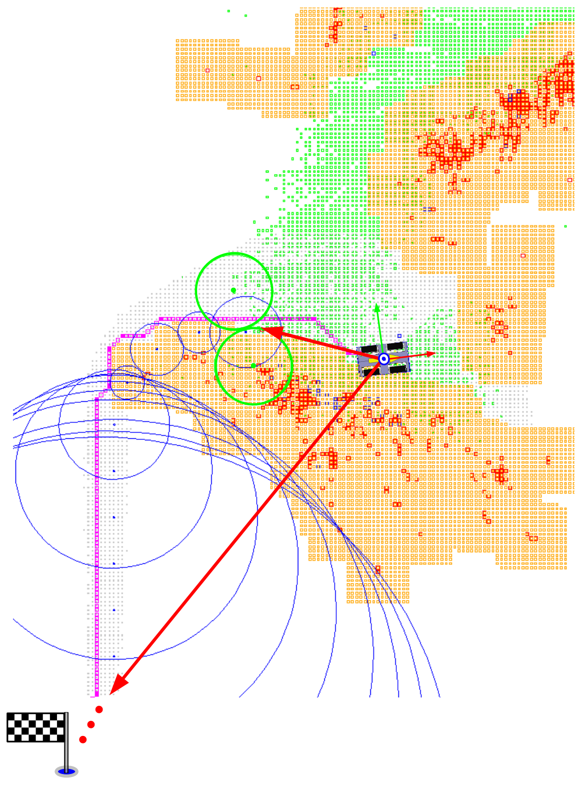
(a) The direct path towards the target draws RAVON to the left side of the clearing.



(b) Initially, the left turn appears more promising from RAVON's point of view (see Figure 11.13c).



(c) As no map material is used in this experiment, the robot continues exploring the trail for several metres.



(d) Finally, the `Local Path Planner` decides to return to the clearing as the way continues to lead away from the target location.

Figure 11.15: Checkpoint (2) is a clearing which challenges the robot with several options.

12. Conclusion and Outlook

In this Doctoral Thesis, a methodology for the design of complex control systems was developed. Embedded into this methodology, a set of design schemata were proposed which assist at guiding the design process. The major focus in that context was the *unification* of information *representation*, the *transformation* of representation between different semantics, and the *fusion* of information on common levels of abstraction.

At the beginning of this thesis, it was stated that system-level design in robotics often merely deals with the question of how to decompose a complex control system into less and less complex subsystems. For that purpose, many architectures propose modularisation schemata which are based on more or less formalised component models which specify fundamental building blocks. In some cases guidelines for the interaction between components are provided. However, these guidelines are often focussed towards a specific class of interactions. In reactive and behaviour-based architectures for instance, the fusion of control data is often strictly regulated while sensor data flow is not schematised in any way. The same holds for the design of representation. Many facets of design thus remain unconstrained which results in rather poor scalability.

The reference models which have become popular in the control engineering community represent the other extreme on the scale. The rigorous prescriptions of such deliberative hierarchical architectures overregulate the design process. In domains where tasks are difficult to rigorously specify a priori, this overspecialisation represents a tremendous burden. Iterative and incremental design is virtually impossible as the resulting control systems are by design monolithic.

Due to their heterogeneous nature hybrid approaches cannot provide a uniform component model. Prescriptions therefore only address the partitioning of control systems into reactive and deliberative layers. Furthermore, guidelines are provided which assist at assigning tasks to the particular layers. The interaction between components is not constrained which makes interface definition a core problem of these approaches. As interfaces may easily grow complex, hybrid architectures are difficult to extend in an incremental fashion.

In summary, control architectures only provide appropriate modelling techniques for a subset of design tasks. Therefore, flexible design schemata are required which strike a balance between standardisation and genericity.

12.1 Evaluation of the Proposed Design Methodology, Schemata, and Guidelines

In this work, a novel design methodology was presented which aims at the thorough schematisation of representation, abstraction, and fusion of information to solve core challenges in robot control. In the following, the results of this attempt shall be presented.

Challenges in Robot Control and Solutions in "Natural Mobile Systems"

In Part I of this work, the proposed schematisation was presented on a theoretical level. As a first step, the core problems in robot control were investigated in order to focus schemata design and methodological considerations. In that context five major challenges, namely *Data Integration* (Challenge 1 on page 21) of sources with different timeliness across time and space, *Configuration Dependence* (Challenge 2 on page 21) of control systems and the limited reusability among different platforms, the *Limited Field of Vision* (Challenge 3 on page 22) of sensor systems, the *Semantic Discrepancy* (Challenge 4 on page 23) between different levels of control, as well as selection of appropriate actions to assure *Control Handover* (Challenge 5 on page 25) between different control strategies were identified.

With these technical issues in mind, the author discussed architectural solutions in nature in order to find out, why "natural mobile systems" are more successful in performing navigational tasks than robotic systems. The aim of this attempt was to identify structural principles rather than trying to rigorously mimic nature at any price. These principles, were cast into design ideas which guided the subsequent schemata design and the development of the proposed design methodology. Evidently, "natural mobile systems" also have the problem of a limited visual range.

However, cats for instance remember obstacles which have left their field of vision for a longer period of time. This idea promotes the deployment of a *Short-term Memory* (Design Idea 6 on page 31) to keep in mind the robot's current situation. In the style of sensor centres in the brain, *Perceptional Separation* (Design Idea 1 on page 28) postulates that *Configuration Dependence* can be minimised by strictly separating perception into self-contained units which rely on a small set of sensor values. These perceptional units can be thought of as filters which derive semantic entities out of data streams from individual senses. In consequence, fusion of information from several perceptional units should preferably be carried out subsequent to the actual extraction of semantics using the concept of *Deferred Fusion* (Design Idea 2 on page 28) to address *Data Integration*.

Inspired by brain models in cognition science which assume generic translation mechanisms between the semantics of different perceptional units, the *Semantic Coupling* (Design Idea 4 on page 30) of information shall be realised in a schematic fashion. Such a generic translation mechanism could be used to lift data from different sources onto a common semantic level (i. e. a common abstraction) such that information can be combined in a straightforward fashion (*Abstract Fusion* (Design Idea 5 on page 30)). This approach further provides the basis to achieve a strict separation of sensor processing and control design (*Sense-Control-Duality* (Design Idea 3 on page 29)) as *Semantic Discrepancy* is modelled and bridged in an explicit fashion.

Schematisation of Representation, Abstraction, and Fusion of Information

On the basis of the design ideas derived from observations in nature, an integrative *design methodology* was developed which combines the strengths of *action-oriented* and *perception-oriented* design approaches without inheriting their weaknesses. In order to support the proposed design methodology, central design ideas were cast into *schemata for representation, abstraction, and fusion of information.* Since representation is a crucial point in data storage and communication, information modelling was regarded as the most delicate part of the proposed schematisation. Therefore, schemata design was carried out in a representation-centred approach. In that context, three design points were followed to achieve the desired degree of standardisation while leaving room for flexibility in terms of well-defined *extensions.* The fundamental thought guiding the representation scheme design was to separate *structure* strictly from *content* (*Structure-Content-Duality* (Design Point 1 on page 38)). While *structures* hold information on locality, *content* represents the semantics tied to entities at a given location. That way, structural transformations can be carried out independent of the translation of semantics and vice versa allowing for the design of generic standard *handlers. Structures* represent the core element of standardisation in the proposed schemata. In contrast, *Content* and *Handlers* feature a standardised base which may be extended on demand using a generic *extension* mechanism (*Content / Handler Standardisation and Genericity* (Design Points 2 on page 38 and 3 on page 38)). On top of the standardised parts, the semantic translation scheme and the fusion scheme were designed. Finally, guidelines for the application of the proposed schemata in control system design have been summarised. This compact reference manual was used as a basis for an application study conducted on a real-life robotic platform.

Schemata Validation in an Application Study

In Part II of this work, the theoretical concepts for schematisation were validated in the design of a control system for the off-road platform RAVON which is developed at the ROBOTICS RESEARCH LAB at the UNIVERSITY OF KAISERSLAUTERN. This study can be regarded as a long-term experiment as it summarises roughly five years of research in off-road robotics.

According to the proposed action/perception-oriented design methodology, control-level design was carried out in an action-oriented fashion. Tasks were decomposed according to the modularisation scheme and adjoined guidelines of the behaviour-based architecture iB2C [Proetzsch 10a]. This procedure yielded a hierarchical behaviour network featuring components which cover the complete spectrum of control strategies (from purely reactive towards fully deliberative) in a coherent fashion. For each component, representation stubs were defined according to the proposed representation scheme. These stubs comprise structural as well as semantic specifications which represent the control-level input for the configuration of the abstraction scheme.

As proposed by the methodology, sensor processing design was carried out in a perception-oriented fashion. Several complex algorithms have been presented which operate on a variety of different data sources, ranging from simple tactile facilities over laser range data towards specialised camera systems. As sensor-level input for the configuration of the abstraction scheme, representation was specified according to the representation scheme.

Finally, abstraction and fusion scheme were configured in an aspect-oriented fashion to distribute information on an as needed basis. On the basis of this configuration, the semantic translation and the abstract fusion generate the requested control-level views from the sensor-level short-term memories in a transparent fashion.

The operational control system was first of all intensely validated using integration testing techniques which represent a central concept in the iB2C development process. In these experiments, the performance of various configuration was evaluated in a statistical fashion. These results document the applicability of the proposed methodology for gradually bridging semantic gaps in robot control systems. The control system was furthermore deployed on RAVON to validate the statistical data in complex real-world scenarios. The control system properties observed in simulation could qualitatively be confirmed in numerous experiments including several successful participations at the European Land Robot Trials (see Appendix B.5).

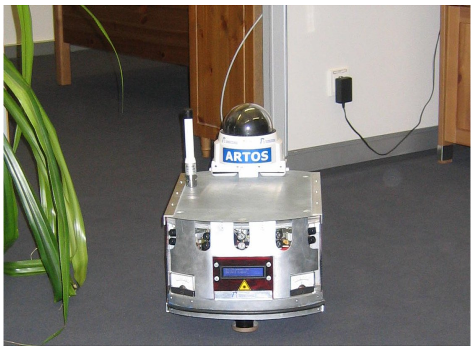
The presented results demonstrate the applicability of the proposed methodology, schemata, and adjoined guidelines to complex real-world control tasks. The proposed design schemata for representation, translation, and fusion of information establish thorough abstractions between components. The conducted application study showed, that the careful application of the proposed guidelines yields inherently extensible and scalable control systems. Furthermore, the design methodology supports large teams of developers as control tasks can be broken down into well-defined components with configurable communication interfaces.

As already stated in Section 3.1, developers working on sensor processing have a different point of view towards representation than control-level designers (remember Example 4 on page 49 (Sense-Control-Duality)). Furthermore, these two groups of people often have a different professional background and favour different development approaches (*action-oriented design* versus *perception-oriented design* – see Section 3.3.2). Bridging the (vertical) *Semantic Discrepancy* (Challenge 4) resulting from these facts was a major aim in the design of the proposed schemata. The thorough abstractions of the representation scheme in combination with the translation mechanisms of the abstraction scheme allow to deal with *Semantic Dispense* in an explicit fashion. That way, the strengths of action orientation and perception orientation were successfully combined into one coherent methodology. Note that these mechanisms may also be used to integrate third-party components and to share software between different target platforms in a straightforward fashion.

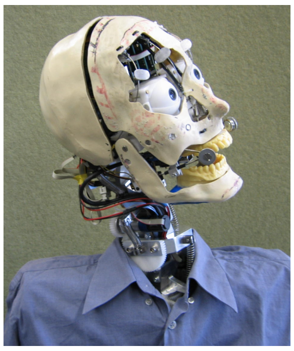
12.2 Future Perspectives

The application study conducted in Part II is nothing but the beginning of the applicability of the proposed design methodology and the proposed schematisation. Several subsystems have already successfully been shared among the ROBOTICS RESEARCH LAB's indoor platforms MARVIN[1] [Wettach 10] (see Figure 12.1a) and ARTOS[2] [Armbrust 10b] and the off-road robot RAVON using the design schemata presented in this work. This comprises sensor processing facilities, as well as entire *Behaviour* networks. The inherent support for modularity via thorough abstractions allows for the fine-grained tailoring of interfaces according to the requirements of the specific platforms.

