Autonomous Visual Rover

Diante Reid, Sean Day, Liem Huynh

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

***Abstract* — In this paper we present the design and methodology used in the creation of the Autonomous Visual Rover. The main objective of the AVR is to use the CMUcam2+ and blob detection to accurately track a blue object in the x, y, and z planes. Furthermore, it will keep a programmable distance of 6 to 10 inches from the object it is tailing and will adjust accordingly to a turn or stop by the object. Therefore, the AVR will never actually come in contact with the object it is tracking. This will be accomplished with a 0% failure rate.**

***Index Terms* — image color analysis, image processing, mobile robots, mobile robot motion- planning, object detection**

I. Introduction

As the scope of modern technology expands and the demands of everyday tasks become more stringent, it is important that the designs of robots are flexible in order to meet and exceed our demands. This means that robots must be versatile, user friendly, and efficient; with that in mind the AVR will meet all of the previously stated expectations. Our basic design for the AVR is a simple yet proficient mobile robot which uses image processing to see and track objects.

Our inspiration for the AVR is to create a device that can help even out the battle field on the war front and keep soldiers safe. Recently the war front has changed dramatically from a normal theater such as a rural country side to urban environments including cluttered cities and remote villages where the chances of collateral damage are immense. The AVR could be deployed to investigate and eradicate suspicious packages such as IEDs (Improvised Explosive Devices) without endangering the lives of soldiers.

The AVR is a light weight, quick and efficient rover equipped with obstacle avoidance and image processing capabilities. The base of the AVR is the combination of the body of a dissected remote control car and a platform made out of wood to hold the components. The CMUcam2+ vision sensor will be mounted on three servos and connected to our custom designed printed circuit board, which will also interface the power supply and weapons system. The AVR will be able to follow and track an object based upon the CMUcam2+ image processing abilities. It will keep a programmable distance of 6 to 10 inches from the object it is tailing and will also detect when the object turns or stops and will adjust accordingly to each action. Therefore, the AVR will never actually come in contact with the object it is tracking.

The main components that encompass the AVR and allow it to function properly are a custom designed PCB, CMUcam2+, one ultrasonic sensor, pan and tilt servos and drive train system. The PCB design is based on the Arduino USB microcontroller and includes and Atmega 328 chip, header pins for input/output ports, three 7805 voltage regulators and two capacitors. The CMUcam2+ is interfaced with the chip on PCB using a hybrid C language that is used to program the Atmega328.

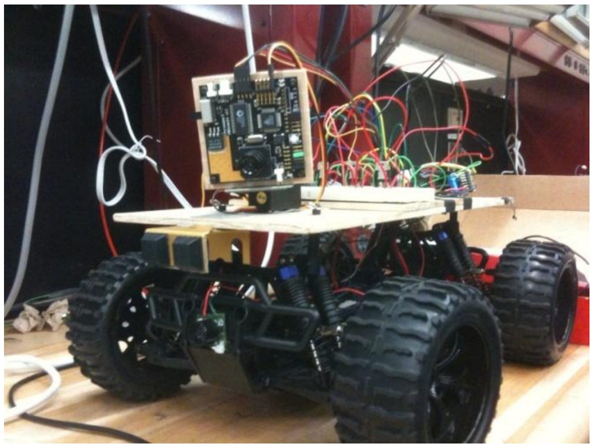

Fig. 1 The AVR

II. Arduino Microcontroller

Although there is a vast variety of microcontrollers on the market that would be suitable to control the AVR, the ATmega family of microcontroller from Amtel stood out from the rest of the crowd. Reasons of selecting this particular type of microcontroller include its cost effective nature, vast amount of documentations available and ease of development.

With a group comprised of only electrical engineers, an easy development environment is very critical. There are many development tools for the ATmega microcontroller family available but the platform that stood out from the rest is the Arduino. The Arduino is an open-source electronics prototyping platform based on flexible, easy-to-use hardware and software. Being an open-source platform, all documentations such as the reference design files are available to us. The Arduino board is straight forward enough that it can be built by hand or purchased preassembled. Our plan is to start the development on a purchased board and then design our own PCB with all the necessary component of the development board plus extra circuitry that are required for this vehicle using their provided design. Extra circuitry such as H-bridges to drive the motors will be integrated all together with the Arduino board in one compact PCB.

The Arduino platform provides a simple programming environment with a good learning curve that can be picked up easily with minimal programming experience. Being an open source environment, there are a great number of examples and tutorials out there to speed up the learning process.

