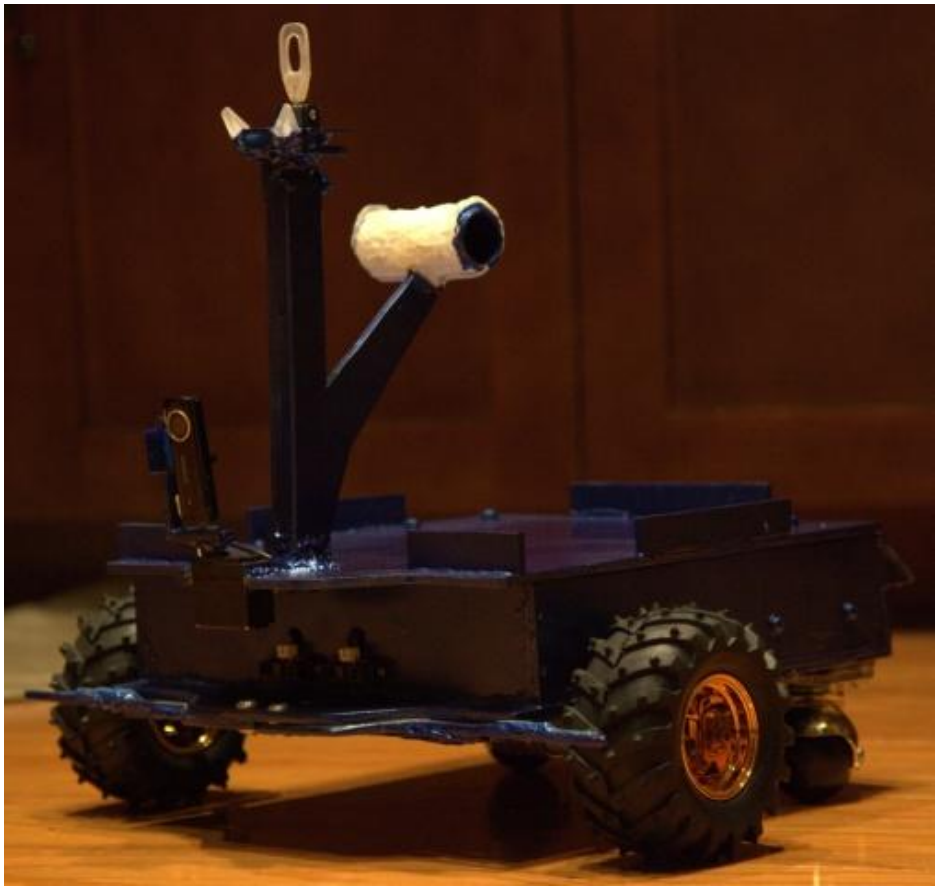
Bird Buggy

Andrew Gray
December 4, 2012



University of Florida
Department of Electrical and Computer Engineering
EEL 5666 – IMDL – Final Report
Instructors: A. Antonio Arroyo, Eric M. Schwartz
TAs: Josh Weaver, Tim Martin

Contents

Abstract

Of all the different bird species in the world, African Grey parrots are the smartest. There intellect allows them to mimic human speech and perform logic thinking. African Grey parrots are so smart; experts compare them to human children:

Figure 1: Mature African Grey.

"The African Grey parrot is considered one of the most intelligent birds and is said by experts to have the cognitive ability of a six-year-old." [1]

Taking advantage of the bird's intelligence, a robot was designed to allow parrot guided transportation. This paper describes the BirdBuggy, an autonomous docking, parrot driven robot. BirdBuggy is able to be guided by a mounted parrot which communicates through a joystick. The Buggy is also able to visually identify a base station with a web camera which is used to dock itself. Able to operate outside of an urban environment, the robot has been outfitted with all terrain tires.

Executive Summary

BirdBuggy is an autonomous robot created to allow a parrot to move about an area without making a mess. The second objective of the BirdBuggy is to autonomously dock with a base station when the parrot has been removed. Wireless communication between the human and robot is done with an XBee wireless communicator.

Propulsion is provided by two forward motors with shaft encoders. Two omni-directional casters were added for stability. Two bump sensors and two IR sensors were placed on the front for obstacle avoidance. Inside the Buggy resides the lipo battery protector, microcontrollers (MC) one and two, the BeagleBoard (BB), merge board, XBee, and a two cell 5 amp hour battery. On top of the buggy are two servos, a web camera, the perch with a two axis joystick, and two LEDs for parrot feedback.