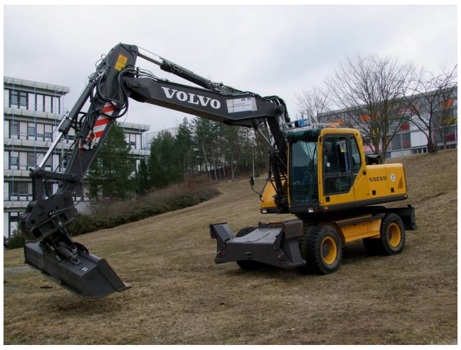
(a) MARVIN in an office environment.



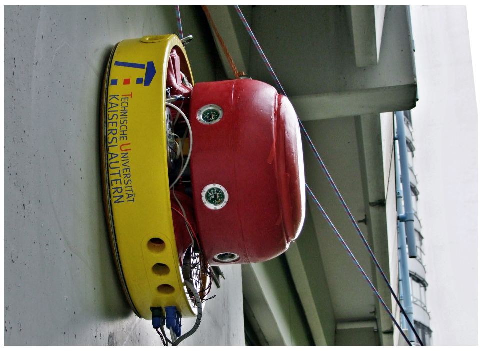
(b) ARTOS in a living room.



(c) The robotic head ROMAN .



(d) AMOBA on the testing ground.



(e) CROMSCI climbing up a wall.

Figure 12.1: The indoor platforms MARVIN and ARTOS and the off-road robot RAVON already share several components designed according to the schemata proposed in this work. In the future further platforms shall be targeted.

As already alluded in Section 2, design schemata usually evolve when casting development experience into a methodology. Even though design schemata are in the first place independent of any concrete application, their capabilities can only be unleashed in practical real-life challenges. In the future, the foundations laid in this work shall therefore be deployed on more platforms developed at the ROBOTICS RESEARCH LAB in order to continue schemata development. This comprises in particular the autonomous excavator

[1] MARVIN → Mobile Autonomous Robotic Vehicle for Indoor Navigation

[2] ARTOS → Autonomous Robot for Transport and Service

(see Figure 12.1d) developed in the context of the AMOBA[3] project [Hillenbrand 10b] as well as the climbing robot CROMSCI[4] [Hillenbrand 10a] (see Figure 12.1e) and the robotic head ROMAN[5] [Mianowski 07] (see Figure 12.1c). The latter two represent crucial benchmarks for the applicability of the proposed methodology apart from the domain of autonomous navigation. A first workshop for initiating this attempt has yielded promising perspectives. The expressiveness of the proposed schemata in general allows for the application to any target platform. Therefore, the more interesting point of investigation is the design of suitable extensions embedded into the schemata to solve specific challenges on the particular platforms. For instance, the design of suitable representations for grasp planning with ROMAN's arms [Hirth 10] or the maintenance of communication situations will certainly unveil completely different demands than navigation tasks.

In the domain of autonomous off-road robotics, the design schemata shall be employed to fuse topological maps extracted from various Geo Information Systems (GIS) [Fleischmann 10]. The representation scheme provides topological maps which allow for the annotation of nodes and edges with arbitrary information. Fine-grained information may therefore be extracted from GIS data bases and modelled in a straightforward and uncondensed fashion. The proposed abstraction scheme can then be used to translate this data for instance into traversability information for arbitrary platforms. Furthermore, the fusion scheme can be extended with graph merging algorithms to fuse multiple sources of information in a similar way as shown in this work.

Another exciting point of investigation is whether the aspect-oriented configuration can be used as a basis for optimising the semantic translation between different semantic levels using learning techniques. The symbolic nature of these mappings may provide the formal basis to formulate learning problems in a suitable way. The test run automation represents a further building block in this attempt. In this context failure might be used as feedback to learning systems without exposing the real robot to danger.

In conclusion the proposed design methodology contributes to the aim of providing thorough design schemata for the development of complex robot control systems. This work furthermore invalidates the criticism – particularly issued by the behavioural community – that the prescriptive parts of representation schemata would impair development flexibility (see Section 1.2). The proposed representation scheme successfully strikes a balance between prescription and genericity such that the schematisation actually promotes flexibility due to the enhanced conceptual clarity. The application study conducted in the context of this work showed the applicability of the proposed methodology to the design of a complex real-world robotic system. The inherent modularity and extensibility of the resulting control system shows the scalability of the proposed approach which is beyond what can be achieved with contemporary architectures.

[3]AMOBA → Autonomer Mobiler Bagger
[4]CROMSCI → Climbing RObot with Multiple Sucking Chambers for Inspection tasks
[5]ROMAN → RObot-huMAN interaction machine

A. Appendix

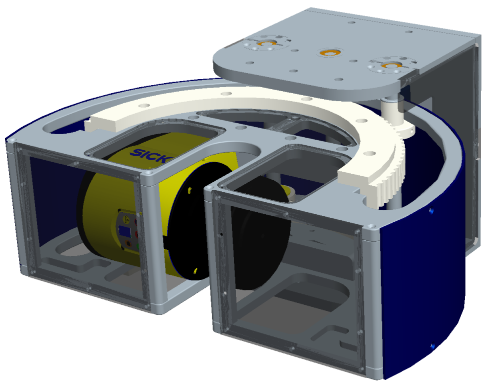
A.1 Deployed Laser Range Finders

The following table summarises the technical data of the 2D LRF used in this work. For additional information visit `http://www.sick.de/`.

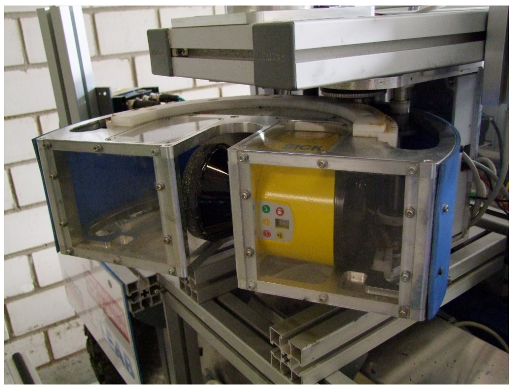
Scanner Type	SICK LMS291	SICK S300	SICK LMS111
Field of View	180 °	270 °	270 °
Angular Resolution	1 ° / 0.5 ° / 0.25 °	0,5 °	0.5 ° / 0.25 °
Frequency (Hz)	100 / 50 / 25	25	25 / 50
Systematic Error (mm)	$\pm 35, \pm 5$	$\pm 20, \pm 5$	± 40
Statistic Error (1σ in mm)	10	± 28	20
Laser Class	1	1	1
Enclosure Rating	IP 65	IP 65	IP 67
Operating Temperature (°C)	$0 \ldots +50$	$-10 \ldots +50$	$-30 \ldots +50$
Max. Distance (m)	80	32	20
Data Interface	RS-422	RS-422	Ethernet
Service Interface	RS-232	RS-232	RS-232
Transfer rate	500 kbaud	500 kbaud	100 Mbit
Power Supply	24 V DC	24 V DC	24 V DC
Power Consumption (Watt)	20 (without heater)	8	12
Storage Temperature (°C)	-30 ... +70	-25 ... +70	-30 ... +70
Weight (kg)	4.5	1.2	1.2
Dimensions W×H×D (mm^3)	$155 \times 210 \times 156$	$102 \times 152 \times 105$	$102 \times 152 \times 106$

A.2 LRF Actuation Unit

In order to obtain a 3D representation of a robot's environment, a common approach is to actuate a planar laser range finder in a controlled fashion (see Section 9.3.1).



(a) CAD model of the 3D LRF.



(b) 3D LRF mounted on RAVON.

Figure A.1: CAD model (a) and real mechanics (b) of the LRF actuation unit designed in the context of this work.

The actuation unit developed in the context of this Doctoral Thesis is a panning mechanism which rotates a SICK S300 such that the scanning centre is precisely on the rotation axis. The mechanical design is such that the full 270° of the deployed 2D LRF are usable as vertical field of vision. The horizontal field of vision is up to 135°. With 25 Hz the SICK S300 yields a horizontal resolution of about 5° when panning at one pass per second.

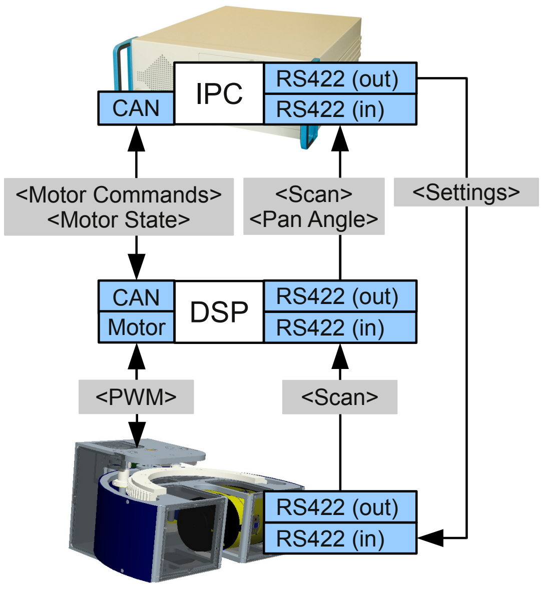
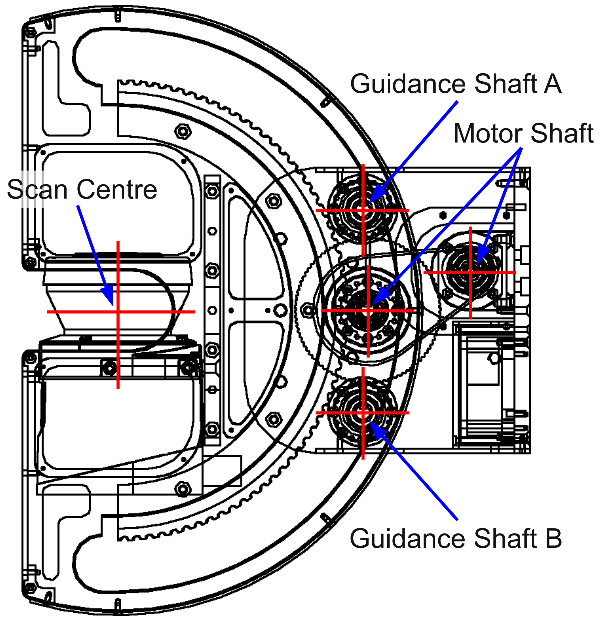
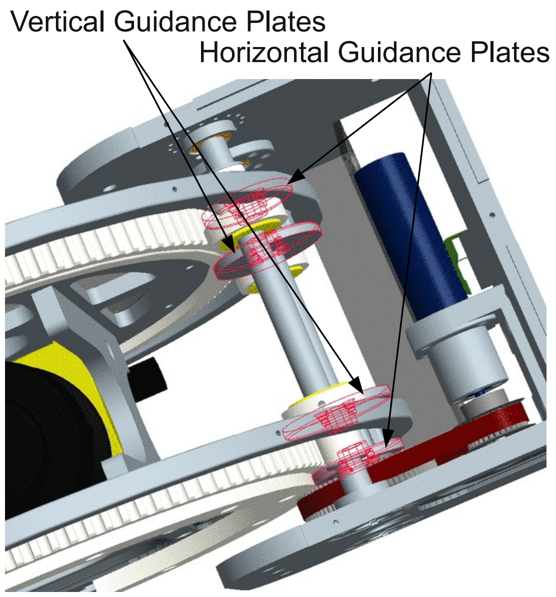


Figure A.2: Electronic connection scheme of the 3D LRF.

