Finally, the results obtained with the proposed methodology MML, demonstrate that the computed low-dimensional space suitably exhibits the dynamic of a given activity, learning it from multiple datasets. Figures 4.2(c), 4.3(c), 4.4(c), and 4.5(a) show how this method learns the relationships among frames of videos related to a similar activity, unfolding the underlying data structure.

The low-dimensional representations found by MML reflects the activity dynamics and the soft correspondence among points of different datasets. Furthermore, our approach identifies a soft correspondence among videos even when they do not share a common similarity appearance, number of observations, and/or resolution. This can be explained because the relationships among samples of different datasets are computed based on an intra manifold comparison (equation (4-2)), the samples are not directly compared on the high-dimensional space, instead of that, the samples are compared by means of their own similarity representations, which is the similarity between a sample an each one of the other samples on the same manifold. The Figures 4.2(d), 4.3(d), 4.3(d), and 4.5(b) confirm it. For instance, Figure 4.5(b) shows the Matrix $\mathbf{A}$ for the last experiment with different amount of samples and input dimensions. And it is possible to observe that due to the conditions of the input data, there are different sub-blocks that represent each manifold and the relationships among them.

5. Motion Activity Recognition

Human motion analysis has emerged in last decade as an important area of research for different fields and applications. One of the most important fields is computer vision systems, due to promising applications such as visual surveillance, human performance analysis, computer-human interfaces (robotic interaction with humans), content-based image retrieval/storage and virtual reality. In particular, the ability to recognize human actions by visual information processing is essential to make that a machine interacts with determination and effort within a human-inhabited environment.

Specifically, the recognition of human actions from video streams has many applications in the surveillance, entertainment, user interfaces, sports, among others [6]. The main purpose of the action recognition is to assign a specific label for a movement either from an image or a video. Normally, the set of actions has a meaning in a certain domain. (e.g. in sign language, the set of actions corresponds to the set of possible words and letters that can be produced). There are two kinds of approaches to perform activity recognition: off-line or on-line, which depends on the requirements of the specific application. In the former former, off-line recognition, an entire video is available and the goal is to identify the type of motion recorded. In online recognition, the goal is to identify the motion type with only a portion of a video, while it is still in progress.

In this sense, the ability to generalize from a small training set is an important feature of any recognition system. Although, there has been much recent progress in the human action recognition field, presenting good results on standard datasets, this level of performance has been achieved with techniques that uses a large amount of training data [38, 39], which could be appropriate for certain tasks, but it is not suitable to assume that large training sets exist for discriminating between broadly different categories of actions [1]. Previous approaches found in the state of the art report good performance in the recognition of different human actions, within the more common are running and walking. However, some of them analyze data taken under constrained conditions [40, 41]. For example, the camera distance to the subject, lightning, angle, homogeneous background, etc, which affects the results and over-fit the systems to certain type of data.

Generally, there are two tightly related steps in building a motion recognition system, i.e., extracting motion features and training a classifier using these features. The most of the relevant work about motion recognition focuses on motion feature selection, including extracting features from 2-D tracking data [39], 3-D tracking information [42] or extracting motion information directly from images [40, 41, 43]. Particularly, in [43] is proposed a feature extracting algorithm called motion history histogram, which is an improvement over the motion history image in terms of encoding the time span of movement, and provided an FPGA (Field Programmable Gate Array) implementation. Moreover, there have been some works on using Support Vector Machines (SVM) to perform human motion recognition [40, 42, 44]. In [44], the motion representation strategy is based on local spatial - temporal features and uses k-means clustering algorithm to extract a set of primitive events, which are fed into an SVM. In [42] a 3-D spatial - temporal description of the motion is required as an input to an SVM. Nevertheless, to find a suitable value for the SVM parameters is a challenging task, normally performed by expert users. In this regard, it is necessary to develop simpler methodologies that works independently of the conditions of data, and with hight reliability to support human action recognition tasks.

In this work, we propose a framework that is applicable to both off-line and on-line recognition. Further, the proposed methodology involves a preprocessing stage that allows to work with video data independently of the record conditions, which improves the performance in the classification phase. The first stage of the methodology consist of a series of operations that allow to detect motion in the video sequences. After frames with no motion are removed from the videos, motion information is represented by a feature image, which is calculated using an infinite impulse response filter (IIR- filter). Then, the frames are resized and data is transform by a dimensionality reduction technique proposed in [45]. Finally, data is classified by three different schemes: Linear Discriminant Classifier (LDC), Quadratic Discriminant Classifier (QDC) and k-Nearest Neighbors Classifier (KNN). We tested the proposed methodology on a benchmark dataset, well known in the state of the art, the Action database (also called KTH database, [46]). We focus on experiments for action recognition, though the same principles could be applied to video retrieval and surveillance tasks.

Next Section (5.1) introduces the proposed scheme for human action recognition. Then, the experimental conditions and results are presented in Section 5.2, and finally in Section 5.3 is the discussion about the experiments.

5.1. Proposed Methodology

In order to analyze videos taken under real conditions, we proposed a methodology (presented in Figure **5-1**) that involves several stages, including preprocessing, nonlinear dimensionality reduction and classification. The main purpose is to be able to work with as few constrains as possible in the input data. There are previous approaches that aim to perform human recognition, and they report good results. However, it is common to find that those approaches work under several restrictions such as lighting control, a specific distance between the subject and the camera, cloth control, angle, among others [40, 41]. For that reason, we propose a methodology less sensitive to the mentioned parameters. In the following sections each stage is explained.

