PERVASIVE SELF-LEARNING WITH MULTI-MODAL DISTRIBUTED SENSORS

Nicola Bicocchi[1], *Matteo Lasagni*[1], *Marco Mamei*[1], *Andrea Prati*[1], *Rita Cucchiara*[2], *Franco Zambonelli*[1]

[1]Dipartimento di Scienze e Metodi dell'Ingegneria, University of Modena and Reggio Emilia, Italy
[2]Dipartimento di Ingegneria dell'Informazione, University of Modena and Reggio Emilia, Italy

ABSTRACT

Truly ubiquitous computing poses new and significant challenges. A huge number of heterogeneous devices will interact to perform complex distributed tasks. One of the key aspects that will condition the role and impact of these new tecnologies on developed societies is how to obtain a manageable representation of the surrounding environment starting from simple sensing capabilities. This will make devices able to adapt their computing activities on an ever-changing environment. This paper presents a framework to promote unsupervised training processes among different sensors. This technology allows different sensors to exchange the needed knowledge to create a model to classify events happening into the environment. In particular we developed, as a case study, a multi-modal multi-sensor classification system combining data from a camera and a body-worn accelerometer to identify the user motion state. The body-worn accelerometer sensor learns a Gaussian mixture model of the user behavior pairing accelerometer data with information coming from the camera and uses it later on to classify the user motion in a fully autonomous way. Overall, the system works in a completely autonomous and unsupervised way. Experiments demonstrate the accuracy of the proposed approach in different situations.

***Index Terms*—** Multimodal data, Distributed Sensors, Unsupervised Motion Classification, Gaussian Mixture.

1. INTRODUCTION

Pervasive computing scenarios comprise a huge number of heterogeneous devices interacting with each other to achieve complex distributed applications. Smart camera and sensor networks could be employed in a variety of applications including environmental monitoring [1, 2, 3], navigation, and human interaction [4]. One central problem underlying such applications is how to obtain a useful representation of surrounding environment. While advances in hardware technologies (e.g., smart cameras, wireless sensors and RFID tags) are making economically feasible to collect a vast amount of information about the system context, it is still very difficult to organize and aggregate all the collected information in a coherent representation exploitable by software services.

In this paper we propose a general approach to create a self-training sensors' ecosystem. The idea consists in having sensors to exchange information and learn a model of some environmental properties under observation. Such a model can then be used to classify events of interest and take actions on the basis of a high-level representation of the system context. This would allow pervasive information and communication systems using the model to autonomously adapt to highly dynamic and open environments. In particular, as depicted in Fig. 1, different sensors can exchange information to build a representation of what is happening in the environment. Once the model has been built, some needed information can be spread to help other sensors to construct a model to interpret their data readings.

More in detail, the approach we propose lets sensors cooperate to create a model to classify some signals. For example, a "trained sensor" (one having already a classification model) can provide ground truth information to an "untrained sensor" (one able to produce raw data only) to enable it running a learning algorithm on its data and become a "trained sensor" itself. The core idea is to use labels coming from on sensor type to train another sensor type.

In addition the paper presents an actual use case of this approach combining data from a camera and body-worn accelerometer.

The rest of this paper will be structured as follows: Section 5 will present related works in the area. Section 2 presents the system architecture. Section 3 presents an implemented use case illustrating our approach and highlights the statistical approach used to run learning and classification algorithms. Section 4 shows experiments and results conducted on our test-bed. Finally Section 6 concludes and outlines future work in the area.

2. DESCRIPTION OF THE SYSTEM

In our framework, every kind of sensor is modeled by means of three modules: *(i)* a data acquisition module dedicated to raw data sampling, *(ii)* a learning module that collects the training data set pairing raw data with external data labels, and runs a learning algorithm to create a classification model

Work partially supported by the project CASCADAS (IST-027807) funded by the FET Program of the European Commission and by the project FREE SURF funded by the Italian MIUR PRIN program.

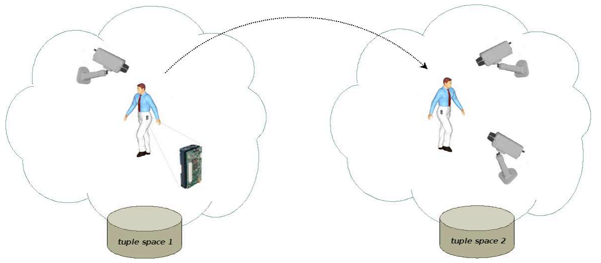

Fig. 1. Sketch of the multi-sensor setup. A user moves between two distinct pervasive environments. This allows pervasive information and communication systems using the model to autonomously adapt to highly dynamic and open environments.

