Real Time Object Recognition in a Dynamic Environment

An application for soccer playing robots

Tino Lourens and Emilia Barakova

GMD-Japan Research Laboratory
2-1 Hibikino, Wakamatsu-ku
Kitakyushu, 808-0135, Japan
http://www.gmd.gr.jp
{tino, emilia}@gmd.gr.jp

Abstract In dynamic environments, such as RoboCup [4], vision systems play a crucial role. In general, systems requiring real-time vision are either implemented in hardware, or as software systems that take advantage of the domain specific knowledge to attain the necessary efficiency. The goal of this paper is to describe a vision system that is able to reliably detect objects in real time and that is robust under different lighting conditions, this in contrast to most models used in robot soccer. The resulting objects serve as input for intelligent prediction of robot behavior [1].

1 Introduction

Fast sensing is advantageous for both biological and artificial systems. Humans can evaluate a visual scene in a fraction of a second. During this period a considerable amount of data is retrieved and processed. Humans very rapidly reduce visual data (the eyes receive most data per time unit) to relevant information, by using knowledge and adaptation. This paper describes a system where visual data is rapidly reduced by using knowledge about the environment. The system is able to recognize objects in real time, and is applied to soccer playing robots. The system is robust to (non-uniform and changing) lighting conditions and differences in color definition. This in contrast to virtually all vision solutions in RoboCup[1], which need tedious tuning for every game.

In most of the color vision systems the first step in data processing are extracting features, assigning pixels to classes, or a combination of both. In general, software solutions for object recognition are not even close to real time. For example, a well known fast method of feature (corners and edges) detection is SUSAN [5]. The newest generation of processing technology might be able to perform this method in real time. Nevertheless, the largest computational effort is needed to map the detected key-points into recognized objects. Hence,

[1] RoboCup is the robot soccer competition.

knowledge possibly in combination with selective attention is essential for real time vision applications, for general tasks in a real world environment.

Assigning pixels to classes has been proven to be successful in RoboCup. The most widely used approach in RoboCup is assigning pixels to color classes. Bruce et al. [2] constructed a thresholding method that is able to classify up to 32 different color classes in few operations. This method is attractive because of its efficient memory usage. We propose a simpler method that assigns a pixel to a class in a single operation by a lookup table.

Colored objects can be recognized as blobs under the assumption that objects are uniform. Bruce et al. [2] accomplished blob detection by color segmentation using run length encoding. The advantage of this method is that it is independent of any knowledge about the environment. The drawbacks of this method however include, poor recognition of partly occluded objects and the relatively high computational cost.

A more attractive approach is proposed by Jamzad et al. [3]. They make use of the perspective view, and state that 1200 single points are sufficient for object detection in RoboCup. In the future, robots are supposed to play against humans by the FIFA rules. The replacement of the boarding by white lines, last year, is a step in that direction. To meet the new requirements, there is a stronger need for detecting small objects, hence so-called scanlines are more suitable than single points.

The paper is organized as follows: Section 2 elaborates on color space reduction, spatial reduction, and cluster extraction to obtain real time object detection in a robot soccer playing environment. The paper concludes with a discussion and future research.

2 Object Recognition in a RoboCup Environment

Detection of objects for a soccer playing robot comprises of four stages: *Color space reduction*; in a RoboCup setting only few colors are relevant, therefore a natural image is reduced to these relevant colors in a single operation by using a preprocessed table. *Spatial reduction.* A perspective grid with a resolution of 10 cm reduces the spatial data to less than 20 percent, which is sufficient to recognize all objects. *Segmentation* is performed by clustering areas of identical color on the grid. *Object extraction*; the objects contain at most 3 different color clusters and are evaluated by size.

2.1 Color Space Reduction

The colors of all objects in RoboCup are defined. These colors are *orange* for the ball, *yellow* and *blue* for goals and corner poles, *green* for the field, *white* for the lines, *black* for the robots, and *cyan* and *magenta* to visually distinguish between teams. We omit cyan, because we are able to distinguish our robots from the others by team communication. We define these seven colors as *color classes* because they cover more than a single (r, g, b) color value.

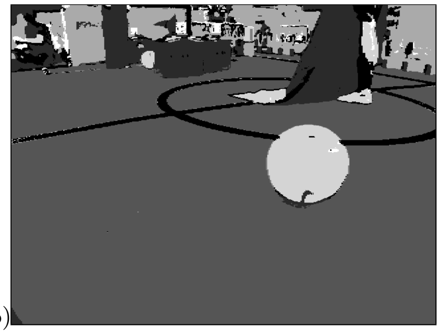
a)  b) 

Figure 1. a) Input image. b) Results of color reduction by attaching every pixel to one of the seven color classes. Note that the image is displayed by a normalized index, to clearly differentiate between the seven classes.

Color space reduction is accomplished by assigning every pixel in an image to a single color class.[2] This reduction is achieved in a single operation by a preprocessed table of $N_1 \times N_2 \times N_3$ elements, where N_k denotes the number of colors in channel $k \in \{1, 2, 3\}$.

