Motion Control in Dynamic Multi-Robot Environments

Michael Bowling **Manuela Veloso**
mhb@cs.cmu.edu mmv@cs.cmu.edu

Computer Science Department
Carnegie Mellon University
Pittsburgh, PA 15213-3890

Abstract

All mobile robots require some form of motion control in order to exhibit interesting autonomous behaviors. This is even more essential for multi-robot, highly-dynamic environments, such as robotic soccer. This paper presents the motion control system used by CMUnited-98, the small-size league champion at RoboCup-98. The team consists of five robots that aim at achieving specific goals while navigating in a limited space shared with the five other opponent robots. We introduce our motion control algorithm, which allows a general differential-driven robot to accurately reach a target point with a desired orientation in an environment with multiple moving obstacles. We describe how the features of our motion controller help to build interesting and robust behaviors. We also briefly compare our system to other motion control techniques and include descriptions and illustrations of the performance of our fully-implemented motion control algorithm.

1 Introduction

For any robotic system motion control is essential to building robust and interesting behavior. This is even more important for multi-robot systems that need to build team behaviors on top of individual behaviors. An example of such a system is robotic soccer. Here, a team of robots must coordinate their actions to push the ball into their opponents' goal. This is complicated by not only the opponent agents trying to prevent this from occurring, but also by the highly dynamic environment. This highly dynamic environment makes many traditional motion planning algorithms impractical since the environment changes before the planner can even finish its path.

This paper examines the motion control algorithm used in CMUnited-98 [Veloso *et al.*, 1999]. This team competed in RoboCup '98 in Paris in the small-size robot league. The team won four of its five games and was the league champion for the second straight year. A great deal of the success of the team can be attributed to the motion control algorithms. It not only made direct contributions by providing smooth and robust motion, but its features allowed us to build powerful individual and team behaviors. In section 2, we give a brief overview of the architecture of our team. It will describe the percepts and actuators available to the motion controller. In section 3, we describe the details of the motion control algorithm. In section 4, we describe how the high-level attacking behaviors effectively made use of our algorithm. Finally, in section 5 we will discuss related work in this area.

2 Team Architecture

The CMUnited-98 small-size robot team is a complete, autonomous architecture composed of the physical robots, a global video camera over-looking the playing field, and several clients as the minds of the small-size robot players. Fig. 1 sketches the building blocks of the architecture. The motion controller resides in the individual client modules and bridges the gap between the output of the vision processing system and the robots' motors.

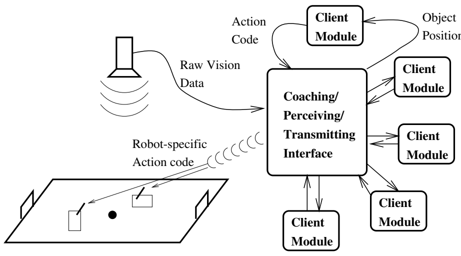


Figure 1: The CMUnited architecture with global perception and distributed reaction.

The vision system provides the input to the motion controller. Since it overlooks the entire field, it can provide a complete view of the world. Our image processing algorithm reliably detects and tracks the position and orientation of our five robots, the position of the five opponents, and the position of the ball. Additionally it uses a Kalman-Bucy filter to provide a reasonably accurate prediction of the ball's velocity, both speed and direction. This information is computed and passed to the client modules approximately every thirtieth of a second.

The output is the motion parameters for the physical robot. The robots have two motors and use differential drive for movement. The client modules, specifically the motion controller, sends the desired wheel velocities to its physical robot using radio communication. The radio communication supports five robots each receiving over twenty commands every second. Additionally, there is no local feedback mechanism on the robots. All control is done using only visual feedback through the motion controller.

A summary of the issues the motion controller must address are given below.

- Ten moving robots ($\leq$ 12cm diameters) on a small walled field (152.5cm by 274cm). Robots are moving at speeds close to one meter per second.
- Vision system providing 30 frames per second. Radio communication able to support approximately 20 commands per second for each of the five robots.
- Must be able to push the ball towards a particular point, specifically towards a goal (50cm wide). It must also be able to intercept a moving ball and avoid fast moving obstacles.

