September 2015

Project Squirrel 2.1

William C. Jones
Worcester Polytechnic Institute

Project Squirrel 2.1

A Major Qualifying Project

Submitted to the Faculty of WORCESTER POLYTECHNIC INSTITUTE In partial fulfillment

of requirements of the Degree of Bachelor Science

Author:

William Jones

Date:

9/10/15

Submitted to:

Professor Michael Gennert, Advisor

Worcester Polytechnic Institute

Abstract

Project Squirrel 2.1 is a direct continuation of Project Squirrel 2.0. The aim of these projects is to develop a robot that can aid the detection of Asian Longhorn Beetles. This project aims to continue the work on the Project Squirrel 2.0 robot in order to have it working as is was designed and then be able to test the capability of the robot. The objective for this project is to have the robot climbing a tree and being able to relay video back to an operator.

Table of Contents

List of figures

Introduction

Project Squirrel is a series of tree climbing robots developed at WPI. The purpose of these robots is to search trees for signs of the Asian Longhorn Beetle (ALB). ALBs spend most of their life inside of trees. Their presence is commonly identified by the holes that they leave in the bark. Infestations end up killing the tree and the rate that the infestation can spread leaves many trees at risk. To stop the ALB and protect the trees $550 million[1] has been spent on eradication programs. With how much is spent to eliminate infestations, making a robot to search trees for infestations could easily be funded. To be successful at the task the robot has to meet certain requirements.

Given the task that the robot is expected to perform it needs to be capable of:

- Detecting the beetles – This is done using a camera to relay images to a human operator.
- Freely moving around on any tree – This is the primary problem that Project Squirrel is attempting to solve.

The current project is to program the latest version of the robot. The robot's ability to climb a tree has not been tested because of unfinished mechanical work. The previous group was able to write a simple server-client application for controlling the robot and several test programs but given the state of the robot, these programs might not be fully tested and refined. By the end of this project the robot should be able to make all of the planned movements and climb a tree.

The primary goals for this project are:

- The robot can move in all the ways that it was designed to move

- An operator can view sensor data from the robot and provide commands for the robot to perform.
- The operator should not be able to give commands that could damage the robot.
- The interface displays the camera feed so that the operator can see what the robot sees.
- To be able to climb 15 cm on a tree with a diameter between 15 and 60 cm
- To be able to grab and release the tree at least 10 times

The secondary goals for this project are:

- Have the robot operate untethered
- Providing the operator with a model of the robot displaying the current orientation of the robot.
- Implement a detection algorithm to watch the camera feed and alert the operator to any possible infestations.
- Be able to climb 30 cm on the tree

Background

Previous Projects

Project squirrel has gone through four iterations thus far and this project aims to pick up where the last ended. Each iteration has been notably different from the others and aims to surpass the accomplishments of the previous version.

The first iteration was the Tree Climbing Robot [2] which was simplistic in design. The robot was a sheet of acrylic as a base and four legs. Each leg had three points of actuation; One allowed the leg to rotate while the other two controlled the angles of the leg. The robot possessed a web-cam allowing images to be sent back to the operator. Unfortunately for the robot the power draw was greater than anticipated, requiring an external power source. The robot was able to hold its position on the tree but the servos could not supply the force needed to climb.