A color triple $c = (r, g, b)$ in the conversion table t is assigned to exactly one color class:

$$t(c) = \begin{cases} \text{white} & \textsf{if} \;\; (\max - \min) < U) \wedge (\text{avg} > T) \\ \text{black} & \textsf{if} \;\; (\max - \min) < U) \wedge (\text{avg} \leq T) \\ i & \textsf{if} \;\; (\max - \min) \geq U) \wedge (|c' - i| \leq |c' - j| \quad \forall j \in C) \end{cases} \tag{1}$$

where U is a uniformity measure, T is a threshold, avg $= |c|$ is the average, min $= \min_3(c)$ is the minimum, max $= \max_3(c)$ is the maximum element value in c. The normalized color $c' = (N_1(r - min)/max, N_2(g - min)/max, N_3(b - min)/max)$, $C = \{orange, blue, yellow, green, magenta\}$ as the set of "real" color classes, and $|x|$ denotes the Euclidian (or L_2) distance. The following settings are used throughout the paper: $U = 50$, $T = 100$, and $N_k = 255$ for every k; the color centers of a class are used in its exact definition (blue $= (0, 0, 255)$, orange $= (255, 165, 0)$, etc.). These settings turn out to be very robust, but can of course be set to the the most desired settings in a particular RoboCup environment. Figure 1 illustrates the results of this algorithm.

2.2 Processing Data in a Perspective View

The robots are equipped with a Sony DFW VL500 CCD camera and have attached a Sony wide angle lens (x0.6 VCL-0637H). The camera is connected to a

[2] The Kmeans algorithm can be used for finding the most appropriate colors in environments where time constraints are less critical and where color settings are not a-priori known.

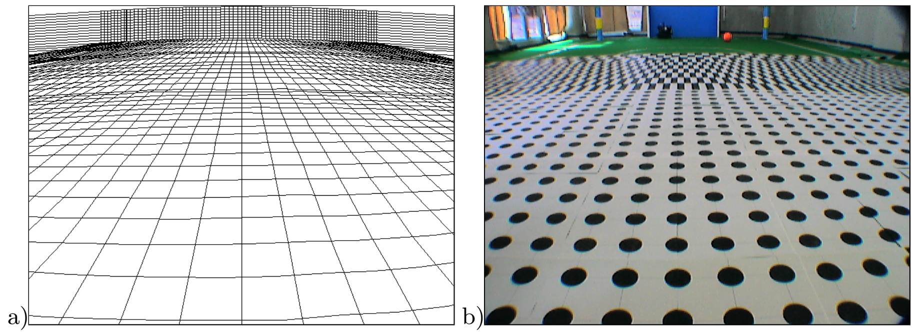


Figure 2. a) Constructed two-dimensional grid. b) Calibration pattern. Equidistant blobs, with centers at 10 cm distance frome each other, are used up to two meters. At a larger distance a 10x25 cm grating pattern is used.

notebook by the IEEE1394 firewire bus. The maximum capacity throughput of the camera is used, which results in 640x480 color (YUV422) images at a 30 Hz framerate.

In RoboCup soccer this setup is sufficient to detect a ball at a 10 meter distance. The image data can be strongly reduced if the sizes of all objects and their positions on the soccer field are taken into account. The smallest static object of interest in a soccer field is the white line (12 cm width) the next smallest object is the ball that has a diameter of about 24 cm.

A perspective grid (Figure 2a) is constructed by using a calibration pattern that contains equidistant blobs that are at 10 cm distance (Figure 2b). Such a grid highly reduces the data, and is still sufficient to recognize all objects in a RoboCup environment. The depthlines (7.08 percent) are completely scanned. Depending on the content of the depthlines, part of the 42 horizontal scanlines (8.75 percent) is scanned. In addition the grid gives the world coordinates from robot perspective (angle and depth to an object) which serve as input for the motion control as well as behavior prediction[1].

2.3 Object Extraction

The objects in a RoboCup setting are all uniform in color. However, in practice, differences in definition of color, reflecting surfaces, illumination of different light sources, as well as, light from outside can have a strong influence on the uniformity and appearance of a color. Actually, this is the major problem in RoboCup vision. The choice of assigning every pixel to an a-priori known small number of color classes (Section 2.1) is robust to these differences in appearance.

