Adaptive Arc Fitting for Ball Detection in RoboCup

Genevieve Coath
University of Melbourne
Dept. of Mechanical & Manufacturing Eng.
Melbourne, Australia
gcoath@ieee.org

Phillip Musumeci
James Cook University
School of Information Technology
Cairns, Australia
p.musumeci@ieee.org

Abstract

This paper presents an edge-based ball detection system for use in RoboCup robots. The algorithm can be economically coded for real-time operation in integer maths digital signal processing units. It allows colour dependencies in existing ball detection algorithms to be relaxed. This work is undertaken as part of the world-wide RoboCup project which aims to stimulate robot development by investigating difficult test problems.

1 Introduction

The international RoboCup competitions stimulate development in mobile robots that function collectively and incorporate a team approach to problem solving. While the RoboCup challenge is only part way towards its stated 2050 goal of achieving autonomous humanoid robots that can defeat humans in soccer, the existing systems are relatively complex and display team cooperation. A hierarchical view of the RMIT University robot system shows a top-level strategy module which coordinates the operation of robot team members using sensory inputs such as vision and touch, and implements strategies via actuator outputs including motor control, steering, and ball kicking systems.

This paper presents research done at RMIT University to enhance the vision system of its middle league (F2000) robots. In particular, the strict colour dependency of previous ball detection systems has been relaxed by developing an adaptive arc identification and location system that processes image data containing edge information. Analysis of arc parameters such as radius, location, and recent trajectory is then used in revised ball identification schemes.

Examples are presented for a number of edge extraction techniques applied to colour image field data, and the adaptive arc detection scheme is shown applied to a ball in full view and also a ball obscured by a (worst case) curved object.

This paper provides a brief description of the existing image processing system in order to set the context for the arc-based scheme. Some preprocessing algorithms used to generate edges are described, including schemes with simplified computation to assist in real-time application, and finally the edge tracking algorithm is presented with field images.

2 Visual Environment

In current competitions, the robot environment is tightly controlled so that robots can interact with a world that can be understood with very limited knowledge. The field lighting levels are specified so that robots do not need to handle significant dynamic lighting effects. In addition, the ball and each goal region may be uniquely coloured to simplify robot orientation and target acquisition. The dependency of current robots on the use of colour in identifying objects is highlighted by the need, in some tournaments, to remove coloured items from spectators because they can confuse the vision systems.

Following the Melbourne 2000 event, a RoboCup rules debate suggested the possibility of increasing the complexity of the environment in which the robots must compete, through such changes as removal of field walls and unique colour schemes. This debate has lead to the current work which attempts to incorporate additional shape information into ball identification while reducing the relative importance of colour.

2.1 Overview of Existing System

The existing vision system [UVD, SBK] uses a Pulnix 1/3" CCD colour analogue camera attached to

a custom interface board that connects a Brooktree video capture device to a Texas Instruments DSP (TMS320C6211).

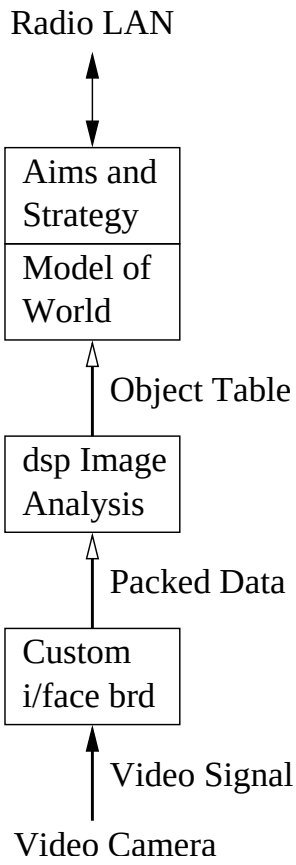


Figure 1. System Information Flows

Figure 1 shows the information flows in the three processing stages. The first stage accepts a video signal from the camera, digitises it, and reduces the video resolution by coding the digital data to 16 bits/pixel to allow more efficient pixel data packing in DSP memory. In the second stage, the enhanced direct memory access (EDMA) controller provided by the DSP is programmed to be a state machine which retrieves image frames from the video capture device at a rate of 25 frames/second and an image size of 450x450 pixels with little CPU intervention.