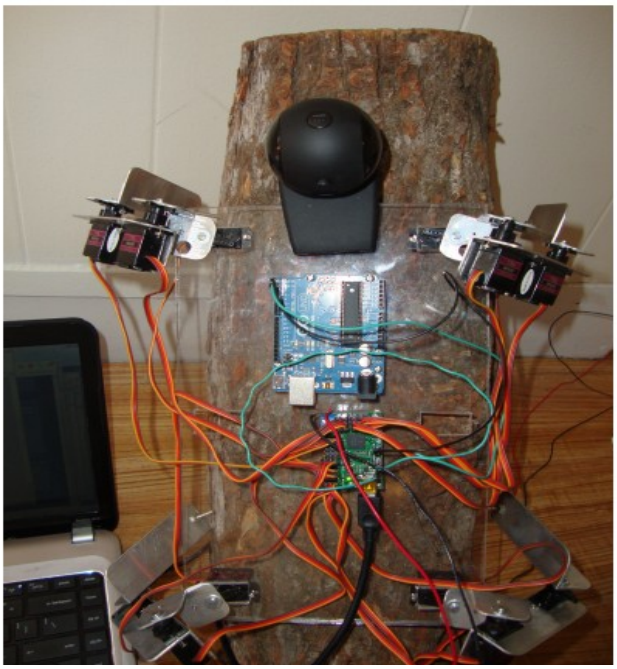

Figure 1- Tree Climbing Robot gripping a sample tree

One of the advantages offered by the first iteration's design is that the legs allow a high degree of flexibility for gripping a tree. The simple design and lightweight parts both help to minimize the weight of the robot which is an important consideration as the robot has to overcome gravity. Having a high resolution camera is a benefit when searching for ALBs since more details can be seen. The biggest problem with this robot is the leg motors. Each of the leg motors drew up to one amp and with the need to constantly power the motors to maintain leg positions on a tree a supply of 8 amps is needed just to hold the tree. This highlights another issue where the motors have to be powered to maintain their position which is an inefficient aspect of the design. A variant of this robot with six legs was tested but the robot did not have the length to keep the legs from colliding with each other. The base for the robot is a piece of acrylic that is mostly unused and could have been trimmed to reduce the weight of the robot.

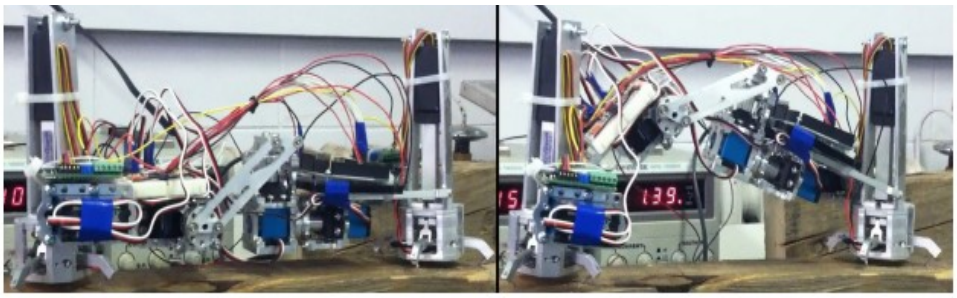

Figure 2-TCR12 Half Stride

The second iteration of the Tree Climbing Robot [3] was a three-segment robot with claws on each end for gripping the tree. The grippers were thoroughly tested to determine the force needed to get a reliable grip on the tree. Each gripper consists of 4 claws that are all controlled by a single linear actuator. For detecting if the gripper was on a surface five push buttons were used on each gripper. The

locomotion of the robot is handled by five servos, three of which provide an inchworm like flexing of the robot allowing it to extend and retract while keeping the grippers facing the tree. The other two servos control the twisting and side to side tilting. This design was able to perform as intended on a horizontal surface but when testing vertically, the motion of the robot caused the gripper to slip.

An advantage offered by the gripper is that controlling it is simple as the gripper is either open or closed. Similarly, the motion of the robot's body is more intuitive than controlling the four leg design of the first iteration. What held the robot back the most were the grippers, which were rather small. The robot has a lot of aluminum in it and while the size is similar to the first iteration, making the base out of aluminum adds a bit of weight to the robot.

The third iteration of the robot [4] used a four-legged design where the legs were spring-loaded. Each leg could move independently of the others however, they were mounted in pairs on the lead screws on both sides of the robot. Figure 3 shows one of the motor pods that drove the lead screws. To help keep the robot stable while it was moving to grab higher on the tree, a spring loaded tail was used to counteract the robot pulling itself off of the tree. For sensing, the robot had limit switches and potentiometers to provide feedback about the positions of the legs. This design was successful at traversing a tree horizontally and was able to complete two gait cycles vertically.

