eclipse technologies

Proof of Concepts - Project Lycanthrope

REVOLUTIONIZE EMPOWER DESIGN

TABLE OF CONTENTS

Executive Summary **7**

Mission Overview **11**

Design Brief 13

Updated Schedule 15

Approach Summary **17**

Risk Management **21**

Introduction 23

Summary of Risks 24

Risk Assessment 26

Risk Outcome 28

System Architecture **31**

Locomotion 33

Capture Methods 34

Transformation 35

Electrical 36

Materials 37

Tracking 38

Communication 39

Testing **41**

Communication System 43

Vision System 55
Walking System 88
Transformation System 102
Ball Manipulation System 107
Trapping Mechanism 115
Bill of Materials 133
Financial Summary 141
Outreach 143
Index 147
Testing Hardware Specifications 149
Works Cited 150
CARPA Approval 153

EXECUTIVE SUMMARY

Eclipse Technologies is currently developing technology that would save the lives of thousands of American soldiers who carry out highly dangerous search and destroy missions. We have designed a Distributed Autonomous Robotic System (DARS) capable of completing similar search and capture (S&C) missions. Three walking robots - codenamed Alpha, DOG-1E5, and DOG-4S1 - must work together to capture one specific "PreyBOT" from a group of three.

Alpha, initially on four legs, must morph into a two legged robot and place an electrically conductive ball in a cup. However, this ball is initially buried somewhere on the field, so the other two members of the team must locate and retrieve the ball, and deliver it to Alpha. Once Alpha successfully completes his task, it will open up the rest of the field, allowing the DOG's to capture the target PreyBOT. The Telekinesis Module allows Alpha to manipulate the steel ball, while the Lykos Module controls transformation. DOG-1E5 and DOG-4S1 are four legged walkers with maximized hunting capabilities through their Theia and Ares Modules, which are responsible for vision processing, color detection, and PreyBOT capture, respectively. They will use a spring trap and a trailer for PreyBOT capture and transport.

We have passed the initial design phase and we have began testing the various different subsystems and modules present in the three robots. The most essential systems in the DARS have been deemed operational. These systems include: communication, vision, locomotion, transformation, and capture methods. The Communication system consists of the robots' abilities to share information with each other. To test the Communication system, the Command Center will ping each robot and see how fast it is able to receive and transmit pings. Our first priority was to test the functionality of the Locomotion system, since the entire mission revolves around the efficiency and speed of the robot's' legs. We ran the various gaits to see if

they actually worked as needed. Both Alpha and the DOGs were put through different situations to assure they would be able to keep their balance. A few actions specific to Alpha and the DOGs were tested, such as manipulating a conductive ball and digging respectively. For Alpha, we tested the strength of the electromagnet with different metals. The feasibility of the capture methods were assessed, from whether or not BOP would trigger during undesired situations. The Vision system deals with the various methods of tracking and identifying field elements, such as PreyBOTS, colors, and RFID tags. The Transportation system tests insure Alpha's ability to transform back and forth in the allotted time slot and if the sensors could successfully trigger the transformation sequence.

Our company will also reach out to prospective engineering students in middle schools to educate and inform them about possibilities in STEM fields. We believe in creating a lasting impression through these young, driven students. Eclipse Technologies has already reached out to Dodson Middle School and we gave a presentation on the principles of engineering and challenged them to build a structure to withstand an earthquake. Eclipse Technologies is one step closer in inspiring the new generation.

At Eclipse Technologies, we strive to drive change and surpass limitations through research, teamwork, and imagination. By taking on this project, we aim to do what seems impossible, and motivate others to contribute to worldwide technological innovation.

MISSION OVERVIEW

DESIGN BRIEF

The CAMS Advanced Research Projects Agency (CARPA) Initiative has challenged Eclipse Technology to design a Distributed Autonomous Robotic System (DARS) to complete a Search and Capture (S&C) mission. One of the robots, codenamed Wolfgang, must be a dual-mode, polymorphic robot that manipulates a steel ball in order to open the field and allow the other two robots, the PackBOTs, to search and either return or destroy their prey. The purpose of the challenge is to design, build, and test an autonomous, dual-mode robotic system capable of performing the S&C mission successfully returning a specified "enemy" target.

