Real-Time Object Tracking for Soccer-Robots without Color Information

André Treptow, Andreas Masselli and Andreas Zell

University of Tuebingen
Department of Computer Science
Sand 1
D-72076 Tuebingen, Germany
{treptow, masselli, zell}@informatik.uni-tuebingen.de
http://www-ra.informatik.uni-tuebingen.de

Abstract. Objects in the RoboCup scenario (soccer playing robots) are identified by their unique color. These visual cues will be removed in the near future so that new vision algorithms are needed to cope with this. In this paper we present a method for detecting and tracking the ball in a RoboCup scenario without the need for color information. We use Haar-like features trained by an adaboost algorithm to get a colorless representation of the ball. Tracking is performed by a particle filter. It is shown that our algorithm is able to track the ball in real-time with 25 fps even in a cluttered environment.

1 Introduction

The RoboCup world until now is designed to be very simple. Unique colors are used for visual identification of different objects like the ball, goals, opponents, etc. Fast and robust color segmentation algorithms have been developed to detect and track objects in this scenario in real-time [4][7].
In the future, visual cues like color will be removed to come to a more realistic setup with robots playing with a "normal" soccer ball. This brings up a new challenge for vision algorithms.
Until now the only approach known to us to deal with the new scenario is presented by Hanek *et. al.* [6]. They describe a so called Contracting Curve Density (CCD) algorithm which uses shape information and a local statistics of RGB values computed on both sides of the expected ball contour. Given a vague estimate of the ball's position, this method is able to extract the contour of the ball even in cluttered environments with different illumination conditions. However, Hanek *et. al.* do not deal with the problem of global detection.
In this paper, we describe a method to detect and track the ball based on simple gray-value features. Inspired by the work of Viola and Jones [5], who developed an algorithm to detect faces in real-time, we implemented their adaboost learning procedure to train ball-detectors. The algorithm used for learning a ball-detector is described in section 2. To track the ball we use a variant of the condensation algorithm described by Isard and Blake[3]. The learned classifier from section 2 is used in section 3 as a measurement model in the condensation algorithm for evaluation of possible ball positions. A similar approach that is used for tracking faces in the field of face detection and recognition can be found in [2]. Different experiments that show the robustness and real-time performance of our ball tracker are presented in section 4. Section 5 concludes the paper and points out topics for future research.

2 Modelling the ball

Recently, Viola and Jones developed a realiable method to detect faces within pictures in real-time [5]. We adopted their algorithm to come to a feature-based description of the ball that does not need color information. An object is described by a combination of a set of simple Haar wavelet like features shown in figure 1.

The advantage of using these simple features is that they can be calculated very quickly with the use of a so called "integral image". An integral image II over an image I is defined as follows:

$$II(x,y) = \sum_{x' \leq x, y' \leq y} I(x',y') \tag{1}$$

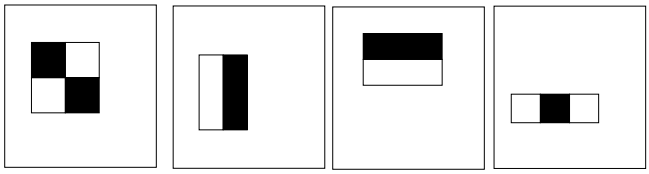

Fig. 1. Four different types of rectangle features within their bounding box. The sum of pixels in the white boxes are subtracted from the sum of pixels in the black areas.

In [5] it is shown that every rectangular sum within an image can be computed with the use of an integral image by four array references. In our implementation the total integral image of the size 320x240 is calculated in less than 2ms on an AMD Athlon XP 1600 processor.