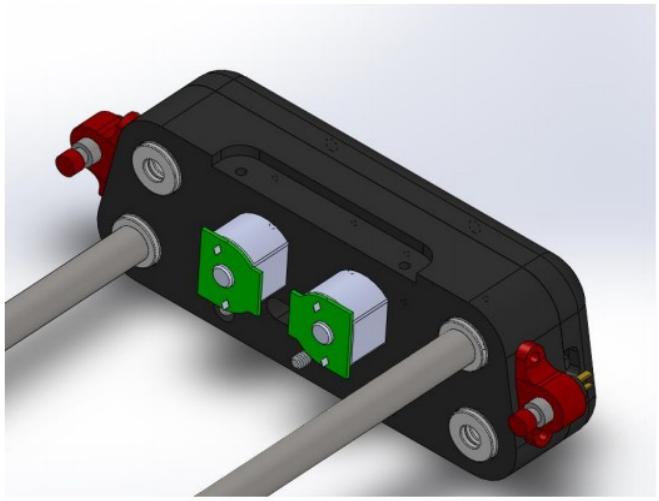

Figure 3-Vex motor pod

The use of spring-loaded legs that had limited range of motion was the key to this robot's ability to climb vertically. They were able to tightly grip the tree much like the gripper from the second iteration while being able to be moved independently allowed the robot to take steps like the first iteration was aiming to do. The limited motion makes the gait simple and easier to implement. The tail gives this robot resistance to the robot naturally wanting to fall away from the tree when only held by the lower grippers. The major disadvantage of this design is that the feet limit the robot's movement to a straight line.

The current iteration of the robot [5] is notably bigger than the previous designs. The robot holds on to the tree using grippers, similar to the second iteration, although in this iteration the grippers are larger and use medical needles as spikes. To extend the robot a lead screw and stepper motor are used and limit switches are used to detect the extremes. The robot is capable of tilting closer or farther from the tree and to the left and right by means of a gimbal. The gimbal is controlled by four motors that are paired together and monitored by potentiometers.

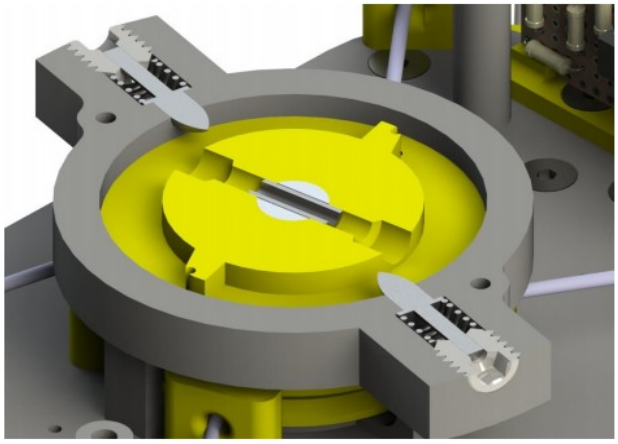

Figure 4-Revised torque release mechanism

The grippers are cleverly implemented to allow a sudden release of tension when the grippers are told to close. Figure 4 shows the torque release mechanism that lets the gripper snap to the tree. The printed parts allow strain sensors to be built into the legs of the robot which provide feedback about how well the grip of the tree is. The use of 3D printed parts helps to reduce the weight of the robot although much of the robot is still made of aluminum. The robot is much larger than the previous three and as a result the weight of robot is also going to be more than the previous versions. The lead screw and guide rod may have difficulty keeping the top half of the robot balanced.

Robot	Weight
Tree Climbing Robot	3 lbs
Tree Climbing Robot 2	2 lbs
Squirrel Robot	6 lbs
Squirrel Robot 2	7 lbs

Figure 5-Weight comparison of all the tree climbing robots

The Raspberry PI

This robot uses the Raspberry Pi B+ to interface with the hardware. The decision to use the

Raspberry Pi B+ was made by the previous team working on the robot and as such the existing electronics are setup to use the Raspberry Pi general purpose input/output (GPIO). In addition to the 40 pin GPIO the Pi has a port for a camera. The camera for the pi is small and light as opposed to the bulky webcam that was used on the first tree climbing robot. The CPU on the Raspberry Pi is a 700 MHz Low Power ARM1176JZFS [6]. With that clock speed the Pi can support an operating system.

As the previous group was not able to leave a Pi with the robot a new one had to be acquired and setup. An 8 GB micro SD card was formatted and NOOBS[New out Of Box Software] was copied to the SD card. Using NOOBS, Raspbian was then installed as the operating system. Raspbian is a port of Debian “Wheezy” a Linux based operating system, for the Raspberry Pi. As part of the setup for Raspbian several configuration options were set. These were to enable the camera, SPI, I2C, Serial, SSH and to not have the Pi start the desktop on startup.

Methodology

Evaluation of the state of the robot

This project is a direct continuation of Project Squirrel 2.0 and dependent on the final state of the robot. Before any work was begun the robot was compared with the design provided in the report and was examined for possible flaws. This evaluation was broken down into three parts, mechanical, electrical, and software.

