Fast Image Processing and Flexible Path Generation System for RoboCup Small Size League

Shinya Hibino[1], Yukiharu Kodama[1], Yasunori Nagasaka[2], Tomoichi Takahashi[2], Kazuhito Murakami[1], and Tadashi Naruse[1]

[1] Aichi Prefectural University, Nagakute-cho, Aichi, 480-1198 Japan
[2] Chubu University, Kasugai, Aichi, 487-8501 Japan

Abstract. Two key features of successful multi-robot systems in RoboCup are robustness of a vision system and optimization of feedback control in order to reach the goal point and generate the action under the team strategy. This paper proposes two new methods. One is a fast image processing method, which is coped with the spatial variance of color parameters in the field, to extract the positions of robots and ball in 1/30 sec. The separation problem in the interlaced format image is solved. Another one is a path generation method in which the robot approaches the goal by changing its direction convergently. With these two algorithms, the real time processing system is realized by generating a stable path under a low quality input image.

1 Introduction

In the physical multi-robot systems in RoboCup[1],[2], it is important to raise the robustness of vision system and to give an optimal feedback in order to control a robot to reach the goal point. There are several technical problems to be solved from the viewpoints of image processing; for example,

- R-, G-, B-values are not constant by shading in the region of the ball in the image, since a golf-ball used in small-size league is a sphere with dimples.
- Lighting is not always uniform in the game field.
- It is difficult to recognize an object moving at high speed, because the object would appear as split patterns in an interlaced format image.

On the other hand, from the viewpoints of processing time to control robots,

- Calculation time should be short because all objects such as a ball and robots move at high speed.
- A small size input image is desirable, however, the quality of image is getting low.

This paper proposes two new methods. One is a fast image processing method, which is coped with the spatial variance of color parameters in the

G.A. Kaminka, P.U. Lima, and R. Rojas (Eds.): RoboCup 2002, LNAI 2752, pp. 53–64, 2003.

field, to extract the positions of robots and ball in 1/30 sec. Although many labeling methods for the connected component have been reported [7,8,9,10], our method is focused on the separation problem in the interlaced format image. This problem is solved with a small overhead for real time image processing. Another one is a path generation method in which the robot approaches the goal by changing its direction convergently. With these two algorithms, the real time processing system is realized by generating a stable path under a low quality input image.

2 Image Processing System

In the small size robot league, an image obtained from a camera installed on ceiling is usually processed to get the position and the direction of each robot and the position of a ball. We also use such images. It is necessary to develop an image processing algorithm which works well for low quality images, since an off-the-shelf camera and frame grabber are used in our system. In this section, we discuss such an algorithm. Throughout this section, a word 'ID' and an 'object' are used for the identification number of each robot and an object region such as a ball, color markers and team markers in the image, respectively.

2.1 System Configuration

Figure 1 shows a configuration of our image processing system. The Video4Linux was used as a driver of frame grabber. The processing is as follows. The frame grabber gives the yuv-image (YUV422). Then, search range is calculated by using history information. For the search range, the segmentation algorithm is applied in order to extract the segments of the object, where the segment is a connected component of pixels which have the color value of specified range. The extracted segments are merged to make a candidate of object by using a labeling algorithm. Finally, the true objects are selected by calculating the size and the location of each object.

It is difficult to get sharp images, since an off-the-shelf camera and frame grabber are used in our system. An example of grabbed image is shown in figure 2. In the figure, a boundary of the object area is blurred and an unexpected scan line is contained. For such an image, the above system should work well. To make a system be robust, we developed a new labeling algorithm which is discussed in sec. 2.3

2.2 Color Image Segmentation for Object Extraction

6 colors should be detected to identify objects, i.e. 1 color for ball (C_1), 2 for team markers(C_2,C_3), and 3 for ID markers (C_4,C_5,C_6). The system classifies each pixel into one of 6 color clusters $C_i (i = 1, 2, ..., 6)$ or the other in the color segmentation process,.

