RC Ghost Rider

Adolph Arieux, James Russick, Paul Shimei

School of Electrical Engineering and Computer Science, University of Central Florida, Orlando, Florida, 32816-2450

***Abstract* — The purpose of this project is to fabricate a cockpit that will move based on information received from a remote controlled vehicle that will in turn be controlled by the cockpit. The information that will be used to move the cockpit will be a combination of information gathered about acceleration acting on the vehicle as well as angle displacement of the vehicle with respect to time. The controls to move the vehicle that will be located on the cockpit are similar to what is to be expected in a normal vehicle, including a steering wheel and pedals.**

***Index Terms* — Cockpit, Actuators, Ghost Rider, Simulator, Remote Controlled**

I. Introduction

The purpose of this project is to fabricate an apparatus which controls an external vehicle's every movement. The apparatus, here on referred to as the Cockpit, will transmit information via a wireless transceiver to an external vehicle, here on referred to as the RC Car or vehicle. The Cockpit will provide driving direction controls as well as regulate the speed and the velocity of the RC Car. The RC Car will consist of a store bought radio controlled vehicle that has been modified and is equipped with a mid grade camera fastened to the front and will stream real time video to the Cockpit display screen. The driver will use the video feed to navigate the RC Car through a prefabricated course or have free reign to roam the terrain. The RC Car will also have a series of sensors that will send information concerning the vehicles disposition to the Cockpit regarding the velocity of pitch and roll as well as the acceleration along the X and Z directions of the RC Car which will be accurately recreated to this projects specifications by a series of three actuators attached under the driver's seat. Using the video information seen on the Cockpit's display monitor the user will press the accelerator or brake pedal to control the speed . In addition, a steering wheel will be used to turn the RC Car. The pedals and the steering wheel are spring loaded to ensure the driver experiences slight resistance while operating the simulator, in order to recreate the experience of driving a vehicle while operation of the simulator is underway. The pedals and steering wheel also contain potentiometers that are realized by the MCU on the cockpit and interpolated to be sent to the RC Car which will act accordingly. There will be adjustable components on the Cockpit which are meant to allow rider's of various size to operate the RC Car comfortably in a seated position.

A. Vehicle Description and Specifications

The vehicle (RC Car) is controlled by the cockpit, but equipped to provide feedback to the cockpit concerning the vehicle's displacement. It is also equipped with a camera to provide the user with optical sensory input and enhanced contollability. The vehicle has dimensions less than 24” x 24”, weigh less than 5lbs, operate at a distance of at least 100ft from the Cockpit, can reach speeds up to 5MPH, and has two batteries, one of which rated at 12 volts and more than 24 WH, the other at at least 7 volts and 15WH.

B. Cockpit Description and Specifications

The cockpit is an apparatus mounted on an actuating platform in which the user will operate in a seated position. It utilizes a steering wheel and pedals to gather information to remotely control the vehicle, tilts based on the acceleration of the RC Car in the directions parallel to the ground plane, jolts proportionally based on the RC Car's obstacles, and has a display relaying the image from the camera. The cockpit must be safe, comfortable, and easy for the user to operate. The maximum Operator Weight Limit is 300lbs, so to not strain the actuators or put the operator in danger. The angular velocity is greater than 8deg/s at max load, and it has independent pitch and roll of +/-15 deg. The cockpit must consume less than 1200W, and be operated within 4 feet clear of obstructions.

II. Subsystem Interfacing

Concerning the two main subsystems of the project, the vehicle and the cockpit, it will be necessary to implement a means of communication between the two. This vital communication will be realized by the implementation of an over the air serial connection using two XBEE communication modules. The modules will relay a 9600 baud serial connection from one subsystem to the other with a minimum range of 100 ft on the 2.4Ghz spectrum.