The DSP analyses the image data and creates an object table with attributes of type (line, solid, etc.), colour, and image coordinates to describe each element. Field information is based on straight line edges detected by a modified Hough transform using integer maths. Information relating to other robots and the ball is derived from colour information.

The ball detection system analyses image scans and detects colour transitions. Regions that match the colour of the ball are then analysed and an estimate of center and diameter are generated. If these parameters are within suitable limits, a ball detection occurs.

To reduce computation, the estimated diameter is obtained from the widest horizontal scan segment with the desired ball colour. If a closer object obscures either side of a ball, then the estimate of center and diameter will be incorrect. Ideally, both the largest vertical and largest horizontal dimension should be used to derive diameter estimates. However, the use of overhead lighting systems in competition venues may cause shadows which make the detection of the bottom edge of the ball unreliable.

The final stage is implemented by interfacing the DSP output to a robot-mounted laptop which processes aims and strategies, and estimates range information based on the vertical position of an object within an image. The laptop also handles the other sensor inputs such as proximity detectors, and outputs such as motor controller commands.

2.2 Deployment of Image Analysis Algorithms

The system was developed by incorporating a shape-based ball identification algorithm into the existing colour-based ball detection scheme. The new algorithm initially provides an additional test of results from the previous ball detection system.

3 Edge-based Arc Detection

Arc detection is used to determine the ball center and diameter as a full circle is not available when the ball is partially obscured. Because there are potentially many other arc sources in a field, additional information such as dimensional constraints, relationship to previous ball sightings, and colour is used to filter out spurious responses.

To detect arcs, edges in the image are enhanced and a contiguous set of straight line segments is constructed between adjacent pixels that may form an edge. By interpreting the straight line segments as chords of an arc, perpendicular projections from each chord are used to identify the centers of these potential arcs. A cluster of centers will then confirm and identify an arc.

3.1 Edge Detection

The edge information is obtained by applying a spatial differentiation operator to the image — see for example [SBA]. For colour independence, this operator can be fed a preprocessed image such as image energy i.e. an L_2 norm is applied to the raw image, or the absolute value of the pixel i.e. an L_1 norm is applied to the raw image. An advantage of preprocessing with an L_1 operator before applying the spatial differentiation operator is a reduction of computation. The application of these operators is now detailed.

Figure 2 shows an ideal ball image sampled horizontally, an image energy curve, its first derivative, and its second derivative.

Any rapid changes in image energy I across a small region give rise to higher peak values in the differen-

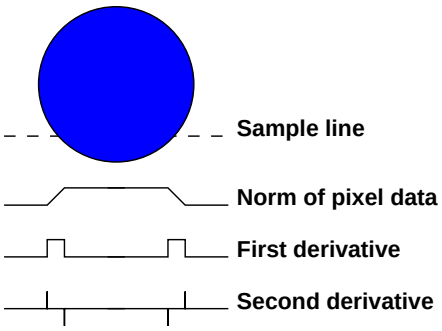


Figure 2. Ideal Ball

tiated image. An approximation to a one-dimensional differential operator at position $x = i\Delta$ is $\frac{dI}{dx} \approx \frac{I(i+1)-I(i)}{\Delta}$, where Δ is the spatial sample interval in the x direction. Extending this to two dimensions gives

$$\frac{\partial^2 I}{\partial x^2} + \frac{\partial^2 I}{\partial y^2} = k\,(I(i+1,j) + I(i-1,j) + I(i,j+1) + I(i,j-1) - 4I(i,j))\,;\mathrm{k} = \mathrm{constant}.$$

Applying this operator to the typical field view of a robot soccer field as shown in Figure 3(a) gives the result shown in Figure 3(b). Previous research in [ERD,UVD] shows that binarisation of RGB components of the image, in an image preprocessing stage prior to differentiation, leads to an enhanced edge display as shown in Figure 4(a). This provides the added benefit of data reduction.

A further improvement in edge definition may be possible for a typical field view if the preprocessing includes: a band-pass filter centered on the target colour; and a norm operator to generate monochrome image data where the maximum output corresponds to the target object's colour. Preprocessing is then completed with binarisation and differentiation to yield the image in Figure 4(b). In effect, this recognises that *typical* field objects have a colour that is clearly different to the target. Note that if the target has a colour that matches one of the dimensions used to represent colour, e.g. red, green or blue in an RGB format, then further computation reductions present themselves.

