

# Dynamically Adaptive Inverted Pendulum Platform

2009 Space Grant Symposium

*Jonathon Cox*

Colorado State University  
Department Of Electrical Engineering

2515 Manet Ct.

Fort Collins CO, 80526  
Email: csutke@gmail.com

## I. INTRODUCTION

The idea of an inverted pendulum platform is not new. In fact it is a fairly well documented and tested concept. The ability to have a mobile platform that is supported by only two wheels creates an enormous advantage for agility and the ability to maneuver in tight spaces due to the smaller base size. This being said, it also has several limitations which have caused the concept to be largely discredited for practical robotics purposes. These include higher power consumption, higher processing requirements and limited design flexibility compared to a conventional statically stable design.

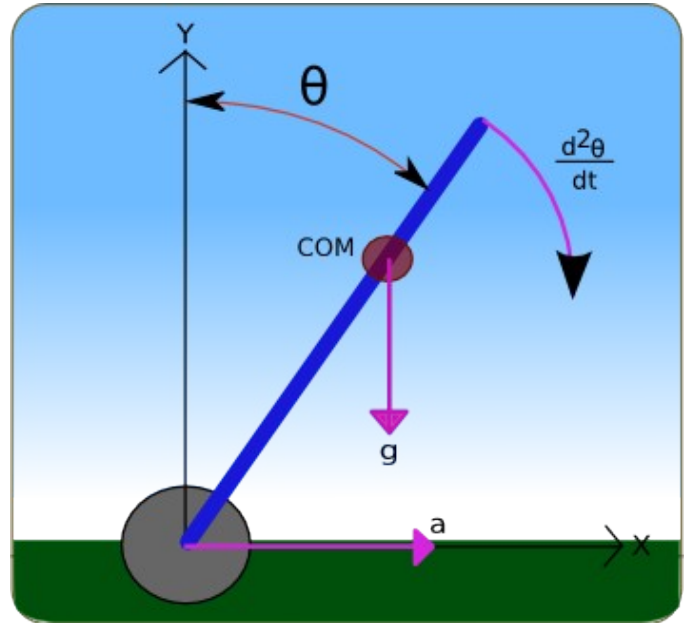
The limited design flexibility comes from the fact that conventional inverted pendulum platform algorithms assume that the device's center of mass does not change. This is an incredibly limiting assumption. For many practical applications, the center of mass needs to be flexible. For instance, what if an inverted pendulum had an arm that extended? In most current algorithms, the platform would either fall over or it would roll forward continuously because the control system thinks it is falling. For some applications, including the Segway personal transporter, this can be useful as a control mechanism. For other applications such as the arm example above, this inhibits the design.

The goal of this project is to solve the above problem by creating an algorithm for a dynamically adaptive inverted pendulum platform. The algorithm will be able to sense a change in the platform's center of mass by monitoring several different inertial sensors and then quickly adapt to maintain balance and minimize unwanted travel.

## II. BASIC THEORY

### A. Physics of the system

Before the control system is covered, it is necessary to discuss the underlying physics that govern the inverted pendulum platform. The diagram presented in Figure 1 shows the free-body diagram for the system with all the relevant forces marked. It is important to note that this derivation is simplified by neglecting the mass of the base, which can be done because of the motor control system that will be discussed later.



**Figure 1: Free Body Diagram of the Inverted Pendulum**

The first goal is to determine the acceleration that the motors need to produce in order to keep the platform's center of gravity over its base. To do this, we can take the moment around an arbitrary point M along the body of the platform between the wheels and center of mass. This is shown in equation 1.

$$\frac{d^2 \theta}{dt} = K \frac{g \sin(\theta) - a \cos(\theta)}{L} \quad (2.1)$$

When the angular acceleration is assumed to be close to zero as will be the case under normal operating conditions, equation 2.1 can be used to derive the acceleration in terms of theta.

$$K \frac{g \sin(\theta) - a \cos(\theta)}{L} \rightarrow a \approx g \tan(\theta) \quad (2.2)$$

$$a \approx K \theta \quad \text{as } \theta \rightarrow 0 \quad (2.3)$$

This result allows the system to be treated as linear when the tilt angle is small, which is the case most of the time.

## B. Control System Approach

Before the specifics of how the platform balances can be discussed, a brief overview of the theory behind the organization of the control system is needed. The complex task of balancing an inverted pendulum platform would be very difficult if it was approached with the intent of creating one transfer function that described the entire system. This could be done in theory but it would be impractical for a real implementation due to dozens of unknown variables related to the physical system. To avoid this and make life easier, the control system is broken down into several basic modules, each of which simplifies the task for the successive module by attempting to deal with very specific unknowns. The division into modules also allows for each module to be developed independently, which greatly simplifies the development process.

