Turning Segways into Soccer Robots

Brett Browning, Jeremy Searock, Paul E. Rybski, and Manuela Veloso
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA USA
{brettb, prybski, mmv}@cs.cmu.edu, jsearock@andrew.cmu.edu

Abstract – **Recently, the company Segway LLC has released a dynamically balancing robot base, the Segway RMP, to complement their Segway scooters for human mobility. These robot bases provide exceptional robustness and capability with the unique feature of dynamic balancing at a human-size scale. We have addressed the challenge of using these Segway RMPs to build robots that are able to autonomously play soccer, building up upon our extensive previous work in this multi-robot research domain. This paper details our investigations towards developing an individually autonomous and capable robot. In particular, the focus is on the electro-mechanical mechanisms required to make a Segway RMP autonomous, able to sense its world, as well as manipulate and kick a soccer ball. In conjunction with the mechanisms required to make the robot physically capable, we detail our investigations into the control algorithms required to enable the robot to perceive, think, and act in real time for a dynamically changing world. While these techniques are applicable to many robot applications, dynamic balancing creates a number of unique challenges and opportunities that must be addressed. We examine these capabilities and limitations of the Segway and provide a detailed analysis of different mechanical and computational techniques to address these limitations. In this paper, we present empirical results examining the performance of our mechanisms and algorithms in the context of a fully functioning system.**

Keywords- Mobile Robots, Autonomy, Mobile robot design, Dynamic balancing robots, Robot soccer

1 INTRODUCTION

Unveiled in December of 2001, the Segway Human Transport (HT) presented an innovative new commercial product for human mobility. The HT's robustness, speed, and versatility make it an attractive transport option over moderate distances (a few kilometers). The two-wheel scooter provides a unique mode of transport due to its self-balancing mode of operation. In May of 2003, Segway released the Segway Robot Mobility Platform (RMP) to a group of researchers [15]. The Segway RMP is, in brief, a modified HT with a shortened handle bar, modified control software, and the addition of a steel table placed 50cm above the axle center to raise the center of mass of the platform. These combination of changes create an extensible robot base that has the robustness and capabilities of the HT, and can be controlled by a computer connected to the platform via a CAN Bus interface[1]. Figure 1 shows both platforms.

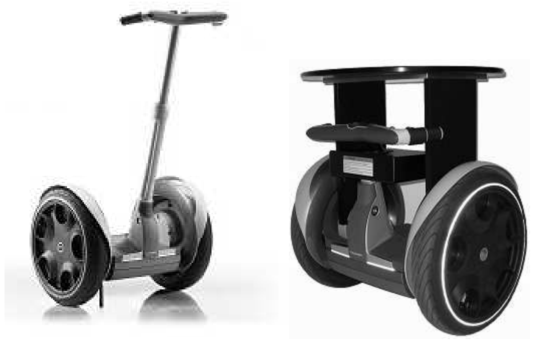


Figure 1. The Segway HT (on the left) and Segway RMP (on the right), which are commercially available from Segway LLC (*http://www.segway.com*).

The availability of the Segway RMP creates the possibility of developing robust capable robots that are able to operate in indoor and outdoor environments. Indeed, many researchers are now developing robots along these lines [18][19][20][21][22]. The work reported in this paper begins with this approach, but focuses on a unique challenge: creating Segway based robots that are able to play soccer. The eventual goal of this research is to create human-robot teams consisting of Segway riding humans and Segway RMPs that are able to perform complex tasks in dynamic,

[1] Controller Area Network (CAN) is a serial bus protocol commonly used in the automotive and micro-controller industries (e.g. *http://www.interfacebus.com/Design_Connector_CAN.html*)

adversarial environments [14]. Extensive research within the RoboCup robot soccer environment by the authors and others ([23][24]), demonstrates that robotic soccer provides a well-founded domain for investigating autonomous robot teams operating in such tasks. In this paper, we focus on the challenge of creating a single robot from a Segway RMP that is autonomous and individually capable of playing soccer. In particular, we focus on three challenges:

- Physical mechanisms to sense, compute, communicate, manipulate and kick the ball, and maintain safety,
- Algorithms to autonomously perceive, think, and act for soccer tasks
- How to integrate these two components together into a robust, real-time system

We have investigated several techniques to address these three challenges, deriving in part from prior research efforts [25][26]. This paper details these techniques, and their resulting performance and inherent limitations. Thus, this paper contributes towards the wider effort of understanding the science behind building capable autonomous robots for dynamic worlds.

The article is structured as follows. The following section describes the Segway RMP as it is commercially available, its physical capabilities and performance characteristics as well as the challenges that must be addressed to turn it into a working, autonomous robot. Section 3 details our investigations into physical mechanisms for addressing the robot soccer challenge. Complementing this, section 4 details the algorithms we have investigated to provide autonomy to the robot. Section **Error! Reference source not found.** describes the final system and its performance, while section 5 concludes the paper.

2 TURNING SEGWAY RMPS INTO ROBOTS

In this section, we describe the Segway RMP platform and its characteristics. This leads to a description of the challenges that must be overcome in order to turn an off-the-shelf Segway RMP into a robot platform.

2.1 The Segway RMP Robot

The Segway RMP is a dynamically balancing robot consisting of a base that contains the micro-controllers, batteries, DC motors and gearboxes that form the core of the robot. Its two wheels are arranged in a differential drive, or wheelchair, format. As there is no third point of support, the robot must dynamically balance to maintain stability. Steel plates mounted approximately 50cm above the axle height raise the center of mass of the platform making inverted pendulum control realizable at moderate control loop speeds of approximately 100Hz. Physically, the unmodified RMP weighs 64kg, stands approximately 76cm tall and can carry a 45kg payload [15]. Although in practice, the latter can be considerably exceeded without undue side effects [19]. The platform is capable of a top speed of 13km/h (~3.6m/s) over a range of 16km for nominal conditions and terrain. It also has a zero turning radius, and a small footprint given its size. For a commercially available mobile robot, the combination of speed, range, and payload is impressive and derives from the considerable engineering effort that went into making the Segway HT a viable commercial product for human use.

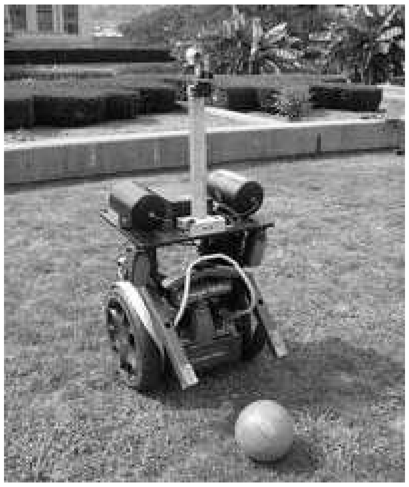


Figure 2. An earlier version of the soccer playing robot.

The Segway RMP runs internal control algorithms based on a Linear Quadratic Regulator (LQR - [1]) to maintain balance. The controller consists of two components: maintaining balance, and executing user velocity commands. Commands can be sent to the RMP via the exposed CAN Bus interface. A computer may send a variety of commands to the RMP via the CAN Bus and can also receiving state updates from the RMP's internal sensors. The RMP has a range of sensors for measuring the pose of the robot (roll, pitch, yaw), wheel velocity, motor currents and so on. The state information is available at a 100Hz update. Table 1 and Table 2 below list the available parameters.