When BirdBuggy is first turned on, the lipo battery protector monitors the batteries to ensure that they are acceptable. After the batteries have passed, power is allowed to flow to the power bridge. From the power bridge, the 7.4v is passed to the BB, MC one, and MC two. The MCs take less than a second to begin, however, the BB takes almost a minute. When the BB is ready, it establishes communication with the two MCs. An audible alert signifies that the system is ready to operate. BirdBuggy starts in parrot slave mode where the parrot controls the movements. MC two monitors the joystick, bump sensors, and IR sensors. The data from the

sensors are sent to the BB via USB. In slave mode MC one monitors both shaft encoders on interrupts and controls the activation of the two parrot feedback LEDs. Commands sent to the board via USB are translated into wheel movements. Thus, when the joystick is moved forward, MC two sends the data to the BB, which in turn sends translated data to MC one which moves the Buggy forward.

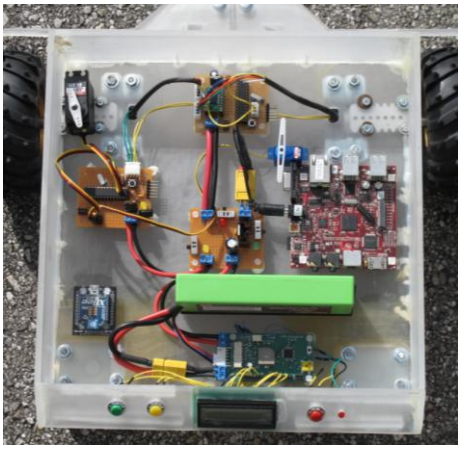


Figure 2: BirdBuggy's internals.

In docking mode, the Buggy becomes autonomous and activates the web camera. The x-axis servo begins panning the camera left and right. Images from the camera are converted from red, green, and blue to hue saturation and value (HSV) on the BB. Thresholds are applied to the HSV images isolating the green sphere color and orange sphere color. Once the Buggy has identified the base station, the orientation of the green sphere in relation to the orange sphere is determined. The Buggy then moves in a perpendicular arc, tracking the spheres with the camera, until the green and orange spheres are vertically aligned. Next, the Buggy turns to face the base station and begins driving forward. Both IR sensors are monitored and when the base station is eight inches away, the Buggy stops, sounds an audible alert and changes to slave mode.

Introduction

Figure 3: Sound activated squirt gun.

Pepper, the parrot, has been a part of our family for over a decade. However, Pepper produces terrible shrieks when left alone.

To combat the shrieks an automated sound triggered squirt gun was created to deter the bird from shrieking (fig 3). At first, the squirt gun proved effective. The bird's screaming would cease after the squirt gun would fire. However, the effectiveness of the water gun did not last. Pepper started utilizing the water as a bird bath and continued to shriek. Next, a remote controlled noise maker was placed near the bird's cage (fig 4). Whenever the bird shrieked, a user could press a button to activate the noise maker startling the bird. This too did not last as the bird became adjusted to the rattling of the noise maker.

Figure 4: Noise rattler.

Instead of startling the bird into muteness, allowing the bird to roam around the house may be a better option. However, because of the messes the bird leaves behind and the possibility of the bird getting stepped on, roaming the house un-attended is not an option. If he could be placed on a mobile platform that could move about the house, hopefully he would stop screaming. Thus the idea for the "BirdBuggy" was born.

