# Design Automation of Computational Aspects of Embedded Heterogeneous Sensor Systems

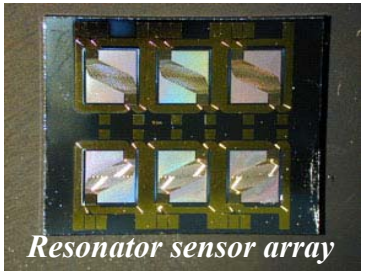
*Kuncup Iswandy*
**and**
*Andreas König*

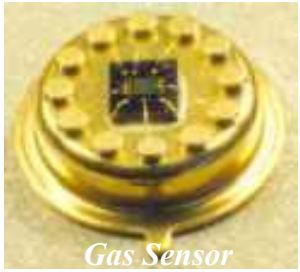
# Overview

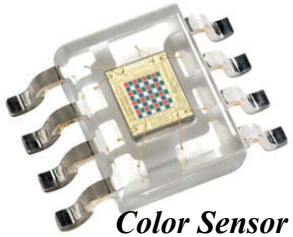
1. *Introduction*
   - *Motivation*
   - *Systematic Design of Intelligent Sensor System*
   - *The Objectives*
2. *Particle Swarm Optimization*
3. *Optimized Feature Computation*
   - *Multilevel Thresholding*
   - *Gaussian Windowing*
   - *Simulation Results*
4. *Optimal Feature Space under Constraints (Cost/Power)*
5. *Conclusions*

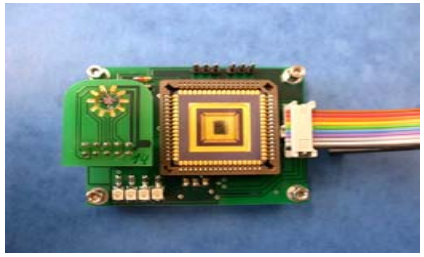
# Introduction
## Motivation

- Various characteristics / electrical interfaces of sensors


*Resonator sensor array*


*Gas Sensor*


***Color Sensor***
***Texas Advanced Optoelectronic Solutions (TAOS)***



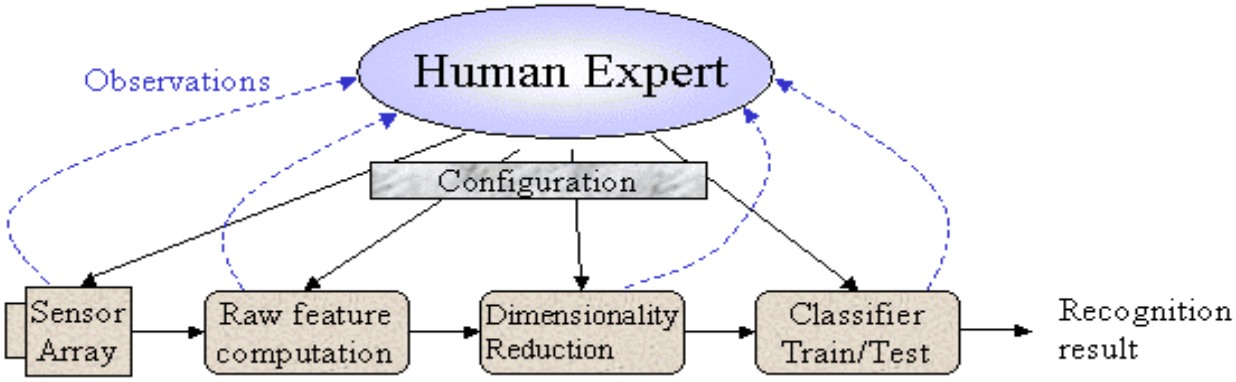
Multi gas sensor system with integrated ASIC for temperature control and signal pre-processing. (*Fraunhofer IPM*)

- Exploitation of sensors in many applications requires sophisticated methods, from:

  conventional signal processing to computational intelligence

# Introduction
## Motivation

- The design of intelligent sensor systems is still predominantly conducted manually by experienced designer, viz.:
  - sensor selection and scene optimization
  - choice of signal processing steps, etc.

