The UT Austin Villa 2005 RoboCup Four-Legged Team

Peter Stone, Kurt Dresner, Peggy Fidelman,
Nate Kohl, Gregory Kuhlmann,
Mohan Sridharan, Daniel Stronger

Department of Computer Sciences
The University of Texas at Austin
1 University Station C0500
Austin, Texas 78712-1188
{pstone,kdresner,peggy,nate,
kuhlmann,smohan,stronger}@cs.utexas.edu
`http://www.cs.utexas.edu/~AustinVilla`

Technical Report UT-AI-TR-05-325

November 11, 2005

Abstract

The UT Austin Villa Four-Legged Team for RoboCup 2005 was a third-time entry in the ongoing series of RoboCup legged league competitions. The team development began in mid-January of 2003 without any prior familiarity with the Aibos. After entering a fairly non-competitive team in RoboCup 2003, the team made several important advances. By the July 2004 competition that took place in Lisbon, Portugal, it was one of the top few teams. After those first two years of intense development, the team's third year was devoted more to spinoff research than to development related to the competition. Building off of the team's previous two technical reports [12, 13], this report details the changes made to the team between RoboCup 2004 and RoboCup 2005 in Osaka. Taken together, this and the previous technical reports provide the history and details of a relatively young RoboCup team that has quickly grown from a nascent project to an ongoing source of diverse and plentiful research results while becoming a highly-seeded team in competitions.

Contents

1 Introduction

RoboCup, or the Robot Soccer World Cup, is an international research initiative designed to advance the fields of robotics and artificial intelligence, using the game of soccer as a substrate challenge domain. The long-term goal of RoboCup is, by the year 2050, to build a team of 11 humanoid robot soccer players that can beat the best human soccer team on a real soccer field [4].

RoboCup is organized into several leagues, including a computer simulation league and two leagues that use wheeled robots. This technical report concerns the development of a team for the Sony four-legged league[1] in which all competitors use identical Sony Aibo ERS-210A and/or ERS-7 robots and the Open-R software development kit.[2]

Since all teams use the same commercial robots, the four-legged league is essentially a software competition. In this report, we detail the development of our team, UT Austin Villa,[3] from the Department of Computer Sciences at the University of Texas at Austin.

For this report, we assume familiarity with the robots' specifications and the rules of the RoboCup games. For full details, see the legged league and Open-R sites footnoted above. Here we describe both our development process and the technical details of its end result, the UT Austin Villa team. In conjunction with our previous technical reports [12, 13] this paper provides full documentation of the algorithms behind our approach. In particular, this report is *not* intended to stand alone; a full understanding requires familiarity with our previous technical reports. Taken together, they provide the history and details of a relatively young RoboCup team that has quickly grown from a nascent project to an ongoing source of diverse and plentiful research results while becoming a highly-seeded team in competitions.

Our team development began in mid-January of 2003, without any prior familiarity with the Aibos. After entering a fairly non-competitive team in RoboCup 2003, the team made several important advances. By the July 2004 competition in Lisbon, Portugal, it was one of the top few teams. After those first two years of intense development, the team's third year was devoted more to spinoff research than to development related to the competition. In this document we fully describe the changes to our 2004 code base, which we previously documented in full detail [13]. As was the case in 2003 and 2004, all of our code is our own, with no parts borrowed from other teams' implementations. We include all of our advances as of RoboCup 2005 in July.

The remainder of the report is organized as follows. Section 2 focuses on the changes that were motivated mainly by the change in the field specifications, including those pertaining to vision, localization, the goalie, and the addition of a dribbling behavior. Sections 3 and 4 detail the changes to our communication infrastructure and strategy. Sections 5 and 6 focus on the changes to the teams locomotion and overall behavior strategies. Section 7 summarizes our recent research results. Section 8 presents our results in the competitions and Section 9 concludes.

2 Adjusting to the new field

Many of the changes were driven by the league's decision to change the playing field used at the competition. For the last few years, games were played on a $2.9m \times 4.4m$ field with $10cm$ white borders all around the field. This year the field size was increased to $4.0m \times 6.0m$ with the actual playing area being $3.6m \times 5.4m$. The major change this year was to remove the borders around the field and replace it with white lines similar to those on the field. In addition, the sizes of the goals were increased ($80cm$ wide instead of $60cm$) and the beacons were moved from the corners of the field towards the center. See Figure 1 for the new field arrangement.

