

Bibliography

- [1] H. Kitano, Y. Kuyinoshi, I. Noda, M. Asada, H. Matsubara, and E. Osawa. RoboCup: A challenge problem for AI. *AI Magazine*, 18(1), pages 73-85, 1997.
- [2] M. Fujita, M. Veloso, W. Uther, M. Asada, H. Kitano, V. Hugel, P. Bonnin, J.-C. Bouramoue, and P. Blazevic. Vision, strategy, and Localization using the Sony legged robots at RoboCup-98. *AI Magazine*, 1999.
- [3] Maayan Roth, Douglas Vail, and Manuela Veloso. A real-time world model for multi-robot teams with high-latency communication. In *Proceedings of the 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2494–2499, Las Vegas, NV, October 2004.
- [4] J. Bruce, T. Balch, and M. Veloso, “Fast and inexpensive color image segmentation for interactive robots,” in *Proc. of IROS-2000*, Japan, October 2000.
- [5] CMVision web page <http://www.cs.cmu.edu/~jbruce/cmvision>
- [6] Lenser, S., and Veloso, M. Sensor resetting localization for poorly modeled mobile robots, *Proceedings of ICRA-2000*, San Francisco, April 2000.
- [7] Lenser, S.; Bruce, J.; and Veloso, M. 2001a. CMPack’00. In Stone, P.; Balch, T.; and Kraetzschmar, G., eds., *RoboCup-2000: Robot Soccer World Cup IV*. Berlin: Springer Verlag. 623–626.
- [8] Lenser, S., Bruce, J. and Veloso, M. (2001). CMPack: A complete software system for autonomous legged soccer robots, *Proceedings of the Fifth International Conference on Autonomous Agents*. Best Paper Award in the Software Prototypes Track, Honorary Mention.
- [9] Juan Fasola. Real-Time Visual Robot Detection and Modeling with Situational Awareness. Senior Research Thesis, Carnegie Mellon University, Pittsburgh, PA, May 2006.
- [10] UTS Unleashed! website <http://www.unleashed.it.uts.edu.au/TeamReports.html>
- [11] Quinlan, M.J. et al.: The 2006 NUbots Team Report. <http://robots.newcastle.edu.au/publications/NUbotFinalReport2006.pdf> (2007)
- [12] rUNSWift website <http://www.cse.unsw.edu.au/~robocup>



Figure 7.1: CMDash'07 team with Four-Legged League 3rd place trophy at RoboCup 2007. Pictured are all team members (from left to right): Somchaya Liemhetcharat, Mike Phillips, Juan Fasola, Greg Delmar, and Manuela Veloso.

shoot on goal behavior, by dodging around visually detected robots only when they are obstructing the view of the teammate according to localization estimates.

7.2.2 Pass Receiving Behavior

The pass to teammate behavior experiments included a teammate robot placed in the crouched position optimal for receiving a pass. However, the teammate robot was static and always in the crouched position, and did not react to successful passes that left the ball at its chest. Clearly, a useful continuation of the research performed would be the creation of an effective pass receiving behavior. The pass receiving behavior would need to have the receiving robot actively coordinate with the passing robot in preparation for a pass, so that it can decide when to appropriately position itself in the crouched position. The receiving robot could also use visual feedback of the ball position to possibly move laterally before crouching, or to not crouch at all in the case of errant passes. In addition, the receiver could actively try and position itself in line of sight with the passing robot, out of the way of opponent robots, to facilitate the passing procedure and the duty of its teammate.

7.3 Competition Results

Our AIBO robot soccer team, CMDash'07, achieved 3rd place at the international RoboCup competition held in Atlanta, Georgia in July 2007. The Four-Legged League competition featured 24 teams from various universities from all around the world. Our team played a total of 7 official matches and was equipped with all the algorithms presented in this thesis, though the passing behavior was not used in competition due to its preliminary nature as stated in the previous section.

The shoot on goal behavior was directly responsible for all but 2 of the 19 goals scored by our robot soccer team. In addition to contributing to the goals scored in regulation time, the shoot on goal behavior performed very well as our team's penalty kicker behavior, as evidenced by its performance in the 3rd place final match. The 3rd place final was a very contested match, in which our team competed against the WrightEagle team from China. The match ended in a 3-3 tie, which forced a penalty shoot-out. The rules and setup of the penalty shoot-out is similar to the shooting experiment in section 5.3, with the main difference being that the goalie robot is allowed to move. The penalty shoot-out was just as contested as the actual game, yet in the end our team's shoot on goal behavior prevailed with a final match score of 6-5.

Our robot soccer team's 3rd place achievement represents our first Four-Legged League international trophy since 2002. Although the advances made in the attacker from the algorithms presented in this thesis undoubtedly helped our team score goals and win games, it was the combined effort of the individual roles of the entire team, including the supporter, defender, and goalie in addition to the attacker that led our team to victory. A picture of the CMDash'07 team members, which all contributed to the success of our robot soccer team, is shown in Figure 7.1.

behaviors. The results show that the visual input-based behaviors, which make use of the novel visual detection algorithms presented and sensor-added behavior pipeline, are able to perform their tasks effectively and are indeed capable of succeeding where previous localization-based pipelined behaviors have failed, namely in the presence of moderate to extreme errors in the robot's localization estimate.

It should be noted that the purpose of the sensor-added behavior pipeline is not to be a total replacement for the localization-based behavior pipeline, or to show the overall ineffectiveness of the localization-based pipeline, but rather to provide an alternative behavior model for certain critical behaviors that have little room for error. These critical behaviors have the characteristic of failure cases that are extremely undesirable, such as kicking out of bounds and losing possession and a scoring chance in the case of the critical shoot on goal behavior. Therefore, for most behaviors the traditional localization-based pipeline is completely appropriate, however for those that require more assurance in their effectiveness and a high degree of accuracy in order avoid the unwanted failure cases, the sensor-added behavior pipeline is presented as a solution.

7.2 Future Work

There are many ways in which the work presented in this thesis can be extended. The visual detection of teammate and opponent robots, for example, can be used to react to robot positions during a designed play or when coordinating with a teammate to decide who should go for the ball. Overall, the visual detection of robots on the field, made more reliable and useful with the addition of the detection of far robots, allows for the creation of an entirely new set of behaviors previously unseen in the RoboCup Four-Legged League. However, the most relevant continuation of the presented work would be the completion of the passing behavior and the creation of a pass receiving behavior.

7.2.1 Passing Behavior

As stated in chapter 6, the pass to teammate behavior described in this thesis is only a preliminary version, and thus a final version must contain some components that have not yet been implemented. These components include shortening the time to locate the teammate before passing. As described, the preliminary version simply turns at a constant speed while running the visual robot tracking algorithm in order to locate the target teammate position, and also does not take into account the 3-second ball holding rule. In an effort to reduce the time required to locate the teammate visually, the robot could be made to turn at full speed until it reaches a certain angle threshold of the estimated teammate position where it would then turn at the constant speed required for tracking. The robot could also use the communicated teammate's position relative to the field landmarks and goals in order to decide when to speed up or slow down as these landmarks become visible while turning.

A final implementation of the passing behavior must also include reasoning about obstructing opponent robots. As is the case when shooting towards the goal, often times there are robots that obstruct the direct view of the target object, in this case the teammate robot. The passing algorithm could be made to handle these cases, again similar to the

Chapter 7

Conclusion

In this chapter we summarize the main contributions of the thesis, discuss future work aimed at expanding and utilizing these contributions, and provide anecdotal evidence of the effectiveness of the algorithms presented, focusing on the ‘shoot on goal’ behavior, from our AIBO robot soccer team’s experience at the recent international RoboCup competition.

7.1 Contributions

The main contributions of this thesis are:

Novel visual detection algorithms of field objects. We present novel algorithms for the visual detection of teammate and opponent robots on the playing field, and of the two uniquely colored goals.

The robot detection algorithm expands on a previous implementation by increasing the detection range to include robots that are up to 6 meters away from the observing robot, while reducing the number of false positives for robot detections within 1.2 meters of the robot. The improved performance of the robot detector allows for increased situational awareness in the robot soccer behaviors.

The goal detection algorithm solves the visual object detection problem introduced by the Four-Legged League’s modification of the structure of the team goals. The goal detection algorithm is able to effectively detect the position of the goal posts, which the robot uses to improve its localization estimates, as well as the goal region itself, which the robot uses while targeting the goal when shooting.

Sensor-added behavior pipeline for improved robot soccer behaviors. We describe a modification to the traditional localization-based behavior pipeline, which introduces a connection from the vision module directly into the behavior module. This sensor-added behavior pipeline allows for the improvement of many robot soccer behaviors, of which we discuss two in detail:

- The shoot on goal behavior
- The pass to teammate behavior

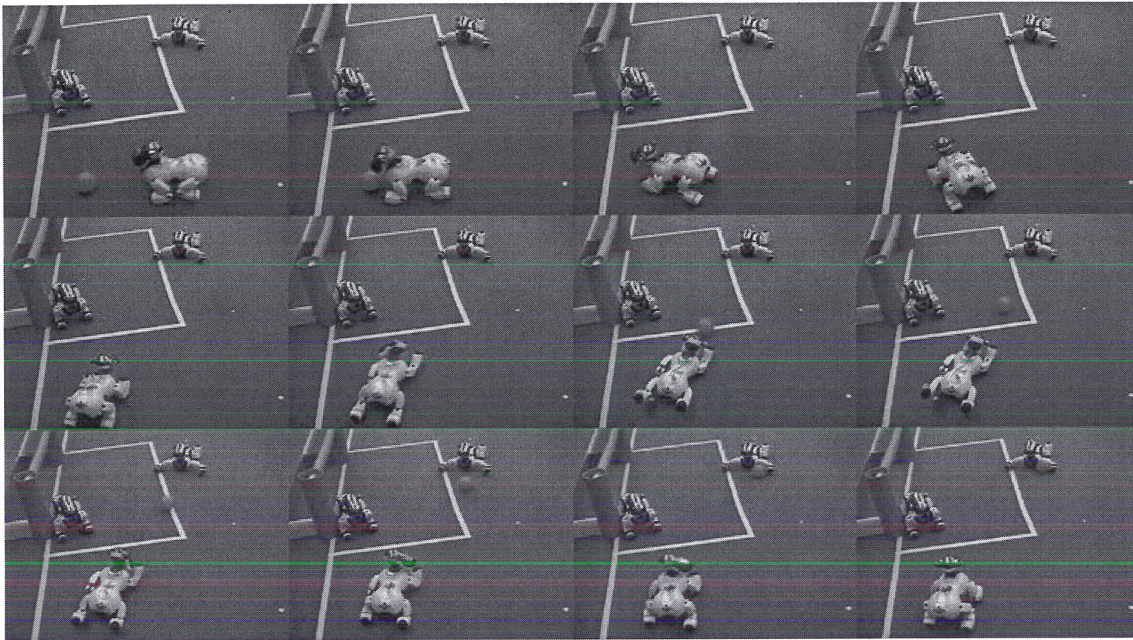
We provide experimental results testing the effectiveness of the visual input-based behavior methods and their traditional localization-based implementations for both

which didn't fail even once at completing a pass to the teammate robot. Like the localization-based behavior, the visual passing behavior sometimes chose the wrong initial turn direction, however it was always able to recover, as it would keep turning and kick the ball forward only after having gained visual confirmation of the awaiting red teammate robot in front of it.

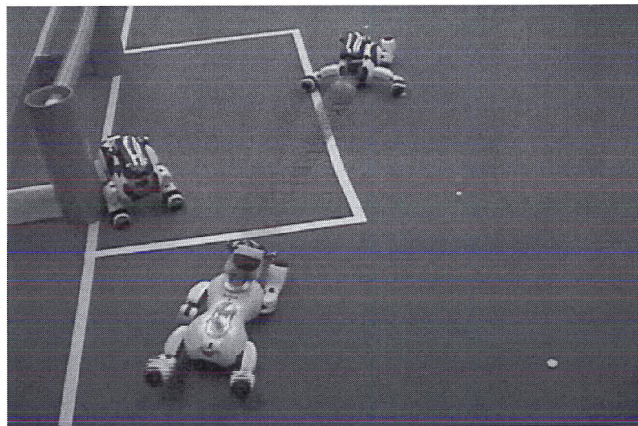
The positional localization error case affected the localization-based behavior, but not as much as the angular error case. In fact, both behaviors performed in line with the expected passing percentages we calculated from the results of the experiment, as the localization-based behavior successfully passed to the teammate robot around 50% of the time, and the visual passing behavior hit on target around 90% of the time, missing on only one occasion.

In summation, the vision-based passing behavior performed extremely well in the experiment and was able to overcome the localization errors imposed by the human operator with ease, which was certainly not the case for the traditional localization-based passing behavior. The vision-based behavior also never once mistook the goalie robot for its red teammate or its teammate for a blue robot, thus validating the visual tracking algorithm used in the behavior. The visual passing behavior in its preliminary form shows great promise for use during actual game play, as future implementations need only worry about how to deal with potential failure cases such as robots occluding a direct view of the teammate, and has the potential to improve the overall team game play in a new and exciting way by opening the doors to fascinating research opportunities previously unexplored by the RoboCup Four-Legged League.

off to one side or the other, causing the ball to barely miss hitting the crouching teammate robot.



(a)



(b)

Figure 6.5: (a) Video capture frames of passing to teammate using visual tracking method. (b) Video capture motion sequence of passing using visual tracking method.

The angular localization error case was the experimental case that caused the biggest disparity between the two passing behaviors. Turning for 1 second while being picked up caused significant trouble for the robot running the localization-based behavior, as it was only able to pass to the teammate robot 7 times out of the 20 trials. Many times the robot would turn itself in the wrong initial direction after grabbing the ball and then kick the ball out of bounds completely off-target in the opposite direction of the teammate robot. The same cannot be said for the vision-based passing behavior,

6.5 Results

The results of the experiment testing the effectiveness of both the localization-based and vision-based passing behaviors, working under three different robot localization scenarios in a traditionally difficult attacking situation, are summarized in Table 6.1.

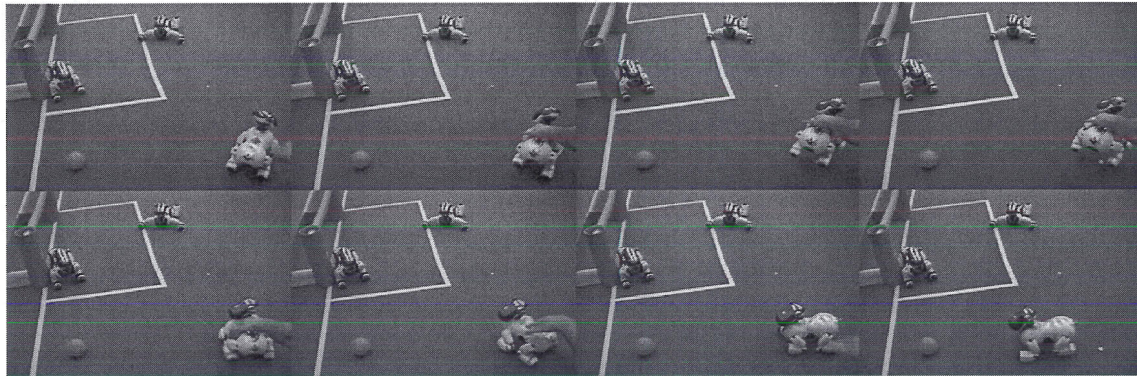
Experiment Cases	Localization-Based Pass To Teammate (# Hits / # Runs)	Vision-Based Pass To Teammate (# Hits / # Runs)
Optimal localization (No disturbance)	14 / 20	16 / 20
Angular localization error	7 / 20	20 / 20
Positional localization error	11 / 20	19 / 20
Passing Percentage (# Success / # Total Runs)	32 / 60 = 53.3%	55 / 60 = 91.6%

Table 6.1: Results of experiment testing passing effectiveness for localization-based and vision-based pass to teammate behaviors.

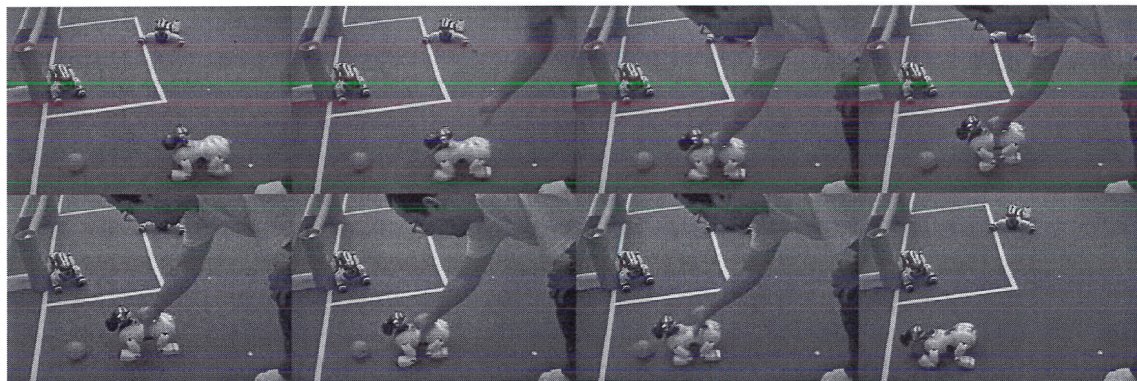
The results show that the vision-based passing behavior outperformed the localization-based passing behavior in all three cases of the experiment, yielding a total pass completion percentage of 91.6% compared to that of 53.3% for the localization-base pass to teammate behavior. The passing percentage for the behaviors is important because it provides a fair estimate as to how well the behavior will perform under normal game conditions, where at any point the robot may be well localized, or mis-localized with respect to its heading, position on the field, or both. Thus, we are able to deduce from the results that, on average, the localization-based passing behavior is expected to have the robot successfully complete a pass to its teammate around 50% of the time, and the vision-based passing behavior is expected to succeed at passing about 90% of the time, which is a significantly better success rate without a doubt. Two examples of successful pass attempts by the vision-based passing behavior can be seen in video capture images shown in Figure 6.5.

In the optimal case where no localization error was manually imposed on the robot, the two behaviors had comparable performance, which was to be expected. The localization-based passing behavior relies on accurate localization estimates, so when the robot's localization estimates are more or less correct, the passing should work fairly well. This conjecture is confirmed by the results that show that the localization-based passing behavior was able to successfully pass the ball to the teammate robot 14 times out of the 20 trials that were run. The vision-based passing behavior also did reasonably well, failing to hit its teammate only on 4 occasions, and when the passes missed they didn't miss by much at all. In fact, the vision-based behavior aligned itself with its teammate robot fairly accurately on all the runs in all three experimental cases, and the missed passes were ones where the robot was lined up pretty good, but the shot veered

approximately 1 second as the robot walks forward towards the ball after it has localized itself on the field at the start of a run. Figure 6.4b shows video capture images of the human operator picking up the robot to introduce forward positional localization error.



(a)



(b)

Figure 6.4: (a) Human operator imposing angular localization error by holding up robot for approximately 1 second while it is turning in place at the beginning of a trial run.

(b) Human operator imposing positional localization error by holding up robot for approximately 1 second as it walks forward towards the ball during a trial run.

Each experimental case was run 20 times for both the localization-based and vision-based passing behaviors, and all three cases followed the same procedure: the robot would start by spending six seconds localizing itself on the field with the aid of the field markers and the far goal, then it would turn to locate the ball before attempting to grab it and make a pass. Each run ended once the robot kicked the ball forward in a pass attempt.

A pass attempt was considered successful if the ball managed to hit any portion of the crouching teammate robot, and unsuccessful otherwise. Passes that hit the teammate robot just slightly on the front paws and rolled off to the side were still considered successful because the passing and receiving behaviors are still preliminary, and it is not unreasonable to assume that a more advanced receiving behavior could have had the teammate robot adjust itself to the path of the ball in order to control it to a stop under the same situation.

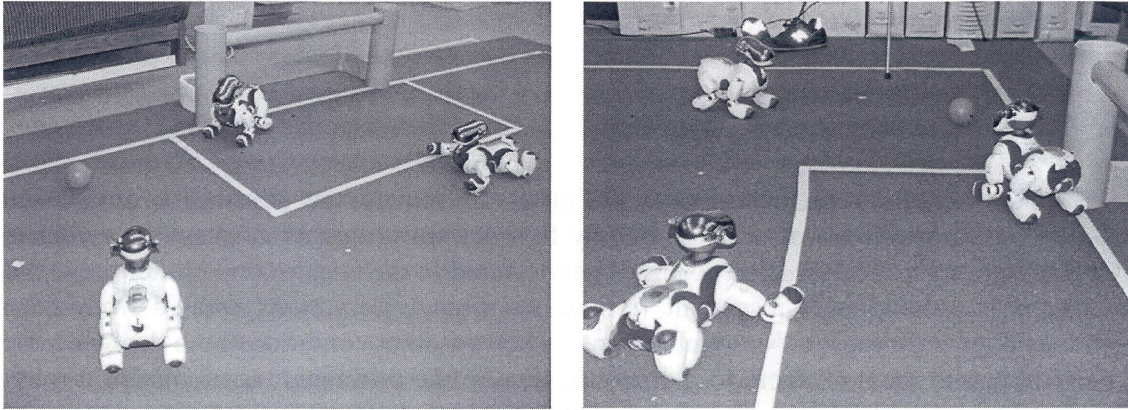


Figure 6.3: Passing experimental setup from two different viewpoints.