III. Vehicle Hardware and Physical Design

A. Physical Design

The vehicle will be composed of a remote controlled car that has been fabricated to the specifications of this project. A DC motor will be used in order to drive the vehicle, while a standard turning servo will be implemented to actually turn the vehicle. A platform has been added to the vehicle to hold the PCB that contains the necessary components for the project. Two batteries will be used for the project, one will be used for the forward motion of the car and the control circuity, while the other will be strictly for the reverse function of the car and the powering of the camera subsystem.

B. Control Interfacing

All control signals will be given by the Atmega328P micro-controller. The servo motor will be controlled by a PWM signal issued by the micro-controller. In order to drive the vehicle two GPIO's will need to be used as digital output lines. These lines will be fed into the motor control circuit that has been designed for this specific purpose. All of the necessary information for the control of the vehicle will be accesses by the aforementioned serial connection to the communication module

C. Feedback Information

In order to adequately move the cockpit the proper information concerning the car's disposition will need to be implemented. This information will be read from a MINIMU-9 sensor board that contains a three axis gyroscope, three axis accelerometer, and magnetometer. For the scope of this project the gyroscope and accelerometer will be utilized. The pitch and roll will be the components simulated by the cockpit, and thus on each sensor one measurement will be neglected The information given from the sensors is a 16 bit unsigned integer. These readings will be calibrated to be used by the actuators on the cockpit to actually move the seat. This is detailed in section VII.

D. Motor Controller

In order to adequately control the motor a circuit to do so needed to be established. To begin two signals are issued by the MCU to the controller circuit. One signal is a clock that tells the shift register that information is arriving, the second is the actual information. Eight bits of data that hold the necessary information for the control of the vehicle are passed to the shift register. Seven of the eight outputs of the shift register are placed through a resistive binary equivalence circuit that will generate an analog voltage. The other signal is places through a small resistive circuit to create a voltage that can be compared to the first output. Both of these voltages are passed into a LF351N that will be used in a comparator configuration. The output of this op-amp will be used to bias the gates of a set of MOSFET's that will actually be delivering the power. Figure 1.1 shows the motor controller.

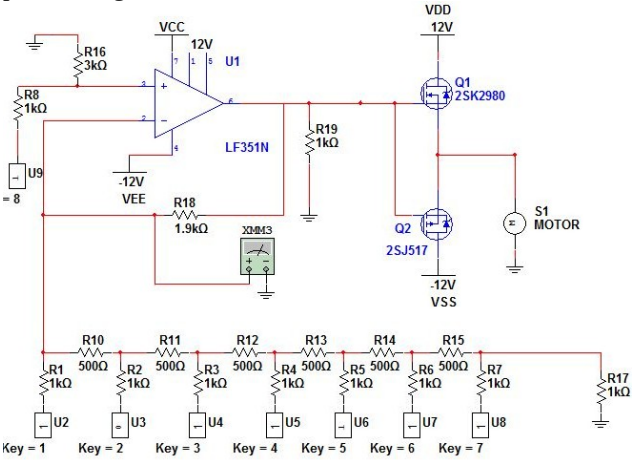

Figure 3.1 The motor controller

IV. Cockpit Fabrication

The Cockpit base is comprised of a 4 foot by 5 foot sheet of diamond plate fastened to a plastic pallet of the same length and width specifications via 12 self tapping screws. The diamond plate is rigid and thick enough to withstand the force the actuators exert on them, while the pallet adds more strength to the base as well as the weight needed to keep the Cockpit from sliding on the floor when in operation.

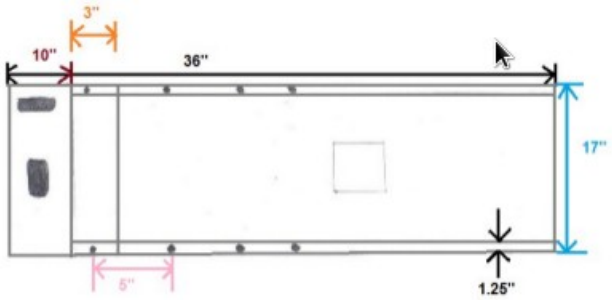

Figure 4.1 Cockpit Floorboard

The base is composed to a top layer and a bottom layer with honey combed columns (12) connecting the top and bottom. The actuators are each placed above a respective column to ensure the strength and stability needed from the base. The height of the base was measured to be 5.5".