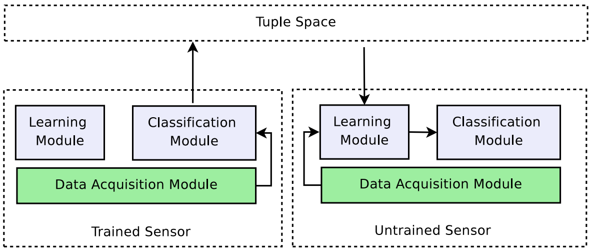

Fig. 2. Functional blocks of the system. The trained sensor classifies data and send it to the tuple space. The untrained sensor pairs labels coming from the tuple space with the sampled data to accomplish the learning phase.

of the data, *(iii)* a classification module that takes the model from the learning module and use it to classify new raw data (see Fig. 2).

All the components participating the process can be divided among three categories:

- *Trained sensors*. These sensors successfully completed the training phase of their classification algorithm or do not have a training phase at all (i.e., the classification algorithm is hard-coded in their program). Trained sensor can sample data from the environment and produce high level events based on the classification of their inputs. These events are pushed towards a tuple space (described below).

- *Untrained sensor*. These sensors are not trained yet. They sample data from the environment and store them in a buffer. Each entry of this buffer is composed by two elements. The first one is the feature vector, while the second one is initially empty and should be filled with a classification label. The training process ends when enough entries of the circular buffer are properly filled with a feature vector and the corresponding label, and the learning algorithm has been executed. At this stage, the sensor can start to sample, classify and publish brand new events (i.e., it has become a *trained sensor*).

- *Tuple spaces*. Every pervasive environment has to be provided with a tuple space. It receives high level events from *trained sensors* and forwards them to every *untrained sensor* that can use them. To avoid broadcast communication while keeping the overall system easy and clear, tuple space's information access is arbitrated by a publish-subscribe mechanism. Trained sensors can freely publish their events. Untrained sensors have to subscribe a template to define the events which they are going to use. Due to this, it is possible to realize a complex environment populated by a huge number of sensors monitoring different aspects of the environment. It is worth noting that, using this mechanism, multiple *trained sensors* can be coordinated to jointly train multiple *untrained sensors*.

The combined use of the above components (together with standard learning algorithms) allows to fulfill the idea of a self-training sensors' ecosystem outlined in Section 1. In particular different sensors can exchange information to build a model of what is happening in the environment by means of the tuple spaces.

3. IMPLEMENTED USE CASE

To verify in practice our ideas, we implemented a specific use case: train a body-worn accelerometer sensor to recognize motion states starting from a trained camera sensor. We are basically interested in recognizing the following motion states: "walking", "running", "standing still" and "falling down". These states can be recognized either by a camera sensor running a video analysis algorithm [5] or by analyzing the acceleration patterns collected by the body-worn low-power accelerometer sensor [6].

We used a PC with attached an analog camera and a Crossbow MicaZ mote equipped with the MTS310 sensor board. In particular:

- The tuple space runs on a dedicated server.

- The camera sensor acquires data and forwards it to a server running the classification module.

- The accelerometer sensor consists of two coupled MicaZ motes able to acquire data and to run the classification module. The motes send data to a server where the learning module is executed. The fact that the classification module of the accelerometer sensor actually runs on one of the MicaZ devices allows to classify the acceleration data even without the availability of any supporting infrastructure.

It is worth noting that functional modules depicted in Fig. 2 can be spreaded over the tuple spaces. By this way, load intensive tasks (e.g., computer vision algorithms and EM) can be executed over high-performing devices and lighter processes (e.g. online classification)over mobile and low-performing ones.

We deployed a trained camera sensor able to recognize the aforementioned motion states and configured it to upload arising events to the tuple space. Then, we introduced a second, untrained, body-worn accelerometer sensor. Once it first enters the camera field of view, it subscribes to the tuple space to receive the labels coming from the camera with information about the user motion state. Acceleration data and camera-based classification are paired together and a model to classify the user motion state on the basis of the sampled acceleration data is built on the fly.