The modules that are used to control the inverted pendulum are listed in the form of a flow diagram in Figure 2. The tan boxes each represent a module of the system that has been isolated during development. The green arrows show the data that is transferred between the modules and the yellow arrows represent feedback.

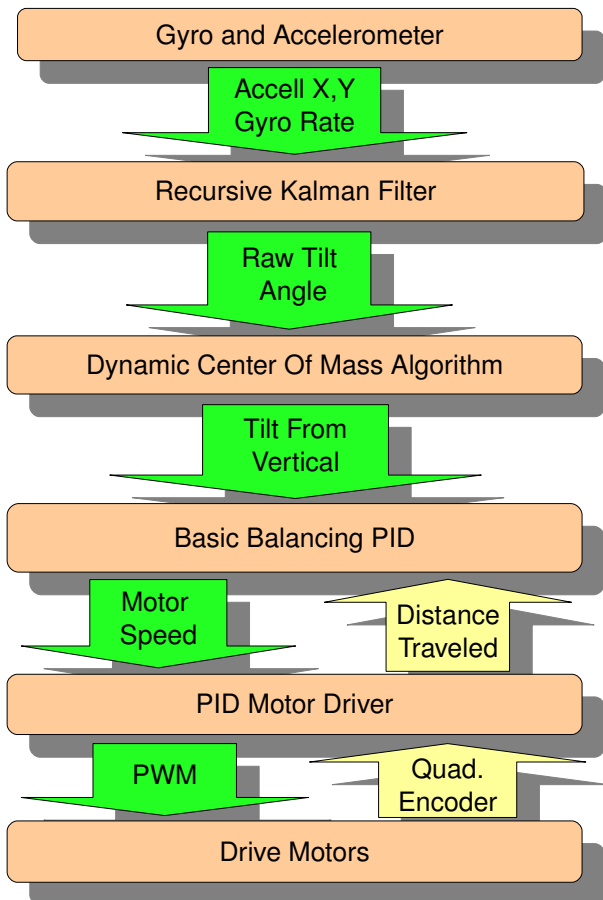


Figure 2: Block diagram of the control system modules

## C. Motor Controllers

Each wheel on the platform has a dedicated motor controller that is responsible for driving the motor at a given speed. The motor controller modules simplify the problem of balancing by taking the mass of the base out of the equation, allowing the main control system to only have to calculate the acceleration required. It accomplishes this by applying a proportional integral derivative controller to each motor. The feedback for the Proportional Integral Derivative (PID) controller is given by the quadrature encoders on the motors which when coupled with the encoder module on the microcontroller return a value that represents the current shaft position. The basic PID controller can then be expanded to drive the motor at a given speed by incrementing the required position of the shaft at a given interval  $\Delta t$ . This is represented analytically in equation 1.1 through 1.3.

$$Err(t) = Position(t) - Setpoint \quad (1.1)$$

$$Output(t) = K_p Err + K_i \int Err dt + K_d \frac{d}{dt} Err \quad (1.2)$$

$$Setpoint = \int Speed dt \quad (1.3)$$

This is obviously required to be implemented using a discrete time approach due to the use of the digital signal processor, which simplifies the integral and derivatives terms of the system to those shown in equations 1.4 through 1.6.

$$K_i \int Err(t) dt \rightarrow \sum_{i=-\infty}^{i=\infty} Err[i] \Delta t \quad (1.4)$$

$$K_d \frac{d}{dt} [Err(t)] \rightarrow \frac{Err[i] - Err[i-1]}{\Delta t} \quad (1.5)$$

$$\int Speed(t) dt \rightarrow \sum_{i=-\infty}^{i=\infty} Speed[i] \Delta t \quad (1.6)$$

Using the discrete time equations of the system, it is a simple matter to implement the controller on the processor. In implementation,  $\Delta t$  is determined by a hardware timer on the processor that is run by interrupts in order to make the control loop execution time irrelevant to the system.

## D. Kalman Filtering Module

The Kalman filtering module is responsible for converting the raw data from the accelerometers and gyroscope into a stable value that represents the tilt angle of the system. The Kalman filter is especially suited for this purpose because it is able to adjust for the noise in the measurements.

The Kalman filter is a recursive algorithm that attempts to predict the position of a linear system and then checks the prediction against the sensor inputs. It then uses the difference between the prediction and the actual state to update a coefficient matrix, essentially 'learning' how the system reacts. It is a very practical for this application because the Micro Electromechanical System (MEMS) sensors have a lot of noise

and the gyroscope by nature of its design drifts over time. The Kalman filter estimates this drift and compensates for it when combining the accelerometer and gyro data to calculate a tilt angle. It is out of the scope of this paper to discuss the intricate details of how a Kalman filter works as the focus of this paper is intended to be on the adaptive balancing algorithms. The Kalman filter is also a very well documented and proven method for this type of application and more information about it and its derivation can be found from several of the sources that are listed in the works cited page.