3 Motion Control Algorithm

The goal of our motion control algorithm is to be as fast as possible while remaining accurate and reliable. This is challenging due to the lack of feedback from the motors, forcing all control to be done using only visual feedback. Our motion control algorithm is robust. It addresses stationary and moving targets with integrated obstacle avoidance. The algorithm makes effective use of the prediction of the ball's trajectory provided by the Kalman-Bucy filter.

We achieve this motion control functionality by a reactive control mechanism that directs a differential drive robot to a target configuration. The mechanism is based on CMUnited-97's motion control [Veloso *et al.*, 1998; Han and Veloso, 1998], but includes a number of major improvements. The target configuration for the motion planner has been extended. The target configuration includes: (i) the *Cartesian position*; and (ii) the *direction* that the robot is required to be facing when arriving at the target position. Obstacle avoidance is integrated into this controller. Also, the target configuration can be given as a function of time to allow for the controller to reason about intercepting the trajectory of a moving target.

3.1 Differential Drive Control for Position and Direction

We begin with some basic control rules. The rules are a set of reactive equations for deriving the left and right wheel velocities, v_l and v_r, in order to reach a target position, (x^*, y^*):

$$\begin{aligned} \Delta &= \theta - \phi \qquad (1) \\ (t, r) &= (\cos^2 \Delta \cdot \mathrm{sgn}(\cos \Delta), \sin^2 \Delta \cdot \mathrm{sgn}(\sin \Delta)) \\ v_l &= v(t - r) \\ v_r &= v(t + r), \end{aligned}$$

where θ is the direction to the target point (x^*, y^*), ϕ is the robot's orientation, and v is the desired speed (see Fig. 2(a))[1]. A few aspects of these equations deserve explanation. The use of $\sin^2$ and $\cos^2$ restricts the values $(t \pm r)$ to the interval $[0, 1]$, which bounds the magnitude of the computed wheel velocities by v. These equations also do not necessarily drive the robot forward, possibly driving the robot backwards towards the target.

We extend these equations for target configurations of the form (x^*, y^*, ϕ^*), where the goal is for the robot to reach the specified target point (x^*, y^*) while facing the direction ϕ^*. This is achieved with the following adjustment:

$$\theta' = \theta + \min\left(\alpha, \tan^{-1}\left(\frac{c}{d}\right)\right),$$

where θ' is the new target direction, α is the difference between our angle to the target point and ϕ^*, d is the distance to the target point, and c is a clearance parameter (see Fig. 2(b).) This will keep the robot a distance c from the target point while it is circling to line up with the target direction, ϕ^*. This new target direction, θ', is now substituted into equation 1 to derive wheel velocities. An example trajectory using these equations is shown in Figure 3 (a).

In addition to our motion controller computing the desired wheel velocities, it also returns an estimate of the time to reach the target configuration, $\hat{T}(x^*, y^*, \phi^*)$. This estimate is a crucial component in our robot's strategy. It is used both in high-level decision making, and for low-level ball interception, which is described later in this section. For CMUnited-98, $\hat{T}(x^*, y^*, \phi^*)$ is computed using a very simple linear function of d, α, and Δ:

