Project no.:	**IST-FP6-FET-16276-2**
Project full title:	**Learning to emulate perception action cycles in a driving school scenario**
Project Acronym:	**DRIVSCO**
Deliverable no:	**D6.1**
Title of the deliverable:	**Algorithm for adaptive subspace transformation based on mutual information and class labels**

Date of Delivery:	14.06.2008	
Organization name of lead contractor for this deliverable:	KUL	
Author(s):	M. Van Hulle, N. Chumerin (KUL)	
Participant(s):	KUL,BCCN,AAU,UMU,VMU	
Work package contributing to the deliverable:	WP6	
Nature:	R	
Version:	2.0 (revised 10.06.2008)	
Total number of pages:	18	
Start date of project:	1 Feb. 2006	**Duration:** 42 months

Project Co-funded by the European Commission		
Dissemination Level		
PU	Public	**X**
PP	Restricted to other program participants (including the Commission Services)	
RE	Restricted to a group specified by the consortium (including the Commission Services)	
CO	Confidential, only for members of the consortium (including the Commission Services)	

Summary: We perform a detailed benchmarking of two adaptive subspace methods that are based on mutual information maximization between the data points projected in the developed subspace and their class labels. However, we found that, when these methods are followed by a classifier, no acceptable results are obtained for large, real-world datasets. We found this in particular to be the case with the Independently Moving Objects (IMO) detection problem, where the IMO and the background are used as class labels. As an alternative, we suggest a specially designed multilayered perceptron for detecting IMOs.

Contents

1 Executive Summary

As mentioned in the Technical Annex, the purpose of Task 6.1 is to develop compact representations of the extracted Structured Visual Events (SVEs) in WP4. Initially, a text mining approach in terms of SVE occurrences (weighted histograms) was proposed, followed by a low-dimensional (subspace) representation in terms of (action, SAE,...) labels by optimizing mutual information between the subspace projected data and the labels (i.e., supervised feature extraction) (Torkkola, 2002). Soon after the start of the project it was realized that a text mining approach was not feasible since the stable detection of Independently Moving Objects (IMOs) was a much harder problem than anticipated. Indeed, the IMO maps of WP4 often appeared as disconnected and sparse clouds of pixels representing the same object. Furthermore, false positives often occurred, thus, pixels falsely labeled as IMOs (see Fig. 1 in D6.2a) Hence, the disambiguation of the SVEs, mentioned in the TA for WP6, received more attention than foreseen.

Torkkola's method was, therefore, put to the test for the stable detection of IMOs given labels of what is an IMO and what is background (supervised approach). But prior to that, we benchmarked Torkkola's and compared it with an alternative technique by Artés-Rodríguez and Leiva-Murillo (2006) on a number of synthetic examples for which the solution is analytically known. This was deemed necessary given the SVE data are noisy and prone to outliers. It turned out that, with Torkkola's or its alternative, no acceptable results were obtained for large, real-world datasets. When applying a classifier, the classification performance was unsatisfactory. Also, the algorithms easily could get trapped in local optima. We also verified all this on the IMO vs. background detection problem just mentioned, where the dimensions consisted of several image cues.

As a remedy, we designed a multilayered perceptron for detecting IMOs and for which encouraging results were obtained. This "cue fusion" solution was the highlight of the first year's review since it enabled us to attach descriptors to the IMOs (such as their tracking accuracy, speed, distance, acceleration,...). These results are reported in D6.2.

Status of the Task:
The original idea of subspace representation based on mutual information was not further considered, after the negative benchmarking results. Instead, a multilayered perceptron was used for cue fusion. We have in the second year also examined a clustering approach, but this is not reported on here. Part of the results reported in this deliverable have been published by us [1].

Revision notes:
In order to answer the criticisms of the reviewers, we have included: 1) a thorough introduction (called here *Executive summary*), where the motivation and the change in the course of WP6 is explained, 2) the status of the task (see above), and 3) a list of the concrete contributions of the papers cited in this

report (see *References*).

2 Introduction – Aim

Feature Extraction (FE) is a more general method in which one tries to develop a transformation of the input space onto the low-dimensional subspace that preserves most of the relevant information. We will further focus on linear FE methods which means that they can be represented by a linear transformation $\mathbf{W} : \mathbb{R}^D \rightarrow \mathbb{R}^d$, $D > d$. Feature Extraction methods can be supervised or unsupervised, depending on whether or not class labels are used. Among the unsupervised methods, Principal Component Analysis (PCA) [2], Independent Component Analysis (ICA) [3], and Multidimensional Scaling (MDS) [4] are the most popular ones. Supervised FE methods (and also Feature Selection methods) either use information about the current classification performance, or use some other, indirect measure. Methods of the first type are called *wrappers*, those of the type *filters*. One expects that, in the case of a classification problem, supervised methods will perform better than unsupervised ones.

Recently, a method has been introduced by Torkkola [5] that has attracted a lot of attention. Consider the data set $\{\mathbf{x}_i, c_i\}$, $i = 1, \ldots, N$ with $\mathbf{x}_i \in \mathbb{R}^D$ the data points, and c_i the class labels taken from the discrete set $\mathcal{C} = \{\mathtt{c}_p\}$, $p = 1, \ldots, N_c$. The objective is to find a linear transformation $\mathbf{W} \in \mathbb{R}^{D \times d}$ for which the mutual information (MI) of the transformed data points $Y = \{\mathbf{y}_i\} = \{\mathbf{W}^T \mathbf{x}_i\}$ and the corresponding labels $C = \{c_i\}$ is maximized. The objective is different from ICA's where MI between the transformed data components is minimized. Also, the presence of the labels C makes the objective different. Torkkola derived an expression for MI based on Renyi's quadratic entropy [6], instead of Shannon's entropy, and a plug-in density estimate based on Parzen windowing.

