People and mobile robot classification through spatio-temporal analysis of optical flow

Plinio Moreno and Dario Figueira and Alexandre Bernardino and José Santos-Victor

Institute for Systems and Robotics (ISR/IST)
LARSyS, Instituto Superior Técnico
Universidade de Lisboa
Lisboa, Portugal
{*plinio, dfigueira, alex, jasv*}*@isr.tecnico.ulisboa.pt*

The goal of this work is to distinguish between humans and robots in a mixed human-robot environment. We analyze the spatio-temporal patterns of optical flow-based features along several frames. We consider the Histogram of Optical Flow (HOF) and the Motion Boundary Histogram (MBH) features, which have shown good results on people detection. The spatio-temporal patterns are composed by groups of feature components that have similar values on previous frames. The groups of features are fed into the FuzzyBoost algorithm, which at each round selects the spatio-temporal pattern (i.e. feature set) having the lowest classification error. The search for patterns is guided by grouping feature dimensions, considering three algorithms: (a) similarity of weights from dimensionality reduction matrices, (b) Boost Feature Subset Selection (BFSS) and (c) Sequential Floating Feature Selection (SFSS), which avoid the brute force approach. The similarity weights are computed by the Multiple Metric Learning for large Margin Nearest Neighbor (MMLMNN), a linear dimensionality algorithm that provides a type of Mahalanobis metric [44]. The experiments show that FuzzyBoost brings good generalization properties, better than the GentleBoost, the Support Vector Machines (SVM) with linear kernels and SVM with Radial Basis Function (RBF) kernels. The classifier was implemented and tested in a real-time, multi-camera dynamic setting.

Keywords: Human robot environments; boosting algorithms; people vs. robot detection; optical-flow based features

1. Introduction

Current trends in robotics research envisage the application of robots within public environments helping humans in their daily tasks. Furthermore, for security and surveillance purposes, many buildings and urban areas are being equipped with extended networks of surveillance cameras. The joint use of fixed camera networks together with robots in social environments is likely to be widespread in future applications. This is the case of urban pedestrian areas in big cities, which are growing in Europe because of mobility and quality-of-life issues.

The pioneer work of the URUS project [36] addressed the implementation of a sensor network with robots that interact with people in urban public areas. A key element of the project is a monitoring and surveillance system composed by a network of fixed cameras that provide information about the human and robot

activities. In the URUS project, the communication framework sends and receives messages between the agents (robots and vision-based algorithms) over a network of wireless routers, ensuring acknowledgment of sucess/failure. The goal of the agents is to make decisions that respond to requests by the people and emergency messages sent by the sensors. For pedestrians, the system was required to pay attention and discern gestures from each person, to be able to detect when someone was in trouble or purposely calling the attention of the system by waving to one of the cameras. Robot detections on the other hand required communication attempts with the robot, to provide it with accurate position information given by the camera, to help the robot improve its self-localization. In addition, the classification of pedestrian and robots will remove false pedestrians and false robots detections that generate false attention signals and uncertainty in the position of the robots. The goal of the MAIS-S project[1] was to add robustness and study scalability issues to the decision making in a similar setup to the URUS project. In particular, if a camera fails the robot should go to the location and "replace" temporarily the fixed camera. Another example is the "robot guard", where the robot is required to go and provide surveillance of an area currently sparsely covered by the human guards. In this context, the ability to discriminate between robots and people was also essential to the successful outcome of the project.

But these multi-camera applications must also consider constraints such as real-time performance and low-resolution images due to limitations on communication bandwidth. Thus, it is fundamental to be able to discriminate between people and mobile robots using low resolution video and efficient features.

1.1. *Our approach*

In this work we address the unexplored issue of discrimination between people and robots, which is essential to the development of surveillance techniques for mixed human-robot environments. The need and benefit of automatic person-robot discrimination is particularly evident in the URUS project [36], where the people vs. robot classification provides the input for other problems. On one hand, detected humans are fed to gesture detection and classification. On the other hand, detected robots are fed to self-localization algorithms in order to improve their pose estimation.

People are notorious in their appearance variability given their infinite clothes possibilities and quite articulate possible poses (as can be seen in Figure 1). Robots even more so, given that there are robots of every size, shape and color, and more keep coming every year. In the URUS project alone we experienced a multitude of different robots, as illustrated in Figure 2.

One thing that distinguishes the two is their motion patterns. If we restrict ourselves to rigid mobile-platform robots, that mostly only have one direction of

[1] http://gaips.inesc-id.pt/mais-s/

Fig. 1. Illustration of person appearance variability. Depending on the clothes, facial features, hair style and configuration of the limbs, appearance-based models for people detection suffer from large variability and ambiguity

motion in the image, their motion patterns should be quite distinguishable from people's non-rigid motion.

Optical flow is one feature that encodes motion patterns. Optical flow is also independent with respect to visual appearance. And it is not limited to high resolution images, being able to capture enough information from only a limited amount of pixels. Therefor using optical flow to separate robots' movement from people's movement is appealing for its independence on people and robot visual appearances (i.e., color, size or shape). Training a classifier based on optical flow also gives more generality than appearance. The same classifier may be applied to several scenarios with different people and different robots, without needing explicit re-training.

Our approach relies on the motion patterns extracted from optical flow, which have been used previously by Viola et al. [41] and Dalal et al. [4, 5] in order to detect pedestrian in images and videos. Viola et al. combine the optical flow with wavelet-based features to model people's appearance, while Dalal et al. compute histograms of the flow directions and histograms of the directions of the spatial derivative of the optic flow. These approaches consider the instantaneous detection of people, so they utilize just one optical flow image for classification. In contrast to these approaches, we propose to build a feature vector that contains the optical flow statistics across several frames (i.e. spatio-temporal volume).

Our model, thus exploits the optic flow differences between the non-rigid people's motion and the robot's rigid displacements.

Figure 3 shows a global view of our approach, which has the following steps: (i) pre-processing, (ii) feature computation and (iii) learning. The learning can be further subdivided in (a) linear dimensionality reduction, (b) feature selection and (c) FuzzyBoost learning. The pre-processing step (Fig. 3(a)) performs background subtraction for detection [2] and nearest-neighbor for tracking, followed by the (dense) optic flow computation [29] in the bounding boxes of the tracked objects. On the feature computation step (Fig. 3(b)), we consider the following raw features:

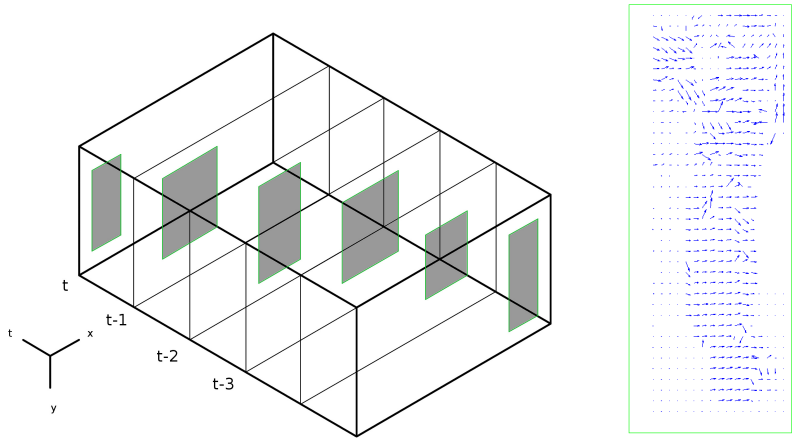
Fig. 2. Illustration of robot appearance variability. Also visible in an EuroNews report at http://youtu.be/Swrwe4NYG4o

- Histogram Of Gradients (HOG) [4], which computes the histogram of the optic flow orientation, weighted by its magnitude. We consider both Cartesian and Polar cells.
- Motion Boundary Histogram (MBH) [5], computed from the spatial gradient of the optical flow. Similarly to HOG, this feature is obtained by weighting the histogram of the optical flow's gradient. We consider both Cartesian and Polar cells.

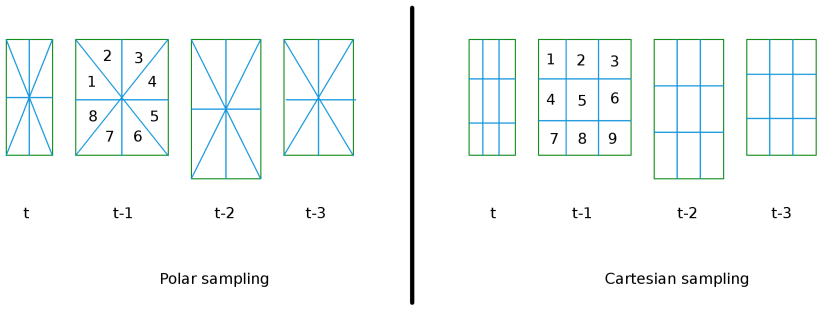
The feature computation is followed by stacking the features of each cell for the previous frames in order to build the feature vector. Thus, the model contains flow statistics of the cells in the previous frames.