This process utilizes a color image segmentation algorithm developed by CMU[3]. In the algorithm, 32 different colors can be segmented simultaneously.

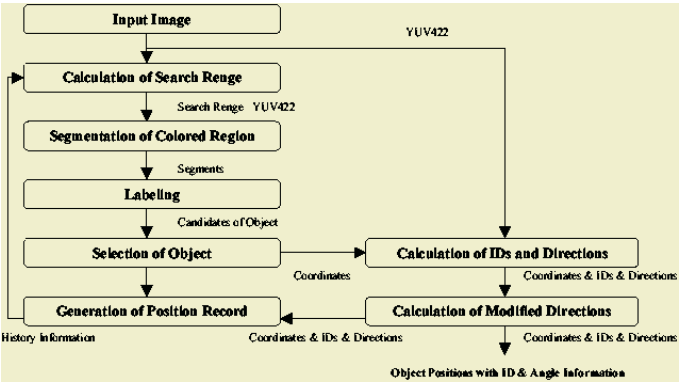


Fig. 1. Image processing system

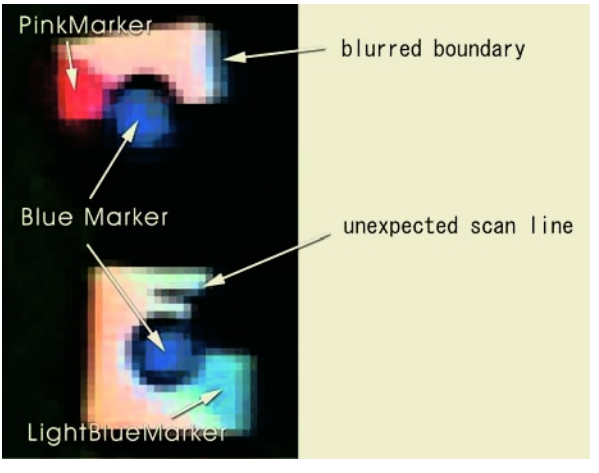


Fig. 2. Example of grabbed image

Thus, we assigned 5 different colors for 1 color cluster, which can compensate the subtle color variations of the object caused by the lighting condition. The image is segmented in this way.

An experimental result under the lighting condition, 700-1500 luxs (which is specified in the regulation of RoboCup small size league), is shown in Table 1. Very high recognition rate is achieved, especially for the ball and team markers (over 99%).

2.3 Labeling Algorithm for Object Extraction

Next step is the labeling process for the segmented image to extract a candidate of the object. Conventional labeling algorithms or methods do not work well for interlaced format image[7,8,9,10], since robots and ball move at high speed.

Table 1. Recognition rate of color markers and balls. It is measured 1000 times at each of the following points; the goals of the own and opponent sides, a center of the field and 4 corners. Green, pink and light blue are ID markers. Ball and team markers should be strictly recognized

Object	Rate(%)
Ball	99.98
Yellow Marker(Team Marker)	99.80
Blue Marker(Team Marker)	99.35
Green Marker	96.23
Pink Marker	98.31
Light Blue Marker	95.28

Even though there is a method which processes either even or odd field of an image, it sacrifices the resolution. Independent processing for even and odd fields makes it difficult to unify the results. In order to overcome these difficulties, a new labeling algorithm called **diagonal alternate spread labeling** was developed. It can cope with the interlaced format image which has blurred objects of moving robots. In the following, i and k denote the scanning parameters for the x coordinate and j and l for y coordinate. For the simplicity of explanation, it is assumed that the image is binary and an object is black.

Our labeling algorithm is given as follows.

Algorithm. diagonal alternate spread labeling

Step0 Let A and B be a set of pixels, respectively, and put $A = \phi, B = \phi$. Let num be a label number, and put $num = 0$.

Step1 Scan the image. If a black pixel (i, j) which is not labeled is found, put it into the set A.