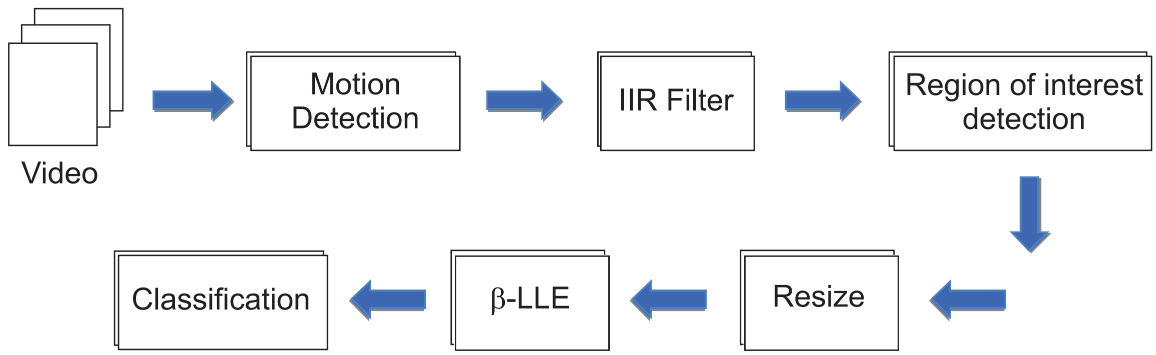


Figure 5-1.: Proposed methodology

5.1.1. Motion Detection

In practice, it possible to find videos that do not present activity or motion 100% of the time. Thence, preprocessing data in order to perform motion detection, becomes an essential step for any application (online or off-line). In Figure 5-2 we take some frames of a video of a person running. In this particular situation the subject performs the action in one direction, then no movement is present in the video, and finally the person runs in the opposite direction.

Figure 5-2.: Video of a person running

Let $\mathbf{X}_{n_v \times p}$ be the input data, with n_v is the number of analyzed videos and p is the input dimension. Then

$$\mathbf{X} = \begin{bmatrix} \mathbf{V}_1 \\ \mathbf{V}_2 \\ \vdots \\ \mathbf{V}_{n_v} \end{bmatrix} \tag{5-1}$$

where $\mathbf{V}_r \in \mathbb{R}^{n_{fr} \times p}$, n_{fr} is the number of frames of r-th video, $r \in \{1, 2, \ldots, n_v\}$, $p = h * w$ being h the number of rows pixels and w the number of column pixels for each frame in gray scale.

In image processing, it is desirable to be able to perform noise reduction. In this regard, the median filter is a nonlinear digital filtering technique often used for this purpose. For that reason, the first step in the motion detection stage is to apply a median filter to each frame in $\mathbf{X}$, in order to reduce noise and produce smoother images. Given a video $\widehat{\mathbf{V}}_r = \begin{bmatrix} \mathbf{v}_1 & \mathbf{v}_2 & \cdots & \mathbf{v}_{n_{fr}} \end{bmatrix}^\top$, where $\mathbf{v}_i \in \mathbb{R}^{1\times p}$ are the vectorized version of the filter frames, we propose to detect motion in each $\widehat{\mathbf{V}}_r$ by

$$\mathbf{d}_i = |\mathbf{v}_i - \mathbf{v}_{i-1}| \tag{5-2}$$

being $\mathbf{d}_i$ the derivative of $\mathbf{v}_i$, and $i = 1, 2, \ldots, n_{fr}$. This operation allows to detect significant changes between video frames, in order to eliminate parts of the video with minimum variability. As a result we obtain the derivative matrix $\mathbf{De} = \begin{bmatrix} \mathbf{d}_1 & \mathbf{d}_2 & \cdots & \mathbf{d}_{n_{fr}} \end{bmatrix}^\top$. Figure **5-3** shows the computed derivative in an example video of a person running.

Figure 5-3.: Derivative applied in a video sequence of a person running

After the derivative, a series of morphological operations are applied to $\mathbf{De}$, in order to facilitate the detection of constant frames, and to avoid consider noise as motion in the video, as follows

- Calculate the extended-maxima transform.
- Perform a morphological closing on the images.
- Finally, fill holes in the binary images.

In Figure **5-4**, is exhibit the effect of the above operations in the process, this is an example of the results to perform this procedure in the derivative presented in Figure **5-3**. As can be seen, in the frames where no movement is detected the result is a white image (constant frames).

Figure 5-4.: Derivative after the morphological operations

Afterward this procedure, the constant frames are identifying by performing a sum of the derivative value pixel by pixel. When the number of pixels with intensity 0 in a frame are more than an specific value (e.g. 90%) then is consider with no motion. In other case, is considered as motion. Once we know if there is motion and where is located the activity in the video sequence, then we proceed to extract the frames where no movement is detected. The result of remove these frames from the given video data is the matrix

$$\mathbf{Z}_r = \begin{bmatrix} \mathbf{T}_1 \\ \mathbf{T}_2 \\ \vdots \\ \mathbf{T}_{n_s}, \end{bmatrix} \tag{5-3}$$

where $\mathbf{T}_j \in \mathbb{R}^{n_{fT} \times p}$, being n_{fT} the amount of frames that have motion in constant sequence, $n_{fT} \in [n_{fT_{\min}}, n_{fT_{\max}}]$, $j \in \{1, 2, \ldots, n_s\}$, and n_s is the number of sub-sequences with motion detected in $\mathbf{V}_r$.

5.1.2. Recursive Filtering - IIR Filter

Recursive filtering is a good option to obtain information about the performed action directly from the frames of the recorded video. Recursive filters are also called Infinite Impulse Response (IIR) filters, since their impulse responses are composed of decaying exponentials. In this regard, the filter response acts like a measure that allows to identify recent motion, which is represented brighter than older motion in the image.

Recursive filters work by adding a fraction of a previous frame to the current image frame. In this way, they smooth the image sequence in time. The degree of filtering can be controlled by varying how much of the previous frame is added to the current frame. As the degree of filtering is increased, moving objects are also blurred in time. This can make moving objects appear to have a *ghost* or *comet-tail*, and the object will also lose contrast, which in this case we used as motion information at each frame. In this sense, at each frame, motion information is represented by a feature image [41].

