A robust method for automatic player detection in sport videos

A. Lehuger[1] S. Duffner[1] C. Garcia[1]

[1] Orange Labs

4, rue du clos courtel, 35512 Cesson-Sévigné

{antoine.lehuger, stefan.duffner, christophe.garcia}@orange-ftgroup.com

Abstract

In the last years, pattern recognition approaches using machine learning techniques to construct object detectors have encountered great success in the domain of visual detection. However, the application of automatic and robust player detection in sport videos is a particularly challenging task because of the small size, and high variability in shape and appearance of the objects to detect, and the influence of noise due to motion blur and video coding. In this paper, we present a sport player detection method based on convolutional neural networks and compare it to the well-known Adaboost approach which is known to perform well in other contexts. Compared to traditional methods, these approaches do not use background substraction after camera motion estimation. Experiments showed that both methods are very robust and effective and that convolutional neural networks perform better with an average gain of 4% on various datasets.

Key words

Convolutional Neural Networks, Adaboost, Automatic Player Detection.

1 Introduction

Since a few years, pattern recognition approaches using machine learning techniques to construct a detector from a large training set have encountered great success in the domain of visual detection [8, 4, 5, 1, 3]. For instance, these techniques are very reliable for faces and cars. In the specific case of detection of players on a football pitch, automatic systems face three challenges : first, targets may undergo drastic changes of appearance due to pose changes, deformations, etc. Secondly the resolution of the images is very low, the target object may only be represented by 200 pixels. Finally, the movement of the camera may cause important blurring effects.

Eventhough improvement of pedestrian detection using motion information [7] is of valuable interest, we choose to consider pure static methods to make our evaluations. Thus this paper describes a player detection system based on neural networks and compares the obtained results with the robust real-time object detector of Viola and Jones [8].

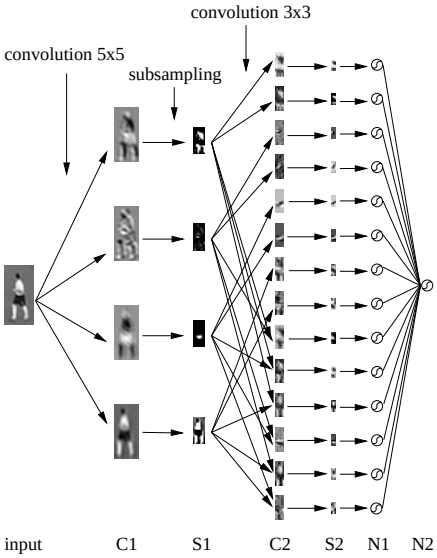

Figure 1 – *The convolutional architecture*

2 System Architecture

The convolutional neural network, shown in Fig. 1, consists of a set of three different kinds of layers. Layers Ci are called convolutional layers, which contain a certain number of planes. Layer C1 is connected to the retina of size 21x43 which receives the image area to classify as player or non-player. Each unit in a plane receives its input from a small neighborhood (biological local receptive field) inside the planes of the preceding layer. The trainable weights (or convolutional mask) forming the receptive field for a plane are forced to be equal for all units in the plane (weight sharing). Each plane can be considered as a feature map that has a fixed feature detector that corresponds to a pure convolution with a trainable mask, applied over the planes in the preceding layer. Finally, a trainable bias is added to the results of each convolutional mask.

Each convolutional layer Ci is typically followed by a subsampling layer Si that performs a local averaging over a neighborhood of four inputs followed by a multiplication by a trainable coefficient and the addition of a trainable bias. This subsampling operation reduces by two the di-

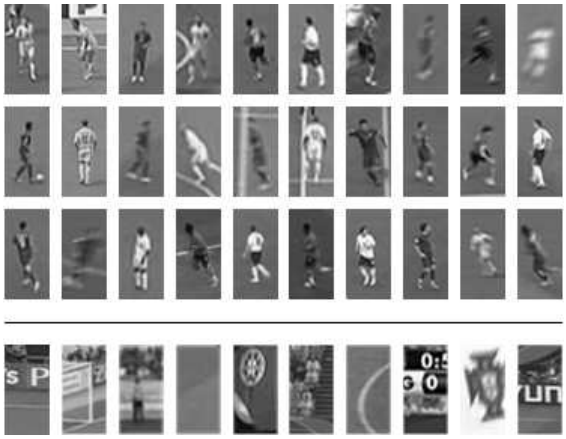

Figure 2 – *Some samples of the training set. The last row shows initial negative examples.*

mensionality of the input and increases the degrees of invariance to translation, scale, and deformation of the learnt patterns.