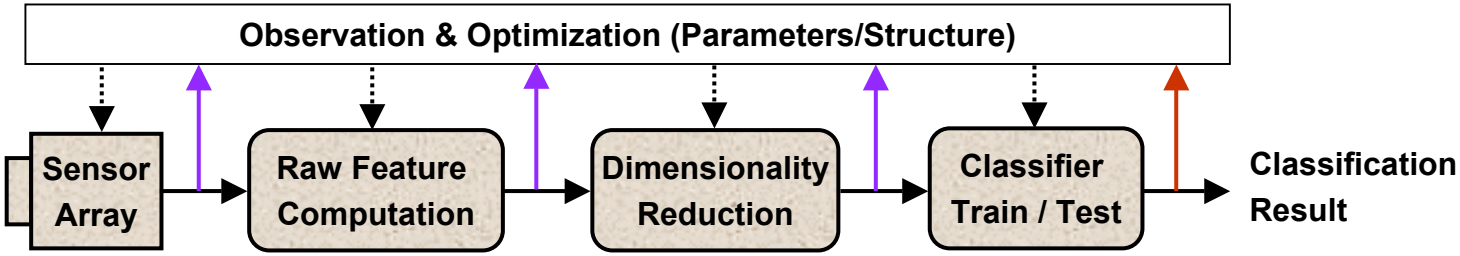



- Consequently:
  - tedious
  - time and labour consuming task
  - suboptimal outcome

# Introduction

## Systematic Design of Intelligent Sensor System

- General concept of automated design of multi sensor system

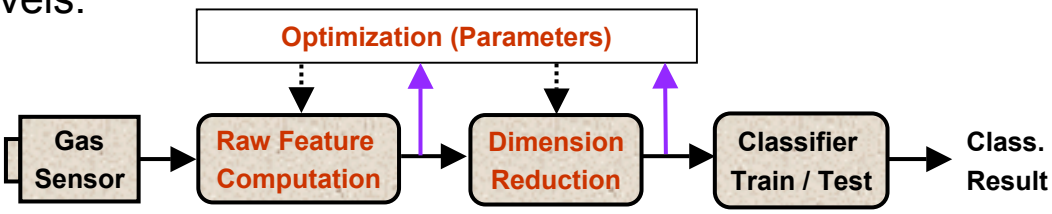



- Another consideration : data fusion
  - Sensor level (observation/measurement)
  - Feature level (features from time/frequency models,etc)
  - Decision level (classifier)

# Introduction

## The Objectives

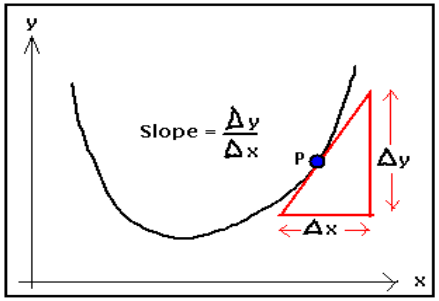
- Global aims of the research work:
  - ✓ Generate a computational platform for design automation according to the concept or architecture of intelligent multi-sensor systems
  - ✓ Collect and develop design methods and tools that automate design procedures on all levels.




- Contents & objectives of this talk:
  - ✓ contribute to the design automation activities for intelligent (multi-) sensor systems
  - ✓ focus on optimization of feature computation
  - ✓ optimizing sensor system design with regarding to optimum cost (e.g., time computation, power, price, etc).