The experiment was run using both the localization-based behavior and the proposed vision-based passing behavior for direct comparison purposes. The experiment was comprised of three separate cases to test the effectiveness of both behaviors under varying game-like conditions. The three cases are:

- 1) Passing under optimal localization conditions
- 2) Passing under angular localization error
- 3) Passing under positional localization error

The first case tests the robot's ability to pass under optimal conditions in which the robot is allowed sufficient time to localize itself on the field before beginning each run and is not disturbed during execution of the runs. The second and third cases both allow the robot to localize itself the same way as in the first case, however in both cases localization error is introduced to the robot by the human operator to simulate the frequent game occurrence of robots pushing against each other while walking towards the ball. Pushing between robots of opposing teams is an unintentional side effect of both robots wanting to take the shortest path towards the ball, and as a result the odometric model of the robot ceases to reflect the actual physical movement of the robot, which in turn leads to errors in the robot's localization estimate. In addition, when the robot is in the corner of the field facing outwards, as is the case in the experiment, it is in the worst position to correctly localize itself based on visual sensor readings because it cannot see any of the field landmarks and it can only see the goal from the side.

The second experimental case tests the ability of the robot to pass to its teammate under angular localization error. The angular localization error is introduced by the human operator picking up the robot as it attempts to turn around after localizing at the beginning of a run. The robot is held in the air for approximately 1 second, which is a reasonable simulation of the robot getting briefly caught up on an opponent robot as it tries to turn around. Figure 6.4a shows video capture images of the human operator imposing angular localization error at the beginning of an experimental run.

The third experimental case tests the robot's ability to pass to its teammate under positional localization error, in the forward direction. Similar to the previous case, the positional error is introduced by having the human operator pick up the robot for

Team color: *Blue*

- History size > 10
- Blue count >= 10
- Red count * 3 < Blue count

The minimum history size condition is to reduce the possibility of passing towards false positive robot observations by first waiting to gather convincing evidence of a robot presence. When false positives get detected in the background clutter they tend to disappear quickly on subsequent frames, so robot observations created from false positives expire after a short amount of time. The two other conditions both deal with the color count, and they are there to gain confidence in the estimated team color of a robot observation to avoid passing to a robot of the opposite team. The minimum count and relative count conditions were based on empirical evidence from running the tracking algorithm while the robot turned at 60 degrees/second. The conditions were modeled primarily over the observation that red robots are much more likely to get blue color estimates than the reverse, especially for robots over 2 meters away, due to segmentation noise and the fact that the uniform area in the image for far robots is extremely small.

The conditions have constant values that are appropriate for a robot turning speed of 60 degrees/second. To put the constants into perspective, a robot turning at 60 degrees/second can expect a robot observation tracked across the whole image to have a maximum history size (i.e. maximum number of recorded observations) of about 20 to 30 frames. So, on average, the minimum required history size, which at this speed is 10 frames, is just less than half the maximum history size. If the robot turning speed changes, the constants should be scaled inverse proportionally. For example, if the turning speed is reduced by a factor of 2 to 30 degrees/second, the constants should be scaled up by a factor of 2.

6.4 Pass Behavior Experiments

To test the effectiveness of the passing behavior, an experiment was performed that placed the robot in one of the most difficult attacking situations on the field: attacking from the opponent corner. The corner case is difficult because the shooting angle is almost non-existent and with a goalie at the goal post, a shot has no chance of scoring. If the attacking robot instead chooses to dribble inwards towards the goal, the goalie robot will most likely clear the ball out of the area. This is the perfect scenario for passing. By passing to a teammate robot that is waiting in front of the wide-open goal, the situation changes from being very difficult for scoring to relatively easy for scoring. This situation is just one of many game situations that can be greatly benefited from the development and use of explicit passing between teammates.

The experimental setup placed the ball 60cm to the left of the goal post and 18cm from the back line, with a static goalie robot placed at the closest goal post facing the ball. The robot was placed approximately 80cm behind the ball towards the midfield, and the teammate robot was placed near the center of the field just outside the goal box and facing the ball. The teammate robot was placed in the crouched position, which is optimal for receiving a pass. Figure 6.3 shows pictures of the experimental setup.

The jumping effect shown when the blue robot was not detected for one frame, and then reappeared beyond the observation matching window on the next frame, occurs infrequently. In most cases, the visual tracking algorithm is able to tolerate three or more consecutive unmatched frames before the detection results jump beyond the matching window. However, the error highlights the preliminary nature of the tracking and passing algorithms, and possibly could be corrected by taking into account the width of the robot bounding boxes during match testing, instead of just the center x-pixel position.

6.3 Passing Algorithm

The preliminary algorithm of the passing behavior, which assumes the robot has control of the ball under its chin initially, contains the following three steps that it executes one after the other:

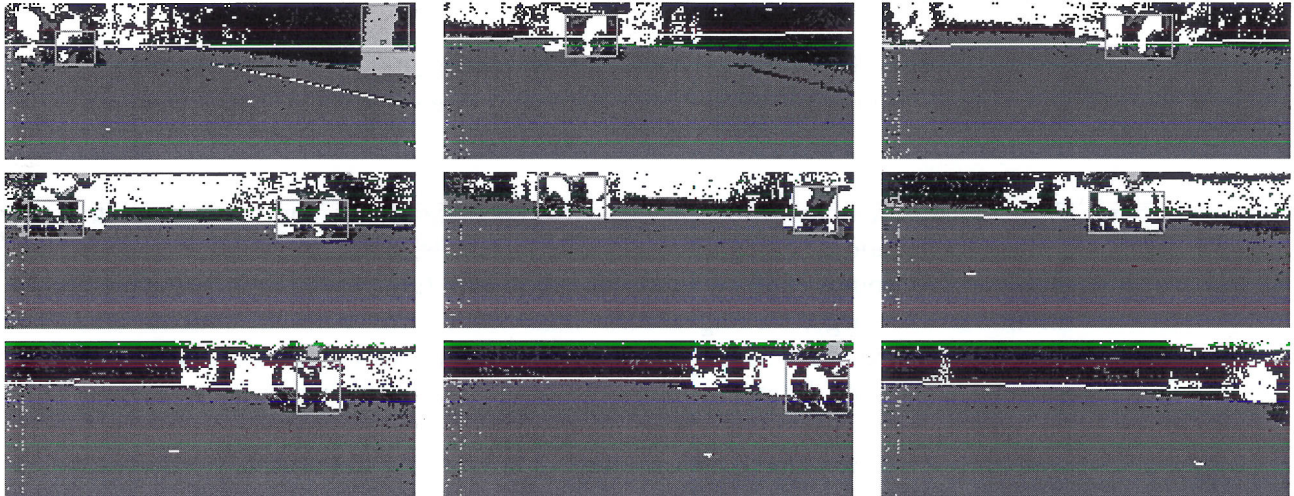
- Turn towards teammate location based on localization: The robot uses its current localization estimate and the location of its teammate, communicated to the robot through wireless communication, and turns in the direction which minimizes the target turning angle. The robot turns with the ball in this direction at a constant speed of 60 degrees/second while running the visual tracking algorithm described in the previous section to locate its teammate through vision. The robot does not change its direction of turning while searching for its teammate, independent of what the communicated teammate position suggests.
- Adjust to visual detection of teammate: Once the robot confirms the position of its teammate with the help of the visual tracking algorithm, it turns to align itself with the detected teammate position. The robot turns at a constant speed of 30 degrees/second and continuously chooses its turn direction for alignment. The alignment objective is to get the visual teammate x-pixel position within a 20 pixel width window around the horizontal center of the image.
- Kick the ball forward to the teammate: After the robot aligns itself such that the visual teammate position is in the center of the image, it kicks the ball forward, which results in a pass towards the teammate.

As described in the first step, the robot turns in place to gain visual confirmation of the teammate position before adjusting and kicking. This visual confirmation occurs when all of the conditions for the robot team color are true with a robot observation in the tracking list; the conditions are:

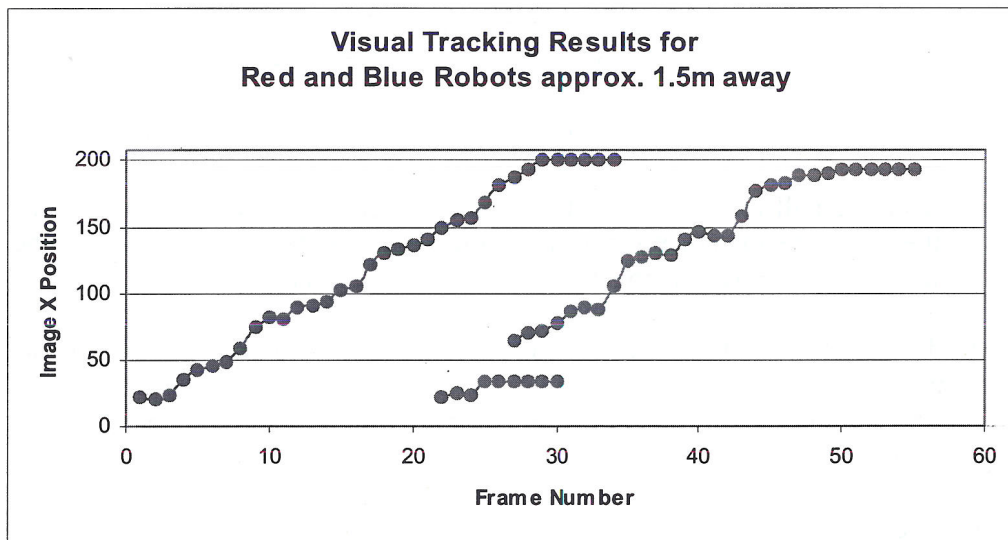
Team color: *Red*

- History size > 10
- Red count >= 6
- Red count * 3 >= Blue count

in vision and on the subsequent frame the detection result had jumped beyond the matching window of the previous location of the robot observation, and therefore the robot detection was not matched to the robot observation a new robot observation was created. This new observation was able to successfully track the blue robot as it translated across the image, while the previous blue robot observation timed out. You can see from the results that all robot observations time out and are removed from the tracking list when they no longer match with robot detection results for 5 consecutive frames, which is the maximum unmatched count from the algorithm description above.



(a)



(b)

Figure 6.2: (a) Robot detection results during visual tracking of red and blue static robots while observing robot rotates at 1.0 radians/sec. (b) Visual tracking results showing the x pixel position in the image of the robot observations tracked over a period of 60 frames (approx. 2 seconds).

- The robot detection results of the current vision frame are matched with robot observations already being tracked in the list. The algorithm iterates through all observations in the list, starting with the one of greatest history size and ending with that of the least, and each iteration attempts to find a match for the robot observation by iterating through the robot bounding box results from vision. A match is found if the horizontal distance between the robot observation's current pixel location and the center of the robot bounding box is less than the matching window size, which is set to 30 pixels. If a match is found for a robot observation in the list, the following operations are performed before continuing along the list:
 - The robot observation's history variables are updated, including the current pixel location, incrementing the red or blue count depending on the team designation of the robot detection, setting the unmatched count to 0, and incrementing the history size.
 - The matching robot bounding box is removed from consideration to avoid matching it to multiple robot observations in the list.
- All robot detection results that could not be matched to robot observations in the list represent new robot observations to track and therefore are added to the tracking list. The history variables of the new robot observation are set to their respective initial values, where the unmatched count is 0 and history size is 1.

6.2.2 Sample Tracking Results

To test the visual tracking algorithm we placed two static robots, one red and the other blue, roughly 1 meter apart from each other and both approximately 1.5 meters away from an observing robot that was running the algorithm. The observing robot was positioned in the exact same way it would be before making a pass, with its head in a fixed position and the ball securely grabbed under its chin. The behavior of the observing robot was to simply rotate in place at a speed of 60 degrees per second.

The purpose of the test was to evaluate the tracking algorithm in a situation where the observing robot would see the red robot first, and then as it continued turning the red robot would translate horizontally across the image and soon the blue robot would enter the field of view so that both robots were visible in the image, and finally as it turned even more the red robot would leave its view and the blue robot would be the only one visible. Figure 6.2 shows the visual tracking results along with images taken from the observing robot's camera showing the robot detection results as it was turning during the test. The images and results are taken from a 2 second time period during the test, which was enough time for the static robots to both enter and leave the field of view of the observing robot while it was turning. The tracking results show the x-pixel location of all robot observations in the list at each frame, with the different pixel locations of a single robot observation connected by a line.

As you can see from the results, the tracking algorithm successfully tracked the two robots in the image across multiple frames as it was rotating in place. However, you will also notice that the tracker mistakenly created two robot observations in the list for the blue robot. This is because there was a frame where the blue robot was not detected

6.2.1 Tracking Algorithm

The passing behavior is run only after the robot has gained possession of the ball underneath its chin, and in order to maintain possession of the ball the robot does not move its head. So from the initial grabbing of the ball, to turning in place to align with its teammate up until executing the passing forward kick, the head remains in a fixed position over the ball. This provides a key characteristic to the tracking algorithm: as the robot rotates to search for its teammate to pass to, the robot observations in the image translate horizontally and not vertically. By using this assumption the visual tracking procedure is greatly simplified. It should be noted, however, that this property and the decision when to start and stop the visual tracking is dependent on the behavior, therefore all of the visual tracking is done at the behavior level using the robot detection results provided by the vision system.

The visual tracking algorithm maintains a list of all the robot observations it is currently tracking, and during each tracking frame the history of all observations in the list is updated. The history variables associated with each robot observation are the following:

- *Current pixel location*: The vision system returns a bounding box for each robot detected in the vision frame. The center of the bounding box is used as the current pixel location in the image for the robot observation.
- *Red count*: The number of frames in which the robot observation was designated as being on the red team.
- *Blue count*: The number of frames in which the robot observation was designated as being on the blue team.
- *Unmatched count*: The number of consecutive frames in which the robot observation was not matched with a robot detection box from vision. This counter gets reset to zero on every frame where a match occurs.
- *History size*: The number of frames throughout the lifetime of the robot observation in the tracking list where the robot observation was successfully matched with a robot detection box from vision.

The tracking algorithm begins with the tracking list of robot observations initially empty, and for every frame that the tracker runs it performs the following steps to update the list:

- The unmatched count is incremented by 1 for all robot observations in the list. If a robot observation has an unmatched count greater than or equal to the maximum allowed, it is removed from the list. The maximum value for the unmatched count is set to 5.
- The list is sorted by history size, such that the robot observation with the highest history size is at the front.
- The detected robot bounding boxes for the current vision frame are retrieved. If there are no detection results, the algorithm exits and no further updates to the list are performed for the current frame.

vision-based solution represents another example of a behavior that is capable of overcoming traditional setbacks due to localization error by introducing a connection from the vision system directly into the behavior for improved decision-making and performance. In the case of the vision-based passing behavior, the primary visual input is the robot detection results, and more specifically, the explicit pixel locations of robots found within the camera image. The proposed behavior input model, as well as the previous localization-based model, are illustrated in Figure 6.1.

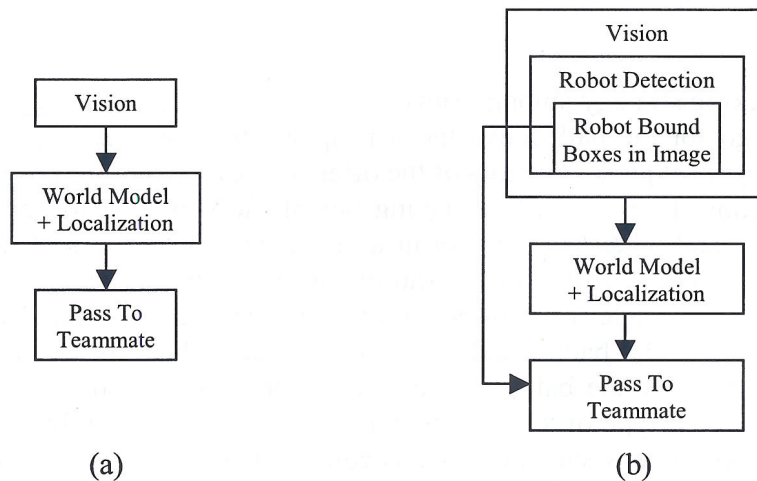


Figure 6.1: (a) The previous pass behavior model, which uses only localization estimates when passing to a teammate. (b) Proposed pass behavior model that uses visual input to explicitly target visually detected teammate robots.

6.2 Visual Robot Tracking

Similar to the shoot on goal behavior, the pass behavior must obtain confirmation of its target through vision before executing a kick. However, obtaining confirmation of a teammate location through vision is a more difficult problem than visually confirming the location of an opponent goal. This is because there are only two goals on the playing field, both of which are color coded and fairly large, so they are more easily detected and the possibility for false positives is low. In the case of robot vision, there are a total of eight robots on the playing field, they are not very large, especially in the image space for more distant robots, and because of this their teammate designation can be noisy at times, and false positives are more likely to appear due to artifacts in the image. Therefore, in order to reduce the possibility of passing to a robot of the opposite team or passing to a false positive detected in the background clutter, we have decided to track robot observations over a period of time to build confidence in the reported team color and existence of the robot on the field.

In this section we will present the visual robot tracking algorithm used in the passing behavior and provide sample results of the tracking algorithm to showcase its performance.

Chapter 6

Passing To Teammates

The overall attacking strategy among teams in the robot soccer Four-Legged League is to have one robot go for the ball, move the ball up the field, and take a shot once close enough to the opponent goal. The role of the offensive teammate is to position itself such that it is supporting the attack while staying out of the way, and to only try and take possession of the ball when it is clearly in a better position to do so than the primary attacker, at which point the roles switch and the primary attacker becomes the supporter. This is very much a single-robot oriented strategy because the offensive teammate is never explicitly given the ball; it only receives the ball when, by chance, through the course of the game play the ball ends up in a better attacking position for it than the primary attacker. This type of team strategy is notably unlike those found in real soccer where explicit passing between teammates is commonplace and basically a necessity for playing well and scoring goals. Explicit passing between teammates is what's missing from the league, and most teams agree it represents the next big challenge to undertake.

In this chapter we will present a novel, yet preliminary, vision-based passing behavior. First we describe our approach and previous localization-based passing methods, and detail the visual robot tracking system used in the proposed behavior. We will then evaluate and directly compare the localization-based passing behavior to the vision-based behavior by analyzing performance results of a passing experiment aiming to solve one of the most difficult attacking situations in robot soccer.

6.1 Passing Behavior Approach

Our team has experimented with passing in the past and, in fact, we implemented localization-based passing to teammate positions communicated over wireless for the competition in 2004. Other teams have implemented similar methods in the past as well. However, this was when playing field still had walls around it and had four landmarks instead of two. So, the robots were better localized and if the passes misfired, the ball would hit a wall and the teammate robot would still be able to retrieve it. This strategy doesn't work as well in the current state of the league because too often balls would be kicked out of bounds, kicked to players of the opposing team that are in the way, and just generally kicked off target or in the wrong direction.

Our proposed solution to the problems faced by localization-based passing, which are primarily due to the inevitable errors in the robot's position and heading estimates, is to have the passing behavior obtain visual feedback and confirmation of the teammate's position before executing a pass to them, to ensure the best possibility of success. This

	Localization-Based Shoot On Goal (# success / # trials)	Vision-Based Shoot On Goal (# success / # trials)
Experiment 1	1/10	9/10
Experiment 2	0/10	10/10
Experiment 3	0/10	9/10

Table 5.3: Results of experiments testing shoot behavior robustness to traditionally difficult scoring situations.

As confirmed by the results in Table 5.3, the localization-based scoring behavior had significant trouble scoring goals under the conditions presented in the three experiments and, in fact, was only able to score one goal out of all its trials. The missed shots in the first experiment were the result of the attacking robot kicking the ball directly into the chest of the goalie robot, which was located in the middle of the goal, and also the target of the localization-based behavior's aiming. All of the missed shots in the second and third experiments were the result of attacking robot kicking the ball out of bounds, which was to be expected. The vision-based scoring behavior, on the other hand, performed exceedingly well under all three experiments by scoring goals on all but two of its trials. Even when its shots missed, the ball came very close to entering the goal; with one shot bouncing off the side of the goalie robot's foot and the other grazing the goal post before going out of bounds. In the first experiment it is very difficult for the attacking robot to score any goals without explicitly avoiding the static goalie in front because the ball is so close to the goalie, however this avoidance is trivial for the vision-based behavior as it simply dodges the goalie to the side before shooting the ball into the visually detected open goal area.

The results of the experiments show that vision-based shoot on goal behavior is capable of overcoming and is certainly robust to the traditional deficiencies of the localization-based shoot behavior, with the ability to: dodge around obstructing robots to make way for clear shots on goal, use visual feedback of goal observations to guide its positioning towards open areas of the goal to increase the scoring chances of shots taken, and overcome any errors in localization, big or small.

