

Advancing the Automated Design of Integrated Intelligent Multi-Sensory Systems with Self-X Properties

Weiterentwicklung des automatisierten Entwurfs
integrierter intelligenter multisensorischer Systeme mit
Self-x-Eigenschaften

vom

Fachbereich Elektrotechnik und Informationstechnik
der Technischen Universität Kaiserslautern
zur Verleihung des akademischen Grades

Doktor der Ingenieurwissenschaften (Dr.-Ing.)
genehmigte Dissertation

von

Kittikhun Thongpull, M.Eng.
geb. in Narathiwat (Thailand)

D 386

Eingereicht am:	4. November 2015
Tag der mündlichen Prüfung:	23. Dezember 2015
Dekan des Fachbereichs:	Prof. Dr.-Ing. Hans D. Schotten
Promotionskommission	
Vorsitzender:	Jun.-Prof. Dr.-Ing. Daniel Görges
Berichterstattende:	Prof. Dr.-Ing. Andreas König Prof. Dr.-Ing. Gerhard Fischerauer

Erklärung

Hiermit versichere ich, dass ich die vorliegende Arbeit selbst angefertigt und verfasst habe und alle benutzten Hilfsmittel in der Arbeit angegeben habe. Die vorliegende Dissertation oder Teile hiervon würden noch nicht als Prüfungsarbeit für eine staatliche oder andere wissenschaftliche Prüfung eingereicht.

Kaiserslautern, den 03.11.2015

Kittikhun Thongpull

Abstract

The recently established technologies in the areas of distributed measurement and intelligent information processing systems, e.g., Cyber Physical Systems (CPS), Ambient Intelligence/Ambient Assisted Living systems (AmI/AAL), the Internet of Things (IoT), and Industry 4.0 have increased the demand for the development of intelligent integrated multi-sensory systems as to serve rapid growing markets [1, 2]. These increase the significance of complex measurement systems, that incorporate numerous advanced methodological implementations including electronics circuit, signal processing, and multi-sensory information fusion. In particular, in multi-sensory cognition applications, to design such systems, the skill-required tasks, e.g., method selection, parameterization, model analysis, and processing chain construction are elaborated with immense effort, which conventionally are done manually by the expert designer. Moreover, the strong technological competition imposes even more complicated design problems with multiple constraints, e.g., cost, speed, power consumption, flexibility, and reliability. Thus, the conventional human expert based design approach may not be able to cope with the increasing demand in numbers, complexity, and diversity. To alleviate the issue, the design automation approach has been the topic for numerous research works [3–14] and has been commercialized to several products [15–18]. Additionally, the dynamic adaptation of intelligent multi-sensor systems is the potential solution for developing dependable and robust systems. Intrinsic evolution approach and self-x properties [19], which include self-monitoring, -calibrating/trimming, and -healing/repairing, are among the best candidates for the issue. Motivated from the ongoing research trends and based on the background of our research work [12, 13] among the pioneers in this topic, the research work of the thesis contributes to the design automation of intelligent integrated multi-sensor systems.

In this research work, the Design Automation for Intelligent COgnitive system with self-X properties, the DAICOX, architecture is presented with the aim of tackling the design effort and to providing high quality and robust solutions for multi-sensor intelligent systems. Therefore, the DAICOX architecture is conceived with the defined goals as listed below.

- Perform front to back complete processing chain design with automated method selection and parameterization
- Provide a rich choice of pattern recognition methods to the design method pool
- Associate design information via interactive user interface and visualization along with intuitive visual programming

-
- Deliver high quality solutions outperforming conventional approaches by using multi-objective optimization
 - Gain the adaptability, reliability and robustness of designed solutions with self-x properties

Derived from the goals, several scientific methodological developments and implementations, particularly in the areas of pattern recognition and computational intelligence, will be pursued as part of the DAICOX architecture in the research work of this thesis. The method pool is aimed to contain a rich choice of methods and algorithms covering data acquisition and sensor configuration, signal processing and feature computation, dimensionality reduction, and classification. These methods will be selected and parameterized automatically by the DAICOX design optimization to construct a multi-sensory cognition processing chain. A collection of non-parametric feature quality assessment functions for the purpose of Dimensionality Reduction (DR) process will be presented. In addition, to standard DR methods, the variations of feature selection method, in particular, feature weighting will be proposed. Three different classification categories shall be incorporated in the method pool. Hierarchical classification approach will be proposed and developed to serve as a multi-sensor fusion architecture at the decision level. Beside multi-class classification, one-class classification methods, e.g., One-Class SVM and NOVCLASS will be presented to extend functionality of the solutions, in particular, anomaly and novelty detection. DAICOX is conceived to effectively handle the problem of method selection and parameter setting for a particular application yielding high performance solutions. The processing chain construction tasks will be carried out by meta-heuristic optimization methods, e.g., Genetic Algorithms (GA) and Particle Swarm Optimization (PSO), with multi-objective optimization approach and model analysis for robust solutions. In addition, to the automated system design mechanisms, DAICOX will facilitate the design tasks with intuitive visual programming and various options of visualization. Design database concept of DAICOX is aimed to allow the reusability and extensibility of the designed solutions gained from previous knowledge. Thus, the cooperative design of machine and knowledge from the design expert can also be utilized for obtaining fully enhanced solutions. In particular, the integration of self-x properties as well as intrinsic optimization into the system is proposed to gain enduring reliability and robustness. Hence, DAICOX will allow the inclusion of dynamically reconfigurable hardware instances to the designed solutions in order to realize intrinsic optimization and self-x properties.

As a result from the research work in this thesis, a comprehensive intelligent multi-sensor system design architecture with automated method selection, parameterization, and model analysis is developed with compliance to open-source multi-platform software.

It is integrated with an intuitive design environment, which includes visual programming concept and design information visualizations. Thus, the design effort is minimized as investigated in three case studies of different application background, e.g., food analysis (LoX), driving assistance (DeCaDrive), and magnetic localization. Moreover, DAICOX achieved better quality of the solutions compared to the manual approach in all cases, where the classification rate was increased by 5.4%, 0.06%, and 11.4% in the LoX, DeCaDrive, and magnetic localization case, respectively. The design time was reduced by 81.87% compared to the conventional approach by using DAICOX in the LoX case study. At the current state of development, a number of novel contributions of the thesis are outlined below.

- Automated processing chain construction and parameterization for the design of signal processing and feature computation.
- Novel dimensionality reduction methods, e.g., GA and PSO based feature selection and feature weighting with multi-objective feature quality assessment.
- A modification of non-parametric compactness measure for feature space quality assessment.
- Decision level sensor fusion architecture based on proposed hierarchical classification approach using, i.e., H-SVM.
- A collection of one-class classification methods and a novel variation, i.e., NOVCLASS-R.
- Automated design toolboxes supporting front to back design with automated model selection and information visualization.

In this research work, due to the complexity of the task, neither all of the identified goals have been comprehensively reached yet nor has the complete architecture definition been fully implemented. Based on the currently implemented tools and frameworks, ongoing development of DAICOX is pursuing towards the complete architecture. The potential future improvements are the extension of method pool with a richer choice of methods and algorithms, processing chain breeding via graph based evolution approach, incorporation of intrinsic optimization, and the integration of self-x properties. According to these features, DAICOX will improve its aptness in designing advanced systems to serve the increasingly growing technologies of distributed intelligent measurement systems, in particular, CPS and Industrie 4.0.

Kurzfassung

Neuere Technologien in den Bereichen verteilter Messsysteme und intelligenter Informationsverarbeitung, wie beispielsweise Cyber-Physical-Systems (CPS), Ambient Intelligence/Ambient Assisted Living (AmI/AAL), dem Internet of Things (IoT) und Industrie 4.0, erfordern die Entwicklung intelligenter integrierter Multi-Sensor-Systeme [1, 2]. Diese Systeme heben die Bedeutung hervor, welche komplexen Messsystemen mit verschiedenen fortschrittlichen Methoden zuteil wird, darunter elektronische Schaltungen, Signalverarbeitung und die Zusammenführung multisensorischer Informationen. Der Entwurf solcher Systeme, besonders der kognitiver Multi-Sensoranwendungen mit Teilbereichen wie Methodenselektion, Parametrisierung, Modellanalyse oder Entwicklung der Prozesskette, ist mit immensem Aufwand verbunden und wird in der Regel von Experten manuell durchgeführt. Zudem verursacht technologischer Wettbewerb zusätzliche Zwänge hinsichtlich Kosten, Zeit, Leistungsaufnahme, Flexibilität und Zuverlässigkeit. Daher kann der konventionelle, auf menschlichen Experten beruhende Ansatz, möglicherweise nicht den steigenden Anforderungen in Anzahl, Komplexität und Diversität genügen. Zur Lösung dieser Schwierigkeiten ist der automatisierte Entwurf Gegenstand zahlreicher Forschungsarbeiten [3–14] und schlägt sich bereits in kommerziellen Produkten nieder [15–18]. Zusätzlich bietet die dynamische Anpassungsfähigkeit intelligenter Multi-Sensorsysteme potentielle Lösungsansätze für den Entwurf zuverlässiger und robuster Systeme. Die Ansätze der intrinsischen Evolution und Self-x-Eigenschaften [19], wie Self-Überwachung, -Kalibrierung und -Reparatur/Heilung, sind dabei mitunter die besten Herangehensweisen. Motiviert durch den andauernden Trend und basierend auf dem Hintergrund früherer Aktivitäten [12, 13] in diesem Gebiet, liegt der Schwerpunkt der vorliegenden Forschungsarbeit im automatisierten Entwurf intelligenter integrierter Multi-Sensorsysteme mit Self-x-Eigenschaften.

In dieser Arbeit wird das System DAICOX (Design Automation for Intelligent COgnitive systems with self-X properties) vorgestellt. Die Architektur dieses Systems zielt darauf ab, den Entwicklungsaufwand zu reduzieren und dabei qualitativ hochwertige sowie robuste Lösungen für intelligente Multi-Sensorsysteme zu liefern. Die Zielsetzung der DAICOX-Architektur umfasst folgende Punkte:

- Vollständiger Front-to-Back Entwurf der Prozesskette mit automatisierter Methodenselektion und Parametrisierung
- Verfügbarkeit zahlreicher Methoden der Mustererkennung
- Verbindung der Entwurfsinformationen mittels interaktiver Benutzerschnittstelle und Visualisierung in Kombination mit intuitiver visueller Programmierung

-
- Bereitstellung hochqualitativer Lösungen, welche konventionelle Ansätze durch Mehrzieloptimierung übertreffen
 - Anpassungsfähigkeit, Zuverlässigkeit und Robustheit entwickelter Lösungen durch Self-x-Eigenschaften

Abgeleitet aus diesen Zielen werden verschiedene wissenschaftliche methodische Entwicklungen und Implementierungen, besonders in den Bereichen Mustererkennung und Computational Intelligence, als Teil der DAICOX-Architektur verfolgt. Die Methodensammlung strebt eine vielfältige Auswahl an Methoden und Algorithmen an, um Datenerfassung, Sensorkonfiguration, Signalverarbeitung und Merkmalsberechnung, Dimensionsreduzierung sowie Klassifizierung abzudecken. Diese Methoden werden durch die DAICOX Entwurfsoptimierung automatisch ausgewählt und parametrisiert, um eine multi-sensorische kognitive Prozesskette zu entwerfen. Es wird eine Sammlung zur qualitativen Erfassung nicht-parametrischer Merkmale mit dem Zweck der Dimensionsreduktion (DR) vorgestellt. Zusätzlich zu üblichen DR-Methoden werden Varianten vorgeschlagen, insbesondere die Gewichtung von Merkmalen. Der Methoden-Pool soll drei verschiedene Klassifizierungs-Kategorien beinhalten. Der Ansatz der hierarchischen Klassifizierung wird vorgeschlagen und auf der Entscheidungs-Ebene als Multi-Sensor-Fusion Architektur entwickelt. Neben der Multi-Klassen-Klassifizierung werden Methoden der Ein-Klassen-Klassifizierung, beispielsweise Ein-Klassen SVM und NOVCLASS vorgestellt, um die Funktionalität der Lösungen zu erweitern, speziell zur Anomalien- und Neuheitserkennung. DAICOX zielt zum Erhalten leistungsfähiger Lösungen auf eine effektive Handhabung des Problems der Methodenselektion und Parametereinstellung für spezielle Anwendungen ab. Die Aufgaben des Prozesskettenentwurfs werden durch meta-heuristische Optimierungsmethoden durchgeführt, wie zum Beispiel Genetische Algorithmen (GA), Partikel-Schwarm-Optimierung (PSO) mit Mehrzieloptimierungsansätzen und Modellanalysen für robuste Lösungen. Zusätzlich zu den automatisierten Entwurfsmechanismen, erleichtert DAICOX die Entwurfsaufgaben mit intuitiver visueller Programmierung und zahlreichen Visualisierungsmöglichkeiten. Das Konzept der DAICOX Entwurfsdatenbank erlaubt die Wiederverwendung und Erweiterung von Lösungen basierend auf bereits erworbenem Kenntnisstand. Daher kann der gemeinschaftliche Entwurf von Maschine und Expertenwissen zum Erhalt erweiterter Lösungen genutzt werden. Insbesondere führen die Integration von Self-x-Eigenschaften sowie intrinsischer Optimierung zu Zuverlässigkeit und Robustheit. DAICOX erlaubt daher die Inklusion dynamisch rekonfigurierbarer Hardware zur Umsetzung der intrinsischen Optimierung und Self-x-Eigenschaften. Als Ergebnis dieser Forschungsarbeit wird eine intuitiv zugänglich, intelligente Multi-Sensorsystem-Entwurfsarchitektur entwickelt mit automatisierter Methodenselektion, Parametrisierung und Modellanalyse, realisiert auf einer Open-Source Multiplattform-Software. Sie enthält eine intuitive Entwicklungsumgebung

mit visueller Programmierung und die Visualisierung von Entwurfsinformationen. Der Entwurfsaufwand wird dadurch minimiert, was in drei unterschiedlichen Applikations-szenarien demonstriert wird: Lebensmittelanalyse (LoX), Fahrerassistenz (DeCaDrive) und magnetische Lokalisierung. DAICOX erzielte in allen Fällen eine bessere Ergebnisqualität als der manuelle Ansatz. Die Klassifizierungsrate wurde um 4,4% (LoX), 0,06% (DeCaDrive) und 11,4% (magnetische Lokalisierung) verbessert. Die Entwicklungszeit wurde im Vergleich zur konventionellen Herangehensweise um 81,87% reduziert. Die neuartigen Beiträge dieser Arbeit zum aktuellen Stand der Entwicklung sind im Folgenden aufgeführt:

- Automatisierter Prozesskettenentwurf und Parametrisierung für den Entwurf von Signalverarbeitung und Merkmalsberechnung
- Neuartige Methoden zur Dimensionsreduzierung, beispielsweise GA und PSO basierte Merkmalselektion und -gewichtung mit multi-objektiver Bewertung der Merkmalqualität
- Modifikation eines Maßes nicht-parametrischer Kompaktheit zur effizienteren Untersuchung der Merkmalraumqualität
- Sensor-Fusion-Architektur auf Entscheidungsebene, basierend auf dem vorgeschlagenen Ansatz der hierarchischen Klassifizierung unter Benutzung von beispielsweise H-SVM
- Sammlung von Ein-Klassen-Klassifizierungsmethoden und die neue Variante NOVCLASS-R
- Toolbox für den automatisierten Entwurf, mit Unterstützung des Front-to-Back Entwurfs, mit automatisierter Modelselektion und Informationsvisualisierung

Aufgrund der Komplexität der Aufgabenstellung wurden in dieser Arbeit noch nicht alle der identifizierten Ziele bereits erreicht und auch die komplette Architektur noch nicht vollständig implementiert. Basierend auf aktuell implementierten Programm-Modulen und Framework wird jedoch die Entwicklung von DAICOX zur vollständigen Architektur verfolgt. Potentielle Verbesserungen umfassen die Erweiterung des Methoden-Pools mit zahlreichen Methoden und Algorithmen, Prozesskettenentwicklung mittels Graphbasiertem evolutionärem Ansatz, Einbeziehung intrinsischer Optimierung und Integration von Self-x-Eigenschaften. Mittels dieser Eigenschaften eignet sich DAICOX zum Entwurf fortschrittlicher Systeme, um der wachsenden Technologie verteilter intelligenter Messsysteme zu dienen, insbesondere in den Feldern CPS und Industrie 4.0.

Acknowledgements

I would like to express my deepest gratitude to my supervisor Professor Dr.-Ing Andreas König for his suggestion of the thesis topic, and also for his incentives, discussions, and guidance throughout my PhD study. I am fully indebted to him for numerous opportunities in research and academic activities he gave to me during my stay.

I would like to thank my second reviewer Prof. Dr.-Ing. Gerhard Fischerauer for his interest in my work.

I would like to thank Prince of Songkla University for the PhD scholarship grant during the first three years of my PhD studies.

I would like to thank my colleagues Dipl.-Ing. Li Li for his supportive collaboration related to his DeCaDrive project, M.Sc. Abhaya Chandra Kammara for the useful discussion and the sharing of code on optimization techniques, and Dr.-Ing. Robert Freier for his help with German language challenges.

Above all I would like to thank my parents and my relatives for their support, encouragement, and motivation, for my success.

Contents

Erklärung	iii
Abstract	iv
Kurzfassung	vii
Acknowledgements	xi
Contents	xi
List of Figures	xvi
List of Tables	xx
1 Introduction	1
1.1 Challenges and Open Issues	3
1.2 Aims of the Thesis	5
1.3 Organization of the Thesis	7
2 Intelligent and Integrated Multi-Sensor Systems	8
2.1 Intelligent and Integrated Multi-Sensor Systems	9
2.1.1 Integrated Sensors	10
2.1.2 Intelligent Sensors	10
2.2 Multi-Sensor Measurement Systems	12
2.2.1 Heterogeneous Sensor Systems	12
2.2.2 Sensor Array	13
2.2.3 Wireless Sensor Networks	14
2.2.4 Virtual Sensor	14
2.3 Multi-Sensor Fusion	15
2.3.1 Architectures for Sensor Fusion	18
2.4 Measurement Deviation	19
2.5 Self-X Properties of Intelligent Integrated Multi-Sensor Systems	20
2.5.1 Evolvable Sensors and Sensor Electronics	23
2.6 Discussion	23
3 Design Automation of Intelligent Multi-Sensor Systems	25
3.1 Motivation	26
3.2 Overview of System Design Automation	28
3.2.1 Sensor Selection and Configuration	28

3.2.2	Sensor Electronics	29
3.2.3	Signal Processing	29
3.2.4	Dimensionality Reduction	30
3.2.5	Classification Design	30
3.3	Multi-Sensor System Design Automation Software	31
3.3.1	Concepts and Frameworks	31
3.3.2	Software Libraries	33
3.3.3	Software Suites	37
3.4	Discussion	41
4	Architecture of an Automated Intelligent Integrated Multi-Sensor Systems with Self-X Capabilities Design Framework	45
4.1	Concept and System Architecture of DAICOX	46
4.1.1	Design Building Blocks	48
4.1.2	DAICOX Design Optimization	49
4.1.3	Self-X Properties of DAICOX Designed Solutions	51
4.1.4	Design Interaction	53
4.2	Sensor Configuration	54
4.2.1	Sensing Element Configuration	54
4.2.2	Sensor Electronics Configuration	55
4.2.3	Automated Sensor Configuration	55
4.3	Signal Processing and Feature Computation	57
4.4	Dimensionality Reduction	59
4.4.1	Feature Extraction	60
4.4.2	Feature Selection	61
4.4.2.1	Optimal Methods	62
4.4.2.2	Heuristic Methods	62
4.4.2.3	Meta-Heuristic Methods	63
4.4.2.4	Filter and Wrapper Evaluation Approaches	64
4.4.3	Feature Weighting	65
4.5	Classification	66
4.5.1	Multi-Class Classification	66
4.5.2	One-Class Classification	69
4.5.3	Hierarchical Classification	71
4.6	Quality Assessment Measures	72
4.6.1	Density Based Methods	73
4.6.2	Distance Based Methods	73
4.7	Model Analysis	78
4.7.1	Data Set Separation	78
4.7.2	Selection Stability	80
4.8	Design Optimization	81
4.8.1	Computational Intelligence Based Optimization Algorithms	81
4.8.1.1	Genetic Algorithms	82
4.8.1.2	Particle Swarm Optimization	84
4.8.2	Multi Objectives Optimization	86
4.8.3	Intrinsic Design Optimization	88
4.8.4	Optimization in Dynamic Environments	89

4.9	Discussion	91
4.9.1	Overview of Parameter Settings	91
5	Implementation of the DAICOX Architecture	93
5.1	Signal Processing and Feature Computation	94
5.2	Dimensionality Reduction	95
5.2.1	Choice of Feature Subset Assessment Functions	96
5.2.2	Automated Feature Selection	99
	Sequential Methods	99
	Meta-Heuristic Methods	100
	Automated Feature Weighting	104
5.3	Classification	105
5.3.1	Multi-Class Classification	106
5.3.2	One-Class Classification	107
5.3.3	Hierarchical Classification	114
5.4	DAICOX Design Optimization	115
5.4.1	Flexible Design Variable Evaluation	115
5.5	DAICOX Software Environment	116
5.5.1	DAICOX Library Interface	117
5.5.2	Choice of Visualization	118
5.5.3	DAICOX Application Design Flow	119
5.6	Discussion	120
6	Realization of DAICOX Design and Integration: Case Studies and Applications	121
6.1	Automated Application Design of LoX	122
6.1.1	Data Acquisition	123
	6.1.1.1 Preliminary Investigation of Self-X Properties Integration	126
6.1.2	Conventional Application Design Approach	128
6.1.3	Automated Design Using DAICOX	129
	6.1.3.1 Automated Dimensionality Reduction Design	129
	6.1.3.2 Classification	133
6.2	DAICOX Automated Design Support for DeCaDrive System	148
6.2.1	Data Acquisition	148
6.2.2	Feature Computation	150
6.2.3	Dimensionality Reduction	151
6.2.4	Classification	152
6.3	DAICOX Automated Design Support for Magnetic Localization System	157
6.3.1	Timebase Synchronization for Magnetic Localization based on Pattern Recognition Approach	158
6.3.2	Dimensionality Reduction	159
6.3.3	Edge Classification Design	160
6.3.4	Simulated Real-Time Edge Classification	161
6.4	Discussion	165
7	Summary and Conclusion	166
7.1	Outcomes and Novel Contributions	167
7.2	Future Improvements of DAICOX	169

List of Abbreviations	171
List of Symbols	173
A DAICOX Software Toolboxes and Libraries	175
B Parameter Settings	179
Bibliography	181
Curriculum Vitae	196

List of Figures

1	(a) IoT development trends forecast [2], (b) 2015 Germany sensor market revenue	2
2	Block diagram of typical intelligent multi-sensor system	2
3	Architecture overview of proposed design optimization	6
4	Structure of modern embedded multi-sensor measurement system	9
5	Key technologies contributing to intelligent integrated multi-sensor systems	11
6	Taxonomy of sensors by their organization and architecture	12
7	Multi-Sensor signal representation transformation	13
8	Typical two dimensional sensor array and its data representation	14
9	Multi-sensor fusion	16
10	Multi-sensor fusion categories based on input and output configurations and three levels hierarchy	17
11	Single fusion network	18
12	Parallel fusion network	18
13	Serial fusion network	19
14	Deviations in each phase of sensor system design process [30, 34]	20
15	Comparison between conventional reconfiguration and self-reconfiguration on intelligent integrated sensor	21
16	Conceptual diagram of self-x mixed-signal SSCE with reconfiguration and actuation resources [86]	22
17	Summarized design steps and environments for intelligent multi-sensor applications	24
18	Standard procedures of intelligent multi-sensor system design	27
19	Traditional design of intelligent multi-sensor system	27
20	Block diagrams of KD process implemented in IDA (adapted from [4])	32
21	Design process of CRN toolbox on MATLAB [6]	33
22	Block diagrams of the design flow used in CSS for classification and fuzzy uncertainty (adapted form [7])	34
23	Sensor abstraction and target platform framework concept block diagram adapted from [8]	34
24	Automated system evaluation schematic proposed in PREn (adapted from [9])	35
25	Design space of vehicle tracking application design using BASELABS [116]	35
26	Image manager structure of the CVB toolkit [16]	36
27	Sensor and algorithm selection flow of freescale sensor fusion adapted from [18]	37
28	Screenshot of Tuchulcha with Charon modules in the design workspace [10]	38

29	Rapid-prototyping feature of QuickCog [13]	39
30	Example of machine vision application designed by visual programming on nVision [17]	40
31	Example of a live classification workspace and feature space visualizations in Orange	40
32	Technological time-line of state of the art	41
33	Automated intelligent multi-sensor system design architecture	47
34	Three designing layers in DAICOX	48
35	The generic building block model and corresponding tasks of IIMSS design in DAICOX	49
36	An example of local optimization at the signal conditioning design step	50
37	Three-Phase system design and deviations in each phase.	51
38	Architecture of self-x properties integration in DAICOX.	52
39	Operation of automated sensor configuration	56
40	Operational flow of signal processing and feature computation	57
41	Illustration of curse of dimensionality paradigm	59
42	Basic processing principle of FS and FE	60
43	Filter approach feature selection	65
44	Wrapper approach feature selection	65
45	Classification structure of OAA and OAO approaches	67
46	Rejection mechanism of the nearest neighbor description	69
47	Proposed hierarchical SVM classification	72
48	Illustration of overlapping measure computation with $k = 5$ at x_i instance of two-class data	76
49	Illustration of intra-class distance and inter-class distance at instance i	76
50	Model generation and evaluation process with three-group data separation	78
51	Taxonomy of commonly used optimization algorithms	81
52	Two types of crossover procedure of GA	83
53	Operations of Genetic Algorithms	83
54	Operations of Particle Swarm Optimization	84
55	Illustration of Pareto approach of two objective optimization problem	87
56	Proposed intrinsic optimization process	89
57	Overview of DAICOX framework implementation in computational intel- ligence domain	94
58	Signal processing and feature computation software flow	95
59	Dimensionality reduction architecture	96
60	Software flow block diagram of sequential AFS method	99
61	Software flow block diagram of meta-heuristic AFS method	100
62	Operation of meta-heuristic AFS method modules investigation	102
63	AFS performance comparison of the "iris" data set	102
64	AFS performance comparison of the "wine" data set	103
65	AFS performance comparison of the "wdbc" data set	103
66	AFS performance comparison of the "ionosphere" data set	103
67	AFS performance comparison of the "yeast" data set	103
68	AFS performance comparison of the "Mechatronic" data set	104
69	Operation of the AFW method module in DAICOX	105

70	Implemented classification methods	106
71	Classification implementation in DAICOX framework	107
72	ROC analysis for one-class classification method	109
73	Computed ROC curves of the "moon" data set	109
74	Computed ROC curves of the "iris" data set	112
75	Computed ROC curves of the "wine" data set	112
76	Computed ROC curves of the "ionosphere" data set	112
77	Computed ROC curves of the "wdbc" data set	112
78	Computed ROC curves of the "yeast" data set	113
79	Computed ROC curves of the "mechatronic" data set	113
80	Implementation of hierarchical classification module	114
81	Flexible design optimization interface	116
82	Example of design variable descriptor of the C-SVM module	116
83	DAICOX software library and visual programming toolbox interface	117
84	Application design steps using DAICOX architecture	120
85	LoX device based application operational architecture	122
86	LoX food ingredients recognition application design overview	123
87	Hardware block diagram of LoX device	124
88	Impedance magnitude and phase spectrum plots of the SoyVine data set	125
89	Proposed hardware structure for self-calibration/trimming	127
90	Flowchart of proposed self-calibration/trimming	127
91	A conventional design workspace for LoX application	128
92	Visual programming workspace of dimensionality reduction experiments	131
93	Radar plot of DR method rankings in differnet peformance characteristics	131
94	Bar plots comparison of DR methods for LoX data sets	132
95	Hierarchical classification processing structure of LoX data set	134
96	Visual programming workspace of multi-class classification design experiments	134
97	Visual programming workspace of one-class classification design experiments	135
98	Radar plot of OCC method ranking of a particular performance characteristics	138
99	Feature space projection of the "UsedOil" data set and its class probability vectors for hierarchical classification	139
100	Feature space projection of the "Oil" data set and its class probability vectors for hierarchical classification	139
101	Feature space projection of the "Beer" data set and its class probability vectors for hierarchical classification	139
102	Feature space projection of the "7Wine" data set and its class probability vectors for hierarchical classification	140
103	Feature space projection of the "Milk" data set and its class probability vectors for hierarchical classification	140
104	Feature space projection of the "WineGly" data set and its class probability vectors for hierarchical classification	140
105	Feature space projection of the "SoyVine" data set and its class probability vectors for hierarchical classification	141
106	Feature space projection of the "Salt" data set and its class probability vectors for hierarchical classification	141

107	Feature space projection of the "Powder" data set and its class probability vectors for hierarchical classification	141
108	Average classification accuracy of OCC methods using LoX data sets with full features	142
109	Average classification accuracy of OCC methods using LoX data sets with AFS	142
110	Average classification accuracy of OCC methods using LoX data sets with AFW	142
111	Average number of stored reference vectors of OCC methods using LoX data sets with full features	142
112	Average number of stored reference vectors of OCC methods using LoX data sets with AFS	142
113	Average number of stored reference vectors of OCC methods using LoX data sets with AWS	142
114	Complete processing chain of a LoX application in an automated design and run-time workspace	144
115	Back-end automated design flowchart of DAICOX	145
116	An example of reduced design space complexity for the complete LoX application design process with constrained design space regarding the parameter settings given in Table 17	145
117	Summary of the design aptness of DAICOX compared to conventional design approach	146
118	DeCaDrive system	148
119	Design flow block diagram of DeCaDrive processing chain	149
120	Overall feature quality results of the DR design process for the DeCaDrive data set	152
121	Feature map of the the DeCaDrive data set with full features	153
122	Feature map of the the DeCaDrive data set with AFS features	153
123	Feature map of the the DeCaDrive data set with AFW features	153
124	Processing structure of DeCaDrive system with hierarchical classification	154
125	Block diagram of DAICOX solution integration for the on-line DeCaDrive system	156
126	Block diagram of the magnetic localization systems	157
127	Coil switching logic sequence of the first two coils.	158
128	Processing structure of classification based edged detection system for measurement synchronization.	160
130	Classification performance over processing window size	161
129	Feature space plot of the labeled data set for training procedure. The blue dots are the data collected from "Idle" state and the red dots are the data collected from "Active" state.	161
131	Simulated real-time edge classification results of recorded data stream at a position near a corner of the test chamber	162
132	Simulated real-time edge classification results of recorded data stream at the center of the test chamber	162
133	Technological time-line of the state of the art and the commencement of DAICOX	170
134	Taxonomy of DAICOX software	176

List of Tables

1	Description of each level of multi-sensor fusion hierarchy	17
2	State of the art comparison of automated design systems for intelligent multi-sensor applications	43
3	State of the art comparison <i>cont.</i>	44
4	Description of basic signal processing methods [124]	58
5	Comparison of feature selection methods	64
6	Support vector machine kernel functions	68
7	Dissimilarity measures for two data vectors X_j and X_i where $x_{i,m} \in X_i$ and $x_{i,m} \in X_j$	74
8	Description of data set separation methods	79
9	Recommended parameter settings	92
10	Signal processing and feature computation in DAICOX	95
11	Description of feature subset assessment functions in DAICOX	97
12	Sensitivity characteristics of feature assessment functions	98
13	Parameter settings for meta-heuristic AFS methods	100
14	Real-world data set descriptions	102
15	Average number of iterations used in the AFS methods investigation	104
16	Parameter settings for stochastic AFW methods	105
17	Detail of parameters and searching range for classification optimization	107
18	Synthetic data sets description. In the feature space plots, the blue dots are the positive class instances, the red dots are outliers. The boundaries between positive clusters and outliers are equal to the maximum 1-NN distance of the positive instances. All data sets contain 500 samples of positive class and 1,000 samples of negative class.	110
19	Classification performance comparison of implemented OCC methods	111
20	Computation performance comparison	111
21	Real-World data set results	111
22	DAICOX library interface descriptions	118
23	Parameters of the LoX device configuration	125
24	LoX data set descriptions	125
25	Possible integrated reconfiguration capabilities for self-trimming process	127
26	Parameter settings for AFS methods used in the LoX case study	130
27	Multi-objective quality measure of full-feature	132
28	Overall result of dimensionality reduction tasks	132
29	Parameter settings for classifier optimization methods	135

30	Tentative design complexity comparison	137
31	Classification performance obtained from flat SVM classifiers designed by conventional approach	137
32	Multi-Class classification results	137
33	Overall design process comparison between conventional approach and DAICOX	147
34	Descriptions of the computed features of DeCaDrive data	150
35	DR process results of DeCaDrive data	151
36	DeCaDrive classification results of reduced training data	155
37	DeCaDrive classification time in $[\mu s]$	155
38	Comparison of the solution obtained by DAICOX and the manual ap- proach from the early work	155
39	Technical details of magnetic localization demonstrator used in this case study	158
40	List of available signal processing and feature computation methods . . .	159
41	Average of the best design solutions of 10 runs	160
42	Feature space quality comparison	160
43	Classification performance in the design phase	161
44	Edge detection performance of the simulated real-time scenarios	163
45	Comparison of generalization edge detection results by using DAICOX compared to manual approach	164
46	DAICOX features with regard to the criteria investigated in the state of the art survey	168
47	DAICOX features with regard to the criteria investigated in the state of the art survey <i>cont.</i>	168
50	DAICOX toolbox interface	177
51	DAICOX toolbox interface <i>cont.</i>	178
52	Experimental results and parameter settings of flat SVM for LoX and DeCaDrive data sets	179
53	Hierarchical SVM parameter settings for LoX and DeCaDrive data sets. .	180

Chapter 1

Introduction

The significant growth of commercial interest and research activities in Cyber Physical Systems (CPS) [20], Ambient Intelligence/Ambient Assisted Living systems (AmI/AAL), the Internet of Things (IoT) [21], and Industry 4.0 [22] boosts enormous developments and the advance of microtechnologies, both for integrated electronics and for integrated sensors. The global production of mobile sensors is predicted by the Tsensor summit society to be at trillion units per years by 2022 as a result from the growing demands in smart sensing technologies [1]. Yole Développement has envisioned the potential future sensing generation to be integrated sensors, expecting that by 2020 as shown in Fig. 1 these will be the solution for the majority of IoT [2]. The sensor market revenue development was estimated to be increased four percent in the third quarter of 2015 by the members of the AMA association¹, which will also increase the revenue to be gained over 140% from 2010. These are in accordance to micro-technology development in increasing new functions to be embedded as reported in the ITRS road map [23] and the More-than-Moore direction [24].

¹<http://www.ama-sensorik.de/en/press/press-releases-2015/>

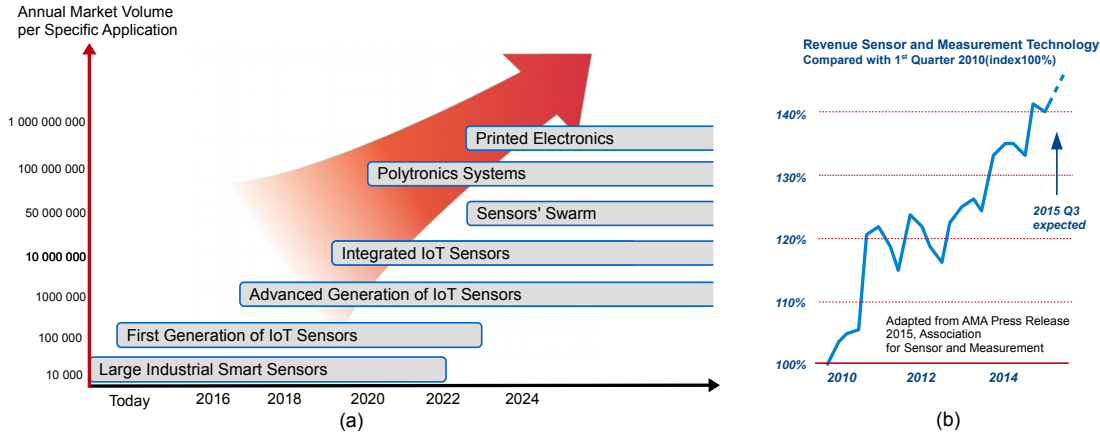


FIGURE 1: (a) IoT development trends forecast [2], (b) 2015 Germany sensor market revenue

According to these technological forecasts and road maps, the demands of intelligent integrated autonomous measurement systems with small, low cost, low-power, flexible, and reliable features will be tremendously increased and will be the main challenges for the design [20]. In particular, as to serve the increasing desires and needs on delicate features and extensive functionalities for up-to-date devices, intelligent sensory system integration become essential in broad applications, e.g., smart environments with autonomous measurement devices [25, 26], advanced driver assistance systems [27], and intelligent condition monitoring and inspection [28, 29]. However, in many complex applications, intelligent solutions with single sensor utilization approach, may be insufficient to derive trivial information in achieving desiderata in potentially complex applications with cost constraint. Therefore, multi-sensor approach accompanied with intelligent capabilities is the potential solution to provide high performance measurement systems at low cost [30, 31, 81].

Intelligent systems for potentially complex pattern recognition tasks with multi-sensory context are composed of a number of processing methods and algorithms ranging from conventional signal processing to advanced computational intelligence. Fig. 2 depicts the standard processing structure of an intelligent multi-sensor system comprised of sensors or a sensor array and corresponding sensor electronics, data acquisition, signal processing, dimensionality reduction, and decision making.

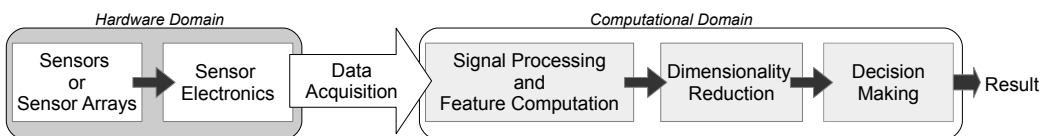


FIGURE 2: Block diagram of typical intelligent multi-sensor system

The developing of such systems involve a number of scientific and technological design procedures, however, at present, the majority of the process is still done manually by expert designers, which are costly and potentially lack availability. In addition, human centered design process tends to produce specialized and singular solutions, i.e., human bias as given in [32], which may not allow flexibility and the quality strongly depends on knowledge and experience of the designer. Moreover, the increasing demands in complex functionality impose multiple design constraints to human-expert centered design approach, that may lead to the delay of design time and extra cost. The challenges in the design of an intelligent multi-sensor system can be summarized into two cases: the first is to increase capability, feature, performance, and robustness of the target system, where the other is to achieve a rapid, flexible and efficient design process. Thus, to pursue the challenges, an autonomous design approach can be a promising solution to comply with those requirements and desiderata, while accelerating the design time and providing good quality solutions at as minimal as possible cost and effort.

1.1 Challenges and Open Issues

The aim of this thesis is motivated from the prior ISE² research focusing on design assistance and design automation of intelligent sensor systems concepts and the need of practical industrial applications. Previous design tool implementation, i.e., proprietary QuickCog [13] with rapid prototyping and visual programming has been used in numerous intelligent multi-sensor applications and research topics. Stefanie Peters [12] presented advanced automatic design framework of image processing systems for visual inspection applications. Extending from image information to generic heterogeneous sensory context, Kuncup Iswandy proposed concepts, methodologies, and frameworks for automated design of intelligent multi-sensor systems including multi-objective optimization [30, 33]. Machine-in-the-loop learning concept has been proposed in [34] to manipulate dynamic perturbations in real-time industrial manufacturing situations. These well established tools, concepts, and methodologies navigate the direction and draw the baseline of this thesis for further implementations and extensions.

In the same research direction, several research topics have been pursued contributing to the design automation of intelligent multi-sensor systems. Several research works have proposed automated methods for sensors configuration [35, 36], which is an important part for some application like gas sensing [37]. The reduction of data dimension are getting broad range of emerging automated methods of discovering best feature subset or their transformations [38–41]. The design of classification process has been the topic of research for automated methods selection and parameterization [42, 43]. However,

²Institute of Integrated Sensor Systems

these studies focus on a particular part or partial process of the pattern recognition system. The comprehensive integration framework of pattern recognition system design have been the subject of several research works. In [44], the authors proposed component based framework for data mining system design. The authors in [45], proposed assistance environment for pattern recognition system based on meta-learning with meta-feature selection. Software tools for assisting the design process have been developed in a growing number of software libraries [6–8] and software suites [10, 14, 17]. The commercial companies also pay significant interest in this area as can be seen from several released products, e.g., Freescale Sensor Fusion [18], BASELABS [15], and nVision [17].

In addition to the well establish design tools, the fast growing demands of complex functionality are often extended beyond the extent of the current developed system ability. Therefore, systems, which are not adequately sustainable or without adaptive capability will be obsolete within a short time. In particular, the maintenance of deployed systems impose cost, time, and operation interruptions, whereas a system can be attractive when it comes with higher degree of independence or self-sustainable in order to cope with dynamic influences or deviations. This issue has been emphasized as a requirement in the NAMUR road map [46] for industrial sensor applications. These gain more and more significance of the dynamically adaptivity integration to intelligent multi-sensor systems. Dynamic reconfiguration capabilities and optimization on run-time platform, i.e., intrinsic optimization, offer flexible adaptation and adjustments during operation time. Thus, self-x properties [19], e.g., self-monitoring [47], -calibrating/trimming [48], and -healing/repairing [49], are among the key features of longterm, dependable, and robust systems. Intensive investigation will be carried out in more detail in chapter 2 and 3.

Concluding from the addressed activities and topics, open issues and research gaps, that will be the potential issues to be tackled can be summarized as follows:

- A number of the investigated works are based on homogeneous sensor information, e.g., image processing.
- Back-end processing approach solely focused, e.g., knowledge discovery or data mining.
- Comprehensive processing chain construction (as shown in Fig. 2) design systems are rarely found.
- Design optimization, in particular, meta-heuristic with multiple objectives and constraints optimization are sparsely integrated.
- The dynamic adaptation of hardware instances, e.g., self-x properties and intrinsic optimization issues are almost untouched.

These gaps and open issues are regarded as the aims and goals of the thesis discussed in the following section.

1.2 Aims of the Thesis

This thesis proposes on advanced design automation of multi-sensor technical cognition systems based on methods and algorithms in the areas of sensor signal processing, multi-sensor fusion, and computational intelligence. Design Automation for Intelligent COgnitive system with self-x properties, DAICOX, architecture is conceived with the aim of reducing design effort and to providing high quality, flexible, and robust solutions. DAICOX will consist of multi-objective design optimization for the design of cognition application processing chains from available processing components in the method pool. Design support of DAICOX shall cover the complete standard building blocks consisting of sensor configuration and data acquisition, signal processing and feature computation, dimensionality reduction, and classification. DAICOX will be elaborated to effectively handle the problem of method selection and parameter tuning for yielding high performance solutions with less design effort. DAICOX is aimed to carry out the design tasks by using meta-heuristic searching approach, e.g., Genetic Algorithm or Particle Swarm Optimization, with multi-objective optimization approach. Moreover, visualization and visual programming will be the features of DAICOX design environment to facilitate design activity and designer interaction. In particular, the integration of self-x properties as well as intrinsic optimization into its platform to gain reliability and robustness are the main goals of the DAICOX architecture. In Fig. 3, the overview structure of the proposed design process, including extrinsic and intrinsic optimization schemes, is illustrated. According to the conceived architecture, the aims of this thesis pursuing towards the highlighted challenges, research gaps, and open issues are formulated as listed below:

- Contribute to the development of design automation architecture for intelligent integrated multi-sensor systems providing high quality solutions with as minimum as possible design time and effort.
- Automated complete processing chain construction from rich choice of methods including data acquisition, signal processing and feature computation, dimensionality reduction, and classification.
- Development of advanced pattern recognition methods, e.g., hierarchical classification approach and One-Class Classification (OCC).
- Overall systematic software integration of the proposed methodologies with intuitive graphic user interface, visualization, and visual programming.

- Meta-heuristic based optimization integration along with multi-objective optimization from choices of assessment function.
- In-the-Loop learning capability with intrinsic optimization concept.
- Dynamic reconfiguration capability implementation on hardware instance with self-x properties to improve robustness and to provide fault tolerance systems.

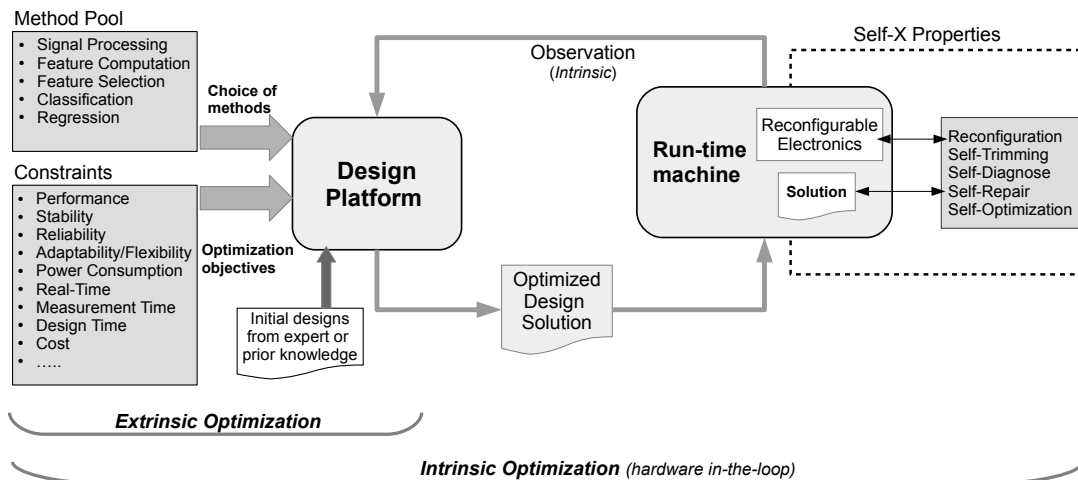


FIGURE 3: Architecture overview of proposed design optimization

1.3 Organization of the Thesis

The content of the thesis is structured as listed below:

- Chapter 2: The principles and background of intelligent multi-sensor system are described. The fundamentals of multi-sensor measurement and signal processing as well as multi-sensor data fusion techniques are provided.
- Chapter 3: State of the art survey of research works and technologies related to design automation of intelligent integrated multi-sensor systems is presented.
- Chapter 4: DAICOX architecture is proposed and elaborated. The methodologies and techniques of the method pool are discussed in detail.
- Chapter 5: The current implementation of the DAICOX architecture are explained along with preliminary studies and experiments.
- Chapter 6: The selected case-studies are presented as well as experiments. The results and discussions are described.
- Chapter 7: The thesis is concluded and discussed on the potential extensions and improvements.

Chapter 2

Intelligent and Integrated Multi-Sensor Systems

Trends in modern measurement and instrumentation technologies are evolving towards the increase in sensing performance and information reliability. Further, the increasing requirements in perception improvement is being beyond the extent of conventional measurement systems with single sensor. One potential approach is the combination of multiple sensors with advanced electronics and intelligent capability. Figure 4 shows the structure of an autonomous multi-sensor measurement system on an embedded platform. A variety of sensing elements or transducers transform physical properties, e.g., mechanical, thermal, optical, magnetic, and biochemical stimulations to electrical representations, e.g., voltage or current. Sensing elements are predominately passive components, which produce weak amount of electrical signal vulnerable to influences, e.g., noise and drift. Thus, analog treatments such as amplification, offsetting, and filtering enhance the sensor signals before the conversion to digital domain using single or multi-channel Analog to Digital Converter (ADC). Digital Signal Processing (DSP) techniques process digital representations of converted signals enhancing quality of the data as well as to extract new information. An embedded processor, e.g., Microcontroller, DSP processor, FPGA, or Application Specific Integrated Circuit (ASIC), is commonly used for handling such tasks. Advanced procedures for calibration or compensation can be conducted involving controlled excitation elements, i.e., actuators. For example, a temperature sensor can be calibrated from relative heat changes generated by a controlled heating element. Complex intelligence measurement systems, e.g., sensor networks, push more significance on the communication part for transferring sensory context to externally interconnected systems.

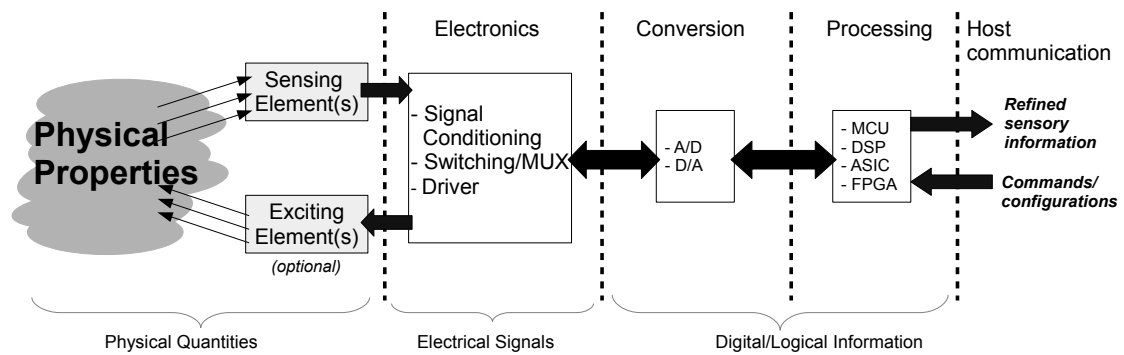


FIGURE 4: Structure of modern embedded multi-sensor measurement system

Although the architecture may seem efficient and dependable to some degree, however, some challenging issues are still open, e.g., performance improvement or robustness. Intelligence and integration approaches play an important role to fill the gap by extending the architecture with advanced flexible electronics and powerful computational intelligence algorithms. The scope of the system can be divided into two fields of integrated sensor systems related to advanced hardware integration and intelligent sensor systems related to sensory context computations. This chapter begins with state of the art survey of ongoing research and technological development activities of intelligent measurement systems. Afterwards, the principles of the methods techniques and algorithms are given before the discussion section summarizes the details of this chapter.

2.1 Intelligent and Integrated Multi-Sensor Systems

The fast development of micro technology is emerging more and more miniaturized sensors and complex integrated circuits. As the fact that a micro scale sensor may produce signal proportional to its dimensions, miniaturized sensors without integration of signal enhancing unit may susceptible to influences. According to that, sensing systems become more and more integrated solution, i.e., realization of several sensing elements, electronic as well as processing components in a single device, e.g, sensor modules in 3D package or MEMs sensors [50]. Moreover, the need of intelligent characteristics of measurement systems are introduced by the increasing requirements on precision, reliability, and stability of complex tasks. Thus, integrated sensor and intelligent sensor concepts play the key roles in the development of measurement technologies.

2.1.1 Integrated Sensors

Complex sensor measurement system integrations have been traditionally done by single and multilayer rigid or flexible boards [51]. However, increasing demand on miniaturizing size, low-cost, and rapid design impose sheer constraints to the solutions. Rapid developments of micro technology and packaging production offer potential integration methodologies in several scales and fashions including System-on-Package [52], 3D Integrated Packaging [53], Sensor on a Chip [54] as well as Lab-on-Chip [55]. These technological concepts have been focused in a growing number of research topics in advanced sensor and intelligent measurement area. Wafer level multi-functional integrated sensor [56] comprises of temperature, humidity, pressure, air speed, chemical gas, magnetic field, and acceleration sensing elements on a single 2×2 mm chip for mobile device in an environmental awareness application. In [57], an optical sensor system composed of an embedded photodetector imaging array and an analog front-end, a mixed-signal processor, and an integrated optical interferometric waveguide were encapsulated in a System-on-Chip SOC package for aqueous and gaseous sensing applications. Thermal tilt sensors, piezoresistive compass and accelerometers, and digital signal processor are integrated in a single chip for orientation determination system [58] using monolithic CMOS technology. In [59], a magnetic field based indoor localization system was proposed with 3D AMR sensors integrated with analog front-end in Active-Multi-Layer (AML) technology, which allows complex integration with low-cost and fast manufacturing time. As the common aim, these technologies combine several components and functions into a single device to realize effective system level solutions for compact size, low cost, and high performance. The other potential advantages incorporate the expansion of functionality and reduction of energy consumption. Intelligent capability, reconfiguration ability, and built-in testing and calibration methodologies will be the advanced features of dependable and robust integrated sensor systems.

2.1.2 Intelligent Sensors

Sustainable and dependable multi-sensor systems rely on the basis of learning and adaptation capabilities, i.e., intelligence. Within the last decades, intelligent sensor terminology has been inferred in diverse ways. Intelligent sensors deliver advantages such as integration of diagnosis and reconfigurability [60]. An intelligent sensor is expected to demonstrate multi-channel sensing and fusion, be able to make decisions based on information or current situations, and adapt or adjust itself regarding to the changing conditions context [61]. Self-diagnose, on-line reconfiguration increase reliability, which is the key factor of intelligent sensors [47]. Self-calibration has been addressed as an important function of intelligent sensors [62]. Intelligent sensors should demonstrate

robustness, dependability, and ability to self-calibrate [63]. In safety applications, intelligent sensors are the vital building blocks for reliable and dependable systems [64]. Concluding from the investigations, important functions and characteristics of intelligent sensors are outlined as follows:

Processing functionality: The ability to provide relevant information in an efficient representation derived from signal processing, data reduction and fusion, context classification, and decision making procedures.

Adaptation capability: The ability to change operational configurations as the reaction to encountering situations by using dynamically reconfigurable techniques such as measurement error correction, self-tuning/adjustment and self-calibration/trimming.

Dependability: addresses the capacity of awareness in erroneous or critical conditions and be able to decide relevant actions to recover healthy status. This character includes self-monitoring, self-repair/healing and fault tolerance system.

Clearly, the design of such systems requires knowledge and experience in diverse fields ranging from conventional signal processing to complex machine learning. In other word, computing tasks consisting of feature computation, pattern recognition and decision making are expected to be embedded in intelligent sensor systems. In addition, reconfigurable sensor electronics are essential to support full flexibility and adaptation of intelligent sensor systems.

Concluding from the investigations, the development of intelligent integrated multi-sensors lays on advances in hardware integration technologies and advances in software algorithms. Key technologies transferring to the design of intelligent integrated multi-sensors systems are depicted in Fig. 5.

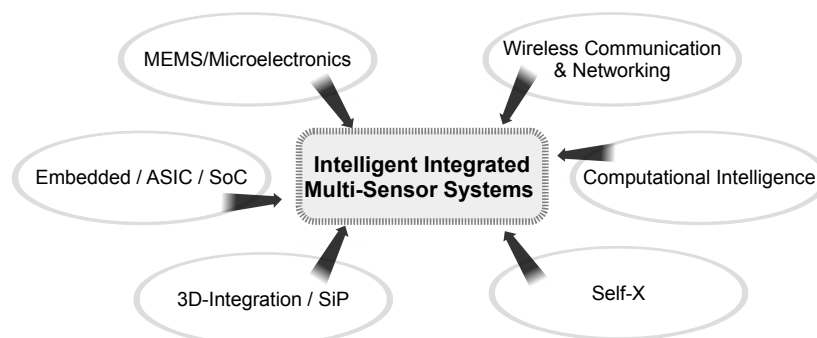


FIGURE 5: Key technologies contributing to intelligent integrated multi-sensor systems

2.2 Multi-Sensor Measurement Systems

This section provides background of measurement systems with multiple sensors. A combination of multiple sensing elements is generally applied to complement or increase sensing capability when singular sensor measurement reaches limitations or faces difficult situations [65]. Multiple sensors provide the ability to enhance signal estimation [66], increase the accuracy of system performance [67], and result in new and robust information [81]. Multi-sensor measurement systems are comprised of a group of sensors, multi-channel or single-channel with multiplexer analog electronics and data conversions, and a processing units. The group of sensors typically are a number of sensing elements, which can be of identical operating principle, i.e, homogeneous, or a combination of various physical quantities, i.e., heterogeneous. Figure 6 shows a taxonomy of sensor categories based on a formation of sensing element. This section provides the fundamentals of the categorized multiple sensor measurement systems. Advance topologies like sensor networks and virtual sensors are also provided as they can contribute to multi-sensor systems as well.

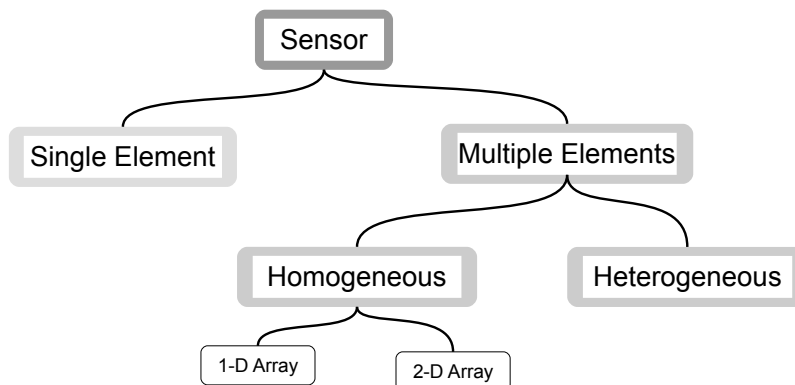


FIGURE 6: Taxonomy of sensors by their organization and architecture

2.2.1 Heterogeneous Sensor Systems

Heterogeneous sensor systems are composed of two or more different types of sensing element delivering a variety of physical representations in order to compensate or enhance perception quality, for instance, pH sensing compensated by temperature of measuring objects [68]. Cooperative operations also utilize heterogeneous sensory information by deriving original representations to provide new information, for instance, the localization system fusing information from gyroscopic and acceleration sensors as well as vision data [69]. Acquired information of a heterogeneous sensor system represents in form of multi-variable signal of dimension K , the number of observed sensor elements. Each observation in discrete time domain results in a vector $\mathbf{x} =$

$\{x_1(n), x_2(n), \dots, x_K(n)\}, n = 1, 2, \dots, N$ where N is the number of samples. The realization of the heterogeneous sensor system data acquisition conducts several components overlapping in analog signal and digital data area, i.e., mixed-signal processing. Starting from the analog signal conditioning delivers refined sensor signals to a multi-channel ADC. Then, the computation parts take place to utilize the converted data with effective signal processing methods. The overview structure of heterogeneous sensor system is illustrated in Fig. 7.

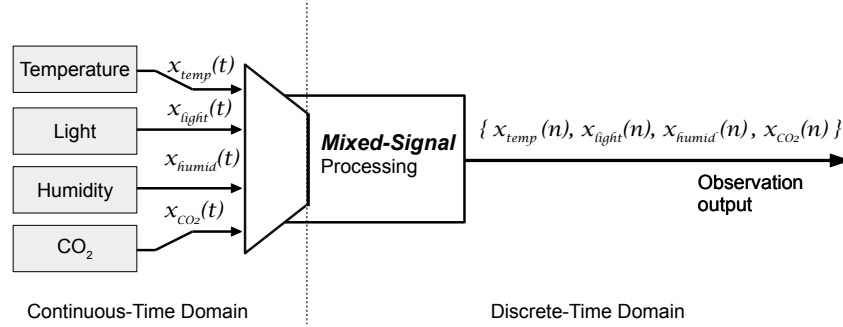


FIGURE 7: Multi-Sensor signal representation transformation

2.2.2 Sensor Array

Sensor array is a configuration type of homogeneous sensing elements or sensing cells in a certain geometry pattern aimed to increase spatial coverage of physical properties. A sensor array consists of a group of homogeneous sensors arranged in both one dimension (1-D) linear array or two dimensional (2-D) grid. The placement of sensing cell is usually in a fixed pitch or known positions, whereas each sensing element may be configured with different properties. For example, a multi-color sensor with different dielectric spectral filters [70] applied in each photo diode of its corresponding spectrum as well as in gas sensor application [71]. Sensor arrays and pattern recognition techniques have been the subject of numerous application areas such as biomedical [72], acoustic vision [73], temperature sensing [74], and tactile sensing [75, 76]. At each observation, the data acquisition of sensor array returns a vector for 1-D array or a matrix for 2-D array whose number of elements is equal to number of sensing cells. In Fig. 8 illustrates a 2-D array of i rows and j columns, which produces data matrix \mathbf{X} with size of $i \times j$ elements. Using sensor array provides numerous advantages such as increasing spatial coverage and obtaining redundant information. However, the computational complexity and measurement time of a sensor array increases quadratically by its dimensions, which are the issues to be tackled for a system design with real-time constraint. Sensor selection and dimensionality reduction techniques are essentially to be considered during the design to obtain efficient and powerful array based sensor systems [77].

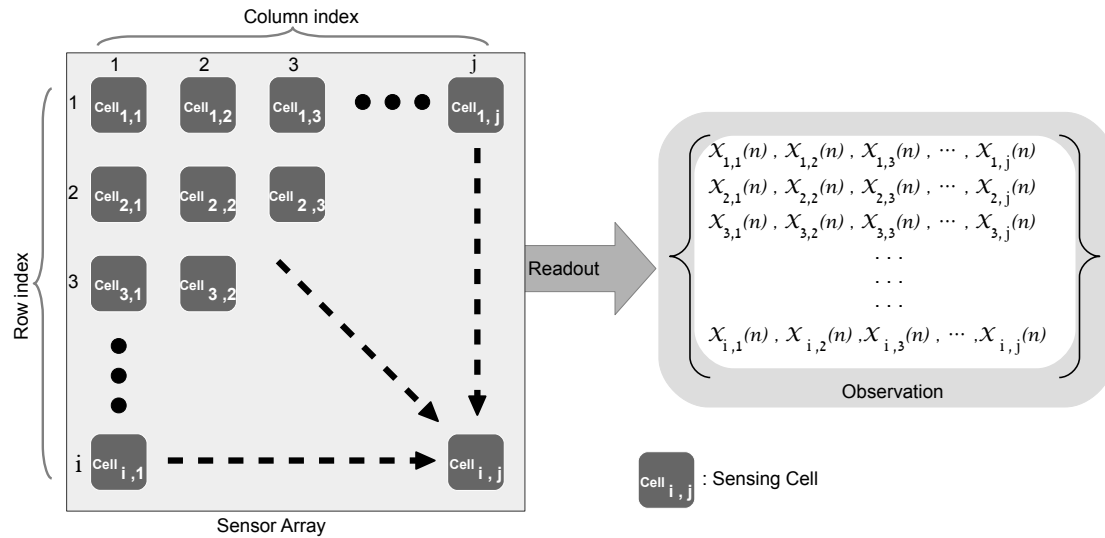


FIGURE 8: Typical two dimensional sensor array and its data representation

2.2.3 Wireless Sensor Networks

The communication technologies of distributed measurement systems have been widely developed from wired connections to radio networks or even globally connected scheme, e.g., Machine to Machine (M2M), CPS, and IoT. Nowadays, Wireless Sensor Networks (WSNs) are applied in numerous domains, e.g., smart home, biomedical, agricultural and industrial automation. WSNs node, a mobile sensor device, comprises of sensing unit, embedded processor, communication radio and power management. The radio unit provides wireless communication channels exchanging information within a network, which generally employs IEEE 802.15.4, ZigBee, Bluetooth or WiFi. Self-powered ability harvesting energy from the environment [78] is the salient feature providing dependability for sensor nodes in term of prolonging operational life-time. As a sensor node is equipped with a dedicated processing unit, extension of adaptive electronics and effective algorithms can realize intelligent integrated multi-sensor systems based on WSNs architecture. Attractive features of WSNs including node localization, self-organizing network and energy harvesting are desirable as well.

2.2.4 Virtual Sensor

A multi-sensor system can be constructed by a number of logical sensor inputs more than the number actual physical sensors by using software created sensors or virtual sensors [79]. Virtual sensors consist of transformation operators, which can be based on deterministic modeling algorithms or statistical analysis techniques. It may give predictions of upcoming data, estimates missing values, and validate measured data. Several virtual sensors can be different in the parameters and configuration of their processing

methods, whereas share the same physical data source. A virtual sensor contains paired variables of the independent input variable from data source (single sensor) and the dependent output variable. For instance, virtual sensors perform data mapping of independent input vector \mathbf{x} to new dependent data value y by using transformation function f .

$$y = f(\mathbf{x}) \quad (2.1)$$

Transformation functions can be model based, i.e., deterministic software sensors or data driven, i.e., black-box or statistical software sensors. The input vector contains a number of components regarded to the size of independent data to be transformed $\mathbf{x} = \{x_1, x_2, \dots, x_d\}$, where d is the number of independent samples. An implementation of temporal sampling based virtual sensor concept has been proposed in [80], where additional sensor inputs were used to compensate the normalization parameters to achieve generalization of the estimation models of the virtual sensors. Statistical virtual sensors have been proposed in [59] by remapping original information using RBF networks and support vector regression techniques.

2.3 Multi-Sensor Fusion

When a number of sensors are connected to a system, appropriate and efficient methods for multi-sensor information combination are essentially important to fully utilizing the capability of multiple sensors. Multi-sensor data fusion [31] is the field, that has been active since decades and is still a dominant part in most research and applications related to intelligent multi-sensor. The authors in [81] addressed the desired outcomes of sensor fusion, that are robustness, extended spatial and temporal coverage, increased confidence, reduced ambiguity, and system performance characteristics, e.g., representation, certainty, accuracy and completeness can be improved by multi-sensor data fusion [31]. Sensor fusion can be described as the process of combining multiple observations taken from sensor sources. Another benefit of multi-sensor fusion is that it can compensate the cross-sensitivity of cheap sensors, thus, high performance and robust measurement systems can be achieved at low-cost. In an observation, sensor related information are encapsulated in a common representation O [31]. The information in a sensor observation may include:

E : The entity of physical property being measured by the sensor.

\mathbf{u} : The spatial location of the sensor, i.e., the geometrical location of the sensor or the position of the measured physical property.

t : The time instant at the point that physical property is measured.

\mathbf{y} : The measured value(s) of the physical property, i.e., sensor measurement.

$\Delta\mathbf{y}$: The errors in \mathbf{y} , e.g., measurement, environment, and calibration error.

All information contained in an observation can be used in the fusion procedures. In a multi-sensor system, the measurement results in multiple sensor observation with regard to the number of observed sensors. For example, multi-sensor system with K sensors, each sensor observation contains a five element tuple of sensor information.

$$O_i = \langle E_i, \mathbf{u}_i, t_i, y_i, \Delta y_i \rangle, i \in \{1, 2, \dots, K\} \quad (2.2)$$

As illustrated 9, showing basic operation idea of multi-sensor fusion, a fusion operator or fusion node takes sensor observations from K sources to reproduce a new data representation output R , which can be represented by observations, features, or decisions depending on fusion structure.

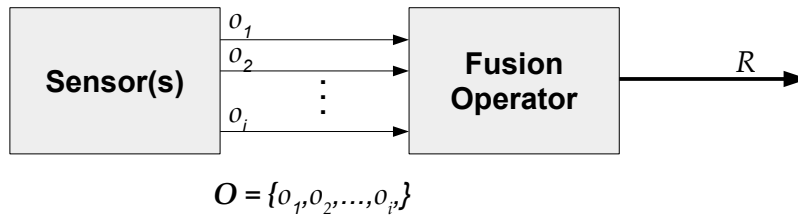


FIGURE 9: Multi-sensor fusion

Multi-Sensor fusion can be classified into three different objectives based on sensor configurations [65]:

Complementary approach takes data from sensor sources at the same time and location to ensure the consistency of observed ones.

Competitive approach acquires homogeneous observation from two or more identical sensors and decide the best one to give output.

Cooperative approach combines two or more physically different sensors to derive new information, which can not be obtained from either of the sensors.

