2008 IEEE International Conference on
Robotics and Automation
Pasadena, CA, USA, May 19-23, 2008

CMDragons: Dynamic Passing and Strategy on a Champion Robot Soccer Team

James Bruce, Stefan Zickler, Mike Licitra, and Manuela Veloso

Abstract— **After several years of developing multiple RoboCup small-size robot soccer teams, our CMDragons robot team achieved a highly successful level of performance, winning both the 2006 and 2007 competitions without losing a single game. Our small-size team consists of five executing wheeled robots with centralized, off-board perception and decision making. The decision making framework consists of a set of layered components, consisting of perception, evaluation and strategy, robot tactics and skills, and real-time navigation. In this paper, we present the strategy, action selection, and execution aspects of our architecture, with a focus on passing as an example of effective coordinated teamwork. The design enabled our robot team to score using multiple methods, from direct shooting up to 3D passes deflected in midair, resulting in a rich set of actions that were difficult for adversaries to counter. We provide several performance quantified claims supported by testing in our laboratory and in competition settings.**

I. Introduction

For the past nine years, our research group has developed entries for the RoboCup world robot soccer championship [1], [2], [3]. Through many revisions and redesigns, we have gained insights into the integrated architectures required to operate successfully in an adversarial multi-robot soccer domain. This paper describes the teamwork of our CMDragons robot team, which has helped bring our most recent competition success. In particular, we focus on dynamic passing, describing it in detail, and also placing it within the context of the large integrated control architecture.

We first introduce the domain. The rules and specifications of the RoboCup robot soccer leagues are dynamic and revised every year to increasingly challenge the research in multi-robot teams. We present the domain and rules as in RoboCup'06, which are equivalent to the ones in RoboCup'07 in terms of the work presented. We then follow by describing our robot soccer architecture. Next, we introduce the teamwork, and follow with individual robot action execution. Finally, we conclude with results and remarks.

II. The RoboCup Small Size Domain

The domain for our robot soccer work is the RoboCup "small-size" league [4]. This league involves teams of five small robots, each up to 18cm in diameter and up to 15cm height. The field of play is a green carpet measuring 4.9m by 3.8m, and an orange golf ball is used as the small-size robot soccer ball. The competition is between two teams of up to five robots each which aim at scoring into the opponent goal. A game consists of two halves of 15 minutes each. No human input is allowed to manipulate or control the robots during the game. The match is refereed by a human according to pre-specified rules, and the referee's signals are encoded and sent via a serial link to each autonomous team controller. Thus the two robot teams must compete using full autonomy in every aspect of the game play.

The robots, in addition to the size limitations, are forbidden from covering more than 20% of the ball, defined by the area of the ball falling within the convex hull of the robot when projected onto the ground plane. Robots are not allowed to gain full control of the ball and remove all of its degrees of freedom. Furthermore, the robots cannot travel more than 0.5m while in contact with the ball, preventing a single robot from dribbling the ball for that long without either passing or kicking. Collectively these limitations are intended to promote passing and other team play.

For the team control, off board sensing, computation, and communication are allowed. This permitted centralized perception and control setup has led to a fast-paced game where sensing is no longer the primary limitation. Robot speeds can exceed 2m/s, and many robots have kicking devices which can propel the ball at speeds of at least 10m/s. Figure 1 shows a snapshot of a small-size robot soccer game.

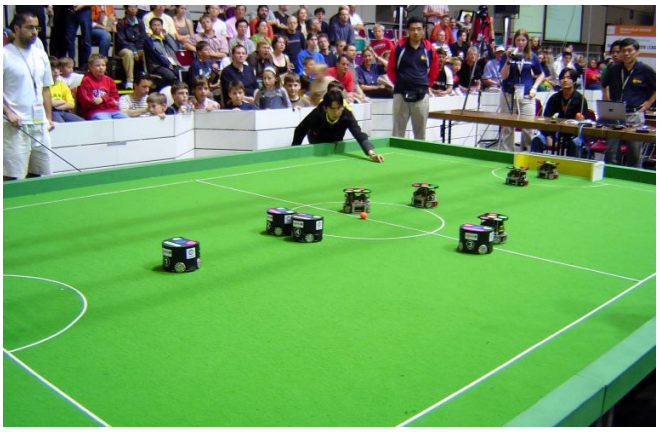

Fig. 1. Two teams competing in RoboCup small-size robot soccer. (Note that the overhead cameras and the off-board computation are not depicted.)

