Detection of Rotational-Invariant Objects Through Regression

Susana Brandão [#1,#3], Manuela Veloso [#2], João Paulo Costeira [#3]

[#1]*ECE,* [#2]*CS Carnegie Mellon University*
5000 Forbes Avenue, Pittsburgh USA,
[#3] *ISR, IST-UTL,*
Av. Rovisco Pais 1, 1049-001 Lisboa Portugal
[1]sbrandao@andrew.cmu.edu
[2]mmv@cs.cmu.edu
[3]jpc@isr.ist.utl.pt

***Abstract*— Visual object detection in robot soccer is fundamental so the robots can act to accomplish their tasks. Current techniques rely on manually highly polished definitions of object models, that lead to accurate detection, but are quite often computationally inefficient. In this work, we contribute with an efficient detection method based on off-line training. We build upon the observation that the robot soccer objects, the ball in particular, is of a well defined color and rotational-invariant shape, investigate an off-line learning approach to modelling such objects. We present our new regression learning approach consisting of two main phases: (i) off-line training, where the objects are automatically labeled off-line by existing techniques, resulting in learned object models through regression, and (ii) online detection, where a given image is efficiently processed in real-time with respect to the learned models. We show comparing results with current techniques comparing both precision and computational load.**

I. Introduction

In robot soccer, vision plays a crucial role on localization and actuation since both task rely on images to provide ground truth for landmarks and objects localization. One of the biggest challenges faced by robot soccer teams is to provide the robot with adequate models for each class of objects in the field. The current paper presents a highly efficient way of recognizing objects in this environment.

The main challenge faced by vision based world modeling is the object representation. Objects in images suffer from occlusion, changes in perspective as well as changes in illumination. To provide the necessary robustness to vision, teams have developed algorithms which work well, but are computationally heavy and/or very thorough and consequently very difficult to implement and generalize. This paper aims at providing a simple and incremental way to model the object and a computationally fast way to identify them during runtime.

When compared with other computer vision problems, the robot soccer environment has the advantage of being invariant in time. All robot soccer fields are alike, the ball and goals have always the same structure and colors are always the same. The robot is constantly faced with very similar problems and it should be able to build upon past experiences to solve new problems. Our approach provides the robot with such capability. It uses past experiences and the results of current state of the art algorithms to construct object models and how to locate them with in the image.

Object detection in images is usually a search problem: we need to go through all the image (either by segmenting or by scanning) and test the hypothesis of whether a given object is in that part of the image. In the proposed approach, we do not need to search for the object in the whole image. Instead, our algorithm provide us a one to one relation between any image and the object position.

Our approach to object detection in robot soccer aims to leverage on the simplicity and repeatability of the domain. The simplicity provide us with the opportunity to use naive tools to tackle problems such as those arising from the translation of objects. The repeatability allow us to train, off-line, object detectors which can be used efficiently during an online phase. Furthermore, the algorithm is capable to provide directly a confidence on its results.

In this paper we focus on ball detection using a principal component regression, pcr, between images containing a robot soccer ball and the ball position in the image. In a pcr, we start by reducing the dimensionalty of our observations through means of a principal component analysis and then perform a linear regression between the reduced observations and their labels. In the case of ball detection in an image, observations will correspond to images and labels to ball positions. The approach is appealing because, while the results from the regression allow us to estimate the position of a ball given a new image, the intermediate principal components allow us to estimate the error in each new detection.

The paper is organized as follows: in section II we describe related work in the area of robotic vision, in section III we present our approach, and in section 4 we present empirical evidence and compare conceptually with current state of the art.

II. Related Work

There are currently many approaches for vision in humanoid robots. Approaches for ball detection range from Neural Networks ([1]), Circular Hough Transform ([2]) and Circle Fitting ([3]). However, in the past edition of robot soccer a large fraction of the teams ([4], [5], [6], [7], [8], [9], [10], [11]) used one of two types of algorithms: i) Scan-line ([10]) and ii) CMvision ([11]) related algorithms: color threshold, runs identification and blob formation.