We chose the Arduino Duemilanove board, which is the latest installment of the platform. One main advantage of this board is that it has a USB interface so the majority of the development work can be done on newer computers without having to go through a Serial interface. Another advantage of the Arduino platform is that the development software is cross platform. This will enable our group to development on both MAC and PC so the works done aren't necessarily confined in the Senior Design lab. This will surely boost productivity.

As far as technical specifications, at the heart of the Arduino Duemilanove board is an ATmega328 microcontroller running at the clock speed of 16 MHz. The ATmega328 has 32KB of flash memory to store codes. This should be more than sufficient for our purpose. Of those 32KB, 2KB are reserved for the Arduino bootloader. As far as Inputs and Outputs, 14 digital I/O and 6 Analog inputs are available. Of those 14 digital I/O, 6 are used to provide PWM output. The PWM outputs are important since we will use them to drive the motors of the vehicle. The Analog inputs will be used to read sensors such as the ultrasonic sensor. Besides the digital and analog I/O mentioned above, there are also two serial ports to send and receive data. TTL serial data will be important since this is how we plan to communicate between the microcontroller and the CMUcam2+. The operating voltage for this board is 5V. Power can be either be supply from via USB plugged into a computer or by an external source such as battery. Recommended input voltage should be around 7-12V but not to exceed 20V.

III. Chassis

The AVR is essentially a robot on wheels, so to accomplish this we chose to use the body of remote controlled truck that uses RC servos to turn. Since we are using an existing RC car as our vehicle's platform, it was easier to find a vehicle and four wheels, rather than a platform with tracks such as a tank. One attractive feature of a track platform is that it is easier to maneuver when in tight spaces. A track platform can very easily make sharp turns or perform a full motion 360 degrees turn without needing much space. To compensate for the less than desirable turning angle, we will have to use better control algorithms working together with the sensors to tell the AVR to avoid such tight spaces.

We chose the traditional two driven wheels on a single motor and two turning wheels as our vehicle's platform. This type of platform can be easily found in most RC cars. Having one single motor for two driven wheels will reduce the amount of control we need to do and no performance will be sacrificed since a single motor is powerful enough to drive the AVR. The two turning wheels will also be controlled with a single servo. We decided that this minimal approach would save us a lot of time and resources to achieve a stable and usable platform for our vehicle.

A. Ultrasonic Sensor

The Maxbotics Company offers multiple types of ultrasonic sensors. The main difference in these sensors is the width of the pulse that is emitted. The wider beam being more sensitive and the narrower beam being better suited for large object detection. The Maxbotics line of sensors also has multiple ways to interface the part with a microcontroller. They offer three different ways to communicate with the microcontroller; pulse width modulation (PWM), analog output, and a RS232 serial output. The pulse width modulation works by sending out a 40 KHz sound wave, when this signal is sent out the PWM pin is set to high; when the echo comes back the PWM pin goes back to low and the length of the pulse divided by two represents the distance to the object. The PWM output gives a pulse width of 147 microseconds/inch, which can be easily interpreted by a simple C program written to the microcontroller. An even easier way to interpret object distance is the analog output. This output gives the object distance by the formula (Vcc/512)/inch. The size of the Maxbotics sensors are small compared to other sensor on the market (.785" x .870"); which is very important because the CMUcamera2+ and PCB board are going to take up a substantial amount of space on the chassis. The sensor itself is not much larger than a quarter; therefore it will take up a minimal amount of space on the AVR. The Maxbotics EZ1 also has the lowest power draw of all the sensors we researched. With only 2ma of current draw it will help prolong the life of the battery between recharging.

B. Motor/Servos

When compared to the other types of motors, the DC motor is the most powerful of them all due to a great amount of research and development for this particular technology. Good DC motors can have up to 90% efficiency but within the scope of this project, the DC motors we will be looking at probably operate around 40-60% efficiency. This is still pretty good for our purpose. A less efficient DC motor will just draw more currents from the power supply. DC motors come in a wide variety of sizes. The ones we will be looking at should be spec around 2-8V. A slight disadvantage of DC motors is that it spins too fast for certain applications. An easy way to get around that is to use it with gear heads to slow down the revolution per minutes. For the AVR, we chose the RS-540RH motor as our main drive train. This is a very powerful motor with an operating voltage range of 3.6-6.0V, 1.4 Amp no load current and 6.45 Amp current draw at maximum efficiency and up to 32Amp when stalled. Due to the powerful nature of this motor, a special H-bridge is required to control it.