These changes were made primarily to encourage better ball control (the ball cannot be pushed all the way down the field by simply pushing it along the borders) and multi-agent behavior in the form of passing (there is more space available on the field for robots to position themselves strategically). The beacon

[1] `http://www.tzi.de/4legged/`
[2] `http://openr.aibo.com/`
[3] `http://www.cs.utexas.edu/~AustinVilla`

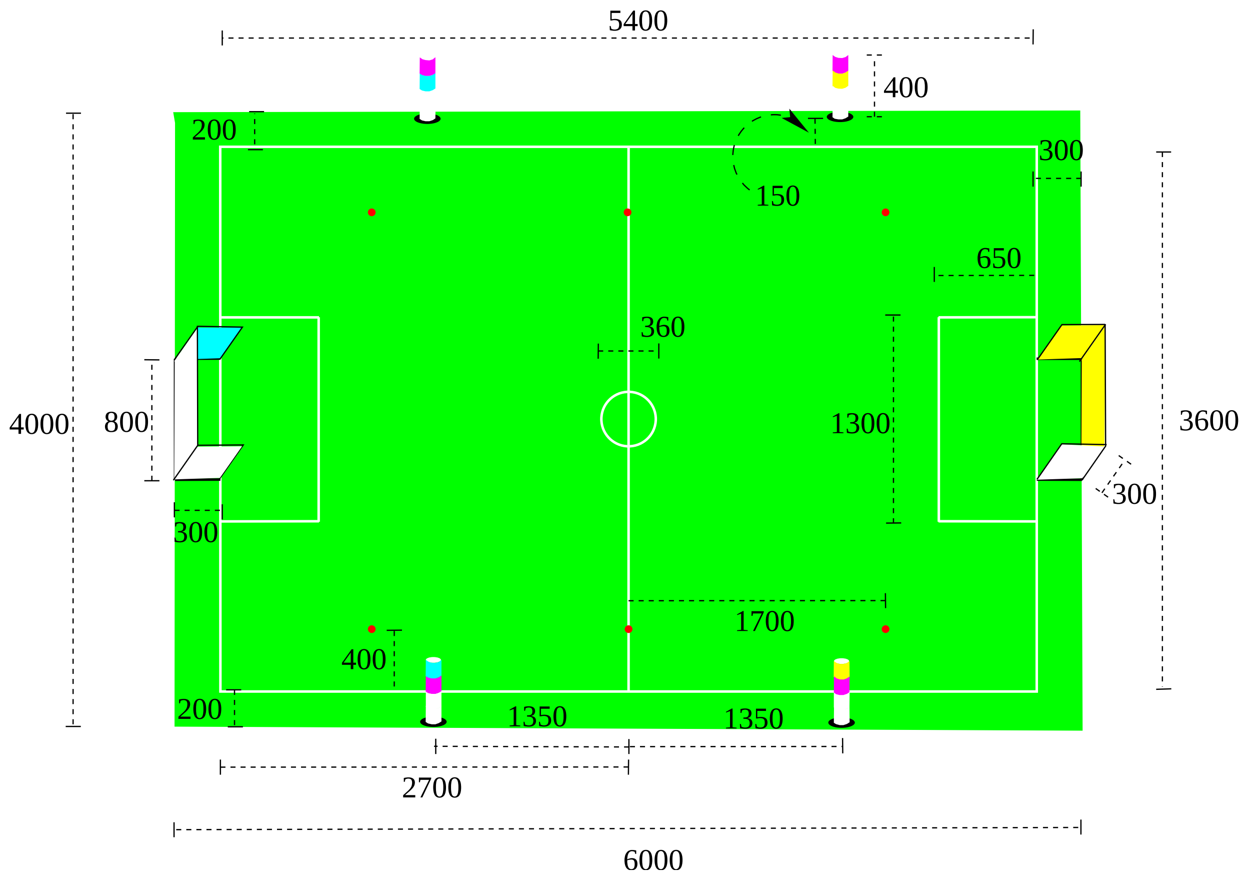

Figure 1: An overhead view of the new field.

positions were changed so that they were not all clumped at either end of the field and the robot does have some visible beacons for localization when it is close to the center of the field.

2.1 Vision

Changes in our vision code focussed mainly on dealing with the modified field arrangement — detection thresholds for the objects such as the goals had to be changed to suit the new sizes. Also, given the larger field, the ball had to be seen from a larger distance without coming up with false positives due to any small orange region.

Another main challenge involved finding the lines on the field robustly. Given that the field borders had been removed, line information assumed added importance because the robot was severely penalized for either walking off the field and/or kicking the ball off the field. This involved changing thresholds and improving the filters incorporated in the process such that the lines are detected without any false positives. The use of lines and line intersections in localization has been documented in the previous technical reports [12, 13].

As a result of the field changes we also had to include several predicates to ensure that the robot would be robust to its surroundings. Since the field borders had been removed and the foot-high white wall around the field no longer existed, the robot could see other objects at field-level outside the field. During competitions, this generally included objects (or spectators' clothing) that were of the same (or similar) color as the objects on the field. The robot could, for example, try to walk off the field because of an orange object on the ground outside the field. The check for additional predicates helped alleviate this problem.

Other changes in vision included a better representation for the ball and the opponents seen by the robot, as compared to our previous representation [13]. We used a similar Gaussian representation for both the ball and the opponents, coupled with a Kalman Filter-like [2] update.

For the ball, the robot maintained an estimate of its position relative to itself. Initially the robot has no idea of the ball position. When the robot sees the ball, the relative distance and angle to the ball (as calculated from the visual data) is used to arrive at a Gaussian representation for the ball. The mean of this Gaussian is the position of the ball relative to the robot while the variance is based on the probability of the ball location — higher probability implies lower variance. The probability estimate in turn depends on the probability of the ball distance and angle estimates (see our previous technical report [13] for detailed information).