Scan-line is a very thorough algorithm, which relies mostly on human modeling of the several elements in the field. It creates color segments based on the scanning of just a few columns in the field. To compensate the information lost between the columns, the algorithm uses human imposed priors on what the segments should be in the robot soccer environment. By looking only at a reduced set of lines, the algorithm is very fast. However, the modeling of each object in the field is quite time consuming. If new elements were added to the field, the reintroduction of the human knowledge would be quite time consuming. Our proposed approach also leverages on the possibility of estimating the variables of interest using a reduced set to a reduced set of pixels in the image. However, object models do not require human intervention to be constructed. They are built upon both synthetic and real data and use labels provided offline by CMvision.

CMvision also relies on color segmentation to identify objects in the image. However, since segments are created based on 4-connectedness, it requires thresholding of almost all the pixels in the image. Afterwards, blobs still have to be sorted by colors and sizes, and finally objects are detected based on how well the largest blobs of the respective color fit to a given model, which is again imposed by humans. All this process, albeit quite accurate, is extremely time consuming and processes a frame at a lower rate then the camera acquires them. Our algorithm, by not requiring blob formation, does not need to scan the whole image and thus is computationally faster.

We can also frame the current work under the more general context of object recognition. In particular, we use the general concepts from Turk et al [12] for faces classification to do ball detection in robot soccer. In their work they performed face recognition and detection through projection into principal components of a set training data. Furthermore, they detect false positives using the distance between an image and its projection into the linear subspace of images generated by the principal components. In our current work, principal component analysis is also used to reduce the space dimensionality and to detect false positives. However, detection is performed using a linear regression between the images in the projected space and the object position.

III. Object Detection By Regression

The main objective of our work is to detect an object in an image with a computationally efficient and easy to implement algorithm that provides control over the error we are incurring. In particular, among the objects present in the robot soccer field, we focus on balls. The algorithm is composed of an offline training stage and an online testing stage. During the offline training stage we perform a linear regression between images and ball position in those images. During the online stage, the linear regression results are used to detected the ball in new images. To perform the regression, we first need to reduce the dimensionality of our images by means of a principal component analysis, pca. The resulting principal components also provide us with a way to estimate our error in the detection of new balls.

We approach the detection problem by creating a one to one affine map between an image with a ball, I, and the ball position in that image, $b = (x, y)$:

$$\mathbf{b} = W^T \mathbf{i} + \mathbf{b}_0 = \tilde{W}^T \tilde{\mathbf{i}} \tag{1}$$

where W is the weight matrix relating image pixels and positions, $\mathbf{b}_0$ is a bias term and $\mathbf{i}$ is the vectorized version of the image, where all the image columns were concatenated into a single vector. To simplify notation, we included the bias term into $\tilde{W}^T = [\mathbf{b}_0, W^T]$ and defined $\tilde{\mathbf{i}}^{\mathbf{T}} = [1, \mathbf{i}]$.

Our objective is to estimate the weight matrix $\tilde{W}$ through a linear regression. The regression can be performed offline using previously labeled images. However, a regression needs more data points than coefficients to estimate. In our case, this implies that we will need at least the same amount of different images as coefficients in the $\tilde{W}$ matrix. Since the $\tilde{W}$ matrix has more than twice the entries as the number of pixels considered, this is clearly infeasible. We would need of a dataset of the order of $\mathcal{O}(10^4)$ images.

We solve the dimensionality problem by taking two complementary approaches: first we sample part of the image pixels, second we perform a pca. For the image sampling we use an uniformly fixed grid. From the pca we retrieve a set of orthogonal vectors corresponding to the directions of larger variance on our dataset and correspond to the subspace of images with balls in the larger space of images. The linear regression can thus be performed in this subspace.

To compute the principal components, we start by considering a vectorized version $\mathbf{i}$ of image I and construct our observations matrix, O, by assigning each image to a row in the matrix. For example, if we had a set of L images , $I_1, ..., I_l, ..., I_L$, with N rows and M columns, our observation matrix would be:

$$O = \begin{bmatrix} i_{1,1,1} & \dots & i_{1,1,M} & i_{1,2,1} & \dots & i_{1,N,M} \\ i_{2,1,1} & \dots & i_{2,1,M} & i_{2,2,1} & \dots & i_{2,N,M} \\ \vdots & \dots & \vdots & \vdots & \dots & \vdots \\ i_{L,1,1} & \dots & i_{L,1,M} & i_{L,2,1} & \dots & i_{L,N,M} \end{bmatrix} \tag{2}$$

where $i_{l,n,m}$ is the color of the pixel with coordinates n, m of the image l.