A grid (Figure 2a) is placed over the image with assigned color classes. All depthlines of the grid are used to scan for objects. On a single depthline all sequences of pixels that belong to the same color class and exceed a fixed minimum length are evaluated. The minimum and maximum y-coordinate of a sequence

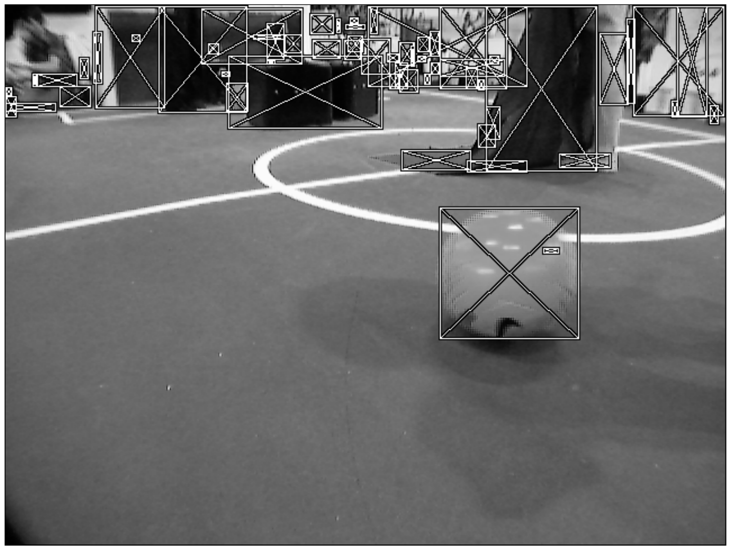


Figure 3. Extracted color clusters of Figure 1a are marked by a rectangle with a cross. Five different color clusters (orange, blue, yellow, black, and magenta) are evaluated. Unfortunately they can not be displayed properly in a grey scale image.

are taken and denote the vertical size of a cluster. Next, all intersections of the sequence with the horizontal scanlines are marked and followed in left and right direction, until another color class is encountered. The maximum and minimum x-coordinates of the followed horizontal scanlines determine the horizontal size of the cluster. If this cluster intersects with an existing cluster it is merged, otherwise a new cluster is added to the set of clusters in a single (time) frame. An illustration of all marked clusters of Figure 1a is given in Figure 3.

When all depthlines are followed in a single frame, the set of segments is complete and object recognition is performed. An object is described by a set of connected clusters and by its size. For example, a corner pole is a blue-yellow-blue object, with a diameter of about 20 cm and a height of about one meter; a ball consists of a single color cluster of about 24 times 24 cm.

2.4 Results

The proposed real time object recognition system is bound by the capacities of the video camera. The load on a 850 MHz P3 notebook is around 90 percent and that of a 2.2 GHz P4 desktop PC is around 35 percent. The initialization needed for the construction of color mapping table, grid, depth map, and map for size estimation are all fully preprocessed, which requires between 10 to 20 seconds, depending on the used machine.

Grabbing and mapping the data into a color segmented image, which can be considered as data retrieval, is most time consuming (240 ms wall clock time for 30 frames on the P4 desktop). Object recognition itself is far less time consuming (102 $\pm$ 5 ms). These measurements are obtained from data taken from two different RoboCup environments. In all cases the number of extracted color clusters are between 20 and 80 in a single frame.

3 Discussion and Future Research

In this paper a real time object recognition model in a RoboCup environment that is robust under varying illumination conditions and differences in color definition is presented. The model includes color and spatial reduction, by assigning pixels to color classes and by using a grid, respectively.

The method provides simple incorporation of additional color classes and allows segments to have more than one color class. For example, between the yellow and orange class a few intermediate color classes can be defined to give a more accurate distinction between a yellow goal and an orange ball. The green color is currently ignored, but segments very well from all other color classes and can be used to determine the field boundaries.

The current setup contains a vision system with a wide angle lens. Omnidirectional vision which is commonly used in RoboCup suffices in resolution. The 360 degree field of view results in simpler hardware and better self localization, and will be used on a robot that is under development. In the model only the grid needs to be recalibrated.

In autonomous systems where vision is included there are three major categorized data streams: *color*, *form*, and *motion*. These three streams are essential for any basic vision system. In real world applications that are not predefined like RoboCup, will require visual attention, knowledge, and learning methodologies to quickly extract relevant information from a huge amount of data.

Both authors are strongly in favor of incorporating biological models of vision and learning into the field of (semi) autonomous robotics.

References

1. E. I. Barakova and T. Lourens. Prediction of rapidly changing environmental dynamics for real time behavior adaptation using visual information. Accepted for 4th Workshop on Dynamic Perception, 2002.
2. J. Bruce, T. Balch, and M. Veloso. Fast and inexpensive color image segmentation for interactive robots. In *IEEE/ISJ International Conference on Intelligent Robots and Systems (IROS 2000)*, October 2000.
3. M. Jamzad, B. S. Sadjad, V. S. Mirrokni, M. Kazemi, H. Chitsaz, A. Heydarnoori, M. T. Hajiaghai, and E. Chiniforooshan. A fast vision system for middle size robots in robocup. In *RoboCup 2001*, 2001.
4. H. Kitano, Y. Kuniyoshi, I. Noda, M. Asada, H. Matsubara, and E. Osawa. Robocup: A challenge problem for ai. *AI Magazine*, 18(1):73–85, 1997.
5. S. M. Smith and J. M. Brady. SUSAN - a new approach to low level image processing. *Int. Journal of Computer Vision*, 23(1):45–78, May 1997.