III. Integrated Robot Soccer Architecture

Our CMDragons team consists of five homogeneous robots each featuring the following hardware capabilities:

1) Omni-directional (holonomic) drive system using four driven wheels.

During this work, all authors were affiliated with the Computer Science Department of Carnegie Mellon University, Pittsburgh PA 15213, USA {jbruce, szickler, mlicitra, mmv}@cs.cmu.edu

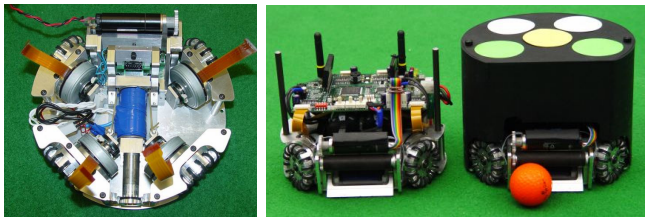

Fig. 2. Top view of the robot drive system, kicker, and dribbler (left), front view of an assembled robot (center), and with protective cover and ball (right). (Thanks in particular to Michael Licitra.)

2) Ball catching and handling device ("dribbler") using a motorized, rubber-coated bar mounted on a hinged damping system.
3) A variable speed "flat" kicker that can kick the ball up to $15m/s$ along the ground.
4) A variable power "chip" kicker that can kick the ball into the air up to a distance of $4.5m$.

The rules' allowance of off board sensing has lead nearly every team to adopt a centralized approach for the majority of the robot control. Figure 3 shows the CMDragons data flow, which is typical of most teams. Perception is provided by two, in our case, overhead cameras, feeding into a central computer to process the image and locate the 10 robots and the ball on the field at a 60Hz rate. We have developed successful algorithms for effective real-time color segmentation and pattern detection [5]. The detected robot locations are fed into an extended Kalman filter for tracking and velocity estimation, and then passed to our soccer strategy and control, which consists of three major components: (1) world state evaluation and play selection; (2) skills and tactics; (3) navigation.

1) *World state evaluation* involves determining predefined high level states about the sensed world, such as whether the team is on offense or defense. This abstraction of the perceived state allows the selection among a set of possible team *plays* [6]. Furthermore, a specific evaluation of alternatives is included to find and to rank possible subgoals, such as determining a good location to receive a pass.
2) *Skills* and *tactics* implement the primitive behaviors for a robot, and can range from the simple "go to point" skill, up to complex tactics such as "pass" or "shoot" [7]. A *role* in a robot team play is defined as a tactic with all parameters fully specified, which is then assigned to a robot to execute. The executed tactic generates a navigation target either directly or by invoking lower level skills.
3) The *navigation* algorithm takes the targets specified by tactics and plans a path using a randomized path planner with safety guarantees [8]. The path is then processed by motion control and dynamic obstacle avoidance to generate robot velocity commands. The resulting commands are sent to the robots by radio.

The entire CMDragons control executes at 60Hz, synchronously with the vision, to optimize its response in real-time to the dynamic and competitive robot soccer game.

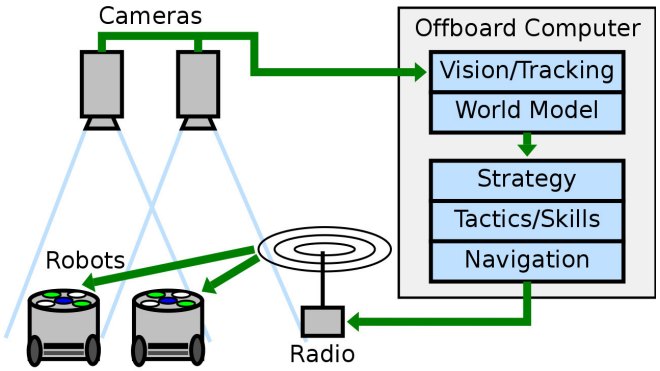

Fig. 3. The overall autonomous architecture for CMDragons.