Furthermore, our images I_l are thresholded versions of those captured by the robot's camera and we consider pixels that do not belong to the ball as background: $i_{l,n,m} = 1$ if the pixel (n, m) has the same color as the ball and $i_{l,n,m} = 0$ if not. To account for differences in ball size due to perspective we need to normalize the rows of our matrix so that they sum 1.

The principal components correspond to the eigenvectors of the covariance matrix of our observations ([13]). This matrix is defined as:

$$C(O) = (O - \bar{O})^T (O - \bar{O}) \tag{3}$$

where $\bar{O}$ is a matrix with the mean of the columns of matrix O.

The principal components obtained, V, are an orthogonal set of images which span our space of images with balls. The total number of components will be equal to the number of different and linearly independent images in the dataset. The principal components for the case of a ball in different places in the image is illustrated in figure 1 where we represent three different principal components for a set of synthetic images. In the examples provided, we see that the components follow an hierarchy of resolution: the first components, which contain more information, have lower spatial frequency. We can thus reduce the image dimensionality on our datasets by projecting their images into the first components of this new basis, as seen in equation eq.4.

$$O_r = O V^T \tag{4}$$

A. Training

After reducing the dimensionality of our images dataset, we perform a linear regression between the reduced images i_{rl} and the known ball position $b_l = (x_l, y_l)$. The result of the linear regression is the set of coefficients $W_r = (w_{r,x}, w_{r,y})$ which solve the linear least squares problem in equation eq. 5 and are given by equation 6.

$$\min_{W_r} \|B - \tilde{O}_r \tilde{W}_r\| \tag{5}$$

$$W_r = (\tilde{O}_r^T \tilde{O}_r)^{-1} \tilde{O}_r^T B, \tag{6}$$

where B is the matrix whose row l is the position vector b_l^T, $\tilde{O}_r = (\mathbf{1}, O_r)$ and $\mathbf{1}$ is a column vector with ones which allow us to incorporate the affine bias term in $\tilde{W}_r$.

Our training dataset is composed of both synthetic images and real images captured by the robot while it was searching and following a ball. For the synthetic dataset, we simulated a ball moving uniformly across the whole image. The resulting images include random noise and occlusion in edges and corners. The synthetic dataset was composed of 768 images which span uniformly the space of images containing a ball. Examples can be found in figure 2. The robot collected data includes balls in different parts of the image, but the sampling is not thorough. The robot is acting according to the ball position and keeps the ball approximatedly in the center of the image. The resulting dataset contains fewer examples of ball on the edges of the image. However, real images introduce the variability on the ball shape which the robot will experience during run-time. From the total of 856 real data images we have ball occlusion on the image edges (figure 3(c)) and by other objects (figure 3(d)). We also have several examples of motion blur (figure 3(e)) and of random noise (figure 3(f)) captured by the robot while searching for the ball in the environment. All the real images were labeled using CMvision.

B. Testing

To find the ball position in a new image, $\mathbf{i}$, we can now use the linear model which we trained in the off-line stage using equation 7.

$$\mathbf{b} = \tilde{\mathbf{i}}_{\mathbf{r}} \tilde{W}_r = \mathbf{i} V^T W_r + b_0 \tag{7}$$

where $\mathbf{b}$ is the ball position and we can compute $V^T W_r$ off-line.

Furthermore, the algorithm allow us to test an image for the existence of balls. This is of particular importance, because we have no mechanism to distinguish between ball pixels and noise pixels of the same color: if we introduce just noise in a new observation, we will still get a prediction for a ball position.

The vectors resulting from the pca describe areas of the image with strong correlation in the dataset. If there is no ball in the image, these correlations will not hold and the projection of the image into the pca vectors will represent an image very different from the original one. By projecting the image in the pca basis and re-projecting it back into the images space again, we can measure how good is our model based on the angle between the two vectors: the original image and the re-projected one.