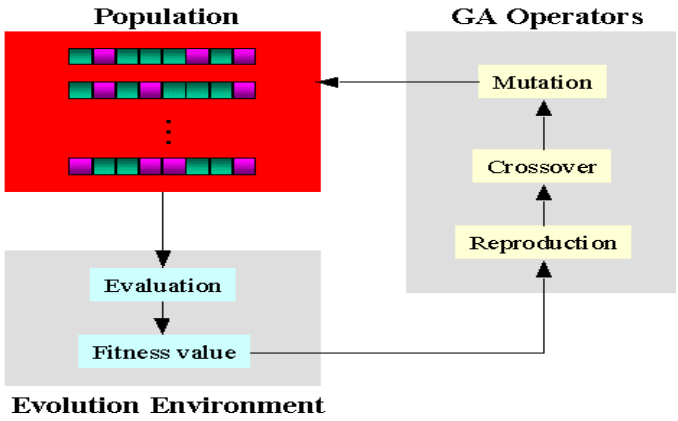
# Optimization

- Some optimization methods:




Gradient descent

- easily trapped to bad local optimum




Genetic Algorithm Evolution Flow

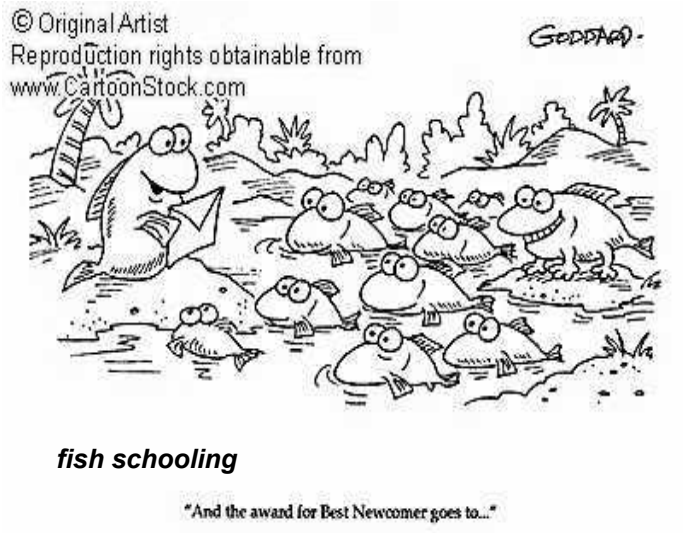
Simulated annealing

- excessive computation time required

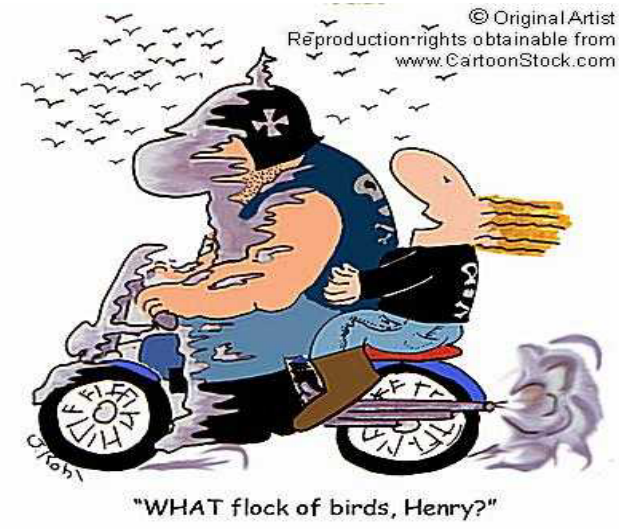
- The fashionable technique : Particle Swarm Optimization

# Optimization
## Particle Swarm Optimization (PSO)

- One of the evolutionary computation techniques
- Population-based search algorithm of swarm theory




***fish schooling***




- A population of random solutions is called as *particles*

# Optimization

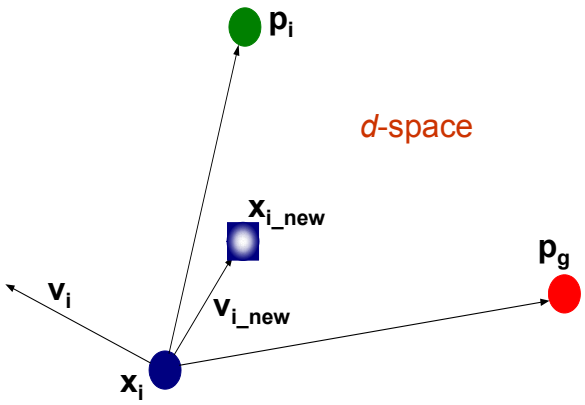
## Particle Swarm Optimization for a real-valued case