### E. Basic Balancing PID

The balancing PID algorithm is a basic Proportional Integral Derivative controller that receives the tilt angle provided by the Kalman filter as the input and returns the acceleration needed from the motors in order to maintain balance. This is the last module necessary to maintain balance for a known physical system. If the system is unknown or dynamic, a basic PID algorithm is not sufficient which leads to the topic of the next section.

### F. Dynamic Adaptivity to a changing Center Of Mass

Up to this point the platform is capable of maintaining balance given that it has been tuned correctly and the center of mass does not change. The dynamic center of mass module uses the tilt angle from the Kalman filter and the sensor inputs in a recursive algorithm to predict the angle that will place the center of gravity over the base. It then feeds this angle to the basic balancing PID module as the reference angle that must be maintained in order to remain stable. It should be stressed that the basic balancing PID module is still being used, the angle that it uses as reference for vertical is what gets changed by the adaptive COM module.

After much thought, it was realized that in order to maintain the stability of the system under a changed center of mass, the base should attempt to maintain a fixed position. This does not mean that it cannot move, it just means that a desired position should be established for the base and then the angle of tilt should be inversely proportional to the distance it must travel away from that position in order to maintain stability. Figure 3 depicts this concept graphically.

Using this method to adjust the reference angle also has the added benefit that it also facilitates the ability to position the platform either remotely or using some other control system by simply changing the setpoint and allowing the system to adapt. Caution must be taken when this is done though that the setpoint is not moved so quickly that the system cannot compensate in time. This would cause a compounding error and most likely lead to the system falling over due to exceeding the motor's capabilities.

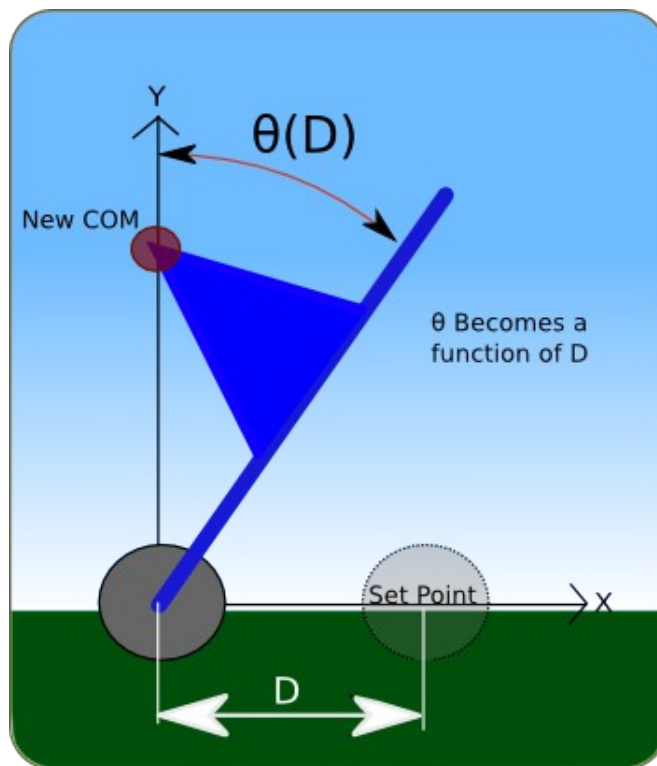


Figure 3: Illustration of how  $\theta$  is a function of the distance from the setpoint.

The first attempt at designing this module used another PID algorithm that incorporated the distance the wheels have traveled as the input to calculate desired tilt angle. This could analytically be described as a second order or nested PID controller. Although this worked for small displacements, it became extremely unstable when the platform had traveled too far because it forced the tilt to go beyond the capabilities of the motors. It also had an enormous overshoot because it could not correctly make the platform tilt in the opposite direction to slow down before the setpoint.

Seeing this, it was determined that a slightly more complex algorithm was necessary in order to relate the tilt angle to the distance from the setpoint. Figure 4 shows the four possible states of the system while trying to reach the setpoint.

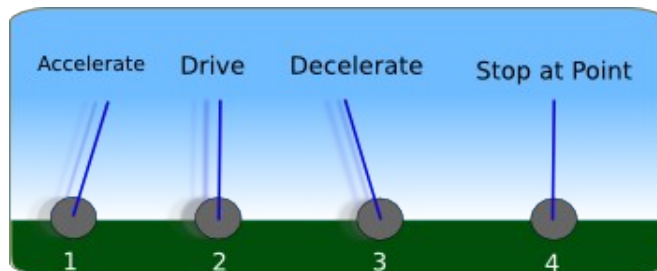


Figure 4: The possible states of the system while attempting to reach a set position.