This was necessary to the design since it gave the needed height to ensure that the front of the floorboard would never come into contact with the ground.

The Cockpit floorboard was fabricated using 3/8th inch (.375) diamond plate aluminum sheet material and 1inch by 1inch, 3/8th inch think angle iron. The dimensions of the floorboard is 17 inches wide by 5 feet long. The angle iron was used to that it runs the perimeter of the the floorboard and has a series of 12, 1/4" thick holes drilled through the top of it. A sheet of diamond plate aluminum with is placed on top and also has a series of 12, 1/4" thick holes drilled through it that line up with those on the angle iron and heavy duty screws are used to fasten the diamond plate to the angle iron.

Each actuator is connected to the floorboard by two angled pieces of aluminum, or "tabs". In the base of each actuator there is a hole of approximately 1 inch in diameter to allow for connection. There is also a hole through each tab so that a nut, bolt, and washer can securely fasten the bottom of each actuator to the base. The orientation of the tabs on the base are same for the front two actuators and opposite for the back actuator.

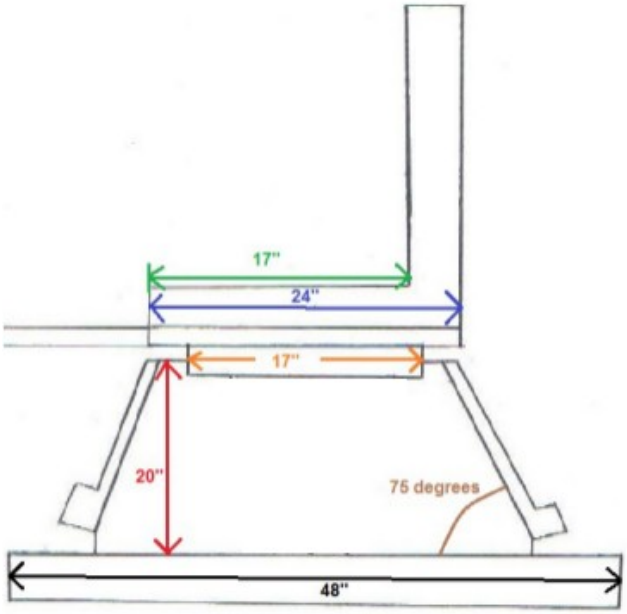

Figure 4.2 Cockpit Seat Configuration

The front tabs are oriented so that the bolt that will connect the actuators run length wise with the floorboard and the back tabs are positioned so that the bolt runs along the width of the floorboard and are orthogonal to the front two bolts. The tabs were "TIG" welded to the base to ensure the most secure connection available. The front two tabs are placed 23" away from each other and are 3" out from the floorboard. The back actuators tab is 2" behind the back of the floorboard, bisect the front two actuators, and is 24" away from the center line of the front two actuators.

On the bottom of the floorboard 3 bolts were "MIG" welded and will be used to connect the top of the actuators to the floorboard. The heads of the bolts were welded to the frame to allow a nut to be placed to ensure that the actuators were connected securely with nuts. The bolts has a thread thickness of 5/8th of an inch and are strong enough to withstand the various forces that are exerted when the actuators are in motion. The front two bolts extend outward from the floorboard perpendicularly, and are 4 inches in length and have a non-threaded or smooth shaft portion that extends from the head of the bolt 2 inch out to the threaded portion where a locking nut will be placed. The back bolt was welded to the floorboard as well, except instead of being welded such that the bolt faces out, it was welding on the inner part of the angle iron and faces in. The bolt has the same specifications as the front two and works in the same manner.

To connect the actuator to its respective bolt on the floorboard a heim joint was used on the front two. The heim joint was welded to the top of the actuator head using a MIG weld. The heim joints offer a f safe and secure connection of the actuators to the floorboard while also allowing for the articulation needed to achieve the desired off set. The heim joints have a larger diameter that the bolts by approximately 1/8". This will allow for the bolt to slide easily through the heim joint and also move in any direction through the joint without binding or rubbing.