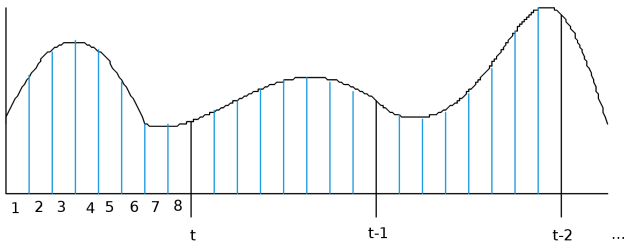
The main difference between our previous work [13] and this paper is the search for spatio-temporal patterns (i.e. searching for several feature dimensions along frames). The main difference in this work to another previous work by us [27] is the added comparisons of our whole feature selection algorithm (that exploits linear dimensionality reduction) with Sequential Floating Feature Selection (SFSS) [32] and Boost Feature Subset Selection (BFSS) [46]. And also added comparisons between the learning part of our algorithm (FuzzyBoost) with Support Vector Machines (SVM) with linear kernels and with Radial Basis Function (RBF) kernels.

In our initial approach, the TemporalBoost algorithm assumes just temporal patterns (i.e. a single feature dimension along frames) for each dimension of the



(a) Illustration of the pre-processing steps. The cuboid represents a slice of the video, the shaded regions are the ouptut of the background detection + tracking at each frame. Then, the optical flow is computed for each track on the sequence. The right-side image shows an example of a person's optical flow





(b) Illustration of the feature computation step. The top figures show the two types of cells considered in this work. At each cell, the optic flow or optic flow derivatives are extracted in order to compute the weighted histogram of the orientations. The bottom figures shows an example of the feature vector, which contains the concatenated weighted histograms of the current and previous frames

Fig. 3. Overview of the pre-processing (top) and feature computation (bottom) steps of our approach.

data samples. Although the TemporalBoost algorithm improves the classification performance, it is ignoring the information contained in the spatio-temporal patterns. In this work we present the advantages of the FuzzyBoost algorithm, which finds both the temporal and spatio-temporal patterns that improve the classification performance. We find that the FuzzyBoost has better generalization properties than the GentleBoost [17], the TemporalBoost [35] and the Support Vector Machines (SVM) with either linear kernels [3] and Radial Basis Function (RBF) kernels on this type of problems.

1.2. *Related Work*

Detection of humans in images is a very active research area in computer vision with important applications such as pedestrian detection, people tracking and human activity recognition. On one hand, the articulated models consider each body part in order to construct the full body model, which allows to detect a person and the pose of each body part. On the other hand, the global models compute image features in a bounding box, which allows just person detection. Although the part-based models provide more information, they have a low efficiency and cannot cope with self-occlusions when compared to the global models. In addition, global models have attained very low false positive rates in challenging scenarios [41, 42, 7, 39, 23, 9]. These reasons have lead to the current trend on people detection: (a) Efficient representations that (b) minimize the false positive rate and (c) will be applied mostly to pedestrian detection for cars.

Efficiency is attained by the application of boosting cascades, which reduces the amount of weak learners evaluated on the image [41]. At each step of the cascade, the weak learner output discards image regions that are not promising detections. Iterative bootstrapping during the learning phase is essential to attain low false positive rates [41, 7, 12]. At each iteration, bootstrapping augments the training set of the negative samples by adding the false positives of the current classifier. Finally, the availability of challenging datasets such as the Caltech pedestrian [8], TUD-Brussels [45] and the new version of INRIA's [39], allows to set standards for evaluation and comparison of the results.

Several features have been proposed to extract the common patterns of people's global appearance: silhouette [6, 43], image gradient [4], color distribution of each limb [33], Haar wavelets [41], optic flow [41, 5], color self-similarity [42], sparse granular features [16], integral images of several color channels [7] and combinations of some of the features mentioned above [42].

Detection of robots in images have become a very popular field of research in the RoboCup[2] framework, which focus on cooperative robot interaction [19, 26, 21]. Previous works have addressed the visual detection of the middle size league robots, which are similar to most of the commercial mobile platforms used for research.

[2] http://www.robocup.org/

The rules of the middle size league provide context information that brings robust features for the soccer game, but with a lack of generalization for other scenarios. For instance, the robots should be mostly black and the teammates and opponents have markers of different colors. The context information and the real-time constraints of the game has biased the utilization of very simple features such as: percentage of black pixels on a region, image region entropy, length of the black/green contours [26, 19]. More complex features include the gradient orientation histogram [26, 19] and the Eigenimages [21].

The boosting algorithms build a strong classifier as an addition of "weak learners", which usually are "decision stumps" [17]. These stumps select a threshold on one of the dimensions of the samples, a procedure that is sensitive to noise and overlapping between classes.

When considering fields such as pattern recognition and regression, several works have shown experimental performance improvements by grouping subsets of data samples in the weak learner.

On one hand, when the context of the problem leads to the definition of the groups, the search for the most relevant groups is constrained to a small number of hypotheses which we refer to as context-oriented search. This type of search depends on the particular assumptions provided by the context of problem, so its application is constrained to certain data sample structure. Examples of context-oriented subset search include the analysis of genomic data [25], image segmentation [1], scale-space regression [31] and human activity recognition [37, 35, 34] amongst others. The main advantage of these approaches is the low computational complexity of the search, while their main drawback is the applicability range.

On the other hand, when the context of the problem do not provide clear hints to define the groups, the search for the most relevant groups lies in the feature selection approaches [15]. The feature selection paradigm relies on a cost function that ranks either single features [15, 20, 10] or subsets of features [28, 32]. This type of approach allows to apply the same search algorithm across several problems, having better generalization properties than the context-oriented search. However, the computational complexity of the search could be prohibitively large, so various heuristics have been proposed to avoid the exhaustive cost function computation. Examples of individual feature selection include: the correlation coefficient [15], mutual information [15] and the RELIEF algorithm [20], amongst others. Examples of subset feature selection include branch-and-bound [28], Sequential Floating Feature Selection (SFSS) for classification [32] and regression [22], amongst others. The more recent dimensionality reduction algorithms [18, 14, 44] address feature selection by mapping the initial data space onto a subspace where regression and classification tasks perform better than in the original space. The map between features is computed from a cost function that considers criteria such as variance maximization [18], relative variance between and across classes [14] and local discrimination properties in the proximity of the decision boundary [44].

We compare the performance of our approach to the classical subset selec-

tion of SFSS[32] and the more recent bootstraping approach by replication. SFSS [32] defines a cost criterion as a function of the feature subset, performing additions/removals of features that increase/decrease the cost criterion. SFSS copes with non-monotonic criterion functions and obtains the very good results while having few parameters than plus l-take away r [38]. We consider SFSS to select the feature subsets, designing an adequate criterion function for the fuzzy decision stumps.

A recent trend on boosting selects features (either single or subsets) using the idea of bootstraps. Bootstraps are sets of samples either constructed by selection (subset) [11, 30] or replication (multiset) [46], designed to capture local feature properties which are not extracted by the global weights of the boosting methods. We consider the Boost Feature Subset Selection (BFSS) [46] that at each round, builds multisets in order to select the best feature and then reduces the sampling probability of the worst samples according to the best feature. The final step is to remove the best feature from the samples and then repeat the procedure until the subset feature size is reached. We design an adequate criterion function for the BFSS algorithm when using fuzzy decision stumps.

2. Target Representation

Regarding the (dense) optical flow computation, we use Ogale et. al. [29] implementation[3], an algorithm that introduces a new metric for intensity matching, based on the unequal matching (i.e. unequal number of pixels in the two images can be correspondent to each other). We chose this algorithm for its good balance between computational load and robustness to noise. Figure 4 shows examples of this flow for a person and a robot.

Previous works have shown the advantages of the Histogram Of Gradient features [4, 24], which divide the image in cells and compute the histogram of the orientation of the gradient weighted by its magnitude. This approach has been applied on the optical flow image and the spatial derivatives of its components [4, 5]. We follow these approaches by considering both the Histogram of Optical Flow (HOF) and the Motion Boundary Histogram (MBH), using two types of cells: Cartesian and polar cells (illustrated in Figure 3(b)).