Thanks to the labels coming from the camera, the whole system uses an unsupervised learning algorithm, therefore, no manual intervention is required.

From that moment on, the accelerometer becomes a *trained* sensor: other than producing raw data, can produce high-level labels describing the user motion. This classification can proceed even when the sensor exits the area covered by the camera.

3.1. Accelerometer Sensor

Learning Module

Accelerometer signal is aligned with the camera one. Once an event is detected by the camera, it is notified to all the subscribing sensors with neglibile communication delays.

The output signal $\mathbf{T} = \{\overrightarrow{a_x}, \overrightarrow{a_y}, \overrightarrow{a_z}\}$ provided by the accelerometer consists of the separate accelerations along the three axes x, y and z. To simplify the representation and reduce the dimensionality, the magnitude A of the sum vector is obtained. In addition, processing the non-oriented scalar magnitude would produce results independent on the actual way the accelerometer is worn and, thus, its orientation.

As stated in [6], the power spectrum of the magnitude A can be a good feature to use. First, the time series $\mathbf{AS} = \{A\}$ provided by the accelerometer is analyzed in the frequency domain through the FFT (Fast Fourier Transform) transform. To reduce the computational load, a small 64 element window with 32 samples of overlap is used. From these 64 elements, one 32 element power spectrum is produced. Ultimately, the DC component is canceled since it affects numerical stability. Summarizing, this procedure converts a time series $\mathbf{AS}$ defined in the spatial domain on $\mathbb{R}$ to a time series $\mathbf{X}$ defined in the frequency domain on $\mathbb{R}^{31}$.

Our cooperative learning approach assigns automatically a label/class to untrained sensor data using the posterior-based classification provided by the trained sensors. Thus, data coming from both types of sensors are paired to build the training set $\mathbf{TS} = \{S_1, S_2, \cdots, S_{N_{TR}}\}$, where N_{TR} is the total number of samples in the training set and $S_i = \langle X_i, C_i \rangle$ represents the i-th sample and is composed by the 31-dimensional accelerometer observation X and the corresponding label C provided by the camera with a majority-voting process.

Following a generative model, we first solve the inference problem of determining the class-conditional densities $p\left(\mathbf{X}|C_i\right)$ for each class C_i individually. Thus, let $\mathbf{X}^{C_i}$ be the set of accelerometer observations in the training set associated in the labelled data to the class C_i. This likelihood can be modelled with a 31-variate mixture of Gaussians (MoG).

In order to estimate its parameters, we employ the well-known EM algorithm and chose to keep the number of MoG components K fixed to two. This choice has been validated by a thorough experimentation which demonstrates that increasing K brings no benefit to the performance.

One important optimization we performed is to force the MoG 31×31 covariance matrices $\mathbf{\Sigma}_k^{C_i}$ to be diagonal, thus assuming that the 31 Fourier components are statistically independent. This notably reduces the size of the parameter set $\mathbf{A}$ used to represent the model, making $\mathbf{A}$ storage on the tiny memory of the body-worn sensor possible.

Classification Module

The set $\mathbf{A}$ of estimated parameters is stored in the mote to be used for motion pattern classification during the on-line testing phase. Since the accelerometer is now trained the on-line classification of a new sample X_{new} is simply performed by applying Bayes rule and MAP (Maximum A Posteriori) framework:

$$C_{new} = C_s \Leftrightarrow s = \underset{\forall r}{\arg\max}\ p\left(C_r|X_{new}, \mathbf{A}\right) \tag{1}$$

where, assuming uniform sample's priori, the posterior class probability can be written as:

$$p\left(C_r|X_{new}, \mathbf{A}\right) \propto p\left(X_{new}|\mathbf{A}, C_r\right) p\left(C_r|\mathbf{A}\right) \tag{2}$$

The first term in the right-side of equation 2 corresponds to the MoG for a given class C_r with parameters $\mathbf{A}^{C_r}$, while the second term can be simplified as $p\left(C_r\right)$ since there is no dependency of the class on the MoG's parameters. The term $p\left(C_r\right)$ represents the prior of class C_r and can be computed as the normalized occurrence of that class in the training set.

3.2. Camera Sensor

Our system is also provided with a standard fixed color camera. The classical computer vision flow considers first to segment moving objects (*segmentation*), then to track them on the field of view of the camera (*tracking*), and ultimately to analyze the objects to classify them and to infer their behavior (*object analysis*).