Command	**Parameters**
Set Velocity	$<v,\omega>$
Kill	None
Set Gain Schedule	{*light, medium, heavy*}
Reset Integrators	None
Set Balance Mode	{*Balance, Tractor, Lock Tractor*}

Table 1. The available Segway RMP commands.

Sensor	**Symbol**	**Sensor**	**Symbol**
Pitch, roll, yaw	ϕ, θ, ψ	Forward Displacement	d_F
Pitch, roll, yaw rates	$d\phi/dt$, $d\theta/dt$, $d\psi/dt$	Gain Setting	g
Wheel Velocity (L/R)	v_L, v_R	Battery Charge	C
Wheel Displacement (L/R)	d_L, d_R	Balance Mode	m
Motor Currents (L/R)	I_L, I_R	Last Command	$<v, \omega>$

Table 2. Sensory state available from the Segway RMP at 100Hz.

2.2 Turning a Segway Into an Autonomous Robot

The Segway RMP provides a base platform from which to construct a robot. To create a complete robot, one must add:

- Independent computation
- External sensors
- A CAN Bus interface to the Segway RMP
- Additional actuators*
- External communications mechanisms*
- Electrical power for additional components

The items labeled with an * are optional and are not required to build a basic autonomous robot. To add these components, many of the engineering challenges faced with any robot platform must be overcome. However, given the size and payload of the RMP, there are few size or weight restrictions on what can be added to the platform. For example, Figure 2 shows an earlier version of the soccer robot with a camera for sensing, and a laptop (not visible) with a wireless card for computation and communications. One challenge is communicating to the RMP via the CAN Bus if a laptop is used as the primary computational device. While many micro-controllers support a CAN interface, most laptops do not. However, there are a number of PCMCIA CAN Bus cards, some with Open Source Windows/Linux drivers[2], as well as USB to CAN converter cables that makes this a solvable task.

The one unique challenge to using the Segway RMP as an autonomous robot resides in its dynamic balancing. Dynamically balancing creates a number of interesting dynamic properties that must be taken into consideration. Dynamic balancing means that it is constantly catching itself from falling over. If anything interferes with that process, say a kill or disable balance command is sent or the platform exceeds its pitch angle limits (typically around 30° from vertical), then the RMP will fall over. Given the height, weight, and high center of gravity, the resulting fall can damage mechanical components or electronics through impact or transmitted shock. Another way to fall over is if the robot hits or gets caught on an obstacle and is unable to execute a motion that will balance itself. In this case, the control loop will try with increasing force to correct for the unbalanced situation. This may lead to recovery, or may compound the situation leading

[2] Such as the Kwaser PCMCIA LAPCanII card (see *http://www.kwaser.com*)

to a harder fall than if balancing was instead disabled. Addressing the safety issues of using the robot, given its weight, size, and power, are a challenging but solvable problem.

A second aspect of dynamic balancing has an impact on the motion of the platform. As it is a single rigid mass on wheels, to accelerate in any direction the platform must first lean in the direction and then catch up with itself to prevent falling over. This creates a number of interesting and related effects. When driving forward from a standstill, the wheels must in fact drive backwards initially to create a lean angle and then accelerate forwards to catch up to the robot. From a controller's perspective, this makes it appear that the robot drove in the opposite direction to the command. A similar situation occurs when changing from acceleration to deceleration. In this case, the wheels must speed up so that the robot goes from leaning forward to leaning backward. Once the backward lean angle is established, the robot can begin slowing down. However, the high center of mass means that the RMP will fall slowly. While this helps make dynamic balancing feasible, it also means that changes in acceleration take a non-negligible amount of time. This contrasts to most electric powered wheeled platforms where changes in acceleration are nearly instantaneous.

Dynamic balancing also has an impact on mechanics and software algorithms. The changing pitch angle of the robot must be taken into account for any actuators and sensors on the robot. For actuators, such as a ball kicking mechanism, this means the contact angle with any external objects is a function of the pitch angle and must be considered. Secondly, the changing pitch angle affects what part of the world external sensors will perceive. Perception algorithms must account for this change in pose to operate effectively.

The work reported in this paper addresses many of these issues. The following sections present the techniques we have developed in order to build an autonomous soccer-playing robot.

3 THE ELECTRO-MECHANICS OF A SOCCER ROBOT

In meeting the challenge of turning a Segway RMP into a soccer-playing robot, we have developed 4 key goals. First, the soccer player must be autonomous by perceiving the world, making decisions, and acting without human intervention. Second, the player must be able to interact with human players by recognizing their presence and communicating. Third, the player must be able to manipulate a ball well enough to be competitive with humans. Lastly, safety must be considered in every aspect to prevent injury to humans and damage to equipment.

With these goals in mind, we have to consider the many challenges that accompany designing and implementing a complete robotic system. Cost effectiveness, processing power, perception, weight distribution, and resistance to shock are all important considerations. The unique motion of the Segway also introduces problems not seen with other platforms. The Segway moves forward by tilting over and driving the wheels in order to rebalance. This motion can lead to the ball becoming stuck underneath the body and wheels of the Segway. This causes the wheels to lose contact or traction with the ground making the Segway unable to sufficiently maintain balance. Any consequential fall could potentially damage equipment. As a solution, guards, consisting of modified rubber mud flaps, were placed in front of the wheels and software was implemented to prevent the RMP from interacting with the ball unless it can safely kick.

Another challenge introduced with Segway Soccer is that there is no unique playing surface. The rules allow for games to be played on grass, Astroturf, cement, or asphalt. Changes in ground softness, clearance, and surface texture alter the dynamics necessary to manipulate the ball. Unlike other robotic soccer platforms, the Segway also tips up to +/-25-30° with respect to the vertical; thus, any attached kicking plate and system will also tip. This requires the manipulation system to be robust enough to manipulate the ball under changing conditions.

With a basic infrastructure in place, we added two laptop computers to interpret the world and control the RMP. One laptop is used to provide the RMP with the capability to process data from a pan/tilt CCD camera and another laptop to quickly decide what action to take and send commands to control the actions of the RMP. The Segway RMP player must possess the capability of consistently recognizing field markers, players, opponents, and the ball. Since the ball and the opponents are continuously moving, a pan tilt servo system mounted at near human eye level is necessary to ensure a complete and robust world observation can be made. The RMP is non-holonomic; therefore, in order to receive a pass, it must look sideways to track the ball as it drives forward or backward to stop the ball with the side of its wheel. This camera mobility allows the robot to drive in one direction while it updates the world model by looking in another.

The RMP must also possess the capability of communicating with a human player. A soccer game requires both humans and robots to quickly communicate with each other in order to effectively position themselves to score a goal. This communication is unique in that in addition to human players being able to tell a robot where to go, a robot can tell a human player where to reposition. Presently, speakers mounted on the RMP allow the robot to communicate with the human players. A 12V sealed lead-acid battery was added to power the control board, speakers, and all the mechanisms. The Segway battery, lead acid battery, and both laptop batteries can be charged with one power-strip that can be plugged into a wall outlet.