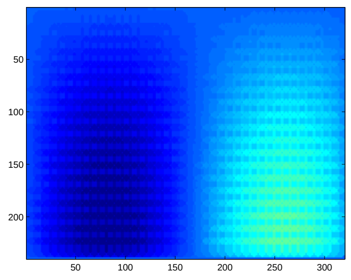
The cosine of the angle between the two images represents our belief in the ball position. Values near 1 correspond to very small angles: the original and the re-projected images are very similar and we have a high probability of having a ball in the image. Values below $\sqrt{2}/2 \simeq 0.7$ correspond to angles larger than $\pi/4$: the original and the re-projected image are pointing to very different regions in space and most probably there is no ball in the image. The re-projected image can be computed using equation 8, and the angle between original and re-projected is given by equation 9.

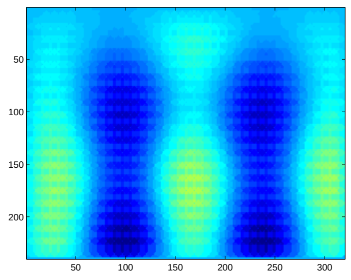
$$\mathbf{i}' = V \mathbf{i}_r = V \mathbf{i} V^T \tag{8}$$

$$\theta_{ball}(\mathbf{i}) = \frac{\mathbf{i} \cdot \mathbf{i}'}{\|\mathbf{i}'\| \|\mathbf{i}\|} \tag{9}$$

The final algorithm for identification of the center of a ball in an image is given by alg.1 and can be separated in three steps: i) first we start by discarding images with less than 3 orange pixels; ii) second we compute the position of an hypothetical ball; iii) compute the confidence of the ball hypothesis and discard the hypothesis if the confidence is less than 0.7.

The algorithm was tested with real robot data again collected while the robot was searching and following the ball. The testing set is composed of 160 images and presents the same characteristics than those in training: occlusions, blurring and random noise. In figure (4) we present examples of the images and detection results. The white squares in both images


(a) 1st component


(b) 10th component

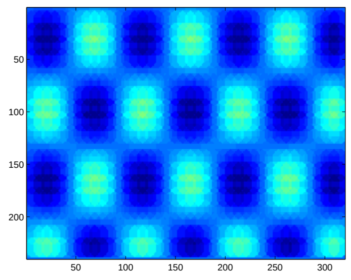
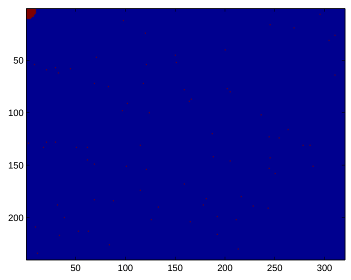
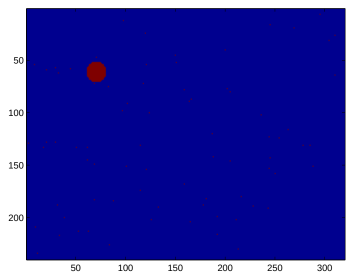

(c) 34th component

Fig. 1. Synthetic data principal components.


(a) Synthetic image, ball in the corner


(b) Synthetic image, ball in the center

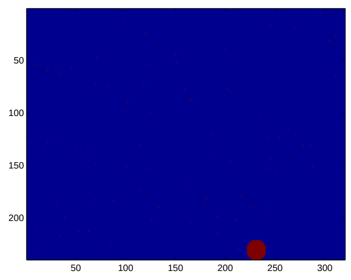
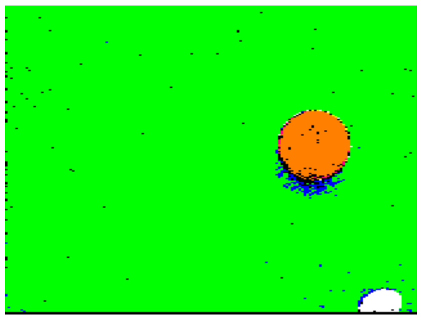
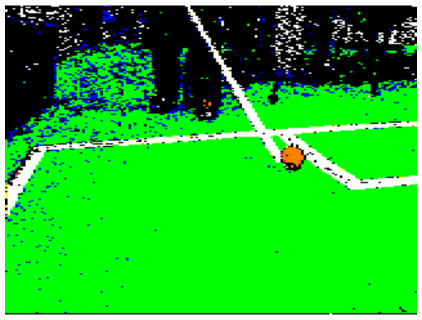
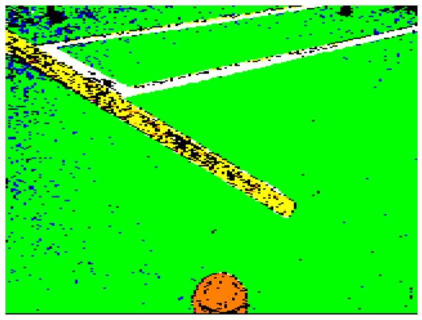
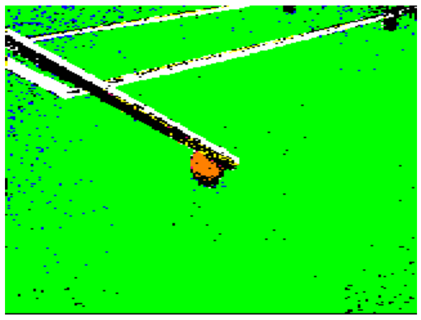

