The Cornell Robocup Team

Raffaello D'Andrea[1], Tamás Kalmár-Nagy[2], Pritam Ganguly[2] and Michael Babish[3]

[1] Sibley School of Mechanical and Aerospace Engineering
Ithaca, NY 14853, USA
rd28@cornell.edu

[2] Department of Theoretical and Applied Mechanics
Ithaca, NY 14853, USA
{nagy,pritam}@tam.cornell.edu

[3] Department of Computer Science
Ithaca, NY 14853, USA
mjb33@cornell.edu

Abstract. This paper describes the Cornell Robocup Team, which won the RoboCup-2000 F180 championship in Melbourne, Australia. The success of the team was due to electro-mechanical innovations (omni-directional drive and a dribbling mechanism) and the control strategies that rendered them effective. As opposed to last year's "role-based" strategy, a "play-based" strategy was implemented, which allowed us to make full use of the robot capabilities for cooperative control.

1 Introduction

The RoboCup competition is an excellent vehicle for research in the control of complex dynamical systems. From an educational perspective, it is also a great means for exposing students to the systems engineering approach for designing, building, managing, and maintaining complex systems.

In an effort to shift the current emphasis of the competition away from simple strategies to more complicated team-based strategies, the main emphasis of this year's team was to play a controlled game. In other words, in a game without ball control, effective strategies essentially consist of overloading the defensive area during a defensive play (the so called "catenaccio" in human soccer, a strategy that is very effective, if not extremely dull and frustrating for the spectators), and shooting the ball towards open space or the goal area in the opponent's half during offensive plays. This was, in fact, the simple role based strategy adopted by our championship team in 1999, which was shown to be extremely effective. In order to bring coordination and cooperation to the RoboCup competition, the Cornell team developed two electro-mechanical innovations (omni-directional drive and dribbling, described in Section 2) *and* the associated control strategies that rendered them effective (described in Sections 3 and 4).

The key system concept employed by the Cornell RoboCup team is that of *hierarchical decomposition*, the main idea of which is depicted in Figure 1.

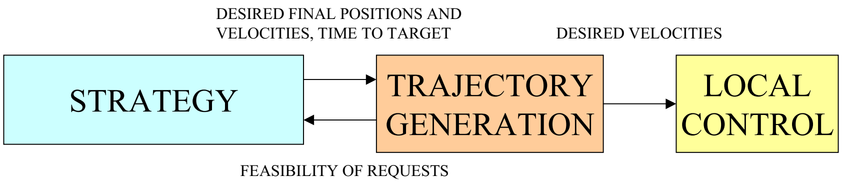

Fig. 1. Hierarchical decomposition

At the lowest level is the local control, which physically occurs on the robots; local feedback loops regulate the wheel velocities about the desired wheel velocities commanded by a centralized computational unit (a workstation). The desired wheel velocities are generated by the Trajectory Generation block; these velocities are generated in such a way to ensure that the robots are physically capable of executing the maneuvers (as limited by power, friction, and stability constraints). The Trajectory Generation block uses desired final position, velocity, and time to target information, generated by the Strategy block, to calculate the desired robot trajectories; feasibility information is sent back to the Strategy block to ensure that only executable commands are sent to the actual robots. The reader is referred to [3] for details on how this general approach can be used to control complex systems.

This decomposition structure renders the real-time vehicle coordination problem tractable. In particular, the high level strategy (described in Section 4) does not need to consider the details of vehicle motion, except for feasibility information; if a desired play is not deemed feasible (for example, leading a pass to another robot is not deemed feasible because the receiving robot cannot execute the interception), alternate strategies are implemented.

The RoboCup competition is relatively mature, with many teams competing at a high level; we have thus decided to emphasize in this paper aspects of our system that we feel are innovative, and forgo a detailed explanation of the sub-systems that are well understood and documented in previous RoboCup publications; interested readers are referred to [5], [1], and [10]. The rest of the paper is organized as follows: in Section 2, a brief description of omni-directional drive and dribbling is presented, followed by an in-depth discussion of the Trajectory Generation scheme in Section 3. A description of team strategy is presented in Section 4.

