VISUAL DETECTION OF LEGGED ROBOTS AND ITS APPLICATION TO ROBOT SOCCER PLAYING AND REFEREEING

JAVIER RUIZ DEL SOLAR, MATIAS ARENAS, RODRIGO VERSCHAE, PATRICIO LONCOMILLA
Department of Electrical Engineering
Universidad de Chile

The visual detection of robots is a difficult but relevant problem in several robotic applications. In the present article, a framework for the robust and fast visual detection of legged-robots is proposed. This framework uses cascades of nested classifiers, the Adaboost boosting algorithm, and domain-partitioning based weak classifiers. Using the proposed framework, frontal, profile and back detectors for AIBO robots (model ERS7), as well as detectors for humanoid robots, are built. The detection rate of the obtained systems is quite high: 90% with an average of 0.1 false positives per image, when the final detections are filtered out using context information (horizon line). In addition, a robot referee that uses these detectors to track players during a soccer game is described. Experiment results showed that the referee achieves very high robot detection rates (98.7% DR with ~1 false detection every 16 images), and fast processing speed.

1. Introduction

Legged robots, and in particular humanoid and four-legged robots (e.g. SONY AIBO robots), are increasingly being used as toys, domestic-work robots, and as experimental platforms to study autonomy and distributed decision-making in teams of mobile robots (e.g. robot soccer). One of the main challenges in legged-robot vision is the development of robust and high-performing vision systems that can operate in dynamic environments (e.g. variable illumination with specular and other reflections as well as shadows, cluttered backgrounds, partial occlusions, and variability in the objects pose), in real-time, and using limited processing power. Limited processing power is an important constraint of processing boards used in standard legged robots (humanoid robots, AIBO robots, etc.). Algorithmic complexity is therefore constrained; there is a trade-off between performance and processing time.

One of the basic skills that legged robots should be equipped with is visual interaction with the environment, including the detection of moving objects such as humans, pets and other robots, which in general terms means deformable objects that can be seen at different distances and viewpoints. For instance, in robot soccer scenarios, the detection of teammates and opponent robots is a key skill needed for good playing. However, the detection of deformable objects, in particular of legged robots, is a difficult task because of the limited processing power, the changing environmental conditions and the changing appearance of the robots, which depends on their relative pose. For instance, most of the systems developed in the robot soccer community are not robust enough in the detection of other players, because they are based on pure color analysis, which is highly dependent on the existing illumination.

In this context, we propose a framework for the robust and fast visual detection of legged-robots which uses nested cascades of classifiers,[1] the Adaboost boosting

algorithm,[2] and domain-partitioning based classifiers,[2] and is based on a face detection framework.[3] Frontal, profile and back detectors for AIBO robots (model ERS7), as well as a humanoid robot detector, are built using the proposed framework. The main strengths of the developed robot detection systems are: the ability to work at multiple scales, the capability of detecting robots at low-resolutions (starting from 24x24 pixels), being illumination invariant to a larger degree (they work in grey scale images and no preprocessing is needed for photometric normalization), and being real-time.

In addition, we have developed a robot referee that uses these detectors to track players during a soccer game. This application is a new extension of the concept of robot soccer, and it is useful for the further testing of the application of our robot detection framework in different situations. The robot referee is specially intended to be used in the RoboCup SPL 2-legged league,[4] and in the RoboCup humanoid league.[5]

This paper corresponds to an extended version of [6,7], in which the robot detection framework and the referee were proposed. This extended version includes a more complete explanation of the proposed framework and the training procedure, a better characterization of the final detectors, the use of context information to reduce false detections, and a comparison with alternative detection methods. The paper is organized as follows. In section 2 state of the art methodologies for robot detection are analyzed. The proposed robot detection framework is presented in section 3. In section 4, the building and validation of detectors of legged robots, to be used in robot soccer applications, are described. The application of the developed detectors in the robot referee is described in section 5. Finally, some conclusions and projections of this work are given in section 6.

2. Related Work

One of the main challenges in legged-robot vision is the development of robust and high-performing vision systems that can operate in dynamic environments, in real-time, and using limited processing power. In many cases, the requirement of having an anthropomorphic body imposes constraints on the sensors that can be used.

In robot soccer most approaches for detecting robots are based on pure color segmentation and on the detection of contrast changes using scan lines (see for example [8,9]). These simple approaches are not robust enough; they are highly dependent on the illumination and background. In [10] a detection system for AIBO robots based on the use of local image descriptors and SIFT features is proposed, but its main limitations are its low processing speed and its reduced performance when highlights are present in the image, which are common in AIBO robots. In [11] , a PCA-SIFT approach is proposed for the detection of deformable robots. The method uses PCA-reduced SIFT descriptors and a voting schema to cluster the descriptors. However, obtained results are just moderate (~77% detection rate with 25% false positive rates for the AIBO case). In [12] a tracking system using 3D shape and color object modeling is used to track a robot with a

catadioptric sensor (omni-directional camera). These sensors are usually used for localization and path planning, thanks to their wide field of vision. However, anthropomorphic body requirements do not allow using this kind of sensors in current humanoid soccer applications (e.g., RoboCup SPL[4] and humanoid leagues[5]).

Robot detection using statistical classifiers is an interesting methodology that has not been sufficiently explored. One of the drawbacks of detection systems based on statistical classifiers is that they are not real-time. The systems based on cascades of boosted classifiers, however, are an exception; they are very fast and accurate at the same time. The Viola & Jones classifier[13] uses a cascade of filters for fast classification. Each filter is trained using Adaboost, and the integral image is used for fast computation of simple, rectangular features (a kind of Haar wavelets). This kind of classifiers allows obtaining fast processing speed and high detection rates. In [14] a combination between color segmentation and zonal grayscale detection with a very restricted number of rectangular features (no cascade classifier) is used to detect AIBO robots, obtaining promising results (the system performance was evaluated in a 3 GHz Pentium 4 processor, not in the robots). In [15] a detection system based on Adaboost and rectangular classifiers for a Human-Robot interaction application is proposed. This application shows acceptable detection rates (robot correctly detected 79% of the time during an evaluation video sequence), but it works only with Black ERS-7 AIBO robots, and can only detect one robot at a time.

The detection framework proposed here is based on the use of nested cascades of boosted classifiers, which have shown better results than standard cascades in detection problems.[1,3] Nested cascades reuse the confidence output of a given layer in the next layer of the cascade, which allows obtaining more compact (faster) cascades and more accurate classifications. In addition, the proposed framework uses domain-partitioning weak classifiers,[2] which, compared to the standard classifiers used in the Viola & Jones work,[13] achieve an improvement in the representation power of the weak classifiers, and reduce the processing and training time.

The use of automated referees and commentators in robot soccer is rather new. In the RoboCup 2006 World Competition the ideas of using automated referees [16] and robot commentators [17,18] for the AIBO robot soccer games were simultaneously proposed. The refereeing tool proposed in [16] is able to comment on and to referee a game, and is further extended in [7] for the case of humanoid robots.

3. Robot Detection Framework

In this section we describe the multiscale robot detection framework (see block diagram in figure 1). First, to detect the robots at different scales, a multiresolution analysis of the images is performed by downscaling the input image by a fixed scaling factor --e.g. 1.2-- (*Multiresolution Analysis* module). This scaling is performed until images of about 24x24 pixels are obtained. Afterwards, windows of 24x24 pixels are extracted in the

Window Extraction module for each of the scaled versions of the input image. The extracted windows could then be pre-processed to obtain invariance against changing illumination, but thanks to the used features we do not perform any kind of preprocessing.

Afterwards, the windows are analyzed by the nested cascade classifier (*Cascade Classification Module*). Finally, in the *Overlapping Detection Processing* module, the windows classified as positive (they contain a robot) are fused (normally a robot will be detected at different scales and positions) to obtain the size and position of the final detections. This fusion procedure is described in.[19]

Using the described framework it is also possible to detect the robots' main orientation. To achieve this, detectors tuned to different robot orientations/views (e.g. frontal, profile and back) should be trained and applied in parallel. Then, in the *Pose Classification* module, the final robot's main orientation is given by the detector having the largest confidence value.