(x,y) position on the field. The experiment was run 10 times with the vision-based behavior and 10 times with the localization-based behavior, again with one shot on goal allowed per run.

5.4.3 Experiment 3

The third experiment measures the ability of the robot to score a goal under high localization error in its body position estimate, i.e. its xy-coordinate field location. As with the second experiment, no goalie is used to improve the attacking robot's chances of scoring under the straining conditions.

The attacking robot is initially placed at the right corner of the goal box and is facing the orange ball, which is placed at the left corner of the goal box. The robot localizes itself at its location before starting each experimental run. The positional localization error is introduced in similar fashion to the second experiment. Once the robot grabs the ball it pauses itself for five seconds, at which point the human operator moves the robot back to its original location and orientation at the right goal box corner facing the opposite corner. The manual replacement causes the robot to operate under a positional localization error of approximately 1.3 meters, while keeping the angular localization estimate correct. As with the two previous experiments, only one shot on goal is allowed per run and the two scoring behaviors are each run 10 times. The experiment is depicted in Figure 5.6 by showing the key components of the trials that were executed.

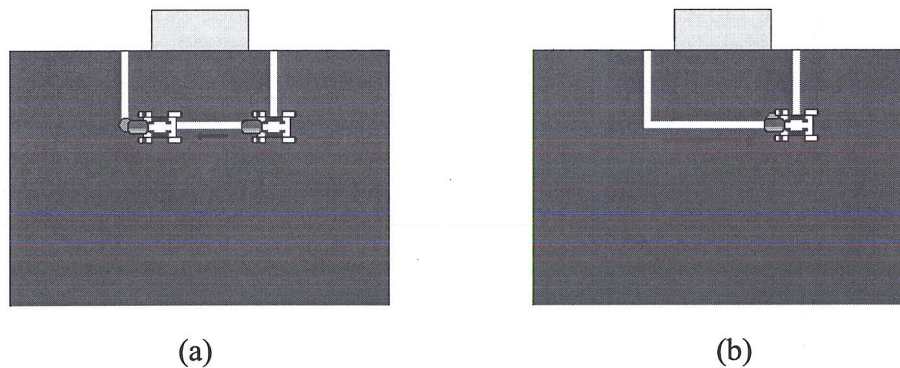


Figure 5.6: Experiment 3 during a run. (a) Attacking blue robot moving from its initial position to just before it grabs the ball. (b) The robot after grabbing the ball and having been manually moved, denoted by the red arrow, to introduce positional localization error.

5.4.4 Experimental Results

The results of the trials of the three experiments on the vision-based scoring behavior and the localization-based scoring behavior are summarized in Table 5.3. Again, the localization-based results are provided only for completeness and confirmation that the situations presented in the experiments are notoriously bad for the localization-based shooting method.

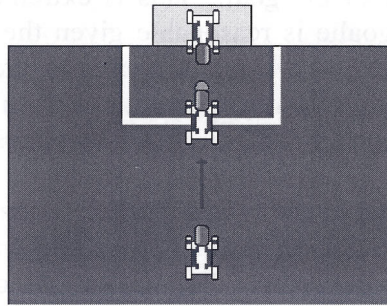


Figure 5.4: Experiment 1 setup. The red robot goalie and the orange ball are both shown in their initial positions. The attacking blue robot is shown in its initial position and just before grabbing the ball.

5.4.2 Experiment 2

The second experiment measures the ability of the robot to score a goal under high angular localization error. No goalie robot is used in order to maximize the probability of success for the attacking robot, as its main opponent is its own incorrect localization estimate. Two illustrative snapshots of the experiment are shown in Figure 5.5.

The attacking robot is initially placed in the center of the field near the midfield line where it localizes itself before starting each run, as in the first experiment, and the ball is initially placed on the front edge of the goal box directly in front of the goal. The angular localization error is introduced by having the robot pause for five seconds after it has gained possession of the ball (with the ball under its chin), during which time the human operator rotates the robot 90 degrees counter-clockwise. After the five second pause terminates the robot continues its scoring behavior as if no pause had been taken, and thus as if no rotation had occurred. This manual rotation causes the robot to obtain a false belief of its body heading, while still maintaining a fairly accurate estimate of its

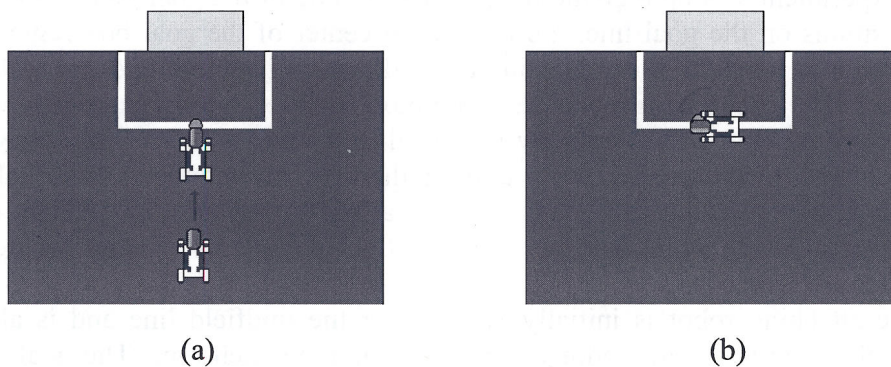


Figure 5.5: Snapshots of experiment 2 during a run. (a) Attacking blue robot moving from its initial position to just before it grabs the ball. (b) The robot after grabbing the ball and having been manually rotated, denoted by the red arrow, to introduce angular localization error.

either scored with a single shot on goal, or required one extra shot after the first shot hit the static goalie in the center of the goal. This is extremely efficient considering that having the first shot hit the goalie is reasonable given the fact that the shot is taken a meter from the goal and there is always some error in the path of the ball when it is kicked. It is especially efficient in comparison with the localization-based method, which hit the goalie more than once on 14 different trials, as opposed to just one for the vision-based behavior.

In summation, this experiment was useful because it provided a means of comparing both shooting behaviors quantitatively, along with demonstrating the overall effectiveness of the shooting behaviors. The vision-based behavior was clearly more effective at scoring goals than the localization-based shooting behavior, as it rarely ever kicked it out of bounds and scored in 95% of the trials. The vision-based shoot on goal behavior was also very efficient at scoring goals, as evidenced by the fact that it hit the goalie robot more than once only on one occasion.

5.4 Robustness Experiments

To test the robustness of the vision-based shoot on goal behavior to errors in localization and to opponent robots occluding shots on goal, three experiments were performed. The first experiment simulates a game situation where scoring is critical, and the other two experiments test the scoring effectiveness of the robot under high localization error. All the experiments were run using both the vision-based and localization-based shoot on goal behaviors separately, however each of the experiments test cases where the localization-based behavior is known to fail, so they were not designed for comparison purposes, but rather only to demonstrate the robustness of the vision-based behavior to traditionally difficult scoring situations. The results of all three experiments are discussed at the end of this section.

5.4.1 Experiment 1

The first experiment sets up a goalie robot in the middle of the goal facing forward with its foot positions on the goal line, the ball in the center of the goal box region, and an attacking robot in line with the goalie and the ball that will attempt to score with a single shot on goal. The experiment replicates a common situation during game play where the ability to score quickly and effectively is crucial: the attacking robot may only get one shot on goal before the goalie clears it from the area, in addition, if the ball is shot inaccurately it could go out of bounds, causing a loss of possession. In both cases the scoring chance is lost, which can be costly in tight competitive games where scoring opportunities are scarce.

The attacking robot is initially placed near the midfield line and is allowed to localize itself using the field landmarks before beginning each run. The goalie is static with its front feet on the goal line and does not move. On each run the attacking robot is allowed only one shot on goal, and a successful run is when the attacking robot scores a goal with its single shot. The experiment was run 10 times with the vision-based behavior and 10 times with the localization-based behavior. The experimental setup is illustrated in Figure 5.4.

	Localization-Based Shoot On Goal	Vision-Based Shoot On Goal
# Goals Scored	24 / 40	38 / 40
# Kicks	109	63
# Shots Hit Goalie	56	18
Scoring Percentage (# Goals / # Runs)	60%	95%

Table 5.1: Results of experiment testing goal-scoring effectiveness/efficiency for localization-based and vision-based shooting behaviors.

The results of the experiment clearly demonstrate the effectiveness of the vision-based shoot on goal behavior and its superiority over the localization-based shoot behavior. The vision-based behavior managed to score an impressive 38 goals from the 40 total runs, only misfiring on two trials and yielding a scoring effectiveness of 95%, whereas the localization-based behavior was only able to achieve a scoring effectiveness of 60% by scoring 24 goals during the 40 runs of the experiment. It is worth noting that the robot has more than enough time to localize itself on the field prior to running each trial, as it spends six seconds looking at field landmarks and the goals, yet the localization-based shooting behavior still managed to kick the ball off-target and out of bounds 40% of the time.

Looking at the criteria towards evaluating the efficiency of the shoot behaviors, we can again see that the vision-based shoot on goal behavior is the clear winner. The localization-based shoot behavior had its shots hit the goalie more than 3 times as much as the vision-based behavior, and the number of kicks taken for the localization-based behavior was approximately 1.7 times the number of kicks taken for the vision-based behavior. Perhaps the efficiency comparison is slightly unfair because the localization-based behavior does not detect and avoid the goalie, so when it shoots towards the center of the goal it always shoots towards the goalie position. Nevertheless, the experiment serves to demonstrate the efficiency of the vision-based shoot behavior on its own. Table 5.2 shows a more detailed analysis of the shots that hit the static goalie.

Number of times the goalie was hit by a shot	Localization-Based Shoot On Goal (# of trials)	Vision-Based Shoot On Goal (# of trials)
0	22	23
1	4	16
2	5	1
More than 2	9	0

Table 5.2: Statistics of shots that hit the static goalie during the scoring experiment

The statistics of the shots that hit the goalie detailed in Table 5.2 suggest the great level of scoring efficiency of the vision-based behavior. The results show that of the 17 trials where the vision-based method hit the goalie, the robot hit the goalie more than once only on one occasion. So on all 40 trials, except one, the vision-based behavior

5.3 Shoot Behavior Comparison Experiment

To test and compare the effectiveness of both the localization-based and vision-based shoot on goal behaviors, an experiment with setup similar to a penalty kick was performed. The setup included one static goalie robot positioned in front of the goal with its right front and rear feet just in front and touching to the goal line. The static goalie was placed sideways, so as to maximize the coverage of its body over the goal opening. The ball was placed just outside the goal box area, 90cm from the center of the goal and in the middle of the field. The attacking robot, which was to run the shooting behavior, was placed 150cm from the center of the goal, in the middle of the field, and facing the opposite goal. Figure 5.3 shows a picture of the experimental setup.

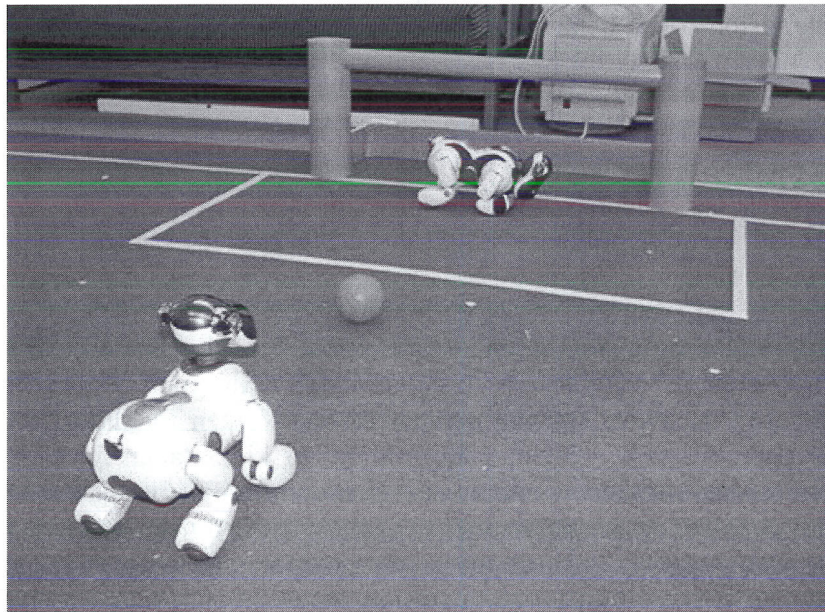


Figure 5.3: The setup for each run of the penalty kick-type experiment testing both shoot on goal behaviors

Each run of the experiment proceeded as follows: the robot would start by spending six seconds localizing itself on the field with the aid of the field markers and the far goal, then it would turn to locate and grab the ball, at which point the shoot behavior would run and hence the robot would try to score a goal. Each run ended once the robot scored a goal or the ball went out of bounds. The experiment included 40 runs for both the localization-based and vision-based behaviors.

The purpose of the experiment was to help measure the general effectiveness and efficiency of the shoot behaviors and provide a means of quantitatively comparing both behaviors. The criteria for measuring the effectiveness included: number of goals scored, number of kicks taken, and number of times the shots taken hit the static goalie. Table 5.1 summarizes the results of the experiment.

state reacts to the timeout in the following way: if the opponent goal is visually detected in the field of view, the state transitions to *Kick*, otherwise the shoot on goal behavior aborts and returns a failure status to avoid a penalty called against the attacking robot for ball holding. The purpose of the transition to *Kick* is to get a shot off even if the robot isn't completely aligned or a robot is slightly blocking the shot because the location of the goal has already been visually confirmed and many times this shot, although not ideal, will help in the attack and even score goals. If not for anything else, taking a shot is meant to be a precaution against situations where the robot is aligned with the goal but in the last frame when the timeout occurs a robot spuriously appears or the goal disappears, causing the attacking robot not to take a shot when it should have. The complete shoot on goal behavior state machine is shown in Figure 5.2.

In summary, the shoot on goal behavior will try to position itself towards an open area of the goal as best it can, by dodging around opponents and visually confirming the location of the goal, to improve its chances of scoring when shooting. However it will also kick the ball if it is facing the goal, but not quite aligned, when the ball holding timeout is reached. In any case, the behavior will not kick the ball if it does not see the goal in its field of view, which avoids the biggest shortcoming of the localization-based shooting behavior: losing a scoring opportunity by not kicking the ball on target towards the goal or kicking the ball out of bounds.

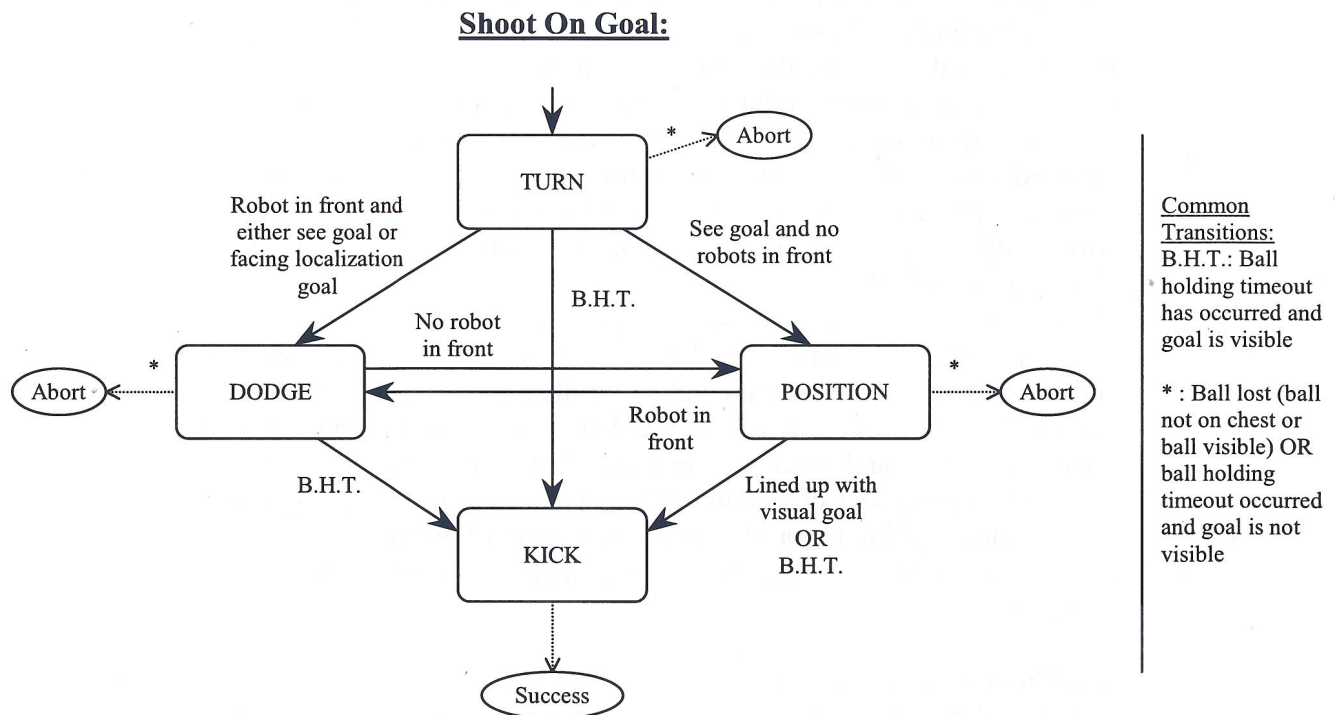


Figure 5.2: The finite state machine of the shoot on goal behavior, with the conditions for successful completion and aborting shown as well.

is unable to see the goal or get a clear shot, resulting in no shot being taken, or when it loses possession of the ball or is forced to abort due to the ball holding timeout. By rule, the robot is only allowed to hold the ball for a maximum of three seconds, after which a penalty is called and the robot is removed from the field. Thus, the shoot on goal behavior must be able to create a clear shot for the robot and kick the ball within this short amount of time.

The behavior has four states: *Turn*, *Dodge*, *Position*, and *Kick*. Following is a description of each.

- *Turn*: The initial state of the behavior. Upon entering the state the direction of turning is chosen which most quickly aligns the robot facing forward with the goal according to its localization estimate. The robot then turns at full speed in this direction until it visually detects a goal in its field of view, in which case the state transitions to *Position* if the view is clear, or to *Dodge* if it also detects a robot obstructing its field of view. The robot will also transition to *Dodge* if it finds a robot obstructing its view and is facing the goal location according to localization, without actually seeing the goal. This handles the case where localization is correct but no goal is detected from vision due to an occluding robot, and therefore the robot must move in order to obtain visual confirmation of the goal.
- *Position*: The robot enters this state after it has detected the opponent goal through vision or if it has dodged around a robot thought to be occluding direct view of the goal, and its purpose is to align the robot directly with an open area of the goal in order to have the best chance of scoring before shooting the ball. The robot turns at a speed relative to its desired turn angle while trying not to overshoot to avoid extra positioning. The turn angle is based on the visually detected goal position, or the localization goal position if the goal has not been seen yet. The state transitions to *Kick* if the robot visually detects and is lined up with an open area of the opponent goal. If at any point the robot detects a robot blocking its view, the state transitions to *Dodge*.
- *Dodge*: The robot enters this state when it detects a robot blocking its potential shot on goal. The attacking robot avoids the obstructing robot by stepping sideways at full speed, with the ball controlled under its chin, in the direction that leads the robot most inwards into the field. This way, the robot is able to dodge opponent robots and gain the best possibility of a clear shot on goal, while avoiding stepping out of bounds. Once there are no robots visually detected in front of the attacking robot, the state transitions to *Position*.
- *Kick*: The robot executes a forward kick, after which the behavior exits successfully.

In addition to state transitions described above, each state besides *Kick* reacts to losing the ball and the ball holding timeout. If the ball is no longer detected to be at the chest position under the chin by analyzing the chest IR readings, or if the ball is visible, the robot is deemed to have lost possession of the ball and the shoot behavior is aborted. The behavior also keeps track of how long it has maintained possession of the ball under its chin, and once the counter reaches 2.8 seconds, the ball holding timeout occurs. Each

localization-based behavior, namely shooting at open areas of the opponent goal instead of always at the center, and dodging around occluding robots to gain clear shots on goal. A diagram illustrating the differences between the two approaches is shown in Figure 5.1.

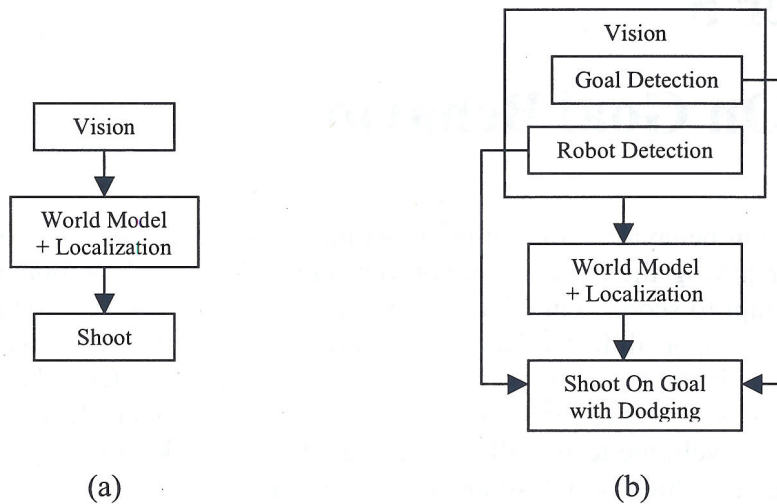


Figure 5.1: (a) The traditional shoot behavior input model, which uses only localization estimates when shooting towards the goal. (b) Shoot behavior input model with new approach that uses visual input to target open areas of the opponent goal and dodge around opponent robots to create clear shot opportunities.