- PSO algorithm [Kennedy and Eberhart ´95]:

  - $v_{id}(t+1) = wv_{id}(t) + C_1 rand()\left(p_{id} - x_{id}(t)\right) + C_2 Rand()\left(p_{gd} - x_{id}(t)\right)$

  - $x_{id}(t+1) = x_{id}(t) + v_{id}(t+1)$

$v_{id}$ is restricted to $v_{\max}$
$w$ : inertia weight; start from 1 to 0.7 (linear change)
$C_1 = C_2 = 2$ : positive constants
Rand() and rand() : random functions, [0,1]
$p_i$ : best previous position of the *i*-th particle
$p_g$ : best particle among all particles



# Optimization

## Particle Swarm Optimization for a binary case

- For binary, particles are assigned binary-valued, e.g., 0 or 1
- Velocity is restricted to an interval value of [-4,4]

- BPSO algorithm [Kennedy and Eberhart ´95]:

  - $$v_{id}(t+1) = wv_{id}(t) + C_1 rand()\left(p_{id} - x_{id}(t)\right) + C_2 Rand()\left(p_{gd} - x_{id}(t)\right)$$

  - $$x_{id} = \begin{cases} 1, & if \quad U(0,1) < sigmoid\left(v_{id}\right) \\ 0, & otherwise \end{cases}$$

  - $$sigmoid\left(v_{id}\right) = \frac{1}{1+e^{-v_{id}}}$$

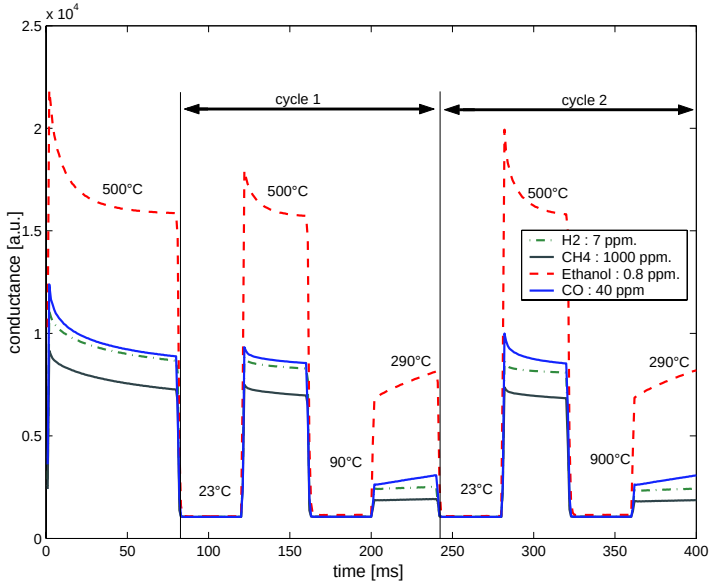
- Parameters $w$, $C_1$, and $C_2$ are same with original PSO

# Optimized Feature Computation

## Gas Detection

- The roles of feature computation techniques:
  - extract the meaningful information of raw data of sensor response
  - reduce the dimension size of variable vector of a pattern
  - increase the computation speed and classification accuracy

- In particular to the application of gas sensor systems, two feature computations have been proposed:
  - *Multi-level thresholding*
  - *Gaussian windowing* (kernel technique)