If the robot does not see a ball in the subsequent frames, the current ball knowledge is modified to account for the robot motion (since the ball position is robot-relative, it has to be suitably modified when the robot moves) and is then degraded, i.e., the variance is increased. This increase in variance is representative of the corresponding decrease in the certainty of the ball position. Once the variance increases beyond an experimentally determined threshold, the robot stops trusting the ball estimate completely.

But if the robot sees a ball in the subsequent frames, it creates a new Gaussian based on the current estimate. Then it modifies the previous estimate to compensate for the motion (if any) and merges it with the current estimate. This merging of Gaussians is weighted by the corresponding variances so that the position corresponding to the merged Gaussian estimate is closer to the estimate with the lower variance. This representation helped keep track of the ball position without having to actually keep looking at the ball. The robot could then shift its focus temporarily to look around and localize but still *remember* the ball location relative to itself. Opponent position estimates are maintained similarly.

To model the ball motion, we added a velocity-based update to the ball model. The robot stores the ball positions over a few frames and after compensating for its motion, uses them to determine the ball velocity and motion direction. This helps when the robot sees the ball move in a particular direction but is not able to continually keep track of the ball, because the velocity based update provides the robot with a ball location where it focuses when it is time to look for the ball again. This helps develop behavior where the robot can look away from the ball but still have a good idea of the ball location; see the section on dribbling (Section 2.4) for an example.

2.2 Localization

In this year's team, we continue to use the Monte Carlo Localization approach from previous years. [12, 13] Our enhancements to basic MCL include the maintenance of landmark histories to create more opportunities for reseeding, a beacon distance correction function found through polynomial regression to allow for distance-based probability updates, and also an extended motion model to increase localization accuracy when navigating to a point. These enhancements and their benefits to accuracy, stability, and recovery are discussed in detail in [7].

It was necessary for us to make some additional changes to localization because of the new field. The change in the field's dimensions required us to transform coordinates from the old field's coordinate system. We felt that in the process of transforming the coordinates, it would be a good idea to move to a team-centric coordinate system. In the new system, $(0, 0)$ is at the center of the field. Positive Y is in the direction of the opponent goal, and positive X is to the right when facing the opponent goal. This team-centric coordinate system allows us to create world state predicates without treating the two teams as separate cases.

Another major change to the field, the removal of the walls, also required a small change to localization. Without walls, the robot may occasionally walk outside of the field lines. We changed the boundary in which the robot's particles may move to permit them to go outside of the field but not off of the green carpeted area. With this enhancement, the robots are aware when they are off of the field. They will not try to chase the ball off of the carpeted area, which would otherwise result in a penalty.

Finally, the new arrangement of beacons around the field allows for the two possible solutions found from triangulation of two landmarks both to be on the field. Previously, the incorrect estimate could be trivially thrown out because it was always outside the field boundaries. On the new field, we actually have to use the observed angles to the landmarks to determine the correct solution. The correct estimate will have roughly the same angle difference between the two landmarks as the difference between the two observed angles. The incorrect estimate will be closer to 2π minus the observed angle difference. It becomes difficult to disambiguate the points when the angle difference is close to π, but at that point, the two position estimates are very close together, so the cost of choosing the wrong point is small.

Altogether, the field changes did not have a very large effect on our localization performance. On the larger field, the beacons were sometimes farther away, making it more difficult to estimate their distances. However, the increased error was roughly offset by the increased frequency of observing beacons due to the change in beacon placement from the corners of the field to the third-way points along the sidelines. Without beacons in the corners, however, localization could sometimes be inaccurate near the goal lines. We were able to compensate for this problem, by changing the robot's behavior near the goal, as will be discussed in Section 6.

2.3 Detecting collisions with the goal

If the goalie has left its normal position in the center of the goal and needs to return there, it typically does so with its back to its destination (to give it a better chance of seeing the ball in play). However, due to the removal this year of the wall bordering the field, this leaves open the possibility that the robot may get stuck on the side of the goal and never be able to reach its destination (see Figure 2).