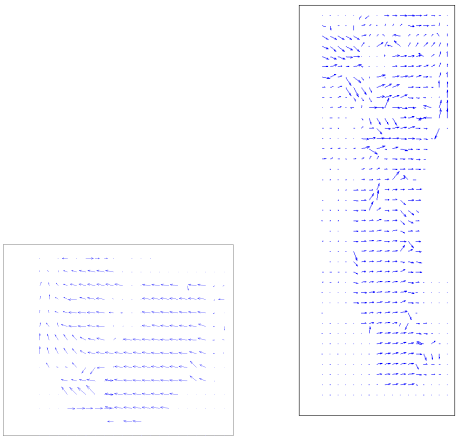
2.1. *Histogram Of Flow*

The most discriminative and efficient features based on gradients compute weighed histogram of the raw features, such as the histogram of gradients (HOG) [4] and Histogram of Optic Flow (HOF) [5]. Given an optic flow image, the first step is to divide the image in cells (according to a Cartesian or Polar sampling strategy) followed by the computation of the histogram of flow orientation weighed by its

[3] http://www.cs.umd.edu/~ogale/download/code.html

(a) Overlaid images of a pioneer robot moving

(b) Overlaid images of a person walking



(c) Optical flow of the robot

(d) Person's optical flow

Fig. 4. Examples of optical flow for robot and person.

magnitude. In difference to the original HOG features, that overlap sampling regions, we don't consider overlapping. In the case of the Cartesian cells, the features are parametrized by the number of intervals the x direction nI_x, the number of intervals in the y direction nI_y and the number of bins nB, which defines $nI_x \times nI_y$ cells. We denote the flow histograms as the row vector computed at frame t as

$$\mathrm{HOF}^t = \left[\mathrm{HOF}^t_{1,1} \ldots \mathrm{HOF}^t_{i,j} \ldots \mathrm{HOF}^t_{nI_x,nI_y}\right] \in \mathbb{R}^{nI_y \cdot nI_x \cdot nB},$$

where $\mathrm{HOF}^t_{i,j} \in \mathbb{R}^{nB}$ denotes the HOF computed at cell (i, j) and frame t. The target representation stacks all the HOF^t vectors in the previous $\tau - 1$ frames,

$$x_i = \left[\mathrm{HOF}^t \ldots \mathrm{HOF}^{t+\tau-1}\right] \in \mathbb{R}^{nI_y \cdot nI_x \cdot nB \cdot \tau}. \quad (1)$$

In the case of the polar cells, the features are parametrized by the number of angular regions nR and the number of bins nB, which defines nR cells.

$$\mathrm{HOF}^t = \left[\mathrm{HOF}^t_1 \ldots \mathrm{HOF}^t_i \ldots \mathrm{HOF}^t_{nR}\right] \in \mathbb{R}^{nR \cdot nB},$$

where HOF^t_i denotes the HOF computed at cell i and frame t. The target representation is

$$x_i = \left[\mathrm{HOF}^t \ldots \mathrm{HOF}^{t+\tau-1}\right] \in \mathbb{R}^{nR \cdot nB \cdot \tau}. \quad (2)$$

2.2. *Motion Boundary Histogram*

The computation of MBH [5] has the following steps: (i) separation of the x and y components of the optical flow into independent images, (ii) compute the gradient of each flow component image, (iii) divide each gradient image in cells, (iv) compute the weighted histogram of the gradient images and (v) stack the weighted histograms of both images in a single feature vector.

The gradient of the flow components captures the local orientations of motion edges, which extract the contours of rigid motions. For instance, the region of a moving forearm has self-similar motion, so the MBH feature will extract the forearm's motion edge. Thus, the person's MBH extracts the contours of the moving limbs. In the case of the ideally clean optical flow of a robot, the MBH extracts the contour of the robot. We do it by considering the two flow components (x and y) as independent images, and taking their gradients. The target representation of the Cartesian cells is as follows:

$$\mathrm{HOG}^t_x = \left[\mathrm{HOG}^t_{1,1} \ldots \mathrm{HOG}^t_{i,j} \ldots \mathrm{HOG}^t_{nI_x,nI_y}\right] \in \mathbb{R}^{nI_y \cdot nI_x \cdot nB},$$

where HOG^t_x denotes the concatenation of the histograms computed on the O^t_x, the x component of the optical flow. $\mathrm{HOG}^t_{i,j}$ denotes the HOG computed at the Cartesian cell (i, j) and frame t of image O^t_x.

$$\mathrm{HOG}^t_y = \left[\mathrm{HOG}^t_{1,1} \ldots \mathrm{HOG}^t_{i,j} \ldots \mathrm{HOG}^t_{nI_x,nI_y}\right] \in \mathbb{R}^{nI_y \cdot nI_x \cdot nB},$$

where HOG^t_y denotes the concatenation of the histograms computed on the O^t_y, the y component of the optical flow. The target representation stacks all the HOG^t vectors in the previous $\tau - 1$ frames,

$$x_i = \left[\text{HOG}_x^t \ \text{HOG}_y^t \ \ldots \ \text{HOG}_x^{t+\tau-1} \ \text{HOG}_y^{t+\tau-1}\right] \in \mathbb{R}^{2 \cdot nI_y \cdot nI_x \cdot nB \cdot \tau}. \quad (3)$$

In the case of the polar sampling of the cells, the target representation is as follows:

$$\text{HOG}_x^t = \left[\text{HOG}_1^t \ \ldots \ \text{HOG}_i^t \ \ldots \ \text{HOG}_{nR}^t\right] \in \mathbb{R}^{nR \cdot nB},$$

where HOG_i^t denotes the HOG computed at the polar cell i and frame t of image O_x^t. The correspondent HOG of the y component is:

$$\text{HOG}_y^t = \left[\text{HOG}_1^t \ \ldots \ \text{HOG}_i^t \ \ldots \ \text{HOG}_{nR}^t\right] \in \mathbb{R}^{nR \cdot nB}.$$

The target representation stacks all the HOG^t vectors in the previous $\tau - 1$ frames,

$$x_i = \left[\text{HOG}_x^t \ \text{HOG}_y^t \ \ldots \ \text{HOG}_x^{t+\tau-1} \ \text{HOG}_y^{t+\tau-1}\right] \in \mathbb{R}^{2 \cdot nR \cdot nB \cdot \tau}. \quad (4)$$

3. The FuzzyBoost algorithm

Boosting algorithms provide a framework to sequentially fit additive models in order to build a final strong classifier, $H(x_i)$. The final model is learned by minimizing, at each round, the weighted squared error

$$J = \sum_{i=1}^{N} w_i (y_i - h_m(x_i))^2, \quad (5)$$

where $w_i = e^{-y_i h_m(x_i)}$ are the weights and N the number of training samples. At each round, the optimal weak classifier is then added to the strong classifier and the data weights adapted, increasing the weight of the misclassified samples and decreasing for the correctly classified ones [40].

In the case of GentleBoost it is common to use simple functions such as decision stumps. They have the form

$$h_m(x_i) = a\delta\left[x_i^d > \theta\right] + b\delta\left[x_i^d \leq \theta\right], \quad (6)$$

where d is the feature index and δ is the indicator function (i.e. $\delta[condition]$ is one if *condition* is *true* and zero otherwise). Decision stumps can be viewed as decision trees with only one node, where the indicator function sharply chooses branch a or b depending on threshold θ and feature value x_i^d. In order to find the stump at each round, one must find the set of parameters $\{a, b, d, \theta\}$ that minimizes J w.r.t. h_m. A closed form for the optimal a and b are obtained and the value of pair $\{d, \theta\}$ is found through exhaustive search [40].