The control software for the 3D LRF provides a dynamic velocity range as well as the possibility to limit the pan angle to the left and right side independent of each other. That



(a) The LRF is attached to a carriage which is suspended on three guidance shafts which reside on a circular path around the LRF's scan centre.



(b) The central guidance shaft transmits the torque provided by a DC motor to the carriage.

Figure A.3: Guidance system of the LRF actuation unit.

way, a smaller horizontal cone can be shifted actively in order to focus on a particular location. This feature can for instance be used to adapt the field of vision in curves.

The embedded controller is based on the custom electronics line developed at the ROBOTICS RESEARCH LAB [Hillenbrand 09]. The basic DSP board may be equipped with various interfaces for connecting hardware via a modular plug on system. As illustrated in Figure A.2, the board assembled for the 3D LRF features a CAN[1] interface, a motor output stage, and a fast serial I/O chip (RS422). The control software runs on an Industrial PC (IPC) and sends commands to the DSP via the CAN interface. The sensor carriage is actuated with a PWM[2]-controlled DC motor[3] which is connected to the DSP board via the motor output stage. The actuation angle is controlled by feedback from the motor encoder. In order to associate actuation angle and data from the scanner as early as possible, the scans are routed through the DSP. Apart from the motor controller, the DSP programme features a state-based analysis system which determines start and stop markers of each scan in the RS422 data stream. For that purpose, the scans are routed through the DSP which forwards the incoming data byte-wise while constantly monitoring for the start and stop markers of scans. Under consideration of the LRF timing, the correct actuation angle is associated to each scan and transferred to the IPC in an additional data packet following the actual scan. That way, offsets between registered scanner data and the pan angle can be avoided.

Recently, this approach was also adopted by the industry and the latest LRF series provide

[1]Controller Area Network (CAN)
[2]Pulse-width Modulation (PWM)
[3]Direct Current motor (DC motor)

an on-board solution for associating encoder signals with range information. The adaption of the DSP software to export encoder data to the LRF rather than parsing the scanner data is currently under development.

The actuated part of the laser pan unit is hooked at three guidance shafts which are located on a common circular path around the LRF's optical centre (see Figure A.3). That way, no mechanical parts impair the scanner's field of vision. The central guidance shaft is connected to the DC motor driving the carriage via a gear belt. The gear belt absorbs shocks and load alternations which frequently occur when negotiating harsh terrain. The cog wheel transmitting the torque from the central guidance shaft to the carriage is flanked by guidance plates which hold the carriage at its place (see Figure A.3b). One set of guidance plates support the inner sides of the carriage cog wheel to provide vertical stability. A second set of guidance plates prevents the carriage from tilting against its socket. These plates furthermore hold the tooth root surfaces of the shaft cog wheel and the carriage cog wheel at a certain distance to each other, such that the cog wheel system cannot get jammed.

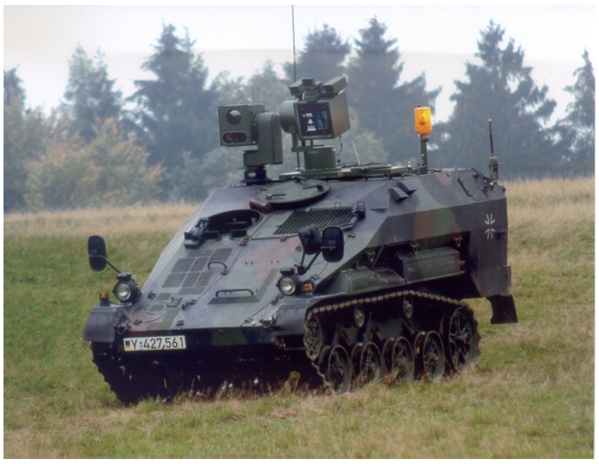
B. Research Activities in Off-road Robotics

This appendix provides a general overview on research in off-road robotics. In that context the various programmes together with the aims and achievements will be highlighted to give the reader a chronological survey on the activities in the field. Note that an exhaustive discussion is way beyond the scope of this work. The programmes, projects, and events have been selected because these were most influential during the development of this Doctoral Thesis.

B.1 The Advent of Autonomous Off-road Robotics

From the very beginning of off-road robotics research in the early nineties of the last century until today many different platforms, locomotion principles and control architectures have been proposed. The *PRIMUS*[1] programme launched by the German federal government yielded first results in 1999. The demonstrator, an airdropable tank of the type "Wiesel 2" (see Figure B.1a), managed to drive up to 30 km/h on low-level roads and open fields. The navigation system was realised using a planner-based deliberative approach. The system was equipped with a 3D-range image camera for obstacle detection. A 2D obstacle map used for path planning was updated at a frequency of 4 Hz. Furthermore, a camera-based contour tracker was used for road following [Schwartz 00]. The focus of the project was autonomous and tele-operated driving in unknown open terrain. Even though the "Wiesel 2" with about four tons counts as a lightweight tank it is physically very robust. Furthermore, the environment was assumed non-cooperative meaning that only obstacles large enough to pose a threat for the vehicle itself would be accounted for. In essence the scenario limits driving situations to wide dirt roads and fields with isolated large obstacles around which a path can be planned a priori. Hence a planner-based deliberative strategy is fully sufficient for the tasks at hand. Advanced local navigation capabilities were not part of the scientific goals as difficult situations could be handled by the operator.

[1]*PR*ogram of *I*ntelligent *M*obile *U*nmanned *S*ystems (PRIMUS)


(a) *PRIMUS* Tank “Wiesel 2” (Photo courtesy of EADS – `http://www.eads.com/`)

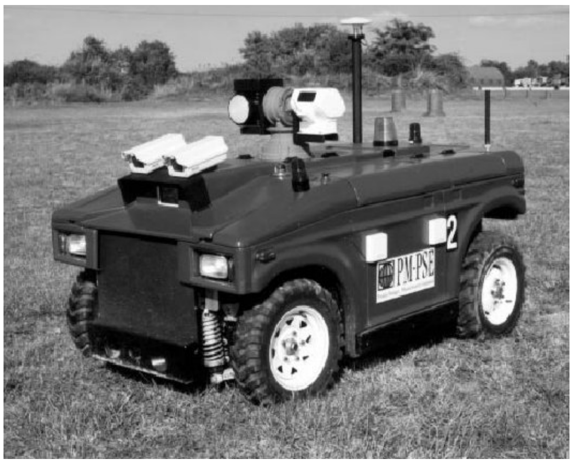

(b) MDARS Exterior Vehicle [Pastore 99]

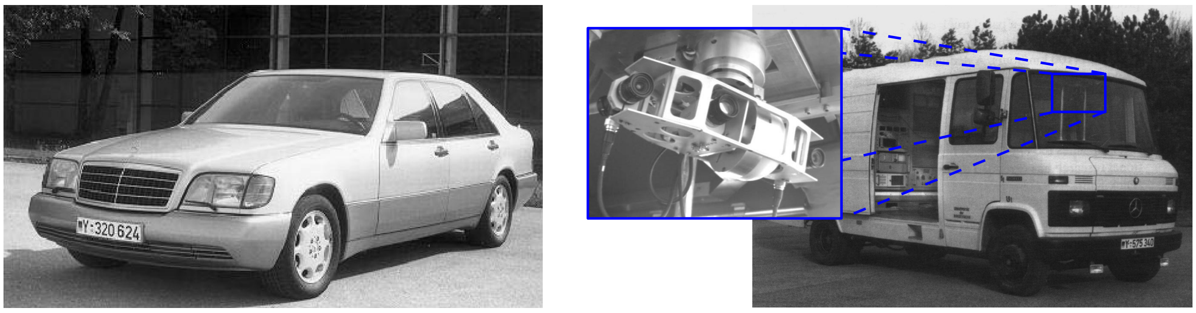
Figure B.1: Right from the beginning off-road robotics research is strictly military.

In parallel to the German efforts the US Department of Defense (DoD) funded several projects going into similar directions as the PRIMUS programme. MDARS-E[2] was part of a larger programme aiming at security applications at military warehouses and storage sites to minimise personal costs [Inderieden 95]. The focus of the MDARS programme was the control of multiple resources like mobile indoor and outdoor robots as well as fixed-place security sensor suites with minimal human supervision. The main research goal was the development of the *Multiple Resource Host Architecture* (MRHA) which allows for the remote access of various security systems from a small number of operator stations. Outdoor navigation represented only a small fragment of the overall programme. Nonetheless, notable achievements have been made within this confined scope.

To the knowledge of the author the MDARS-E demonstrator (see Figure B.1b) represents the first outdoor vehicle following a hybrid navigation approach. The operator provides the robot with routes which are assembled by an off-line planner/dispatcher facility. The robot follows the series of waypoints avoiding obstacles on the way in a reactive manner. In order to augment reliability of the obstacle detection facilities a multi-sensor fusion approach was developed. The data from a radar system, a multi-line laser range finder, several ultrasonic sensors, and a stereo head was registered into a local obstacle map. On the basis of histograms resampled from this map a reactive obstacle avoidance system modified velocity and steering parameters [Pastore 99].

Note that the scenario allows for permanent radio connection between the mobile platforms and the operator station due to the limited extent of storage facilities. Furthermore, the complete environment can be provided to the vehicle in terms of detailed map material. This renders a semi-autonomous system architecture feasible in which the robot calls the operator in case difficult manoeuvres have to be carried out. Therefore, only simple obstacle avoidance strategies together with world-knowledge-based additional features (e. g. keep on the right side of a road) were integrated on the vehicle.

[2]Mobile Detection Assessment and Response System - Exterior (MDARS-E)


(a) VaMP (Prometheus III) [Dickmanns 94]

(b) VaMoRs (AutoNav) [Baten 98, Siedersberger 01]

Figure B.2: Testbeds developed in the context of Prometheus III and AutoNav.

B.2 AutoNav: US-German Off-road Cooperation

Interestingly the efforts of both countries in the domain of local navigation had been complementary so far. The Germans banking on planning mechanisms while the Americans also made use of reactive schemes. In 1995 the US-German cooperation programme *AutoNav* was launched which was dedicated to autonomous driving on low-level roads and open fields. Two years into the programme in 1997 the group of Prof. Dickmanns (UniBW[3] Munich / Germany) joined the *AutoNav* project. The group was already well-known for remarkable results in autonomous navigation on roads which were achieved in the Eureka-funded Prometheus project cycle (see Figure B.2a). One goal of *AutoNav* was to integrate the off-road suited stereo-based obstacle detection from MDARS-E with the deliberative 4D perception and control architecture EMS-Vision[4] developed in Prometheus III [Baten 98, Siedersberger 01].

The UniBW VaMoRs[5] vehicle which was equipped with the multi-camera stereo system `MarVEye`[6] during the Prometheus programme was now supplemented with real-time image processing hardware of the VFE[7] series by Sarnoff Corporation. The VFE computed dense 3D point clouds from which obstacles were extracted and passed to the 4D perception and control system in terms of object bounding boxes. To yield more robust obstacle information objects were corroborated over time and reported only if detected in several subsequent frames. The control system is organised in a knowledge representation which holds information about both the vehicle state and the environment. The core structure is the object-based scene tree in which obstacle data is maintained. A state-based situation assessment system judges the current driving conditions and initiates evasive or stopping manoeuvres if applicable. For that purpose relevant obstacles residing in or nearby the vehicle's driving tube are assigned fuzzy values which are used for symbolic reasoning.

Though driving scenarios were still limited to wide roads and open fields with isolated and distinct obstacles the proposed 4D perception and control approach certainly yields many

[3]UniBW → Universität der Bundeswehr – Federal Armed Forces University

[4]EMS-Vision → Expectation-based, Multi-focal, Saccadic vision

[5]VaMoRs → Mercedes Benz 508D van equipped with a stereo system, dedicated image processing hardware and four PCs.