In many cases, a combination of several fusion nodes is applied in a multi-level fusion scheme. Hence, multi-sensor fusion can be categorized into a three-level hierarchy [81]: data level, feature level, and decision level fusion. Data level fusion is the first level in sensor fusion architecture, that combines several sources of original raw data producing new refined data with identical representation or to extract new features. Feature level fusion combines a set of extracted features, for instance, mean, variance, color, textures,

or positions into new feature vector or decision information, i.e., classification. Decision level fusion is the highest level that combines multiple decisions' information to judge the final decision output. The details and method examples of each level in a multi-sensor fusion hierarchy are given in Table 1. An additional subcategory defined by fusion node input and output configurations [81] is introduced and combined with the three-level fusion model to avoid ambiguity in categorization. Figure 10 illustrates the relation between the three-level categorization and its I/O based sub category.

TABLE 1: Description of each level of multi-sensor fusion hierarchy

Fusion Level	Description and Example
Data: Generate new data by modifying/enhancing input data set	Data smoothing, zooming, resampling, filtering, estimation, remapping, and pixels calibration of segmented image
Feature: Extract new feature registration from sensor data and provide fused information to decision level	Model fitting, Object detection, Texture analysis Feature extraction, Feature selection, Classification, Multi-Dimensional Scaling
Decision: Combined transformed information	Rule-induction, Classification Tree, Fuzzy rule, Ensemble Classification Hierarchical Classification

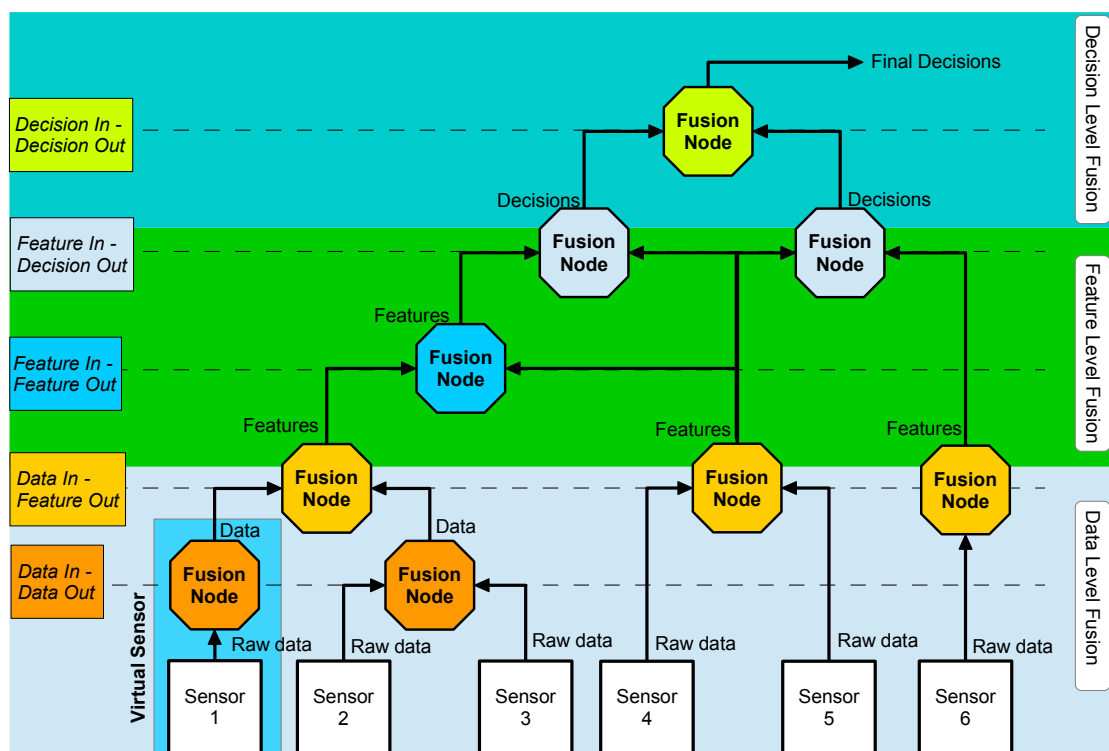


FIGURE 10: Multi-sensor fusion categories based on input and output configurations and three levels hierarchy

2.3.1 Architectures for Sensor Fusion

In general, a multi-sensor fusion system consists of multiple fusion nodes, which can be organized in three types of network arrangement or architecture [31, 81]:

Single Fusion Node: The simplest way of dealing with a multi-sensor fusion problems is to combine all observations into a single fusion node as shown in Fig. 11. It is also called centralized fusion.

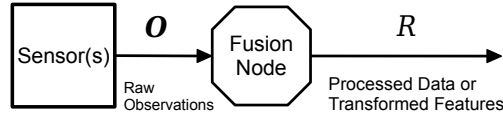


FIGURE 11: Single fusion network

Parallel Fusion Network: A number of fusion nodes are composed in this architecture. Each node separately processes its own sensor observations, resulting in fused information, which are connected to the central fusion node as input channels for the final level fusion node as depicted in Fig. 12.

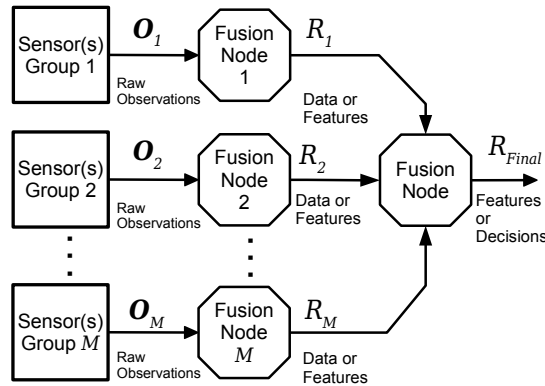


FIGURE 12: Parallel fusion network

Serial Fusion Network: The fusion nodes are constructed in a cascade connection arrangement as illustrated in Fig. 13. Each node dependently processes with one or more information from another nodes. The entire operation of the network exhibits a sequential scheme. The first node in the sequence is independent from any nodes output while others require information from their precedent sequences.

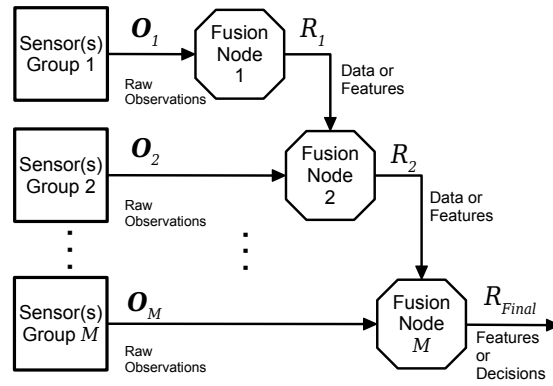


FIGURE 13: Serial fusion network

2.4 Measurement Deviation

High quality intelligent measurement systems are required to provide high accuracy [82] with high repeatability [83] of measured information. However, measurement uncertainty is the major challenge of the requirement, that reflects the lack of exact knowledge of the value of the measurand [84]. Deviations in measurement systems are among the major causes of measurement uncertainty, which are the common phenomena of sensors and sensor electronics operations. Source of the deviations can be manufacturing tolerance and environmental influences. Deviations can be categorized as given below [30, 34, 85, 86]:

Static Deviation: During the hardware components manufacturing, static deviations are introduced due to industrial process tolerances. For example, deviation of the transistor dimensions and thickness of the layers. By individually analyzing a device, static deviations can be determined and later be compensated or eliminated. Trimming techniques can be manually employed to minimize the errors affected from static deviations during deployment phase, however, this technique is time consuming and costly operations.

Dynamic Deviation: Dynamic deviations are usually caused by environmental dynamics at different time and locations. The source of variations may be in the form of heat, vibration, and illumination as well as electrical influences, e.g., noise and induced magnetic field. These quantities stochastically disturb the systems and are difficult to anticipate during the design phase, which will degrade performance of the systems. Dynamic reconfiguration capability on hardware instance is the solution to cope with this issue.

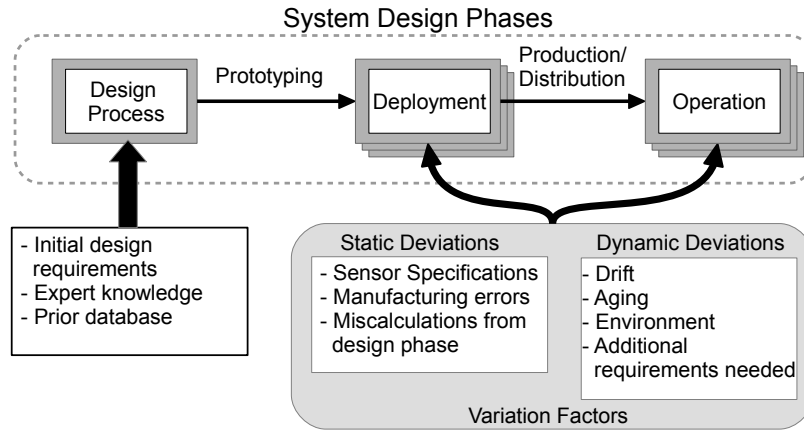


FIGURE 14: Deviations in each phase of sensor system design process [30, 34]

As depicted in Fig. 14, once a design is finished the deployment of prototypes conducts the adjustment of parameters and configurations compensating to the static and dynamic deviations. Thereafter, the tuned prototype will be used as a mold for high volume productions for the real products. However, the operations of the final products are still vulnerable to unanticipated dynamic deviations at different installation sites and run-time, that will ultimately deteriorate the performance. More over, additional requirements will lead to the necessity of adjusting deployed hardware instances. Of cause, re-adjustments are needed at every unit and in some case redesigns may have to be considered. These are costly and time consuming tasks. Therefore flexible and adaptable sensor systems with reconfiguration capability show the potential ability to tackle the issues to maintain performance at different operating situations.

2.5 Self-X Properties of Intelligent Integrated Multi-Sensor Systems

Intelligent multi-sensor systems face the challenges of increasing demand for robustness as well as fault tolerance on system integrations as to maintain quality over operation life-time. The main challenge is to deal with the aforementioned deviations, that inevitably decrease system performance from the designed specifications. In addition, supporting rapid prototype design with adaptation capability on hardware instances is also a key issue for reducing time to market. Thus, situation awareness together with responding actions over self-reconfiguration capabilities are the potential approaches to cope with dynamic influences, improve overall performance, and reduce time and effort in the development. Figure 15 depicts the overview structure of intelligent sensor systems, where the left pane is a conventional reconfiguration approach while the other one is integrated with self-x concept. Both share the same common procedures. First, received

commands will be interpreted into sensor measurement actions. The prior database of static deviations can be used to configure sensors and sensor electronics. Second, the sensor signals, which are compensated from static deviations, will be transferred to the sensor fusion process for giving the final result. However, dynamic deviations still may occur during the measurements and cause unpredictable errors. By adding observation and assessment units for optimization on the device, i.e., *intrinsic optimization*, the dynamic influences can be reduced or eliminated, thus, retaining high quality of the final result.

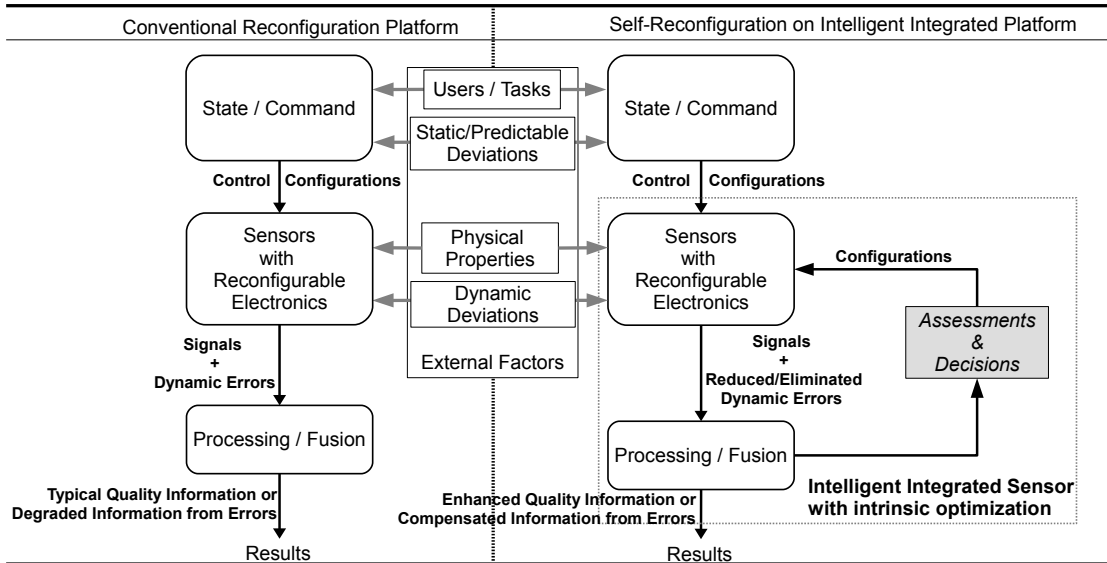


FIGURE 15: Comparison between conventional reconfiguration and self-reconfiguration on intelligent integrated sensor

Self-X features for intelligent multi-sensor are categorized as follows.

Self-Reconfigurable Sensors: Self-Reconfigurable sensors are integrated with dynamic software and hardware capabilities. Software is flexible by its nature, however, reconfigurable software require additional programmable memory space to store dynamic functions and parameters. Generic dynamic hardware architecture [87] based on reconfigurable analog electronics is comprised of digital control switches to select a variety of passive components and programmable gain amplifiers. Advanced integrated configurable mixed-signal electronics concept was proposed to enable self-x properties [85]. The block diagrams of self-x supported mixed-signal reconfigurable sensor electronics are depicted in Fig. 16.

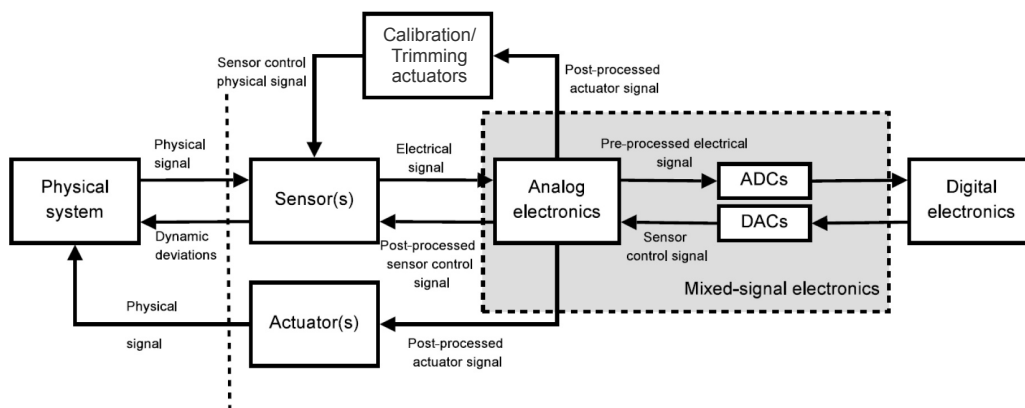


FIGURE 16: Conceptual diagram of self-x mixed-signal SSCE with reconfiguration and actuation resources [86]

Self-Monitoring: or self-diagnosis is the ability of determining current states of system performance and health status and detect occurred faults as well as predicting the sensor failure and maintenance supervision requirement. This is also denoted by other different names, e.g., self-validation or self-checking [47], where the functionality is almost identical. When sensors are not in acceptable conditions, self-calibrations task should be engaged.

Self-Calibration/Trimming: is the ability of a sensor system, that autonomously re-adjusts calibration parameters [48]. Evaluation results from self-diagnosis procedures trigger self-calibration to perform. This function requires action operators, i.e., controlled micro actuators and switching devices to provide reference physical quantities for compensation process. Self-Trimming is the advanced level of self-calibration, where hardware adjustments using reconfigurable electronics will take place to recover sensor signals.

Self-Repair/Healing: In some situations, self-calibration may not be able to retain deteriorated performance, which can be defined as fault state [47, 61]. Thus, faults recovery procedure must be taken place. Self-repair or self-healing [49] are capable to determine the actions maneuvering the erroneous situations to recover sensor status. Redundant hardware is the common implementation of this feature including additional switching matrix, spare circuit areas or cells, or identical copies of sensor electronics.

2.5.1 Evolvable Sensors and Sensor Electronics

The term evolvable sensor and sensor electronics denotes the cooperation of reconfigurable electronics and evolutionary optimization operators with the aim to let the sensor adapt itself not only to sustain undesirable situations but also to improve quality [88]. Reconfigurable electronics can be carried out with different scale of programmable platforms ranging from transistor level to integrated analog array and digital logic blocks on FPGA [89]. Comprehensive integration has been developed to cope with mixed signal scenarios on Field Programmable Mixed-Signal Array FPMA [89].

Evolutionary Computation (EC) [90] is the main background of the optimization tasks. Optimization algorithms based on EC mimics biological mechanisms for instance Genetic Algorithms (GA) [91] or Particle Swarm Optimization (PSO) [92]. The process can be done on a real platform or done in a simulator or design platform, i.e., intrinsic and extrinsic evolution respectively. The optimal solution will be then interpreted to the configurations of the reconfigurable electronics or programmable devices.

The integration of evolvable sensor concept as started with DAICOX is a promising solution for longterm dependable intelligent multi-sensor systems.

2.6 Discussion

Multi sensor systems have numerous advantages through the integration of redundant and diverse sensory information. Integration technologies create more and more miniaturized yet powerful sensors and electronics enabling an intelligent multi-sensor solution in a single device. Intelligent multi-sensor systems can be integrated with advanced information processing and multi-sensor fusion to exploit sensor performance beyond the extent of single sensor approach. Self-X properties and the evolvable sensor concepts can provide higher reliability and stability to the systems, leading to longterm dependable and fault tolerant systems. These will realize longterm and stable measurement system solutions for intelligent integrated multi-sensor system application. Despite the delicate features, in fact, the design of complex intelligent measurement systems requires knowledge and experience of the designer for intensive elaboration in all parts of the systems. Moreover, design goals and requirements are usually restricted by multiple constraints, that impose additional burden onto the design. As depicted in Fig. 17, designing tasks handle a number of design procedures ranging from sensor electronics to complex multi-sensor fusion. The design process is expected to achieve all given requirements, however, compromising and trading-off may be taken place due to difficult constraints. As a result, the best solution demands huge effort, cost, and time. To tackle with the caveats of human-based design, the potential approach is to automate the design process by

using advanced optimization techniques to find the best methods and parameters for a particular design case, where self-x concept will provide well trailered solution for each hardware instance. Thus, potentially good quality solutions can be obtained by taking as least as possible effort and knowledge requirements from the designer, thus, affordable cost and time.

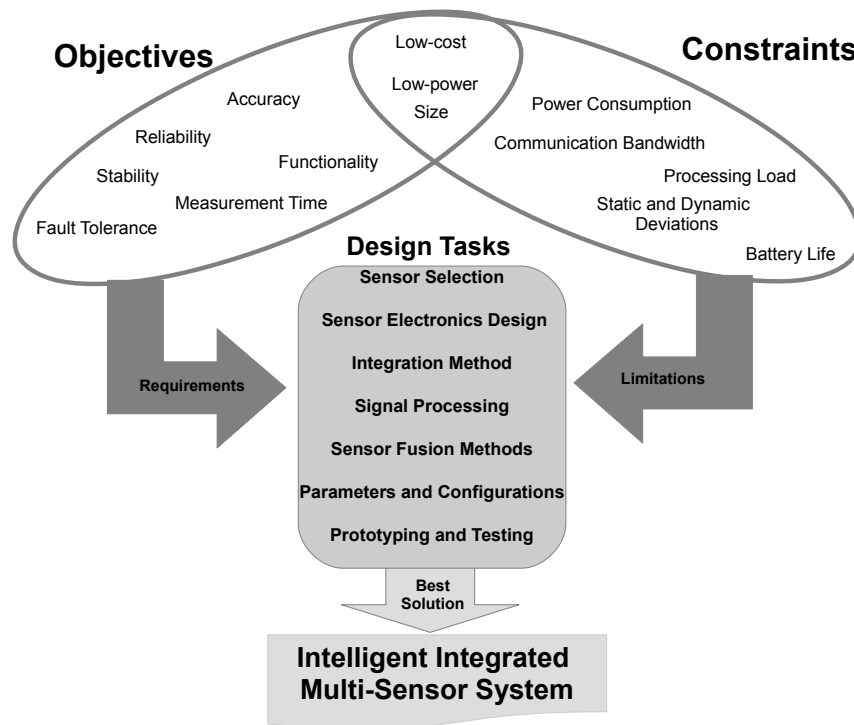


FIGURE 17: Summarized design steps and environments for intelligent multi-sensor applications

Chapter 3

Design Automation of Intelligent Multi-Sensor Systems

Due to fast technology development and strong market competition more and more complex applications of intelligent integrated multi-sensor systems have to be tackled at less development time. Due to the increasing complexity, a solution, that satisfies desired requirements and objectives, becomes highly difficult to obtain. A designer of such systems is expected to having expert skills in several areas ranging from complex hardware design to intelligent computation software programming. Even with highly skilled and experienced persons, it is often the case, that many designs consume time and cost and sometimes yield suboptimal results. An autonomous design process approach for intelligent multi-sensor systems is the main motivation of this work with the aim to yield high quality and rapid solutions with diminished requirement of designer's expertise and effort. In this chapter, recent research and technological development activities related in the field of design automation for intelligent integrated multi-sensor systems are investigated to derive the concept and system architecture proposed in this thesis.

3.1 Motivation

In general, an intelligent sensory system or product design starts from target functionality and specifications requested from users or customers. These requirements are forwarded to a (human) designer for creating a design prototype from the designer's experience or prior knowledge. Once the first prototype blue print is approved, evaluation tasks take place afterwards to assess the performance regarding to the desired specifications. Frequently, the target functionality and the performance may not achieve the commitments or satisfaction, thus, revisions and adjustments of the prototype are needed. For tuning and tracking the performance, an optimization procedure is the main task of finding the best solution to achieve such problems. Once the prototype implements all the desired specifications, then it will be used as a final design solution, i.e., template, for the deployment or production. In some scenarios, an exceptional decision making can be made by a human supervisor if part of goals can not be satisfied due to constraints or limitations. The typical system design flow with prototype optimization can be summarized as shown in Fig. 18. In conventional designs, the flow predominately occupies experienced designers at each designing step. In an intelligent multi-sensor application, the design involves numerous tasks in several disciplines including sensors and electronics, signal processing, dimensionality reduction, and decision making as illustrated in Fig. 19. The design of such a system addresses the problem of method selection and its relevant configurations at every processing block, which is an extremely high dimensional search problem especially in applications involving complex functionalities. Human based manual design approach may not be able to effectively handle such intricate search problems with reasonable time, effort, workload, and cost requirements and may involve errors and failures.

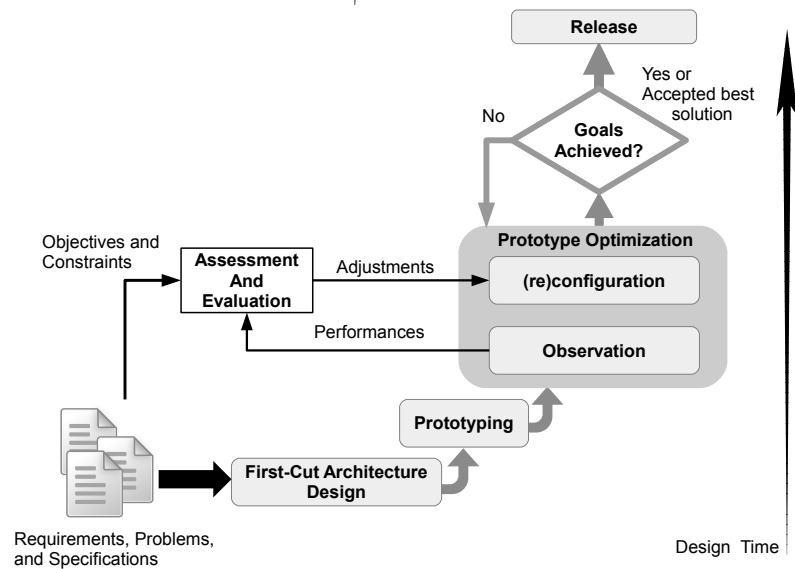


FIGURE 18: Standard procedures of intelligent multi-sensor system design

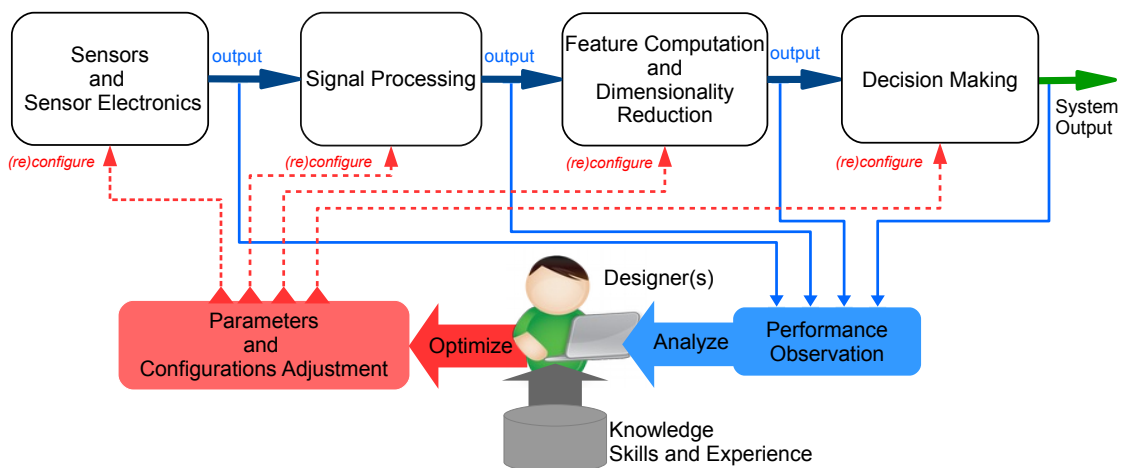


FIGURE 19: Traditional design of intelligent multi-sensor system

These emphasize inefficacious aspects of manual and human based designs, that pave the way of the thesis motivation in contributing to the design automation of intelligent multi-sensor systems with the aims to provide high quality measurement and decision making systems while minimizing design effort and time. Automated design, computer aided and designer assistance frameworks, have been the subjects predominately in the field of semiconductor circuit design automation. Several groups pursue the similar approach in data mining [3, 4, 9] and computer vision [10, 93] applications. But, the multi-sensor measurement area seems to be rarely visited from the contribution towards the design automation. For this reason, we will discuss here the concept, that fills the gap by contributing to the design automation of intelligent multi-sensor systems. Recently, the methodology and tool implementations of intelligent multi-sensor system

design automation have been established at ISE [12, 13, 30], which define the research direction and the development in this thesis. To elaborate an extension from the ISE earlier works, a number of the recently related research and technological works have been investigated and given in detail in the following sections. The investigation is split into two main parts. The first focuses on the works, that attempt to assist or automate the design of the system using methodologies and technologies related to intelligent integrated multi-sensor systems. The latter provides the information and reviews the existing frameworks and software tools.

3.2 Overview of System Design Automation

The investigations in this section are grouped into five fields of study categorized by the essential processing components, i.e., building blocks, of the intelligent integrated multi-sensor system architecture.

3.2.1 Sensor Selection and Configuration

Sensor configuration is an essential step in the design of multi-sensor systems, that should be carefully considered at the beginning of the design. Increasingly emerging sensors expand the number of choices for selection, that make this task more and more tedious, in particular, in multi-sensor applications. The task of selecting a relevant set of sensors for a measurement application is predominately manual and done by exposing the sensors to the real stimuli, which is extremely time consuming, costly, and error prone. Moreover, lack of knowledge of the designer may result inconsistency of sensor information, that degrades system performance or lead to failures. These intensify the development and research in assisting and automating sensor selection as well as configuration tasks. Performance of application specific systems, for instance, gas sensing depends on proper selection of sensing elements in the sensor array [37] to target compounds, that can significantly improve classification rate [94]. Identifying noisy sensors to be omitted from measurement can increase the degree of confidence and reliability of multi-sensor fusion systems [95]. Frequency range or bandwidth selection is also another aspect of sensor selection for impedance measurement applications [96], which can improve accuracy and reduce the amount of data to be processed as well as measurement time. Along with sensor selection, sensor parameters setting should be performed consistently with the set of selected sensors. In [97], the authors proposed an automated sensor parameter setting method by using Receiver Operating Characteristic (ROC) information over sensor fusion operators. The method requires only two user-defined parameters regardless of the number of connected sensors. Considering the

robustness issue, information reliability [98] is a significant factor used in sensor selection stage of dependable sensor system design.

3.2.2 Sensor Electronics

Sensor electronics are the front-end parts responsible for conditioning electrical sensor signals. Proper configurations of sensor electronics ensure full exploitation of dynamic range and resolution of the data in the digital domain. In [99], it is suggested that each sensor application requires an individual optimization of signal conditioning circuitry configurations to attain the highest resolution at the lowest noise level. Analog filter plays a major role in sensor conditioning to improve the selectivity of desired signals and eliminate noise. Automated analog filter compiler [100] derives the transfer function approximation and prototype synthesis from given specifications to generate the filter elements of selected network structure. By using a genetic algorithm for optimization, an automated analog circuit design system [101] was developed and demonstrated with three analog filter and amplifier design configurations. The results from generated circuits achieved all specifications without effort or knowledge taken from the designer. Modular concept of the design and configurations of sensor analog front-end architecture [102] provides a flexible way to build a complete sensor platform out of configurable block schematics.

3.2.3 Signal Processing

The signal processing part of an intelligent sensor typically involves statical calculations, digital filters, and signal estimators. A well-known computer aid tool for digital filter design is the commercial Digital Signal Processing (DSP) System Toolbox in MATLAB¹, which provides an extensive library for filter design, analysis, and implementation. In the similar fashion, SciPy [103] provides signal processing libraries facilitating digital filter design, which are free open-source and based on Python² language. These tools, however, are manual design approach with numerous sensitive parameters and the quality of a result depends on the knowledge and experience of the designer. The combination of an optimization algorithm to the design of digital filters is usually applied to obtain the best filter performance by automatically adjusting relevant parameters, hence, reducing the design effort. A gradient search algorithm was adapted to a filter model selection and parameterization framework in [104]. The framework roughly initializes the parameters by using the least-square method prior to optimization procedures, that have proven to provide better result quality with reduced resource consumptions compared to

¹The MathWorks, Inc. www.mathworks.com

²Python Software Foundation www.python.org

traditional approach. Typically, the search landscape of a digital filter design problem is generally non-linear and multi-modal, thus, meta-heuristic search algorithms are the favorable choices in order to avoid local minima. Particle Swarm Optimization (PSO) is applied in [105] to autonomously search for a filter coefficient vector, that gives minimum error and also preserve stability of the designed filter. The approach provided better performance of designed filters compared to the DSP System Toolbox tool in MATLAB.

3.2.4 Dimensionality Reduction

Reducing data dimension can diminish unnecessary computation load and can improve correct prediction rate [106]. Dimensionality reduction techniques attempt to find a smaller subset selected from the original input space with similar or even better information quality. Automated Features Selection (AFS) [107] automatically forms the best subset justified by certain measures associated with recognition performance. Alternatively, Automated Feature Weighting (AFW) [41] approach applies a weight factor to a feature, that can provide finer resolution of the feature subset quality. Exhaustive search approach, that enumerates all possible feature subset construction can ensure the discovery of the optimal subset. However, it imposes a computational burden with the complexity of $O(2^N)$ where N is the number of features, which may impractical for multi-sensor applications with high dimensional data. To reduce the cost of computation, AFS methods are usually implemented in heuristic forms, i.e., sequential selection [38], sequential floating [40], and oscillation scheme [108]. In fact, these methods may suffer from being trapped in local optima, which can be tackle by stochastic search approaches, such as, e.g., GA [109] and PSO [110]. In addition to focusing solely on recognition performance, multi-objective optimization approaches for feature selection [111], which takes feature cost of acquisition into account providing high recognition quality feature subsets with reasonable measurement cost, i.e., computational load, measurement time, and power consumption.

3.2.5 Classification Design

Due to numerous emerging algorithms and methodologies for classification, choosing a suitable one to a specific application is becoming a tedious part of designing multi-sensor cognition applications. A constructive classification operator should consist of a proper classifier algorithm with relevant parameter settings. Several approaches established towards the effort in automatically select algorithms, models, and parameters for classification tasks. Classifier selection toolbox [112] is a system for learning algorithm evaluation and selection based on meta-features, i.e., the statistical information

of a data set, to advise designer in the design of classification applications. By using estimators, which describe the best relevant classifier for a specific type of sensor from a given multi-sensory dataset, the classifier combination framework [113] increases the reliability and accuracy of the measurements. Evolutionary algorithms are usually employed for solving classifier design problems, which typically are high dimensional search space [3, 114, 115]. Particle Swarm Model Selection (PSMS) [3] is a technique for the selection of effective (individual) classification models as well as parameters through Particle Swarm Optimization (PSO) algorithm. GA based classification design with classifier ensemble approach [115] has been developed for multi-sensor pattern recognition systems by using weighted meta-features for enhancing the classification performance.

Concluding from the investigations, the common aim of these works is to reduce effort, knowledge requirement, and time of a design process as well as to provide superior performance outperforming the manual design approach. Taken these as ambitions, several software frameworks, libraries, and toolboxes have emerged, contributing to the automation of the design process for intelligent multi-sensor systems. The recent development of the software is investigated in the following section.

3.3 Multi-Sensor System Design Automation Software

This section focuses on the current state of existing frameworks and software tools related to the design automation for intelligence multi-sensor systems. The investigation is composed of three groups, first, the concept and framework group, which points out on proposed ideas, second, the surveys on available libraries or toolboxes, and the third is the collection of design automation software.

3.3.1 Concepts and Frameworks

The investigation provided here are the conceptual approaches of several research works, that provide architecture of automated design process in the field related to multi-sensor cognition system.

Full Model Selection

Full Model Selection (FMS) [3], developed by the INAOE³ institute, is a framework for pattern recognition system design aiming in increase classification performance with less time spending on designing, developing and optimizing. By giving a data set and requirements, FMS is capable of designing a viable recognition system by searching through the method pool containing pre-processing methods, feature selection agents, and learning algorithms as well as their parameters. Ensemble classification approach

³the National Institute of Astrophysics, Optics and Electronics

was employed to accomplish generalization of the best solution by increasing the diversity of the individual models. Hyperparameter optimization is proposed to account optimization control parameters in the search procedure using PSO. The software is open-source and multi-platform developed in MATLAB.

Intelligent Discovery Assistant

Bernstein et al. of University of Zurich developed Intelligent Discovery Assistant (IDA) [4], the tool that aims to support the design of Knowledge Discovery (KD) processes. The Knowledge Discovery structure in IDA, by taking the background of pattern recognition, consists of automated data pre-processing, rule-induction algorithms, and post-processing of models as shown in Fig. 20. The IDA assists its users in choosing processes to execute, for example, by ranking the process (heuristically) based on given tasks, data, and goals from the user. The decision of selecting the final solution can be made only by the user. The final selected solution can be exported for external execution in run-time but no option of reconfiguration on generated solutions available.

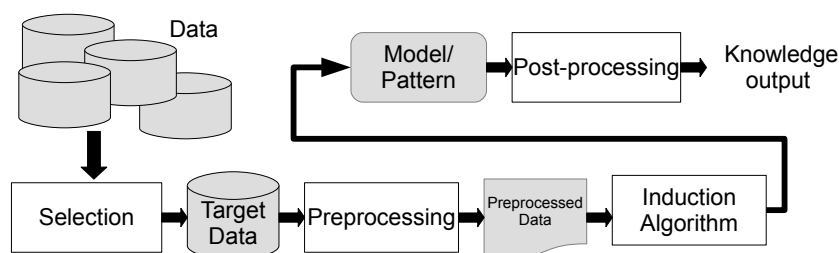


FIGURE 20: Block diagrams of KD process implemented in IDA (adapted from [4])

Self-Tuning of Teachless Process Monitoring Systems

The researchers of IFW⁴ university of Hannover [5] developed a Self-tuning Process Monitoring System (SPMS) based on multi-criteria sensor signal evaluation. The aim of the work is to reduce manual parameterization tasks and to improve process monitoring performance and reliability by using proposed self-tuning approach to the framework. The criteria consist of overall assessment, sensitivity and robustness of the monitored process, which are used in the parameter optimization. The processing of the framework consists of heterogeneous sensor data acquisitions and choice of feature extraction methods. The optimization of the system employs GA to adjust parameters of process control as well as features' computation parameters.

⁴Institute of Production Engineering and Machine Tools <http://www.ifw.uni-hannover.de/das-ifw.html?&L=1>

3.3.2 Software Libraries

Several concepts and frameworks have been implemented in the form of software plug-in or library. Using this kind of software usually requires another software or compiler, e.g., MATLAB or Python in order to execute as well as interact with users. The libraries are either open-source free-license or commercially available.

CRN

Bannach and Lukowicz at University of Passau have developed the Context toolkit [6] to reduce effort in the design of context-aware applications. The tool contains a collection of signal processing and pattern recognition algorithms. The information handling operator of the toolbox is designed to support distributed wireless sensor networks including time based synchronization of data streams. The reusable and parameterizable building blocks concept yields flexibility of the designed and configured software components when deployed in different target platforms. CRN facilitates the design process by providing graphical programming plug-in as shown in Fig. 20. The software is open-source and runs on MATLAB.

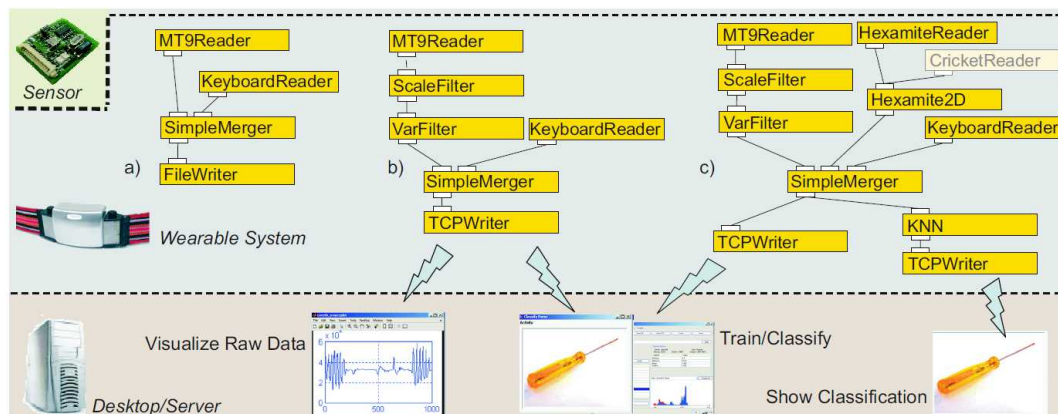


FIGURE 21: Design process of CRN toolbox on MATLAB [6]

CCS

Context Classification System [7] is a software for context aware system application design developed by the IRB⁵ research group. The system consists of multi-sensor data acquisition, feature extraction, data mapping, and fuzzy classification depicted in Fig. 22. The software facilitates in training data manipulation, however, parameters of the processing algorithms have to be adjusted manually. The framework was developed on Python and C language and implemented on mobile phone platforms using information from standard equipped sensors.

J-Sens

In [8] the authors developed an Application Program Interface (API) of the hardware

⁵The Institute of Operating Systems and Computer Networks <https://www.ibr.cs.tu-bs.de/>

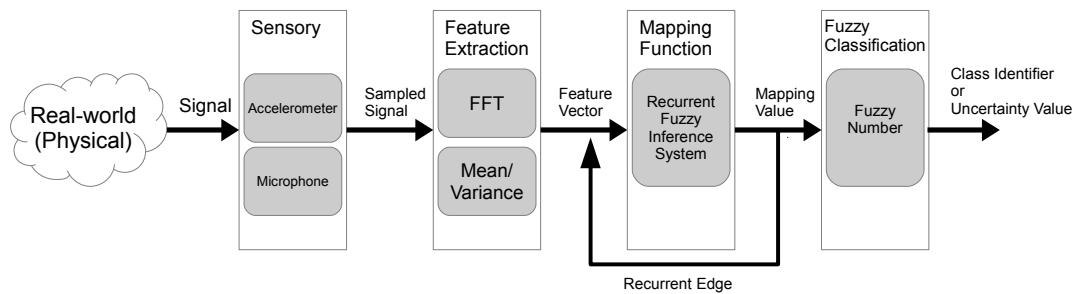


FIGURE 22: Block diagrams of the design flow used in CSS for classification and fuzzy uncertainty (adapted from [7])

abstraction layer for wireless sensor nodes over Java⁶ Integrated Development Environment (IDE) platform. The abstraction hardware representation offers the designer a transparent access of all sensor and actuator control functions to design an application with the least knowledge requirement in the hardware domain. The API supports modular design over IDE-based centralized development cycle with real-time debugging of a particular sensor environment, that allows the designer to prototype applications in hardware independent fashion.

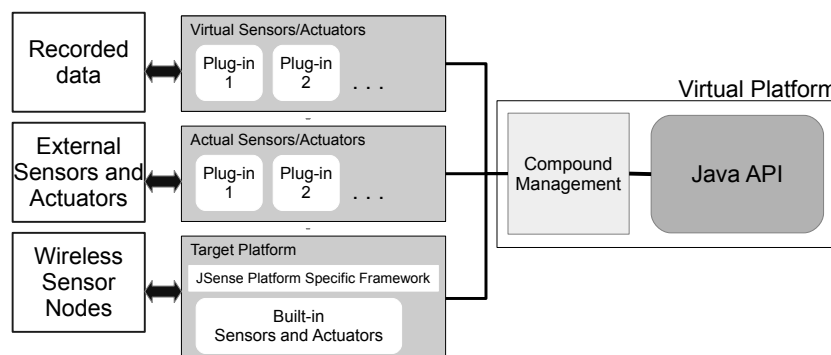


FIGURE 23: Sensor abstraction and target platform framework concept block diagram adapted from [8]

PREn

Pattern Recognition Engineering (PREn) [9] from a research group of the DFKI⁷ is a toolbox, that runs over RapidMiner⁸, a data mining software. It provides automated model construction and system evaluation for pattern recognition system design helping a user to analyze data sets with choices of classifier to find a relevant classifier for a given data set as depicted in Fig. 24. The predicted accuracy of the available classifiers will

⁶Java www.oracle.com/java

⁷Multimedia Analysis and Data Mining Competence Center <http://madm.dfki.de/>

⁸RapidMiner, Inc. www.rapidminer.com

be displayed to the user for manual selection, which are generated by support vector machine for regression by using meta-features generated from the data set [45]. The selected classifiers are parameterized with an exhaustive grid search optimization. The experiments done by repository benchmark data sets show that the toolbox helped in reducing knowledge requirement and time for pattern recognition applications design.

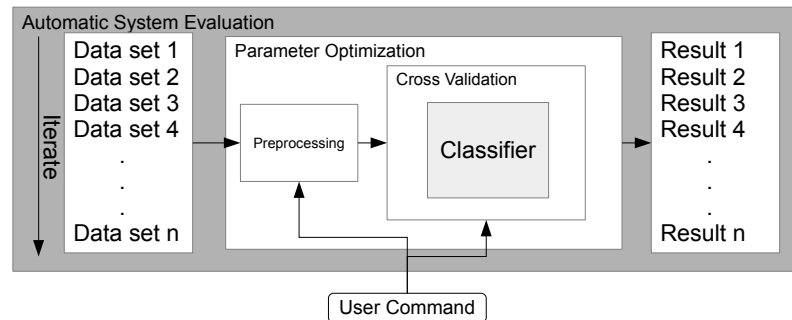


FIGURE 24: Automated system evaluation schematic proposed in PREn (adapted from [9])

BASELABS

BASELABS [15] is a software tool for rapid prototyping of system design and the parameterization for Advanced Driver Assistant Systems (ADASs) [116]. The software is associated with a library of ready-to-use sensor components for geometrical sensing and positioning, that provides assistance in sensor utilization and interfacing. Several physical sensor interfacing standards are supported, e.g., USB, CAN, FireWire, etc. The pattern recognition components of the software are based on probabilistic tracking algorithms, e.g., Kalman, Bayesian and particle filter. The software has been used to implement an example of a camera-based vehicle tracking system as a case study. The libraries used in the software are developed in Microsoft Visual Studio⁹ suit.

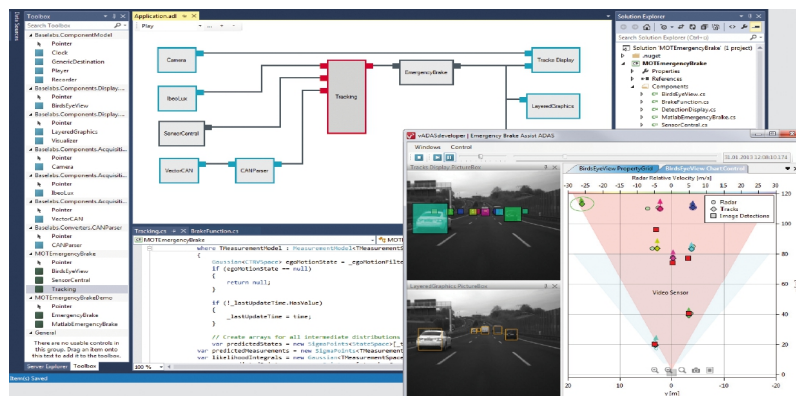


FIGURE 25: Design space of vehicle tracking application design using BASELABS [116]

⁹Microsoft Visual Studio www.visualstudio.com

Common Vision Blox

Common Vision Blox (CVB)¹⁰ [16] is an open architecture, hardware independent toolkit for imaging applications. The Image Manager in CVB offers hardware and platform flexibility by providing hardware independent acquisition API and the specific versions for both Windows and Linux operating system. A number of signal processing and feature computation libraries are provided to facilitate complex image classification tasks. These properties allow the change between different hardware without extensive re-development. A change of hardware during the design phase is possible at any time without increasing development time and losing low level control.

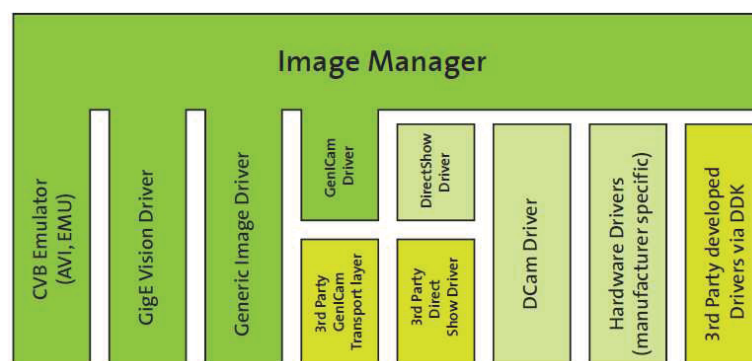


FIGURE 26: Image manager structure of the CVB toolkit [16]

GENESIS

Stefanie Peters at ISE TU-Kaiserslautern has proposed GENESIS [12], a comprehensive architecture for design automation of image processing systems. In this work, numerous texture analysis operators have been established including choice of signal processing, feature computation, and classification. In particular, Non-Linear and Oriented Kernels (NLOK), a novel method, that combines both first and second order statistics for feature segmentation, has been presented. These method are subject to optimization by GA and PSO to their optimum setting and parameters. Evolutionary strategy approach, e.g., genetic programming, has been adopted to optimize the processing chain, i.e., graph-based method breeding.

¹⁰STEMMER IMAGING GmbH <http://www.commonvisionblox.com/en/elements/fundamentals>

3.3.3 Software Suites

Investigation of software aided design tools is provided, which is composed of a collection of methods and algorithm libraries orchestrated by a central user interface software. Beside the main features in designing intelligent multi-sensor systems, visual and modular based system development environment, point-and-click, and drag and drop for the creation of processing flows are also considered in the investigation. These features are becoming common in modern computer-aided software, that simplify the creation of work-flow and reduce time and effort in the design process.

Freescal Sensor Fusion

Freescal Sensor Fusion [18] is a free library package designed for inertial and magnetic sensor families from Freescal¹¹. It comes with a GUI software for design, analysis and signal representation visualization providing an intuitive environment for multi-sensor fusion application design. User define tasks and specifications are used to derive a suitable code template and its configurations, that will be programmed to the target hardware. For an extensive analysis, the library package provides a plug-in to MATLAB for complex simulations and parameterizations. The software supports both Android and Windows operating systems, however, many advanced signal processing algorithms and visualization features are available only in the latter.

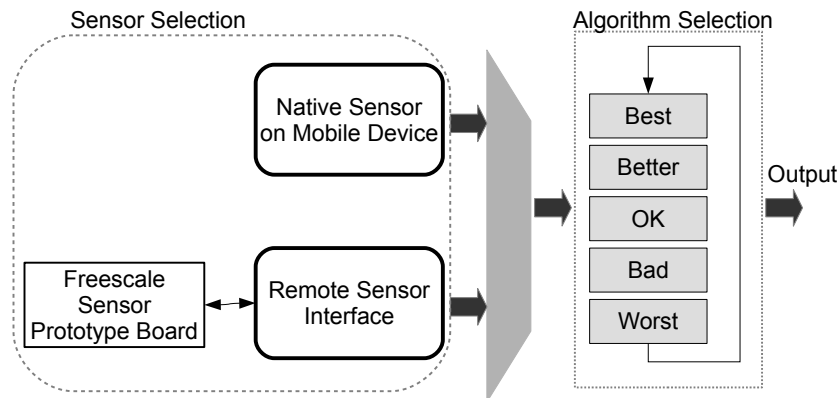


FIGURE 27: Sensor and algorithm selection flow of freescal sensor fusion adapted from [18]

RAPTOR

Rapid Application Prototyping Tool for Object Recognition (RAPTOR) [93] is a development environment software, that aims to support the design of image pattern recognition systems based on template matching technique. The application development process in the software is comprised of sensor data acquisition, data manipulation, pre-processing, and filter design. The software provides a hardware simulation of a design

¹¹Freescal Semiconductor, Inc www.freescal.com

for performance analysis and solution tuning before being exported to the real platform. This feature reduces effort, time, and cost of hardware implementation during the deployment. However, the flexibility of the deployed solution was not considered in this software.

Charon and Tuchulcha

The HCI¹² research group in the University of Heidelberg has developed Charon [10] an open-source framework for computer vision prototyping based on C++. The approach targets on implementing computer vision applications in a modular fashion. Each part of the computer vision algorithm constructed in a workspace is called a module, that can be visually connected to other algorithm modules on a design space of the tool named Tuchulcha responsible for GUI work-flow configuration application. Reusable work-flow is an interesting feature of the software helping to simplify the deployment in various target platforms from a single design center. Charon provides a collection of modules for optical flow estimation, 3D reconstruction and general image processing algorithms.

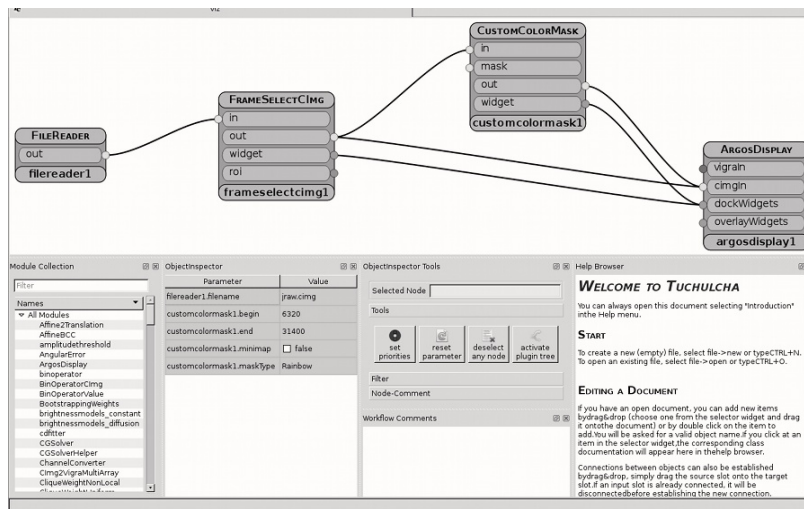


FIGURE 28: Screenshot of Tuchulcha with Charon modules in the design workspace [10]

KEEL

Knowledge Extraction based on Evolutionary Learning KEEL¹³ [11] is an open source Java software, that features knowledge extraction capabilities to assist the design of pattern recognition systems. The software employs an evolutionary algorithm to predict a model consisting of pre-processing, feature selection and post-processing for a given task. A set of statistical analysis methods is included for the analysis of the characteristics of a given data and the performance of algorithm candidates. Fuzzy rules learning approach is used to trade-off the accuracy with the interpretability of a designed model.

¹²Heidelberg Collaboratory for Image Processing <http://hci.iwr.uni-heidelberg.de/>

¹³ KEEL: Knowledge Extraction based on Evolutionary Learning <http://www.keel.es/>

QuickCog

QuickCog [13] is a fast and transparent automated design environment of multi-sensor cognition systems with self-learning capability. The main aim of the software is to accelerate the design process from the key features including data acquisition, visual programming, interactive data manipulation, feature space visualization, automatic feature selection, and pattern recognition. These features enable rapid prototype and reusable of designed solutions, which alleviate the need of expert knowledge and reduces design time for potentially complex multi-sensory cognition system applications. The software has been used in a broad range of automated visual inspection applications, e.g., medical laboratory process control [34] and semiconductor manufacturing inspection [117], while being also inherently suitable for general pattern recognition applications.

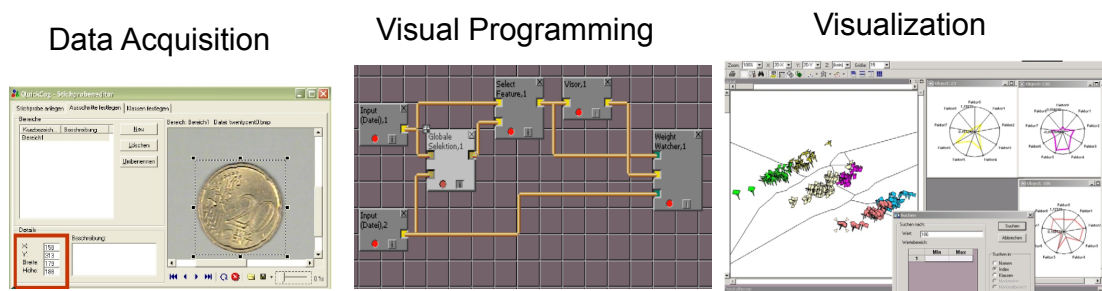


FIGURE 29: Rapid-prototyping feature of QuickCog [13]

nvision

nVision¹⁴ [17] is a development environment for industrial image processing. The main system architecture consists of nVision Designer and nVision Run-Time. An image processing task can be designed via graphical programs over nVision Designer including performance analysis and visualization. Once the final solution is approved, the compiled design will be downloaded to target platforms, that have been installed with nVision run-time software. This concept helps in facilitating the deployment phase especially in the case of different hardware platforms.

¹⁴Impuls Imaging GmbH www.impuls-imaging.com

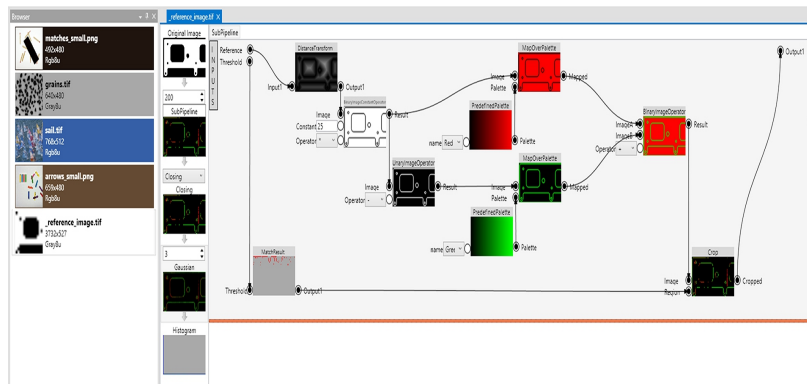


FIGURE 30: Example of machine vision application designed by visual programming on nVision [17]

Orange

Demsar et al. of the Bioinformatics Lab at the University of Ljubljana have developed Orange [14], a machine learning and data mining suite for data analysis through visual programming and Python scripting. Various useful data manipulation pre-processing and classification toolboxes, called Widgets, are provided. To design a pattern recognition task, a user can simply drag and drop the widgets and connect them with each other, creating a relevant processing flow. As it is fully open-source and non-commercial, Orange is an attractive choice for the development of additional functions, features, and effective methods of computational intelligence extended from its standard available modules.

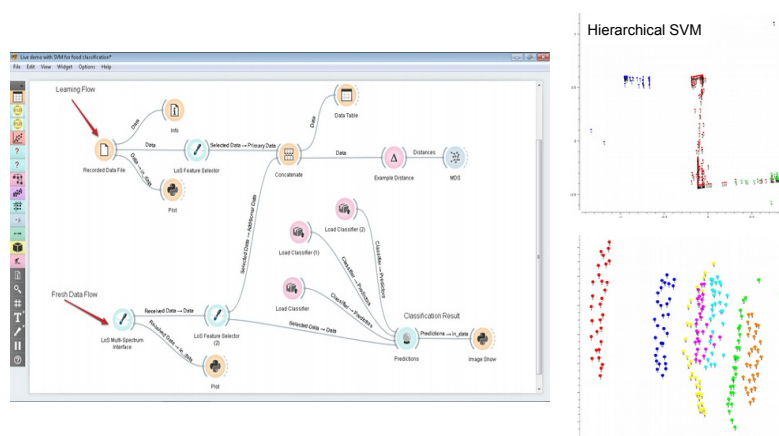


FIGURE 31: Example of a live classification workspace and feature space visualizations in Orange

3.4 Discussion

In this chapter, the techniques methods and algorithms, that contribute to the design automation of every processing component in the intelligent multi-sensor system have been surveyed and presented. Numerous software tools and libraries emerged serving in the trend of automating the design of system architecture were selected for the investigation. The similar objective of the investigated works is to provide effective solutions to the design process, that reduce effort, cost, time, and error while increasing quality and performance outperforming the human-based manual approach. The summary and features comparison of the surveyed works are given in Table 2 and 3. Visual programming and modular based design flow have become the important features in modern computer-aided design software. The design optimization is essential for the design automation of potentially complex intelligent multi-sensor systems. Evolutionary computation based meta-heuristic search algorithms have proven their applicability and efficacy in several investigated works. The capability of reusing the same designed solution in different installation sites or run-time platforms can significantly reduce effort and cost of set-up and deployment and revision during operations.

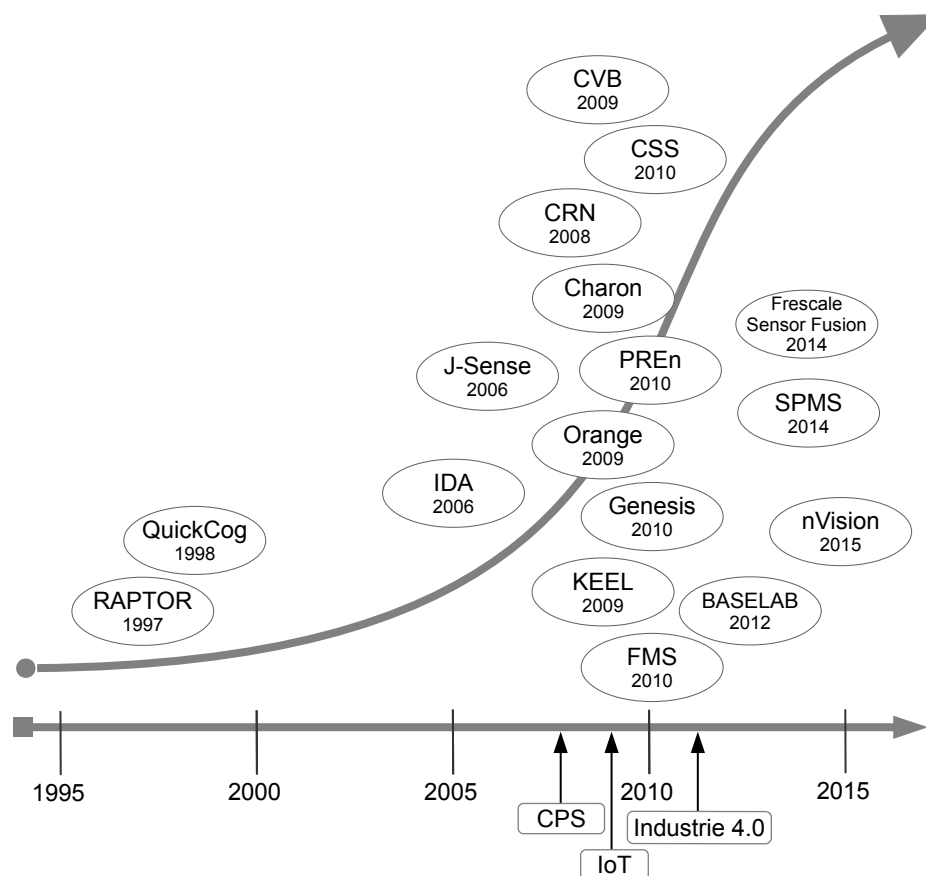


FIGURE 32: Technological time-line of state of the art

In contrast to the intensive development in this field, some salient features are apparently neglected. Complete processing chain design as well as flexibility and adaptability of the designed system on the hardware level can only be found in a few examples. The incorporation of self-x properties in dealing with reliability and robustness issue was rarely regarded. Intrinsic evolution and advanced multi-objective optimization approach seems to have been overlooked in the studied works.

Concluding from the recent development, the contribution of the thesis is to provide intuitive rapid and flexible design environment for intelligent multi-sensor systems via visual programming and informative visualization. To attain high performance and robustness solutions, advanced multi-objective optimization techniques together with the reconfigurable capability are considered to enable self-x properties and intrinsic evolution of the solutions. These features are the baseline of the development and elaboration of this thesis for conceiving an automated system design architecture, which will be described in the following chapters.

TABLE 2: State of the art comparison of automated design systems for intelligent multi-sensor applications

SoA.	Sensor Type and Interface Type	Sensory Data Format	Dimensionality Reduction	Data Analysis and Mining	Decision Making	Optimization/ Learning	Knowledge Rule Import/Export
FMS [3]	N/A Load From Database	-	Feature Selection/ Ranking, PCA	✓	Automated (Based on k -fold CV)	PSO Based Single-Objective	N/A
IDA [4]	N/A Load From Database	-	Feature Selection	✓	Manual (select by user)	N/A	Export (executable selected model)
SPMS [5]	Machinery Sensors Analog	Heterogeneous, Multi-Channel	Feature Extraction	N/A	N/A	GA Based multi-objective	N/A
CRN [6]	Sensor Modules	Heterogeneous, Multi-Channel	N/A	✓	N/A	N/A	✓
CSS [7]	Mobile Phone Sensors	Heterogeneous, Multi-Channel	Feature extraction	✓	N/A	N/A	✓
J-Sense [8]	Sensor Node Generic Sensors	Heterogeneous, Multi-Channel	N/A	N/A	Manual (select by user)	N/A	N/A
PREn [9]	N/A Load From Database	N/A	N/A	✓	Manual (select by user)	Exhaustive	N/A
BASELABS [15]	Automotive CAN Bus	Heterogeneous, Multi-Channel	N/A	✓	Manual (select by user)	N/A	N/A
CVB [16]	Machine Vision FireWire, USB	Image Data	Feature Extraction	✓	Manual (select by user)	N/A	✓
GENESIS [12]	Machine Vision (standard camera interface)	Image Data	Feature Extraction	✓	✓	GA&PSO Based multi-objective	✓
FSF [18]	Inertial Sensors I ² C, SPI, USB	Heterogeneous, Multi-Channel	N/A	N/A	Manual (select by user)	N/A	✓
RAPTOR [93]	Machine Vision (standard camera interface)	Image Data	Feature Extraction	N/A	Manual (select by user)	N/A	✓
Charon [10]	Machine Vision (standard camera interface)	Image Data	Feature Extraction	N/A	Manual (select by user)	Gradient based	Yes Import &Export
KEEL [11]	N/A Load From Database	N/A	Feature Selection Feature Extraction	✓	Automated	EA based	N/A
QuickCog [13]	Generic Sensors USB, FireWire	Heterogeneous, Multi-Channel	Feature Selection Feature Extraction	✓	Automated	N/A	N/A
nVision [17]	Machine Vision USB, FireWire	Image Data	Feature Extraction	✓	Automated	N/A	✓
Orange [14]	N/A Load From Database	-	Feature Selection Feature Extraction	✓	Automated	N/A	N/A

TABLE 3: State of the art comparison *cont.*

SoA.	Operating Platform*/ Dependent Tool	License	GUI and Visualization	Graphical Programming	Extendable Library /Programming Language	Reusable/ Executable Solution	Self-X
FMS [3]	WIN,MAC,Linux with Matlab	Open-Source	N/A	N/A	Yes Matlab scripts	N/A	N/A
IDA [4]	WIN,MAC,Linux with Weka/Java	Non-Public	Only GUI	N/A	N/A	✓ one-time code generation	N/A
SPMS [5]	WIN,MAC,Linux with Matlab	Non-Public	N/A	N/A	N/A	N/A	Self-Tuning
CRN [6]	WIN,MAC,Linux with Matlab	Open-Source	✓ Signal Analysis	✓	✓ C++	✓	N/A
CSS [7]	WIN,MAC,Linux	Open-Source	✓ Signal Analysis	✓	✓ C++	✓	N/A
J-Sense [8]	WIN,MAC,Linux with Java IDE	Open-Source	Only GUI	N/A	✓ Java	✓	N/A
PREn [9]	WIN,MAC,Linux over RapidMiner	Open-Source	Only GUI	✓ on RapidMiner	✓ Java	N/A	N/A
BASELABS [15]	WIN MS-Visual Studio	Commercial	✓ Signal Analysis	✓	✓ C++	✓	N/A
CVB [16]	WIN,Linux	Commercial	✓ Signal Analysis	N/A	✓ C++	✓	N/A
GENESIS [12]	WIN	-	✓ Image Analysis	N/A	✓ C++	✓	N/A
FSF [18]	WIN Android	Free	✓ Signal Analysis	N/A	✓ C++, Python	✓	N/A
RAPTOR [93]	WIN	Commercial	✓ Signal Analysis	N/A	N/A	N/A	N/A
Charon [10]	WIN,MAC,Linux	Open-Source	✓ Image Analysis	✓ (via Tuchulcha)	✓ C++	✓	N/A
KEEL [11]	WIN,MAC,Linux with Weka	Open-Source	Only GUI	N/A	✓ Java	N/A	N/A
QuickCog [13]	WIN	Commercial	✓ Feature Space, Image Analysis	✓	N/A	✓	N/A
nVision [17]	WIN	Commercial	✓ Image Analysis	✓	N/A	✓	N/A
Orange [14]	WIN,MAC,Linux with Python	Open-Source	✓ Feature Space Analysis	✓	✓ Python/C++	N/A	N/A

Chapter 4

Architecture of an Automated Intelligent Integrated Multi-Sensor Systems with Self-X Capabilities Design Framework

Intelligent Integrated Multi-Sensor Systems (IIMSS) for complex recognition tasks are composed of advanced methods and algorithms with numerous parameters. Designing such systems involves exacting tasks, such as method selection, combination, and parameterization, which together require tremendous design effort. For this reason, we establish the Design Automation for Intelligent COgnitive system with self-X properties, DAICOX, architecture. The architecture tackles the design effort reduction by adopting design automation concepts emerged in other domains, e.g., integrated circuit design and machine vision. DAICOX contrives an IIMSS by automatically selecting processing components from the method and algorithm pool and adjusting their parameters to achieve desired requirements. In particular, multi-objective optimization is the key function of the architecture ensuring the highest quality of the solution and achieving all constraints. More important, it is incorporated with self-x properties to enable the intrinsic evolution capability on run-time platforms. Visualization and intuitive visual programming offer the efficient interaction between the DAICOX and its users. Finally, the DAICOX architecture shall provide a fast and transparent design environment, that potentially yields high performance and adaptive multi-sensory measurement solutions.