IV. Teamwork

Multi-robot domains can be categorized according to many different factors. One such factor is the underlying parallelism of the task to be achieved. In highly parallel domains, robots can complete parts of the task separately, and mainly need to coordinate to achieve higher efficiency. In a more serialized domain, some part of the problem can only be achieved by one robot at a time, necessitating tighter coordination to achieve the objective efficiently. Robot soccer falls between parallel and serialized domains, with brief periods of joint actions. Robot soccer is a serialized domain primarily due to the presence of a single ball; At any given time only one robot from a team should be actively handling the ball. Multi-robot coordination needs to reason about the robot that actively addresses the serial task (ball), the *active* robot, and to assign *supporting* objectives to the other team members [9]. These supporting robots add a parallel component, as they can execute actions in a possibly loosely coupled way to support the overall team objective.

A. Multi-Robot Role Assignment

A large number of methods exists for task assignment and execution in multi-agent systems, which have been organized in a comprehensive taxonomy [10]. Direct conflicts can arise among multiple executing behaviors, as well as additional complications, when the number of tasks does not match the number of robots [11]. Task assignment methods in robot soccer can handle tight coordination using messaging between the behaviors controlling each agent [12]. Instead we have developed the STP (skills-tactics-plays) architecture, which uses predefined plays for multi-robot coordination with dynamic play selection and role assignment [7].

In addition to the assignment of tasks to agents, there still lies the problem of describing how each robot should implement its individual task. Potential fields present an efficient solution for local obstacle avoidance [13], though with the inherent limitations local minima and the inability to represent hard action constraints. The SPAR algorithm [9] describes a method combining binary constraints with linear objective functions to define a potential over the workspace, but is only used for navigation target selection rather than direct actions.

The CMDragons task allocation follows the STP architecture [7]. It adopts a split of active and support roles and solves each of those subtasks with a different method. Active roles that manipulate the ball generate commands directly, receiving the highest priority so that supporting roles do not interfere. Supporting roles are based on optimization of potential functions defined over the configuration space, building upon SPAR [9]. With these two distinct solutions, part of our CMDragons control is optimized for the serialized aspect of ball handling, while another part is specialized for the loosely coupled supporting roles. We address the need for the even more tight coupling that is present in passing plays through behavior dependent signaling between the active and supporting behaviors [12].

B. Objective Evaluation Functions

The CMDragons navigation targets of supporting roles are determined by world state evaluation functions defined over the entire soccer field. Each function holds the world state external to a robot constant, while varying the location of the robot to determine a real valued evaluation of that state within the workspace. Hard constraints are represented using multiplicative boolean functions, whereas soft constraints are modeled as general real-valued functions. We specifically focus on presenting the evaluation of the supporting roles for the passing teamwork.

1) Passing Evaluation: Figure 4 shows an example of the general form of the pass position evaluation function, which evaluates each point p using several variables given the current world state, as shown:

- d is the reachable area, which is defined by the perpendicular distance the robot can travel in the time it takes a pass to reach a perpendicular centered on p;
- a is the subtended angle of d centered at the current ball location;
- b is the angular width of the unobstructed angle toward the goal from point p;
- c is the angle between the pass to p, and the shot from p to the center of the free angle on the goal.

The variables are defined towards the evaluation of a pass followed by a possible shot on goal. The variables are similarly defined if the evaluation considers a pass to another target instead of a goal.

Based on these general variables, we then define several types of passes of possible passes, which are evaluated:

- A **one-touch pass-and-shoot** that intercepts the moving ball to kick it at the goal. The evaluation estimates the time t as the pass length divided by the pass speed, plus the shot length divided by the shooting speed. An angular preference $k(c)$ is calculated which increases linearly from 0 at $c = 0$ to 1 at $c = 45°$. It stays at 1 until $c = 90°$, where it decreases rapidly to 0. The evaluation is then $[k(c) \min(a, b)/t]$.
- A **pass-receive-turn-shoot** that explicitly receives the ball and then aims and shoots. The evaluation estimates the time t as the sum of the pass time, turning time for c, and shot time. The evaluation is then $[\min(a, b)/t]$.