Once the motion of the system had been broken down as in figure 4, an equation that represented the tilt angle as a function of the distance to the setpoint and the current speed was derived.

The first step of the derivation is to realize that the platform can only accelerate at a maximum rate limited by the motors. This translates into a maximum tilt angle that is safe for the system which we will call  $\alpha$ . In other words, when the platform is at angle  $\alpha$ , it is accelerating as quickly as possible. At small angles the rate of acceleration is also directly proportional to the tilt angle which leads to equation 2.1:

$$a = k_a \theta \text{ for } \theta < \alpha \quad (2.1)$$

The second physical limitation of the system is the maximum speed that the motors can run at. We will represent this as  $\beta$ . This becomes a more limiting factor because the motors cannot drive at their full speed when 'cruising'. This is because they need to be able to accelerate in order to get in front of the center of mass and slow the platform down. A safe value for the cruising speed which will give the platform enough margin of error to deal with imperfections in the terrain is one half of the maximum velocity of the motors. Therefore,

$$V_{cruise} = \frac{1}{2} \beta \quad (2.2)$$

The maximum velocity the platform can reach is also potentially limited by the distance it has to achieve that speed. This means that the platform must not be accelerating when it passes the point  $D/2$ . These characteristics are summed up in Figure 5 and 6.

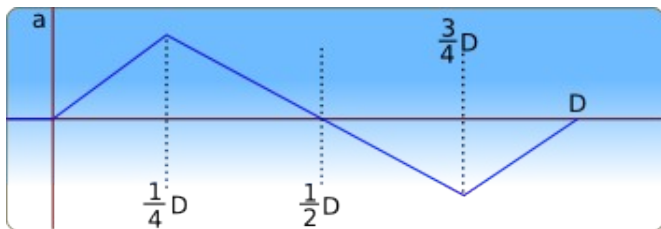


Figure 5: Acceleration as a function of distance when the distance is the limiting factor.

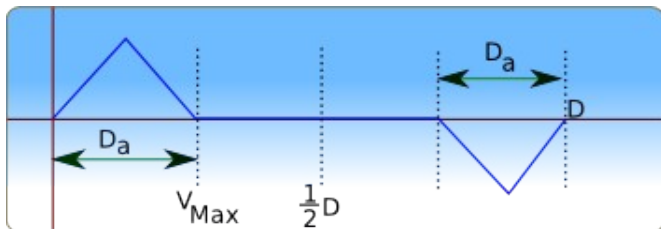


Figure 6: Acceleration as a function of distance when the max speed is the limiting factor.

From these two graphs, it can be seen that in order to prevent from going over the cruising speed, the platform must

start decreasing it's acceleration when it reaches half of the cruising speed. It must also be noted that for the first version of this algorithm the time rate of change of acceleration ( $x'''$ ) is set to be linear in order to simplify the problem. A better performing system could probably be derived such that  $x'''$  is not constant.

The algorithm also takes advantage of the fact that the distance needed to decelerate on a constant friction surface is roughly the same as the distance needed to accelerate. It uses this to its advantage to calculate the distance before the setpoint at which it must start decelerating in order to not overshoot the target.

In software, the algorithm is implemented using several if statements that determine which state in figure 4 the system is in and then determines the correct tilt angle based on the distance still required to travel.