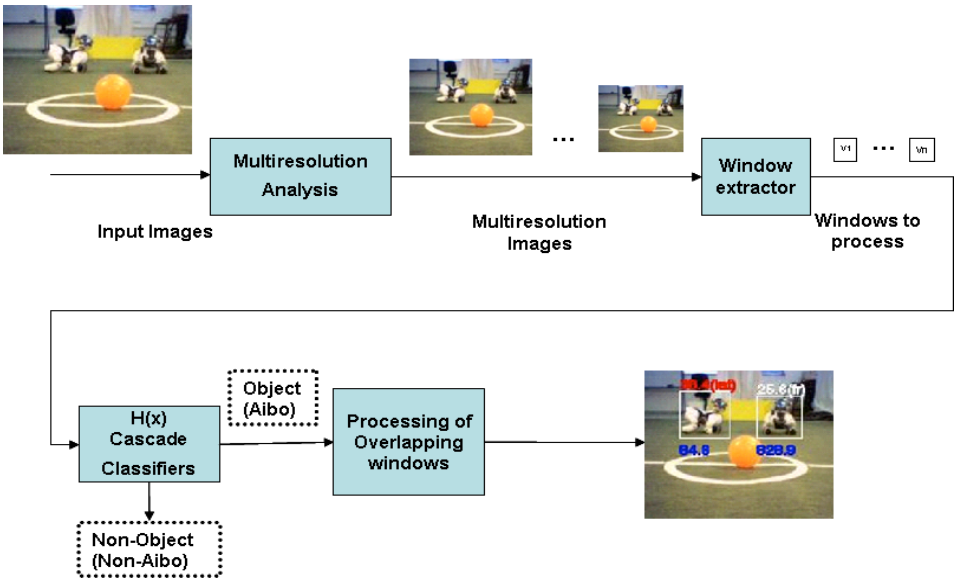


Fig. 1. Block diagram of the robot detection framework.

3.1 *Learning using nested cascades of classifiers*

The key concepts used in the considered framework are nested cascades, boosting, and domain partitioning classifiers. Cascade classifiers consist of several layers (stages) of classifiers of increasing complexity to obtain fast processing speed together with high accuracy. Nested cascades allow high classification accuracy and higher processing speed by reusing in each layer the confidence given by its predecessor. Adaboost [2] (a

Boosting algorithm) is employed to find highly accurate hypotheses (classification rules) by combining several weak hypotheses (classifiers). We use domain partitioning weak hypotheses [2], each one having a moderate accuracy, and giving self-rated confidence values that estimate the reliability of each prediction.

As already mentioned, a nested cascade of boosted classifiers is composed by several integrated (nested) layers, each one containing a boosted classifier. The whole cascade works as a single classifier that integrates the classifiers of every layer. Weak classifiers are linearly combined, obtaining a strong classifier. A nested cascade, composed of *M* layers, is defined as the union of *M* boosted classifiers H_C^k each one defined by:

$$H_C^k(x) = H_C^{k-1}(x) + \sum_{t=1}^{T_k} h_t^k(x) - b_k \qquad (1)$$

with $H_C^0(x) = 0, h_t^k$ the weak classifiers, T_k the number of weak classifiers in layer *k*, and b_k a threshold (bias) value that defines the operation point of the strong classifier. The class assigned to the output corresponds to the sign of *H(x)*. The output of H_C^k is a real value that corresponds to the confidence of the classifier and its computation makes use of the already evaluated confidence value of the previous layer of the cascade (see figure 2).

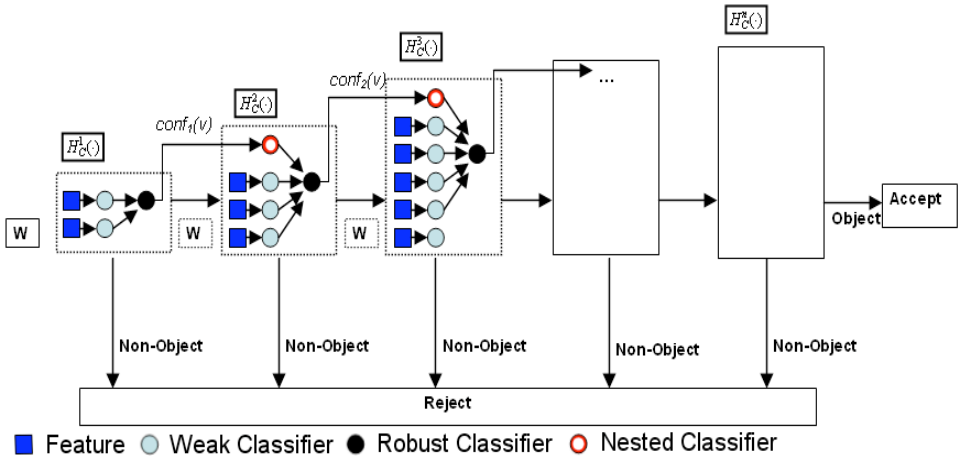


Fig. 2. Nested Cascade of Boosted Classifiers.

3.2 *Design of the strong and weak classifiers*

The weak classifiers are applied over features computed in every pattern to be processed. A single feature is associated to each weak classifier. Domain-partitioning weak hypotheses make their predictions based on a partitioning of the domain *X* into disjoint

blocks $X_1,...,X_n$, which cover all X, and for which $h(x)=h(x')$ for all $x, x' \in X_j$. Thus, the weak classifiers prediction depends only on which block X_j a given sample instance falls into. In our case the weak classifiers are applied over features, therefore each feature domain F is partitioned into disjoint blocks $F_1,...,F_n$, and a weak classifier h will have an output for each partition block of its associated feature f:

$$h(f(x)) = c_j \ni f(x) \in F_j \tag{2}$$

For each classifier, the value associated to each partition block (c_j), i.e. its output, is calculated so that it minimizes a bound of the training error and at the same time a loss function on the margin [2]. This value depends on the number of times that the corresponding feature, computed on the training samples (x_i), fall into this partition block (histograms), and on the class of these samples (y_i) and their weight $D(i)$. To minimize the training error and the loss function, c_j is set to:

$$c_j = \frac{1}{2}\ln\left(\frac{W_{+1}^j + \varepsilon}{W_{-1}^j + \varepsilon}\right), \quad W_l^j = \sum_{i:f(x_i)\in F_j \wedge y_i = l} D(i) = \Pr\left[f(x_i) \in F_j \wedge y_i = l\right], \text{ where } l = \pm 1 \tag{3}$$

where ε is a regularization parameter. The outputs, c_j, from each of the weak classifiers, obtained during training, are stored in a LUT to speed up its evaluation. The real Adaboost learning algorithm is employed to select the features and training the weak classifiers $h_t^k(x)$.

The main idea of cascade classifiers is to process most non-object windows as fast as possible, and to process carefully the object windows and the object-like windows. We manage this by setting the maximum allowed False Positive Rate - FPR (*fprMax*) and the minimum allowed True Positive Rate - TPR (*tprMin*) per layer, while the minimum number of features is selected such that *fprMax* and *tprMin* are achieved. For details on the cascade's training algorithm see [3].

3.3. *Selection of the training examples*

Every window of any size in any image that does not contain an object (e.g. an AIBO robot) is a valid non-object training example. Obviously, to include all possible non-object patterns in the training database is not an alternative. To define such a boundary, non-object patterns that look similar to the object should be selected. This is commonly solved using the bootstrap procedure,[20] which corresponds to iteratively train the classifier, each time increasing the negative training set by adding examples of the negative class that were incorrectly classified. When training a cascade classifier the bootstrap can be applied in two different situations: before starting the training of a new layer and for re-training a layer that was just trained. According to our experience, it is important to use bootstrap in both situations. The *external* bootstrap is applied just one time for each layer, before starting its training, while the *internal* bootstrap can be applied several times during the training of the layer. The bootstrap procedure in both

cases is the same with only one difference, before starting an external bootstrap all negative samples collected for the training of the previous layer are discarded (see [3] for details).

4. Building and Validation of Detectors of Legged Robots

4.1. *Training of the AIBO and humanoid robot detectors*

In this section we describe the training of the AIBO and humanoid robot detectors. In the case of the AIBO robots, we built detectors for 3 different main orientations (*Frontal*, *Profile* and *Back*), which are required for the correct detection of these robots. In the case of the humanoid robots, a single detector was built, designed to work with different robot orientations.

During the training of the cascades two sets are used: training and validation. We will explain how the training dataset is obtained; the procedure to generate the validation dataset is analogous. To obtain the training set used at each layer of the cascade classifier, two types of databases are needed. One them consists of cropped windows of positive examples (e.g., frontal AIBOs). The second one consists of images not containing the object to be detected, and it is used during the bootstrap procedure to obtain the negative examples (see section 3).