Considering a subsequence $\mathbf{T}_j = \begin{bmatrix} \mathbf{t}_1 & \mathbf{t}_2 & \cdots & \mathbf{t}_{n_{fT}} \end{bmatrix}^\top$ found in a video $\mathbf{V}_r$, a weighted average at time a, is computed as

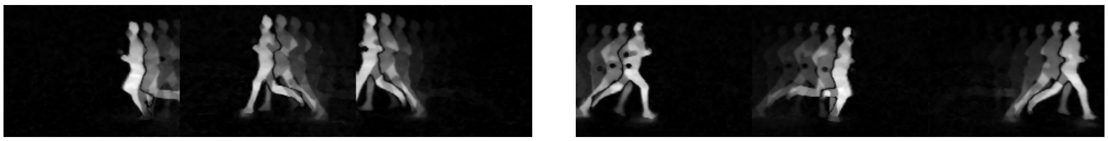
$$\boldsymbol{\rho}_a = \tau \mathbf{t}_{a-1} + (1-\tau)\,\boldsymbol{\rho}_{a-1}, \tag{5-4}$$

where $\boldsymbol{\rho}_a \in \mathbb{R}^{1\times p}$, $\mathbf{t}_a \in \mathbb{R}^{1\times p}$ is the image at time a, $a = 1, 2, \ldots, n_{fT}$, and τ is a scalar between 0 and 1. The feature image $\mathbf{f}_a$ is calculated by

$$\mathbf{f}_a = |\rho_a - \mathbf{t}_a|\,. \tag{5-5}$$

Then, for $\mathbf{T}_j$, the filter output is $\mathbf{F}_j = \begin{bmatrix} \mathbf{f}_1 & \mathbf{f}_2 & \cdots & \mathbf{f}_{n_{fT}} \end{bmatrix}^\top$, with $\mathbf{F}_j \in \mathbb{R}^{n_{fT}\times p}$. If $\tau = 1$ then $\mathbf{t}_a$ will be equal to the previous frame, if $\tau = 0$ then $\mathbf{t}_a$ will remain constant, and thence $\mathbf{F}_j$ will be equal to the foreground. The idea is that the feature image captures the temporal changes in the video sequence, and therefore, moving objects result in a fading trail behind them or blurring. Some characteristics are implicit in this representation: speed (spread of the trail) and direction (gradient of the region).

Figure 5-5 shows the output of the IIR filter in a sequence of a person running (after extracting no motion frames), as can be notice the recent motion is present in each frame, giving a blur of the person performing the activity.



(a) First detected motion (b) Second detected motion

Figure 5-5.: Extracted motions

Finally, in order to remove possible noise present in the frames after the IIR filter a median filter is applied to $\mathbf{F}_j$, and then a morphological filter is employed to facilitate the next stage. This morphological operation sets a pixel to 1 if five or more pixels in its 3-by-3 neighborhood are 1s; otherwise, it sets the pixel to 0 (considering a binary image). An example is shown in Figure 5-6. The outcome of the morphological filter is noted as $\mathbf{Fm}_j = \begin{bmatrix} \mathbf{fm}_1 & \mathbf{fm}_2 & \cdots & \mathbf{fm}_{n_{fT}} \end{bmatrix}^\top$

(a) First detected motion (b) Second detected motion

Figure 5-6.: Result after morphological filter

It is important to emphasize that the main advantage of this technique (Recursive filtering) is that is suitable for real-time applications.

5.1.3. Region of Interest Detection

At this stage of the proposed methodology we already obtained data with motion and the IIR filter was applied. Thence, to identify in further stages what kind of activity is generating the motion, a region of interest is found. For instance, if a person is waving his hands, the region of interest should be located around the hands and arms, which are the ones that produce changes between frames due to movement.

Considering the information obtained from the last stage $\mathbf{Fm}_j$, the idea is to find the height and width of the subjects by calculating their position in each image. For this purpose, we look for changes in pixel intensity to determine the first pixel in both rows and columns in which begin the silhouette of the subject as in Figure5-7.

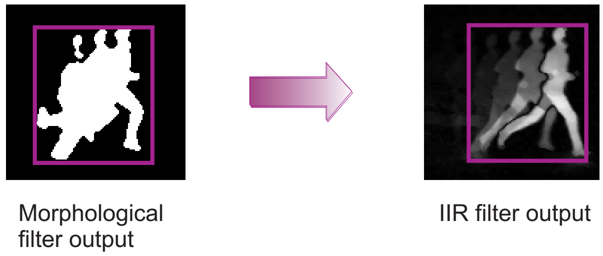
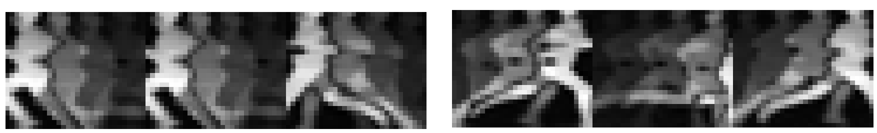


Figure 5-7.: Derivative after the morphological operations

5.1.4. Resize

The goal of this stage is to set a standard resolution for all the video frames. In this regard, the chosen resolution was based on state of the art. Thus, the images were resized so that they have an specific number of rows and columns, in particular each frame is resize to $h_r \times w_r$ pixel resolution. An example is shown in Figure **5-8**



(a) (b)

Figure 5-8.: Example of the resized frames

The resized process is performed using a bicubic interpolation and antialiasing. Thus, the output pixel value is a weighted average of pixels in the nearest 4-by-4 neighborhood, and an antialiasing is performed when shrinking the images.

The resize operation is executed on $\mathbf{F}_j$, but considering the region of interest found based on $\mathbf{Fm}_j$. Thence, the obtained output is $\tilde{\mathbf{F}}_j \in \mathbb{R}^{n_{fT} \times p_r}$, being $p_r = h_r w_r$, h_r is the resized height and w_r is the resized width.