The mission will begin with Wolfgang in a four-legged state. It must traverse the field and find its way into the morphing chamber, where it will be exposed to ultraviolet (UV) light and a specific sound frequency. These will trigger Wolfgang's transformation into a two-legged walker. After three minutes, Wolfgang will revert back into a four-legged state. While in its two-legged state, Wolfgang will be tall enough to activate the switch that gives the PackBOTs access to the rest of the field. The switch itself is comprised of an electrically conductive ball. The ball will be buried under varying colored patches of sand in the field which the PackBOTs will identify and locate the correct colored patch of sand to dig out an electrically conductive ball for Wolfgang. After the barrier is lowered, the PackBOTs will be allowed to enter the field and must scan each PreyBOTs' Radio Frequency Identification (RFID) tag in order to return the correct enemy target. The PreyBOTs will be constructed by an anonymous entity and the design will be revealed on the date of the mission. If the PackBOTs are unable to capture and return the prey while alive, the PackBOTs can hit a kill switch to destroy the PreyBOTs making it much easier to return the correct target to the holding cell. CARPA would prefer if the

PackBOTs returned the prey alive to a holding cell, but if the PackBOTs choose to destroy the prey and return the correct target, the mission will still be considered a success.

REQUIREMENTS FOR THE LYCANTHROPE PROJECT:

WOLFGANG

- Must not exceed a height of 20 centimeters (cm) while in a four legged state
- Must not exceed a height of 30 cm in a two legged state
- Must not exceed a total weight of 8 kilograms (kg)
- Must automatically transform back onto four legs 3 minutes after transformation
- Must be able to detect color, UV light, and sound frequencies

PACKBOTS

- Must not exceed a height of 20 cm
- Must not exceed a weight of 4.5kg
- Must be able to wirelessly communicate with each other, Wolfgang, and the mission control center
- Must be able to read a passive RFID chip
- Must capture and return proper PreyBOT to a designated area.
- The PackBOTs must be able to dig out the electrically conductive ball from a depth of three inches

UPDATED SCHEDULE

Item	Task	% Complete	Sept	Oct	Nov	Dec	Jan	Feb	Mar	Apr	May	June
1	**CDR 1: Systems Performance Specifications**	100%										
2	**CDR 2: Proof of Concept**	100%										
	Final Computer Aided Designs											
	Subsystems Testing Procedures											
	Prototype- Net											
	Prototype- Ramp											
	Prototype- Wolfgang											
	Prototype- PackBOTs											
	Test Designs											
	Analyze Results											
	Revisions to Designs (1)											
	Test Revised Robots											
	Analyze and Summarize Results											
	POC Rough Draft											
	POC Rough Draft Presentation											
	POC Preparation											
	Final Revision											
3	**CDR 3: Technical Data Package**	25%										
	Finalized Wolfgang Design											
	Finalized PackBOTs Designs											
	Robot Assembly											
	Finalize Part Drawings											
	Finalize Bill of Materials											
	Test Requirements											
	Rough Draft of Data Package											
	Finalized Technical Data Package											

Key	
	Complete
	In Progress
	Not Started

Item	Task	% Complete	Sept	Oct	Nov	Dec	Jan	Feb	Mar	Apr	May	June
4	**CDR 4: Trade Expo**	0%										
	Wolfgang											
	PackBOTs											
	Engineering Notebooks											
	Performance Scenario/ Strategy											
	Packaging											
	User Manual											
	Software Programs											
	Documentary											
	Team Portfolio											
5	**CDR 5: Mission Performance**	**0%**										
	Scenario Completion											
	Sensor Expo											

Key	
	Complete
	In Progress
	Not Started

APPROACH SUMMARY

To complete the S&C mission assigned to us at Eclipse Technologies by the CARPA Initiative, we will incorporate the Lycanthrope Pack. The Pack will consist of three entities: Wolfgang (Alpha), PackBOTs (DOG-1E5, DOG-4S1), and two Trap System(s) (BOP! and MotherGoose). Wolfgang and PackBOTs function similarly by using shared subsystems: Locomotion, Tracking, and Communication. Wolfgang additionally uses the Transformation subsystem. The Trap system(s) consists of the Capture Methods subsystem. The Lycanthrope Pack will asynchronously work together to complete the S&C mission efficiently and effectively.

The Transformation subsystem takes action when Alpha steps into the morphing chamber. Sensors will read values for UV light and audio and allow Alpha to morph into two legs. Three minutes after leaving the chamber, the subsystem will force Alpha to revert back into a four legged state.