To detect an object, a classifier has to be trained consisting of several discriminating features within a subwindow. Remembering that the number of features within a box sized 24x24 is greater than 160000 (far more than the number of pixels!), one has to select a small set of features that describe the object that has to be detected. Adaboost [8] is a mechanism so select a low number of good classification functions, so called "weak classifiers" to form a final "strong classifier" which is a linear combination of the weak classifiers. In the context of learning features, each weak classifier $h_j(x)$ consists of one feature f_j:

$$h_j(x) = \begin{cases} 1 & : \text{ if } p_j f_j(x) < p_j \theta_j \\ 0 & : \text{ otherwise} \end{cases} \tag{2}$$

where θ_j is a threshold and p_j a parity indicating if f_j has to be greater or less than the threshold for a positive classification. The algorithm to select a predefined number of features given a training set of positive and negative example images is shown in figure 2.

1. Input: Training examples (x_i, y_i), $i = 1..N$ with positive $(y_i = 1)$ and negative $(y_i = 0)$ examples.
2. Initalization: weights $w_{1,i} = \frac{1}{2m}, \frac{1}{2l}$ with m negative and l positive examples
3. For t=1,...,T:
   (a) Normalize all weights
   (b) For each feature j train classifier h_j with error $\epsilon_j = \sum_i w_{t,i}|h_j(x_i - y_i)|$
   (c) Choose h_t with lowest error ϵ_t
   (d) Update weights: $w_{t+1,i} = w_{t,i}\beta_t^{1-e_i}$ with $e_i = \begin{cases} 0 & : x_i \text{ correctly classified} \\ 1 & : \text{otherwise} \end{cases}$ and $\beta_t = \frac{\epsilon_t}{1-\epsilon_t}$
4. Final strong classifier: $h(x) = \begin{cases} 1 & : \sum_{t=1}^{T} \alpha_t h_t(x) \geq 0.5 \sum_{t=1}^{T} \alpha_t \\ 0 & : \text{otherwise} \end{cases}$ with $\alpha_t = log(\frac{1}{\beta_t})$

Fig. 2. Adaboost learning algorithm as proposed in [5].

3 Tracking the ball

Particle filtering [1], which is known as condensation in the area of computer vision, provides a statistical framework to the problem of object tracking. Given a history of measurements $Z_t = \{z_0, ..., z_t\}$ and a vector X_t describing the object-state, the idea of condensation is to approximate the posterior $p(X_t|Z_t)$ by a finite sample set

$$S_t = \{x_t^{(i)}, \pi_t^{(i)}\}, \ i = 1...N. \tag{3}$$

Every $x_t^{(i)}$ describes a possible object-state weighted with

$$\pi_t^{(i)} \propto p(z_t|X_t = x_t^{(i)}) \text{ (observation model).} \tag{4}$$

The samples evolve over time by a resampling mechanism proportional to $\pi^{(i)}$ and a dynamic model $p(x^{(t+1)}|x^{(t)})$ that predicts object states for the next time step. The estimate of the object-state at time t is the weighted mean over all sample-states:

$$\hat{X}_t = E(S_t) = \sum_{i=1}^{N} \pi_t^{(i)} x_t^{(i)}. \tag{5}$$

Due to the sampling mechanism condensation provides robustness and is able to deal with multiple state hypotheses.

In this paper, we apply condensation to the problem of ball-tracking. The object is decribed by the state-vector

$$x_t^{(i)} = [x_I, y_I, s_I, v_x, v_y, v_s]^T \tag{6}$$

where the position of the ball in the image is described by (x_I, y_I), the velocity in x- an y-directions is (v_x, v_y), s_I represents the size of the ball and v_s is the velocity in size. The dynamic model is implemented as a simple random walk:

$$x_{t+1} = Ax_t + u_t \tag{7}$$

This is a first order process where A defines a movement with constant velocity. Small random changes in velocity are modelled by addition of a random vector u_t. This motion model is quite simple and can be extended in the future. However, experiments indicate that robust tracking can be achieved even with random walk dynamics.