Prior to Torkkola, Bollacker and Ghosh [7] proposed an incremental approach to MI maximization that was derived by rewriting the original MI objective function as a sum of MI terms between the one-dimensional projections and the corresponding class labels. A polytope algorithm was used for the optimization and histograms for estimating the probabilities. Very recently, a method based on the same reformulation of the MI objective function was introduced by Leiva-Murillo and Artés-Rodríguez (2006) [8]. However, they used gradient descent as an optimization strategy, and expressed the one-dimensional MI terms as one-dimensional negentropies, which were then estimated using Hyvärinen's robust estimator [9].

The purpose of this report is to perform an in-depth comparison of the two MI based FE methods. The report is structured as follows. Section 3 briefly describes the FE method based on quadratic MI maximization, as proposed by Torkkola. In Section 4 we also briefly describe the approach proposed by Leiva-Murillo and Artés-Rodríguez. In Section 5, we describe a number of measures for FE comparison, and in Section 6 we explain our comparison methodology. The results of the comparison are given in Section 7, followed by a Discussion in

Section 8. In Section 9, we consider feature extraction followed by a classifier, and consider the detection of Independently Moving Objects (IMOs) from their backgrounds as a classification problem for a large, real-world data set. We found no satisfactory classification performance (*i.e.*, IMO segmentations in real movies). We suggest an alternative strategy based on a multilayered perceptron (MLP) classifier. Finally, we conclude this report in section 10.

3 Torkkola's Method

Given two random variables X_1 and X_2 with joint probability density $p(x_1, x_2)$ and marginal probability densities $p_1(x_1)$ and $p_2(x_2)$, the mutual information (MI) can be expressed as:

$$I(X_1, X_2) = K(p(x_1, x_2), p_1(x_1)p_2(x_2)), \tag{1}$$

with $K(\cdot, \cdot)$ the Kullback-Leibler divergence.

In order to estimate MI, Torkkola and Campbell [5] use the quadratic measures K_C or K_T originally introduced by Principe and co-workers [6]:

$$K_C(f, g) = \log \frac{\int f^2(\mathbf{x})d\mathbf{x} \int g^2(\mathbf{x})d\mathbf{x}}{\left(\int f(\mathbf{x})g(\mathbf{x})d\mathbf{x}\right)^2} \tag{2}$$

$$K_T(f, g) = \int \left(f(\mathbf{x}) - g(\mathbf{x})\right)^2 d\mathbf{x}. \tag{3}$$

For continuous-valued Y and discrete-valued C, using (1), (2) and (3), one can derive two types of MI estimates:

$$I_C(Y, C) = \log \frac{V_{(cy)^2} V_{c^2y^2}}{(V_{cy})^2}, \tag{4}$$

$$I_T(Y, C) = V_{(cy)^2} + V_{c^2y^2} - 2V_{cy}, \tag{5}$$

where:

$$\begin{aligned} V_{(cy)^2} &= \sum_{c \in \mathcal{C}} \int_{\mathbf{y}} p^2(\mathbf{y}, c)d\mathbf{y}, \\ V_{c^2y^2} &= \sum_{c \in \mathcal{C}} \int_{\mathbf{y}} p^2(c)p^2(\mathbf{y})d\mathbf{y}, \\ V_{cy} &= \sum_{c \in \mathcal{C}} \int_{\mathbf{y}} p(\mathbf{y}, c)p(c)p(\mathbf{y})d\mathbf{y}. \end{aligned} \tag{6}$$

The class probability can be evaluated as $p(\mathsf{c}_p) = J_p/N$, where J_p is the number of samples in class c_p. The density of the projected data $p(\mathbf{y})$ and the joint

density $p(\mathbf{y}, c)$ are estimated with the Parzen window approach [10]:

$$\begin{aligned} p(\mathbf{y}) &= \frac{1}{N}\sum_{i=1}^{N} G(\mathbf{y}-\mathbf{y}_i, \sigma^2 I) \\ p(\mathbf{y}, \mathsf{c}_p) &= \frac{1}{N}\sum_{i=1}^{J_p} G(\mathbf{y}-\mathbf{y}_{pj}, \sigma^2 I), \end{aligned} \tag{7}$$

with $G(\mathbf{x}, \Sigma)$ the Gaussian kernel with center $\mathbf{x}$ and covariance matrix Σ, and $\mathbf{y}_{jp}$ the j-th sample in class c_p. In order to reduce the number of parameters to optimize, Torkkola proposes a parametrization of the desired matrix $\mathbf{W}$ in terms of Givens rotations in $\mathbb{R}^D$. As a result, there are only $d(D-d)$ parameters (rotation angles) to optimize instead of D^2. Obviously, the maximal number of parameters to estimate occurs for d near $D/2$. The computational complexity of the method is claimed to be $O(N^2)$.

4 Artés-Rodríguez's Method

In the Artés-Rodríguez method, an objective function (*global* MI) in terms of the sum of *individual* MI's is considered:

$$I_{AR}(Y, C) = \sum_{i=1}^{d} I(y_i, c) = \sum_{i=1}^{d} I(\mathbf{w}_i^T \mathbf{x}, c), \tag{8}$$

with $y_i = \mathbf{w}_i^T \mathbf{x}$ the data projected onto direction $\mathbf{w}_i$, and $\mathbf{w}_i \in \mathbb{R}^D$ the i-th column of the desired orthonormal matrix $\mathbf{W}$.