### III. SYSTEM OVERVIEW

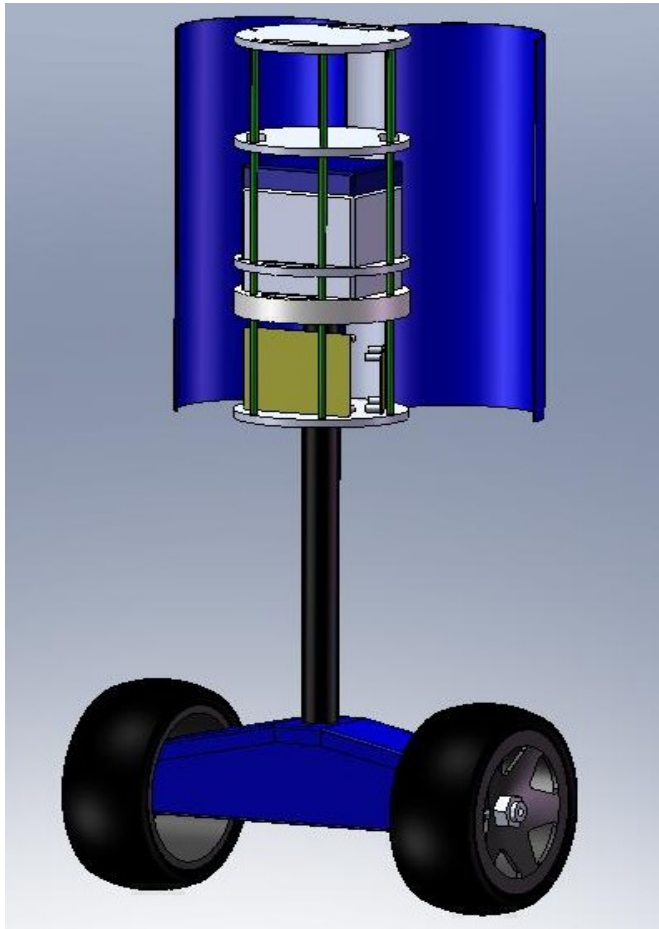
#### A. Hardware

The test hardware created to host the inverted pendulum control system has been created from common, off the shelf components. The two brushed motors used are manufactured by Pittman Inc. and are designed to have very low backlash. Each motor includes a 256 CPR quadrature encoder. The basic frame is constructed from 14 gauge aluminum sheet metal that has been folded to form the major body components. A one inch hollow aluminum tube is used as the 'backbone' of the platform to provide rigidity under motion. It also serves as a conduit for the wires that connect to the motors. The wheels used are 1/6 scale radio controlled car wheels that have custom machined hubs to match the motors. These wheels were chosen because they are foam filled and provide a small amount of damping to the system, therefore making it more stable on hard surfaces. The disadvantage is that the higher traction also forces the motors to draw more current when turning. Power for the system is provided by a 5Ah sealed lead acid battery. It is not an optimal choice but the alternative lithium-ion or NiMH solutions were prohibitively expensive and would require more complex power conditioning to protect the cells.

#### B. Design Compromises

Several aspects of the hardware design involved a trade off between two benefits. The biggest compromise was the decision of where to place the center of mass for the platform. The higher the center of mass, the less reactive the control system needs to be because the platform will not fall as fast. The downside is that it makes the system less maneuverable because it cannot force the system to lean in order to accelerate as quickly. This can be compared to trying to balance a broomstick on your hand. If it is a long broomstick, it is easy to balance. If it is short, it is harder because it requires a faster reaction time.

Another compromise had to be made when I came to deciding what size wheels to use. Larger wheels provide more ground clearance and speed but they also increase the effect of the motor backlash and reduce the torque and therefore acceleration. Smaller wheels give much higher acceleration but cause much more vibration when encountering a bump.



**Illustration 1: A CAD model of the platform hardware with the access panels open**

### C. Control System – Sensors

There are three types of sensors used on the platform to maintain balance, a gyroscope, an accelerometer and quadrature encoders on the motors. The gyroscope and accelerometer are both MEMS technology devices that are interfaced with a high precision analog to digital converter. The shaft encoders are 256 CPR infrared quadrature type encoders.

The placement of the MEMS sensors also plays a big role in how the platform performs. Ideally, the sensors should be placed at the center of gravity for all the mass above the base as this will be the point of rotation. If the sensors are too high or too low, the accelerometer will react to the dynamic

acceleration of the base and introduce error into the measurements.

### D. Control System – Motor Drivers

Each motor has a dedicated motor driver which is connected to the main control board by a simple serial network. The motor driver boards are based on the 16 bit Microchip PIC30F4012 digital signal processor which is clocked at 160Mhz. This device was chosen because it has several peripherals that make it an ideal choice for motor control including a quadrature encoder module, a universal serial asynchronous receiver transmitter module (USART) and a powerful processor core with optimized floating point instructions. To supply current to the motor, the LMD18200T single chip H-Bridge controller was chosen because it has a very simple interface and is cheaper and smaller than a discrete H-Bridge approach. The motor driver board also includes a status LED that indicates the direction of the motor and any errors that might occur.

### E. Control System – Man Control Board

The main control board for the system is based on the 16 bit Microchip PIC30F4011 digital signal processor. This processor was chosen because it has the following features:

- 30 MIPS operation
- 2 USART modules
- 10 bit A/D converter with adjustable references
- 28 I/O Pins
- Self Writable Program Memory (for boot loader)
- In Circuit Serial Programmable
- DSP Optimized core with hardware 32x32 bit multiplication.
- Free ANSI C compiler available
- Internal PLL clock multiplier

The clock source for the processor is provided by a 20Mhz oscillator that is internally multiplied by an internal Phase Locked Loop multiplier which multiplies it by six to obtain an operating frequency of 160Mhz. For communication, one of the USART modules is configured to 9600 baud and connected to a wireless serial modem which is connected to a computer. The other USART module is connected to the bus that controls the motor drivers and is configured to 115700 baud to increase the responsiveness of the motors.