Mechanical

The first issues noted were about the state of the robot's legs. One needle was missing and another was broken. This was not much of an issue as spare needles were available. A more significant problem was that one of the blocks that held a needle was broken. This was a problem because no spare parts were available. The next issue was that the hardware for controlling the yaw axis of the gimbal was not mounted although it was in with the spare parts. A potentially large issue is that some screws appear to be missing.

Based on the observations it was inferred that the robot could wind up the grippers. Given the state of the gimbal only the pitch axis could be used. The stepper motor, lead screw, and guide rail appeared slightly unsteady but should still allow the robot to extend. The missing hardware on the gimbal however does prevent the use of the yaw axis.

Electrical

The robot does not currently have a micro controller. This and a few missing wires were the only noticeable issues with the electronics. The robot does have a prototyping board with a ribbon cable for connecting to the Raspberry Pi, 7 motor drivers, 16 analog to digital converter (ADC) channels, and an accelerometer. One issue with the board is that there are two sets of power wires that

hang free of the board.

For the grippers the motors were wired to two of the motor drivers and the potentiometers are wired to two of the ADC channels. The top gripper has wires for force sensors in the legs but they are not connected to the board. The gimbal has two motors and potentiometers that should allow for control of the pitch axis. The yaw axis of the gimbal does not have the motors or potentiometers needed to be controlled. The lead screw motor is wired to a motor driver but the limit switch is not connected. With the extra parts was a camera for the Raspberry Pi and two ultra sonic sensors.

Software

The code was neat and organized and appeared to have all of the code needed for the robot to operate. The Java code for the client compiled and was able to display the user interface. Given the electrical and mechanical state of the robot there is concerns about how thoroughly the code has been tested by the previous group. When testing the client and server that the previous group had programmed it was noted that some of their networking code was commented out. Testing for a connection between the client and robot did not provide a working link. The driver level code was tested with much better results. The motor and potentiometer objects were properly implemented and usable. One instance of a gimbal motor had the assigned pin switched which was a minor problem.

Streaming

Being able to stream video back to the user is one of the primary tasks that the robot has to perform. Both OpenCV (Open Computer Vision) and Video for Linux (V4L) were considered and a mix of the two had the best results. V4L was chosen for the ease of creating the stream which can be started with two shell commands. The stream is configured to broadcast an http stream on a designated port. One of the tested viewing methods was to use the VLC media player which is V4L based. The

other method that was tested was a python script that used OpenCV to read and display the stream. The figure below show the approximate delays observed for several streaming methods.

Stream	Client	Delay (s)
VLC (640x480)	VLC player	5
VLC (640x480)	Python w/ OpenCV	2
VLC (1920x1080)	Python w/ OpenCV	5

Figure 6-Video delays from different streams

Wireless

The previous group had acquired a USB wireless adapter. Giving the Pi wireless allows for controlling the robot untethered which is a great benefit for a robot that has to examine a tree and its branches. The down side to using wireless is that it has to be manually configured to connect to a net work. There were no issues with connecting to either a normal wireless router or a mobile hot spot. Connecting to a mobile hot spot is ideal for this robot as it will likely be operated outdoors and it limits the need to reconfigure for another network.

Calibration program

The Setup.py script was designed for testing the basic functionality of the robot and to calibrate the sensors. The script grew to accommodate directional calibration data for the motors to allow software correction. The script begins with a menu allowing the user to calibrate the robot, test the calibration, and exit. When the user choses to calibrate the robot and no configuration exists, the user is directed to the motor direction calibration. During this it walks the user through each motor and has them set the motor to the correct direction. After the motors are calibrated the user is then directed through the sensor calibration. Some of the gimbal calibration is automated to limit the tasks that the user is required to do. If a configuration already exists when the user goes to calibrate the robot it will

give them the option of choosing to use the existing configuration, creating a new configuration, or update parts of the existing configuration.

The option of testing the calibration allows the user to control all of the motors and view sensor data. This mode is meant to give users a minimal interface for tests that require direct control of the motors. There are no software limits and it is possible to damage the robot although this interface is not targeted at the common user.

Server-Client program

The Server and Client programs are designed for the everyday use of the robot. The everyday use case for this robot is that a user turns the robot on, opens the client on a laptop or tablet, uses the robot, and puts the robot away. The reasons that a server-client approach was used are to keep the user tasks simple and to limit the possible damage to the robot that a user could accidentally cause.