Assuming the original data is whitened, each individual MI can be estimated as:

$$I(y_i, c) = \sum_{p=1}^{N_c} p(\mathsf{c}_p)\left(J(y_i|\mathsf{c}_p) - \log\sigma(y_i|\mathsf{c}_p)\right) - J(y_i), \tag{9}$$

with $y_i|\mathsf{c}_p$ the projection of the p-th class' data points onto the $\mathbf{w}_i$ direction, $J(\cdot)$ the negentropy, and $\sigma(\cdot)$ the standard deviation. Hyvärinen's robust estimator [9] for the negentropy is used:

$$\begin{aligned} J(z) \approx\ & k_1\left(E\{z\exp(-z^2/2)\}\right)^2 \\ & + k_2\left(E\{\exp(-z^2/2)\} - \sqrt{1/2}\right)^2, \end{aligned} \tag{10}$$

with $k_1 = 36/(8\sqrt{3}-9)$ and $k_2 = 24/(16\sqrt{3}-27)$. In a *top-down* scheme, one should sequentially (thus, one-by-one) obtain the projection directions $\mathbf{w}_i$ thereby preserving the two constraints: $\|\mathbf{w}_i\| = 1$ and $\mathbf{w}_i^T\mathbf{w}_j = 0$ for $1 \le j < i$. The second constraint means that each projection direction must be searched in the subspace orthogonal to the projection directions already obtained, and this causes the search for each new projection direction to be carried out in a

subspace of decreasing dimension. The sequence of individual MI's obtained in this way is also decreasing: $I(y_i, c) > I(y_j, c)$ for $i < j$. The *bottom-down* scheme involves a sequential removing of the directions with minimum individual MI's between the variables and classes.

5 Measures for Comparison

In order to compare the two FE methods, we use four different MI estimators: the two mentioned above, I_C and I_{AR}, the binned estimator I_B, and the one proposed by Kraskov and co-workers [11], namely, the $I^{(2)}$ estimator (rectangular version).

The most straightforward, and most widely used method to estimate the MI between two variables X and Y is the histogram-based approach. The support of each variable is partitioned into bins of finite size. Denoting by $n_x(i)$ ($n_y(j)$) the number of points falling in i-th bin of X (j-th bin of Y), and $n(i,j)$ the number of points in their intersection, we can estimate MI:

$$I_B(X,Y) = \log N + \frac{1}{N}\sum_{i,j} n(i,j) \log \frac{n(i,j)}{n_x(i) n_y(j)}. \tag{11}$$

Unfortunately this estimator is biased, even in the case of adaptive partitioning. Another disadvantage of the binned estimator is the high memory requirements in the high-dimensional case.

Kraskov MI estimator $I^{(2)}$ is based on entropy estimation using k-nearest neighbor statistics. Let X and Y are normed spaces with norms $\|\cdot\|_X$ and $\|\cdot\|_Y$ respectively. Consider new space $Z = X \times Y$ with norm $\|\cdot\|_Z$ which for every $\mathbf{z} \in Z,\ \mathbf{z} = (\mathbf{x}, \mathbf{y})$ is defined as

$$\|\mathbf{z}\|_Z = \max\{\|\mathbf{x}\|_X, \|\mathbf{y}\|_Y\}.$$

For fixed natural k let us denote by $\epsilon(i)/2$ the distance from $\mathbf{z}_i$ to its k-th neighbor, and by $\epsilon_x(i)/2$ and $\epsilon_y(i)/2$ the distances between the same points projected into the X and Y subspaces. Denoting by

$$\begin{aligned} n_x(i) &= \#\left\{\mathbf{x}_j : \|\mathbf{x}_i - \mathbf{x}_j\|_X \le \epsilon_x(i)/2\right\}, \\ n_y(i) &= \#\left\{\mathbf{y}_j : \|\mathbf{y}_i - \mathbf{y}_j\|_Y \le \epsilon_y(i)/2\right\} \end{aligned}$$

MI can be estimated by

$$I^{(2)}(X,Y) = \psi(k) - \frac{1}{k} - \langle \psi(n_x) + \psi(n_y) \rangle + \psi(N), \tag{12}$$

here $\langle \dots \rangle = N^{-1} \sum_{i=1}^{N} E\left\{\dots(i)\right\}$ is averaging operation both over all $i = 1, \dots, N$ and over all realizations of the random samples and $\psi(x)$ is digamma function:

$$\psi(x) = \frac{1}{\Gamma(x)} \frac{d\Gamma(x)}{dx}. \tag{13}$$

It satisfies the recursion $\psi(x+1) = \psi(x) + 1/x$ and $\psi(1) = -C$ where $C = 0.5772156\dots$ is the Euler-Mascheroni constant. For large x, $\psi(x) \approx \log x - 1/2x$.

Data set name	Dimension (D)	Number of samples (N)	Number of classes (N_c)
Iris	4	150	3
Pima	8	500	2
Glass	9	214	7
Pipeline flow	12	1000	3
Wine	13	178	3

Table 1: Information about used real data sets

6 Comparison Methodology

In order to have a fair comparison, we use the original source code of the Artés-Rodríguez algorithm (courtesy of Leiva-Murillo and Artés-Rodríguez) and the publicly available implementation of Torkkola's approach `MeRMaId-SIG` by Kenneth E. Hild II [12]. For the Artés-Rodríguez algorithm, we choose the top-down scheme. We consider both synthetic and real world data sets. The synthetic data set consists of a variable number of equal-sized, normally distributed clusters (modes) in $\mathbb{R}^D$. The clusters centers are Gaussianly distributed with variance equal to 3. All data sets are centered and whitened before applying the respective FE methods. We consider $N_c = 3, \ldots, 10$ clusters, and use 1000 data sets with $d = 1, \ldots, D-1$ subspace dimensions. The MI estimators' means and standard deviations for the 1000 data sets are then plotted as a function of the subspace dimensionalities d. For the real-world data sets, we compute the MI estimates for each possible subspace dimension d. The Pipeline Flow data set was taken from Aston University[1]. The rest of the real-world data sets were taken from the UCI Machine Learning Repository[2]. If the data dimensionality was more than 9, we did not evaluate I_B (binned estimator) due to memory limitations.