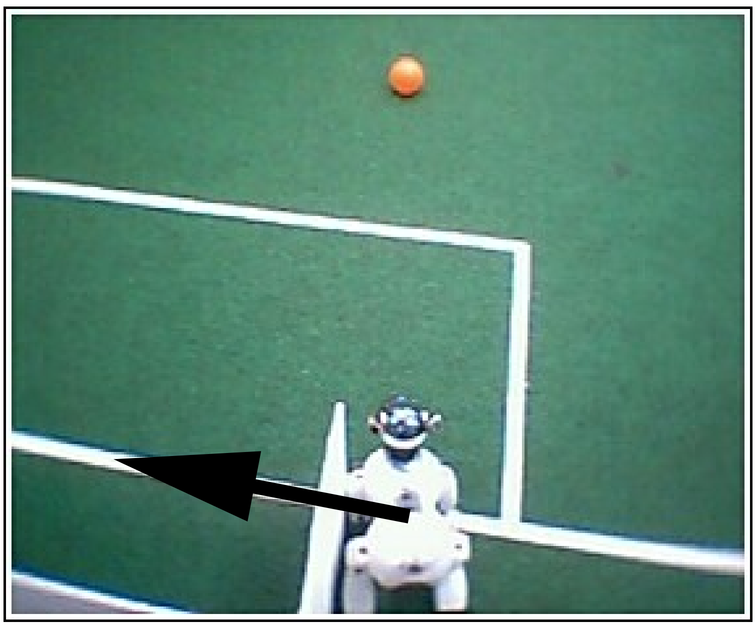

Figure 2: An example of how the goalie can get stuck on the goal. Here the goalie is trying to move in the direction of the arrow, but is running into the side of the goal. It cannot see directly that it is stuck, because it must pay attention to the ball and the rest of the field.

We implemented a straightforward localization-based solution to this problem, since the goalie can most likely see other landmarks while it is facing out into the field this way. Because it can see these landmarks, it will realize it has not moved even though its odometry is likely telling it it *has* moved (since it is actively walking sideways, trying to get back to its post).

If the goalie believes itself to be at least 300 millimeters from its post and less than 150 millimeters from the back end of the field (where the white line is), and if it also believes it has moved less than 50 millimeters in the last 4 seconds, it will conclude that it is stuck on the edge of the goal. It will then attempt to unstick itself by trying to move to a point 100 millimeters inside of the closest outside corner of the goalie box, and then it will resume trying to get back to its post in the middle of the goal. This motion and the criteria that determine when it executes are summarized in Figure 3.

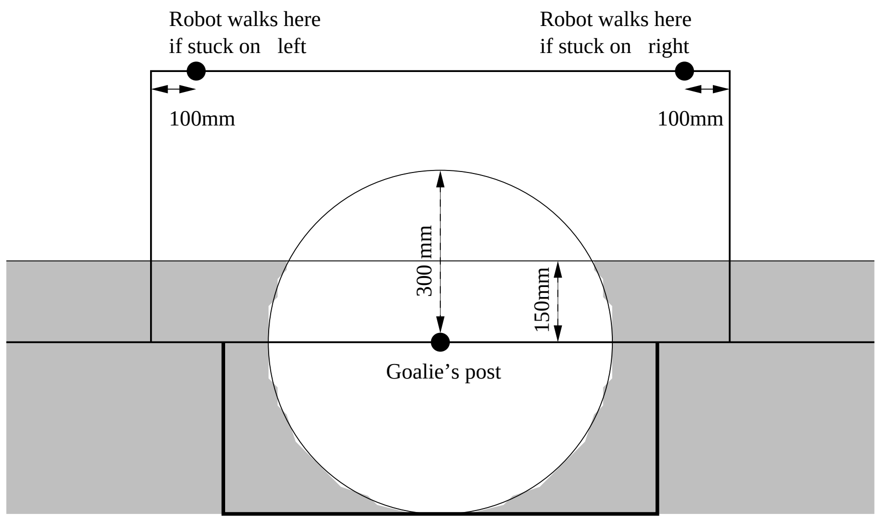

Figure 3: Diagram of when and how a goalie will attempt to free itself from being caught on the outside of the goal. If the goalie believes itself to be in the shaded region and believes that it has moved less than 50 mm in the last 4 seconds, it will conclude it is stuck on the side of the goal. It will then walk toward one of the labeled points on the goalie box, determined by which side it believes it is on, and then will move back toward its post.

2.4 Dribbling

One of the changes to our 2005 code was the addition of a "dribble" behavior. This change was largely motivated by the removal of the walls from the edge of the field. Our team's behavior from previous years focused primarily on getting to the ball quickly and moving it down the field as far as possible with each possession. This strategy made sense on an enclosed field, where the penalty for missed passes and poorly-aimed kicks was negligible. With the removal of the walls from the edge of the field and the introduction of penalties for kicking the ball out of bounds, this "grab-and-kick" strategy became less effective. A more controlled behavior that moved the ball in smaller increments was implemented to address this issue.

The initial version of the dribble dealt with three objects on the field:

1. The robot in possession of the ball
2. The ball
3. The target object, towards which the robot should dribble

The basic idea was that as the robot approached the ball, it should align itself such that the ball was directly between the robot and the target. This alignment was to occur at a pace proportional to the distance between the ball and the robot. For example, if the robot was very close to the ball it would circle around the ball without getting much closer to it, until it had achieved the proper alignment. If the robot was far away from the ball, it could focus more on getting closer to the ball and less on getting the right angle to the ball. Once the robot was close enough and lined up properly, it would walk directly into the ball, knocking it forwards a small amount. After this dribble, if the ball was still aligned between the robot and the target, the robot would continue forward. If the ball was out of alignment, the robot would switch back to the previous get-aligned behavior.

The problem with this initial version of the dribble was that the robot often spent too much time stutter-stepping to get the ball aligned between it and the target. Noise in the Aibos actuators and in the estimation of the position of the ball prevented the robot from believing that it was properly aligned. This hesitation slowed the robot's approach to the ball, often costing us possessions.