When the client is first opened it brings the user to a networking screen and allows the user to specify the IP and port to connect to the server running on the robot. After connecting to the robot the user can then switch to the control screen and begin sending the robot commands. The commands for the gripper are restricted to two states, open and closed. The sensor value that separates the two states is set when the setup is run. The controls for the lead screw allow it to be set to extend or retract, which is determined by state of the limit switches. The controls for the gimbal are arraigned in a 3 by 3 grid and allow for the user to adjust the pitch and yaw of the gimbal as well as returning it to the centered position. Some of the planned features were being able to 'lock' the robot to the tree, integrate the display into the client rather than as a separate program, and having more control over the lead screw.

The client communicates with the server using a specific format where the client sends a command and the server send a response back to the client. The client is configured to wait for the response from the server. The function for handling the interaction with the server needs to be guarded

to keep the Java events from all trying to message the server at the same time. On the server side there is a non-blocking socket in the control loop listening for commands from the client. The use of this non-blocking socket allows for the socket to be in the control loop and not in a separate thread waiting for a command.

The server has several threads for handling the initial network socket and any active connections from a client. When the server is started it creates a thread that handles new connections this is the socket that the client initially connects with. After the connection is established this thread then spawns a new thread that runs the robot control loop and listens for the clients commands. The control loop starts with checking the socket for any commands from the client. If a command has been received it is parsed and the associated action is executed, a response is sent back to the client and the loop continues. The server is able to parse commands for the gripper states, lead screw state, gimbal movement and commands requesting sensor data. This gives the user access to all of the motors and sensors on the robot. Due to the current implementation of the server there are some possible issues that exist. The biggest issue is the possibility of multiple clients connecting at the same time, when this happens there is the possibility of having two instances of the robot controller running trying to move the robot to conflicting positions.

Stepper Motor

During this project it was found that the stepper motor was unable to provide the force necessary to move one segment of the robot when the other segment was fixed in place. This problem was mostly that the motor was not strong enough for the robot. It should be noted that the tension created by the poorly aligned guide rail did have a negative impact on the stepper motor. In order to replace the stepper motor the required torque needed to be calculated.

As the stepper motor is only moving one half of the robot at a time the weight of the bottom which is the heaviest was used as the weight. To account for the weight that a larger stepper motor might add an extra kilogram was added to the 2 kg weight of the bottom. For the coefficient of friction 0.3 was used as the guide for the lead screw is steel. The efficiency was kept low at 10% to keep from under estimating the torque. The breakaway torque of the lead screw was determined to be approximately 0.1 N*m. The last important factor was the angle of the lead screw for which an angle of 90 degrees was used. A safety factor of 1.5 was chosen as an extra precaution to ensure the stepper motor would have enough torque. The calculated torque was determined to be 1.6 N*m and the new stepper motor could supply a torque of 1.2 N*m which just meets the minimum torque without the safety factor. The new motor is a NEMA 23 sized stepper motor and approximately 8 times larger than the motor the previous team had.

While looking into stepper motors it was observed that a NEMA 11 stepper motor, the size of the old motor, would need 12V to operate effectively. The previous team did not take this into account and was driving the old motor with 7.2V and using a motor driver that was only rated for up to 11V. This lack of power was contributing the problems with the old motor. The current motor needs approximately 12V at 1A to be able to move the robot and needs to be supplied power externally as the current system cannot supply 12V. Without updating the robot's power system the robot once again has to be tethered to operate properly.

Conclusion

Results

The robot was tested on its ability to climb a tree in both a vertical and a horizontal position. The horizontal testing was for bench testing the robot and for determining how well the robot would be able to navigate the branches of a tree. For the first stage of testing a section of a 2-by-4 was used to test the basic functionality of the robot. Later stages of testing were done on trees of various sizes.

The tests performed using a 2-by-4 to represent the tree were to determine the baseline performance of the robot in a controlled manner. The grippers were first tested with the robot in a horizontal position to check that the robot could grab the 'tree'. After determining that the robot could get a grip, the holding capacity of the needles was tested and found to be able to hold while the robot was upside-down. The last test of the grippers was to determine if the robot could hang with one gripper which was shown to be possible. The gimbal was tested in both horizontal and vertical positions with the bottom segment fixed in place. The gimbal was able to perform as intended with the exception of the pitch axis being unable to overcome gravity when the robot is horizontal. The test of the lead screw in the horizontal and vertical positions was a disappointment as the lead screw was unable to move without help to keep the lead screw perfectly straight. The other observation made during testing is that the needles blunt easily and become unable to embed themselves in the tree. A worrisome observation is that when the grippers are properly adjusted, they cannot provide the force needed to remove the needles from the trees.