Layers C1 and C2 perform convolutions with trainable masks of dimension 5x5 and 3x3 respectively. Layer C1 contains four feature maps and therefore performs four convolutions on the input image. Layers S1 and C2 are partially connected. Mixing the outputs of feature maps helps in combining different features, thus in extracting more complex information. In our system, layer C2 has 14 feature maps. Each of the four subsampled feature maps of S1 is convolved by two different trainable masks 3x3, providing eight feature maps in C2. The other six feature maps of C2 are obtained by fusing the results of two convolutions on each possible pair of feature maps of S1. Layers N1 and N2 contain simple sigmoid neurons. After the preceding steps of feature extraction and input dimensionality reduction the role of these layers is to perform classification. In layer N1, each neuron is fully connected to exactly one feature map of layer S2. The unique neuron of layer N2 is fully connected to all the neurons of the layer N1. The output of this neuron is used to classify the input image as player or non-player. For training the network, we used the classical backpropagation algorithm with momentum modified for being used in convolutional networks as described in [2]. Desired responses are set to -1 for non-player and to +1 for player.

3 Training Methodology

The system employs a boot-strapping strategy [6] where the set of negative training examples is iteratively augmented by applying the neural network to images that do not contain players and subsequently extracting the produced false alarms. The procedure is detailed in table 1. In step 1, a validation set is built and used for testing the generalization ability of the network during learning and, finally, selecting the weight configuration that performs best on it. This validation set is kept constant through all

1. Create a validation set of 50 player images and 50 non-player images randomly extracted and excluded from the initial training set. It will be used to choose the best performing weight configuration during steps 3 and 8.
2. Set $BIter = 0$, $ThrFa = 0.8$.
3. Train the network for 60 learning epochs. Use an equal number of positive and negative examples in each epoch. Set $BIter = BIter + 1$.
4. Gather false alarms from a set of 100 video frames with network answers above *ThrFa*. Collect at maximum $2,000$ new examples.
5. Concatenate the newly created examples to the non-player training set.
6. If $ThrFa \geq 0.2$ set $ThrFa = ThrFa - 0.2$.
7. If $BIter < 6$ go to step 3.
8. Train the network for 60 more learning epochs and exit.

Tableau 1 – *The proposed bootstrapping scheme.*

the bootstrapping iterations, in contrast to the training set which is updated. In step 3, the backpropagation algorithm is used with the addition of a momentum term for neurons belonging to the N1 and N2 layers. Stochastic learning was preferred versus batch learning. For each learning epoch, an equal number of examples from both classes are presented to the network giving no bias toward one of the two classes.

The generation of the new patterns that will be added to the non-player training set is carried out by step 4. The false alarms produced in this step force the network, in the next iteration, to refine its current decision boundary for the player class. At each iteration, the false alarms, giving network answers greater than *ThrFa*, and therefore strongly misclassified, are selected. As the network generalizes from these examples, *ThrFa* is gradually reduced until reaching 0. In this way, some redundancy is avoided in the training set. The learning process is stopped after six iterations, when convergence is noticed, i.e. when the number of false alarms remains roughly constant. This procedure helps in correcting problems arising in the original algorithm proposed in [6] where false alarms were grabbed regardless of the strength of the network answers. Finally, the controlled bootstrapping process added around $8,000$ non-player examples to the training set.