Hardware is needed to be able to protect the components of the Segway from damage during a fall. The laptop computers and ball manipulation system components are mounted as close as possible to the bottom of the Segway reducing their falling distance and the shock they will need to absorb. The laptops are also securely fastened with Velcro straps preventing them from being ejected from the confines of the RMP body. The laptop mounting is attached using two of the three mounting screws on the inside of the wheel housing. Angle bracket can be added to these screws in order to support a length of sheet aluminum. The goal in the laptop mounting is to allow easy access while preventing damage.

Steel safety stands were also added to reduce the total distance the Segway will fall once it is no longer capable of dynamically balancing. In a soccer game, the RMP always has the potential of falling over due to high speeds, uneven terrain, and interaction with the ball or other players. The stands mount onto the side plates of the RMP and only allow it to fall over 30 degrees from the vertical. The key in designing safety stands is to keep the stands as compact as possible. Since the RMP is capable of a zero turning radius and can spin fairly fast, protruding stands are a potential safety hazard. If the stands are too compact, a high-speed fall could cause the RMP to fall over top of the stands.

A catching mechanism was implemented giving the robot the capability of robustly manipulating a soccer both for catching a pass and for turning with the ball prior to executing a pass. The robot can drive to the ball, grab it, and then quickly turn with the ball before passing or shooting. This mechanism consists of two high torque metal gear hobby servos that raise and lower a plastic ring.

A microcontroller was added to control the pan/tilt RC servo-motors, the solenoid valves to activate the pneumatic kicking system, the catching mechanism, to monitor the tank pressure, and to receive feedback from the hobby servo potentiometers. This board can communicate with the laptops through a serial port interface.

3.1 Ball manipulation systems

One of the main challenges to using a Segway RMP to play soccer lies in designing a ball manipulation system that allows a Segway platform to kick a ball to the scale of an outdoor human game. The need for passing in the game and the inability of the robot to safely propel the ball by running into it necessitates the development of a kicking mechanism. We present an analysis of several types of kicking mechanisms as well as a detailed implementation and evaluation of a pneumatic kicker

3.1.1 Kicking System Design Considerations

A ball manipulation system can be described as a mechanical manipulator used to accelerate a ball to a desired velocity in a desired trajectory. This can be achieved in many different ways with various actuators. The most common systems come from the realm of robot soccer as seen in RoboCup [7] competitions. These include pneumatic, spring, solenoid, rack and pinion, and rotating plate systems. A careful analysis of the following factors is needed to determine which kicking system best fits a given platform and environment:

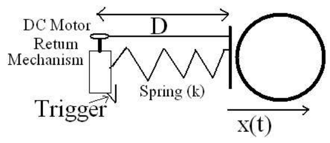
• Speed	• Accuracy	• Kick capacity
• Response time	• Recovery time	• Safety
• Complexity	• Weight	• Size
• Power	• Reliability	• Maintenance

Table 3. The factors to consider for kicking mechanisms.

The actuating system must be chosen with these considerations in mind. For each option, we present the basic system components, the mathematical models necessary to properly specify an appropriate actuator, and an example comparing each option to the pneumatic system we implemented and describe in section V.

3.1.2 Spring Loaded Mechanisms

Spring kicking mechanisms use an extension or compression spring(s) to store and then release energy to propel the ball. As such, a mechanism is needed to tension the spring(s) and a trigger to release the stored energy. Such mechanisms must be robust, and are non-trivial to design. Apart from the obvious complexity, spring strength is coupled to kicking power, but a more powerful spring is more difficult to retract and hold. This relationship leads to potential problems during a soccer game where the time to reload a powerful spring can take several seconds if a cheaper less powerful motor is used. Springs do provide the best power density out of the given the options [4][5]. For a spring with a spring constant of *k*, a kick length of *D*, and a kicking mass of *m*, the equation of motion is given in Figure 3.



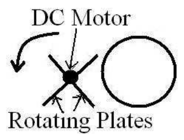
$$\ddot{x} = -kx \cdot m^{-1}, \qquad x(0) = D, \dot{x}(0) = 0$$
$$x(t) = D\cos\left(\sqrt{k/m}\,t\right)$$

Figure 3. The spring-based kicking mechanism.

For the Segway, the pneumatic kicking system model predicts the ball will be kicked at a 4.3 m/s max velocity. Using the above equations, one can determine the spring constant, *k*, necessary to achieve a similar speed with a similar stroke length to the pneumatic model (0.1524m). Assuming a kicking mass of 0.85kg, a spring constant of 676 N/m would be necessary. This would require a force of at least 103N in order to load and hold the spring at 0.1524m for a kick. Depending on motor size and consequentially cost, the spring would take one to several seconds to reload for another kick. Complexity, reload time, and cost are the liming factors for the spring kicker design.

3.1.3 Rotating Plate Mechanisms

Rotating plate kickers consist of two or more flat surfaces, bars, or other contacts arranged in a balanced paddleboat configuration. This technique was developed by [1]. The shaft of the paddle wheel is connected to a DC motor. The angular velocity of the paddle wheel determines the end velocity of the ball, although there is great variability due to the potential variation in the contact point. Pulse Width Modulation can be used to vary the speed of the wheel and thus vary the power of the kick. Rotating plate mechanisms require a significant amount of space to mount the paddle wheel and the drive motor. Furthermore, for larger robots, rotating plates become extremely dangerous to human operators. A rotating plate mechanism scaled to the size of a Segway would have to be approximately 18cm by 38cm. The plates would be rotating fast enough and with enough power to cause injury to humans who happen to fall off of their HT into a kicking device. As a result, we do not consider a rotating plate mechanism in depth.



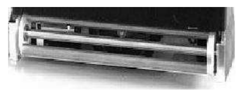
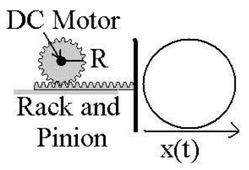


Figure 4. The rotating plate kicking mechanism. Picture taken of robots described in [1].

3.1.4 Rack and Pinion System

Rack-and-pinion systems are driven by DC motors and thus the ball velocity is dependent on the output power of the motor [3]. For a rack and pinion motor system with a back emf of k_e, voltage of *V*, forward torque per amp of *K*, terminal resistance of R, pinion radius of *r*, gear ratio of *N*, and total kicking components mass of *m*, the following are the equations of motion:



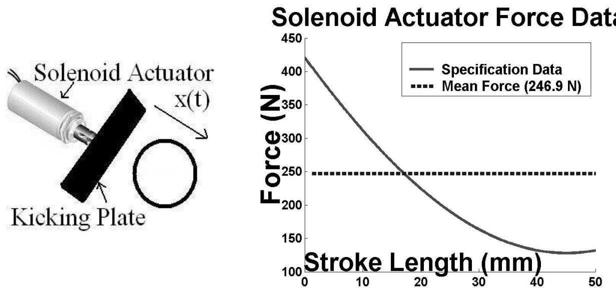
$$\ddot{x} = \frac{K}{mr^2R}(Vr - k_e\dot{x}), \qquad x(0) = 0, \dot{x}(0) = 0,$$
$$x(t) = \frac{Vr}{k_e}t - \frac{mr^3RV}{NKk_e^2}\left(1 - e^{-\frac{k_eKN}{mr^2R}t}\right)$$

Figure 5. Rack-and-pinion kicking mechanism.