2 Omni-Directional Drive and Dribbling

A very effective way of position control was implemented by the Cornell Robocup Team in the 2000 competition. The omni-directional drive consists of three pairs of wheels (see Figure 2). Each pair of wheels has an active degree of freedom in the direction of the rotation of the motor and a passive one perpendicular

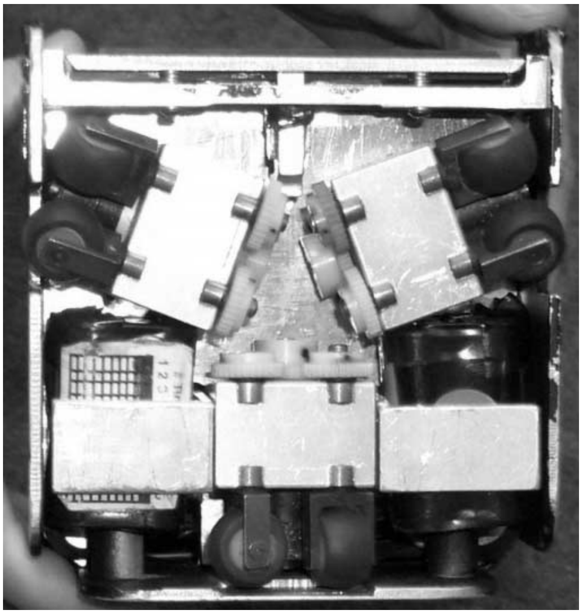

Fig. 2. Bottom view, omni-directional drive

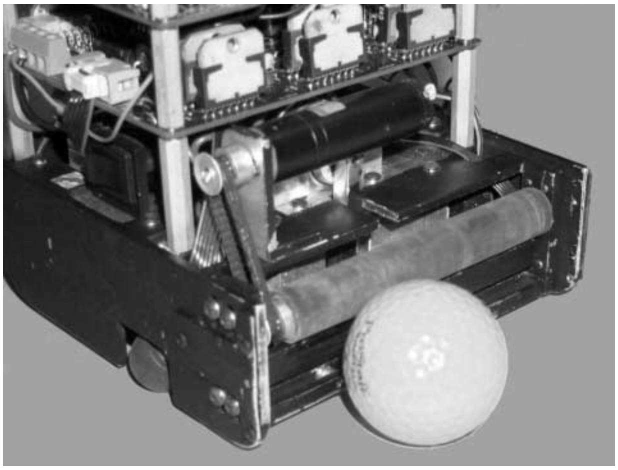

Fig. 3. Dribbling mechanism

to it (see [4], for example). Since the drive directions are pair-wise linearly independent, the translation and the rotation of the robot can independently be controlled by a judicious choice of drive velocities.

The dribbling mechanism is a rotating bar with a latex cover placed just above the kicking mechanism (see Figure 3). Upon contact with the ball, the rotation of the bar imparts a backward spin on the ball; the bar is strategically placed such that the net component of the force on the ball is always towards the robot, and this is achieved without violating the 20 % convexity rule.

The omni-directional drive, coupled with the dribbling mechanism, greatly increases the *potential* capabilities of the robots (we stress the word potential, since it is not obvious that a real-time control strategy can be developed to fully utilize these features). The main capability which is rendered possible by this combination is the effective receiving of passes, which must be central to any sophisticated team-based strategy.