To address this hesitation, we changed the dribble behavior to more resemble our previous offensive behavior, which focused on getting to the ball quickly. As with our old behavior, the robot would approach the ball single-mindedly, grab the ball with its chin, and turn to face the target direction. However, instead of kicking at this point, our dribbling robots released the ball and walked forward. Just as with our initial dribble, this movement would knock the ball forwards a small amount. The robot would then approach the ball and repeat this grab-turn-and-dribble behavior again.

As with many aspects of our Aibo code, we found there to be a trade-off between staring at the ball and localizing. With the dribble, we found this to be especially problematic, since the robots do not have a lot of time to look around for beacons when the ball is right in front of them. Being able to localize while dribbling called for a robust model of the ball and the ability to localize quickly and effectively.

With our Gaussian ball position and velocity model (Section 2.1), the robot does have the ability to look away from the ball (to localize) while maintaining a good idea of the ball location. But, when the robot is dribbling the ball, it is actually moving with the ball and is no longer able to see it (since it is using its head to scan around). In this situation, the robot's relative ball knowledge is not accurate because it assumes that the ball is still at its last known position. We addressed this problem by adding a special case to the velocity-based update to the ball model; the dribbling action is equivalent to a ball motion and is considered similarly. When the robot moves, the ball estimate is subjected to a velocity update using the robot's velocity and direction of motion. We determined that this lets the robot retain an accurate notion of the ball location when the robot is dribbling the ball or when it is turning with the ball tucked under its chin.

We were then able to include the dribble action in the behavior (Section 6). The robot could dribble the ball and while doing so could take some time to localize itself by scanning around with its head.

In addition to maintaining its own ball estimate the robot also took into account the estimates provided by its teammates. We describe this process in Section 4.

3 Communication Infrastructure

This year's competition was the first in which teams no longer used the TCPGateway mechanism to communicate between robots. As a result of this, we needed to handle our own low-level communication. The main task was to develop a low-level mechanism that could use the same high-level interfaces established in previous years.

3.1 TCP vs. UDP

One of our primary decisions was to decide between using the Transmission Control Protocol(TCP) and the User Datagram Protocol(UDP). While TCP is more reliable, we determined that the additional overhead associated with using TCP was not worth the reliability benefits it bestowed. Meanwhile, a convenient feature of UDP is the ability to broadcast without assembling and sending multiple messages.

Rather than creating separate mechanisms for broadcast and direct robot-to-robot communication, we decided, at the low level, to just send every message to every robot. To enable direct robot-to-robot communication, we simply attached a header to each message indicating which robots were the desired recipients. In the end, most of the messages sent by our robots are intended to be broadcast anyway.

3.2 Varying Packet Size

In the varying environments in which our robots play, we have noticed that sometimes large packets are consistently dropped. In these particular environments, we have found that shrinking our packets and breaking up data into multiple messages often allows us to work around this problem. However, breaking up messages like this incurs an overhead that we would like to avoid. To cope with this, we designed our system to have a variable packet size. This introduced several new challenges — messages must be broken up, sent, and then reconstructed on the other side. If a message is reconstructed improperly (for example if one part is missing), the robot that attempts to interpret the data according to the understood protocol could experience a fatal segmentation fault.

Verifying data was accomplished by a simple checksumming mechanism. The data was broken up into a sequence of 4-byte chunks. Each chunk was interpreted as an integer, and then the integers were added and the sum was placed into a 4-byte integer (overflows were allowed). This checksum was prepended as part of a custom header to each packet.

Detecting dropped packets is also very important. If a packet was missed, even attempting the checksumming procedure presents difficulties. In order to solve this, we included packet numbers in each header. The first packet would be given a packet number of $1000000 + n$ where n is one less than the total number of packets that would be sent. The next packet would have a number of $n - 1$, and then each subsequent packet would have a packet number that was one less. This had several major advantages. First, we could always detect the first packet in a sequence of packets and know how many subsequent packets to expect. Second, we could always determine if a packet is the last packet (if its packet number is 0) in a sequence and get ready to receive a new message starting with the next packet. This system allowed us to reliably detect (although not reconstruct) missed packets. Due to our communication strategy design (Section 4), it is not critical that we receive every message, but rather that we have the most up-to-date information available. Thus we determined it was more worth our while to simply ignore a corrupted or incomplete message and wait for the next one than to attempt to reconstruct the message. If the network suffers a period of instability or congestion, we'd rather just ignore the information sent during that period and pick up immediately when the problem subsides — significantly delayed information is only slightly better than no information at all.

4 Communication Strategy

A single robot cannot always keep track of the ball because it cannot keep the entire field in view all the time. One of the major goals of multi-agent platforms is the effective sharing of information between agents so that each individual agent's resources are used optimally. The robot must use the communicated information from its teammates to make informed planning decisions. However, to use this communicated information intelligently, the robot must decide which estimates to use when.

Each robot communicates its observations to its teammates. This includes its pose, its current estimate of the position of the ball and the opponents, and other state information. Each robot broadcasts its time-stamped state information to its teammates. The conversion of the robot-relative observations to global positions is based on the robot's own position and orientation on the field; the global position uncertainties are therefore a function of the relative observation uncertainties and the uncertainty in the robot's own position. A robot communicates information only if it is sufficiently certain of its own position estimate and even then, it only communicates the observations that it is reasonably sure of.