A rack and pinion system is comparable to the other options but the price and design requirements are more significant. Using an 80W Maxon motor [12], a 0.015m pinion radius, 1:1 gear ratio, and a total kicking mass of 1.3 kg, the rack and pinion system can accelerate the ball to a theoretical velocity of 6.7 m/s in 0.1524 m (6in). With a motor efficiency around 75% and the friction forces acting against the sliding rack, the actual velocity will be closer to 3.5 m/s. An 80W motor plus drive electronics, however, is an expensive proposition. Moreover, due to the mounting of the handlebars and width of the kicking surface, a two motor system will most likely be required. Hence, a rack and pinion is unfeasible for use on a Segway platform.

3.1.5 Solenoid Systems

A solenoid kicker consists of a solenoid that creates a magnetic field around a shaft that is propelled by the field and accelerated away from the solenoid [13]. The shaft is returned by a built in return spring. Consider a solenoid kicking system with a current of *I*, ampere turns of *N*, plunger radius of r, a return spring constant of *k*, and a total kicking mass of *m*. The equation of motion and resulting numerical approximation are given in Figure 6.



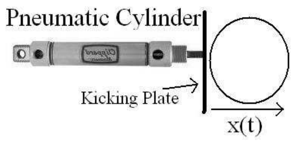
$$\ddot{x} = m^{-1}\left(\tfrac{1}{2}\mu_0\pi\left(\tfrac{rNI}{x}\right)^2 - kx\right), \qquad x(0)=0 \quad \dot{x}(0)=0$$
$$\ddot{x} = \mathrm{m}^{-1}\left(\mathrm{F}_{\mathrm{avg}} - kx\right)$$
$$x(t) = \tfrac{F_{avg}}{k}\left(1-\cos\left(\sqrt{\tfrac{k}{m}}\,t\right)\right)$$

Figure 6. The solenoid kicking solution and its force characteristics.

A solenoid system becomes more impractical with larger robots. Most commonly available solenoids produce approximately 400N of force and generally have small stroke lengths on the order of 0.0254m (1in), which limit its ability to effectively contact a ball. For the solenoid shown in Figure 6, assuming a return spring constant *k* of 99 N/m, and a kicking mass of 1.5 kg, a ball would be kicked at 4.1 m/s over 0.0508m stroke length (2in). With the effects of friction and motor efficiency the actual speed would be closer to 3 m/s. This is comparable to the pneumatic system but the smaller stroke length limits is ability to effectively manipulate a ball during a game. The high voltage requirement also raises safety issues. Pneumatic Systems

Pneumatic piston systems usually consist of one or more actuating cylinders, a gas reservoir, solenoid valves to control the air flow, a source of compressed gas in the form of compressed air or liquid carbon dioxide (CO_2), and a regulator to maintain a specified pressure. The decision between CO_2 and compressed air depends on the availability of CO_2. Most air compressors and air tanks nominally operate up to only 150 psi while CO_2 tanks fill to several thousand psi. The higher pressure allows greater output force leading to a more powerful kick. Additionally, the high density of liquid CO_2 means that considerably more gas can be compressed into a smaller volume. Therefore the kicker can have considerably more capacity. The major drawback of CO_2 is that it is not readily transportable as it is considered as dangerous goods. Similarly, it may not be readily available in overseas locations. Additionally, its rapid expansion during each kick results in thermal issues such as the formation of condensation, which may be problematic for electronic parts. As a result, a compressed air approach is often used instead.

Figure 7 shows the pneumatic cylinder set up. The kicking plate can vary in material and shape dependent upon application. A pneumatic system offers a wide range of options in its configuration and employment as one or more pistons can be fired at the same time or in synchrony to achieve a directional kick. For a pneumatic system with power factor, *f*, combined kicking mass, *m*, return spring constant, *k*, and operating at a pressure, *P*, the equations of motion are:



$$\ddot{x} = m^{-1}\left(fP - kx\right) \qquad x(0)=0, \dot{x}(0)=0,$$
$$\ddot{x}(t) = fP\cdot k^{-1}\left(1-\cos\left(\sqrt{\tfrac{k}{m}}\,t\right)\right)$$

Figure 7. The pneumatic kicking system.

For a Segway, a suitable cylinder would be approximately 0.254m (10in) long, 0.01905m in diameter and produce 274N for force at 140 psi. The pistons are the only moving parts and the air tank consumes the most space. The price of a pneumatic system is also fairly cheap. A functional system can be bought for less than \$100. The air used to power the cylinders is naturally accessible and can be refilled quickly during a soccer game with an onboard air compressor. The system has a low chance of malfunctioning and becoming inoperable during a game because the only moving parts are the cylinder shafts [2].

3.2 Performance Characteristics

With the considerations presented, we chose to use a pneumatic approach due to its relative simplicity, low cost, and transportability. Figure 8 shows the resulting arrangement.

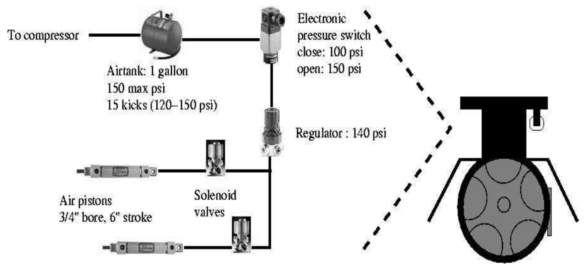


Figure 8. Schematic diagram of the implemented pneumatic kicking system.

Two 0.01905m bore, dual acting pneumatic cylinders were chosen as the main actuating components. This size cylinder provides adequate power with a sturdy shaft that can sustain unexpected stress. Dual acting cylinders do not have a return spring to reset the cylinder shafts back to their original position. This allowed us to implement a return mechanism with just enough force to reset the kicking plate without significantly affecting the output force. We used 4 x No.64 (3.5in x 1/4in) rubber bands to return the kicking plate. Two cylinders also allow for directional kicking.

3.2.1 Air Resevoir Options

The air reservoir can be designed in two different ways. The reservoir can be large enough to hold enough kicks for the entire game or an onboard air compressor can refill a smaller reservoir. If the robot has enough room to house a larger tank, not having an air compressor allows the overall system to be simpler. We used a 1 gallon tank, which provides a sufficient number of kicks as seen in Figure 9. We have an onboard compressor that turns on after 15 kicks and shuts off when the tank pressure reaches 150 psi. The compressor is controlled by a microcontroller that also monitors a mechanical pressure switch that opens at 150 psi. As a result, the operation of the compressor is completely automated.