Vision-based approaches to the shoot on goal behavior have been used successfully in the Four-Legged League for the past few years, starting with its introduction by UTS Unleashed![10] at RoboCup 2004 when there were still walls around the field. Since then and since the removal of the walls many teams have followed suit by implementing their own version of the behavior, and teams like Nubots[11] and rUNSWift[12] have scored many goals as a result of using the behavior and have done very well in the competitions. The vision-based shoot on goal behavior has proven to be one of the league’s key improvements in game play and represents a great example of a behavior that selectively uses visual information to override previously gathered information about the world state and robot location, in an attempt to eliminate sources of error when performing a critical task. In the next section we will present our own algorithm for the shoot on goal behavior and subsequently demonstrate its effectiveness with experimental results.

5.2 Shooting Algorithm

The shoot on goal behavior is used when the attacking robot is in the vicinity of the opponent goal and is activated only after the robot has gained possession of the ball and controls it under its chin. The behavior executes successfully when the robot has found a clear shot and kicks towards an open area of the goal. Failure cases occur when the robot

Chapter 5

Shoot On Goal Behavior

The shoot on goal behavior is one of the most important behaviors in robot soccer. It is responsible for having the attacking robot kick the ball towards the opponent goal in a best effort attempt to score. Put simply, it is the behavior that scores goals. A behavior with this type of responsibility necessarily requires careful implementation with attention to detail, because if the behavior performs poorly it negatively affects the team's overall performance, and such is the case with the shoot on goal behavior. If the shoot behavior doesn't do its job well, the team will not score goals, thus making it very difficult for the team to win games, which is undesirable to say the least.

In this chapter we will discuss two different approaches that we have used to implement the shoot behavior: the traditional localization-based approach and the new visual input based approach. We will then describe the vision-based shoot on goal algorithm in detail and provide results of experiments comparing both approaches in practice and demonstrate the overall effectiveness of the new vision-based approach.

5.1 Shoot Behavior Approach

The traditional way to implement this behavior was to have the attacking robot trust its current localization estimate and turn and kick towards the center of the global goal position in an attempt to score. Clearly, this method relies greatly on the accuracy of the robot's localization estimate at the moment right before kicking, and is especially vulnerable to errors in the robot's heading estimate. For many years our team has used this approach for shooting on the opponent goal, and with much success. However, up until recently, walls enclosed the field of play such that the ball would never be allowed to exit the field. This diminished the shoot behavior's reliance on accurate localization, because if the shots on goal were off target they would hit the back wall and allow our team to regain control of the ball close to the opponent goal and try again. Now, there are no walls surrounding the field of play and when balls are kicked out the back line they are replaced at the midfield line, which gives the opposing team a great opportunity to start an attack. The removal of the walls along with the rule changes has forced us to re-evaluate our shooting behavior and implement a new version that is capable of overcoming any errors in the robot's localization estimate during execution.

In order to overcome the deficiencies of the localization-based approach we have chosen a new approach which uses visual input to confirm the location of the opponent goal before performing a kick. In addition, by introducing visual input to the behavior it allowed us to implement functionality that simply wasn't possible with the original

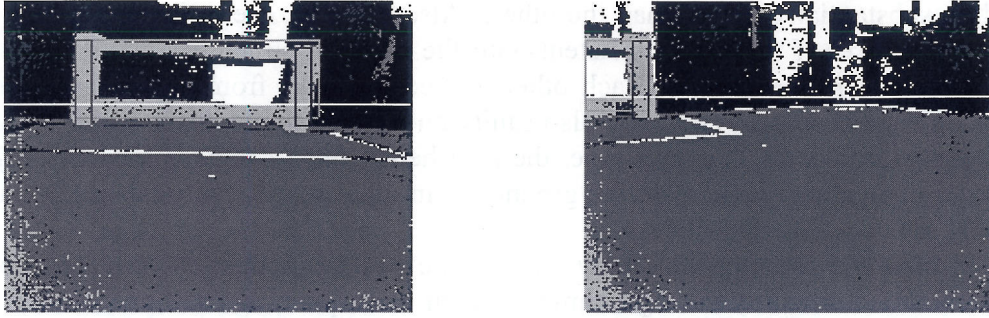


Figure 4.3: Various goal detection results taken from an observing robot at different locations on the field and camera positions. The red box denotes the left post, and the blue post denotes the right post.

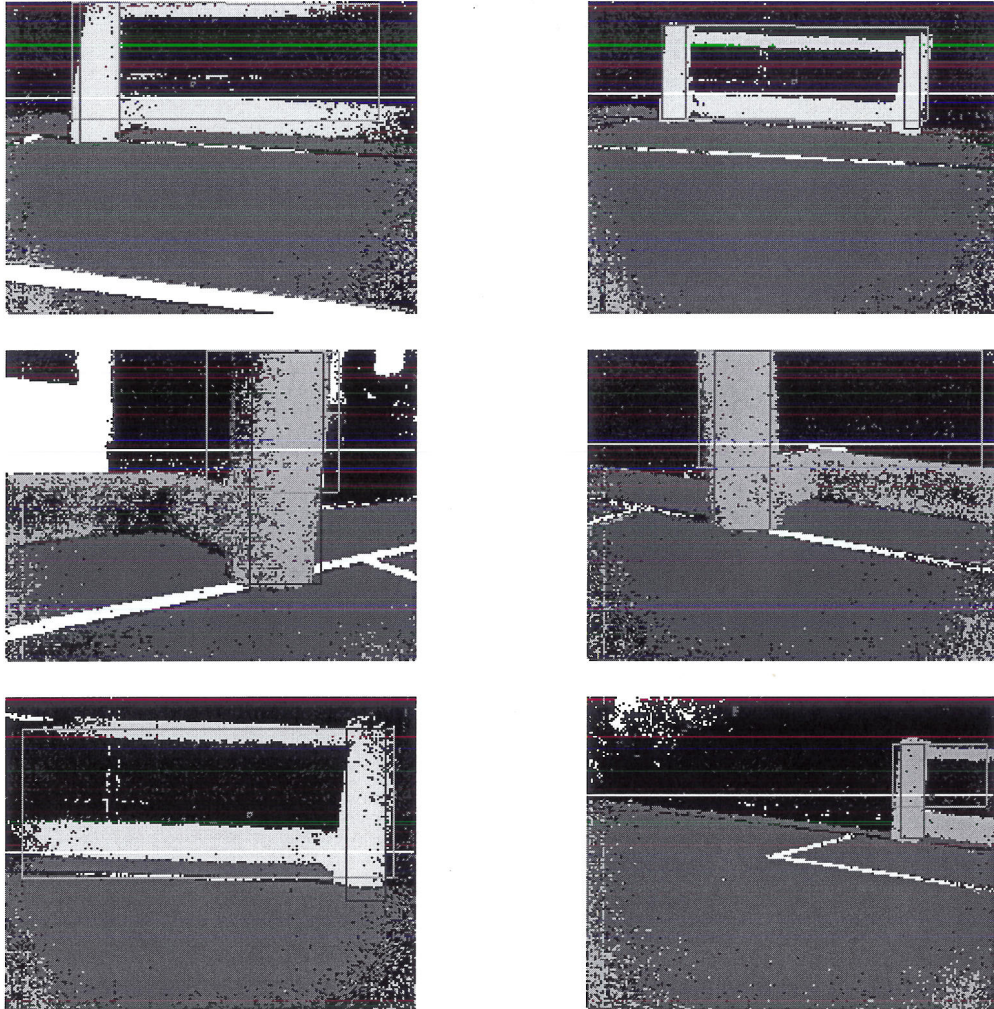
The results show the effectiveness of the goal detection algorithm, as it is able to reliably detect the goal region and goal posts from the viewpoint of various locations on the field. The results also show that the goal detection algorithm is capable of handling multiple cases of the goal position in the image, whether it be off to one side or the other, close and dominant in the image, or far and with both posts visible.

should be substantially bigger than the other. Also, the distance estimates to both posts can be used to see if they are consistent with the physical structure of the goal. If the posts estimates are too close to each other, or too far apart from each other, the least reliable one is discarded. There are also sanity checks for single posts, based on certain characteristics for posts. For example, the post height must not be below the projected plane from the camera parallel to the ground, or in other words, the post height must be verifiably above the head of the robot.

If after the running the sanity checks, the posts still remain, or at least one of them, they are returned by the algorithm with their corresponding positions relative to the robot, calculated from the distance estimates, pixel position bearing and camera location.

4.2 Visual Detection Results

To demonstrate the effectiveness of the algorithm, Figure 4.3 provides sample visual goal detection results for goal images captured by an observing robot at multiple positions on the field.



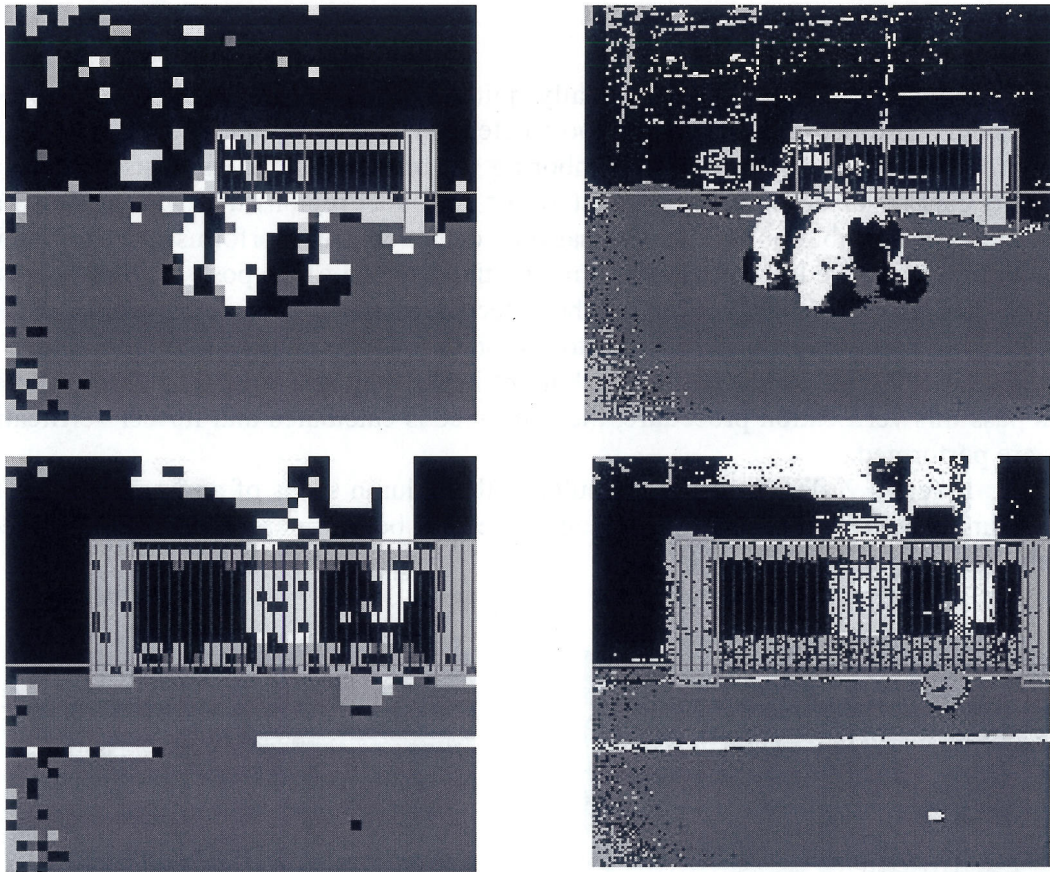


Figure 4.2: Images showing both the reduced color segmented image, and the original along with the goal column analysis results and goal post detections. Red columns are posts, blue columns are crossbars, and green columns are bottom portions.

4.1.3 Post Verification and Distance Estimation

The distance to the detected posts is calculated similar to the far robot detection method, where the distance based on projecting a ray from the camera through the pixel position of the post to the ground plane is compared to the distance returned by using the focal length of the camera, pixel height of the post, actual physical height of the post and the ratio property of similar triangles. It is the same procedure, except that with the posts, the focal distance method is also used to compute the distance based on the width of the post.

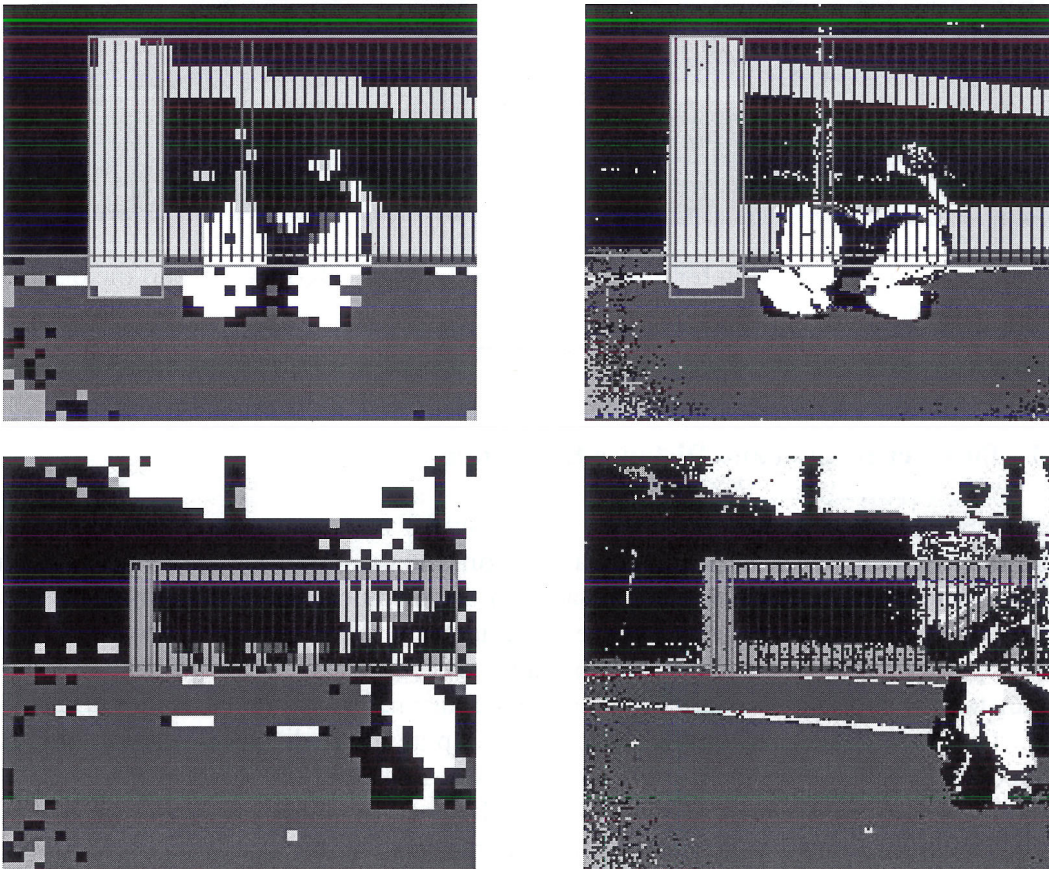
Once all three methods are used to compute their distance to post, the final distance is chosen among them based on the result that returned the closest distance, as this is deemed to be the most reliable. The goal detection algorithm allows for detecting goal posts within about 2 to 2.5 meters from the robot.

Once the final distance is calculated for the detected goal posts, a series of sanity checks are performed to verify the goal posts. Most of the sanity checks occur when there a two posts detected in the goal, as these two posts need to agree with each other on many things. For example, the posts should have relatively the same width, no post

4.1.2 Goal Post Detection

The detection of the goal posts is actually quite simple once the columns of the goal region have been identified. The goal post detection algorithm first iterates through the classified columns and groups all neighboring columns classified as posts. There are only two goal posts, so only the two outermost goal post regions that were grouped are considered for possible posts. The visual post detection only performs one verification procedure, the rest are left for when estimating the distance to the post, and that is ensure that detected posts are not “broken”. The detected posts are considered broken if there exists a path from one side of the post to the other through non-goal colored pixels. A path is a collection of connected pixels, but not in the diagonal direction. If the detected posts pass this verification procedure, their distance is calculated and further verification tests are performed.

. Figure 4.2 shows example results of the column scans of an image, where the post columns are red, crossbar columns are blue, and bottom portion columns are green, and the resulting post detections.



the vision module. This is the same image reduction from the near robot detection algorithm, and it only runs once, but both vision algorithms make use of it.

b) Goal Region Analysis: The largest goal-colored region of the image is analyzed by scanning the columns within its bounding box, and identifying each column as a post, crossbar, or bottom portion. Each column type has its own characteristics; for example, a crossbar column would have goal-colored pixels near the top followed by a greater number of non-goal-colored pixels.

c) Goal Post Detection: Neighboring columns labeled as posts within the goal region's bounding box are grouped to determine the location and edges of the goal posts. The center of the posts is useful for localization and the inner edges can be useful for behaviors that shoot towards the inside of the goal. The height of the posts is used for distance estimation, however when the post is close and the actual height cannot be determined, the width is used instead.

4.1.1 Goal Region Analysis

Once the largest goal colored region in the image is identified, it is checked against the field horizon line. If the goal region intersects the field horizon line and is of a large enough area, it is considered to be the goal. Once the goal is located in the image, its bounding box is analyzed to help determine the location of the posts, which are the most important features of the goal, as the robot uses the relative post locations for localization. The goal region itself, and the robot's bearing to it, is also returned by the algorithm so that behaviors may use the location of the goal as additional visual input when executing a task, such as shooting towards the goal.

The algorithm scans each column in the goal bounding box, identifying each as a post, crossbar, or bottom portion. Each type of column is identifiable by the location of the goal colored pixels along the column.

Post columns ideally will have every pixel in the column be of the goal color, and this is what the algorithm looks for when discerning whether or not the column is a post. Of course, the algorithm allows for some tolerance of non-goal colored pixels, but they must be spread out across the column, or it is liable to instead get labeled as a crossbar or bottom portion.

Crossbar columns start with the goal colored pixels at the top and quickly turn to non-goal colors. If the column follows this transition and halfway down the column from the top there are at least as many non-goal pixels as there are goal pixels, the column stops the scan and labels the column a crossbar.

Bottom portion columns occur when the crossbar of the goal is not visible in the image, most likely because the observing robot is too close to the goal. These columns are characterized by having goal colored pixels at the bottom and at least as much non-goal colored pixels on top.

Once all of the columns in the goal region's bounding box have been scanned, the algorithm moves on to the detection of the posts. The post detection procedure is discussed in the following sub-section.

Chapter 4

Goal Detection

The introduced modification of the structure of the goals in the Four-Legged League has led to the development of a new goal detection algorithm. The new goals have two posts, one on each side, a top crossbar, and small walls on the bottom positioned where the net would be in a human soccer goal. The goals are of a single solid color as in previous years, and are either cyan or yellow, depending on which side of the field they are located. Figure 4.1 shows pictures of the two goals. The robot is able to localize itself on the field using the new goal more accurately than using the previous goal model, which consisted of two side walls and one wall in the back all of the same height and color, because verifying the location of the goal edges is now easier due to the discrete posts.

