CMDragons 2011 Team Description

Joydeep Biswas, Can Erdogan, Philip A. Etling, Yuan Tao, and Manuela Veloso

Carnegie Mellon University
{joydeepb,mmv}@cs.cmu.edu
{cerdogan,petling,yuantao}@andrew.cmu.edu

Abstract. In this paper we present an overview of CMDragons 2011, Carnegie Mellon's entry for the RoboCup Small Size League. Our team builds upon the research and success of RoboCup entries in previous years.

1 Introduction

Our RoboCup Small Size League entry, CMDragons 2011, builds upon the ongoing research used to create the previous CMDragons teams (1997-2003,2006-2010) and CMRoboDragons joint team (2004, 2005). Our team entry consists of five omni-directional robots controlled by an offboard computer. Sensing is provided by two overhead mounted cameras linked to the offboard computer. The software then sends driving commands to the individual robots. This paper describes the robot hardware and the offboard control software required to implement a robot soccer team.

2 System Overview

Our team consists of seven homogeneous robot agents, with five being used in a game at any point in time. In Figure 1, an example robot is shown with and without a protective plastic cover. The hardware is mostly the same as used in RoboCup 2006-2009. We believe that our hardware is still highly competitive and allows our team to perform close to optimal within the tolerances of the rules. One noticable hardware improvement for 2010 however, was a new dribbler-mount assembly, better protecting the robot's infrared sensors and dribbler motor. Besides this hardware improvement, we focus most of our efforts on improving the software to fully utilize the robots' capabilities instead.

2.1 Robot Hardware

Each robot is omni-directional, with four custom-built wheels driven by 30 watt brushless motors, each featuring a reflective quadrature encoder. The kicker is a large diameter custom wound solenoid attached directly to a kicking plate. It is capable of propelling the ball at speeds up to $15m/s$, and is fully variable so that controlled passes can also be carried out. The CMDragons robot also

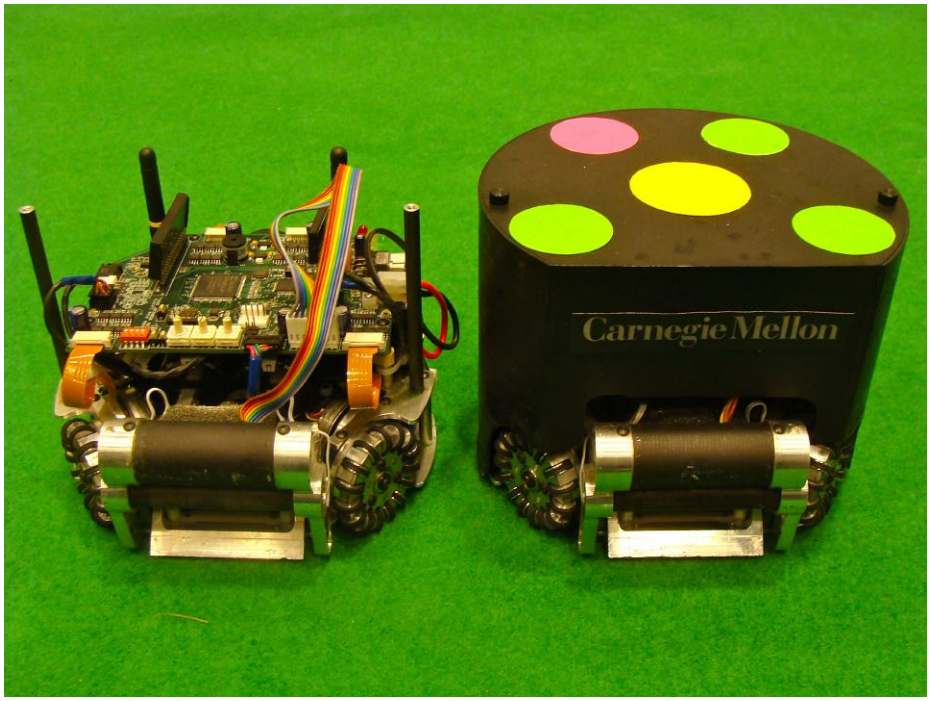

Fig. 1. A CMDragons robot shown with and without protective cover.

has a chip-kicking device, implemented by a custom-made flat solenoid located under the main kicker, which strikes an angled wedge visible at the front bottom of the robot. It is capable of propelling the ball up to $4.5m$ before it hits the ground. Both kickers are driven by a bank of three capacitors charged to $200V$. Ball catching and handling is performed by a motorized rubber-coated dribbling bar which is mounted on an hinged damper for improved pass reception. A more detailed description of the robot's design and electronics can be found in [1].