The main control board contains it's own regulated and filtered power supply as opposed to a system wide power supply in an effort to reduce amount of noise that reaches the processor. This becomes a significant issue in a system that contains large inductive loads such as motors. The main board also has three buttons with indicator LEDs that are useful for

debugging and controlling the system during development. On the final system, one button is used enable and disable the integral and derivative terms of the balancing PID module. The second button is used to enable the dynamic center of mass module and the third is used to disable the motor drivers. Being able to disable the integral and derivative terms in the basic controller is very helpful when the platform needs to be picked up because it prevents the integral from becoming huge.

#### F. Computer Interface

A computer interface was developed which connects via a wireless serial modem in order to develop and control the platform. The interface includes the ability to display the current motor commands as well as all of the sensors. For remote operation of the platform, the computer interface also allows for a standard computer joystick to be used to send driving commands to the system.

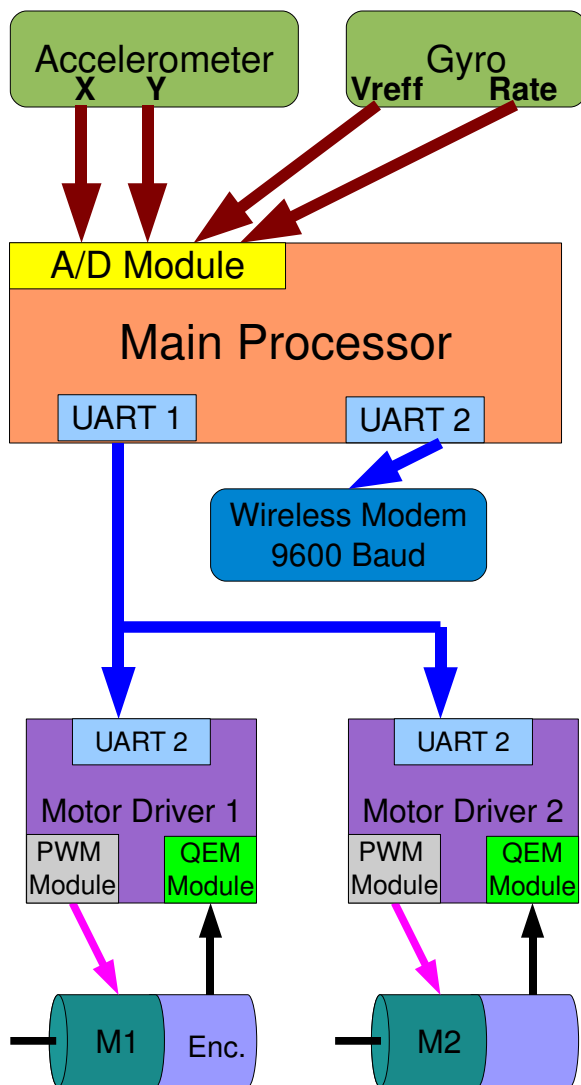


Figure 7: Hardware Flow Diagram

## IV. SOFTWARE

### A. Introduction

The software for the system was written entirely using C++ and the Microchip C30 compiler. The C30 compiler was chosen because it has optimized support for floating point numbers on the dsPIC family of processors and the headers for the processors that will be used have already been written. It also allows for objects to be created which increases the organization of the code.

### B. Hardware Abstraction

Before the development of each module began, Several functions and interrupt routines were written to provide seamless access to the different system resources without the need to worry about the precise hardware implementation. Table 1 shows each hardware peripheral and the associated purpose it was written to support.

Hardware	Implementation	Purpose
A/D Module	Interrupts	Read Sensor Data and apply FIR filter
Quadrature Encoder Module	Function	Read Motor Shaft Position
Timer 1	Interrupts	Provide $\Delta t$ for calculations
UART 1	Interrupts	Provide Computer Interface
UART 2	Interrupts	Provide Motor Driver Interface
PWM Module	Interrupts	Drive The Motors
Buttons	Function	Poll The Input Buttons
Initialization	Function	Configure all the hardware on startup

Table 1: Breakdown of the hardware abstraction

For the hardware that uses interrupts, global variables were defined which give access to the data obtained from the interrupts. This presents the situation where they may be updated while in the middle of a calculation, which can be solved by assigning a global variable to a temporary variable while doing a calculation. The function implementations don't have this issue as they are updated only when they are called.

### C. Organization

The main portion of the code is based around an infinite loop that executes about 60 times a second. This loop

iteratively executes the Kalman filter module, basic balancing PID module and the adaptive COM module to generate an output. The time base for the calculations ( $\Delta t$ ) is generated by a highly precise hardware timer peripheral. This is much more precise than relying on the execution time of the loop and it also allows the timing of the loop to be changed without affecting the other modules.

Each module is written as a block of code in the main control loop which references the global variables for the input and output data. This leads to a more simplistic system and allows it to be optimized for speed.