The Locomotion subsystem is everything that has to do with the movement of the legs. Each leg of every robot will utilize three servos to allow for three degrees of freedom. These degrees of freedom allow for a variety of movement gaits to be utilized on four legged mode and two legged mode. They also allow for the manipulation of the ball and for digging.

To track the different PreyBOTs on the field, the Tracking subsystem alerts all robots of where the PreyBOTs are on the field. Each robot is equipped with a webcam. The webcam along with the tracking program allows the PackBOTs to follow and locate the different targets.

The Communication subsystem allows the robots to send and receive information to one another. Using a publish and subscribe system, the robots can upload and receive knowledge to and from the Command Center instantly. This allows the robots to coordinate their search with ease.

The Trap subsystem consists of two methods, MotherGoose and BOP. Both will be dragged out onto the field as soon as the gate is opened. The opposing PreyBOTs will wander into the traps and the traps will immobilize them, making for easy capture.

REVISIONS:

TRANSFORMATION SUBSYSTEM

After further calculations, the tail will no longer help in transforming. It would shift the center of gravity away from the ground as it stands up. In order to lift Alpha up, the front legs will push off the ground as the back legs move Alpha to an upright position.

LOCOMOTION SUBSYSTEM

The CARPA Initiative has added the challenge of digging to retrieve the stainless steel ball. In order to dig out the ball, the PackBOTs will have shovel-like attachments on their feet that will allow them to efficiently scoop dirt. The attachments will be able to come off after the digging phase is complete.

To simplify manufacturing, corner gussets will connect the body pieces together instead of using metal frames to connect the Plexiglas sides.

TRACKING SUBSYSTEM

After testing with the vision software, it is determined that only one camera is necessary per robot to determine the field location of the stainless steel ball and the different PreyBOTs.

RISK MANAGEMENT

INTRODUCTION

Risk management is defined as the care and control of all circumstances outside the control of the parties involved that may have a negative or positive effect on project outcome. Although risk management is a continuous process that spans the entire project, the implementation of procedures is especially crucial in this stage of the project, as the outcome of potential risks can have a detrimental effect on team success.

In order to systematically assess each risk, Eclipse Technologies has newly implemented a comprehensive procedure that takes into consideration the various aspects of a risk that may affect its impact. Such considerations include:

- Likelihood
- Severity
- Cause
- Consequence
- Action

After thorough analysis of these aspects, Eclipse Technologies is able to formulate a representative rating of each risk, and subsequently determine the correct course of action for each risk.

SUMMARY OF RISKS

MISSION EXECUTION

The greatest challenge that Eclipse Technologies faces in this project is designing a robot that can reliably walk on four legs. This task comes with many risks in regards to robot performance during the mission, with the main risk being robot balance.

As all three robots are autonomous, communication from each robot to the Command Center and communication between the three robots are crucial for a successful mission execution. Eclipse Technologies assumes that all three robots will be able to communicate effectively with the Command Center, as the actions of the robots rely on human input from the Command Center. Without effective communication, the productivity and usability of the robots would be greatly impaired.

Furthermore, the team assumes that all sensors and cameras will be fully functional throughout the duration of the mission, which includes the successful transfer of data from each device back to the Command Center. This assumption is imperative as failures in sensory input would dramatically increase the level of difficulty in completing the mission. Without knowledge of the specific location or position of the robots, it would be extremely difficult for human control to accurately guide the robots.

MANUFACTURING

A substantial risk that Eclipse Technologies considered is the use of 3D printers in the manufacturing process. Due to the small size for the robot and its constituent parts, Eclipse Technologies decided to rely on 3D printers as the main method of fabrication. There are two risks that accompany the use of 3D printing. First, Eclipse Technologies does not always have reliable access to 3D printers. The team has access to two printers: one that is owned by a company friend, and one that is owned by CARPA. The team does not have continuous access to the one owned by CARPA, and the other printer involves the second risk of inconsistent printing quality, another risk that comes with using 3D printers.

In order to 3D print, as well as virtually test our designs, modeling with computer-aided design software is a crucial step in the development process. In order to have the most effective simulations, Eclipse Technologies utilizes the newest version of Autodesk Inventor. Files from Inventor 2016 may have compatibility issues with files from older versions.