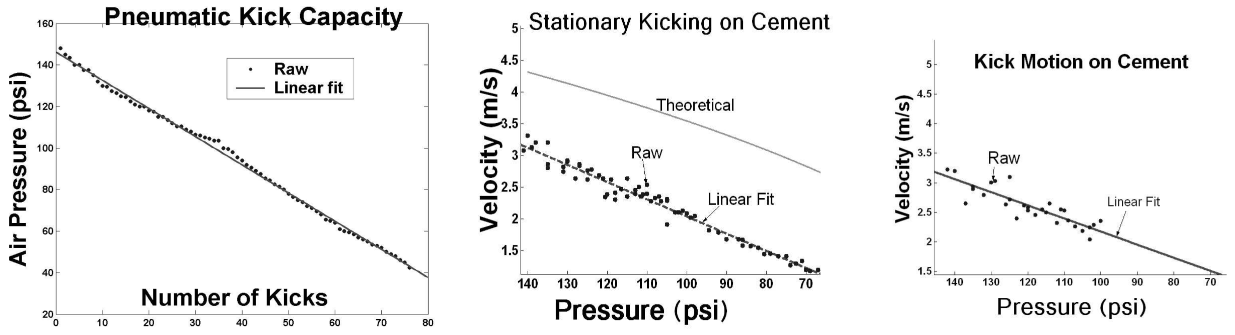


Figure 9. Kicking performance. The left figure shows the drop in pressure with each kick. The middle and right figures show the kicking speed as a function of tank pressure for stationary kicks and kicks involving motion of the Segway platform respectively.

The cylinders accelerate the ball to a max velocity of approximately 3.5 m/s, which is sufficient for a two on two game of Segway soccer. The velocity can increase to 4.5 m/s if the Segway RMP is moving at the ball when it kicks it. The theoretically predicted top speed for a stationary kick is approximately 4.3 m/s. The loss in velocity is due to the efficiency of the pneumatic cylinders, an imperfect impact with the ball, and ground friction. An experiment was setup using one of the cylinders, a small kicking plate, and a golf ball. The velocity of the golf ball was measured on a cement floor. This experiment was designed to significantly lower the effects of impact and friction losses. Through these tests, it was determined that the pneumatic cylinder alone had an efficiency of 75%. These losses are due to several factors including cylinder friction, exiting air resistance, and flow rate limitations. Impact losses and ground friction account for an additional 2% loss. The theoretical and experimental kick speed versus cylinder pressure plot is shown in Figure 9 (middle). Furthermore experiments were conducted measuring the speed of the ball when the Segway RMP played back a kick motion in which it swung its base forward and simultaneously kicked (Figure 9 right).

3.2.2 Accuracy

The kick is sufficiently accurate as seen by the distribution in Figure 10. The mean is 122 mm and the standard deviation is 175 mm. The mean error can be mostly accounted for by experimental error in lining up the kick. In practice, this mean and variance will be modified by the robots ability to position itself next to the ball.

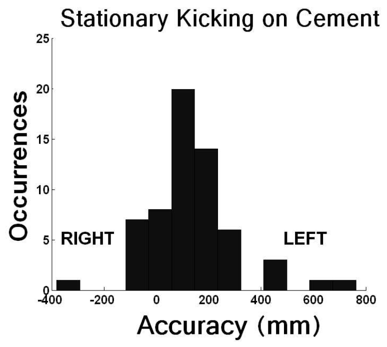


Figure 10. Accuracy histogram of stationary kicking.

4 INTELLIGENCE FOR A SOCCER ROBOT

Figure 3 shows the complete control architecture for the RMP. The gray boxes show the main processing path that makes up perception, cognition, and action. In an environment occupied by other fast moving agents, the ability to perceive and respond to situational changes in minimum time is essential. Hence, it is critical to overall robot performance that the control loop formed by the gray modules operates at full frame rate with minimum latency. The white boxes show the supporting development environment, which although not critical during execution play a central role in the pace of the development cycle and therefore in the robustness of the result. We now describe each major component and its role in the overall hierarchy, namely; perception, skills and tactics.

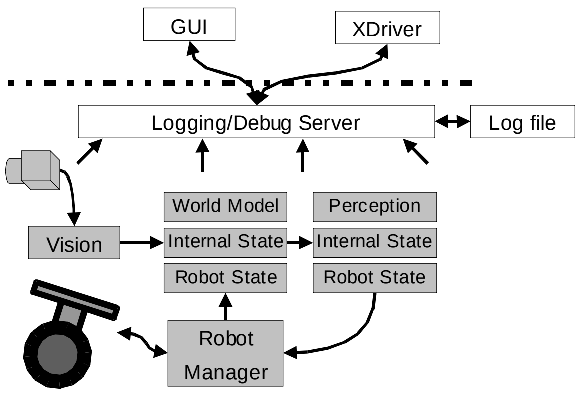


Figure 11. The control hierarchy used for the robot. The gray modules provide the system autonomy while the white modules aid in development. X-driver is a tele-operation program.

4.1 Intelligent Perception: Vision and Tracking

For environments within which the RMP must operate, there are few sensors that can compete with color vision for low cost, compact size, high information content and throughput, and relatively low latency. Thus, our chosen path is decidedly vision centric where a color camera provides each robot with its only external sensor. Given the range of environments that the robot operates in -- outdoors on grass fields, or indoors when the weather is poor – perception quickly becomes an overriding challenge. That is, to develop *fast* vision algorithms that provide the robot with timely, but relatively noise free information that is robust to variations in lighting intensity. To complicate issues, only a fraction of the available processing power is dedicated to vision as the remainder must be used for tracking, modeling, behaviors, navigation and motion control. Achieving these goals is one of the key challenges to be addressed for using the RMP. To our knowledge, there are no vision algorithms that offer all of these features.

A number of fast, color-based algorithms and freely available source libraries have been developed for static lighting conditions (e.g. [26]). However, these libraries do not as yet extend to variable, or changing lighting conditions. Our approach uses region growing [36], where seeds are first provided from any regions that found in the previous frame that were large enough. The remainder of the seeds are chosen uniformly. Each region is grown using a fixed homogeneity constraint based on the distance of the new color pixel in YUV space from the average color of the region grown thus far. That is, if the new pixel has color $c_j = (y_j, u_j, v_j)^T$, it is added to the region R_i if it is a neighbor of an existing pixel in the region, $c_j \in \mathrm{N}(c_k), c_k \in R_i$, and it is sufficiently close to the region mean, $(y_j - \hat{y}_i < \tau_y) \wedge (u_j - \hat{u}_i < \tau_u) \wedge (v_j - \hat{v}_i < \tau_v)$. The region mean is updated after each pixel addition and has the value $\hat{c}_i = |R_i|^{-1} \sum_{R_i} c_j$.

Once regions are identified and the summary statistics for each region are calculated, high level vision processing begins by identifying the regions that are potential objects of interest: the ball, field markers, teammates or opponents. Regions that are close to the color of interest, using a Euclidean distance metric and threshold, are labeled accordingly. That is for prototype p : $R^p = \{R_i \mid |\hat{c}_i - c^p| < \tau^p\}$. A region may be labeled as belonging to more than one class type. For each object of interest, the regions are then based on objects' expected geometric properties as in our earlier work [25]. Figure 12 shows an example of the vision processing and its output.

The goal of vision is to provide as many valid estimates of objects as possible (i.e. a low false-positive rate). Estimates of the global positions of the obstacles are derived using a lens distortion model, a pin-hole projective model for the camera, and knowledge of the robot's respective tilt angles. Tracking then fuses this information to track the most interesting objects of relevance to the robot. At the time of writing our current multi-hypothesis tracker is still under active development. Ongoing work is focused on developing a true probabilistic multi-hypothesis tracker.