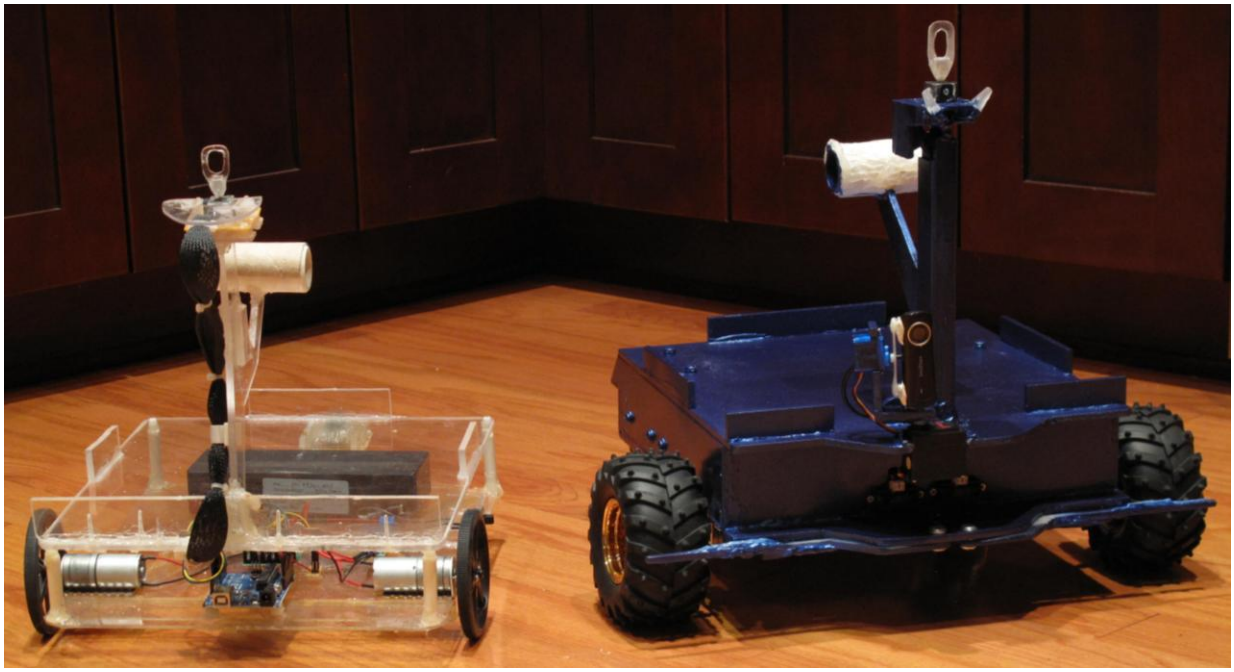


Figure 5: Original BirdBuggy on the left with the new BirdBuggy on the right.

The original Bird Buggy is an open-loop, two-wheel drive, tri-cycle platform powered by a 12v battery and controlled by a MC. The parrot is placed on the perch and interacts with the Buggy through a joystick. The movement of the joystick corresponds to movement of the cart. Because of the open loop feedback, the cart has several deficiencies: it is unable to drive in a straight line, and it cannot avoid obstacles. The goal of the new "Bird Buggy" is to solve the previous problems and to add some additional functionality to the platform.

This report will cover the construction and details of the new "Bird Buggy" autonomous parrot perch. The buggy will operate in two primary modes: parrot slave (PS) and return to base (RTB). In PS, the parrot will be in control. Moving the joystick will result in movement of the perch. Upon parrot driving completion and parrot return to cage, RTB will be initiated via remote control.

Integrated System

BirdBuggy uses a BB for the high level decisions and two MCs for low level decision making. Shaft encoders, bump switches, IR range finders, a joystick, and a web camera provide information for the BB to make decisions. The block diagram in below shows all of the components and the communication ties between.