5.1.5. Class Label Locally Linear Embedding – β-LLE

An extension of the LLE approach to deal with several manifolds (patterns) is presented in [45], which employs class labels as extra information to guide the procedure of dimensionality reduction allowing to figure out a suitable representation for each one of them. In this way, this approach (that we called β Locally Linear Embedding – β-LLE) aims to construct a NFE algorithm that preserves the local geometry of the data, providing a discriminative strategy during the embedding procedure. The conventional LLE algorithm is reformulated changing the cost function (3-10) as

$$\min_{\mathbf{Y}} \Psi\left(\mathbf{Y}, \beta\right) = \min_{\mathbf{Y}} \left\{ \sum_{i=1}^{n} \left\| \mathbf{y}_i - \sum_{j=1}^{n} w_{ij}\mathbf{y}_j \right\|^2 - \beta \sum_{i=1}^{n} \left\| \mathbf{y}_i - \sum_{j=1}^{n} \gamma_{ij}\mathbf{y}_j \right\|^2 \right\}, \tag{5-6}$$

with $\beta \in \Re^{+}$, and subject to $\sum_{i=1}^{n} \mathbf{y}_i = \mathbf{0}$ and $\sum_{i=1}^{n} \mathbf{y}_i\mathbf{y}_i^{\top}/n = \mathbf{I}_{m\times m}$. Furthermore

$$\gamma_{ij} = \begin{cases} 0, \text{ if } i = j \\ \frac{1}{n-1}, \text{ if } \mathcal{P}\left(\mathbf{y}_i\right) \neq \mathcal{P}\left(\mathbf{y}_j\right) \\ -\frac{1}{n-1}, \text{ if } \mathcal{P}\left(\mathbf{y}_i\right) = \mathcal{P}\left(\mathbf{y}_j\right) \end{cases}, \tag{5-7}$$

being $\mathcal{P}\left(\cdot\right)$ a function that determines the class label of the objects, and β is a tradeoff between the preservation of the local geometry of the high dimensional data and the representation induced by the class labels [45]. For solving the minimization problem, it is possible to rewrite (5-6) as

$$\min_{\mathbf{Y}} \Psi\left(\mathbf{Y}, \beta\right) = \min_{\mathbf{Y}} \left\{ \mathrm{tr}\left(\mathbf{Y}^{\top}\left(\mathbf{M} - \beta\tilde{\mathbf{M}} \right) \mathbf{Y} \right) \right\} \quad \text{s.t.} \quad \begin{cases} \mathbf{1}_{1\times n}\mathbf{Y} = \mathbf{0}_{1\times n} \\ \frac{1}{n}\mathbf{Y}^{\top}\mathbf{Y} = \mathbf{I}_{m\times m} \end{cases} \tag{5-8}$$

where $\mathbf{M} = \left(\mathbf{I}_{n\times n} - \mathbf{W}^{\top}\right)\left(\mathbf{I}_{n\times n} - \mathbf{W}\right)$ and

$$\tilde{\mathbf{M}} = \left(\mathbf{I}_{n\times n} - \Gamma^{\top}\right)\left(\mathbf{I}_{n\times n} - \Gamma\right), \tag{5-9}$$

being $\Gamma \in \Re^{n\times n}$ a matrix whose elements γ_{ij} are computed as in (5-7). Thence, it is possible to calculate the m eigenvectors of $\mathbf{M} - \beta\tilde{\mathbf{M}}$, which are associated to the m smallest eigenvalues after discarding the eigenvector related to some eigenvalue equal or close to zero.

Note that the β parameter in (5-6) is a tradeoff between the reconstruction error and the margin between objects belonging to different classes. If $\beta = 0$, we have the original mapping of LLE, and as β increases the separation between classes is larger. For a given β it is possible to find the output $\mathbf{Y}_{\beta}$ that minimizes the cost function (5-6) [45]. Next, the reconstruction error e_R and the margin μ can be also computed as function of $\mathbf{Y}_{\beta}$ for each value of β, as in equation (5-10)

$$\begin{aligned} e_R\left(\beta\right) &= \mathrm{tr}\left(\mathbf{Y}_{\beta}^{\top}\mathbf{M}\mathbf{Y}_{\beta}\right) \\ \mu\left(\beta\right) &= \mathrm{tr}\left(\mathbf{Y}_{\beta}^{\top}\tilde{\mathbf{M}}\mathbf{Y}_{\beta}\right) \end{aligned}. \tag{5-10}$$

Looking forward the simultaneous minimization of the reconstruction error $e_R\left(\beta\right)$ and the maximization of the margin $\mu\left(\beta\right)$, in [45] the parametric plot $e_R\left(\beta\right)$ versus $\mu\left(\beta\right)$ is used as a tool to study the behavior of these quantities [47, 48]. Algorithm 5 describes the β-LLE method.

Algorithm 5 – β Locally Linear Embedding – β-LLE

Require: data matrix in this case $\tilde{\mathbf{F}}$, number of neighbors k, output dimension m, set of β values Ξ.

1: Find the k nearest neighbors for each point.
2: Compute the weight matrix $\mathbf{W}$ by minimizing (3-3).
3: For each $\beta \in \Xi$ compute the matrix $\tilde{\mathbf{M}}$ as in (5-9).
4: Compute the output matrix $\mathbf{Y}_\beta$ for each $\tilde{\mathbf{M}}$ by minimizing (5-8) as in original LLE.
5: Employ the parametric plot $e_R(\beta)$ versus $\mu(\beta)$ using (5-10) to find $\mathbf{Y}$.

5.2. Experimental Set Up and Results

In order to validate the proposed methodology for activity recognition we compare against different conditions and techniques. In the first place, we apply the preprocessing stage but we do not perform dimensionality reduction. Then, based on the scheme described in Figure **5-1**, we use three dimensionality reduction techniques besides β - LLE: The traditional linear method PCA and two nonlinear algorithms studied in Chapter 3, LLE and LEM.