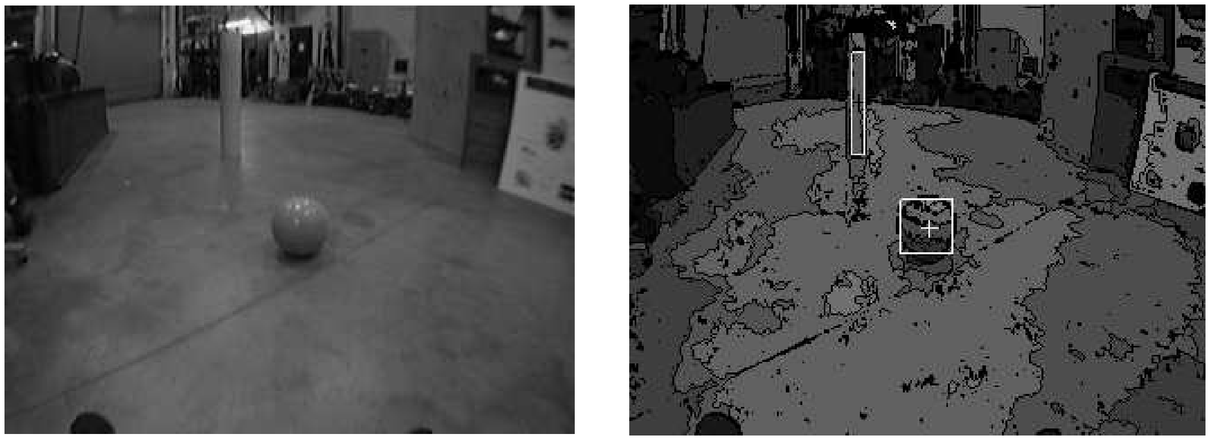


Figure 12. The left image shows a raw image in an indoor scene. The right image shows the resulting regions, with the identified ball region and yellow marker outlined with a bounding box.

4.2 Robot Control Hierarchy

In our previous work we developed and thoroughly validated a hierarchical, behavior based control system called Skills-Tactics-Plays (STP) [27] for adaptive multi-robot control in adversarial, highly dynamic environments. The architecture consists of Skills for low-level control policies, Tactics to encapsulate a complete single robot behavior, and Plays for team coordination. We have applied the STP control hierarchy to the Segway problem. Here we focus on skills and tactics that encapsulate single robot behavior.

- **Robot Control:** At the lowest level of the hierarchy are motion control and obstacle free navigation, key components to any mobile robot. These modules are adapted versions of our prior work, [27] and [28], respectively, and will not be discussed in detail here. Above motion control are the skills, the building blocks of individual behavior.
- **Skills:** Each skill is a focused control policy for carrying out complex actions over some limited set of world states. For Segway soccer an example skill is the action to actually kick the ball, or to position behind a ball in order to kick it towards a target.
- **Tactics:** A tactic encapsulates a complete single robot behavior. An example tactic includes shooting the ball on goal, receiving the ball from a teammate, or defending the goal.
- **Plays:** A play encapsulates team behavior by encoding a sequence of tactics to be carried out by each team member [35]. Plays are beyond the scope of this paper.

Skills are the action primitives for tactics, thus a tactic consists of instantiating a sequence of skills to execute, in other words a finite state machine, where the sequence of execution depends upon how the perceived world state changes.

Tactics affect the world by instantiating skills in sequence to execute and by passing each executing skill parameters derived from the world state to affect its operation as it executes. For example to shoot the ball at the goal, the shoot tactic executes a sequence of skills such as **gotoBall**, **positionForKick**, and when aimed at the target the final kick skill. Finally, following the usual behavior based approach [29], tactics and skills execute in parallel at frame rate (30Hz).

With only minor variations all of the tactics employed on the RMP were developed previously in[27]. At this high-level of behavior, the tactics evaluate the world and determine target points for kicking or moving. As most of the details of each action are contained in the underlying skills, we have found that the tactics transfer effectively from one platform to another completely different platform but for a very similar task. In contrast, the underlying skills are highly dependent up on the hardware and do not transition effectively. To avoid the necessity of redeveloping a new skill set, we followed the novel approach of developing skill acquisition systems for rapidly acquiring new skills through training by a human operator.

4.3 Rapid Skill Acquisition and Use

Within the STP framework, skills form the closest interface to the physical robot hardware. Each skill, therefore, is highly dependent upon the physical properties of the robot and its environment and plays a major role in determining overall robot performance. The dependency of skills on the physical properties of robot and environment mean that skills rarely transfer well from one environment to the next, or from one robot platform to another, where deviations are the norm. When one considers that in practice, developing a skill requires considerable hand tweaking of control parameters, a notoriously error prone and time intensive operation, it seems apparent that some form of automated skill acquisition is needed.

One interesting approach to skill acquisition is reinforcement learning (e.g [34]). Although there have been significant advancements in reinforcement learning, there are still a number of issues that must be address in order to learn effective robot control where *it could be used as a sub-component of an existing control hierarchy*. The primary limitation is that of long training time. Even the most advanced algorithms still take a considerable amount of time to achieve even moderate levels of performance.

We have taken a different approach to skill acquisition, where skills commands are generated by generalizing from example trajectories provided earlier by a human operator via tele-operation. There have been a number of examples in the literature (e.g. [34][33][32] and [30]) where tele-operation or learning from observation has been effectively used to learn to execute a task thus motivating our approach. Our particular approach is inspired by two observations. First, it is usually relatively easy to tele-operate a wheeled robot, even a dynamically balancing one, through the approximate sequence of motions required to execute a skill. Second, skills developed by hand are often a direct, if complex, function mapping from sensory state to output commands. Our key assumption is that the commands given by the human operator are noisy samples from some fixed, but unknown function of the world state. The goal of skill acquisition is therefore to provide estimates of this function for world states experienced by the robot based on the provided samples.

There are numerous function approximation techniques that are available. However, we have focused our investigations on locally weighted regression (LWR - [30]). LWR provides robust function estimation using a minimal amount of training data, is able to generalize by interpolation across unseen inputs, and is amenable to fast, efficient implementations. All of these criteria are of importance to skill acquisition in the given setting. Additionally, LWR has a long, successful history of use in robot control problems [31].