(c) Synthetic image, ball in the bottom

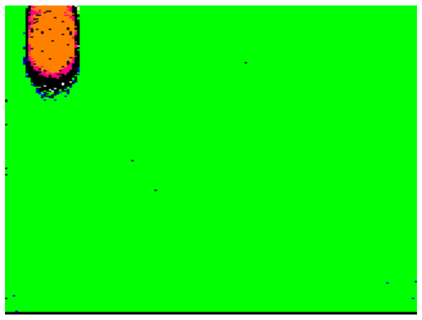
Fig. 2. Examples of synthetic images used in training. All the balls have the same size, but each image has a ball in a different position, including cases where the ball is occluded by the edges and corners.


(a) Large Ball


(b) Small Ball With Random Noise


(c) Occlusion on the edge of the image


(d) Occlusion by object


(e) Movement Blur

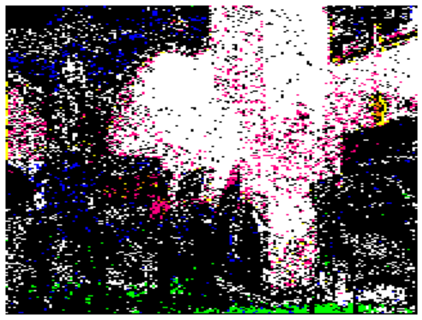

(f) Random Noise

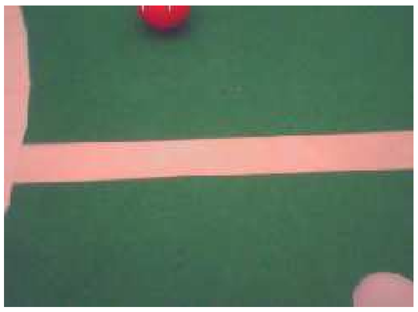
Fig. 3. Examples of real images used for training and testing. Real images datasets covered all types of images with balls presented to robots during a robot soccer match: balls of different sizes, occlusion, motion blur and random noise.

correspond to a localization, but not to a bounding box. The size of the square serves only illustrative purposes.

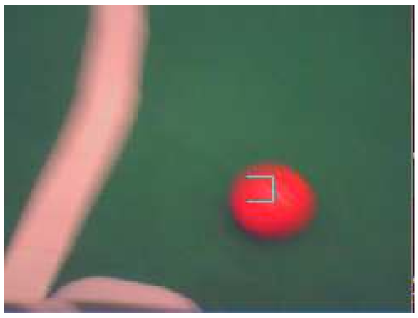
We want to compare the performance both in time and accuracy of our method with respect to other methods. In particular

Algorithm 1 Identify center of a ball

```
for all $(\mathbf{i}) \in O_t$ do
  if $\#(OrangePixels) < 3$ then
    return FALSE
  else
    $\mathbf{b} = \mathbf{i}W^T$
    $\mathbf{i} = V\mathbf{i}V^T$
    $\theta_{ball}(\mathbf{i}) = \frac{\mathbf{i}\cdot\mathbf{i}'}{|\mathbf{i}'||\mathbf{i}|}$
    if $\theta_{ball}(\mathbf{i}) <= 0.7$ then
      return FALSE
    else
      return TRUE
    end if
  end if
end for
```



(a) Example of detection under occlusion



(b) Example of detection under motion blurring

Fig. 4. Examples of detection under difficult conditions: occlusion and blurring.

we want to compare them against CMvision, which was used as the ground truth in training. To achieve that, we used both methods offline, running each one 1000 times per frame in a Pentium 4 at 3.20GHz. The processing for CMvision included thresholding, blob formation and ball detection, while our approach included only thresholding and ball detection.