3.2 Radius Projection

Given a region of interest, our algorithm chooses as a reference a pixel in the edge enhanced image with significant energy[1] i.e. point p0 in Figure 5. Then, a nearby pixel p1 with significant energy is located such that the two points form a chord of length $> l$. A perpendicular projection through the mid-point of chord p0-p1 indicates potential locations for the center of the arc p0-p1. This process is then repeated for a sequence of pixels pairs located on the potential arc, and the intersections between the sequential projections leads to a cluster of potential center points. This example shows two chords p0-p1 and p1-p2 with projections intersecting at a potential arc center.

The minimum length constraint l is imposed on each chord because the location of each point is spatially discretised based on the pixels in the sampled image, and so very small length chords would introduce excessive error in the calculation of centers.

When this algorithm is applied to an unobscured ball image, the result shown in Figure 6(a) is obtained. In this example, the algorithm has been run a number of times with slightly different starting points, so that a larger number of projections have been produced. The clustering of intersections between successive projections gives the location of the center of the arc (or in this case, circle).

Note that, while we have highlighted the projections in this image, the actual system is only concerned with the intersection point of pairs of equations that are illustrated by the projections from pairs of adjacent chords.

An advantage of this algorithm is that it can function on obscured balls. In Figure 6(b), a number of irregular objects are covering parts of the right hand side of the ball which is also slightly distorted vertically. A number of potential centers are identified, in rough proportion to the size of the corresponding arc. In this example, an arc associated with the ball is a dominant image feature so the ball center is correctly found. If the ball is mostly obscured, then the result is unpredictable without additional information. The algorithm is now described in detail.

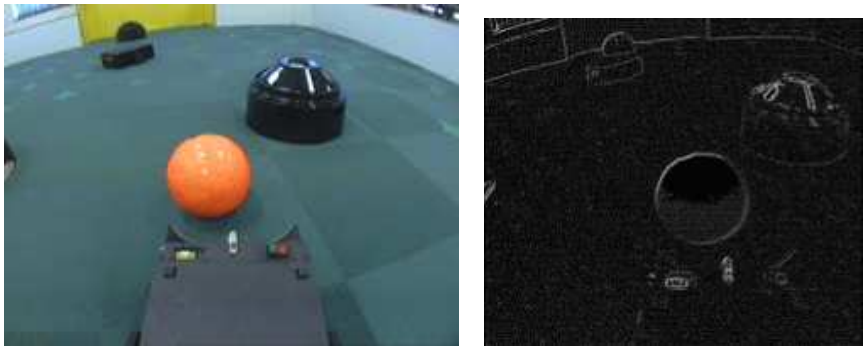
3.3 Arc Location Algorithm

As mentioned previously, the points located on the potential arc or circle have a minimum spacing of l enforced. The algorithm starts from the initial point identified for the largest edge image pixel, and then generates a sequence of points along the potential arc or circle.

1. Obtain edge image via application of two-dimensional differentiation operator or similar transform;

2. Find the magnitude of the maximum pixel e_{max}

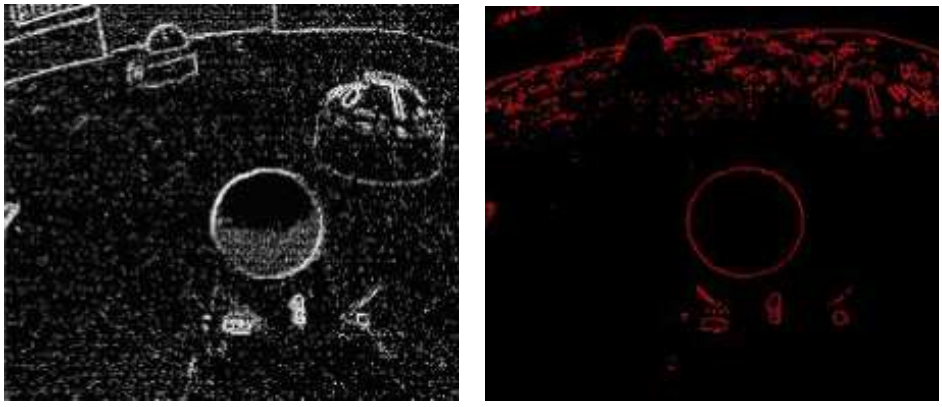
[1] In off-line testing, the initial pixel may be chosen as the largest value in the edge enhanced image for comparison purposes.