There is a 4 foot length of aluminum square tubing, that is 3 feet from the back of the floorboard and 2 feet from the front. It is TIG welded on the center line of the floorboard. The video display unit and steering column connect to this. The video display unit is located at the top of the square tubing and is connected by two bolts, the top of which is 2" below the top of the column. There is a joint which allows the monitor to pitch up, this bracketing is what is connected to the column. The monitor will be connected to the bracketing with a series of 4 screws.

The steering column is located on the center line of the column, 2 feet from the tab or the bottom. It will run

perpendicularly to the the square tubing that the video display unit is fastened to. The steering column is composed of 12 inches of 1.25 inch thick square tubing, and 6 inches of 1.125 inch thick square tubing. Each has been predrilled with 4 holes, this allows for the steering column to be extended or retracted by the user. A plastic steering wheel from a golf cart will be used.

At the end of the floorboard will be two pedals. They are also constructed of diamond plate. A spring loaded hinge join is used to provide the resistance and also return the pedals to the same positioning. The pedals will be connected to a length of diamond plate that is 17 inches wide by 12 inches long with 3 screws on each pedal. The platform that the pedals are connected to will be connected to the floorboard in the same manner as the steering column, with predrilled holes and a cotter key. This will allow the platform to move forwards and back at the riders' discretion.

The seat is bolted and welded to the to the top of the floorboard. Four bolts connect the base of the seat to the floorboard with the back 2 bolts 3 inches from the end. The back two bolts are welded to the floorboard and the front two are bolted through the diamond plate portion of the floorboard. The seat is a 1997 Chevy Silverado drivers' seat. The seat has the ability to slide forwards and back on the track located on the bottom and also can be tilted forwards and backward. A single lap belt is installed and fastens using the seats' seat belt belt receiver to ensure that the rider of the simulator is safe during the entirety of the experience.

V. Cockpit Hardware Design

The cockpit hardware is focused heavily around the processes of the micro-controller. The processing platform utilized is the Netduino. The Netduino is controlled by an AT9SAM7X512 Atmel 32-bit micro-controller with a speed of 48MHz, 6 analog inputs, and 14 GPIO (of which, 4 are PWM). The Netduino is programmed in it's own version of C#. The subsystems of the cockpit hardware include “Operator Controls”, “Transceiver Communication”, and “Motor Control”. All the hardware is contained in what is currently known as the “Power Box”. The printed circuit board is powered by a 24V 1A supply. This supply is regulated to 12V to power the Netduino. All circuitry of the cockpit uses a common ground, however, the positive end of the power supply for the PCB is different than the positive end of the power supply for the actuator movement.

A. Actuators

The actuators utilized on the cockpit were essential to the design. Servo City 450lbs. Thrust DC Actuators capable of 6 inches of extension were used. These are intended to be operated at 12V at a stall current of 20A. These actuators move 2.89in/s with no load, and 1.90in/s with max load. The current rating is not fully utilized, due to weight constraint specifications of the project. A 15A fuse is used in series to prevent excessive current draw. The actuators are equipped with 10K potentiometers, which are used in programming, as to not attempt to over extend the actuators, but also as a means of setting up a drift function to allow the cockpit to return to a stable center position over time.

B. Operator Controls

The “Operator Controls” subsystem is the direct interface utilized by the user. These include digital inputs from the “Turn-On Switch” and the “Reverse Control” switch, and analog inputs from potentiometers in the steering wheel, the gas pedal, and the brake pedal. The reference is connected at ground to low, and the 3.3V from the Netduino's supply to high.

C. Transceiver Communication

The “Transceiver Communication” subsystem utilizes a UART serial port connection with the Xbee. The Xbee is powered by 3.3V regulated from the 24V supply to the PCB.