3.1. *Fuzzy weak learners optimization*

We propose to include the feature set (F) as an additional parameter of the decision stump, as follows:

$$h_m^*(x_i) = \frac{1}{||F||}\left(a\, F^T\delta\left[x_i > \theta\right] + b\, F^T\delta\left[x_i \leq \theta\right]\right), \tag{7}$$

which can be rearranged in order to put a and b in evidence

$$h_m^*(x_i) = a\,\frac{F^T\delta\left[x_i > \theta\right]}{||F||} + b\,\frac{F^T\delta\left[x_i \leq \theta\right]}{||F||}. \tag{8}$$

where $x_i \in \mathbb{R}^D$ and the vector $F \in \mathbb{Z}_2^D$, denotes a D dimensional vector with binary components, and the non-zero components of F define a feature set. The vector F chooses a group of original sample dimensions that follow the indicator function constraints of Eq. (7) in order to compute the decision stump $h_m^*(x_i)$. Note that the feature sets of classic decision stump are

$$\mathcal{F} = \{F_1, \ldots F_d \ldots, F_D\}, \text{where } F_d = \begin{bmatrix} 0 \\ \vdots \\ 1_d \\ \vdots \\ 0 \end{bmatrix}. \tag{9}$$

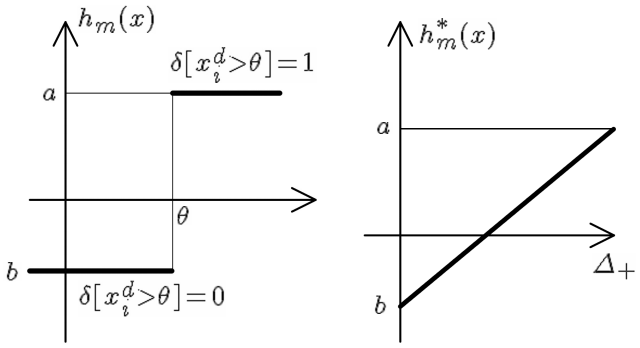
Therefore, the vector F generalizes GentleBoost by considering additional feature dimensions. From Eq. (8) it is easier to see that the selector F is replacing the indicator function (i.e. a true or false decision) by an average of decisions. The new functions are:

$$\Delta_+(x_i, \theta, F) = \frac{F^T\delta\left[x_i > \theta\right]}{||F||}, \tag{10}$$

$$\Delta_-(x_i, \theta, F) = \frac{F^T\delta\left[x_i \leq \theta\right]}{||F||} \tag{11}$$

and they compute the percentage of features selected by F that are above and below the threshold θ. The functions Δ_+ and $\Delta_- = 1 - \Delta_+$ of Eq. (11) sample the interval [0 1] according to the number of features selected (i.e. non-zero entries of F). For example, if $||F|| = 2$ this yields to $\Delta \in \{0,\ 1/2,\ 1\}$, if $||F|| = 3$ to $\Delta \in \{0,\ 1/3,\ 2/3,\ 1\}$ and so on. The new weak learners, the fuzzy decision stumps, are expressed as $h_m^*(x_i) = a\Delta_+ + b\Delta_-$.

We illustrate in Fig. 5 the difference between the classic decision stumps and our proposed fuzzy stumps. The response of the decision stump is either a or b according to the feature point x_i^d, while the fuzzy stump response is a linear function of Δ_+ that weights the contribution of the decisions a and b, thus the name fuzzy stump.



(a) (b)

Fig. 5. Response of the weak learners: (a) decision stumps and (b) fuzzy stumps

Replacing the fuzzy stumps of Eq. (8) in the cost function (Eq. (5)), the optimal decision parameters a and b are obtained by minimization,

$$a = \frac{\bar{\nu}_+\bar{\omega}_- - \bar{\nu}_-\bar{\omega}_\pm}{\bar{\omega}_+\bar{\omega}_- - (\bar{\omega}_\pm)^2}, \qquad b = \frac{\bar{\nu}_-\bar{\omega}_+ - \bar{\nu}_+\bar{\omega}_\pm}{\bar{\omega}_+\bar{\omega}_- - (\bar{\omega}_\pm)^2}, \tag{12}$$

$$\text{with} \quad \begin{array}{ll} \bar{\nu}_+ = \sum_i^N w_i y_i \Delta_+^T, & \bar{\nu}_- = \sum_i^N w_i y_i \Delta_-^T, \\ \bar{\omega}_+ = \sum_i^N w_i \Delta_+^T, & \bar{\omega}_- = \sum_i^N w_i \Delta_-^T, \\ \bar{\omega}_\pm = \sum_i^N w_i \Delta_-^T \Delta_+^T. & \end{array}$$

Algorithm 1: Generation of feature sets $\mathcal{F}$ of Eq. (7) using the Temporal-Boost algorithm [35]. Line 3 sets ones at components $F_{ij}(d)$ where feature bin, cell and frame conditions are fulfilled.

input : Spatio-temporal feature, such as HOF (1), (2), MBH (3), (4) with nB bins and nC cells
output: $\mathcal{F} = \{F_{11}, \dots F_{ij} \dots, F_{nBnC}\}$
1 **for** each bin $b_i \quad i = 1 \dots nB$ **do**
2 **for** each cell $c_j \quad j = 1 \dots nC$ **do**
3 $F_{ij}(d) = \delta[d_b = b_i \wedge d_c = c_j \wedge d^t = t, \dots, t+\tau-1]$;
4 **end**
5 **end**

There is no closed form to compute the optimal θ and F, thus exhaustive search is usually performed. Although finding the optimal θ is a tractable problem, the

search for the best F is NP-hard thus generally impossible to perform. In previous work, we assumed the temporal similarity of each feature dimension in order to build the feature sets $\mathcal{F}$ [35]. Alg. 1 shows the feature set selection of TemporalBoost, a heuristic that builds temporal stripes in the spatio-temporal feature volume. In this work we address the search for sets in the full spatio-temporal volume, guiding the search and reducing the number of possible candidates through dimensionality reduction algorithms.

Dimensionality reduction algorithms, as explained below, provide a projection matrix that we explore in order to find feature set candidates. Figure 6 shows the FuzzyBoost algorithm, which relies on the sets of features

$$\mathcal{F} = \{F_{11}, \dots F_{ij} \dots, F_{n_{\text{rows}} n_s}\}, \tag{13}$$

provided by a feature search on a linear projection matrix L with n_{rows} rows and a predefined number of intervals n_s. In the following section we present the algorithm that searches for $\mathcal{F}$ using a linear dimensionality reduction technique.

(1) Given:
$(x_1, y_1), \dots, (x_N, y_N)$ and $\mathcal{F} = \{F_{11}, \dots F_{ij} \dots, F_{n_{\text{rows}} n_s}\}$. Data $x_i \in X$, $y_i \in Y = \{-1, +1\}$ and feature sets $\mathcal{F}$ provided by a feature search on a linear projection matrix L with n_{rows} rows and a predefined number of intervals n_s.
Set $H(x_i) := 0$, initialize the observation weights $w_i = 1/N$, $i = 1, 2, \dots, N$

(2) Repeat for $m = 1, \dots, M$

  (a) Find the optimal weak classifier h_m over (x_i, y_i, w_i) using the feature sets $\mathcal{F}$.
  (b) Update weights for examples $i = 1, 2, \dots, N$, $w_i := w_i e^{-y_i h^*_m(x_i)}$

(3) Compute the strong classifier as $H(x_i) = \sum_m^M h^*_m(x_i)$ and classify the sample x_i according to $\operatorname{sgn} H(x_i)$

Fig. 6. FuzzyBoost algorithm

3.2. *The search space for the feature set*

The search for the feature set is a NP-hard problem so exhaustive search could take a prohibitive long time. For instance, consider a case with twenty dimensions ($d = 20$), the number of available feature sets is already $conb = \sum_{k=1}^{20} C_k^{20} \approx 10^6$. Since the computational complexity is exponential on the number of dimensions, the search is infeasible for databases with a high dimensionality. In addition, some of the dimensions do not provide useful information for discriminative purposes.

In a previous work [27], we introduced an algorithm that relies on linear dimensionality reduction techniques in order to find good set candidates for the FuzzyBoost. The linear mapping

$$x^* = Lx \tag{14}$$

contains relevant information about the correlations between dimensions of the original feature space. Our proposal analyzes each row of the matrix L (row projection vector) by grouping vector components with similar values. Our rationale follows the weight similarity approach: if the weight of a dimension in the (row) projection vector is similar to other dimension(s) in that vector, this implies some correlation level between those dimensions. Amongst three dimensionality reduction algorithms, Principal Component Analysis (PCA), Linear Discriminant Analysis (LDA) and Multiple Metric Learning for large Margin Nearest Neighbor (MMLMNN) Classification [44], MMLMNN provided the best results [27]

The recently proposed MMLMNN method [44] attempts to learn a linear transformation of the input space, such that each training input should share the same labels as its k nearest neighbors (named target neighbors) and the training inputs with different labels (named impostors) should be widely separated. These two terms are combined into a single loss function that has the competing effect of attracting target neighbors on one hand, and repelling impostors on the other. The optimization of the cost function obtains the projection matrix L that is related to the Mahalanobis metric $M = L^T L$. The MMLMNN method learns multiple locally linear transformations instead of a single global linear transformation, which is the case of the Large Margin Nearest Neighbor classifier (LMNN) [44]. The multiple transformations allow to model better the decision boundaries on highly nonlinear problems, due to the association of different Mahalanobis distance metrics for different parts in the input space. The parts (clusters) are defined by the class label, so there is a projection matrix for each class. Thus, MMLMNN acts as LDA but considers local constraints to build the linear transformation.