4 Player Localization

Fig. 3. depicts the process of player localization. In order

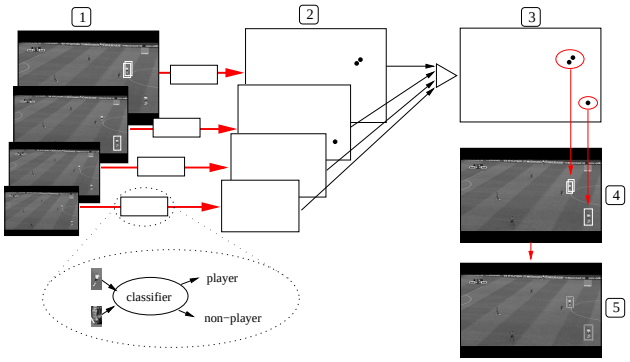

Figure 3 – *Multi-scale player localization*

to detect player patterns of different sizes, the input image is repeatedly subsampled by a factor of 1.2, resulting in a pyramid of images (step 1).
As mentioned earlier, each image of the pyramid is entirely convolved at once by the network. For each image of the pyramid, an image containing the network results is obtained (step 2).
After the processing by this detection pipeline, player candidates (pixels with positive values in the result image) in each scale are mapped to the scale of the input image (step 3). They are then grouped according to their proximity in image and scale spaces. Each group of player candidates is fused into a representative player whose center and size are computed as the centroids of the centers and sizes of each candidate, weighted by their individual network responses. After applying this grouping algorithm, the set of remaining representative player candidates serve as a basis for the next stage of the algorithm, in charge of fine player localization and eventually false alarm dismissal.
To do so, a local search procedure is performed in an area around each player candidate center in image scale-space (step 4). A reduced search space centered at the player candidate position is defined in image scale-space for precise localization of the player candidate. It corresponds to a small pyramid centered at the player candidate center position covering ten equally distant scales varying from 0.8 to 1.5 times the scale of the player candidate. For every scale, the presence of a player is evaluated on a rescaled grid of 16×16 pixels around the corresponding player candidate center position. We observed that true player usually give a significant number of high positive responses in consecutive scales, which is not often the case for non-players. In order to discriminate true players from false alarms, it resulted efficient to take into account both number and values of positive answers. We therefore consider the volume of positive answers (the sum of positive answer values) in the local pyramid in order to take the classification decision. Based on the experiments described in the next section, a player candidate is classified as player if its corresponding volume is greater than a given threshold *ThrVol* (step 5). The bottom-right image of Fig.3 shows the position and size of the detected player after local search.

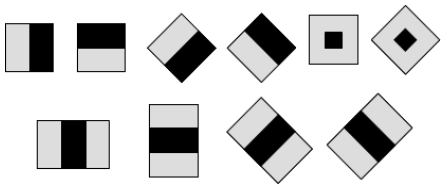

Figure 4 – *Haar-like features are the input to the basic classifers for the Adaboost-based approach.*

5 The Adaboost-based approach

The well-known AdaBoost algorithm aims at constructing a ”strong” classifier as a linear combination of simple ”weak” classifiers.

The feature used in a particular weak classifier (Fig.4) is specified by its shape, position within the region of interest and the scale. The sums of pixel values over a rectangular regions are calculated rapidly using integral images.

The training process uses AdaBoost to select a subset of features and construct the classifier. The learning algorithm chooses from a heterogenous set of filters in each round. The AdaBoost algorithm also picks the optimal threshold for each feature. Each round of AdaBoost chooses from the total set of the appearance features, the feature with lowest weighted error on the training examples. The resulting classifier uses intensity information in order to maximize detection rates. Viola and Jones [8] showed that a single classifier for face detection would require too many features and thus be too slow for real time operation. They proposed a cascade architecture to make the detector efficient (see Fig.5). Each classifier in the cascade is trained to achieve very high detection rates, and law false positive rates. Stages of the cascade are added until the overall target for false positive and detection rate is met.

A 21 layer cascaded classifier was trained to make the generic player detector. Each stage of the cascade was a boosted classifier trained using a set of 1850 positive examples and 1850 negative examples. Each positive training example was scaled and aligned to a base resolution of 12 by 24 pixels taken from the video sequences. Negative examples are extracted from images which do not contain players. Positive and negative examples are shown in Fig.2.

The detection threshold of each newly added classifier is adjusted so that the false negative rate is very low. Validation is performed using full images which contain marked positive examples. The threshold of the newly added classifier is set so that at least 99.5% of the players that were correctly detected after the last stage are still correctly detected. The cascade training algorithm also requires a large set of images to scan for false positives. These false positives form the negative training examples for the subsequent stages of the cascade. We use a set of 3500 full images which do not contain pedestrians for this purpose.