After performing the basic tests on the robot, the grip on different size trees began. The range of sizes used for testing ranged from 5 cm to 68 cm in diameter. The minimum diameter tree that the

needles could reliably grip was 12 cm. The maximum diameter that the robot could hang from was found to be approximately 63 cm. With larger trees the grip of the robot is weakened due to the springs not being able to release all of their stored energy before the needles hit the tree.

More testing was performed after changing out the stepper motor for a stronger one. In a horizontal position the motor did not have any trouble moving the rear segment of the robot. With the lead screw extended the robot was attached to the 2-by-4 and positioned vertically for this test the robot was able to pull up the rear segment about 7.5 cm which is the most climbing the robot had done during this project. It is noted that the increased weight of the stepper motor requires the grippers to be at their best to keep the robot on the tree. The robot was unable to perform a full climbing test due to the failure of the top gripper's outer ring. The ring supports the spring-loaded pins so that the force can be applied to the legs and without the support provided the pins cannot handle the load.

Social Implications

This robot is equipped with a camera and will raise concerns over privacy. The issue has already arisen with cameras on drones and the potential for misuse that is created. This robot will have some of the same problems that these drones have and this could have an impact on their usefulness in the fight against ALBs.

Future considerations

The results obtained from testing the robot make several issues apparent. To get sufficient force to grab the tree the grippers are spring loaded and designed to release when a set tension is reached. To prevent excessive wear on the mechanism the torsion should be set to the minimum required to fully retract the claws. With the gripper properly tuned the the force needed to remove the needles from the

trees cannot be supplied. Additionally with the needles stuck in the tree too much force can cause the string pulling the leg to slip break

While the needles were a good choice for the grippers' claws, they deform to quickly to be practical for the normal operation of the robot. The needles were chosen for the sharp point that they possess but the tip of the needles bend with repeated use, blunting the needle and preventing grip of the tree. In addition to being easily blunted, the weight that the needles have to support leads to the needles bending. One alternative to needles would be to use nails. Nails are similar to needles and would not require large changes to the design.

The lead screw was poorly implemented on the robot. Both the lack of necessary force and the lack of stability contributed to its poor performance. Using a stronger stepper motor is recommended for providing the force needed for the robot to extend and retract. To increase the stability of the lead screw one or two more guide rods should be added. The holes that the guide rods go through should be deep enough to keep the guide rods in square.

The gimbal motors function as they were intended when the robot is vertical but they cannot fight gravity when the robot is vertical. The robot needs to be able to navigate the branches of a tree which can put the robot in a horizontal position. The simple solution would be to just use stronger motors on the gimbal.

This robot is fairly heavy and to save weight some of the parts were 3D printed. The legs on robot were printed but the legs are designed to snap into the tree. Having seen these parts break during testing there is concern for the durability of the legs when used continuously for a full search of a tree. It is recommended that the robot be fully modeled in a CAD program and run a stress analysis to determine what the parts should be made out of to survive the stresses while keeping the weigh low.

Currently to keep the wire organized the ones going to the same sensor or motor are bundled together. To further organize the wires, bundles that are near each other are tied together. The biggest problem with the wiring is that the robot has two sections and there is no ideal way to run the wires from the bottom half to the top. It is recommended that a wiring harness should be added to keep wires from hanging around.

The robot was designed to use force sensors in the top gripper and an ultra sonic sensor on each gripper. The force sensors are in the legs but have not been wired and tested and the ultra sonic sensors have not been mounted on the robot or tested. Adding force sensors to the bottom gripper should also be considered although a third ADC chip might be needed to support the additional sensors.

The client user interface is currently laid out for with minimal consideration for the average user. The first priority was setting up basic control of the robot using a client-server layout not a clean user interface. Now that the basic controls have been implemented more effort should be spent to arrange the interface for users. In addition to cleaning up the interface, the camera display should be merged into the client instead of existing as a separate program.

Despite being unable to test how well the robot can climb a tree, there may be undiscovered issues when trying to climb to a side. The way that the current robot moves it may not be able to make contact with the tree when leaning to the side. The first tree climbing robot may be a better choice for overall mobility on a tree. If the first robot design is revisited worm gears should be used on the joints on the legs so that the motors do not need power to maintain position. To further reduce the weigh the web-cam could be replaced with a smaller camera like the one for the Raspberry Pi.