Our approach works in two phases; recording and playback. During the recording phase, a human operator guides the robot through a sequence motions by providing a sequence of commanded actions:

$$a_i = \left(a_i^1, a_i^2, \ldots, a_i^k\right)^T$$

A special-purpose recording tactic stores the sent command a_i, as well as the corresponding relevant world state:

$$x_i = \left(x_i^1, \ldots, x_i^d\right)^T$$

for later recall. During playback, the skill uses these data-points to approximate the actions of the recording phase. The key assumption is that the recorded actions are sampled from some unknown function so that $a_i = f(x_i)$. During playback the skill approximates this function, using its samples, using LWR, such that i.e. $a(t) = \hat{f}(x(t))$ as the robot moves in the world. Concretely, we have:

$$a(t) = \hat{f}(x(t)) = \frac{\sum_i K(x(t), x_i) \bullet a_i}{\sum_i K(x(t), x_i)}$$

where $K(\cdot\bullet,\bullet)$ is a kernel function, which in practice is a Gaussian given by:

$$K(x(t), x_i) = e^{\frac{-(x(t)-x_i)^2}{2\bullet h^2}}$$

There are three issues to implementing the LWR function approximation. The first is to provide for fast function approximation. Following the usual approach, we store the x_i 's in a Kd-tree for fast nearest neighbor search [31]. Additionally, we limit the horizon of the search, for some pre-defined $K_{\min}$ to:

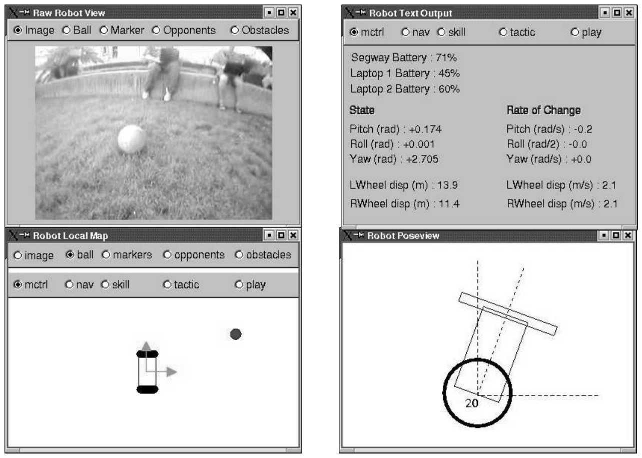
$$H_{\max} = \sqrt{2h^2 / \ln(K_{\min})}$$

Thereby limiting the extent of the evaluations. Secondly, one must choose the so-called bandwidth parameter, h . We chose a global value of h by hand. An alternative approach would be to use cross-validation, or a locally varying bandwidth parameter [30]. Although there are some arguments for the usefulness of the latter, in our current work it proved unnecessary. Finally, while LWR works well for interpolation, it is well recognized that it becomes unpredictable for extrapolation beyond the underlying data set, i.e. outside of the convex hull formed by the x_i s. For robot control this can cause serious limitations. However, given that skills are by definition only defined over a sub-set of the state space, say S_{skill} , provided this set of states is a sub-set of the convex hull formed by the x_i 's no difficulties should occur. This is achieved by ensuring the skill training data covers S_{skill} . As a secondary safety measure, should the normalization factor be too small the output is set to the null vector and warning notifications are given.

4.4 Development Infrastructure

One of the key aspects to robot development that is often overlooked in the literature relates to the support infrastructure to aid development. In our experience, good development infrastructure can greatly ease the development burden, however, there has been no scientific study of what 'good' development infrastructure is. Based on our prior experiences, we have developed a number of infrastructure tools for the Segway RMP to aid development. Concretely, we have developed a Debug Server and GUI client for providing contextual text and graphic debug information at execution time. We have also developed a logging/playback system to be able to record what the robot 'sees', 'thinks', and 'does' and play it back for later analysis. Finally, we have developed an off-line vision testing tool to speed up the vision development process. We now describe in detail each of these components.

Figure 13 shows the GUI output for allowing a remote user to view several aspects of the robot's sensory and internal state. This is particularly useful for developing behaviors for the robot as one can quickly see what the robot's model of the world is. The GUI client programs connect to the Debug server (see Figure 11) over a TCP socket. In the *RawRobotView* display, the output of the vision system can be seen and the user can use it to view the output of the various tracking modules. For instance, the outputs from the ball or obstacle tracking modules can be directly viewed. Other output interfaces, such as the *RobotLocalMap,* show an ego-centric view of the robot and show the positions of detected objects around it, while the *RobotPoseView,* shows an ego-centric side view of the robot so that the reported tilt angle can be visualized. Finally, all of the text output generated from the soccer server, such as debug and state information, can be viewed on the *RobotTextWindow* display.


Raw Robot View
[✓ Image] Ball Marker Opponents Obstacles
Robot Text Output
[✓ mctrl] nav skill tactic play
Robot Local Map
image [✓ ball] markers opponents obstacles
[✓ mctrl] nav skill tactic play
Robot Poseview
20

```
Segway Battery : 71%
Laptop 1 Battery : 45%
Laptop 2 Battery : 60%

State                        Rate of Change
Pitch (rad) : +0.174         Pitch (rad/s) : -0.2
Roll (rad) : +0.001          Roll (rad/2) : -0.0
Yaw (rad) : +2.705           Yaw (rad/s) : +0.0

LWheel disp (m) : 13.9       LWheel disp (m/s) : 2.1
RWheel disp (m) : 11.4       RWheel disp (m/s) : 2.1
```

Figure 13. The GUI output showing the raw video from the Segway (top left), the robot's state real-time information (top right), the robot's local world model (bottom left), and a graphic of the robot state (bottom right). The GUI may connect to the robot 'live' or to a log for off-line analysis.

The Logging server allows the Segway to record all sensor and state information at a configurable rate, so that it can be analyzed off-line as well as replayed. The latter is especially useful given the speed of action and the large information content. The user can request a number of different sensor channels to log, where these might be all of the robot's pose and velocity state information, the raw video, the segmented video, or even the specific positions of the tracked targets (ball and players). Logging all of the raw video data along with all of the robot state information is fairly processor intensive and is not typically done unless there is a specific need for it.

In order to test and debug the video processing algorithms, the raw frames of video can be loaded into a *vtest* program, which will process each frame using all of the Segway's video processing code. This is particularly useful for gathering large amounts of video data for testing purposes. The robot can simply be tele-operated around the environments where it will be expected to operate, and new video processing code can be tested without having to drive the robot.

5 CONCLUSIONS

In this paper we have presented our work with the Segway RMP. We have shown how the Segway RMP as a base platform can be developed into a fully functional autonomous robot capable of playing soccer. Turning a Segway RMP into a soccer-playing robot requires a combined approach to the mechanics, electronics, and software control. We have developed the necessary electro-mechanical components to enable a Segway RMP to:

- Sense using a color based camera mounted on a pan-tilt unit
- Manipulate the ball using compressed gas kickers, and motor driven trapping mechanisms
- Fall safely, in the advent of a loss of balance
- Operate autonomously, with wireless and speech-based communications for development and operation purposes

In conjunction with the electro-mechanical apparatus, we have developed the software algorithms necessary to enable autonomous operation. Additionally, we have developed infrastructure that aids in the development process by providing readily used logging and debugging facilities. Much work remains to reach the goal of humans and robots playing soccer together, and this forms the future directions of this research endeavor.

ACKNOWLEDGMENTS

This research was sponsored by the United States Army under Grant No. DABT63-99-1-0013. The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing official policies or endorsements, either expressed or implied, of DARPA, the US Army, or the US Government.