[6]MarVEye → Multi-focal active / reactive Vehicle Eye [Dickmanns 03]

[7]VFE → Vision Front End [Mandelbaum 98] – VFE 100 and later VFE 200 were deployed

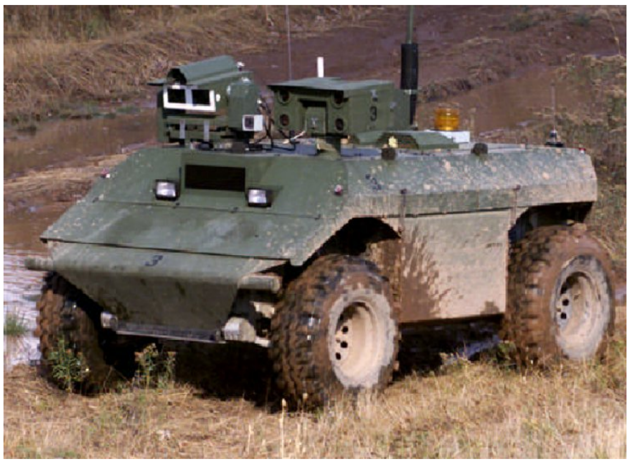


Figure B.3: Demo III Experimental Unmanned Vehicles (XUV) [Lacaze 02]

useful ideas for robust off-road navigation. In tight terrain featuring extensive vegetation the approximation of obstacles with bounding boxes as well as the representation in an object-based scene tree might not be precise enough for effective manoeuvring. Sensor noise and fragmented obstacle readings will probably result in structures which are difficult to group into distinct entities which could be maintained in a sparse representation.

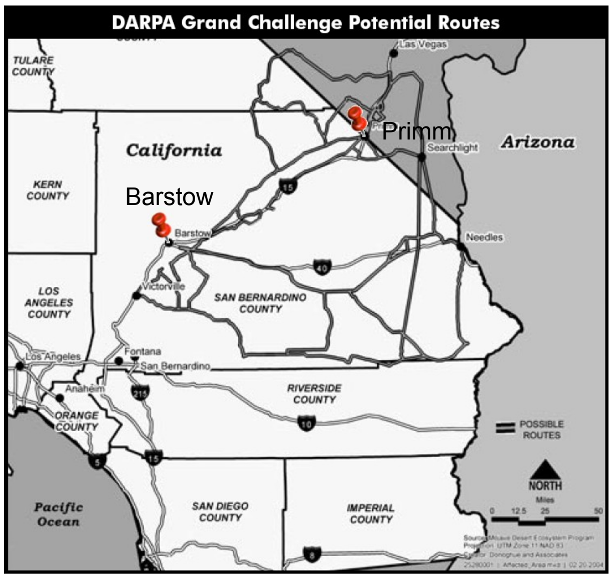
B.3 Demo III Targets Vegetated Terrain

Since the late 1980s the Joint Robotics Program (JPR)[8] was responsible for the consolidation of all robotic-related acquisition programmes of the US DoD. After demonstrations of short-term realisable tele-operated unmanned systems resulting from the Demo I programme a sequel project was jointly funded by glsjpr and DARPA[9] from 1991 until 1996. The aim of Demo II was the development of autonomous on- and off-road navigation capabilities with a strong focus on the enhancement of supervisory control techniques (compare MDARS programme). Operator workload per vehicle should be kept as minimal as possible, yet manual intervention as last resort lead to the design of a semi-automated system architecture. Problems and shortcomings were identified in concluding field exercises carried out with regular troops. As this was the first time the end user was in charge of the final evaluation process the success of the experiment has to be seen as a milestone in military robotics. Encouraged by the achievements of the Demo II evaluation the third extension of the programme was designed towards tighter cooperation between developers and future users. For a more detailed overview over the Demo programme series as well as interconnections and overlaps with other programmes see [Shoemaker 98].

Motivated by the experiences from its predecessor project and results yielded in other programmes funded by the Department of Defence the Demo III project had very ambitious objectives. In a time frame of only four years four demonstrators should be built up which would be capable of day and night navigation under tactical conditions at speeds between 16 and 32 km/h. Furthermore, cluttered and vegetated environments were explicitly added

[8]The JPR was founded in 1989 by the Congress of the United States of America

[9]Defense Advanced Research Projects Agency (DARPA) – `http://www.darpa.mil/`



(a) DARPA GC 2004 track through Mojave desert [DARPA 04] (Photo courtesy of DARPA)

(b) Sandstorm 2004 from CMU [Whittaker 04]

Figure B.4: DARPA GC 2004: From Los Angeles to Las Vegas.

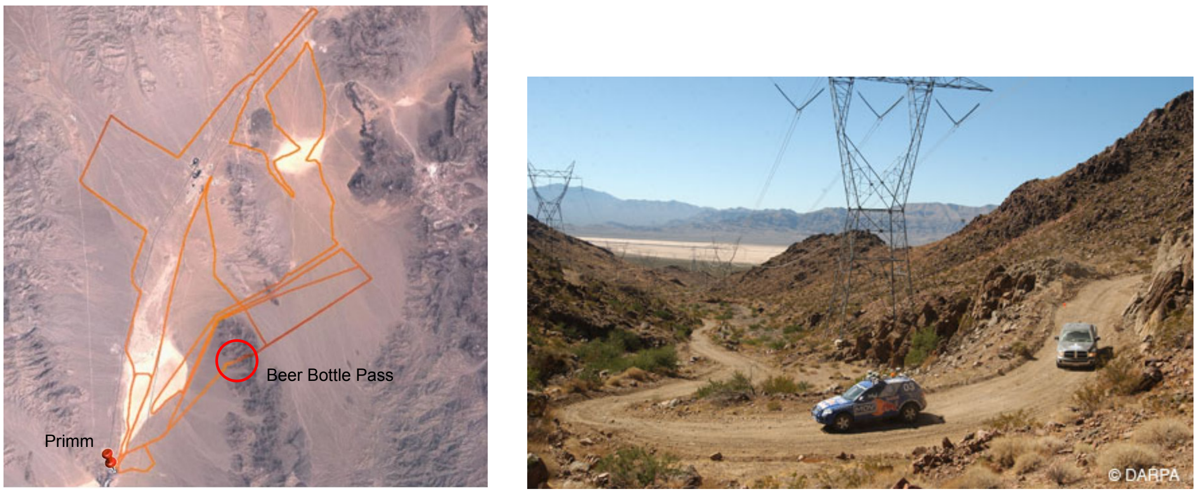
to the objectives of the project. In order to manage the complexity of the overall system the VaMoRs 4D approach ported towards off-road navigation during the *AutoNav* project was combined with the deliberative NIST[10] reference control architecture RCS[11]. The control system modularisation approach of RCS had already been approved in space robotics and automation technology. The resulting reference model for the Demo III vehicles (see Figure B.3) was named 4D/RCS [Albus 98] and became a NIST recommendation for autonomous ground vehicles in 2002 [Albus 02].

B.4 The DARPA Grand Challenge: Fast Navigation in Unrehearsed Terrain

In recent years off-road robotics has made great advances and conquered more and more difficult terrain. Sophisticated data structures, algorithms, and architectures have been evolved to manage the increasing complexity of mobile systems operating in uncertain environments. So far the development was limited to supervised autonomous systems which would call the operator in case difficulties were encountered. Despite all the success many problems apparently remain unsolved with the systems developed so far. With the aim to accelerate the development of unmanned ground vehicles which would satisfy military requirements DARPA announced a robotic competition in July 2002: the DARPA Grand Challenge. The focus of the Grand Challenge was long-term robustness and navigation at high speeds.

[10]National Institute of Standards and Technology (NIST) – `http://www.nist.gov/`

[11]Real-time Control System (RCS)



(a) DARPA GC 2005 track **(b)** Stanley from Stanford University traversing the Beer Bottle Pass [Thrun 06]

Figure B.5: DARPA GC 2005: The round course through Mojave desert (a) was completed first by the robot Stanley from Stanford University (b) (Photos courtesy of DARPA).

From Los Angeles to Las Vegas in 10 hours: GC 2004

The first competition took place in 2004 and was carried out on a 241 km long desert track between Los Angeles and Las Vegas (see Figure B.4a). During the race the course was cleared of all interfering traffic and lead parallel to the Interstate 15 from Barstow (California) to Primm (Nevada). The teams were provided with a data file (RDDF[12]) containing GPS waypoints as well as a corridor width and a speed limit for each path segment. In total the course was defined by 2586 waypoints (i.e. distance between waypoints about 100 m) and the minimum corridor width was about 3.5 m. Speed limits ranged from 8 km/h to 80 km/h. The vehicles were required to follow the waypoints without manual intervention and complete the course in a time window of only 10 hours. The winning vehicle's team would be awarded $1 million price money. More than 100 teams applied for participation and after the qualifying fifteen vehicles started in the final race. In the end none of the finalists accomplished more than 5 % of the distance. Carnegie Mellon University's (CMU) *Sandstorm* travelled furthest clearing 11.78 km.

Mojave Desert Round Trip: GC 2005

The disillusion of the 2004 Grand Challenge results did not last very long. In October 2005 the race was repeated with the prize money doubled. A new 212 km track was planned through Mojave desert starting and ending in Primm (Nevada). As the year before the circuit was specified in the RDDF format. Two hours before the race the teams were provided with the data files for mission planning. The waypoint density was increased such that the average distance between two waypoints was about 75 m. In curves and at tighter spots, waypoints were placed only a few metres apart. In total 2935 waypoints were used to specify the complete course. Almost 200 teams registered

[12]Route Description Data File (RDDF)

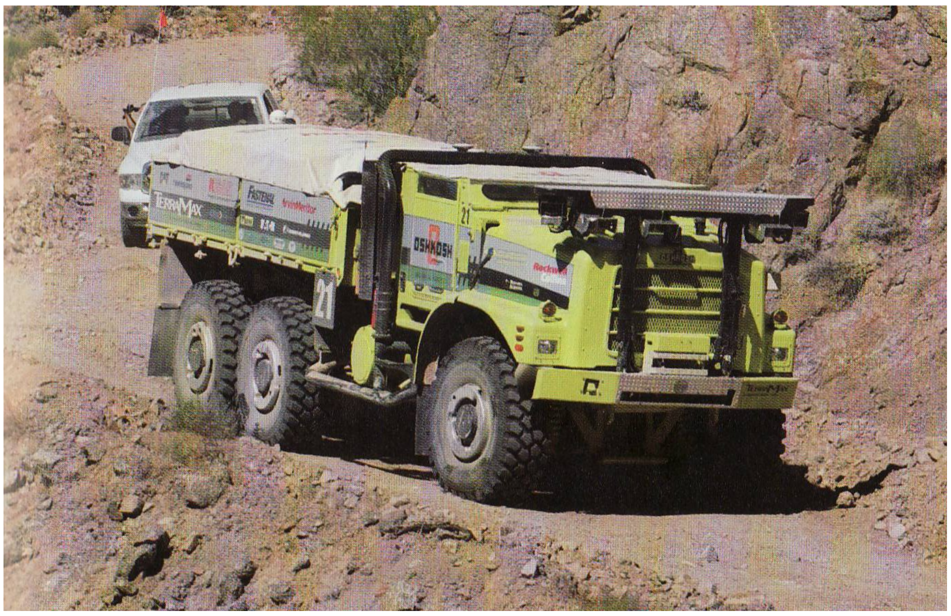


Figure B.6: TerraMax™was the largest vehicle participating in the 2005 DARPA GC. (Photo courtesy of DARPA [Buehler 07])

for the competition and in the end 23 teams started on the racing days. The robot Stanley from Stanford University needed 6:53 h and was the first vehicle to complete the circuit. Figure B.5b shows Stanley traversing the Bottle Beer Pass which was feared by all participants as the most difficult passage on the track. The ten hours time limit was undercut by three further robots: CMU's Sandstorm and H1ghlander as well as Kat-5 by Gray Insurance Company. The 16-ton-truck TerraMax™ (see Figure B.6) by Oshkosh Truck Company[13] completed the track on the next day and accomplished the course in 12:51 h. This is remarkable even though the DARPA time limit was not met as for this huge vehicle (LWH: 8.02 m $\times$ 2.48 m $\times$ 3.55 m [OTC 08]) about 3 m minimal corridor width is definitely tight. For both Grand Challenges it can be stated that the high waypoint density and the additional corridor information made global path planning virtually irrelevant. Some teams, e. g. Stanford Racing, computed smother paths before the race. Mostly in order to keep away from the more cluttered track boundaries and to mitigate sharp turns. The given waypoints were located on dirt roads and mountain tracks which were well-traversable with an off-the-shelf all-terrain vehicle and wide enough even for large trucks to pass (remember TerraMax™). Furthermore, obstacles were located rather at the road boundaries and were clearly distinguishable from the ground. Vegetation or water hazards were not part of the obstacle repertoire. On the other hand the tight time limit required the vehicles to achieve speeds that make certain assumptions about the terrain structure inevitable. The main problems to be solved for the DARPA Grand Challenge were automated guidance, track keeping and obstacle avoidance at high speeds.