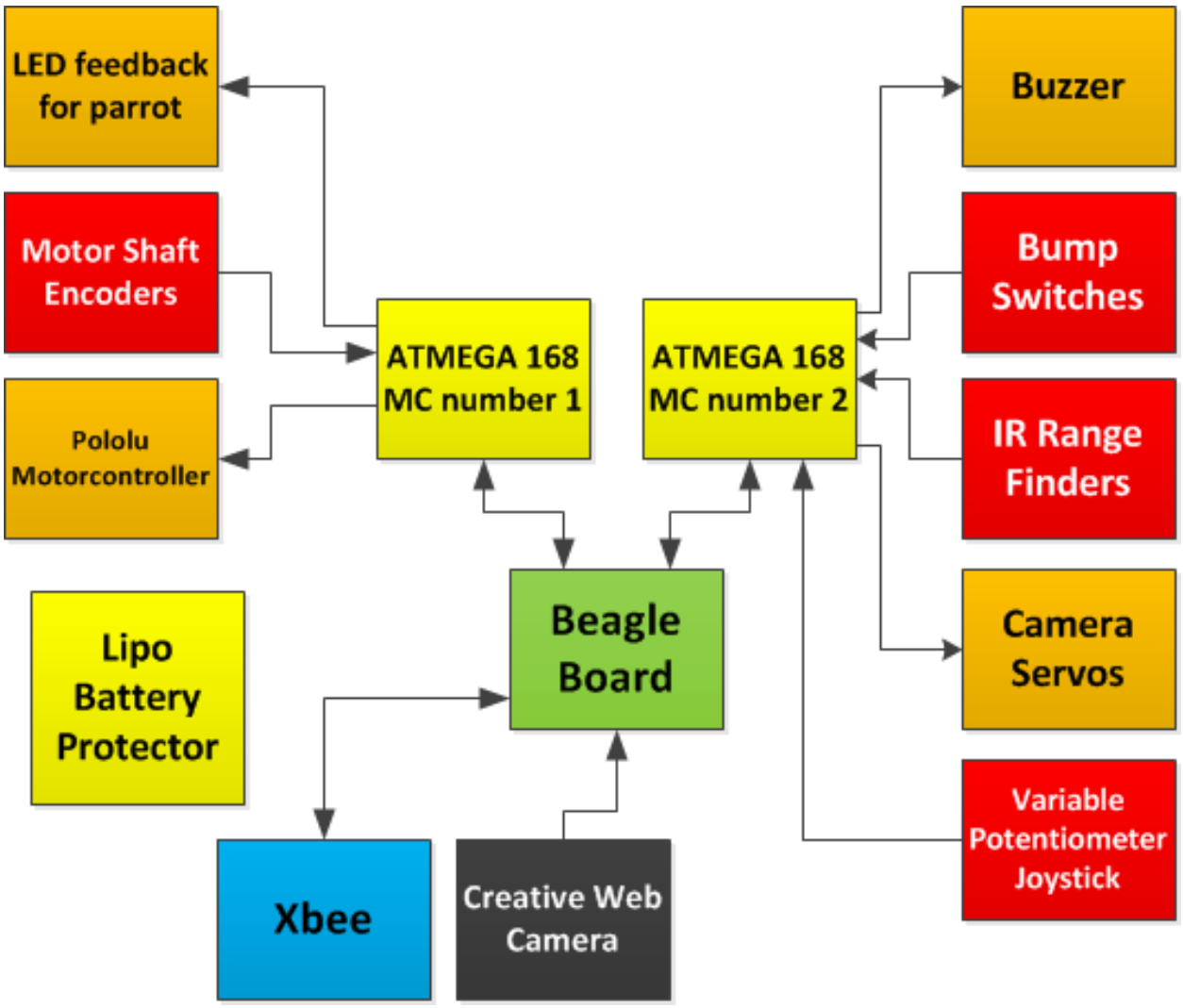


Figure 6: BirdBuggy system block diagram

Mobile Platform

Figure 7: The spring loaded hinges made troubleshooting easier.

For stability reasons, a two wheel drive system with casters for stability was utilized. Aggressive all terrain wheels were selected for off-road capability and some shock absorption. Both motors were mounted to aluminum L brackets. The weight of the robot bent the brackets and U-shaped clamps were added to add rigidity to the motors and to protect the shaft encoders. The casters with spherical wheels vise cylindrical wheels minimized the amount of twist applied to the robot when transitioning from a turn to moving forward. A previous model of the BirdBuggy would sometimes be off course by up to five degrees because the caster was dragging.

The majority of the platform is made of polycarbonate plastic bonded together with two-part epoxy. Taps and set screws were added to increase rigidity. Capped spacers were used to provide a pivot point for the bump sensor arms. A PVC pipe was covered in silicon epoxy and used as the stand for the perch. All wires connected to the perch were passed through the polycarbonate stem holding the perch up. This protected them from the parrot's destructive beak. Two spring loaded hinges were installed inside the Buggy for easy access to the internal components.

Actuation

The camera pan and tilt system consists of two servos and an L-bracket. A small HI-TEC servo provides tilt and a larger HI-TEC servo pan. Able to pan and tilt up to 180 degrees, the platform provided a widerange to allow a large viewing angle for the camera.