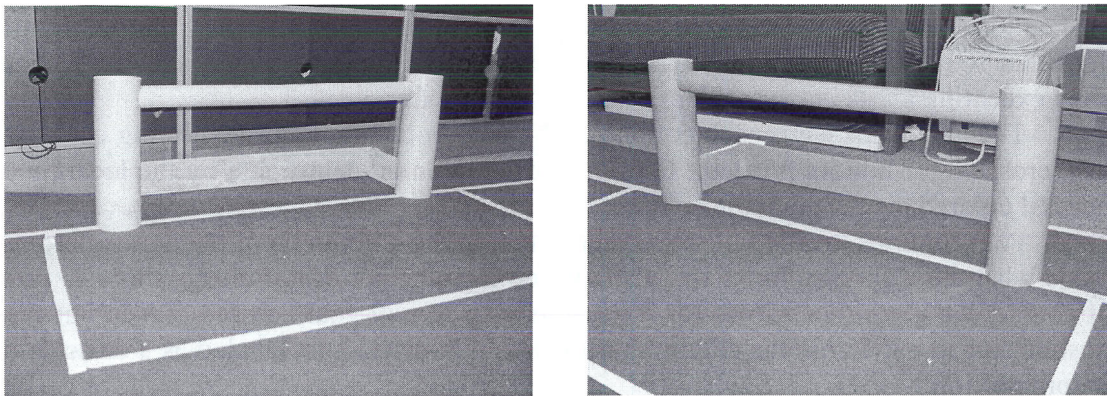


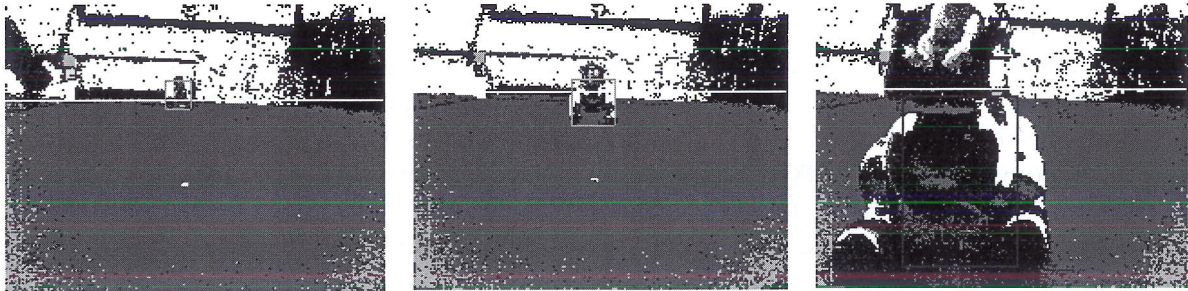
Figure 4.1: The yellow and cyan goals

4.1 Goal Detection Algorithm

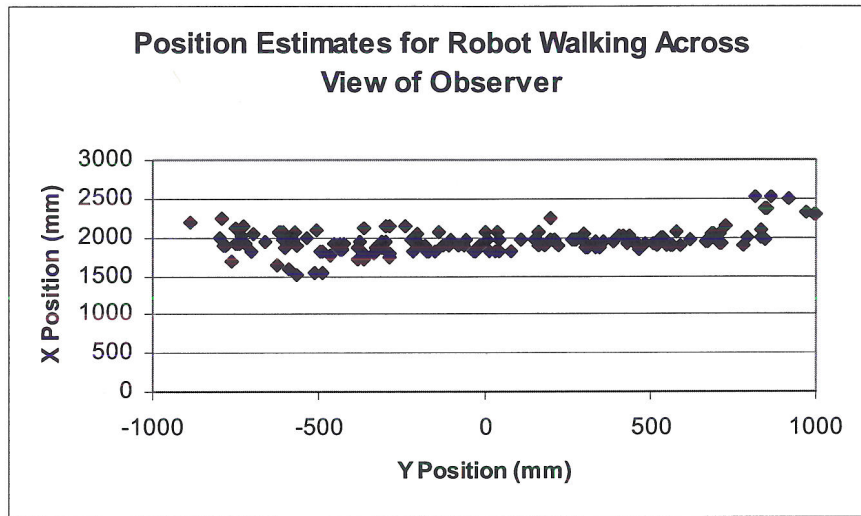
The goal detection algorithm focuses on finding the location of the goal posts in the image, as the posts provide the most information about the robot's relative location to the goal, which helps in self-localization. The algorithm is composed of three main parts:

a) Image Reduction: The original 208x160 color segmented image is reduced by a factor of four in both the horizontal and vertical directions by reading pixels from scan lines that are parallel and perpendicular to the image horizon. The horizon aligned scan lines allow the image processing to be independent of the camera rotation. This image rescaling simplifies the recognition task, and is available to all detection methods within

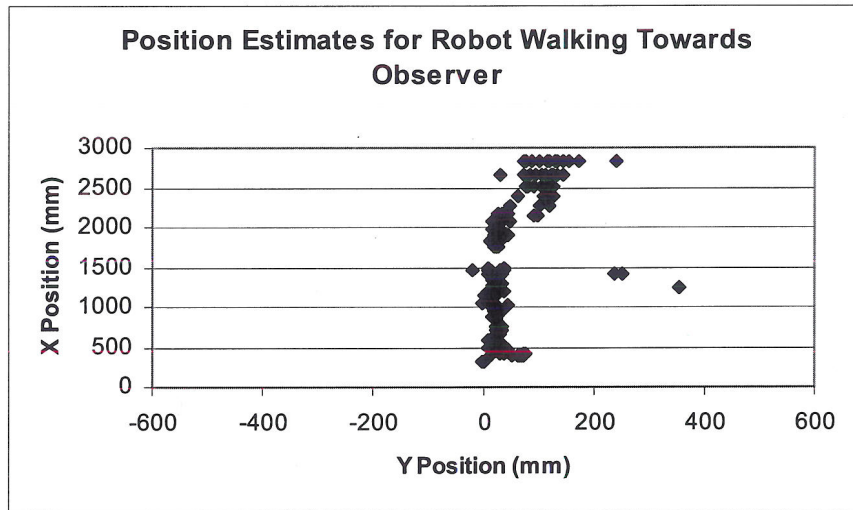
The recorded (x,y) positions are in body local coordinates, where the x-axis represents forwards/backwards displacement and the y-axis represents lateral displacement. The standing robot maintains a fixed head position looking forward for the duration of the experiment. In this experiment, the robot was able to successfully detect the moving robot as it walked across its viewpoint and towards it. The position estimates for the detected robot were fairly accurate as well, with the robot estimating the 2m walking line correctly and the oncoming robot's position along the forward x-direction starting from 3m away.



(b)

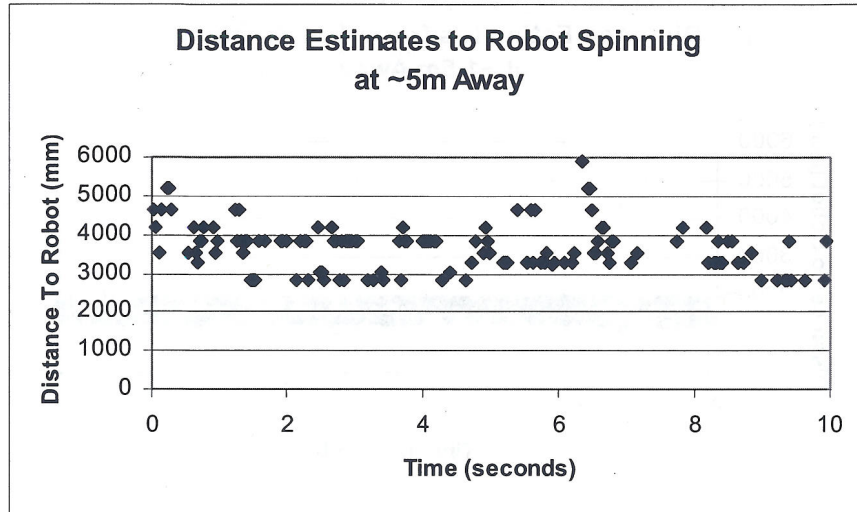


(c)



(d)

Figure 3.13: (a) Images of robot walking in front of viewing robot at $\sim 2\text{m}$ away
 (b) Images of robot walking towards viewing robot from about $\sim 3\text{m}$ away
 (c),(d) Recorded robot location estimates during time of robot walking



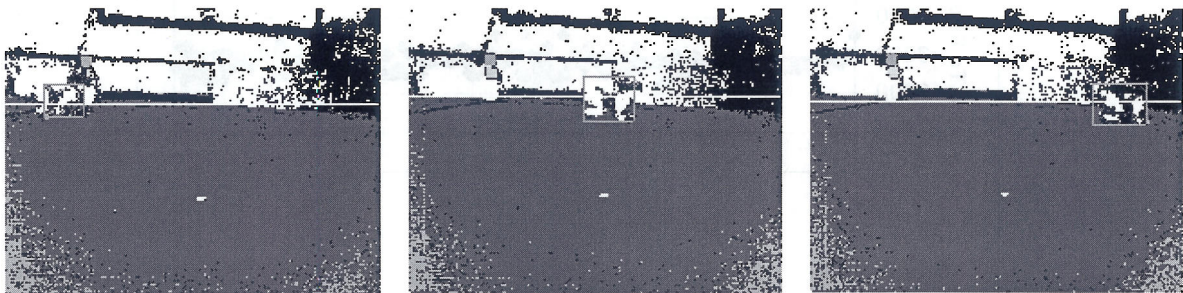
(e)

Figure 3.12: (a) Robot at ~1.5m away (b) Robot at ~2m away (c) Robot at ~3m away (d) Robot at ~5m away (e) Graphs of recorded distance estimates for observed spinning robot

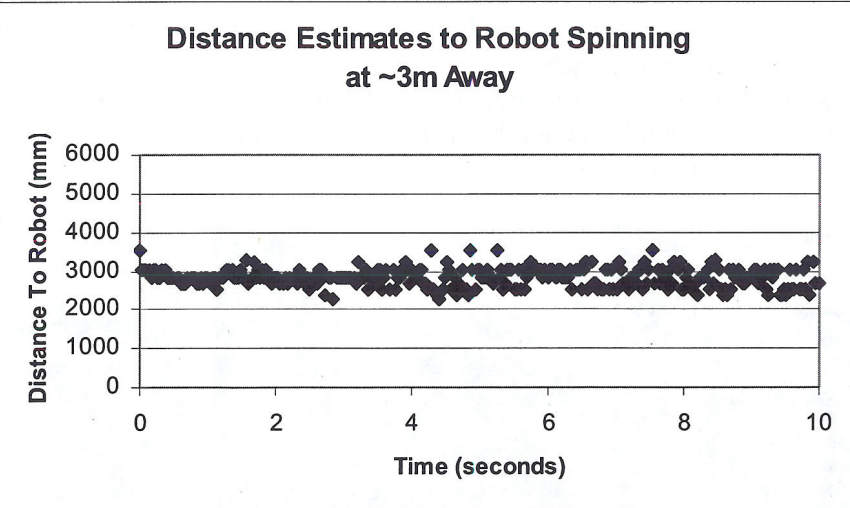
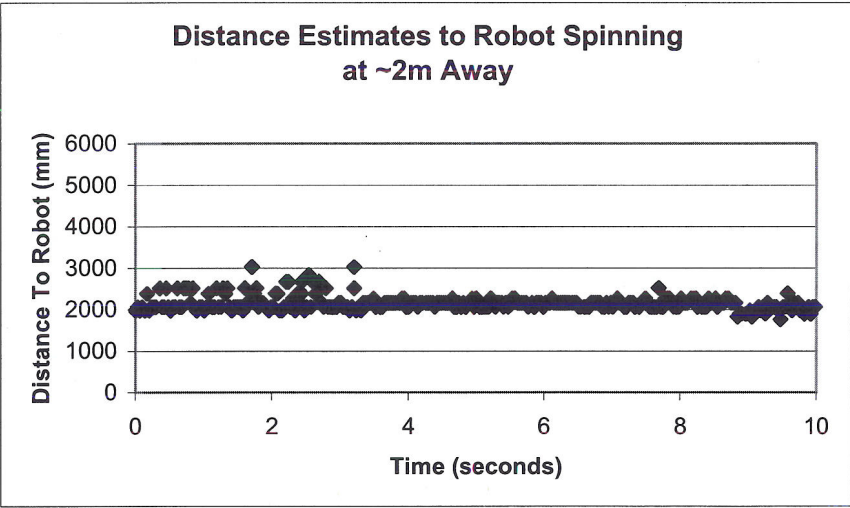
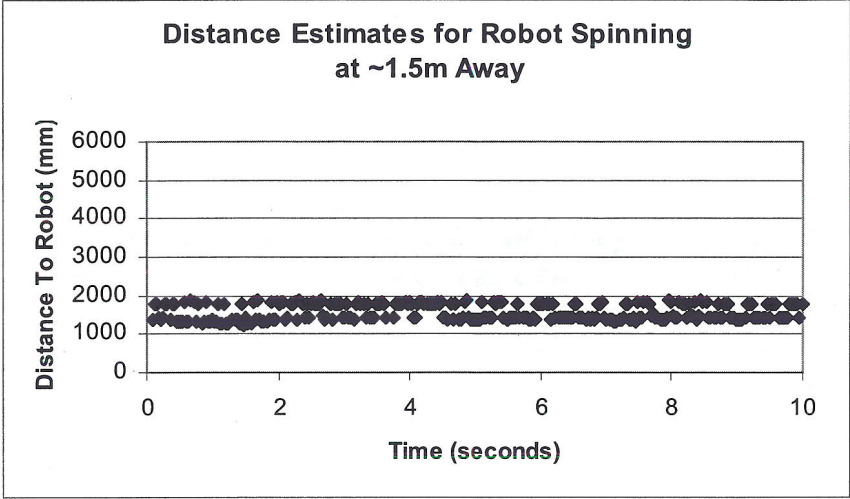
In this experiment, the robot was able to reliably detect the robot at all its orientations as evidenced by the frequency of the detection points. The spinning robot was rotating at approximately 60 degrees/second, so it was fast enough for the observing robot to capture all orientations of the robot. The distance estimates were fairly good, usually falling within a 500mm window of the correct distance, which is acceptable for far-range robot detections. The exception was the 5 meter case, where the distance estimates were jumpy and the detection of the robot not as frequent. Still, the distance estimates returned were generally above 3 meters, which suggests that behaviors could threshold distance results to determine their accuracy.

3.8.2 Experiment 2

The second experiment involves a standing robot observing a moving robot directly in front of it walking from left to right along a line a distance of approximately 2m from the robot, and also walking directly towards it starting from approximately 3m away. Figure 3.13 shows the results which include both the x and y coordinates of the detected robot during the time of robot crossing.



(a)



intersect with the ground plane at all. The robot box pixel height can be jumpy at times, which causes the distance estimates also to be noisy and incorrect.

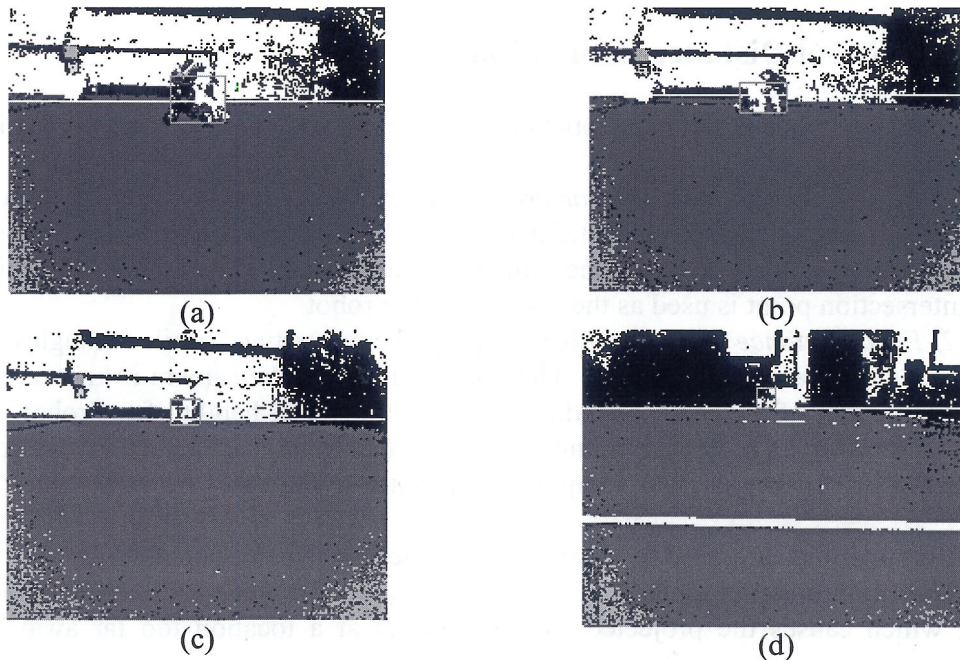
For each robot box, both methods are used to compute the estimated distance to the robot. The default final distance to return is the one computed from the pixel height method, which is more reliable for robots that are farther away. However, as detected robots get closer, the projection estimates are more reliable. Therefore, the ray projection distance is used when it is less than the pixel height distance, and is greater than the minimum far distance (1.2 meters) or if the far robot box is not skinny (box height > box width), which generally signifies that the pixel height method returned a reasonable distance estimate. In general, the distance returned is from the method deemed to be the most accurate for the given robot detection given the circumstances of the detection frame.

3.8 Experimental Results

To test the effectiveness of the robot detection algorithm we ran two experiments: the first tests the ability to detect robots at multiple different views at varying distances, and the second tests the detection of robots at various locations in the image.

3.8.1 Experiment 1

The first experiment involves a standing robot observing another robot spinning in-place directly in front of it at four separate distances: 1.5m, 2m, 3m, and 5m away. The robot being observed is spinning in-place in order to test the detector's ability to recognize robots at all possible orientations, while the standing robot maintains a fixed head position facing forward during observation. Experimental results are shown in Figure 3.12.



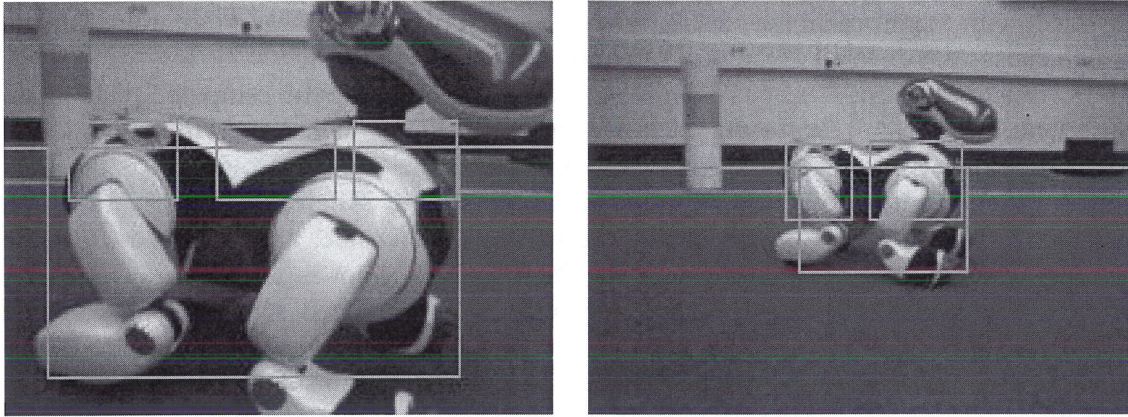


Figure 3.11: Images showing close robots with near and far robot detection boxes overlapping. The far robot boxes are subsequently removed, and the near robot boxes validated.

3.6 Far Robot Team Color Estimation

The team color designation procedure for the robot boxes is actually quite simple and works fairly well in practice. As the columns get scanned for edge responses and robot box creation, the algorithm keeps track of the number of red pixels that are found. This way, the number of red pixels in each robot box created is known. A robot detection box with 3 or more red pixels is designated as a robot on the red team, otherwise it is designated as a robot on the blue team. The red pixel count is used instead of the blue count because the blue color often times appears in the image as noise, or in shadows and other dark colors, and so the blue pixels are less likely than the red pixels to be part of a robot uniform.

3.7 Far Robot Distance Estimation

There are two methods that the far robot detection algorithm uses to compute the distance to detected robots:

1) *Ray projection from the camera to the ground plane*: A ray from the camera position is projected through the center pixel of the bottom line of the far robot bounding box and the intersection point with the estimated ground plane is computed. The distance to this intersection point is used as the distance to the robot.

2) *Focal distance and robot pixel height*: The property of similar triangles is used to compute the distance to the robot. The focal distance of the camera is known, as well as the pixel height of the robot bounding box, and the actual height of the robot. Using the ratio for sides of a similar triangle the distance from the camera to the robot is computed, which is then used to compute the distance to the robot from the center of the body.

Both methods are used to compensate for deficiencies or potential errors with the other. The ray projection method fails because the ground plane estimate is not always correct, which causes the projected ray to intersect at a location too far away or not

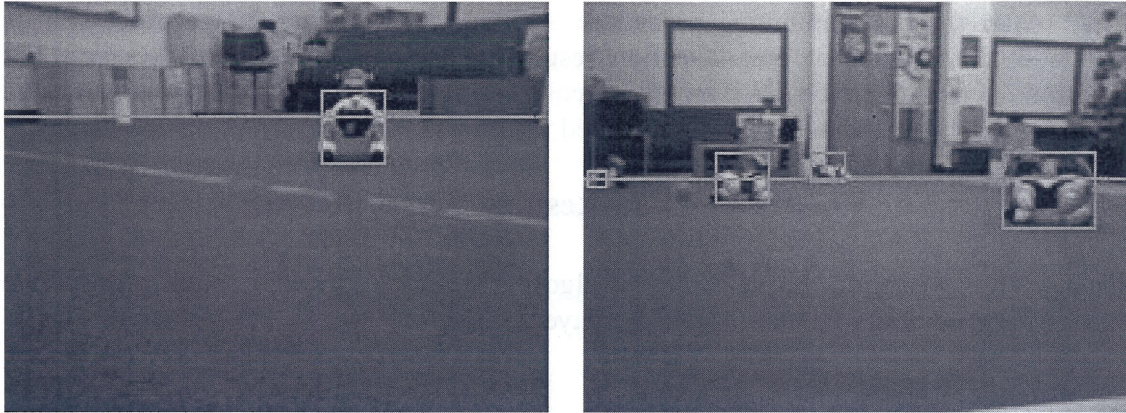


Figure 3.10: Collection of visual far robot detection results, showing the estimated field horizon line and the final robot detection boxes.