The observation probability of each sample is derived from the results of the learned classifier described in section 2. To calculate the weight $\pi^{(i)}$, the ball-detector is evaluated at the position (x_I, y_I) of the sample. Instead of using the output of the strong classifier, which is a binary value, we rate each sample according to the weighted sum of the weak classifiers:

$$\pi^{(i)} \propto \delta \sum_{j=1}^{T} \alpha_j h_j(x) \tag{8}$$

where α_j, h_j are the weighted weak classifiers (see section 2) and δ is a penalty factor to penalize samples that are not classified as the ball (response of strong classifier is zero).

The cameras on our robots have a fixed field of view so that the size of the ball in the image is always the same for a special (x, y) position in the image (as long as the ball is on the ground). In a calibration process we build a lookup table for each pixel in the image that gives the corresponding size of the ball at this position. Samples in the particle filter are given a zero weight if their size does not match the calibrated size within a given tolerance.

To improve further the robustness of tracking, we reinitialize 10% of the samples having lowest weights.

4 Experiments

4.1 Training

First of all we collected a set of positive examples showing the ball under different viewing conditions and a set of negative examples that was built from random cropped picture regions that do not contain the ball. To come to a better negative example set yielding lower false positive rates we successivly included missdetections from images that do not contain the ball into the negative example set. Most of the images in the example sets have mirrored counterparts in the same set ending up with 1100 positive and 10000 negative examples. The sets are split randomly into a training and a test set containing 550 positive and 5000 negative examples each. Overall we used much more negative than positive examples to keep a low false positive rate. All training examples had the size 19x19 and were normalized to have zero mean and a standard deviation of 1.0.

We trained a detector with the adaboost algorithm until the resulting strong classifier labelled all examples in the training set correctly, ending up with 137 weak classifiers. Detection, false positive and correct classified rates over the number of weak classifiers on the training set are shown in figure 3. The training of the 137-feature classifier required about 80 minutes on an AthlonXP 1600 processor.

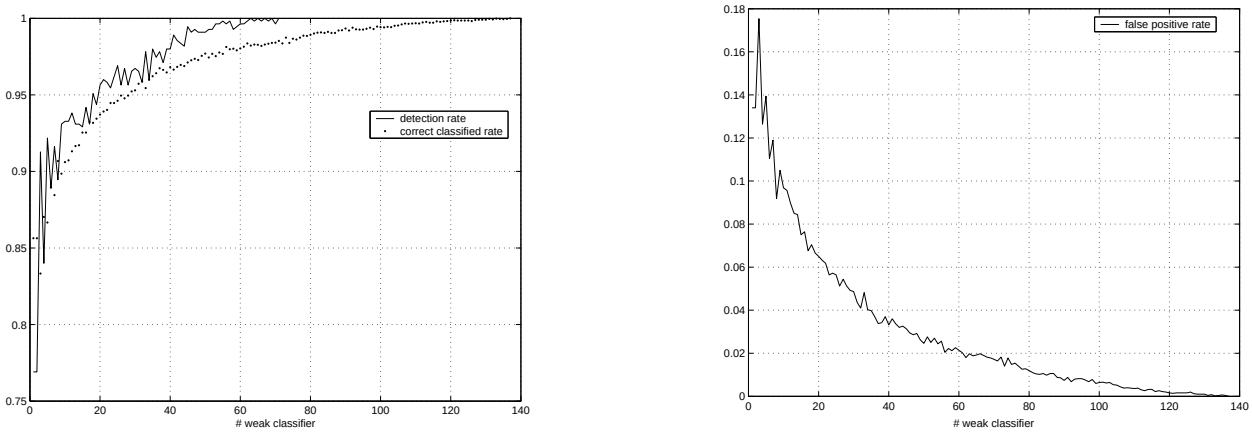

Fig. 3. Detection and false positive rates during training

4.2 Detection and tracking results

On our test set with 550 positive and 5000 negative examples, we achieved a total detection rate of 91.64% with a false positive rate of 0.015% leading to a total rate of 97.86% of examples that were correct classified. The results on the test set are shown in table 1.