For all the parts that Eclipse Technologies cannot personally 3D print, the team must assume the risk of ordering parts from outside sources. This risk is inevitable, as ordering parts is the most effective way of acquiring parts due to the variety in available parts and affordable costs. However, the convenience of ordering parts comes with the risk of shipping delays, misplaced orders, or defective parts.

Another important step in the manufacturing process is testing. Testing is the only method that will allow the team to know if the potential solutions are effective, and what else the team needs to improve on. Although testing is crucial, the process comes with the risk of damaging parts, as the parts are constantly being used and manipulated.

RISK ASSESSMENT

The first step in analyzing a risk is to generate both a scale rating and a categorical rating of each risk. The scale rating is generated based on the likelihood and severity; each risk is assessed and assigned a number based on the degree of the likelihood of the risk occurring and the potential severity of its effect. The greater the number, the higher the potential risk. Then, the two numbers are added to get the total scale rating.

Quantitative Scale Rating						
		Severity				
		1	**2**	**3**	**4**	**5**
Likelihood	**1**	2	3	4	5	6
	2	3	4	5	6	7
	3	4	5	6	7	8
	4	5	6	7	8	9
	5	6	7	8	9	10

The next step is to derive a categorical rating using the scale rating. This allows Eclipse Technologies to easily classify a risk, and also efficiently communicate the results with the rest of the team.

Qualitative Rating	
Scale Rating	*Categorical Rating*
2-4	Acceptable
5-6	Minimal
7-8	Significant
9-10	Critical

This procedure is then applied to each risk, and is summarized in the following table.

Risk Rating Summary					
	Risk	*Likelihood*	*Severity*	*Scale Rating*	*Categorical Rating*
1	Robot imbalance	3	5	8	Significant
2	Robot communication failure	2	5	7	Significant
3	Sensory input hardware failure	2	5	7	Significant
4	Lack of access to 3D printers	4	2	6	Minimal
5	Inadequate 3D printers	3	3	6	Minimal
6	Autodesk Inventor incompatibility	3	2	5	Minimal
7	Damaging parts while testing	2	4	6	Minimal
8	Shipping delays	1	3	4	Acceptable

RISK OUTCOME

After the scale and categorical ratings are generated, Eclipse Technologies then takes into consideration the remaining aspects besides likelihood and severity. This includes the cause of the risk, the potential consequence, and successive action. These aspects are then identified for each risk, and summarized in the Composite Risk Management Table.

In general, Eclipse Technologies automatically proceeds with risks that receive an acceptable or minimal rating, and refrains from proceeding with risks that receive a critical rating. Risks that receive a rating of significant require case-by-case analysis, though, and the team takes into consideration all aspects of the risk as summarized in the Composite Risk Management Table.

Composite Risk Management Table

	Risk	*Rating*	*Cause*	*Consequence*	*Action*
1	Robot imbalance	Significant	Faulty design, difficulties in execution, rough terrain, physical obstacles	Robot will lose ability to move, and thus cannot continue with the mission	Invent methods to stabilize robot, find recovery methods for robot to regain balance
2	Robot communication failure	Significant	Raspberry Pi failure, wireless connection failure	Team members would be unable to control robot; robot would remain immobile	Test hardware for reliability; choose dependable, high-quality hardware

	Risk	*Rating*	*Cause*	*Consequence*	*Action*
3	Sensory input hardware failure	Significant	Sensor failures, camera failure	Difficulties orienting the robot	Use dependable hardware, test hardware for reliability
4	Lack of access to 3D printers	Minimal	Insufficient number of printers available, unable to use CARPA printer during school holidays	Delays in printing schedule, unable to print parts on time	Reserve excess time to compensate for limited access to printers
5	Inadequate 3D printers	Minimal	Low-quality printers	Difficulties while printing, needing to reprint parts	Print parts with caution to eliminate the need to reprint
6	Autodesk Inventor incompatibility	Minimal	Discrepancies in the version of Inventor on each computer	Unable to open files, spending time to convert files	Establish a uniform version for the entire team
7	Damaging parts while testing	Minimal	Rigorous testing, carelessness, human error	Needing to replace broken parts, inability to continue testing and/or manufacturing	Be very careful when testing, designate a portion of resources to extra parts
8	Shipping delays	Acceptable	Problems with the company, lost orders, backordered parts	Setbacks in manufacturing schedule	Reserve excess time to anticipate delays or other complications