This chapter is organized as follows. The next section describes the details of the DAICOX architecture. The following four sections (4.2, 4.3, 4.4, and 4.5) explain methods and algorithms used for the realization of the design building blocks in DAICOX. Section 4.6 explains quality assessment measures for the design evaluation. The details of model evaluation and selection mechanism are provided in Section 4.7. Section 4.8 gives the details of the optimization algorithms and techniques for multi-objective and dynamic environment optimization. Finally, the summary is given in Section 4.9 along with the recommended parameter settings of the presented methods.

4.1 Concept and System Architecture of DAICOX

DAICOX is a Computational Intelligence (CI) based systematic design architecture for IIMSS. The architecture inherits the concept and methodology from state of the art and our¹ previous research attempting to automate the design process of complex multi-sensory measurement systems. DAICOX stands in the overlap field of Computer-Aided Design (CAD) and Computer-Aided Engineering (CAE), where the first is the use of software in creation, modification and optimization and the latter is the assistance in engineering analysis tasks. CI and machine learning are the main background of DAICOX as means of automatically generating and optimizing IIMSS. One of ISE early works on the design automation has been carried out in the field of industrial machine vision systems [12]. The similar concept has been extended to general sensor applications in the work of Iswandy et al. [30] with multi-objective optimization and the introduction of self-x concept. Visual programming and choice of interactive visualizations in QuickCog provide fast and transparent automated design of general recognition systems. These are the baselines, that constitute functionality as well as features of the DAICOX architecture. In particular, the extension of flexibility and adaptability at hardware level with self-x properties is the strong focus point advancing the current tools and methodologies. Thus, the main objectives of the DAICOX are to:

- Provide rich information via interactive user interface and visualization along with intuitive visual programming and transparent design environment,
- Deliver high quality solutions outperforming human-manual based designs by using multi-objective optimization,
- Speed up the design process by rapid prototyping, and
- Gain the adaptability, reliability and robustness of designed solutions with self-x properties.

¹Institute of Integrated Sensor Systems

These are the properties and key features consolidated in the architecture of DAICOX system, which is elucidated in Fig. 33.

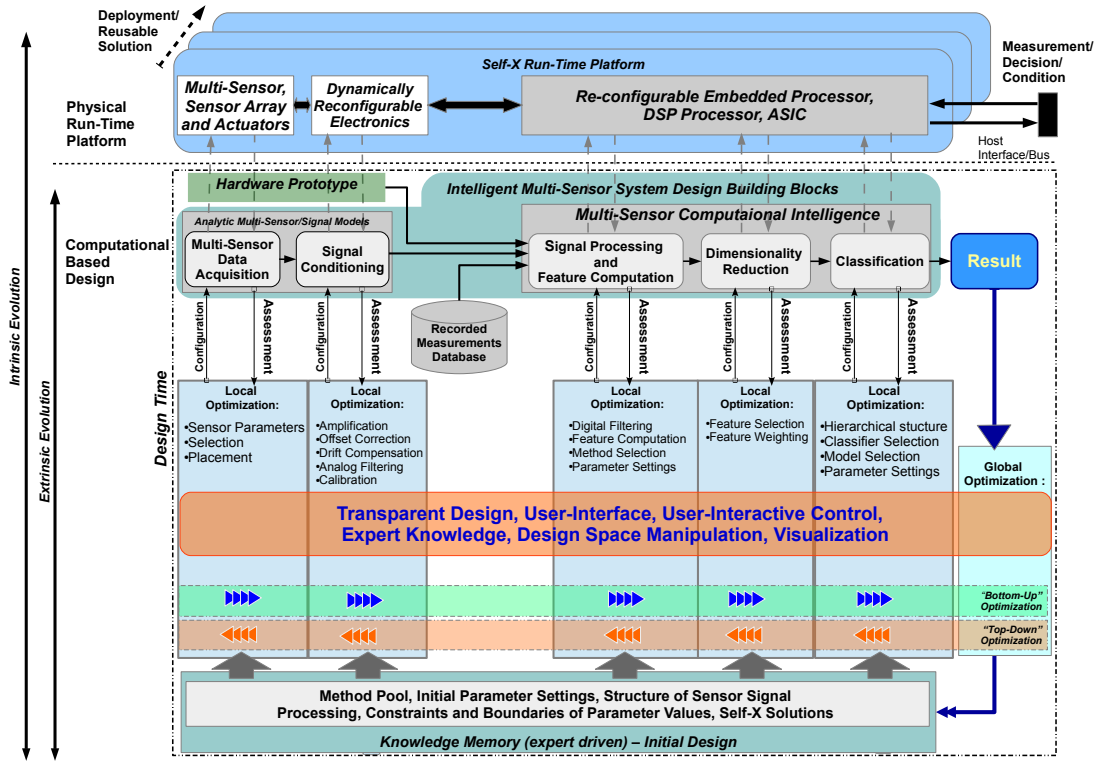


FIGURE 33: Automated intelligent multi-sensor system design architecture

Here, the brief discussion on the concept of DAICOX architecture is given before specific detail explanations in the following subsections. The blocks at the top of architecture are dynamically reconfigurable hardware platforms for supporting intrinsic optimization as well as self-x properties. The functionality of the platform is constructed from the DAICOX’s standard building blocks. The building blocks of IIMSS represent the processing components of multi-sensory cognition tasks.

Common design and optimization tasks comprise method selection from available methods and parameters tuning. In particular in sensory recognition system design optimization, three possible data sources can be used, including deterministic sensor models, that simulate the signal representation of the sensors, recorded measurements stored in the database, and live measurements by using multi-channel data acquisition modules.

The entire design process of DAICOX is performed in three layers as illustrated in Fig. 34. In the user-interaction layer, the designer is provided with user-friendly interaction to the design process, e.g, obtain ongoing design quality via visualization or create an initial design by using visual programming. This feature is useful especially in early stage of the design. The DAICOX design-time automation layer consists of the core procedures of the design automation process including advanced multi-objective optimization, standard

building blocks, method pools, and design database. For intrinsic optimization, a design solution can be transferred to the DAICOX run-time platform to evaluating the quality and performance at each evolution. Thus, a well-tailored design solution of an individual run-time platform is obtained. With self-x properties, the optimization tasks can be realized dedicatedly on the run-time platform, that realizes the concept of *machine-in-the-loop* evolution, which potentially gains reliability and robustness to its run-time operations.

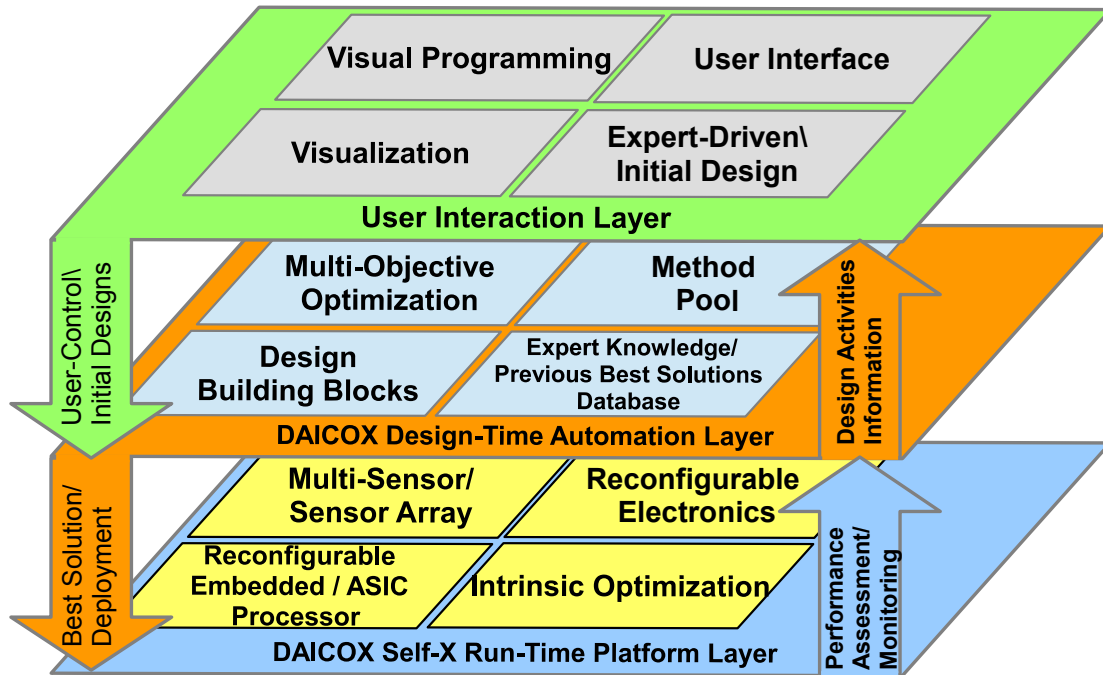


FIGURE 34: Three designing layers in DAICOX

4.1.1 Design Building Blocks

DAICOX constructs a processing chain of IIMSS based on the generic building blocks as illustrated in Fig. 35. Graph-based structures cooperating of multiple methods can be employed inside a particular block. Automatic method selection and parameter setting procedures are carried out by a meta-heuristic searching algorithm with multiple assessment criteria (multi-objective optimization). The method pool contains elaborated methods and functions in terms of specialized scientific software modules and libraries. Beside advanced methods available in the building block, DAICOX also provides seamless links between blocks to exchanging information without complicated reformatting. Adding or editing algorithms and methods can be achieved simply by accessing the method pool via DAICOX user interface. The precise details of the methods and algorithms as highlighted in Fig. 35 are given in Section 4.2 for sensor and sensor electronic

configuration, Section 4.3 for signal processing and feature computation, Section 4.4 for dimensionality reduction, and classification in Section 4.5.

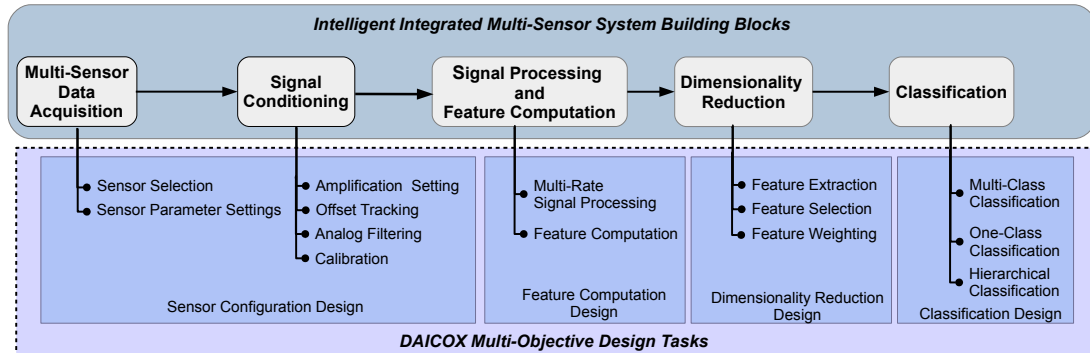


FIGURE 35: The generic building block model and corresponding tasks of IIMSS design in DAICOX

4.1.2 DAICOX Design Optimization

Design Time Automation (DTA) is the core operation of the DAICOX architecture. DAICOX optimization unit is the major part of DTA, that aims to reduce time and effort in designing IIMSS as well as to provide high quality and robustness solutions. The DAICOX optimization unit is elaborated with the key features as listed below.

- Method selection, parameter, and building block structural optimization.
- Multi-scale optimization scheme, e.g., *local* and *global* optimization.
- Inclusion of physical hardware devices or run-time platforms in optimization loop (*machine-in-the-loop* optimization)
- Multi-objective optimization
- Choice of flexible and expendable evaluation functions
- Supervised(monitored), semi-supervised, unsupervised optimization
- Adaptive optimization for lowering the effort in configuring optimizers
- Robust solutions, e.g, dynamic environment optimization

Global and Local Optimization: Due to high complexity of IIMSS design problems, the concurrent design of building blocks, i.e, global optimization, may require substantially huge computational resource, which is impractical in most scenarios. Thus, a distributed approach mimicking the *divide-and-conquer* paradigm, i.e., Local Optimization (LO) [12] is proposed. As a result, a global optimization problem is transformed

into a set of smaller design problems with reduced search space. LO recursively decomposes a design problem of the complete generic building block structure presented earlier into sub-problems of a particular block, i.e., local block. Therefore, local performance assessment is used for optimization process. At each block, performance assessment is given by information depending on functionality, for example, feature subset quality of the dimensionality reduction block. LO sequentially evolves a system by taking the best result from the neighborings as input, at every block with two choices of directions; *bottom-up* and *top-down*. Typically, the *bottom-up* scheme is common as a design usually starts from given desired sensory contexts, that leads to the selection of physical sensors/sensor array at first, whereas the *top-down*, for instance, starts a design with a specific classifier. The hybrid approach by including assessments from other building blocks, e.g., classification accuracy, where parameters or configurations of the particular designing block are only subject to change from the optimization procedure. Figure 36 displays an example of LO procedure at the signal conditioning block.

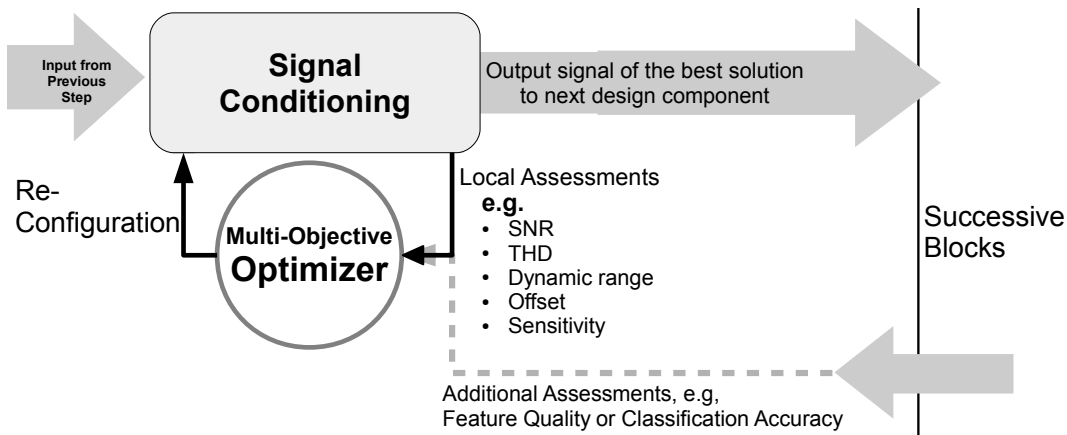


FIGURE 36: An example of local optimization at the signal conditioning design step

Intrinsic and Extrinsic Evolution: Typical design optimization problems are based on *extrinsic evolution* approach, i.e., evaluate design solutions by behavioral, deterministic, or computational models. These require precise analytical expressions, that are expected to include all possible variation factors in order to assure the stability system performance in operation time. In real-world situations, as shown in Fig. 37, static and dynamic deviations, drift phenomena and environmental perturbations occur inevitably after the deployment phase, that potentially deteriorate the system performance. For this reason, the inclusion of an actual hardware instance into the optimization loop, denoted as *intrinsic evolution* as shown on top of Fig. 33, is a promising means to cope with dynamic influences. Involving a real hardware instance into the evolution process allows the exploitation of parameter adaptation to compensate deficiencies of the extrinsic approach providing robust and time-independent performance. The realization of intrinsic evolution can be extended to self-x properties, i.e., self-monitoring/assessment

and self-optimization, as they share the same background of dynamically reconfigurable hardware system.

Commonly, optimization tasks rely on the *learning-from-examples* concept, one of many machine learning paradigms, which requires data or supervised information, i.e., class affiliation. In an intrinsic optimization, which *a priori* knowledge may be unavailable or no explicit goals or targets, these informations still can be made available during run-time by adding reference actors, e.g., heater or illumination. But, for some applications like gas sensing or liquid analysis, this approach may be infeasible or too expensive. A potential candidate for this issue is the unsupervised optimization approach, that performs local adaptation, a similar mechanism of auto encoding technique in neural networks.

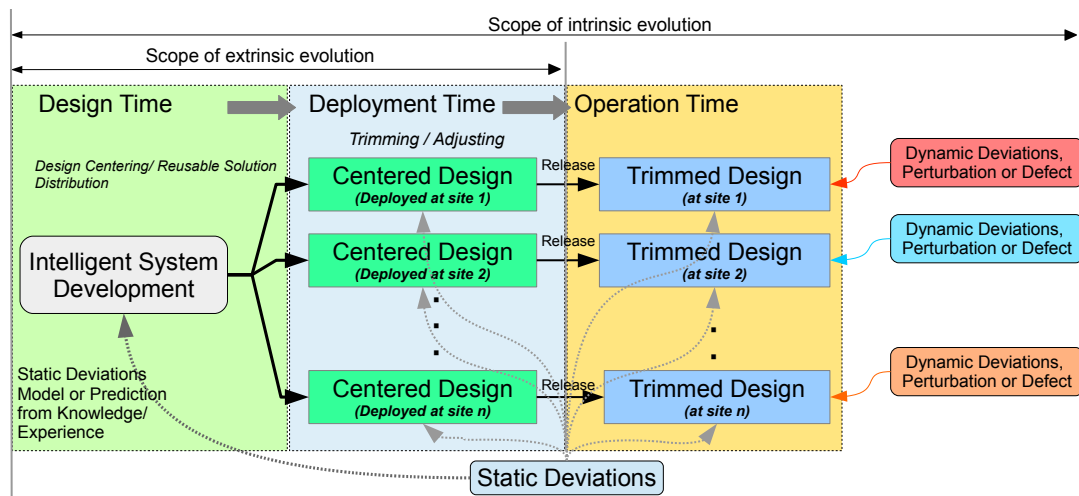


FIGURE 37: Three-Phase system design and deviations in each phase.

4.1.3 Self-X Properties of DAICOX Designed Solutions

Adaptiveness and flexibility are the essence in achieving longterm reliability, robustness and sustainability. Organic computing [19] is a recent research area consisting of autonomous and cooperating subsystems, that mimics organic mechanisms of living creatures on computing machines, i.e., self-x properties [118–120]. As design robustness is among the important design objectives of DAICOX, thus, the ability of embedding self-x properties into designed solutions is conceived. DAICOX is proposed with the capabilities, that encapsulate the properties of self-x to IIMSS as listed below:

- Self-Monitoring/Diagnose
- Self-Calibration/Trimming
- Self-Healing/Repairing

- Self-Optimization

The functionality of self-x properties are composed of dynamically reconfigurable and efficient adaption mechanisms. Thus, for IIMSS applications, the essential components are dynamically reconfigurable hardware, e.g., sensors electronics on run-time platforms. Several scales of implementation including, functional level via programmable analog ICs or digital sensors to microscopic level via integrated reconfigurable analog arrays.

In addition, most of the adaptive operations in self-x process are composed of several complex computation tasks. Hence, choices of high performance embedded processors, e.g, High-Speed Microcontrollers, Digital Signal Processors (DSPs) or Application Specific Integrated Circuits (ASICs) can be chosen based upon the functional complexity of property and application. For instance, self-optimization usually comes with arithmetical burden and large routines, that DSPs may be the good candidate in implementation. The realization of these properties, in particular to IIMSS, requires a cooperative structure of several complex components as depicted in Fig. 38.

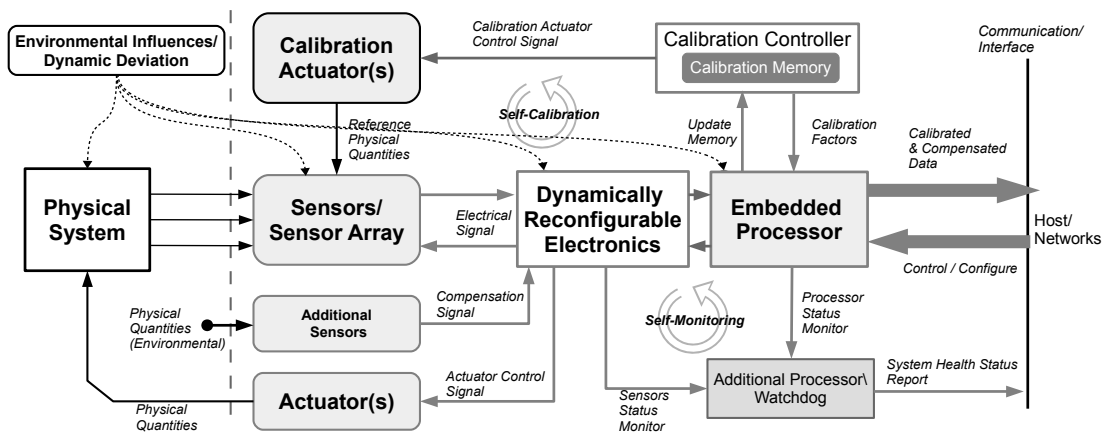


FIGURE 38: Architecture of self-x properties integration in DAICOX.

Proposed self-x properties are the potential aspects of providing rapid-prototyping, flexible and computational effective solutions with robustness and dependable characteristic at low cost, effort, and short design as well as maintenance time.

4.1.4 Design Interaction

Design interaction of the DAICOX architecture consists of four main parts as illustrated at the top layer of Fig. 34. The aim of this layer is to provide the designer with intimate links to the automated design activities in order to navigate and reinforce the ongoing design process. Features and functionality of DAICOX design interaction layer are explained as follows.

Visual Programming

The up to date trend in software development framework is graphical-based programming. This allows programming with visual expressions consisting of modular elements, which each represents a particular function. A processing flow or graph, usually as an initial design, can be simply created over drag-and-drop and signal connecting actions. This fashion provides an intuitive and fast track for system development as well as requires no expert skill in designing a system. The ability of returning information to preceding modules or blocks, i.e, feedback graph, would be a desirable feature, that allows advanced functionality over the system.

User Interface

DAICOX provides full access to all design steps via its intuitive User Interface (UI). Configurations and parameters of a design module can be observed and defined in its corresponding UI. In prototyping or debugging stage, operational characteristics and performance of a system can be observed and analyzed by the designer over real-time signals monitoring. Meaningful information can also be given to the designer in terms of the design guideline or wizard.

Interactive Visualization

Perceptive and associative capabilities of humans are powerful tools to solve complex problems like hidden knowledge extraction or non-obvious correlations identification. To exploit this, DAICOX provides supportive information using appropriate visual representations, e.g., feature space observation. However, most of the informations relating to the design of IIMSS are high-dimensional, thus, an effective feature transformation is needed to reduce the information into a human perceptible data space i.e., 2D or 3D. To do this, multivariate projection methods, e.g, Linear Discriminant Analysis (LDA) or Multi-Dimensional-Scaling (MDS) are choices to be incorporated with interactive visualization.

Design Space Exploration

In an optimization problem, the efficient procedure, that governs the search operator to focus in an attaining search region is a salient task to provide promising outcomes at reasonable time and resource consumption. Design Space Exploration (DSE) is the activity to let the designer explore design alternatives before engaging the full scale search operation with constrained or limited feasible search landscape. Choice of visualizations can be chosen in cooperation with the proposed visualization part. UI and visualization are also required in order to establish bidirectional interaction between the designer and DSE.

Expert Driven Design and Design Database

Starting a design from scratch for an IIMSS is laborious and time-consuming. In particular, due to lack of knowledge and experience designers may return deceptive solutions, which potentially are sub-optimal. The efficient way is to initialize the design with a design seed or information from the Design Database (DD). This approach results significantly compact search space, that potentially embrace the optimum. The information stored in DD are in two possible forms; the initial design given by the designer, i.e., *Expert Driven* and the a priori knowledge or experience of previous designs' best solutions. The latter case significantly reduces design effort in contrast to starting a design from scratch. Flexibility and extensibility of the design template are necessary in order to support changes in functionality or improvement.

4.2 Sensor Configuration

For a measurement system, setting relevant sensor parameters and configurations is an essential task to obtain high consistency between measuring physical properties and acquired sensory data. Flexibility is required in two domains involved for automating this process, i.e., the setting of parameters governing operational characteristics of physical sensors and the configuration of sensor electronics. Reconfigurable electronics, that enable dynamic adjustment on hardware level, play major role in realizing the automation process of sensor configuration.

4.2.1 Sensing Element Configuration

Possible physical sensor parameters, that can be reconfigured dynamically may account the selection of sensor elements to be operated or mechanical configurations, e.g, camera angle or focus position. In some applications, an action of controlling environmental properties at measurement time of a sensor using actuators can be a kind of sensor

configuration, for instance, heating or illumination. In contrast to conventional heating element, micro peltier is an advanced cooling/heating technology with high power density, e.g., thermo-harvester from Micropelt [121]. For magnetic measurement applications, using AMR sensor, e.g., AFF755 from Sensitec [122], in particular, offset caused from manufacturing tolerances can be canceled by driving current to an integrated metal layer to select sensitive orientation, i.e., flipping [59, 123]. Sensor selection can be implemented on software level by excluding the data of deselected sensors. But this approach is ineffective because excluded sensors still consume energy and measurement time and cost. An efficient choice is to use controllable analog switches, e.g., CMOS analog switches or multiplexers as well as MEM switches. The utilization of sensor electronics can also be improved when a single sensor front-end is connected with multiple sensors elements.

4.2.2 Sensor Electronics Configuration

This task adjusts sensor electronic parameters with the aim to effectively enriching sensor signals before delivering them to further processing components. Thus the adjustments are mainly related to sensor signal conditioning domain. Dynamically reconfigurable electronic devices are the essentials for enabling flexibility of the circuits. In *functional level*, numerous modern digital sensors and analog electronic devices provide access via digital interface to control or adjust their operational parameters such as gain, offset compensation, cut-off frequency, etc. This level provides flexibility only at some degree due to limited availability of adjustable functions on a particular device or sensor. Higher flexibility can be obtained in *programmable level*, which applies changes to internal analog circuit structures, e.g, transistors in Field Programmable Transistor Array (FPTA). However, the price tag for this solution is the lack of utilization due to redundant areas and switching components. The trade-off solution for the issue is the Field Programmable Medium-granular mixed signal Array (FPMA) [85], that provide considerably high flexibility with efficient resource requirement.

4.2.3 Automated Sensor Configuration

The operation of autonomous sensor configuration task consists of a hardware setting generation, a configuration assessment, and a search operator. The acquired measurements are used to evaluate the performance of present sensor settings in configuration assessment procedure. Several signal characteristics can be considered together in the evaluation and configuration searching, which is multi-objective optimization, for instance DC offset level, Signal-to-Noise Ratio (SNR), Total Harmonic Distortion (THD), and Dynamic Range. A search procedure is responsible for generating a set of sensor

settings and evaluate them to find the best configuration candidate. Meta-heuristic algorithms given in Section 4.8 can be employed. The processing steps of automated sensor configuration in DAICOX are illustrated in Fig. 39. Typically, these procedures perform on a dedicated design platform, i.e., PC or High-Performance embedded computer, with an communication interface to the sensor run-time platform in order to exchange optimization information, thus, the intrinsic evolution is already obtained. However, this approach is still vulnerable to dynamic deviations, whereas the sustainable approach combines all the procedures on the run-time platform, i.e., *machine-in-the-loop* [34].

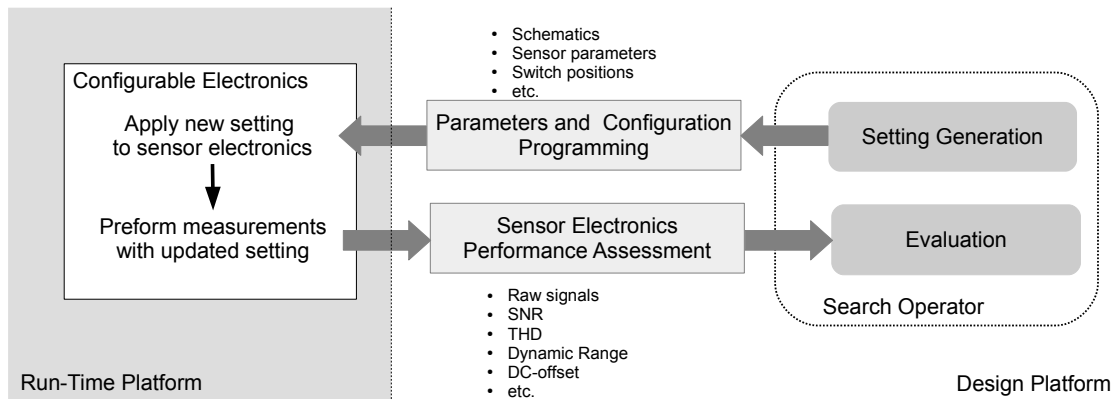


FIGURE 39: Operation of automated sensor configuration

4.3 Signal Processing and Feature Computation

Sensory data usually need some treatments at a very first stage of pattern recognition tasks such as noise removal, drift compensation, dynamic enhancement, shaping, and scaling. Signal Processing (SP)² helps condition sensor signals by signal to signal transformation and manipulation. SP methods predominately are data preserving, i.e., the feature vector of an output data is identical to its input one, whereas some methods create new information space extracted from input data, e.g., frequency, distribution, projection analysis, which are denoted as feature computation in this thesis. Feature computation extracts and condenses information by heuristic techniques using transformations.

The term Feature Computation (FC) is frequently interchangeable with *Feature Extraction* (FE) in several literatures. Two common aims of FC and FE are to create new information correlated to the original data and to compress the original data into a smaller dimensions space. Therefore, to avoid ambiguity between the terms, in this thesis we denote FC as the group of methods, that produce new information and denote FE as the group of methods for dimensionality reduction, which is explained in the next section.

In complex multi-sensor pattern recognition systems, both SP and FC are employed. Therefore, a joining process of two data spaces, i.e, concatenation, is needed to form a single data space being used for further processing steps. The overview of SP and FC processing flow is illustrated in Fig. 40. Several SP and FC methods [124] commonly used in multi-sensor related applications are summarized in Table 4.

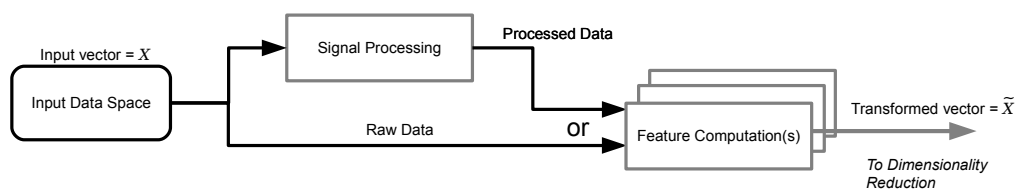


FIGURE 40: Operational flow of signal processing and feature computation

²In this thesis, all SP methods are of digital signal domain

TABLE 4: Description of basic signal processing methods [124]

Method	Description	Type
Arithmetic	Common arithmetical operations, usually used for signal reshaping including : addition, subtraction, multiplication, and division.	SP
Blending	A linear combination between signals commonly used for signal mixing, noise canceling, or compensating.	SP
Normalization	The conversion all sensor values to a common scale. For multi-sensor data, each feature is normalized individually.	SP
Digital Filtering	Remove unwanted signals from input data. Finite Impulse Response (FIR) and Infinite Impulse Response (IIR) filter are the common choices, where the first has better stability and linear response while the latter behave more closely to analog filter.	SP
Correlation	Generate the value of similarity between two signals, i.e., cross correlation. Auto correlation uses single signal by using temporal shift.	FC
Data Statistics	Basic statistical information of data, e.g., minimum, maximum, mean, median, standard deviation, and variance	FC
Statistical Moments	Extract properties of similarity in statistics or geometric commonly applied in computer vision applications, e.g., Moment Invariant, Gabor filters and wavelets, and Fractal Analysis	FC
Frequency Analysis	Generate frequency spectrum from time-domain data, e.g., Discrete-Fourier-Transformation (DFT). For non-stationary time-signals, Short-Time-Fourier-Transformation (STFT) or Wavelet Transform are generally used.	FC
Distribution Analysis	Similar concept to frequency analysis but by means of frequency of occurrence, e.g., Histogram, Density estimation, and Multi-Level Thresholding (MLT)	FC

4.4 Dimensionality Reduction

Due to the growing number of emerged sensors as well as their ability in providing multiple information, e.g., multi-spectrum color sensors, thus, the data of multi-sensor measurement systems are becoming more and more high dimensional. Although, a beneficial aspect of using multiple sensors is to improve recognition performance, however, some sensor information or features may be redundant and can deteriorate the performance in some cases. The term to address this paradigm is *curse of dimensionality* [125], which implies the augmentation of additional features over a certain size can cause degradation instead of improvement in performance. The assumption of curse of dimensionality effect is illustrated in Fig 41.

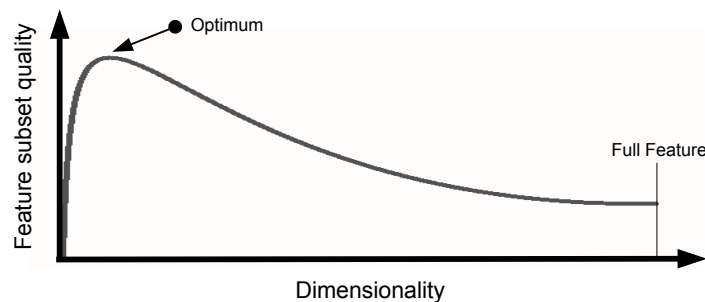


FIGURE 41: Illustration of curse of dimensionality paradigm

There are several reasons supporting the necessity to perform DR in a pattern recognition task as listed below.

- The reduction in bandwidth of the data. (improvement in measurement speed)
- The relevant set of features. (improvement in recognition performance)
- The modest set of feature. (improvement in economically computation)
- The visualization purpose. (two or three dimensions data representation)

In general pattern recognition literature, two approaches are defined in DR, Feature Selection (FS) and Feature Extraction (FE), based on their underlying mechanisms. FS constructs a subset by including or excluding features of the original input vector space without modification on information of the selected features., i.e, data preserving approach. FE, transforms all features in the original input vector space, usually by using a linear or non-linear combination, to generate new lower dimensional features. An intuitive principle comparison of the approaches is depicted in Fig. 42. The descriptions of these approaches are given in the following subsections.

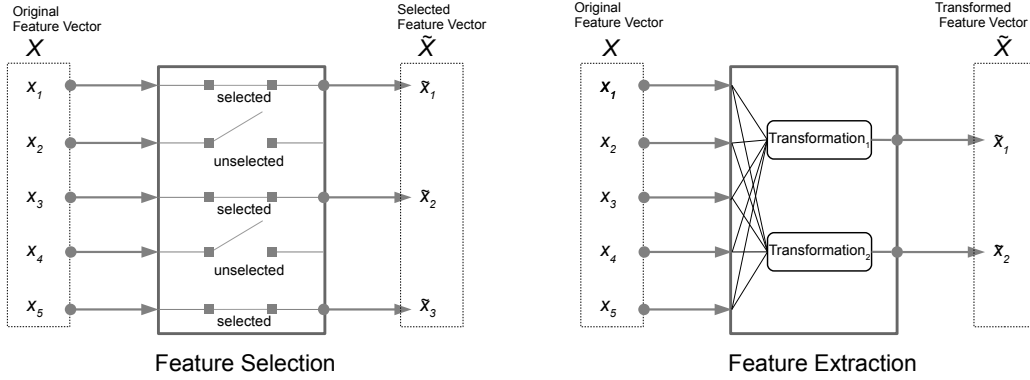


FIGURE 42: Basic processing principle of FS and FE

4.4.1 Feature Extraction

Feature Extraction (FE) finds a set of transformations for generating a new feature space, i.e., transformed space, in a lower dimension from the initial space. Thus, the aim is to replace the original data variables by a smaller set of variables, that carries as similar information as the original one. A linear or non-linear combination are typically used in transformations and can be either supervised or unsupervised. Maximum class separability is the criterion of finding the transformation in a supervised case, i.e., class labels are required. An unsupervised case uses underlying statistical characteristics, e.g, maximum variance, of the data themselves for judging the quality. The optimization problem of FE is performed focusing on the set of allowable transformations, \mathcal{A} , to find the best transformation \tilde{A} of an given input matrix \mathbf{X} , that satisfies

$$J(\tilde{A}) = \max_{A \in \mathcal{A}} J(A(\mathbf{X})) \quad (4.1)$$

where J is the criterion function [117, 126]. Therefore, the new feature vector, i.e, transformed vector, can be obtained from $Y = \tilde{A}(\mathbf{X})$.

Several approaches based on subspace projection methods are widely used such as Principal Component Analysis (PCA) [127] and Linear Discriminant Analysis (LDA) [127]. PCA attempts to find k projections axes, which are orthogonal to each other, of which providing maximum variance (principle axes), where $k < |\mathbf{X}|$. PCA is a signal-representation technique meaning that it does not use class label information, i.e, unsupervised.

LDA, in contrast, tries to find a k -dimensional subspace based on maximum class separability, which requires class label information (> 2 classes), i.e., supervised, thus, the maximum transformation allowance for LDA is limited at $k \leq C - 1$, where C is the number of classes in a given data.

These methods, however, are linear transformation, which basically may not be able to cope with data, that is not linearly separable. Kernel PCA can solve non-linear problems by mapping an initial feature vector into a higher dimensional space then perform the original PCA. Non-linear subspaces projection methods, e.g, Self-Organizing Maps (SOM) [127], or Multi-Dimensional Scaling (MDS) [128], are specifically designed to deal with non-linear data characteristic.

SOM constructs a k -neuron network, whose inputs are connected to all features on the initial data, where k is the target dimensions of mapped space. The training process in SOM adjusts the weight vector, that tries to preserve as much information as in the input vector, also called the *topology-preserving* approach.

MDS non-linearly maps high dimension data into a low-dimensional sub space (typically 2 or 3 dimensions), which brings it a popular DR for visualization purpose. The MDS process accompanied with a criterion with regard to the scaling dissimilarities, which is the stress function, that can be Sammon [129] or Kruskal's [130] stress.

Fractal Analysis (FA) [131] assessing fractal characteristics of patterns in two dimensions. It has been mainly used in image analysis applications for texture segmentation and feature extraction. An intuitive explanation of FA is that it tries to find a repeat pattern (fractal dimension) in different geometrical scales. The processing consists of central projection and wavelet transformation. The first attempts to transform an original pattern to one-dimension pattern, where the latter generates a sub-patterns, i.e., extracted features.

4.4.2 Feature Selection

Feature Selection (FS) performs a binary selection of every feature in an initial feature space to remove irrelevant and redundant features. FS has a significant advantage on its results that they can be directly transferred to the physical level, i.e., activating sensors in respect of the optimal feature subset. To autonomously perform an FS task, denoted as Automated Feature Selection (AFS), two important processes are involved; feature set construction, and feature set quality assessment. The general procedure of AFS takes a given original feature vector $X = \{x_i | i = 1, \dots, M\}$, $X \in \mathbf{X}$ of M features. The optimization problem of FS, as formulated in Eq. 4.2 [117, 126], searches for the best feature subset $\tilde{\mathbf{X}}$ from the set of all possible subset formations \mathcal{X} , where J is the assessment function with some criterion. The cardinality, $|\tilde{\mathbf{X}}| \leq d$, of the new feature vector is expected as $d \ll n$.

$$J(\tilde{\mathbf{X}}) = \max_{\mathbf{X}' \in \mathcal{X}} J(\mathbf{X}') \quad (4.2)$$

4.4.2.1 Optimal Methods

Optimal methods for feature selection, e.g., exhaustive search, guarantee the discovery of the best feature subset as it explores all members in \mathcal{X} . This method may be impractical for multi-sensor applications due to its computational burden with $O(2^n)$ complexity. Branch and Bound (B&B) [132] claims to deliver the optimal within the maximum bound at $\Theta(n^2)$ complexity, by neglecting the searching paths with less contribution to assessment criteria. However, the discovery of the optimal is certain only under the monotonic circumstance.

4.4.2.2 Heuristic Methods

Sub-Optimal or heuristic methods are the alternatives from optimal ones, that aim to reduce computational complexity of the search procedure. Sequential Search is one of the simplest and widely used method, that follows only one path of the complete search tree. Two variations are affiliated: Sequential Forward Search (SFS) and Sequential Backward Search (SBS) where the first starts from an empty feature set then adds a feature at every search step and the latter starts from the full initial feature vector then excludes a feature at every step. These methods are faster compare to B&B with $O(n^2)$ complexity, however, they suffer from being trapped in a nested search tree. By applying a stopping criterion, the searching time can be reduced, where $d < n$ is the number of features of at the stopping point.

It is clear that the reduction in computation comes with the price of taking risk of the nested feature subset problem. The solution mitigating this issue is to add a dynamic direction mechanism to the search procedure, i.e, the combination of forward and backward movement. This approach will give a chance for the search procedure to revisit an ongoing search path rather than constantly move toward one direction.

”**plus l -take away r ”**, also called l - r [133], performs successive augmentation and segregation process. Two parameters, l and r , control the number of features in the incremental and removal step respectively. The limitation of this method is the lack in theoretical explanation to determine the appropriate settings of l and r , i.e., often arbitrarily set as ad-hoc.

Sequential Floating [40] inherits the plus l - r method with no control parameters. The procedure consists of both SFS and SBS, where the beginning of the search can be performed by either of them. At each search step, both SFS and SBS are performed but the searching direction will maintain in one way as long as the assessment result is better than the other, otherwise reverse the searching direction. The performance of this method is considered as good as B&B with much less computational time [133]. It

is possible that search processes may face an endless cycle, i.e, forever loop, which can be prevented by applying book-keeping.

Oscillating Search [134] tries to optimize a feature subset of a desired cardinality d . The search starts with d dimensions subset, which can be initialized by using SFS/SBS or random generation, then performs oscillations. The oscillation procedure applies SFS to add features until reaching the upper bound $d + \Delta$ then perform SBS toward the lower bound $d - \Delta$, where Δ is a user defined parameter. The search will stop when the current feature size reaching back at d dimensions with an improvement in feature quality. Setting higher Δ value results in more thorough searches but costs longer time to complete the process. This method has been reported to overcome the "nesting problem" and outperform SFFS/SBFS [134].

4.4.2.3 Meta-Heuristic Methods

Sequential mechanism of heuristic methods is naive and greedy behavior, that may be prone to local optima. Even with several variations previously mentioned to alleviating the problem, considerably high computational complexity is the price tag.

Meta-heuristic approach is different to sequential method in the way that it uses stochastic mechanisms. The simplest way to explain this is that a set feature vectors are randomly generated by a search procedure. The main advantages of using meta-heuristic for feature selection are listed below:

- Invulnerable to the "*nested feature subsets*" problem.
- Does not require the monotonicity assumption.
- An optimum can be reached without completely exploring solution space.
- Ability to parallelize search procedure (population based in GA or PSO).
- Well-suited for multiple criteria assessment.

Evolutionary computation based algorithms have been widely adapted to the feature selection purpose, e.g., Simulated Annealing (SA) [135] or Genetic Algorithms (GA) [136]. SA is based on the annealing process of thermal systems and performs a stochastic search. While SA searches the optimal on a single solution, GA, a population based algorithm, performs search with multiple solutions concurrently.

Genetic Algorithm (GA) [136] comprises of three main components: the individual representation, individual modification, and assessment evaluation. More details of GA are given in Section 4.8.1.1. Each individual represents a n -bit binary pattern, where n is set to equal the initial feature set dimension. The evaluation process uses the binary

pattern to construct a feature subset before giving to an assessment function with respect to some criteria. Search procedures of GA consist of selection, reproduction, mutation, and crossover procedures, together modify the individuals at every iteration to find the best solution.

Particle Swarm Optimization [110] is a population based algorithm, which searches for an optimal by moving individuals over the search space. This means that each individual has a location, that is coordinated by a number of dimensions, which is equal to the initial features size. The individuals move towards two locations, own best (local) and population best (global) positions with regard to the feature subset quality. PSO for feature selection outperforms heuristic methods as well as its counterparts, e.g. GA, as reported in [110]. Concise details of PSO are given in Section 4.8.1.2.

Table 5 shows comparisons of the addressed feature selection methods. O refers to the tight computational complexity, Θ refers to the maximum computational complexity if the upper bound is unknown, and i is the maximum number of iterations.

TABLE 5: Comparison of feature selection methods

Method	Optimal	Computational Complexity	monotonicity independent	Required parameters
Exhaustive	Optimal	$O(2^n)$	yes	-
B&B	Optimal	$\Theta(2^n)$	no	-
SFS/SBS	Sub-Optimal	$O(n^2)$	no	-
l - r	Sub-Optimal	$O(n^2)$	no	l and r
SFFS/SBFS	Sub-Optimal	$\Theta(2^n)$	no	-
Oscillating	Near-Optimal	$\Theta(2^n)$	no	d and Δ
Meta-Heuristic	Near-Optimal	$O(p \times i)$	yes	Algorithm based (see 4.8)

4.4.2.4 Filter and Wrapper Evaluation Approaches

Feature subset assessment can be categorized into two approaches based on the classifier dependency of assessment function. *Filter approach* is independent from classifier, that evaluates a feature subset by intrinsic properties of the data. The computation in assessment functions based on filter approach uses direct information from the data such as distance, variance, dependency, and consistency. Several assessment methods and algorithms for the filter approach are given in Section 4.6. *Wrapper Approach* uses a predetermined learning algorithm to assess a feature subset. The proper implementation of the wrapper approach is relatively complex since it involves classifier training,

parameter tuning and validation. The wrapper approach usually achieves higher recognition performance due to the fact that the feature subset is used to model and tune the classifier. Computationally expensive and the risk of overfitting are the drawbacks of this approach. The filter approach is significantly the faster in searching and better in generalization performance, however it tends to selecting the full-feature set for the optimal solution, thus, a certain dimension cut-off should be given to avoid the aggressive growing of feature subset. Figure 43 and 44 summarize the general concepts of the filter and wrapper approaches respectively.

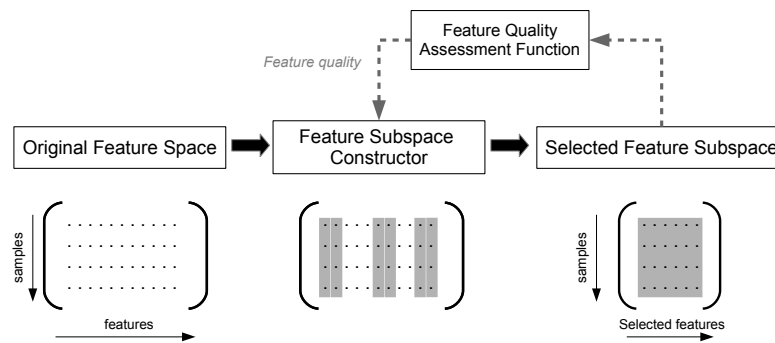


FIGURE 43: Filter approach feature selection

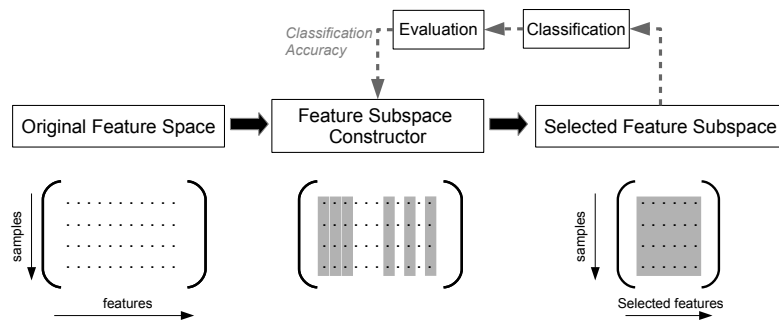


FIGURE 44: Wrapper approach feature selection

4.4.3 Feature Weighting

An advanced variant of feature selection is to replace the binary selection procedure by real value linear combination, i.e., Feature Weighting (FW) [41]. An FS problem can be described as a linear combination between an initial feature vector X and a selection vector A_S as formulated in Eq. 4.3. Where w_n is either 0 or 1 in FS case, while a real number interval $w_n \in [0, 1]$ is applied in the FW case. The construction of feature subset \tilde{X} excludes all elements with zero value or smaller than a defined threshold,

finally $|\tilde{X}| < |X|$.

$$\begin{matrix} X \\ \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \end{matrix} \cdot \begin{matrix} A_S \\ \begin{bmatrix} w_1 & 0 & 0 & 0 \\ 0 & w_2 & 0 & 0 \\ & \vdots & \vdots & \\ 0 & 0 & 0 & w_n \end{bmatrix} \end{matrix} = \begin{matrix} \tilde{X} \\ \begin{bmatrix} \tilde{x}_1 \\ \tilde{x}_2 \\ \vdots \\ \tilde{x}_n \end{bmatrix} \end{matrix} \quad (4.3)$$

Automated Feature Weighting (AFW) comprises as similar procedures as AFS, where the search problem is much larger since it has more choices to be selected for each feature. The search problem of AFW can be illustrated as a parameter tuning task by varying weights (parameters) within the interval. The step size controls the resolution of solutions, which of course exponentially increases computational complexity of the search problem.

Thus, this leads to the complexity of $O(l^n)$ where l is the number of weight steps. Parametric weight setting [137] can help realize AFW with computationally feasible, however, the performance depends on the model prediction and fine weight setting is difficult to achieve by this approach. Meta-Heuristic methods are the potential candidates to realize of AFW with good quality results under reasonable computational cost.

4.5 Classification

Classification is the problem of identifying which group (class label) the given observation (data vector) belongs to based on the pattern lies on the data, i.e., pattern recognition. Several effective classification algorithms and techniques categorized into three groups, i.e., multi-class classification, one-class, and hierarchical classification, are given in this section.

4.5.1 Multi-Class Classification

Multi-Class Classification (MCC) problem deals with the identification of an unknown pattern to a group of classes. A multi-class classifier judges a given M features data vector $X = \{x_m | m = 1, \dots, M\}$, also denoted as *unknown pattern*, to a member of an L classes class label vector $\Omega = \{\omega_1, \omega_l, \dots, \omega_L\}$. The vast majority of classification methods are multi-class classification. While some methods are able to directly handle problems with more than two classes, e.g, k -NN, many can deal only with two-class problems, i.e. binary classifications. To use a binary classification method for a multi-class (more than three classes) problem, a mechanism, that decompose a multi-class problem into a number of binary classification problems is required.

One-Against-One (OAO) and *One-Against-All* (OAA) are commonly used approaches for constructing a structure of multiple binary classifiers in MCC problems. The OAA approach creates a binary classifier for each class, which is trained to distinguish the patterns of the corresponding class from the patterns of all other classes. Instead of a symbolic or logical result, each classifier outputs the probability of a given pattern to its corresponding class, i.e., decision or confident value. For a given unknown pattern, the binary classifiers compute decision values for the final decision, that justifies and outputs the final class value associated with the maximum probable classifier.

OAO, also known as *pairwise coupling*, constructs a binary classifier for each pair of classes. This means that a problem with L classes, the total number of $L(L - 1)/2$ classifiers will be trained to separate the patterns of one class to another. A voting process of all classifier results decides, which class a given unknown pattern belongs. The illustration of the classification structures of both OAO and OAA approach are given in Fig. 45. OAA requires $O(N)$ classifiers while OAO, instead, requires $O(N^2)$, however, in problems with high number of data instances and if the training time increases non-linearly respect to the number of data vectors, then OAO is the faster choice. In the following, widely used classification algorithms are explained.

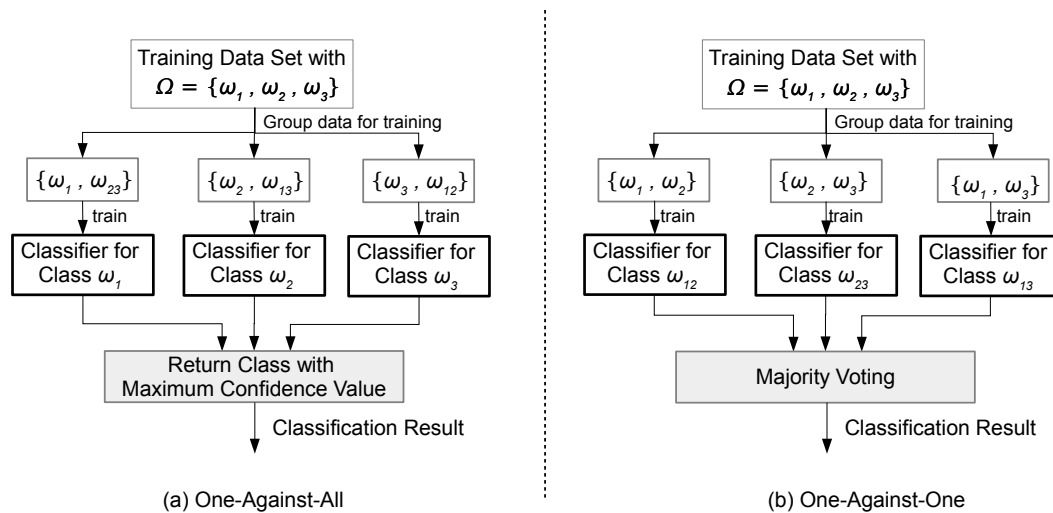


FIGURE 45: Classification structure of OAA and OAO approaches

Nearest Neighbor Methods

(k -NN) is considered among the classic non-parametric classification algorithms and is a multi-class classification algorithm by nature. To classify an unknown pattern, the distance (e.g. Euclidean) from that pattern to each of prototype example (from training data) is measured. The group of k nearest neighbors are identified, then classification can be performed by using either the voting approach, that makes a decision based on the frequency of class occurrence, or, the volumetric approach, which is based on

actual distance weighting of each neighbors contribution. The value of k can be either user defined or automatically determined using a dedicated data set and a validation technique [138]. Some variations of k -NN attempt to reduced the number of prototypes to increase generalization performance and reduce computational load. Reduced Nearest Neighbor (RNN) [139] is proposed by extending the early reference vector reduction mechanism, i.e., Condensed Nearest Neighbor (CNN) [140], to tackle the drawback of CNN that it may select some vectors that are far from the decision boundary. RNN incorporates a post processing step to revise selected reference vectors by heuristically removing a vector and if no miss-classification resulted from the removal then the vector will be deleted. This mechanism results in slightly smaller subset of selected reference vectors compare to one obtained by CNN.

Support Vector Machine

Support Vector Machines (SVMs) [141] are among the most robust and successful classification algorithms. SVMs, inherent binary classifiers, provide a good generalization performance and independent of the distributions of the patterns. A principle of the basic linear version of SVMs tries to maximize the margin of a linear hyperplane, which separates two classes data vectors. The *soft-margin* [141] concept is introduced for non-separable data vectors by penalizing some error points with the slack variable. The sum of the slack variables is associated with the parameter C of the optimum hyperplane determination process, i.e., establishing support vectors. For large C the higher tolerance of error points is admitted, whereas the lower C tends to minimize the margin. The appropriate setting range of the parameter C is [1,10000] [142].

SVMs can be extended to deal with non-linear problems by employing the feature space transformation technique, i.e., kernel trick [143]. Kernel function $K(\mathbf{x}_i, \mathbf{x})$ transforms an original feature space into a higher dimension space where a linear hyperplane can be applied. Choice of kernel functions are given in Table 6. The Gaussian Radial Basis Function (RBF) kernel is one the most effective and frequently used approaches. The parameter γ should be set within [0.00001,10] range [142].

TABLE 6: Support vector machine kernel functions

Function	Transformation $K(\mathbf{x}_i, \mathbf{x})$
Linear	$\mathbf{x}_i \mathbf{x}$
Polynomial	$(\mathbf{x}'_i \mathbf{x} + 1)^d$
Gaussian RBF	$\exp(-\gamma \mathbf{x}'_i - \mathbf{x})$
Exponential RBF	$\exp(-\gamma(\mathbf{x}'_i - \mathbf{x})^2)$
Sigmoid	$\tanh(a\mathbf{x}'_i \mathbf{x} - \delta)$

4.5.2 One-Class Classification

The Multi-Class Classification (MCC) paradigm aims to classify an unknown pattern into one of pre-defined classes (minimum two). Using a multi-class classifier with unknown patterns which do not belong to any of those classes will return misleading results.

One-Class Classification (OCC) is an approach to create a classifier, that accepts the patterns of target (positive) class and rejects others (negative or outliers). To create an OCC classifier, the aim is to build a model, that maximizes the correction in acceptance of positive patterns (True Positive) while minimizing the chance of accepting the negative patterns (False Positive) and vice versa for the rejection mechanism. The challenge in designing OCC classifiers is that in the case of only data of positive target are available.

The motivation for OCC is in broadening scenarios, e.g., monitoring faults in a machine, where only the information of the normal condition is available. Another interesting capability of OCC is the extension for novelty detection or anomaly detection, which can identify unknown patterns, that do not exist in the training data set or database. For these reasons, a number of methods and algorithms have been proposed to solve the OCC problem, which are explained in the following.

Nearest Neighbor Based OCC

Nearest Neighbor Description (NN-D) was proposed in [144] by applying the local density computation. An unknown pattern z is accepted to the positive class if the local density to its nearest prototype NN^{tr} is less than the local density of the NN^{tr} to its nearest prototype NN_k^{tr} (see Fig. 46 where $k = 1$). The threshold value ρ can be varied (typically set at 1.0) to adjust the sensitivity of the detection. For high dimensional data, this method should be performed with a mapped subspace [144]. Due to classification model of this method relies on the individual position of the prototypes, high number of training samples may be required to avoid the overfitting problem.

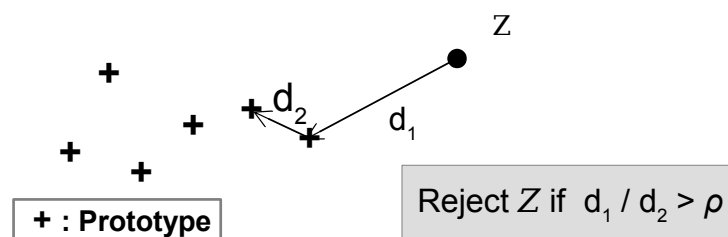


FIGURE 46: Rejection mechanism of the nearest neighbor description

One-Class Support Vector Machines

Support Vector Machine has been adapted to solve OCC problems in [145] by creating a hyper-sphere covering the positive class data vectors in a mapped feature space, i.e., Support Vector Data Description (SVDD). SVDD is suggested that using the Gaussian

RBF kernel will produce well fit descriptions, but higher number of data vectors is required to support the flexibility of the boundary. Instead of using a hyper-sphere, in [146], a method, that uses a hyper-plane as in the original SVM was proposed. This approach is based on ν -support SVM (ν -SVM) [147], which is the variation from the original C -support SVM. The modification of ν -SVM for one-class problems exploits the parameter ν to control the boundary of the outliers fraction (upper bound) and the support vectors of the target class fraction (lower bound). These bounds are the margin of the separating hyperplane, which any pattern drops in this area is classified as outliers. The setting of ν is in the range of [0,1], where it is suggested to be set at 0.05 [146]. Both methods have proposed techniques for generating artificial negative data vectors that help improve performance of the model when outliers data are unavailable. Several SVM based OCC techniques investigated in [148] have reported that when the Gaussian RBF is used the performance of the classifier is highly sensitive to the kernel width setting (γ).

Novelty Classifier

Novelty Classifier (NOVCLASS) [149] is a novelty detection method taking account the concept of Background Classification (BC) and OCC. The classifier model is based on hyperspherical classification method, which is a special case of the Restricted Coulomb Energy (RCE) [150] network. The classification model of NOVCLASS represents a network of prototypes t_j constructed from an N samples training data set, $\mathbf{t} = \{t_j \mid j = 1, \dots, N\}$. In the feature space, each prototype forms a hypersphere that locates at its center t_j with radius R_j .

NOVCLASS incorporates the BC concept by assigning a given feature space (of all classes) into the target class, i.e., *selected background*, whereas the area outside the feature space is defined as rejection region. An unknown pattern x activates a hypersphere j if $S(\|x - t_j\| < R_j)$ indicating that the pattern is inside the target class region, thus, classified as positive or *Normal* pattern. Any patterns that are outside the target class region are classified as *Anomaly* or *Novel*. Using the hypersphere approach in BC and OCC requires an additional determination process of hyperspheres' radii due to unavailability of negative class data samples. In NOVCLASS, all hyperspheres are constructed with uniform radii, $R_j = R_{max} \forall j$, which is determined according to the maximum 1-NN distance of every instance in the training data (see. Eq. 4.4).

$$R_{max} = \max_{j=1}^N \left(\min_{i=1, i \neq j}^N \|x_i - x_j\| \right) \quad (4.4)$$

Thus, the novelty classification process of an unknown pattern x is computed as the following steps:

1. Search the nearest prototype t_{NN} of x by:

$$\text{find } t_{NN} \text{ from : } d_{t_{NN}} = \min_{j=1}^N d_{t_j} \quad (4.5)$$

where $d_{t_j} = \|x - t_j\|$

2. Then classify the pattern x by:

$$y = f(x) = \begin{cases} \text{normal for : } \|x - t_{NN}\| < R_{max} \\ \text{novel for : } \|x - t_{NN}\| \geq R_{max} \end{cases} \quad (4.6)$$

In the training phase, a scaling factor η can be applied to $\eta \times R_{max}$ to control the denseness of the classifier model. In case the prototype selection is applied, a large η produces a coarse model with a small number of prototypes, whereas $\eta < 1$ produces a fine and compact normal class region with larger number of prototypes.

4.5.3 Hierarchical Classification

Conventional classification approaches process input data from a complete feature space (after pre-processing and DR). In a multi-sensor system, especially heterogeneous one, it may be the case that the classification flexibility on a particular sensor channel is required. For example, different sensor channels may influence the classification performance differently, e.g., different in signal precision or robustness, thus, a local optimization should be performed with regard to a particular sensor.

Hierarchical Classification (HC) is proposed to tackle the issue by adapting the ensemble classification approach. The HC consists of structured multiple classifiers in multi-level scheme and a hierarchical decision making process, thus, multi-sensor fusion at the decision level is performed here. The similar approach have already presented using Probabilistic Neural Network (PNN) [34] and Support Vector Machine [151]. Each classifier is dedicated to a feature subspace associated to a particular sensor channel or a user defined subspace. Thus, the optimizations problems are solved locally by determining optimum parameters, that fit to a particular subspace. The proposed prototype of the HC is heuristically based on multi-class SVM as the first stage of study. The concept of the proposed HC using the SVM (H-SVM) approach is elucidated in Fig. 47.

Assuming a given data obtained from S sensors, at the first processing stage, a data splitter separates the complete feature space into S subspaces. Thus, S classifiers are created and trained for each subspace in the first level classification. The output from those classifiers are vectors containing class probability estimates, denoted as p_{class} . Consider the One-Against-One approach solving an L -class problem at each subspace, the total number of binary SVM classifiers generated by H-SVM is $k =$

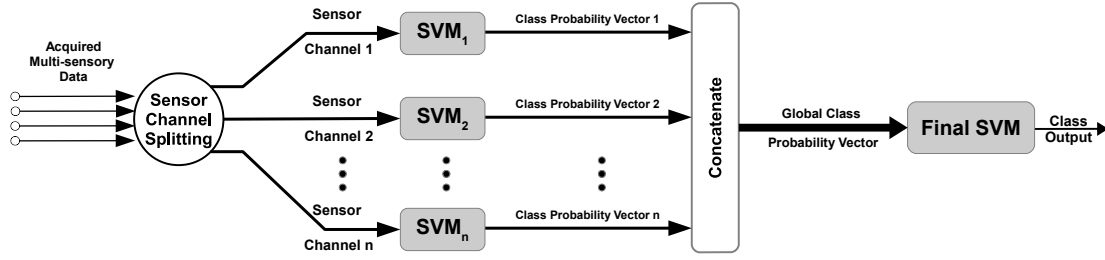


FIGURE 47: Proposed hierarchical SVM classification

$L(L - 1)/2$, which each is used to compute pairwise class probability estimates for $p_{class} = \{p_{ij} | j, i = 1, \dots, L \text{ and } j \neq i\}$. Therefore, a set of p_{class} vectors are created in the first level classification. The estimation of pairwise class probabilities for an L -class problem of a given pattern X from i th and j th classes is formulated as follows.

Given data with L classes, to classify an unknown pattern X , a classification estimates:

$$p_i \approx P(y = i | X), \quad i = 1, \dots, L. \quad (4.7)$$

Considering one-against-one approach, pairwise class probabilities r_{ij} can be estimated as:

$$r_{ij} \approx P(y = i | y = i \text{ or } j, X) \quad (4.8)$$

$$\text{where } r_{ij} \approx \frac{1}{1 + e^{A\hat{f} + B}},$$

\hat{f} is the decision value of trained SVM classifier model, and the estimates A and B are obtained by minimizing the negative log-likelihood of the training data [152]. In the training procedure, H-SVM figures \hat{f} , A and B and performs the optimization of the SVM parameter C and the RBF Gaussian kernel γ to a particular sensory channel. A concatenation operator joins all p_{class} vectors to form a global class probability vector p_{global} . The global decision making is computed from an SVM classifier at the final level classification, which is trained by using the p_{global} as input. At this level of the classification hierarchy, the final level classifier can also serve as an information fusion node at the decision level. A discrete classification result, i.e., class label, is given at this level.

4.6 Quality Assessment Measures

Designing a pattern recognition system one may focus on maximizing classification rate alone, but this poses the requirement on relevant model and parameters settings as well as expensive computation. Several processing components can be designed by assessing

some criteria other than classification accuracy. For example, up to the design building block of signal processing and feature computation, some signal related criteria, e.g., MSE or SNR, can be used. In particular to the designing of dimensionality reduction methods, numerous assessments, that analyze underlying statistical properties of data, e.g., nonparametric compactness, and overlap of class region, are good in generalization and are computationally cheap relative to using classification models. Regarding these reasons, robust measures with a few or entirely free of parameters, model assumptions, and intricate training requirements are presented in this section.

4.6.1 Density Based Methods

A number of information measures rely on data distribution characteristic by using probabilistic modelings. The computation of these methods generally involves the estimation of class conditional probability density functions (pdf's) by using an available training set under assumptions of normal distribution. Joint Entropy method, which measures the uncertainty associated with a set of classes by determining the overlapping region of distributions. Extended from the entropy concept, Mutual Information (MI) [153] takes into account both individual and joint distributions. The use of MI for feature quality assessment is in a pairwise scheme, that determines the mutual uncertainty of between classes with respect to the certainty of a particular class. MI accounts for high-order statistics, however, it involves numerical integration of complex functions, which leads to expensive computation especially when the number of dimensions is large. In particular, in pattern recognition applications, a pdf is typically estimated from the histogram of a data set, which requires a relevant setting of the bin size as well as a appropriate estimation method in order to achieve an appropriate pdf that describes the true distribution of the data.

4.6.2 Distance Based Methods

In contrast to obtaining distribution functions, distance metric based methods assess feature space quality by using dissimilarity between patterns, i.e., *distance*, directly from vectors of measurement. These approaches also have advantage over classifier based approaches as they do not depend on the model of decision making and have less or no parameters to set. These methods comprise of a distance measure function d , i.e., *metric*, between two pattern vectors $X_j, X_i \in \mathbf{X}$ of $|\mathbf{X}|$ dimensions resulting in a

distance value, which satisfies the following conditions.

$$\begin{aligned}
 d(X_i, X_j) &\geq 0 \\
 d(X_i, X_j) &= 0 \text{ iff } X_i = X_j \\
 d(X_i, X_j) &= d(X_j, X_i) \\
 d(X_i, X_j) &\leq d(X_i, X_k), d(X_j, X_k)
 \end{aligned} \tag{4.9}$$

Table 7 lists a number of the common distance measures and their formulations [133]. The choice of a particular metric usually depends on application. The consideration may take into account of the number of features (dimension), the requirement of shape of the contour around the class mean (rotational equality), and cost of computation.