Our robot is designed for full rules compliance at all times. The robot fits within the maximum dimensions specified in the official rules, with a maximum diameter of 178mm and a height of 143mm. The dribbler holds up to 19% of the ball when receiving a pass, and somewhat less when the ball is at rest or during normal dribbling. The chip kicking device has a very short travel distance, and at no point in its travel can it overlap more than 20% of the ball due to the location of the dribbling bar. While technically able to perform kicks of up to $15m/s$, the main kicker has been hard-coded to never exceed kick-speeds of $10m/s$ for full rule compliance.

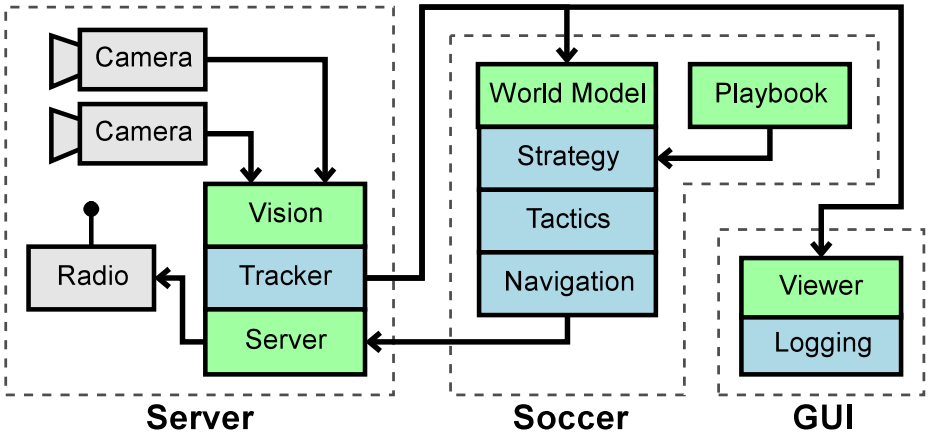

Fig. 2. The general architecture of the CMDragons offboard control software.

2.2 Software

The software architecture for our offboard control system is shown in Figure 2. It follows the same overall structure as has been used in the previous year, outlined in [2, 1]. The major organizational components of the system are a server program which performs vision and manages communication with the robots, and two client programs which connect to the server via UDP sockets. The first client is a soccer program, which implements the soccer playing strategy and robot navigation and control, and the second client is a graphical interface program for monitoring and controlling the system.

The server program consists of vision input, tracker, radio, and a multi-client server. The vision input is supplied via ethernet from the RoboCup SSL shared vision system SSL-Vision [3]. Some of the integration details are described in section 3 of this paper. Tracking is achieved using a probabilistic method based on Extended Kalman-Bucy filters to obtain filtered estimates of ball and robot positions. Additionally, the filters provide velocity estimates for all tracked objects. Further details on tracking are provided in [4]. Final commands are communicated by the server program using a RS232 radio link.

The soccer program is based on the STP framework [4]. A world model interprets the incoming tracking state to extract useful high level features (such as ball possession information), and act as a running database of the last several seconds of overall state history. This allows the remainder of the soccer system to access current state, and query recent past state as well as predictions of future state through the Kalman filter. The highest level of our soccer behavior system is a strategy layer that selects among a set of plays [5, 6]. Below this we use a tree of tactics to implement the various roles (attacker, goalie, defender), which in turn build on sub-tactics known as skills [4]. One primitive skill used by almost all behaviors is the navigation module, which uses the RRT-based

ERRT randomized path planner [7–9] combined with a dynamics-aware safety method to ensure safe navigation when desired [10]. It is an extension of the Dynamic Window method [11, 12]. The robot motion control uses trapezoidal velocity profiles (bang-bang acceleration) as described in [13, 4]. Additionally, our system features a detailed physics-based simulator based on rigid-body dynamics as described in [2]. One focus of our work this year is on modelling opponent behaviours from logged game data, as described in Section 4.

3 Vision Hardware and Software

CMDragons 2011 operates using SSL-Vision as its vision system [3]. In our lab, we use two Firewire 800 cameras (AVT Stingray F-46C) which provide a 780×580 progressive video stream at 60Hz. SSL-Vision is released as open source and is therefore available to all teams. In order to use SSL-Vision, the "Vision" component in Figure 2 represents a network client that receives packets from the SSL-Vision system. These packets will contain the locations and orientations of all the robots, as well as the location of the ball. However, data fusion of the two cameras and motion tracking will continue to be performed within our system, as SSL-Vision does not currently support such functionality.

SSL-Vision has the capability to report multiple balls on the field. We use this feature to selectively track balls on the field, which has proven to be especially useful while sharing the field with other teams during testing and setup time during RoboCup. The ball tracker selects the ball detection reported by SSL-Vision which is closest to the next frame prediction of the tracker. Additionally, the ball tracker can be selectively reset to track the ball closest to the center of the field, which is how we define "our" ball during

4 Modeling Opponent Behavior

Based on our experience in the SSL, we first conjecture that each team has a finite set of pre-defined plays and within a game, depending on the world state, they commit to the execution of a single play. The team description reports of several years from previous years also support our hypothesis. Secondly, we note our extensive logging architecture where for an individual game, we record the position and orientation of every robot in the game, the position of the ball and the game state depending on the referee calls (free kick, goal kick and etc.).