Now, we test the above mentioned feature extraction methods on a real world database, the Action dataset, which is a benchmark dataset in the state of the art [46]. This database is conformed by six kinds of human actions (walking, jogging, running, boxing, handwaving and handclapping) performed several times by 25 subjects in four different scenarios: outdoors, outdoors with scale variation, outdoors with different clothes and indoors as illustrated in Fig **5-9**. All sequences were taken with a static camera with 25fps frame rate. The sequences were down-sampled to 120×160 ($p = 19200$) pixels and have a length of four seconds in average. For testing, the videos with scale variation (zoom) and/or noticeable shadows were discarded. Besides, we randomly choose 30 videos ($n_v = 30$), 5 videos for each one of the six activities (minimum one for subject).

Figure 5-9.: Examples from Action database

The generalization abilities for the provided experimental conditions are tested by a 10 folds cross-validation scheme. In regard to the early stages of the proposed methodology the parameters were set as follows: the median filter applied in the motion detection stage was performed using a 3-by-3 neighborhood. A morphological filter disc type was used. Then, for the motion detection a minimum window of 10 and maximum 20 frames was established to extract de parts of the video with no activity. The frames are removed if the 90% (or more) of the image pixels are 0, otherwise is considered as motion. Now, the parameter τ in the IIR filter is set to 0.5. As we mentioned in section 5.1.4, the final frame size was set to 25×30.

After that process, 141 videos (sub-video sequences found in the motion detection stage) were employed for training and testing. We obtained an space $\tilde{\mathbf{F}}$ with $n_{fT} = 141$, $p = 750$ and $C = 6$, considering the resized process. In Table **5-1** are presented the specific amount of videos for each activity, and the amount of frames.

Characteristic	**Activity**						
	Walking	Jogging	Running	Boxing	Hand waving	Hand clapping	Total
Number of videos	25	21	20	25	25	25	141
Amount of frames	516	441	349	489	492	490	2777

Table **5-1**.: Characteristics of $\tilde{\mathbf{F}}$

From Table **5-1**, we can infer that there are two ways to analyze $\tilde{\mathbf{F}}$, according to the amount of videos $\tilde{\mathbf{F}}_{141\times750}$ and according to the amount of frames $\tilde{\mathbf{F}}_{2777\times750}$. In this line of analysis, we perform two strategies: video classification and image classification. In the former strategy, the label for each video is chosen as the mode (majority voting) of the label found for the frames of the video.

The number of nearest neighbors for the NFE methods is chosen using a new approach [31], which computes an specific number of neighbors for each input object, a brief description of this methodology is presented in Appendix A. The dimension of the embedding space is fixed looking for a 95% of expected local variability (see Appendix A.2), leading in an output dimension of $m = 4$.

Three classifiers are tested: linear discriminant classifier (LDC), quadratic discriminant classifier (QDC) and k-nearest neighbors classifier (KNNC). The number of neighbors for this classifier is optimized with respect to the leave-one-out error of the training set.

In figure **5-10** is show the procedure to analyze new samples after training the system.

5.2.1. Results

Table **5-2** and **5-3** present the classification accuracy, standard deviation and confidence interval for the video classification and image classification strategies. In addition, Table **5-4** to **5-33** shows the confusion matrices for each classifier with each one of the conditions

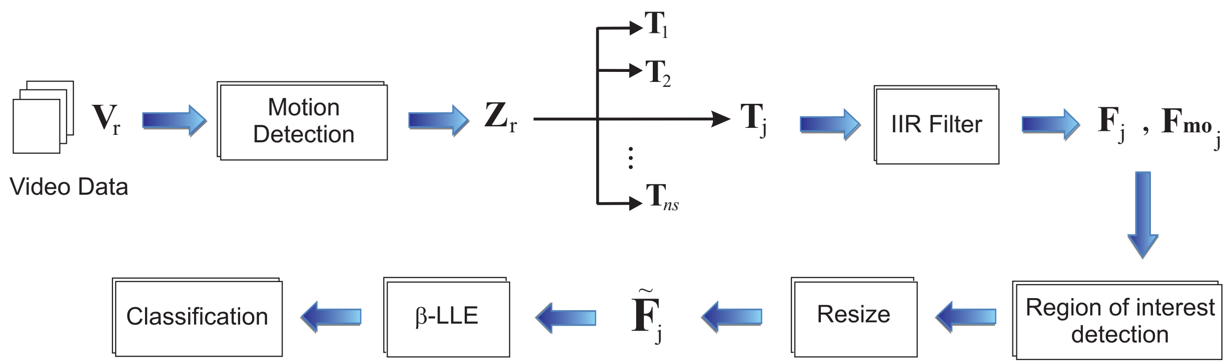


Figure 5-10.: Scheme used for new samples

mentioned in the experimental set up (No dimensionality reduction, PCA, LLE, LEM and β LLE).

Classifier	Classification Accuracy ± Standard Deviation				
	Without DR	PCA	LLE	LEM	β LLE
LDC	48.7473 ± 11.3255	50.3883 ± 7.7237	52.5238 ± 13.7744	33.7573 ± 16.0388	85.8832 ± 5.3789
	CI = $[40.64, 56.84]$	CI = $[44.86, 55.91]$	CI = $[42.67, 62.37]$	CI = $[22.28, 45.22]$	CI = $[82.03, 89.730]$
QDC	18.4725 ± 3.7750	63.5788 ± 10.4543	60.1575 ± 14.3836	50.9240 ± 10.5782	90.1360 ± 5.4782
	CI = $[15.77, 21.17]$	CI = $[56.10, 71.05]$	CI = $[49.86, 70.44]$	CI = $[43.35, 58.49]$	CI = $[86.21, 94.05]$
KNN	90.8022 ± 9.5041	87.4615 ± 8.2779	75.7106 ± 7.4377	73.4840 ± 9.9066	91.8246 ± 8.0512
	CI = $[84.00, 97.60]$	CI = $[81.53, 93.38]$	CI = $[70.45, 80.96]$	CI = $[66.39, 80.57]$	CI = $[86.06, 97.58]$

Table **5-2**.: Classification Accuracy and Confidence Interval – Video Classification