Two Pololu motors drive the Buggy and provide wheel position with shaft encoders. A Pololu motor controller provides two H-bridges for forward and reverse direction of the motors. MC one operates the motor controller using pulse width modulation while monitoring the shaft encoders.

In order to ensure that both wheels spin at the same speed a proportional integral derivative controller was created in software. After fine tuning, the controller enabled the wheels to maintain a constant RPM while minimizing overshoot. This resulted in very straight movements because of the controlled wheel speeds.

During slave mode, when the parrot is driving, MC two monitors the 2-axis joystick. The positions of the joystick are broken into five areas: forward, backward, left, right, and stop. When the joystick is placed in one of the areas, MC two continuously sends directional commands to the BB. The BB then relays the command to MC one. MC one takes the command and sets a decaying counter based on the number of shaft encoder ticks. If the command is repeated, the counter is reset regardless of whether it had finished or not. For obstacle avoidance, MC 2 restricts what is sent to the BB. If a bumper is triggered, MC two sends a bumper signal to the BB and ignores the joystick commands until the avoidance maneuver has been completed. The BB relays the message to MC one which conducts a backing left or right turn depending on which bumper is hit. Should the IR sensors detect an obstacle in front, MC two will sound an alert and will not send a command for the joystick positions of: forward, left, and right. The parrot will be forced to move backward.

During autonomous docking mode, the BB controls the servo position and the movements of the Buggy. Initially the BB uses the camera pan and tilt system to identify the base station. Once found and the orientation is determined, the BB orients the Buggy so that the camera is centered on the base station in a window between two bearings. While the camera tracks the base station, the Buggy moves forward orienting itself about the camera moving left or right until the camera is once again between the two bearings. Movements during this stage are more drastic with greater speed and distance per move. When the spheres are vertically aligned and the Buggy begins its approach, the movements become less drastic allowing for greater accuracy during the docking stage.

Sensors

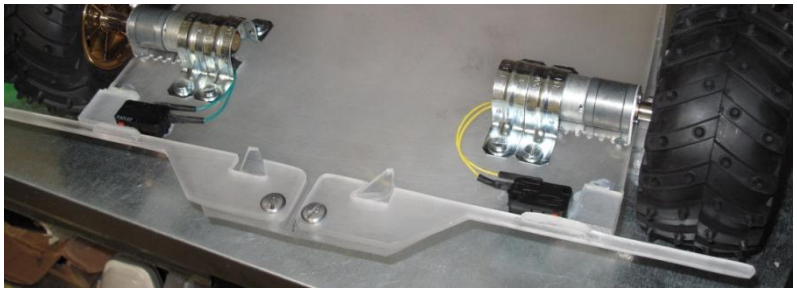


Figure 8: Bump sensors in contact with the bump "paddles" underneath the Buggy.

Contact switches were recycled from an old microwave and put to use as the bump switches. 5v were applied to one end with the other connected to an interrupt with a pull down resistor. Pressing the switch connected the circuit pulling the interrupt pin high. Bump "paddles" were created to increase the contact area of the switch.

The joystick was made from a 10K 2-axis variable potentiometer. 5v, ground, and two wires for analog voltage readings were connected. On top of the potentiometer was placed a loop like piece of polycarbonate for the parrot to manipulate with his beak. Values from the analog to digital converter (ADC) pins were used to determine in which direction the bird wanted to move.

IR sensors were used to detect obstacles without physical contact. The sensors are used during slave mode to prevent the parrot from hitting an obstacle and during docking mode to detect the base station. Each sensor was supplied 5v and ground. The analog signals were monitored by MC 2 on two ADC pins.

Figure 9: Pepper the parrot using the joystick.

For visual identification and base station position orientation, a Creative Live! cam chat HD was used. The camera provided 640 by 480 image resolution and a frame rate of up to 30 frames per second (FPS). However, the BB had difficulty with the resolution resulting in a low FPS (~one FPS). The low FPS severely slowed the Buggy because the docking pace was determined by the speed of the image processing. To combat the slow speed, during search mode, the resolution was reduced to 320 by 240 (~three FPS). During the approach phase where movement speed is reduced, the image resolution is reduced further to 160 by 120 speeding up the frame rate to around five FPS. The camera was mounted vertically (90 degrees clockwise) allowing the pivot point to be closer to the perch stem.

Figure 10: Creative Live! cam chat HD.