D. Motor Control

The “Motor Control” subsystem takes up the majority of the mass of the “Power Box”. A visual representation of the circuit can be found in Figure 5.1. Input for the 3 Motor Controllers is received from 3 of the PWM channels from the Netduino, and is changed into a DC signal using a buck converter(seen left of the operational amplifier), which is applied to the gates of three paralleled power MOSFETs (IRFP250N), which are operating in the linear region. These are driven on the drains by a Duracomm 24V 50A unipolar switching power supply. Because this is not a bi-polar supply, use of a relay at the source of the MOSFETs to invert the signal to actuators must take place. These relays are driven by smaller relays on the PCB, which changes state from 3 “sign” GPIO pins. It should be minded that due to high current utilized by the actuators, proper precautions had taken place to keep the power MOSFETs cool (i.e. heat sink and fans, compensating wire width). It should also be noted that the MOSFETs pass a large amount of current, so wire is soldered to the pins of them, rather than having them

directly connected to the PCB. Diodes have been placed across magnetic devices to prevent voltage back feed in use of mechanical operating pieces. The PWM for each motor controller operates at 500Hz, using high widths ranging from 0-69ms, allowing approximately -14 to14V to the actuators.

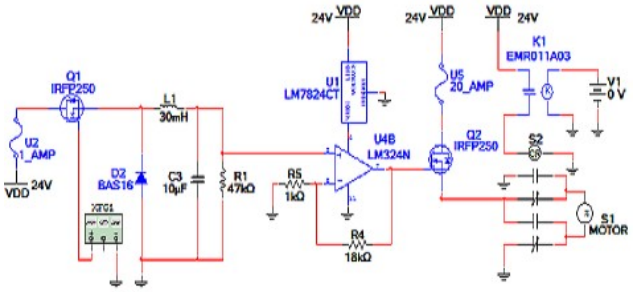

Figure 5.1 DC Actuator controller

VI. Software Overview

In order for the vehicle and the cockpit to interact correctly two separate software implementations must be created. In one instance, it will be necessary for the car to always receive information from the cockpit to control it's movements. In the other instance, the cockpit will always need accurate and up to date information in order to properly mimic the forces that the vehicle is undergoing. To adequately account for both of these positions the software must synchronize and pass information at least ten times per second to keep the delay to less than 100ms. The synchronization needs to be done is such a way so that it is not possible for either MCU to enter into an infinite loop. This will be done by allowing one MCU to reset in the case where too much time has elapsed between synchronizations. The MCU on the vehicle will preform this task to ensure that there are no software lock ups. In addition the calculations will need to be done quick enough to allow for plenty of data transmission between the systems. This will be done by distributing the calculation workload between the MCU's, such that the delta time in between synchronizations is approximately equal. The MCU on the vehicle will control the vehicle, gather sensory information, preform calculations on the information as to fine tune it, and transmit and receive data. The MCU located on the cockpit will gather feedback information about the actuators, control the actuator movement, gather sensor information for the pedals and steering wheel, and transmit and receive data.

VII. Vehicle Specific Software

A. Overview

The software on the vehicle is being written in a slightly modified version of C. There consists a group of functions for various tasks that will need to be done on a regular basis. Figure 7.1 shows the necessary functions.

```
void setup();
void loop();
bool synchronize();
void dataTransfer();
void controlMotors();
void accelRead();
void gyroRead();
void SWprint(byte data);
byte SWread();
byte[] calcAccel(int x, int y);
byte[] calcGyro(int x, int y);
```

Figure 7.1 Necessary Functions

The MCU begins by running the setup() function, which will declare all necessary ports and set all the variables to their respective initialization states, from setup() controlMotors() is called once to ensure that the vehicle does not move. The program then moves onto the function called loop() where it will stay for it's entirety. The first action of the loop is the synchronize() function where communication is established with the MCU. This function calls dataTransfer() which uses the SWprint() and SWread() commands to properly move information, in addition a watchdog timer will reset the MCU in the case that synchronize does not occur. After communication has been properly established and a predefined set of dummy information is read in, it is now time to do something useful. The MCU then calls accelRead() and gyroRead() to acquire a set of information. Following the acquisition of this data calcAccel() and calcGyro() are called to modify the necessary variables so that data can properly be transferred. Now, the MCU moves onto actually controlling the vehicle by calling the controlMotors() command. This is the end of the loop and this cycle will repeat indefinitely while the vehicle is on.