TABLE 7: Dissimilarity measures for two data vectors X_j and X_i where $x_{i,m} \in X_i$ and $x_{j,m} \in X_j$

Metric	Denoted as	Formulation
Euclidean	d_e	$\ X_j - X_i\ _e = \sqrt{\sum_{m=1}^{ X } (x_{i,m} - x_{j,m})^2}$
City-Block	d_{CB}	$\ X_j - X_i\ _{CB} = \sum_{m=1}^{ X } x_{i,m} - x_{j,m} $
Canberra	d_{Ca}	$\ X_j - X_i\ _{Cr} = \sum_{m=1}^{ X } \frac{ x_{i,m} - x_{j,m} }{ x_{i,m} + x_{j,m} }$
Chebyshev	d_C	$\ X_j - X_i\ _C = \max_{m=1}^{ X } (x_{i,m} - x_{j,m})$

Euclidean metric d_e measures distance in the most natural manner among others. The square root of the total sum in d_e can be neglected (denoted as squared euclidean) without losing the monotonicity on the dissimilarity function but it will not anymore be a metric. City-block metric d_{CB} , also called Manhattan, costs a little cheaper computation than the d_e , that can be a choice if the speed is important. Canberra metric d_{Ca} , which is the weighted version of d_{CB} , is suitable for calculations taking non-negative and bounded values, however, a zero values in any vectors requires special attention in implementation. Chebyshev d_C is the cheapest and is often used in cases where the execution speed is so critical.

By taking into account the distances of dissimilarity, several nonparametric feature space quality measures have been proposed for a particular analysis on characteristics of the feature space scatter in multi dimensions, e.g., class region overlapping or constriction.

Class Overlapping Region

Based on the k -NN principle, class overlapping region measure q_{OV} [154] is a numeric nonparametric quality criterion measuring between classes overlapping areas in feature space. The concept is motivated by the idea of nonparametric distance matrix and the edited nearest neighbor [155] approach. q_{OV} indicates the degree of discriminant between classes in feature space, where the highest quality (1.0) refers to the nonexistence of overlapping class regions within the range of k neighbors from all data instances. The increasing number of different class neighbors at an instances, i.e., overlapping of class regions as depicted in Fig. 48, proportionally decreases the quality towards zero. Given an N instances data set affiliated to a L -class label vector $\Omega = \{\omega_1, \omega_2, \dots, \omega_N\}$ where $\forall \omega \in \{l_c : l_c = 1, 2, \dots, L\}$, the calculation of q_{OV} is formulated in Eq. 4.10.

$$q_{OV} = \frac{1}{L} \sum_{c=1}^L \frac{1}{N_c} \sum_{j=1}^{N_c} \frac{\sum_{i=1}^k q_{NN_i} + \sum_{i=1}^k n_i}{2 \sum_{i=1}^k n_i} \quad (4.10)$$

where

$$n_i = 1 - \frac{d_{NN_{ji}}}{d_{NN_{jk}}} \quad (4.11)$$

and

$$q_{NN_i} = \begin{cases} n_i : \omega_i = \omega_j \\ -n_i : \omega_i \neq \omega_j \end{cases} \quad (4.12)$$

n_i is the weighting factor for i th nearest neighbor NN_{ji} , $d_{NN_{ji}}$ is the distance from the instance i to NN_{ji} and in the same hold for $d_{NN_{jk}}$ where NN_{jk} is the farthest distance neighbor within k members. The distance calculation can be one of the approaches given in Table 7 but generally euclidean distance is employed. The top level summation can be omitted if the number of instances affiliated to each class are equal. Only one parameter, k , needs to be defined by a user, which typically is set at 5 to 10 for computational reason as suggested in [154]. Automatic determination of k can be made straightforward by taking the number of instances per class but should not be exceeded the suggestion. q_{OV} provides a very fine-grained value range and and, thus, is well suited for optimization schemes.

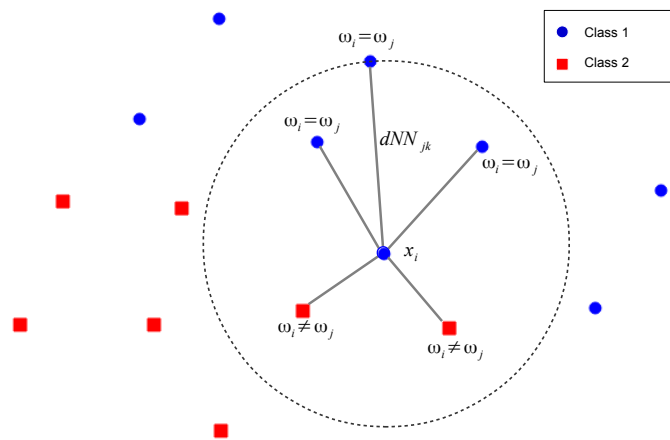


FIGURE 48: Illustration of overlapping measure computation with $k = 5$ at x_i instance of two-class data

Compactness Measure

Inspired by linear and non-linear discriminant analysis, nonparametric compactness measure q_c [107] is proposed to measure the quality of within class (*intra-class*) compactness and between classes (*inter-class*) separability in a feature space as illustrated in Fig. 49. Thus, in q_c , two numeric measures are part of the calculation; $q_{C-intra}$ and $q_{C-inter}$. The necessity of q_c may arise in the case that q_{OV} reaches the maximum quality (no overlap) yet improvements can be further perceived in terms of intra-class regions contraction or inter-class regions repulsion. In such case, using $q_{C-intra}$ and $q_{C-inter}$ as complementary measures delivers feature subsets with highest possible quality. Due to the fact that q_c performs the concurrent determination of $q_{C-intra}$ and $q_{C-inter}$, thus, an optimization using q_c is already multi-objective.

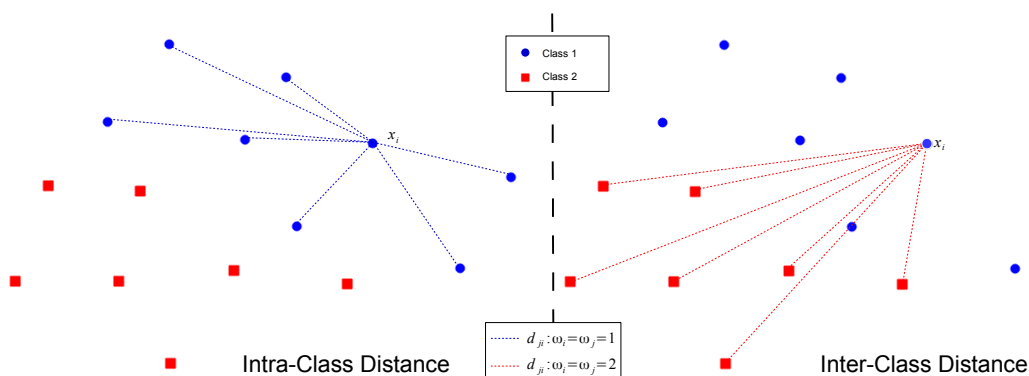


FIGURE 49: Illustration of intra-class distance and inter-class distance at instance i

The calculations of $q_{C-intra}$, $q_{C-inter}$, and q_c are describes as:

$$q_{C-intra} = \frac{1}{L} \sum_{c=1}^L \frac{2}{N_c(N_c - 1)} \sum_{i=1}^{N-1} \sum_{j=i+1}^N \delta(\omega_i, \omega_j) \delta(\omega_i, \omega_c) d_{i,j} , \quad (4.13)$$

$$q_{C-inter} = \frac{1}{N^B} \sum_{i=1}^{N-1} \sum_{j=i+1}^N (1 - \delta(\omega_i, \omega_j)) d_{i,j} , \quad (4.14)$$

and

$$q_c = w(1 - q_{C-intra}) + (1 - w)q_{C-inter} , \quad (4.15)$$

where

$$N^B = \sum_{i=1}^{N-1} \sum_{j=i+1}^N 1 - \delta(\omega_i, \omega_j) ,$$

and

$$\delta(\omega_i, \omega_j) = \begin{cases} 1 & : \omega_i = \omega_j \\ 0 & : \omega_i \neq \omega_j \end{cases} .$$

The numerical values of $q_{C-intra}$ and $q_{C-inter}$ are unbounded in contrast to q_{OV} , therefore an additional normalization step for each selection or configuration should be performed. The final combination for q_c , which is similar to the agglomerative multi-objective optimization approach, also requires a user defined weighting factor w , typically is 0.5.

Separability Measure

Separability measure q_S [107] exploits the RNN-classifier to obtain the selected reference vectors T_{RNN} from the iterative training procedure [139] which removes the redundant reference vectors from the original classification model (k -NN). The number of selected prototype depends on the separability of the feature space. q_S is determined by

$$q_S = \frac{1}{L} \sum_{i=1}^L \frac{N_i - (T_{RNN_i} - 1)}{N_i} , \quad (4.16)$$

where T_{RNN_i} is the number of selected reference vectors, N_i denotes the number of patterns affiliated to class ω_i , and L is total number of classes. In the linearly separable feature space, which means only one reference vector is selected for each class, q_S returns its optimum value (1.0). The computation of q_S is $O(N)$ complexity, the fastest compared to previously presented measures. However, the coarse resolution is the drawback.

4.7 Model Analysis

The generalization capability of a pattern recognition method relates to its performance on independent data, i.e., test data. Model performance analysis is essentially important as it navigates the choice of methods, models, and parameters to find a finally chosen model by some measure criteria. Choosing the optimum complexity of any chosen model depends on the quantity and the quality of the data creating the model, i.e, training data. The well-known paradigm is the *overfitting* problem, which refers to a very complex model, that performs well or best on the training data but, when coping with independent data the performance will be deteriorated. However, reducing complexity too much may result in a model that inadequately describes structure in the data. This dilemma requires a strategic procedure for model assessment, that balances between the goodness of fit and the capability of maintaining performance on independent data.

In this section, a number of methods aiming in maximizing generalization performance of a pattern recognition model are explained. In addition, interesting issues relating to model quality in terms of reliability, robustness and stability also given in this section.

4.7.1 Data Set Separation

The common and appropriate approach for attaining both fitness and generalization performance is to randomly divide an initial data set \mathbf{X} into three parts: a training set, a validation set, and a test set. The training set \mathbf{X}_{train} is used to generate or fit the models; the validation set $\mathbf{X}_{validation}$ is used to evaluate the models for model selection; and finally the test set \mathbf{X}_{test} is used to assess the generalization performance of the final selected model. These procedures are elucidated in Fig. 50.

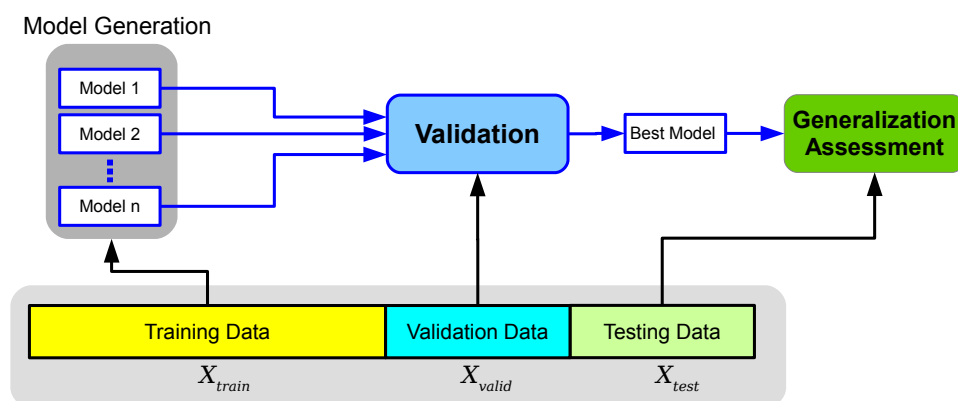


FIGURE 50: Model generation and evaluation process with three-group data separation

A systematic data separation method is required here in order to appropriately divide data into several sub groups. In particular, the mechanism, that constructs a number of

variations of training data and validation data plays the vital role in achieving the goal of generalization performance. Thus, several methods have been established to serve these reasons as explained in Table 8 [31].

TABLE 8: Description of data set separation methods

Method	Description
<i>k</i> -Fold Cross Validation (<i>k</i> -CV)	CV divides the available instances into <i>k</i> disjoint subsets (folds). The model selection runs <i>k</i> iterations of the <i>i</i> th fold validation of a model generated from the remainder folds. The setting of <i>k</i> is generally recommended between 5 to 10.
Holdout	An initial data set is randomly separated into two folds with size proportion to a ratio r_H , where $r_H = 0.5$ is identical to 2-fold CV. If the available data size is large, the holdout can provide adequately good results with cheapest computation cost. Random sampling is the common technique for selecting data instances to each fold.
Leave-One-Out Cross Validation (LOOCV)	The same hold as in the <i>k</i> -CV approach, where <i>k</i> is equal the number of data instances <i>N</i> . Each iteration, a model generated from <i>N</i> - 1 instances is validated by the left over instance. LOOCV is said to provide low biased models, but contains a large variance [31]. The computational effort is also considerably high due to a large number of observations as well as the largest data for model generation among others.
Bootstrap	Create a number of bootstrap subsets of size <i>N</i> by sampling N_s from \mathbf{X} with N_r replacements, where $N_r + N_s = N$. Bootstrap can provide lower variance compare to LOOCV while having high number of possible subsets, which is useful for a small size data set.

It has been broadly noticed that, the model selection *k*-fold cross validation could lead to choose a model with low chance of over-fitting. The optimum *k* setting is still an open issue to date, whereas the commonly used value in literatures is 10, 5, or 3. LOOCV and Bootstrap have powerful potential with regard to their finer variation of subsets. They should be the first choice when dealing a problem with little data. Automatic determination of *k* in *k*-CV is an interesting issue to find a rational *k* setting independently by analyzing statistical information of data.

4.7.2 Selection Stability

Stability of a model is an indicator of its tolerance capacity when facing with independent data or situations. In dimensionality reduction domain, a selection stability has been proposed in [156] to measure the stability of the selected subset by adapting the Leave-One-Out validation with distribution analysis. For a given data set \mathbf{X} of N instances, the LOO procedure is used to validate N variations of feature vectors each designed by $N - 1$ instances data via an FS method. The frequency of appearance of each feature $\rho_i, i \in \{1, 2, \dots, |\mathbf{X}|\}$ is recorded in this process and is used for the final selection. The final subset can be constructed by ranking approach or by a certain threshold a in range $[0,1]$, where features with $\rho_i/n \geq a$ will be selected. The selection stability of overall validation process is calculated as:

$$S_{fs} = \frac{(\rho_{\max} - \rho_{\min})}{|\mathbf{X}|} \cdot 2 \sum_{i=1}^{|\mathbf{X}|} |\rho_i - \bar{\rho}| \quad (4.17)$$

where ρ_{\max} , ρ_{\min} , and $\bar{\rho}$ is maximum, minimum, and average frequency of occurrence respectively. S_{fs} is normalized in range $[0,1]$ where 1 indicates the highest stability. The instability of the selection decreases S_{fs} towards zero. The indication of an inappropriate selection is expressed by low value of S_{fs} , that reflects low contrast of feature subset variation as the information of all features is needed.

A stability measure for classifier model assessment based on $0-1-loss$ function has been proposed in [157]. *Stability Cost* ($S(g)$) is an estimation of stability of the predictor g , which is a chosen classifier model with tuned parameters. The estimation of S is formulated as:

$$S(g) = \frac{1}{n} \sum_{i=1}^n \delta \{g_{\mathbf{X}_{train}}(x_{test,i}) \neq g_{\mathbf{X}_{test}}(x_{test,i})\} \quad (4.18)$$

where δ is the $0-1-loss$ function of classification results, $g_{\mathbf{X}_{train}}$ and $g_{\mathbf{X}_{test}}$ is the learned classifier from the separated train and test data respectively. $S(g)$ is 0 in the perfect stability case, whereas large value represents significant instability. In the case that g is obtained by the *empirical risk minimization* approach, the stability cost yields an upper bound on the generalization classification performance of g .

4.8 Design Optimization

Up to this point, it is obvious that the design of a multi sensor intelligent system from front to back consists numerous tasks of model and method selection and parameter tuning, which lead to more and more complex optimization problems. A number of available performance measures pose the requirement of multiple-objective optimization. To obtain a system with high robustness in real world situations, the optimization task should take into account dynamic effects from environment. These reasons have summed up the foundation of the design optimization process in DAICOX. Taxonomy of optimization algorithms categorized by searching mechanism is illustrated in Fig. 51. Thus, this section presents the details of design optimization procedures, which take into account meta-heuristic algorithms, multi-objective optimization approaches, and techniques for optimization in dynamic environment.

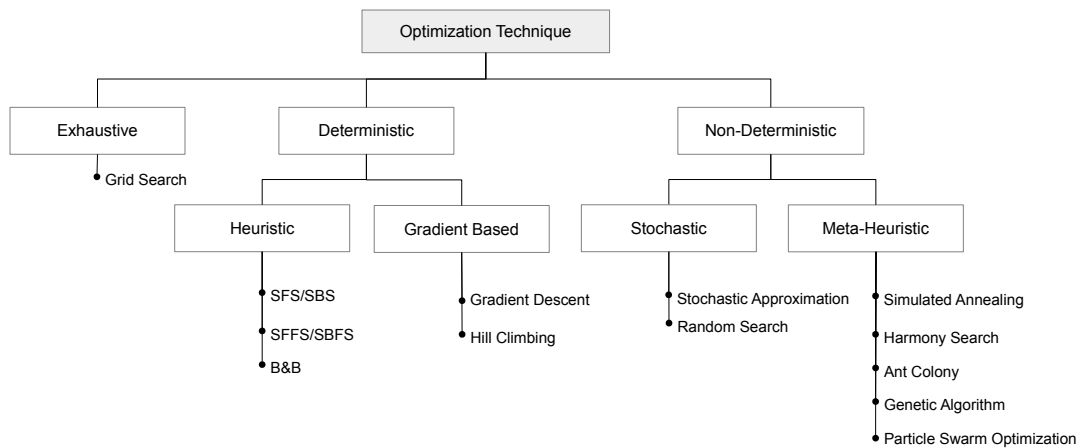


FIGURE 51: Taxonomy of commonly used optimization algorithms

4.8.1 Computational Intelligence Based Optimization Algorithms

Computational Intelligence (CI) have been adopted in several optimization algorithms. Two bio-inspired families for optimization are established in part of CI: *swarm intelligence* and *evolutionary algorithm*. These methods are also referred as *meta-heuristic* approaches, which combine stochastic and strategic characteristics to the search procedure. The concept of swarm intelligence is inspired from the behavior of a population, i.e., *swarm*, and the social interaction between its individuals, while evolutionary algorithm mimics the natural evolution process based on the Darwinian evolution idea.

4.8.1.1 Genetic Algorithms

Genetic Algorithms (GA) [158], an evolutionary algorithm based approach, represents problem or design variables to its population in the chromosome form. Each individual in the population contains a string of genes whose dimension is the number of design variables. The evolutionary process in GA, i.e., *generation*, comprises the manipulation of chromosomes (design solutions) through these operations:

- Initialization: create a set of population for the first generation. Population initialization is a vital part of GA for achieving good solution within reasonable time. The diversity of population is the common consideration to ensure the full exploration over the search space. Some applications may have a specific way for initialization such as in feature selection, some individuals can be obtained by using a heuristic method, e.g., SFS/SBS.
- Selection: select some individuals as the parents for creating offspring by the recombination process for the next generation. The standard selection methods are based on uniform random, roulette-wheel, or tournament selection [158]. For uniform selection, the probability of selection is controlled by parameter P_s , which is typically set at 0.5 meaning that half of the population will be selected for recombination.
- Crossover: a recombination process, that creates offspring from the selected parents. Every offspring contains a modified chromosome of its parents created from the *crossover procedure*. One or more random cutting point can be used to crossover a part of chromosome between the parents as illustrated in Fig. 52 for binary representation. In case of real number representation, arithmetic crossover can be used to generate two offspring using the following equation:

$$\begin{aligned} x_{offspring_1} &= \alpha \cdot x_{parent_1} + (1 - \alpha) \cdot x_{parent_2} \\ x_{offspring_2} &= (1 - \alpha) \cdot x_{parent_1} + \alpha \cdot x_{parent_2} \end{aligned} \quad (4.19)$$

where x_{parent_1} and x_{parent_2} are the genes from the first and second parent, respectively, and α is a random number from the interval [0,1].

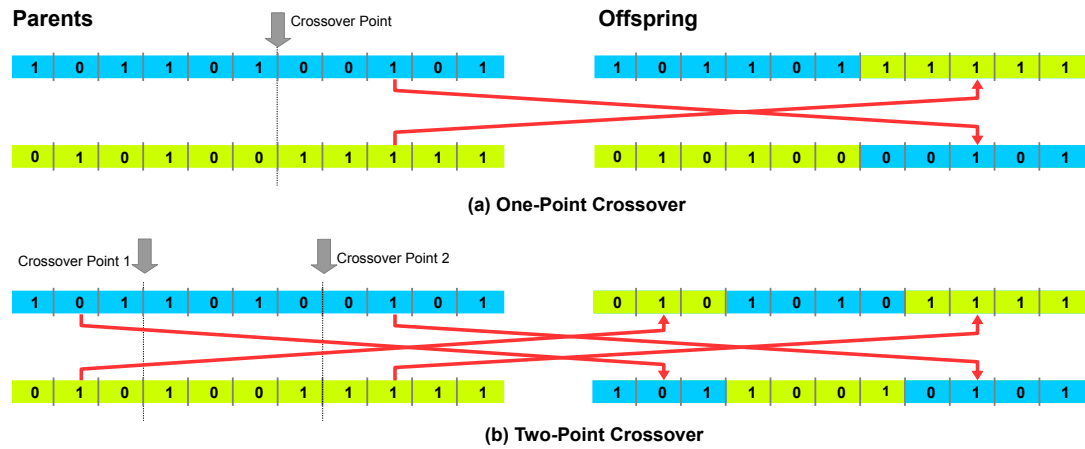


FIGURE 52: Two types of crossover procedure of GA

- Mutation: Random mutations alter some genes of selected chromosomes. For binary value encoded genes, the *bit-flipping* operation is used. For real value genes, the current value will be replaced by the random value based on a Gaussian distribution. The selection probability is controlled by the mutation rate μ parameter, which is suggested to set at 0.2 [158], that means 20% of the population will be mutated. Typically, the best chromosome is not included in the mutation procedure.
- Reproduction: The selection of individuals among the previous generation and current offspring is needed in order to maintain the population size before starting the next generation. The removal process can be based on individual age or fitness ranking.

The block diagram illustrated in Fig. 53 shows the basic operation of GA. After the population is initialized, selection, crossover, mutation, and reproduction are performed inside the generation loop. This loop is the evolutionary process, which keeps continuing until some criteria are fulfilled or the maximum number of generations is reached.

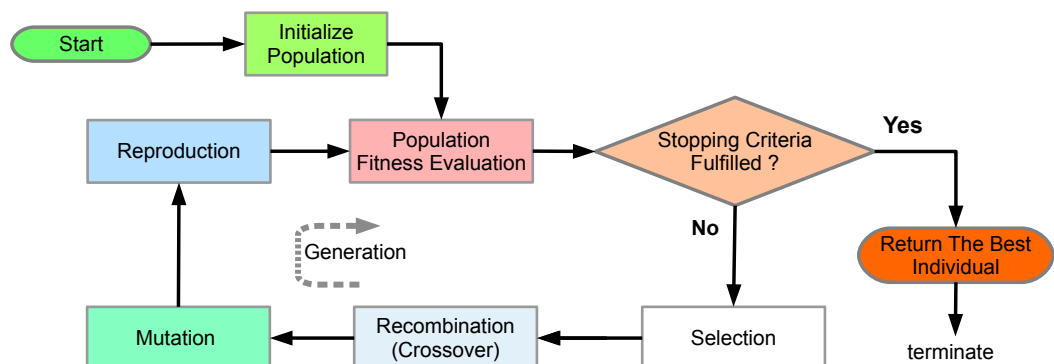


FIGURE 53: Operations of Genetic Algorithms

4.8.1.2 Particle Swarm Optimization

Particle Swarm Optimization (PSO) is inspired by the individual and social behavior of creatures such as bird flocking, fish schooling and bee swarming. PSO is a population based optimization method similar to GA but each individual represents its location in the search space instead of chromosome. Design variables are encoded to an individual in terms of coordination in multi-dimensional space. The population update procedure of PSO consists of one operation, which moves individuals (particles) throughout the search space. The movement of a particle is the combination of its current position and its velocity. The new velocity of each particle is the attraction from particle's best p_b and population best g_b positions associated with inertial weight w , and individual C_1 and social C_2 learning ratio. The velocity v_t and position x_t update of a particle are formulated as in Eq. 4.20 where $rand()$ is a random number generator of the interval $[0,1]$. The optimization process of PSO is elucidated in Fig. 54.

$$\begin{aligned} v_t &= wv_{t-1} + C_1rand()(p_b - x_{t-1}) + C_2rand()(g_b - x_{t-1}) \\ x_t &= x_{t-1} + v_t \end{aligned} \quad (4.20)$$

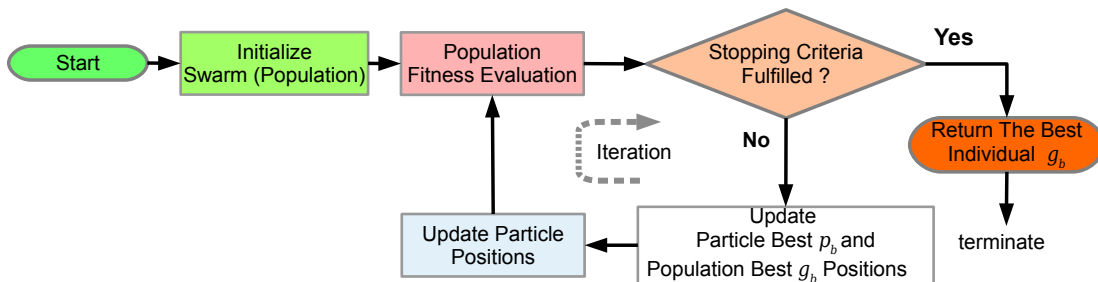


FIGURE 54: Operations of Particle Swarm Optimization

Due to the simplicity, the calculations of the position and velocity may lead to some undesirable behaviors of particle dynamic. Thus, additional control mechanisms are suggested to be incorporated to ensure appropriate movements of every particle as well as to increase the performance of PSO.

- **Position Confinement:** Ensures that the new position is still inside the search space boundaries before evaluating individual's fitness. This is due to the majority of design variables are constrained within limited extent, hence, individuals containing some infeasible values should not be valid. The treatment for the individuals ventured outside the boundaries can be the discarding of the individual or the relocation of the individuals by random, absorbing or reflecting mechanism [159].

- **Weight Decay:** To obtain better result resolution, the inertia weight w should be decreased over time (iteration) in order to gain the finer movements of particles. As suggested in [160] the empirical relevant range of w is [0.4,0.9] and the inertia weight at i th iteration can be obtained by:

$$w_i = w_{max} - \frac{i}{i_{max}}(w_{max} - w_{min}) \quad (4.21)$$

where $[w_{max}, w_{min}]$ can be user defined otherwise use the suggested values and i_{max} is the maximum number of iterations.

- **Maximum Velocity:** Restrict the velocity before updating the position in order to prevent violation in the movement of a particle. Maximum velocity setting v_{max} can be determined from the position boundary of each dimension: $x_{max,d}$ and $x_{min,d}$ where $d \in \{1, 2, \dots, D\}$ is the dimensional index of a D dimensions problem. Thus, the maximum velocity of dimension d is computed as: $v_{max,d} = \eta(x_{max,d} - x_{min,d})$ where $0.0 < \eta \leq 1.0$ is the scaling factor, which is typically set in the interval [0.2,0.5].
- **Population Size:** The optimum number of individual n_p is still an open issues to date. One can assumes that an appropriate choice of n_p is equal to the number of dimension of the design variable. It is practical only in problems with low-dimensional design variable. For high dimension problems, n_p can be set to $10 + 2\sqrt{D}$, which is a common formulation in literatures.

Another importance issue of PSO is the representation modification. Naturally, a particle moves in a real number representation search space, which means that the changing in a design variable can only be of continuous value. Therefore, in case of a design variable contains discrete values, the proper conversion is needed in order to interpret the current particle position into the correct format of the design variable before an evaluation.

For the binary representation, the new position can be updated by using a probability threshold of the velocity [161]. The new position of particle is updated by:

$$x_{t,d} = \begin{cases} 1 & \text{if } s(v_{t,d}) > rand() \\ 0 & \text{if } s(x_{t,d}) < rand() \end{cases} \quad (4.22)$$

where s is the sigmoid function. For the integer representation, the computed velocity must be rounded prior to the position update procedure. Thus, the new particle position in integer format is updated as: $x_{t,d} = x_{t-1,d} + round(v_{t-1,d})$.

4.8.2 Multi Objectives Optimization

In real world engineering application design problems, multiple goals, specifications and constraints are involved, which is the typical scenario in the design of intelligent multi-sensor systems. Thus, the design is Multi-Objective Optimization (MOO). MOO considers more than one objective functions to be optimized simultaneously with an appropriate handling procedure. An MOO problem with n objectives can be described as:

$$\begin{aligned} \min \text{ or } \max \mathbf{y} &= \mathbf{f}(\mathbf{x}) = \{f_1(\mathbf{x}), f_2(\mathbf{x}), \dots, f_n(\mathbf{x})\} \\ \text{subject to } \mathbf{x} &= \{x_1, x_2, \dots, x_d\} \in \mathbf{X} \\ \mathbf{y} &= \{y_1, y_2, \dots, y_n\} \in \mathbf{Y} \end{aligned} \quad (4.23)$$

where \mathbf{f} is the objective or fitness function vector, \mathbf{y} is the objective value vector in the objective space \mathbf{Y} and \mathbf{x} is design vector containing d elements of the design parameters (or design variables). Clearly, MOO finds the best design vectors, that satisfies all criteria in \mathbf{f} , which is the main challenge of MOO problems. Constrained optimization problems are the major practice of MOO. For example, given a feature selection task with limited number of features as a constraint, the cardinality of the feature vector will be taken account into the optimization process as an additional assessment. In advanced cases, information from other processing blocks can also be used to constrain the optimization process, e.g., measurement time, signal quality, power consumption, classification accuracy, model sensitivity, etc. This will keep the search procedure in the potential landscape of feasible solutions matching those (soft) constraints and nullify ones that fail to cope with the constraints, i.e., infeasible solutions.

Two common approaches in handling MOO problems, i.e., *weighted agglomeration* approach and *Pareto* method are explained.

Weighted Agglomeration Approach

Weighted Agglomeration (WA) approach, which is also called weighted-sum, transforms multiple objectives into an aggregated scalar objective function f_{sum} . Each objective value y_n is multiplied by a weight factor w_n as expressed in Eq. 4.24, then the optimization process takes y_{sum} as an assessment value of \mathbf{x} for every evolution.

$$y_{sum} = f_{sum}(\mathbf{x}) = \sum_{n=1}^N w_n f_n(\mathbf{x}) \quad (4.24)$$

where

$$\sum_{n=1}^N w_n = 1$$

WA is among the most convenient approaches due to its simplicity, that requires only few extra calculation steps. However, the appropriate procedure to set the weights is an open issue. In addition, the summation is satisfied under the assumption on convex circumstance. Therefore, variations of WA have been the subject in dealing with the problems. Adaptive weight setting [162] has been proposed to gain the ability of WA in dealing with the non-convex optimization problem. In [163], the weight vector is part of the evolutionary process, thus, a set of design solutions and weight settings are dynamically adapted to a particular problem simultaneously.

Pareto Approach

In the concept of *Pareto* optimality, a set of solutions of a multi-objective optimization problem cannot improve any dimension in the objective vector without degradation in another. Assume a maximization problem and consider two design vectors \mathbf{a} and \mathbf{b} where $\mathbf{a}, \mathbf{b} \in \mathbf{X}$. Then \mathbf{a} dominates \mathbf{b} iff:

$$\begin{aligned} \forall i \in \{1, 2, \dots, n\} : f_i(\mathbf{a}) \geq f_i(\mathbf{b}) \quad \wedge \\ \exists j \in \{1, 2, \dots, n\} : f_j(\mathbf{a}) > f_j(\mathbf{b}) \end{aligned} \quad (4.25)$$

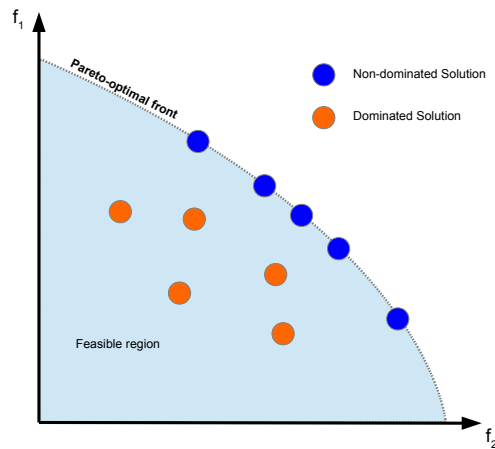


FIGURE 55: Illustration of Pareto approach of two objective optimization problem

By this notion, the set of *nondominated* design vectors, which are not dominated from any candidates, is the *Pareto front* [164] as illustrated in Fig. 55. The goal of Pareto based MOO is to collect as many of nondominated design vectors as possible. Thus, the selection of the final solution among others in a Pareto front is an important issue.

Ranking is applied in [165], where the optimization starts by collecting all nondominated individuals to create a subpopulation and remove it from the population before repeating the process until all individuals are classified. The ranking is based on the size of the subpopulation.

Nondominated Sorting (NS) [166] ranks individuals in the same manner as the previous method. A set of nondominated individuals is assigned by a dummy fitness value instead of explicit ranking. The fitness value is then degraded by the *share* factor, which is calculated from the distance of each solution to its two closest neighbors in the design space. By using this technique, optimization process actually searches within Pareto-optimum regions.

4.8.3 Intrinsic Design Optimization

A well-tailored solution of a multi-sensor cognition application can be obtained by concurrently adjusting hardware configurations as design variables of the optimization process. In particular, in robustness issues, the process modelling are commonly rather based on stationary model. Ultimately, the deployed model or system solution will fail after some time without on-site adaptation when dealing with non-stationary situations. These together strongly increase the importance of intrinsic optimization. The design optimization problems dealing with dynamic deviations are categorized in the hierarchy listed below.

- Extrinsic nominal design assumes stationary process in the design model. Solutions are identical during deployment.
- Extrinsic design with predicting process statistics, drift, aging, etc., to compensate solutions' parameters during deployment.
- Intrinsic (static) design incorporates the instance in the design loop from a host which requires an observer to compute performance metrics and optimize (basic self-x properties).
- Intrinsic (dynamic) implements performance metrics and optimization on the instance (complete self-x functionality). The design repeated or cyclical with information observed by itself from basic to complex level of information processing.

The integration of intrinsic optimization can be in two approaches; instance adaptation which performs optimization on the design platform, and dynamic adaptation which embeds optimization process in run-time machines. The first allows fine-tuning of a solution with regard to a hardware instance in order to cope with deviations, e.g., manufacturing tolerances, during deployment phase while the latter is more superior that it is capable to recover from deviations during operation time. To realize this feature, DAICOX allows the inclusion of a hardware instance in design optimization loops to provide intrinsic evolution of a designed solution. The operation mechanism of the instance adaptation approach incorporates communication between an optimization operator and

a hardware instance as depicted in Fig. 56. To enable hardware instance adaptation capability, digital reconfiguration, e.g., using FPGA, is the typical approach, but in particular, analog/mixed-signal adaptation is more attractive for intelligent multi-sensor applications, thus, advanced reconfigurable devices are considered including FPTA and FPMA. The hardware instance optimizer tunes hardware configurations with regard to some criteria which can be based on signal representation of the hardware instance, e.g., MSE, SNR, or THD or based on information in data domain, e.g., classification accuracy or feature quality measure. In particular, the design of the standard building blocks may provide some design constraints to the hardware optimization process. The live measurements acquired from the hardware instance can be included in the data source for standard building blocks optimization. For the dynamic adaptation approach, the challenge is that entire process is carried out by the run-time platform in order to obtaining longterm robustness of the optimized solution.

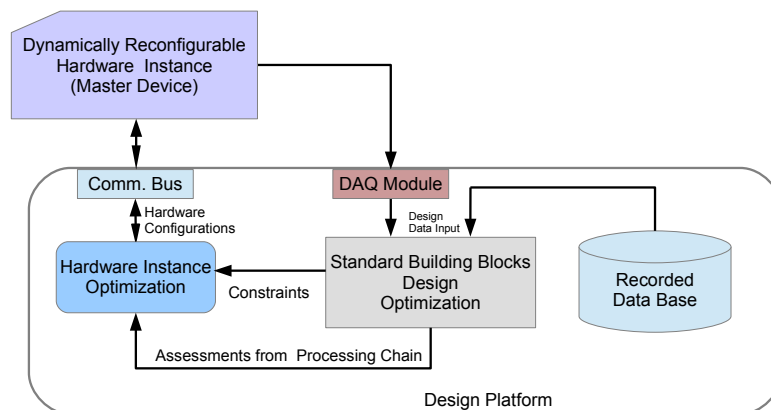


FIGURE 56: Proposed intrinsic optimization process

4.8.4 Optimization in Dynamic Environments

A conventional engineering system design assumes static environment, which expects a system to provide constant performance at any operation time. In reality, environmental uncertainties occur over time resulting deviations in the performance. This issue is the motivation of self-x properties in IIMSS, because run-time operations will have to face environmental perturbations. In particular to dynamic deviation issues, dynamically adaptive optimization techniques, that adjust system configurations and parameters respect to changes are salient.

When environmental changes occur, assessment value of a particular design vector can be varied in each evaluation [167], thus, the peaks in the search space of each generations

may also be moved. The typical suggestion to tackle this issue is to find the set of robust solutions, which are diverse and as close as possible to the optima. For these reasons, several optimization techniques aiming to tackle dynamic environmental problems are established, which can be categorized by the mechanisms given in the following.

Increasing Diversity

Diversity can be increased by enforcing the population resetting over a certain period of time or as soon as changes are detected [168], where the sign of change can be, e.g., a reduction in the best assessment values, or re-evaluation of old solutions. This approach is the simplest, however, the knowledge gained from the past evolutions is completely ignored that may not be an economically good choice.

Maintain Diversity

This approach contains no explicit action to react to changes. Instead, it maintains the diversity of population and concurrently avoid the convergence of a large number of individuals over time. This mechanism guarantees that the population may not be trapped in local optima when changes occur. Charged Particle Swarms CPSO [169] applies the repulsion mechanism to prevent particles get too close to each other. Compound PSO [170] deviates particles from their original locations proportional to the velocity reductions in each particles, which is an implicit indication of convergence. The advantage of maintaining diversity is that it may be effective for dynamic problems with severe changes and occur intermittently.

Memory Based

The main focus of this approach is to reuse the information gained in the past, which can be the locations of previously found optima. In some types of changes in environment, the optima may be moved to the vicinity of their previous locations if the changes are periodic or recurrent, which is the situation the approach performs best. Memory enhanced model [171] divides the population into two groups, where the first exchanges individual information with an external memory while the other keeps searching for new peaks and stores data to the memory if new peaks are found. Instead of directly modifying the information in memory, Dynamic Memory Model [172] updates stored information by slightly moving all individuals in the memory towards newly found peaks.

Multi-Population

This approach divides the whole population into multiple sub populations in order to handle several areas within the search space simultaneously. Diversity can be maintained by re-allocating, splitting, and merging to avoid the overlap between sub populations. Shifting Balance GA [173] constructs a population consisting of a number of a small size subpopulations for exploring the search space. Once a new peak is found, it will be tracked by some the remainder individuals. Anti-convergence mechanism was introduced

in Multi-swarm PSO (mPSO) to ensure one free swarm to continue exploring the search space while the rest keep tracking discovered peaks.

4.9 Discussion

In this chapter, the DAICOX architecture is presented with the aim of tackling the design effort and to providing high quality and robust solution for intelligent multi-sensor systems. The DAICOX architecture consists of multi-objective design optimization to designing a system from available processing components in the method pool. Numerous methods and algorithms to be included to the method pool ranging from sensor configuration to classification are explained in this chapter. Sensor configuration part configures physical settings of sensing elements and electronics to enhance the signal quality. Signal processing and feature computation improve the quality of acquired data and also create new information. Feature weighting is proposed as a potential candidate for dimensionality reduction providing a finer scale of selection. Beside multi-class classification, presented one-class classification methods, e.g., One-Class SVM and NOVCLASS show the desirable ability for some advanced tasks, e.g., anomaly and novelty detection. The design tasks of these methods are carried out by presented meta-heuristic optimization methods, e.g., GA or PSO, with multi-objective optimization approach. DAICOX is conceived to effectively handle the problem of method selection and parameter setting for a particular application yielding high performance solutions. In particular, DAICOX will integrate self-x properties as well as in-the-loop dynamic environments optimization into its platforms to gain reliability and robustness of the performance. These concepts, techniques, ideas, and methods constitute the aims and goals for the DAICOX architecture implementations, which is presented in the next chapter.

4.9.1 Overview of Parameter Settings

Several methods and algorithms presented and explained in this chapter require pre-defined parameters for their operations. The recommended parameter settings, which typically are empirically derived from experiments or case studies, for each method are summarized in Tab 9. The values are given in terms of proper range or specific number, where two numeric types are given: integer(int.) number and real number. More detail in parameter setting of a particular method can be found in its corresponding text in this chapter.

TABLE 9: Recommended parameter settings

Method	Parameter	Type	Description
Feature Selection			
plus l -take away r	l and r	int.	$1,2,\dots$, no. of features $\times 0.25$
Oscillating	d	int.	no. of features $\times 0.5$
	Δ	int.	$0.1 \times d$
Feature Extraction			
PCA	d	int.	$1,2,\dots$, no. of input features
LDA	d	int.	$1,\dots$, no. of classes-1
SOM	η	real	< 0.01
Classification			
k -NN	k	int.	$1,2,\dots$, no. of instances per class
SVM	C	real	[1,10000]
(RBF kernel)	γ	real	[0.00001,10]
NN-D	k	int.	1
	ρ	real	< 1.0
SVDD	C	real	> 1.0
	C	real	$nSV \times C < 1.0$ (when outliers data available)
	γ	real	[1,25]
Oneclass-SVM	ν	real	[0.01,0.9] (recommended at 0.05)
NOVCLASS	η	real	[0.1,2]
Feature Quality Measure			
Overlap q_{OV}	k	int.	[5,10]
Compactness q_c	w	real	{0,1.0} (typical at 0.5)
Model Analysis			
Holdout	r_H	real	{0,1.0} (typical at 0.25,0.5 or 0.75)
k -CV	k	int.	5 or 10
Optimization			
GA	P_m	real	0.2
	P_c	real	[0.5,1.0]
PSO	C_1	real	1.149
	C_2	real	1.149
	w	real	[0.4,1.2] (or time dependent within the range)
	v_{max}	real	$0.25(x_{max} - x_{min})$
	p	int.	$10 + 2\sqrt{d}$

Chapter 5

Implementation of the DAICOX Architecture

The presented motivation, background, and ongoing research stimulate the increasing implementation of tools and methods for the design automation of intelligent multi-sensor systems. Thus, the contribution to the comprehensive framework, which is derived from the DAICOX architecture proposed in the previous chapter, is presented in this chapter. This chapter presents the current status on the implementation and instantiation of methods and system applications. The overview of the established framework is elucidated in Fig. 57. The design process is done in one direction based on *bottom-up* approach, where the optimization process performs sequentially block by block. In particular, the implementation of effective optimization techniques strongly focused on meta-heuristic approach along with multi-objective optimization concepts are incorporated to the design optimization part of the framework. At each design step, the optimization process is performed in the local approach, whereas the inclusion of additional assessments or fitness functions from other blocks is also possible by multi-objective optimization, for instance, designing a feature selection task by taking account of classification performance.

All implemented methods discussed in this chapter are developed in form of method library for DAICOX architecture, i.e., DAICOX software library. The implementation of DAICOX software library is conceived with the aim of:

- Open source and multi-platform,
- Flexibility and modularity,
- Extendable in both creating new libraries or modify/upgrade present ones, and
- Support inclusion/extension from external libraries.

DAICOX software is mainly developed in Python programming language. Designing a system using the DAICOX can be done in two manners; Python scripting or visual programming. The first requires insight understanding of the DAICOX library structure and programming, where the latter represents software abstraction via graphical modules, i.e., Widgets. Visual programming in DAICOX provides a convenient and intuitive way for developing sensory technical cognition applications.

Several methods implemented in the framework are given in detail as well as their preliminary experimental results. In the following subsections, implemented methods' groups categorized by the standard building blocks are explained. Then, in Section 5.4 the design optimization mechanism for the implemented framework is described. The information on software structure and environment is given in Section 5.5.

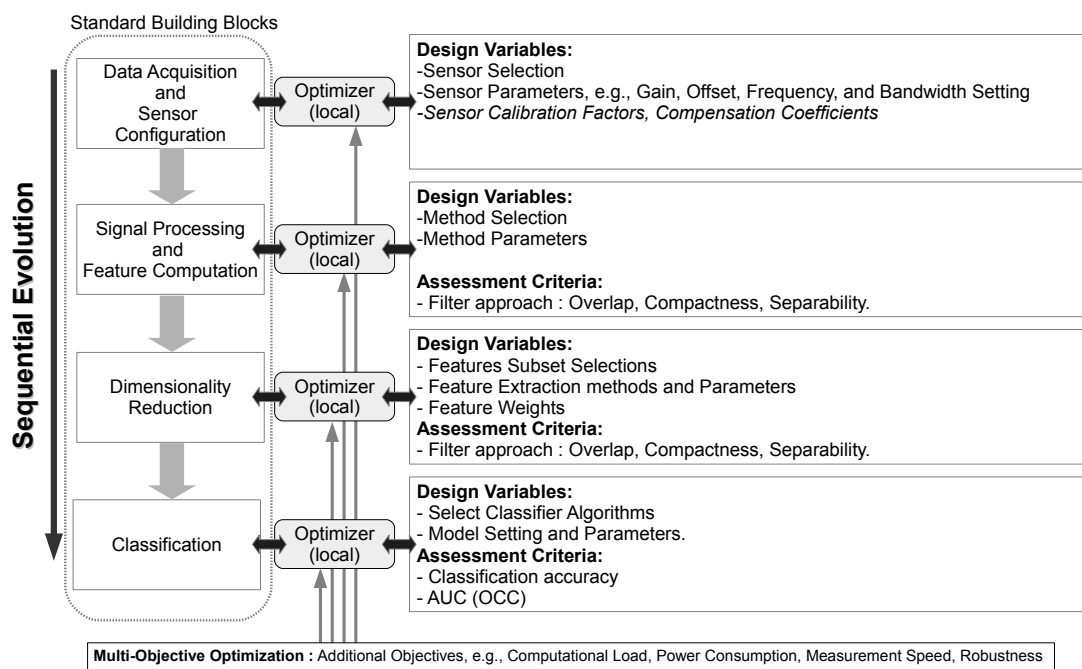


FIGURE 57: Overview of DAICOX framework implementation in computational intelligence domain

5.1 Signal Processing and Feature Computation

The current implementation of Signal Processing (SP) and Feature Computation (FC) comprises methods from self-developed and external standard available libraries, e.g., Scipy and Orange. For each included external library, a library wrapper is used to associating data, control, and parameters between native library and DAICOX software environment. The standard method interface for SP and FC is depicted as software flow block diagram in Fig. 58. The common interface descriptions are manifested by

the DAICOX library interface, which is described in Section 5.5.1. The DAICOX library interface ensures the proper operation of external libraries, when using together with other methods in a processing chain. In particular, to some methods with free parameters, the inclusion of these parameters into DAICOX design optimization is also applicable. The currently available SP and FC methods, ranging from basic arithmetic to signal transformation, are given in Table 10. These methods are employed in part of the experiments and case studies given in the next chapter. Introducing a new processing method from existing library can be achieved by creating a wrapper script, which transforms input and output data formats as well as required parameters.

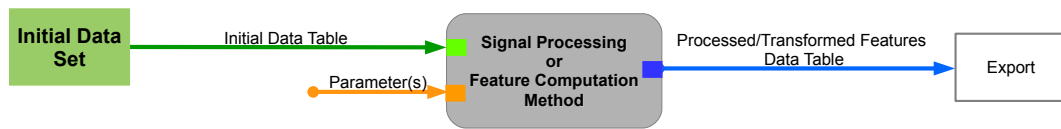


FIGURE 58: Signal processing and feature computation software flow

TABLE 10: Signal processing and feature computation in DAICOX

Method Group	Functions	Source
Statistic	Basic statistical information of data	DAICOX
Normalization	Data matrix normalization with reusable coefficients	DAICOX
Smoothing	Signal smoothing function, e.g., Gaussian filter, Moving average	DAICOX
Function Approximation	Polynomial curve fitting	Scikit
Filtering	Digital IIR filter	Scikit
Convolution/Cross-Correlation	Data matrices convolution/cross-correlation	DAICOX
FFT/STFT	Frequency domain transformation	Scikit
PCA	Transform Feature Space (Unsupervised)	Orange
LDA	Transform Feature Space (Supervised)	Scikit

5.2 Dimensionality Reduction

The implementation of Dimensionality Reduction (DR) process in DAICOX follows the software structure illustrated in Fig. 59. Three main DR categories are proposed including Automated Feature Selection (AFS), Automated Feature Weighting (AFW), and

Feature Extraction. For feature extraction, two methods are included in the method pool by using external library, e.g., PCA from Orange library and LDA from Scipy library. The proposed DR process takes account of several feature quality assessment functions and the capability of multi-objective optimization using weighted agglomerative approach. The model analysis part employs Hold-out Random Sampling (HRS) and k -fold Cross Validation (k -CV). The consideration of solution stability and acquisition cost can be incorporated in the decision making process to select the high quality with high robustness feature subset.

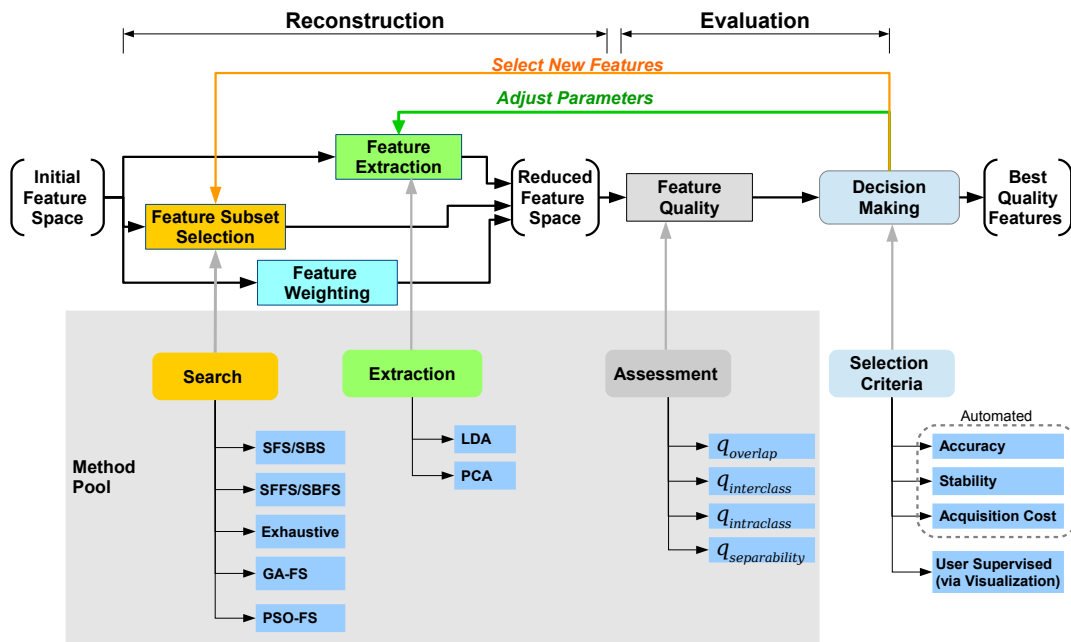


FIGURE 59: Dimensionality reduction architecture

5.2.1 Choice of Feature Subset Assessment Functions

A number of feature quality assessment functions are available in the DAICOX library with the main purpose for dimensionality reduction tasks. Non-parametric feature space measures are selected in the implementation in order to provide better generalization performance rather than classifier based measure (wrapper approach). The list of currently available assessment functions is given in brief details in Table 11, where precise information of each method can be found in Section 4.6.2.

TABLE 11: Description of feature subset assessment functions in DAICOX

Assessment	Indication	Output	Parameters
Overlap q_{OV}	Class overlapping region	Normalized [0, 1]	k
Compactness q_C	Within-Class region compactness ($q_{C-intra}$) Between-Class region separation ($q_{C-inter}$)	Unnormalized	-
Separability q_S	Class regions separation quality (Linearly separable)	Normalized [0, 1]	-

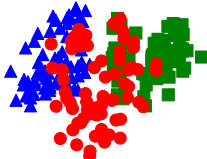
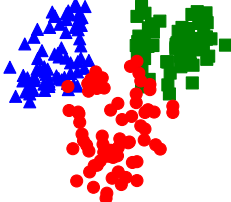
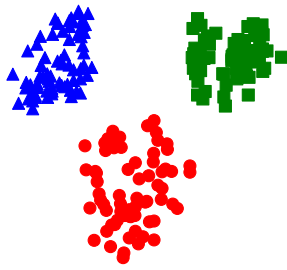
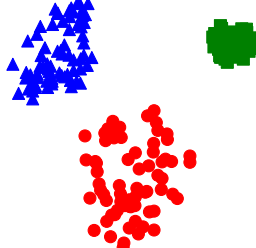
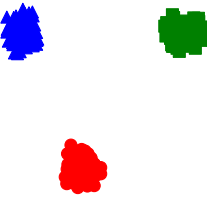
As mentioned in the previous chapter, the calculation of the compactness measure q_C suffers from non-normalized distance metrics, which results in unbounded assessment values. In an optimization task, this can cause problems and mislead to irrelevant evolutions especially in stochastic approaches. The way to solve this is to compute $q_{C-intra}$ normalized by the value of the original set (full-feature), $q_{C-intra_0}$, at the beginning of a search. The calculation of the relative $q'_{C-intra}$ is given as:

$$q'_{C-intra} = 1.0 - q_{C-intra}/q_{C-intra_0}, \quad (5.1)$$

where $q_{C-intra_0}$ is the measure of the full-feature data set. The improvement of class region compactness increases the value towards 1.0, which is an ideal case, where all points of the same class lay at a location in feature space (zero variance). However, negative value is still unbounded, but, it indicates the deteriorated quality from the original set, thus, a clipping technique is suggested.

Table 12 shows the sensitivity characteristics of each implemented feature quality measure methods in different simulated scenarios from artificially generated data sets. At the first step, an initial feature space of 3-class data set, each class has 350 data samples, is given with some overlap areas between classes. $q_{C-intra_0}$ is computed in this step. All class clusters depart away from each other by some distance in step 2 simulating the smaller overlap regions compared to step 1, thus, q_{OV} and q_S increases indicating the better quality of the feature space. $q_{C-inter}$ is also sensitive in this case as the mean of each cluster is shifted, while $q'_{C-intra}$ is insensitive in this case as the formation of each cluster remains unchanged ($q'_{C-intra} = q_{C-intra_0}$). Step 3 repeats the same action by moving each cluster further until no overlap area remains. $q_{overlap}$ and q_S are saturated, whereas $q_{C-inter}$ still sensitive to the improvement (increasing space between class clusters) in this situation. The sensitivity of $q'_{C-intra}$ can be seen in step 4 where the class 2 cluster (in green squared shape) is shrunk indicating the improvement (compacting) of the cluster. The change occurs only in $q'_{C-intra}$ of class 2 while $q'_{C-intra}$ of other classes remain unchanged until step 5 where all clusters are compacted.

TABLE 12: Sensitivity characteristics of feature assessment functions

Step	Feature Space	$q'_{C-intra}$	$q_{C-inter}$	$q'_{C-intra}$ (per class)	q_{OV}	q_S
1		0.000~	0.425~	class 1: 0.000~ class 2: 0.000~ class 3: 0.000~	0.768~	0.705~
2		0.000~	0.518↑	class 1: 0.000~ class 2: 0.000~ class 3: 0.000~	0.941↑	0.905↑
3		0.000~	0.609↑	class 1: 0.000~ class 2: 0.000~ class 3: 0.000~	1.000↑	1.000↑
4		0.161↑	0.666↑	class 1: 0.000~ class 2: 0.595↑ class 3: 0.000~	1.000↑	1.000↑
5		0.665↑	0.764↑	class 1: 0.669↑ class 2: 0.595~ class 3: 0.711↑	1.000↑	1.000↑

↑ refers to expected increasing and ~ refers to expected similarity of the assessment value compared to the previous step.

5.2.2 Automated Feature Selection

Automated Feature Selection (AFS) is the main feature of DR process in currently implemented DAICOX architecture. By taking the concept of methods already available in QuickCog as a baseline, several AFS methods have been implemented and presented here. A number of extensions with advanced searching topology, in particular, meta-heuristic approaches, have been elaborated as well. In addition, the novel feature weighting method proposed in Chapter 4 Section 4.4.3 is also implemented as an alternative choice for DR process. Aforementioned feature subset assessment functions are used in evaluation phase of searching procedures, i.e., filter approach.

Sequential Methods Four sequential based feature selection methods are implemented in DAICOX including SFS, SBS, SFFS, and SBFS. These methods are among the simplest search procedures of DR tasks. They are considerably fast when dealing with low-dimensional data, but, they may be prone to sub-optima if there exists non-monotonicity. The other advantage is that they require no parameter. SFS and SBS are the most simplest and fastest methods while SFFS and SBFS take higher number of iterations to complete a search but are less susceptible to sub-optima. All AFS methods are implemented in modular scheme as illustrated in the software flow block diagram, Fig. 60. The scheme provides high flexibility in designing DR process as well as facilitates future inclusion of new methods. For feature selection case, a method module requires an input data table in supported formats, e.g., Python Array, NIF, and Orange Data Table and an assessment function object. The combination of multiple assessment functions for multi-objective optimization is also possible and can be configured in the assessment function module. A feature selection module outputs two types of its best result; data table of the selected feature subset and the feature subset descriptor. The first is useful during the design and the latter will be used as selection template for operational purpose.

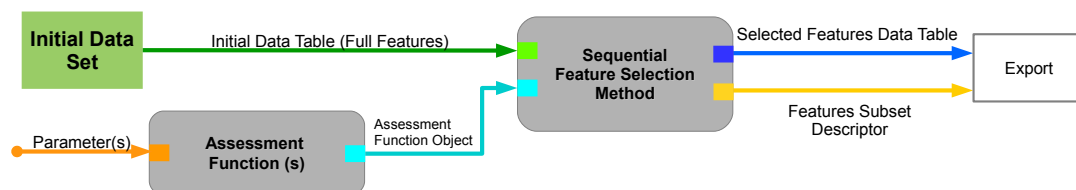


FIGURE 60: Software flow block diagram of sequential AFS method

TABLE 13: Parameter settings for meta-heuristic AFS methods

Method	Parameters and Settings
GA-FS	Population Size = $10 + 2\sqrt{d}$, Max. Iteration = 50, Selection Rate = 0.5, Mutation Rate = 0.2
PSO-FS	Population Size = $10 + 2\sqrt{d}$, Max. Iteration = 50, $C_1=2.0$, $C_2=2.0$, $w=1.2$

d is the number of features of initial data.

Meta-Heuristic Methods Two methods based on meta-heuristic are implemented, GA-FS and PSO-FS, using Genetic Algorithms and Particle Swarm Optimization, respectively. In both cases, the searching procedure employs the general purpose optimization module implemented in DAICOX library. The modules can be used in several purposes covering all design steps in DAICOX standard building blocks. In particular, in DR tasks, the meta-heuristic optimization, e.g., GA or PSO and their corresponding DAICOX modules, can also serve to improve DR quality in visualization, replacing gradient decent techniques applied in the standard approach. The software flow of implemented Meta-Heuristic AFS using general purpose optimization module is illustrated in Fig. 61. The modularity of this structure facilitates the implementation of method variations by just replacing the optimization module with a new method. Two additional components are used in this flow; the feature selector module that applies a given feature subset descriptor to create a feature subset data, and, the object method caller that computes the assessment value from a given function object and data table. The optimization procedure consists of the feature subset generation and modification with regard to assessment results. The best feature subset data table and descriptor will be returned when the optimization task is finished. Using meta-heuristic algorithms requires proper parameter settings in order to ensure the discovery of good quality, or even optimal, solutions. Parameter setting has been the topic in several references and the majority of suggestions are from empirical studies. In our implementation parameters are predefined as given in Table 13, which follow the suggestions given in [109] for GA-FS and in [160] for PSO-FS.

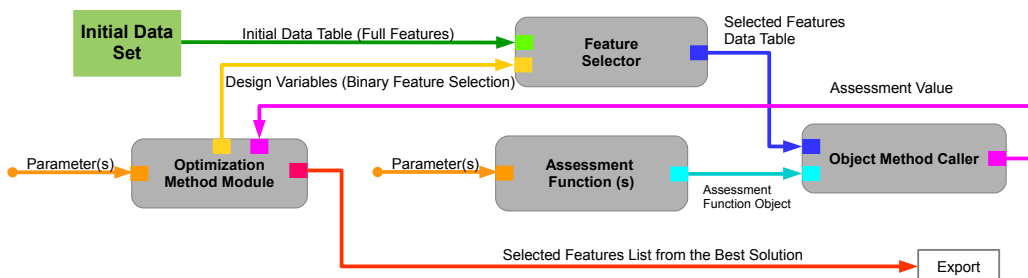


FIGURE 61: Software flow block diagram of meta-heuristic AFS method

For operational investigation, the implemented methods were tested with real-world data sets of conventional classification problems taken from the UCI repository [174]. Table 14 provides information of the data sets. The objective of this investigation is to instantiate a design of dimensionality reduction process using implemented DAICOX framework, that helps the designer to select relevant feature selection methods for a particular application as well as to analyze newly implemented methods. New methods can be included by simply adding new method processing blocks into the software flow depicted in Fig. 62. The module processing block represents the software flow shown in Fig. 61 and Fig. 60 for a particular implemented method. Four assessment functions including q_{OV} , q_S , $q'_{C-intra}$, $q_{C-inter}$ are combined. In order to indicate the recognition performance with good generalization, 10-fold cross validation is employed on training data set, which is sampled by Hold-out Random Sampling (HRS) method with the size of 50% of the complete data. The data are uniformly sampled from the original data set to form two data subsets with the same class proportion of the original data. Thus, the recognition performance results are evaluated on the remaining data, i.e., testing data set. The results are visualized in the box plots showing in Fig. 63 to 68 comparing different searching methods by each assessment criterion. Feature space visualization is applicable for this set-up, which will be used in the experiments and cases studies described in the next chapter.

Sequential methods can provide considerably good performance in several cases outperformed meta heuristic approaches in some data sets. However, the number of used iterations grew by the feature size as can be observed in Table 15 while fixed (by setting) in meta-heuristic approaches. This shows an advantage of meta heuristic approaches, when dealing with high dimensionality data, where sequential methods may not be practical. The better results for meta-heuristic approaches can be achieved by increasing number of searching candidates, constrain search space, and extending maximum number of iteration.