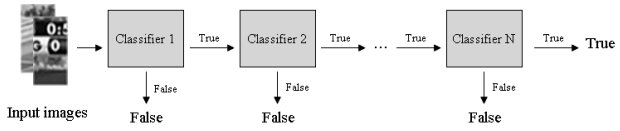

Figure 5 – *Cascade architecture. Input is passed to the first classifier with decides true or false (player or not player). A false determination halts further computation and causes the detector to return false. A true determination passes the input along to the next classifier in the cascade.*

6 Experimental Results

In order to evaluate the performance of the proposed method and to compare it to the Adaboost-based approach we conducted several different experiments on the detection of football players in gray-scale sport videos. The videos used to train and test the systems are recordings from a mobile camera with fixed position (see Fig.6).

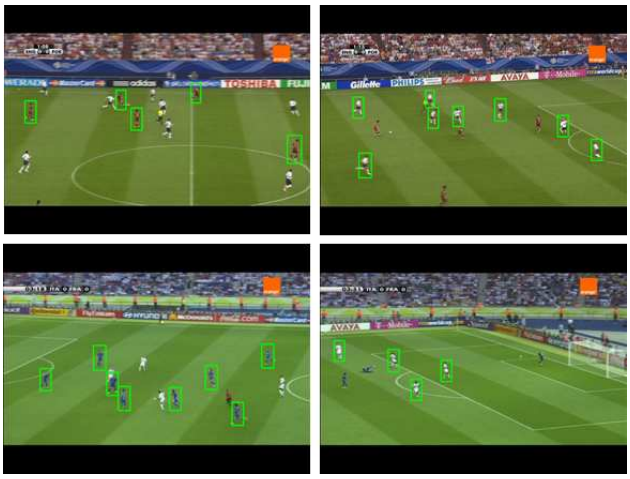

Figure 6 – *One sample frame from each sequence we used for training. The manually green marked boxes over players are extracted to construct the training set.*

The final detector is scanned across the image at multiple scales and locations. Scaling is achieved by scaling the detector itself, rather than scaling the image. This process makes sense because the features can be evaluated at any scale with the same cost. Shown results were obtained using a set of scales a factor of 1.05 apart.
Two types of experiments have been conducted.

6.1 Specific team players detectors

In the first experiment we extracted examples of players from two different football matches. Then, four independant detectors, one for each team, were trained and tested on the respective team. Note that each team wears a different color thus leading to different levels of contrast compared to the background. Each detector was trained using a set of 250 positive examples. During training, the only pre-treatment made is the initial rescaling to satisfy the required intial conditions. Some results for the static case are shown in Fig. 7. The rate of false detection FA is easily calculated as follows :

$$FA = \frac{\text{Total of False Alarms}}{\text{Total of expected detections}} \quad (1)$$

	Neural Network		Viola and Jones	
Team	Detection	False D.	Detection	False D.
Eng	98.88%	0%	94.38%	2.25%
Por	94.25%	1.15%	89.66%	0%
Fra	87.91%	1.10%	74.72%	8.79%
Ita	84.47%	1.94%	83.50%	0%
Total	91.08%	1.19%	85.41%	2.70%

Tableau 2 – *Detections and false alarms calculated frames taken from 6000 frame sequences of England-Portugal and Italy-France FIFA World Cup 2007 matches. One for each team was trained.*

Method	Detection	False detection
Neural Network	81.50%	1.73%
Viola and Jones	78.03%	1.73%

Tableau 3 – *Detections and false alarms with the generic player detector.*

Table 2 lists the good detection and false alarm rates for our system as well as for the Adaboost system. Convolutional neural networks performs better, especially for the ”Fra” Test. The adaboost method shows some difficulties to distiguish all white players and goal posts in the penalty area.

6.2 Generic player detector

In the second experiment, we extracted about 1850 examples from eight different matches to make a single detector for every match. We took care of considering very heterogeneous conditions to construct this training set (afternoon and night matches, different colors of shirts, different stadiums and distances from the camera to the pitch).

Table 3 lists the detection and false detection rate for our system as well as for the Adaboost system. Experiments show that convolutional neural networks perform better when trained on an heterogeneous dataset. Results are presented for the same low level of false detection which is an important criteria to couple detection with a tracking method.