B. Synchronize()

In order for the synchronize() function to not default, communication with the cockpit must be established within two seconds, otherwise the MCU will reset to allow for the proper acquisition of data. This function will read in three bytes of information, which will be the start byte, motor control byte and the servo control byte.

The MCU will then transmit the five bytes of information necessary. See section VIII A for more information.

C. ControlMotors()

Because the motor controller has been designed for the specific purpose of this project, interfacing with it is not inherently intuitive. This function will output a clock signal that will be used to clock in data to the shift register. Another signal that holds the binary data will need to be ready at each instance of the clock pulse. This clock should be intermittent and only be initialed when the data in memory is changed, otherwise the data being moved in is frivolously altered.

D. SensorRead()

Because of the similarities between gyroRead() and accelRead() they will be explained as one. The sensorRead() functions utilize an I2C bus at 100kHz in order to acquire the data necessary. These function take the 16 bit uint values from the registers on board the respective chips and place them into the global variables that are associated with them

E. SWprint() and SWread()

In order to create a UART serial connection it will be necessary to establish functions that will preform the connection, because the MCU does not inherently support such functionality. Because of this issue, two GPIO's from the MCU will need to be used as a software serial connection. In order to do this the output of the MCU needed to be correctly manipulated using an oscilloscope and adding the delays associated with a 9600 baud connection.

F. CalcAccel() and CalcGyro()

The information that is drawn from the sensors is raw data, and will need to be adjusted in order to be properly used. For the purposes of this project, the 16 bit uint values will need to be truncated to 8 bit uint values. In addition because of the sensitivity of the sensor it is possible to gain inconsistent data. For these reasons the accelerometer data will be processed by a polynomial function of the following form.

$$s(r) = \alpha_1 r^{(\alpha_2)} + \beta_1 r^{(\beta_2)} + c \qquad (1)$$

The corresponding value s(r) represents each sensor readings 16 bit uint value that has been altered to cancel out the jitter, while the coefficients are a function of input and the previous input in an attempt to linearized the data, but to do so in a simple, quick algorithm. In order to cancel out the drift that occurs over time when using a gyroscope it will also be necessary to use a filter with the accelerometer data to calibrate the gyroscope. A complementary filter was used in this case. The filter is implemented in the form of (2)

$$\theta n = \alpha(\theta_{n-1} + \delta\phi) + \beta\psi \qquad (2)$$

The new angle is calculated by multiplying the gyroscope reading by the difference in time since the last reading and adding it to the old angle. This value is multiplied by a coefficient and the added to the accelerometer reading multiplied by a coefficient. The coefficients are calculated on a trial and error basis that is determined by the sensitivity of the sensors being used. It is also noteworthy that the coefficients alpha and beta need to add up to one in order to satisfy the criteria of calculating the new angle. At this point a new angle has been established and this can be used to show the disposition of the vehicle. This new angle will be used in conjunction with the old angle and divided by the change in time to calculate change in angle. See (3).

$$\delta\phi_{calc} = (\theta n - \theta_{n-1})/\delta t \qquad (3)$$

The measured change in angle and the calculated change in angle will then be added and divided by the two to get the average rate of change that will then be used by the cockpit. This is seen in (4);

$$\delta\phi_{new} = (\delta\phi_{calc} + \delta\phi)/2 \qquad (4)$$

At this point the information is ready to be truncated to its 8 bit uint form and sent off to the cockpit for use in the programming. The information that will be sent is the value calculated for the new rate of change in both pitch and roll, as well as the normalized acceleration values.

VIII. Cockpit Specific Software

Due to the nature of the processing platform, it should be noted that output pin values can only be adjusted in the main() function. The ever-lasting while loop initializes by using the “turn-on” switch to high, and ends when the switch outputs low to the switch pins.