Step2 Do the following.
1) For a pixel (i, j) in the set A, if it is not labeled, label it with num. Then, search the following 8 pixels.

$$(i+2, j+2), (i+2, j-2), (i-2, j+2), (i-2, j-2),$$
$$(i+1, j+1), (i+1, j-1), (i-1, j+1), (i-1, j-1)$$

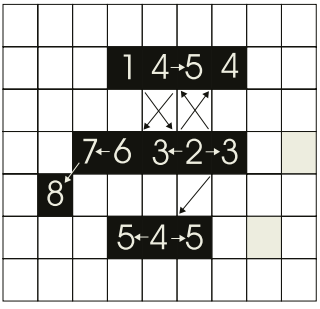
For each pixel, if it is black, put it into the set B.
2) For a pixel (k, l) in the set B, if it is not labeled, label it with num. Then, search the following 4 pixels.

$$(k, l+1), (k, l-1), (k-1, l), (k+1, l)$$

For each pixel, if it is black, put it into the set A.
3) Repeat *Step2* while new black pixels are gotten.

Step3 Increment num and repeat *Step1* and *2* while there are unlabeled black pixels.

Figure 3 shows a labeling example.



This is an example of segments recognized as one object. The number added to the pixel is a processing order in Step 2 of the labeling algorithm. Since the scan is sequential, the first pixel of an object is always the upper left pixel which is numbered 1

Fig. 3. An example of labeling

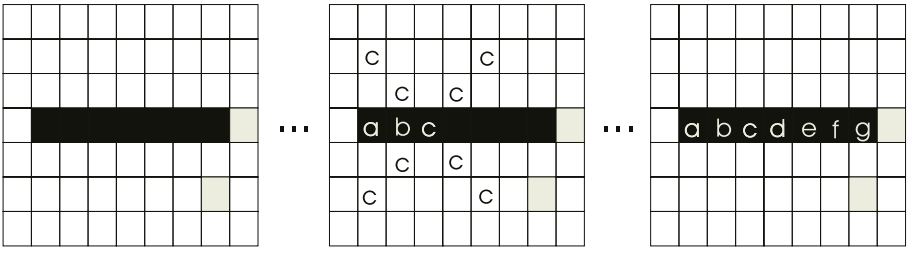
Effectiveness of the Proposed Labeling Algorithm. This algorithm can detect line segments, where each of them appears by every two lines, as one object as shown in figure 3. This solves the interlace problem of alternate appearance of black and white lines when the object moves. This algorithm can detect moving objects up to speeds of $v_{max}[cm/sec] = d[cm]/(1/60[sec]) = 60d[cm/sec]$, where, d is the size of an object (ex. $240cm/sec$ for a $4cm$ ball). If the object moves over the maximum speed v_{max}, it is completely separated into 2 objects in the image.

This algorithm has another unique characteristics that a vertical or horizontal line segment of length n and width 1 (pixel) would be divided into n pieces of different objects of size 1 (pixel), because the search to the horizontal and vertical directions occurs after the search to the diagonal directions in Step 2. An example is shown in figure 4. This algorithm works well for such straight line segments that are unexpectedly generated around the boundary of the field or objects shown in figure 5, because the system would delete small size(pixel size) objects as noises by a simple thresholding.

2.4 ID Recognition

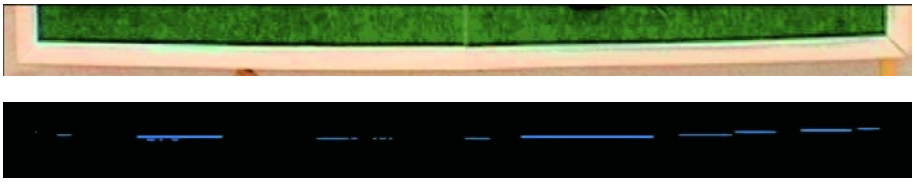
ID Recognition Method. The image processing module recognizes the ID and the direction of each robot in our system. Each robot has a black and white plate on its top as shown in figure 6. In the image, the plate region is detected as an object by the image processing module. The ID of robot can be decided by measuring the sector of white region. To do so, we prepared the reference table. The size of the table is 40 or 44 entries[1], that is suited for the size of the robot in an image. The reference table consists of entry number, angle, x and y distance from the center of the plate as shown in table 2.

