ECE 3992
Senior Project Proposal

MIPRover: A Two-Wheeled Dynamically Balancing Mobile Inverted Pendulum Robot

6 May 2005

Prepared By: Kevin E. Waters

Department of Electrical and Computer Engineering
University of Utah
Salt Lake City, UT 84112

MIPRover: A Two-Wheeled Dynamically Balancing Mobile Inverted Pendulum Robot

I. Introduction

The goal of this project is to design and build a two-wheeled dynamically balancing mobile inverted pendulum robot. Perhaps the most well known example of a two-wheeled inverted pendulum device is the Segway people mover, and NASA is experimenting with Robonauts (robot astronauts) that use a Segway type system for locomotion. The concept of balancing an inverted pendulum is simple, and can be likened to attempting to balance a broom. When you try and balance a broom on the floor (brush end up), it wants to fall over. However, when you try and balance the same broom in your hand (again, brush end up), you compensate for the broom's tendency to fall over by moving your hand in the direction it is falling, in an effort to keep it upright. What you are doing is applying active control to stabilize an inherently unstable system. This project will take this simple concept and attempt to create a two-wheeled robot that will utilize active control to stabilize an otherwise unstable system.

II. Motivation

This project was selected because of the unique challenges presented by trying to balance an inverted pendulum using a two-wheeled platform, and the combination of hardware and software required to achieve this goal. It will involve a blending of mechanical, electrical, and computer engineering solutions to solve the problem, which may lead to future practical applications. More than anything, it seems like it will be a fun project to work on.

III. Project Overview

The basic concept of a two-wheeled dynamically balancing robot is quite simple: drive the wheels in the direction the upper part of the robot is falling; and when the wheels are driven to stay under the robot's center of gravity, the robot will remain balanced. The robot will use these properties to move forward or backwards. To move forward, the wheels are first rotated backward to create a forward tilt. To compensate, the wheels are then moved forward to try and rebalance. To keep moving, a slight tilt is maintained as the robot moves forward, while at the same time trying to maintain a constant distance to the ground. To achieve this, two sets of sensors are required: sensors to measure the proximity of the robot with respect to the ground; and sensors to measure the speed of the robot. From this data, four items are used to define motion and position, which are then used to drive and balance the robot. These are:

1. Tilt angel,
2. Angle velocity (the first derivative of the tilt angel),
3. Horizontal position, and
4. Horizontal velocity (the first derivative of the horizontal position).

These values are summed and fed back to the motor as voltage, which drives the wheels, which are used to balance the robot. Figure 1 shows the relationship of these simple inputs and outputs. Mathematically, this is a fourth order nonlinear system which must be linearized to maintain vertical equilibrium.

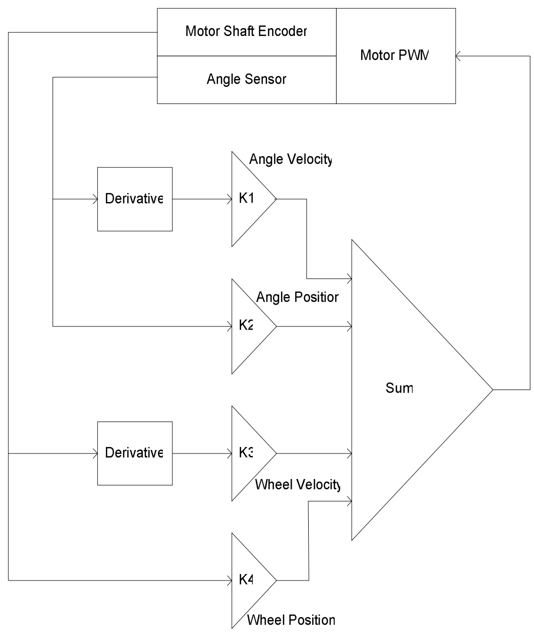


Figure 1. Four items are used to define motion and position.

IV. Project Tasks

The project will be comprised of two main tasks: Design and build a prototype; and then use the information gained from this task to develop a final design from which the MIPRover will be built. The prototype will be a simplified version of the final design, and will be used as a means to learn and gather useful information that will be vital for the success of the final design. The final design should be fully autonomous, with the possibility of collision detection and remote user interaction if design, time, and budget allow.

A. The Prototype

The prototype will be an uncomplicated design that will contain simple off-the-shell components that can be easily assembled and configured. This is so that a working prototype can be quickly designed and built, which will be the springboard for the final design.