**Sensor response patterns during two temperature cycles**

- *Cooperation with LMT – Saarland University (Prof. Dr. rer. nat. A. Schütze)*

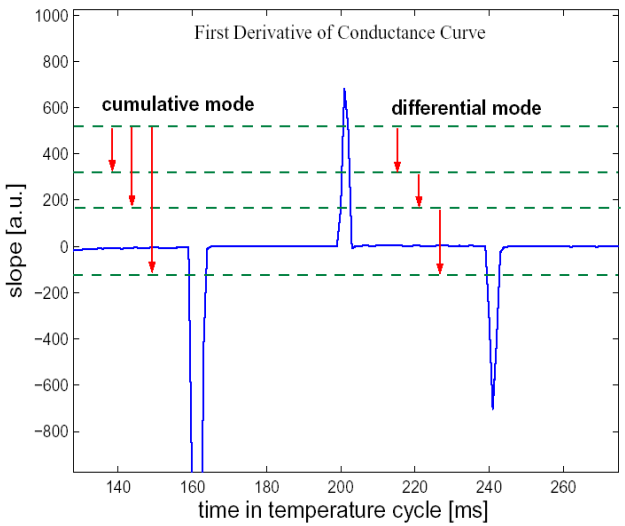
# Optimized Feature Computation
## Multi-Level Thresholding (MLT)

- MLT computes the features with similar to histogram and amplitude distribution
- Two methods of MLT, i.e., differential (DM) and cumulative (CM) modes
- The features of MLT can be computed as

$$z_i = \sum_{s=1}^{N_r} \delta(y_s, T_p, T_q)$$

$$\delta(y_s, T_p, T_q) = \begin{cases} 1 & T_p \le y_s \le T_q \\ 0 & otherwise \end{cases}$$

$y_s$ : magnitude value of sensor signal with s = 1, 2, ..., $N_r$
$N_r$ : total samples of a pattern
$i$ : a number of features ($i$ = $T$ - 1)
$T$ : a number of thresholds used
$Tp$ and $Tq$ : level-values
with $q$ = (2, 3, ... $T$) and $p$ = $q$ – 1 for DM, and
with $q$ = $T$ and $p$ = (1, 2, 3, ... $T$-1) for CM



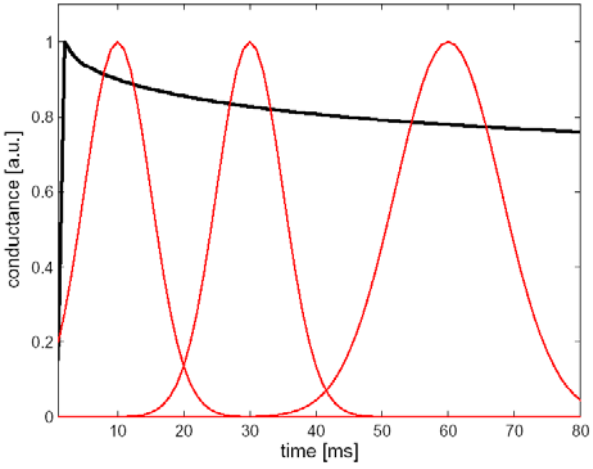

**Feature computation of MLT for a gas stimulus presentation of *first derivative* of conductance**

# Optimized Feature Computation

## Gaussian Windowing Function (GWF)

- Extract features directly from conductance curves (transient responses)
- Each feature is represented by a kernel-base, i.e., a Gaussian exponential function [Courte et al. 2003]
- Parameters : mean $\mu$ and standard deviation $\sigma$

- The features of Gaussian Windowing can be computed as
  - $$z_i = \sum_{s=1}^{N_r} y_s \cdot G(s, \mu_i, \sigma_i)$$
  - $$G(s, \mu_i, \sigma_i) = \exp^{-\frac{1}{2}\left(\frac{s-\mu_i}{\sigma_i}\right)^2}$$




**Feature computaion of Gaussian windowing (window time slicing) for a normalized conductance curve**

# Optimized Feature Computation

## Data Description and Parameter Setting

- Applying a benchmark data of a gas sensor system
- Types of gases, i.e., $H_2$, $CH_4$, *ethanol* and *CO*
- The data set consists of 810 measure values and 264 patterns
- Separated into training (144 patterns) and testing (120 patterns) sets
- Each experiment is repeated using 10 runs
- Each run is limited up to 100 iterations