$$\hat{T}(x^*, y^*, \phi^*) = w_d d + w_\alpha \alpha + w_\Delta \Delta.$$

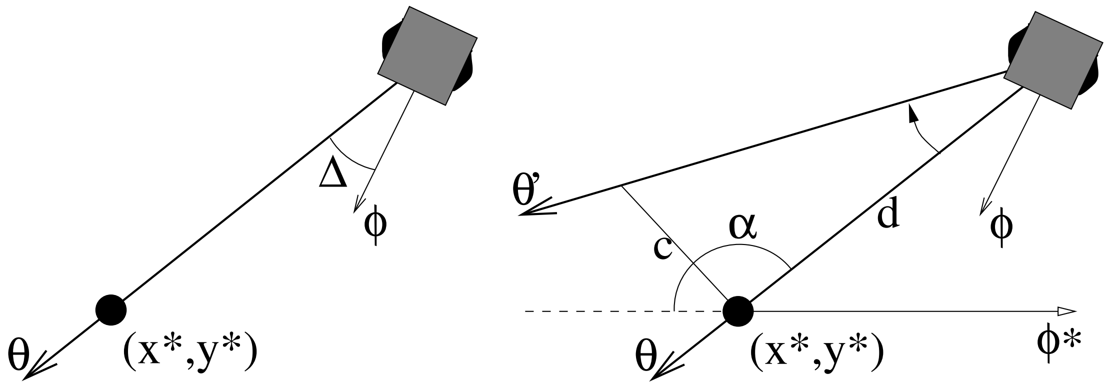
The weights were set by simple empirical measurements. w_d is the inverse of the robot's translational speed; w_Δ is the inverse of the robot's rotational speed; and w_α is the inverse of the speed of the robot when traversing a circle of radius, c. It is interesting to note that even this crude time estimate can be incredibly useful for building more complex behaviors, which are discussed later in this paper.

3.2 Obstacle Avoidance

Obstacle avoidance was also integrated into the motion control. This is done by adjusting the target direction of the robot based on any immediate obstacles in its path. This adjustment can be seen in Fig. 4.

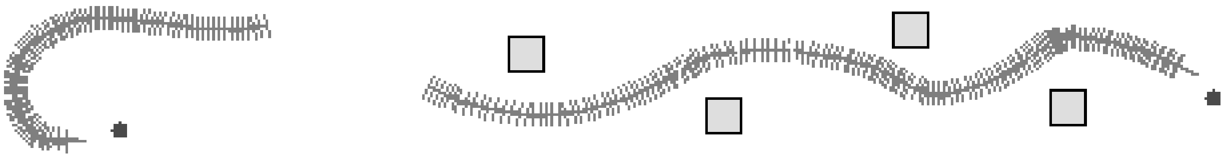
If a target direction passes too close to an obstacle, the direction is adjusted to run tangent to a preset allowed

[1] All angles are measured with respect to a fixed coordinate system.



(a) (b)

Figure 2: (a) The parameters used to reach a target configuration (x^*, y^*), without a specified target orientation. (b) The adjustment of θ to θ' to reach a target configuration of the form (x^*, y^*, ϕ^*).



(a) (b)

Figure 3: Example trajectories. (a) This illustrates reaching a point with a specific orientation. The target point is the position of the ball, and the specified orientation is to the right. (b) An example trajectory illustrating obstacle avoidance.

clearance for obstacles. Since the motion control mechanism is running continuously, the obstacle analysis is constantly replanning obstacle-free paths. This continuous replanning allows for the robot to handle the highly dynamic environment and immediately take advantage of short lived opportunities. Figure 3 (b) shows an example trajectory.

This technique can be viewed as a path planner using only a one-step lookahead. Hence, it sacrifices completeness for the performance needed to handle the dynamic environment. Section 5 will briefly compare this technique with traditional path planning.

3.3 Moving Targets

One of the real challenges in robotic soccer is to be able to control the robots to intercept a moving ball. This capability is essential for a high-level ball passing behavior. CMUnited-98's robots successfully intercept a moving ball and several of their goals in RoboCup-98 were scored using this capability.

This interception capability is achieved as an extension of the control algorithm to aim at a stationary target. Fig. 5(a) illustrates the control path to reach a stationary target with a specific direction, using the control mechanism described above. Our extension allows for the target configuration to be given as a function of time, where $t = 0$ corresponds to the present,

$$f(t) = (x^*, y^*, \phi^*).$$

At some point in the future, t_0, we can compute the target configuration, $f(t_0)$. We can also use our control rules for a stationary point to find the wheel velocities and estimated time to reach this hypothetical target as if it were stationary. The time estimate to reach the target then informs us whether it is possible to reach it within the allotted time. Our goal is to find the nearest point in the future where the target can be reached. Formally, we want to find,

$$t^* = \min\{t > 0 : \hat{T}(f(t)) \leq t\}.$$

After finding t^*, we can use our stationary control rules to reach $f(t^*)$. In addition we scale the robot speed so to cross the target point at exactly t^*.