1. Hardware

The principal hardware for the prototype is the RCX controller, also known as "The Brick," which consists mainly of the Hitachi H8/3292 Microcontroller. This microcontroller is comprised of: a Hitachi H8/300 16 MHz CPU Core, with an on-chip address/data/control bus, and on-chip memory of 60kB ROM and 2kB RAM; a RAM module; and motor controls. This will be used to control the operation of the prototype rover, and will act as the sensors for measuring the speed of the wheels, and thus the speed of the robot. The RCX uses 6 AA batteries for power to the microcontroller, and a 9V battery as power for the wheel motors.

The proximity sensors used for determining tilt angle and tilt velocity will consist of Electro-Optical Proximity Detector Sensors EOPDs. These are visible light proximity sensors, and are similar to infrared (IR) proximity sensors, but use a visible light source instead of an IR light source. The visible light will provide a higher resolution and sensitivity between different colored surfaces than the IR sensors, which is critical for following a course.

The motors that will be used for driving the robot are 9v electric mini-motors, with one connected to each wheel. These motors provide high-torque, and can run forwards as well as backwards, all of which are critical for maintaining balance. These motors will connect directly to the motor controllers imbedded on the H8/3292 microcontroller.

The body of the robot will be “home-made,” and will consist of any hardware needed to operate the prototype MIPRover. The body will house all the hardware listed above, as well as any additional hardware that may be deemed necessary.

2. Software

The RCX controller uses an embedded open-source operating system called BrickOS. This operating system is compatible with C/C++/Assembly programs, and supports the gcc/g++ compilers. C will be the programming language of choice, with assembly used as needed, and the software control system will be uploaded to the controller for operation of the robot.

The basic functionality of the MIPRover will be autonomous operation, which will consist primarily of: balancing in place; spinning in place; and following a predefined course. For balancing, the robot will be programmed to maintain an upright position, with the only movement being small corrections to maintain balance. To spin in place, the robot will be programmed to run each motor in opposite directions, while maintaining balance. To follow a predefined course, the proximity sensors will not only be used to maintain a constant distance from the ground, they will also be used to follow a line on the ground, with corrections made not only for balance and forward motion, but to follow the line.

The final function of the software is to know when it’s time to stop. This can be when it has completed a task, or if no corrections have been made for the last second, which is highly unlikely for a system as unstable as this, then it is likely that the robot has tipped over, and it will shut down.

B. The Final Design

The final design of the MIPRover is not well defined at this time. It will depend almost entirely on the results of designing, building, testing, and operating the prototype. Another reason for the uncertainty is that there are many different ways of handling the aspects of balancing a two-wheeled inverted pendulum robot, with the biggest unknown the right combination of hardware and software needed. For example, this is complicated by the fact that there are many different ways of determining proximity to an object: IR, visible light, sound, laser, just to name a few. Add to this the combination of gyroscopes and/or accelerometers that can be used to maintain balance, each of which can "drift" over time, and the complexity of the systems grows quickly. And finally, the type of controllers needed to drive these different combinations of components, and the software needed, makes it almost impossible to predict with any degree of certainty what will be needed right from the beginning. The final design will hopefully be an evolution of the prototype, or it may go in an entirely opposite direction of the prototype.

V. Interfaces

There are three main interfaces to the prototype robot: the software control system to microcontroller upload interface; the EOPDs proximity sensor and microcontroller interface; and the wheel to microcontroller interface.

A. The Software Control System to Microcontroller Interface

The prototype will initially be designed to operate autonomously. This means the only interface between the operator and the robot will be when the software control system is uploaded to the RCX controller. This can be done either through an IR interface, or through a serial connection. Any remote control by the operator would be through the IR capabilities of the RCX controller.

B. The EOPDs to Microcontroller Interface

The EOPDs are designed to work with the RCX controller. This means that the sensors simply plug into the RCX, and the issues should be primarily software; with little, if any, hardware compatibility issues.

C. The Motors to Microcontroller Interface

As stated above, the H8/3292 microcontroller has imbedded motor controllers, and like the EOPDs, 9v electric mini-motors are designed to work with the RCX controller. Again, this means that the motors simply plug into the RCX, and any issues should be primarily software; with little, if any, hardware compatibility issues.

VI. Testing and Integration

A. The Prototype

For the prototype robot, the hardware is already well matched, and should go together and operate with few problems. The biggest challenge will be adjusting the software control system to work with the hardware in maintaining balance, as well as eventually performing the autonomous operations outlined above. This “learning” process will be invaluable in designing and building the final rover model. Unit and integration tests will be performed at each stage of the build process to minimize problems that will arise with the final prototype design. In a system as inherently unstable as this robot is, the operation of each system depends greatly on the reliability of all the other systems.