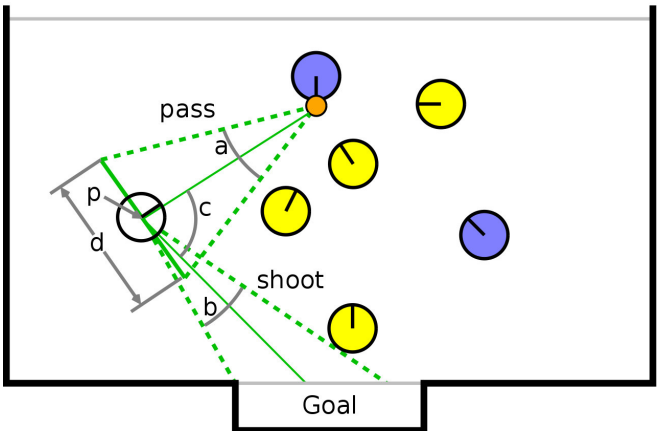

Fig. 4. An example demonstrating the pass evaluation function inputs. The input values for evaluating a point p are the subtended angle a of the reachable area d, the unobstructed goal angle b, and the angle c between the pass and shoot centerlines. These values are combined in specific evaluation functions to achieve the desired passing behavior.

- Partial **chip-pass variants** of the above passing methods, where a chip shot is used to pass partway to the receiver, but dropping soon enough that it will roll by the time it reaches the receiver. The receiver robot can position itself for the pass. These chip-pass variants are used when there are adversarial robots or team member robots strategically blocking the direct passing path.
- A **direct chip deflection "header"** where a chip pass is calculated to a reachable robot position (a part of d), with a target point that passes over d at a height midway up the robot. The robot deflects the ball directly into the goal, so the pass and shoot speed are identical. The evaluation is then identical to the one-touch pass-and-shoot evaluation.

Figure 5 shows the resulting plots from two example passing situations, with the pass evaluation function for one-touch pass-and-shoot overlaid on a field. The location of the ball and other robots causes very different target locations to be selected for the supporting robot. Because large portions of the field are covered by a continuous evaluation, the existence of an unobstructed maximum is likely.

2) Properties: While the exact parameters and weights applied in evaluation functions are highly application dependent, and thus not of general importance, the approach has proved of useful throughout many revisions of our system. In particular, with well designed functions, the approach has the useful property that large areas have a nonzero evaluation. This feature provides a natural ranking of alternative positions so that conflicts can be resolved. Thus multiple robots can be placed on tasks with conflicting actions, or even the same task; The calculation for a particular robot simply needs to discount the areas currently occupied by other robots. Direct calculation of actions, as is used for active roles, does not inherently provide ranked alternatives, and thus leads to conflicting actions when other robots apply the same technique.