3.2.1. *Computing F from L*

Given the linear mapping L computed by MMLMNN, each row of the matrix is considered separately in order to extract feature set candidates. The sets are built by selecting the components of the row vector having very similar values and discarding components having very low values (see Alg. 2). The quantitative measures of closeness and low values are: the size of the similarity interval (Δ_s in Alg. 2) and the lower threshold (s_0 in Alg. 2). The values of the projection matrix are scaled as follows: $\mathcal{L}_{ij} = \frac{|L_{ij}|}{\max(L)}$, which ensures that $0 < \mathcal{L}_{ij} \leq 1$.

The lower threshold $s_0 \in [0, 1[$ removes components of $\mathcal{L}_i$ having low projection weights, which are the less meaningful dimensions. The number of intervals $n_s \in \mathbb{N}$ defines the size of the similarity interval Δ_s (line 2 of Alg. 2), which leads to the criterion to group dimensions with similar weights (line 5 of Alg. 2). On one hand, a large number of intervals n_s implies a low Δ_s that will group a few dimensions. On the other hand, a low number of intervals will generate a larger feature set F_{ij}. In order to see the effect of several choices of s_0 and n_s, we apply the Algorithm 2 using several pairs (s_0, n_s) for the linear mapping L (see Section 4.2).

In addition to the introduced new algorithm that relies on linear dimensional-

Algorithm 2: Generation of feature sets F of Eq. (7) from a scaled linear mapping $\mathcal{L}$

input : s_0 lower threshold, n_s number of intervals, $\mathcal{L}$ projection matrix
output: $F_{ij} \quad i = 1 \dots n_{\text{rows}} \quad j = 1 \dots n_s$

```
for each projection (row) vector L_i do
    compute Δ_s = (max(L_i) − s_0)/n_s;
    for j = 1 ... n_s do
        compute s_j = s_0 + (j − 1)Δ_s;
        F_ij = δ[s_j ≤ L_i < s_j + jΔ_s];
    end
end
```

ity techniques to select subsets of features, we also test and compare against two previously introduced approaches that rank subsets of features: The SFSS [32] and the BFSS [46] algorithms.

3.2.2. *Sequential Floating Feature Selection (SFSS)*

The SFSS algorithm creates and updates subsets of features according to a feature selection criterion. The algorithm begins with an empty set, then evaluates the inclusion of a new feature to the current set by evaluating the criterion function. The feature that maximizes the criterion function across all the augmented sets is added to the current set. The subsequent step is to evaluate the exclusion of a feature from the current set according to the criterion function on the reduced sets. If the function's value of one of the reduced sets is greater than the value of the criterion function of the current set, the feature is removed from the set. Finally, if a feature is removed (Line 12 of Alg. 3), the exclusion step is performed again on the reduced set (Line 14 of Alg. 3), otherwise the inclusion step is performed (Line 16 of Alg. 3). The Alg. 3 shows the SFSS algorithm.

SFSS can be applied both on top-down and bottom-up searches, copes with non-monotonic criterion functions, avoids the "nesting-effect" (i.e. not being able to remove a wrongly added feature) and just has one parameter to tune, the size of the subset k. The criterion function for the FuzzyBoost should consider the classification performance of the individual features (i.e. decision stump performance) and the correlation between features in order to rank groups of features. The criterion function is as follows:

$$\text{GFS}(x^F) = \frac{1}{N}\left(\sum_{i\in\mathcal{P}}\sum_{d\in F}\text{sgn}H(x_i^d) - \sum_{i\in\mathcal{N}}\sum_{d\in F}\text{sgn}H(x_i^d)\right) \tag{15}$$

$$\text{SFSS}(x^F) = \text{GFS}(x^F) + \sum_{d_i,d_j\in F, i\neq j, i<j}\rho_{i,j}(x_i^F, x_j^F), \tag{16}$$

Algorithm 3: Exhaustive SFSS algorithm. The feature set is initialized with every dimension $\{1, \ldots, d, \ldots, D\}$.

input : H strong GentleBoost classifier, k subset size, data samples x, labels y
output: $\mathcal{F} = \{F_1, \ldots F_d \ldots, F_D\}$

for each feature d **do**
  $F_d = d$
  $j = 0$
  while $j < k$ **do**
    Inclusion step
      $d^* = \arg\max \text{SFSS}(x^{F_d \cup d})$
      $F_d = F_d \cup d^*$
      $j = j + 1$
    Conditional exclusion step
      $d^- = \arg\max_{d \in F_d} \text{SFSS}(x^{F_d - d})$
    if $\text{SFSS}(x^{F_d - d}) > \text{SFSS}(x^{F_d})$ **then**
      $F_d = F_d - f^-$
      $j = j - 1$
      go to Conditional exclusion step
    else
      go to Inclusion step
    end
  end
end

where $\mathcal{P}$ denotes the positive samples, $\mathcal{N}$ the negative samples, F the feature set, x^F the selected features F of sample x and $\rho_{i,j}(x_i^F, x_j^F)$ the correlation coefficient between feature i and j. Eq. (16) has two components: (i) The Group Feature Score (GFS) (Eq. (15)) that computes the average of the contribution of the features in the set F to the GentleBoost sample classification and (ii) the correlation coefficient between features that belong to F.

3.2.3. *Boost Feature Subset Selection (BFSS)*

The BFSS algorithm builds incrementally a subset of features according to an individual feature score function. The algorithm begins with an empty set, then generates a bootstrap sample set using sample probabilities, $p(x_i)$. The individual feature score is computed on the bootstrap set and the top feature is added to the feature subset. The subsequent step is to find the worst w samples in the bootstrap set according to the top feature. The individual score of the top feature is computed in sets with removed samples and the top w scores select the worst samples. Finally, the sample probabilities of the other samples (best samples) are

reduced by a constant factor and the top feature is removed from the samples. This process is repeated until the subset size is reached.

BFSS is motivated by the fact that individual feature scores do not provide information to the subsequent feature selection steps. In order to provide information, the sample probability is reduced on the best classified samples by the current top feature, thus increasing the sample probability on the most difficult samples of the current feature. Then on the subsequent selection steps, the features selected are more likely to address the most difficult samples according to the previously selected features. The Alg. 4 shows the BFSS algorithm. The parameter values are: $m_b = 2N$, where N is the number of training samples, $k = \frac{1}{2}N$, $w = 0.96N$, $\epsilon = 0.96$.

Algorithm 4: BFSS algorithm

input : H strong GentleBoost classifier, w worst set size, ϵ decreasing factor, subset size k, bootstrap set size m_b, data samples x, labels y

output: $\mathcal{F} = \{F_1, \dots F_j \dots, F_k\}$

$p(x_i) = 1/N$, where N is the number of samples;
$j = 0$;
$F_{j-1} = \emptyset$;
while $j < k$ **do**
  Generate bootstrap sample set $\mathcal{B} = \{b_1, \dots, b_{m_b}\}$ with size m_b, using $p(x)$ and x;
  Compute IFS($b_i \in \mathcal{B}$);
  $d^* = \arg\max \text{IFS}(\mathcal{B}^d)$;
  $F_j = F_{j-1} \cup d^*$;
  Select the worst w samples $\mathcal{B}_{worst}$;
  Update the $p(x_i) = \epsilon p(x_i)$ where $x_i \in \mathcal{B} - \mathcal{B}_{worst}$;
  Remove f^* from x;
  $j = j + 1$;
end

BFSS has shown experimental advantages on gene-based classification and acts like boosting algorithms, but needs to tune three parameters: the size of the bootstrap set, the size of the worst set of samples and the degradation sampling factor. The Individual Feature Score (IFS) function is as follows:

$$\text{IFS}(x^d) = \frac{1}{N}\left(\sum_{i\in\mathcal{P}} \text{sgn}H(x_i^d) - \sum_{i\in\mathcal{N}} \text{sgn}H(x_i^d)\right). \tag{17}$$

The Individual Feature Score (IFS) (Eq. (17)) computes the average of the contribution of the strong classifier of GentleBoost.