In this section, we introduce an architecture that exploits the comprehensive amount of log data regarding the game play of a team to model its behavior. The key idea is to represent the spatiotemporal behavior of a robot as an instance of a geometric trajectory path (Figure 3) and to categorize sets of trajectories into different clusters. Below, we will first briefly discuss several challenges with this approach and then, discuss how this idea can be utilized to make online predictions of future world states and take pre-emptive actions.

The first question is how to define the beginning and end of a trajectory. We define a time period, an episode, which starts with the referee rewarding

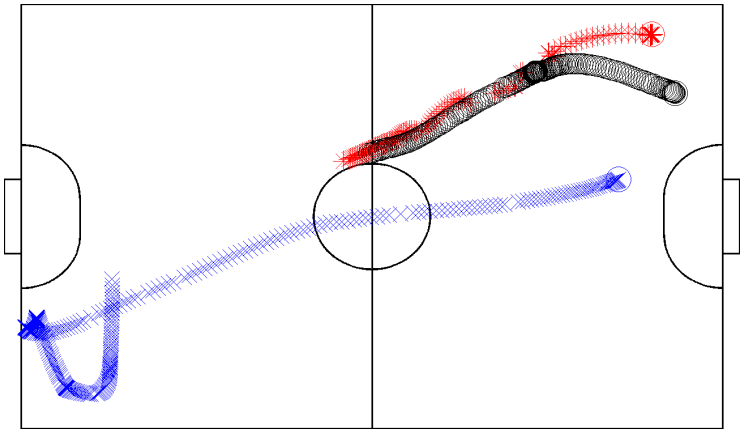

Fig. 3. The behavior of a team throughout an episode where blue and black robots position for a pass and the ball (in red) is passed to the black one. Note that the actuator robot is not shown.

the possession of the ball to a team and ends with the collision between a robot and the ball after its initial actuation. An episode lasts about 3s in average. The second question is how to quantify the similarity of sets of trajectories between different episodes. We utilize the geometric Hausdorff distance to measure the similarity of individual trajectories. To quantify the similarity between sets of trajectories, we first find a matching between the individual trajectories of the two sets and compute the sum of the distances of the most similar trajectories. In Figure 4, we provide the results of applying hierarchical clustering on the similarity values between the 29 episodes recorded in a game play in SSL 2010.

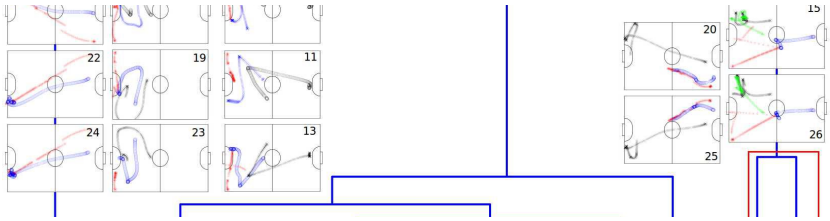

Fig. 4. A sample clustering of sets of robot trajectories into behavior patterns. The red boxes represent the clusters obtained by hierarchical clustering. The trajectory sets in the same cluster are instances of the same behavior pattern.

Lastly we discuss how opponent modeling can be used to make predictions within the game. Assume we have a set of clusters where each cluster is made of sets of trajectory patterns and the beginning of a new episode is detected, and a new pattern is being observed. We compute the distance between a cluster

and a new, partial pattern by computing the distance between each pattern in the cluster and the new pattern and taking a mean. To compute the distance between a complete pattern and a partial pattern, we shorten the complete pattern so that the duration of the pattern is the same as the duration of the partial pattern. Our experiments on the real data from 4 teams in SSL 2010 games, we can correctly classify %70 of behavior patterns by observing %30 of any input trajectory set. In summary, by using opponent modeling and online detection, we seek to provide tools for a more adaptive game play.

5 The Attacker Control System

The CMDragons robots perform motion profiling off-board, on the central computer. This raises three problems, namely:

System latency: Latency in the control loop introduces a phase delay between the expected and actual motion profiling. This however is minimized by forward-predicting the observed world state and computing the motion profile on this future state.

Hesitation: Precise motion control can lead to pauses while changing target locations due to switching of motion profiles.

Underperformance: The robot's motion profile is computed using expected robot acceleration and top speeds, although the true values might differ, and in certain cases the robot might actually be capable of exceeding the expected values.