TABLE 14: Real-world data set descriptions

	Data Set					
	iris	wine	ionosphere	wdbc	yeast	mechatronic
Instances	150	178	351	569	186	1,775
Features	4	13	33	20	79	24
Classes	3	3	2	2	3	4
Examples/Class	50-50-50	59-71-48	225-126	212-357	35-30-121	250-500-850-175

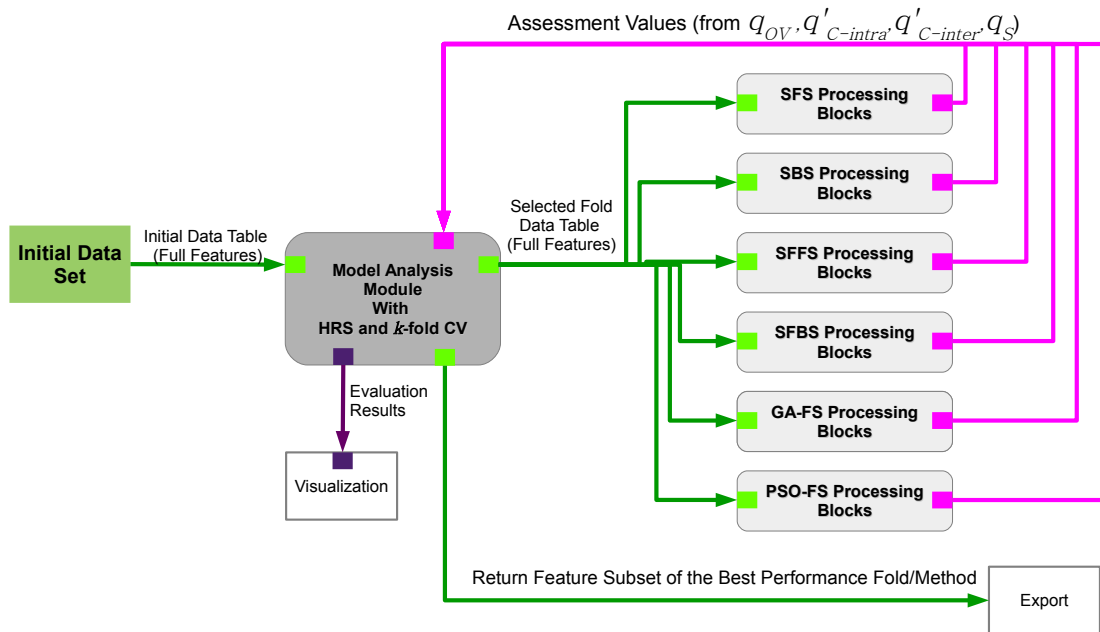


FIGURE 62: Operation of meta-heuristic AFS method modules investigation

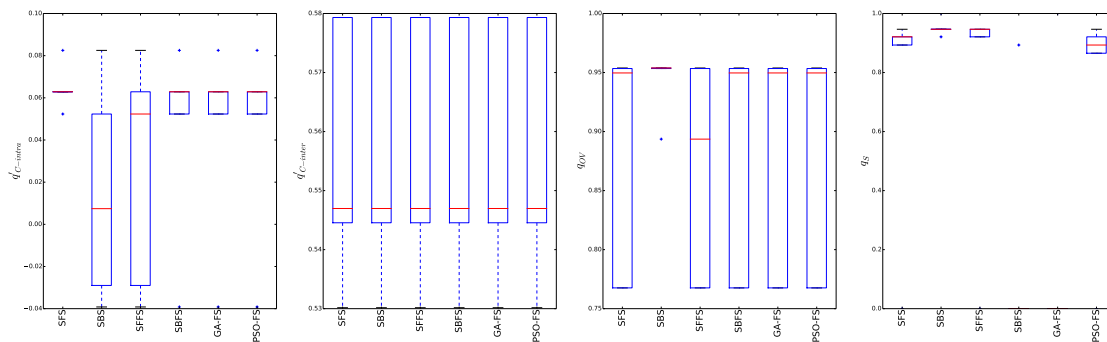


FIGURE 63: AFS performance comparison of the "iris" data set

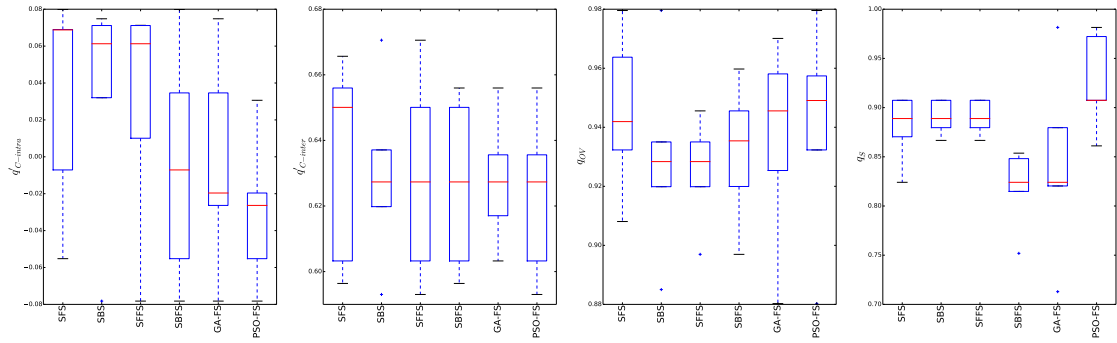


FIGURE 64: AFS performance comparison of the "wine" data set

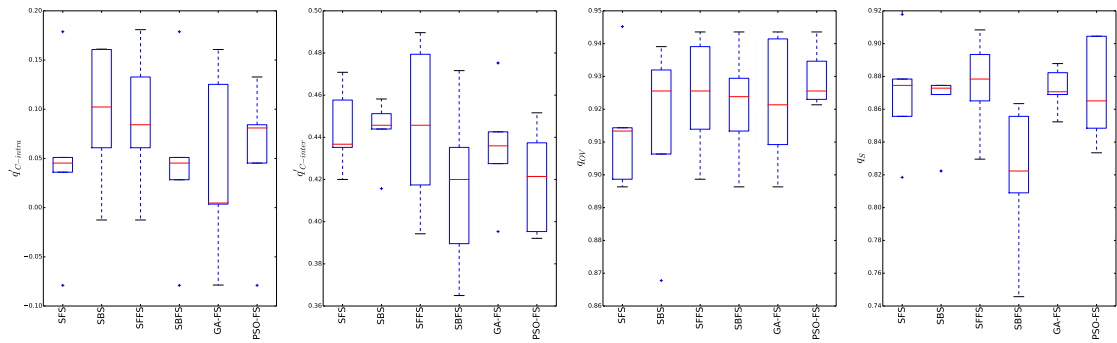


FIGURE 65: AFS performance comparison of the "wdbc" data set

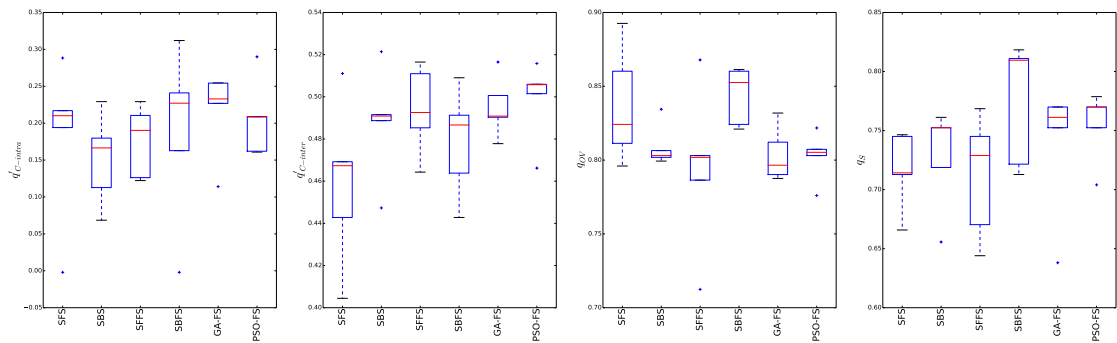


FIGURE 66: AFS performance comparison of the "ionosphere" data set

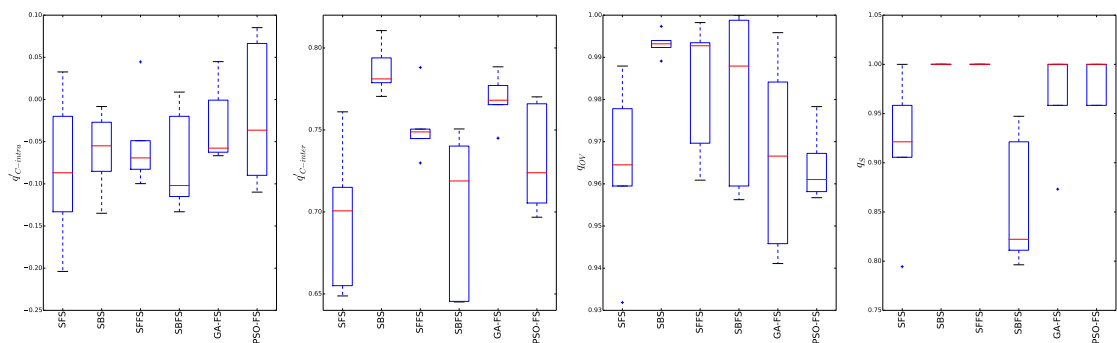


FIGURE 67: AFS performance comparison of the "yeast" data set

TABLE 15: Average number of iterations used in the AFS methods investigation

Data Set	Method					
	SFS	SBS	SFFS	SBFS	GA-FS	PSO-FS
iris	12(0.0)	12(0.0)	40.20(0.2)	38.60(0.2)	100(0.0)	100(0.0)
wine	156(0.0)	156(0.0)	229.4(3.7)	242.3(1.1)	100(0.0)	100(0.0)
ionosphere	1,056(0.0)	1,056(0.0)	1,179(14)	1,092(14)	100(0.0)	100(0.0)
wdbc	380(0.0)	380(0.0)	548.1(8.0)	566.8(5.6)	100(0.0)	100(0.0)
yeast	6,162(0.0)	6,162(0.0)	7,089(22)	6,986(14)	100(0.0)	100(0.0)
mechatronic	552(0.0)	552(0.0)	593.2(8.2)	612.4(11)	100(0.0)	100(0.0)

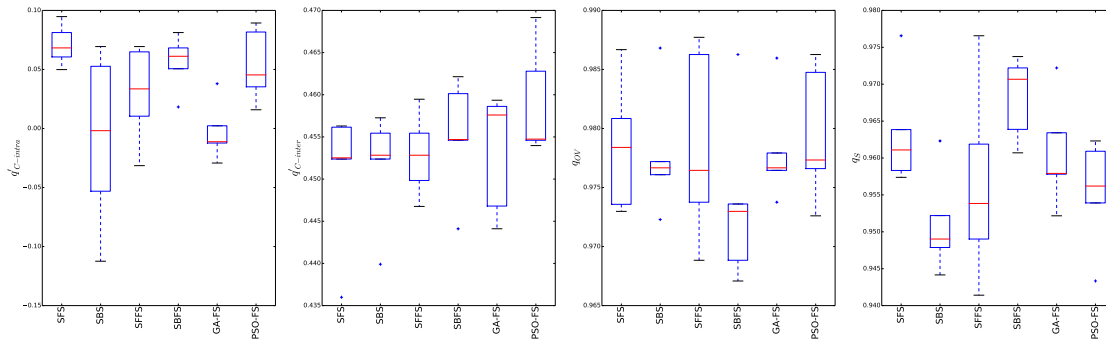


FIGURE 68: AFS performance comparison of the "Mechatronic" data set

Automated Feature Weighting Automated Feature Weighting (AFW) [41] is an alternative method for DR process in DAICOX, that provides a finer scale of selection compared to the binary selection approach. The implementation of AFW method employs meta heuristic search approach due to potentially large search space as mentioned earlier in Chapter 4, Section 4.4.3. The implementation of AFW is illustrated by software flow structure in Fig. 69. At present, PSO or GA can be selected at the optimization module to perform the search procedure responsible for weight adjustment task. The extension of currently implemented AFW with alternative optimization methods can be made by only introducing new optimization modules into the proposed flow while the rest of the process remains the same. A normalized data set is required in order to obtain commensurate assessment of weighted data. Thus, normalization coefficients are vital information for further uses of the designed AFW module, e.g., in operation time, which can be applied later using the implemented normalization module. The weight data module applies a given weight vector generated from the optimizer to create a weighted data table before being evaluated by selected assessment criteria in the object method caller module. The best solution will be exported in form of weight vector along with normalization coefficients. A reduction scheme is also implemented by using a pre-defined threshold value, that eliminates features associated with weight value below the threshold assuming low contribution to feature space quality. The threshold can also be

included into the optimization loop allowing concurrent evolution together with feature weights.

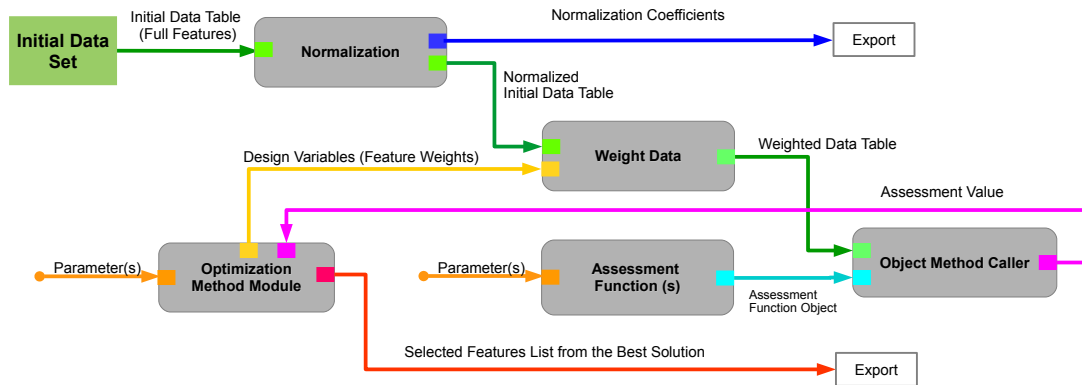


FIGURE 69: Operation of the AFW method module in DAICOX

TABLE 16: Parameter settings for stochastic AFW methods

Method	Parameters and Settings
GA-FW	Population Size = $10 + 2\sqrt{d}$, Max. Iteration = 50, Selection Rate = 0.5, Mutation Rate = 0.2, Weight Threshold = 0.1
PSO-FW	Population Size = $10 + 2\sqrt{d}$, Max. Iteration = 50, $C_1=1.2$, $C_2=1.2$, $w=1.0$, Weight Threshold = 0.1

d is the number of features of initial data.

5.3 Classification

The DAICOX architecture provides a number of classification methods implemented in three categories; multi-class classification, one-class classification, and hierarchical classification. For SVM based methods, the implementation extended from the LIBSVM [175] open-source library. The overview of implemented classification method taxonomy is depicted in Fig. 70. All methods are provided in form of a modular library and compatible with the DAICOX library interface. Therefore, the combination of classification task with other processing parts, e.g., signal processing, and dimensionality reduction can be made with small effort. In particular, the optimization of parameter settings can be effectively performed by DAICOX design optimization with choice of available optimization modules.

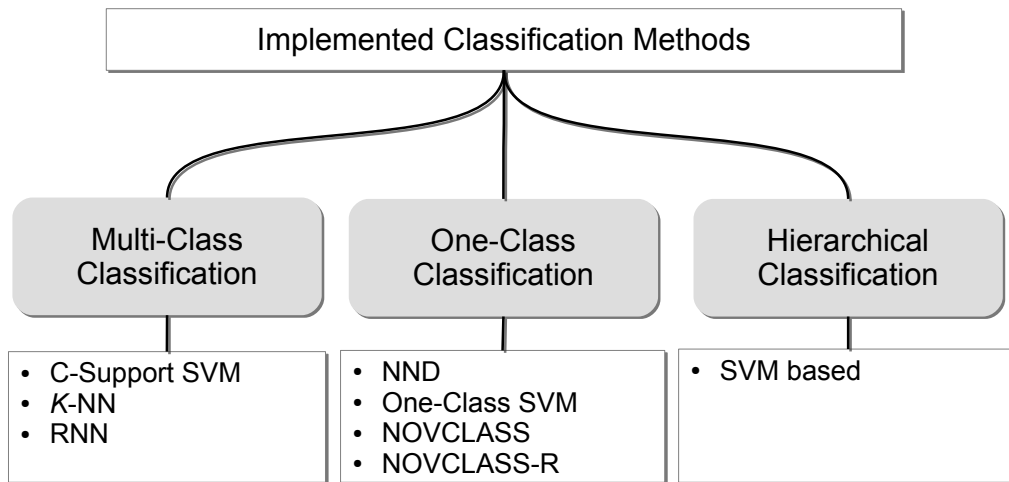


FIGURE 70: Implemented classification methods

5.3.1 Multi-Class Classification

To solve multi-class classification problems, DAICOX provides standard classifier modules from existing libraries with capability of inclusion to DAICOX design optimization. A number of Python-based classification libraries, e.g., Scikit-learn or Orange can be easily added to the method pool by using library wrapper. A library wrapper manages data format translation between DAICOX supported data format (Orange data table) and the format of an external library. Parameter optimization can be performed by available optimization method modules by describing parameter information in the parameter descriptor, whose more detail can be found in Section 5.4.1. Currently available multi-class classification methods are k -NN, RNN, and C-SVM. The choices of kernel function for C-SVM includes linear, polynomial, and Gaussian RBF kernel, which can also be incorporated to the optimization process for selection.

A generic automated classification design is based on software flow block diagram illustrated in Fig. 71, which is the baseline for classifier design in experiments and case studies presented in the next chapter. An input data set is split into 2 subsets for training and testing processes by using hold-out random sampling technique. The training data set is again split into k subsets by using k -fold cross validation for model analysis in order to obtain high generalization performance. At every optimization iteration, classifier parameters generated from employed optimizer will be applied to create classifier objects, each uses data from selected fold. The best performance classification object will be exported at the end of optimization process for run-time uses. Parameter searching ranges for optimization process are given in Tab 17. The ranges can be adjusted by the designer in order to constrain optimization in a potential search space.

TABLE 17: Detail of parameters and searching range for classification optimization

	Method	
	k -NN and RNN	C-SVM
Parameters	$k \in 1, 2, 3, \dots, N - 1$ where N is the number of training example per class	$C \in [1, 10000]$ $\gamma \in [1e^{-5}, 10]$ (100 discrete steps for Grid Search)

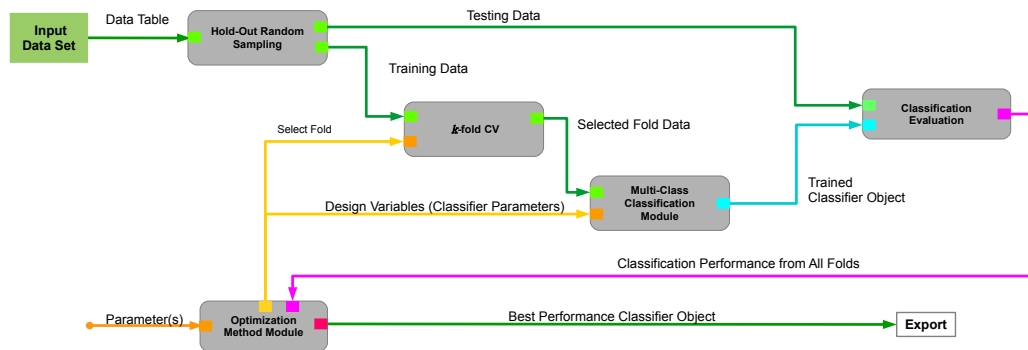


FIGURE 71: Classification implementation in DAICOX framework

5.3.2 One-Class Classification

Four methods are selected and implemented for One-Class Classification (OCC); NN-D, One-Class SVM (OC-SVM), NOVCLASS, and its variation NOVCLASS-R. All methods are implemented in DAICOX classification module with similar software flow to multi-class classification as depicted in Fig. 71 except that a given data set for training will be regarded as a single positive class and a classification result is in boolean format. Thus, to classify a specific class from a multi-class data set the training data should be filtered into sub-data sets of a particular class. Class specific data filtering function is supported DAICOX to facilitate OCC application designs. OC-SVM uses the ν -SVM model from the LIBSVM library for implementation.

A variation of NOVCLASS is proposed by incorporating part of iterative training mechanism [149] to reduce the required number of the prototypes and dynamically adjust their radii. The additional training steps extended to the original NOVCLASS are described in algorithm 1. This variation is denoted as NOVCLASS-R, where R refers to the prototype reduction mechanism.

The preliminary study of the implemented one-class classification methods has been conducted with artificial data sets. To investigate performance in several data characteristic,

Algorithm 1: NOVCLASS-R Training Process

```

Given  $N$  samples training data set  $X_{train}$ ;
Randomly chose a sample  $\mathbf{x}$  from  $X_{train}$  as the first Ref. vector.  $\mathbf{t}_1$ ;
Compute distance matrix  $D$  of  $X_{train}$ ;
 $r_1 = \max.$  distance of 1-NN of all samples in  $X_{train}$ ;
 $R = \{r_1\}$ ;
 $T = \{\mathbf{t}_1\}$ ;
for  $\mathbf{x}_n \in X_{train}$  do
    for  $\mathbf{t}_i \in T$  do
        if  $d(\mathbf{x}_n, \mathbf{t}_i) > r_i$  then
            add  $\mathbf{x}_n$  to  $T$  as  $\mathbf{t}_{i+1}$ ;
             $r_{i+1} = d(\mathbf{x}_n, \mathbf{t}_i)$ ;
        end
    end
end

```

four data sets are created based on general paradigms of information distribution, which are frequently found on real-world situations. The information of the data sets are explained in Table 18 as well as the visualization of their feature space. Further, real-world data sets are also used to investigate the classification performance of the implemented methods.

The classification performance of OCC method can be evaluated and optimized by using Receiver-Operating-Characteristic (ROC) [176] analysis techniques. The ROC curve of a one-class classifier model indicates the classification characteristic within an interval of threshold value θ . The threshold interval starts from the value where all patterns are accepted and ends at the value where all patterns are rejected. Soft classification output, which is the probability estimation of a pattern to the positive or normal class, is used to compute an ROC curve. A perfect classifier represents a rectangle shape of ROC curve, which refers to 100% correct acceptance rate, i.e., True Positive (TP) with 100% correct rejection rate, i.e., True Negative (TN) as indicated at the top right corner of Fig. 72. The Area Under Curve (AUC) is used to measure the quality of a classifier, where in the perfect case the area is equal to 1.0. The smaller number of AUC indicates higher number of misclassification. Given a test data set X with N positive patterns X_{pos} and M negative patterns X_{neg} , the TP rate and the TN rate are computed as in Eq. 5.2 and Eq. 5.3 respectively, where the function I returns integer 1 for true case or 0 for false case. Practically, some negative patterns or outliers will be accepted and vice versa for positive patterns as represented in the blue curve in Fig. 72.

$$TP = \frac{1}{N} \sum_{i=1}^N I(p(\mathbf{x}_i) > \theta), \mathbf{x}_i \in X_{pos} \quad (5.2)$$

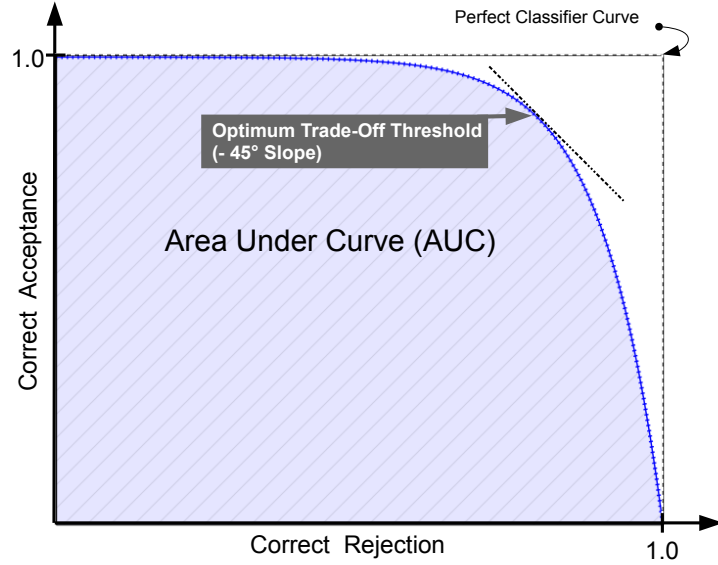


FIGURE 72: ROC analysis for one-class classification method

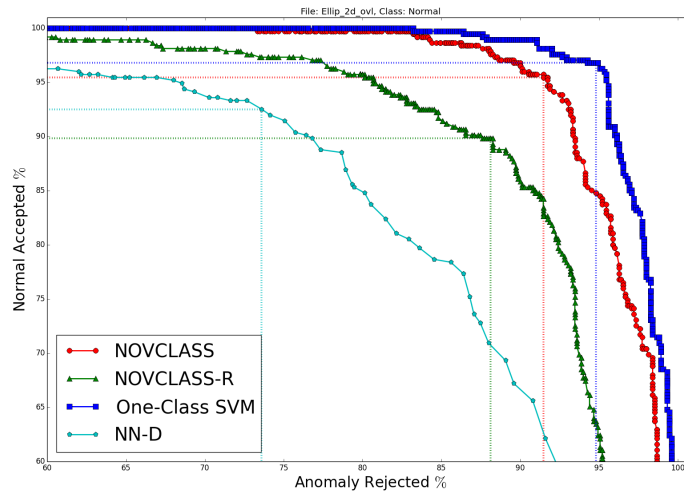


FIGURE 73: Computed ROC curves of the "moon" data set

$$TN = \frac{1}{M} \sum_{i=1}^M I(p(\mathbf{x}_i) < \theta), \mathbf{x}_i \in X_{neg} \quad (5.3)$$

The ROC analysis can also be used to find an optimum threshold value in case a trade-off between TP and TN is necessary. The best threshold value is where the ROC curve has -45 degree slope indicating the optimal trade-off between TP and TN , which has minimum structural risk of the model. An example of determined threshold values, which are selected by considering the slope on curve, AUC, TP , and TN , is illustrated in Fig. 73.

For the synthetic data experiments, the classification performance results are given in Table 19 and the computational performance are given in Table 20. Table 21 contains

classification results from the real-world data set. All results value are an average of ten runs. At each run, 5-fold cross validation was employed for model analysis and selection. The ROC plots of the results from all data set also given in Fig. 74 to Fig. 79. For OC-SVM method, the parameter ν was set to 0.5 and the parameter γ was set to 0.01 in all cases. The scaling factor parameter η for NOVCLASS and NOVCLASS-R was set to 1.0 in all cases. NOVCLASS-R has significantly reduced number of reference prototypes compared to the original NOVCLASS as shown in Table 20 indicating the effectiveness of the proposed prototype reduction mechanism.

TABLE 18: Synthetic data sets description. In the feature space plots, the blue dots are the positive class instances, the red dots are outliers. The boundaries between positive clusters and outliers are equal to the maximum 1-NN distance of the positive instances. All data sets contain 500 samples of positive class and 1,000 samples of negative class.

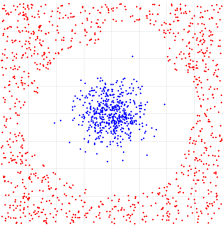
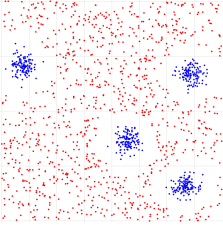
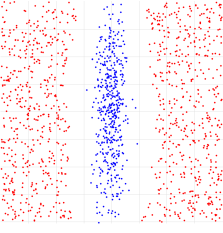
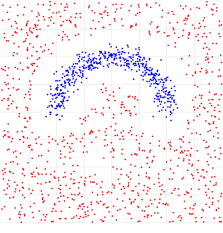
Feature Space	Data Set Name	Description
	GausNorm	Single data cluster with normal distribution, a unit covariance matrix is used.
	GausMulti	Multi-Modal of Gauss-Normal model. This mimics the scenario of a multi-class data are affiliated into a single positive class
	Elliptic	Simulates data with different variance in each feature.
	Moon	The more complex case than the Elliptic. Data is uniformly distributed in a half circle with normal distribution representing non-convexity.

TABLE 19: Classification performance comparison of implemented OCC methods

Performance	Classification Method	Data Set			
		GausNorm	GausMulti	Elliptic	Moon
AUC	NN-D	0.842 (0.012)	0.955(0.031)	0.927 (0.022)	0.900 (0.009)
	OC-SVM	0.993 (0.089)	0.560 (0.203)	0.946(0.051)	0.987(0.012)
	NOVCLASS-R	0.961(0.020)	0.976(0.066)	0.972(0.034)	0.943(0.086)
	NOVCLASS	0.976(0.042)	0.984 (0.081)	0.981 (0.100)	0.978 (0.075)
TP rate	NN-D	0.800 (0.032)	0.949 (0.046)	0.931 (0.019)	0.925 (0.023)
	OC-SVM	0.955(0.045)	0.960(0.026)	0.941(0.068)	0.979 (0.076)
	NOVCLASS-R	0.984 (0.076)	0.971(0.075)	0.984 (0.022)	0.899(0.040)
	NOVCLASS	0.976(0.009)	0.987 (0.015)	0.984 (0.025)	0.955(0.092)
TN rate	NN-D	0.799 (0.100)	0.828(0.064)	0.623 (0.071)	0.736 (0.030)
	OC-SVM	0.965 (0.099)	0.328 (0.087)	0.851(0.064)	0.917 (0.132)
	NOVCLASS-R	0.867(0.019)	0.904(0.087)	0.893(0.49)	0.881(0.050)
	NOVCLASS	0.915(0.076)	0.932 (0.058)	0.923 (0.041)	0.915(0.030)

TABLE 20: Computation performance comparison

Performance	Classification Method	Data Set			
		GausNorm	GausMulti	Elliptic	Moon
t_{train} [s]	NN-D	6.384(0.1)	6.654(0.1)	6.698(0.2)	7.067(0.1)
	OC-SVM	5.491(0.9)	5.73(0.8)	5.744(0.9)	6.107(0.7)
	NOVCLASS-R	3.233(0.1)	2.99(0.1)	3.167(0.2)	3.348(0.1)
	NOVCLASS	3.346(0.0)	3.410(0.0)	3.487(0.0)	3.698(0.0)
\bar{t}_{class} [μ s]	NN-D	448.0(0.1)	418.7(0.1)	373.3(0.0)	416.0(0.1)
	OC-SVM	64.00(2.1)	69.33(3.0)	64.00(3.7)	69.33(1.5)
	NOVCLASS-R	90.67(18.8)	125.3(24.0)	101.3(29.2)	117.3(32.1)
	NOVCLASS	233.1(120)	230.5(90.1)	238.0(143)	228.0(101)
Stored Prototypes	NN-D	125(0.0)	125(0.0)	125(0.0)	125(0.0)
	OC-SVM	4.2(0.5)	52.1(1.2)	52.5(1.7)	28.4(0.9)
	NOVCLASS-R	8.7(0.6)	17.1(1.3)	9.0(0.9)	13.7(1.4)
	NOVCLASS	125(0.0)	125(0.0)	125(0.0)	125(0.0)

TABLE 21: Real-World data set results

Data Set	Class	Classification Rate (TP, TN)			
		NN-D	OC-SVM	NOVCLASS-R	NOVCLASS
iris	1	(1.00, 1.00)	(1.00, 1.00)	(1.00, 1.00)	(1.00, 1.00)
	2	(0.56, 0.94)	(1.00, 0.90)	(1.00, 0.94)	(0.96, 0.90)
	3	(0.96, 0.56)	(0.92, 0.98)	(0.92, 0.86)	(0.88, 0.90)
wine	1	(0.73, 0.90)	(1.00, 1.00)	(0.93, 0.90)	(0.90, 0.88)
	2	(0.49, 0.72)	(0.91, 0.78)	(0.91, 0.59)	(0.94, 0.46)
	3	(0.75, 0.69)	(0.96, 0.97)	(1.00, 0.42)	(0.92, 0.60)
ionosphere	1	(0.79, 0.66)	(0.74, 0.89)	(0.96, 0.70)	(0.24, 0.96)
	2	(0.88, 0.87)	(0.89, 0.94)	(0.99, 0.64)	(0.93, 0.88)
wdbc	1	(0.75, 0.62)	(0.94, 0.87)	(0.84, 0.89)	(0.89, 0.89)
	2	(0.11, 0.95)	(0.00, 1.00)	(0.16, 0.95)	(0.13, 1.00)
yeast	1	(0.94, 0.25)	(0.94, 0.97)	(0.94, 0.99)	(0.89, 1.00)
	2	(1.00, 1.00)	(1.00, 0.94)	(1.00, 0.97)	(1.00, 1.00)
	3	(1.00, 1.00)	(0.98, 1.00)	(0.98, 1.00)	(1.00, 1.00)
mechatronic	1	(1.00, 0.99)	(1.00, 0.99)	(1.00, 0.99)	(1.00, 1.00)
	2	(0.91, 0.78)	(0.92, 0.93)	(0.67, 0.84)	(0.90, 0.83)
	3	(0.98, 0.85)	(0.97, 0.97)	(0.99, 0.84)	(0.86, 0.95)
	4	(1.00, 1.00)	(1.00, 0.98)	(0.99, 0.99)	(0.99, 0.99)

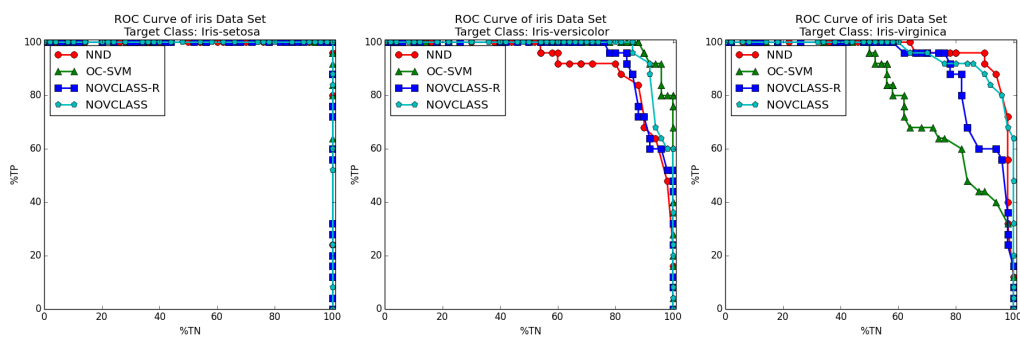


FIGURE 74: Computed ROC curves of the "iris" data set

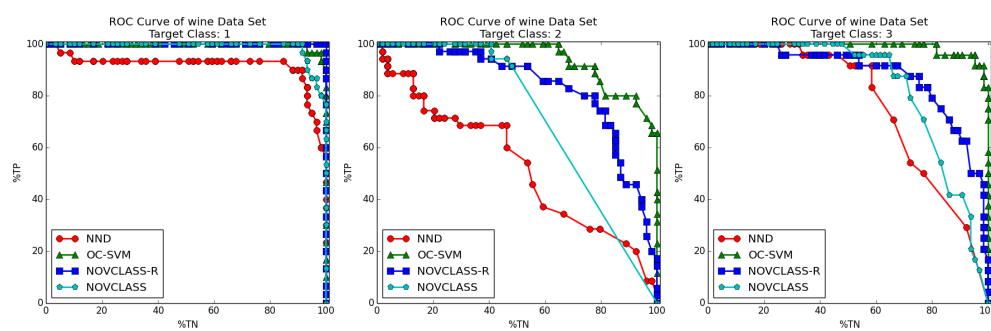


FIGURE 75: Computed ROC curves of the "wine" data set

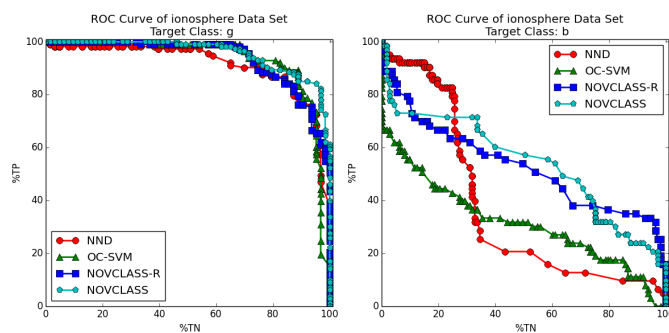


FIGURE 76: Computed ROC curves of the "ionosphere" data set

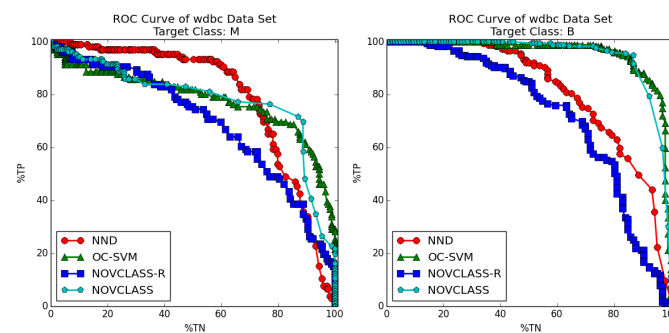


FIGURE 77: Computed ROC curves of the "wdbc" data set

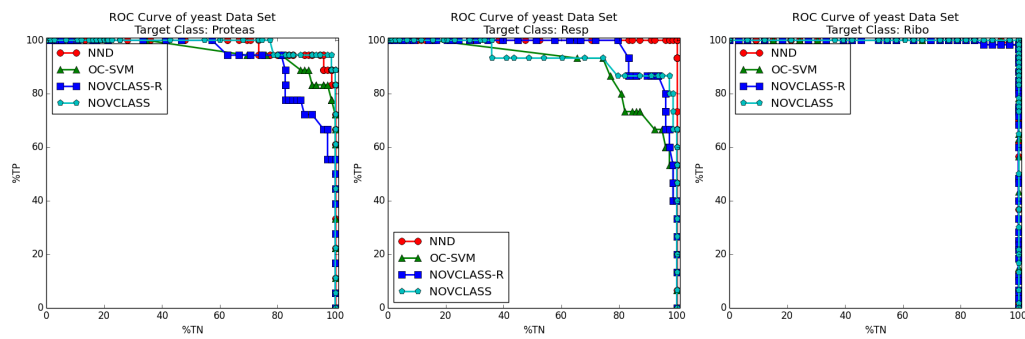


FIGURE 78: Computed ROC curves of the "yeast" data set

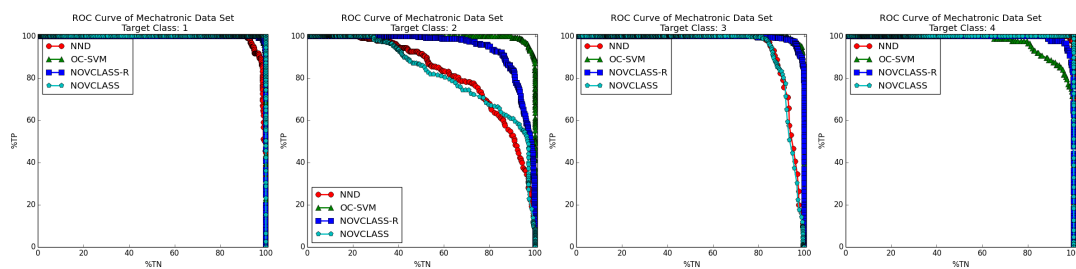


FIGURE 79: Computed ROC curves of the "mechatronic" data set

All implemented methods have performed with considerably good performance in the synthetic data set except only one case of OC-SVM with "GausMulti" data set. This is due to the fact that SVM may face a problem when handling data with multiple clusters. However, OC-SVM outperformed other methods in several cases with fastest classification time in all cases. The proposed NOVCLASS-R has close performance to its original, in contrast, providing light weight classifier model with significantly reduced number of stored prototypes as well as faster classification time. Considering designing time, the NOVCLASS family is the least training time consumption method. The fastest was achieved by NOVCLASS-R. This may contradict to what was reported for NN-D that it takes no training time. But in practice, when creating a classifier, the model analysis should be performed. In these experiments 10-fold cross validation was employed in the training process. Thus, a number of classifications of each model was performed for evaluation. NN-D takes longest classification time in almost all cases, which also causes the longest training process among the others. This is due to its classification procedure, which has to compare all stored prototypes to each given unknown pattern to find the nearest neighbor. In contrast, NOVCLASS sequentially observes each prototype and classifies an unknown pattern as positive if it is inside the prototype's radius. Thus, the maximum bound of the classification time is equal to NN-D in the

case of a given pattern is negative, but, faster in dealing with positive patterns. The results from the real-world data sets have proven applicability and effectiveness of the implemented methods in practical scenarios with perfect classification in several cases. This will guarantee the potential capability of using the implemented OCC methods for real applications given in the next chapter.

5.3.3 Hierarchical Classification

From the proposed background elaborated in the previous chapter in 4.5, the Hierarchical Classification (HC) approach has been implemented in DAICOX based on C-SVM algorithm. The proposed hierarchical classifier is a multi-sensor fusion architecture at the decision level. It consists of multiple classifier and operator modules as illustrated in Fig. 80 instantiating a 3 sensory channels HC structure using C-SVM (H-SVM). In contrast to the complexity of the flow, the constructed modules operate and represent as a single multi-class classification module, thus, taking as similar design effort as a conventional flat classification approach. Each classifier in the first level generates a soft-classification output, which is a vector of probability estimations of a given pattern to each class. The final classifier performs multi-sensor fusion at decision level from a concatenated soft-classification vector predicting a class associated with the vector, i.e., class label. Any classifier methods, which are able to generate soft-classification output, can be used in the proposed HC by replacing the classifier modules in Fig. 80. Model selection and optimization can be conducted during the design by using the same setup used for the multi-class design problem given in Fig. 71. The uses as well as performance discussion of the proposed HC are given in the experiment and case-studies in the next chapter.

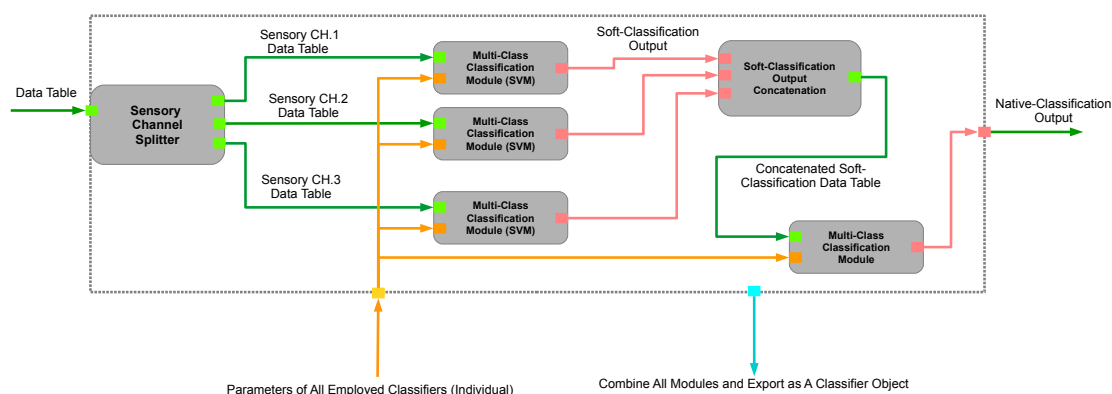


FIGURE 80: Implementation of hierarchical classification module

5.4 DAICOX Design Optimization

As already mentioned in previous sections, the DAICOX architecture provides choice of optimization methods in designing of every standard building block. In this section, the details of optimization software structure are explained along with the concept of flexible optimization interface, which enables applicability of any implemented optimization methods in all design procedures. The DAICOX design optimization is elaborated with meta-heuristic optimization approaches and multi-objective assessment capability. Currently implemented optimization methods comprise simple Exhaustive (Grid Search), Genetic Algorithms (GA), and Particle Swarm Optimization (PSO). With the modular method concept, DAICOX allows and facilitates the inclusion of new optimization method implementation. The DAICOX library interface also utilizes the full exploitation of a method as it can be used in all design of the standard building blocks.

5.4.1 Flexible Design Variable Evaluation

Usually, an implementation of optimization is task specific and the optimization procedure is integrated in part of the processing method. The complication arises when one wants to reuse the implemented optimization routine for other design problems, which may ultimately lead to tedious and redundant tasks. For this reason, the modular optimization method concept is proposed with flexible interface between optimization routine and a particular design problem using the DAICOX library interface. The main role of this concept is the transformation layer depicted in Fig. 83, which associates information exchange between method module of a design problem and optimization operator. A method module reports information about its available parameters, precision types, and value ranges using the design variable descriptor, the common format of design variable representation as shown in Fig. 82. The transformer is in charge of converting and scaling each given optimization design variable from the optimization operator using the precision and value range of the parameters it is associated with. An employed optimization procedure is allowed to modify the values of the "Params" vector during search procedure. The method module applies the parameters at every change in the "Params" vector to generate a method object, which will be used for an evaluation process. Evaluation result of each given design variable returns to the optimization operator (normalized real value in the interval of [0,1]) as the information to modifying the solution candidates for next iteration. As a result, an implemented optimization algorithm can be utilized to several design tasks without modification effort.

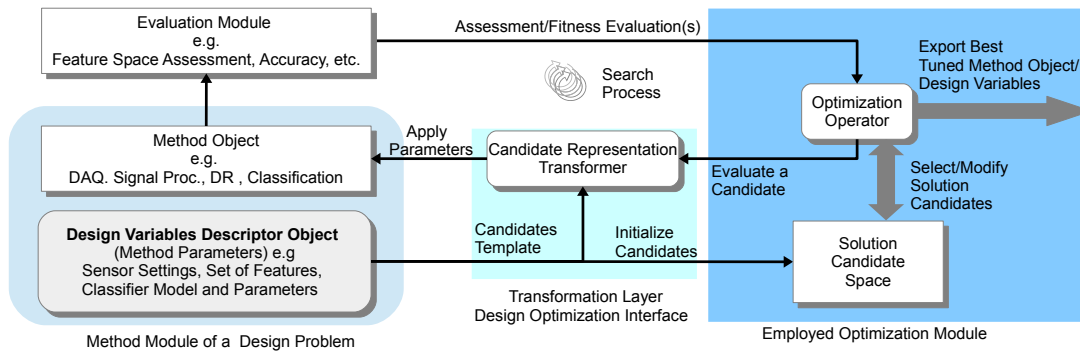


FIGURE 81: Flexible design optimization interface

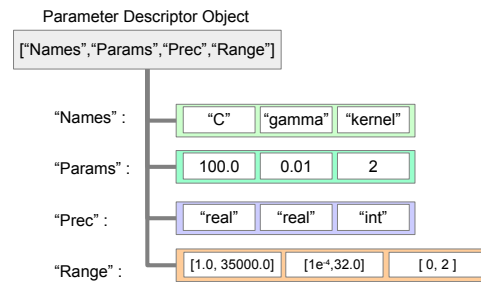


FIGURE 82: Example of design variable descriptor of the C-SVM module

5.5 DAICOX Software Environment

Software development for DAICOX architecture is based on Python programming language as to support the aims of open-source and multi-platform software. Core computation of each processing method represents as a python module script, thus, denoted as method module. DAICOX library is a collection of the available method modules structured in Python standard name space, i.e., Python package. DAICOX toolboxes comprise the linked methods from the DAICOX library by using widget wrappers for visual programming environment in the Orange Canvas software. The widget wrappers are scripts, that load method modules and mask the interface and signal control into visual programming workspace for the purpose of intuitive design activity. In addition, the use of the DAICOX library is independent from the Orange Canvas a designer can also design a processing chain manually using Python script. This also allows the creation of an executable script of a designed processing chain using visual programming. The overview of the DAICOX library and toolbox software structure is shown in Fig. 83. Some library developed in other languages, e.g., C or C++, can also be included to the DAICOX library by using a library converter, for C/C++ cases, BOOST [177] is

employed. The BOOST library converter is a interpreter between C++ source codes and the Python programming language, which allows the integration of ISE's previous C/C++ based works into DAICOX architecture.

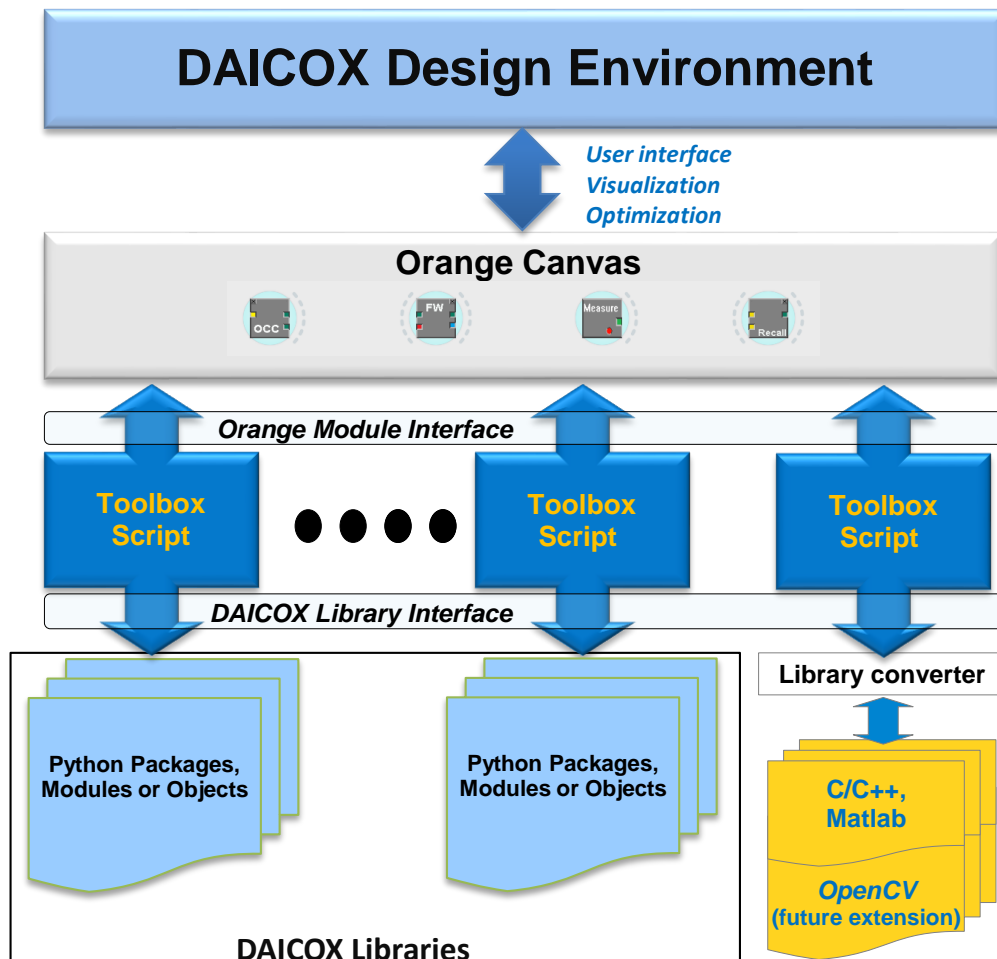


FIGURE 83: DAICOX software library and visual programming toolbox interface

5.5.1 DAICOX Library Interface

The proposed modular software requires consistent information exchange for every pair of interconnected modules in order to provide seamless operations in a processing chain. DAICOX Library Interface (LI) is a unified mechanism, that governs information exchanges between method modules. Major aim of the interface is to ensure relevant inter-module communications and facilitate the extension of each new library module.

Two attributes are described for each method module including the Input field, which provides required input information and the Output field, which provides the type of possible output information. The descriptions of LI for all method types are given in Table 22. Proper operation can be ensured for the development of new method module by regarding the DAICOX library interface.

TABLE 22: DAICOX library interface descriptions

Module Type	Input	Output
Data Acquisition	Parameters, Commands	Orange Data Table
Signal Processing	Orange Data Table, Parameters	Method OBJ, Parameter Descriptor
Assessment Function	Parameters	Measure OBJ
Feature Selection	Orange Data Table, Measure OBJ	Selected Orange Data Table, Selected Feature List
Classification	Orange Data Table, Parameters	Method OBJ, Parameter Descriptor
Designer	Method OBJ(s)	Designed Method OBJ, Visualization
Optimization	Method OBJ, Parameter Descriptor, Measure OBJ	Optimized Method OBJ

*all OBJ are referred as a Python object.

5.5.2 Choice of Visualization

DAICOX provides a number of visualization features in several representations, e.g., signal, feature space, and performance statistic. The signal representation visualization takes direct information from data table to visualize in corresponding domain, e.g., time signal, frequency domain, and symbolic data. For heterogeneous multi-sensor information, palette visualization concept proposed in [123] is an intuitive choice for multiple signal representations. DAICOX is integrated with feature space visualization of multi-dimensional data. Mapping methods are required to transform high-dimensional data into a 2D visualization space. Several approaches are available to be selected including linear method using PCA and LDA or the more powerful non-linear MDS based on distance-preserving using Sammon Mapping is the preferable choice, which can be incorporated with the available DAICOX optimization modules. These methods are included in the standard library of Orange. In design optimization, performance information visualization, e.g., assessment versus evolution plot, can be used to analyze the

quality of designed solution and further improvement or fine-tuning can be conducted by the designer.

5.5.3 DAICOX Application Design Flow

Figure 84 illustrates the overall design flow for a cognition application using DAICOX architecture. The flow represents standard procedures in the design of multi-sensor cognition application from front to back, which is the baseline for the designs in experiments and case studies given in the next chapter. Requirements or specifications are used to derive an initial design, which can be created by using Orange Canvas a visual programming software by selecting method module and connect their signal to form a processing chain, i.e., workspace. It is also possible to create a processing flow using Python script. A set of previous design solutions stored in the design database can be reused as a template, when starting a new application design of similar task, hence, gaining the knowledge from previous experience and significantly saving design time and effort. Design optimizations take place after all method modules are properly given with their required input, then sequential evolutions will be conducted. The sensory database provide recorded data as data source for the design optimization in off-line mode. A hardware instance, e.g., data acquisition can also be used to provide fresh measurements as data source for design optimization. Reconfigurable capability of the hardware instance can realize the intrinsic optimization, which will be pursued next. The designer can perceive information of the design process via choice of visualizations to analyze the design solutions for making a decision on the final output solution. The final solution will be used to generate a Python executable script, which performs the optimized processing tasks independently without requirement of the Orange Canvas. The main purpose for Python executable script generation is the integration of the designed solution with Graphic User Interface (GUI). The more detail on GUI integration in several application cases will be given in the next chapter. The final solution will also be stored in the design database for amendment purpose of current version and for new application design.

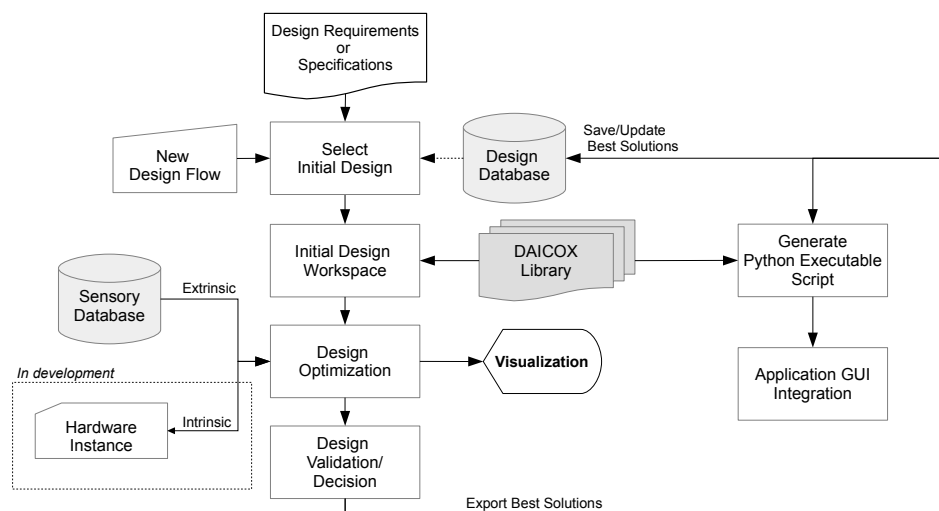


FIGURE 84: Application design steps using DAICOX architecture

5.6 Discussion

A number of processing methods have been implemented in this chapter to contribute to the proposed DAXICOX architecture. Modular software library concepts have been proposed to provide high degree of flexibility in both usage and development. The implemented methods serve all functions in the standard building blocks including, signal processing and feature computation, dimensionality reduction, and classification. In particular, four one-class classification methods for novelty/anomaly detection have been implemented and tested with considerably good performance. All implemented methods can be optimized by the proposed DAICOX design optimization with choice of optimization algorithms and multi-objective optimization capability. Software environments of DAICOX provides intuitive link to the designer with design database, visual programming, and choices of visualization. The complete processing chain can be exported into a single executable file for run-time purpose and user interface integration. All the features and currently available methods of DAICOX architecture support the design of intelligent multi-sensor system with reduced effort and design time. The comprehensive experiments and case studies are given in the next chapter to quantitatively demonstrate capability and effectiveness of the current DAICOX architecture in generic real-world sensor application designs.

Chapter 6

Realization of DAICOX Design and Integration: Case Studies and Applications

The currently implemented DAICOX architecture has been utilized in three advanced application design scenarios presented in this chapter. The main aim of each application design task is to demonstrate the capabilities as well as performance of automating the design process and delivering good quality solutions with less design effort than the human designer. The design process and design optimization have been performed by using implemented DAICOX libraries, toolboxes, and frameworks presented in the previous chapters. The first application design case study deals with basic food analysis application employing an autonomous multi-sensor device and an advanced pattern recognition processing chain. Driving assistance system is the focus area of the second case study, where heterogeneous multi-rate sensor information are fused for on-line identification of driver drowsiness status. The last case study employed DAICOX in assisting the design of pattern recognition based edge detection for timebase self-synchronization mechanism of wireless sensor nodes in a magnetic localization system.

6.1 Automated Application Design of LoX

The interest in food analysis has been growing significantly as a result of numerous situations of food scamming, contamination and adulteration. This motivates the development of Lab-on-X (LoX) [26, 178] devices to serve in several applications related to living assistance system, interactive health care, and food quality/authenticity checking. The LoX devices are autonomous intelligent multi-sensor data acquisition units for the E-Taster [179] assistance system, an intelligent environment for on-line cooking assistance. Thus, the information processing architecture as shown in Fig. 85, which comprises of data acquisition, sensory database, and pattern recognition tasks, e.g., dimensionality reduction and classification, is conceived. Therefore, the design of a processing chain for the E-Taster application involves a number of complex, knowledge requiring, and time consuming tasks. To facilitate the E-Taster application design tasks, the currently implemented DAICOX architecture has been used in this case study with the main focus in food substance classification applications. The LoX technical cognition application design steps using DAICOX are illustrated in Fig. 86. The beginning step is the data collecting procedure, where the data sets used during the design can be obtained from two sources; sensory database of previous recorded data sets and fresh measurements from a connected LoX device. The latter case also allows on-line preliminary data analysis before committing the new data into the sensory database as well as performance analysis of a complete processing chain. The implementation, and experimentation in all steps of the design flow are explained in detail in the following subsections.

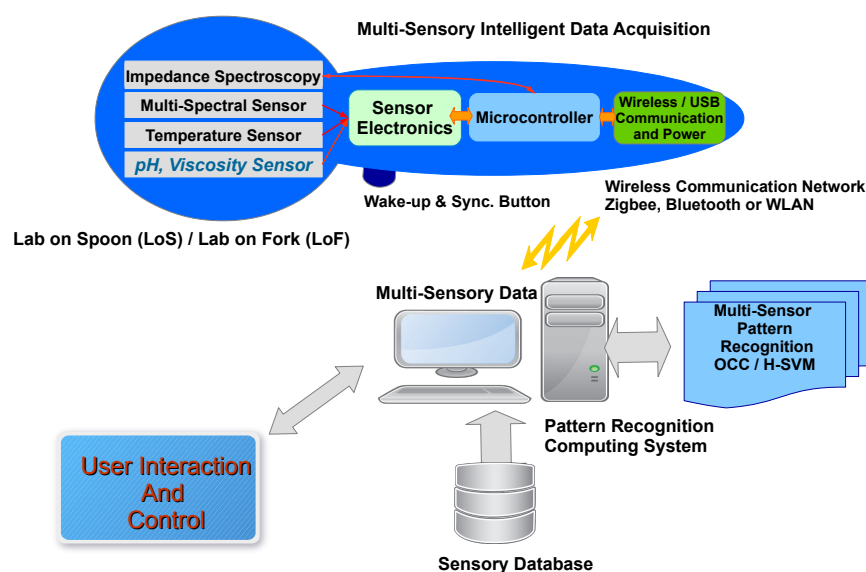


FIGURE 85: LoX device based application operational architecture

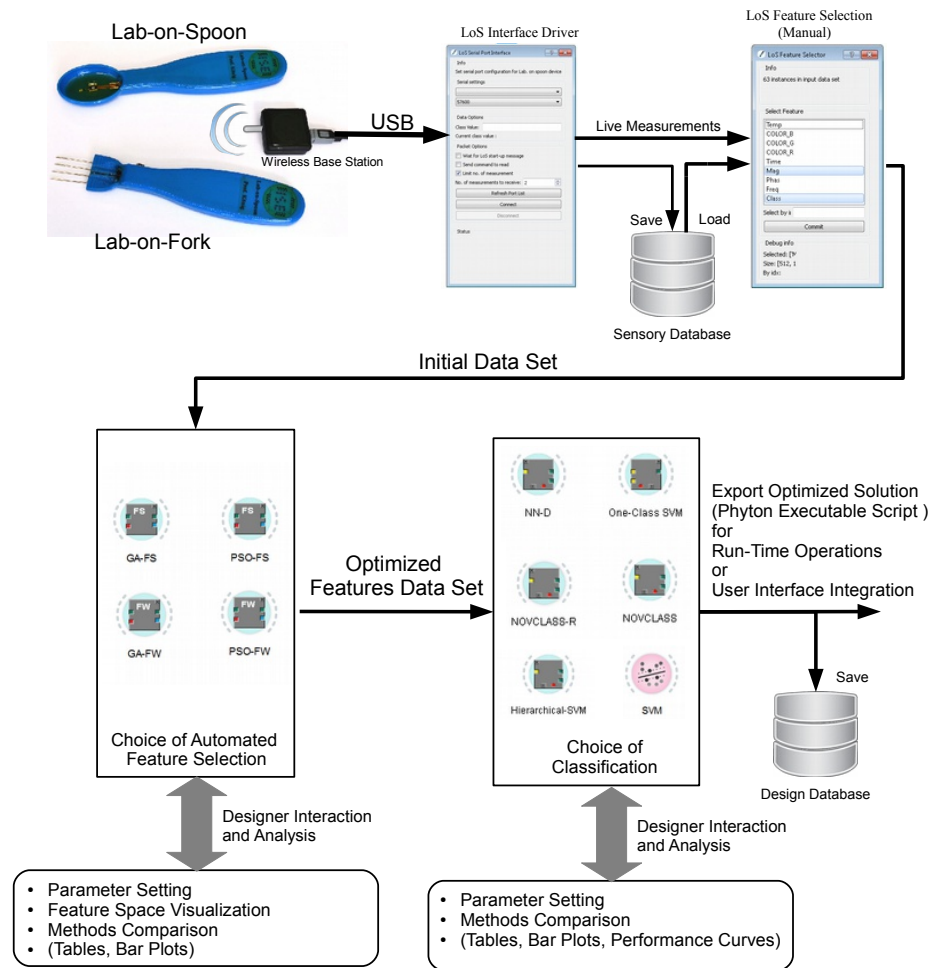


FIGURE 86: LoX food ingredients recognition application design overview

6.1.1 Data Acquisition

LoX devices are realized as embedded autonomous measurement units for delivering multi-sensory context to a soft-computing host machine. The hardware block diagram of the currently implemented LoX is depicted in Fig. 87. The current sensor palette includes a pt 10k temperature sensor with a custom calibrated front-end, a MAZeT MMCS6 [180] multi-color sensor with a configurable transimpedance amplifier MCD04 [181] and active illumination, and the AD5933 [182] embedded network analyzer for impedance spectroscopy measurement applied with gold-plated electrodes. The impedance spectroscopy requires a standard analog front-end (AFE) for low- as well as high impedance measurements wide variety of measurable types of liquid. Figure 88 shows examples of acquired impedance information of the SoyVine data set in both magnitude and phase curves. The multi-sensory information of a liquid acquired by a LoX device represents a signature, which can be identified by using a pattern recognition process. Measurement readout and control and host communication are in the

responsibility of an Arduino MCU board. Two communication options, wired via USB port and wireless via IEEE802.15.4 standard using XBee module, are available. The hardware configurations and parameters of current LoX devices are given in Table 23. Calibration can be intuitively achieved by clamping an external reference resistor to the measurement electrodes or by using reference liquid. The current development of self-x properties integration is pursuing towards dynamic reconfiguration capability of the LoX hardware instance. This integration will allow complex functions in dealing with sustainability and robustness issues, e.g., self-monitoring, self-calibration, and self-trimming, thus, maintaining long-term quality and performance.

The information of a measurement acquired by a LoX device results in 1,040 features for the currently implemented LoX devices, which will be denoted later as MS-LoX. In the previous development of LoX, the color data were obtained from an RGB sensor, thus, resulting in 1,027 features. The design sensory database is the collection of recorded data sets from several food classification scenarios as described in Table 24. In the experiments, the data sets are split into training and testing subset with 1:2 ratio using hold-out random sampling method. This results in 10 instances of training data and 20 instances of testing data. 5-fold cross validation was employed for model analysis in both data subsets in all experiment cases.

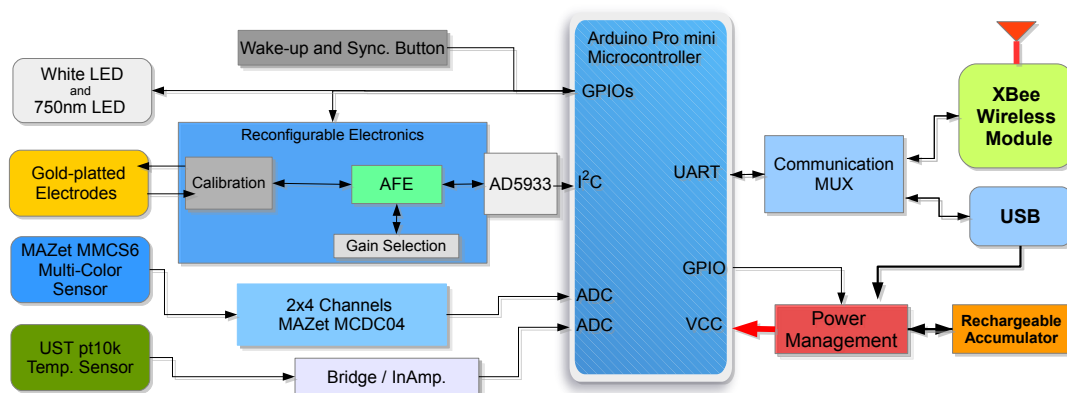


FIGURE 87: Hardware block diagram of LoX device

TABLE 23: Parameters of the LoX device configuration

IS frequency range	10kHz to 100kHz		
IS impedance range	1k Ω -10M Ω		
IS frequency resolution	0.1Hz		
IS gain	1 or 2		
IS excitation voltage	0.2, 0.4, 1.0, or 2.0 Volt		
IS R_{FB}	2.2k Ω		
ADC configurations	Arduino	AD5933	MCDC04
ADC input	10-bit	12-bit	16-bit
ADC bit-resolution	3.22 mV/LSB	4.93 μ V/LSB	20 fA/LSB
ADC span	3.3V	2.0 V	3.3V
ADC sampling rate	15ks/Sec	1 Ms/Sec.	400ks/Sec.
Transimpedance amp.	$R_{ref} = 500$ k Ω .		
Sensor information	Temp., 16 wave lengths multi-color (MS-LoX) or RGB-color sensor, 1024 points of Magnitude and Phase		

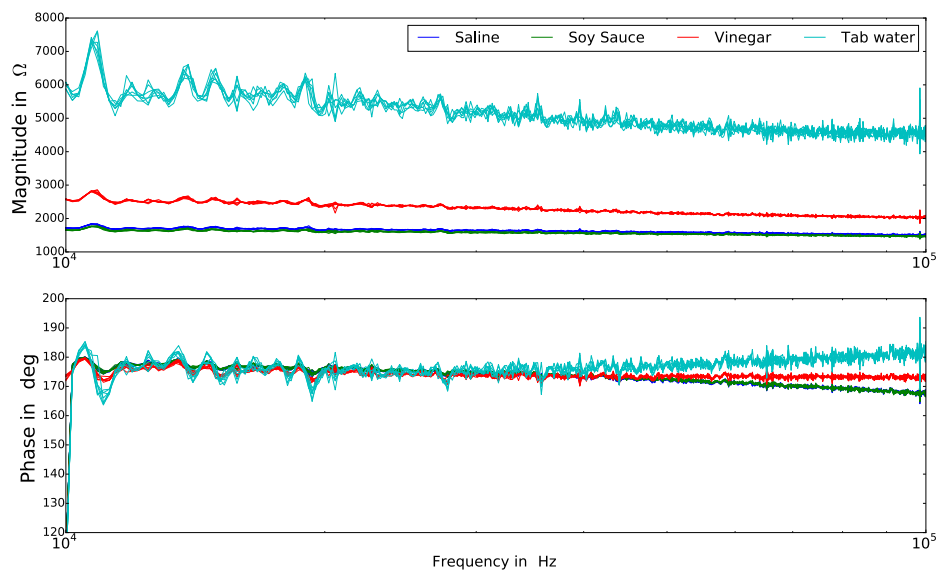


FIGURE 88: Impedance magnitude and phase spectrum plots of the SoyVine data set

TABLE 24: LoX data set descriptions

Data set Name	No. of (class; sample)	DAQ Device	Description
UsedOil	2; 60	LoS	Fresh and used cooking oil
Oil	3; 90	LoS	3 types of cooking oil
Beer	4; 120	LoS	4 brands of beer
7Wine	7; 210	LoS	7 different kinds of wine
SoyVine	3; 90	MS-LoS	Soy, Vinegar, and Tap water
WineGly	11; 330	MS-LoS	Contamination analysis of wine
Milk	4; 120	MS-LoS	Degradation analysis of milk
Salt	3; 90	MS-LoF	Salt and salt adulterated with chalk powder
Powder	9; 270	MS-LoF	9 powder and granular ingredients

6.1.1.1 Preliminary Investigation of Self-X Properties Integration

The improvement of the hardware instance architecture by dynamic reconfiguration and self-x features based on switching devices and reconfigurable electronic front-ends as illustrated in Fig. 89 is currently pursued. Self-calibration of the IS can be performed by connecting external reference resistors to the inputs, where analog CMOS switches or DC-MEMS switches [119] can be the switching devices. In addition to using dedicated reference resistors, a standard liquid, oils of different color, with reference characteristics obtained from laboratory instruments can be used for the calibration process, which can serve a broader range of sensors type, e.g., color, pH, and viscosity just from single reference source. The calibration coefficients of the hardware instance will be updated from the acquired data to compensate measurement errors using reference information, i.e., deterministic models of the calibration resistors. Two-Point, multi-point, or piecewise [82] calibration approach can be employed here. The procedures follow the diagrams given in Fig. 90. The processing of the procedure can be conducted at a design platform using bi-directional communication, i.e., intrinsic (static), or at the run-time platform, i.e., intrinsic (dynamic), depending on computational resource requirement.