B. The Final Design

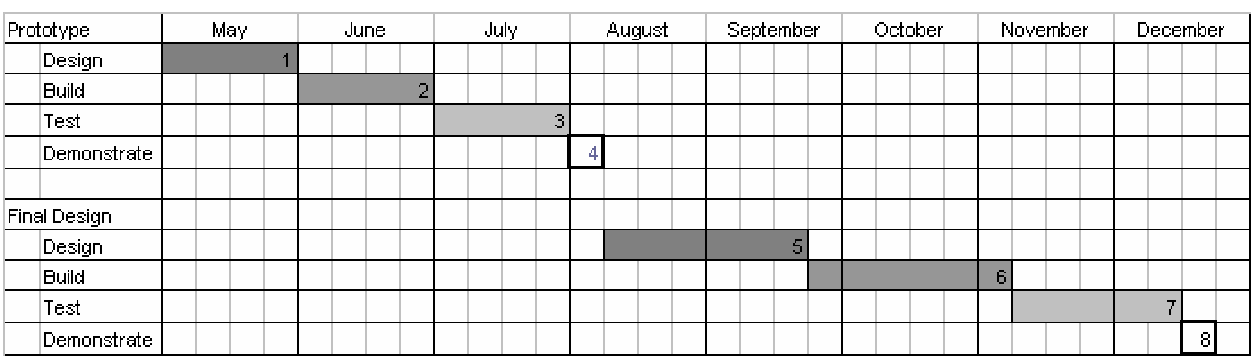
For the final rover model, integration of the hardware will be a greater challenge, and will thus require more time integrating and testing. It will be very important that the right hardware components be matched. Once the hardware is assembled, the lessons learned from developing the software control system for the prototype should make it a little easier in developing the control system for the final robot design. Here again, unit and integration testing will be used at each stage of assembly to assure compatibility and reliability, and to minimize the likelihood of problems propagating and carrying through the process.

VII. Communication Plan

Since I am a group of one, there should be no group communication issues. However, it is important that regular communication be maintained with the senior project advisor, and that regular meetings take place. This will be done in order to demonstrate that the project is progressing on schedule; that milestones are being met; to discuss project related issues; and to keep problems that will arise from becoming project derailing issues. A Web site will be established and updated regularly to keep those who are interested informed on the progress of the project.

VIII. Schedule and Milestones

A. Schedule



B. Milestones

1. End of May – Prototype has been built.
2. End of June – Testing of prototype.

3. End of July – Prototype has full functionality.
4. 1st week of August – Prototype is ready to be demonstrated.
5. 3rd week of September – Final design complete.
6. 1st week of November – MIPRover built.
7. 2nd week of December – Testing of MIPRover complete
8. End of semester – Demonstrate MIPRover.

IX. Risk Assessment

A. The Prototype

There is less risk associated with the prototype robot that with the final design of the MIPRover. The prototype has less hardware than the hardware that will be used in the final design, and the hardware has been designed to work together. The only problem associated with the hardware would be availability of parts, but these are common robot parts used by those who build robots regularly. The biggest risk with the prototype will be with the software control system, and being able to fine tune it to be able to control a two-wheeled self-balancing robot.

B. The Final Design

There is considerably more risk with the final design. The complexity of the hardware is significantly greater, and therefore the complexity of the software control system will greater as well. The biggest risk here is getting the right combination of hardware sensors and controllers, and then marrying it to the right control system. However, it will be harder to pinpoint risk areas for this part of the project, since the final design depends largely on the results of the prototype phase of the project. Therefore, great care will be taken during the final design phase to minimize the potential risks, and detailed plans will be created and followed in this regard.

X. Bill of Materials

A. The Prototype

Part Description	Qty	Cost
Body	1	$ 50.00
Wheels @ $3 ea.	2	6.00
RCX Controller "Brick"	1	165.00
Technic 9V Electric Motor 2 @ $9 ea.	2	18.00
Hitechnic Electro-Optical Proximity Detector 2 @ $40 ea.	2	80.00
AA Batteries 6 @ $1.75	1	1.75
9V Battery	1	1.00
	Total Cost:	$ 321.75

Body
Home-made, cost is estimated

Wheels
This includes axels, wheels and tires
RobotStore: www.robotstore.com

RCX Controller
RobotStore: www.robotstore.com

Technic 9V Electric Motor
RobotStore: www.robotstore.com

Electro-Optical Proximity Detector
Hitechnic: www.hitechnic.com

Batteries
Home Depot: Centerville, UT

gcc/g++ Compiler
Free download off Internet

B. The Final Design

These details will be unknown until after the prototype has been completed.