The training dataset is used to train the weak classifiers using Adaboost, and the validation database is used to decide when to stop the training of a layer, and to select the bias values of the layer (see section 3). To obtain positive examples (cropped windows) the following procedure was employed. First, a rectangle bounding the robot was annotated. Then, a square centered on this rectangle, and of size equal to the largest size of the rectangle was obtained. Finally this square was cropped and downscaled to a 24x24 pixel size. In the case of the humanoid robots, two windows were cropped from each robot used during training, one corresponding to the upper half of the robot (torso and head), and the other to the lower part (mostly legs). This allowed us to obtain a larger number of training examples with high variability. This was also made to allow the detection of either the upper or the lower part of the robot independently (using only one detector). The reason is that during a game a robot will see many times only a part of the other robots in the field, and this information should be sufficient for a successful detection.

In the case of the databases used to train the AIBO detectors, the positive examples were obtained from videos captured using the AIBOs' cameras and using external cameras under real-world playing conditions (variable illumination, occlusions, etc.). The sources used to build the humanoids training and validation sets were videos obtained using the same camera employed in our humanoid robots (Philips ToUCam III - SPC900NC), and videos from other humanoids robots obtained from the RoboCup Humanoid league website.[5] The number of images used in each database is shown in

Table 1. Figure 3 shows some positive examples used to train the *Frontal* and *Profile* AIBO detectors.

Table 1. Summary of the databases used for training. NPE: Number of Positive Examples. NNI: Number of Negative Images.

Class	NPE (training)	NPE (validation)	NNI (training)	NNI (validation)
Frontal AIBOs	3,115	3,115	5,946	2,550
Left AIBOs	4,263	3,624	5,946	2,550
Back AIBOs	1,528	1,528	5,958	2,562
Humanoids	3,506	3,500	5,958	2,562

(a) (b)

Fig. 3. Examples (24x24 pixels) used for training: (a) *Frontal* AIBOs, and (b) *Profile* AIBOs.

The training procedure is an iterative process. Basically, it consists on iteratively adding layers to the cascade, where each layer is also built iteratively by adding weak classifiers. The parameters of this procedure control the trade-off between the detection rate, the false positive rate and the processing time (see [3] for details):

- Maximum FPR per layer (*fprMax*),
- Minimum TPR per layer (*tprMin*),
- Number of bootstrap steps,
- Number of bootstrap examples,
- Initial number of negative examples, and
- Sampling factor: Percentage of features considered for the training of the cascade.

For each classifier the whole training process is repeated from 5 to 10 times. Each time, the classifier's performance and accuracy is evaluated using the validation set, and parameters are adjusted to generate a better detector. The used *sampling factor* stayed between 30% and 40%, affecting principally the speed of the training. The number of bootstrap steps was kept between 3 and 6, but for most of the final detectors, 4 steps were used. Each step of the procedure clears the weak classifiers of the stage being trained, and adds the number of bootstrapped examples (between 300 and 600) to the initial number of negative examples (between 2,000 and 3,000) for the re-training. This helps building a stronger cascade, in particular in its early stages. The *fprMax* per layer was kept between 20% and 50%, and best results were obtained with about 30%. The *tprMin* per layer was tuned between 99.50% and 99.99%, with the best result obtained with 99.90%. The final cascades have between 10 and 12 layers. For all cascades, the first stage has 9 weak classifiers (fast evaluation), while the last one has up to 50 weak classifiers (high complexity and high classification ability).

4.2. *Evaluation of the detectors*

The detection results are presented in terms of Detection Rate (DR) versus Number of False Positives (FP) in the form of ROC curves (Receiver Operation Characteristic curves) and tables, while the robot's orientation estimation results are presented using a confusion matrix. The analysis of the processing speed of the system is also presented.

4.2.1. *Evaluation databases*

To evaluate the proposed system, two databases were constructed: one for the AIBO robots (AIBODetUChileEval) and one for the Humanoid robots (HDetUChileEval). These databases are made available in [21] for future comparisons. No image of the training or validation sets is part of these databases. The AIBODetUChileEval database contains AIBOs in three orientations (frontal, profile, back), while the HDetUChileEval database consists of images containing humanoid robots, which are different robot models than the ones used to train the system. These images are from real-world soccer scenarios, and they include many changes in illumination, contrast, and background. Table 2 contains details on the number of AIBO/humanoid robots in each of these datasets.

Table 2**.** Summary of the database used for evaluation of the AIBO detection system and the Humanoid detection system. NI: Number of Images. NR: Number of robots. IS: Image size.

		AIBOs			Humanoids	
Database	NI	NR (frontal)	NR (profile)	NR (back)	NR	IS
AIBODetUChileEval	724	344	489	180	-	208x160
HDetUChileEval	244	-	-	-	493	640x480

4.2.2. *Detection results*

The performance of the proposed robot detection system was evaluated in the AIBODetUChileEval and the HDetUChileEval databases. These results are presented in terms of the DR versus FP (Figure 4a and Table 3), and percentage of correct robot's orientation classification (Table 4).

In the AIBOs database, the first test consisted in evaluating each detector independently on the specific class it was trained to detect (e.g. *Frontal* detector detecting *Frontal* AIBOs). In this evaluation, AIBOs appearing under views different to the ones being detected were ignored, i.e. they were neither counted as false positives nor as correct detections. As it can be observed in the ROC curves of figure 4a, the best performing AIBO detector is the *Profile* detector, followed by the *Back* detector, and then by the *Frontal* detector. For a DR of ~90%, the *Profile* detector has 70 FP in 724 images, while the *Back* and *Frontal* detectors have 166 and 254 FP, respectively (see

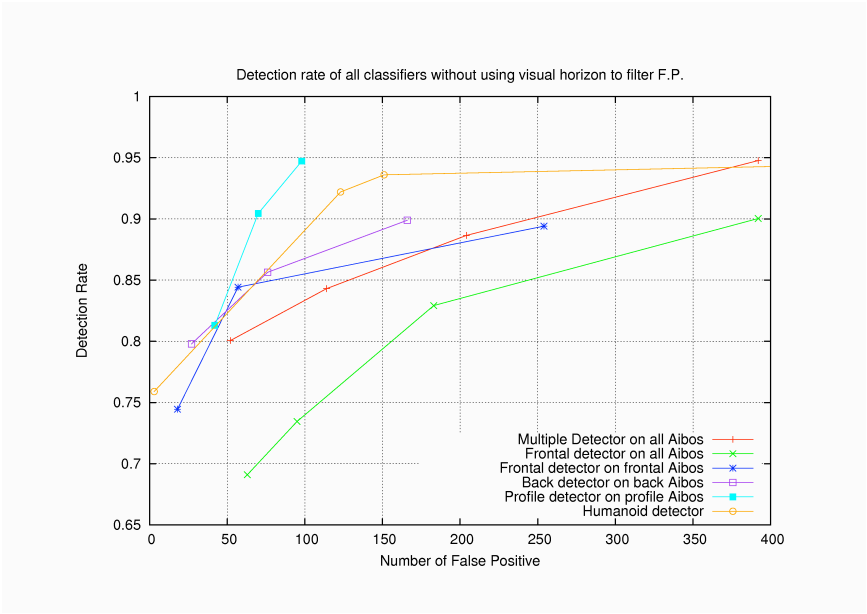
table 3). The profile detector achieves a DR of 94.7% with 98 FP in 724 images, which can be considered reasonably good for real soccer applications.

The second test consisted in evaluating the performance of a particular detector when detecting all robot's orientations, including the ones they were not trained to detect. In this case the detectors were able to detect AIBOs in all orientations, showing a reasonably good detection rate; e.g. the *Frontal* detector obtained a 90 % DR of AIBOs under all orientations with 392 FP. This means that even without training a detector for all orientations, the characteristics of the rectangular features still find a high resemblance between them, mainly due to the contrasts present in Aibo robots.