The algorithms are implemented using quite different, yet simple optimization techniques: the Artés-Rodríguez algorithm employs a simple adaptation of the learning rate during evaluation, while the `MeRMaId-SIG` uses a pure gradient ascent with constant learning rate and fixed number of iterations. Due to this, a fair comparison of the run times is not straightforward. Therefore, we determine the number of float-point operations (flops) needed for one gradient evaluation of each algorithm. It is a more relevant measure than the average computing time because it does not depend on the optimization techniques used by these algorithms.

The flops were obtained using the `flops` function (in Matlab 5.3) on data sets with $N \in \{1000, 2000, 3000, 4000\}$ and $D \in \{4, 8, 12, 16\}$.

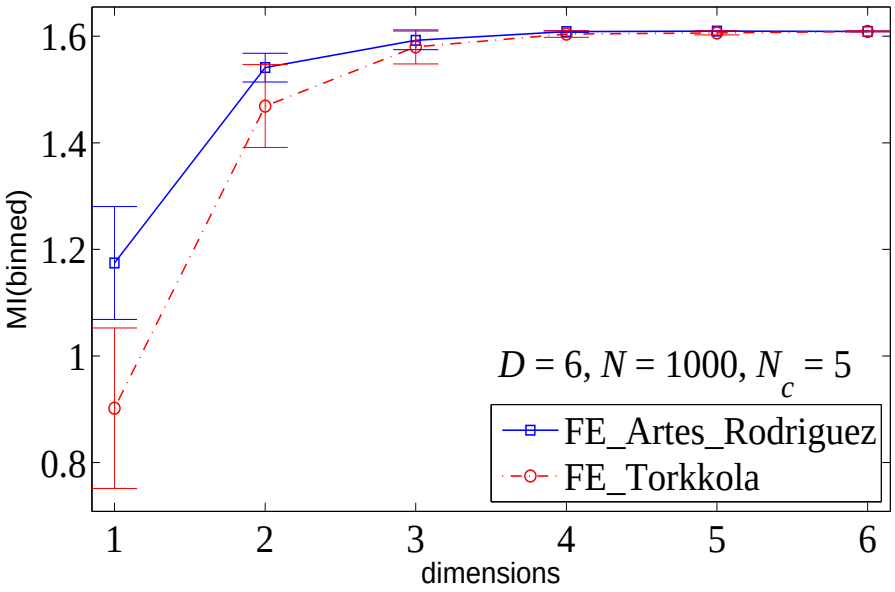

Figure 1: Mean of I_B vs. d

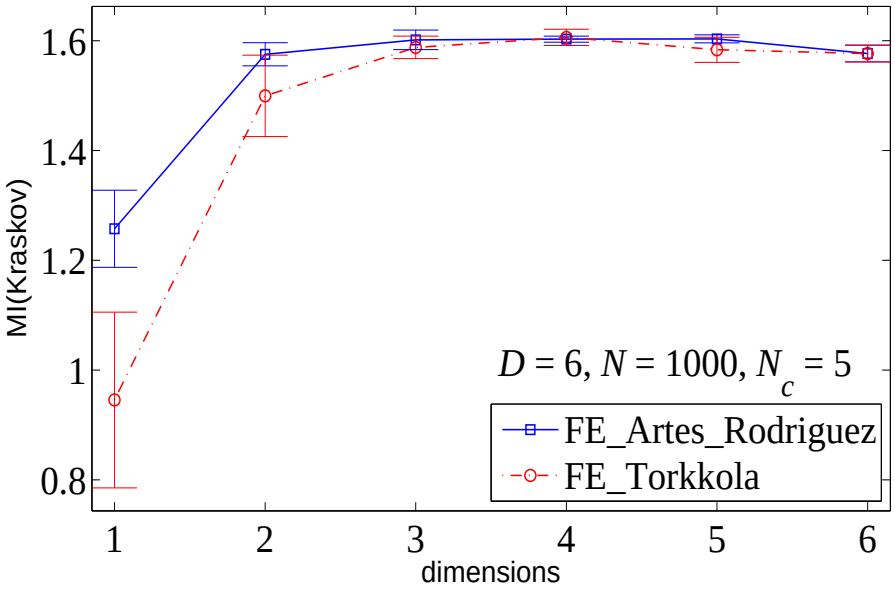

Figure 2: Mean of $I^{(2)}$ vs. d

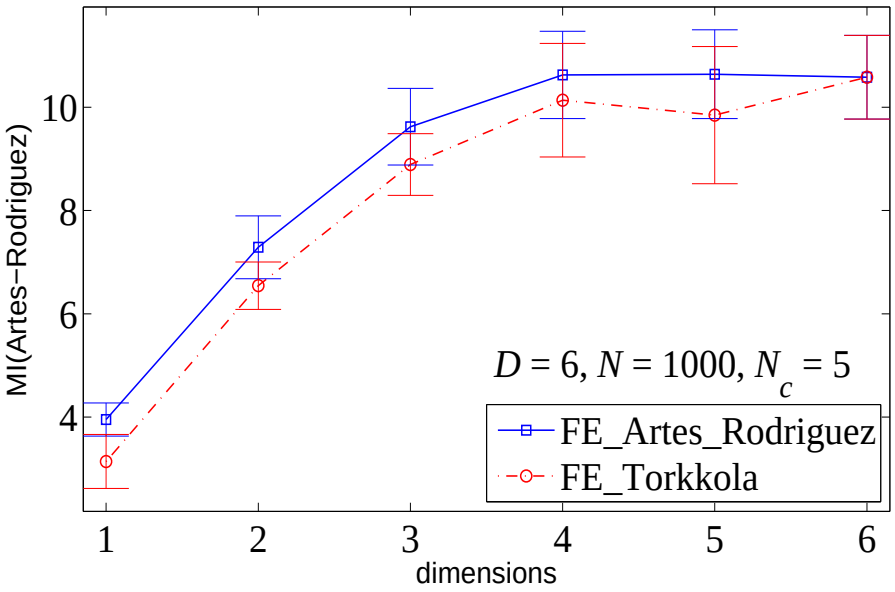

Figure 3: Mean of I_{AR} vs. d

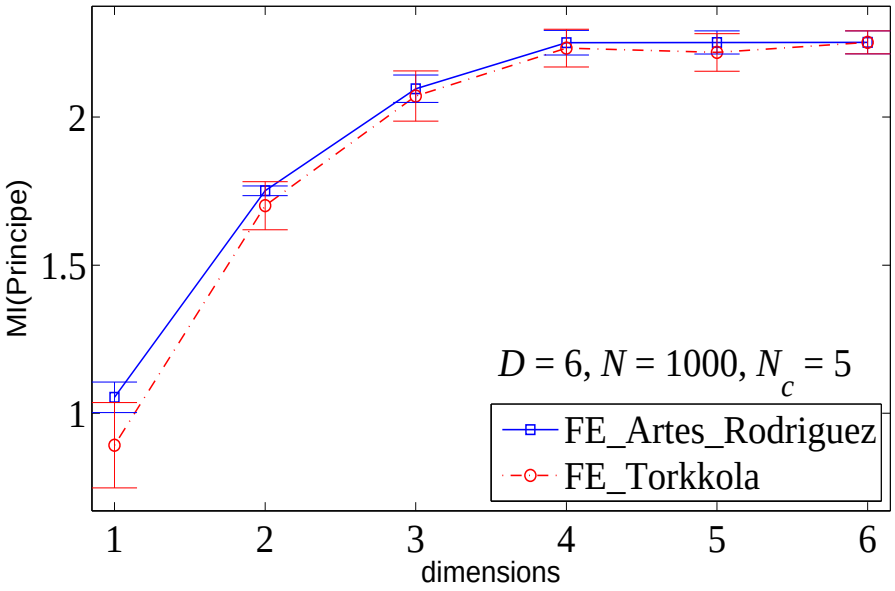

Figure 4: Mean of I_C vs. d

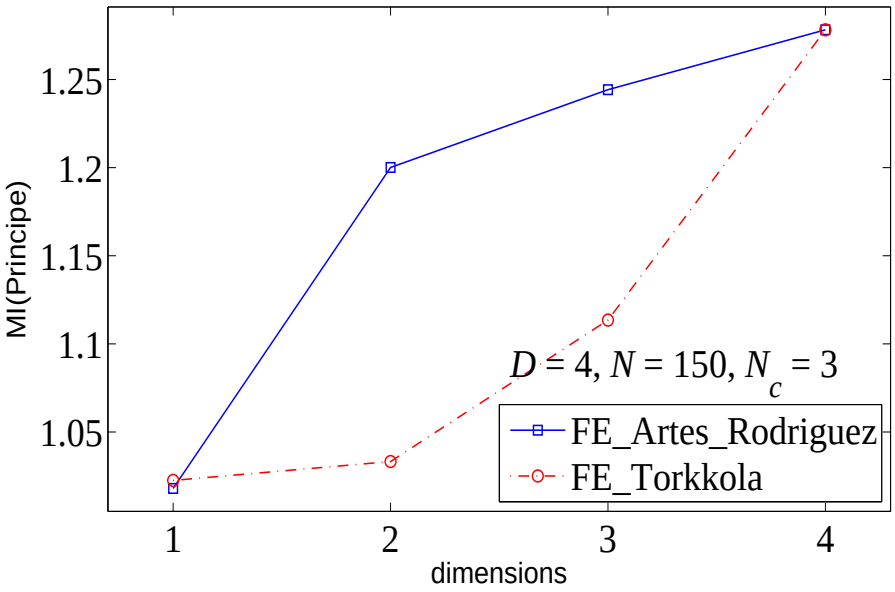

Figure 5: I_C versus d for Iris Plants Database

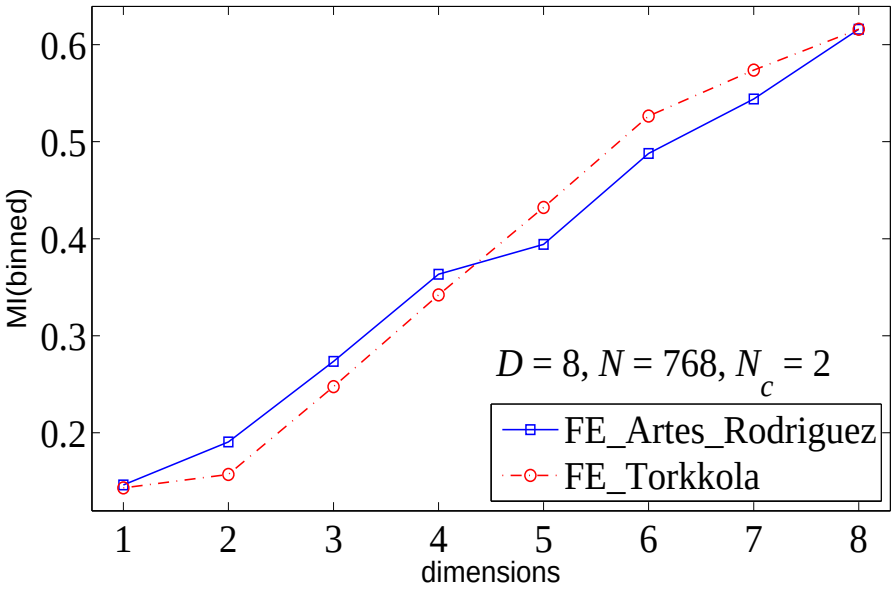