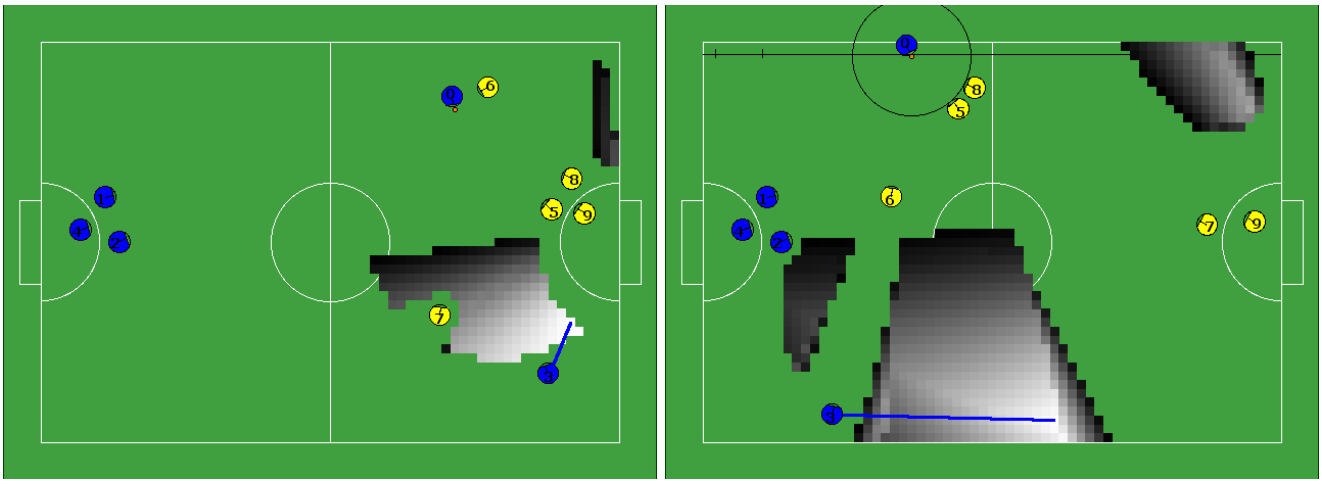

Fig. 5. Two example passing situations with the passing evaluation metric overlaid. The evaluation values are shown in grayscale where black is zero and the maximum value is white. Values less than 0.5% of the maximum are not drawn. The maximum value (in white) is further indicated by the end of the line extending from the current position of the supporting robot, representing the position to which it will move to receive the pass.

V. Action Execution

Numerous low level skills are required to implement a robot soccer team, many of which, such as shooting on a goal, we have previously described (e.g., [7]). Two of the more unique parts of our current CMDragons team are the *delta-margin metric* for choosing when to kick a ball with aiming a shot, and the *one-touch pass-and-shoot* method to achieve the robot soccer equivalent of the "one-touch" goal shot from human soccer.

A. Delta-Margin Shooting Metric

The shooting metric is a skill that must determine the appropriate time of energizing the kicking device to execute an aimed shot. The input to the shooting metric is a range of angles $[g_0, g_1]$ representing the target, along with the robot's current angle $\alpha(t)$. This data is provided each frame as a data stream, and the output of the metric is the a binary value of whether to kick or not. During this time, the robot will position itself to aim at the center of the angular range. This problem is similar to an automated assembly task where a part is positioned and then dropped into place. In both cases, there is a trade off between probability of correct placement (i.e., within tolerance), and the time used in positioning. Ideally we would like something that maximizes the probability of correct placement while minimizing the total time taken in positioning. In the robot soccer environment, this is complicated by the fact that the target angles can change as a function of time $[g_0(t), g_1(t)]$. This situation is similar to an assembly task where both parts to be mated are moving.

Our method for shooting relies on the assumption that the probability of shooting within the tolerance is proportional to the margin, where the margin is defined as the angular distance to the nearest edge of the target range. Formally, we can define the margin function $m(t)$ as shown in Equation 1, which in turn depends on the angle normalization function N, as shown in Equation 2.

$$m(t) = \max\left[N(\alpha(t) - g_0(t)), N(g_1(t) - \alpha(t))\right] \quad (1)$$

$$N(a) = \begin{cases} N(a + 2\pi) & \text{if } a < -\pi, \\ N(a - 2\pi) & \text{if } a > +\pi, \\ a & \text{otherwise.} \end{cases} \quad (2)$$

(3)

Using the definition of $m(t)$, we can define the binary shooting function $S(t)$ as shown in Equation 4. The first case will prevent shooting unless the margin is within the tolerances The second case case will shoot when the margin is nearer to the optimal margin than the constant fraction β (we use $\beta = 0.9$). The third case, which is the primary contribution of this method, prevents shooting as long as the margin is increasing with time. In all remaining cases the metric will elect to shoot.

$$S(t) = \begin{cases} 0 & \text{if } m(t) < 0, \\ 1 & \text{if } m(t) > \beta(g_1(t) - g_0(t))/2, \\ 0 & \text{if } m(t) > m(t-1), \\ 1 & \text{otherwise.} \end{cases} \quad (4)$$

This method has worked extremely well in practice, as it appears to strike a good balance between the conflicting options of shooting as early as possible (to capture short-lived opportunities) and waiting to improve the aim (to lower the probability of missing the target). Though simple to compute, it captures all of the following qualitative properties:

- Try to aim for the center of the target angular range.
- If an angular range is widening, delay shooting since the miss probability is dropping with time.
- If an angular range is narrowing, take the shot since the miss probability is increasing with time.
- If aiming past a moving object (such as a goalkeeper), delay shooting iff our goal probability is improving faster than the opponent is decreasing it.