## V. SYSTEM RESULTS

### A. Basic Balancing Module

When the system is configured to balance using only the basic PID module, it is very stable and has very little oscillation when sitting still. The system is also very capable of balancing on various different surfaces because the motor drivers adapt and compensate for the change in friction to keep the wheels turning at the desired speed.

The basic balancing algorithm performs very poorly on a sloped surface because it does not include any type of compensation for the change in angle, therefore it will continuously accelerate down the ramp until it maxes out the motors and falls over.

The basic balancing module also performs poorly when an outside impulse is applied. It will accelerate to maintain stability but it requires a long distance to come to a stop.

### B. Dynamic Center Of Mass Module

When the dynamic center of mass module is enabled, the performance of the platform is greatly improved. The module allows the platform to adapt quickly when an impulse is applied because it attempts to maintain its current position. It also adapts well when a constant force is applied to the platform because it adjusts the tilt angle in an attempt to maintain its position which in turn introduces an equal and opposite force.

The platform has the same response if a large mass is added to one side in order to shift the center of mass. It will quickly adapt and maintain its position as opposed to driving forever.

The dynamic center of mass module still has trouble on sloped surfaces because the surface changes the entire dynamics of how the pendulum responds to input. That being said, it is much better at it than the basic balancing mode as it is still able to maintain balance with a slight oscillation.

## VI. IMPLICATIONS FOR THE AEROSPACE INDUSTRY

## VII. CONCLUSION

As a final statement I think it would be justifiable to say that the project was a major success in controlling an inverted pendulum platform. The development of the demo platform was completed in just under three months including the mechanical, electrical and software systems. It has also made a large step toward making this method of locomotion practical for use in robotic systems that require a very small footprint and high mobility.

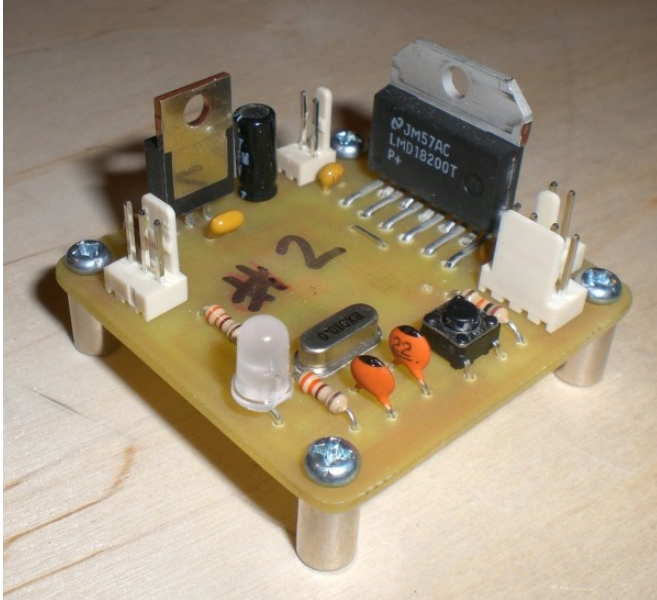
The project has also introduced many other possible opportunities for advancement, such as modifying the system to be capable of balancing in two dimensions in order to be able to handle more rigorous terrain.

More information about this project and video clips of the demo platform can be found at [www.campusaudio.com](http://www.campusaudio.com).