- Population size is 20 individuals for both GA and PSO
- The number of nearest neighbors is set to five for the overlap (NPOM) measurement and the *k*NN voting classifier
- The classification accuracy is estimated using holdout method and leave-one-out cross-validation approach

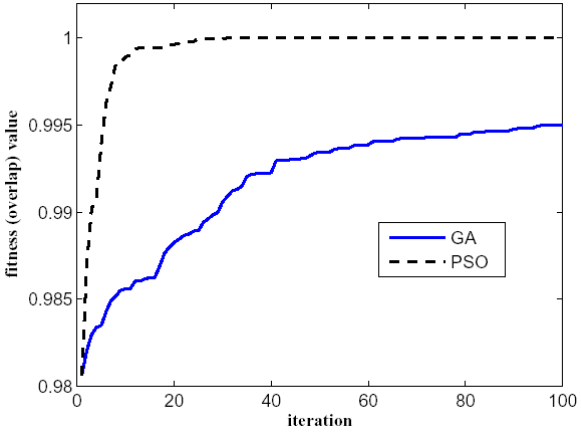
# Optimized Feature Computation

## Comparison between GA and PSO for MLT

**MLT differential mode** (L = 9)

| Method | overlap | Recognition accuracy (*k*NN) | | |
|---|---|---|---|---|
| | $q_o$ | train(%) | test(%) | test-LOO(%) |
| | Mean / Std | Mean / Std | Mean / Std | Mean / Std |
| **GA** | 0.9950 / 0.0035 | 99.44 / 0.55 | 99.67 / 0.58 | 99.17 / 0.79 |
| **PSO** | 1.00 / 0 | 100 / 0 | 100 / 0 | 99.83 / 0.35 |

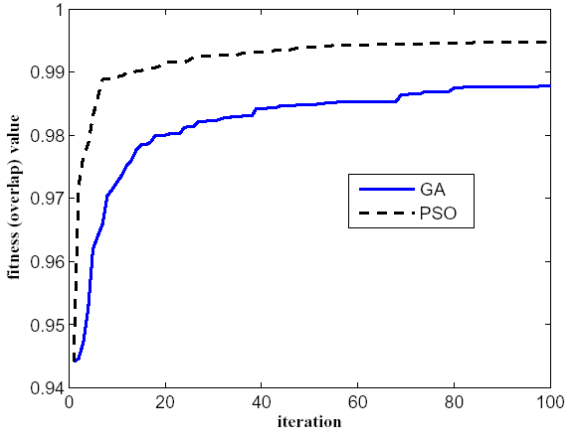
- **Manual (by expert) with (L = 12)** : 99.16%




**MLT cumulative mode** (L = 5)

| Method | overlap | Recognition accuracy (*k*NN) | | |
|---|---|---|---|---|
| | $q_o$ | train(%) | test(%) | test-LOO(%) |
| | Mean / Std | Mean / Std | Mean / Std | Mean / Std |
| **GA** | 0.9878 / 0.0044 | 98.89 / 0.36 | 99.50 / 6.36 | 98.67 / 1.48 |
| **PSO** | 0.9953 / 0.0024 | 99.10 / 0.34 | 99.92 / 0.89 | 99.83 / 0.35 |