Besides the main drive train, the AVR also utilizes 3 RC servos to operate the turning wheel and the CMUcam2+'s pan and tilt system. We chose the HS-311 standard RC servo for this purpose. The servo can output up to 42 in-oz of torques at a reasonable 4.8V and it can be easily control using the built in servo class of the Arduino software package.

C. H-Bridge

In order to output high current to support the DC motor, a special H-Bridge will be employed. We chose the 15Amp motor driver from Polulu for this application. This h-bridge is capable of outputting 15Amp of continuous current and up to 170Amp peak current for situation such as stalling. This h-bridge also comes with onboard control that allows us to use PWM to control the rotation and speed of the DC motor. Using a simple 40kHz PWM, we can control the speed of the motor. The motor speed can be varies using PWM duty cycle and direction can be control using a simple digital output. A HIGH output will enable to motor to go one direction, LOW will drive it the other way. The truth table below will explain all the operations. Although breaking will stop the motors from spinning instantaneously, it doesn't necessary mean the vehicle will comes to stop at the exact moment the breaking process is commanded. A small workaround using software is used to make sure that the vehicle break in a timely manner. The vehicle will be instructed to go backward for a few seconds after the break command is given to cancel out any forward momentum the vehicle has during the course of the run.

Motor Driver Truth Table				
PWM	**DIR**	**OUTA**	**OUTB**	**Operation**
H	L	L	H	Forward
H	H	H	L	Backward
L	X	L	L	Brake

Fig. 2 Motor Driver Truth Table

D. Printed Circuit Board

For compact footprint and a more professional appeal, we plan to combine most of our circuitry onto one single PCB. After the initial prototyping with the development board and breadboard, we will come up with a final design that can be printed out on a single circuit board. We plan to the surface mount soldering technique to put together our PCB. Although it might not be possible to surface mount all the components, for this scenario we will then use through-hole soldering.

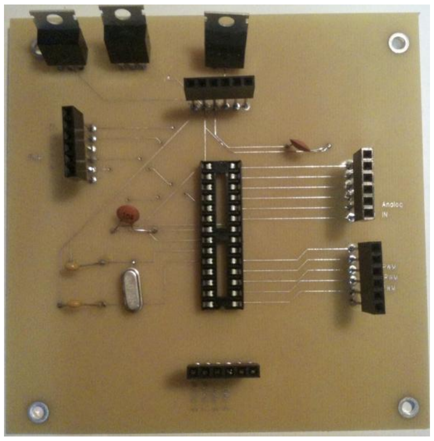

Fig. 3 Printed Circuit Board

The process of printing out a custom circuit board is fairly easy to understand. A circuit design was completed using a PCB design software and then sent to a manufacturer to print out the PCB using their industrial grade equipments. The cost associated with printing a circuit board depends on the size of the circuit board and the number of layers. For our purpose, a 2 layer PCB is sufficient. 2 layers PCB tend to be fairly cheap compared to higher number of layers. Since most manufacturers encourage people to use their custom PCB software, the software to design such circuits tends to be free of cost.

IV. Image Processing

The purpose of this image processing system is to locate a given object based on its color. For the scope of the AVR, a blue object will be tracked. To ensure that the correct color is being detected an LED on the CMUcam2+ will light solid yellow when it sees the object. This feat will be accomplished through blob detection.

A. CMUcam2+

By researching numerous vision sensors and possible configurations, we determined that developing an original image processing system which is more efficient than those already available is beyond the scope of this project. That being said, we chose to use the CMUcam2+, shown in figure 3.8.1, which was created by The Robotics Institute at Carnegie Mellon University. The CMUcam2+ is a low-cost, low-power vision sensor for mobile robots that can perform many different kinds of on-board, real-time vision processing. It was devised using a SX52 microcontroller which is interfaced with an OV6620 Omnivision CMOS camera and communicates via RS-232 or TTL Serial port. It has the following capabilities:

- Image Down Sampling
- Find the centroid of any tracking data
- Gather mean color and variance data
- Gather a 28 bin histogram of each color channel
- Up to 160 x 255 Resolution
- Control 5 servo outputs
- Track user defined color blobs at up to 50 Frames Per Second