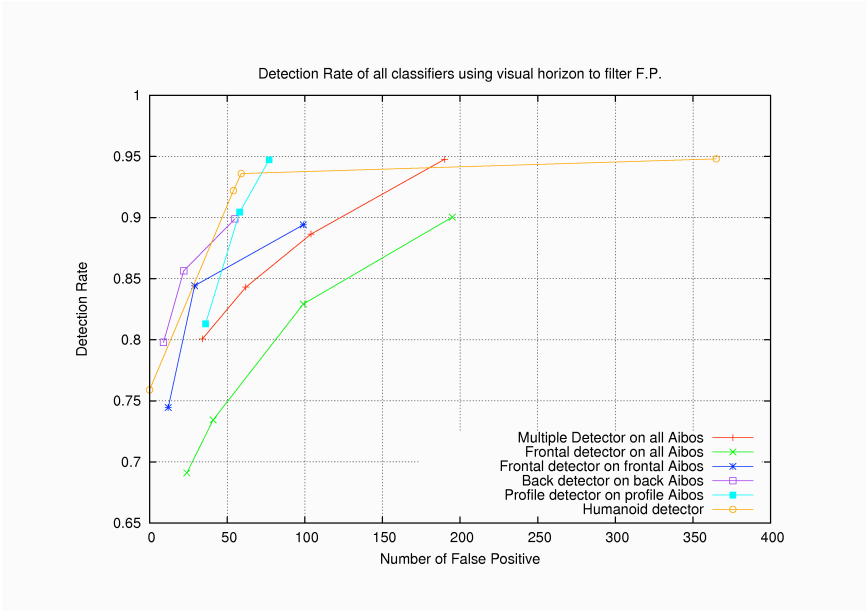
The third test (*Multiple detectors in all AIBOs*) consisted in running all AIBOs detectors (*Frontal*, *Profile* and *Back*) in parallel. Given that in some cases the three detectors detected the same AIBOs, the final detections were obtained by selecting all non-overlapping detections, and merging overlapping detections by choosing the one with the highest confidence. It is important to notice that in this case the number of false positives slightly increased, e.g. a DR of 94.8% was obtained with 392 FP in 724 images. In other words, it is possible to arbitrate among the output of the detectors without increasing considerably the number of FP, although this alternative is about 3 times slower than using the individual detectors.

The purpose of the last AIBO test was to classify the robot´s orientation using the detectors. For this, the *Frontal* detector was used as a generic detector (using the same parameters that obtained a 90% DR with 392 FP), followed by a verification of the detections using the specific *Frontal*, *Profile*, and *Back* detectors. Afterwards, the orientation was estimated by taking the output of the specific detector that gave the largest confidence value. Out of the 1,013 AIBOs, 657 were classified, i.e. the orientation was estimated. Out of them, the orientation was correctly estimated in 519 cases (79% correct classification rate). Table 4 shows the confusion matrix of the robot's orientation estimation for these AIBOs. The *Frontal* and *Profile* classifiers show the best results, classifying correctly ~90% and ~80% of the *Frontal* and *Profile* AIBOs, respectively.

Finally, performance of the humanoid detector was also tested (Figure 4a and Table 3). A 92.2% detection rate was obtained with 123 false positive in a total of 244 images. This is quite high considering that the system was trained using examples corresponding to different humanoid robot models than the ones used in the evaluation. The humanoids have a clear contrast shape that is easier to identify (almost like a black rectangle), although it also makes them easy to be miss classified with other elements in the image. This means that the boundary between Humanoid and non-Humanoid is very thin, and the detector has to be finely tuned so that it does not have too many false positives, while still having a good detection rate.



(a)



(b)

Fig. 4. (a) ROC curves of the detectors in the AIBODetUChileEval and HDetUChileEval databases. (b) Same as (a) but using the visual horizon information to filter false detections. See main text for details.

Table 3. Selected operation points – DR (Detection Rate) versus FP (Number of False Positives) of the evaluated AIBO and humanoid robot detectors. FFP = Filtered False Positives.

Detector / Target	**DR**	**FP**	**FFP**	**DR**	**FP**	**FFP**	**DR**	**FP**	**FFP**	**DR**	**FP**	**FFP**	**DR**	**FP**	**FFP**
Frontal / Frontal AIBOs				89.4	254	99	84.4	57	29				74.5	18	12
Profile / Profile AIBOs	94.7	98	77	90.4	70	58				81.3	42	36			
Back / Back AIBOs				89.9	166	55	85.6	76	22	79.8	27	9			
Frontal / All AIBOs				90.0	392	195				82.9	183	99	73.4	95	41
Multiple / All AIBOs	94.8	392	190	88.6	204	104	84.3	114	62	80.1	52	34			
Humanoids	93.6	151	59	92.2	123	54							75.9	3	0

Table 4. Confusion Matrix of the AIBO´s orientation estimation using the robot detection system.

True Class / Predicted Class	Frontal AIBOs	Profile AIBOs	Back AIBOs
Frontal AIBOs	91.63 %	11.64 %	33.87 %
Profile AIBOs	3.72 %	81.45 %	15.32 %
Back AIBOs	4.65 %	6.92 %	50.81 %

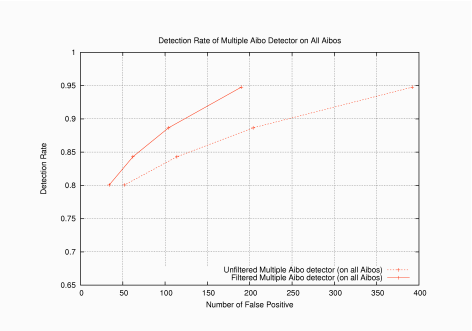
4.2.3 *Detection results using context information*

During the evaluation of the detectors we noticed that many false positives appeared on the top part of the image, where robots are less likely to be since they are always on the ground. Thus, the robot detection results can be improved if context information is used. The robot running the detectors can use the information of their camera pose to filter out false detections. First, the robot compute the horizon line (also called visual horizon) using the camera pose, defined as the intersection between a projection plane P, perpendicular to the optical axis and centered in the focal point, and a horizontal plane H, parallel to the ground, and at the same height that the camera. Second, detection windows whose center is located above the visual horizon of the image are filtered out, because any object appearing in the image above the horizon line means it is located above the ground.

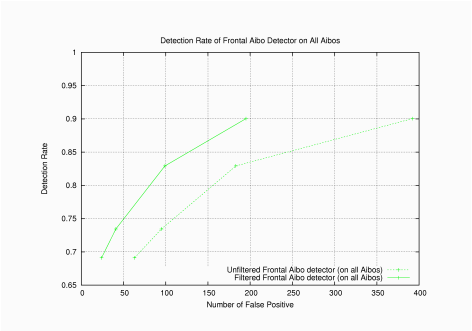
The filtering of false detection using the horizon line was incorporated in our robot detection systems. The results were quite satisfactory since we obtained a significant

reduction of false positives without influencing the detection rate (see table 3, figures 4b and 5). In the AIBOs case, for a DR of ~90%, the FP are reduced from 70 to 58 in the case of the *Profile* detector, from 166 to 55 in the case of the *Back* detector, and from 254 to 99 in the case of the *Frontal* detector. Thus, around 50% of the false detections are eliminated thanks to the use of the context filter. In the humanoids case more than 50% of the false detections are eliminated by using the context filter. For instance, for a DR of 92.2%, the FP are reduced from 123 to 54.

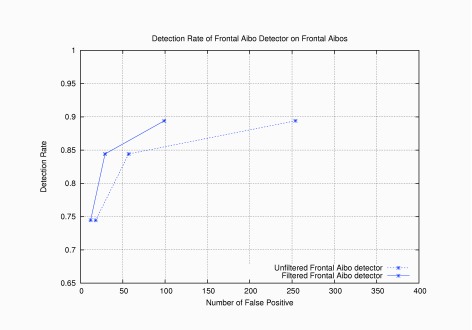
In figure 5 the ROC curves of all detectors are shown and compared with and without using the context filter. In all cases not a single good detection was filtered out; for any FP value the DR is higher in the case of using filtering. This shows that the context information is extremely useful, and that it should always be taken into account when available. Figure 6 and 7 shows some examples of positive and false detections. False detections are filtered using the horizon line (in green).