Unfortunately, t^*, cannot be easily computed within a reasonable time-frame. We approximate this value, t^*, by discretizing time with a small time-step. We then find the smallest of these discretized time points that satisfies our estimate constraint. An example of this is shown in Fig. 5(b), where the goal is to hit the moving ball.

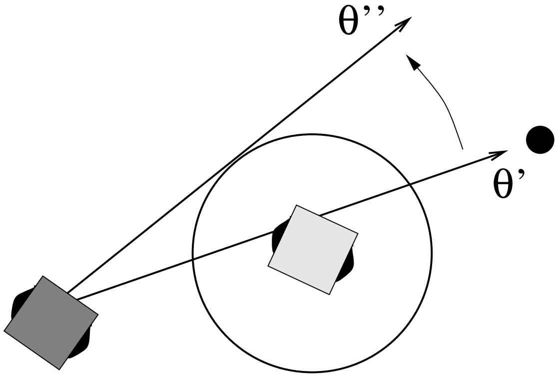


Figure 4: The adjustment of θ' to θ'' to avoid immediate obstacles.

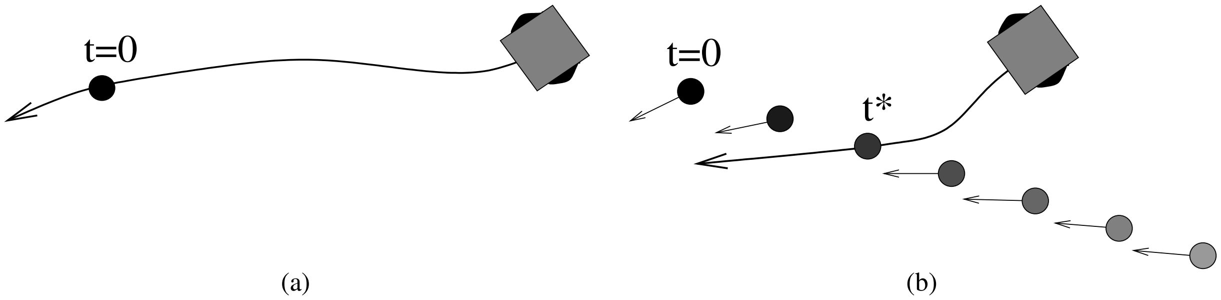


Figure 5: (a) Control for a stationary target. (b) Control for a moving target.

The target configuration as a function of time is computed using the ball's predicted trajectory. Our control algorithm for stationary points is then used to find a path and time estimate for each discretized point along this trajectory, and the appropriate target point is selected.

4 Using Motion Control

We have described how our motion controller computes the wheel velocities to reach a target configuration, (x^*, y^*, ϕ^*), which may even be a function of time, $f(t)$. It is the responsibility of individual and team behaviors to select the appropriate target configurations for each of the robots. The features of our motion controller often simplify this problem. We will examine how these features help build two attacking behaviors, shooting and passing. Also, we will show how it contributes to our team attacking behavior, which involves a decision theoretic action selection.

4.1 Individual Behaviors: Shooting and Passing

We first developed individual behaviors for passing and shooting. For both behaviors the positional portion of the target position is the ball's position, since the goal is to push the ball. Additionally, we can use the ball's predicted trajectory to make the position a function of time, according to the trajectory.

The directional portion of the target configuration determines where the ball is pushed. The passing behavior specifies a direction that is a small amount in front of the designated receiver. For shooting, a more complex target direction is computed. Simply pushing the ball towards the center of the goal will do nothing to avoid pushing the ball into the goalie. Instead, we want to push the ball towards the largest unblocked portion of the opponent's goal. This is done by selecting the largest unblocked angular section of the goal and aiming for the angle that bisects it. Figure 6 illustrates the selected target configuration to achieve passing and shooting.