Each robot needs to combine the estimates communicated by its teammates. When merging the ball estimates from the teammates with its own estimate, the robot primarily trusts what it sees over what is communicated. The teammates' estimates of the ball are considered only when the robot does not have a good estimate of its own. In the case where the certainty of its own ball position estimate is low, it merges the teammates' estimates and uses the result to decide the direction in which it should start searching for the ball. Without the communicated ball information, the robot typically spends a lot of time recovering when it loses sight of the ball, especially after sudden ball movements such as kicks. But when the robots incorporate communicated ball information, if one robot sees the ball, its teammates are able to walk in the most probable direction to recover the ball's position quickly.

We found that it is better for the robot to know the likely locations on the field where it might be obstructed. Therefore, in our case, robots always merge communicated opponent estimates. Once again, the estimates with higher certainty are given proportionately more importance. This information is used mainly to avoid the opponents. We observed that using the merged opponent estimates provides a significant improvement in the robot's behavior. For example, when the robot is close to an opponent (even one it cannot see), it is able to determine that it is necessary to clear the ball quickly.

The state information and knowledge of the ball and opponent positions are collectively used in the role-assignment and role-switching processes that are not much different from that used by us in previous years (see [13] for complete details).

5 Walking

Other than an increase in the flexibility of foot loci, our walking machinery is the same as in previous years[13].

In the past, foot loci have been half ellipses, but this year we changed the machinery to allow for loci that are any fraction of an ellipse. By re-learning the forward gait for this new parameterization using the method of Kohl and Stone[6], we arrived at the walk used in the 2005 competitions, in which the foot loci are approximately 55% of an ellipse and the speed in our lab is 390 mm/sec. (For comparison, the speed of our fastest forward walk with half-elliptical loci was 335 mm/sec in our lab.)

6 Behavior

The changes in the behavior module this year were mainly made to tackle the field changes and to also have the robots coordinate better as a team. In addition, we improved the extent to which the actions in the roles such as the defender are well-defined and the transitions between roles are smooth, primarily by creating an an efficient representation for sharing information.

One main field change made this year was the removal of the field borders, which dramatically increased the chances of the ball rolling off the field, especially when the ball is close to the field borders. In addition to the normal kicking behavior, we also added the dribbling behavior (Section 2.4) which sometimes causes the ball to be pushed around by the robot even when it does not see it. In addition to our ball representation scheme that ensures accurate ball knowledge even if the ball is occluded for a few seconds, we had to make behavioral changes to increase the likelihood that the ball stays on the field when it is moved or kicked.

In the dribbling phase, we had the robot scan around while moving the ball so that it can localize and move the ball accurately. The robot generally dribbles the ball towards the opponent's goal, while trying to avoid opponents. But, dribbling is not initiated in certain specific sections of the field; it is not done when the robot is in the defensive third or is somewhere close to the offensive corners of the field. In the defensive third, the robot tries to clear the ball rather than try to dribble it. While doing so it kicks the ball into the field, towards the closer goal edge. When the robot is in one of the corners of the field, the direction of the robot's kick must be especially precise to get the ball into the goal. Missed shots from this position can often result in the ball going off the field. To overcome this problem, the robot moves the ball away from the end-line such that a larger section of the goal is open so that a kick will be more robust to direction uncertainty. To do so, the robot just grabs the ball and sidesteps away from the end line.

In addition to shooting towards the goal, the robot also performs reactive correction to account for the the errors in localization. Since overshooting the edges of the goal can lead to the ball going off the field and being replaced at an unfavorable position, the robot tries to correct its kick direction reactively. When the robot gets really close to the goal, it uses the visual input and while trying to align in the direction of the opponent's goal, it also tries to shoot at the largest visible goal region, even if that does not correspond exactly with the direction suggested by localization. Doing so also helps shoot around the goal-keeper who is trying to guard the goal.

Until last year, the lack of an efficient ball representation system made it difficult to effectively coordinate the robots on the team, especially those that did not have a good knowledge of the ball position. But with the current ball representation (Section 2.1) and communication (Section 4) most of the robots in the team have an accurate knowledge of the ball location, even when they are unable to actually see the ball. The robot can now dribble without having to continuously keep the ball in focus. In addition, it can perform behaviors such as *active localization* which help achieve a trade-off between keeping the ball in focus and keeping the robot well-localized. Here, the robot occasionally shifts its focus from the ball to actively look for landmarks to improve its localization estimate. We decide when to actively localize based on the uncertainty in the localization estimate. If the localization certainty falls below a threshold, the robot uses its current pose estimate and known geometry of the world to predict the relative positions of the various landmarks. Then, it uses this knowledge to *plan* the motion of its head (pan, tilt) that should allow it to see the closest markers. Because performing such an action could cause the robot to lose track of the ball's position, especially when the ball is close by, active localization is performed only when the robot is a sufficient distance (800mm) away from the ball. We have found that such active localization noticeably decreases the robots' overall pose uncertainty.