3 Trajectory Generation and Control

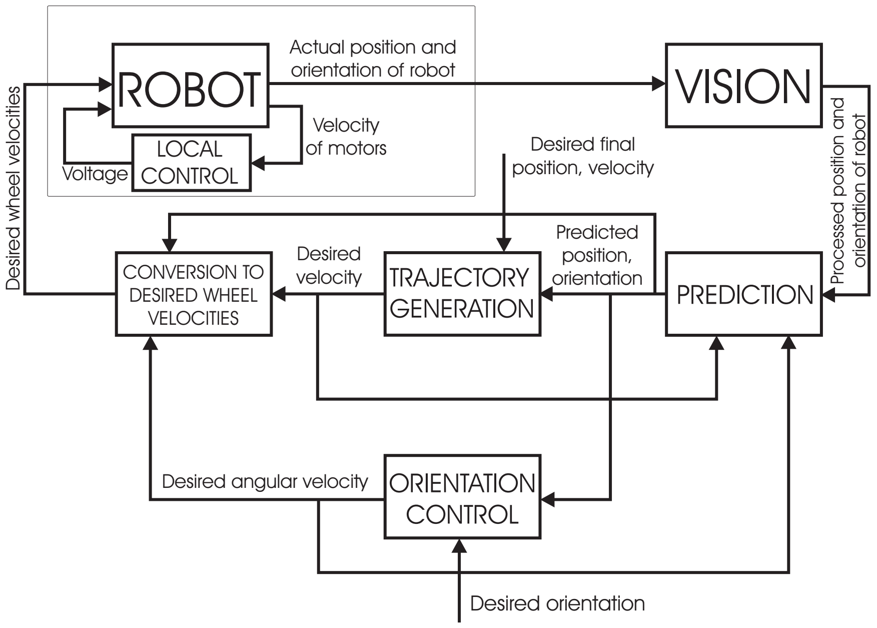
ROBOT
Actual position and orientation of robot
VISION
LOCAL CONTROL
Voltage
Velocity of motors
Desired wheel velocities
Desired final position, velocity
Processed position and orientation of robot
CONVERSION TO DESIRED WHEEL VELOCITIES
Desired velocity
TRAJECTORY GENERATION
Predicted position, orientation
PREDICTION
Desired angular velocity
ORIENTATION CONTROL
Desired orientation

Fig. 4. Block diagram representation of trajectory generation

The overall trajectory generation and control scheme for one robot is depicted in Figure 4. The Vision block feeds the calculated position and orientation of the robot to the Prediction block, which estimates the position and orientation of the robot *in the robot temporal frame* based on the vision data and the history of the commanded velocities. The Trajectory Generation block solves a relaxation of an optimal control problem to calculate the future robot velocity profile required to reach the prescribed final position and final velocity in either the shortest possible time, or in a prescribed amount of time using the least amount of control effort; a similar step occurs for the robot orientation. The dynamics of the motors and the robots are taken into account to ensure that the generated velocity profiles are feasible. This data is then converted to wheel velocities, which is sent to the robot via wireless communication. A local control loop on the robot regulates the actual wheel velocities about the desired wheel velocities. Note that the overall system feedback is achieved by solving an optimal control problem for every robot at every frame (approximately 30 times per second).

3.1 Feasible Trajectory Generation

As described in Section 1 and depicted in Figure 1, desired wheel velocities are calculated by the Trajectory Generation block based on desired final position and velocity information passed down by the Strategy block. Two types of trajectories are generated: minimum time and minimum control effort. Minimum time trajectories are called upon when a robot must move to a given location, with a given final velocity, as quickly as possible; minimum control effort trajectories are called upon when a robot must arrive at a given location, with a given final velocity, at a given time. A minimum control effort trajectory is clearly not feasible if the minimum time counterpart takes longer to execute.

The full equations of motion of an omni-directional vehicle are complex enough to render the real-time computation of optimal trajectories impossible with today's computational resources. In addition, it is desirable for feasible trajectories to be generated very quickly to allow high-level strategies to explore as many scenarios as possible. Our approach is based on relaxation: can a simplified, computationally tractable optimal control problem be solved whose solution yields feasible, albeit sub-optimal, trajectories? The omni-directionality of our robots permits a positive answer to the above question. The basic idea is to decouple the three degrees of freedom (two translational and one rotational), to solve an optimal control problem for each degree of freedom independently.