However, in an extreme situation, the calibration approach alone may not be able to cope with large deviations. Thus, trimmings of the hardware instance configurations and parameters are required, which are conventionally done by hand. These tasks can be performed automatically, i.e., self-trimming, by incorporating additional reconfigurable electronics to adjust circuit characterizations, e.g., gain and offset. Advanced analog front-end ICs, e.g., AD5933 and MCDC04, provide reconfiguration capability to adjust their internal parameters as given in Table 25. They can be the trimming parameters, thus, the scope of adjustments is broaden and more levels of deviation can be tackled. Further, these ICs are usually integrated with operational condition related sensors, e.g., temperature sensor, which can be used as a source of self-compensation process to eliminate influences from environment. As a result, a potentially robust hardware instance will be obtained. The combination of self-calibration with self-trimming will offer a solution for longterm dependable measurement systems.

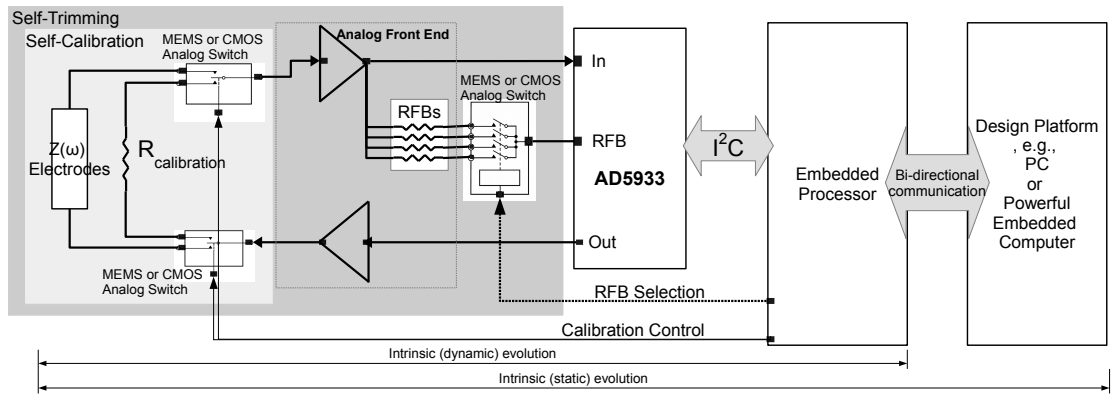


FIGURE 89: Proposed hardware structure for self-calibration/trimming

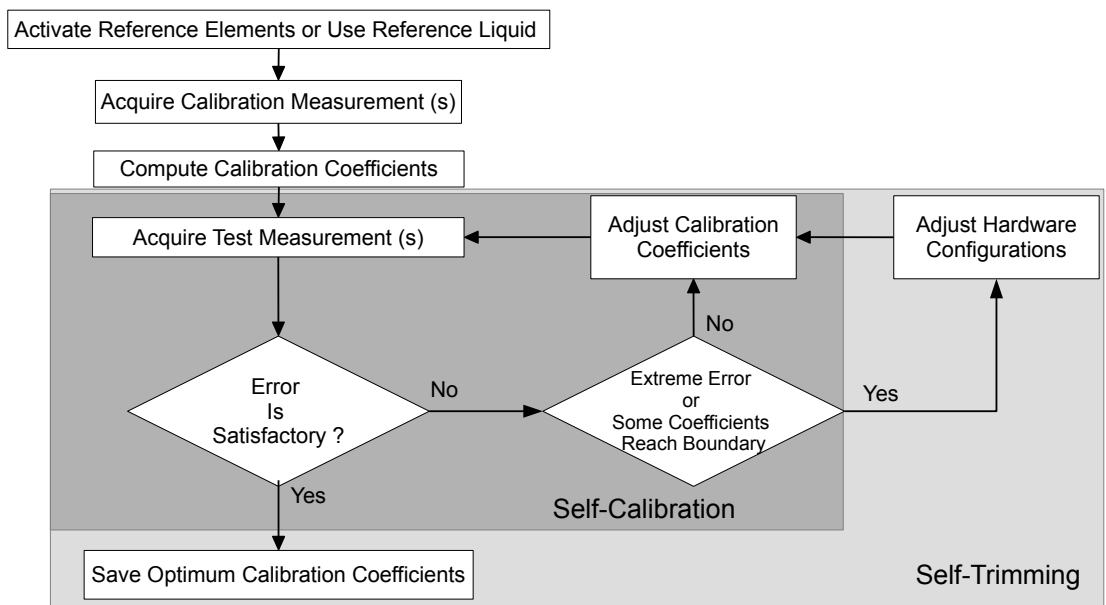


FIGURE 90: Flowchart of proposed self-calibration/trimming

TABLE 25: Possible integrated reconfiguration capabilities for self-trimming process

Device	Parameters
AD5933	Gain, Excitation voltage, ADC settling time, Start and stop sweeping frequency Number of sweeping points
MCDC04	Gain, Reference voltage

6.1.2 Conventional Application Design Approach

The conventional and manual designs of LoX technical cognition applications have been conducted as first cut designs, where automation was less involved in, thus, the design effort was at the ceiling amount compared to the proposed automated design approach. This creates criteria of assessment purpose in terms of design work load as well as solution quality when using DAICOX in the same applications. The conventional design approach uses standard existing toolboxes and few extra self-developed modules, which has been used as standard framework for early experiments in [26, 183, 184]. Table 52 shows the results obtained by using the conventional framework from the data sets discussed in Table 24 with the same training and testing data size as used throughout the entire experiment using DAICOX. The design procedures follow the flowchart given in Fig. 91, where partially automated process can be conducted in some parts, e.g., DR design or classifier parametrization. These, however, are based on trial-and-error together with the rule of thumb approach, which is time consuming and requires a lot of effort.

The most of the design time is at the repetitions of reconfigurations of both DR and classification design processes, which depends upon experience of the designer and also the complexity and difficulty of the design problems. Therefore, DAICOX takes over the complete design process of the flowchart by requiring almost no task specific configurations or settings, which will be explained in detail with experimental results in the next following sections.

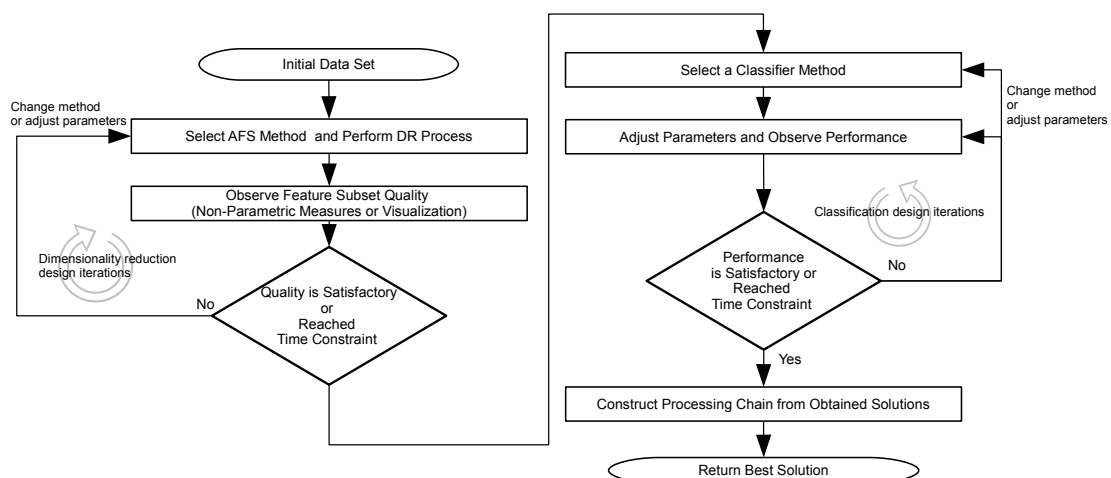


FIGURE 91: A conventional design workspace for LoX application

6.1.3 Automated Design Using DAICOX

The currently implemented DAICOX architecture provides design automation of the complete processing chain of an LoX application. At the beginning of a design task, the dimensionality reduction process is performed with choices of method as well as result analysis and visualization. The best quality feature subset will be selected automatically based on performance and will be forwarded to the classification design tasks. Automatic parameter optimization of the classification task is performed using the DAICOX classifier design toolbox. The best performance classifier will be selected automatically and embedded into the final processing chain solution. The explanation and experiments of automated LoX application design using DAICOX at each processing block are given in the following sub sections.

6.1.3.1 Automated Dimensionality Reduction Design

Dimensionality Reduction (DR) is an essential design task for LoX applications due to the high dimensional data provided from the LoX device. As pointed out in the previous chapter in Section 5.2.2 that the meta-heuristic approaches can find considerably good quality solutions within reasonable number of iterations. Thus, the dimensionality reduction design tasks for LoX application employ the implemented meta-heuristic feature selection and feature weighting approaches, e.g., GA-FS, PSO-FS, GA-FW and PSO-FW. The visual programming workspace illustrated in Fig. 92 was used to perform the experiments and to obtain the results in both table and graphical representations. The parameter settings of all methods are given in Table 26 for GA-FS and PSO-FS and Table 16 for GA-FW and PSO-FW in the previous chapter. The search procedures of all methods are performed with a multi-objective assessment function, which is composed of several feature space quality assessments, e.g., q_{OV} , $q'_{C-intra}$, $q_{C-inter}$, and q_S . In this work, all criteria are combined by using the weighted agglomeration approach with weight-averaged scheme as given in Eq. 6.1.

$$q_{overall} = \frac{q_{OV} + q'_{C-intra} + q_{C-inter} + q_S}{4} \quad (6.1)$$

The results of DR tasks for LoX application are given in Table 28 as well as bar plots in Fig. 94. The dimensional reduction capability d is provided in the result table, which indicates the size of the best feature subset dimensionality in percent compared to full feature. t_{search} is the time used in searching for one given data fold. All given values are averaged values from 5-fold cross validation applied to the testing data subset and the smaller numbers in parentheses are the corresponding standard deviations. The initial value of $q_{overall}$ assessments, which are measured from full features of a particular

TABLE 26: Parameter settings for AFS methods used in the LoX case study

Method	Parameters and Settings
GA-FS	Population Size = $10 + 2\sqrt{d}$, Max. Iteration = 100, Selection Rate = 0.5, Mutation Rate = 0.2
PSO-FS	Population Size = $10 + 2\sqrt{d}$, Max. Iteration = 100, $C_1=2.0$, $C_2=2.0$, $w=1.2$

d is the number of feature of initial data.

data set, are given in Table 27 for comparison with the results of the DR methods. An intuitive methods' comparison is visualized in the radar plot given in Fig. 93 representing the methods' ranking of a particular characteristics, where the highest rank is in the outermost. For the classification time, the first rank is the fastest searching method. This representation helps to analyze the methods and provides the overview perspective for method selection. The best feature subset quality of each reduction scheme was selected for classification design task explained in the following section.

All conducted DR methods discovered significantly better feature assessments in all data sets compared to the conventional ones. Genetic algorithms have shown their capability in searching for better solutions, than those, which are discovered by PSO in both feature selection and feature weighting schemes. Comparing between the two schemes, feature weighting outperformed feature selection in all cases, of course, with the price tag of a higher computational cost. GA also performed the fastest in all feature selection cases, which is due to their searching mechanism, that is effective for binary selection problems. The classification results of classification design experiments can be used to confirm the quality of the selected feature subset. It can be noted that GA based DR methods are good choices concluded by this application instance. However, in general, method comparison and analysis are still necessary, which is already facilitated by DAICOX.

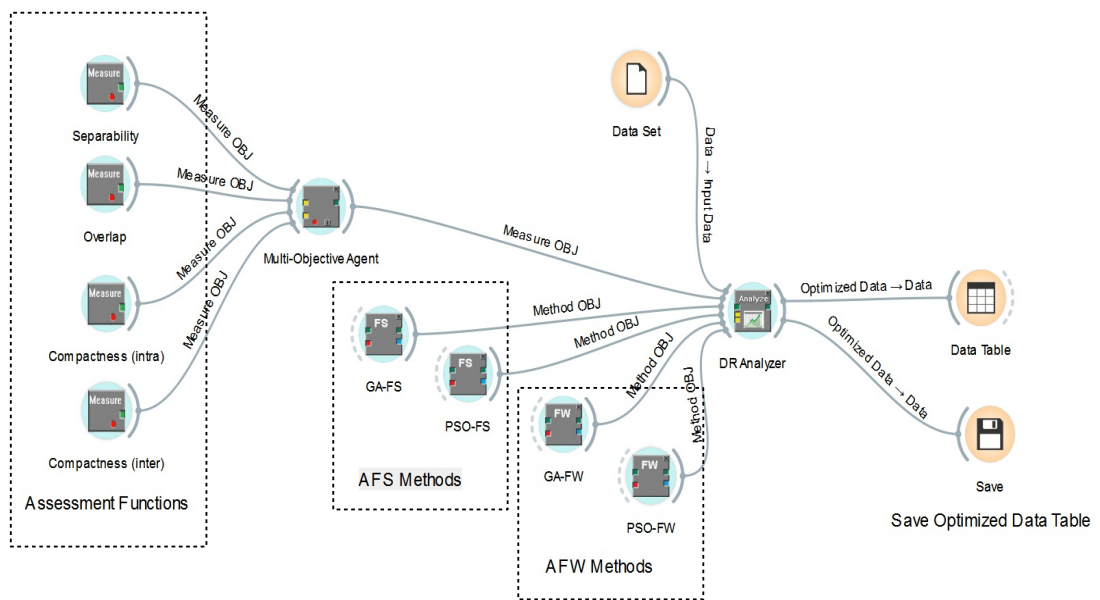


FIGURE 92: Visual programming workspace of dimensionality reduction experiments

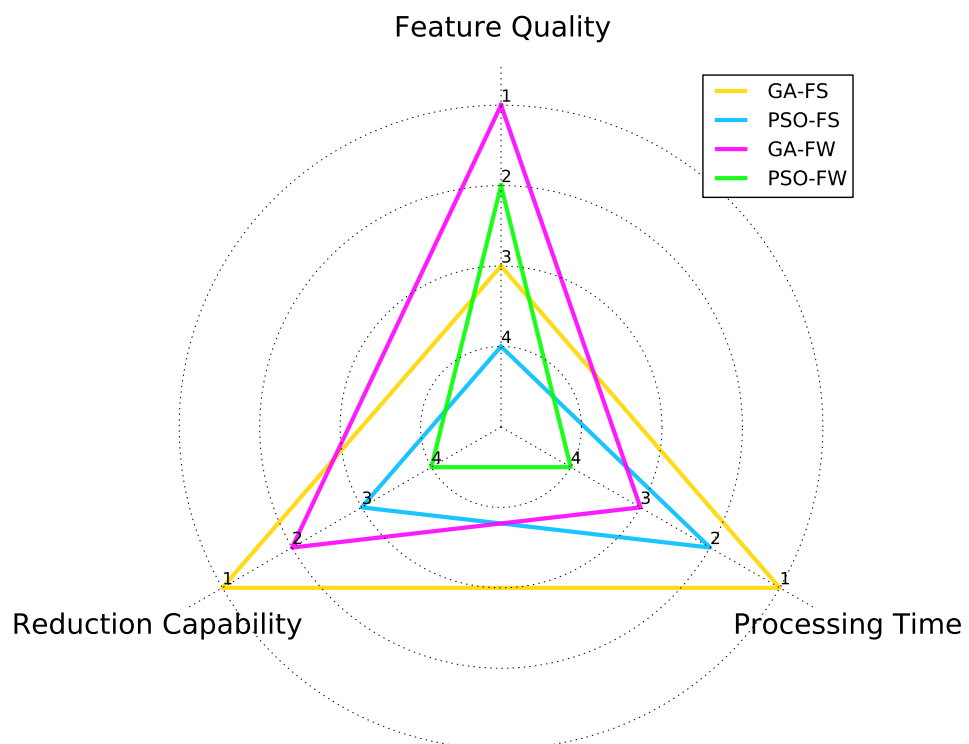


FIGURE 93: Radar plot of DR method rankings in different performance characteristics

TABLE 27: Multi-objective quality measure of full-feature

UsedOil	Oil	Beer	7Wine	SoyVine	Milk	WineGly	Powder	Salt
0.667	0.588	0.528	0.505	0.636	0.540	0.477	0.580	0.625

TABLE 28: Overall result of dimensionality reduction tasks

Data Set	GA-FS			PSO-FS		
	$q_{overall}$	d	$t_{search}[s]$	$q_{overall}$	d	$t_{search}[s]$
UsedOil	0.660(0.065)	56.6%(23.3%)	15.63(5.06)	0.586(0.039)	50.0%(2.5%)	24.35(0.25)
Oil	0.714(0.055)	53.8%(8.0%)	18.65(2.06)	0.662(0.044)	39.2%(16.5%)	28.06(2.00)
Beer	0.751(0.014)	40.8%(2.7%)	18.10(1.03)	0.732(0.019)	40.3%(5.8%)	33.28(1.24)
7Wine	0.654(0.020)	30.5%(6.7%)	26.50(3.74)	0.669(0.016)	45.8%(2.7%)	55.66(0.67)
SoyVine	0.817(0.014)	45.3%(20.4%)	17.03(5.35)	0.802(0.011)	44.1%(10.7%)	28.65(1.31)
Milk	0.652(0.015)	52.1%(15.5%)	27.03(6.00)	0.672(0.021)	47.2%(4.4%)	40.54(1.17)
WineGly	0.536(0.018)	58.5%(17.0%)	76.64(17.07)	0.556(0.020)	48.7%(1.4%)	98.99(1.22)
Powder	0.727(0.007)	55.2%(19.3%)	34.21(9.12)	0.718(0.019)	48.6%(2.3%)	48.35(0.54)
Salt	0.815(0.002)	48.1%(16.8%)	16.86(4.43)	0.817(0.004)	46.6%(4.5%)	29.12(0.61)

Data Set	GA-FW			PSO-FW		
	$q_{overall}$	d	$t_{search}[s]$	$q_{overall}$	d	$t_{search}[s]$
UsedOil	0.761(0.039)	78.2%(6.9%)	35.94(0.74)	0.700(0.041)	59.2%(22.1%)	63.24(1.98)
Oil	0.787(0.041)	62.9%(23.9%)	30.24(2.17)	0.730(0.020)	26.0%(14.7%)	65.91(0.22)
Beer	0.756(0.008)	17.4%(7.4%)	24.69(1.91)	0.742(0.007)	11.7%(7.4%)	114.0(0.98)
7Wine	0.684(0.008)	73.7%(18.3%)	93.21(3.88)	0.657(0.015)	34.2%(7.5%)	155.1(4.31)
SoyVine	0.796(0.011)	37.3%(25.9%)	22.68(2.14)	0.798(0.011)	21.6%(10.7%)	66.75(0.33)
Milk	0.734(0.008)	79.7%(12.2%)	49.77(2.09)	0.709(0.031)	41.1%(25.3%)	91.65(0.43)
WineGly	0.616(0.024)	74.9%(17.9%)	126.69(3.06)	0.603(0.025)	55.0%(11.9%)	205.65(1.03)
Powder	0.683(0.010)	76.3%(18.1%)	60.99(1.48)	0.705(0.031)	49.3%(17.9%)	107.1(0.50)
Salt	0.840(0.002)	59.3%(5.4%)	30.15(1.07)	0.839(0.007)	25.2%(9.4%)	66.03(0.11)

Results obtained with fixed random seed number = 12081983

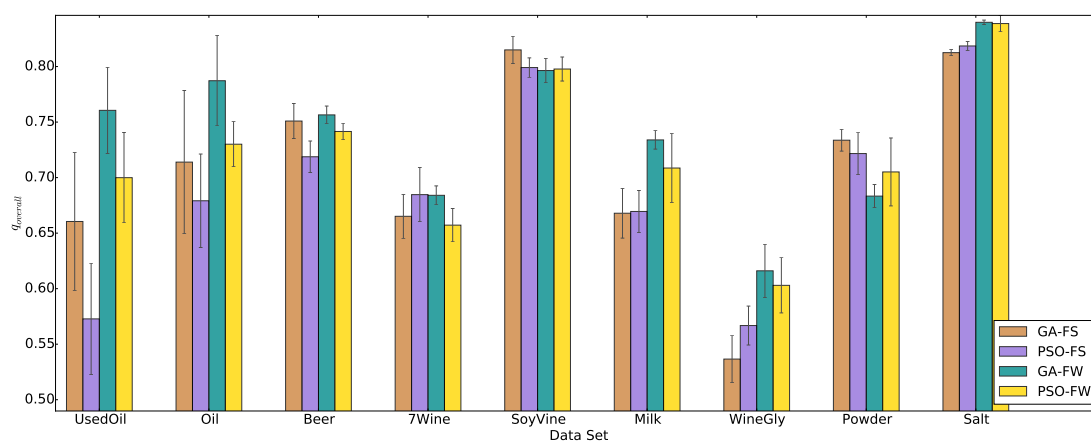


FIGURE 94: Bar plots comparison of DR methods for LoX data sets

6.1.3.2 Classification

Classification is a key processing component of the E-Taster system in terms of providing a final decision output for food ingredients recognition. In practical uses, both accuracy and robustness of a classifier are the primary consideration aspects in order to ensure the correctness of prediction results in operations with fresh measured data, i.e., live classification. Thus, the classification design experiments were conducted and demonstrated using toolboxes and libraries of the currently implemented DAICOX architecture for automating parameterization and model analysis procedures. Three classification approaches have been conducted in the experiments, e.g., multi-class classification, hierarchical-classification, and one-class classification. SVM-C classification was used in the multi-class classification case to serve as a conventional classification method design with an extension of parameter optimization using GA and PSO.

The more complex classification approach was performed in multi-level scheme using the proposed hierarchical-classification approach. For LoX data, three classifiers have been established in the first classification level, each for an individual sensory channel, corresponding to physical data sources, which are color, magnitude of impedance, and phase of impedance spectrum. The processing flow of the hierarchical classification for LoX data is depicted in Fig. 95. The dimensionality reduction was performed locally at each sensory channel by employing the methods discussed in the previous section. The first level classifiers create class probability vectors, i.e., soft-output, from a given input instance. These vectors will be concatenated and given to the final classifier in the decision making fusion level. The visualization of the concatenated class probability vectors obtain from multiple input instances can be used to evaluate and analyze the performance of the first level classifiers. The scatter plots of the generated class probability vectors compare to original data for each data set are given in Fig. 99 to Fig. 107. The plots are generated by using the Multi-Dimensional Scaling (MDS) technique with Sammon's stress mapping approach.

The latter case of classification design is based on One-Class Classification (OCC) approach using the implemented methods addressed in the previous chapter. The majority of classification problems of the E-Taster system are given with a specific target class, i.e., authenticity checking and anomaly detection. In the case of encountering with data, that do not belong to the sensory database, i.e., novel or abnormal data, OCC approach is capable to reject the data or report novelty of the data for further actions. In such case, using multi-class classification approach may return surprising and unsuitable generalization due to the lack of sufficient training data for negative cases. The design process for an OCC classifier requires an additional step of single class data separation for the creation of the classifier model while the evaluation process uses multi-class data

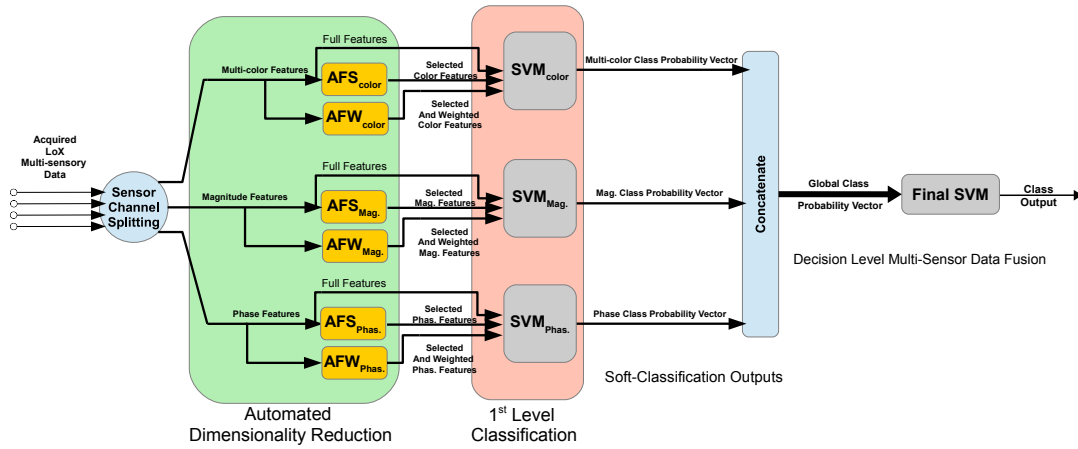


FIGURE 95: Hierarchical classification processing structure of LoX data set

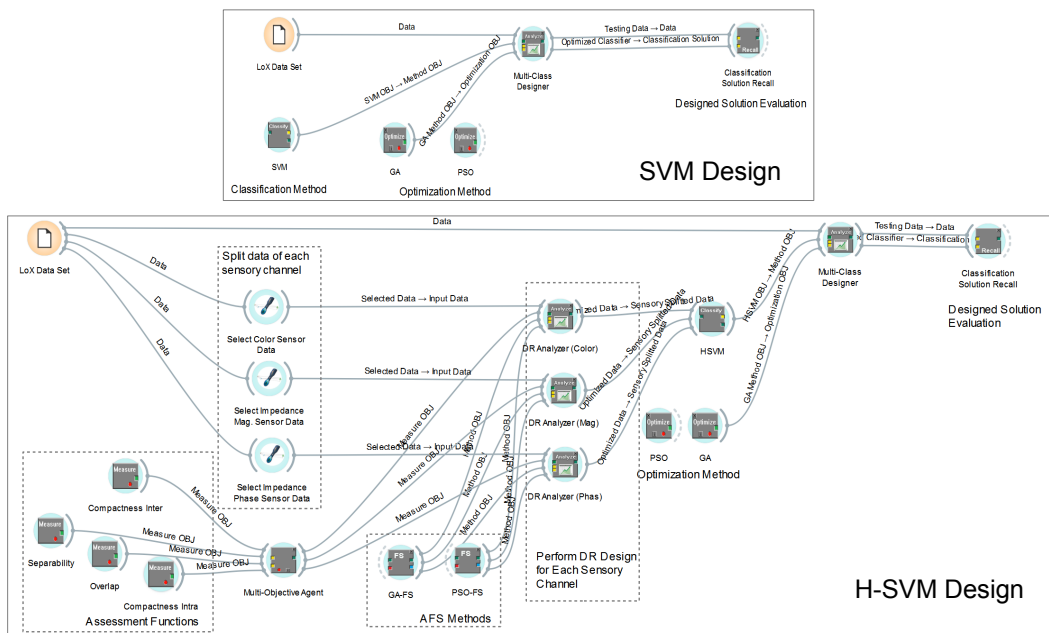


FIGURE 96: Visual programming workspace of multi-class classification design experiments

to observe both correct acceptance and correct rejection performance. Figure 97 illustrates the visual programming workspace of the OCC design used in the experiments.

In all classification design experiments, the data sets are applied with three variations of feature subsets obtained from the previous step including full-feature, AFS and AFW. In multi-class classification scenarios, the experiments were performed by using visual programming workspace as shown in Fig. 96. For SVM based classification methods, the parameter C and γ are obtained by an optimization procedure employing GA and PSO

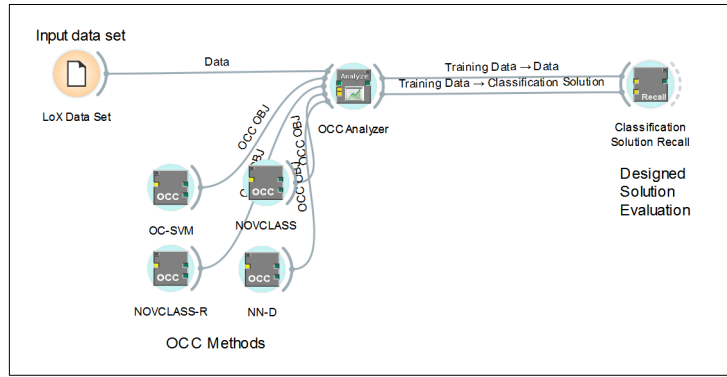


FIGURE 97: Visual programming workspace of one-class classification design experiments

to tune and search for the best classifier. In every design case, the best performance classifier is selected based on the model analysis results using 5-fold cross validation in order to ensure the generalization and robustness of the selected solution. The parameters of the employed optimization methods are given in Table 29.

TABLE 29: Parameter settings for classifier optimization methods

Method	Parameters and Settings
GA	Population Size = 20 , Max. Iteration = 50, Selection Rate = 0.5, Mutation Rate = 0.2
PSO	Population Size = 20, Max. Iteration = 50, $C_1=1.2$, $C_2=1.2$, $w=1.0$,

The experimental results for multi-class and hierarchical classification cases are given in Table 32. All given numbers are averaged values from 5-fold cross validation applied to the testing data subset and the smaller numbers in parentheses are standard deviation. Figure 108 to 110 show classification performance results of OCC based classification tasks. The classification accuracy for the OCC based classification is the average of true positive and true negative rates of a particular method. The number of stored reference vectors of all experiments are given in Figures 111 to 113, which indicate computational cost of a particular case.

Classifications results from all experiments cases show the considerably good performance of the implemented classification methods in dealing with real world scenarios. 100 percent classification rate has been achieved in several cases. Proposed hierarchical classification approach provided better classification performance compared to standard SVM-C. The feature space plots of class probability matrices generated from the first level classification for each data set indicate significant improvement in feature space quality. In some cases, the scatter plots show linearly separable data meaning that

the final classifiers may not need to be complex and, thus, better generalization performance can be achieved. The trade-off is the high computational cost, which reflects long classification and training time.

In OCC cases, the proposed NOVCLASS classification outperformed both NN-D and OC-SVM in several cases. The classification results of OCC methods are comparable to the conventional multi-class cases except in the case of the UsedOil data set. Based on the results obtained in all experiments, the rankings of the OCC methods are given in Fig. 98 where "Best TP" and "Best TN" is the number of highest true positive and true negative classification respectively, "NetSize" is the number of stored reference vector, t_{train} is the training duration, and t_{class} is the classification duration. As already investigated in the previous chapter, NOVCLASS-R stored the least number of reference vectors but still returned high classification rate and even outperformed the standard NOVCLASS algorithm in some cases. This indicates the effectiveness of the proposed NOVCLASS method in terms of classification performance and computational cost. The OCC results have also confirmed the potential usage of one-class classification approaches in LoX application for novelty/anomaly detection with high classification rate.

Table 30 gives the design complexities of the conducted multi-class classification design experiments. The design complexities in this case study are expressed in terms of design effort and design duration. The design effort is evaluated by the number of design steps, that require designer interactions, for instance, sensory data base access and assignment, data set separation, parameter setting, classification model generation, validation, and model selection, where all the tasks have to be manipulated manually by the designer. In contrast, DAICOX requires only two user activities of assigning data set to perform and define optimization parameter settings only at one time at the beginning of a design process. More important, the design effort increases by the number of design problems, in this case by the number of data sets, for the conventional design approach, whereas DAICOX requires no additional effort from the increased number of problems. The tedious and knowledge requiring tasks of decision making can be automatically executed by DAICOX, while the better results were also achieved. The shorter design time is reflected by the reduced design effort and it should be noted that manual decision makings will aggravate further the already much slower design process compared to using DAICOX. This indicates 81.87% reduction in terms of human(expert) resource consumption. The computation time is the duration, where automated processes take part, that depends on the processing speed of the design platform, e.g., PC. The classification accuracies have been improved in all cases by using DAICOX, in particular, by using H-SVM approach, whereas computational cost is the price tag of the enhanced solutions, which can be practically tackled by using powerful distributed computing units.

TABLE 30: Tentative design complexity comparison

	Conventional/Manual	DAICOX	
No. of design steps (per data set)	19	2	
No. of design steps (9 data sets)	171	2	
No. of decisions (by designer) (per data set)	6	0	
No. of decisions (by designer) (9 data sets)	54	0	
Design Time [s] (per data set)	120 (<i>approx.</i>)	15 (<i>approx.</i>)	
Design Time [s] (9 data sets)	1,080 (<i>approx.</i>)	15 (<i>approx.</i>)	
Average Computation Time [s] (9 data sets)	248.5	225.8 (SVM-C)	3,178.8 (H-SVM)
Total Design Time [s] (9 data sets)	1328.5	240.8 (SVM-C)	3,193.8 (H-SVM)
Average Classification Accuracy (9 data sets)	93.41%	97.74% (SVM-C)	98.81% (H-SVM)

*The experiments were performed on a PC with Intel Core i7 CPU at 2.8 GHz and 6 GB RAM

TABLE 31: Classification performance obtained from flat SVM classifiers designed by conventional approach

Data Set								
UsedOil	Oil	Beer	7Wine	SoyVine	WineGly	Milk	Salt	Powder
90.00%	96.67%	92.50%	92.99%	93.33%	78.18%	97.00%	100.00%	100.00%

TABLE 32: Multi-Class classification results

Data Set	SVM-C		
	Full	AFS	AFW
UsedOil	92.50% _(1.10%)	75.00% _(2.96%)	92.50% _(1.10%)
Oil	100.00% _(0.00%)	95.00% _(0.61%)	100.00% _(0.00%)
Beer	98.75% _(0.14%)	98.75% _(0.14%)	100.00% _(0.00%)
7Wine	95.71% _(0.35%)	95.00% _(0.40%)	97.14% _(0.23%)
SoyVine	100.00% _(0.00%)	100.00% _(0.00%)	100.00% _(0.00%)
Milk	100.00% _(0.00%)	100.00% _(0.00%)	100.00% _(0.00%)
WineGly	90.00% _(0.61%)	88.18% _(0.70%)	87.73% _(0.73%)
Powder	100.00% _(0.00%)	98.33% _(0.15%)	99.17% _(0.08%)
Salt	100.00% _(0.00%)	100.00% _(0.00%)	100.00% _(0.00%)
Data Set	H-SVM		
	Full	AFS	AFW
UsedOil	100.00% _(0.00%)	100.00% _(0.00%)	95.00% _(0.75%)
Oil	100.00% _(0.00%)	100.00% _(0.00%)	100.00% _(0.00%)
Beer	98.75% _(0.14%)	100.00% _(0.00%)	100.00% _(0.00%)
7Wine	99.29% _(0.06%)	97.86% _(0.18%)	99.29% _(0.06%)
SoyVine	100.00% _(0.00%)	100.00% _(0.00%)	100.00% _(0.00%)
WineGly	90.00% _(0.61%)	87.27% _(0.75%)	90.00% _(0.61%)
Milk	100.00% _(0.00%)	100.00% _(0.00%)	100.00% _(0.00%)
Powder	100.00% _(0.00%)	100.00% _(0.00%)	100.00% _(0.00%)
Salt	98.33% _(0.21%)	100.00% _(0.00%)	100.00% _(0.00%)

Results obtained with fixed random seed number = 12081983

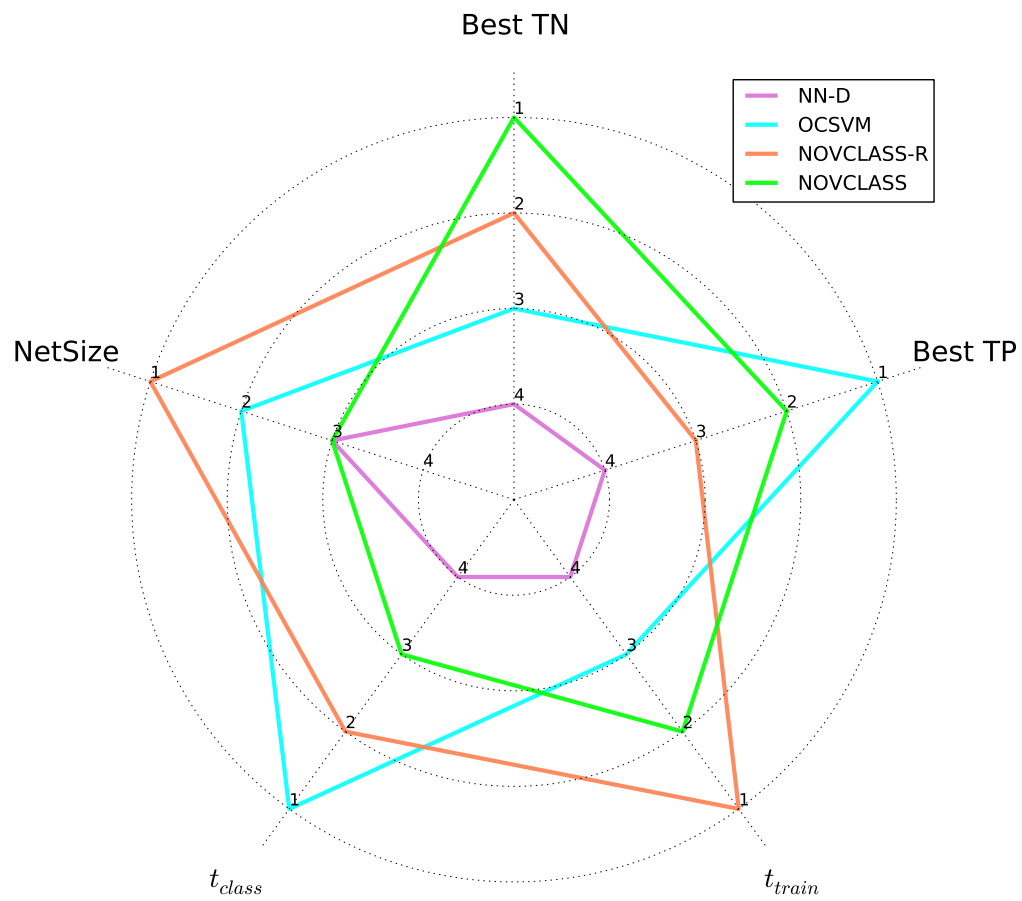


FIGURE 98: Radar plot of OCC method ranking of a particular performance characteristics

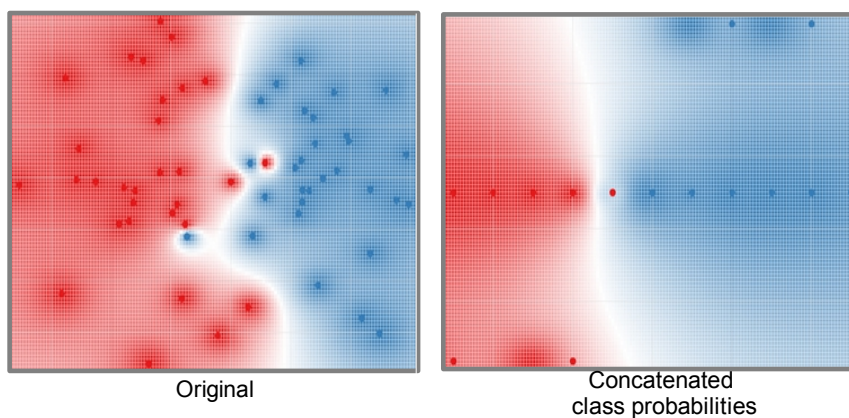


FIGURE 99: Feature space projection of the "UsedOil" data set and its class probability vectors for hierarchical classification

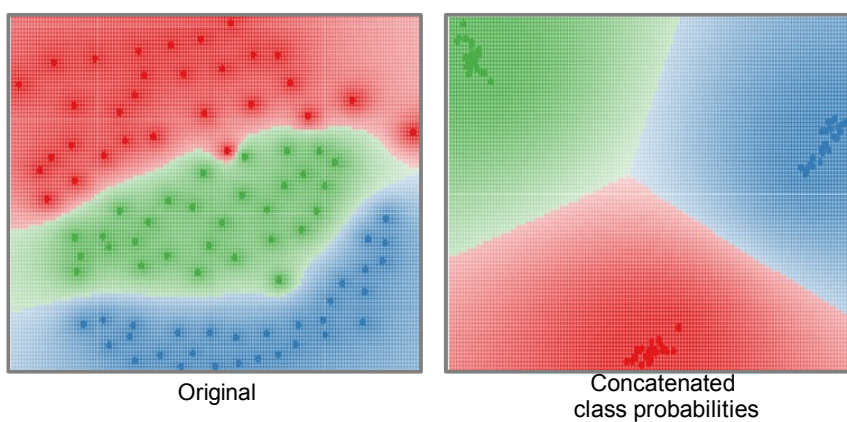


FIGURE 100: Feature space projection of the "Oil" data set and its class probability vectors for hierarchical classification

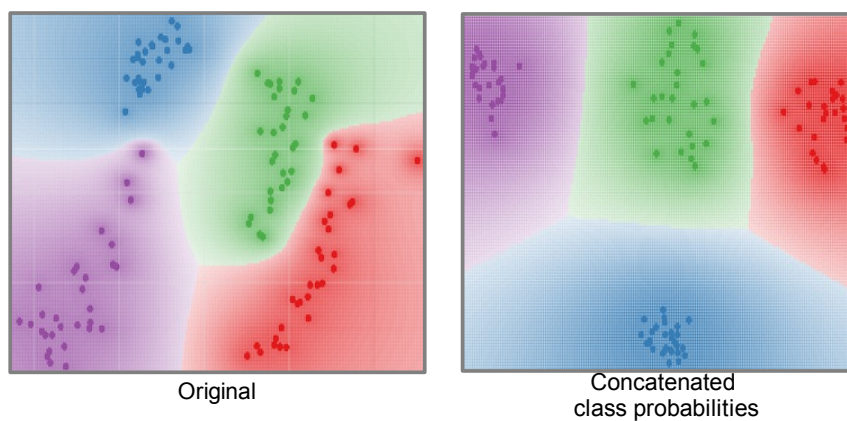


FIGURE 101: Feature space projection of the "Beer" data set and its class probability vectors for hierarchical classification

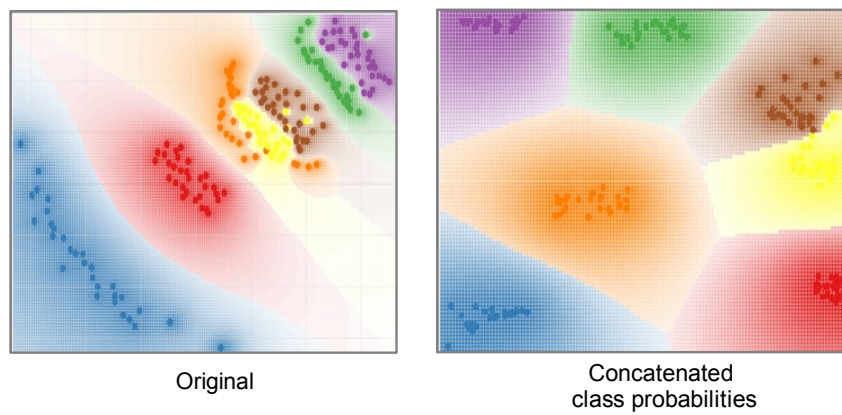


FIGURE 102: Feature space projection of the "7Wine" data set and its class probability vectors for hierarchical classification

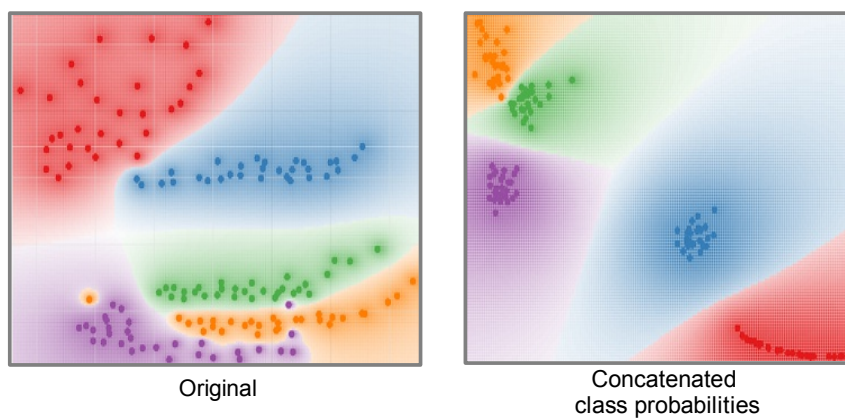


FIGURE 103: Feature space projection of the "Milk" data set and its class probability vectors for hierarchical classification

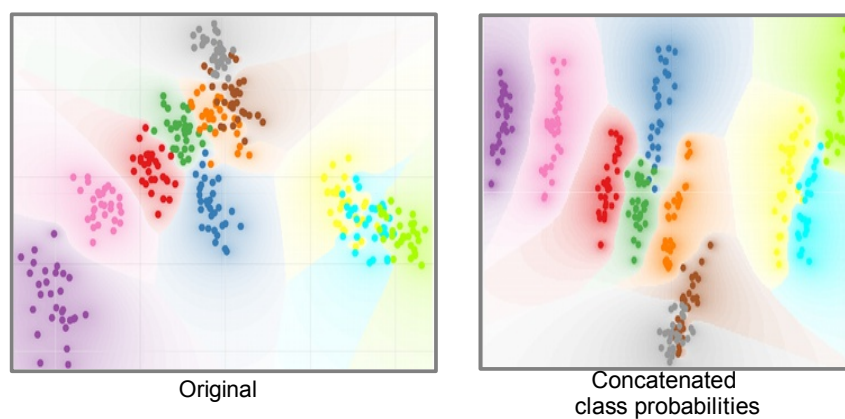


FIGURE 104: Feature space projection of the "WineGly" data set and its class probability vectors for hierarchical classification

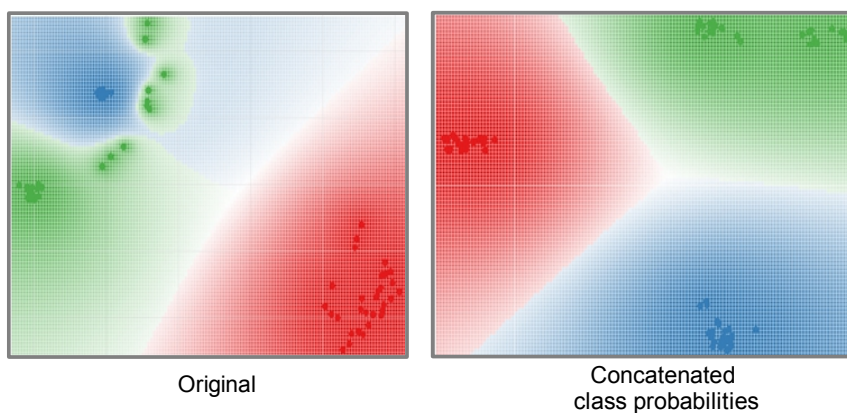


FIGURE 105: Feature space projection of the "SoyVine" data set and its class probability vectors for hierarchical classification

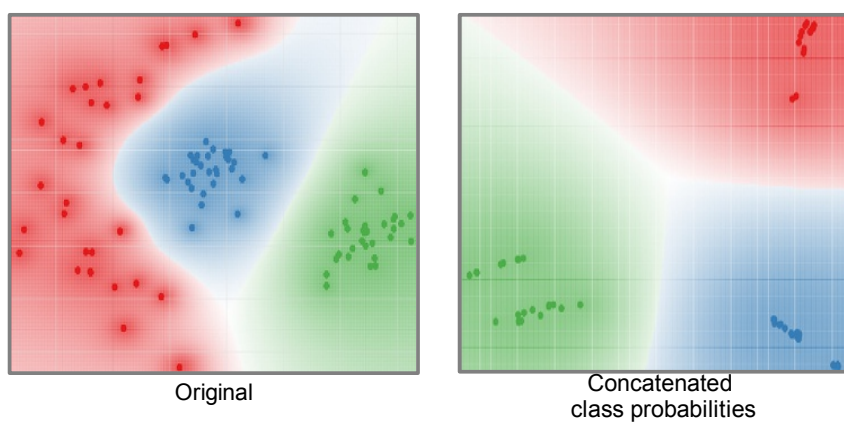


FIGURE 106: Feature space projection of the "Salt" data set and its class probability vectors for hierarchical classification

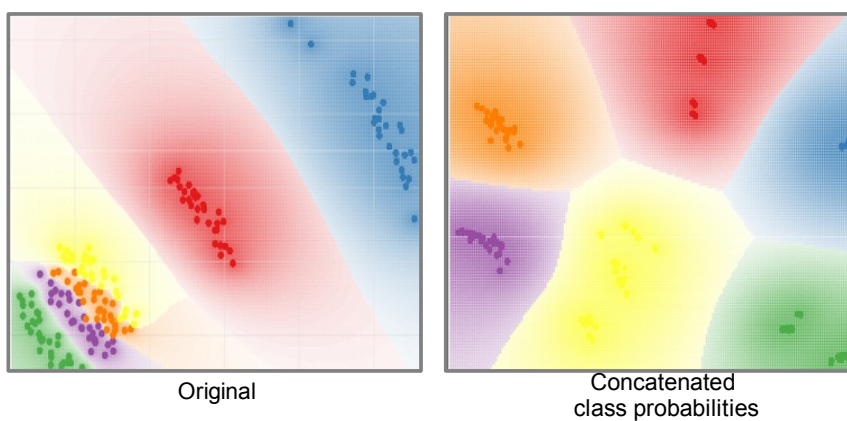


FIGURE 107: Feature space projection of the "Powder" data set and its class probability vectors for hierarchical classification

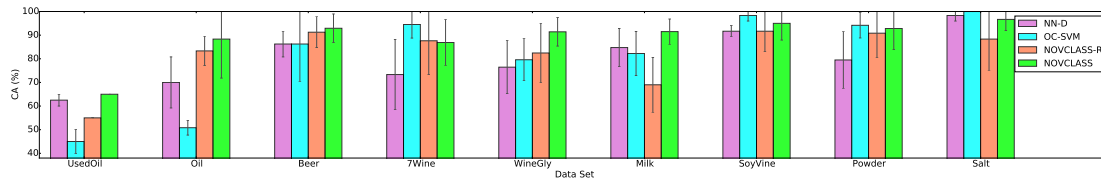


FIGURE 108: Average classification accuracy of OCC methods using LoX data sets with full features

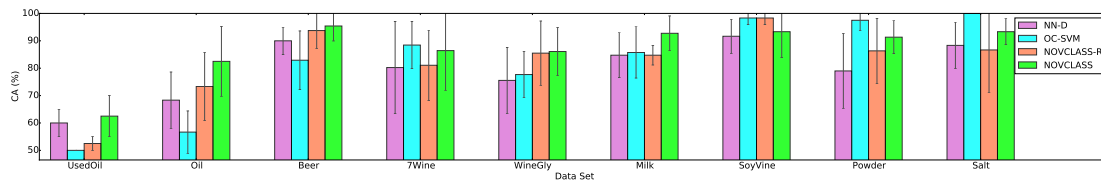


FIGURE 109: Average classification accuracy of OCC methods using LoX data sets with AFS

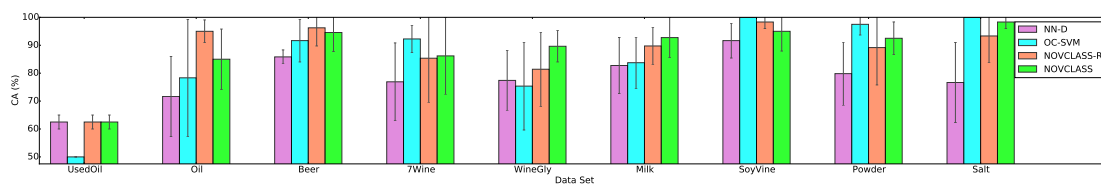


FIGURE 110: Average classification accuracy of OCC methods using LoX data sets with AFW

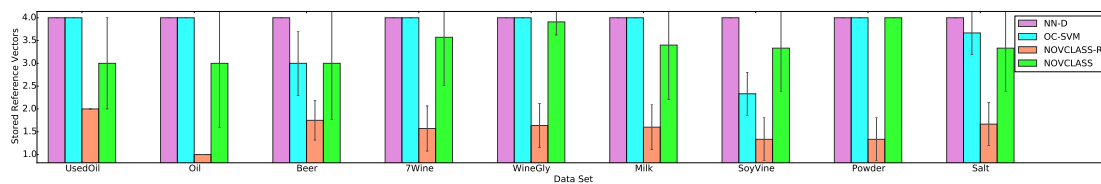


FIGURE 111: Average number of stored reference vectors of OCC methods using LoX data sets with full features

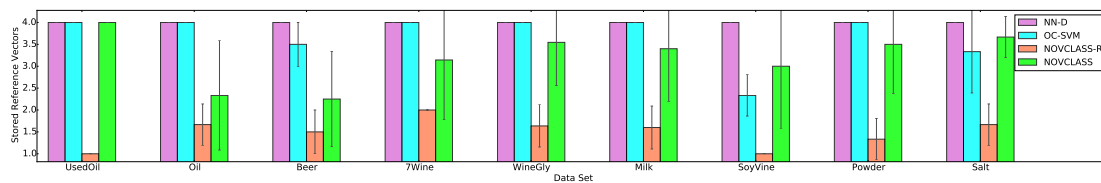


FIGURE 112: Average number of stored reference vectors of OCC methods using LoX data sets with AFS

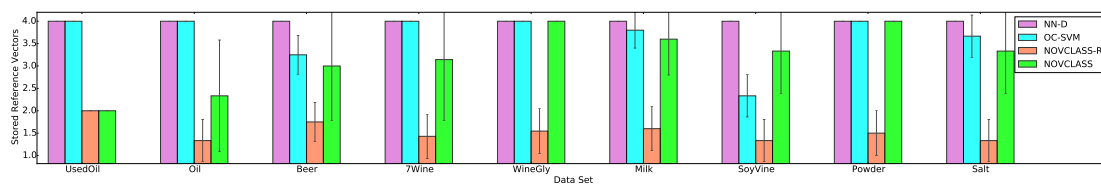


FIGURE 113: Average number of stored reference vectors of OCC methods using LoX data sets with AWS

A complete processing chain, comprising of data acquisition, dimensionality reduction, and classification, design can be performed in a single design workspace as illustrated in Fig. 114. This design scheme is useful especially at the beginning of an application, i.e., starting from scratch, in particular, with a non expert designer. The entire process evolves sequentially from the beginning building block, e.g., data acquisition, to a consecutive block until the final processing task, e.g., classification. Once an optimized instance of a complete processing chain is selected (automatically or manually by the designer), the refined workspace of the chain can be stored in the design database as a template for new application design (with constrained optimization). Additionally, the processing chain itself can be bred or fine tuned within constrained search space and additional data and information. Breeding techniques, e.g., genetic algorithms or evolutionary strategies are potential candidates for this approach. By this concept, the complete intrinsic optimization can be realized by embedding the processing chain workspace and DAICOX library to a run-time platform. The selected final complete solution, i.e., a designed processing chain comprised of data acquisition and sensor configurations, dimensionality reduction, and classification, can be exported into a standard DAICOX executable object for the purpose of further system integration, e.g., live-classification application with GUI, as has been integrated to the current E-Taster software environment.

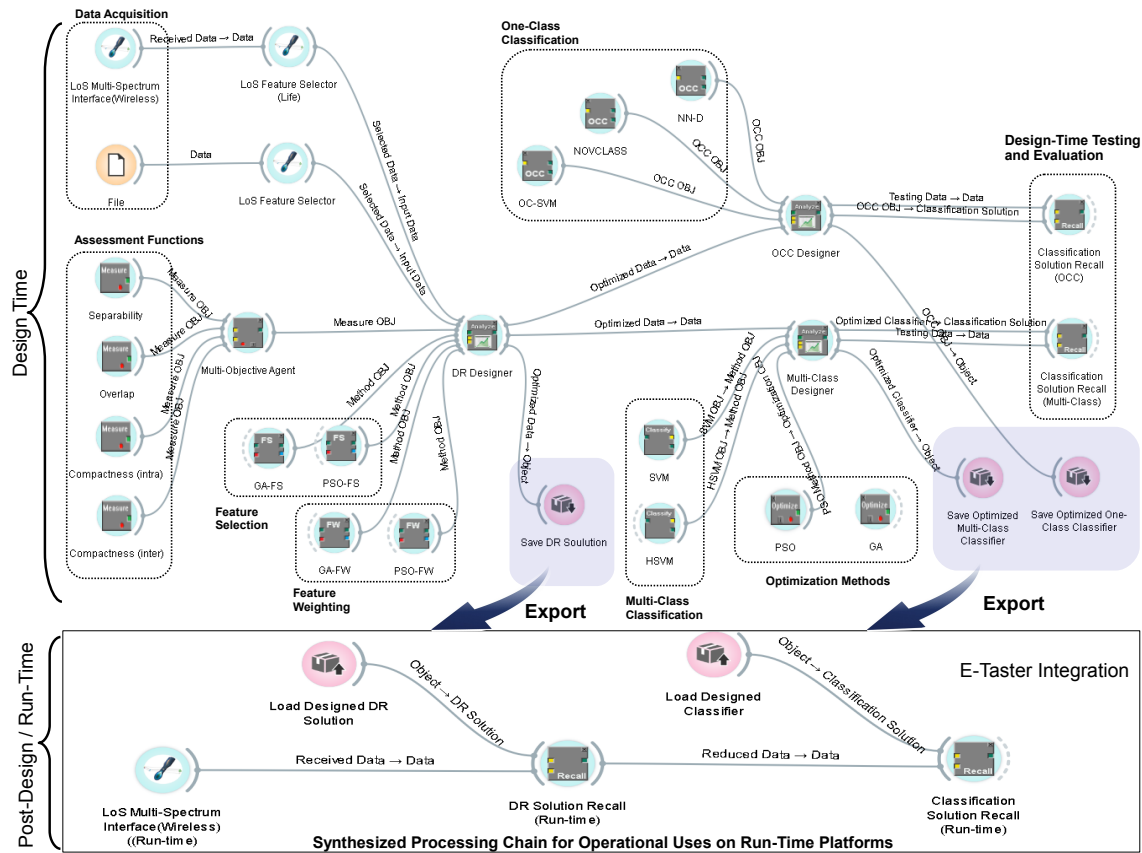


FIGURE 114: Complete processing chain of a LoX application in an automated design and run-time workspace

All experiment cases starting from dimensionality reduction to classification have shown the capability of currently implemented DAICOX architecture in terms of speed-up and facilitate the design process. The summarizing comparisons of the overall design process using conventional approach and DAICOX based on the information and results from the experiments are given in Table 33 with the graphical overview in radar plot as given in Fig. 117. Clearly, DAICOX has significant advantages over manual or conventional approach in all comparing aspects. Figure 115 summarizes the complete design steps, where the designer interactions can take place in addition to the complete design automation flow. Comparing to the conventional flow as have already shown in Fig. 91, the entire design process can be handled by the DAICOX design environment, which reduces significant amount of design effort. The design space complexity of the complete processing chain design for a LoX application with AFW and H-SVM is illustrated in Fig. 116, which is based on sequential evolution scheme and constrained local optimization approach. Also, information of design process status and the intuitive reports of results' quality were given to the designer for design process analysis. Thus, total application design effort and time can be reduced due to a number of automated procedures

and the designer may be involved only in decision making of the final complete solution, which can also be decided autonomously.

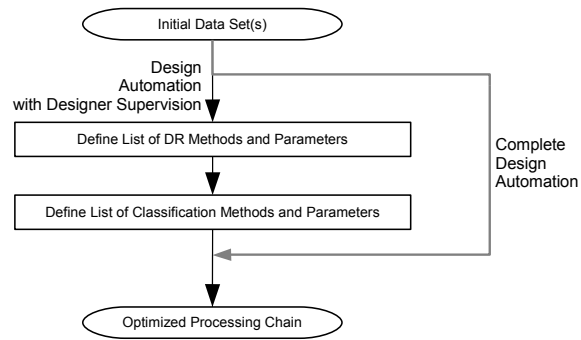


FIGURE 115: Back-end automated design flowchart of DAICOX

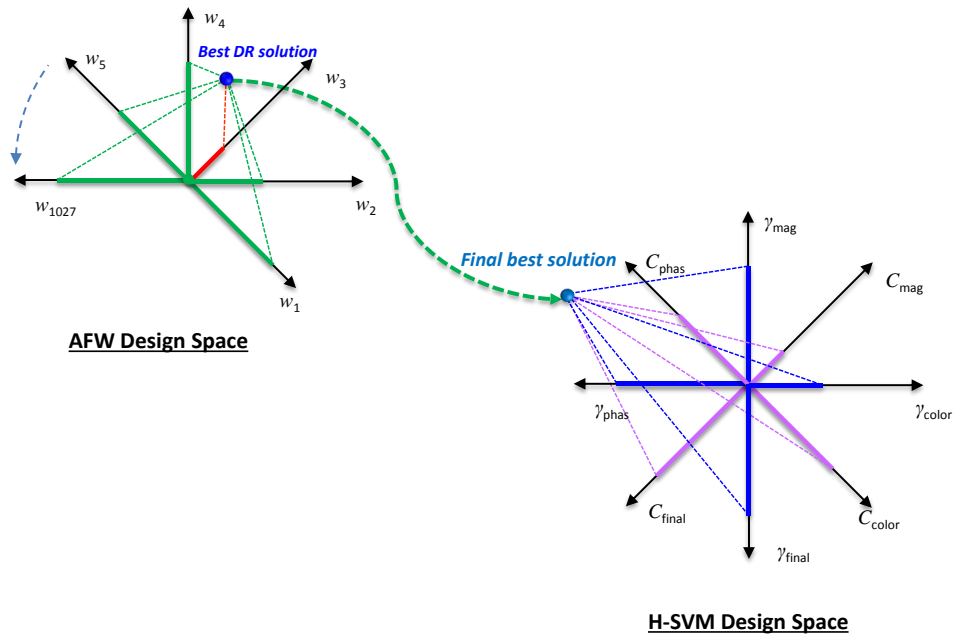


FIGURE 116: An example of reduced design space complexity for the complete LoX application design process with constrained design space regarding the parameter settings given in Table 17

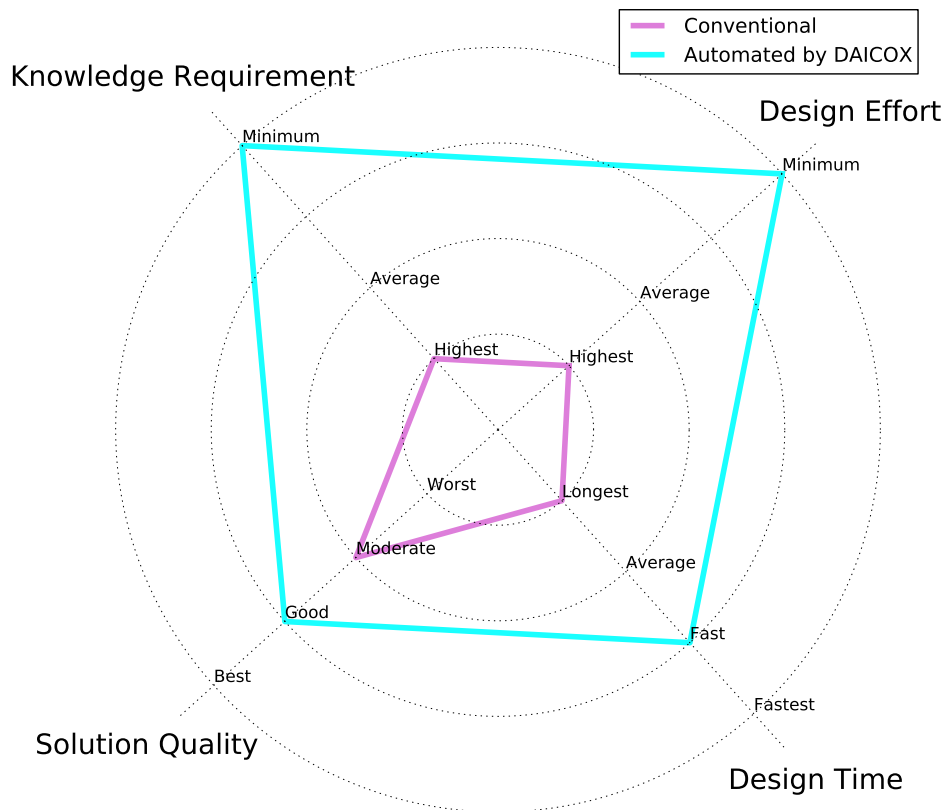


FIGURE 117: Summary of the design aptness of DAICOX compared to conventional design approach

TABLE 33: Overall design process comparison between conventional approach and DAICOX

Classification	
Manual	DAICOX
	<ul style="list-style-type: none"> + Design optimization discovered good quality solutions without need of designer's knowledge. The experienced designer can also help in guiding the search for even better solutions by constraining search space.
<ul style="list-style-type: none"> - The solution quality depends heavily on the designer's experience in performing method selection and parameterization. 	<ul style="list-style-type: none"> + Achieved 5.4% better classification accuracy in average compared to manual approach (see Table 30)
<ul style="list-style-type: none"> - In all experiments, the total number of 351 parameters need to be adjusted by hand in manual design approach. 	<ul style="list-style-type: none"> + Seven optimization parameters (see Table 29) were set covering all experiments, where some can be set automatically based on type of searching problems.
<ul style="list-style-type: none"> - Model analysis may imposes huge effort as it multiplies the design procedure, e.g., $k * (k - 1)$ repetitions in k-fold cross validation case. 	<ul style="list-style-type: none"> + All design cases carried out by employing the appropriate data set separation method with model analysis using k-fold cross validation technique with automated best method selection.
	<ul style="list-style-type: none"> + Reduced design time by 81.87% (see Table 30)
	<ul style="list-style-type: none"> + Design effort does not increase by the size of the design problems (see Table 30)
Processing Chain	
Manual	DAICOX
<ul style="list-style-type: none"> - Manual scripting is usually used to create a processing chain of a design solution. 	<ul style="list-style-type: none"> + Automatically generated an optimized solution in both script or visual programming workspace.
<ul style="list-style-type: none"> - Adjusting the designed solution requires effort in and programming knowledge. 	<ul style="list-style-type: none"> + Designed solution can be adjusted or reconfigured via visualization programming.

6.2 DAICOX Automated Design Support for DeCaDrive System

In this section, the DAICOX architecture was used to facilitate the multi-sensory cognition system design for driving assistance applications. This case study stems from a collaboration with the PhD research project conducted by Li, the DeCaDrive [27, 185], an embedded multi-sensor driver assistance system. IR-depth, vision, vehicle data, e.g., steering wheel sensor information, as well as biomedical information of the driver, e.g., pulse rate or skin impedance, are collected and collaboratively processed for drowsiness detection. In this case study, DAICOX serves as an alternative design platform to the existing solution, which is based on MATLAB and Microsoft Visual Studio by moving to the Orange multi-platform open-access environment and Python programming language. The integration of DAICOX designed solutions into intuitive and informative graphic user interface software environment and the on-line classification are the main focus. The DeCaDrive processing chain is comprised of data acquisition, multi-rate signal processing and feature computation, dimensionality reduction, and classification, which all can be designed by the current DAICOX architecture. Thus, designing the entire process requires a number of complex tasks as shown in Fig. 119, which are facilitated by DAICOX as described in the next subsections.