Data Sets	Classification Accuracy ± Standard Deviation				
	Without DR	PCA	LLE	LEM	β LLE
LDC	37.9872 ± 6.7339	37.5822 ± 4.6363	45.3742 ± 6.7571	29.2643 ± 11.0522	72.6052 ± 4.5875
	CI = $[33.17, 42.80]$	CI = $[34.26, 40.89]$	CI = $[40.54, 50.20]$	CI = $[21.35, 37.17]$	CI = $[69.32, 75.88]$
QDC	19.5183 ± 4.2963	49.5920 ± 4.7459	40.3189 ± 5.9758	40.9155 ± 6.0277	74.3620 ± 4.4348
	CI = $[16.44, 22.59]$	CI = $[46.19, 52.98]$	CI = $[45.04, 53.59]$	CI = $[36.60, 45.22]$	CI = $[71.18, 77.53]$
KNN	73.0324 ± 5.2134	63.2293 ± 3.4650	54.9953 ± 3.9397	50.3244 ± 4.6350	75.4449 ± 4.1844
	CI = $[69.30, 76.76]$	CI = $[60.75, 65.70]$	CI = $[52.17, 57.81]$	CI = $[47.00, 53.64]$	CI = $[72.45, 78.43]$

Table **5-3**.: Classification Accuracy and Confidence Interval – Image Classification Accuracy

5.2.2. Confusion Matrices

	Walk	Jog	Run	Box	Hwav	Hclp
Walk	96.66	18.33	10.00	83.33	38.33	90.00
Jog	3.33	66.66	35.00	0.00	0.00	0.00
Run	0.00	5.00	55.00	0.00	0.00	3.33
Box	0.00	0.00	0.00	8.33	0.00	0.00
Hwav	0.00	5.00	0.00	8.33	61.66	0.00
Hclp	0.00	5.00	0.00	0.00	0.00	6.66

Table **5-4**.: Without DR + LDC - Video Average Classification Accuracy

	Walk	Jog	Run	Box	Hwav	Hclp
Walk	40.00	40.00	40.00	30.00	23.33	31.66
Jog	10.00	10.00	10.00	20.00	6.66	18.33
Run	0.00	0.00	0.00	3.33	11.66	0.00
Box	20.00	20.00	20.00	20.00	23.33	20.00
Hwav	20.00	20.00	20.00	13.33	20.00	20.00
Hclp	10.00	10.00	10.00	13.33	15.00	10.00

Table **5-5**.: Without DR + QDC - Video Average Classification Accuracy

	Walk	Jog	Run	Box	Hwav	Hclp
Walk	100.00	0.00	0.00	0.00	0.00	0.00
Jog	0.00	75.00	5.00	11.66	5.00	0.00
Run	0.00	25.00	95.00	10.00	0.00	0.00
Box	0.00	0.00	0.00	78.33	0.00	0.00
Hwav	0.00	0.00	0.00	0.00	95.00	5.00
Hclp	0.00	0.00	0.00	0.00	0.00	95.00

Table **5-6**.: Without DR + KNN - Video Average Classification Accuracy

	Walk	Jog	Run	Box	Hwav	Hclp
Walk	67.17	17.05	12.96	65.89	33.38	48.95
Jog	13.56	39.86	33.42	5.61	5.35	5.90
Run	7.03	26.01	40.60	3.70	3.64	5.34
Box	5.46	4.52	4.10	15.41	4.21	4.33
Hwav	2.13	5.99	1.99	5.91	41.35	12.53
Hclp	4.63	6.54	6.91	3.45	12.04	22.93

Table **5-7**.: Without DR + LDC - Image Average Classification

	Walk	Jog	Run	Box	Hwav	Hclp
Walk	39.66	40.00	40.00	26.02	23.31	30.33
Jog	10.33	10.00	10.00	20.37	13.57	19.23
Run	0.00	0.00	0.00	7.09	6.41	1.33
Box	20.00	20.00	20.00	20.00	23.33	20.00
Hwav	20.00	20.00	20.00	13.75	20.00	17.83
Hclp	10.00	10.00	10.00	12.5	14.32	11.26

Table **5-8**.: Without DR + QDC- Image Average Classification Accuracy

	Walk	Jog	Run	Box	Hwav	Hclp
Walk	78.23	2.28	3.99	6.83	0.00	0.00
Jog	11.10	60.91	18.63	13.68	3.84	4.57
Run	10.67	35.18	77.10	14.56	2.56	2.65
Box	0.00	1.41	0.25	54.30	1.34	3.17
Hwav	0.00	0.20	0.00	7.22	89.50	14.45
Hclp	0.00	0.00	0.00	3.37	2.74	75.14

Table **5-9**.: Without DR + KNN - Image Average Classification Accuracy

	Walk	Jog	Run	Box	Hwav	Hclp
Walk	86.66	10.00	0.00	18.33	0.00	11.66
Jog	10.00	66.66	65.00	11.66	0.00	23.33
Run	0.00	0.00	0.00	0.00	0.00	0.00
Box	3.33	13.33	0.00	61.66	15.00	0.00
Hwav	0.00	0.00	0.00	0.00	75.00	65.00
Hclp	0.00	10.00	35.00	8.33	10.00	0.00

Table **5-10**.: PCA + LDC - Video Average Classification Accuracy

	Walk	Jog	Run	Box	Hwav	Hclp
Walk	83.33	10.00	5.00	8.33	0.00	0.00
Jog	5.00	31.66	25.00	0.00	0.00	0.00
Run	8.33	35.00	70.00	0.00	0.00	0.00
Box	3.33	13.33	0.00	86.66	0.00	8.33
Hwav	0.00	10.00	0.00	0.00	90.00	78.33
Hclp	0.00	0.00	0.00	5.00	10.00	13.33

Table **5-11**.: PCA + QDC - Video Average Classification Accuracy

	Walk	Jog	Run	Box	Hwav	Hclp
Walk	100.00	10.00	0.00	0.00	0.00	0.00
Jog	0.00	55.00	10.00	5.00	0.00	0.00
Run	0.00	35.00	90.00	3.33	0.00	0.00
Box	0.00	0.00	0.00	91.66	0.00	0.00
Hwav	0.00	0.00	0.00	0.00	100.00	21.66
Hclp	0.00	0.00	0.00	0.00	0.00	78.33