- **Manual (by expert) with (L = 5)** : 97.16%



# Optimized Feature Computation

## Gaussian Windowing feature computation optimized by PSO

- The searching of best result runs with a variety number of kernels

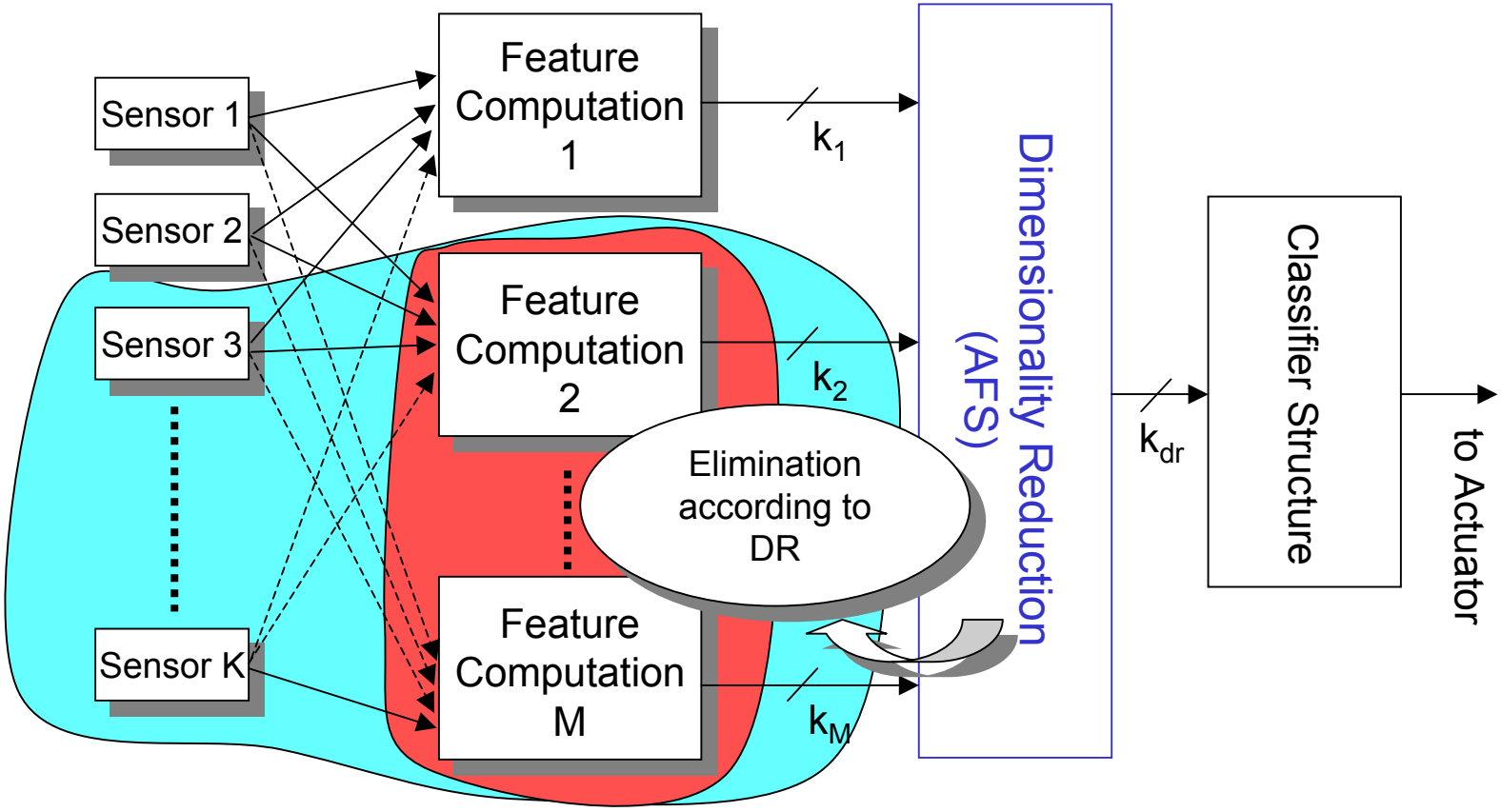
| No. Of Kernel (x 10) | overlap $q_o$ Mean / Std | Recognition accuracy (*k*NN) train(%) Mean / Std | test(%) Mean / Std | test-LOO(%) Mean / Std |
|---|---|---|---|---|
| **3** | 0.9806 / 0.0044 | 97.91 / 0.65 | 99.00 / 0.66 | 95.50 / 2.29 |
| **4** | 0.9791 / 0.0081 | 97.78 / 0.79 | 99.00 / 0.77 | 95.83 / 1.76 |
| **5** | 0.9794 / 0.0021 | 98.13 / 0.03 | 99.67 / 0.43 | 96.08 / 1.11 |
| **6** | 0.9797 / 0.0034 | 97.71 / 0.74 | 98.75 / 0.90 | 94.92 / 2.17 |
| **7** | 0.9795 / 0.0015 | 98.13 / 0.57 | 99.25 / 0.73 | 96.92 / 1.11 |
| **8** | 0.9786 / 0.0027 | 97.92 / 0.46 | 99.00 / 0.53 | 95.67 / 0.95 |
| **9** | 0.9786 / 0.0031 | 97.92 / 0.46 | 99.08 / 0.61 | 95.83 / 1.36 |
| **10** | 0.9787 / 0.0016 | 98.13 / 0.47 | 99.75 / 0.40 | 96.08 / 0.88 |