A better knowledge of the ball also helped better define the individual roles and the transitions between the roles. For example, the defender positions itself in the defensive half of the field such that it is lined up for blocking a kick to the goal from the current known ball position. Also, the robot that is trying to support the defender or the attacker, the *supporter*, positions itself, facing the ball, on the other side of the field from the robot it is trying to support, and its target location is decided based on the position of the ball and the robot it is supporting. In general, its x-coordinate is decided based on whether the ball is in the right or left half of the field and its target y-coordinate is set such that it puts the supporter slightly ahead of the ball location. This makes sure that the supporter does not obstruct the path of its teammate, and if the robot primarily in charge of the ball (attacker or defender) is unable to gain control of it and/or kicks it into an opponent, the supporter is suitably positioned to track the ball and get to it as fast as possible. To prevent rapid changes in target supporter pose, the desired pose is modified iff there is a substantial change in the ball position. This creates a focused attack on the ball and provides a more coordinated team strategy.

7 Research Results

Though the competitions are motivating, exciting and fun, the main goal of the UT Austin Villa robot legged soccer team is to produce cutting edge research results. Over the past three years, we have used the Aibo robots and our RoboCup team code infrastructure as the basis for novel research along several interesting directions. In this section, we provide the titles of these research papers. Full details are available from our team web-page at `http://www.cs.utexas.edu/~AustinVilla/?p=research`.

Vision:

- *Real-Time Vision on a Mobile Robot Platform* [9]
- *Towards Illumination Invariance in the Legged League* [10]
- *Towards Eliminating Manual Color Calibration at RoboCup* [11]
- *Autonomous Color Learning on a Mobile Robot* [8]

Localization

- *Practical Vision-Based Monte Carlo Localization on a Legged Robot* [7]
- *Simultaneous Calibration of Action and Sensor Models on a Mobile Robot* [15]

Joint control

- *A Model-Based Approach to Robot Joint Control* [14]

Locomotion

- *Policy Gradient Reinforcement Learning for Fast Quadrupedal Locomotion* [6]

- *Machine Learning for Fast Quadrupedal Locomotion* [5]

Behavior learning

- *Learning Ball Acquisition on a Physical Robot* [3]

Robot surveillance:

- *Continuous Area Sweeping: A Task Definition and Initial Approach* [1]

8 The Competitions

In the RoboCup initiative, periodic competitions create fixed deadlines that serve as important motivators. In 2005, we entered the Third U.S. Open competition as well as the Ninth International RoboCup Competition. This section describes our results and experiences at those events.

8.1 U.S. Open

The Third U.S. Open RoboCup Competition was held in Atlanta, GA from May 7th to 10th, 2005.[4] Eight teams competed in the four-legged league, and were divided into two groups of four for a round robin competition to determine the top two teams which would advance to the semi-finals. The three other teams in our group were from Georgia Institute of Technology, Spelman College, and Columbia University/CUNY. After finishing in first place in the group, we advanced to the semi-finals against a team from the University of Pennsylvania, and eventually the tournaments third place game against Columbia. The results of our four games are shown in Table 1. Links to videos from these games are available at `http://www.cs.utexas.edu/~AustinVilla/?p=competitions/US_open_2005`.

Opponent	Score (us-them)	Notes
Georgia Tech	4–0	
Columbia/CUNY	3–0	
Spelman	7–0	
Penn	0–1	Semi-final
Columbia	8–0	

Table 1: The scores of our 5 games at the U.S. Open.

Overall, our performance was quite strong, giving up only a single goal and scoring 22. Unfortunately, the one goal against was in the semi-final against Penn in a very close game. CMU eventually beat Penn 2–1 in the final (in overtime). But as an indication of how evenly matched the 3 top teams are, we beat CMU in an exhibition match 2–1. We also played an exhibition match against a team from Dortmund, the winner of the 2005 German open, and lost 2–0.

8.2 RoboCup 2005

The Ninth International RoboCup Competition was held in Osaka, Japan from June 13th to 19th, 2005.[5] 24 teams competed in the four-legged league and were divided into eight groups of three for a round robin competition. The top 16 teams then moved on to a second round robin with 4 teams in each group to determine the top two teams which would advance to the quarter-finals. The teams in our initial group were JollyPochie from Kyushu University and Tohoku University in Japan; and UChile from Universidad de Chile.

[4] `http://www.robocup-us.org/2005/`
[5] `http://www.robocup2005.org`

After finishing first in our group, we advanced to the second round robin in a group with CMDash from Carnegie Mellon University, EagleKnights from ITAM in Mexico, and BabyTigers from Osaka University. After finishing 2nd in that group, we advanced to the quarter-finals against rUNSWift from UNSW in Australia. The results of our 6 games are shown in Table 2. Links to videos from these games are available at `http://www.cs.utexas.edu/~AustinVilla/?p=competitions/roboCup_2005`.

Opponent	**Score (us-them)**	**Notes**
JollyPochie	3–0	
UChile	2–0	
CMDash	1–2	
EagleKnights	9–0	
BabyTigers	3–0	
rUNSWift	1–7	Quarter-final

Table 2: The scores of our six games at RoboCup.