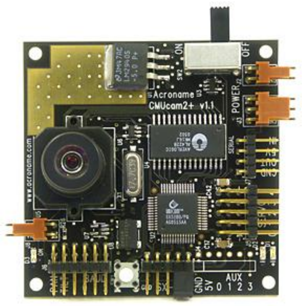

Fig. 4 The CMUcam2+

The CMUcam2+ is connected to the computer through TTL or RS232 serial port, where RS232 is the most common setup used. The CMUcam2+ is able to convert an image in a series of pixels by gathering different colors of light through manufactured grids of boxes on its chip. These boxes are stimulated by the passing light and create different voltages that are proportional to the amount of light present. There is a red channel, blue channel, and two green channels in the CMUcam2+ as well as a range from 16 to 240 value of voltage for each channel. There is an extra green channel because the human eye is most sensitive to green and it also helps fill in the grid to be evenly spaced across the sensor. The sensor contains 356 columns and 292 rows of light sensitive cells arranged as a grid, using the RGB format. The camera module then takes data from 2 sensor rows at a time to generate each line output from the camera module. From there the CMUcam2+ then takes the data and outputs it as a shortened pixel data. Now that the image has been broken into pixels it is possible to track color. This is done by specifying a minimum and maximum allowable value for the RGB channels, in order to specify a specific color. In the CMUcam2+ the maximum value is 240 and the minimum value is 16 for each channel. In our case the values for the blue ball to be tracked are “0 80 0 80 50 255”.

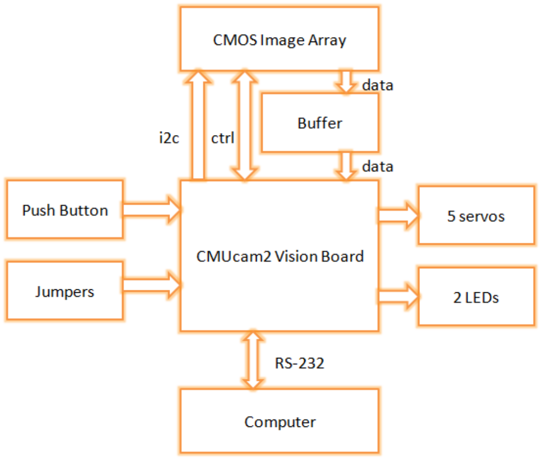

Fig. 5 CMUcam2+ Block diagram

B. Blob Detection

In our project we will use the CMUcam2+ ability to perform blob detection in order to track a blue soccer ball. The CMUcam2+ allows for the blob to be tracked at 50 frames per second. Blob detection is an algorithm used to determine if a group of connecting pixels are related to each other by detecting points and regions in the image that are either brighter or darker than the surrounding. This technique is very efficient when trying to identify separate objects in a scene or count the number of objects in a scene. To perform blob detection using the CMUcam2+ we first grab an image. After that we set the threshold for the limits of what acceptable shades of blue to be seen by the vision sensor. Then we find the middle mass of the soccer ball which is where the average location of all pixels of the selected color occurs. Since we are only tracking one object the middle mass is present in the center of the soccer ball. Using blob detection we have been able to gain the best results tracking the object over other techniques including edge detection and pixel differencing.

V. Power Supply

For this project, we are simply going to prove that the technology used in the AVR is both reliable and practical. The power supply needs to be able to accommodate an array of parts that are going to be present on the AVR.

Part	Watts
Ultrasonic Sensor	50mW
H Bridge	10W
Steering Servo	75mW
CMUCam2+	1.5W

Fig. 6 Power Needs Table

The entire vehicle is going to run off a 7.4V lithium ion battery rated at 2000 mAH which should give the vehicle a substantial run time between recharging. Lithium ion technology was the obvious choice because no other battery technology has a higher milliamp hour rating. When different parts on the AVR are operating they are going to cause voltage drops to occur; this could cause damage to a part or cause a part to act abnormally. To prevent these problem voltage regulators will be used to sustain a constant voltage and eliminate any question on how much voltage a certain part is seeing. We are using the 7805 voltage regulator, which can handle up to 1A of max current. This is more than sufficient to power any part on the AVR. The max current draw is going to come from the CMUCam2+ which draws about 200mA of current during operation. All of the parts except for the CMU camera are going to run on a 5V supply voltage. The Ardunio Duemilanove board is capable of supplying 40mA of current per I/O pin that is able to power most of the parts on the AVR.