Results for processing time are presented in figure 5. We achieve an average processing time of about one fourth than CMvision. Furthermore, our the worst case was still better than the best case in CMvision.

In terms of accuracy we still achieved good results. To compare both methods, we measured the distance between the center of the ball detected using each algorithm. In the

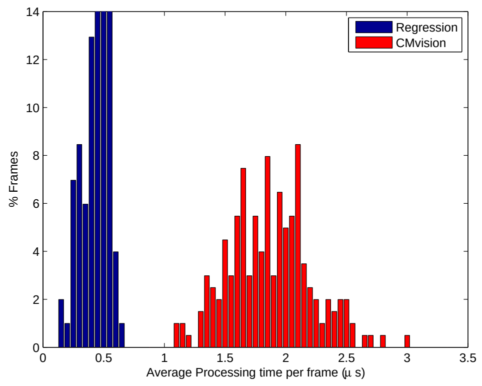


Fig. 5. Comparison of processing times using both approaches. Times correspond to the average processing time for one frame in a Pentium 4 at 3.20 GHz. The average was estimated by processing the same frame 1000 times.

graphics of figure 6 we show that the predicted position with our approach is included inside the bounding box of CMvision output 90% of the frames. Since the bounding box corresponds to a physical 3D region with the ball diameter, the error in centimeters in our algorithm will not be larger than ± 3 cm when compared to the CMvision error, independently of the error in term of pixels. From the remaining 10%, 1 frame has a distance between predictions of more than 100 pixels while the rest differs on around 20 pixels. The larger error is a consequence of an over conservative error detection. Our algorithm discarded one image which actually contained a ball because the confidence was bellow the threshold of 0.7. However, the remaining ones reflect the different ways both algorithms deal with occlusion. While CMvision always finds the center of the ball inside the orange segment in the image, our algorithm is capable of extrapolating the center to outside the visible part of the ball. An example of such extrapolation can be seen in image 4(a).

IV. Results Discussion and Conclusions

Results show that, in robot soccer, it is possible to leverage past experience to create simple and adequate models of objects without the need of computationally expensive algorithms nor explicit modeling of objects. By accumulating past images and using the current state of the art algorithms to provide ground truth, we gain access to an unlimited number of labeled data which can be used for training the coefficients of a regression. The resulting algorithm is faster than the one used for training but without affecting precision considerably. Furthermore, the algorithm is capable of identifying its own error, which allows for online validation of its results.

There were three important conditions contributing for the success of the algorithm: the object symmetry, the scenario invariance and chromatic simplicity and, finally, the existence of very accurate algorithms capable of providing ground truth for training.

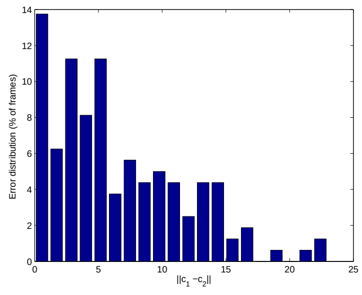
The object rotational symmetry simplifies most of the usual computer vision problems: we do not have to deal with rotations nor self occlusion. The ball has always the same shape independently of its position with respect to the robot, apart from a scalling factor. Other objects in the field do not share this property and would require the extraction of features more sophisticated than just the pixels value. The symmetry allowed us to construct a model based on a simple regression. If they had greater variance than the number of pixels, the mapping between objects and coordinates would not be linear.

The scenario invariance and simplicity in terms of colors simplifies the learning problem. In particular, it allows to segment the objects from the background and from each other. This, associated with the symmetry of the objects themselves, greatly reduces the dimensionality of our problem: we do not need to use all the image. Only those pixels which are of the same color as our interest object.