A. Serial Port Communication Functions

Serial port communication is quite easily simply defining the serial port, and commanding it. Transmission takes place in the main function, however, the receive function

is defined as an event handler, which stores the data in the buffer of the processing platform . Three bytes are submitted to initialize the communication between the RC Car and the Cockpit (1 handshake byte, 2 data bytes – 1 for the DC motor, the other for the steering servo utilizing the steering wheel, gas pedal, brake pedal, and reverse switch). The event handler requires 6 bytes from the car: handshake byte, 2 bytes for pitch and roll, 2 bytes for velocity of pitch and roll, and a final handshake byte. If both handshake bytes are not received, then it is claimed that the transmission has errors, and ignores the data until resubmitted. Every 20 cycles of receiving, the buffer clears.

B. Calibration Routine

In order to allow for a similar user experience for each person riding the simulation, regardless of weight and height, a short calibration routine will be implemented to establish a set of coefficients that can be used by the cockpit to better cater to the individual. The calibration routine will extend each actuator fully and determine the rate of change in actuator velocity and position with respect to time. These coefficients are important and will be later used in the Drift Function.

C. Actuator Control Functions

The pulse width, direction, and actuator displacement are determined by the data received from the RC car, as interpreted by the RC Car's Atmega328P MCU. Thus, the pitch and roll velocities are scaled to meet the proportional displacement of the RC Car. However, the displacement of the actuators is not safe to assume based on the duration of the PWM, so the displacement of the actuators indicated by the potentiometers they are equipped with will be utilized. '0' displacement will be further indicated by a lack of change in velocity or pitch and roll of the vehicle, and will steadily displace itself to center extension on the actuators at a reasonable speed. Actuators do not freely move, however. To achieve the desired pitch and roll, actuators work together to meet the goals of realism and safety. It is essential to maintain the center of gravity of the apparatus on top of the actuators to some limiting factor. The reaction voltage found by scaling (5) and (6), will be calculated by the processor using the values from the *Potentiometer Calculus Function* after passing the values calculated from extending actuators to solve for $d\ell/dt$ (the velocity function output). The value of the change of angle of the retracting actuator(s) must be the additive inverse of the change of angle of the extending actuator(s). This is absolutely necessary, as not all people have the same center of gravity or weight. Safety is of the greatest concern, and so is stability in this project. Capital letters denote constants. Let H be the height of the central axis relative to the bottom of the apparatus, W be the horizontal distance of the bottom of any actuator from the central ball and socket, R be the distance of the top of any actuator from the origin in which rotation takes place, θ_1 be the angle made between the top of the actuator to the to the central ball and socket (relative to being set completely horizontally), θ_2 be the angle made by the bottom of an actuator to the top of an actuator (relative to the normal axis from the ground plane), L be the fully retracted length of the actuator, and ℓ be the length of extension by the actuator. Then, the system can be modeled as follows:

(5)

$$\frac{d\theta_1}{dt} = \pm \frac{d\chi}{dt}\left(\frac{H^2\chi + W\sqrt{H^4 + H^2W^2 - H^2\chi^2}}{(H^2+W^2)\sqrt{H^4 + H^2W^2 - H^2\chi^2}}\right)\left[1-\left(\frac{\sqrt{H^4 + H^2W^2 - H^2\chi^2} - W\chi}{H^2+W^2}\right)^2\right]^{-\frac{1}{2}}$$

$$\ell = \sqrt{2R[H\sin(\theta_1) - W\cos(\theta_1)] + H^2 + W^2 + R^2} - L \quad (6)$$