VI. System Architecture

The flowchart below shows a top-level overview of the AVR. This flowchart dictates the flow of data and all the processes involves in this autonomous vehicle. The target and environmental variables will be detected using the CMUcam2+ and the Ultrasonic Range Finder. The CMUcam2+ will detect a target and then process the information to obtain the target's centroid location. After processing, the output of the CMUcam2+ will be a string of various data to be interpreted by the Sensors Manager. The Maxbotix Ultrasonic Range Finder behaves the same way. Ultrasonic signal bouncing off the target will be picked up by the range finder. The range finder will use those signals to create a Pulse-width Modulation signal and output that to the Sensors Manager. From those two sources of information, sensor manager will be able to find out how far is it away from the target and where is the target's centroid. Those data will then be passed onto the controller to perform calculations that will ultimately be converted into voltages (Pulse-width Modulation) to drive the actuators. The actuators here would be the DC motors mounted on the chassis. This closed loop system will keep looping until the termination condition is met.

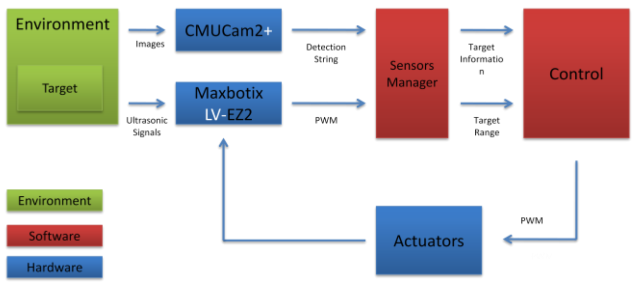

Fig. 7 System Architecture Overview

For this vehicle to be fully autonomous, it must be able to navigate itself through the environment. Vision based navigation will be a big part of this project but additional information from various sensors such as the ultrasonic sensor and the movement of the vehicle will also come to play. To stay within the scope of this project, the operating environment will be controlled. Most likely the environment will be in a brightly lit, indoor, room temperature, leveled ground with minimal obstacles. The vehicle must be able to avoid simple obstacle and avoid collision along its way to the target. Information about its surrounding will be obtained through various sensors.

VII. Algorithms

A. Sensors Manager

The sensors manager acts as an executive function that will mainly deal with the hardware-software interfacing of the vehicle. All raw sensor data will go through this function and the sensors manager's job is to process these data so it can be fed to the controller.

B. Ultrasonic Sensors Data

The onboard hardware of the ultrasonic sensor will do most of the processing job but some work is still need to be done in order to translate the output data into the real distance between the sensor and its detected object. There are many possible ways to read the output data of this sensor such as direct connection to the serial port, analog data or digital data. The easiest way to interpret these data using a microcontroller without taking up the TX and RX port is through the Pulse Width port. The PW port of this ultrasonic sensor will output data in form of a pulse-width modulation signal. The distance can be calculated by dividing the pulse width by 147 microseconds per inches. For example: if an object is 10 inches away from the sensor, a value of 0.00147 will be read from the sensor's output using the microcontroller's digital in port. The microcontroller will then take this value and divide it by 147 microseconds to get the 10 inches. It is also possible to convert such values to the metric system using the microcontroller if desired.

C. CMUcam2+ Data

Detecting and tracking an object can be a very difficult task without the right hardware. The sophisticated CMUcam2+ greatly simplified the process. One of the operating modes of the CMUcam2+ is color tracking. Simply put, the camera will use some signal-processing algorithm to isolate a given color within its line of sight and extract information about the location. Within the scope of this project, the operating environment of the AVR will be controlled. The target will be a bright colored object. Colors such as green, red, blue will most likely be used. The CMUcam2+ will be programmed isolate this bright colored object from everything else within its line of sight. The centroid of this bright colored object will be extracted and sends back to the microcontroller. Along with the location of the centroid, a confident level will also be sends back. Using the location and level of confident given, we can successfully track the object as it moves through time. The objective now would be to follow the object and always keep the centroid within sight. The detection data coming from the CMUcam2+ is pretty reliable since it has already been processed and filtered by the on board hardware.