The most exciting game was our first official matchup against CMU. It was a very close game, with UT Austin Villa taking a 1–0 lead before eventually losing 2–1. Meanwhile, the match against rUNSWift demonstrated clearly that our team was not quite at the top level of the competition. Indeed two other teams (NuBots and the German Team) were also clearly stronger than UT Austin Villa. Our goal for next year will be to close that gap.

9 Conclusions and Future Work

The experiences and algorithms reported in this technical report document the third year of the UT Austin Villa legged-league robot team.

There are still many directions for future improvements to our team, as noted throughout this report. We plan to continue our development toward future RoboCup competitions. But more importantly, we continue to make use of this code base as a fully functional research platform and are using it for investigations in various directions as summarized in Section 7.

Overall, developing a competitive RoboCup soccer team has been a rewarding learning experience. We look forward to building from it in the future and continuing to contribute to the RoboCup initiative.

Acknowledgments

Thanks to Ben Bradley, Selim T. Erdoğan, Laurel Issen, and Nick Jong for their contributions to the team development process. The authors would also like to thank Sony for developing the robots, as well as the previous RoboCup legged-league teams for forging the way and providing their source code and technical reports as documentation. This research is supported in part by NSF CAREER award IIS-0237699 and ONR YIP award N00014-04-1-0545.

References

[1] Mazda Ahmadi and Peter Stone. Continuous area sweeping: A task definition and initial approach. In *The 12th International Conference on Advanced Robotics*, July 2005.

[2] Y. Bar-shalom, X. R. Li, and T. Kirubarajan. *Estimation with Applications to Tracking and Navigation.* Wiley and Sons, 2001.

[3] Peggy Fidelman and Peter Stone. Learning ball acquisition on a physical robot. In *2004 International Symposium on Robotics and Automation (ISRA)*, August 2004.

[4] Hiroaki Kitano, Minoru Asada, Yasuo Kuniyoshi, Itsuki Noda, and Eiichi Osawa. RoboCup: The robot world cup initiative. In *Proceedings of the First International Conference on Autonomous Agents*, pages 340–347, Marina Del Rey, California, February 1997.

[5] Nate Kohl and Peter Stone. Machine learning for fast quadrupedal locomotion. In *The Nineteenth National Conference on Artificial Intelligence*, pages 611–616, July 2004.

[6] Nate Kohl and Peter Stone. Policy gradient reinforcement learning for fast quadrupedal locomotion. In *Proceedings of the IEEE International Conference on Robotics and Automation*, May 2004.

[7] Mohan Sridharan, Gregory Kuhlmann, and Peter Stone. Practical vision-based monte carlo localization on a legged robot. In *IEEE International Conference on Robotics and Automation*, April 2005.

[8] Mohan Sridharan and Peter Stone. Autonomous color learning on a mobile robot. In *Proceedings of the Twentieth National Conference on Artificial Intelligence*, July 2005.

[9] Mohan Sridharan and Peter Stone. Real-time vision on a mobile robot platform. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, August 2005.

[10] Mohan Sridharan and Peter Stone. Towards illumination invariance in the legged league. In Daniele Nardi, Martin Riedmiller, and Claude Sammut, editors, *RoboCup-2004: Robot Soccer World Cup VIII*, pages 196–208. Springer Verlag, Berlin, 2005.

[11] Mohan Sridharan and Peter Stone. Towards eliminating manual color calibration at robocup. In Itsuki Noda, Adam Jacoff, Ansgar Bredenfeld, and Yasutake Takahashi, editors, *RoboCup-2005: Robot Soccer World Cup IX*. Springer Verlag, Berlin, 2006. To appear.

[12] Peter Stone, Kurt Dresner, Selim T. Erdoğan, Peggy Fidelman, Nicholas K. Jong, Nate Kohl, Gregory Kuhlmann, Ellie Lin, Mohan Sridharan, Daniel Stronger, and Gurushyam Hariharan. UT Austin Villa 2003: A new RoboCup four-legged team. Technical Report UT-AI-TR-03-304, The University of Texas at Austin, Department of Computer Sciences, AI Laboratory, 2003.

[13] Peter Stone, Kurt Dresner, Peggy Fidelman, Nicholas K. Jong, Nate Kohl, Gregory Kuhlmann, Mohan Sridharan, and Daniel Stronger. The UT Austin Villa 2004 RoboCup four-legged team: Coming of age. Technical Report UT-AI-TR-04-313, The University of Texas at Austin, Department of Computer Sciences, AI Laboratory, October 2004.

[14] Daniel Stronger and Peter Stone. A model-based approach to robot joint control. In Daniele Nardi, Martin Riedmiller, and Claude Sammut, editors, *RoboCup-2004: Robot Soccer World Cup VIII*, pages 297–309. Springer Verlag, Berlin, 2005.

[15] Daniel Stronger and Peter Stone. Simultaneous calibration of action and sensor models on a mobile robot. In *IEEE International Conference on Robotics and Automation*, April 2005.