7 Conclusions

We have presented a sport player detection method based on convolutional neural networks and compared it to the well-known Adaboost approach which is known to perform well in other contexts. Compared to traditional methods, these approaches do not use background substraction after

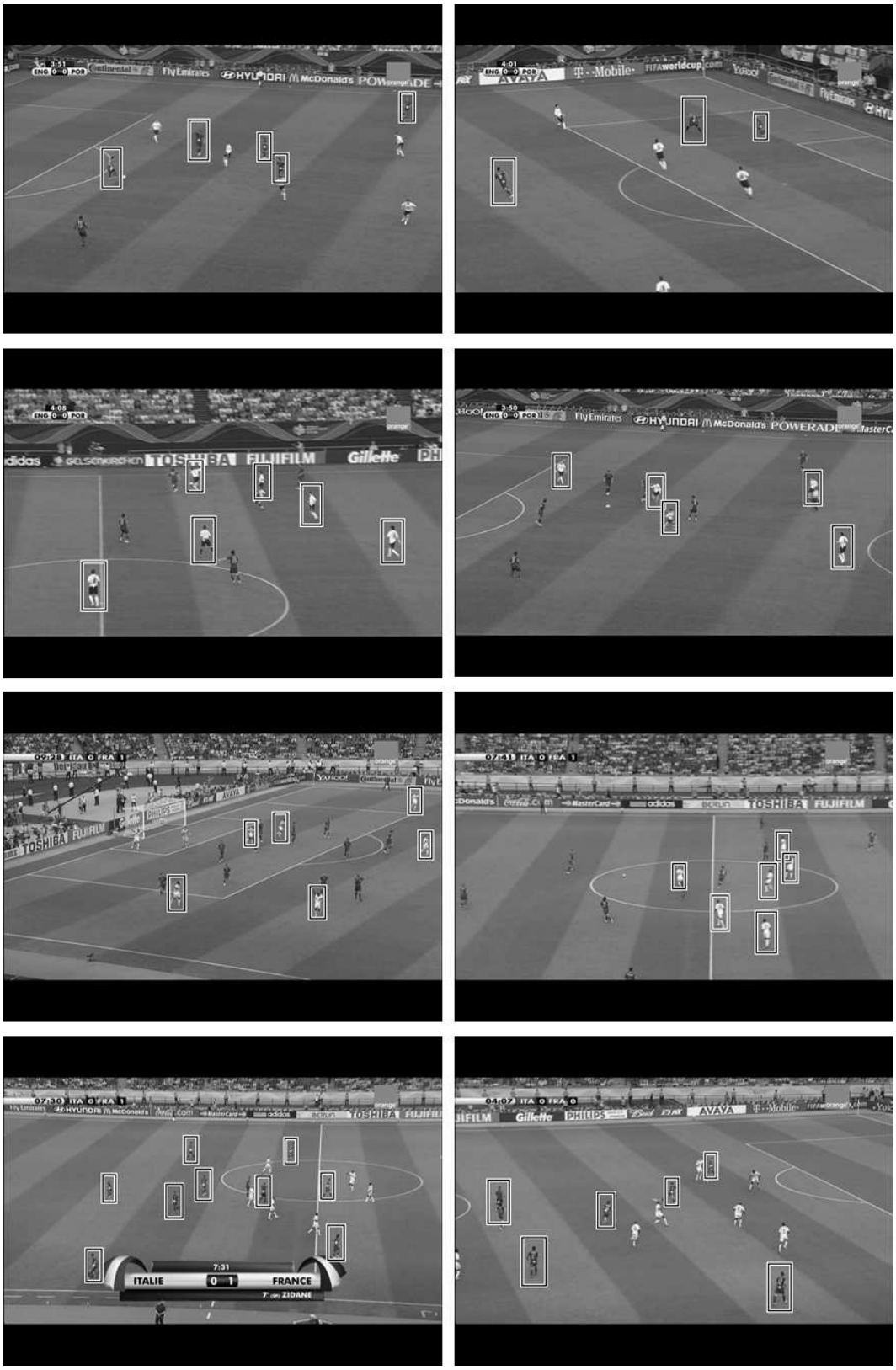

Figure 7 – *Examples of detections and non-detections taken from 6000 frame sequences of England-Portugal and Italy-France FIFA World Cup 2007 matches. Four specific convolutional neural networks detectors were trained (one for each team).*