Table **5-12**.: PCA + KNN - Video Average Classification Accuracy

	Walk	Jog	Run	Box	Hwav	Hclp
Walk	61.54	11.73	13.16	21.00	8.79	18.02
Jog	16.62	51.22	50.00	19.82	5.22	18.36
Run	1.31	3.38	1.04	0.00	0.83	0.16
Box	9.65	14.22	9.20	44.59	15.96	14.65
Hwav	6.17	6.85	4.67	3.09	50.73	41.88
Hclp	4.68	12.57	21.90	11.48	18.44	6.89

Table **5-13**.: PCA + LDC - Image Average Classification Accuracy

	Walk	Jog	Run	Box	Hwav	Hclp
Walk	65.85	14.24	12.75	13.31	2.44	5.50
Jog	10.57	26.17	21.00	13.18	1.05	5.08
Run	13.37	33.19	54.98	2.55	1.25	3.14
Box	7.10	16.85	7.39	58.19	6.26	17.36
Hwav	0.00	5.06	1.46	3.96	63.53	43.42
Hclp	3.10	4.45	2.40	8.79	25.45	25.46

Table **5-14**.: PCA + QDC - Image Average Classification Accuracy

	Walk	Jog	Run	Box	Hwav	Hclp
Walk	83.41	15.42	15.03	8.93	2.33	5.08
Jog	5.64	40.37	21.03	8.26	1.23	4.25
Run	7.11	29.74	59.79	5.92	0.16	0.66
Box	1.88	8.37	3.54	60.05	6.02	11.86
Hwav	0.14	0.20	0.00	9.70	77.73	25.13
Hclp	1.78	5.88	0.58	7.10	12.49	52.98

Table **5-15**.: PCA + KNN - Image Average Classification Accuracy

	Walk	Jog	Run	Box	Hwav	Hclp
Walk	91.66	53.33	40.00	33.33	0.00	0.00
Jog	8.33	31.66	60.00	3.33	0.00	5.00
Run	0.00	0.00	0.00	0.00	0.00	0.00
Box	0.00	10.00	0.00	56.66	5.00	13.33
Hwav	0.00	0.00	0.00	6.66	75.00	40.00
Hclp	0.00	5.00	0.00	0.00	20.00	41.66

Table **5-16**.: LLE + LDC - Video Average Classification Accuracy

	Walk	Jog	Run	Box	Hwav	Hclp
Walk	96.66	43.33	45.00	5.00	0.00	0.00
Jog	0.00	15.00	20.00	8.33	0.00	8.33
Run	0.00	31.66	35.00	0.00	0.00	0.00
Box	0.00	0.00	0.00	76.66	0.00	10.00
Hwav	0.00	0.00	0.00	3.33	70.00	26.66
Hclp	3.33	10.00	0.00	6.66	30.00	55.00

Table **5-17**.: LLE + QDC - Video Average Classification Accuracy

	Walk	Jog	Run	Box	Hwav	Hclp
Walk	100.00	18.33	15.00	0.00	0.00	0.00
Jog	0.00	28.33	45.00	0.00	0.00	0.00
Run	0.00	38.33	40.00	0.00	0.00	0.00
Box	0.00	10.00	0.00	96.66	5.00	5.00
Hwav	0.00	0.00	0.00	0.00	81.66	3.33
Hclp	0.00	5.00	0.00	3.33	13.33	91.66

Table **5-18**.: LLE + KNN - Video Average Classification Accuracy

	Walk	Jog	Run	Box	Hwav	Hclp
Walk	80.55	44.43	38.46	25.12	1.08	3.48
Jog	12.95	31.02	50.14	11.77	3.33	10.32
Run	0.63	3.63	1.95	0.35	0.00	0.16
Box	3.11	10.31	3.62	45.85	12.89	18.31
Hwav	0.00	0.40	0.25	9.64	58.09	29.50
Hclp	2.74	10.19	5.55	7.23	24.59	38.20

Table **5-19**.: LLE + LDC - Image Average Classification Accuracy

	Walk	Jog	Run	Box	Hwav	Hclp
Walk	83.64	36.43	37.03	11.81	0.79	1.98
Jog	10.04	16.62	20.82	12.14	3.34	6.23
Run	4.32	31.10	36.64	3.22	0.66	0.48
Box	0.65	6.12	1.28	49.22	8.20	16.74
Hwav	0.00	0.90	0.73	5.04	56.04	29.34
Hclp	1.33	8.79	3.47	18.55	30.93	45.20

Table **5-20**.: LLE + QDC - Image Average Classification Accuracy

	Walk	Jog	Run	Box	Hwav	Hclp
Walk	79.64	22.10	20.65	7.02	0.25	1.00
Jog	11.84	30.13	35.17	9.58	2.03	4.07
Run	8.77	29.20	35.74	4.31	0.89	0.25
Box	4.74	12.97	6.62	58.77	9.10	16.88
Hwav	0.00	1.28	0.53	7.12	67.20	22.05
Hclp	0.00	4.30	1.26	13.17	20.50	55.73

Table **5-21**.: LLE + KNN - Image Average Classification Accuracy

	Walk	Jog	Run	Box	Hwav	Hclp
Walk	75.00	70.00	60.00	53.33	23.33	31.66
Jog	3.33	0.00	15.00	0.00	0.00	0.00
Run	0.00	5.00	5.00	0.00	0.00	0.00
Box	8.33	8.33	10.00	36.66	16.66	0.00
Hwav	0.00	0.00	0.00	0.00	18.33	11.66
Hclp	13.33	16.66	10.00	10.00	41.66	56.66