Figure 6: I_B versus d for Pima Indians Diabetes Database

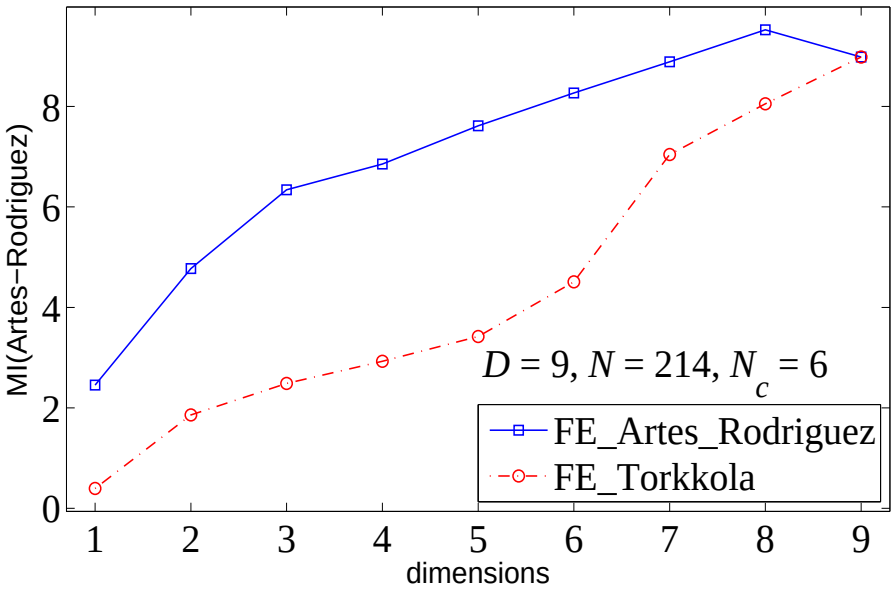

Figure 7: I_{AR} versus d for Glass Identification Database

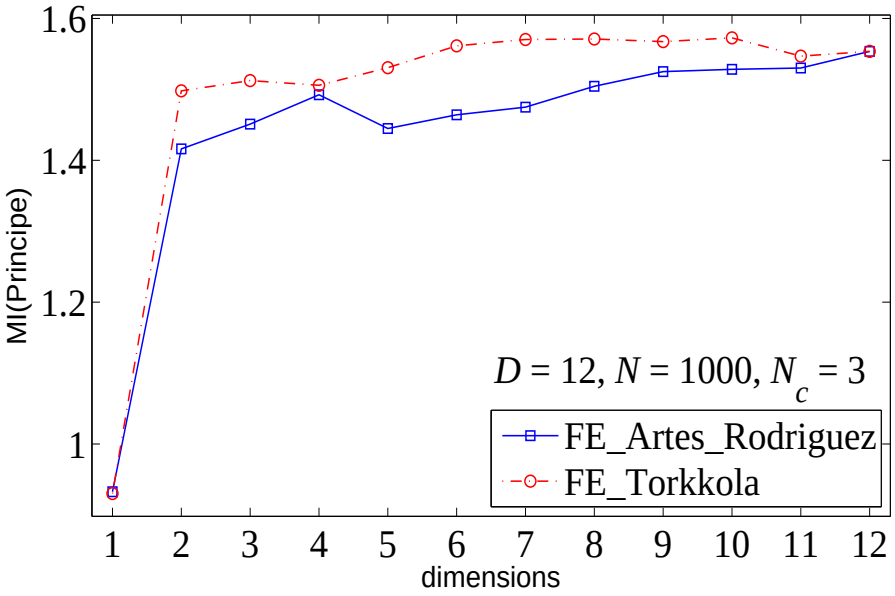

Figure 8: I_C versus d for Pipeline Flow data

Data set	Approach	$\langle I_B \rangle$	$\langle I_C \rangle$	$\langle I_{AR} \rangle$	$\langle I^{(2)} \rangle$
Iris	AR	1.0391	1.1541	5.0251	0.9944
	Torrkola	1.0181	1.0565	4.0409	0.9561
Pima	AR	0.3428	0.2089	0.8528	0.1628
	Torrkola	0.3461	0.2026	0.4140	0.1678
Glass	AR	0.7078	0.4212	6.8401	0.5952
	Torrkola	0.7430	0.4409	3.8368	0.4764
Pipeline	AR	1.0814	1.4331	20.461	1.0668
	Torrkola	1.0749	1.4889	5.9973	1.0605
Wine	AR	1.0668	1.5019	8.3007	0.8798
	Torrkola	0.9009	1.2422	3.0588	0.7194

Table 2: Averages of the estimated MI for all real data sets considered and $d = 1, \ldots, D-1$; $\langle I_B \rangle$. Note that the estimates were computed only for $d < 9$) (see text).

7 Results

For the synthetic data sets we show only the case of $D = 6$, $N = 1000$ and $N_c = 5$ (Figs. 1–4). The results for the real-world data sets are shown in Figs. 5–8 and in Table 2.

The speed comparison results are shown in Table 3. The case $D = 8$ for Torkkola's method is shown in Fig. 9 in more detail. We do not show the plots for the Artés-Rodríguez approach because each gradient evaluation needs the same number of flops for all $d = 1, \ldots, D-1$.

For data sets with fixed numbers of samples N, fixed dimensions D and different numbers of clusters N_c, gradient evaluation in both methods needs almost the same numbers of floating point operations (the deviation in flops for constant N, D and $N_c \in \{5, 10, 20\}$ was less than 1%). This is the reason why we present here the comparison only for $N_c = 5$ and different N and D values. The CPU time should grow with increasing N_c, however, it stays almost constant. We explain this by the highly optimized manner Matlab treats matrix computations: for fixed N, the more classes we have, the more portions of the data (with smaller sizes) are processed in a vectorized manner.

8 Discussion

The results show that, for most data sets, the Artés-Rodríguez approach yields better results. From our point of view, one of the reasons of the better performance of the Artés-Rodríguez algorithm is the fact that I_{AR} is more smoother and has less local optima than the other measures, including the I_C metric used in Torkkola's, with almost coinciding maxima. This is illustrated in Fig. 10.