In order to define a general-purpose approach, we have defined a framework where visual objects are classified into three classes: actual visual objects, shadows, or "ghosts", i.e.

apparent moving objects typically associated with the "aura" that an object that begins moving leaves behind itself. Further details can be found in [7].

In this paper, we describe an approach of appearance-based probabilistic tracking, specifically conceived for handling all occlusions. The algorithm uses a classical predict-update approach. It takes into account not only the status vector containing position and speed, but also the *Appearance Memory Model* AMM and the *Probabilistic Mask* PM of the shape. AMM represents the estimated aspect (in RGB space) of the object's points: each value $AMM(p)$ represents the "memory" of the point's appearance, as it has been seen up to now. In the probability mask $PM(p)$ each value defines the probability that the point p belongs to the object.

Without going into much details, these two features are used both to perform the matching between the existing tracks and the objects detected in the current frames, and to detect the presence of occlusions. Occlusions are detected by analyzing a measure of confidence and likelihood computed on the probability mask PM. Further details can be found in [8].

Classification Module

Given the observation I^t, the algorithm bases its classification on the analysis of the tracked person. It is worth noting that the computer vision algorithm is capable to track multiple targets but needs to associate that (or those) target(s) wearing the accelerometers with the corresponding target ID(s). This association can be obtained with the same synchronization procedure described in Section 3.1, based on a specific event such as a jump. Thanks to sophisticated tracking algorithms, the distinction between "standing still" (class S), "walking" (class W) and "running" (class R) is almost straightforward. Given that the frame rate is constant, the movement speed can be easily estimated, even though perspective distortion can affect strongly the estimation.

Let H^t and V^t be the height of the bounding box and the velocity of the tracked person in I^t, respectively. The velocity is estimated through the distance (in pixels) covered by the person's centroid between I^{t-1} and I^t. By using a discriminative approach we can infer the posterior class C_i probabilities as follows:

$$p\left(C_i = S|I^t\right) = \frac{1}{\sqrt{2\pi}\sigma_1} exp\left\{-\frac{\left(V^t\right)^2}{2\sigma_1^2}\right\} \tag{3}$$

$$p\left(C_i = W|I^t\right) = \frac{1}{\sqrt{2\pi}\sigma_2} exp\left\{-\frac{\left(V^t - Th_1\right)^2}{2\sigma_2^2}\right\} \tag{4}$$

$$p\left(C_i = R|I^t\right) = \frac{1}{1 + exp\left\{-\left(V^t - Th_2\right)\right\}} \tag{5}$$

$$p\left(C_i = F|I^t\right) = \frac{1}{1 + exp\left\{-\left(\Delta H - Th_3\right)\right\}} \tag{6}$$

In other words, a person is classified as standing still if his/her velocity is almost zero, modeled with a zero-mean Gaussian (see equation 3) with a small standard deviation σ_1 (set to 1 in our tests). The same applies for classifying walking people (equation 4), but with a Gaussian centred on a positive value Th_1 (set to 5) and with a large standard deviation σ_2 (set to 3). Moreover, the person is classified as running if the velocity is greater than a threshold Th_2 (set to 10 pixels). Thresholding is performed with a logistic sigmoid function centred on the value Th_2 (see equation 5). Logistic sigmoid function provides a smoother transition than the classical Heaviside function used when thresholding. Finally, the classification of falling people is performed by looking at the variation $\Delta H = H^{t-1} - H^t$ of the height of the bounding box: if this variation is greater than a threshold Th_3 (set to 5) this means that the height is changing fast and this is a cue for a fall.

	Fall	Stand	Walk	Run
Fall	**84.27%**	8.70%	4.76%	2.28%
Stand	0.00%	**86.43%**	13.00%	0.57%
Walk	2.71%	0.48%	**94.81%**	2.00%
Run	0.43%	9.12%	1.72%	**88.73%**

Table 1. Confusion matrix of camera sensor classification.

Given the equations 3-6, a MAP approach provides the class maximizing the posterior probabilities:

$$C_j \;\mid\; j = \underset{\forall k}{\arg\max} \;\; p\left(C_k|I^t\right) \tag{7}$$

4. EXPERIMENTAL RESULTS

To evaluate the performances of our system we did several tests.