REFERENCES

[1] S. Behnke et. al., "Using Hierarchical Dynamical Systems to Control Reactive Behavior." *RoboCup-99:Robot Soccer World Cup III.* Berlin: Springer, 2000, p.189

[2] M. Ferraresso et. al., “Collaborative Emergent Actions Between Real Soccer Robots.” *RoboCup-2000:Robot Soccer World Cup IV*. Berlin: Springer, 2001, p.297

[3] Ng Beng Kiat et. al., “LuckyStar II-Team Description Paper.” *RoboCup-2000:Robot Soccer World Cup IV*. Berlin: Springer, 2001, p.543

[4] G. Wyeth et. al., “UQ RoboRoos: Achieving Power and Agility in a Small Size Robot.” *RoboCup-2001:Robot Soccer World Cup V*. Berlin: Springer, 2002, p.605

[5] R. Cassinis et. al., "Design for a Robocup Goalkeeper." *RoboCup-99:Robot Soccer World Cup III*. Berlin: Springer, 2000, p.255

[6] A. Bredenfeld et. al., “GMD-Robotst.” *RoboCup-2001:Robot Soccer World Cup V*. Berlin: Springer, 2002, pp. 648-9

[7] M. Asada *et. al*. "An overview of RoboCup-2002 Fukuoka/Busan". AI Magazine, 24(2): pages 21-40, Spring 2003

[8] M. Nicolescu, M. J Mataric, "Learning and Interacting in Human-Robot Domains", Special Issue of IEEE Transactions on Systems, Man, and Cybernetics, Part A: Systems and Humans , Vol. 31, No. 5, pages 419-430, C. C. White and K. Dautenhahn (Eds.), 2001

[9] M.B. Dias and A. Stentz. "Opportunistic Optimization for Market-Based Multirobot Control". Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems IROS 2002, 2002

[10] M. Ferraresso, et al., “Collaborative Emergent Actions Between Real Soccer Robots.” RoboCup-2000:Robot Soccer World Cup IV. Berlin: Springer, 2001, pp.297-300

[11] N. Kiat, Q. Ming, T. Hock, Y. Yee, and S. Yoh, “LuckyStar II-Team Description Paper.” RoboCup-2000:Robot Soccer World Cup IV. Berlin: Springer, 2001, pp. 543-546

[12] Maxon Motors, “F 2260, Graphite Brushes, 80 Watt, No.880,” http://www.mpm.maxonmotor.com] pp. 95

[13] Solenoid City, “Push Type Tubular Solenoid, Series S-70-300-H,” [http://www.solenoidcity.com]

[14] “Skill Acquisition and Use for a Dynamically-Balancing Soccer Robot”, Brett Browning, Ling Xu, and Manuela Veloso, The Nineteenth National Conference on Artificiial Intelligence, in press.

[15] H. Nguyen, J. Morrell, K. Mullens, A. Burmeister, S. Miles, N. Farrington, K. Thomas, and D. Gage. “Segway Robotic Mobility Platform”, SPIE Proc. 5609: Mobile Robots XVII, Philadelphia, PA, October 2004.

[16] Robotic Mobility Platform, URL: *http://www.segway.com/segway/rmp/*, Segway LLC.

[17] Segway Robotic Mobility Platform, URL: *http://www.spawar.navy.mil/robots/land/SegwayRMP/SegwayRMP.html*, Space and Naval Warfare Systems Center, San Diego.

[18] R. Brooks, L. Aryananda, A. Edsinger, P. Fitzpatrick, C. C. Kemp, U. O’Reilley, E. Torres-Jara, P. Varshavskaya, and J. Weber, “Sensing and Manipulating Built-for-Human Environments,” Int. J. of Humanoid Robotics, Vol. 1, No. 1 (2004)

[19] R. O. Ambrose, R. T. Savely, S. M. Goza, P. Strawser, M. A. Diftler, I. Spain, and N. Radford, “Mobile Manipulation using NASA's Robonaut,” Proc. Int. Conf. on Robotics and Automation (ICRA'04), New Orleans, LA, April-May 2004.

[20] J. L. Krichmar, G. M. Edelman, “Brain-Based Devices: Intelligent Systems Based on Principles of the Nervous System,” Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, Las Vegas, NV, October 2003.

[21] A. Howard, D. F. Wolf, and G. S. Sukhatme, “Towards 3D Mapping in Large Urban Environments,” IEEE/RSJ Int.Conf. on Intelligent Robots and Systems, Sendai, Japan, Sep 2004.

[22] H. G. Nguyen, et al., “A Segway RMP-based robotic transport system,” SPIE Proc. 5609: Mobile Robots XVII, Philadelphia, PA, October 2004.

[23] Kitano, H.; Kuniyoshi, Y.; Noda, I.; Asada, M.; Matsubara, H.; and Osawa, E. 1997. “RoboCup: A challenge problem for AI”. AI Magazine 18(1):73–85.

[24] Polani, D.; Browning, B.; Bonarini, A.; Yoshida, K. (Eds.) “RoboCup 2003: Robot Soccer World Cup VII”. LNCS Lecture Notes in Artificial Intelligence , Vol. 3020

[25] Stengel, R. F. “Optimal Control and Estimation”, Dover Publications, September 1994.Bruce, J., Balch, T., Veloso, M., “Fast and Inexpensive Color image Segmentation for Interactive Robots”, proceedings of the 2000 IEEE International Conference on Intelligent Robotics and Systems.

[26] Bruce, J., Bowling, M., Browning, B., Veloso, M., “Multi-Robot team Response to a Multi-Robot Opponent Team”, proceedings of the 2003 International Conference on Robotics and Automation.

[27] Bruce,J., Veloso, M., “Real-Time Randomized Path Planning for Robot Navigation”, proceedings of the 2002 IEEE International Conference on Intelligence Robots and Systems.

[28] Arkin, R. “Behavior-Based Robotics (Intelligent Robotics and Autonomous Agents), Bradley Books, 1998.

[29] Lenser, S.; Bruce, J.; and Veloso, M. 2001. “CMPack: A complete software system for autonomous legged soccer robots”. In Proceedings of the Fifth International Conference on Autonomous Agents. Best Paper Award in the Software Prototypes Track, Honorary Mention.

[30] Cleveland, W., and Loader, C. 1995. “Smoothing by local regression: Principles and methods”. Technical report, AT T Bell Laboratories, Murray Hill, NY.

[31] Atkeson, C. G.; Moore, A. W.; and Schaal, S. 1997. Locally weighted learning. Artificial Intelligence Review 11(1-5):11–73.

[32] Schaal, S., and Atkeson, C. 1998. Constructive incremental learning from only local information. Neural Computation 10(8):2047–2084.

[33] Billard, A., and Mataric, M. 2001. Learning human arm movements by imitation: Evaluation of biologicallyinspired connectionist architecture. Robotics and Autonomous Systems 37:145–160.

[34] Bagnell, J. A.; Kakade, S.; Ng, A.; and Schneider, J. 2003. Policy search by dynamic programming. In Proceedings of Neural Information Processing Systems, NIPS 16.

[35] Bowling, M.; Browning, B.; and Veloso, M. 2004, Plays as effective multiagent plans enabling opponentadaptive play selection. In Proceedings of the International Conference on Automated Planning and Scheduling (ICAPS’04).

[36] Forsyth, D. A., Ponce, J. Computer Vision: A Modern Approach. Prentice Hall, 2002.