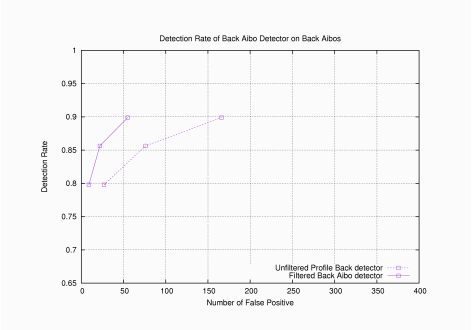
(a)



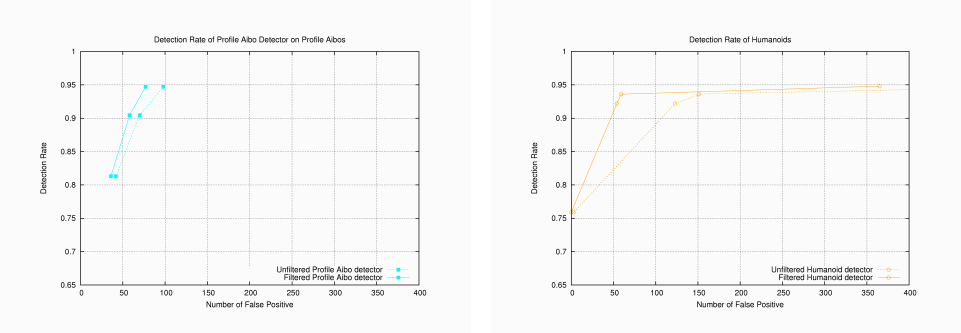
(b)



(c)

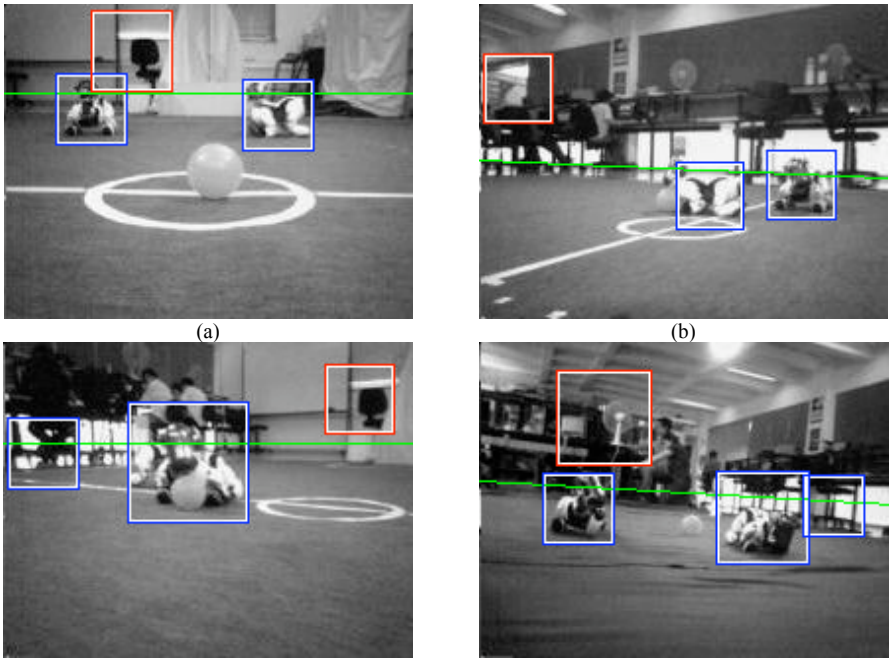


(d)



(e) (f)

Fig. 5. ROC curves on the AIBODetUChileEval and HDetUChileEval databases with and without using the visual horizon information to filter false detections. (a) Multi-view detector on all views, (b) Frontal detector on all views, (c) Frontal detector on frontal view, (d) Back detector on back view, (e) Profile detector on Profile view, (f) Humanoid detector. See main text for details.



(a) (b)

(c) (d)

Fig. 6. Detection results obtained with the frontal AIBO detector on images of the AIBODetUChileEval database. Filtered/non-filtered detections are marked as solid-line/dashed-line boxes.

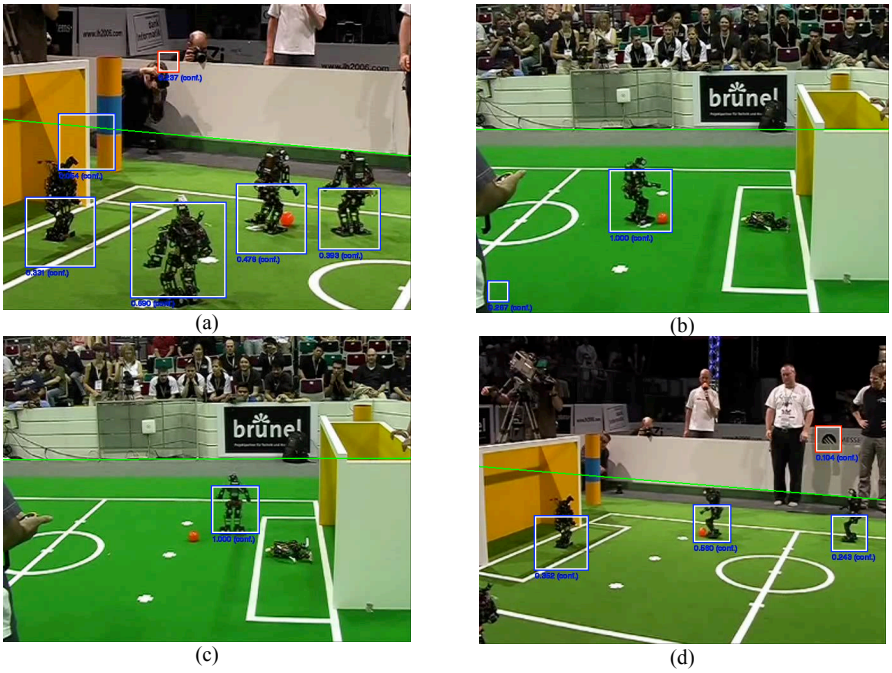


Figure 7. Detection results of the Humanoid detector on the HDetUChileEval database. Filtered/non-filtered detections are marked as solid-line/dashed-line boxes.

4.2.4 Effect of Training Procedures in the Detectors

To show explicitly some of the advantages of the proposed training procedures, we carried out an experiment in which only parameters of the bootstrap procedure were modified. In this experiment, the use of external bootstrap versus the use of internal and external bootstrap at the same time, is analyzed. In both cases, we train the different layers of the cascade using the same final number of training examples (3,600). In the first case, only external bootstrap, the 3,600 examples are all collected during external bootstrap. In the second case, internal and external bootstrap, 2,400 examples are collected during the external bootstrap, and then, 1,200 examples are collected in three iterations of internal bootstrap, 400 in each iteration. The experiment was implemented for the training of a *Frontal* detector of AIBO ERS7 robots. The obtained detectors were tested using the AIBODetUChileEval database. Using the same detection parameters in both cases, the obtained operating points are 89.80% detection rate with 453 false positives when only external bootstrap is used, and 89.40% detection rate with 254 false positives when internal bootstrap is added. This corresponds to almost reducing the number of false positives to a half for a given detection rate. This result clearly shows the benefit of using internal bootstrap.

4.2.5 Processing time

The processing time of the robot detectors is analyzed in three different platforms, and considers the use of different number of scales in the detectors (they are multi-scale). These platforms are: the internal processor of the AIBO ERS7 robots, a standard notebook computer, and a standard desktop PC. ERS7 robots have a 64bit RISC Processor (MIPS R7000) running at 576 MHz, 64MB RAM, and a color-camera of 208x160 pixels that delivers 30fps. The main characteristics of the notebook and desktop PC are, respectively, 1.73 GHz Intel Core Duo with 1GB of RAM, and 2.66 GHz Intel Core i7 desktop with 4GB of RAM, both running Windows XP. The frame rate depends mainly on the scaling factor used to obtain the scaled version of the images (1.5 or 1.2), and the number of scales skipped by the detection system.

Table 5 shows the average frame rate (in frames per second) delivered by the *Frontal* AIBO detector. In all cases the full soccer control library of the robots is running (Uchile1 control library [22]). The detector works fine with a scaling factor of 1.15 or 1.2, and skipping 1 or 2 of the first scales, which allows obtaining 2.2 fps in the AIBO's processor. This allows using the detector in our four-legged team, considering that it is not necessary to detect the robots in each frame, but every 3-7 for frames (every 90-210 milliseconds). In a desktop computer we can attain rates of more than 30 frames per second under the same conditions. This is important for our referee application, which will be described in the next section

Table 5. Frame rate (in frame per seconds) of the frontal AIBO detector.