Each degree of freedom can be captured by the following differential equation:

$$\ddot{z} = \alpha u - \beta \dot{z}, \tag{1}$$

where z is the displacement or rotation, u is the effective voltage, and α and β are physical parameters which can be calculated from the physical characteristics of the robots and motors. Note that equation (1) captures, to a very high level of accuracy, the dynamics of a single motor driving a load [6]. The problem becomes a constrained boundary value problem when the initial position, initial velocity, final position, and final velocity are prescribed:

$$z(0) = z_0, \quad z(t_F) = z_F, \quad \dot{z}(0) = v_0, \quad \dot{z}(t_F) = v_F \tag{2}$$

subject to the physical constraint $|u| \leq u_{max}$. Since α and β are fixed, we can introduce new length and time scales and assume, without loss of generality, that $\alpha = \beta = u_{max} = 1$. In this non-dimensional form, the maximum achievable velocity has magnitude 1, achieved asymptotically when $\ddot{z} = 0$. We will thus assume that $|v_0| < 1$ and that $|v_F| < 1$.

When performing a minimum control effort maneuver, the decoupling of the degrees of freedom does not pose a problem, since a time of execution is specified. When performing a minimum time maneuver, however, the degrees of freedom will invariably take different amounts of time to execute. This is easily solved by executing a minimum time trajectory for the *slowest* degree of freedom, and executing minimum control effort trajectories for the remaining degrees of freedom.

The minimum time and minimum control effort solutions are provided below. The amount of time required to calculate a trajectory for one robot was on the order of 1000 FLOPS (floating point operations), which is an extremely small number. In particular, as of the writing of this paper a typical desktop computer can execute upwards of 100 MFLOPS per second; at a sampling rate of 30 times per second, the computational resources required to calculate one robot trajectory amounts to less than 0.03 percent of the time allotted to execute one complete system cycle.

Minimum Time: The minimum time problem consists of finding a velocity profile $u(t)$ such that the boundary conditions in (2) are satisfied for as small a t_F as possible. It can be shown (see [7], for example), that the boundary value problem always has a solution, and that the voltage profile which minimizes time t_F consists of a piecewise constant $u(t)$ composed of two segments of magnitude 1. This type of control strategy is commonly referred to as "bang-bang" control. In particular, the following must be solved for U, t_1, and t_2:

$$\ddot{z} + \dot{z} = U, \;\; 0 < t \leq t_1 \tag{3}$$

$$\ddot{z} + \dot{z} = -U, \;\; t_1 < t \leq t_1 + t_2 = t_F, \tag{4}$$

where U is either 1 or -1. Subject to the constraints in (2), we can solve the above to yield

$$t_1 = t_2 - \frac{C}{U}, \;\; \exp(t_2 - \frac{C}{U}) = \frac{v_0 - U}{(v_F + U)\exp(t_2) - 2U}, \tag{5}$$

where $C := z_0 + v_0 - z_F - v_F$. The second expression is equivalent to a quadratic equation in $\exp(t_2)$, with solution:

$$\exp(t_2) = \frac{1 \pm \sqrt{1 - \exp(C/U)(1 + v_F/U)(1 - v_0/U)}}{1 + v_F/U}. \tag{6}$$

Four different cases must be considered (two for U, and two for equation (6)). A solution with $t_1 > 0$ and $t_2 > 0$ always exists; the one which yields a minimum value of $t_1 + t_2$ is chosen. Note that once U and t_1 are determined, a closed form solution for $\dot{z}(t)$ can be constructed for $0 \leq t \leq t_1 + t_2$. The computational effort required to solve for t_1, t_2, and U is less than 200 FLOPS.