First, we recorded with the camera a person moving inside our department, acting in several motion states. At the same time, the same person wore the accelerometer sensors on his belt. While the scene was recorded, we manually collected the ground truth. The camera sampled at 30Hz, while the accelerometers at 100Hz. We reported in Fig. 3 a snapshot of the data we collected in this experiment. This plot shows the three accelerations (on X, Y and Z direction) with superimposed both the ground-truth and the camera classifications, in order to show the correspondence of accelerations' changes with mition variations.

We used these data in two ways. First, we compared the labels produced by the camera with the ones manually collected to measure the computer vision algorithm performances. Table 1 presents the confusion matrix of this classification. Looking at the reported confusion matrix of the camera sensor classification, it is possible to see that the vision algorithm works pretty well, having a precision of 88.59% and a recall of 88.56%.

Second, we measured the classification performance of the accelerometer, once it has been trained with the labels

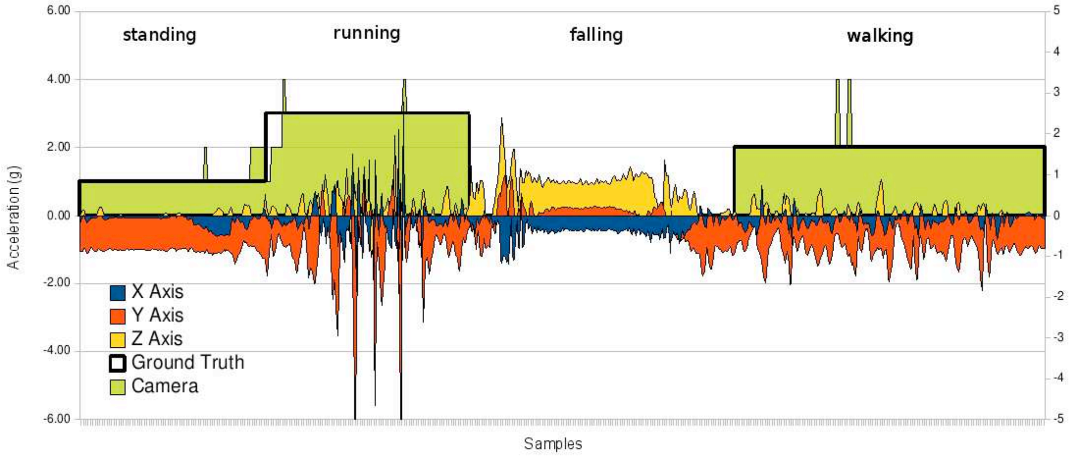

Fig. 3. A snapshot of the multi-sensor training set we used. Camera's classification labels and ground truth are superimposed over acceleration time series.

coming from the camera. We compared the labels produced by the accelerometer with the ground truth to obtain the confusion matrix reported in Table 2. Overall the classification has a precision of 85.25% and a recall of 69.50%. The poor performance on detecting falls (that contributes also to decrease the overall recall of the system) is due to the fact that the falling action is composed of a transient phase in which the accelerometer values can vary significantly. A more correct modeling of this action would be to take the sequence of observations into account, for instance with a HMM.

Although accelerometer classification performance is quite good, it is possible to see a reduction of performance with respect to the vision classification. This is due to two simple facts: first and most importantly, a belt-worn accelerometer produces a signal that is less descriptive than a full-fledge video sequence. Accordingly, the features extracted from the acceleration signal are less capable of discriminating among high-level behaviors (e.g., falling and walking) than the video sensor. Second, the accelerometer is trained with labels coming from the camera sensor and thus it is affected also by the errors coming from the camera.

However these considerations have to be applied only to the algorithms we used in this use case which are not the main contribution of this paper. It is worth considering that the framework we propose can support a moltitude of others algorithms which would perform much better.

To better understand the contribution of this latter source of errors, we conducted another experiment aimed at evaluating the robustness of the approach with respect to the camera classification errors.

We run a number of learning phases of the accelerometer data, passing to the accelerometer learning module increasingly corrupted class labels: we corrupted the ground truth information with errors sampled accordingly to the camera sensor confusion matrix reported in Table 1. Then, we tested the accelerometer classification performance. The graph in Fig. 4 shows on the x-axis the percentage of errors introduced, varying from 0 to 100 percent. Precision and recall are reported on the y-axis. As expected, classification capabilities are highly correlated with the quality of training data. Introducing an error of 20% halves the precision and recall levels.