SYSTEM ARCHITECTURE

LOCOMOTION

ALPHA

- 4 legs
- Hind legs are heavier
- 3 Degrees of Freedom and 3 servos per leg
- Z rotation and X rotation the fore legs
- Bigger back feet
- Crawl Gait
- 9-axis IMU
- Descent, Drag, and Return (Power Stroke)
- Recovery Sequence
- Inverse Kinematics
    - FABRIK

PACKBOTS

- Front legs have yaw rotation about x and z planes
- Back legs have yaw and roll about x axis
- Trot Gait
- Crawl Gait
- 1243/1342 Rotary Gait
- Backward Movement Sequence
- Recovery Sequence

MOTHER GOOSE

- 4 wheels
- Hook mechanism
- DFRobot 270 degrees' servo

CAPTURE METHODS

BOP

- Deployment
- Increased rod length
- Neon green untripped side and red tripped side
- Nylon Net with mesh size of 0.75 in. and thickness of 1.32 mm
- Wood Structure = 1/16 in. plywood

MOTHER GOOSE

- Frame out of plexiglass
- Electrical
    - All devices running at 1.8 A
    - 5 V Battery at 2000 mAH
    - 1 Hour Runtime
    - 3 Port Distribution Board
- Ultrasonic Range Finder
- ABS or HDPE Plastic

JAW

- 3 teeth per jaw
- Tooth length of 1 cm
- Jaw opening is 8 cm wide
- Bite is 4 cm deep

TRANSFORMATION

LYKOS MODULE - PROGRAMMING

- Transform Function
    - Violet Sub Module - UV Light Detection
        - UV sensor Breakout
    - Apollo Sub Module - G Note Detection
        - Numpy

ARTICULATION

- **One Inch Diameter electroconductive ball**
    - Multiple options
    - Located beneath loosely packed soil
    - Cup 18 cm off floor
- **Concave paws**
    - Electromagnets
- Raspberry Pi 2 Model B
- Arduino Uno

- Polulu Mini Maestro 18-Channel Servo Controller
- One Leg Ball Transportation

ELECTRICAL

ALPHA

- 7.4V Tenergy Battery
- 18-Port Servo Controller Mini Maestro
- 7.4v DC to 12v DC Stepper
- 7.4v to 5v Voltage Stepper
- 7.4v to 3.3v Voltage stepper
- Total Amperage = 4.82A
- Max Amperage = 6.82A
- Bread Board
- Lowered voltage to 6V
- Slack in wiring
- Extra wire not secured
- 12v DC electromagnet
- Sparkfun UV Sensor
- UltraMax Sonic Sensor
- Odroid C1+
- IMU 9250
- LS 7204 inverter

- Sparkfun Redboard
- 2 Macro to USB cords
- Mirco USB to power

PACKBOTS

- Total Amperage = 4.40A
- Max Amperage = 6.40A
- 7.4V Lithium Polymer Battery with 6600mAH of run time
- 62 minute robot run time
- Bread Board
- MIni Maestro 18 port servo controller
- 6V Voltage Stepper
- LS 7204 inverter
- IMU 9250
- UltraMax Sonic Sensor
- Odroid C1+
- 2 Macro to usb Cords
- 1 Micro to power cord
- Protective Sheath over wires
- Slack in wiring
- Extra wire not secured

MATERIALS

ALPHA

- Frame made from Plexiglass sheets
- 0.098 in. sheets with holes
- Back legs made of .16 in. plexiglass
- Front legs made of .1 in. plexiglass

PACKBOTS

- Overall Frame made from Plexiglass sheet
- Sheets of ABS plastic

TRACKING

PREYBOTS

- 2 HD Cameras
    - Mounted Parallel
    - 2 Degrees of Freedom
    - Open CV
    - HSV color space
    - Disparity Mapping
- Ares Module - Trap Control and PackBOT coordination
    - Path Conflict Fixing
- Theia Module - Vision
    - Quick Field Mapping and Position System
    - Kalman Filter

- Camshift

ALPHA

- 2 HD Cameras
  - Mounted Parallel
  - 2 Degrees of Freedom
  - Open CV
  - HSV color space
  - Disparity Mapping
- Theia Module
  - Color/image detection of ball location
  - ORB detection algorithm
  - Perspective Transform
    - shape detection
  - Bruteforce Hamming Matcher
  - RANSAC homography