(a) Typical RoboCup Field

(b) Differentiated Image (smaller after processing)

Figure 3. Field Views



(a) Binarised & differentiated

(b) 1-colour, binarised & differentiated

Figure 4. Field Views with Preprocessing

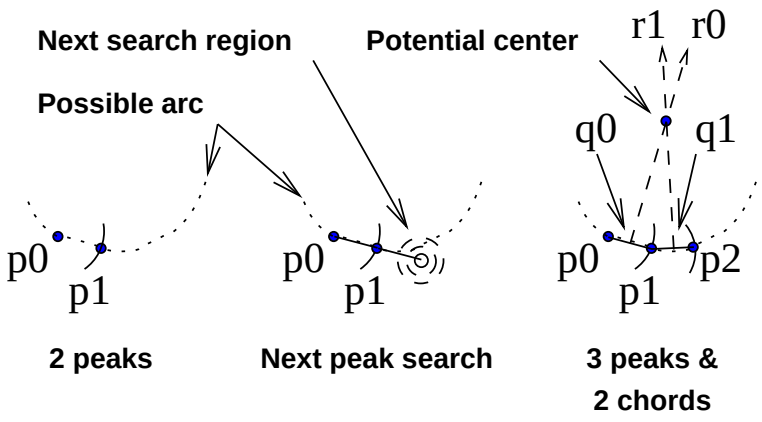
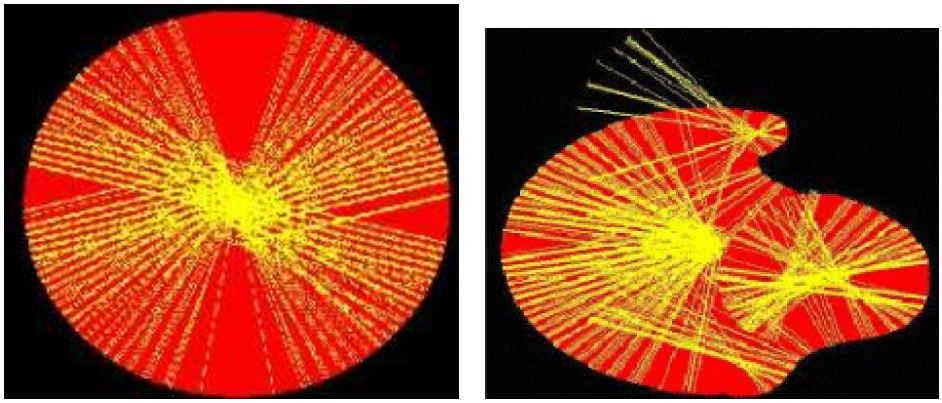


Figure 5. Edge Tracking (3 stages shown)



(a) Test Circle Detection (b) Obscured Circle

Figure 6. Projections for Center Location

in edge image, and choose α such that threshold $e_t = \alpha e_{max}$ can be used as a lower bound to identify significant peak values $e \geq e_t$. In our work, we typically use $\alpha = 0.9$;

3. Designate the first point $p_0 = \{x_0, y_0\}$ where $e(p_0) = e_{max}$;
4. Find the next point, $p_1 = \{x_1, y_1\}$, where $e(p_1) \geq e_t$ and $\|p_1 - p_0\|^2 > l^2$, by conducting a localised horizontal and then vertical scan for a pixel with significant energy in the next search region found by extending the current chord (illustrated in the 2^{nd} stage of Figure 5). Note that calculation of a square root to ensure adequate spacing of points p_i is avoided by testing distance squared;
5. Identify the chord mid-point $q_0 = \{(x_1 + x_0)/2, (y_1 + y_0)/2\}$. The perpendicular projection from point q_0 to r_0 is defined in terms of a vector dot product i.e. $\overline{r_0 - q_0}.\overline{p_1 - p_0} = 0$;
6. Repeat the previous two steps to generate the point sequences $\{p_0, p_1, p_2, \ldots\}$, $\{q_0, q_1, q_2, \ldots\}$, and a set of equations for $\{r_0, r_1, r_2, \ldots\}$;
7. Choose to solve for intersection equations for $\{r_i, r_{i+1}\}$ i.e. sequential projections, to generate potential center locations c_i. Solving for intersection of adjacent projections also minimises storage of point data;
8. Finally, an average of the coordinates of potential center locations c_i gives an estimate of the arc (or ball) center. For more complex field views, the image may be segmented and the average c_i in a segment suggests a localised arc center.