4.2 Team Behavior: Decision Theoretic Action Selection

Given the individual behaviors, we must select an active agent, the agent that will go to the ball, and an appropriate behavior, passing or shooting. This is done by a decision theoretic analysis that uses a single step look-ahead. With n agents there are n^2 choices of actions involving shooting or a pass to another agent followed by that agent shooting. An estimated probability of success for each pass and shot is computed along with the time estimate to complete the action, which is provided by the motion controller. With

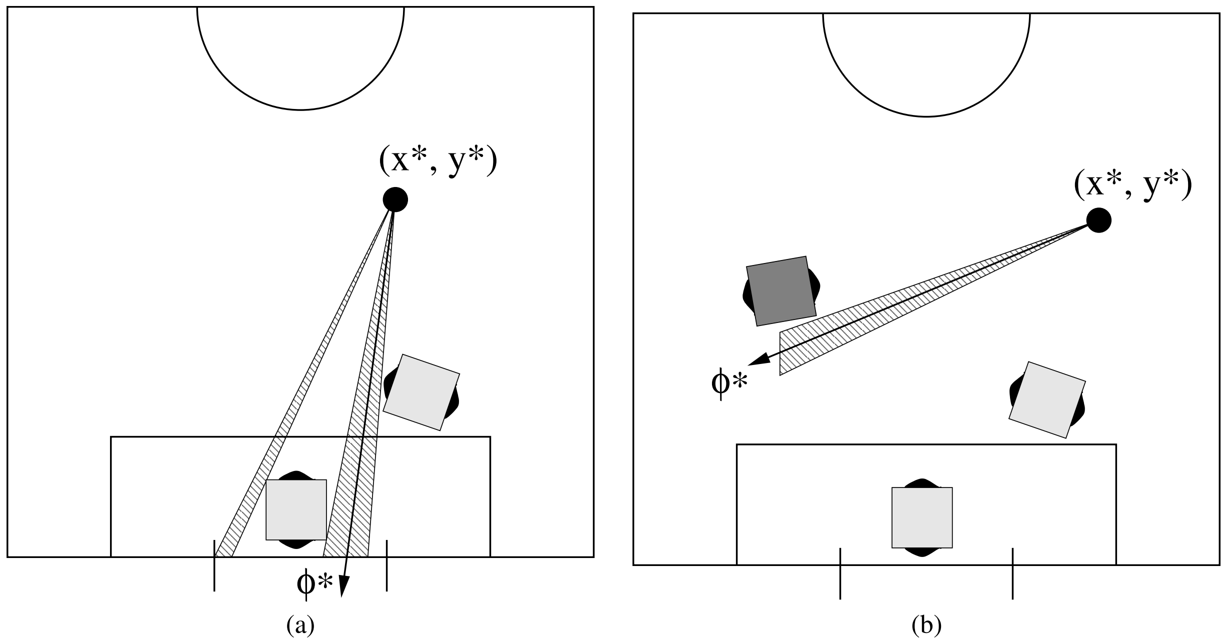


Figure 6: An example of possible aiming positions given the position of the ball and two opponents. The largest angle is chosen and ϕ^* is the bisection of the angle.

these estimates, a value for each action is computed,

$$\text{Value}_{\text{pass}} = \frac{\text{Pr}_{\text{pass}}\text{Pr}_{\text{shoot}}}{\hat{T}(f_{\text{pass}})} \qquad \text{Value}_{\text{shoot}} = \frac{\text{Pr}_{\text{shoot}}}{\hat{T}(f_{\text{shoot}})}$$

The action with the largest value is selected, which determines both the active agent and its behavior. Table 1 illustrates an example of the values for the selection considering two attackers, 1 and 2. CMUnited- 98 uses a heuristic function to estimate the success probabilities of passing and shooting.

It is important to note that this action selection is occurring on each iteration of control, approximately 30 times per second. The probabilities of success, estimates of time, and values of actions, are being continuously recomputed. This allows for quick changes of actions if shooting opportunities become available or collaboration with another agent appears more useful.