To counter the effect of these problems, we implemented an "Attacker Control System". The Attacker Control System has two main features:

1. The motion profile parameters (acceleration and velocity limits) are separate for AI calculations and for execution. Specifically, the parameters used for AI calculations are more conservative than the true robot parameters, while the execution parameters marginally exceed the true parameters.
2. Intercept and target locations are explicitly modified by a proportional-derivative (PD) controller

The PD controller is implemented as follows. Let the target location of the attacker, as computed by the AI be denoted by l_d. Let the current robot location be denoted by l_r. The modified target location $\tilde{l}_d$ is given by,

$$\tilde{l}_d = l_d + k_p(l_d - l_r) + k_d \frac{d(l_r)}{dt} \tag{1}$$

The proportional and derivative gains k_p, k_d are hand-tuned, and two separate sets are used during the accleration and the deceleration stages. The modified target location $\tilde{l}_d$ is then used for motion profiling using the execution motion profile parameters.

Competition	Result
US Open 2003	1st
RoboCup 2003	4th
RoboCup 2004	4th [1]
RoboCup 2005	4th [1]
US Open 2006	1st
RoboCup 2006	1st
China Open 2006	1st
RoboCup 2007	1st
US Open 2008	1st
RoboCup 2008	2nd
RoboCup 2009	Eliminated during quarter-final
RoboCup 2010	2nd

Table 1. Results of RoboCup small-size competitions for CMDragons from 2003-08

6 Conclusion

This paper gave a brief overview of CMDragons 2011, covering both the robot hardware and the software architecture of the offboard control system. The hardware has built on the collective experience of our team and continues to advance in ability. The software uses our proven system architecture with continued improvements to the individual modules. The CMDragons software system has been used in four national and eight international RoboCup competitions, and the competition results since 2003 are listed in Table 1. We believe that the RoboCup Small Size League is and will continue to be an excellent domain to drive research on high-performance real-time autonomous robotics.

References

1. Bruce, J., Zickler, S., Licitra, M., Veloso, M.: CMDragons 2007 Team Description. Technical report, Tech Report CMU-CS-07-173, Carnegie Mellon University, School of Computer Science (2007)
2. Zickler, S., Vail, D., Levi, G., Wasserman, P., Bruce, J., Licitra, M., Veloso, M.: CMDragons 2008 Team Description. In: Proceedings of RoboCup 2008
3. Zickler, S., Laue, T., Birbach, O., Wongphati, M., Veloso, M.: SSL-vision: The shared vision system for the RoboCup Small Size League. RoboCup 2009: Robot Soccer World Cup XIII (2010) 425–436
4. Browning, B., Bruce, J.R., Bowling, M., Veloso, M.: STP: Skills tactics and plans for multi-robot control in adversarial environments. In: Journal of System and Control Engineering. (2005)
5. Bowling, M., Browning, B., Veloso, M.: Plays as effective multiagent plans enabling opponent-adaptive play selection. In: Proceedings of International Conference on Automated Planning and Scheduling (ICAPS'04). (2004)

[1] Provided software component as part of a joint team with Aichi Prefectural University, called CMRoboDragons

6. Bruce, J.R., Bowling, M., Browning, B., Veloso, M.: Multi-robot team response to a multi-robot opponent team. In: Proceedings of the IEEE International Conference on Robotics and Automation, Taiwan (May 2003)
7. Bruce, J.R., Veloso, M.: Real-time randomized path planning for robot navigation. In: Proceedings of the IEEE Conference on Intelligent Robots and Systems. (2002)
8. LaValle, S.M., James J. Kuffner, J.: Randomized kinodynamic planning. In: International Journal of Robotics Research, Vol. 20, No. 5. (May 2001) 378–400
9. James J. Kuffner, J., LaValle, S.M.: RRT-Connect: An efficient approach to single-query path planning. In: Proceedings of the IEEE International Conference on Robotics and Automation. (2000)
10. Bruce, J.R., Veloso, M.: Safe multi-robot navigation within dynamics constraints. Proceedings of the IEEE **94** (July 2006) 1398–1411
11. Fox, D., Burgard, W., Thrun, S.: The dynamic window approach to collision avoidance. IEEE Robotics and Automation Magazine **4** (March 1997)
12. Brock, O., Khatib, O.: High-speed navigation using the global dynamic window approach. In: Proceedings of the IEEE International Conference on Robotics and Automation. (1999)
13. Bruce, J.R.: Real-Time Motion Planning and Safe Navigation in Dynamic Multi-Robot Environments. PhD thesis, Carnegie Mellon University (Dec 2006)
14. Zickler, S., Laue, T., Birbach, O., Wongphati, M., Veloso, M.: SSL-Vision: The Shared Vision System for the RoboCup Small Size League. In: Proceedings of the RoboCup Symposium 2009 (To Appear)