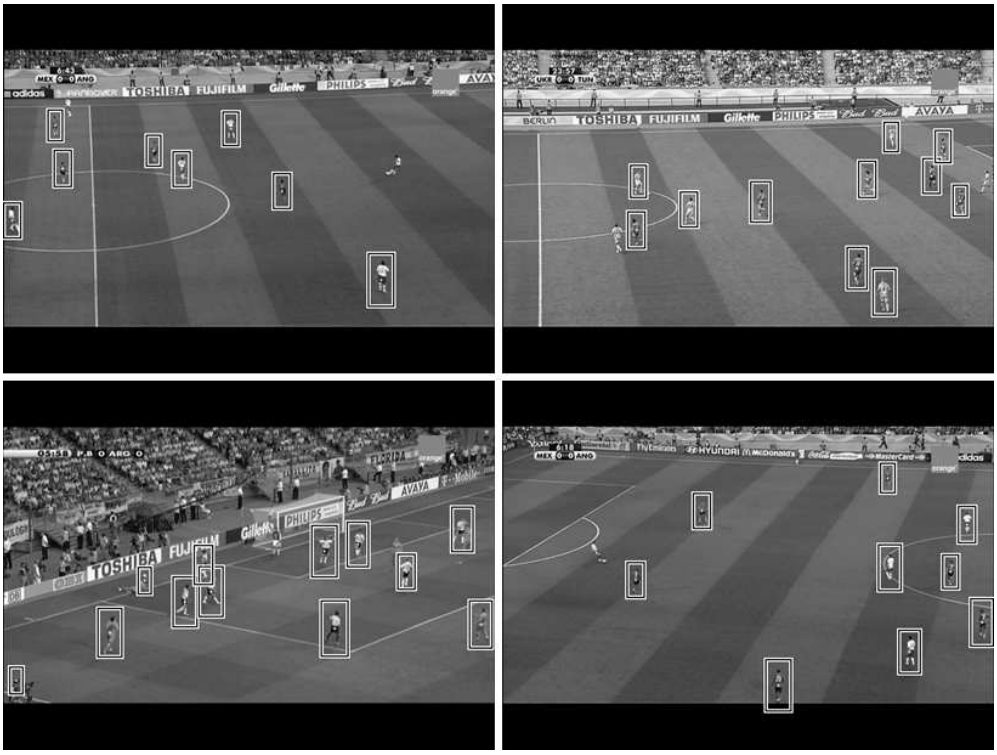

Figure 8 – *Examples of detections, non-detections and false alarms taken from 6000 frame sequences of several FIFA World Cup 2007 matches. One unique detector was trained on 1850 examples from eight other different matches.*

camera motion estimation. Experiments showed that both methods are very robust and effective but convolutional neural networks perform better on all datasets. The proposed system perform a cascade of convolution and subsampling operation, that are easily implemented via image pipeline, and thus allows to detect in less than 100 ms all players at different scales in a 720 x 540 pixel image on a 3.0 GHz P4 processor. Using optimized image processing routines, this can be further improved in a staightforward way, to reach real time processing. As an extension, the idea of building efficient detectors that combine both motion and appearance cues will be applicable to our method as well and probably improve the results.

References

[1] C. Garcia and M. Delakis. Convolutional face finder: A neural architecture for fast and robust face detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(11):1408–1423, 2004.

[2] Y. LeCun, L. Bottou, , Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[3] C. Papageorqiou, , M. Oren, , and T. Poggio. A general framework for object detection, 1998. Proc. Intl. Conference on Computer Vision.

[4] H. Rowley, S. Baluja, and T. Kanade. Neural network-based face detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(1):23–38, 1998.

[5] H. Schneiderman, , and T. Kanade. A statistical method for 3d object detection applied to faces and cars, 2000. Proc. Intl. Conference on Computer Vision.

[6] K. Sung and T. Poggio. Example-based learning for view-based human face detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(1):39–51, 1998.

[7] P. Viola, , M. J. Jones, , and D. Snow. Detecting pedestrians using patterns of motion and appearance, 2005. Proc. Intl. Journal of Computer Vision.

[8] P. Viola and M. J. Jones. Rapid object detection using a boosted cascade of simple features, 2001. Proc. Intl. Conference on Computer Vision and Pattern Recognition.