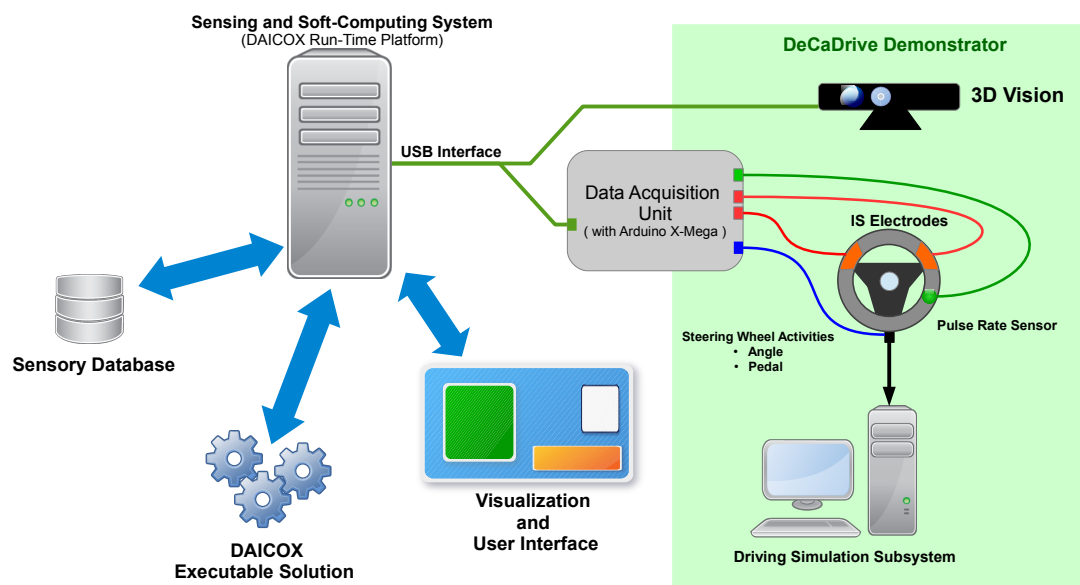


FIGURE 118: DeCaDrive system

6.2.1 Data Acquisition

The DeCaDrive framework has been realized based on a standard PC based driving simulation, sensing, and soft computing subsystems [185]. This comprises a IR-depth

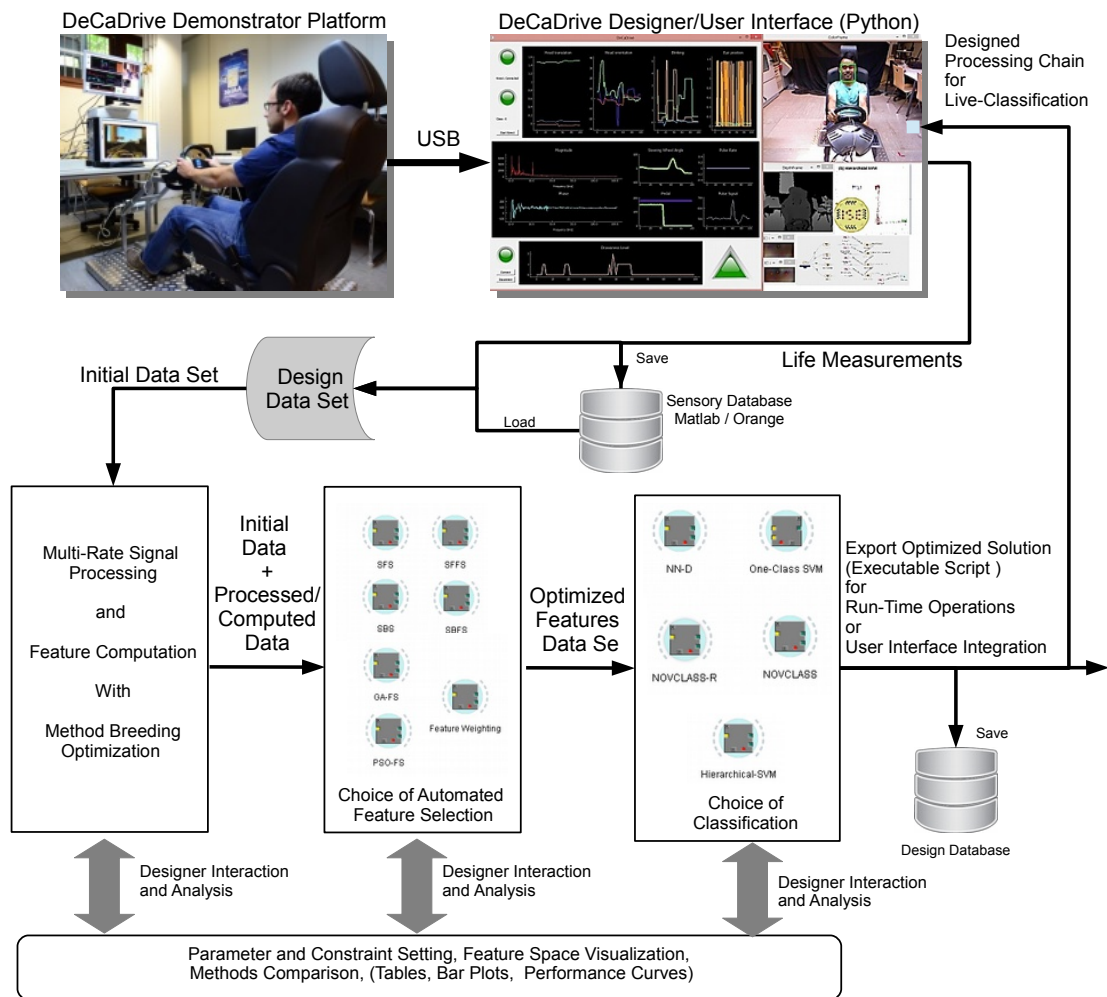


FIGURE 119: Design flow block diagram of DeCaDrive processing chain

camera, using Kinect sensor, for eye finding and facial features, which has been extended to an intelligent multi-sensor system incorporating sensor signal processing as well as diversified embedded sensor interfaces, e.g., pulse rate sensor, steering angle and related driving behavior sensors, and, in particular, impedance spectroscopy which has been reported in [27] with significant contribution to recognition performance improvement. As illustrated in Fig. 124, data links of heterogeneous sensors on steering wheel are connected to the microcontroller based front-end of the multi-sensor interfaces subsystem. The IR-depth camera, as a key component of the image sensing subsystem is connected to PC-based back-end directly via USB interface. The driving simulation subsystem, which is run on a dedicated PC, is used for data collection and for performing a chain of pattern recognition methods for on-line classification. In the current multi-sensory information processing architecture, a data set is collected from depth vision, steering angle, brake and throttle level, driver pulse rate, and impedance spectrum from skin measurement, then, fused at the feature level. The data sets are obtained from five test subjects each conducted by 60 minutes driving simulation on the DeCaDrive system,

where the concise information of physical conditions and simulated driving situations can be found in [27]. Data sets of complementary sensors are synchronized on the same timebase information before being conveyed to feature computation by using re-sampling and interpolation techniques based of time stamps information of each sensory data.

6.2.2 Feature Computation

The synchronized multi-sensor data streams are used to compute a set of features listed in Table 34. Facial information features are computed from preprocessed Kinect data, which includes locations of head, eyebrows, mouth, nose, and other facial components in three dimensions. Eyelid blinking activity information are obtained from object tracking process proposed in [186]. Driving behavioral features are computed from steering wheel sensor resulting in statistical and frequency domain information of steering wheel activities. The feature computation methods, e.g., FFT and function estimation, employ the DAICOX feature computation libraries adapted from the standard Scikit toolbox. Driver state features are health related information of the driver obtained from pulse sensor for heart rate and from impedance spectroscopy for driver's skin impedance. Based on the outcome of feature computation process, these features are fused on the feature level to construct input vectors for pattern classification process explained in the following subsection.

TABLE 34: Descriptions of the computed features of DeCaDrive data

Feature	Sensor	Index	Description
Facial	Kinect	1-3	Head position in x,y and z coordinates
		4-6	Head orientation in x,y and z coordinates
		7,8	Translation and rotation head velocity
		9	Mean Eyebrow position
		10,11	Eye lid closing freq. and duration
		12,13	Steering activity in 1° and 3°
Driving Behavior	Steering Wheel	14	Standard deviation of steering activity
		15	Percentage of minimum steering activity
		16	Mean of absolute position
		17	Steering velocity
		18	Center of FFT-band of steering wheel position
		Driver State	Pulse Sensor
21,22	Mean and std. deviation		
23	Coefficient a (slope) after linear fitting		
IS Sensor	24		Coefficient a after exponential fitting
	25-27		Coefficient a, b and c after polynomial fitting
	28		Coefficient a (slope) after linear fitting
		29-31	Coefficient a, b and c after polynomial fitting

6.2.3 Dimensionality Reduction

The discussed feature computation processes result in 31 features for each timebase synchronized processing window. In early work [185], sequential forward search (SFS) has been used to obtain a feature subset of 8 dimensions applied to the Kinect features performed on QuickCog platform using the q_{OV} measure. In this case study, more variations of the DR process for DeCaDrive are implemented by employing all available feature reduction methods of the current DAICOX architecture. Particularly, the richer choice of feature assessment function is incorporated for the search procedure. The multi-objective assessment function is identical to the previously discussed LoX design cases (see Eq. 6.1). In addition to automated solution selection, the intuitive result reporting via box plot illustrated in Fig. 120 will help the designer in choosing a relevant DR method for a processing chain. The randomly sampled 10% of the data set was used for training procedures, while the remaining was used for generalization procedure. 5-fold cross validation was employed for model analysis and model selection. In this design instance, the sequential methods have shown their efficacy when dealing with moderate dimensionality data. Feature map plots shown in Fig. 121 to 123 can also be used to investigate the performance of the DR process. The SFFS discovered the best quality feature subset with regard to the employed multi-objective criteria, however, the standard deviation was also the highest, whereas feature weighting approaches provided relatively better average result with small deviation. The best feature subset discovered for each approach was selected to be used in the classification design process explained in the next section.

TABLE 35: DR process results of DeCaDrive data

Method	Feature Quality $q_{overall}$	Reduction Rate (%)	$t_{search}[sec.]$	
Feature Selection	SFS	0.547 _(0.031)	27.7% _(7.5%)	41.78 _(1.95)
	SBS	0.501 _(0.018)	56.8% _(25.4%)	54.49 _(5.46)
	SFFS	0.520 _(0.038)	20.6% _(1.6%)	66.92 _(5.55)
	SBFS	0.488 _(0.012)	67.1% _(11.8%)	89.35 _(4.51)
	GA-FS	0.489 _(0.019)	67.7% _(14.7%)	35.86 _(2.05)
	PSO-FS	0.461 _(0.005)	56.8% _(12.0%)	39.96 _(0.20)
Feature Weighting	GA-FW	0.518 _(0.022)	94.2% _(2.4%)	43.49 _(0.60)
	PSO-FW	0.512 _(0.015)	61.3% _(18.1%)	64.94 _(4.68)

Results obtained with fixed random seed number = 12081983

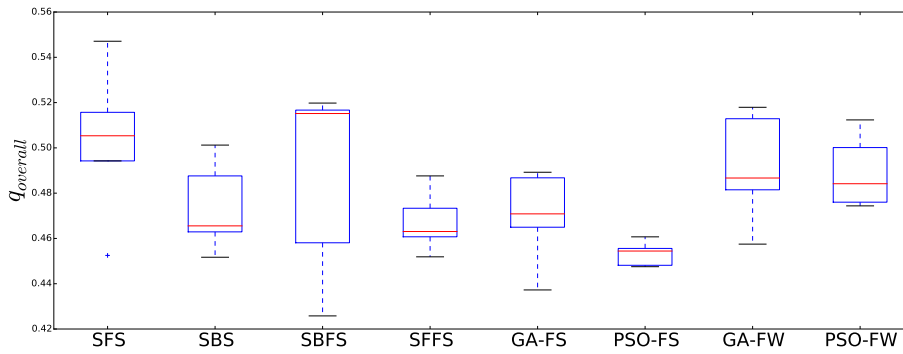


FIGURE 120: Overall feature quality results of the DR design process for the DeCaDrive data set

6.2.4 Classification

Classification is the vital process of the DeCaDrive system, that fuses the computed multi-sensory information to predict the driver's drowsiness state. Three drowsiness levels are defined; Active, Transition, and Drowsy state, which were used to label every recorded data instance. The current DAICOX architecture can serve in automating the classification design process with less effort and experience requirement. The classifier used in the conventional work [27] was built upon a multilayer perceptron (MLP) with manual model selection and parameterization. In this case study, four classification methods were selected to the automated classification design process with parameter optimization and automated model selection. The list of selected classifier consists of kNN, RNN, SVM-C and H-SVM, where the parameter optimization tasks employ GA and PSO. The data set separation and model analysis have the same configuration as used in the DR design process.

The average classification accuracy results are given in Table 36, where the average classification time results are given in Table 37. All classification designs were performed in both manual and automated design approach by using advanced DAICOX design concept. The experimental results in this case study are obtained by the classifiers trained with only 2.5% of the amount of data used in the previous work [27] which applied 8 out from 10 folds for training. The reduced number of training samples reflects better generalization performance while the accuracy may have to be traded-off but the classifier will be robust to large deviations. The classification performance and the general result validity of the DeCaDrive case study is strongly influenced by the rather limited database and the somewhat subjective 'Ground truth' definition of the data.

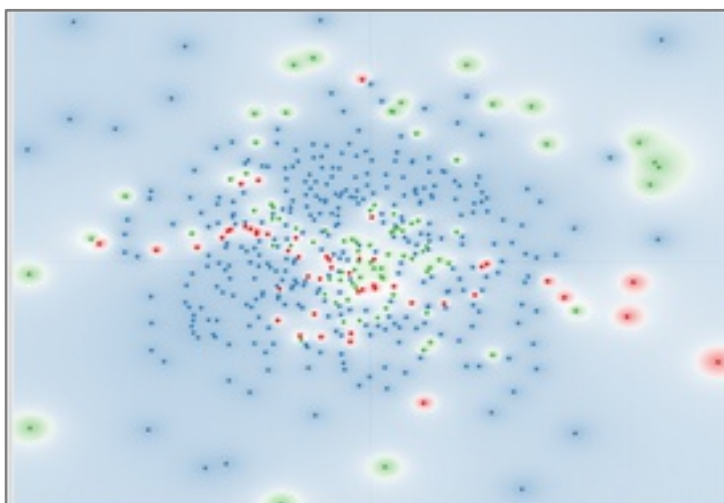


FIGURE 121: Feature map of the the DeCaDrive data set with full features

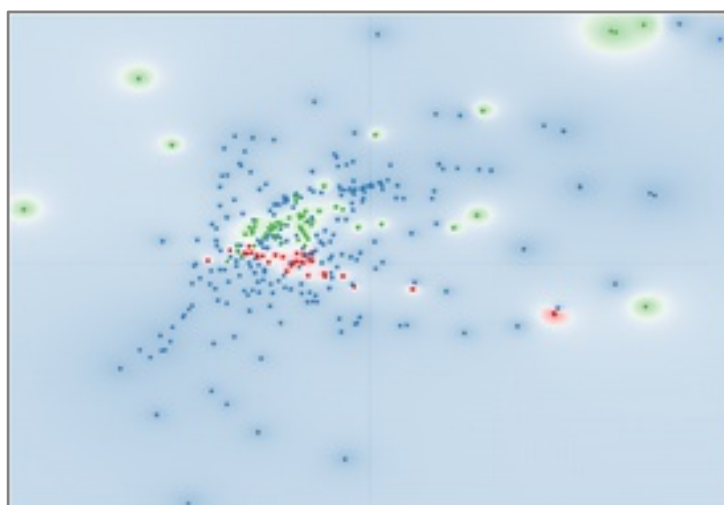


FIGURE 122: Feature map of the the DeCaDrive data set with AFS features

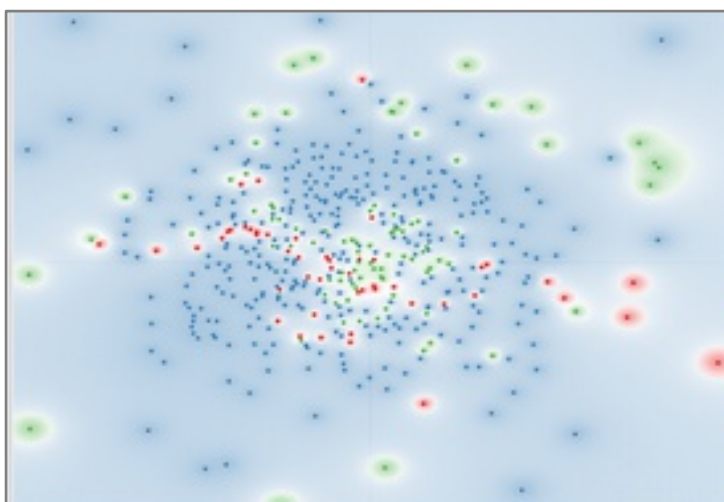


FIGURE 123: Feature map of the the DeCaDrive data set with AFW features

DAICOX outperformed the conventional manual design approach [27] in all cases in terms of classification performance and design effort. The highest classification accuracy was achieved by the H-SVM with applied AFS. The classification time is also an important issue, as the aim of the project is to classify driver state of drowsiness in real-time. Thus, prototype based classifiers may store substantially large number of reference vectors as to get the best fit model. The time used for a classification depend mainly on method's computations and the number of stored prototypes as can be seen if Table 37. The highest classification time method is k -NN due to all training samples are stored to the classification model, whereas the fastest classifier is SVM. RNN significantly reduced the classification time, while improving the classification performance compared to k -NN. Concluding from the obtained results, the standard SVM can be a good candidate for real-time classification purpose. The extension of the H-SVM approach by using distributed computation will result in better classification accuracy with the same speed as the conventional SVM.

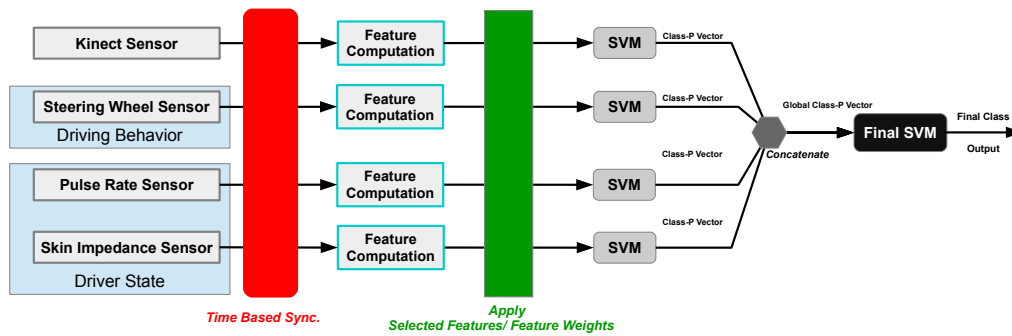


FIGURE 124: Processing structure of DeCaDrive system with hierarchical classification

Table 38 contains classification results obtained from the classifiers generated from the same training data size in accordance with the early work [27] as to investigate the performance comparison of automated design approach and the manual design. The optimized processing chain for the drowsiness detection system has been obtained using the DAICOX automated design environment and encapsulated in a DAICOX run-time executable file. The on-line pattern recognition process of the DeCaDrive incorporates an optimized solution into a comprehensive software environment structures as shown in Fig. 125. In the design optimization layer, a processing chain, e.g., graph based in visual programming, can be given from scratch by a designer or chosen from previous design solutions. The design optimization of DAICOX sequentially evolves the chain by automatically adjusting method parameters and selecting the best performance method of each building block. Then the optimized processing chain can be exported into a

TABLE 36: DeCaDrive classification results of reduced training data

Method	Conventional	DAICOX		
	Full	Feature Set		
		Full	AFS	AFW
kNN	72.34%(0.25%) $k = 5$	76.39%(0.28%) $k = 18$	88.72%(0.14%) $k = 20$	75.72%(0.28%) $k = 20$
RNN	69.84%(0.37%) $k = 5$	71.43%(0.31%) $k = 24$	85.13%(0.19%) $k = 11$	67.61%(0.33%) $k = 21$
SVM	77.51%(0.44%)	80.35%(0.24%)	85.53%(0.19%)	80.45%(0.24%)
H-SVM	-	86.60%(0.18%)	89.21%(0.15%)	84.18%(0.20%)

TABLE 37: DeCaDrive classification time in [μs]

Method	Conventional	DAICOX		
	Full	Feature Set		
		Full	AFS	AFW
kNN	351.75(11.42)	353.15(13.22)	225.87(8.66)	200.23(7.57)
RNN	166.61(6.09)	163.64(5.90)	55.24(2.84)	68.76(2.30)
SVM	32.71(3.14)	32.63(3.21)	10.72(1.55)	7.23(0.47)
H-SVM	-	330.76(9.20)	163.83(2.33)	138.34(4.77)

Results obtained with fixed random seed number = 12081983

TABLE 38: Comparison of the solution obtained by DAICOX and the manual approach from the early work

	Full	AFS	AFW
Li's approach [27] (Manual)	99.22%	99.60%	no report
DAICOX (used 8 of 10 data folds for training)	99.48%	99.66%	99.66%

DAICOX run-time executable file, which can be later utilized and integrated as a multi-sensory cognition processing part of a practical software system. In the DeCaDrive project, a designed solution was integrated to a real-time graphic user interface software for live-classification demonstration purpose. An additional part of the software is the library linker, adapted by using BOOST [187] for Python, which associates the Kinect data acquisition part developed using on C++ to the provide real-time data streams for the processing chain. Thus, this is an instance of the flexibility and seamless operation of DAICOX solution integration with multiple platform/programming languages.

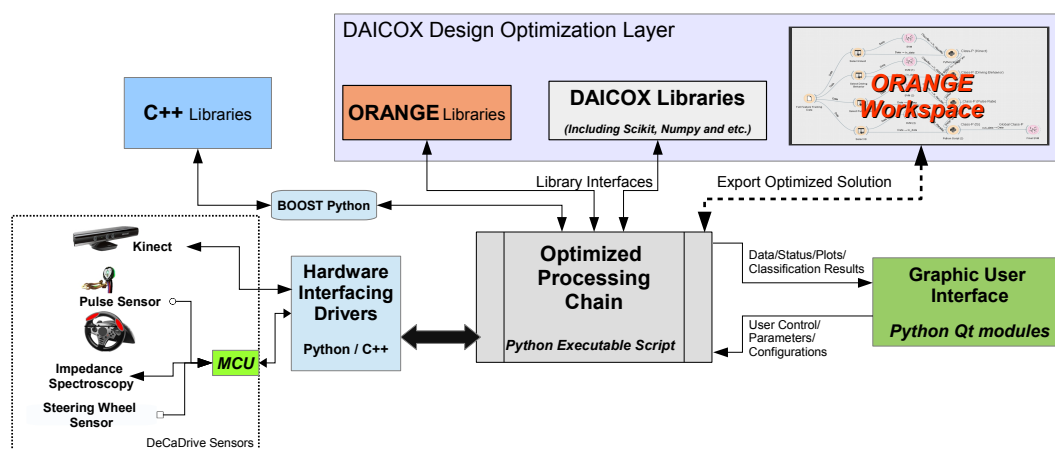


FIGURE 125: Block diagram of DAICOX solution integration for the on-line DeCaDrive system

Solutions of improved performance have been obtained by using DAICOX. Classification results of 99.48% for the full data and 99.66% for the AFS case are the outcomes from effective meta-heuristic optimization under the identical conditions to Li's work [27]. It should also be noted that, the same amount of work and effort of configuring and finding parameters in Li's works [27, 185], which took months to complete, using DAICOX to automatically design the same problem, the design effort minimized to few lines of Python script and small number of steps on the visual programming, which can be done approximately within an hour, and let the design optimization searching for the solution. The searching time in the optimization is the only main time consuming part of the design, which can be reduced by using more powerful computation units or distributed computing machines. Thus, DAICOX has proven itself in this case study as a fast and effort-reduced yet productive and well-performing intelligent system design environment.

6.3 DAICOX Automated Design Support for Magnetic Localization System

In this last case study, the DAICOX architecture has been used to facilitate a design problem of ISE's indoor magnetic localization system [59]. In contrast to common indoor localization techniques, e.g., radio signal based using Received Signal Strength Indicator (RSSI), the system uses artificially generated magnetic fields from fixed position copper wound coils. The inspiration for this approach is to serve the operation in industrially relevant scenarios, in particular, the localization of submerged sensory nodes in stainless-steel fermentation tanks in brewery industry, where radio signal based approaches may be impractical due to high signal absorption in liquid media, which can be alleviated by using very low frequency or even DC magnetic field. The currently developed system consists of a central soft-computing and control unit, a electronic coil switching controller and a high current power supply, magnetic field generation coils fixed in reference positions of the measurement chamber, and autonomous multi-sensor wireless sensor nodes as illustrated in Fig. 126. The generated quasi-DC magnetic field of a certain pulse width, i.e., DC plateau, is acquired by 3-axis AMR sensors equipped on the sensor node. The distance between the source (coil) of the field and the sensor node is estimated from the received magnetic field strength. Combining the distance information from several known-position coils, a 3-dimensional position can be determined by using triangulation or multi-lateration techniques. Advanced mapping approaches have been presented in [59], which significantly improved the accuracy and robustness of the localization mechanism. The comprehensive operation details as well as implemented computational techniques can be found in [59, 123]. Hardware settings and configurations are repeated in Table 39.

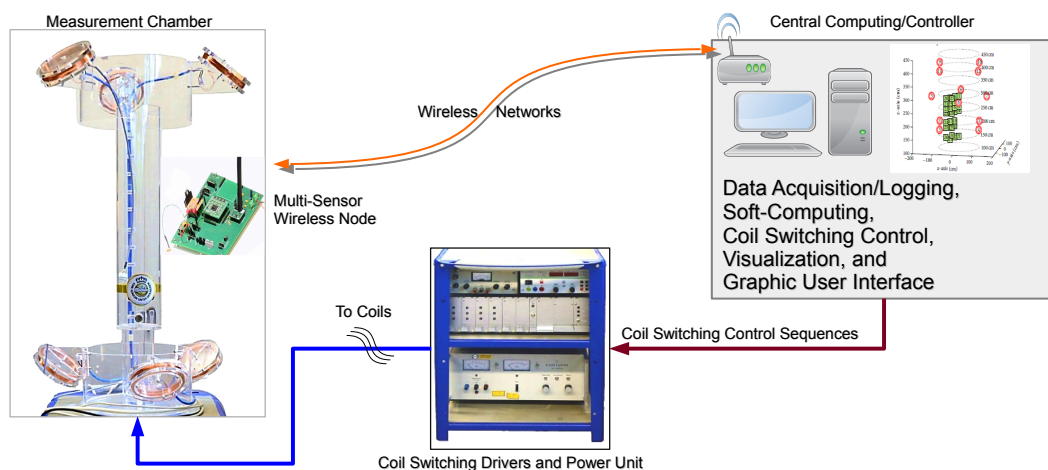


FIGURE 126: Block diagram of the magnetic localization systems

TABLE 39: Technical details of magnetic localization demonstrator used in this case study

dimensions [cm]	R12xH60	sensor type	AFF755
number of coils	6	DAQ system	XMEGA 256A3
coil diameter [cm]	12	ADC resolution	12 bit
number of windings	120	No. of readout/plateau	128
coil current [A]	3	No. of positions	44
coil placement	spherical	num. of coil rings	2

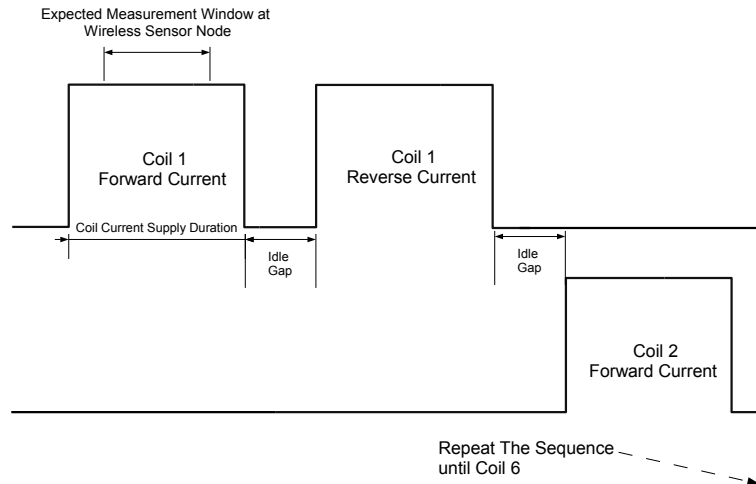


FIGURE 127: Coil switching logic sequence of the first two coils.

6.3.1 Timebase Synchronization for Magnetic Localization based on Pattern Recognition Approach

The discussed distributed sensor node magnetic localization system crucially depends on the knowledge of the timing of each coil's activation on each wireless sensor node. As the coils are sequentially driven in a standard defined pattern, the sensor read-out timing has to be matched with the coil activation time windows (see Fig. 127). A misaligned read-out timing can result in erroneous signal interpretation and a significant deviation in estimated sensor positions, thus, on-line independent synchronization, i.e., self-synchronization is required.

In the recent development of the localization system, we presented a classification based synchronization system as an alternative to conventional approaches [188]. The design task involved in pattern recognition design process, which DAICOX came in to facilitate the design. The main function of the synchronization process is to determine the temporal occurrence of the first coil sequence by detecting the edge shape signal of the generated magnetic field mimicking the same principle applied in magnetic recording application [189]. By this, we adapted pattern recognition techniques to detect edges in a certain processing window size, and, if an edge is detected, estimated timing from

the window position will be used to adjust the sensors' clock for next measurements. The overview of proposed on-line synchronization mechanism is elucidated in Fig. 128. The raw input data is represented by 2,500 ADC samples, i.e., processing window, with 100-sample sliding step. The window size setting was determined from a preliminary study of classification performance over the processing window size as shown in Fig. 130, where it is the point that classification rate begins to stable. Every new 100 samples, the edge detection is performed, which consists of a feature computation task and a one-class classification. The window position will be used to estimate the temporal position of a detected edge and then update the measurement timing.

6.3.2 Dimensionality Reduction

At every processing window, the processing of a raw ADC data stream will be performed by selected Signal Processing (SP) and Feature Computation (FC) methods as listed in Table 40. Several methods can be combined in a processing chain, which was obtained by using the automated SP and FC design toolbox, where a set of selected method combination and optimized parameters were acquired with reduced design effort. The method selection procedure was performed by using GA as search operators and the same feature quality assessment used in the previous case studies measures $q_{overall}$ (see Eq. 6.1) as an assessment function. The optimized processing chain and parameters are given in Table 41 where the average values from 10 runs of all design variable (method selection and parameterization) were used to determine the final solution. Numerous variations of processing chain formation and parameter setting have been discovered from the design optimization process. The final selected method combination was chosen based the highest number of occurrences, thus, LDA and PCA were selected to construct the optimized processing chain for signal precessing and feature computation.

TABLE 40: List of available signal processing and feature computation methods

Method	Description	Parameter Range
Stat	Data Statistics, including mean, median, std., and variance.	-
Polyfit	Compute polynomial function coefficients	[1,8]
HIST	Histogram	bin [2,20]
GausSmooth	Gaussian filter smoothing function	order [0,3], σ [0,10]
FFT	Time to frequency domain transformation	-
PCA	PCA feature computation	No. of PC [1,20]
LDA	LDA feature computation	No. of projections [1]

TABLE 41: Average of the best design solutions of 10 runs

Method	Parameters
PCA	No. of PC = 20
LDA	No. of projections = 1

TABLE 42: Feature space quality comparison

	Original Features	All Methods Selected	DAICOX
	Raw ADC Stream	manual parameterization	optimized processing chain
<i>Q_{overall}</i>	0.623	0.450	0.778

Results obtained with fixed random seed number = 12081983

6.3.3 Edge Classification Design

Four one-class classification methods have been employed for edge recognition task. The training procedure of classification model generation was performed with a supervised and labeled data set, which contains 40 samples of actual edge signal and 68 samples recorded without coil activations. The first are affiliated as “Active” and the latter are affiliated as “Idle” where the “Active” class instances are used to create classification model, i.e., target or normal class. The mapped feature space plot of the data is shown in Fig. 129, which indicates better separation in the computed feature information. The classification performance of the design phase of all methods are given in Table 43. The results are obtained by using 3-fold cross validation. The “Idle” instances were used to evaluate the rejection performance. Automated classifier threshold determination proposed in Section 5.3.2 was performed by using AUC information to find an optimal decision threshold for a particular classifier.

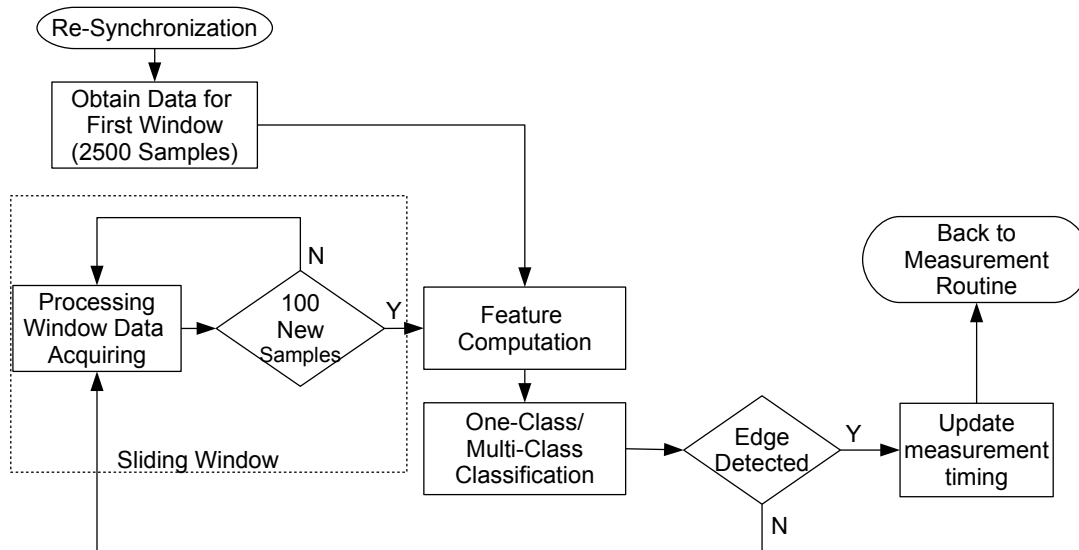


FIGURE 128: Processing structure of classification based edged detection system for measurement synchronization.

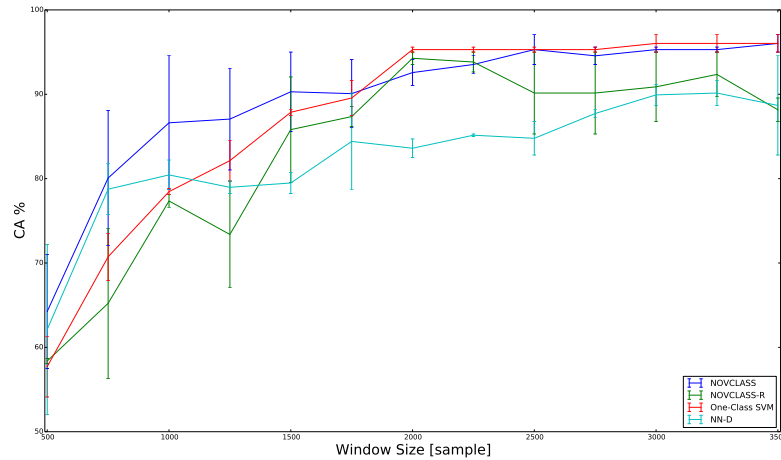


FIGURE 130: Classification performance over processing window size

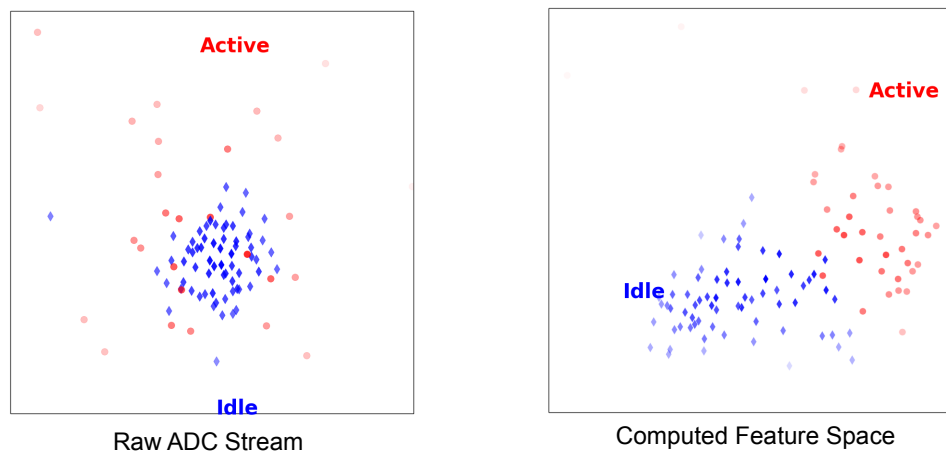


FIGURE 129: Feature space plot of the labeled data set for training procedure. The blue dots are the data collected from "Idle" state and the red dots are the data collected from "Active" state.

TABLE 43: Classification performance in the design phase

Method	True Positive (%)	True Negative (%)
NN-D	98.0 (2.0)	78.0 (1.0)
OC-SVM	99.0 (5.0)	90.0 (3.0)
NOVCLASS-R	100.0 (0.0)	97.0 (4.0)
NOVCLASS	100.0 (0.0)	96.0 (4.0)

6.3.4 Simulated Real-Time Edge Classification

The results in Table 43 represent the preliminary investigations of the capability of the OCC methods in detecting edge signal within the labeled data set. In the next step, elaborated experiments were performed by simulating the real-time operation of the proposed edge detection system. Three recorded AMR sensor data streams of 160,000

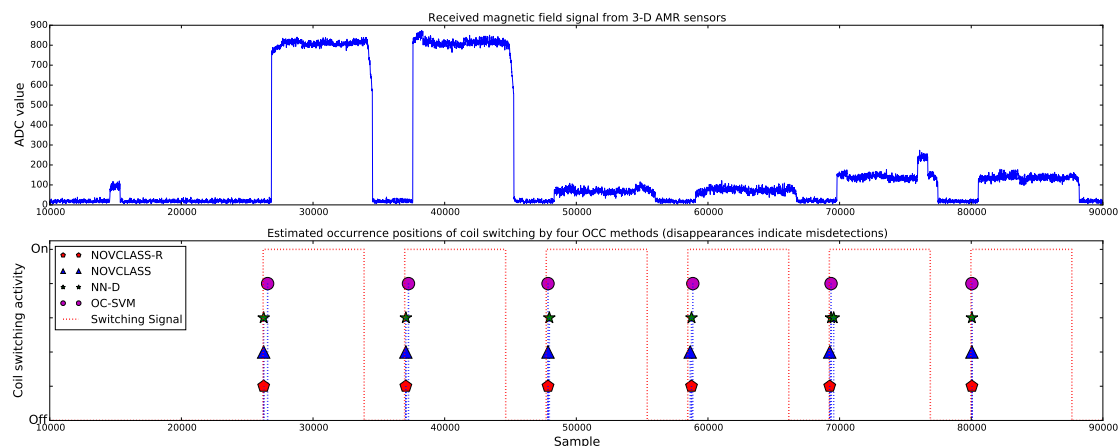


FIGURE 131: Simulated real-time edge classification results of recorded data stream at a position near a corner of the test chamber

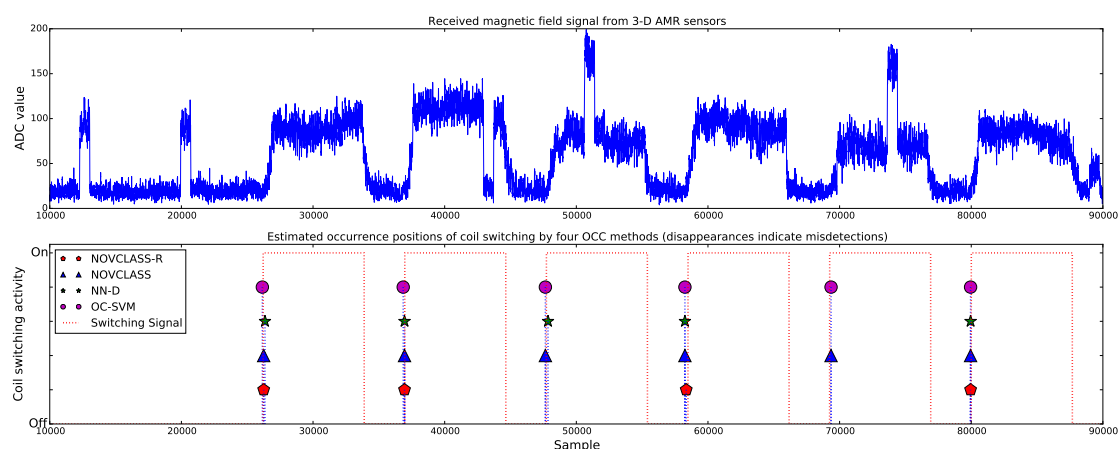


FIGURE 132: Simulated real-time edge classification results of recorded data stream at the center of the test chamber

samples and each were used as simulated real-time signal feed into the processing windows. The edge detection system performed at every new 100 ADC samples, therefore, 1,575 classifications were performed in each data stream. 72 actual edges are contained in the data streams and are associated with recorded coil switching signal in a separated channel, which is used as ground truth for performance evaluation. Figure 131 and 132 illustrate a plot of the first section of the AMR sensor data stream in the top strip with edge detection results in the lower strip, where the markers show detection results of employed OCC methods. The coil switching signal curves can be used as a reference to determine miss detection, i.e., false negative, and the timing error of the estimation. The summarized detection performance are given in Table 44.

TABLE 44: Edge detection performance of the simulated real-time scenarios

Method	72 Actual Edges			Timing Error [μs]
	Detected Edges	Missed Edges	Spurious Detections	
NN-D	43	29	5	176.5
OC-SVM	52	20	8	194.4
NOVCLASS-R	41	31	2	162.8
NOVCLASS	50	22	1	142.2

In Fig. 131 the data was obtained at a position close to the first coil's pulse, which produced the clear edge shape signal as represented in the first two pulses in the data curves. All recorded data streams contain a level of background noise and spurious pulses, which is shown in Fig. 132 for a more extreme case. The robustness of the edge detection system can be proven in this circumstance. The overall results indicate the potential capability of using classification based edge detection system in part of on-line synchronization mechanism. The timing error of the estimated edge location compared to the actual coil switching signal can be an important issue for future development, where a remapping approach can be employed to compensate multiple edges themselves. The discovery of the first edge is also necessary for re-adjusting sensors' clock. An undetected first edge leads to incomplete timing information especially when the last pulses are also undetected. The different pulse width or pulse gap can be defined as an identity of a particular coil so that the timing between two consecutive edges can be used to identify a coil in the sequence. Then, by this piece of information, the temporal position of the first pulse can be consistently estimated for appropriately resynchronizing the measurement timing.

The design effort in this case study was determined just by the selection and labeling of the training data set, which is common in typical classification design tasks. The parameter tuning tasks were done by DAICOX as well as the model analysis for selecting an optimized classifier of each method. Table 45 compares overall classifications results of the solution obtained manually and by using DAICOX in the simulated real-time scenario. The manual design approach was performed without classification threshold tuning and model analysis. It is clear that all design cases performed by DAICOX produced better performance solutions. This is due to the fact that the generalization performance of OCC methods strongly depends on a decision threshold, which is dramatically different in the OC-SVM case. In DAICOX, an automatic determination of a relevant and generalized threshold setting is supported.

TABLE 45: Comparison of generalization edge detection results by using DAICOX compared to manual approach

	Manual [59]		DAICOX	
	CA (%)	Timing Error [μs]	CA (%)	Timing Error [μs]
NN-D	91.01	239.8	97.65	176.5
OC-SVM	72.43	361.84	98.98	194.4
NOVCLASS-R	91.24	182.4	98.85	162.8
NOVCLASS	94.32	157.9	99.23	142.2

The investigation of pattern recognition based timebase synchronization system has proven the feasibility of DAICOX, where high detection performance was achieved. DAICOX has been used to automatically design the pattern recognition methods throughout the complete processing chain and find the best suited one. The applicability of using DAICOX for this application instance can be listed below.

- The design effort has been significantly reduced in comparison to the previous work [59], that was manually developed on MATLAB platform [59].
- The dependence of achievable solution quality and performance on the designer's knowledge and expertise has been reduced.
- Visualizations of DAICOX provide better understanding to the designer in order to enhance the solution obtained from automated process.

Summarizing from the highlighted advantages, in accordance with the previous case studies, the outcomes in this case study emphasize the aptness and advantage of using DAICOX in broad applications.

6.4 Discussion

DAICOX was successfully applied to three real-world practical application case studies and results equal or superior to the existing manually designed solutions could be demonstrated. The presented experiments have been conducted with the automated design process features of DAICOX given in the following.

- Automated method optimization of a complete processing chain
- Automated method selection and processing chain combination
- Optimization of feature computation and signal processing methods' parameters and automated method combination
- Design optimization from choices of methods with multi-objective optimization
- Design performance visualization and visual programming

The reported results indicate the significance of employing DAICOX for automatically design intelligent multi-sensor systems in delivering good quality solutions. The design effort has been reduced in several cases as a result from the automated design process, design optimization, visual programming, and information visualization features. The design assistance environment of DAICOX help accelerate technical cognition application designs with less knowledge requirements from the designer. These achievements partially fulfill the goals of the complete DAICOX architecture, where numerous improvements will be in the next steps of developments. In particular, intrinsic optimization concept and self-x properties will be in the focus to obtain well tailored and fully robust solutions for multi-sensor cognition applications. In addition, open issues for ongoing and future development, e.g., graph-based processing chain breeding optimization will also be pursued next.

Chapter 7

Summary and Conclusion

The emerging intelligent and distributed technologies, e.g., IoT/IIoT, AAL/AmI, Industrie 4.0, and CPS/CPDS have a strong and increasing need for integrated intelligent multi-sensory cognition systems in rich diversity and rapid availability [1, 2]. However, the majority of design approaches of such systems are human expert based, which is prone to suboptimal results, time consuming, and costly. To tackle these issues, the design automation for intelligent integrated multi-sensor systems has been pursued and presented in the thesis.

The Design Automation for Intelligent COgnitive system with self-X properties: DAICOX architecture has been conceived with the aim to reduce the design effort and provide high quality and robust solutions. The goal of the proposed DAICOX architecture consists of multi-objective design optimization algorithms, a collection of pattern recognition methods, intuitive design environment for fast and transparent application developments. In particular, dynamic reconfiguration, intrinsic evolution, and self-x properties concepts are aimed on to be integrated into hardware instances for providing dependable and robust measurement system solutions.

Summarizing from the outcomes of the research work, due to the complexity of the task in the research work, neither all of the established goals have been achieved yet nor has the complete architecture definition been fully implemented. From the developed tool, framework, and conducted real-world application case studies, the outcomes and the novel contributions are listed in the following outlines. Together, according to the investigated state of the art research and tools in the same area of the thesis interest, Table 46 and 47 show the features of the currently implemented DAICOX architecture outlined by the same criteria of the survey as presented in Chapter 3 (see Table 2 and Table 3), where the highlighted cells indicate better features compared to the best feature available in the state of the art. In Section 7.1, the key results and novelties will be summarized, and in Section 7.2, the potential future improvements and open issues of

the developed framework and tool are discussed.

7.1 Outcomes and Novel Contributions

- A comprehensive intelligent multi-sensor system design architecture with automated method selection, parameterization, and model analysis was developed.
- The design automation is incorporated with multi-objective meta-heuristic design optimization, that provided high performance solutions and outperformed standard conventional design results as listed below.
 - 5.4% better classification rate in average in the LoX application designs.
 - Outperformed classification rate (99.66%) of the early system (99.60%) in the DeCaDrive case study.
 - 11.4% classification rate improvement in the magnetic localization case study.
 - Reduced human (expert) resource consumption by 81.8% compared to conventional approach in the LoX case study.
- A number of developed methods and variations have been presented including:
 - Automated processing chain construction and parameterization for signal processing and feature computation.
 - Advanced dimensionality reduction methods, e.g., feature weighting based on GA and PSO with multi objective feature quality assessment.
 - A novel modification of a non-parametric compactness measure for feature space quality assessment.
 - Decision level sensor fusion architecture based on hierarchical classification approach using SVM, i.e., H-SVM.
 - A collection of one-class classification methods and a novel algorithm variation, i.e., NOVCLASS-R, for resource reduction.
 - Automated design toolboxes supporting front to back design with automated model selection and information visualization.
- An intuitive design environment for multi-sensor cognition applications is developed, which includes visual programming concept and design visualizations.
- Knowledge base and reuse of previously designed solutions for further elaboration and extension, as well as serving as seed solutions to speed-up the design of new applications.
- Extendable modular method libraries and toolboxes developed in complying with open-source multi-platform software.

TABLE 46: DAICOX features with regard to the criteria investigated in the state of the art survey

	Sensor Type and Interface Type	Sensory Data Format	Dimensionality Reduction	Data Analysis and Mining	Decision Making / Learning	Optimization/ Learning	Knowledge Rule Import/Export
DAICOX	Generic Multi-Sensor DAQ, Sensor Networks and Load From Database	Heterogeneous, Multi-Channel	Feature Computation/ Feature Selection/ Feature Weighting	✓ H-SVM and OCC	Automated (Based on k -fold CV)	GA and PSO Based Multi-Objective	Import/Export
Best of SoA	Sensor Node Generic Sensors [8]	Heterogeneous, Multi-Channel [5–8, 15]	Feature Selection Feature Extraction [3, 13, 14]	✓ Multi-Class Classification [3, 4, 6, 7] [9, 12–15]	Automated (Based on k -fold CV) [3, 12]	GA and PSO Based Multi-Objective [12]	Import/Export [10]

TABLE 47: DAICOX features with regard to the criteria investigated in the state of the art survey *cont.*

	Operating Platform/ Dependent Tool	License	Visualization	Graphical Programming	Expendable Library /Programming Language	Reusable/ Executable Solution	self-x
DAICOX	WIN,MAC,Linux with Python	Open-Source	Signal and Feature Space Visualization	✓ (via Orange Canvas)	✓ Python/C++	✓ Executable Python Script and GUI integration	In prototype
Best of SoA	WIN,MAC,Linux with Python [14]	Open-Source [3, 6–8] [9, 10, 14]	✓ Feature Space Image Analysis [13, 14]	✓ [6, 7, 9, 15] [10, 13, 14]	✓ Python/C++ [14]	✓ [6–8, 15] [10, 12, 13]	Self-Tuning [5]

7.2 Future Improvements of DAICOX

The current state of the research work of the thesis is being pursued towards the complete DAICOX architecture concept, where the currently implemented tool and framework can serve as a development vehicle. The list of potential future improvements is given in the following.

- Extension to a richer choice of methods and algorithms of the method pool, in particular, signal processing and feature computation.
- Graph-based processing chain breeding and optimization approach.
- Intrinsic optimization in both static and dynamic approach.
- Complete self-x functionalities integration with dynamically reconfigurable electronics of hardware instances.
- Extension to support industrial applications, in particular, automated intelligent condition monitoring system design [28, 29, 190] in compliance to the Industrial Internet of Things (IIoT) and Cyber Physical Production Systems (CPPS) technologies.

These objectives will consolidate the DAICOX architecture for further contributions to advanced sensing and measurement technologies, in particular, CPPS, Industrie 4.0, and IIoT. Based on ISE previous research works, pursued within the last two decades, ongoing enhancements of this thesis, and future improvements, DAICOX has the potential to become an EDA system for complex applications in diverse application domains, e.g., meeting the postulated design support demands of Industrie 4.0, CPS/CPPS, IoT/IIoT, and general distributed intelligent measurement systems.

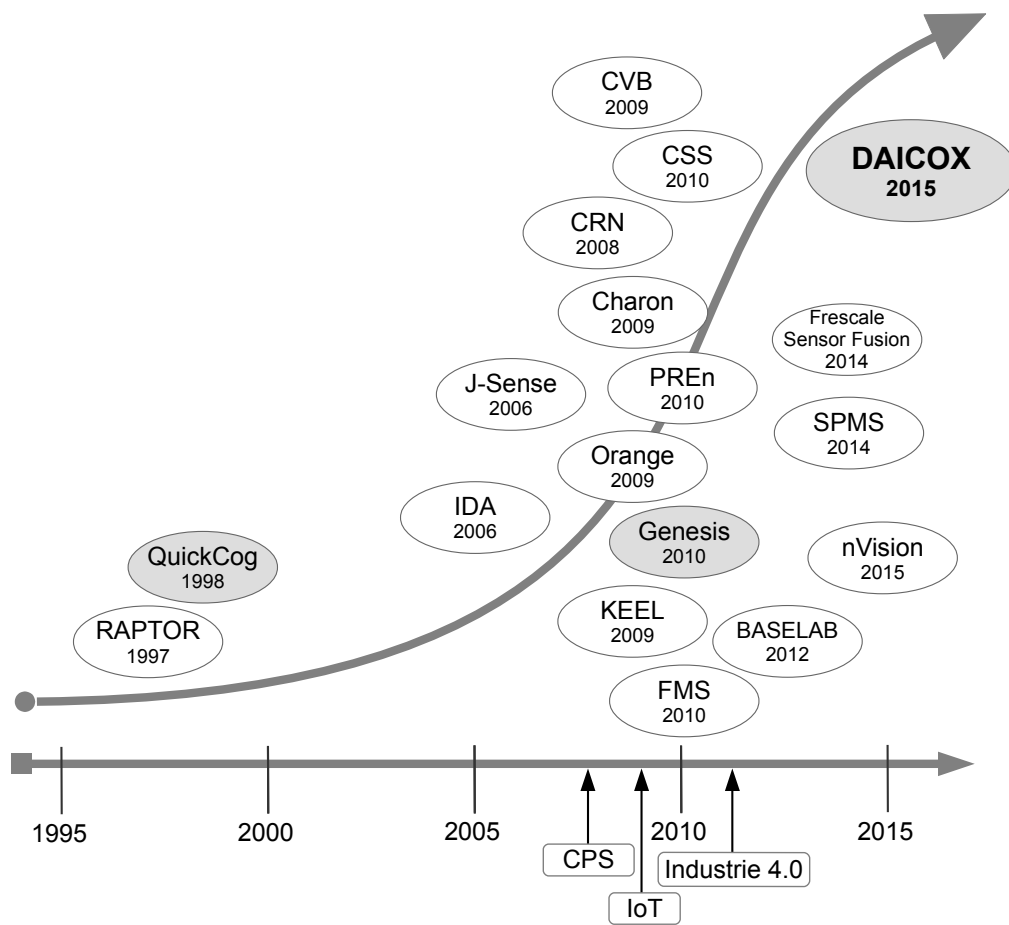


FIGURE 133: Technological time-line of the state of the art and the commencement of DAICOX

List of Abbreviations

ADC	Analog to Digital converter
AFE	Analog Front-End
AFS	Automated Features Selection
AFW	Automated Features Weighting
AmI/AAL	Ambient Intelligence/Ambient Assisted Living systems
ASIC	Application Specific Integrated Circuit
AUC	Area Under Curve
CA	Classification Accuracy
CI	Computational Intelligence
CPS	Cyber Physical Systems
CPPS	Cyber Physical Production Systems
DR	Dimensionality Reduction
EC	Evolutionary Computation
FC	Feature Computation
FE	Feature Extraction
GA	Genetic Algorithm
GUI	Graphic User Interface
HC	Hierarchical Classification
IIMSS	Intelligent Integrated Multi-Sensor Systems
IoT	Internet of Things
IIoT	Industrial Internet of Things
<i>k</i>-CV	<i>k</i> -Fold Cross Validation
<i>k</i>-NN	<i>k</i> -Nearest Neighbor Classification
LDA	Linear Discriminant Analysis
LoX/LoS/LoF	Lab-on-X/Spoon/Fork
MDS	Multi-Dimensional-Scaling
MOO	Multi-Objective Optimization
NN-D	Nearest Neighbor Description
NOVCLASS	Novelty Classifier
OCC	One-Class Classification
PCA	Principal Component Analysis

PSO	Particle Swarm Optimization
RBF	Radial Basis Function
RNN	Reduced Nearest Neighbor
ROC	Receiver Operating Characteristic
SFS/SBS	Sequential Forward/Backward Search
SFFS/SBFS	Sequential Forward/Backward Floating Search
SP	Signal Processing
SVDD	Support Vector Data Description
SVM	Support Vector Machines
TP	True Positive
TN	True Negative

List of Symbols

Global definitions, unless otherwise defined	
X or \mathbf{x}	Input data vector
\mathbf{X}	Input data matrix (data set)
\mathcal{X}	Possible feature subsets of X
L	Number of classes
M	Number of Features
N	Number of Instances
A	Feature transformation function
\mathcal{A}	Possible set of transformations
J	Feature assessment criteria
$ X = d$	cardinality of X
O	Computational complexity
K	Kernel function

Support Vector Machine	
C	Penalty control parameter of SVM [175]
ν	Boundary control parameter of SVM
γ	Kernel width of RBF kernel function

NOVCLASS	
t	Reference vector (prototype)
R	Radius of reference vector
η	Scaling factor

Distance metric	
d	Distance or dissimilarity function value

None parametric quality measure	
q	quality assessment function
dNN	Distance of nearest neighbor
ω	Class label
δ	kronecker delta function

Optimization	
x	Design variable vector
D	Design variable dimensions
f	Fitness functions
GA:	
P_s	Selection rate
μ	Mutation rate
PSO:	
v	Particle velocity
p_b	Particle best
g_b	Population best
C_1	Social learning factors
C_2	Individual learning factors
w	Inertia weight

Appendix A

DAICOX Software Toolboxes and Libraries

This appendix provides technical information for the purpose of practical uses and the extensions of the developed software of the currently implemented DAICOX. Figure 134 shows the taxonomy of the implemented DAICOX software structure categorized by its functionality. Each implemented method is provided in form of both Python script and Orange widget toolbox for visual programming. Using method widgets, the interface information required by each method are provided in Tab. 50 and Tab 51.

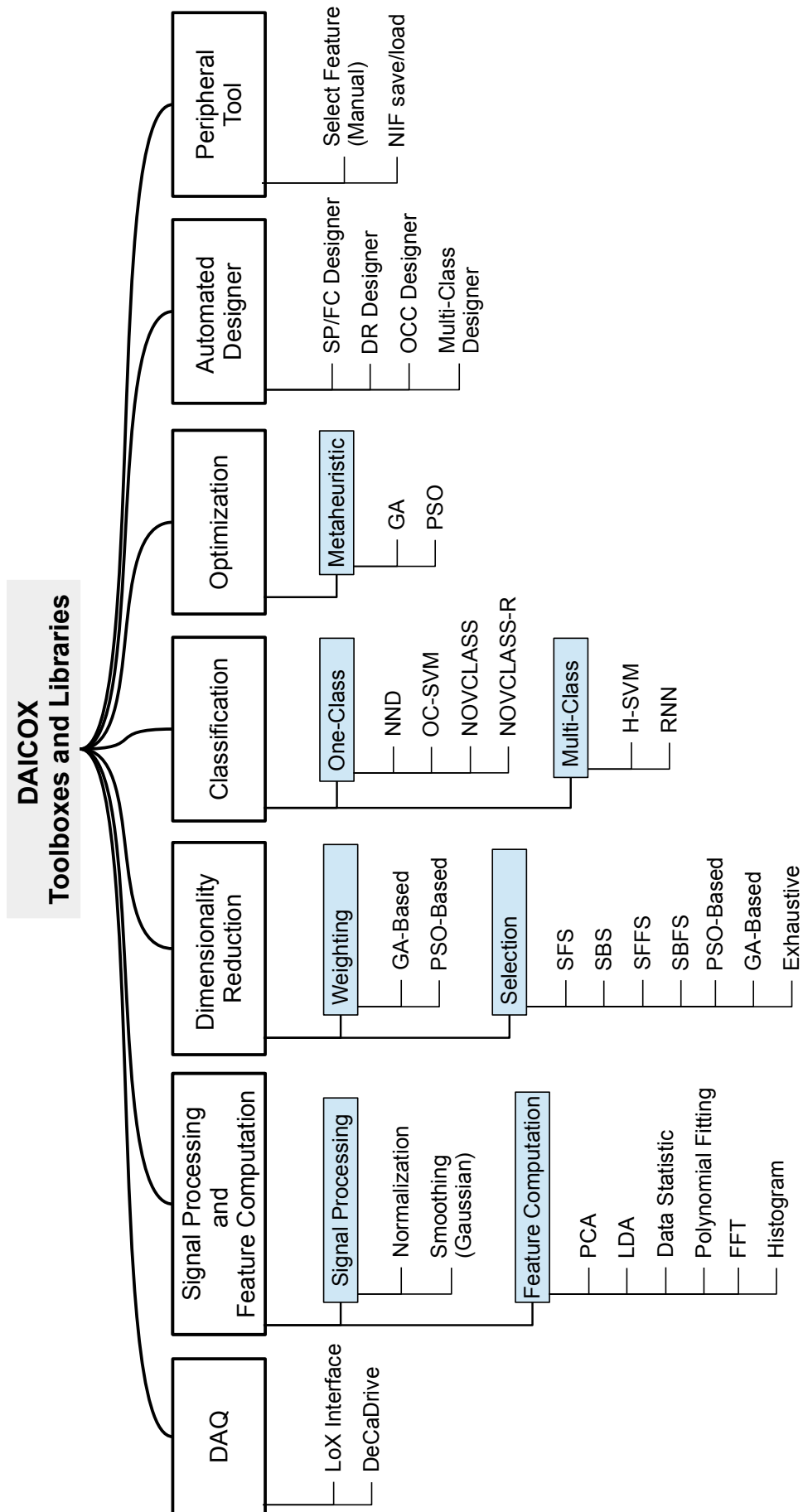


FIGURE 134: Taxonomy of DAICOX software

TABLE 50: DAICOX toolbox interface

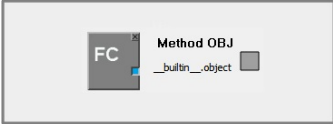
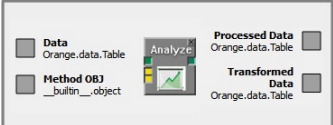


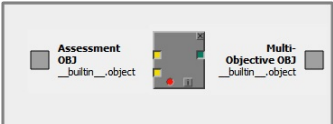
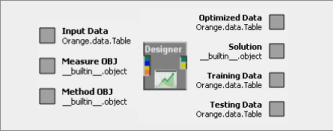
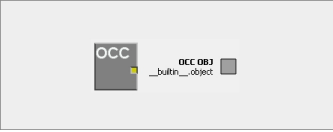
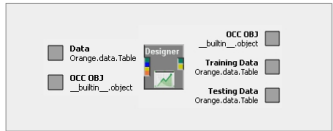
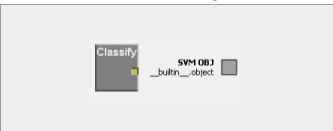
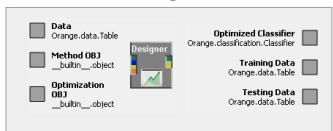

Type/ Widget Interface	Input Interface Output Interface	Toolboxes (Named Widgets)
<p>Feature Computation & Signal Processing Method</p> 	<p>Input</p> <ul style="list-style-type: none"> - <p>Output</p> <ul style="list-style-type: none"> • Method Object 	<ul style="list-style-type: none"> - Stats - Polyfit - HIST - FFT - Normalize - GausSmooth - LDA - PCA
<p>Feature Computation & Signal Processing Designer</p> 	<p>Input</p> <ul style="list-style-type: none"> • Training Data • Data • Method Object (Multiple) • Optimization Object <p>Output</p> <ul style="list-style-type: none"> • Processed Data • Transformed Data 	<ul style="list-style-type: none"> - SP&FC Designer
<p>Feature Selection & Feature Weighting Method</p> 	<p>Input</p> <ul style="list-style-type: none"> • Data • Assessment Object <p>Output</p> <ul style="list-style-type: none"> • Selected Feature Data • Selected Feature Domain • Method Object 	<ul style="list-style-type: none"> - SFS - SBS - SFFS - SFBS - PSO-FS - GA-FS - PSO-FW - GA-FW
<p>Feature Assessment Function</p> 	<p>Input</p> <ul style="list-style-type: none"> - <p>Output</p> <ul style="list-style-type: none"> • Assessment Object 	<ul style="list-style-type: none"> - Overlap - Compactness - Separability
<p>Multi-Objective Agent</p> 	<p>Input</p> <ul style="list-style-type: none"> • Assessment Object(Multiple) <p>Output</p> <ul style="list-style-type: none"> • Assessment Object 	<ul style="list-style-type: none"> - Multi-Objective Agent

TABLE 51: DAICOX toolbox interface *cont.*

Type/ Widget Interface	Input Interface Output Interface	Toolboxes (Named Widgets)
<p>Dimensionality Reduction Designer</p> 	<p>Input</p> <ul style="list-style-type: none"> • Data (Multiple) • Method Object (Multiple) • Assessment Object <p>Output</p> <ul style="list-style-type: none"> • Best Solution Data Table • Best Solution Object 	<p>- DR Designer</p>
<p>One-Class Classification Method Object</p> 	<p>Input</p> <ul style="list-style-type: none"> • - <p>Output</p> <ul style="list-style-type: none"> • Method Object 	<p>- NN-D - OC-SVM - NOVCLASS</p>
<p>One-Class Classification Designer</p> 	<p>Input</p> <ul style="list-style-type: none"> • Data Table • Method Object (Multiple) <p>Output</p> <ul style="list-style-type: none"> • Optimized Method Object 	<p>- OCC Designer</p>
<p>Multi-Class Classification Method Object</p> 	<p>Input</p> <ul style="list-style-type: none"> • - <p>Output</p> <ul style="list-style-type: none"> • Method Object 	<p>- kNN - RNN - HSVM</p>
<p>Multi-Class Classification Designer</p> 	<p>Input</p> <ul style="list-style-type: none"> • Data Table • Method Object (Multiple) • Optimization Object <p>Output</p> <ul style="list-style-type: none"> • Optimized Method Object 	<p>- Multi-Class Designer</p>
<p>Designed Solution Recaller</p> 	<p>Input</p> <ul style="list-style-type: none"> • Data Table • Solution Object <p>Output</p> <ul style="list-style-type: none"> • - 	<p>- SPFC Solution Recall - DR Solution Recall - Classification Solution Recall</p>

Appendix B

Parameter Settings

This appendix provides the obtained parameter settings of all conducted experiments.

TABLE 52: Experimental results and parameter settings of flat SVM for LoX and DeCaDrive data sets

Data set	Flat SVM (full features)		Flat SVM with AFS		Flat SVM with AFW	
	C	γ	C	γ	C	γ
UsedOil	3.45e+03	1e-05	4.83e+03	9.99	3.45e+03	1e-05
Oil	3.45e+03	1e-05	8.21e+03	0.0615	3.45e+03	1e-05
Beer	4.74e+03	10.0	3.28e+03	1e-05	4.74e+03	10.0
7Wine	8.89e+03	7.56	3.45e+03	1e-05	3.45e+03	1e-05
SoyVine	3.45e+03	1e-05	3.45e+03	1e-05	3.45e+03	1e-05
Milk	8.89e+03	7.56	9.56e+03	0.0101	3.45e+03	1e-05
WineGly	5.49e+03	0.0501	3.14e+03	1e-05	3.86e+03	1e-05
Salt	9.45e+03	1.17	9.48e+03	0.54	8.77e+03	1.29
Powder	1.0	1e-05	2.69e+03	0.326	9.96e+03	0.196
DeCaDrive	3640.777	6.015	3433.727	8.650	2640.986	9.353

TABLE 53: Hierarchical SVM parameter settings for LoX and DeCaDrive data sets.