[1] The size of a table is changed according to the situation, because the field size in the image changes with the camera arrangement in the hall.



This is an example of object that each pixel is recognized as a different object. Black pixels are candidates of objects, however, each pixel is labeled with different alphabet (a,b,...,i). White pixels with an alphabet show that they are searched in step 2 in conjunction with the black pixel with the same alphabet (Labels d,..,i are omitted here)

Fig. 4. An example of labeling to a straight line



Blue color regions appear near a field boundary by the characteristics of frame grabber. The color segmentation algorithm detects them as the candidate segments

Fig. 5. An example of straight lines caused by color blur

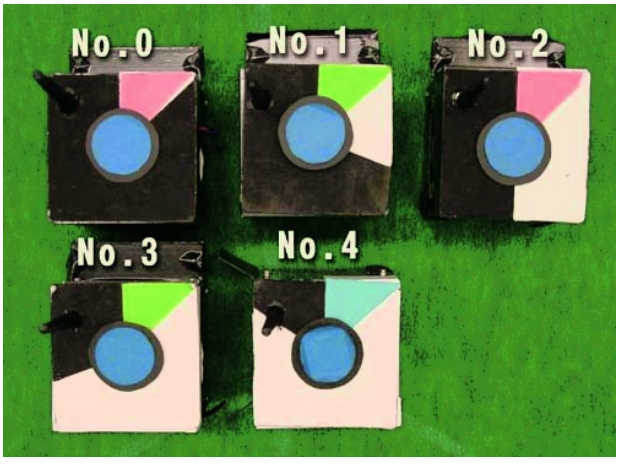


Fig. 6. ID plate

This table is applied to the robot region as shown in figure 7. The image processing module detects a center of the object and refers the pixel values which are pointed by the table entries. The module saves the entry number

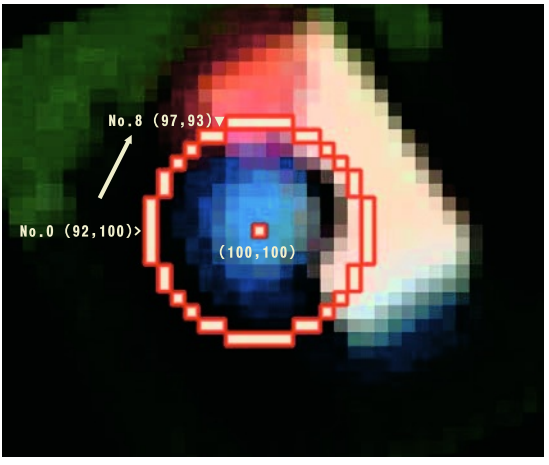


Fig. 7. Applying the reference table to the robot region

Table 2. ID reference table

Entry No.	Angle	X distance	Y distance
0	0	-8	0
1	7	-8	-1
2	14	-8	-2
3	24	-7	-3
4	30	-7	-4
5	38	-6	-5
6	50	-5	-6
7	58	-4	-7
8	65	-3	-7
9	75	-2	-8
10	83	-1	-8

First 11 entries of the 44 entries table are shown

corresponding to the point that the pixel value changes from white to black(it is No.8 in fig.7), and saves the number of pixels whose color is white. In addition, as RoboCup rule allows ID plate to attach other color markers other than black and white, we have set up all markers other than black to be recognized as white by thresholding. It is not difficult to decide this threshold.

This processing is applied to each robot. The ID is obtained by counting the number of pixels judged to be white, and the direction of the robot is given by the table entry corresponding to the pixel whose value changes from black to white.

Resolution and Verification. The ID decision does not depend on the direction of the robot, since the reference points are arranged around the circle. The

Table 3. Measured direction

Measured direction	Frequency
172	165
180	731
188	96
196	8

Measured 1000 times

angle resolution is about $8°(= 360°/44)$. This processing is operated only once to each robot, so the processing time is negligible.