The potential center locations c_i can be interpreted as a vote for a potential arc center and therefore a vote for the presence of a ball. An additional filtering stage under development is to reject c_i values that are located more than a specified distance from a potential center, in order to obtain a more tightly bound cluster that is not influenced by outliers. For the current robots, an ad-hoc filtering stage rejects a potential center that fails to have 60% of its center estimates within ± 20 pixels of the horizontal center or 60% of its center estimates within ± 30 pixels of the vertical center. However, this scheme needs to be recast in relation to the object size.

Where ambiguous results are obtained, additional information must be used to clarify the location of a single ball. For the RMIT University systems, this arc detection algorithm is being used to enhance reliability of other ball location schemes.

3.4 Algorithm Tuning

The size of l can be found by testing the algorithm on the top and bottom arcs of a test ball, and then on the far left and far right arcs of a test ball. The smaller l is, the better tracking is obtained for the edge. However, projection errors due to the spatial discretisation due to horizontal and vertical (scan) image sampling become significant if l is made too small. For the range of ball sizes encountered when viewing a ball at a range of distances, l is chosen so that the chord subtending each arc is approximately 10 or more pixels long.

The threshold selection parameter α is chosen so that a suitable number of preprocessed edge peaks are

generated — if α is too small, the edge tracker can be confused by noise while if α is too high, any arcs may be undersampled. It is possible to add a feedback loop that adapts α up or down in order to achieve a reasonable number of peaks. However, the effect of α is not unduly sensitive in typical field lighting conditions due to the use of preprocessing stages (binarisation and differentiation).

4 Conclusion

We have presented an edge tracking algorithm and arc location scheme that has been successfully applied to RoboCup field images. A number of image data preprocessing strategies have been discussed, with features such as data reduction and colour-centered edge enhancement. The edge tracking algorithm provides a means of locating arcs and circles to identify the soccer ball. In the implementation on the TMS320C6211 DSP, the actual chord and associated perpendicular projection equations were implemented with minimal transcendental function calls because this DSP does not have a floating point unit.

The availability of alternative ball detection algorithms allows comparison testing and also focusses attention on benchmarking. Previous RMIT RoboCup systems have concentrated on getting a single algorithm working for each critical stage so comparison testing was not possible. In the current robot architecture, target tracking is performed at higher levels so comparisons will apply to the processing of individual image frames with the assumption that a recently acquired target allows analysis to be restricted to smaller parts of an image over a short time interval.

As the whole system must operate in real-time, it is desirable that partial results obtained in the sample time be of use — the algorithm proposed here can identify a ball before all of the visible circumference has been traversed if a suitable number of potential centers has been found i.e. an early result is possible without identifying all chords or processing all significant edge pixels.

We acknowledge the RMIT University RoboCup teams of 1999 and 2000, and specifically thank Mark Makies, Duy Khuong Do, Sy Quang Dinh, Karl Fantone, Alan Harvey, and Shereene Jesurajah. We thank Colin Lemmon (JCU) and the anonymous reviewers for improving this paper.

5 References

ERD "Machine Vision - Theory, Algorithms, Practicalities", E.R. Davies, 2nd edition, Academic Press, 1997.

SBA "Diffuse edge fitting and following: a location-adaptive approach", A.L. Shipman, R.R. Bitmead and G.H. Allen, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. PAMI-6, no. 1, January 1984, pp. 96-102.

UVD "Field Understanding System and Vision System", RMIT United Vision & DSP Group, 2000, Internal Report.

SBK "RoboCup 2000: Robot Soccer World Cup IV", Peter Stone, Tucker R. Balch, Gerhard K. Kraetzschmar (editors), Lecture Notes in Computer Science 2019, Springer 2001, ISBN 3-540-42185-8.