Where

$$W = (L+\ell)\sin(\theta_2) + R\cos(\theta_1) \quad (7)$$
$$H = (L+\ell)\cos(\theta_2) - R\cos(\theta_2)$$

$$\chi = \frac{1}{2R}[(L+\ell)^2 - H^2 - W^2 - R^2] = H\sin(\theta_1) - W\cos(\theta_1) \quad (8)$$

$$\theta_1 = \pm\arccos\left(\frac{\sqrt{H^4 + H^2W^2 - H^2\chi^2} - W\chi}{H^2+W^2}\right) \quad (9)$$

$$\frac{d\chi}{dt} = \frac{d\ell}{dt}\left(\frac{L+\ell}{R}\right) \quad (10)$$

D. Drift Function

In order to create a desired user experience it will be necessary to move the actuators accordingly, however a system in which the actuators slowly return to their center point will be necessary to allow for the continued simulation of the vehicles motion over time. This will be implemented from within the programming on the cockpit via a drift function. For the purposes of this project an approximation of an under-damped response will be used to add an element of realism and to further enhance the user experience. When looking at (11), it shows the exponential function which will be used in the linear regression process to approximate the under-damped system.

$$v(t)=\alpha_1 t^{(\alpha_2 t)}+\beta_1 t^{(\beta_2 t)} \quad (11)$$

The experimental coefficients that will be used in (11), are to be established in the aforementioned calibration routine that will be run at start up. This will allow the specific under-damped response approximation to be catered to the individual rider, but also allows the programmer the freedom to adjust the coefficients based on the real world variations needed in a project of this scope.

E. User Input

In order to properly use the analog input that is being taken in by the potentiometers located on the steering wheel and the pedals, the Netduino must read this value and convert it into usable information. The analog input is truncated and the most significant 8 bits are placed into a single byte for each sensor. The information for the steering is complete at this time, however the total velocity value that the car needs will need to be a function of the gas pedal and the brake pedal. To easily determine the value, the uint from the brake pedal will be subtracted from the using of the gas pedal. If the car is to fully stop the value that should be sent is 00xH

VII. Safety Precautions

Safety was a concern in the making of the prototype of this project. High speed actuator motion could become a risk to the operator, if proper care is not taken. Thus, mitigation was necessary. The use of a truck seat provides comfort and stability for the back of the user. Programming the cockpit so that it would not cause the actuators to move rapidly also protects the rider. A three point harness will also be used to protect the rider, as well as a helmet.

High-current application also warrants preventative measures. High-current application could cause some components to fail, thus posing a high-cost risk. Also, high current makes some components hot, thus, causing minor explosion. Thus, the power box was made to protect the user from such an event, and the system was designed in a robust enough fashion to protect the apparatus.

VIII. Conclusion

To conclude, all specifications and goals have been met. The RC Car is responsive and robust, smaller than 24”x24”, weighs less than 5 lbs, and can operate at approximately 100 feet from the cockpit. The Cockpit can support an operator up to 300 lbs, has an angular velocity greater than 8deg/s(under max load), has independent pitch and roll of +/-15deg, consumes less than 1200W. Robust control was also established in a high-current application. Standards of safety were met.

Biography

James Russick is a senior at the University of Central Florida. He plans to graduate in May 2012 with a Bachelor's of Science in Electrical Engineering. After graduation he plans to continue to expand his business, Epixam LLC.

Paul Shimei is a senior at the University of Central Florida. He plans to graduate in May 2012 with a Bachelor's of Science in Electrical Engineering. After graduation he plans to work for Bionetics Corporation in Cape Canaveral.

Adolph Arieux is a senior at the University of Central Florida. He plans to graduate in May 2012 with a Bachelor's of Science in Electrical Engineering.

Acknowledgment

The authors wish to acknowledge the assistance and support of Richard Barrett of ITT Exelis for the continued support and assistance in the process of establishing a working prototype

References

[1] Netduino. Secret Labs LLC , 2010. Web. 2 Oct 2011. <http://netduino.com/>.

[2] "450lbs Thrust Linear Actuator." Servo City. N.p., 2011. Web. 22 Oct 2011. <http://www.servocity.com>.

[3] "Force Dynamics 401 Specifications." Force Dynamics. Force Dynamics, n2009. Web. 18 Sept 2011. <http://www.force-dynamics.com/401/>.