4. Experimental results

We consider the target representations described in section 2: (i) HOF using Cartesian cells of Eq. (1), (ii) HOF using polar cells of Eq. (2), (iii) MBH using Cartesian cells of Eq. (3), (iv) MBH using polar cells of Eq. (4) and (v) combination of HOG and MBH with Polar cells. Regarding the learning algorithms, we consider the GentleBoost [17], the FuzzyBoost with feature set search [27], the Support Vector Machines with linear kernel [3] and the SVM with Radial Basis Function (RBF) kernel. The search space of FuzzyBoost is performed with three algorithms: (i) The similarity between the components of the MMLMNN transformation, (ii) the SFSS and (iii) the BFSS.

We compute the six different flow-based features on three scenarios: people walking, people loitering and robot moving. The motion patterns of people walking and robot moving will be properly extracted by optical flow-based features, so they are the nominal classification scenario. People loitering on the other hand, is a difficult situation as it provides small optical flow values. Both people walking and loitering are very common activities, therefore we decide to focus on them in this work. Figure 7 shows the setup of each scenario, which includes video sequences from 10 cameras and was recorded for our initial approach to this problem [13].

We implemented the feature computation and learning algorithm that provide a good trade-off between accuracy and speed. The parameters of the learning algorithm are extracted from the indoor setup with 10 cameras. We tested the software in an outdoors dynamic environment, running it on the video feed of the camera and classifying different unseen people and robots at the same time[4].

4.1. *Database and scenario assumptions*

We grabbed five groups of sequences, where each one includes images from 10 cameras. One group with a person walking, another group with a different person walking, two groups with the same pioneer robot moving in two different conditions, and the last group with a third person loitering. The people class videos have a total of 9500 samples of the optical flow and the robot class videos have a total of 4100 samples. The segmentation and tracking of the moving objects in the scene are provided by LOTS background subtraction for detection [2] and nearest neighbor for tracking. The LOTS algorithm provides the bounding boxes of the regions of interest and its respective segmented pixels. Nearest neighbor is computed between the center points of the two bounding boxes.

We follow a cross validation approach to compare the classification results, building two different groups of training and testing sets. The people loitering data is always in the testing set, each person belongs to the training set for one of the experiments, and each pioneer robot sequence belongs to the training set once. The recognition rates below are obtained through averaging the rates of the two tests.

[4] http://youtu.be/fGQtUcfZf9A

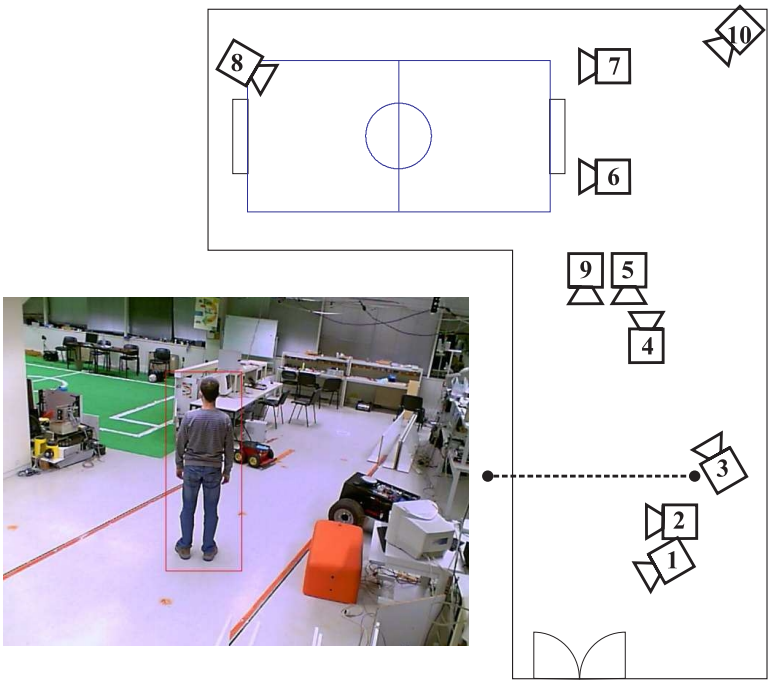


Fig. 7. Experimental setup for training scenario

4.2. *Parameters of the feature computation*

The spatio-temporal volume is constructed by stacking the feature vector of the current frame with the vectors of the previous four frames. The Cartesian HOF of Eq. (1) has eight bins and three cells in each direction, which yields $nB = 8$, $nI_y = 3$, $nI_x = 3$, $\tau = 5$ so $x_i \in \mathbb{R}^{360}$. The Polar HOF of Eq. (2) has eight bins and eight cells, which yields $nB = 8$, $nR = 8$, $\tau = 5$ so $x_i \in \mathbb{R}^{320}$. The Cartesian MBH of Eq. (3) has eight bins and three cells in each direction, which yields $nB = 8$, $nI_y = 3$, $nI_x = 3$, $\tau = 5$ so $x_i \in \mathbb{R}^{720}$. The Polar MBH of Eq. (4) has eight bins and eight cells, which yields $nB = 8$, $nR = 8$, $\tau = 5$ so $x_i \in \mathbb{R}^{640}$. The concatenation of Polar HOF and Polar MBH yields a feature vector $x_i \in \mathbb{R}^{960}$.

4.3. *Parameter selection of the feature search*

We define a set of pairs (s_0, n_s) in order to see the effect of the parameter selection in the performance of the generated feature sets in the FuzzyBoost algorithm. We set three low thresholds $s_0 \in \{0.1, 0.2, 0.3\}$, and three number of intervals, as follows: (i) $(0.1, 9)$, $(0.1, 18)$ and $(0.1, 27)$ for the first s_0, (ii) $(0.2, 8)$, $(0.2, 16)$ and $(0.2, 24)$ for the second s_0 and (iii) $(0.3, 7)$, $(0.3, 14)$ and $(0.3, 21)$ for the third s_0. The rationale behind this choice is to have Δ_s intervals with the same length across the different s_0 values, which allows to evaluate the pairs (s_0, n_s) fairly. For each pair (s_0, n_s), we

Fig. 8. On top, we have examples of person and robot training samples, and in the bottom we show samples of real-time classification in a dynamic outdoors setting featuring several different robots and people at the same time (visible at http://youtu.be/fGQtUcfZf9A). Red indicates a Person classification and Green indicates a Robot classification.

apply the Alg. 2 using the MMLMNN method in order to generate the feature sets $\mathcal{F}$. Then, $\mathcal{F}$ is applied on the FuzzyBoost algorithm of Figure 6 in a fixed number of rounds $M = 1000$. The quantitative evaluation considers two measures: (i) The maximum recognition rate attained on the testing set and (ii) the True Positives (TP) vs. False Positives (FP) curve.

4.4. *Discussion*

In the case of HOF-based features, Figure 9 and Figure 10 show that the FuzzyBoost algorithm with MMLMNN feature search finds spatio-temporal patterns (i.e. feature sets) that improve the detection performance of the GentleBoost and the TemporalBoost. The best performance is provided by the HOF using polar cells and the FuzzyBoost with MMLMNN feature search using $s_0 = 0.3, \Delta_s = 0.05$.

In the case of MBH-based features, the results of Figure 9 and Figure 10 suggest that the MMLMNN search heuristic cannot cope with the size of the feature space, while the SFSS and BFSS are able to extract meaningful feature sets. The best performance is provided by the MBH using polar cells and the FuzzyBoost with SFSS feature search.

In the case of the concatenation of HOG polar and MBH polar features, the results of Figure 11 show that the FuzzyBoost algorithm with MMLMNN feature search finds spatio-temporal patterns (i.e. feature sets) that improve the detection performance of the GentleBoost. The best performance is provided by the HOF + MBH with polar cells and the FuzzyBoost with BFSS feature search, followed by FuzzyBoost with MMLMNN feature search using $s_0 = 0.2, \Delta_s = 0.033$.