Reading data from the CMUcam2+ is more complicated than the Ultrasonic Sensor. Since the CMUcam2+ is a much more complex piece of hardware. It has many operating modes and output data. For our purpose, we want to extra the location of the centroid of the detected object. To make this process easier, two types of functions have been created to read and write information to the CMUcam2+. The reason we need two types of function is because we don't always expect to get information back from the CMUcam2+. Only certain commands require the CMUcam2+ to talk back. A "Set" function will enable us to command the CMUcam2+ without expecting any information back. Such commands that do not require any information back include: reset, set color, idle, etc. The other types of commands such as requesting a centroid position or servo position requires the CMUcam2+ to talk back to the microcontroller. This type of command will be called "Get" commands. Get commands expect to receive something back from the CMUcam2+ through the serial port.

Before any type of information can be extracted, the camera must be initialized to our liking. The camera will interface without microprocessor through the UART serial RX and TX port. Various things must be set first before meaningful data can be extracted. First and foremost, we have to enable raw serial data transferring and set the right baud rate so the camera would be able to communicate with the microcontroller. Baud rate effects how fast the data are coming into the microcontroller. For our purpose, a slow baud rate of 9600 is good enough.

After the connection has been established, a command from the microprocessor will tell the camera which color to track. This data must be format in a way that the camera will be able to understand. The format of this command should be in the form of a series of RGB values. The microcontroller must specify which color it wants the camera to track. 6 values will be given to the camera from the microcontroller. These values are as follow:

[Minimum Red Maximum Red Minimum Blue Maximum Blue Minimum Green Maximum Green].

After a specific RGB value has been initialized, the camera will start to track that color. If an object of the color is found within the camera's field of vision, the camera will start to track that color. Information about the object can be extracted using the color tracking command. The microcontroller will then ask the camera to output the available data of the centroid of that tracked object. The output of the camera will be a string of raw data that will need to be interpreted by the sensors manager function. The format of the raw data should follow: [mx my x1 y1 x2 y2 pixels confidence panservo tiltservo] with mx and my being the x and y location of the centroid of the tracked object. The x1 y1 x2 y2 values follow make up the output of the rectangle that is the region of the camera's field of vision. From those 6 values, the microcontroller will be able to calculate and see where exactly is the object is relative to the vehicle. The confidence level will also give us a good sense of how good the raw data is. Using a certain threshold in the sensors manager, we can reject data with low confidence to ensure the accuracy of the location of the tracked object. Lastly, we have panservo and tiltservo which are the electrical angles of the Pan and Tilt servo. From these electrical angels, we can convert them into actual mechanical angle to see where the pan and tilt servo are actually pointing.

Raw data coming from the CMUCam2+ are in a string format with each value delimited by a space. This string of data will require some parsing in order to convert the string information to integers for the controller to do calculation.

From the location of the centroid, the sensors manager function can calculate the location of the tracked object in relative to the location of the vehicle. This information will be useful to the controller because by knowing where the object is we can successfully command the vehicle to move towards it. Along with ultrasonic sensor's range, we now have the exact location of the tracked object.

D. Controller

The controller will use the telemetry data coming from the sensors and determine how to control the actuators to perform tracking and following. Before any action is taken place, the controller must first take the given telemetry data and make sense of it. Since the AVR employed a Pan and Tilt system, we now have two coordinate frames to deal with (Body frame and CMUcam frame). Some calculation is required to align these two coordinates together. As seen in the figure below, the body frame will be fixed and CMUcam2+ frame will make an angle with the body frame as the panning servo move in the horizontal axis. We will call this angle beta. Since all of the target's data coming from the CMUcam2+ will be in a form of x and y coordinate of the centroid, we just adjust this centroid value to represent the real target's centroid location relative to where the vehicle's body is pointing. The target's centroid information coming from the CMUcam2+ will be aligned to the body frame by adding an offset using the following equations:

$$offset = 44\sin(\beta) \quad (1)$$

$$centroid\_b = centroid\_c + offset \quad (2)$$

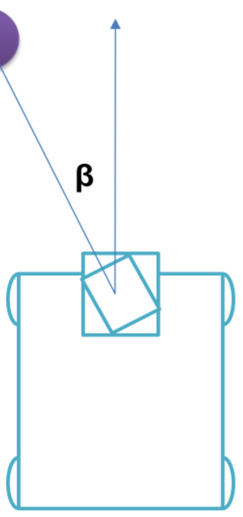

Fig. 8 Coordinate Frame Alignments

To avoid fault detection and have a better confident in the tracked object, the controller will save the last 10 looks in an array and keep statistics of how many detections acquired within the last 10 looks. If the percentage of detections is higher than a certain threshold, the controller will know that it's a good detection and then move on. If it's a bad detection, no more action will be required until a good detection is found. Once a good detection is found, the controller will use the centroid value of the tracked object and determine how to command the turning wheels and the drive train. The flow chart below will explain the basics step that the controller will go through to determine the right commands for the actuators.