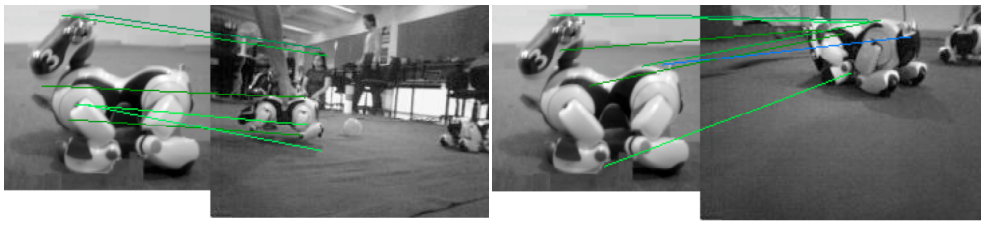
Configuration	Frame Rate (fps) in an AIBO processor	Frame Rate (fps) in a Laptop PC	Frame Rate (fps) in a Desktop PC
scaling 1.15 - no scale skipped	0.7	3.4	7.9
scaling 1.15 - skip 1st scale	1.1	6.7	14.9
scaling 1.15 - skip 1st,2nd scale	1.3	9.1	21.0
scaling 1.15 - skip 1st,2nd,3rd scale	1.9	11.1	30.1
scaling 1.2 - no scale skipped	0.8	4.8	9.9
scaling 1.2 - skip 1st scale	1.6	9.1	22.8
scaling 1.2 - skip 1st,2nd scale	2.2	12.5	36.1
scaling 1.2 - skip 1st,2nd,3rd scale	2.9	16.7	59.8

4.3. *Comparison with Alternative Detection Methods*

In a first experiment, the detection of AIBO ERS7 robots using local image detectors and SIFT descriptors, implemented using the detection system described in,[10] is analyzed. In this work, the detection of AIBO robots was achieved using robot's head reference images containing a player number (1 to 4), which was important in the matching process, as the numbers provided stable SIFT descriptors. In that scheme, a detection rate of ~80% was obtained.

In order to compare this detection system with the one here proposed, we used the AIBODetUChileEval database. In the experiments three full-body images, *Frontal*,

Profile and *Back* robot examples, were selected as reference images. The obtained results are very poor, with a detection rate of ~5%, for each of the three prototype images. The main reasons for this lower performance are due to (i) the small number of descriptors appearing on the robot's image areas, as the robot body is composed by regular white and red regions without any texture, (ii) the small size of the area covered by the robots in the test images, and (iii) the high symmetry of the robot's body, which disturbs the matching process. Figure 8 shows two examples of incorrect matches between pair's of reference-test images.



(a) (b)

Figure 8. Examples of wrong detections using the SIFT matching methodology. (a) Part of the prototype's descriptors are matched against the background, (b) Some robot's parts are incorrectly matched.

In a second experiment, the here proposed system is compared against an OpenCV [24] implementation of a cascade detector (OpenCV's *HaarTraining*). This cascade detector implements,[24] which is an extension of Viola&Jones detector.[13] The main differences between the here proposed detector and the OpenCV one are due to the use of different (i) weak classifiers (domain partitioning weak classifiers vs. decision stumps), (ii) boosting algorithm (real Adaboost vs. GentleBoost[24]), (iii) features (standard vs. extended rectangular features), (iv) cascade's types (nested vs. standard), and (v) training procedures (use or not use of internal bootstrap procedures).

As the main idea is to compare both detection system under the same conditions, the same datasets, and parameters (whenever possible) for the training and evaluation of the OpenCV based detector, are used. Figure 9 shows the detection results of frontal AIBO robots (AIBODetUChileEval dataset), in terms of ROC curves. As is can be observed, the unfiltered and filtered detectors have a much better performance than the OpenCV detector (the OpenCV was run without horizon filtering). The amount of false positives returned by the here proposed detectors is always much smaller than the one obtained by the OpenCV implementation, no matter how high the detection rate is.

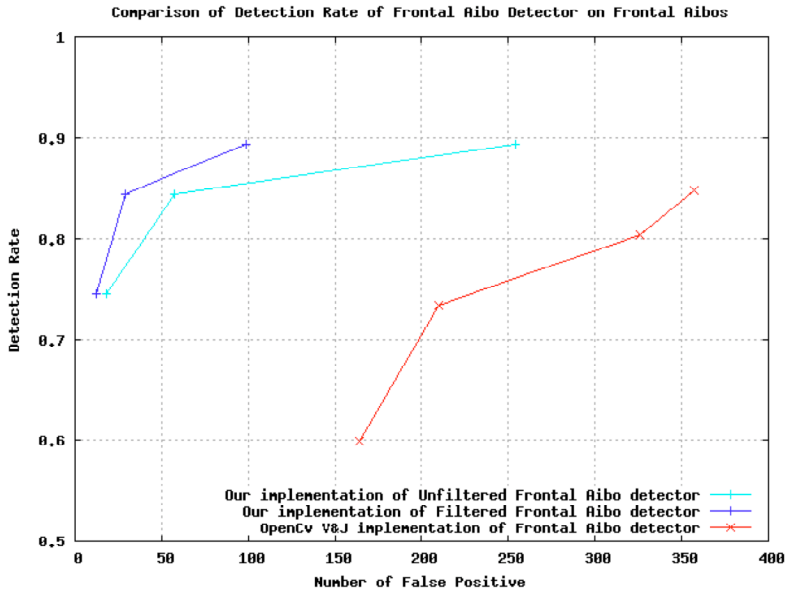


Figure 9. ROC curves of the proposed and OpenCV detectors in the AIBODetUChileEval database.

5. Proposed robot referee

A robot referee that uses the proposed robot detector to track players during a soccer game was developed. This application is a new extension to the concept of robot soccer, and it is useful to further test the applicability of our robot detection framework. The tasks required for refereeing are complex and require a very high degree of reliability (a goal not being counted is unacceptable), and they have to be solved in real-time.

5.1. *Robot hardware*

As robot referee we use a service robot (*Bender* [25]), whose main hardware components are (see a detailed description in [26]):

- A chest that incorporates a tablet PC as the main processing platform. The screen of the tablet PC allows: (i) the visualization of relevant information for the user, and (ii) entering data thanks to the touch-screen capability.

- The robot's head incorporates two CCD cameras and pan-tilt movement of the whole head. One of its most innovative features is the capability of expressing emotions.

- The arms of the robot are designed for allowing the robot to manipulate objects. Each arm has 6 degrees of freedom, 2 in the shoulder, 2 in the elbow, 1 for the wrist and 1 for the gripper.

- A mobile platform, in which all described structures are mounted. The platform provides mobility (differential drive), and sensing skills (1 laser sensor, 16 infrared, 16 ultrasound, and 16 bumpers). One interesting feature is that the relative angle between the mobile platform and the robot body can be manually adjusted. For the task of refereeing, the angle is set to 90 degrees. This allows the robot to have a frontal view of the field while moving along one of the field sides, even though it has a differential drive configuration (see figure 10).

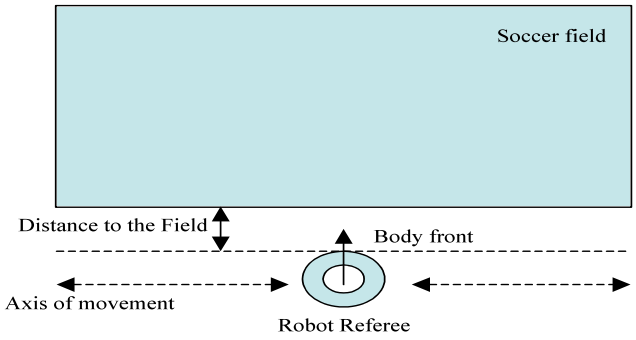


Fig. 10. Robot referee positioning.

5.2. *Robot controller*

The block diagram of the proposed robot referee controller is shown in figure 11. The system is composed by seven main modules *Object Perception*, *Visual Tracking*, *Self-localization*, *Refereeing*, *Motion Control*, *Speech Synthesis*, and *Wireless Communications*, and makes use of two databases: *Rules* (input) and *Game Statistics* (output).