Minimum Control Effort: The minimum control effort problem is based upon the minimum time solution. In particular, let $t_F^* = t_1^* + t_2^*$ denote the time required to execute the minimum time trajectory. It can be shown that for all $t_F > t_F^*$, there exists $t_1 > 0$, $t_2 > 0$, and $0 < \gamma \leq 1$ such that equations (5) are satisfied with $t_F = t_1 + t_2$ and $|U| = \gamma$. In other words, a bang-bang trajectory with lower control effort can always be constructed such that the desired state is reached in the prescribed amount of time. Unlike the minimum control effort problem, a closed form solution cannot be constructed; a simple iterative scheme can be used, however, to solve for t_1 and t_2 which executes in less than 1000 FLOPS.

A second scheme for generating fixed time trajectories can be employed if 1000 FLOPS is too computationally expensive. The equations of motion can be expressed in first order form as follows $\dot{x} = Ax + Bu$, where $x = (z, \dot{z})$. It can readily be shown (see [2], for example), that for a given t_F, the control strategy $u(t)$ which minimizes the 2-norm of u is given by

$$u(t) = B^* \exp(A(t_F - t))W^{-1}(x_F - \exp(At_F)x_0), \tag{7}$$

where

$$W = \int_0^{t_F} \exp(A\tau)BB^* \exp(A^*\tau)d\tau. \tag{8}$$

We can then solve for x as follows:

$$x(t) = \exp(At)x(0) + \int_0^t \exp(A(t-\tau))Bu(\tau)d\tau. \tag{9}$$

The above expression can be solved analytically for $x(t)$; the second component of $x(t)$ yields the desired velocity profile $\dot{z}(t)$. The computational effort required to solve for $\dot{z}(t)$ is less than 100 FLOPS. Note however, that the minimum 2-norm trajectory need not be feasible; in particular, there may exist a t such that $|u(t)| > 1$, which violates the actuator constraints. This was not found to be a problem in practice, however, and was the algorithm used in the competition.

3.2 Latency Determination

An integral component of the trajectory control is the Prediction block; this block constructs the optimal estimate of the robot position and orientation in the robot temporal frame. In order to make this estimate, the overall latency of the system must be known precisely. A simple and robust method was devised to accurately estimate the overall system latency, described below.

Consider the angular control of a robot:

$$\dot{\theta}(t) = v_\theta(t), \tag{10}$$

where $v_\theta(t)$ is the commanded angular rate. When the commanded rate is set to a multiple of the observed angular position, the following expression is obtained:

$$\dot{\theta}(t) = -k\theta(t-\tau), \tag{11}$$

where τ is the overall system latency. Consider solutions of the form $\theta(t) = ce^{\lambda t}$, where λ is a complex number. This yields the following characteristic equation:

$$\lambda + ke^{-\lambda\tau} = 0 \tag{12}$$

The locations of the (infinitely many) roots λ determine system stability (negative real parts correspond to stable behavior). The motion is oscillatory (with a decaying or growing envelope) and the change from stable to unstable behavior will take place as a root crosses the imaginary axis, i.e., $\lambda = i\omega$ (see [9], for example). This yields

$$\omega = k, \quad \tau = \frac{\pi}{2k}. \tag{13}$$

This leads to the following method for estimating the system latency: find the value of k for which oscillations neither grow nor decay. The observed frequency of oscillation should be k radians per second, and the system latency $\frac{\pi}{2k}$. For our system, it was indeed observed that the frequency of oscillation was equal to k, validating the approach. The system latency was calculated to be 0.12 seconds.

3.3 Obstacle Avoidance

The Cornell team was able to implement effective obstacle avoidance mainly due to the hierarchical decomposition of overall robot control into trajectory generation and the higher level strategy algorithms. The algorithm we used was developed for circular obstacles (the robots are approximated by their circumscribing circle), and is similar to the one presented in [8]. Even though the obstacles are treated as static, the algorithm works well since it is run every frame. Extension of this algorithm to include instantaneous velocities of obstacles (both friendly and opponent) is straightforward.