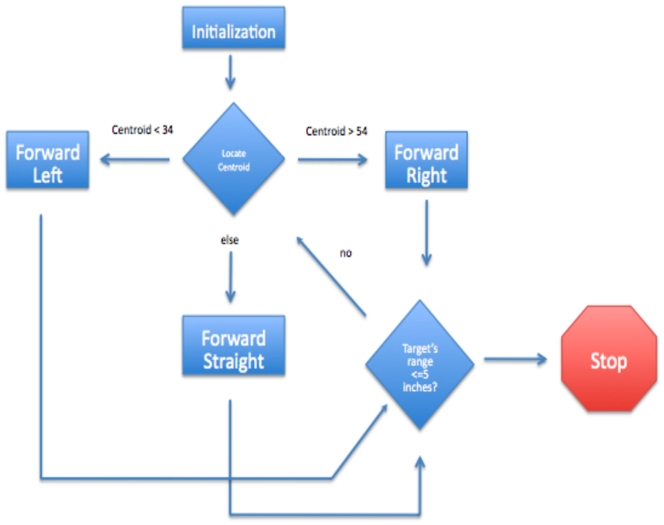

Fig. 9 Guidance Flowcharts

The speed of the AVR is relative to the distance between the AVR's body and the target's body. The further away the target is, the faster the DC motor will spin. And as it gets closer to the target, the speed will be adjusted in accordance to the relative distance between the two bodies. This whole process is done through a simple proportional controller. The speed of the motor is controlled through the H-bridge. Using a PWM at 40kHz frequency, the speed can be specifies by the amount of the duty cycle. The duty cycle of the PWM can range from 0% to 100%. Since the DC motor we use is very powerful, 100% duty cycle would be overkill and a waste of battery power. For our purpose, we are varying the duty cycle from 10%-40% depending on the proportional controller. Anything more than 40% duty cycle won't be necessary.

VIII. The Engineers

Diante Reid

Diante Reid is currently a senior electrical engineering student at the University of Central Florida. After graduation, he plans to enter the workplace with a company such as Lockheed Martin or Harris Engineering. The fields which interest him most are power systems and communications. Once securely established in the workplace, he plans to pursue a master's degree in business administration.

Liem Huynh

Liem Huynh is currently a senior electrical engineering student at the University of Central Florida. After graduation, he plans to work for the Johns Hopkins University Applied Physics Lab with the Signal and System Analysis Group within the National Security Technology Department while pursuing a master's degree in Electrical Engineering with specialization in Robotics and Control.

Sean Day

Sean Day is currently a senior as the University of Central Florida. After graduation, he plans to enter the job market with a focus on microelectronics and signal processing. While working full time as an electrical engineer Sean plans to study for a master's degree with a focus on microelectronics.

Acknowledgement

The authors wish to acknowledge the assistance and support of Dr. Richie. Also, we'd like to thank Dr. Flitsiyan, Dr. Haralambous, and Dr. Wasfy for taking time out of their respective schedules to participate in the presentation of our project.

References

[1] *Arduino Serial Servo Control*. (n.d.). Retrieved 6 7, 2009, from http://principialabs.com/arduino-serial-servo-control/
[2] Charles L. Phillips, R. D. (1999). *Feedback Control Systems*. Prentice Hall.
[3] *Freeduino*. (n.d.). Retrieved 6 1, 2009, from http://www.freeduino.org/
[4] *Power Regulation*. (2009, July 27). Retrieved July 27, 2009, from web-ee: www.web-ee.com
[5] *PCB123*. (n.d.). Retrieved 6 28, 2009, from http://pcb123.com/
[6] *PROGRAMMING - COMPUTER VISION TUTORIAL Part 3: Computer Vision Algorithms*. (n.d.). Retrieved August 6, 2009, from SOCIETY OF ROBOTS: http://www.societyofrobots.com/programming_computer_vision_tutorial_pt3.shtml
[7] University, A. R. (2003). *CMUcam2 Manual v1.06 for the CMUcam2 v1.0 firmware.* Retrieved August 6, 2009, from http://www.cs.cmu.edu/~cmucam2/CMUcam2_manual.pdf