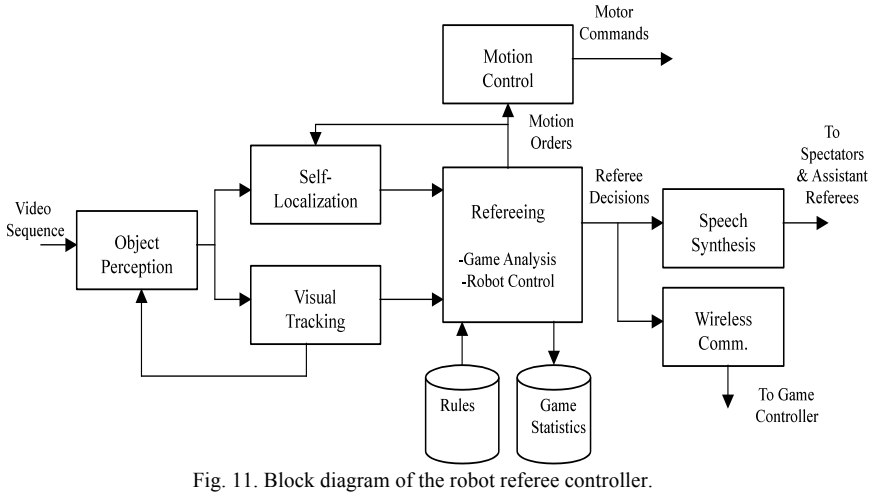


Fig. 11. Block diagram of the robot referee controller.

The *Object Perception* module has two main functions: object detection and object identification. First, all the objects of interest for the soccer game (field carpet, field and goal lines, goals, beacons, robot players, ball) are detected using color segmentation and some simple rules, similar to the ones employed by any RoboCup soccer robot controller (e.g., [8,9,22]). No external objects, as for example, spectators or legs of assistant referees or team members are detected (in some leagues assistant referees and team members can manipulate the robots during a game). The identification (identity determination) of goals, beacons and the ball is straightforward, because each of them has a defined form and color composition. The identification of field and goal lines is carried using the relative distance from the detected lines to the robot referee, and to the already identified beacons and goals. The detection of the robot players is performed using the multiscale robot detectors described in the previous sections. The information of the detected robots is passed to the *Visual Tracking* module to further process the information and to keep track of the robots during the game.

The *Visual Tracking* module is in charge of tracking the moving objects. The implemented tracking system is built using the mean shift algorithm,[27] applied over the original color image. The seeds of the tracking process are the detected ball and robot players. As in,[28] a Kalman Filter is employed for maintaining an actualized feature model for mean shift. In addition, a fast and robust line's tracking system was implemented (see description in [16]). Using this system, it is not necessary to detect the lines in each frame. Using the described perception and tracking processes, the system is able to track in real time (30 fps) all game moving objects and the lines.

The *Self-localization* module is in charge of localizing the robot referee. As in the case of the robot players, this functionality is achieved using the pose of the landmarks (goals and beacons) and the lines, and odometric information. The only difference being that in the case of the robot referee, the movements are not executed inside the field, but outside, along one of the field sides (see figure 10).

The *Refereeing* module is in charge of analyzing the game dynamics and the actions performed by the players (e.g. kicking or passing), and detecting game relevant events (goal, ball out of the field, illegal defender, etc.). This analysis is carried out using information about static and moving detected objects, and the game rules, which are retrieved from the *Rules* database (see description in [7]). In addition, this module is in charge of the referee positioning. The module should keep the referee outside of the field, but at a constant distance of the field side (the referee should move along one of the field sides), it should control de robot's head and body movement for allowing the robot to correctly *follow the game*, by avoiding obstacles and without leaving the field area (in case that the ball or a player leave the field.). It is important that the referee always perceives and follows the main elements of game-play. This is done by following the ball, and estimating the position of the players in the field. In future implementations we plan to use several cameras to have more information of the activities in the field. The outputs of this module are refereeing decisions (e.g. goal was scored by team A) that

are sent to the *Speech Synthesis* and *Wireless Communication* modules, motion orders that are sent to the *Motion Control* module, and game statistics (e.g. player 2 from team A score a goal) that are stored in the corresponding database.

The *Motion Control* module is in charge of translating motion orders into commands for the robot motors. These commands allow the control of the robot pose, the robot head pose, the robot facial expressions, and the robot arms.

Finally, the *Speech Synthesis* and *Wireless Communication* modules communicate the referee decisions to the robot players (using the *Game Controller* tool[4]), human assistant referees and spectators. Wireless communication is straightforward, while speech synthesis is achieved using the CSLU toolkit.[29]

5.3. *Robot detection in a refereeing task*

In this section robot detection results are reported. The detectors were tested using video sequences (with different configurations) taken in our laboratory. In total 5,293 frames were taken for a preliminary analysis of our system. These videos correspond to short play sequences; some examples can be seen in figures 12 and 13.

The robot detection module was programmed to detect robots every 5 to 10 frames. The robots are tracked in the remaining frames. With this detection/tracking combination we can estimate the robots' positions in every frame, and process more frames per second since the tracking of robots is very fast. If a new robot appears on the image, or if a robot is lost by the tracking module, it is quickly detected by the detection module and then passed to the tracking system. To measure the performance of the algorithm we counted all the robots that the detection and tracking system correctly found versus the number of robots that appear in all the frames, and the number of false detections. Table 6 shows the obtained results. These results show that the robots were correctly detected or tracked in almost all frames, having approximately one false detection every 16 frames. Usually, false detections appeared in consecutive frames, mainly because once a false detection is made, it is then tracked with the mean shift algorithm until the next detection frame comes. These results are quite good and show that the detector is working as intended. Once a robot is correctly detected the tracking system works remarkably well (the mean shift system has no problem tracking objects in these environments).

Table 6. Summary of the results for the robot detection and tracking. NF: Number of frames. NR: Number of robots. DR: Detection Rate of detected and tracked robots. FP: Number of false positives.

NF	NR	DR	FP
5,293	3,405	98.7%	334

6. Conclusions

In this article a framework for the robust detection of mobile robots using nested cascades of boosted classifiers was proposed. This framework was used to build detectors for humanoid and AIBO robots. The main module of the system corresponds to a nested cascade of boosted classifiers, which is designed to perform fast detections with high detection rates, and a very low number of false positives. Using this cascade classifier, an exhaustive multi-scale search is performed, and robots appearing at different scales and positions are detected. The nested cascade classifiers are trained using real Adaboost. Each (strong) classifier is built using domain-partitioning weak classifiers, implemented using LUTs. This allows obtaining compact cascades, with high processing speed, and low error rates.

The detection rate of the obtained systems is quite high, when the final detections are filtered out using context information (horizon line). For instance, the DR of *Profile* AIBOs is ~90%, with 58 false detection in 724 images (1 false detection every 12.5 frames). Similar results are obtained for the AIBO´s *Frontal* and *Back* detectors, as well as for the humanoids detector. It is important to note that the humanoid detector was evaluated in images containing humanoid's models different to the ones used to train it, showing a high generalization capability of the system, and allowing the use of the detector as a generic humanoid detector. Even thought the detection system was not designed to estimate the main orientation of the AIBO robots, it was possible to estimate it with a good accuracy. The system correctly estimated the robot's orientation in 79% percent of the detected and verified AIBOs.

The performance of the AIBO frontal detector was compared with two competitive detectors, one implemented using local features and SIFT' matching and one cascade detector built using the OpenCV library. The here proposed detector showed a much higher performance than the other two.

The applicability of the robot detectors is tested in a robot referee application in which the implemented humanoid detector is used. This refereeing system is able to detect and identify all humanoid-league defined field objects, and performs the detection and tracking of the moving objects in real-time. Experimental results shown that the referee achieves very high robot detection rates (98.7% DR with ~1 false detection every 16 images). As future work, we plan to characterize the performance of the complete refereeing system, to use external cameras to cover the whole soccer field, and to be able to correctly analyze complex game situations.

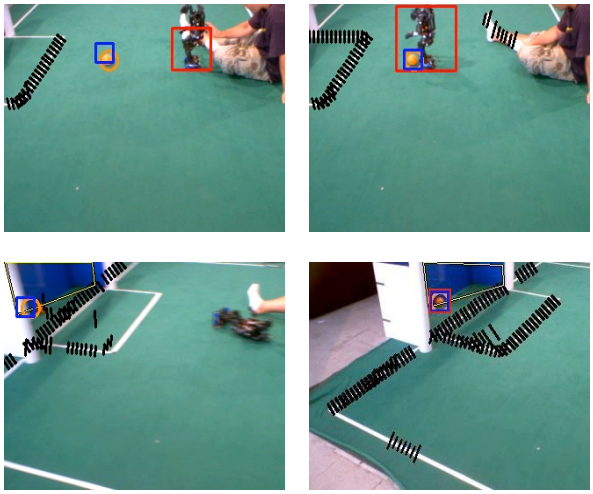


Fig. 12. Selected frames from a robot scoring sequence. Robot/ball tracking window in solid-/dashed-line.