As evidenced by the visual results, the far robot detection algorithm is capable of detecting multiple robots in a single image, robots at various different distances from the observer, and robots positioned at many different orientations. The results also show that the algorithm is able to tolerate some amount of blurring due to movement of the head of the observing robot. In the results shown, there is one false positive in the background in the image. False positives can appear in the background clutter, but are usually rare in normal environments and tend to disappear on subsequent frames taken of the scene. The algorithm is actually quite robust to background clutter, as you can see from the visual result images, which all contain many edges and sharp contrast in the background near the horizon line, yet the algorithm is capable of eliminating these edges from consideration for robot detection results.

3.5 Far and Near Robot Detection Overlap

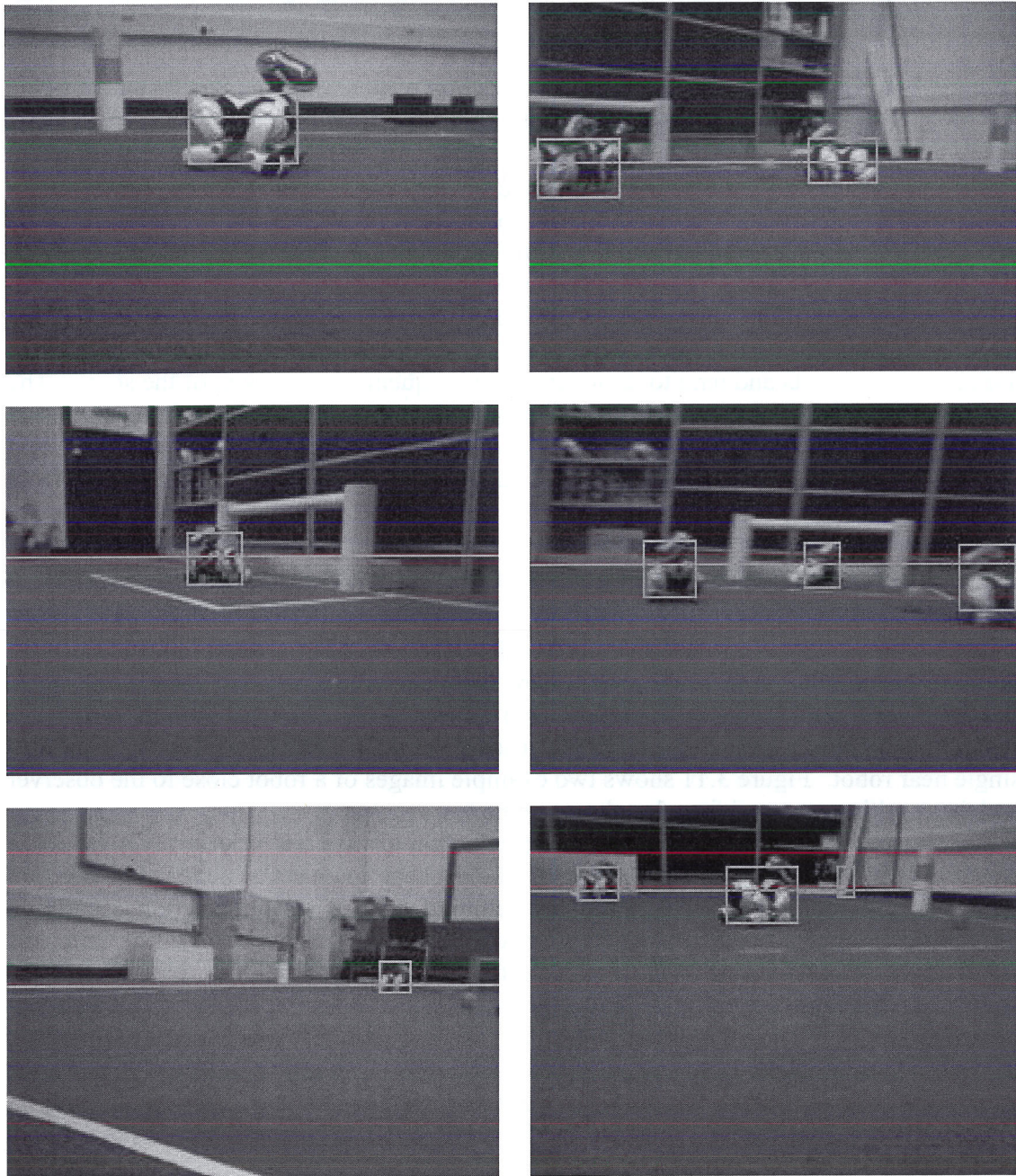
There are times when far robot detection results overlap with near robot detection results, in fact, this is always the case when there are near robots in the image. The edge detection results common in far robots also appear in near robots, though with more separation between edge clusters, which leads to multiple far robot observations in a single near robot. Figure 3.11 shows two example images of a robot close to the observer and the resulting near and far robot detection boxes.

To resolve the issue, the near robot detection always has priority over far robot detection results. Therefore, all far robot boxes that intersect a near robot box are removed from consideration and are not returned by the robot detection algorithm. Furthermore, since all near robots have far robot detection results attached to them, the far detection results can be used to validate the near detection results. This way, if the near robot detection algorithm produces a robot box that does not intersect any far robot boxes, then it is discarded to avoid returning false positive near robot boxes. Of course, this validation procedure only occurs if the far robot detection algorithm is run in the current processing frame, for example when the camera is level. There is no extra validation procedure for near robots if the far robot detection algorithm does not run in a given frame.

After the box combining procedure is finished, no further image processing is done and the final robot box detection results represent the output of the far robot detection algorithm, excluding the team color and robot position/distance estimates, which are produced by the methods described in sections 3.6 and 3.7.

3.4 Visual Far Robot Detection Results

Visual results of the far robot detection algorithm, which include images of robots at various distances and orientations, are displayed in Figure 3.10.



or directly away from the camera, because both of these robot orientations have a middle region that produces a very small amount of edge responses. These split-up boxes generally have the characteristic of being close to each other in the image space, skinny (box height > box width), and with similar box heights. The detection algorithm attempts to group boxes pertaining to the same physical robot by iterating through the list of detected robot boxes, comparing each box to the next in the list, and applying the following criteria:

Criteria for combining two robot boxes:

- Both boxes are skinny, meaning their box height > box width
- Both boxes have width < 100 pixels
- The boxes have similar height below the horizon line, satisfying at least one of these conditions:
 - Height difference ≤ 3 pixels
 - Neither box height below horizon line is less than half of the other
- The boxes are close to each other, satisfying at least one of the following conditions:
 - Distance between closest box edges < 10 pixels
 - Distance between closest box edges < Either of the box widths

If two boxes in the robot box detection list satisfy these conditions, then the first box (the box closest to the front of the list) is merged into the second box, and the robot box combining procedure continues by then comparing this newly combined robot box with the next robot box in the list. This way, multiple boxes that are part of the same physical robot have the possibility of being combined. Figure 3.9 shows an example image where multiple boxes were detected for a single robot initially, and the result of the detection after running the robot box combining procedure.

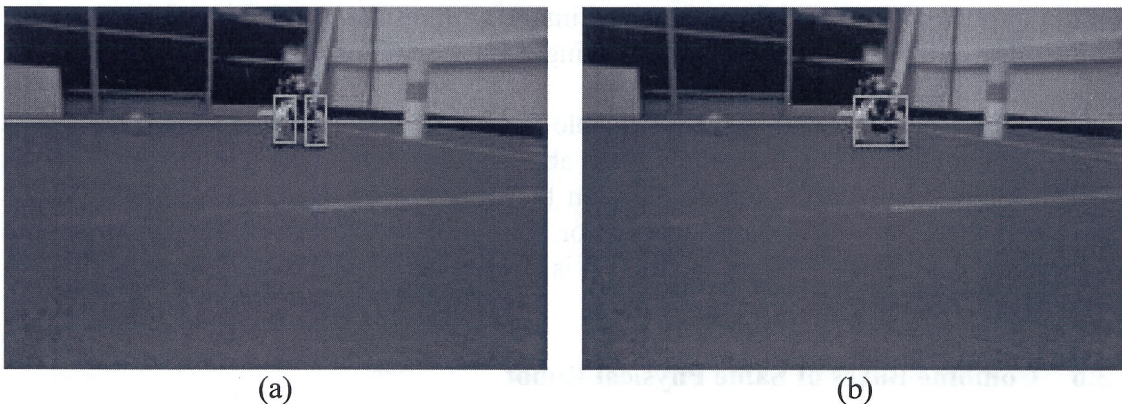


Figure 3.9: (a) Initial robot detection results for scene with robot on playing field facing observer. (b) Robot detection results of same scene after running the procedure to combine robot boxes of the same physical robot.

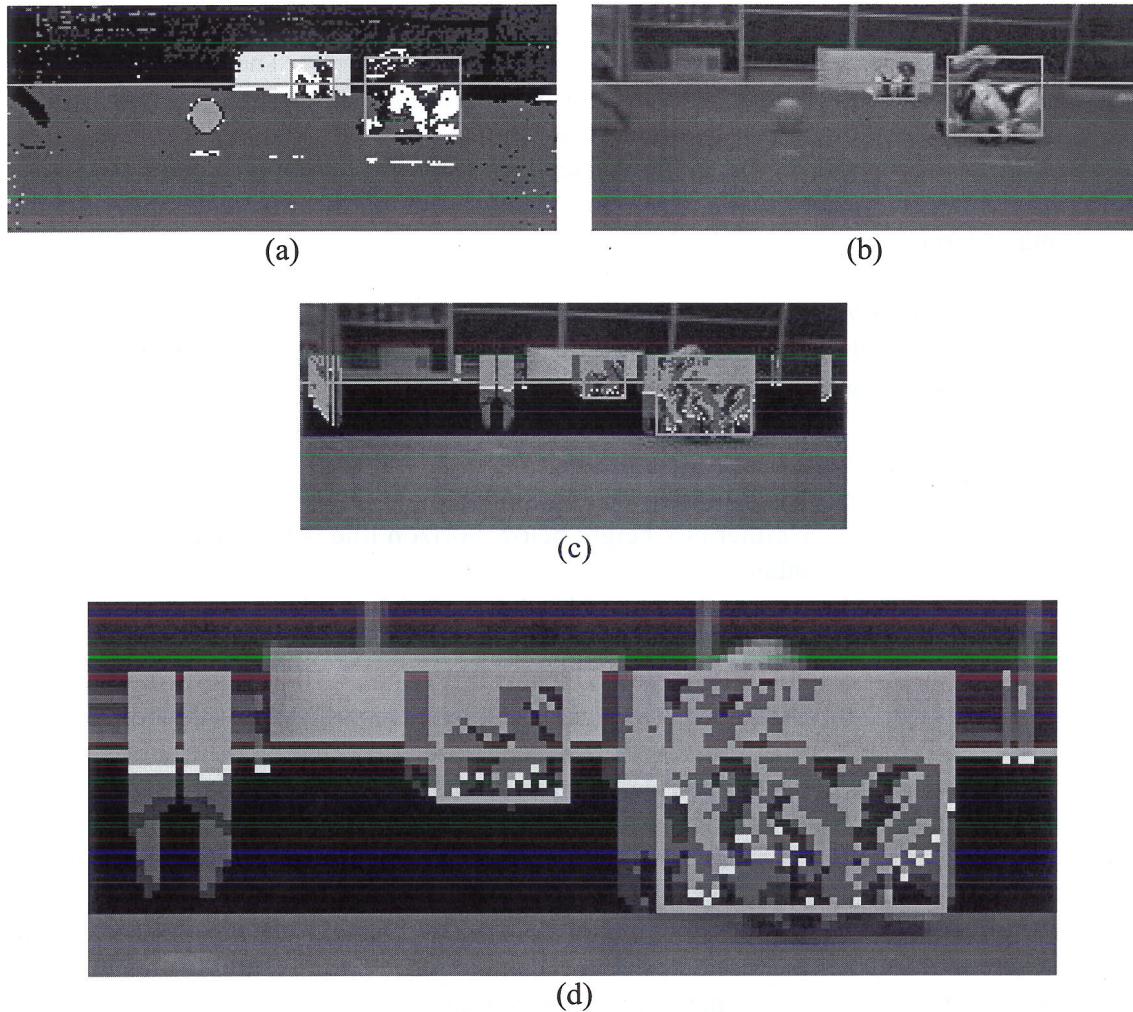


Figure 3.8: Sample far robot detection results with illustration of column scanning for building robot boxes. (a) Color segmented image with results. (b) Grayscale image with results. (c) Illustration of column scanning. (d) Close up of algorithm illustration.

Notice that if a column scan is unable to find an edge response whose strength is greater than or equal to 30, the column scan aborts after reaching the horizon line. This is because robots of the algorithm assumption that robots must intersect the field horizon line. So if there are no edge responses at or below the horizon line, there is no need to continue scanning up the column because it is already assumed that no robot parts will be found in the column.

3.3.6 Combine Boxes of Same Physical Robot

There are times when the algorithm detects two or more separate robot boxes for the same physical robot. These boxes are close to each other, but are separated because the algorithm did not find edges with strength ≥ 30 in the columns between them. This typically happens when the robot being observed is oriented directly facing the camera,

3.3.5 Robot Boxes

The far robot detection algorithm tries to build robot boxes as it scans through the image. Once one column is verified to have robot properties (field end, strong edges), a robot box is started. The algorithm moves across the image from left to right, column by column, so it only builds one robot box at a time. The idea for a robot box is to group neighboring columns with normal to strong edges, that is, each column must have at least one edge with magnitude ≥ 30 and validated with an end of field position below the horizon line. Each subsequent column is evaluated for addition to the current box, and the robot box expands only when the column scans find and add strong edge points. No regular edge points can control robot box boundaries except for those designated as pixel positions for the start of a robot/end of field in their column. If a subsequent column scan does not find an end of field position before the horizon line, or does not contain edges, the current robot box is closed and added to the robot box detection list if it satisfies the following conditions:

Robot box conditions:

- Box width ≥ 4 pixels
- Number of strong light-to-dark edges ≥ 3
- Number of strong dark-to-light edges ≥ 3

Again, strong edges are pixels with a gradient in the x-direction of magnitude greater than or equal to 80. The first robot box condition attempts to eliminate possible false positives, as even the farthest robots tend to have a pixel width of at least 4 pixels. The second and third conditions try to enforce the robot characteristic of having strong edge clusters of light-dark and dark-light transitions. The minimum number for each such transition is set to 3 to accommodate the furthest of robots detected, however the robot boxes generally contain many pixels with these edge properties.

Robot box expands only when columns find and add strong edge points, no regular edge points can control box boundaries except for those designated as pixel positions for the start of robot/end of field in their column. Figure 3.8 shows far robot detection results along with an illustration of the column scanning/robot box building process.

The legend for the illustration in Figure 3.8(c),(d) is the following:

- Red pixels: Light-to-dark transition of edge with strength ≥ 80 .
- Blue pixels: Dark-to-light transition of edge with strength ≥ 80 .
- Green pixels: Edge response with strength ≥ 30 .
- Yellow pixels: End of field positions for each column.
- Orange pixels: No edge response (i.e. edge strength < 30)
- Darkened pixels: Column results before end of field found, and are not used due to the end of field position being above them (Robot must be fully above field end point).
- Black pixels: No edge response with strength ≥ 30 found yet in column scan.

scanning the column up until 10 pixels above the field horizon. Once at least one edge with strength ≥ 80 is found above the field end, the algorithm creates a potential robot bounding box starting with this column. The robot box is created by adding pixel points to it, and it expands to fit all points contained within it. The first pixel points added to a robot bounding box when it is created is the pixel point for the start of the robot (the end of field position) and the first pixel point found above the field end with a strong edge magnitude. As other strong edges are found, they are also added to the current robot box to expand its boundaries. The robot box is not complete until the next column is scanned, since neighboring columns with strong edge results are grouped together. Figure 3.7 shows example scenes with their corresponding strong edge detections in the scanned portion of the image near the field horizon line. The red pixels represent strong light-to-dark transitions and the blue pixels represent strong dark-to-light transitions.

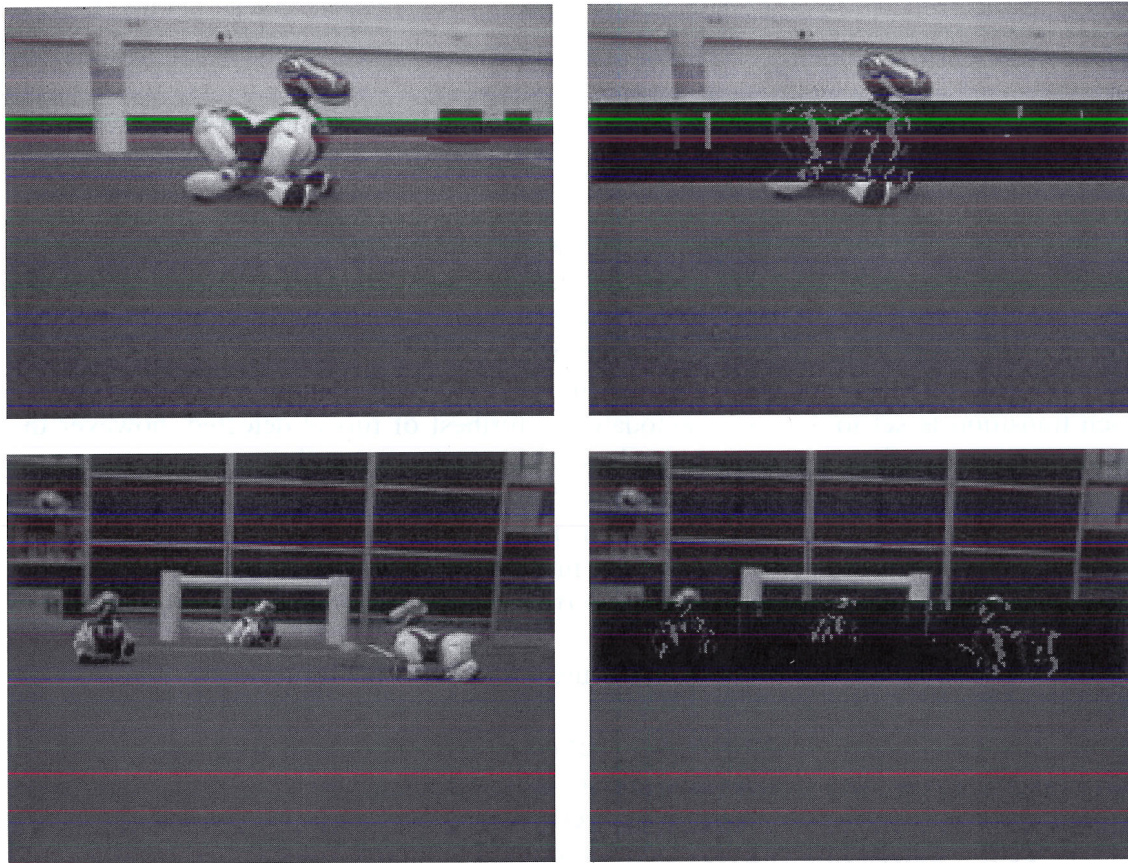


Figure 3.7: Two scenes of robots on the playing field, along with their strong edge detection results (edge strength ≥ 80). Red pixels are light-to-dark transitions and blue pixels are dark-to-light transitions, both of which are characteristic of robots in grayscale images.

range of 0-255. The strong threshold is meant to categorize edge responses characteristic of robots, yet not of the typical background noise seen near the field horizon of an image. However, robots also produce edge responses of lesser magnitude due to shadows and the contours of the robot, so the algorithm also keeps track of edges with strength ≥ 30 , which is a more normal edge response value and common among many objects in typical images taken on the field.

The columns of the horizon image window are scanned from bottom to top, and if no regular edge responses are found in a column (edge magnitude ≥ 30), then the column is not considered to be part of any robot, and the algorithm moves on to the next column. If the column scan finds a pixel with edge magnitude ≥ 30 , then the column may be part of a robot and the scan continues up the column to validate this possibility by first finding the end of the field pixel position along the column, and then confirming the existence of another normal sized edge response above this position.

3.3.3 Finding End of Field Along Column

After detecting the first normal sized edge response along the column, the algorithm must then find the pixel position along the column denoting the end of the field, in order to discard the normal sized edges found from field lines and the ball. Basically, if field colors are found above the initial edge response reading, then that edge is not considered to be part of a robot. The end of field position must be located on or below the field horizon line or else the column is discarded.

To find the end of the field along the column, the algorithm keeps track of the ratio between object colors and field colors. Field colors are green, orange, and yellow, which denote the green field, orange ball, and yellow goal. The cyan goal color is not included because cyan is almost always classified in the image noise along the borders and also on other objects, so it cannot be used as a definitive field color. Object colors are the colors other than these, such as black, white, red and blue. There is a counter for the number of field colors, and number of object colors found while scanning. If at any point the number of field colors becomes greater than or equal to the number of object colors, then both counters are reset to zero and a marker is set to the current pixel position along the column, as it might become the end of field position if object colors become prevalent along the column before hitting the horizon line. The marker is initially set to the initial edge response location.