	Fall	Stand	Walk	Run
Fall	**15.00%**	21.25%	63.75%	0.00%
Stand	0.00%	**77.63%**	22.37%	0.00%
Walk	0.00%	1.96%	**97.20%**	0.84%
Run	0.79%	0.79%	10.24%	**88.19%**

Table 2. Confusion matrix of accelerometer sensor classification. Note that the data at the basis of this confusion matrix are not the same of the camera sensor confusion matrix. The data considered by the camera are used by the accelerometer for training only.

To summarize the discussion about the performances of the system it is possible to see that: *(i)* the algorithms proposed to track moving persons and classify their motion state are suitable for this application scenario; *(ii)* the algorithm we choose to classify acceleration data is working properly; *(iii)* our idea of a self-training sensor ecosystem is feasible and could help to reduce resources needed to deploy near-coming smart cameras and sensor networks infrastructures.

5. RELATED WORKS

In [9] was first proven that under certain condition is possible to exploit classification labels coming from a classifier to train another one. Since unlabeled samples can be obtained signif-

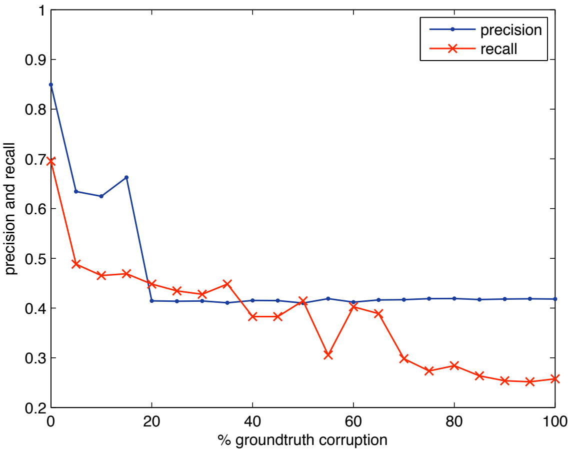

Fig. 4. Graph showing accelerometer classification performance with respect to the camera classification errors.

icantly easier than labeled samples the main goal would be to take advantage of the statistical properties of the labeled and unlabeled data by using a semi-supervised approach. Hence, a seed classifier, that was trained from a smaller number of labeled samples, can be improved by taking into account a large number of available unlabeled samples.

The work described in [10] presents a multi-modal, multi-sensor classification system using body-worn microphones and accelerometers. Sensors' outputs are independently classified and their values are merged together using different kinds of information fusion mechanisms. Other than the choice of sensors being used (we use cameras instead of microphones), the main conceptual difference between this and our work is that we combine sensors also in the learning phase, following the cooperative learning paradigm described above. Moreover, our system has been designed to allow also the individual sensors (especially the accelerometer) to work in isolation.

Another interesting work in the direction of multi-modal multi-sensor motion classification is presented in [11]. This work combines accelerometer-based classification with data coming from a RFID-glove reporting information about the (RFID tagged) objects touched by the user (e.g., if the user is touching a tooth-brush, it is unlikely that he is running). The main difference with our proposal is that while this work focuses only on the classification algorithms we propose a general framework not tied with specific solutions.

Although the general idea of the training sensor ecosystem is not bounded to specific pattern recognition algorithms. It is worth reporting some works at the state of the art in this area. In fact, our proposal is intended to let these algorithms to cooperate with each other seamlessly.

The analysis of human movements has been a very active research area in the computer vision community. Gavrila in [12] surveyed the existing approaches to whole-body or hand motion analysis. Generally speaking, most of the reported approaches focus on specific human motions and are heavily based on a model used for the system training. For instance, Yam *et al*. [13] proposed a system for people identification based on the analysis of their gait. The system extracts leg motion by temporal template matching in which the periodic motion of leg is used as a model. Fourier analysis is employed to analyze the periodicity of the leg motion. However, this approach can only work on pure lateral views and is basically capable to distinguish only between walking and running.

Urtasun and Fua [14] exploited a more general model and used temporal motion models based on PCA to formulate the human body tracking problem as one of minimizing differentiable objective functions. Moreover, a multi-activity database is accessed to compare extracted features with a theoretically infinite set of human motion models.