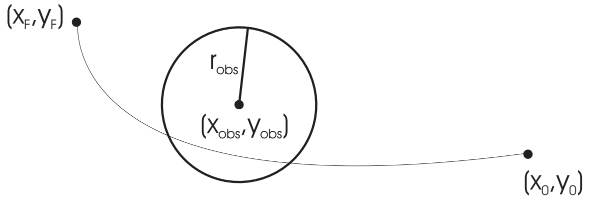

Fig. 5. Trajectory intersecting an obstacle

If the trajectory from the starting point (x_0, y_0) to the final point (x_F, y_F) intersects a single circular obstacle, the path has to be modified (Figure 5). In this case, the algorithm generates two new paths, with via-points located on lines perpendicular to the tangents to the circle from the starting point (Figure 6). The best path is then chosen (for example, the faster one). A straightforward extension of the above algorithm is used for multiple obstacles.

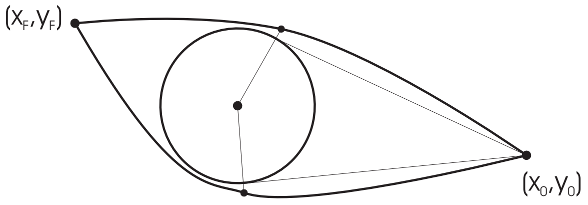

Fig. 6. Avoiding a single obstacle.

4 Strategy

The effectiveness of Cornell's trajectory generation system made it much easier to develop the high level strategy. The design only needs to choose a destination for each robot, and trust that the low level control can move the robot towards that destination. This year, Cornell abandoned its "role-based" strategy in favor of a "play-based" strategy. The new system is a finite state automaton based on a playbook, as depicted in Figure 7. At any point during a game, the robots are involved in a specific play (such as Offense, Defense, or PassPlay) and have specific goals to carry out. The main advantage of this design is that it was extremely easy to develop and debug, and very easy to adapt during the competition when we discovered problems with our strategy. The main disadvantage is that it is dependent on human intuition for its strategy - the system had no way to adapt to unexpected situations. In the future we hope to develop a team with many different strategy options, and the ability to choose the best strategy based on the current opponent.

5 Conclusions

The proposed increase in the field size will greatly reward teams that implement effective team play. We strongly feel that the dribbling mechanism will greatly improve the quality of the game, and allow teams to effectively use sophisticated team-based strategies. The main benefit of the omni-directional drive mechanism is a simplification of the resulting control problem, which *greatly* reduces the computation required for generating nearly optimal trajectories, and thus free up computational resources for higher level control decisions; we do not feel, however, that it will be a necessary feature of future competitive teams. It is clear, however, that successful future teams must seriously address dynamics and control issues, such as estimation, coping with system latency, robustness, and optimal control; only by doing so can the full benefits of team play and cooperation be achieved.