COMMUNICATION

CHATTER MODULE

- Wi-Fi
- WAMP protocol
- Router at 802.11n
- Crossbar.io

- Remote Procedure Call Pattern
- Asynchronous Design
    - Python 3.4's Asyncio Module

ZEUS MODULE - MISSION CONTROL

- Command Display
- Redis
- PostgreSQL
- Open CV
- Chrome
- PyPy
- Python 3
- Point Cloud Library
- Ubuntu 14.04 LTS

TESTING

COMMUNICATION SYSTEM

RED ECLIPSE FRAMEWORK

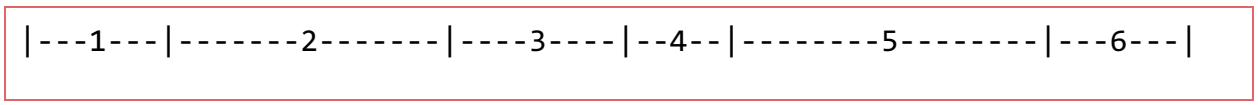
In a massive and complex framework such as the one Eclipse Technologies has designed, it is difficult to rely on serial programming - waiting for one operation to finish before passing it on to another. In the mission field, communication with the outside world is perhaps more important than the operation itself, even if the robots run fully autonomously. To combat this problem, Eclipse Technologies' framework uses the new *asyncio* Python module and an event-driven framework to ensure success on mission day.

The layout of the framework is quite simple. A computer running Crossbar.io in PyPy on Ubuntu 15.10 (Wily Werewolf) acts as the relay center. PyPy is a JIT distribution of Python which compiles many functions to machine code during runtime, drastically improving speed. All robots and computers connect to a router. The computer running Crossbar.io has a static IP. During startup, all devices will connect to the Crossbar.io running on the server. Once communication is established, the devices can then exchange their IPs (which may be dynamic) and begin communication.

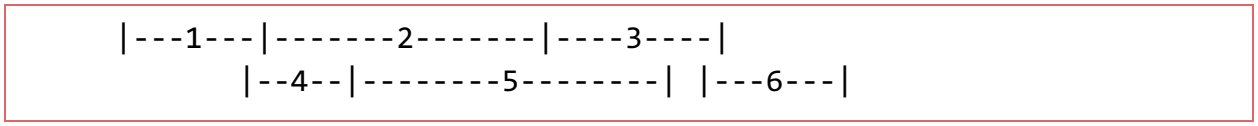
ASYNCHRONOUS VS. SERIAL

The Red Eclipse Framework, which powers all of Eclipse Technologies' robots, utilize *asyncio* for communication. Asynchronous design allows handling of time-consuming and often blocking I/O operations without interrupting other concurrent CPU-intensive operations. Asynchronous operations are often quite puzzling, as they are able to handle blocking operations with only one thread. A simple way to understand the difference between synchronous and asynchronous operations is to use an everyday example - getting ready in the mornings.

Suppose an arbitrary individual, Monica, wakes up late on a particular morning. She has to drink her daily dose of coffee and eat toast. To get coffee, she has to perform three operations: (1) prepare the coffee machine, (2) wait for the coffee machine, and (3) drink the coffee. To get toast, Monica must (4) put bread in the toaster, (5) wait for the toaster, and (6) eat the toast. A serial or synchronous framework would do the following.



Monica soon realizes that this serial method of preparing her morning meal always made her late. Thus, she resolves to try the asynchronous method of preparing her morning meal.



In the asynchronous and event-driven design, Monica takes advantage of the fact that operations 2 and 5 have nothing to do with her. After preparing coffee and while waiting for the coffee machine, Monica goes to put the bread in the toaster (4). The coffee finishes before the toast, so Monica drinks the coffee (3). While she is drinking coffee, the toast finishes. However, since she is already drinking coffee, Monica finishes her coffee before moving on to consume the toast (6). Note that while this is not exactly twice as fast, Monica saves a lot of time because she can now do other things while she waits for operations that don't involve her, such as brewing the coffee and toasting the bread, to finish.