[13]Oshkosh Truck Company – `http://www.oshkoshcorporation.com/`
More information on TerraMax™: `http://www.terramax.com/`

B.5 European Land Robot Trials: Realistic Scenarios for Off-road Mobility

Since 2006 European action forces formulate the demands for robotic support in the scenarios of the ELROB[14]. In contrast to the DARPA Grand Challenge competitions the ELROB is a trial where no explicit ranking is published by the organisers. Furthermore, the scenarios are open to any robotic solution ranging from tele-operated approaches to fully autonomous solutions. As the focus of ELROB is on short-term realisable robotics the scenarios are very close to real-life applications with demands which tend to attract semi-autonomous solutions. The aim of the event is to bring together companies, research groups and potential customers in order to discuss needs, problems and solutions. In even years the European armed forces specify the trial scenarios in the uneven years civilian applications are in the focus.

Over the years four core disciplines have emerged which constitute the frame for the scenarios which are slightly changed every year. The core disciplines shall briefly be introduced in the rest of this section.

Reconnaissance and Surveillance

In this scenario the robot has to explore an unknown territory and identify objects of interest. The objects of interest have to be documented with images of reasonable quality and tagged with GPS position data. The accuracy of the localisation information may only vary a few metres from the actual position of the object. Further credits can be earned if the report to the operator features a terrain map which indicates the locations of interest in a graphical form. The terrain may vary from hilly grassland over steep rocky slopes to intensely vegetated areas. Metre-deep waterholes and muddy ditches have to be anticipated as well. So far the nasty environmental conditions have put forth very practical approaches which bank to a certain degree on tele-operation.

Transport – Mule

The transport scenario targets the capability of the vehicles to carry out repetitive tasks. The robots have to shuttle between two camps as often as possible in a given time limit while carrying a payload of up to 50 kg. From the application point of view functionality for logistics shall be presented. As for *Reconnaissance and Surveillance* the route consists of mixed off-road terrain which may feature a wide variety of obstacles. In contrast to the scenario above a traversable track which is visible in freely available Geographic Information System (GIS) data can be assumed.

Camp Security

In this scenario, a camp has to be protected. One or more robots are supposed to patrol and detect intruders. People entering the camp have to be asked for authorisation. In case the authorisation is refused or the person act uncooperative for instance by running away, an alarm shall be triggered in the control centre.

[14]European Land Robot Trial (ELROB) – `http://www.elrob.eu/`

Autonomous Navigation

The autonomous navigation scenario attempts to test the usability of the robotic system in unknown environments. In particular the capability to reach a destination area with limited information under strict time constraints are the focus of this discipline. The robot and the operator team are taxied to an unknown starting point, where the operators are provided with a map indicating the target location. The robot has to reach the target location in a given time limit with only the means available on site. The operator station will not have a line of sight to the destination area and radio uplink may not be available at all times. As for *Transport Mule* a traversable off-road track can be obtained from freely available GIS data. A mission plan – in terms of waypoints or further map material – is not provided a priori. All preparations necessary have to be carried out on site and are regarded as part of the mission. Note that the strict time limit does not allow for elaborate mission planning. In 2008 the time limit was 45 minutes for ranges between 0.5 to 3.0 km air-line distance.

Bibliography

[Abouaf 98] J. Abouaf, "Trial by Fire: Teleoperated Robot Targets Chernobyl", *IEEE Computer Graphics and Applications*, vol. 18, pp. 10–14, 1998.

[Agre 87] P. E. Agre, D. Chapman, "Pengi: An Implementation of a Theory of Activity", in *National Conference on Artificial Intelligence (AAAI)*. 1987, pp. 268–272.

[Alami 98] R. Alami, R. Chatila, S. Fleury, M. Ghallab, F. Ingr, "An Architecture for Autonomy", *International Journal of Robotics Research*, vol. 17, pp. 315–337, 1998.

[Albus 02] J. S. Albus, "4D/RCS: A Reference Model Architecture for Intelligent Unmanned Ground Vehicles", in *Proceedings of the SPIE 16th Annual International Symposium on Aerospace/Defense Sensing, Simulation and Controls*. Orlando, FL, USA, April 1–5 2002.

[Albus 87] J. Albus, "NASA/NBS Standard Reference Model for Telerobot Control System Architecture (NASREM)", NIST, Tech. Rep., 1987.

[Albus 93] J. S. Albus, "A reference model architecture for intelligent systems design", in *An Introduction to Intelligent and Autonomous Control*. Kluwer Academic Publishers, 1993, pp. 27–56.

[Albus 95] J. S. Albus, "The NIST Real-time Control System (RCS) An Applications Survey", in *AAAI Spring Symposium*. 1995.

[Albus 98] J. S. Albus, "4-D/RCS: a reference model architecture for Demo III", in *International Symposium on Intelligent Control (ISIC)*. Gaithersburg, USA, September 1998, pp. 634–639.

[Alon 06] Y. Alon, A. Ferencz, A. Shashua, "Off-road Path Following using Region Classification and Geometric Projection Constraints", in *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 1. New York, NY, USA: IEEE Computer Society (Washington, DC, USA), June 17–22 2006, pp. 689–696.

[Arkin 97] R. C. Arkin, T. Balch, "AuRA: Principles and Practice in Review", *Journal of Experimental and Theoretical Artificial Intelligence*, vol. 9, pp. 175–189, 1997.

[Arkin 98] R. Arkin, *Behaviour-Based Robotics*, MIT Press, 1998.

[Armbrust 07] C. Armbrust, “Mobile Robot Navigation Using Dynamic Maps and a Behaviour-Based Anti-Collision System”, Diploma thesis, Robotics Research Lab, Department of Computer Sciences, University of Kaiserslautern, September 30 2007. unpublished.

[Armbrust 09a] C. Armbrust, T. Braun, T. Föhst, M. Proetzsch, A. Renner, **B.-H. Schäfer**, K. Berns, “RAVON — The Robust Autonomous Vehicle for Off-road Navigation”, in *Proceedings of the IARP International Workshop on Robotics for Risky Interventions and Environmental Surveillance 2009 (RISE 2009)*, IARP. Brussels, Belgium, January 12–14 2009.

[Armbrust 09b] C. Armbrust, **B.-H. Schäfer**, K. Berns, “Using Passages to Support Off-road Robot Navigation”, in *Proceedings of the 6th International Conference on Informatics in Control, Automation and Robotics 2009 (ICINCO 2009)*, Institute for Systems and Technologies of Information, Control and Communication (INSTICC). Milan, Italy, July 2–5 2009, pp. 189–194.

[Armbrust 10a] C. Armbrust, T. Braun, T. Föhst, M. Proetzsch, A. Renner, **B.-H. Schäfer**, K. Berns, “RAVON – The Robust Autonomous Vehicle for Off-road Navigation”, in *Using robots in hazardous environments: Landmine detection, de-mining and other applications*, Y. Baudoin, M. K. Habib, Eds., Woodhead Publishing Limited, 2010, ch. RAVON – The Robust Autonomous Vehicle for Off-road Navigation. ISBN: 1 84569 786 3; ISBN-13: 978 1 84569 786 0.

[Armbrust 10b] C. Armbrust, S. A. Mehdi, M. Reichardt, J. Koch, K. Berns, “Using an Autonomous Robot to Maintain Privacy in Assistive Environments”, *Security and Communications Networks: Special Issue on Privacy and Security in Pervasive e-Health and Assistive Environments*, 2010. to be published.

[Armbrust 10c] C. Armbrust, M. Proetzsch, **B.-H. Schäfer**, K. Berns, “A Behaviour-based Integration of Fully Autonomous, Semi-autonomous and Tele-operated Control Modes for an Off-road Robot”, in *Proceedings of the 2nd IFAC Symposium on Telematics Applications*, IFAC. Politehnica University, Timisoara, Romania, October 5–8 2010. invited paper.

[Baddeley 00] A. D. Baddeley, “The episodic buffer: a new component of working memory?”, *Trends in Cognitive Science*, vol. 4, pp. 417–423, 2000.

[Baddeley 09] A. Baddeley, M. W. Eyenck, M. C. Anderson, *Memory*, New York, USA: Psychology Press, 2009.

[Baddeley 74] A. D. Baddeley, G. Hitch, *The psychology of learning and motivation: Advances in research and theory*, New York: Academic Press, 1974, vol. 8, ch. Working memory, pp. 47–89.

[Baddeley 96] A. D. Baddeley, S. D. Sala, “Working memory and executive control”, *Philosophical Transactions of the Royal Society of London*, vol. 351, pp. 1397–1404, 1996.

[Barbera 84] A. J. Barbera, J. S. Albus, M. L. Fitzgerald, L. S. Haynes, “RCS: The NBS real-time control system”, in *Robots 8 Conference and Exposition*. Detroit, USA, June 4–7 1984.

[Baten 98] S. Baten, M. Luetzeler, E. D. Dickmanns, R. Mandelbaum, P. J. Burt, “Techniques for Autonomous, Off-Road Navigation”, *IEEE Intelligent Systems*, vol. 13, no. 6, pp. 57–65, 1998.

[Bergel 06] A. Bergel, S. Ducasse, O. Nierstrasz, R. Wuyts, “Stateful Traits”, in *International Smalltalk Conference (ISC 2006)*, ser. LNCS, vol. 4406. Springer, 2006, pp. 66–90.

[Bergh 00] C. Bergh, B. Kennedy, L. Matthies, A. Johnson, “A compact and low power two-axis scanning laser rangefinder for mobile robots”, in *Seventh Mechatronics Forum International Conference*. 2000.

[Blake 07] E. S. Blake, E. N. Rappaport, C. W. Landsea, “The Deadliest, Costliest, and most Intense United States Tropical Cyclones from 1851 to 2006 (and other Frequently Requested Hurricane Facts)”, National Hurricane Center (USA) - http://www.nhc.noaa.gov/pdf/NWS-TPC-5.pdf, 2007.

[Bonarini 03] A. Bonarini, G. Invernizzi, T. H. Labella, M. Matteucci, “An architecture to coordinate fuzzy behaviors to control an autonomous robot”, *Fuzzy Sets and Systems*, vol. 134, no. 1, pp. 101–115, 2003.

[Borenstein 91] J. Borenstein, Y. Koren, “The Vector Field Histogram - Fast Obstacle Avoidance for Mobile Robots”, *IEEE Transactions on Robotics and Automation*, vol. 7, no. 3, pp. 278–288, 1991.

[Braid 06] D. Braid, A. Broggi, G. Schmiedel, “The TerraMax autonomous vehicle: Field Reports”, *J. Robot. Syst.*, vol. 23, no. 9, pp. 693–708, 2006.

[Braun 07] T. Braun, J. Wettach, K. Berns, “A Customizable, Multi-Host Simulation and Visualization Framework for Robot Applications”, in *13th International Conference on Advanced Robotics (ICAR07)*. Jeju, Korea, August 21–24 2007, pp. 1105–1110.