true\predicted	positives	negatives
positives (550)	504	46
negatives (5000)	73	4927

Table 1. Results of 137-feature detector on test set

To track the ball we used 450 samples so that one timestep of the condensation algorithm required about 35ms on an AthlonXP 1600 processor with a framesize of 320x240. For each frame, the integral image and a so called squared integral image (required for image normalization) are calculated in about 6ms. We recorded 12 different sequences to test the ball tracker under different conditions. We were able to track random movements of the ball up to speeds of 1m/s. The distance of the ball from the camera ranged from 20cm up to 2m. The limitation of 2m is a result of the extreme wide-angle lense mounted on the camera, so that objects that are far away have an extremly small size in the projected image plane and could not be tracked. Some of the best tracking results are shown in figure 4, 5 and 6. The weighted mean of the best 20% of all particles is considered to be the answer of the ball tracker and marked with a box. The particles were initialized randomly in the first frame. In one sequence, the initial position of the ball was not found and the tracker was distracted, but averaged over the remaining 11 test sequences, the ball position was found within the first 8 frames of the sequence (min: 2 frames, max: 25 frames). The initialization phase is shown in figure 7. In our test sequences the ball did not move during this phase.
Tracking through occlusions was also possible as shown in figure 5. The tracker is even able to recover from distraction as one can see in the experiment shown in figure 6. However, there are situations in which the tracker is unable to recover due to false detections of the strong classifier. Therefore, improving the classifier e.g. by incorporating information about shape will be topic of future research. The situation that the ball is not seen in the image at all is not addressed within this paper. In this case the weighted mean of the n best particles cannot be treated to be the position of the object. One way to deal with this is to return a positive answer only if a ball is detected by the strong classifier within the position suggested by the particle filter. Another possibility is to set a classification threshold for the weighted mean of the detection results from the best particles.

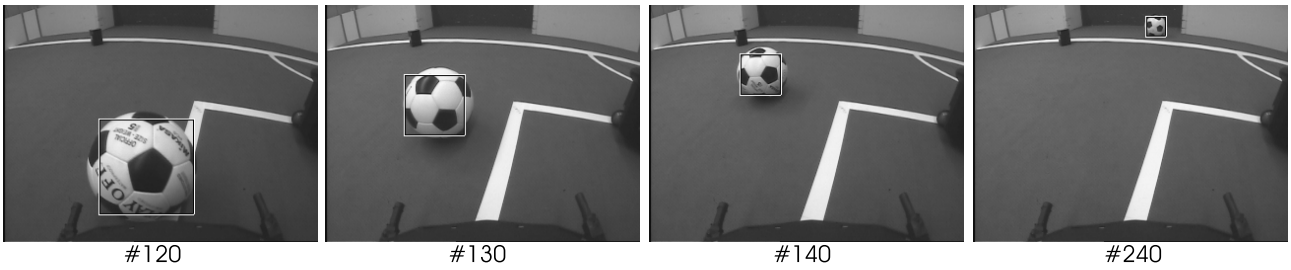

Fig. 4. Tracking the ball.