Monica is analogous to a thread in the *asyncio* event loop. Network operations and proxy operations, which consume massive amounts of time but do not involve any CPU time from the thread, can be done using futures. A future is an initially unknown value that can be fulfilled at a later time. In the Monica example, the coffee machine will beep and the toaster will pop out the toast to indicate to Monica that a future is complete. Similarly, the operating system will signal to the thread of the completion of an I/O operation so that the thread can return to process the future.

IMPLEMENTATION

The box below indicates a small portion of the code that will run in the event-loop on all robots. In real-life situations, not all operations support futures. For example, initializing the servo controller and computing some constants require some CPU, but also has a lot of USB I/O time. For operations that don't fully support futures, a thread is used.

```
def __init__(self, *args, **kwargs):
  # Create a thread executor for slightly CPU-bound async functions.
  self.executor = ThreadPoolExecutor(4)
  self.robot = None
[snip]
async def initialize(self):
  loop = asyncio.get_event_loop()
  try:
      self.robot = await loop.run_in_executor(self.executor, async_initialize)
      self.call('zeus.info', 'Successfully initialized!')
  except:
      self.call('zeus.info', 'Failed to initialized!')
```

The purpose of branching off a thread is to ensure the functionality of the event loop. If the main event loop is blocked by an operation that is either CPU-intensive and doesn't support thread switching or an I/O operation that doesn't support futures, the entire communication system could be at risk. In simpler terms, if Monica got an important phone call but kept on drinking coffee, she would miss the call. For a robot, missing the ping packet (call) can cause Crossbar.io to assume that connection has been lost and falsely send certain modules into panic.

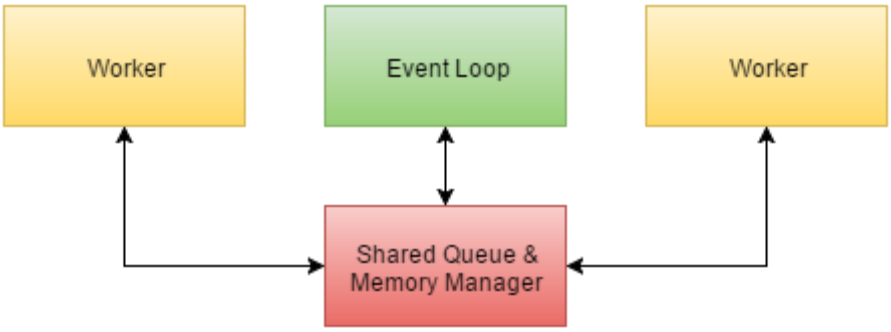
MODULE COMMUNICATION

Due to the importance of the event loop, it is nearly impossible to combine all modules into one process. CPU-intensive processes like vision processing or IMU data fusing would inevitably slow down the event loop and processing drastically. Multithreading is also highly problematic in Python due to the Global Interpreter Lock (GIL). However, Eclipse Technologies has developed a novel method of proxy communication that allows asynchronous worker to worker and worker to event loop communication.

MEMORY EFFICIENT MULTIPROCESSING

The low memory capacity of the ODROID precludes the possibility of *multiprocessing*, a Python module designed for spawning multiple processors. However, this spawning is a fork operation, meaning that the child process duplicates the memory of the parent. This is extremely inefficient, as most of Eclipse Technologies' modules are independent of one another. In addition, forking will quickly overfill memory. To be safe, only 500 MB of memory should be used. Loading all of the modules consumes at most 100 MB, plus an additional 50 MB for processing space. Forking would bring the memory very close to max. However, there are many module repeats in child processes that are not needed.

To combat this problem, Eclipse Technologies uses a memory manager. The manager is a separate Python process whose sole purpose is to pass data between processes. It has a very low memory consumption of ~5 MB during startup and is a good tradeoff between latency and memory efficiency.



The manager works by proxy. When a worker or the event loop wants to modify a variable, it does so by pickling the object and sending it to the manager, which will then reflect the change on future request by other processes. This operation is very fast. Over 20,000 operations can happen per second if when the manager shares data using pipes on Windows and unix sockets on Linux.

The main method of communication between the event loop and its worker is by means of a proxy FIFO queue. When the event loop receives an instruction from the command center, it puts a command into the queue. The worker processes will consume the command. If the command requires some form of return data, the event loop can asynchronously wait for the worker to return data via a different FIFO queue.