[1] http://www.ncrg.aston.ac.uk/GTM/3PhaseData.html
[2] http://www.ics.uci.edu/~mlearn

D	N	Torkkola	Artés-Rodríguez
4	1000	0.165	0.253 . . . 0.390
	2000	0.329	0.505 . . . 0.778
	3000	0.493	0.757 . . . 1.166
	4000	0.657	1.009 . . . 1.554
8	1000	0.438	1.976 . . . 5.121
	2000	0.874	3.900 . . . 9.977
	3000	1.310	5.824 . . . 14.833
	4000	1.746	7.748 . . . 19.689
12	1000	0.841	6.977 . . . 27.540
	2000	1.677	13.517 . . . 50.544
	3000	2.513	20.057 . . . 73.548
	4000	3.349	26.597 . . . 96.552
16	1000	1.372	17.556 . . . 104.446
	2000	2.736	33.192 . . . 175.022
	3000	4.100	48.828 . . . 245.598
	4000	5.464	64.464 . . . 316.174

Table 3: Comparison of floating point operations (in Mflops) needed for one gradient evaluation.

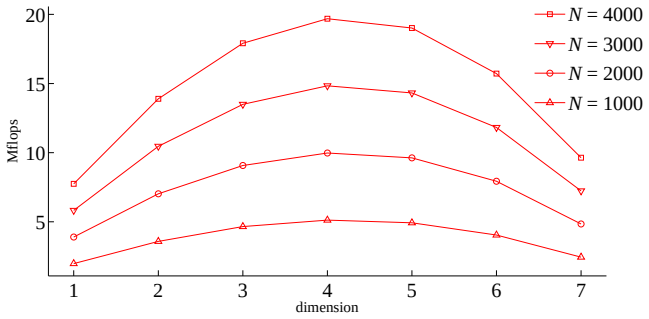

Figure 9: Plots of the floating point operations required for Torkkola's gradient evaluation for $D = 8$ and different N. It should not come as a surprise that the shape of plots reflect the quadratic nature of the number of parameters to optimize (see text).

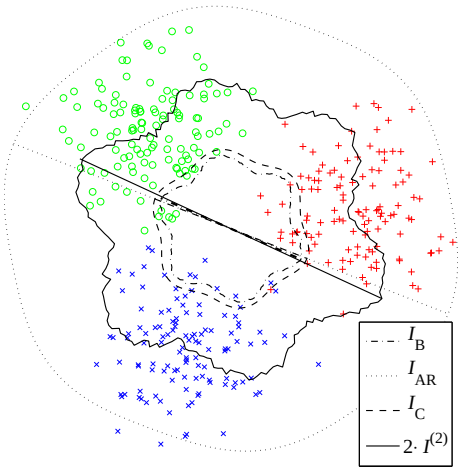

Figure 10: Plots of I_B, I_{AR}, I_C and $I^{(2)}$ (which is doubled for the sake of exposition) as a function of the angle of the direction on which the data points are projected, given a two-dimensional data set consisting of 3 equal sized Gaussian clusters. For each plot the direction of the maxima is indicated with a line segment. It can be clearly seen that I_{AR} is smoother than the other measures, with almost coinciding maxima.

One should also remind that in the Artés-Rodríguez approach, for all computations of the gradient, one-dimensional projections are used, whereas Torkkola's approach gradient evaluations are based on data of dimensionality $d(D-d)$. This could be beneficial as well.

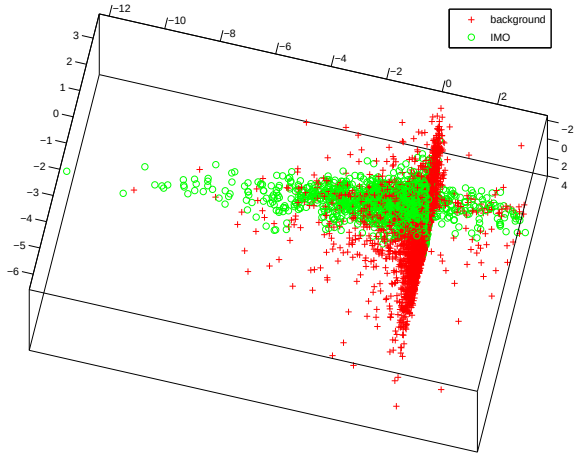

Figure 11: Result of real data (50 frames of *city3* sequence) subspace transformation (feature extraction) from 9-D early vision feature space to 3-D space, using Torkkola's method. Only 10000 samples are plotted for the sake of exposition.

Another issue is data preprocessing. In Torkkola's only PCA is used as data preprocessing, whereas Artés-Rodríguez employs a more sophisticated technique: successively PCA and SIR (Sliced Inverse Regression) are used, which already yields a quite good MI result.

9 Application to IMO detection

The application above is only valid for relatively small datasets (no more then 10^5 samples). For larger data sets, the application of these methods become problematic because they rely on batch learning. Another important issue is that in reality the dimensionality can be high. We consider the detection of Independently Moving Objects (IMOs) based on several cues, that were previously extracted from the image, such as optic flow, average flow, disparity of the previous, current and next frames, ON/OFF contours, static (1D) disparity, grey scale image, (x, y) position coordinates of the pixels, and so on... We consider two types of classes: IMOs and background (non-IMOs). Hence, we perform cue fusion given a specific task, namely, the separation of the two classes as much as possible. In our simulations, the number of cues varies between 8 to 20, de-