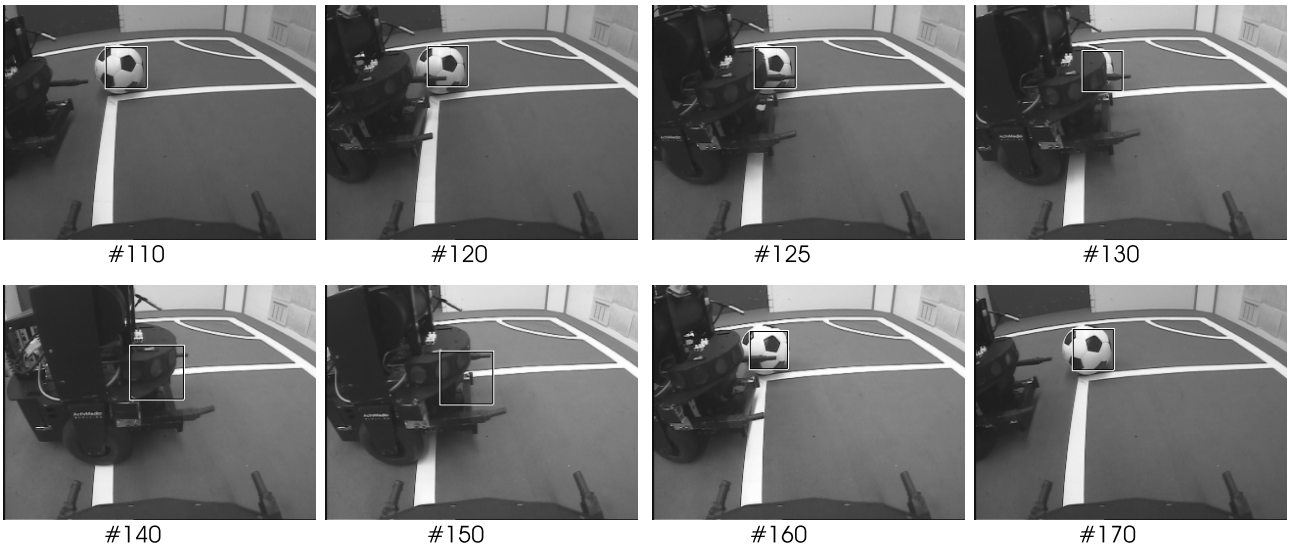

Fig. 5. Tracking through occlusion.

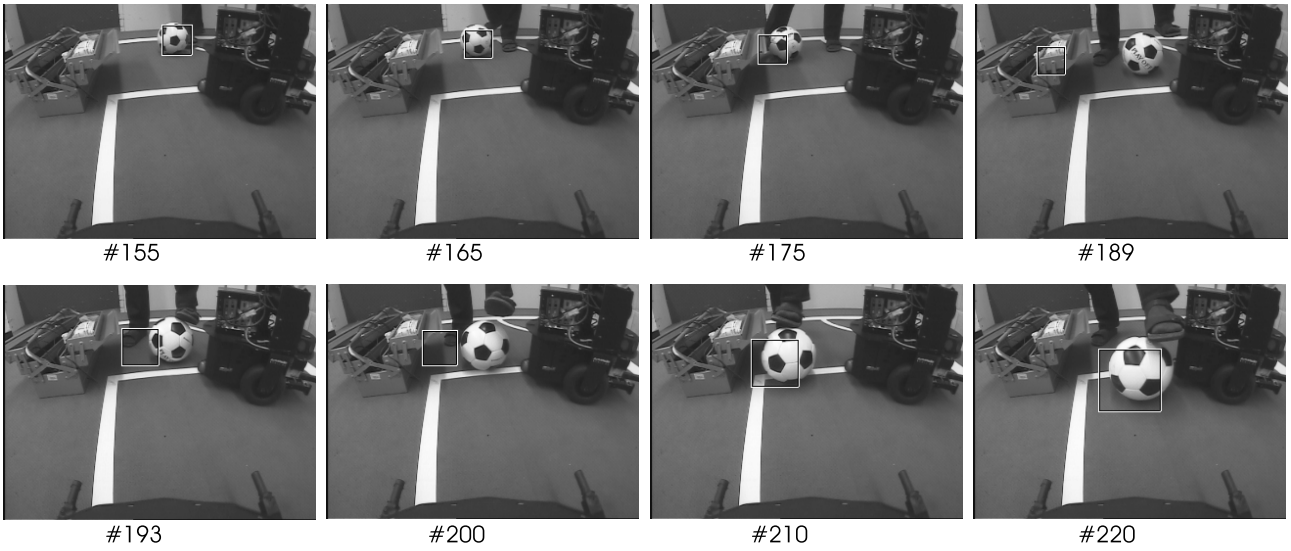

Fig. 6. Tracker is distracted by clutter.