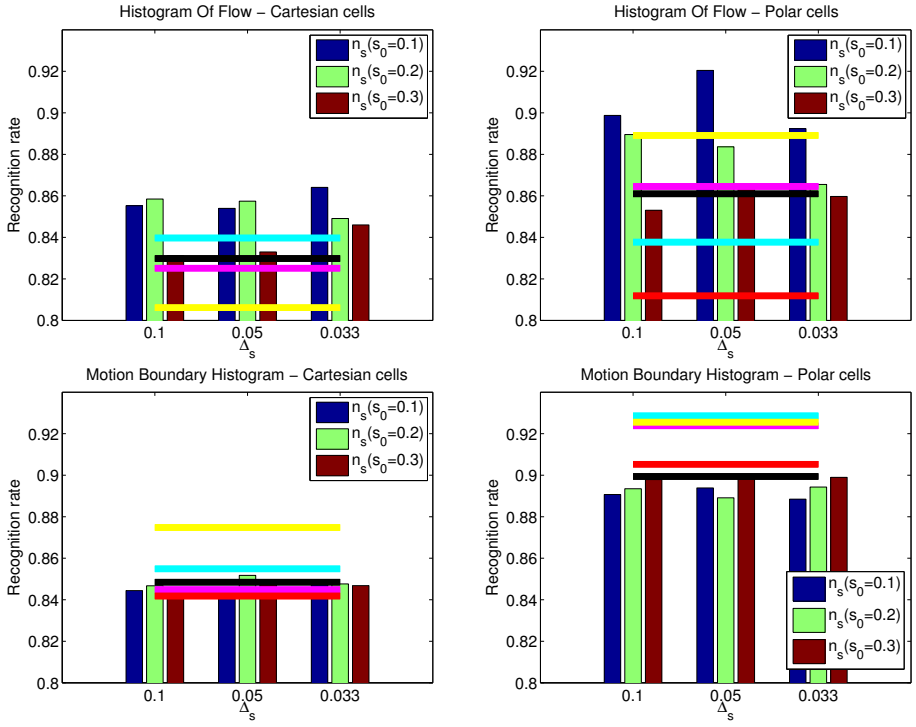


Fig. 9. The vertical bars indicate the performance for several parameters of the FuzzyBoost with the MMLMNN search. The horizontal bars show the performance of SVM with linear kernel (red), SVM with RBF kernel (yellow), GentleBoost (black), FuzzyBoost with SFSS (cyan), and FuzzyBoost with BFSS (magenta). On the top left plot, the SVM with linear kernel performance is very poor (54.35%) so it is not visible

Table 1 summarizes the results by taking the recognition rate at Equal Error Point (EEP). We remark that the concatenation of features HOG and MBH improves the performance in almost all the classifiers. The best overall result is

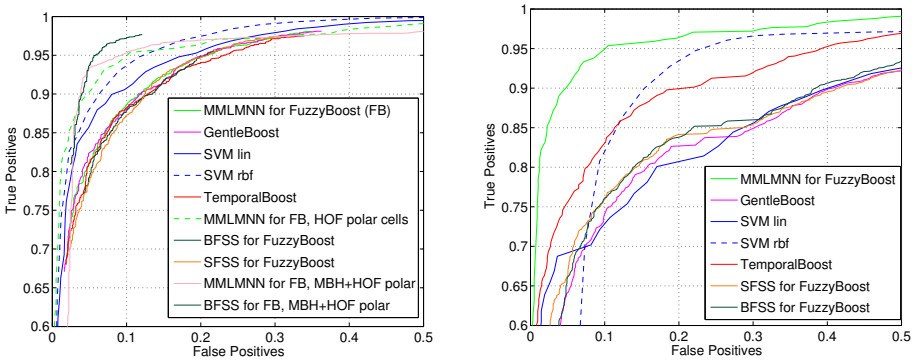


Fig. 10. True Positives vs. False Positives curves of MBH using polar cells (left) and HOF using polar cells (right). In the MBH plot is also displayed the best result of the HOF polar (FuzzyBoost with MMLMNN) and the top two results of the HOF polar + MBH polar for comparison purposes. The plots were created by varying the threshold of the strong classifier $H(x)$

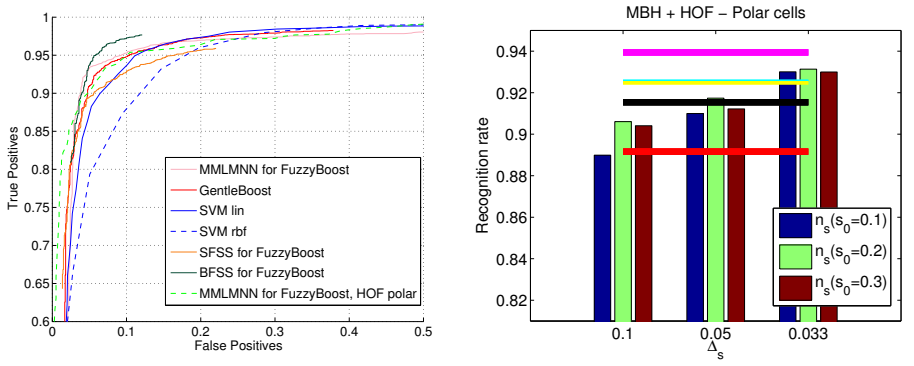


Fig. 11. True Positives vs. False Positives curves of the concatenation of MBH polar cells and HOF polar cells(left), where is displayed the curve of HOF polar (FuzzyBoost with MMLMNN). On the right side we show the accuracy of the classifiers for the concatenation of MBH polar and HOF polar, following the same presentation of the individual features in Figure 9.

provided by the FuzzyBoost with BFSS feature search.

4.5. *Real-Time Implementation*

Finally we take a compromise between the accuracy of the classifier and speed. We implemented HOF features extracted from polar cells and the FuzzyBoost classifier. The feature sets during training are provided by the MMLMNN heuristic. We test this implementation in an outdoor environment, aiming for real-time classification of several pedestrians and robots and the same time.

The larger part of the computational load is the dense optical flow computation.

Table 1. Summary of results (Recognition rate % at the Equal Error Point (EEP) of the Fig. 10). The bold underlined value is the best classification rate attained over all features and learning algorithms. The bold results highlight the best performance for each type of feature across the different learning algorithms

	HOF polar	MBH polar	HOF polar + MBH polar
SVM linear	80.07	90.6	91.83
SVM RBF	87.56	**92.01**	89.11
TempBoost	87.01	88.76	-
GentleBoost	81.85	88.95	93.35
FuzzyBoost + MLMNN	**93.01**	89.01	93.84
FuzzyBoost + BFSS	82.26	88.42	**94.57**
FuzzyBoost + SFSS	82.25	88.02	91.85

It must be computed efficiently because the output of the classifier is associated to messages streamed to other components that require low latency. Our choice was to subsample the images before computing the dense optical flow, defining a subsample ratio parameter according to the image size. We vary this parameter from 8 to 16 depending on the camera resolution.

In addition, we chose to extract optical flow only when a target has taken a "step" of S meters. This choice is motivated by three issues: (1) we wished to standardize the optical flow extracted from the moving targets – one pixel movement of a target close to the camera corresponds to much less movement than one pixel movement of a target "far away" from the camera; (2) optical flow from very small displacements are known to be noisier; (3) we wish to reduce the computational load and thus prevent computing optical flow between every frame.

Therefor, to reduce computational load, prevent very small flows, and guarantee some measure of standardization in the optical flows computed, we set this step parameter S to 1 meter. To do this, extrinsic camera calibration is required and the assumption of a flat ground plane.

The algorithm was able to run at 7 Frames Per Second on a single core machine, with 84.4% classification rate.

5. Conclusions

In this work we propose to analyze the spatio-temporal similarities of optical flow based features in order to distinguish people from robots. We compare two optical flow features, their combination and four learning algorithms. We test the Histogram Of Flow (HOF) [4], the Motion Boundary Histogram (MBH) [5] and their concatenation, extracted from Cartesian cells or polar cells. We applied the GentleBoost algorithm, the TemporalBoost algorithm, the SVM algorithm and the FuzzyBoost algorithm [27]. For the SVM we applied the linear and RBF kernels, and for the FuzzyBoost we compared three different feature search algorithms: Boost Feature Subset Selection (BFSS), Sequential Floating Feature Selection (SFSS)

and the Multiple Metric Learning for large Margin Nearest Neighbor (MMLMNN). FuzzyBoost is able to improve the classification rate by considering spatio-temporal decision functions (i.e. feature sets) at each round, a procedure that provide generalization capabilities over the common single feature decision functions.

The experimental results show that the concatenation of HOF and MBH consistently outperforms the individual features. In addition, FuzzyBoost almost always outperforms GentleBoost, TemporalBoost and SVM algorithms. The best overall result is provided by the FuzzyBoost with BFSS feature search, using the concatenation of HOF and MBH extracted from polar cells.

Finally, we implemented the FuzzyBoost with HOF features extracted from Polar Cells, which provides a trade-off between accuracy and speed. This implementation was tested in a real-time multi-camera dynamic scenario.

6. Acknowledgements

This work was supported by FCT [UID/EEA/50009/2013], partially funded by POETICON++ [STREP Project ICT-288382], the FCT Ph.D. programme RBCog and FCT project AHA [CMUP-ERI/HCI/0046/2013].