The end of field position is found when the number of objects colors (*num_obj*) essentially overtakes the number of field colors (*num_field*), starting from the marker position. The two cases are: if marker-to-horizon distance > 8 and *num_obj* > 7 , or if marker-to-horizon distance ≤ 8 and *num_obj* ≥ 4 , then the end of field position is found and it is set to the marker position.

3.3.4 Light-to-Dark Transitions

The characteristic edges for robots are the combined light-dark and dark-light transitions with edge strength ≥ 80 . Once the end of the field in the column has been found, the algorithm keeps track of how many of each transition, light-dark and dark-light, are found above the end of the field position, or rather, the start of the potential robot, while

contrast between the robot body and its team uniform color is distinctively strong in the horizontal direction, such that it suffices to distinguish between robots and background clutter. Therefore, the vertical edge detection is not needed to reliably detect the robots, which is a good thing because it reduces processing time.

The distinguishing factor of the robot and uniform is that the image intensity transitions from light to dark, and then from dark to light in a very small window in the horizontal direction. This is the transition from robot body to uniform, and then back to robot body, and it happens for any orientation of the robot because the robot uniform has four individual faces, one for each side of the robot. In addition to this pattern, the magnitudes of the edge responses are very strong, which normally allows the robots to noticeably stand out from the background clutter, which also may contain horizontal edges.

In addition to the raw grayscale image, the far robot detection algorithm also uses the color segmented image during processing, even though it is noisy. The color image helps to filter out edge results from common features of the field, such as the field lines and the ball, which both produce horizontal edges, though not as strong as those coming from robots. The edges caused by lines and other field items could interfere with the bounding box estimate for the robots, making them much too large and hence causing bad distance projections. So, in short, the color segmented image is used to better distinguish the edge results coming from robots and those coming from lines or the ball, which should be discarded.

The far robot detection algorithm, in a nutshell, finds robots in the image by finding clusters of strong edge results that contain multiple light-dark and dark-light transitions, a distinctive characteristic of robot-uniform contrast. These clusters represent the far robot hypotheses, whose bounding boxes and relative positions from the robot are returned by the algorithm.

The following sub-sections will provide more details and elaborate on all of the steps of the algorithm.

3.3.2 Edge Detection

In order to find the robot clusters of strong edge results, the algorithm scans the image window around the horizon line column-by-column, starting from 20px below the field horizon and moving up to 10px above the field horizon. As it moves up each column, it computes the image gradient in the x-direction for the current pixel, which is accomplished by subtracting the neighboring pixels in the x-direction and multiplying by two:

- Image gradient in x-direction at image location (x, y) :
$$G_x(x, y) = (I_m(x+1, y) - I_m(x-1, y)) \times 2$$

Using the above formula, it can be seen that light-dark transitions produce a negative gradient, while dark-light transitions produce a positive gradient. The magnitude of many edge responses for robots is strong, with strong meaning having horizontal edge strength ≥ 80 . This threshold value was chosen after observing the edge responses of many images with robots in them, all of which have grayscale pixels in the

5) *Camera is not rotated*: The algorithm assumes that the head of the robot is level so that it need not worry about handling rotations of the camera, which adds more complexity and processing time. The assumption is enforced because if the head position is deemed to be rotated or unable to see the field horizon, the algorithm does not run. This assumption doesn't seem to negatively affect the robot, by enforcing unnecessary rules, as the most natural way to look for objects in the distance is to level the head with the field horizon in the image.

3.3.1 Far Robot Detection Overview

The main difference between the far robot detection algorithm and the near robot detection algorithm is that the far case primarily deals with the raw grayscale image, rather than the color segmented image, which is what the near algorithm uses. The grayscale image is used instead because the color segmented image is very noisy and much information is lost when converting from the raw image into the segmented image. All the information in the image is valuable when dealing with objects, such as far away robots, that cover only a small area in the image space. The algorithm must be able to effectively differentiate the robots from the background noise, which is very difficult when relying solely on the color segmentation. Figure 3.6 shows two images from the same scene, one grayscale and the other segmented, for comparison.



Figure 3.6: (a) Grayscale image of robot on playing field. (b) Color segmented image of the same scene.

The far robot detection algorithm only processes pixels in the image region around the field horizon, specifically from 10 pixels above the horizon to 20 pixels below the horizon, to speed up processing. If a robot were close enough to extend beyond the 20 pixels, the near robot algorithm would detect it; therefore there is a seamless integration between the two algorithms.

The algorithm computes the image gradient, or edge detection, in the x-direction for each pixel in the image window around the horizon. The edge detection results provide the distinguishing features that allow for detecting far robots in the image. The algorithm only computes the gradient in the x-direction, horizontally, because the

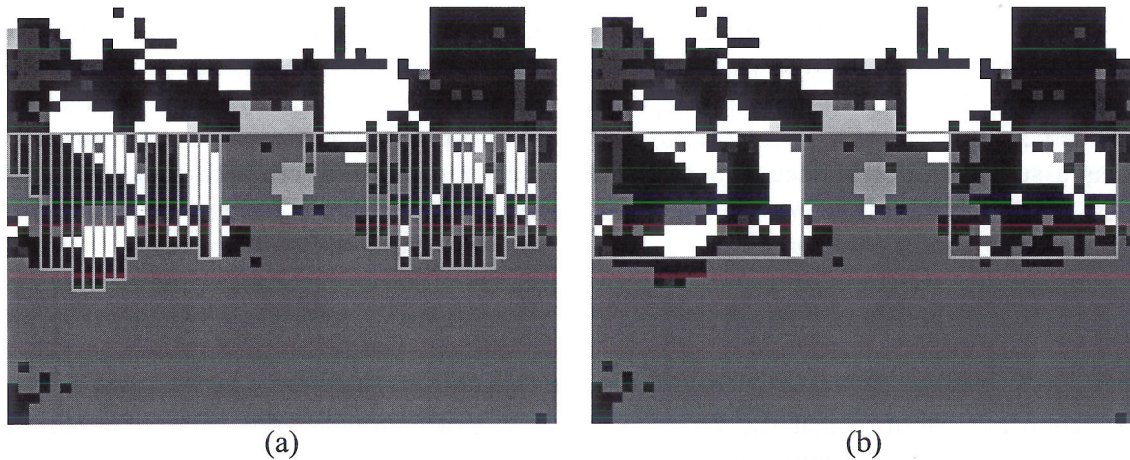


Figure 3.5: (a) Reduced image with initial column scan results for obstacle colors shown
 (b) Reduced image with final robot bounding boxes after grouping neighboring object columns

3.2.4 Near Robot Color and Distance Estimates

The team color is determined by searching for red regions within the bounding box of the detected robots. If convincing red regions are found within a robot's bounding box, the robot is said to be on the red team; blue team otherwise. The distance to a detected robot is calculated by taking the center pixel of the bottom line of its bounding box and projecting a ray through it onto the ground plane. The distance to the intersection point is considered to be the distance to the robot. The near robot detector is able to detect robots up to a maximum distance of approximately 1.2 m away.

3.3 Far Robot Detection Algorithm

Just as with the near robot detection algorithm, the far robot detection algorithm relies on important assumptions in order to simplify the detection procedure and speed up processing time. These assumptions are:

1) *Field horizon*: Same as with near robot detection algorithm. The field horizon is the line in the image that best separates the field from the background.

2) *Robots intersect field horizon*: Also the same assumption from the near case. All robots are assumed to be on the field, and since the field horizon is the line that denotes the beginning of the field in the image, all robots must intersect this line.

3) *Robots appear small in image*: Since the robots are far away from the observer (greater than 1.2m away), they will not be very big in the image space. This assumption allows the algorithm to focus processing on a small window surrounding the field horizon, instead of on the entire image.

4) *Team uniform contrasts with robot*: All the robots in competition must wear their team uniform, which is either dark red or dark blue in color. The uniform colors contrast with white, the color of the AIBO robot. The algorithm takes advantage of this contrast to identify robots by their edge detection results.

3.2.2 Finding the Field Horizon

The field horizon as described earlier is the line that best separates the background from the playing field in the image. The playing field will always be green in color, and the background will most likely not be. Therefore, finding the field horizon in the image constitutes scanning the image columns from top to bottom, searching for the start of green pixels that denote the beginning of the playing field. The start of the playing field is found once four green pixels are detected sequentially down an image column. Once all columns in the image have been scanned and the starting points of the field are recorded, the highest point in the image where the start of green is recorded denotes the location of the field horizon. Only one point is necessary to determine the field horizon line, as the image has already been rotated to world coordinates, so the field horizon line will always be a horizontal line. If there are no start of field points in a given image, i.e. the field is not visible, then the field horizon line is set to be at the top of the image. Many times when the field horizon is at the top of the image, the whole image is interpreted as being an obstacle on the field, which is often true in this scenario, e.g. the robot is off the field looking away from the field, or there actually is another robot obstructing the view of the robot.

3.2.3 Detecting Obstacle Regions

Using the assumption that robots are always on the field, we can search below the field horizon for pixels that are not green (not field) and not orange (not ball) and group them to form obstacles, which will eventually become our detected robots. Each column in the image is scanned below the field horizon searching for the start of a sequence of green/orange pixels, at which point the scanned length along the column is recorded and scanning is initiated on the following column. The amount of green or orange pixels necessary in a sequence is different depending on where the sequence lies in the image. For example, if there is one green pixel immediately below the field horizon, that is enough to abort scanning along that image column, however if only one green pixel were found further down the column away from the field horizon, it wouldn't be enough to stop scanning. Generally, if three green pixels are found in sequence, the scanning is aborted, or if four green pixels are found, in sequence or not. This procedure essentially finds the length along each column of an obstacle that intersects the field horizon line. Columns with small lengths are thrown out to prevent creating obstacle regions on account of image noise. Neighboring columns with lengths a reasonable amount apart are grouped together. The lengths of all the columns within a group are averaged to find the length for that group. Using this length, along with the starting and ending column positions, a bounding box is created for each group; these bounding boxes represent the obstacles located on the field. Resulting bounding boxes that are too small or in some way inadequate, i.e. too "skinny" or too lengthy (the width and height dimensions are multiple factors apart), are removed. The final bounding boxes are considered to encompass robots and represent the output of the robot detector. An illustrative interpretation of the column scanning and obstacle grouping are shown in Figure 3.5.

3.2.1 Preprocessing the Segmented Image

Before the near robot detection algorithm can begin searching for robots within the segmented image, it must be preprocessed in order to speed up the detection process. There are two steps in the preprocessing, which are:

1) *Rescaling segmented image*: The color segmented image is scaled downwards by a factor of four along each dimension using a simple nearest neighbor approach, reducing the original 208x160 resolution image to 52x40. An example of image reduction is shown in Figure 3.4(a),(b).

2) *Rotate image to world space coordinates*: The robot contains three degrees of freedom in its neck and head, therefore when the camera is rotated the image coordinates do not correctly correlate with world coordinates. In order to ensure that the “up” direction in image space is the same as the “up” direction in world space (perpendicular to the ground plane), the segmented image is rotated by the opposite amount of the camera rotation. Correcting for the rotation of the camera allows the algorithm to remain robust to different head positions and gaze directions. An example of image rotation is shown in Figure 3.4(c),(d).

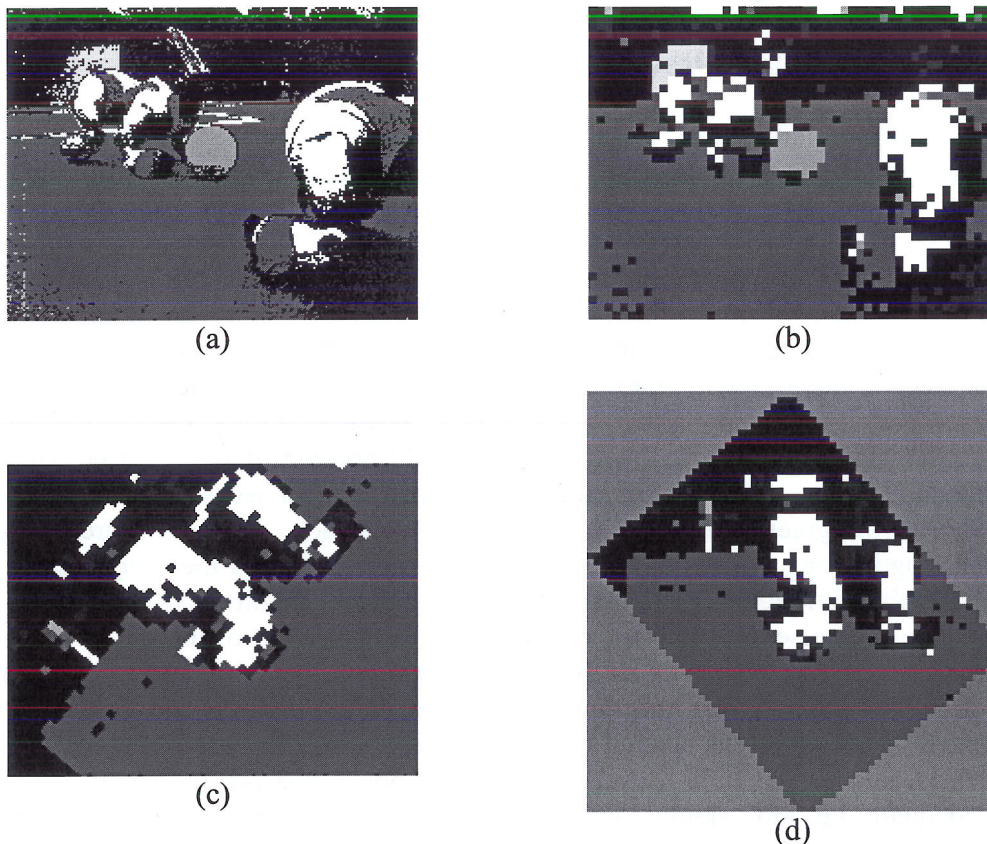


Figure 3.4: (a) Original 208x160 color segmented image from robot camera
(b) Reduced 52x40 image of same view as in (a)
(c) Reduced segmented image of vision frame captured from rotated camera (head)
(d) Reduced image corrected for rotation of camera in (c) – ready for processing

3.2 Near Robot Detection Algorithm

The near robot detection algorithm is described in detail, along with experimental results demonstrating its effectiveness, in my Senior Research Thesis [9]. A summary of the algorithm is presented here for completeness and to introduce components used by the vision algorithms discussed later in this thesis.

All the images that need to be processed come from the head of an AIBO that is standing on the field, therefore certain assumptions about the image are made to aid and speed up the detection of objects that are considered to be on the field. There are three main assumptions:

1) *Field horizon*: There exists a horizontal line that best separates the playing field from everything above it, called the field horizon. Every pixel below this horizon line belongs to either part of the field or something on the field.

2) *Objects not green*: Objects on the field, such as robots, people's legs and other things, are not green in color since the field is green and a distinction must be made between objects on the field and the field itself. Therefore, every pixel below the field horizon that is not green is part of an object that is located on the field.

3) *Objects intersect field horizon*: Objects standing on the field will always intersect the field horizon. This assumption is made because the robots to be detected will most likely always be standing, and from the point of view of another standing robot the playing field cannot be seen over the robot being viewed.

By using these assumptions the detection of robots in the image can be processed much more rapidly than if these assumptions were excluded.

There are three main steps in the robot detection algorithm, which are:

- Preprocessing the segmented image
- Finding field horizon location
- Identifying discrete objects (robots) below field horizon

An overview of the near robot detection algorithm is presented in Figure 3.3.

```
Algorithm I: FINDNEARROBOTS (image)  
  
min ← ∞  
RESCALEIMAGE(image)  
for each column col in image  
    row ← FINDSTARTOFGREEN(image, col)  
    if row < min  
        min ← row  
fieldHorizon ← min  
for each column col in image  
    objscol ← FINDOBJECTEND(image, fieldHorizon, col)  
return FORMROBOTBOUNDINGBOXES(fieldHorizon, objs)
```

Figure 3.3: Algorithm for detecting near robot locations within image frame.

One way to address the problem of robot detection is to search for red or blue blobs in the image, and if the blobs conform to some constraints, a robot is detected. In theory, this seems to be a reasonable solution, however the color of the blue uniforms is so dark that in the color segmented images they appear to be black. Since the background color is usually black, and there are a lot of shadows and segmentation noise that is also black, finding blue robots in the image with this technique is not as easy as it seems. If one were to try and correct the segmentation to emphasize the recognition of the color blue of the uniforms, then a lot of black pixels would become blue, and the problem would still remain. Also, there is more than one uniform patch on any given robot, and so determining whether or not blobs are part of the same robot or multiple robots correctly is non-trivial. Furthermore, the uniforms are only one aspect of the robots, and intuitively it seems that by using only the uniform color information, much information is ignored that would otherwise help to greatly improve the detection of robots. For these reasons we have decided not to use uniform colored blobs for robot detection, but focus rather on finding objects located on the playing field. The objects detected on the playing field are assumed to be robots, as the only objects on the field during game play, besides the ball, are robots. Figure 3.2 shows images taken from the robot's camera that are typical during game play, and constitute examples of the visual problem space.

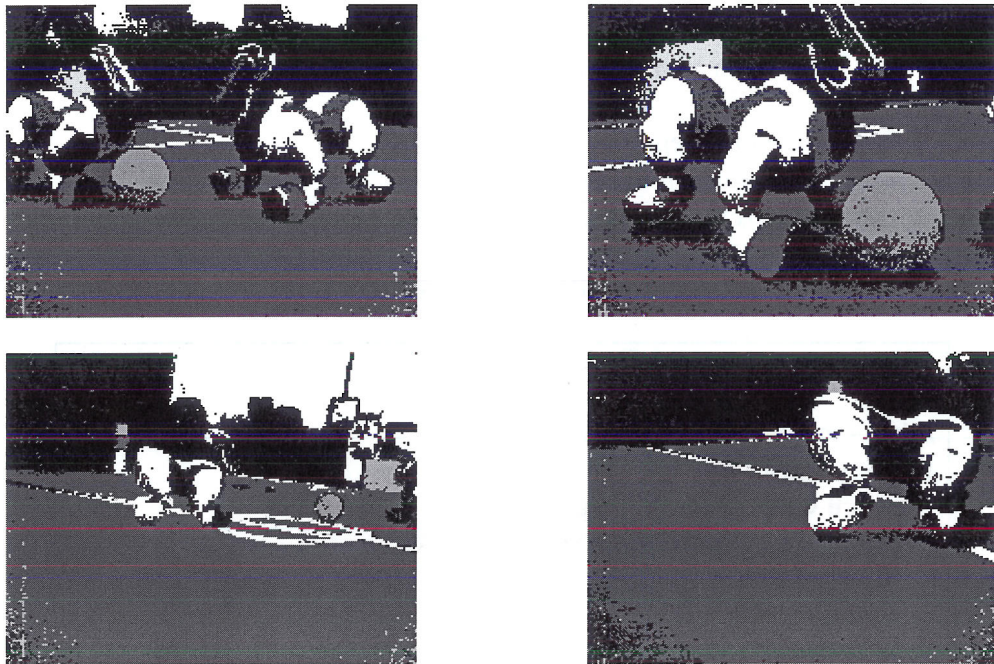


Figure 3.2: Color segmented images of vision frames taken from the robot's camera similar to those during actual game play

Chapter 3

Robot Detection

In the RoboCup soccer environment there are specific objects of interest whose positions on the field, when known by the robot, can be very helpful in making appropriate strategic decisions. For example, the robots on the field must be able to visually detect the location of the orange ball in order to move towards it in an attempt to perform some action on it, like kicking in the direction of the opponent goal or passing to a teammate. Another example are the landmarks on the sides of the playing field, whose locations relative to the robot help the robot determine where it is located on the field and how it is oriented, and at the same time, in what direction the opponent goal is located. The objects of interest that we are concerned about are the teammate and opponent robots. The detection and modeling of teammate and opponent robots on the field provides for the creation and execution of a variety of different behaviors and actions the robot can perform in response to such information. In this chapter we will describe our method of visual detection of robots and present experimental results on its effectiveness.

3.1 Visual Robot Detection Problem

The AIBO robots are white, in the shape of a small dog with four legs and a moving head. In a robot soccer game, the robots wear special "uniforms," which are colored patches of different (strange) shapes that cover parts but not all of their white body. The patches are either red or blue to denote the team but all the robots in a team are indistinguishable. Figure 3.1 shows examples of robots in their colored uniforms.

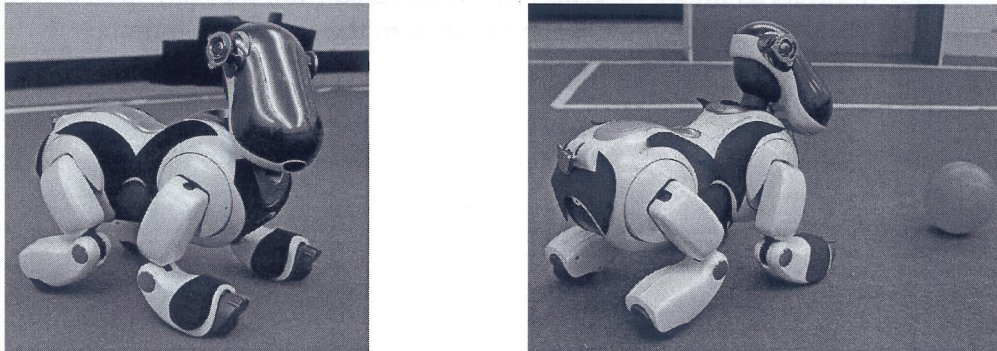


Figure 3.1: AIBO robots with RoboCup colored uniforms for both teams

to the motion module. In short, the visual processing must not only run in real-time, but run in about one third of the time it takes for a new vision frame to be recorded just so the other computation can be completed; in practice, this amounts to approximately 11 milliseconds. Furthermore, within the visual processing, many separate objects must be detected, which further limits the time allowed for any given detection method.