[Braun 09a] T. Braun, *Cost-Efficient Global Robot Navigation in Rugged Off-Road Terrain*, ser. RRLab Dissertations, Verlag Dr. Hut, 2009. ISBN 978-3-86853-135-0.

[Braun 09b] T. Braun, **B.-H. Schäfer**, K. Berns, “Topological Large-Scale Off-road Navigation and Exploration - RAVON at the European Land Robot Trial 2008”, in *IROS’09: Proceedings of the 2009 IEEE/RSJ international conference on Intelligent robots and systems*. St. Louis, MO, USA: IEEE Press, October 11–15 2009, pp. 4387–4392.

[Brenneke 03] C. Brenneke, O. Wulf, B. Wagner, “Using 3D Laser Range Data for SLAM in Outdoor Environments”, in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Las Vegas, Nevada, USA, Oktober 27–31 2003, pp. 188–193.

[Brooks 86] R. A. Brooks, “A Robust Layered Control System for a Mobile Robot”, *IEEE Journal of Robotics and Automation*, vol. RA-2, no. 1, pp. 14–23, April 1986.

[Buehler 07] M. Buehler, K. Iagnemma, S. Singh, *The 2005 DARPA Grand Challenge: The Great Robot Race*, 1st ed., Springer Publishing Company, Incorporated, 2007.

[Cardelli 85] L. Cardelli, P. Wegner, "On understanding types, data abstraction, and polymorphism", *ACM Comput. Surv.*, vol. 17, no. 4, pp. 471–523, December 1985.

[Castano 03] A. Castano, L. Matthies, "Foliage Discrimination using a Rotating Ladar", in *IEEE International Conference on Robotics and Automation*. 2003.

[CFEG 06] CFEG. *Environmental Consequences of the Chernobyl Accident and their Remediation: Twenty Years of Experience*. Chernobyl Forum Expert Group Environment. 2006.

[Chatila 85] R. Chatila, J. Laumond, "Position referencing and consistent world modeling for mobile robots", in *IEEE International Conference on Robotics and Automation (ICRA)*. 1985, pp. 138–145.

[Chen 06] Q. Chen, "Intelligent off-road navigation algorithms and strategies of Team Desert Buckeyes in the DARPA Grand Challenge 2005: Field Reports", *J. Robot. Syst.*, vol. 23, no. 9, pp. 729–743, 2006.

[Connell 92] J. H. Connell, "SSS: a hybrid architecture applied to robot navigation", in *IEEE International Conference on Robotics and Automation. Proceedings.*, vol. 3. May 1992, pp. 2719–2724.

[Cooper 05] K. Cooper, S. Liddle, S. Dascalu, "Experiences Using Defect Checklists in Software Engineering Education", in *CAINE*. 2005, pp. 402–409.

[Corbetta 91] M. Corbetta, F. Miezin, S. Dobmeyer, G. Shulman, S. Petersen, "Selective and Divided Attention during Visual Discriminations of Shape, Color, and Speed: Functional Anatomy by Positron Emission Tomography", *The Journal of Neuroscience*, 1991.

[DARPA 04] DARPA, "The DARPA Grand Challenge Commemorative Program", http://www.darpa.mil/grandchallenge04/, March 8–13 2004.

[Dep 04] Department of Defense - United States of America, http://www.stormingmedia.us/86/8610/A861044.html. *2004 Army Transformation Road Map*. July 2004.

[Dickmanns 03] E. D. Dickmanns, "An advanced vision system for", in *1st International Workshop on In-Vehicle Cognitive Computer Vision Systems (IVCCVS)*. 2003.

[Dickmanns 94] E. Dickmanns, "The seeing passenger car VaMoRs-P", in *International Symposium on Intelligent Vehicles*. 1994, pp. 24–26.

[Dijkstra 59] E. W. Dijkstra, "A Note on Two Problems in Connexion with Graphs", *Numerische Mathematik*, vol. 1, pp. 269–271, 1959.

[Dijkstra 82] E. W. Dijkstra, "On the role of scientific thought", *Selected Writings on Computing: A Personal Perspective*, pp. 60–66, 1982.

[Dirac 42] P. Dirac, “The Physical Interpretation of Quantum Mechanics”, in *Royal Society London*, vol. A 180. 1942, p. 1–39.

[Dusha-Gudym 05] S. I. Dusha-Gudym, “Transport of Radioactive Materials by Wildland fires in the Chernobyl Accident Zone: How to Address the Problem”, *International Forest Fire News (IFFN)*, no. 32, pp. 119–125, January–June 2005.

[Elfes 87] A. Elfes, “Sonar-Based Real-World Mapping and Navigation”, in *IEEE Journal of Robotics and Automation*, vol. RA-3, no. 3. 1987, pp. pp. 249–265.

[Emanuel 05] K. Emanuel, “Increasing destructiveness of tropical cyclones over the past 30 years”, Nature Vol. 436 - Letters, 2005.

[Ester 96] M. Ester, H.-P. Kriegel, J. Sander, X. Xu, “A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise”, in *International Conference on Knowledge Discovery and Data Mining (KDD)*. 1996.

[Firby 89] R. J. Firby, “Adaptive Execution in Complex Dynamic Worlds”, Dissertation, Graduate School of Yale University, New Haven, CT, May 1989. Technical Report YALEU/CSD/RR #672.

[Fitschen 07] P. Fitschen, *Die Transformation der US-Streitkräfte*, 1 ed., Peter Lang Verlag Frankfurt, Juli 2007.

[Flatt 98] M. Flatt, S. Krishnamurthi, M. Felleisen, “Classes and mixins”, in *POPL 98́: Proceedings of the 25th ACM SIGPLAN-SIGACT symposium on Principles of programming languages*. New York, NY, USA: ACM, 1998, pp. 171–183.

[Fleischmann 10] P. Fleischmann, “Integration von Luftbildern in RAVONs Navigationssystem zur Lokalisierung und weitsichtigen Planung”, Diploma thesis, Robotics Research Lab, Department of Computer Sciences, University of Kaiserslautern, June 2010. unpublished.

[Fong 03] E. Fong, W. Adams, F. Crabbe, A. Schultz, “Representing a 3-D Environment with a 2 1/2-D Map Structure”, in *IEEE International Conference on Intelligent Robotics and Systems (IROS)*. 2003.

[Garcia 03] R. Garcia, J. Jarvi, A. Lumsdaine, J. G. Siek, J. Willcock, “A comparative study of language support for generic programming”, in *OOPSLA 03: Proceedings of the 18th annual ACM SIGPLAN conference on Object-oriented programing, systems, languages, and applications*. New York, NY, USA: ACM, 2003, pp. 115–134.

[Garcia 07] R. Garcia, J. Jarvi, A. Lumsdaine, J. Siek, J. Willock, “An extended comparative study of language support for generic programming”, *Journal of Functional Programming*, vol. 17, no. 02, pp. 145–205, 2007.

[Gat 92] E. Gat, “Integrating Planning and Reacting in a Heterogeneous Asynchronous Architecture for Controlling Mobile Robots”, in *Proceedings of the Tenth National Conference on Artificial Intelligence (AAAI)*. 1992.

[Gat 98] E. Gat, “Three-Layer Architectures”, in *Artificial Intelligence and Mobile Robots*, D. Kortenkamp, R. P. Bonasso, R. Murphy, Eds., AAAI Press / The MIT Press, 1998, pp. 195–210. ISBN: 0-262-61137-6; title of the draft: On Three-layer Architectures.

[Gerthsen 89] C. Gerthsen, H. Kneser, H. Vogel, *Physik*, 16 ed., Springer, 1989.

[Green 95] R. O. Green, J. Dozier, “Measurement of the spectral absorption of liquid water in melting snow with an imaging spectrometer”, in *Summaries of the Fifth Annual JPL Airborne Earth Science Workshop*, J. Publication, Ed. January 23–26 1995, pp. 91–94.

[Greenpeace 06] Greenpeace. *Nuclear Power and Terrorism.* Greenpeace UK, Canonbury Villas, London, N1 2PN. January 2006.

[Haas 06] E. Haas, E. Bartholomé, B. Combal, “A map of temporary water bodies in Western Africa”, in *GlobWetland Symposium.* ESA, Frascati, Italy, October 19–20 2006.

[Hart 68] P. E. Hart, N. J. Nilsson, B. Raphael, “A Formal Basis for the Heuristic Determination of Minimum Cost Paths”, in *IEEE Transactions of Systems Science and Cybernetics*, vol. 4, no. 2. July 2 1968, pp. 100–107.

[Hebert 03] M. Hebert, N. Vandapel, “Terrain Classification Techniques from LADAR Data for Autonomous Navigation”, in *Collaborative Technology Alliances conference.* May 2003.

[Hillenbrand 07] C. Hillenbrand, K. Berns, “Modulare Sicherheits- und Sensorsysteme für autonome mobile Roboter realisiert im Forschungsfahrzeug Marvin”, in *Autonome Mobile Systeme 2007*, K. Berns, T. Luksch, Eds. Springer Verlag Berlin, 2007, pp. 133–138.

[Hillenbrand 09] C. Hillenbrand, “Sicheres Klettern eines radgetriebenen Roboters mit Unterdruckkammern an porösen Flächen”, Dissertation, Technische Universität Kaiserslautern, 2009.

[Hillenbrand 10a] C. Hillenbrand, *Sicheres Klettern eines radgetriebenen Roboters mit Unterdruckkammern an porösen Flächen*, ser. RRLab Dissertations, Verlag Dr. Hut, 2010. ISBN 978-3-86853-352-1.

[Hillenbrand 10b] C. Hillenbrand, D. Schmidt, N. Bennett, P. Bach, K. Berns, C. Schindler, “Feasibility Study for the Automation of Commercial Vehicles on the Example of a Mobile Excavator”, in *Commercial Vehicle Technology 2010 - Proceedings of the 1st Commercial Vehicle Technology Symposium (CVT 2010)*, K. Berns, C. Schindler, K. Dreßler, B. Jörg, R. Kalmar, J. Hirth, Eds. Kaiserslautern, Germany: Shaker Verlag, March 16–18 2010, pp. 22–31. ISBN 978-3-8322-9040-5.

[Hirth 10] J. Hirth, K. Berns, K. Mianowski, “Designing Arms and Hands for the Humanoid Robot ROMAN”, in *Proceedings of the IEEE International Conference on Mechanical Engineering, Robotics and Aerospace (ICMERA).* Bucharest, Romania, December 2–4 2010, pp. 63–67.

[Hong 00] T. H. Hong, M. Abrams, T. Chang, M. Shneier, “An Intelligent World Model for Autonomous Off-Road Driving”, *Computer Vision and Image Understanding*, 2000.

[Hong 02] T. H. Hong, C. Rasmussen, T. Chang, M. Shneier, "Fusing Ladar and Color Image Information for Mobile Robot Feature Detection and Tracking", in *7th International Conference on Intelligent Autonomous Systems*, M. Gini, W. M. Shen, C. Torras, H. Yuasa, Eds. IOS Press, March 2002, pp. 124–133.

[IAEA 02] IAEA. *Calculating the new global nuclear terrorism threat.* International Atomic Energy Agency. The Science of the Total Environment 284 pp. 269-272 ed., 2002.

[IAEA 07] IAEA. *Combating Illicit Trafficking in Nuclear and other Radioactive Material.* International Atomic Energy Agency, Vienna. 2007.

[Inderieden 95] R. S. Inderieden, H. R. Everett, T. A. Heath-Pastore, R. P. Smurlo, "Overview of the Mobile Detection Assessment and Response System", in *DND/CSA Robotics and KBS Workshop*. October 1995.

[Jäger 95] H. Jäger, "The Dual Dynamics Design Scheme for Behavior-based Robots: A Tutorial", GMD, St. Augustin, Tutorial, Decemver 19 1995.

[Jansen 05] P. Jansen, W. van der Mark, J. van den Heuvel, F. Groen, "Colour based off-road environment and terrain type classification", *Intelligent Transportation Systems, 2005. Proceedings. 2005 IEEE*, pp. 61–66, 2005.