Table 3 shows the accuracy of measured direction. In the experiment, a robot was placed in the horizontal direction and the direction was measured 1000 times. It is clear from the table that 73% of the measurements give the right direction, 26% give the directions with the error of $\pm 8°$.

3 Path Generation

Although we have designed the image processing system with angle resolution of 8° so far, the accuracy is not enough to control the direction of the robot. Inaccuracy of the direction should be compensated in the path generation system.

The path generation system generates and evaluates a path from a current position to a target position, where the target position is given by the strategy system[4]. Moreover, there is a constraint to control the direction, since our robot has only 2 wheels.

For the robot with 2 wheels, Cornell Big Red [5] generated the path by using polynomials in RoboCup '99. CMUnited-99 [6], also in RoboCup '99, proposed a method to control the direction of robot depending on the difference between the direction of the target point and the forward direction of the robot at the target point. We propose, in this section, a method to control the direction of robot depending on the curvature of a geometric path and the difference between the direction of the target point and the robot's orientation. With this method, curved paths shown in figure 9 are generated.

3.1 System Configuration

A path generation system consists of a path generator, a path evaluator, an auxiliary target point generator, a simple predictor and a velocity generator, as shown in figure 8. This system works as follows;

1. The path generation system receives the target position, the robot direction and the action (stop, shoot, etc.) at the target position from a strategy system.
2. The path generator generates a working path by using a curvature control variable and sends it to the path evaluator. The path evaluator estimates the required time to arrive at the target position and finds out the path not to violate the RoboCup soccer rules.

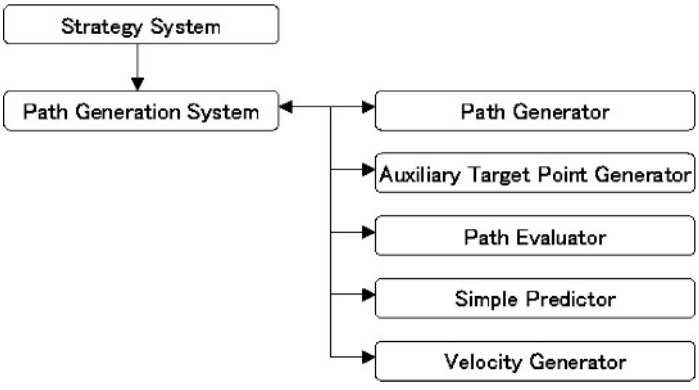
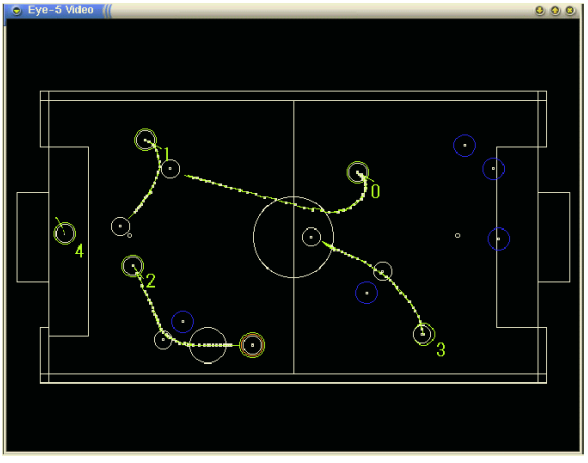


Fig. 8. Path generation system

3. Determine an advancing direction of a robot based on an evaluation result.
4. If there is an obstacle on the path or if robot direction at the target does not satisfy the given direction, the auxiliary target point is added for avoiding collision or for satisfying the direction at the target. Then, a new path is calculated again.
5. The new path is evaluated again.
6. Iterate step 3 - 5 by modifying the value of curvature control variable, and get an optimal path.
7. The velocity generator generates the velocities of wheels which move on the optimal path.
8. The simple predictor corrects the velocities of wheels to compensate the delay of processing time of image processing system. The corrected velocities are sent to the robot through a radio system.