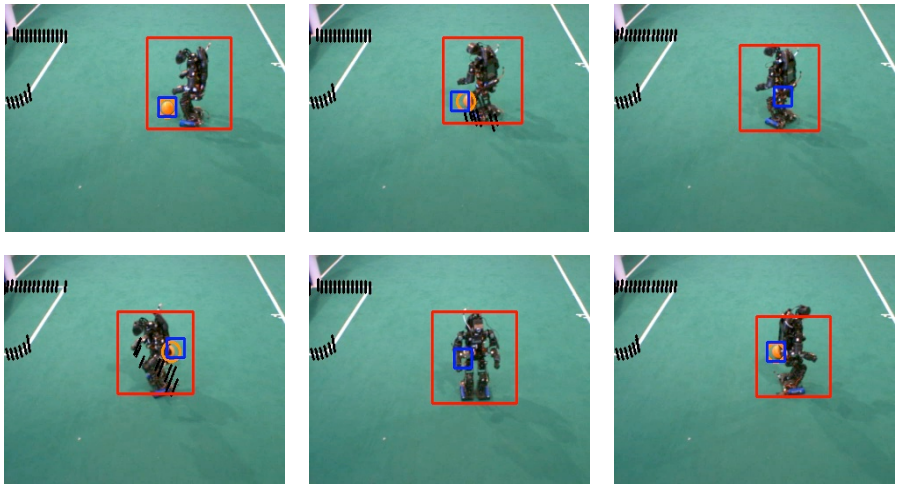


Fig. 13. Selected frames from ball tracking and robot detection. Robot/ball tracking window in solid-/dashed-line.

References

1. B. Wu, H. Ai, C. Huang, and S. Lao, Fast rotation invariant multi-view face detection based on real Adaboost, *6th Int. Conf. on Face and Gesture Recognition –* (Seoul, Korea, *FG 2004*), pp. 79–84.
2. R.E. Schapire and Y. Singer, Improved Boosting Algorithms using Confidence-rated Predictions, *Machine Learning*, 37(3) (1999) 297-336.
3. R. Verschae, J. Ruiz-del-Solar, M. Correa. A Unified Learning Framework for object Detection and Classification using Nested Cascades of Boosted Classifiers. *Machine Vision and Applications*, 19 (2), (2008) 85-103.
4. RoboCup SPL League Official Site. Available on June 2009 in: http://www.tzi.de/spl/bin/view/Website/WebHome
5. RoboCup Humanoid League Official Site. Available on June 2009 in: http://www.tzi.de/humanoid/bin/view/Website/WebHome
6. M. Arenas, J. Ruiz-del-Solar, Detection of Aibo and Humanoid Robots using Cascades of Boosted Classifiers. *Lecture Notes in Computer Science 5001* (*RoboCup* Symposium *2007*), (Springer 2008) pp. 449-456.
7. M. Arenas, J. Ruiz-del-Solar, S. Norambuena, S. Cubillos, A robot referee for robot soccer, *Lecture Notes in Computer Science 5399* (RoboCup Symposium 2008), (Springer 2009) pp. 426-438.
8. T. Röfer *et al.*, *German Team 2005 Technical Report*, RoboCup 2005, Four-legged league. Available on February 2006 in: http://www.germanteam.org/GT2005.pdf
9. M. J. Quinlan et al., The 2005 NUbots Team Report, *RoboCup 2005*, Four-legged league. Available on February 2006 in: http://www.robots.newcastle.edu.au/publications/NUbotFinalReport2005.pdf.
10. J. Ruiz-del-Solar, and P. Loncomilla, (2009), Robot Head Pose Detection and Gaze Direction Determination using Local Invariant Features, *Advanced Robotics*, 23 (3) (2009) 305-328.
11. S. Zickler, A. Efros, Detection of Multiple Deformable Objects using PCA-SIFT, *in Proc. 22nd AAAI Conf. on Artificial Intell*. (Vancouver, Canada, July 22-26 2007), pp. 1127-132.
12. M. Taiana, J. Gaspar, J. Nascimento, A. Bernardino, P. Lima. 3D Tracking by Catadrioptic Vision Based on Particle Filter, *Lecture Notes in Computer Science 5001* (Springer 2008) pp. 77-88.
13. P. Viola and M. Jones, Fast and robust classification using asymmetric adaboost and a detector cascade, *Advances in Neural Inform. Processing System* 14, (MIT Press, 2002), pp. 1311-1318.
14. J. Fasola, M. Veloso, Real-time Object Detection using Segmented and Grayscale Images, *Int. Conf. on Robotics and Automation* (*ICRA IEEE Press*) (2006), pp. 4088-4093.
15. J. E. Young, E. Sharlin, and J. E. Boyd, Implementing Bubblegrams: The Use of Haar-Like Features for Human-Robot Interaction, *IEEE Conf. on Automation Science and Engineering* (CASE) (Shanghai, China, October 7 – 10, 2006), pp. 298-303.
16. J. Ruiz-del-Solar, P. Loncomilla, and P. Vallejos, An automated refereeing and analysis tool for the Four-Legged League. *Lecture Notes in Computer Science 4434* (RoboCup Symposium 2006) (Springer, 2007), pp. 206-218.

17. M. Veloso, N. Armstrong-Crews, S. Chernova, E. Crawford, C. Mcmillen, M. Roth, D. Vail, A Team of Humanoid Game Commentators, (*6th IEEE-RAS Int. Conf. on Humanoid Robots*) (4-6 Dec. 2006*)*, pp. 228-233.
18. M. Veloso, N. Armstrong-Crews, S. Chernova, E. Crawford, C. Mcmillen, M. Roth, D. Vail, S. Zickler, A Team of Humanoid Game Commentators, *Int. Journal of Humanoid Robotics* (IJHR), (Sept. 2008) 5 (3) 228-233.
19. R. Verschae, J. Ruiz-del-Solar. A Hybrid Face Detector based on an Asymmetrical Adaboost Cascade Detector and a Wavelet-Bayesian-Detector, *Lecture Notes in Computer Science 2686* (2003) pp. 742-749.
20. K. Sung, T. Poggio, Example-Based Learning for Viewed-Based Human Face Deteccion, *IEEE Trans. Pattern Anal. Mach. Intell.,* 20 (1) (1998) 39-51.
21. Evaluation Database. Available on June 2009 in: http://vision.die.uchile.cl/
22. J. Ruiz-del-Solar, P. Guerrero, M. Arenas, P. Loncomilla, R. Palma-Amestoy, P. Vallejos, D. Monasterio, G. Díaz, J. Fredes, (2007). UChile Kiltros 2007 Team Description Paper, (*CD Proceedings of the RoboCup 2007 Symposium*, July 9 – 10, Atlanta, USA).
23. OpenCV Official Wiki site: http://opencv.willowgarage.com/wiki/
24. R. Lienhart, A. Kuranov, V. Pisarevsky, Empirical analysis of detection cascades of boosted classifiers for rapid object detection. *Lecture Notes in Computer Science* 2781 (*DAGM 2003*) (Springer 2003) pp. 297-304.
25. Bender robot official website: http://bender.li2.uchile.cl/
26. J. Ruiz-del-Solar, M. Correa, F. Bernuy, S. Cubillos, M. Mascaró, J. Vargas, S. Norambuena, A. Marinkovic, and J. Galaz.. UChile HomeBreakers 2008 Team Description Paper, *RoboCup Symposium (*July 15 – 18, Suzhou, China *2008).*
27. D. Comaniciu, V. Ramesh, and P. Meer, Kernel-Based Object Tracking, *IEEE Trans. on Pattern Anal. Machine Intell.* 25 (5) (2003) 564 – 575.
28. N.S. Peng, J. Yang, and Z. Liu, Mean shift blob tracking with kernel histogram filtering and hypothesis testing, *Pattern Recognition Letters*, 26 (2005), 605-614.
29. CSLU toolkit Official Website: http://cslu.cse.ogi.edu/toolkit/