Figures 6 and 7 show two examples of the shooting method executing on a real robot with the relevant variables plotted over time until the kick is taken. The experiment setup was a single robot 1.5m from an open goal. In the first example, the margin increases to the maximum, and the kick is taken due to the zero crossing of the margin delta. In the second example, the margin stops improving so the shot is taken before the maximum (the ball was rolling away from the robot causing its aim to stop improving).

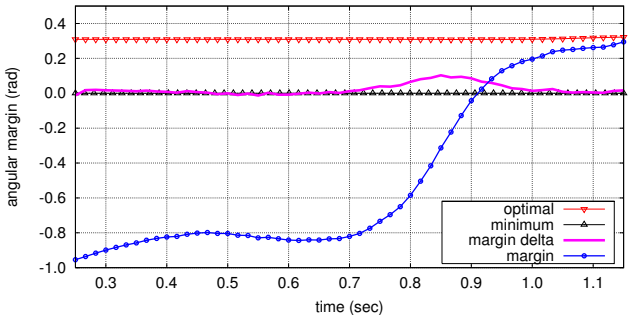

Fig. 6. An example plot of the delta-margin shooting metric reaching a maximum margin. Shot is taken at $t = 1.15$.

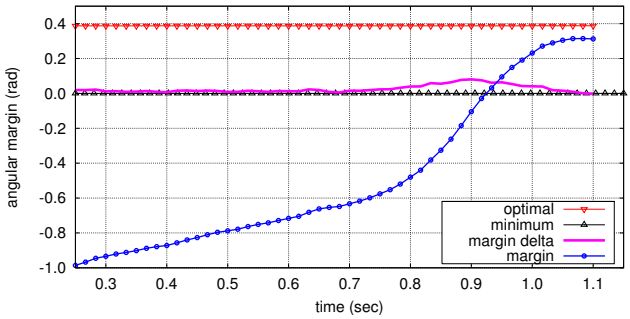

Fig. 7. An example plot of the delta-margin shooting metric reaching a zero-crossing of the margin delta. Shot is taken at $t = 1.1$.

B. One-touch Aiming

The one-touch pass-and-shoot skill is a method for intercepting a moving ball to shoot it at the goal, and corresponds to a "one-touch" strike in human soccer. This skill combines our existing ball interception and target selection routines with a method for determining the proper angle to aim the robot to accurately redirect the ball to the target. In order to calculate the proper aiming angle, a model of how interaction with the kicker will affect the speed of the ball is needed. In particular, while the kicker adds a large forward component to the ball velocity, effects from the ball's original (incoming) velocity are still present and non-negligible.

Figure 8 illustrates the model for the one-touch aiming. R_h and R_p represent the normalized robot heading and perpendicular, respectively. After having explored numerous options to determine the final ball velocity (v_1), we settled on a robust model as a weighted sum of three components:

- Initial ball velocity v_0 damped by the dribbling device. This is a tangential velocity along R_p, or $(R_p \cdot v_1)R_p$, which is scaled by a damping factor $\beta \in [0, 1]$.
- Initial ball velocity v_0 reflected by the robot heading R_h. This is expressed as vector reflection of v_0 by R_h, scaled by a constant $\gamma \in [0, 1]$.
- Additive velocity imparted by the kicker along R_h. The kicker provides an impulse that could propel a ball at rest to speed k (i.e., $\|v_0\| = 0 \rightarrow \|v_1\| = k$). Because the kicker is part of the moving robot, k is the sum of the kicking device speed and the robot speed along R_h.

Using this model, we then estimate v_1 as:

$$\hat{v_1} = \beta(R_p \cdot v_0)R_p + \gamma(v_0 - 2(v_0 \cdot R_h) \cdot R_h) + kR_h \quad (5)$$

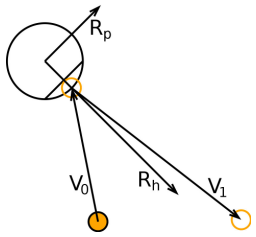

Fig. 8. The model for one-touch pass-and-shoot. The final velocity of the ball v_1 contains a component of the initial velocity v_0, and thus is not parallel to the robot heading R_h.