[Joyeux 10] S. Joyeux, F. Kirchner, S. Lacroix, "Managing plans: Integrating deliberation and reactive execution schemes", *Robotics and Autonomous Systems*, vol. 58, no. 9, pp. 1057 – 1066, 2010. Hybrid Control for Autonomous Systems.

[Koch 05] J. Koch, C. Hillenbrand, K. Berns, "Inertial Navigation for Wheeled Robots in Outdoor Terrain", in *5th IEEE Workshop on Robot Motion and Control (RoMoCo)*. Dymaczewo, Poland, June 23–25 2005, pp. 169–174.

[Konolige 97] K. Konolige, A. Saffiotti, "The Saphira Architecture: A Design for Autonomy", in *Journal of Experimental and Theoretical Artificial Intelligence (JETAI) 9.* 1997, pp. 215–235.

[Lacaze 02] A. Lacaze, K. Murphy, M. DelGiorno, "Autonomous Mobility for the DEMO III Experimental Unmanned Vehicles", 2002.

[Lalonde 06] J.-F. Lalonde, N. Vandapel, D. Huber, M. Hebert, "Natural terrain classification using three-dimensional ladar data for ground robot mobility", *Journal of Field Robotics*, vol. 23, no. 10, pp. 839 – 861, November 2006.

[Lamon 06] P. Lamon, S. Kolski, R. Siegwart, "The SmartTer - a Vehicle for Fully Autonomous Navigation and Mapping in Outdoor Environments", in *Proceedings of the 9th International Conference on Climbing and Walking Robots*. September 2006.

[Langer 94] D. Langer, J. Rosenblatt, M. Hebert, "A Behavior-Based System for Off-Road Navigation", in *IEEE Journal of Robotics and Automation.* 1994.

[Leedy 06a] B. M. Leedy, "Two Minds for one Vehicle: A Case Study in Deliberative and Reactive Navigation", Diploma thesis, Virginia Polytechnic Institute and State University, http://nsdl.org/, March 29 2006.

[Leedy 06b] B. M. Leedy, J. S. Putney, C. Bauman, S. Cacciola, J. M. Webster, C. F. Reinholtz, "Virginia Tech´s twin contenders: A comparative study of reactive and deliberative navigation: Field Reports", *Journal of Field Robotics*, vol. 23, no. 9, pp. 709–727, 2006.

[Lieb 05] D. Lieb, A. Lookingbill, S. Thrun, "Adaptive Road Following using Self-Supervised Learning and Reverse Optical Flow", in *Robotics: Science and Systems.* Cambridge, Massachusetts, USA, June 8–10 2005, pp. 273–280.

[Macedo 01] J. Macedo, R. Manduchi, L. Matthies, "Ladar-Based Discrimination of Grass from Obstacles for Autonomous Navigation", in *ISER 0́0: Experimental Robotics VII.* London, UK: Springer-Verlag, 2001, pp. 111–120.

[Mandelbaum 98] R. Mandelbaum, M. Hansen, P. Burtt, S. Bated, "Vision for Autonomous Mobility: Image Processing on the VFE-200", ISIC/CIRNISAS Joint Conference, Gaithersburg / USA, September 1998.

[Manduchi 04] R. Manduchi, A. Castano, A. Talukder, L. Matthies, "Obstacle Detection and Terrain Classification for Autonomous Off-road Navigation", in *Autonomous Robots.* 2004.

[Matarić 97] M. J. Matarić, "Behavior-Based Control: Examples from Navigation, Learning, and Group Behavior", *Journal of Experimental and Theoretical Artificial Intelligence - Special Issue on Software Architectures for Physical Agents*, vol. 9, no. 2–3, pp. 323–336, 1997.

[Matthies 03] L. Matthies, A. Rankin, "Negative Obstacle Detection by Thermal Signature", in *IEEE/RSJ International Conference on Intelligent Robots and Systems.* 2003.

[Matties 03] L. Matties, P. Bellutta, M. McHenry, "Detecting water hazards for autonomous off-road navigation", in *International Conference on Intelligent Robots and Systems.* Las Vegas, Nevada, USA: Jet Propulsion Laboratory, National Aeronautics and Space Administration, October 27 2003.

[McVea 06] D. McVea, K. Pearson, "Long-Lasting Memories of Obstacles Guide Leg Movements in the Walking Cat", *The Journal of Neuroscience*, 2006.

[Mianowski 07] K. Mianowski, N. Schmitz, K. Berns, "Mechatronics of the Humanoid Robot ROMAN", in *Sixth International Workshop on Robot Motion and Control (RoMoCo).* Bukowy Dworek, Poland, June 11–13 2007, pp. 341–348.

[Miura 02] J. Miura, Y. Negishi, Y. Shirai, "Mobile Robot Map Generation by Integrating Omnidirectional Stereo and Laser Range Finder", in *In Proceedings of 2002 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems.* 2002, pp. 250–255.

[Moravec 89] H. P. Moravec, D. W. Cho, "A Bayesian Method for Certainty Grids", in *AAAI Spring Symposium on Robot Navigation.* 1989, pp. 57–60.

[Moravec 96] H. Moravec, “Robot Spatial Perception by Stereoscopic Vision and 3D Evidence Grids”, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, Tech. Rep. CMU-RI-TR-96-34, September 1996.

[Nicolescu 02] M. N. Nicolescu, M. J. Matarić, “A Hierarchical Architecture for Behavior-Based Robots”, in *Proceedings of the First International Joint Conference on Autonomous Agents and Multi-Agent Systems*. Bologna, Italy, July 15–19 2002, pp. 227–233.

[Nüchter 06] A. Nüchter, K. Lingemann, J. Hertzberg, “Extracting Drivable Surfaces in Outdoor 6D SLAM”, *Proceedings of the 37rd International Symposium on Robotics (ISR '06)*, May 2006.

[Office 99] U. G. V. J. P. Office, “Joint Architecture for Unmanned Ground Systems (JAUGS)”, Redstone Arsenal, Alabama 35898, Tech. Rep., 1999. AMSAM-DSA-UG.

[OTC 08] OTC. *MTVR: Medium Tactical Vehicle Replacement – Brochure*. Oshkosh Truck Company, http://www.oshkoshdefense.com/. 2008.

[Pastore 99] T. H. Pastore, H. R. Everett, K. Bonner, “MOBILE ROBOTS FOR OUTDOOR SECURITY APPLICATIONS”, in *American Nuclear Society (ANS) 8th International Topical Meeting on Robotics and Remote Systems*. 1999.

[Patel 05] K. Patel, W. Macklem, S. Thrun, M. Montemerlo, “Active Sensing for High-Speed Offroad Driving”, in *Proceedings of the 2005 IEEE*. April 2005.

[Pradalier 08] C. Pradalier, A. Tews, J. Roberts, “Vision-based Operations of a Large Industrial Vehicle - Autonomous Hot Metal Carrier”, *Journal of Field Robotics*, 2008.

[Proetzsch 10a] M. Proetzsch, *Development Process for Complex Behavior-Based Robot Control Systems*, ser. RRLab Dissertations, Verlag Dr. Hut, 2010. ISBN 978-3-86853-626-3.

[Proetzsch 10b] M. Proetzsch, T. Luksch, K. Berns, “Development of Complex Robotic Systems Using the Behavior-Based Control Architecture iB2C”, *Robotics and Autonomous Systems*, vol. 58, no. 1, pp. 46–67, January 2010. doi:10.1016/j.robot.2009.07.027.

[Proetzsch 10c] M. Proetzsch, F. Zimmermann, R. Eschbach, J. Kloos, K. Berns, “A Systematic Testing Approach for Autonomous Mobile Robots Using Domain-Specific Languages”, in *KI 2010: Advances in Artificial Intelligence*, ser. Lecture Notes in Computer Science, R. Dillmann, J. Beyerer, U. Hanebeck, T. Schultz, Eds., Springer Berlin / Heidelberg, 2010, vol. 6359, pp. 317–324.

[Putney 06] J. S. Putney, “Reactive Navigation of an Autonomous Ground Vehicle Using Dynamic Expanding Zones”, Diploma thesis, Virginia Polytechnic Institute and State University, http://nsdl.org/, 2006.

[Quinlan 93] S. Quinlan, O. Khatib, “Elastic bands: Connecting path planning and control”, in *Proceedings of IEEE Int. Conference on Robotics and Automation*. Atlanta, 1993, pp. 802–807.

[Ranganathan 03] A. Ranganathan, S. Koenig, “A Reactive Robot Architecture with Planning on Demand”, in *Proceedings of the 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems*. Las Vegas, Nevada, USA, October 27–31 2003, pp. 1462–1468.

[Rankin 04] A. Rankin, L. Matthies, A. Huertas, “Daytime water detection by fusing multiple cues for autonomous off-road navigation”, in *Proc. 24th Army Science Conference*. Orlando, Florida, USA: Jet Propulsion Laboratory, National Aeronautics and Space Administration, November 29 – December 2 2004.

[Rankin 06] A. Rankin, L. Matthies, “Daytime water detection and localization for unmanned ground vehicle autonomous navigation”, in *Proc. 25th Army Science Conference*. November 2006.

[Rankin 08] A. L. Rankin, L. H. Matthies, “Daytime Mud Detection for Unmanned Ground Vehicle Autonomous Navigation”, in *Proc. 25th Army Science Conference*. 2008.

[Renner 06] A. Renner, “Orientierungsbestimmung bei mobilen Robotern unter Verwendung von Magnetfeldsensoren”, Project thesis, Robotics Research Lab, Department of Computer Sciences, University of Kaiserslautern, September 2006.

[Renner 08] A. Renner, “Water Detection for mobile Off-road Robotics”, Diploma thesis, Robotics Research Lab, Department of Computer Sciences, University of Kaiserslautern, 2008. unpublished.

[Rosenblatt 95] J. Rosenblatt, C. Thorpe, “Combining multiple goals in a behavior-based architecture”, in *Proc. IEEE Conference on Intelligent Robots and Systems*. 1995.

[Rosenblatt 97a] J. Rosenblatt, “Utility Fusion: Map-Based Planning in a Behavior-Based System”, in *Proceedings of FSR ‘97 International Conference on Field and Service Robotics*. 1997.

[Rosenblatt 97b] J. K. Rosenblatt, “DAMN: A Distributed Architecture for Mobile Navigation”, Dissertation, Carnegie Mellon University, 1997.

[Saffiotti 95] A. Saffiotti, K. Konolige, E. H. Ruspini, “A multivalued logic approach to integrating planning and control”, *Artificial Intelligence*, vol. 76, no. 1–2, pp. 481–526, 1995.

[Sarwal 04] A. Sarwal, J. Nett, D. Simon, “Detection of Small Water-Bodies”, PercepTek Robotics, 12395 N. Mead Way, Littleton, CO 80125, industrial/commercial XAARL, December 2004.

[Schaerli 05] N. Schaerli, “Traits – Composing Classes from Behavioral Building Blocks”, Dissertation, University of Bern, February 2005.

[Schäfer 05a] **B.-H. Schäfer**, T. Luksch, K. Berns, “Obstacle Detection and Avoidance for Mobile Outdoor Robotics”, in *EOS Conference on Industrial Imaging and Machine Vision*. 2005.

[Schäfer 05b] **B.-H. Schäfer**, M. Proetzsch, K. Berns, “Extension Approach for the Behaviour-Based Control System of the Outdoor Robot RAVON”, in *Autonome Mobile Systeme*. Stuttgart, Germany, December 8–9 2005, pp. 123–129.

[Schäfer 05c] **B.-H. Schäfer**, M. Proetzsch, K. Berns, “Stereo-Vision-Based Obstacle Avoidance in Rough Outdoor Terrain”, in *International Symposium on Motor Control and Robotics (ISMCR)*, Brussels, Belgium. November 2005.