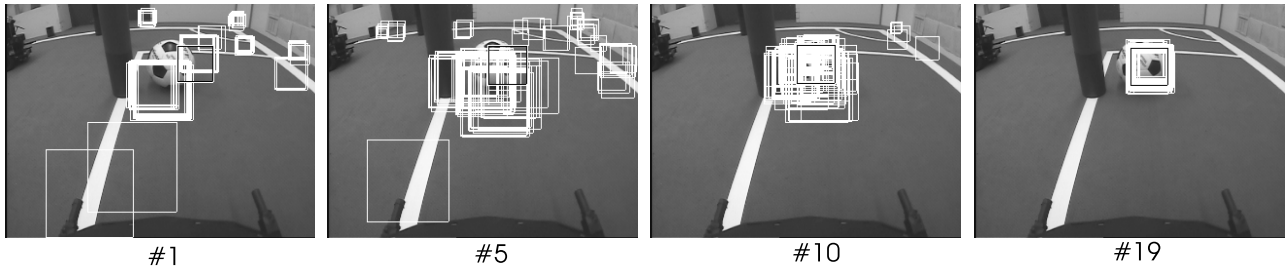

Fig. 7. Initialization: Best 20% of all particles.

5 Conclusion and future work

In this paper, we presented a method to detect and track a ball which is not marked with a special color in a RoboCup environment. The results of the adaboost feature learning algorithm are integrated into a condensation tracking framework making it possible to track a learned feature based representation of the ball even in cluttered environments in real-time. Besides the work of Hanek *et. al.*, who do not cover the problem of global detection of the ball, it is the first approach to deal with the problem of detecting and tracking objects in the RoboCup domain that do not have unique colors.
Experiments showed that tracking in real time is possible even if the ball is occluded by other objects. Before our robots are able to play with a non-colored ball, some more attention has to be paid to situations where no ball is present at all. Strategies for recovering from false detections have to be implemented to improve robustness. One possible solution is to integrate a measure of shape into the particle and to extend the detector to a cascade of detectors.

References

1. N. de Freitas A. Doucet and N. Gordon, editors. *Sequential Monte Carlo Methods in Practice*. Springer, New York, 2001.
2. G. Shakhnarovich, P. A. Viola and B. Moghaddam. A Unified Learning Framework for Real Time Face Detection and Classification. In *5th IEEE International Conference on Automatic Face and Gesture Recognition*, Washington D.C, 2002.
3. M. Isard and A. Blake. Condensation – conditional density propagation for visual tracking. *International Journal of Computer Vision*, 29(1):5–28, 1998.
4. J. Bruce, T. Balch and M. Veloso. Fast and inexpensive color image segmentation for interactive robots. In *Proc. of the 2000 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS '00)*, volume 3, pages 2061–2066, 2000.
5. P. Viola and M.J. Jones. Robust real-time object detection. In *Proc. of IEEE Workshop on Statistical and Theories of Computer Vision*, 2001.
6. R. Hanek, T. Schmitt, S. Buck and M. Beetz. Towards RoboCup without Color Labeling. In *RoboCup International Symposium*, Fukuoka, Japan, 2002.
7. T. Bandlow, M. Klupsch, R. Haneka and T. Schmitt. Fast Image Segmentation, Object Recognition and Localization in a RoboCup Scenario. In *3. RoboCup Workshop, IJCAI'99*, 1999.
8. Y. Freund and R.E. Schapire. A short introduction to boosting. *Journal of Japanese Society for Artificial Intelligence*, 14(5):771–780, September 1999.

This article was processed using
the TEX macro package with ECMR2003 style