Resources

[1] Williams, T. (n.d.). Eradication Nation | American Forests[Online].Available: https://www.americanforests.org/magazine/article/eradication-nation/

[2] Read,Erick T.Student author -- RBE, Gostanian,Justin Student author -- CS, Gennert,Michael A.Faculty advisor -- CS. “Design and Construction of a Tree Climbing Robot”. Worcester, MA: Worcester Polytechnic Institute; 2012.

[3] Giovacchini,Ryan Joseph Student author -- RBE, Cobane,Eric James Student author -- RBE, Campbell,Ian A.Student author -- ME, Murray,Thomas Grant Student author -- RBE ME, Hou,Zhikun Faculty advisor -- ME, Gennert,Michael A.Faculty advisor -- CS. “Design and construction of a tree-climbing robot”. Worcester, MA: Worcester Polytechnic Institute; 2013.

[4] Simpson,Matthew James Student author -- RBE, Mistretta,Louie-John Vincenzo Student author -- ECE, Ilacqua,David Christopher Student author -- ME, DeMaio,Emanuel Student author -- RBE, Gennert,Michael A.Faculty advisor -- CS, Stafford,Kenneth A.Faculty advisor -- ME. “Project Squirrel”. Worcester, MA: Worcester Polytechnic Institute; 2014.

[5] Stylos,Alexander Christopher Student author -- RBE, Pounds,David C.Student author -- ID, Brown,Elizabeth Sarah Student author -- ME, Strickland,Michael L.Student author -- ME, Gennert,Michael A.Faculty advisor -- CS, Stafford,Kenneth A.Faculty advisor -- ME. “Project Squirrel 2.0 -- A Tree Climbing Robot”. Worcester, MA: Worcester Polytechnic Institute; 2015.

[6]Adafruit(n.d.). Raspberry Pi Model B+ [Online]. Available: https://www.adafruit.com/datasheets/pi-specs.pdf

Appendices

Appendix A - Setup instructions

Raspberry Pi

Install NOOBS on a micro sd card

Connect pi to a monitor, keyboard and Internet

Using the NOOBS installer, install Raspbian

In raspi-config

- Set password
- Enable Camera
- Enable SSH
- Enable SPI
- Enable I2C
- Enable Serial

Setup WIFI

- https://www.raspberrypi.org/documentation/configuration/wireless/wireless-cli.md

Create a directory for software

Clone the Git repository

Install i2c and smbus

- http://skpang.co.uk/blog/archives/575

Install spidev

- http://raspberrypi-aa.github.io/session3/spi.html

Install vlc

- sudo apt-get install vlc

Start stream with stream.sh

Start setup with setup.py

Start serer with server.py

Client

Install SSH

Linux: openSSH

Windows: Putty

Install nmap

Install Git

Install eclipse and its Git plug-in

Install openCV for python

Acquire a copy of the Git repository

Import the client project from the Git repository

Build client

Edit view_stream.py

On line 5 change the ip to have the pi's current ip

Appendix B - Software guide

Setup

The setup is intended to be run while connected to the robot over SSH. The setup has two modes, one for configuring the motors and sensors and another for controlling the robot. The configuration mode has several options for creating and updating the configuration data. The configuration is split into two parts, motor configuration and sensor calibration. The motor configuration is for setting the direction of the motors using software and was included in the setup to limit the need to move wires around on the robot. The sensor calibration is used to get the limits for the potentiometers so that soft limits can be implemented.

The second mode is for testing the robot while allowing full control of the motors. This allows for basic use of the robot which makes it possible to test the robot. This mode is no intended for average users and does not have software limits on the gimbal or lead screw.

Server

The server is intended to be the program on the robot that the user interacts with via the client. It currently handles connection with the client and can reconnect with the client if the connection

is lost. The server is limited to 5 connections before it exits to make it easier to test the server. The server can currently control the grippers and send sensor data back to the client.

Client

The client has two panels one for connecting to the server and another for controlling the robot. The networking panel allows the user to specify the ip and port to connect to the robot on and allows the user to connect or disconnect from the server. The second panel has buttons to control the grippers, lead screw, and gimbal. Data from the sensors is visible to the user but is currently only updated infrequently.

Stream

The Stream is started by using stream.sh which enables the kernel module and initializes the stream. The stream can be viewed by any program that can be used to watch a stream such as VLC media player. Using the view_stream.py uses openCV and was found to have the smallest delay