[Schäfer 06] **B.-H. Schäfer**, K. Berns, “RAVON - An autonomous Vehicle for Risky Intervention and Surveillance”, in *International Workshop on Robotics for risky intervention and environmental surveillance - RISE*. 06 2006.

[Schäfer 07a] **B.-H. Schäfer**, P. Hahnfeld, K. Berns, “Real-time Visual Self-localisation in Dynamic Environments”, in *Autonome Mobile Systeme*. 2007, pp. 50–56.

[Schäfer 07b] **B.-H. Schäfer**, M. Proetzsch, K. Berns, “OBSTACLE DETECTION IN MOBILE OUTDOOR ROBOTS - A Short-term Memory for the Mobile Outdoor Platform RAVON”, in *International Conference on Informatics in Control, Automation and Robotics (ICINCO)*. Angers, France, May 9–12 2007, pp. 141–148.

[Schäfer 08a] **B.-H. Schäfer**, A. Hach, M. Proetzsch, K. Berns, “3D Obstacle Detection and Avoidance in Vegetated Off-road Terrain”, in *IEEE International Conference on Robotics and Automation (ICRA)*. Pasadena, USA, May 13–19 2008, pp. 923–928. ISBN 978-1-4244-1646-2.

[Schäfer 08b] **B.-H. Schäfer**, M. Proetzsch, K. Berns, “Action/Perception-Oriented Robot Software Design: An Application in Off-road Terrain”, in *IEEE 10th International Conference on Control, Automation, Robotics and Vision (ICARCV)*. Hanoi, Vietnam, December 17–20 2008.

[Schäfer 09a] **B.-H. Schäfer**, T. Braun, “Studie Wassererkenner (Zwischenbericht) – TRVS-AQVI-010809-HSTB”, HS Robotic Solutions GbR, Robotics Research Lab, unpublished, August 2009.

[Schäfer 09b] **B.-H. Schäfer**, J. Wettach, “Studie Wassererkenner (Abschlussbericht) – TRVS-AQVI-011209-HSJW”, HS Robotic Solutions GbR, Robotics Research Lab, unpublished, December 2009.

[Schmitz 05] N. Schmitz, “Satellitenortung und Sensorfusion zur Lokalisierung von Fahrzeugen in unstrukturierter Umgebung”, Diploma thesis, Robotics Research Lab - University of Kaiserslautern, December 2005. unpublished.

[Schöner 92] G. Schöner, M. Dose, “A dynamical systems approach to task-level system integration used to plan and control autonomous vehicle motion”, *Robotics and Autonomous Systems*, vol. 10, no. 4, pp. 253–267, 1992.

[Schoppers 87] M. J. Schoppers, “Universal Plans for Reactive Robots in Unpredictable Environments”, in *International Joint Conference on Artificial Intelligence (IJCAI)*. Milan, Italy, 1987.

[Schröter 05] D. Schröter, "Region & Gateway Mapping: Acquiring Structured and Object-Oriented Representations of Indoor Environments", Dissertation, Institut für Informatik der Technischen Universität München, München, Germany, November 2005.

[Schwartz 00] I. Schwartz, "PRIMUS: Autonomous driving robot for military applications", in *Unmanned ground vehicle technology*, vol. 4024. 2000, pp. 313–323.

[Shoemaker 98] C. M. Shoemaker, J. A. Bornstein, "The Demo III UGV Program: A Testbed for Autonomous Navigation Research", in *Proceedings of the IEEE International Symposium on Intelligent Control*. Gaitersburg, MD, September 1998.

[Siedersberger 01] K. H. Siedersberger, M. Pellkofer, M. Luetzeler, E. D. Dickmanns, A. Rieder, R. Mandelbaum, L. Bogoni, "Combining EMS-Vision and Horopter Stereo for Obstacle Avoidance of Autonomous Vehicles", in *Lecture Notes In Computer Science*, vol. 2095. 2001, pp. 139–156.

[Singh 02] M. Singh, S. Singh, "Spatial Texture Analysis: A Comparative Study", in *ICPR (1)*. 2002, pp. 676–679.

[Smaragdakis 01] Y. Smaragdakis, D. Batory, "Mixin-Based Programming in C++", *LNCS Springer*, vol. 2177, 2001.

[Stentz 94] A. Stentz, "Optimal and Efficient Path Planning for Partially-Known Environments", in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA '94)*, vol. 4. San Diego, CA, USA, May 8–13 1994, pp. 3310–3317.

[Strachey 00] C. Strachey, "Fundamental Concepts in Programming Languages", *Higher-Order and Symbolic Computation*, vol. 13, pp. 11–49, April 2000.

[Talukder 02] A. Talukder, R. Manduchi, A. Rankin, I. Matthies, "Fast and Reliable Obstacle Detection and Segmentation for cross-country Navigation", in *IEEE Intelligent Vehicles Symposium*. June 2002, pp. 610–618.

[Thorpe 03] C. Thorpe, J. Carlson, D. Duggins, J. Gowdy, R. MacLachlan, C. Mertz, A. Suppe, B. Wang, "Safe Robot Driving in Cluttered Environments", in *11th International Symposium of Robotics Research*. Siena, Italy, October 19–22 2003.

[Thrun 03] S. Thrun, D. Hähnel, D. Ferguson, M. Montemerlo, R. Triebel, W. Burgard, C. Baker, Z. Omohundro, S. Thayer, W. Whittaker, "A System for Volumetric Robotic Mapping of Abandoned Mines", in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. 2003.

[Thrun 06] S. Thrun, M. Montemerlo, H. Dahlkamp, D. Stavens, A. Aron, J. Diebel, P. Fong, J. Gale, M. Halpenny, G. Hoffmann, K. Lau, C. Oakley, M. Palatucci, V. Pratt, Pasc, "Stanley: The Robot that Won the DARPA Grand Challenge", *Journal of Field Robotics*, vol. 23, no. 9, p. 661–692, September 2006.

[Triebel 06] R. Triebel, P. Pfaff, W. Burgard, "Multi Level Surface Maps for Outdoor Terrain Mapping and Loop Closing", *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2006.

[Unnikrishnan 09] R. Unnikrishnan, M. Hebert, "Fast Extrinsic Calibration of a Laser Rangefinder to a Camera", Carnegie Mellon University, Tech. Rep., 2009.

[Urmson 06] C. Urmson, C. Ragusa, D. Ray, J. Anhalt, D. Bartz, T. Galatali, A. Gutierrez, J. Johnston, S. Harbaugh, H. Y. Kato, W. Messner, N. Miller, K. Peterson, B. Smith, J. Snider, S. Spiker, J. Ziglar, W. R. Whittaker, M. Clark, P. Koon, A. Mosher, J. Struble, "A Robust Approach to High-Speed Navigation for Unrehearsed Desert Terrain", *Journal of Field Robotics*, vol. 23, no. 8, pp. 467–508, August 2006.

[Wellington 04] C. Wellington, A. Stentz, "Online Adaptive Rough-Terrain Navigation in Vegetation", in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, vol. 1. April 2004, pp. 96–101.

[Wellington 06] C. Wellington, A. Courville, A. Stentz, "A Generative Model of Terrain for Autonomous Navigation in Vegetation", *The International Journal of Robotics Research*, vol. 25, no. 12, pp. 1287–1304, December 2006.

[Wettach 10] J. Wettach, K. Berns, "Dynamic Frontier-Based Exploration with a Mobile Indoor Robot", in *Joint Conference of the 41st International Symposium on Robotics (ISR 2010) and the 6th German Conference on Robotics (ROBOTIK 2010)*. 2010.

[Whittaker 04] W. R. L. Whittaker, "Red Team DARPA Grand Challenge Technical Paper", http://www.darpa.mil/grandchallenge04/, April 2004.

[Wolff 95] L. B. Wolff, "Applications of Polarization Camera Technology", *IEEE Expert: Intelligent Systems and Their Applications*, vol. 10, no. 5, pp. 30–38, 1995.

[Wooden 07] D. Wooden, M. Powers, D. C. MacKenzie, T. Balch, M. Egerstedt, "Control-Driven Mapping and Planning", in *Proceedings of the 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*. San Diego, CA, USA, October 29 – November 2 2007, pp. 3056–3061.

[Worst 02] R. Worst, F. Kirchner, "KURT2 - Eine mobile Plattform für die Robotikforschung", in *Robotik 2002*, VDI, Ed., vol. 1679. 2002, pp. 389–394.

[Wulf 03] O. Wulf, B. Wagner, "Fast 3D Scanning Methods for Laser Measurement Systems", in *International Conference on Control Systems and Computer Science (CSCS14)*. Bucharest, July 2003.

[Xie 07] B. Xie, H. Pan, Z. Xiang, J. Liu, "Polarization-Based Water Hazards Detection for Autonomous Off-road Navigation", in *2007 IEEE International Conference on Mechatronics and Automation*. Harbin, China, August 5–8 2007, pp. 1666–1670.

[Yu 06] C. Yu, D. Zhang, "A New 3D Map Reconstruction Based Mobile Robot Navigation", in *International Conference on Signal Processing (ICSP)*. 2006.

[Zhang 08] Y. Zhang, J. Jiang, "Bibliographical review on reconfigurable fault-tolerant control systems", *Annual Reviews in Control*, vol. 32, no. 2, pp. 229–252, December 2008.

Index

Acronyms

2D LRF 2D Laser Range Finder. 76, 134, 136, 227

3D LRF 3D Laser Range Finder. 137, 143, 147, 156, 159, 160, 166, 167

ARS Acute Radiation Syndrome. 1

CAD Computer Aided Design. 76

CAN Controller Area Network. 229

CMU Carnegie Mellon University. 14

CT Computer Tomography. 27

DAMN Distributed Architecture for Mobile Navigation. 17

DARPA GC DARPA Grand Challenge. 3

DBSCAN Density Based Spatial Clustering of Applications with Noise. 157

DC motor Direct Current motor. 229

DoD Department of Defense. 232, 234

DOF Degrees of Freedom. 87

DOM Degrees of Manoeuvrability. 87, 88

DSP Digital Signal Processor. 76, 182

ELROB European Land Robot Trial. 238

GIS Geographic Information System. 80, 123, 124

GPS Global Positioning System. 78, 80

IAEA International Atomic Energy Agency. 2

IMU Inertial Measurement Unit. 78

IPC Industrial PC. 229

JPR Joint Robotics Program. 234

LADAR Laser Detection and Ranging. 142

LWIR Long-wave Infrared. 170

M-ELROB Military European Land Robot Trial. 3

MDARS-E Mobile Detection Assessment and Response System - Exterior. 232

MRT Magnetic Resonance Tomography. 27

MWIR Mid-wave / Thermal Infrared. 170, 171

NASREM *NASA/NBS Standard REference Model* for Telerobot Control System Architecture. 11

NBS National Bureau of Standards. 11

NIR Near Infrared. 170

NIST National Institute of Standards and Technology. 11, 235

OA Obstacle Avoidance. 206

OA+PP Obstacle Avoidance + Local Path Planner. 206

OA+PP+PD Obstacle Avoidance + Local Path Planner + Passage Detection. 206

PET Positron Emission Tomography. 27, 28

PRIMUS *PR*ogram of *I*ntelligent *M*obile *U*nmanned *S*ystems. 231

PWM Pulse-width Modulation. 229

RCS Robot Coordinate System. 61–63

RCS Real-time Control System. 11–13, 37, 138, 140, 150, 235

RDDF Route Description Data File. 236

S(M)PA Sense-(Model)-Plan-Act. 11

SAD sum of absolute differences. 183

SCS Sensor Coordinate System. 134, 150

SLAM Simultaneous Localisation and Mapping. 142–144

SWIR Short-wave Infrared. 170, 171

TAURUS Test Automation Runtime System. 200

TÜV Technischer Überwachungsverein. 128

UML Unified Modeling Language. 39

VSP Virtual Sensor Probe. 62, 63, 119

WCS Working Coordinate System. 59–61, 63, 150, 162