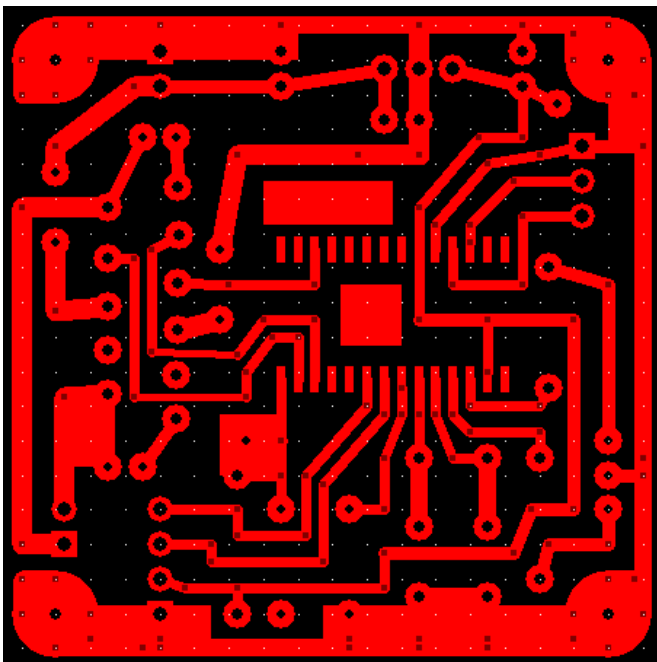
## REFERENCES

- (1) Charles L. Phillips, John M. Parr and Eve A. Riskin, "Signals, Systems and Transforms," Fourth Edition, 2008.
- (2) Serway, Jewett, "Physics For Scientists and Engineers", Volume 1, 2004.
- (3) Polking, Boggess, Arnold, "Differential Equations", Second Edition, 2006.
- (4) "Control Tutorials For Matlab," Carnegie Mellon University, <http://www.engin.umich.edu/group/ctm/examples/pend/invpen.html>.
- (5) Steve Hassenplug, "Controlling the inverted Pole" [http://www.restena.lu/convict/Jeunes/inverted\\_pole/inverted\\_pole\\_journal.htm](http://www.restena.lu/convict/Jeunes/inverted_pole/inverted_pole_journal.htm)
- (6) Tom Pycke, "Kalman Filtering of IMU data, <http://tom.pycke.be/mav/71/kalman-filtering-of-imu-data>
- (7) Microchip Inc. "MPLAB C30 C Compiler Users's Guide", 2005.
- (8) Microchip Inc. , "AN964 – Software PID Control of an Inverted Pendulum Using the PIC16F684", 2004.

A. Motor Drivers

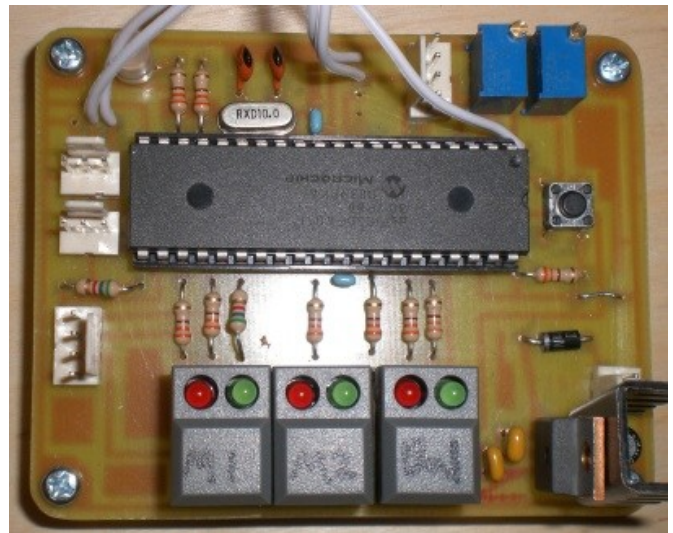


Picture 1: This is a picture of one of the completed motor driver boards. Each board is marked to indicate it's address on the serial bus.

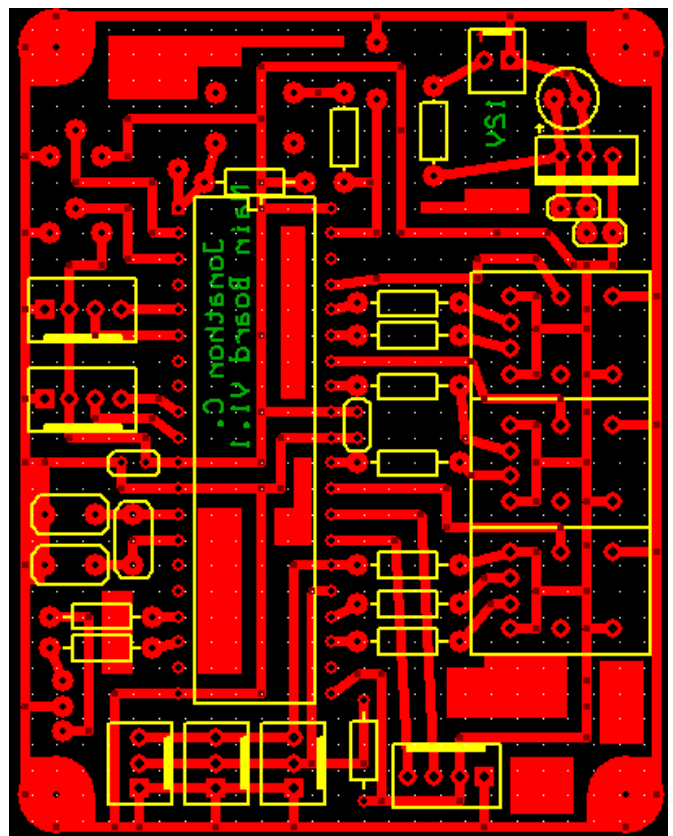


Picture 2: This is the layout that was created to make the motor driver boards.

Main Control Board



Picture 3: This is the completed main control board with the programming cable attached.



Picture 3: This is the layout that was created for the main system board.



**Picture 4: This is the demo system balancing in my living room. The enclosure for the electronics is not yet complete. The rods on top are just temporary mounting rods that have not been cut yet.**



**Picture 4: Another view of the demo platform in action.**