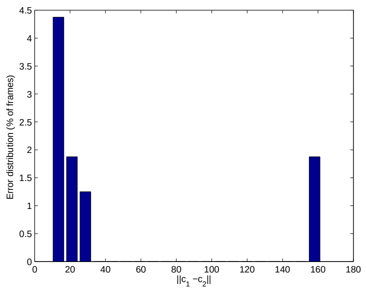
These two conditions justify our linear model relation between pixels and positions in the image. Consider an image from robot soccer after thresholding: all the ball pixels have the same value and we can discard all other non orange pixels as background. In a first order approach, we could compute the balls position by simply computing the mean of this thresholded image. Our algorithm goes a bit beyond simple mean: we can extrapolate balls positions and we can estimate the error of our detections.

Finally, the availability of very accurate algorithms such as CMVision, is essential to have the required training data. Without this tool, we would have to label the data by hand, which would be extremely time consuming and would prevent us from obtaining relevant training sets.

As open an issue for future work we consider the detection of objects in robot soccer with more complex features such as field lines and goals. These objects present themselves as a more interesting challenge since the linear approach is not expected to work: the objects are no longer invariant to rotations and suffer more from occlusion during the match.



(a) Both algorithms agree on the ball position



(b) Algorithms do not agree on ball position

Fig. 6. Distribution of the distance between the centroids obtained using both approaches. Results are separated according to whether the regression algorithm found the ball inside the bounding box of CMVision detection. Only a small percentage of the frames did not agree on the detection

Acknowledgment

This work was partially supported by the Carnegie Mellon/Portugal Program managed by ICTI from Fundação para a Ciência e Tecnologia. Susana Brandão holds a fellowship from the Carnegie Mellon/Portugal Program, and she is with the Institute for Systems and Robotics (ISR), Instituto Superior Técnico (IST), Lisbon, Portugal, and with the Department of Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh, PA. The views and conclusions of this work are those of the authors only.

References

[1] M. Nieuwenhuisen and S. Behnke, "Nimbro spl german open 2010 team description," 2010.
[2] E. Hashemi, O. A. Ghiasvand, M. G. Jadidi, A. Karimi, R. Hashemifard, M. Lashgarian, M. Shafiei, S. M. Farahani, K. Zarei, F. Faraji, M. A. Z. Harandi, and E. Mousavi, "Mrl team description 2010 standard platform league," 2010.
[3] T. Hester, M. Quinlan, P. Stone, and M. Sridharan, "Tt-ut austin villa 2009: Naos across texas," 2009.
[4] H. L. Akn, T. Mericli, N. E. Özkucur, C. Kavaklioğlu, and B. Gokc̈e, "Cerberus10 spl team," 2010.
[5] S. Czarnetzki, S. Kerner, O. Urbann, M. Hofmann, S. Stumm, and I. Schwarz, "Nao devils dortmund team report 2010," 2010.
[6] J. Brindza, A. Lee, A. Majumdar, B. Scharfman, A. Schneider, R. Shor, and D. Lee, "Robocup standard platform league team report 2009," 2009.
[7] P. D. K.-U. Jahn, D. Borkmann, T. Reinhardt, R. Tilgner, N. Rexin, and S. Seering, "Nao-team htwk, team research report 2009," 2009.
[8] "Team description paper 2010 austrian kangaroos," 2010.
[9] E. Polosetski, V. Sadov, A. Levy, A. Shilony, A. Sevet, E. Alon, M. Traub, G. Kaminka, and E. Kolberg, "Burst team report robocup 2009 spl," 2009.
[10] T. Röfer, T. Laue, J. Müller, A. Burchardt, E. Damrose, A. Fabisch, F. Feldpausch, K. Gillmann, C. Graf, T. J. de Haas, A. Härtl, D. Honsel, P. Kastner, T. Kastner, B. Markowsky, M. Mester, J. Peter, O. J. L. Riemann, M. Ring, W. Sauerland, A. Schreck, I. Sieverdingbeck, F. Wenk, and J.-H. Worch, "B-human team report and code release 2010," 2010, only available online.
[11] J. Bruce, T. Balch, and M. Veloso, "Fast and inexpensive color segmentation for interactive robots," in *In Proceedings of IROS-2000*, 2000, pp. 2061 – 2066.
[12] M. A. Turk and A. P. Pentland, "Face recognition using eigenfaces," in *Proceedings. 1991 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. IEEE Comput. Sco. Press, 1991, pp. 586–591.
[13] C. M. Bishop, *Pattern Recognition and Machine Learning*. Springer, 2007.