References

1. Shai Avidan. Spatialboost: Adding spatial reasoning to adaboost. In In Proc. European Conf. on Computer Vision, pages 386–396, 2006.
2. Terrance E. Boult, Ross J. Micheals, Xiang Gao, and Michael Eckmann. Into the woods: Visual surveillance of noncooperative and camouflaged targets in complex outdoor settings. Proceedings Of The IEEE, 89(10):1382–1402, October 2001.
3. Corinna Cortes and Vladimir Vapnik. Support-vector networks. Machine Learning, 20:273–297, 1995. 10.1007/BF00994018.
4. Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In CVPR '05: Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 1, pages 886–893, Washington, DC, USA, 2005. IEEE Computer Society.
5. Navneet Dalal, Bill Triggs, and Cordelia Schmid. Human detection using oriented histograms of flow and appearance. In European Conference on Computer Vision, 2006.
6. R. Diaz De Leon and L.E. Sucar. Human silhouette recognition with fourier descriptors. In Pattern Recognition, 2000. Proceedings. 15th International Conference on, volume 3, pages 709–712 vol.3, 2000.
7. P. Dollár, Z. Tu, P. Perona, and S. Belongie. Integral channel features. In BMVC, 2009.
8. P. Dollár, C. Wojek, B. Schiele, and P. Perona. Pedestrian detection: A benchmark. In CVPR, June 2009.

9. Piotr Dollár, Ron Appel, Serge Belongie, and Pietro Perona. Fast feature pyramids for object detection. PAMI, 2014.
10. Martin Drauschke and Wolfgang Förstner. Comparison of adaboost and adtboost for feature subset selection. In PRIS 2008, Barcelona, Spain, 2008.
11. B. Efron and R.J. Tibshirani. An introduction to the bootstrap. Monographs on Statistics and Applied Probability; 57. Chapman & Hall/CRC, New York, 1993.
12. M. Enzweiler and D.M. Gavrila. Monocular pedestrian detection: Survey and experiments. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 31(12):2179 –2195, dec. 2009.
13. Dario Figueira, Plinio Moreno, Alexandre Bernardino, José Gaspar, and José Santos-Victor. Optical flow based detection in mixed human robot environments. In Proc. of ISVC, 2009.
14. R. A. Fisher. The use of multiple measurements in taxonomic problems. Annals of Eugenics, 7(2):179–188, 1936.
15. Isabelle Guyon and André Elisseeff. An introduction to variable and feature selection. J. Mach. Learn. Res., 3:1157–1182, March 2003.
16. Chang Huang, Haizhou Ai, Yuan Li, and Shihong Lao. Learning sparse features in granular space for multi-view face detection. Automatic Face and Gesture Recognition, IEEE International Conference on, 0:401–407, 2006.
17. T. Hastie J. Friedman and R. Tibshirani. Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors). Annals of Statistics, 28(2):337–407, 2000.
18. I. T. Jolliffe. Principal component analysis. Springer, New York, 2002.
19. Ulrich Kaufmann, Gerd Mayer, Gerhard Kraetzschmar, and Günther Palm. Visual robot detection in robocup using neural networks. In RoboCup 2004: Robot Soccer World Cup VIII, pages 262–273, 2005.
20. Kenji Kira and Larry A. Rendell. The feature selection problem: traditional methods and a new algorithm. In Proceedings of the tenth national conference on Artificial intelligence, AAAI'92, pages 129–134. AAAI Press, 1992.
21. Sascha Lange and Martin Riedmiller. Appearance-based robot discrimination using eigenimages. In RoboCup 2006, pages 499–506, 2007.
22. Jiang Li, Michael T. Manry, Pramod Lakshmi Narasimha, and Changhua Yu. Feature selection using a piecewise linear network. IEEE Transactions on Neural Networks, 17(5):1101–1115, 2006.
23. Joseph Lim, C. Lawrence Zitnick, and Piotr Dollár. Sketch tokens: A learned mid-level representation for contour and object detection. In CVPR, 2013.
24. D. Lowe. Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision, 60(2):91–110, 2004.
25. Yihui Luan and Hongzhe Li. Group additive regression models for genomic data analysis. Biostatistics, 9(1):100–113, 2008.
26. Gerd Mayer, Ulrich Kaufmann, Gerhard Kraetzschmar, and Günther Palm. Biomimetic Neural Learning for Intelligent Robots. Springer Berlin / Heidel-

berg, 2005.
27. Plinio Moreno, Pedro C. Ribeiro, and J. Santos-Victor. Feature set search space for fuzzyboost learning. In Proceedings of the IbPRIA 2011, 2011.
28. P.M. Narendra and K. Fukunaga. A branch and bound algorithm for feature subset selection. IEEE Transactions on Computers, 26:917–922, 1977.
29. Abhijit S. Ogale and Yiannis Aloimonos. A roadmap to the integration of early visual modules. International Journal of Computer Vision, 72(1):9–25, april 2007.
30. Michael Oren, Constantine Papageorgiou, Pawan Sinha, Edgar Osuna, and Tomaso Poggio. Pedestrian detection using wavelet templates. Computer Vision and Pattern Recognition, IEEE Computer Society Conference on, 0:193, 1997.
31. Jin-Hyeong Park and Chandan K. Reddy. Scale-space based weak regressors for boosting. In ECML '07: Proceedings of the 18th European conference on Machine Learning, pages 666–673, Berlin, Heidelberg, 2007. Springer-Verlag.
32. P Pudil, J Novoviov, and J Kittler. Floating search methods in feature selection. Pattern Recognition Letters, 15(11):1119–1125, 1994.
33. Deva Ramanan, David A. Forsyth, and Andrew Zisserman. Tracking people by learning their appearance. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 29(1):65–81, Jan. 2007.
34. P.C. Ribeiro, P. Moreno, and J. Santos-Victor. Introducing fuzzy decision stumps in boosting through the notion of neighbourhood. Computer Vision, IET, 6(3):214–223, 2012.
35. Pedro Canotilho Ribeiro, Plinio Moreno, and José Santos-Victor. Boosting with temporal consistent learners: An application to human activity recognition. In Proc. of 3rd International Symposium on Visual Computing, pages 464–475, 2007.
36. A. Sanfeliu and J. Andrade-Cetto. Ubiquitous networking robotics in urban settings. In Workshop on Network Robot Systems. Toward Intelligent Robotic Systems Integrated with Environments. Proceedings of 2006 IEEE/RSJ International Conference on Intelligence Robots and Systems (IROS2006), October 2006.
37. Paul Smith, Niels da Vitoria Lobo, and Mubarak Shah. Temporalboost for event recognition. In International Conference on Computer Vision, volume 1, pages 733– 740, October 2005.
38. S.D. Stearns. On selecting features for pattern classifiers. In Proceedings of the 3rd International Conference on Pattern Recognition, pages 71–75, 1976.
39. Matteo Taiana, JacintoC. Nascimento, and Alexandre Bernardino. An improved labelling for the inria person data set for pedestrian detection. In JooM. Sanches, Luisa Mic, and JaimeS. Cardoso, editors, Pattern Recognition and Image Analysis, volume 7887 of Lecture Notes in Computer Science, pages 286–295. Springer Berlin Heidelberg, 2013.
40. A. Torralba, K.P. Murphy, and W.T. Freeman. Sharing visual features for multiclass and multiview object detection. IEEE Transactions On Pattern

Analysis and Machine Intelligence, 29(5):854–869, 2007.
41. Paul Viola, Michael J. Jones, and Daniel Snow. Detecting pedestrians using patterns of motion and appearance. International Journal of Computer Vision, 63(2):153–161, 2005.
42. Stefan Walk, Nikodem Majer, Konrad Schindler, and Bernt Schiele. New features and insights for pedestrian detection. In Conference on Computer Vision and Pattern Recognition (CVPR), San Francisco, 06/2010 2010. IEEE, IEEE.
43. Liang Wang, Tieniu Tan, Huazhong Ning, and Weiming Hu. Silhouette analysis-based gait recognition for human identification. IEEE Transactions on Pattern Analysis and Machine Intelligence, 25(12):1505–1518, 2003.
44. Kilian Q. Weinberger and Lawrence K. Saul. Distance metric learning for large margin nearest neighbor classification. J. Mach. Learn. Res., 10:207–244, June 2009.
45. Christian Wojek, Stefan Walk, and Bernt Schiele. Multi-cue onboard pedestrian detection. In CVPR, pages 1–8, 2009.
46. Xian Xu and Aidong Zhang. Boost feature subset selection: A new gene selection algorithm for microarray dataset. In International Conference on Computational Science (2), pages 670–677, 2006.