- No experience reference is available

# Optimal Feature Space under Constraints

## Dimensionality Reduction for Implicit Power Saving

- Structural simplification of first-cut design according to DR results:



# Optimal Feature Space under Constraints

## Feature Selection with Acquisition Cost

- Majority of approaches do not consider cost (e.g., power dissipation) of features / grouped features within feature subset selection.
- One rare example by (Paclik & Duin 2002).
- Our approach incorporates inspiration from this work and evolutionary computation.

- Accumulative expression:

$$K = q_{ov} + A \times \left(1 - \frac{C_S}{C_T}\right)$$

where

$A$ : weight of feature cost parameter

$C_S$ : sum of cost of active features

$C_T$ : sum of all cost of features

# Optimal Feature Space under Constraints

## Experiments and Results - Iris Data

- Common benchmark: consists of 4 features, 3 classes and 150 patterns (75 patterns for train and test).
- Dataset is repeated 4 times with different arbitrary cost assignment per feature.
  - Cost assignment 1 : [ 2, 3, 14, 16, 10, 8, 4, 15, 6, 5, 20, 3, 3, 12, 18, 17 ]
  - Cost assignment 2 : [ 4, 1, 20, 18, 3, 4, 17, 20, 1, 2, 15, 15, 2, 3, 18, 22 ]

| Iris Data | $q_{ov}$ | Cost | Classification rate (%) | Selected Features |
|---|---|---|---|---|
| 16 features | 0.95417 | 156 | 90.333 | All |
| FS (5 of 16) | 0.98200 | 65 | 94.667 | 3, 4, 8, 12, 16 |
| FS + Cost 1 | 0.98056 | 7 | 96.000 | 7, 12 |
| FS + Cost 2 | 0.96318 | 16 | 93.333 | 2, 12 |

# Optimal Feature Space under Constraints

## Experiments and Results – Eye Tracking Image Data

- Consist of 3 Groups: Gabor (12), ELAC (13) and LOC (33 features)
- Each group has 2 classes and 133 patterns (72 patterns for train and 61 for test) computed from 17 * 17 grey value images.
- Simplified cost model: multiplication is assumed the cost of 10 addition

  Gabor : 6358 / feature  LOC : 1445 / feature

  ELAC : 3179 / feature

| Eye-Image Data | $q_{ov}$ | Cost | Classification rate (%) | Selected Features |
|---|---|---|---|---|
| 58 features | 0.95482 | 165308 | 98.361 | All |
| FS (17 of 58) | 1.00 | 62713 | 98.361 | 1, 3, 8, 9, 11, 12, 14, 15, 16, 18, 21,28, 29, 34, 38, 54, 58 |
| FS + Cost | 0.99976 | 21675 | 98.361 | 12, 14, 18, 21, 37, 38, 39, 54 |

# Conclusions

- Optimization of feature computation (MLT and GWF) and dimensionality reduction proved feasible and salient
- The experiment result showed PSO performs better than GA
- Optimization approach outperformed the existing expert solution !
- Basic investigations of cost aware feature selection delivered encouraging result for a benchmark problem and simple model

***Future Work***:

- Extension by additional signal processing and classification techniques will be considered
- Develop novel methods of data fusion for optimized sensor system design
- Aspects of adaptive processes, e.g., self-calibration process will also be considered

# Thank You