Ultimately, recent advances in statistical pattern recognition and computer vision techniques contribute to a new era of research on human motion understanding, as demonstrated by the quite recent special issue on Computer Vision and Image Understanding journal [15]. As an example among the many existing proposals, Robertson and Reid [16] model human behaviors as a stochastic sequence of actions and evaluate the likelihood by using HMM. The observations come from position, velocity and motion descriptors and matching is performed against a labeled database of actions.

6. CONCLUSIONS

In this paper we presented a novel approach to promote unsupervised training processes among different sensors. This technology allows different sensors to exchange the needed knowledge to create a model to classify events happening into the environment. This allows pervasive information and communication systems using the model to autonomously adapt to highly dynamic and open environments. We described also an use case of this approach combining data from a camera and body-worn accelerometer. In particular the accelerometer is trained by the data coming from the camera to recognize four user motion states.

7. REFERENCES

[1] T. Abdelzaher, Y. Anokwa, P. Boda, J. Burke, D. Estrin, L. Guibas, A. Kansal, and S. Madden, “Mobiscopes for human spaces,” *IEEE Pervasive Computing*, vol. 6, no. 2, pp. 20 – 29, 2007.

[2] R. Cucchiara, A. Prati, and R. Vezzani, “A multi-camera vision system for fall detection and alarm generation,” *Expert Systems Journal*, vol. 24, no. 5, pp. 334–345, 2007.

[3] M. Paskin, C. Guestrin, and Jim McFadden, “A robust architecture for inference in sensor networks,” in

International Symposium on Information Processing in Sensor Networks. ACM Press, Los Angeles (CA), USA, 2005.

[4] D. Patterson, L. Liao, K. Gajos, M. Collier, N. Livic, K. Olson, S. Wang, D. Fox, and H. Kautz, “Opportunity knocks: a system to provide cognitive assistance with transportation services,” in *International Conference on Ubiquitous Computing*. ACM Press, Nottingham, UK, 2004.

[5] R. Cucchiara, C. Grana, A. Prati, and R. Vezzani, “Probabilistic posture classification for human behaviour analysis,” *IEEE Transactions on Systems, Man, and Cybernetics, Part A: Systems and Humans*, vol. 35, no. 1, pp. 42–54, Jan. 2005.

[6] R. W. DeVaul and S. Pentland, “The mithril real-time context engine and activity classification,” Technical Report, MIT Media Lab, 2003.

[7] R. Cucchiara, C. Grana, M. Piccardi, and A. Prati, “Detecting moving objects, ghosts and shadows in video streams,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 10, pp. 1337–1342, Oct. 2003.

[8] R. Cucchiara, C. Grana, G. Tardini, and R. Vezzani, “Probabilistic people tracking for occlusion handling,” in *Proceedings of IAPR International Conference on Pattern Recognition (ICPR 2004)*, Aug. 2004, vol. 1, pp. 132–135.

[9] A. Blum and T. Mitchell, “Combining labeled and unlabeled data with co-training,” in *COLT: Proceedings of the Workshop on Computational Learning Theory, Morgan Kaufmann Publishers*, 1998.

[10] J. Ward, P. Lukowicz, G. Troster, and T. Starner, “Activity recognition of assembly tasks using body-worn microphones and accelerometers,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 10, pp. 1553 – 1567, 2006.

[11] I. Kim, S. Im, E. Hong, S. Ahn, and H. Kim, “Adl classification using triaxial accelerometers and rfid,” in *International Conference on Ubiquitous Computing Convergence Technology*. Beijing, China, 2007.

[12] D. M. Gavrila, “The visual analysis of human movement: A survey,” *Computer Vision and Image Understanding*, vol. 73, no. 1, pp. 82–98, Jan 1999.

[13] C.-Y. Yam, M.S. Nixon, and J.N. Carter, “On the relationship of human walking and running: Automatic person identification by gait,” in *ICPR (1)*, 2002, pp. 287–290.

[14] R. Urtasun and P. Fua, “3d human body tracking using deterministic temporal motion models,” in *ECCV (3)*, 2004, pp. 92–106.

[15] A. Hilton, P. Fua, and R. Ronfard, “Foreword: Special issue on modeling people: Vision-based understanding of a persons shape, appearance, movement, and behaviour,” *Computer Vision and Image Understanding*, vol. 104, no. 2-3, pp. 87–89, 2006.

[16] N. Robertson and I. Reid, “A general method for human activity recognition in video,” *Computer Vision and Image Understanding*, vol. 104, no. 2-3, pp. 232–248, 2006.