We determined the model parameter values experimentally, by recording the incoming and outgoing velocity at a variety of angles and kick speeds, and calculating the most likely values. We found that $\beta = 0.1$ and $\gamma = 0.5$ best modeled the data, but these values are likely to be dependent on the exact robot design, and can even be affected by the field surface. The calibration procedure is straightforward however, so we have not found this to be a major limitation.

Of course, the forward model alone is not sufficient, as the control problem requires solving for the robot angle given some relative target vector g. To invert the model, we use bisection search. The bounding angles for the search are the original incoming ball angle (where the residual velocity component would be zero) and the angle of target vector g (where we would aim for an ideal kicker with total damping ($\beta = 0, \gamma = 0$). The actual solution lies between these limits, and we can calculate an error metric e by setting up Equation 5 as a function of the robot angle α.

$$\begin{aligned} R_h(\alpha) &= \langle \cos\alpha, \sin\alpha \rangle \\ R_p(\alpha) &= \langle -\sin\alpha, \cos\alpha \rangle \\ \hat{v_1}(\alpha) &= \beta(R_p(\alpha) \cdot v_0)R_p(\alpha) + kR_h(\alpha) + \\ &\quad \gamma(v_0 - 2(v_0 \cdot R_h(\alpha)) \cdot R_h(\alpha)) \quad (6) \\ e(\alpha) &= \hat{v_1}(\alpha) \cdot g \quad (7) \end{aligned}$$

Thus when $e(\alpha) > 0$ the solution lies with α closer to g, while if $e(\alpha) < 0$ the solution is closer to v_0. A solution at the value of α where $e(\alpha) = 0$, so bisection search is simply terminated whenever $\|e(\alpha)\| < \epsilon$. While it is possible to invert the model so that search is not required, using a numerical method for determining α allowed rapid testing of different models and parameters. The complexity of the approximation is $O(\log(1/\epsilon))$, which has proven adequately fast in practice.

We have found the one-touch aiming method to work with passes between $2m/s$ and $4m/s$, and has proven faster than a more explicit receive-turn-shoot approach. The main limitation of the approach is that the angle between the pass and the shot on goal should generally lie between 30 and 90 degrees. The receive-turn-shoot approach can be used when the angle is not amenable to the pass-and-shoot approach.

We also adapted the 2D version of one-touch aiming to the 3D problem of soccer "headers." The chip kicker is used to kick the ball in the air, and a dynamics model of the ball fit a parabolic trajectory to the observed ball position. This allows a robot to intercept a ball while still in the air to deflect it into a goal. Because the kicker is not used, the model for aiming is pure reflection ($\beta = 0, \gamma = 1.0$). The interception method used is to drive to the point where the ball will reach a particular height above the ground (halfway up the flat part of the robot's protective cover). Due to the decreased accuracy of chip kicks, this type of pass normally does not allow the receiving robot to remain at a fixed location, and depends heavily of the receiving robot adjusting to intercept the ball. Despite the low probability of a successful pass compared to other methods $P[\text{success}] = 0.3$, when it succeeds it has a high chance of scoring as it leaves little time for the opponent team to react.

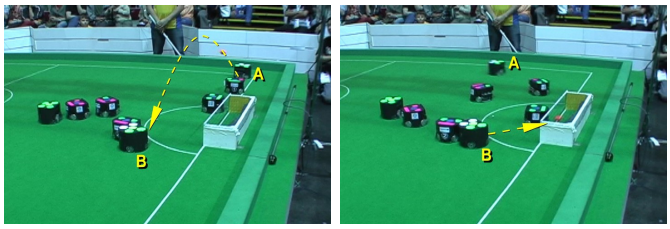

Fig. 9. An example header from the RoboCup 2006 semi-final match. Robot A passes to robot B, which then deflects the ball into the goal. Robot B chooses its angle using one-touch aiming in order to deflect into the gap left by the goalkeeper.

VI. Results and Conclusion

This paper gives an overview of the CMDragons methods for teamwork, and in particular passing. The overall control combines many modules and consists of numerous algorithms which must operate together as part of a tightly integrated autonomous multi-robot system. To demonstrate how the passing contributed to the success of the teams, Figure 10 shows a tally of how goals were scored by the two CMDragons'06 and CMDragons'07 world champion teams in the RoboCup competitions. Robots in both offense and defense plays could pass and shoot, using the shared evaluation functions. Some goals were even scored by directly intercepting an opponent's kick and immediately scoring a goal. Other goals were scored on set plays (corner kicks, free kicks, and penalty kicks). The values in the direct and passing columns show that passing played a significant role in addition to direct shots on goal. Numerous goals were scored both with flat passes and chip passes. The overall diversity in scoring demonstrates how our team architecture can successfully operate in various game play situations.