3.2 Path Generator

The degree of control freedom is two and the control of direction is limited, because our robot has two wheels. Considering this and the fact that the target position where each robot should go changes from hour to hour, it is realistic to generate a path which the robot approaches the target position by asymptotically changing its direction. We call this a **direction converging path generation**. Figure 9 shows the paths generated by this method. In the figure, paths are superimposed on the field image. Each double circle with a number is a current position of our robot and the end point of each path is a target position. A triple circle at the target position is a ball. The dotted circles are opponent robots. If opponent robots stand on the generated path, subtargets are put near the opponents to avoid collision. In the case, the robot goes to the subtargets at first and then goes to the target. The robots number 2 and 3 have a subtarget and the robots number 0 and 1 do not. The goal keeper (number 4) does not move



This figure shows generated paths in a certain time slice of a real game. Each dotted circle is an opponent robot. Each double circle with a number is our robot and each solid circle is a target point. A triple circle is a ball. For robot 3, the strategy system gives a subtarget point in order to avoid an enemy. So is for robot 2. In case of robot 2, one more subtarget point is generated (a large solid circle in front of the ball) to adjust a robot direction as facing the ball

Fig. 9. Planned paths in a real game

in this figure. Note that the path is generated only to determine the velocity for next Δt time step[2]. The path is newly generated every Δt time step.

Our path generation algorithm is given as follows.

Algorithm. direction converging path generation

Step1 Let $p(=(x,y)),(v_l,v_r),\mathbf{u}$ be position, velocity (of left and right wheels) and forward direction vector of a robot, respectively. Let t be the current time. Calculate the curvature κ and robot velocity $v = \frac{v_l+v_r}{2}$. See literature [5] for detail.

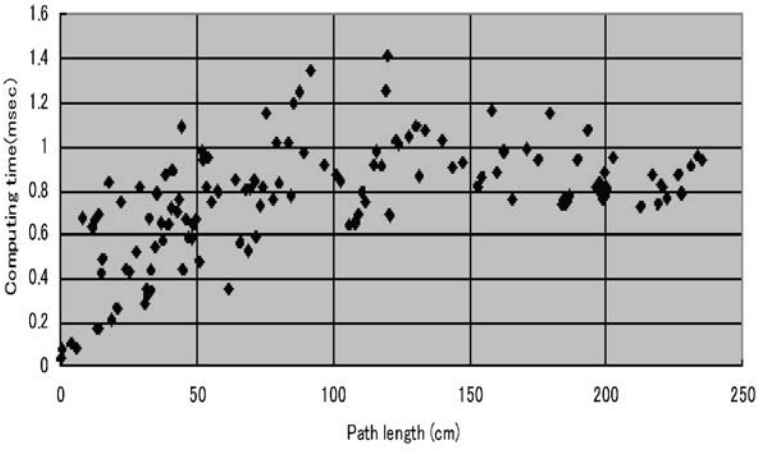
Step2 Get a goal position $p' = (ox, oy)$. This is given by the strategy algorithm.

Step3 Let $\overrightarrow{pp'}$ be a vector directed from the current position to the goal position and let θ be an angle between vectors $\overrightarrow{pp'}$ and $\mathbf{u}$.

Step4 Give a curvature at the time $t + \Delta t$ by $\kappa_{new} = \theta \times n_a / R_A$, where R_A is a constant and n_a is a variable depending on subgoal(s). (This equation generates a path which has a large curvature at first and a small as approaching to the goal. See fig. 9.)

Step5 Putting $dr = 1/\kappa - 1/\kappa_{new}$, calculate a velocity variation of robot $|dv| = |S/dr|$, where S is a constant. Let $v_m(\kappa)$ be a maximum robot speed when a curvature κ is given. Give new velocity by $v_{new} = v + dv$ if $v_{new} < v_m(\kappa_{new})$, otherwise by $v_{new} = v - dv$. Then, calculate a new position at the time $t + \Delta t$.

[2] Image processing speed determines the Δt. In our system, $\Delta t = 33msec$.