Behaviors

BirdBuggy operates in two different modes: PS and autonomous docking mode or RTB. When first powered on, the Buggy defaults to parrot slave mode. During this mode, obstacle avoidance and parrot feedback is utilized. MC two monitors the obstacle avoidance sensors providing pre-made maneuvers to avoid objects. Because the parrot does not know how to turn right, red and green LEDs were mounted next to the joystick in order to provide feedback to the bird. When BirdBuggy turns left, MC one activates the red LED. If the parrot turns right, green is illuminated. When the bird has been removed, the owner activates RTB mode via wireless communications using XBees.

Figure 11: BB has identified the base station

RTB mode relies on the web camera for most of its information. The camera image is used to locate the base station, determine the base station orientation, determine the spheres' areas, and to determine the bearing of the station in relation to the Buggy. When RTB is activated, the BB uses the camera to identify the base station. First the image is converted to HSV and the new image is split into images with a green threshold and an image with an orange threshold. Using the thresholds, area of the green and orange values is calculated. If both values are above a set value, the center of the green and the center of the orange blobs are calculated. If the difference between the two centers of the two objects is less than a given value then the BB has identified the base station. If any of the steps fail, the BB pans the servo 8 degrees and starts the process over.

Once the base station has been identified, the orientation of the spheres is determined. Depending on whether the green sphere is to the right or left of the orange, the Buggy will turn until the station bears between 125 and 150 degrees relative or 25 and 50 degrees relative. When the station bearing is correct, the Buggy moves forward. The process is performed until both spheres are vertically aligned. Once aligned, the Buggy turns to directly face the station and proceeds forward maintaining alignment. Movements during this stage are slower and more precise allowing a more accurate dock. The IR sensors are continually polled during this time to determine the distance to the base station. When the station has been detected by the IR sensors and has an approximate range of eight inches, the Buggy stops and is considered docked. Next the Buggy sounds an audible alert and switches back to PS.

Experimental Layout and Results

Bird Buggy went through several iterations in determining how the Buggy would return to the base station. Initially, an IR emitting base station was considered. The base station would actively transmit IR light at a fixed frequency. On the Buggy, several IR receivers would be placed in a way that would allow 360 degree coverage. Operating much like the Roomba robotic vacuum cleaner, the Buggy would have turned toward the base station and closed the distance by driving forward. The IR idea lost interest because of the inability to determine orientation of the base station in relation to the Buggy. IR light that was reflected off a wall also interfered with the Buggy's detection of the base station. Finally, the range of the IR transmitter / receiver pair was not acceptable.

Looking in a different direction, a USB web cam was considered. The camera would allow the Buggy to determine the orientation of the base station relative to itself. Also, the camera could detect the station at a longer range with less risk of focusing on noise.

While the camera seemed like the best solution, it came with many more requirements than the IR base station setup: a webcam, computer vision software, and a processor that can handle video. While there are many different brands of cameras most can broken into two interface types: USB or wireless IP based. The wireless IP camera required an external computer to operate therefore it was not considered. The USB camera required a computer more powerful than the 8-bit MCs being used to drive the motors and use the sensors. The BB was selected for its processing power, cost, and operating system. OPENCV is a must for any robot using computer vision. Python was selected because of its ease of use. Finally, the BB would work best when operating on a lower processor intensive operating system like Linux vice Windows.

Using OPENCV an image was captured from the camera, converted from red, green, blue (RGB) to hue, saturation, and value (HSV). Next the computer used HSV to identify the green sphere and the orange sphere. After the location of the two spheres had been determined, the Buggy would decide on how to maneuver towards the base station.