LoX case study									
Data set		C_{color}	γ_{color}	C_{mag}	γ_{mag}	C_{phase}	γ_{phase}	C_{global}	γ_{global}
Full Feature	UsedOil	$7.71e^3$	5.45	$9.19e^3$	4.72	$4.64e^3$	8.39	$3.22e^2$	2.7
	Oil	$7.56e^3$	9.76	$9.91e^3$	7.56	$8.73e^3$	8.16	$7.56e^3$	7.56
	Beer	1.0	$1e^{-5}$	$7.07e^2$	$1e^{-5}$	1.0	1.9	1.0	1.6
	7Wine	$3.35e^3$	3.03	$2.13e^3$	0.0867	$2.92e^3$	0.589	$8.77e^2$	0.149
	WineGly	$5.05e^3$	2.71	$6.63e^2$	$1e^{-5}$	$4.64e^3$	$1e^{-5}$	$3.14e^2$	0.0391
	SoyVine	$3.16e^2$	2.73	$1.02e^3$	5.89	$2.04e^3$	4.95	$9.54e^3$	2.75
	Milk	$3.59e^2$	7.37	$2.28e^3$	0.0188	$2.6e^3$	0.00284	$1.08e^3$	0.138
	Salt	$4.5e^2$	7.36	$3.11e^3$	$1e^{-5}$	$3.48e^3$	$1e^{-5}$	$1.47e^3$	0.184
	Powder	$7.9e^3$	7.56	$7.56e^3$	7.56	$7.56e^3$	8.45	$7.56e^3$	8.32
DeCaDrive case study									
		C_{Kinect}	γ_{Kinect}	C_{DrBhv}	γ_{DrBhv}	$C_{DrState}$	$\gamma_{DrState}$	C_{global}	γ_{global}
DeCaDrive	Full	$5.88e^3$	9.11	0.1	4.59	$1.09e^4$	2.98	$2.78e^3$	0.857
LoX case study									
Data set		C_{color}	γ_{color}	C_{mag}	γ_{mag}	C_{phase}	γ_{phase}	C_{global}	γ_{global}
AFS	UsedOil	$4.5e^2$	7.36	$3.11e^3$	$1e^{-5}$	$3.48e^3$	$1e^{-5}$	$1.47e^3$	0.184
	Oil	$4.68e^3$	2.66	84.0	9.43	$1.95e^3$	4.27	$1.2e^2$	1.67
	Beer	1.0	$1e^{-5}$	$2.12e^3$	$1e^{-5}$	1.0	$1e^{-5}$	1.0	4.81
	7Wine	$2.91e^2$	4.58	$1.94e^3$	0.0136	$2.16e^3$	$1e^{-5}$	$3.19e^3$	0.115
	WineGly	$5.01e^2$	2.67	$2.38e^3$	0.0386	$2.95e^3$	0.146	$1.1e^3$	0.153
	SoyVine	$3.16e^2$	2.73	$1.02e^3$	5.89	$2.04e^3$	4.95	$9.54e^3$	2.75
	Milk	$1.47e^3$	5.54	$6.37e^3$	0.409	$4.11e^3$	$1e^{-5}$	$1.29e^3$	0.227
	Salt	$4.5e^2$	7.36	$3.11e^3$	$1e^{-5}$	$3.48e^3$	$1e^{-5}$	$1.47e^3$	0.184
	Powder	$1.36e^3$	$1e^{-5}$	$9.32e^2$	$1e^{-5}$	1.0	$1e^{-5}$	1.0	8.17
DeCaDrive case study									
		C_{Kinect}	γ_{Kinect}	C_{DrBhv}	γ_{DrBhv}	$C_{DrState}$	$\gamma_{DrState}$	C_{global}	γ_{global}
DeCaDrive	AFS	$2.7e^3$	0.356	$2.42e^4$	0.0256	$1.32e^4$	1.45	$5.58e^2$	0.203
LoX case study									
Data set		C_{color}	γ_{color}	C_{mag}	γ_{mag}	C_{phase}	γ_{phase}	C_{global}	γ_{global}
AFW	UsedOil	$6.73e^3$	$1e^{-5}$	$4.07e^2$	8.73	1.0	$1e^{-5}$	1.0	0.354
	Oil	$3.16e^2$	2.73	$1.02e^3$	5.89	$2.04e^3$	4.95	$9.54e^3$	2.75
	Beer	$8.23e^3$	8.12	$8.75e^3$	0.143	$7.5e^3$	7.58	$7.5e^3$	0.0709
	7Wine	$1.94e^3$	1.59	$2.41e^3$	2.08	$2.58e^3$	$1e^{-5}$	$1.09e^3$	0.284
	WineGly	$2.99e^2$	3.56	$1.65e^3$	0.00327	$3.11e^3$	$1e^{-5}$	$8.64e^2$	0.137
	SoyVine	$9.3e^3$	0.582	$4.74e^2$	0.208	$1.17e^3$	8.24	$8.53e^3$	0.926
	Milk	1.0	$1e^{-5}$	1.0	6.01	$3.64e^3$	$1e^{-5}$	1.0	1.12
	Salt	$3.16e^2$	2.73	$1.02e^3$	5.89	$2.04e^3$	4.95	$9.54e^3$	2.75
	Powder	$5.18e^3$	3.88	$3.44e^3$	0.0946	$4.15e^3$	6.41	$1.45e^3$	0.481
DeCaDrive case study									
		C_{Kinect}	γ_{Kinect}	C_{DrBhv}	γ_{DrBhv}	$C_{DrState}$	$\gamma_{DrState}$	C_{global}	γ_{global}
DeCaDrive	AFW	$5.72e^3$	1.72	$1.65e^4$	6.48	$1.22e^4$	4.1	$1.26e^4$	3.55

Bibliography

- [1] Tsensors Summit. Tsensors Summit for Trillion Sensor Roadmap. URL <http://www.tsensorssummit.org/Resources/TSensors%20Roadmap%20v1.pdf>. Accessed: Dec. 2015.
- [2] Yole Développement. Technologies & Sensors for the Internet of Things: Businesses & Market Trends 2014-2024). URL http://www.i-micronews.com/images/Flyers/MEMS/Yole_Technologies_and_Sensors_for_the_Internet_of_Things_June_2014_Report_web.pdf. Accessed: Dec. 2015.
- [3] H.J. Escalante, M. Montes, and E. Sucar. Ensemble Particle Swarm Model Selection. In *Neural Networks (IJCNN), The 2010 International Joint Conference on*, pages 1–8, July 2010. doi: 10.1109/IJCNN.2010.5596915.
- [4] A. Bernstein, F. Provost, and S. Hill. Toward Intelligent Assistance for a Data Mining Process: An Ontology-Based Approach for Cost-Sensitive Classification. *IEEE Transactions on Knowledge and Data Engineering*, 17:503–518, 2005.
- [5] B. Denkena, D. Dahlmann, and J. Damm. Self-Tuning of Teachless Process Monitoring Systems with Multi-Criteria Monitoring Strategy in Series Production. *Procedia Technology*, 15(0): 614 – 621, 2014. ISSN 2212-0173. doi: <http://dx.doi.org/10.1016/j.protcy.2014.09.022>. URL <http://www.sciencedirect.com/science/article/pii/S2212017314001376>. 2nd International Conference on System-Integrated Intelligence: Challenges for Product and Production Engineering.
- [6] D. Bannach, P. Lukowicz, and O. Amft. Rapid Prototyping of Activity Recognition Applications. *Pervasive Computing, IEEE*, 7(2):22–31, April 2008. ISSN 1536-1268. doi: 10.1109/MPRV.2008.36.
- [7] H. Guenther, F. E. Simrany, M. Berchtold, and M. Beigl. A Tool Chain for a Lightweight, Robust and Uncertainty-Based Context Classification System (CCS). In *Architecture of Computing Systems (ARCS), 2010 23rd International Conference on*, pages 1–9, Feb 2010.
- [8] S. Santini, R. Adelman, M. Langheinrich, G. Schätti, and S. Fluck. Jsense – Prototyping Sensor-Based, Location-Aware Applications in Java. In Hee Yong Youn, Minkoo Kim, and Hiroyuki Morikawa, editors, *Ubiquitous Computing Systems – Third International Symposium, UCS 2006, Seoul, Korea, October 11-13, 2006, Proceedings*, volume 4239 of *Lecture Notes in Computer Science*, page 300–315, Berlin Heidelberg New York, oct 2006. Springer, Springer.
- [9] F. Shafait, M. Reif, C. Kofler, and T. Breuel. Pattern Recognition Engineering. In *RapidMiner Community Meeting and Conference*. Online, 9 2010.
- [10] J.-M. Gottfried and D. Kondermann. Charon Suite Software Framework. In *IPOL 2012 Meeting on Image Processing Libraries*, 2012. URL <http://hci.iwr.uni-heidelberg.de/Staff/jgottfried/papers/IPOL12-charonSuite/charon.pdf>.

- [11] J. Alcalá-Fdez, L. Sánchez, S. García, M.J. del Jesus, S. Ventura, J.M. Garrell, J. Otero, C. Romero, J. Bacardit, V.M. Rivas, J.C. Fernández, and F. Herrera. Keel: a Software Tool to Assess Evolutionary Algorithms for Data Mining Problems. *Soft Computing*, 13(3):307–318, 2009. ISSN 1432-7643. doi: 10.1007/s00500-008-0323-y. URL <http://dx.doi.org/10.1007/s00500-008-0323-y>.
- [12] S. Peters. *Automatischer Entwurf und Autokonfiguration von Bildverarbeitungssystemen für die industrielle Oberflächeninspektion*. PhD thesis, Lehrstuhl Integrierte Sensorsysteme TU Kaiserslautern, Forschungsberichte Integrierte Sensorsysteme Band 03, Herausgegeben von Prof. Dr.-Ing. A. König, Germany, 2011. ISBN 978-3-941438-63-7. URL <https://kluedo.ub.uni-kl.de/frontdoor/index/index/docId/2192>.
- [13] A. König, E. Michael, and W. Robert. QuickCog Self-Learning Recognition System - Exploiting Machine Learning Techniques for Transparent and Fast Industrial Recognition System Design. *Image Processing Europe*, Sept./Oct:10–19, 1999.
- [14] J. Demšar, T. Curk, A. Erjavec, Č. Gorup, T. Hočevar, M. Milutinovič, M. Možina, M. Polajnar, M. Toplak, A. Starič, M. Štajdohar, L. Umek, L. Žagar, J. Žbontar, M. Žitnik, and B. Zupan. Orange: Data Mining Toolbox in Python. *J. of Machine Learning Research*, 14:2349–2353, 2013. URL <http://jmlr.org/papers/v14/demsar13a.html>.
- [15] S. Robin, R. Eric, M. Norman, and L. Holger. Pattern Recognition Prototyping Tool. In *19th ITS World Congress, Vienna, Austria, 22/26*, pages 1–8, Oct 2012. URL https://www.baselabs.de/fileadmin/user_upload/BASELABS/Downloads/Papers/Rapid_Prototyping_of_ADAS_and_ITS_applications_on_the_example_of_a_vision-based_vehicle_tracking_system.pdf.
- [16] STEMMER IMAGING GmbH. Common Vision Blox (cvb), . URL <http://www.commonvisionblox.com/en/elements/fundamentals>. Accessed: Dec. 2015.
- [17] Impuls Imaging GmbH. Machine Vision Software für die Lehre, Sep 2014. URL <http://www.impuls-imaging.com/?p=1338>. Accessed: June 2015.
- [18] Freescale Semiconductor Inc. Freescale Sensor Fusion Library for Kinetics Data Sheet. Technical report, 2014. URL http://cache.freescale.com/files/sensors/doc/data_sheet/XSFLK_DS.pdf.
- [19] C. Müller-Schloer, H. Schmeck, and T. Ungerer. Organic Computing. *Informatik-Spektrum*, 35(2):71–73, 2012. ISSN 0170-6012. doi: 10.1007/s00287-012-0599-2. URL <http://dx.doi.org/10.1007/s00287-012-0599-2>.
- [20] E. A. Lee. Cyber Physical Systems: Design Challenges. Technical Report UCB/EECS-2008-8, EECS Department, University of California, Berkeley, Jan 2008. URL <http://www.eecs.berkeley.edu/Pubs/TechRpts/2008/EECS-2008-8.html>.
- [21] A. Kevin. That 'Internet of Things' Thing, June 2009. URL <http://www.rfidjournal.com/articles/view?4986>. Accessed: Dec. 2015.
- [22] J. Jürgen. Was hinter Begriffen wie Industrie 4.0 steckt. <http://www.computer-automation.de/steuerungsebene/steuern-regeln/artikel/93559/>. Accessed: Dec. 2015.
- [23] International Technology Roadmap for Semiconductors. ITRS 2013 Edition Summary, 2013. URL <http://www.itrs.net/ITRS%201999-2014%20Mtgs,%20Presentations%20&%20Links/2013ITRS/Summary2013.htm>. Accessed: Dec. 2015.

- [24] P. Cogeaz M. Graef B.Huizing W. Arden, M. Brillouet and R. Mahnkopf. More than Moore White Paper. International Technology Roadmap for Semiconductors (ITRS). Technical report. URL www2.imec.be/content/user/File/MtM%20WG%20report.pdf.
- [25] E. M. Tapia, S. Intille, and K. Larson. Activity Recognition in the Home Using Simple and Ubiquitous Sensors. In Alois Ferscha and Friedemann Mattern, editors, *Pervasive Computing*, volume 3001 of *Lecture Notes in Computer Science*, pages 158–175. Springer Berlin Heidelberg, 2004. ISBN 978-3-540-21835-7. doi: 10.1007/978-3-540-24646-6_10. URL http://dx.doi.org/10.1007/978-3-540-24646-6_10.
- [26] A. König and K. Thongpull. Lab-on-Spoon a 3-D Integrated Hand-Held Multi-Sensor System for Low-Cost Food Quality, Safety, and Processing Monitoring in Assisted-Living Systems. *Journal of Sensors and Sensor Systems*, 4(1):63–75, 2015. doi: 10.5194/jsss-4-63-2015. URL <http://www.j-sens-sens-syst.net/4/63/2015/>.
- [27] L. Li, B. Thomas, and A. König. Can Impedance Spectroscopy Serve in an Embedded Multi-Sensor System to Improve Driver Drowsiness Detection ? In *Int. Workshop on Impedance Spectroscopy*, pages 48–49, September 2013.
- [28] M. Kohlert. *Multi-Sensory Data Analysis and On-Line Evaluation for Advanced Process Control and Yield Optimization in Polymer Film Industry*. PhD thesis, Lehrstuhl Integrierte Sensorsysteme TU Kaiserslautern, Forschungsberichte Integrierte Sensorsysteme Band 06, Herausgegeben von Prof. Dr.-Ing. A. König, Germany, 2015. ISBN 978-3-95974-002-9. URL <https://kluedo.uni-kl.de/frontdoor/index/index/docId/4175>.
- [29] N. Helwig and A. Schütze. Intelligentes Condition Monitoring von Hydraulischen Anlagen. In *Messtechnisches Symposium 2014*, pages 121 – 128, 2014. doi: 10.5162/AHMT2014/P1.
- [30] K. Iswandy and A. König. Methodology, Algorithms, and Emerging Tool for Automated Design of Intelligent Integrated Multi-Sensor Systems. *Algorithms*, 2(4):1368–1409, 2009. ISSN 1999-4893. doi: 10.3390/a2041368. URL <http://www.mdpi.com/1999-4893/2/4/1368>.
- [31] H.B. Mitchell. *Multi-Sensor Data Fusion: An Introduction*. Springer, 2007. ISBN 9783540715597. URL <http://books.google.de/books?id=1kXwQhJmOMIC>.
- [32] G. L. Pappa and A. A. Freitas. *Automating the Design of Data Mining Algorithms*. Springer Berlin Heidelberg, 2010. ISBN 9783642025426.
- [33] K. Iswandy and A. König. Automated Design of Dependable Intelligent Sensory Systems with Self-X Properties. In A. König, A. Dengel, K. Hinkelmann, K. Kise, R. Howlett, and L. Jain, editors, *Knowledge-Based and Intelligent Information and Engineering Systems*, volume 6884 of *Lecture Notes in Computer Science*, pages 155–166. Springer Berlin Heidelberg, 2011. ISBN 978-3-642-23865-9. doi: 10.1007/978-3-642-23866-6_17. URL http://dx.doi.org/10.1007/978-3-642-23866-6_17.
- [34] E. Michael, R. Siegfried, and A. König. Industrial Application of Machine-in-the-Loop-Learning for a Medical Robot Vision System - Concept and Comprehensive Field Study. *Computers & Electrical Engineering*, 34(2):111 – 126, 2008. ISSN 0045-7906. doi: <http://dx.doi.org/10.1016/j.compeleceng.2007.10.006>. URL <http://www.sciencedirect.com/science/article/pii/S004579060700081X>. Advances on Computer-based Biological Signal Processing Techniques.
- [35] P.M. Szczówka, A. Szczurek, and B.W. Licznarski. On Reliability of Neural Network Sensitivity Analysis Applied for Sensor Array Optimization. *Sensors and Actuators B: Chemical*, 157(1): 298 – 303, 2011. ISSN 0925-4005. doi: <http://dx.doi.org/10.1016/j.snb.2011.03.066>. URL <http://www.sciencedirect.com/science/article/pii/S0925400511002620>.

- [36] M. J. Korsten, P. E. van der Vet, and P. P. L. Regtien. A System for the Automatic Selection of Sensors. In *Proc. IMEKO (International Measurement Confederation) XVI, Vienna, Austria*, pages 211–216, 2000. ISBN 3-901888-11-X.
- [37] J.W. Gardner, P. Boilot, and E.L. Hines. Enhancing Electronic Nose Performance by Sensor Selection Using a New Integer-Based Genetic Algorithm Approach. *Sensors and Actuators B: Chemical*, 106(1):114 – 121, 2005. ISSN 0925-4005. doi: <http://dx.doi.org/10.1016/j.snb.2004.05.043>. URL <http://www.sciencedirect.com/science/article/pii/S0925400504004071>. {ISOEN} 2003 - Selected Papers from the 10th International Symposium on Olfaction and Electronic Noses.
- [38] I. Guyon and A. Elisseeff. An Introduction to Variable and Feature Selection. *J. Mach. Learn. Res.*, 3:1157–1182, March 2003. ISSN 1532-4435. URL <http://dl.acm.org/citation.cfm?id=944919.944968>.
- [39] K. Iswandy and A. Koenig. Feature-Level Fusion by Multi-Objective Binary Particle Swarm Based Unbiased Feature Selection for Optimized Sensor System Design. In *Multisensor Fusion and Integration for Intelligent Systems, 2006 IEEE International Conference on*, pages 365–370, Sept 2006. doi: 10.1109/MFI.2006.265670.
- [40] P. Pudil, J. Novovičová, and J. Kittler. Floating Search Methods in Feature Selection. *Pattern Recognition Letters*, 15(11):1119 – 1125, 1994. ISSN 0167-8655. doi: [http://dx.doi.org/10.1016/0167-8655\(94\)90127-9](http://dx.doi.org/10.1016/0167-8655(94)90127-9). URL <http://www.sciencedirect.com/science/article/pii/0167865594901279>.
- [41] A. König. A Novel Supervised Dimensionality Reduction Technique by Feature Weighting for Improved Neural Network Classifier Learning and Generalization. In *6th Int. Conf. on Soft-Computing and Information/Intelligent Systems*, volume 4, pages 746 – 753, Oct 2000.
- [42] J. W. Lee and C. Giraud-Carrier. Automatic Selection of Classification Learning Algorithms for Data Mining Practitioners. *Intell. Data Anal.*, 17(4):665–678, July 2013. ISSN 1088-467X. doi: 10.3233/IDA-130599. URL <http://dx.doi.org/10.3233/IDA-130599>.
- [43] A. Bernstein, F. Provost, and S. Hill. Toward Intelligent Assistance for a Data Mining Process: An Ontology-Based Approach for Cost-Sensitive Classification. *IEEE Trans. on Knowl. and Data Eng.*, 17(4):503–518, April 2005. ISSN 1041-4347. doi: 10.1109/TKDE.2005.67. URL <http://dx.doi.org/10.1109/TKDE.2005.67>.
- [44] J. Kätevä, P. Laurinen, T. Rautio, J. Suutala, L. Tuovinen, and J. Röning. Se-155 dbsa: A Device-Based Software Architecture for Data Mining. In *SAC*, pages 2273–2280. ACM, 2010. ISBN 978-1-60558-639-7. URL <http://dblp.uni-trier.de/db/conf/sac/sac2010.html#KatevalRSTR10>.
- [45] M. Reif, F. Shafait, M. Goldstein, T. Breuel, and A. Dengel. Automatic Classifier Selection for Non-Experts. *Pattern Analysis and Applications*, 17(1):83–96, 2014. ISSN 1433-7541. doi: 10.1007/s10044-012-0280-z. URL <http://dx.doi.org/10.1007/s10044-012-0280-z>.
- [46] VDI/VDE. NAMUR/GMA-Roadmap Prozess-Sensoren 2015+, 2009. URL https://www.vdi.de/fileadmin/vdi_de/redakteur_dateien/gma_dateien/Prozess-Sensoren_2015+.pdf. Accessed: Dec. 2015.
- [47] F. Brissaud, A. Barros, C. Bérenguer, and D. Charpentier. Reliability Analysis for New Technology-Based Transmitters. *Reliability Engineering & System Safety*, 96(2):299 – 313, 2011. ISSN 0951-8320. doi: <http://dx.doi.org/10.1016/j.res.2010.09.010>. URL <http://www.sciencedirect.com/science/article/pii/S0951832010002139>.

- [48] G.Y Tian, Z.X Zhao, and R.W Baines. A Fieldbus-Based Intelligent Sensor. *Mechatronics*, 10(8): 835 – 849, 2000. ISSN 0957-4158. doi: [http://dx.doi.org/10.1016/S0957-4158\(99\)00078-1](http://dx.doi.org/10.1016/S0957-4158(99)00078-1). URL <http://www.sciencedirect.com/science/article/pii/S0957415899000781>.
- [49] R. Misra and C. Mandal. Self-Healing for Self-Organizing Cluster Sensor Networks. In *India Conference, 2006 Annual IEEE*, pages 1–6, Sept 2006. doi: 10.1109/INDCON.2006.302833.
- [50] N.M. Jokerst, Lin Luan, S. Palit, M. Royal, S. Dhar, M. Brooke, and T. Tyler. Progress in Chip-Scale Photonic Sensing. *Biomedical Circuits and Systems, IEEE Transactions on*, 3(4):202–211, Aug 2009. ISSN 1932-4545. doi: 10.1109/TBCAS.2009.2020693.
- [51] D. J. Nagel. Microsensor Clusters. *Microelectronics Journal*, 33(1–2):107 – 119, 2002. ISSN 0026-2692. doi: [http://dx.doi.org/10.1016/S0026-2692\(01\)00110-0](http://dx.doi.org/10.1016/S0026-2692(01)00110-0).
- [52] R.R. Tummala and V.K. Madiseti. System on Chip or System on Package? *Design Test of Computers, IEEE*, 16(2):48–56, Apr 1999. ISSN 0740-7475. doi: 10.1109/54.765203.
- [53] D. Choudhury. 3D Integration Technologies for Emerging Microsystems. In *Microwave Symposium Digest (MTT), 2010 IEEE MTT-S International*, pages 1–4, May 2010. doi: 10.1109/MWSYM.2010.5514747.
- [54] D.D. Kim, M.A. Brooke, N.M. Jokerst, J.J. Lillie, M.A. Thomas, and S.E. Ralph. Data Acquisition Sensitivity Determination of a Sensor-on-a-Chip Integrated Microsystem. In *Sensors, 2004. Proceedings of IEEE*, pages 1297–1300 vol.3, Oct 2004. doi: 10.1109/ICSENS.2004.1426419.
- [55] V. Shukla, N.B.Z. Ali, F.A. Hussin, and M. Zwolinski. On Testing of Media Based Digital Microfluidics Biochips. In *Quality Electronic Design (ASQED), 2013 5th Asia Symposium on*, pages 60–65, Aug 2013. doi: 10.1109/ASQED.2013.6643565.
- [56] C.L. Roozeboom, V.A. Hong, C.H. Ahn, E.J. Ng, Y. Yang, B.E. Hill, M.A. Hopcroft, and B.L. Pruitt. Multifunctional Integrated Sensor in a 2x2 mm Epitaxial Sealed Chip Operating in a Wireless Sensor Node. In *Micro Electro Mechanical Systems (MEMS), 2014 IEEE 27th International Conference on*, pages 773–776, Jan 2014. doi: 10.1109/MEMSYS.2014.6765755.
- [57] D.D. Kim, M.A. Thomas, J.J. Lillie, K.S. Dennis, B.M. Comeau, M.A. Brooke, N.M. Jokerst, S.E. Ralph, and C.L. Henderson. Integrated Mixed-Signal Optoelectronic System-on-a-Chip Sensor. In *Circuits and Systems, 2005. ISCAS 2005. IEEE International Symposium on*, pages 1738–1741 Vol. 2, May 2005. doi: 10.1109/ISCAS.2005.1464943.
- [58] B. Alandry, N. Dumas, L. Latorre, F. Mailly, and P. Nouet. A CMOS Multi-Sensor System for 3D Orientation Determination. In *Symposium on VLSI, 2008. ISVLSI '08. IEEE Computer Society Annual*, pages 57–62, April 2008. doi: 10.1109/ISVLSI.2008.80.
- [59] D. Groben, K. Thongpull, A. C. Kammara, and A. König. Neural Virtual Sensors for Adaptive Magnetic Localization of Autonomous Dataloggers. *Advances in Artificial Neural Systems*, 2014: 1–17, 2014. doi: 10.1155/2014/394038.
- [60] F.-L. Lian, J. Moyne, and D. Tilbury. Network Design Consideration for Distributed Control Systems. *Control Systems Technology, IEEE Transactions on*, 10(2):297–307, March 2002. ISSN 1063-6536. doi: 10.1109/87.987076.
- [61] S. Mekid. Further Structural Intelligence for Sensors Cluster Technology in Manufacturing. *Sensors*, 6(6):557–577, 2006. ISSN 1424-8220. doi: 10.3390/s6060557. URL <http://www.mdpi.com/1424-8220/6/6/557>.

- [62] José Rivera, Gilberto Herrera, Mario Chacón, Pedro Acosta, and Mariano Carrillo. Improved Progressive Polynomial Algorithm for Self-Adjustment and Optimal Response in Intelligent Sensors. *Sensors*, 8(11):7410–7427, 2008. ISSN 1424-8220. doi: 10.3390/s8117410. URL <http://www.mdpi.com/1424-8220/8/11/7410>.
- [63] S. Mekid, T. Schlegel, N. Aspragathos, and R. Teti. Foresight Formulation in Innovative Production, Automation and Control Systems. *Foresight*, 9(5):35–47, 2007. doi: 10.1108/14636680710821089.
- [64] A. Mkhida, J.-M. Thiriet, and J.-F. Aubry. Integration of Intelligent Sensors in Safety Instrumented Systems (SIS). *Process Safety and Environmental Protection*, 92(2):142 – 149, 2014. ISSN 0957-5820. doi: <http://dx.doi.org/10.1016/j.psep.2013.01.001>.
- [65] H. F. Durrant-Whyte. Sensor Models and Multisensor Integration. *Int. J. Rob. Res.*, 7(6):97–113, December 1988. ISSN 0278-3649. doi: 10.1177/027836498800700608. URL <http://dx.doi.org/10.1177/027836498800700608>.
- [66] A.M. Rao and D.L. Jones. A Denoising Approach to Multisensor Signal Estimation. *Signal Processing, IEEE Transactions on*, 48(5):1225–1234, May 2000. ISSN 1053-587X.
- [67] R.C. Luo, Chih Chia Chang, and Chun Chi Lai. Multisensor Fusion and Integration: Theories, Applications, and its Perspectives. *Sensors Journal, IEEE*, 11(12):3122–3138, Dec 2011. ISSN 1530-437X. doi: 10.1109/JSEN.2011.2166383.
- [68] S. Bhadra, D.S.Y. Tan, D.J. Thomson, M.S. Freund, and G.E. Bridges. A Wireless Passive Sensor for Temperature Compensated Remote pH Monitoring. *Sensors Journal, IEEE*, 13(6):2428–2436, June 2013. ISSN 1530-437X. doi: 10.1109/JSEN.2013.2255519.
- [69] A. Martinelli. Vision and IMU Data Fusion: Closed-Form Solutions for Attitude, Speed, Absolute Scale, and Bias Determination. *Robotics, IEEE Transactions on*, 28(1):44–60, Feb 2012. ISSN 1552-3098. doi: 10.1109/TRO.2011.2160468.
- [70] P.-J. Lapray, X. Wang, J.-B. Thomas, and P. Gouton. Multispectral Filter Arrays: Recent Advances and Practical Implementation. *Sensors*, 14(11):21626–21659, 2014. ISSN 1424-8220. doi: 10.3390/s141121626. URL <http://www.mdpi.com/1424-8220/14/11/21626>.
- [71] O.A. Postolache, J.M.D. Pereira, and P.M.B.S. Girao. Smart Sensors Network for air Quality Monitoring Applications. *Instrumentation and Measurement, IEEE Transactions on*, 58(9):3253–3262, Sept 2009. ISSN 0018-9456. doi: 10.1109/TIM.2009.2022372.
- [72] Y. Makino, S. Ogawa, and H. Shinoda. Flexible EMG Sensor Array for Haptic Interface. In *SICE Annual Conference, 2008*, pages 1468–1473, Aug 2008. doi: 10.1109/SICE.2008.4654890.
- [73] J. Tiete, F. Domínguez, B. da Silva, L. Segers, K. Steenhaut, and A. Touhafi. Soundcompass: A Distributed MEMS Microphone Array-Based Sensor for Sound Source Localization. *Sensors*, 14(2):1918–1949, 2014. ISSN 1424-8220. doi: 10.3390/s140201918. URL <http://www.mdpi.com/1424-8220/14/2/1918>.
- [74] Wen-Pin Shih, Li-Chi Tsao, Chian-Wen Lee, Ming-Yuan Cheng, Chienliu Chang, Yao-Joe Yang, and Kuang-Chao Fan. Flexible Temperature Sensor Array Based on a Graphite-Polydimethylsiloxane Composite. *Sensors*, 10(4):3597–3610, 2010. ISSN 1424-8220. doi: 10.3390/s100403597. URL <http://www.mdpi.com/1424-8220/10/4/3597>.
- [75] J. M. A. Morgado. Large Area Tactile Sensor for Smart Surfaces with Low Power Consumption. ”masterarbeit, Prof. Dr.-ing. A. König”, ISE, TU Kaiserslautern, Kaiserslautern, June 2011.

- [76] Fernando Vidal-Verdú, Óscar Oballe-Peinado, José A. Sánchez-Durán, Julián Castellanos-Ramos, and Rafael Navas-González. Three Realizations and Comparison of Hardware for Piezoresistive Tactile Sensors. *Sensors*, 11(3):3249–3266, 2011. ISSN 1424-8220. doi: 10.3390/s110303249. URL <http://www.mdpi.com/1424-8220/11/3/3249>.
- [77] A. Flores-Caballero, D. Copaci, Ma. D. Blanco, L. Moreno, J. Herrán, I. Fernández, E. Ochoteco, G. Cabañero, and H. Grande. Innovative Pressure Sensor Platform and its Integration with an End-User Application. *Sensors*, 14(6):10273–10291, 2014. ISSN 1424-8220. doi: 10.3390/s140610273. URL <http://www.mdpi.com/1424-8220/14/6/10273>.
- [78] S. Roundy, P. K. Wright, and J. Rabaey. A Study of Low Level Vibrations as a Power Source for Wireless Sensor Nodes. *Computer Communications*, 26(11):1131 – 1144, 2003. ISSN 0140-3664. doi: [http://dx.doi.org/10.1016/S0140-3664\(02\)00248-7](http://dx.doi.org/10.1016/S0140-3664(02)00248-7). URL <http://www.sciencedirect.com/science/article/pii/S0140366402002487>. Ubiquitous Computing.
- [79] Marie H. Masson, Stéphane Canu, Yves Grandvalet, and Anders Lynggaard-Jensen. Software Sensor Design Based on Empirical Data. *Ecological Modelling*, 120(2-3):131 – 139, 1999. ISSN 0304-3800. doi: [http://dx.doi.org/10.1016/S0304-3800\(99\)00097-6](http://dx.doi.org/10.1016/S0304-3800(99)00097-6).
- [80] K. Iswandy and A. König. Hybrid Virtual Sensor Based on RBFN or SVR Compared for an Embedded Application. In A. König, A. Dengel, K. Hinkelmann, K. Kise, R. Howlett, and L. Jain, editors, *Knowledge-Based and Intelligent Information and Engineering Systems*, volume 6882 of *Lecture Notes in Computer Science*, chapter chapter 34, pages 335–344. Springer, 2011. ISBN 978-3-642-23862-8.
- [81] W. Elmenreich. *Sensor Fusion in Time-Triggered Systems*. PhD thesis, Technische Universität Wien, Institut für Technische Informatik, Treitlstr. 3/3/182-1, 1040 Vienna, Austria, 2002.
- [82] G. van der Horn and J. Huijsing. *Integrated Smart Sensors: Design and Calibration*. The Springer International Series in Engineering and Computer Science. Springer, 1998. ISBN 9780792380047. URL <http://books.google.de/books?id=tC0ZTdZqiUC>.
- [83] R. Frank. *Understanding Smart Sensors, Second Edition*. Artech House, Inc., Norwood, MA, USA, 2nd edition, 2000. ISBN 0890063117.
- [84] JCGM. Evaluation of Measurement Data — Guide to the Expression of Uncertainty in Measurement, 2008. URL <http://www.iso.org/sites/JCGM/GUM/JCGM100/C045315e-html/C045315e.html?csnumber=50461>. Accessed: Dec. 2015.
- [85] S. K. Lakshmanan. *TOWARDS DYNAMICALLY RECONFIGURABLE MIXED-SIGNAL ELECTRONICS FOR EMBEDDED AND INTELLIGENT SENSOR SYSTEMS*. PhD thesis, Lehrstuhl Integrierte Sensorsysteme TU Kaiserslautern, Forschungsberichte Integrierte Sensorsysteme Band 01, Herausgegeben von Prof. Dr.-Ing. A. König, Germany, 2009. ISBN 978-3-941438-03-3. URL <https://kluedo.ub.uni-kl.de/frontdoor/index/index/docId/2052>.
- [86] P. M. M. Tawdross. *Bio-Inspired Circuit Sizing and Trimming Methods for Dynamically Reconfigurable Sensor Electronics in Industrial Embedded Systems*. PhD thesis, Lehrstuhl Integrierte Sensorsysteme TU Kaiserslautern, Forschungsberichte Integrierte Sensorsysteme Band 02, Herausgegeben von Prof. Dr.-Ing. A. König, Germany, 2011. ISBN 978-3-941438-62-0. URL <https://kluedo.ub.uni-kl.de/frontdoor/index/index/docId/1920>.
- [87] A.H. Taner and J.E. Brignell. Aspects of Intelligent Sensor Reconfiguration. *Sensors and Actuators A: Physical*, 47(1-3):525 – 529, 1995. ISSN 0924-4247. doi: [http://dx.doi.org/10.1016/0924-4247\(94\)00955-H](http://dx.doi.org/10.1016/0924-4247(94)00955-H). URL <http://www.sciencedirect.com/science/article/pii/S092442479400955H>.

- [88] M. M. B. Vellasco, R. S. Zebulum, and M. A. Pacheco. *Evolutionary Electronics: Automatic Design of Electronic Circuits and Systems by Genetic Algorithms*. CRC Press, Inc., Boca Raton, FL, USA, 1st edition, 2001. ISBN 0849308658.
- [89] D. Keymeulen, R. Zebulum, Y. Jin, and A. Stoica. Fault-Tolerant Evolvable Hardware using Field-Programmable Transistor Arrays. *Reliability, IEEE Transactions on*, 49(3):305–316, Sep 2000. ISSN 0018-9529. doi: 10.1109/24.914547.
- [90] J.A. Walker, M.A. Trefzer, S.J. Bale, and A.M. Tyrrell. Panda: A Reconfigurable Architecture that Adapts to Physical Substrate Variations. *Computers, IEEE Transactions on*, 62(8):1584–1596, Aug 2013. ISSN 0018-9340. doi: 10.1109/TC.2013.59.
- [91] E. Benkhelifa, A. Pipe, G. Dragffy, and M. Nibouche. Towards Evolving Fault Tolerant Biologically Inspired Hardware using Evolutionary Algorithms. In *Evolutionary Computation, 2007. CEC 2007. IEEE Congress on*, pages 1548–1554, Sept 2007. doi: 10.1109/CEC.2007.4424657.
- [92] P. Tawdross and A. Köing. Investigation of Particle Swarm Optimization for Dynamic Reconfiguration of Field-Programmable Analog Circuits. In *Hybrid Intelligent Systems, 2005. HIS '05. Fifth International Conference on*, pages 6 pp.-, Nov 2005. doi: 10.1109/ICHIS.2005.66.
- [93] S.A. Mills, J.P. Karins, and R.B. Dydyk. Pattern Recognition Prototyping Tool. In *Image Processing, 1997. Proceedings., International Conference on*, volume 3, pages 312–315 vol.3, Oct 1997. doi: 10.1109/ICIP.1997.632102.
- [94] E. Phaisangittisagul, H. T. Nagle, and V. Areekul. Intelligent Method for Sensor Subset Selection for Machine Olfaction. *Sensors and Actuators B: Chemical*, 145(1):507 – 515, 2010. ISSN 0925-4005. doi: <http://dx.doi.org/10.1016/j.snb.2009.12.063>. URL <http://www.sciencedirect.com/science/article/pii/S0925400509010181>.
- [95] A. A. Khan and M.A. Zohdy. A Genetic Algorithm for Selection of Noisy Sensor Data in Multi-sensor Data Fusion. In *American Control Conference, 1997. Proceedings of the 1997*, volume 4, pages 2256–2262, Jun 1997. doi: 10.1109/ACC.1997.608983.
- [96] S. Kun and R.A. Peura. Selection of Measurement Frequencies for Optimal Extraction of Tissue Impedance Model Parameters. *Medical & Biological Engineering & Computing*, 37(6):699–703, 1999. ISSN 0140-0118. doi: 10.1007/BF02513370. URL <http://dx.doi.org/10.1007/BF02513370>.
- [97] P.W. Pachowicz and A. Williams. Automated Selection of Fusion Parameters Through a Segmentation of Multi-Sensor ROC Curves. In *Information Fusion, 2005 8th International Conference on*, volume 1, pages 8 pp.-, July 2005. doi: 10.1109/ICIF.2005.1591889.
- [98] F. Kobayashi, D. Masumoto, and F. Kojima. Sensor Selection Based on Fuzzy Inference for Sensor Fusion. In *Fuzzy Systems, 2004. Proceedings. 2004 IEEE International Conference on*, volume 1, pages 305–310, July 2004. doi: 10.1109/FUZZY.2004.1375739.
- [99] M. Reza. Low Noise Signal Conditioning for Sensor-Based Circuits. Technical report, Analog Devices, Inc., 2010. URL <http://www.analog.com/media/en/technical-documentation/technical-articles/MS-2066.pdf>. Technical Article MS-2066.
- [100] R.K. Henderson, Li Ping, and J.I. Sewell. Analog Integrated Filter Compilation. *Analog Integrated Circuits and Signal Processing*, 3(3):217–228, 1993. ISSN 0925-1030. doi: 10.1007/BF01239362. URL <http://dx.doi.org/10.1007/BF01239362>.

- [101] J.D. Lohn and S.P. Colombano. A Circuit Representation Technique for Automated Circuit Design. *Evolutionary Computation, IEEE Transactions on*, 3(3):205–219, Sep 1999. ISSN 1089-778X. doi: 10.1109/4235.788491.
- [102] W. Bracke, P. Merken, R. Puers, and C. Van Hoof. Generic Architectures and Design Methods for Autonomous Sensors. *Sensors and Actuators A: Physical*, 135(2):881 – 888, 2007. ISSN 0924-4247. doi: <http://dx.doi.org/10.1016/j.sna.2006.07.028>. URL <http://www.sciencedirect.com/science/article/pii/S0924424706005292>.
- [103] E. Jones, T. Oliphant, P. Peterson, et al. SciPy: Open Source Scientific Tools for Python, 2001–. URL <http://www.scipy.org/>. Accessed: May 2015.
- [104] K. Tsujino, W. Kobayashi, T. Onoye, and Y. Nakamura. Automated Design of Digital Filters for 3-D Sound Localization in Embedded Applications. In *Acoustics, Speech and Signal Processing, 2006. ICASSP 2006 Proceedings. 2006 IEEE International Conference on*, volume 5, page V, May 2006. doi: 10.1109/ICASSP.2006.1661284.
- [105] R. Singh and S.K. Arya. Optimization of IIR Digital Filters Using Particle Swarm Optimization. In *Communication, Information Computing Technology (ICCICT), 2012 International Conference on*, pages 1–7, Oct 2012. doi: 10.1109/ICCICT.2012.6398172.
- [106] K. Fukunaga and Warren L.G. Koontz. Application of the Karhunen-loève Expansion to Feature Selection and Ordering. *Computers, IEEE Transactions on*, C-19(4):311–318, April 1970. ISSN 0018-9340. doi: 10.1109/T-C.1970.222918.
- [107] A. Köing. Dimensionality Reduction Techniques for Multivariate Data Classification, Interactive Visualization, and Analysis-Systematic Feature Selection vs.Extraction. In *Knowledge-Based Intelligent Engineering Systems and Allied Technologies, 2000. Proceedings. Fourth International Conference on*, volume 1, pages 44–55 vol.1, 2000. doi: 10.1109/KES.2000.885757.
- [108] P. Somol and P. Pudil. Oscillating Search Algorithms for Feature Selection. In *Pattern Recognition, 2000. Proceedings. 15th International Conference on*, volume 2, pages 406–409 vol.2, 2000. doi: 10.1109/ICPR.2000.906098.
- [109] M.E. ElAlami. A Filter Model for Feature Subset Selection Based on Genetic Algorithm. *Knowledge-Based Systems*, 22(5):356 – 362, 2009. ISSN 0950-7051. doi: <http://dx.doi.org/10.1016/j.knosys.2009.02.006>. URL <http://www.sciencedirect.com/science/article/pii/S0950705109000410>.
- [110] K. Iswandy and A. Koenig. Feature-Level Fusion by Multi-Objective Binary Particle Swarm Based Unbiased Feature Selection for Optimized Sensor System Design. In *Multisensor Fusion and Integration for Intelligent Systems, 2006 IEEE International Conference on*, pages 365–370, Sept 2006. doi: 10.1109/MFI.2006.265670.
- [111] K. Iswandy and A. König. Feature Selection with Acquisition Cost for Optimizing Sensor System design. *Advances in Radio Science*, 4:135–141, 2006.
- [112] S. Ali and K. A. Smith. On Learning Algorithm Selection for Classification. *Applied Soft Computing*, 6(2):119 – 138, 2006. ISSN 1568-4946. doi: <http://dx.doi.org/10.1016/j.asoc.2004.12.002>. URL <http://www.sciencedirect.com/science/article/pii/S1568494605000049>.
- [113] K. Faceli, A.P.L.F. de Carvalho, and SO. Rezende. Combining Intelligent Techniques for Sensor Fusion. *Applied Intelligence*, 20(3):199–213, 2004. ISSN 0924-669X. doi: 10.1023/B:APIN.0000021413.05467.20. URL <http://dx.doi.org/10.1023/B%3AAPIN.0000021413.05467.20>.

- [114] D. Ruta and B. Gabrys. Application of the Evolutionary Algorithms for Classifier Selection in Multiple Classifier Systems with Majority Voting. In Josef Kittler and Fabio Roli, editors, *Multiple Classifier Systems*, volume 2096 of *Lecture Notes in Computer Science*, pages 399–408. Springer Berlin Heidelberg, 2001. ISBN 978-3-540-42284-6. doi: 10.1007/3-540-48219-9_40. URL http://dx.doi.org/10.1007/3-540-48219-9_40.
- [115] R. Xu and L. He. Gacem: Genetic Algorithm Based Classifier Ensemble in a Multi-Sensor system. *Sensors*, 8(10):6203–6224, 2008. ISSN 1424-8220. doi: 10.3390/s8106203. URL <http://www.mdpi.com/1424-8220/8/10/6203>.
- [116] Vector Informatik GmbH. Developing Multisensor Applications Quickly and Conveniently, 2015. URL http://vector.com/portal/medien/cmc/factsheets/vADASdeveloper_FactSheet_EN.pdf. Accessed: Dec. 2015.
- [117] A. König and A. Gratz. Advanced Methods for the Analysis of Semiconductor Manufacturing Process Data. In NikhilR. Pal and Lakhmi Jain, editors, *Advanced Techniques in Knowledge Discovery and Data Mining*, Advanced Information and Knowledge Processing, pages 27–74. Springer London, 2005. ISBN 978-1-85233-867-1. doi: 10.1007/1-84628-183-0_2. URL http://dx.doi.org/10.1007/1-84628-183-0_2.
- [118] Sensor-trends 2014: Trends in Zukunfsorientierten Sensortechnologien. Technical report, AMA Fachverband fuer Sensorik e.V Redaktionsteam., 2013.
- [119] M. A. Johar. *Exploration and Design of DC MEMS Switches for Integrated Self-x Sensory Systems*. PhD thesis, Lehrstuhl Integrierte Sensorsysteme TU Kaiserslautern, Forschungsberichte Integrierte Sensorsysteme Band 05, Herausgegeben von Prof. Dr.-Ing. A. König, Germany, 2015. ISBN 978-3-943995-87-9. URL <https://kluedo.ub.uni-kl.de/frontdoor/index/index/docId/4090>.
- [120] R. Freier. *Ein universelles und dynamisch rekonfigurierbares Interface für eingebettete und intelligente Multi-Sensor-Systeme mit Self-x Eigenschaften*. PhD thesis, Lehrstuhl Integrierte Sensorsysteme TU Kaiserslautern, Forschungsberichte Integrierte Sensorsysteme Band 04, Herausgegeben von Prof. Dr.-Ing. A. König, Germany, 2015. ISBN 978-3-943995-81-7. URL <https://kluedo.ub.uni-kl.de/frontdoor/index/index/docId/3966>.
- [121] Micropelt GmbH. Micropelt Thermogenerators, . URL <http://micropelt.com/thermogenerator.php>. Accessed: Dec. 2015.
- [122] Sensitec GmbH. AFF755B Datasheet, . URL <http://www.sensitec.com/english/products/magnetic-field/aff755.html>. Accessed: Dec. 2015.
- [123] A. König, A. Kammara, D. Groben, and K. Thongpull. Dynamically Reconfigurable Integrated Sensor Electronics for Magnetic Localisation of Distributed Autonomous Sensor Nodes. In *Proceedings SENSOR 2013*, pages 334 – 339, 2013. ISBN 978-3-9813484-3-9. doi: 10.5162/sensor2013/C1.1.
- [124] A. V. Oppenheim, R. W. Schaffer, and John R. Buck. *Discrete-Time Signal Processing (2Nd Ed.)*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1999. ISBN 0-13-754920-2.
- [125] K. Fukunaga. *Introduction to Statistical Pattern Recognition*. Elsevier Science, 1990. ISBN 9780122698514.
- [126] J. Kittler. *Feature Selection and Extraction*. Academic Press, Inc., Orlando, San Diego, New York, Austin, London, Montreal, Sydney, Tokyo, Toronto, 1986.

- [127] S.S. Haykin. *Neural Networks and Learning Machines*. Number v. 10 in Neural networks and learning machines. Prentice Hall, 2009. ISBN 9780131471399.
- [128] A. Buja and D. F. Swayne. Visualization Methodology for Multidimensional Scaling. *Journal of Classification*, 19(1):7–43, 2002. ISSN 0176-4268. doi: 10.1007/s00357-001-0031-0. URL <http://dx.doi.org/10.1007/s00357-001-0031-0>.
- [129] J.W. Sammon. A Nonlinear Mapping for Data Structure Analysis. *Computers, IEEE Transactions on*, C-18(5):401–409, May 1969. ISSN 0018-9340. doi: 10.1109/T-C.1969.222678.
- [130] J. De Leeuw and I. Stoop. Upper Bounds for Kruskal’s Stress. *Psychometrika*, 49(3):391–402, 1984. ISSN 0033-3123. doi: 10.1007/BF02306028. URL <http://dx.doi.org/10.1007/BF02306028>.
- [131] K. López-de Ipiña, H. Egiraun, J. Sole-Casals, M. Ecay, A. Ezeiza, N. Barroso, P. Martinez-Lage, and U. Martinez-de Lizardui. Feature Extraction Approach Based on Fractal Dimension for Spontaneous Speech Modelling Oriented to Alzheimer Disease Diagnosis. In Thomas Drugman and Thierry Dutoit, editors, *Advances in Nonlinear Speech Processing*, volume 7911 of *Lecture Notes in Computer Science*, pages 144–151. Springer Berlin Heidelberg, 2013. ISBN 978-3-642-38846-0. doi: 10.1007/978-3-642-38847-7_19. URL http://dx.doi.org/10.1007/978-3-642-38847-7_19.
- [132] W. L.G. Koontz, P. M. Narendra, and K. Fukunaga. A Branch and Bound Clustering Algorithm. *Computers, IEEE Transactions on*, C-24(9):908–915, Sept 1975. ISSN 0018-9340. doi: 10.1109/T-C.1975.224336.
- [133] J.P.M. de Sa. *Pattern Recognition: Concepts, Methods, and Applications*. Springer, 2001. ISBN 9783540422976.
- [134] P. Somol and P. Pudil. Oscillating Search Algorithms for Feature Selection. In *Pattern Recognition, 2000. Proceedings. 15th International Conference on*, volume 2, pages 406–409 vol.2, 2000. doi: 10.1109/ICPR.2000.906098.
- [135] J. C.W. Debusse and V.J. Rayward-Smith. Feature Subset Selection within a Simulated Annealing Data Mining algorithm. *Journal of Intelligent Information Systems*, 9(1):57–81, 1997. ISSN 0925-9902. doi: 10.1023/A:1008641220268. URL <http://dx.doi.org/10.1023/A%3A1008641220268>.
- [136] H. Vafaie and I. F. Imam. I.: Feature Selection Methods: Genetic Algorithms vs Greedy-Like Search. In *In: Proceedings of the International Conference on Fuzzy and Intelligent Control Systems*, 1994. URL <https://cs.gmu.edu/~eclab/papers/fuzzy.pdf>.
- [137] D. Wettschereck, D. Aha, and T. Mohri. A Review and Empirical Evaluation of Feature Weighting Methods for a Class of Lazy Learning Algorithms. *Artificial Intelligence Review*, 11(1-5):273–314, 1997. ISSN 0269-2821. doi: 10.1023/A:1006593614256. URL <http://dx.doi.org/10.1023/A%3A1006593614256>.
- [138] G. Guo, H. Wang, D. Bell, Y. Bi, and K. Greer. kNN Model-Based Approach in Classification. In Robert Meersman, Zahir Tari, and Douglas C. Schmidt, editors, *On The Move to Meaningful Internet Systems 2003: CoopIS, DOA, and ODBASE*, volume 2888 of *Lecture Notes in Computer Science*, pages 986–996. Springer Berlin Heidelberg, 2003. ISBN 978-3-540-20498-5. doi: 10.1007/978-3-540-39964-3_62. URL http://dx.doi.org/10.1007/978-3-540-39964-3_62.
- [139] G. Gates. The Reduced Nearest Neighbor Rule. *Information Theory, IEEE Transactions on*, 18(3):431–433, May 1972. ISSN 0018-9448. doi: 10.1109/TIT.1972.1054809.
- [140] P. Hart. The Condensed Nearest Neighbor Rule. *Information Theory, IEEE Transactions on*, 14(3):515–516, May 1968. ISSN 0018-9448. doi: 10.1109/TIT.1968.1054155.

- [141] C. Cortes and V. Vapnik. Support-Vector Networks. *Machine Learning*, 20(3):273–297, 1995. ISSN 0885-6125. doi: 10.1007/BF00994018. URL <http://dx.doi.org/10.1007/BF00994018>.
- [142] S.-W. Lin, Z.-J. Lee, S.-C. Chen, and T.-Y. Tseng. Parameter Determination of Support Vector Machine and Feature Selection Using Simulated Annealing Approach. *Applied Soft Computing*, 8(4):1505 – 1512, 2008. ISSN 1568-4946. doi: <http://dx.doi.org/10.1016/j.asoc.2007.10.012>. URL <http://www.sciencedirect.com/science/article/pii/S156849460700141X>. Soft Computing for Dynamic Data Mining.
- [143] K. Muller, S. Mika, G. Ratsch, K. Tsuda, and Bernhard Scholkopf. An Introduction to Kernel-Based Learning Algorithms. *Neural Networks, IEEE Transactions on*, 12(2):181–201, Mar 2001. ISSN 1045-9227. doi: 10.1109/72.914517.
- [144] D. M.J. Tax. One-Class Classification. PhD Thesis, Delft University of Technology, Delft, June 2001.
- [145] D. M.J. Tax and R. P.W. Duin. Support Vector Data Description. *Machine Learning*, 54(1):45–66, 2004. ISSN 0885-6125. doi: 10.1023/B:MACH.0000008084.60811.49. URL <http://dx.doi.org/10.1023/B%3AMACH.0000008084.60811.49>.
- [146] B. Schölkopf, J. C. Platt, J. C. Shawe-Taylor, A. J. Smola, and R. C. Williamson. Estimating the Support of a High-Dimensional Distribution. *Neural Comput.*, 13(7):1443–1471, July 2001. ISSN 0899-7667. doi: 10.1162/089976601750264965. URL <http://dx.doi.org/10.1162/089976601750264965>.
- [147] B. Schölkopf, A. J. Smola, R. C. Williamson, and P. L. Bartlett. New Support Vector Algorithms. *Neural Comput.*, 12(5):1207–1245, May 2000. ISSN 0899-7667. doi: 10.1162/089976600300015565. URL <http://dx.doi.org/10.1162/089976600300015565>.
- [148] S. S. Khan and M. G. Madden. One-Class Classification: Taxonomy of Study and Review of Techniques. *CoRR*, abs/1312.0049, 2013. URL <http://arxiv.org/abs/1312.0049>.
- [149] A. König, P. Windirsch, and M. Glesner. Massively Parallel VLSI-Implementation of a Dedicated Neural Network for Anomaly Detection in Automated Visual Quality Control. In *Microelectronics for Neural Networks and Fuzzy Systems, 1994., Proceedings of the Fourth International Conference on*, pages 354–363, Sep 1994. doi: 10.1109/ICMNN.1994.593731.
- [150] D.L. Reilly, L.N. Cooper, and C. Elbaum. A Neural Model for Category Learning. *Biological Cybernetics*, 45(1):35–41, 1982. ISSN 0340-1200. doi: 10.1007/BF00387211. URL <http://dx.doi.org/10.1007/BF00387211>.
- [151] B. Waske and J.A. Benediktsson. Fusion of Support Vector Machines for Classification of Multisensor Data. *Geoscience and Remote Sensing, IEEE Transactions on*, 45(12):3858–3866, Dec 2007. ISSN 0196-2892. doi: 10.1109/TGRS.2007.898446.
- [152] J. C. Platt. Probabilistic Outputs for Support Vector Machines and Comparisons to Regularized Likelihood Methods. In *ADVANCES IN LARGE MARGIN CLASSIFIERS*, pages 61–74. MIT Press, 1999.
- [153] A. Al-Ani and M. Deriche. Feature Selection Using a Mutual Information Based Measure. In *Pattern Recognition, 2002. Proceedings. 16th International Conference on*, volume 4, pages 82–85 vol.4, 2002. doi: 10.1109/ICPR.2002.1047405.

- [154] A. König, M. Eberhardt, and R. Wenzel. A Transparent and Flexible Development Environment for Rapid Design of Cognitive Systems. In *Euromicro Con., 1998. Proceedings. 24th*, volume 2, pages 655–662 vol.2, Aug 1998.
- [155] R. Alejo, J.M. Sotoca, R.M. Valdovinos, and P. Toribio. Edited Nearest Neighbor Rule for Improving Neural Networks Classifications. In Liqing Zhang, Bao-Liang Lu, and James Kwok, editors, *Advances in Neural Networks - ISNN 2010*, volume 6063 of *Lecture Notes in Computer Science*, pages 303–310. Springer Berlin Heidelberg, 2010. ISBN 978-3-642-13277-3. doi: 10.1007/978-3-642-13278-0_39. URL http://dx.doi.org/10.1007/978-3-642-13278-0_39.
- [156] K. Iswandy and A. Koenig. Towards Effective Unbiased Automated Feature Selection. In *Hybrid Intelligent Systems, 2006. HIS '06. Sixth International Conference on*, pages 29–29, Dec 2006. doi: 10.1109/HIS.2006.264912.
- [157] T. Lange, M. L. Braun, V. Roth, and J. M. Buhmann. Stability-Based Model Selection. *Advances in Neural Information Processing Systems*, 15:633–642, 2003.
- [158] R. L. Haupt and S. E. Haupt. *Practical Genetic Algorithms*. John Wiley & Sons, Inc., New York, NY, USA, 1998. ISBN 047-1188735.
- [159] W. Chu, X. Gao, and S. Sorooshian. Handling Boundary Constraints for Particle Swarm Optimization in High-Dimensional Search Space. *Information Sciences*, 181(20):4569 – 4581, 2011. ISSN 0020-0255. doi: <http://dx.doi.org/10.1016/j.ins.2010.11.030>. URL <http://www.sciencedirect.com/science/article/pii/S0020025510005839>. Special Issue on Interpretable Fuzzy Systems.
- [160] Y. Shi and R. C. Eberhart. Parameter Selection in Particle Swarm Optimization. In V.W. Porto, N. Saravanan, D. Waagen, and A.E. Eiben, editors, *Evolutionary Programming VII*, volume 1447 of *LNCS*, pages 591–600. Springer Berlin Heidelberg, 1998. ISBN 978-3-540-64891-8. doi: 10.1007/BFb0040810. URL <http://dx.doi.org/10.1007/BFb0040810>.
- [161] J. Kennedy and R.C. Eberhart. A Discrete Binary Version of the Particle Swarm Algorithm. In *Systems, Man, and Cybernetics, 1997. Computational Cybernetics and Simulation., 1997 IEEE International Conference on*, volume 5, pages 4104–4108 vol.5, Oct 1997. doi: 10.1109/ICSMC.1997.637339.
- [162] I.Y. Kim and O.L. de Weck. Adaptive Weighted Sum Method for Multiobjective Optimization: a New Method for Pareto Front Generation. *Structural and Multidisciplinary Optimization*, 31(2): 105–116, 2006. ISSN 1615-147X. doi: 10.1007/s00158-005-0557-6. URL <http://dx.doi.org/10.1007/s00158-005-0557-6>.
- [163] P. Hajela and C.-Y. Lin. Genetic Search Strategies in Multicriterion Optimal Design. *Structural optimization*, 4(2):99–107, 1992. ISSN 0934-4373. doi: 10.1007/BF01759923. URL <http://dx.doi.org/10.1007/BF01759923>.
- [164] E. Zitzler and L. Thiele. Multiobjective Evolutionary Algorithms: a Comparative Case Study and the Strength Pareto Approach. *IEEE Transactions on Evolutionary Computation*, pages 257–271, 1999.
- [165] C. M. Fonseca and P. J. Fleming. Genetic Algorithms for Multiobjective Optimization: Formulation, Discussion and Generalization, In *Proceedings of the 5th International Conference on Genetic Algorithms*, pages 416–423, 1993. ISBN:1-55860-299-2.

- [166] X. Li. A Non-Dominated Sorting Particle Swarm Optimizer for Multiobjective Optimization. In *Genetic and Evolutionary Computation — GECCO 2003*, volume 2723 of *Lecture Notes in Computer Science*, pages 37–48. Springer Berlin Heidelberg, 2003. ISBN 978-3-540-40602-0. doi: 10.1007/3-540-45105-6_4. URL http://dx.doi.org/10.1007/3-540-45105-6_4.
- [167] X. Cui, C.T. Hardin, R.K. Ragade, T.E. Potok, and A.S. Elmaghraby. Tracking Non-Stationary Optimal Solution by Particle Swarm Optimizer. In *Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing, 2005 and First ACIS International Workshop on Self-Assembling Wireless Networks. SNPD/SAWN 2005. Sixth International Conference on*, pages 133–138, May 2005. doi: 10.1109/SNPD-SAWN.2005.77.
- [168] H. G. Cobb. An Investigation Into the Use of Hypermutation as an Adaptive Operator in Genetic Algorithms Having Continuous. Technical report, 1990. Office of Naval Research, Arlington, VA URL <http://www.dtic.mil/dtic/tr/fulltext/u2/a229159.pdf>.
- [169] T. M. Blackwell and Peter J. Bentley. Dynamic Search with Charged Swarms. In *Proceedings of the Genetic and Evolutionary Computation Conference, GECCO '02*, pages 19–26, San Francisco, CA, USA, 2002. Morgan Kaufmann Publishers Inc. ISBN 1-55860-878-8. URL <http://dl.acm.org/citation.cfm?id=646205.682961>.
- [170] L. Liu, D. Wang, and S. Yang. Compound Particle Swarm Optimization in Dynamic Environments. In Mario Giacobini, et.al, editors, *Applications of Evolutionary Computing*, volume 4974 of *Lecture Notes in Computer Science*, pages 616–625. Springer Berlin Heidelberg, 2008. ISBN 978-3-540-78760-0. doi: 10.1007/978-3-540-78761-7_67. URL http://dx.doi.org/10.1007/978-3-540-78761-7_67.
- [171] J. Branke. Memory Enhanced Evolutionary Algorithms for Changing Optimization problems. In *Evolutionary Computation, 1999. CEC 99. Proceedings of the 1999 Congress on*, volume 3, pages –1882, 1999. doi: 10.1109/CEC.1999.785502.
- [172] C.N. Bendtsen and T. Krink. Dynamic Memory Model for Non-Stationary Optimization. In *Evolutionary Computation, 2002. CEC '02. Proceedings of the 2002 Congress on*, volume 1, pages 145–150, May 2002. doi: 10.1109/CEC.2002.1006224.
- [173] M. Wineberg and J. Chen. The Shifting Balance Genetic Algorithm as More than Just Another island model ga. In Kalyanmoy Deb, editor, *Genetic and Evolutionary Computation — GECCO 2004*, volume 3103 of *Lecture Notes in Computer Science*, pages 318–329. Springer Berlin Heidelberg, 2004. ISBN 978-3-540-22343-6. doi: 10.1007/978-3-540-24855-2_28. URL http://dx.doi.org/10.1007/978-3-540-24855-2_28.
- [174] M. Lichman. UCI Machine Learning Repository, 2013. URL <http://archive.ics.uci.edu/ml>.
- [175] C.-C. Chang and C.-J. Lin. Libsvm: A Library for Support Vector Machines. *ACM Trans. Intell. Syst. Technol.*, 2(3):27:1–27, May 2011.
- [176] Charles E. Metz. Basic Principles of {ROC} Analysis. *Seminars in Nuclear Medicine*, 8(4): 283 – 298, 1978. ISSN 0001-2998. doi: [http://dx.doi.org/10.1016/S0001-2998\(78\)80014-2](http://dx.doi.org/10.1016/S0001-2998(78)80014-2). URL <http://www.sciencedirect.com/science/article/pii/S0001299878800142>.
- [177] B. Dawes and D. Abrahams. Boost c++ libraries. URL www.boost.org. Accessed: Sep 2015.
- [178] A. König. Lab-on-Spoon Multi-Sensorial 3D-Integrated Measurement System for Smart-Kitchen and AAL Applications, . URL "www.eit.uni-kl.de/koenig/projects/LabonSpoon.html". Accessed: Oct. 2015.

- [179] A. König. The E-Taster Assistance System with Lab-on-Spoon and Lab-on-Fork as 'Electronic Tongues', URL www.eit.uni-kl.de/koenig/projects/E-Taster.html. Accessed: Oct. 2015.
- [180] MAZeT GmbH. Multiple Color Sensors mmcs6cs, 2015. URL <http://www.mazet.de/en/products/jencolor/multiple-color-sensors>. Accessed: Dec. 2015.
- [181] MAZeT GmbH. Sensor Signal Amplifiers, 2015. URL <http://www.mazet.de/en/products/jencolor/multi-channel-transimpedance-amplifier>. Accessed: Dec. 2015.
- [182] Analog Devices. Analog Devices, AD5933 Datasheet, 2011. URL http://www.analog.com/static/imported-files/data_sheets/AD5933.pdf. Accessed: Dec. 2015.
- [183] K. Thongpull and A. König. An Emerging Framework for Automated Design of Multi-Sensor Intelligent Measurement Systems Applied to Lab-on-Spoon in Food Analysis. In Thomas Fröhlich and Eberhard Manske, editors, *XXIX. Messtechnisches Symposium 2015*, pages 149–156, Berlin, 2015. De Gruyter Oldenbourg. doi: 10.1515/9783110408539-019.
- [184] K. Thongpull and A. König. Application-Specific Partially Automated Design of Multi-Sensor Intelligent Lab-on-Spoon System. In *Messtechnisches Symposium 2014*, pages 111 – 120, 2014. doi: 10.5162/AHMT2014/4.3.
- [185] L. Li, K. Werber, C. Carlos F., D. D. Khac, G. Ander, and A. König. Multi-Sensor Soft-Computing System for Driver Drowsiness Detection. In *World Conference on Soft-Computing*, December 2012.
- [186] K. Werber. Untersuchung von Fahrerassistenzsystemen zur Fahrer- zustands- und Absichtserkennung mit Multisensorik, 2012. Diplomarbeit, ISE, TU-Kaiserslautern.
- [187] D. Abrahams and R. W. Grosse-Kunstleve. Building Hybrid Systems with Boost.Python. *C/C++ Users Journal*, 21:29–36, 2003.
- [188] C. V. Nelson and B. C. Jacobs. Magnetic Sensor System for Fast-Response, High Resolution, High Accuracy Three-Dimensional Position Measurements, September 1998. US Patent Applicant: The John Hopkins University, USA.
- [189] J.M. Cioffi, W.L. Abbott, H.K. Thapar, C.M. Melas, and Kevin D. Fisher. Adaptive Equalization in Magnetic-Disk Storage Channels. *Communications Magazine, IEEE*, 28(2):14–29, 1990. ISSN 0163-6804.
- [190] BMBF. Sensorbasierte Elektroniksysteme für Anwendungen für Industrie 4.0 (SElekt I4.0), 2014. URL <http://www.bmbf.de/foerderungen/25379.php>. Accessed: Dec. 2015.

Curriculum Vitae

M. Eng. Kittikhun Thongpull

Permanent Address Prince of Songkla University, Hatyai, Thailand

Education Prince of Songkla University, Hatyai, Thailand
Electrical engineering
2010 M.Eng.

Prince of Songkla University, Hatyai, Thailand
Electrical engineering
2008 B.Eng.

Experience Technische Universität Kaiserslautern, ISE, EIT,
Research assistant (Wissenschaftlicher Mitarbeiter, teilzeit)
Oct. 2014 - present

Technische Universität Kaiserslautern, ISE, EIT,
Doctoral student
(Scholarship granted by Prince of Songkla University)
Nov. 2011 - Oct. 2014

Prince of Songkla University Hatyai,
Dept. of Electrical Engineering
Lecturer
Sep. 2010 - Nov. 2011