Each plot shows generated path length and its runtime for a robot (Processor is Pentium III 800 MHz. Operating system is Linux)

Fig. 10. Runtime for randomly selected generated-path in a real game

Step6 Calculate repeatedly the steps from *step1* to *step5* and check whether the path reaches the given goal or not. (Fig. 9 shows the result of this calculation in a real game.) If the path reaches the goal, it is OK. If not (if over M times repeated), recalculate these steps by changing the constant R_A until the path reaches the goal. This computation gives the robot velocity of next Δt time period.

3.3 Runtime Evaluation

Figure 10 shows the performance of our path generation system. In the figure, each plot shows generated path length and its runtime for a robot in a real game. The path generator runs on the Pentium III 800 MHz processor under the Linux operation system.

The runtime for typical path with length of 100 cm takes about 1 msec from the figure. This is short enough to make the image processing, strategy and path generation computation within 1/30 sec on the host processor.

The runtime disperses over the vertical axis, since the path generation time, for the same length paths, depends on the curvature of the geometrical path, i.e. the computation time of the path with large curvature takes long time and the small takes short time.

4 Concluding Remarks

In this paper, we proposed a fast and robust image processing method which is coped with the variance of color parameters in the field and a new labeling

algorithm, "diagonal alternate spread labeling algorithm". It was clarified experimentally that these method and algorithm are highly effective and robust to extract moving objects up to 240 cm/sec without any condition that it is an interlaced format image or not.

We also show the path generation algorithm in which the robot approaches the goal by changing its direction convergently.

These two algorithms realized the fast and robust robot system. Although our system installed these algorithms works well in real time, there are some remaining issues. As the robustness of the image processing under the spatial variance of lighting is improved, it still remains the problem for improving the robustness under the time variance of lighting.

Acknowledgements

This paper was partially supported by Grant-in-Aid for General Scientific Research (C) of Ministry of Education, Culture, Sports, Science and Technology, and the Tatematsu Foundation and AI Research Promotion Foundation.

References

1. M. Veloso et al eds. "Lecture Notes in Artificial Intelligence 1856, RoboCup-99: Robot Soccer World Cup III", Springer, 2000.
2. P. Stone et al eds. "Lecture Note in Artificial Intelligence 2019, RoboCup 2000: Robot Soccer World Cup IV", Springer, 2001.
3. J. Bruce, T. Balch, M. Veloso "Fast and Inexpensive Color Image Segmentation for Interactive Robots", Proc. 2000 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, pp. 2061 - 2066, 2000.
4. Y.Nagasaka, K.Murakami, T.Naruse, T. Takahashi and Y.Mori: "Potential Field Approach to Short Team Action Planning in RoboCup F180 League", in P. Stone et al eds., Lecture Note in Artificial Intelligence 2019, RoboCup 2000: Robot Soccer World Cup IV, pp.345-350, Springer, 2001.
5. R. D'Andrea and J. Lee "Cornell Big Red Small-Size-League Winner", AI magazine, Vol. 21, No. 3, pp. 41 - 44, 2000.
6. M. Veloso, M. Bowling, S. Achim "The CMUnited-99 Small Robot Team", Team Descriptions Small and Middle Leagues, Linköping University Electronic Press, 1999.
7. A.Rosenfeld and A.C.Kak, "Digital Picture Processing", 2nd ed., Vol.2, Chap.10, Academic Press, 1982.
8. R.M.Haralick, "Some neighborhood operations", in Real Time/Parallel Computing Image Analysis, M.Onoe, K.Preston,Jr., and A.Rosenfeld Eds., Plenum Press, New York, pp.11-35, 1981.
9. Y.Shirai, "Labeling connected regions", in Three-Dimensional Computer Vision, Springer-Verlag, pp.86-89, 1987.
10. K.Suzuki et al., "Fast Connected-Component Labeling Based on Sequential Local Operations in the Course of Forward Raster Scan Followed by Backward Raster Scan", Proc. of ICPR2000, Vol.2, pp.434-437, Barcelona(Spain), 2000.