RoboCup provides objects of interest that are of discrete pre-defined colors, so as to simplify the task of visual detection of these objects. Our vision module is separated into both low-level and high-level processing. The low-level processing handles the conversion of the given vision frame from YUV-color space into the discrete color space of RoboCup colors, which include black, white, green, orange, yellow, cyan, pink, red, and blue. For this, the CMVision [4,5] algorithm is used. The conversion is achieved by performing a simple lookup in a color calibration table that identifies which discrete color is mapped to the given YUV color. An example conversion is shown in Figure 2.3. Although the table lookup runs very fast, the creation of this color table is extremely tedious and involves manually labeling pixels from images captured from the robot on the field under the same lighting conditions as expected during a game. The low-level processing also groups neighboring pixels of the same discrete color into color regions so as to facilitate detection methods at higher level processing. The high-level vision processing makes use of the segmented image and color regions while performing the detection methods for all the objects of interest, such as the ball, landmarks, goals, field lines, and robots.

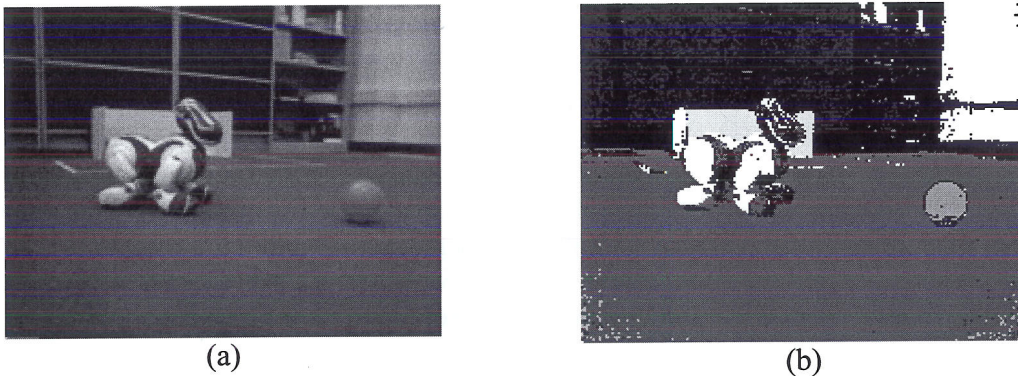


Figure 2.3: (a) RGB image of vision frame captured from the camera of an AIBO
(b) Color segmented image result from the same vision frame

sensing. The camera provides images at a resolution of 208x160 pixels and has a field of view of approximately 55°, which combined with the 180° pan of the head allows the robot to effectively scan the entire area in front of it. The robot is designed to be fully autonomous, with the ability to use its sensors to perceive the world around it and onboard computation to select and execute actions.

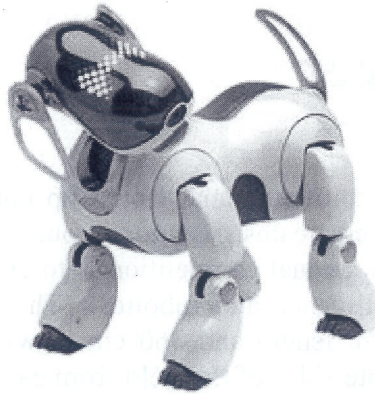


Figure 2.2: The Sony AIBO ERS-7

2.2 Overview of on board Modules

There are five modules that are responsible for the autonomous functionality of the robot: vision, localization, world model, behaviors and motion. The vision module is responsible for the perception of the environment, namely by identifying and reporting distances and bearings of field objects after analyzing the images returned by the onboard camera. The localization module estimates the location and orientation of the robot on the field by using the information on the relative distance and angle of the landmarks [6]. When no markers are visible the location of the robot is estimated based on the robot's last known position, the motions executed and the sensor readings. The world model module is responsible for gathering information and modeling the locations of field objects that are not static, such as the ball and other robots [3].

The information provided by the localization and world model modules is used by the behaviors to select the next action to execute, such as walking forward or kicking. The motion module interprets the command set by the behaviors and determines how to manipulate the joints in order to carry out the command.

2.3 Vision Module

All visual information about the robot's environment is gathered from the onboard camera located in the head of the robot. The camera outputs new vision frames at approximately 30 fps. In order to take full advantage of this onboard capability, the robot must be able to perform all necessary computation on the current vision frame before receiving the next vision frame. This computation includes not only performing all visual detection methods for the various objects on the field, but also updating localization estimates, world model information, running behaviors, and outputting motion commands

Chapter 2

RoboCup Soccer

Our domain of research is the international RoboCup competition [1], specifically the Four-Legged League [2], where teams of autonomous robots play soccer against one another. The goal behind this annual competition is to encourage and promote research in the areas of artificial intelligence and robotics. The robot game environment is a playing field that is 540 cm in length and 360 cm in width with green carpet surface. There are two goals on opposite sides of the field from each other, both measuring 80 cm wide and 30 cm tall. Two uniquely colored landmarks are positioned on the sides of the field to help the robots localize their positions on the field. The game is played with a small orange plastic ball, and each team has four players. Wireless communication between robot teammates is acceptable, however all processing must be performed onboard the robots and no human intervention is allowed. Figure 2.1 shows a model of the playing field.

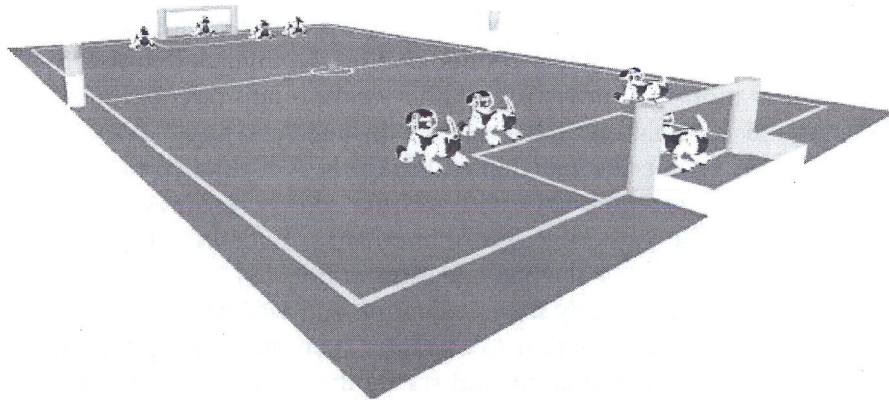


Figure 2.1: The robot soccer playing field

2.1 Robot Platform

The robot platform used in this research is the Sony AIBO ERS-7 Entertainment Robot, which is a quadruped robot designed to look like a small dog. The robot design is shown in Figure 2.2. The robot contains a total of twenty degrees of freedom with three in the head, three for each leg, and the remaining five in the tail, mouth, and ears. There is a single 64-bit 576 MHz MIPS processor on the robot which is used for all onboard computation. The robot also has 64 MB of RAM, two infrared distance sensors, a wireless wavelan card, and a CMOS camera located in its head that it uses for all visual

1.2 Overview of Chapters

The thesis is divided into two parts: vision and behaviors. Chapter 2 will provide a discussion on the RoboCup domain, AIBO platform, and the different on board modules used for robot soccer. Chapter 3 will start the vision part of the thesis, as it presents the algorithm for the visual detection of teammate and opponent robots, which is split up into two separate cases, one for near robots and another for far robots. Chapter 4 will then discuss the algorithm for the detection of the goals, which also makes a distinction between the near and far cases. Results showing the effectiveness of both algorithms will be provided, and in the case of robot vision, results from experimental tests will also be provided.

Behaviors that make use of the visual robot detection and goal detection results include shooting on the opposing goal, passing toward a visual teammate, navigation with obstacle avoidance and various other opponent awareness behaviors. Figure 1.3 shows the visual input structure of these behaviors. The first two behaviors will be discussed in detail in Chapters 5 and 6 respectively. These chapters will also present the results of experiments that have been run to test each behavior's effectiveness and performance in comparison to previous methods of the same behaviors.

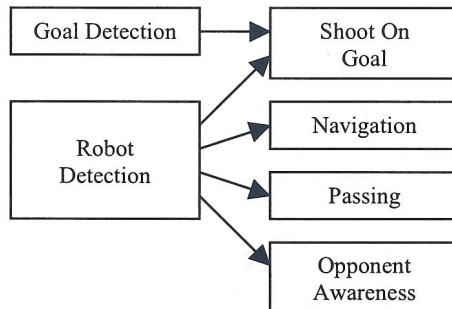


Figure 1.3: Visual input structure for behaviors seeking visual feedback

The thesis will conclude by providing a comprehensive analysis of the work accomplished as well as discussing future work and how our robot soccer team, equipped with the work presented in this thesis, has fared in recent RoboCup competitions.

confirmation of accurate positioning has been achieved. This proposed solution has been shown to dramatically reduce our team's chances of inadvertently performing an incorrect action, such as kicking the ball out of bounds, and significantly increase our team's chances of performing the correct action, such as shooting on the opposing goal and scoring.

The shoot on goal behavior can benefit greatly from the proposed input model by using visual goal detection and robot detection results as additional input. Figure 1.2 illustrates the different approaches that the shoot behavior can have based on the two input models. By using the goal detection results, the shoot behavior can enforce that the robot only shoot the ball when the goal is detected to be within the appropriate kicking range and line of sight, thus increasing the chances of good shots on goal and essentially eliminating the possibility of kicking the ball completely off target. In addition, the robot detection input allows the behavior to decide when it should try and dodge around robots that occlude direct sight of the goal, thus improving the robot's chances of detecting the opposing goal for a shooting opportunity. Hence, by modifying the shoot behavior to receive and actively pursue visual feedback, we have given it the ability to overcome any localization errors, which was not possible with the previous input model. The introduced changes have also increased the probability of successful shots on goal and significantly reduced the chances of missed opportunities due to kicking the ball out of bounds.

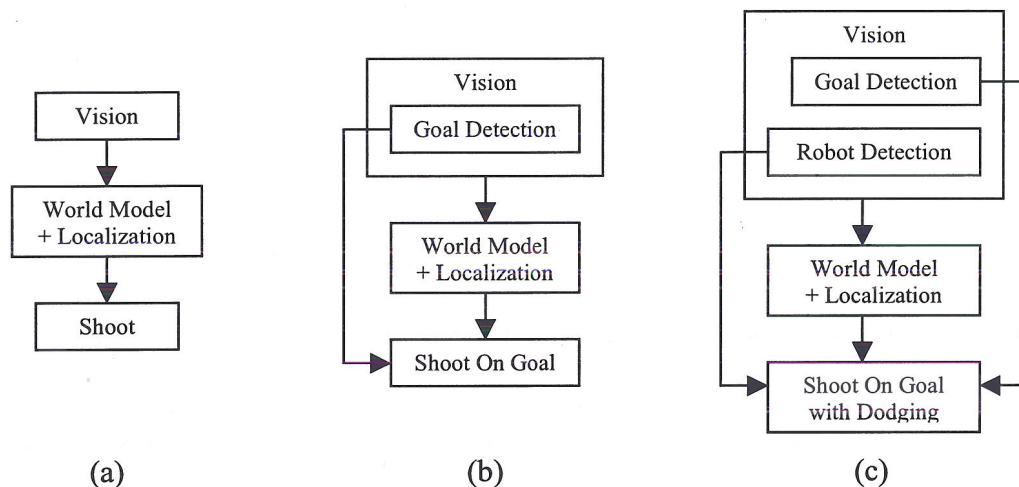


Figure 1.2: (a) The previous shoot behavior, which uses only localization estimates when shooting towards the goal. (b) Shoot behavior using proposed input model with visual goal detection results as additional input. (c) Shoot behavior with proposed input model and visual robot detection results as well as goal sightings for additional input.

1.1 Behavior Input Model

We have worked for several years on the complex problems faced by a team of soccer playing robots and the behavior input model has remained essentially the same throughout [7,8]. This model, common to other robot architectures, consists of feeding the results of the visual object detection methods into the robot's localization estimates and world model, which is the robot's state estimation of the field of play and game in general. The behaviors then use this state estimation when making decisions on what to do during the game, such as choosing the manner and direction to kick the ball. In this model there is no direct link between the vision module and behaviors, they are only linked indirectly through the world model and localization estimates. Figure 1.1a sketches this pipelined architecture. We propose a new method of behavior input in this thesis which allows vision to be a direct input into behaviors, in addition to the localization and world model estimates, so that a robot can make more informed decisions during game play. Figure 1.1b shows a flow chart of the proposed model for comparison.

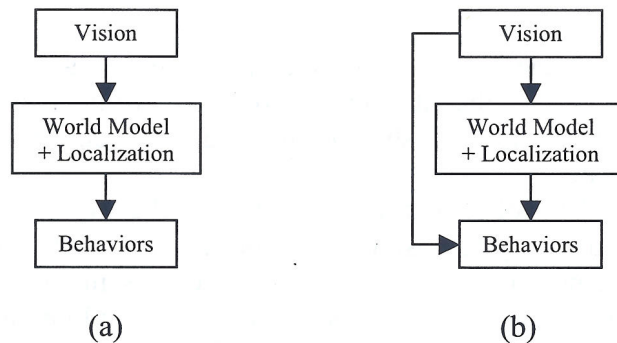


Figure 1.1: (a) The pipelined behavior input model
(b) The sensor-added pipelined behavior input model

In this thesis, we show that the pipelined method for behavior input is brittle to localization errors, and is not as robust as necessary for the most competitive robot soccer matches. When playing against tough opponents we notice that scoring chances, which are seldom against strong teams, can be lost because the attacking robot blindly trusts its own localization estimate and when shooting towards the goal, inadvertently kicks the ball out of bounds due to slight errors in the estimate. The slight errors are expected due to the fact that against tougher teams, the robots fight for the ball more, by pushing up against other robots, and are occluded by opponent robots more often, altogether leading to incorrect odometry estimates and fewer sensor updates to localization.

Our new sensor-added pipelined behavior architecture, as presented in this thesis, solves this problem by allowing the robot to receive and actively pursue feedback from vision to re-affirm its position estimate when performing crucial behaviors, such as when shooting towards the opponent goal, and to only actuate on the ball once visual

Chapter 1

Introduction

Robot soccer presents a variety of challenging problems for an autonomous mobile robot, such as the AIBO, especially as no accurate global models of the environment are provided from an external source such as an overhead camera. Problems common to robot soccer include: real-time visual perception of objects, world modeling, localization, multi-agent coordination, and strategy assessment. Our AIBO robot soccer team consists of four robots, each with their own unique characteristics that define their role in the team. The roles of our current team are defined as such:

- *Attacker*: The robot that goes for the ball. It is the duty of the attacker to gain possession of the ball as quickly as possible and to score on the opposing team's goal. Once the attacker has possession of the ball it will either choose to move the ball up field, shoot on the opponent goal (if within range), or pass to a teammate.
- *Supporter*: The robot that supports the attacker offensively. The supporter will continually try to position itself on the field such that it can provide the most assistance to the attacker. This includes moving to a position for a potential pass, being in a position to easily retrieve loose balls, being ready for shot deflections, and generally staying clear of possible shots on goal by the attacker.
- *Defender*: The robot that guards the defensive zone, which generally includes the half of the field where the team's goal is located. The defender will wait patiently until the ball enters the defensive zone, at which point it will try to clear the ball up field for the offensive players to attack. It also stays clear of possible ball clearing by the goalie.
- *Goalie*: The robot that guards the goal box area directly in front of the team's goal. The goalie represents the last line of defense on the team and it will try to position itself to best block shots on goal from the opposing team, and to clear the ball anytime it enters the goal area.

This thesis will focus primarily on the role of the attacker, which we present in terms of its perception and behaviors. However, every robot role in the team benefits from the advances made in the attacker, as they can incorporate attacker tactics into their own behaviors with little or no modifications.

5	Shoot On Goal Behavior	35
5.1	Shoot Behavior Approach	35
5.2	Shooting Algorithm	36
5.3	Shoot Behavior Comparison Experiment	39
5.4	Robustness Experiments	41
5.4.1	Experiment 1	41
5.4.2	Experiment 2	42
5.4.3	Experiment 3	43
5.4.4	Experimental Results	43
6	Passing to Teammates	45
6.1	Passing Behavior Approach	45
6.2	Visual Robot Tracking	46
6.2.1	Tracking Algorithm	47
6.2.2	Sample Tracking Results	48
6.3	Passing Algorithm	50
6.4	Pass Behavior Experiments	51
6.5	Results	54
7	Conclusion	57
7.1	Contributions	57
7.2	Future Work	58
7.2.1	Passing Behavior	58
7.2.2	Pass Receiving Behavior	59
7.3	Competition Results	59
	Bibliography	62

Contents

1	Introduction	1
1.1	Behavior Input Model	2
1.2	Overview of Chapters	4
2	RoboCup Soccer	5
2.1	Robot Platform	5
2.2	Overview of on board Modules	6
2.3	Vision Module	6
3	Robot Detection	8
3.1	Visual Robot Detection Problem	8
3.2	Near Robot Detection Algorithm	10
3.2.1	Preprocessing the Segmented Image	11
3.2.2	Finding the Field Horizon	12
3.2.3	Detecting Obstacle Regions	12
3.2.4	Detecting Obstacle Regions	13
3.3	Far Robot Detection Algorithm	13
3.3.1	Far Robot Detection Overview	14
3.3.2	Edge Detection	15
3.3.3	Finding End of Field Along Column	16
3.3.4	Light-to-Dark Transitions	16
3.3.5	Robot Boxes	18
3.3.6	Combine Boxes of Same Physical Robot	19
3.4	Visual Far Robot Detection Results	21
3.5	Far and Near Robot Detection Overlap	22
3.6	Far Robot Team Color Estimation	23
3.7	Far Robot Distance Estimation	23
3.8	Experimental Results	24
3.8.1	Experiment 1	24
3.8.2	Experiment 2	26
4	Goal Detection	29
4.1	Goal Detection Algorithm	29
4.1.1	Goal Region Analysis	30
4.1.2	Goal Post Detection	31
4.1.3	Post Verification and Distance Estimation	32
4.2	Visual Detection Results	33

CHAPTER 1

The first chapter of the book discusses the basic concepts of the theory of groups. It begins with the definition of a group and the properties of groups. The chapter then discusses the subgroups of a group and the quotient groups. The chapter concludes with a discussion of the isomorphism theorem for groups.

Acknowledgments

I would like to thank my advisor, Manuela Veloso, for giving me the opportunity to work and perform research in her lab every year I've been at Carnegie Mellon. Manuela, thanks for everything. I would also like to thank all the lab members, Sonia Chernova, Douglas Vail, Colin McMillen, Scott Lenser, James Bruce, and Paul Rybski, who were always right there whenever I needed help throughout the years. To my team members, Somchaya Liemhetcharat, Mike Phillips, and Greg Delmar, thanks for all your hard work and dedication, we did it! Finally, I would like to thank my family for all their love and support; this is for you.

Abstract

Visual object recognition, world state estimation, and localization are challenging problems for an autonomous mobile robot. The problems are especially difficult on the AIBO robot platform, which must rely only on local sensor information retrieved from a single monocular camera with limited viewing angle. In the RoboCup domain, specifically the Four-Legged League, teams of autonomous AIBO robots play soccer against one another. This thesis focuses on the creation of intelligent robot soccer behaviors that rely on direct visual input to carry out critical tasks where previous localization-based behaviors have lacked in effectiveness and efficiency. We present novel algorithms for the real-time visual detection of the goals and teammate and opponent robots on the playing field, the latter of which is capable of detection ranges up to three times greater than in a previous implementation, aimed at improving overall team game play. The thesis discusses the ‘shoot on goal’ behavior and the ‘pass to teammate’ behavior, which were both modified to make use of direct visual input from the new vision algorithms for improved performance. Experimental results comparing the effectiveness of the visual input-based behaviors against their traditional localization-based implementations are also provided. The thesis concludes with a discussion of future work and how the robot team, equipped with the algorithms presented, has fared during actual games at the international RoboCup competition.

Keywords: robot soccer, vision, object detection, autonomous behaviors, AIBO, RoboCup, mobile robots

Real-Time Visual Input for Intelligent Robot Soccer Behaviors

Juan P. Fasola

August 17, 2007

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Thesis Committee:
Manuela Veloso, Chair
Paul E. Rybski

Submitted in partial fulfillment of the requirements
for the degree of Master of Science

Copyright © 2007 Juan P. Fasola