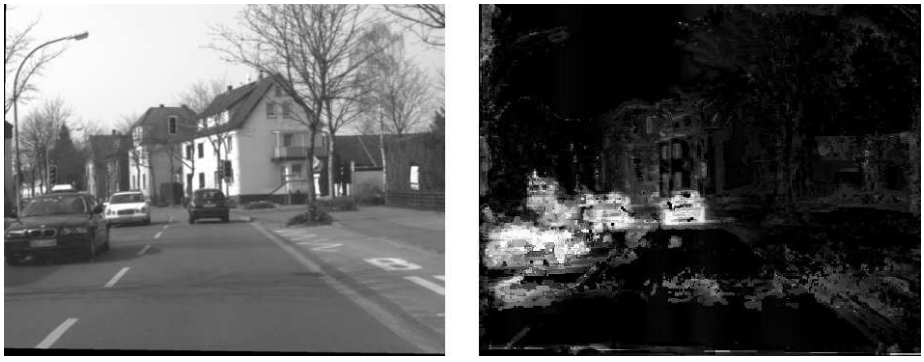

Figure 12: The left image is a rectified version of the original (left) one, which is frame number 63 of the *city3* sequence. The right image is the result of the MLP-classification of the left image.

pending on the initial setup (*i.e.*, set of cues selected for further processing). A result of Torkkola's feature extraction method, applied on a real movie sequence (*city3*), is shown in Fig. 11. We found that, by reducing the dimensionality of the original cue space, we actually made the ensuing classification task more difficult. We can explain this by noting that the classifier is trained separately from the adaptive subspace algorithm (which only uses the class labels for finding the optimal subspace but does not take into account the geometry of the actual classifier).

One way to improve the classification performance is to combine cue fusion with classification and train them as one classification paradigm. This can be done in the framework of multilayered perceptrons (MLPs), where cue fusion and classification stages are represented by different layers. In our experiments, we tried different MLP configurations, but all of them had 3 layers with (4–8) linear neurons in the first layer, (8–16) nonlinear neurons in the second layer and one linear output neuron. The first layer could be considered as the fusion part, the second layer coupled with the third one – as the classification part. A typical result for the *city3* sequence is shown in Fig. 12.

Because of the better performance with the MLP, we propose to switch from MMI feature extraction methods followed by classification to MLP classifiers.

10 Conclusion

In line with the TA, we have considered Torkkola's [5] supervised subspace method based on mutual information: We have benchmarked two supervised subspace methods based on mutual information: not only the algorithm of Torkkola [5], but also that of Artés-Rodríguez and Leiva-Murillo [8]. By testing on synthetic data sets, we found that these algorithms only work well for relatively small datasets. This was confirmed in the case of a real world segmen-

tation problem where the subspace was followed by a classifier: the idea was to find a subspace that optimally segments Independently Moving Objects (MOs) from their backgrounds. As inputs we used several visual cues, all of which were obtained from WP4. This led to disappointing results. As an alternative, we developed a cue fusion paradigm based on a multilayer perceptron (MLP). This yielded much better results, so that we decided to continue with this paradigm for disambiguating IMOs (further reported on in D6.2).

References

The papers referred to are concerned with techniques and algorithms for subspace development, *i.e.*, the transformation of the input space onto a low-dimensional subspace. In our case, we look for a subspace that best separates SVEs from the background. At the time of the TA writing, Torkkola's feature selection (Torkkola, 2002) method seemed very promising. In the reference list the papers cited are briefly commented, as required by the reviewers.

[1] N. Chumerin, and M.M. Van Hulle, "Comparison of two feature extraction methods based on maximization of mutual information," in *IEEE Workshop on Machine Learning for Signal Processing* (Maynooth, Ireland, 6-8 September 2006), 2006, pp. 343-348.
Publication of part of the results reported here.

[2] J.E. Jackson, *A User's Guide to Principal Components*, Wiley, New York, 1991.
The standard work on PCA, an unsupervised method for developing subspaces.

[3] A. Hyvärinen, J. Karhunen, and E. Oja, *Independent component analysis*, John Wiley & Sons, 2001.
The standard work on ICA, another unsupervised method for developing subspaces.

[4] F.W. Young, "Multidimensional scaling: History, theory, and applications," R.M. Hamer, Ed. 1987, Hillsdale, NJ: Lawrence Erlbaum Associates.
A review on MDS, a non-linear subspace transformation technique.

[5] K. Torkkola and W. Campbell, "Mutual information in learning feature transformations.," in *ICML*, 2000, pp. 1015–1022.
The article on which this report is based.

[6] J.C. Principe, J.W. Fisher III, and D. Xu, "Information theoretic learning," in *Unsupervised Adaptive Filtering*, Simon Haykin, Ed., New York, 2000, Wiley.
This article describes an approximation of mutual information, on which Torkkola's is based.

[7] K.D. Bollacker and J. Ghosh, "Linear feature extractors based on mutual information," in *Proceedings of the 13th International Conference on Pattern Recognition*, 1996, vol. 2, pp. 720–724.
This article describes a method for subspace development based on the direct estimation of mutual information.

[8] A. Artés-Rodríguez and J. M. Leiva-Murillo, "Maximization of mutual information for supervised linear feature extraction," submitted.
This article offers an alternative method to Torkkola's.

[9] A. Hyvärinen, "New approximations of differential entropy for independent component analysis and projection pursuit," in *Advances in Neural Information Processing Systems*, Michael I. Jordan, Michael J. Kearns, and Sara A. Solla, Eds. 1998, vol. 10, pp. 273–279, The MIT Press.
This article describes how mutual information maximization can be done with negentropy maximization.

[10] E. Parzen, "On the estimation of a probability density function and mode," *Annals of Mathematical Statistics*, vol. 33, pp. 1065–1076, 1962.
The classical paper on density estimation with Gaussian kernels.

[11] A. Kraskov, H. Stögbauer, and P. Grassberger, "Estimating mutual information," Phys. Rev. E, 69, 066138, 2004.
A new distance-based method for estimating mutual information.

[12] K.E. Hild II, D. Erdogmus, K. Torkkola, and J.C. Principe, "Sequential feature extraction using information-theoretic learning," in press.
A paper that describes the publicly available implementation of Torkkola's approach