Situation	Flat-Direct		Flat-Pass		Chip-Pass	
	'06	'07	'06	'07	'06	'07
Offense Shots	21	30	7	7	1	1
Defense Shots	5	0	0	0	0	0
Interceptions	2	1	0	0	0	0
Corner Kicks	0	0	6	0	2	5
Free Kicks	0	1	7	5	3	0
Penalty Kicks	1	1	0	0	0	0

Fig. 10. Classification of recorded shots that resulted in a goal for our CMDragons'06 and CMDragons'07 teams. (8 of our 63 goals in 2006 and 10 of our 61 goals in 2007 were not recorded and thus not accounted for in this table.)

Beyond robot soccer, our work contributes a fully integrated multi-robot architecture with challenging demands for exact robot positioning to enable the performance of a task cooperatively. The low level skills give a method for binary decision making under real-values tolerances in a dynamic system, and a method for deflecting an object dynamically using pure acceleration control. We hope that our research can aid others in designing robot systems to robustly work in dynamic, multi-robot domains with fast-changing environments.

References

[1] M. Veloso, P. Stone, and K. Han, "The CMUnited-97 robotic soccer team: Perception and multiagent control," *Robotics and Autonomous Systems*, vol. 29 (2-3), pp. 133–143, 1999.

[2] M. Veloso, M. Bowling, S. Achim, K. Han, and P. Stone, "The CMUnited-98 champion small robot team," in *RoboCup-98: Robot Soccer World Cup II*. Springer Verlag, 1999.

[3] J. R. Bruce, M. Bowling, B. Browning, and M. Veloso, "Multi-robot team response to a multi-robot opponent team," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2003.

[4] H. Kitano, M. Asada, Y. Kuniyoshi, I. Noda, and E. Osawa, "Robocup: The robot world cup initiative," in *Proceedings of the IJCAI-95 Workshop on Entertainment and AI/ALife*, 1995.

[5] J. Bruce and M. Veloso, "Fast and accurate vision-based pattern detection and identification," in *Proceedings of the IEEE International Conference on Robotics and Automation*, Taiwan, May 2003.

[6] M. Bowling, B. Browning, and M. Veloso, "Plays as effective multiagent plans enabling opponent-adaptive play selection," in *International Conference on Automated Planning and Scheduling*, 2004.

[7] B. Browning, J. R. Bruce, M. Bowling, and M. Veloso, "STP: Skills tactics and plans for multi-robot control in adversarial environments," in *Journal of System and Control Engineering*, 2005.

[8] J. R. Bruce and M. Veloso, "Safe multi-robot navigation within dynamics constraints," *Proceedings of the IEEE*, vol. 94, pp. 1398–1411, July 2006.

[9] M. Veloso, P. Stone, and M. Bowling, "Anticipation as a key for collaboration in a team of agents: A case study in robotic soccer," in *Proceedings of SPIE Sensor Fusion and Decentralized Control in Robotic Systems II*, vol. 3839, 1999.

[10] B. P. Gerkey and M. J. Mataric, "A formal analysis and taxonomy of task allocation in multi-robot systems," *International Journal of Robotics Research*, vol. 23, no. 9, pp. 939–954, 2004.

[11] E. Uchibe, T. Kato, M. Asada, and K. Hosoda, "Dynamic task assignment in a multiagent/multitask environment based on module conflict resolution," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2001, pp. 3987–3992.

[12] A. D'Angelo, E. Menegatti, and E. Pagello, "How a cooperative behavior can emerge from a robot team," in *Proceedings of the Symposium on Distributed Autonomous Robotic Systems*, 2004.

[13] R. B. Tilove, "Local obstacle avoidance for mobile robots based on the method of artificial potentials," *General Motors Research Laboratories, Research Publication GMR-6650*, September 1989.