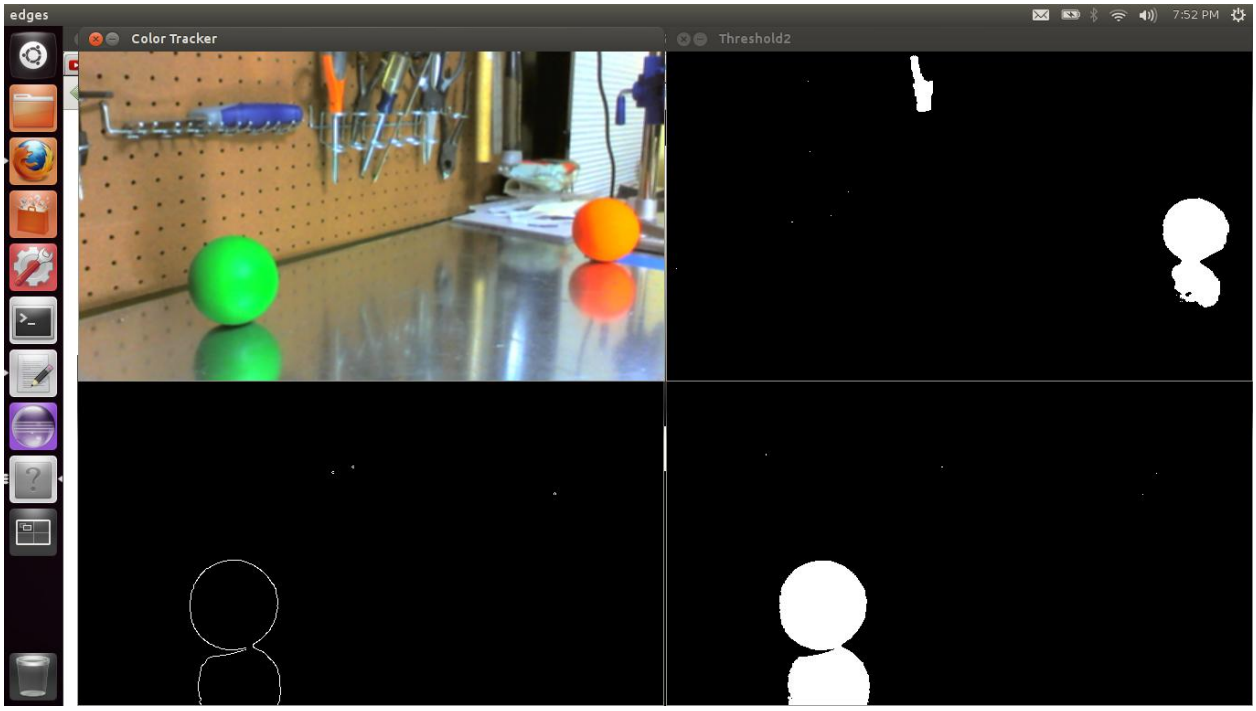


Figure 12: Top left: actual image, top right: orange threshold, bottom right: green threshold, bottom left: green edge detection.

Conclusion

After building the first version of BirdBuggy, I realized that I needed a new platform. The older version consisted of a motor controller, a microcontroller and two motors. Operating open loop, the robot always has a slight left turn and was unable to avoid obstacles. I felt that Pepper the

parrot deserved better and begin laying plans for a new platform. Taking advantage of the freedom of choice for IMDL, I decided to build a new platform for my bird in the class. Before the IMDL, I had never worked with Linux, OPENCV, Python, or an ARM based processor like the one on BeagleBoard. In order for my robot to work I had to learn the new languages, libraries and environments. After completing this project, I now have a few more tools in my tool bag.

While BirdBuggy was able to complete its tasks, there is much room for improvement. While I had timer based delays on joystick inputs preventing rapid forward and reverse movements, the parrot still managed to drive the Buggy erratically. Changing the joystick inputs to a graph based approach would allow the parrot to move with greater resolution and a smoother ride. During the docking mode, calculating how much the camera must move to center on the spheres vice moving a set amount would reduce the number of movements and increase the operating frequency. Finally, a faster processor would greatly increase the speed of docking and would allow for faster movements.

Documentation

- ATMEGA 168 datasheet: http://www.atmel.com/Images/doc2545.pdf
- BeagleBoard-xM: http://beagleboard.org/
- Pololu motor controller: http://www.pololu.com/catalog/product/713
- Xbee 2.4 GHz series 2 datasheet: ftp://ftp1.digi.com/support/documentation/90000866_A.pdf
- Creative Live! cam chat HD web camera: http://www.creative.com/mylivecam/products/product.aspx?catID=1&pid=20164

Appendices

- [1] Newspaper article on African Grey parrot: http://news.bbc.co.uk/2/hi/asia-pacific/7414846.stm
- The computer code can be found at the Bird Buggy website: https://sites.google.com/site/birdbuggy109/eel-5666-files