5 Related Work

An alternative to our purely reactive algorithm is to use a complex motion planning algorithm. A number of these algorithms are summarized by Latombe [Latombe, 1991]. These techniques find complete obstacle free paths, but yet have difficulties in the robotic soccer domain. Since the environment is highly dynamic with the obstacles constantly moving, planned paths would need to be constantly reevaluated. Also, path planning often needs to be done for a large number of proposed trajectories before the high-level action can even be selected. These traditional algorithms are simply too slow for the continuous real-time execution that is demanded in robot soccer.

Another approach to motion control [Rowstron *et al.*, 1998] uses a reactive mechanism with fast hardware-supported feedback, via motor encoders and on-board sensors. This makes use of a slower decision loop to provide high-level commands, and a fast control loop to perform these commands. The control loop uses the motor encoders to perform accurate movements and on-board sensing for immediate obstacle avoidance and ball manipulation. This was successfully used by CURF (Cambridge University Robot Football) in RoboCup '98. One drawback to this technique is that the fast control loop does not have access to the complete sensors (i.e. the global view of the field), and short lived opportunities, which may not be recognized by the local sensors, often cannot be exploited . ISpace [Hashimoto *et al.*, 1998], another team that competed in RoboCup '98, used a similar technique, but due to onboard vision could possibly overcome this drawback.

Additionally, there are also other reactive control systems for remotely controlled robots[Santos-Victor and Carreira, 1998].

6 Conclusion

We've described the motion control algorithm used in CMUnited-98. The algorithm incorporates obstacle avoidance, and has an extended target configuration that includes orientation and can be given as a function of time. In addition to the details of the algorithm, we also described how its features simplifies the building of individual and team

Attacker	Action	Probability of Success Pass	Shoot	Time(s)	Value
1	Shoot	–	60%	2.0	0.30
1*	Pass to 2	60%	90%	1.0	0.54
2	Shoot	–	80%	1.5	0.53
2	Pass to 1	50%	40%	0.8	0.25

Table 1: Action choices and computed values are based on the probability of success and estimate of time. The largest-valued action (marked with the *) is selected.

behaviors. The system was integral part of the team's success.

References

[Han and Veloso, 1998] Kwun Han and Manuela Veloso. Reactive visual control of multiple non-holonomic robotic agents. In *Proceedings of the International Conference on Robotics and Automation*, Belgium, May 1998.

[Hashimoto *et al.*, 1998] Tomomi Hashimoto, Toru Yamaguchi, and Hideki Hashimoto. Multi-vision RoboCup system using ISpace. In Minoru Asada, editor, *Proceedings of the Second RoboCup Workshop*, pages 527–537, 1998.

[Latombe, 1991] Jean-Claude Latombe. *Robot Motion Planning*. Kluwer Academic Publisher, 1991.

[Rowstron *et al.*, 1998] A. Rowstron, B. Bradshaw, D. Crosby, T. Edmonds, S. Hodges, A. Hopper, S. Lloyd, J. Wang, and S. Wray. CURF: Cambridge university robot football team. In Minoru Asada, editor, *Proceedings of the Second RoboCup Workshop*, pages 503–509, 1998.

[Santos-Victor and Carreira, 1998] José Santos-Victor and Carlos Carreira. Vision based remote control of cellular robots. *Journal of Robotics and Autonomous Systems*, 1998.

[Veloso *et al.*, 1998] Manuela Veloso, Peter Stone, Kwun Han, and Sorin Achim. CMUnited: A team of robotic soccer agents collaborating in an adversarial environment. In Hiroaki Kitano, editor, *RoboCup-97: The First Robot World Cup Soccer Games and Conferences*. Springer Verlag, Berlin, 1998.

[Veloso *et al.*, 1999] Manuela Veloso, Michael Bowling, Sorin Achim, Kwun Han, and Peter Stone. The CMUnited-98 champion small robot team. In Minoru Asada and Hiroaki Kitano, editors, *RoboCup-98: Robot Soccer World Cup II*. Springer Verlag, Berlin, 1999.