Table **5-22**.: LEM + LDC - Video Average Classification Accuracy

	Walk	Jog	Run	Box	Hwav	Hclp
Walk	16.66	10.00	0.00	0.00	6.66	0.00
Jog	30.00	15.00	20.00	20.00	0.00	0.00
Run	53.33	63.33	80.00	5.00	0.00	0.00
Box	0.00	11.66	0.00	75.00	8.33	0.00
Hwav	0.00	0.00	0.00	0.00	25.00	10.00
Hclp	0.00	0.00	0.00	0.00	60.00	90.00

Table **5-23**.: LEM + QDC - Video Average Classification Accuracy

	Walk	Jog	Run	Box	Hwav	Hclp
Walk	95.00	15.00	15.00	8.33	0.00	0.00
Jog	0.00	66.66	55.00	5.00	0.00	0.00
Run	0.00	18.33	30.00	0.00	0.00	0.00
Box	5.00	0.00	0.00	83.33	0.00	0.00
Hwav	0.00	0.00	0.00	0.00	78.33	18.33
Hclp	0.00	0.00	0.00	3.33	21.66	81.66

Table **5-24**.: LEM + KNN - Video Average Classification Accuracy

	Walk	Jog	Run	Box	Hwav	Hclp
Walk	62.19	60.01	58.78	46.30	18.17	28.19
Jog	9.43	4.51	12.28	2.16	0.33	1.33
Run	2.22	3.61	3.93	0.00	0.00	0.00
Box	15.83	15.02	15.77	31.89	17.73	17.01
Hwav	0.00	0.00	0.00	0.00	23.13	9.79
Hclp	10.31	16.82	9.21	19.63	40.62	43.48

Table **5-25**.: LEM + LDC - Image Average Classification Accuracy

	Walk	Jog	Run	Box	Hwav	Hclp
Walk	26.94	5.67	3.52	5.90	6.70	7.83
Jog	23.84	19.87	23.71	20.25	2.18	5.38
Run	40.50	57.96	66.90	10.68	0.43	2.83
Box	5.79	15.51	3.80	54.36	13.07	16.46
Hwav	0.72	0.00	0.29	1.28	24.90	8.26
Hclp	2.19	0.98	1.75	7.51	52.69	59.20

Table **5-26**.: LEM + QDC - Image Average Classification Accuracy

	Walk	Jog	Run	Box	Hwav	Hclp
Walk	56.56	17.03	16.46	14.54	1.76	1.72
Jog	15.33	36.16	39.80	12.00	2.74	5.31
Run	13.67	28.90	31.08	3.56	0.16	0.95
Box	13.07	14.52	10.09	58.77	7.48	11.42
Hwav	0.54	1.07	0.28	7.29	65.02	29.48
Hclp	0.80	2.29	2.25	6.75	22.84	51.09

Table **5-27**.: LEM + KNN - Image Average Classification Accuracy

	Walk	Jog	Run	Box	Hwav	Hclp
Walk	95.00	0.00	0.00	0.00	0.00	0.00
Jog	0.00	48.33	35.00	3.33	0.00	0.00
Run	5.00	51.66	65.00	0.00	0.00	0.00
Box	0.00	0.00	0.00	96.66	0.00	0.00
Hwav	0.00	0.00	0.00	0.00	100.00	0.00
Hclp	0.00	0.00	0.00	0.00	0.00	100.00

Table **5-28**.: β-LLE + LDC - Video Average Classification Accuracy

	Walk	Jog	Run	Box	Hwav	Hclp
Walk	95.00	0.00	0.00	0.00	0.00	0.00
Jog	5.00	60.00	20.00	0.00	0.00	0.00
Run	0.00	40.00	80.00	0.00	0.00	0.00
Box	0.00	0.00	0.00	100.00	0.00	0.00
Hwav	0.00	0.00	0.00	0.00	96.66	0.00
Hclp	0.00	0.00	0.00	0.00	3.33	100.00

Table **5-29**.: β-LLE + QDC - Video Average Classification Accuracy

	Walk	Jog	Run	Box	Hwav	Hclp
Walk	100.00	0.00	0.00	0.00	0.00	0.00
Jog	0.00	65.00	20.00	3.33	0.00	0.00
Run	0.00	35.00	80.00	0.00	0.00	0.00
Box	0.00	0.00	0.00	96.66	0.00	0.00
Hwav	0.00	0.00	0.00	0.00	100.00	0.00
Hclp	0.00	0.00	0.00	0.00	0.00	100.00

Table **5-30**.: β-LLE + KNN - Video Average Classification Accuracy

	Walk	Jog	Run	Box	Hwav	Hclp
Walk	83.40	4.83	2.05	3.81	0.00	0.16
Jog	3.26	45.69	37.21	6.05	0.81	1.50
Run	13.07	48.19	60.43	12.79	3.57	3.11
Box	0.25	0.86	0.00	68.43	1.30	4.23
Hwav	0.00	0.00	0.00	3.57	89.74	8.89
Hclp	0.00	0.37	0.29	5.32	4.55	82.06

Table **5-31**.: β-LLE + LDC - Image Average Classification Accuracy

	Walk	Jog	Run	Box	Hwav	Hclp
Walk	85.74	9.92	7.91	2.89	0.00	0.00
Jog	5.97	44.09	28.16	6.11	1.23	0.91
Run	6.81	37.66	61.64	3.37	0.39	0.00
Box	1.46	7.59	1.97	77.67	3.01	6.56
Hwav	0.00	0.00	0.00	1.08	84.86	6.72
Hclp	0.00	0.72	0.29	8.85	10.49	85.79

Table **5-32**.: β-LLE + QDC - Image Average Classification Accuracy

	Walk	Jog	Run	Box	Hwav	Hclp
Walk	88.08	9.02	6.12	4.10	0.25	0.16
Jog	5.82	48.71	29.79	5.87	2.20	0.52
Run	5.08	36.75	62.31	4.70	0.35	0.75
Box	1.00	4.38	1.47	74.40	1.87	4.15
Hwav	0.00	0.21	0.00	5.00	88.06	9.39
Hclp	0.00	0.89	0.29	5.91	7.25	84.99

Table **5-33**.: β-LLE + KNN - Image Average Classification Accuracy