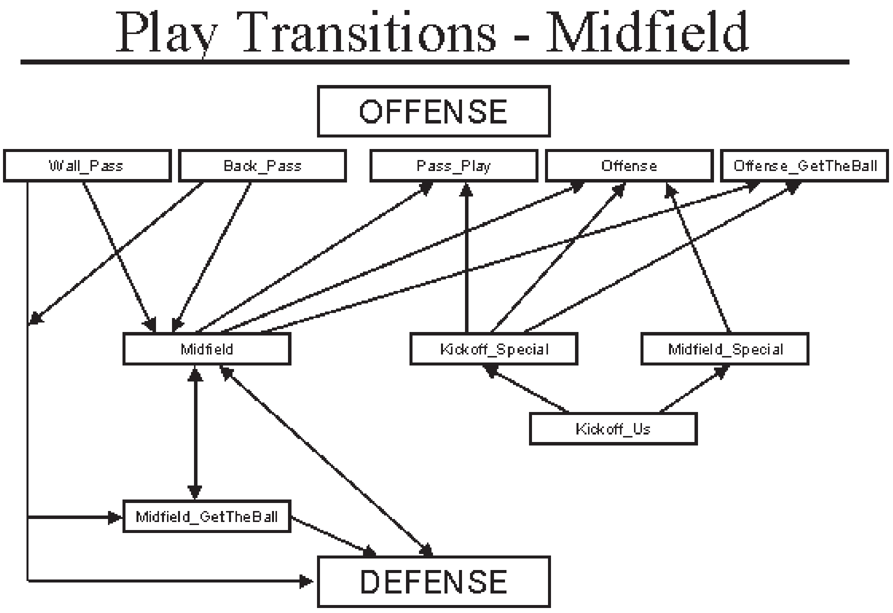

Fig. 7. Example of Playbook.

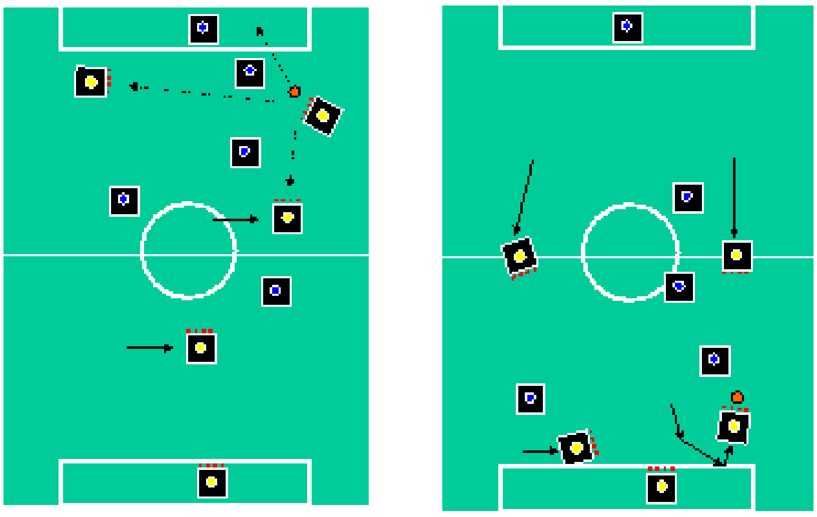

Fig. 8. Example of Plays. Left diagram, offensive play; right diagram, defensive play.

6 Acknowledgements

The lead author would like to thank all of the students involved in the RoboCup project at Cornell University, past and present, for their dedication to the project, and for their numerous accomplishments. We would also like to thank our many sponsors for their continuing support, in particular Jim Morgan at Applied Materials, John Swanson at ANSYS, Lockheed Martin, and United Technologies.

References

1. M. Asada and H. Kitano, editors. *RoboCup-98: Robot Soccer World Cup II.* Lecture Notes in Computer Science. Springer, 1999.
2. G. E. Dullerud and F. Paganini. *A Course in Robust Control Theory.* Springer, 2000.
3. E. Frazzoli,M. A. Dahleh and E. Feron. Robust Hybrid Control for Autonomous Vehicle Motion Planning. In *Conference on Decision and Control*, 2000.
4. M. Jung,H. Shim,H. Kim and J. Kim. The Miniature Omni-directional Mobile Robot OmniKity-I(OK-I). In *International Conference on Robotics and Automation*, volume 4, pages 2686-2691, 1999.
5. H. Kitano, editor. *RoboCup-97: Robot Soccer World Cup I.* Lecture Notes in Computer Science. Springer, 1998.
6. K. Ogata. *Modern Control Engineering.* Prentice-Hall, third edition, 1997.
7. D. A. Pierre. *Optimization Theory with Applications.* Dover, second edition, 1986.
8. Z. Shiller. Online sub-optimal obstacle avoidance. In *International Conference on Robotics and Automation*, volume 1, pages 335–340, 1999.
9. G. Stépán. *Retarded Dynamical Systems.* Longman Harlow, 1989.
10. M. Veloso, E. Pagello, and H. Kitano, editors. *RoboCup-99: Robot Soccer World Cup III.* Lecture Notes in Computer Science. Springer, 2000.