OVERCOMING GIL

GIL presents a significant problem in Python. Unlike other languages, Python is unable to execute multiple bytecodes in parallel in different threads running in the same process. This issue means that spawning multiple threads for CPU-intensive tasks does not speed up processing at all as true multithreading is not achieved due to GIL.

To combat this problem, Eclipse Technologies uses a RAII (Resource Acquisition Is Initialization). In all time consuming functions that do not involve any Python variables, such as a tracking function shown below, GIL is released from C++ and automatically reacquired before boost.python performs final variable conversions from C++ to Python.

```
class releaseGIL {
public:
 inline releaseGIL() {
    save_state = PyEval_SaveThread();
 }

 inline ~releaseGIL() {
    PyEval_RestoreThread(save_state);
 }
private:
 PyThreadState *save_state;
};

[snip]

bool reinit(const cv::Mat& image, Tuple boundingBox)
{
 releaseGIL unlock;

 Rect bb = Rect(
    extract<T>(boundingBox[0]),
    extract<T>(boundingBox[1]),
    extract<T>(boundingBox[2]),
    extract<T>(boundingBox[3])
    );

 return reinit_(image, bb);
}
```

The release of GIL allows other threads to execute. Long-term and short-time trackers can work in conjunction walking algorithms using only one worker process. This allows the framework to be extremely efficient at exploiting multicore processors like the one in the ODROID. Tests show that a non-GIL release process can exploit only 31% of the entire ARM processor while the GIL release version can exploit up to 75%.

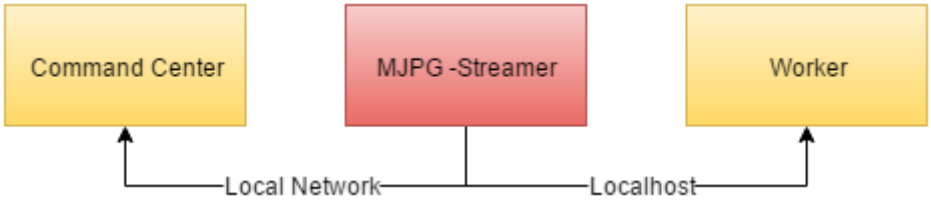
WORKER MANAGEMENT

Workers are self-contained processes with an arbitrary number of threads. The threads have synchronization primitives like locks so that certain parts of the code can be sequenced properly. However, the processing threads never directly communicate with the shared proxy manager. A separate process is spawned when communication is required as communicating with the manager is an I/O operation which can block CPU-intensive tasks like vision processing even thought it itself takes little to no CPU.

Worker processes also has the ability to start and stop thread to save CPU. In the case of transformation for example, vision threads and video streaming would be paused temporarily (not destroyed) so that Apollo can access the microphone on the camera. After transformation is complete, the worker will destroy the entire Apollo and Violet modules to save memory and CPU. This management is fully automatic and greatly improves performance while keeping memory usage in check.

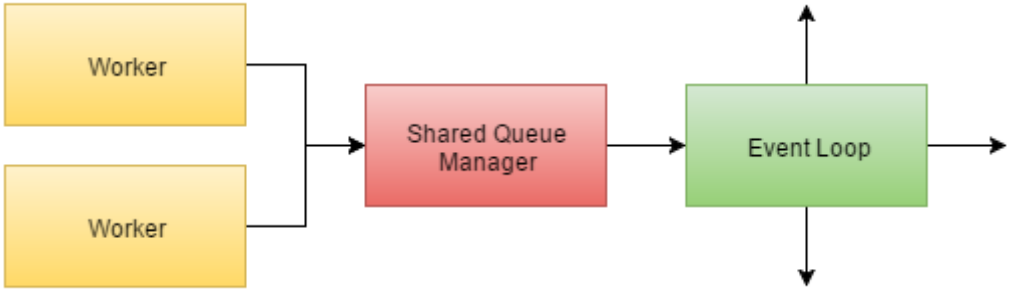
LARGE BINARY DATA AND LOGGING

Crossbar.io utilizes JSON to pass data while the proxy memory manager uses serialization by pickling. Neither of those methods are suited for large binary data transfer such as the passing of image data. To allow for rapid image acquisition for both the worker processes, neither the workers nor Zeus has direct control of the camera. Instead, a streamer streams the camera on localhost so that both all systems connected to the local network can access the data without consuming any CPU from parsing an image that already exists.



An additional benefit is that most USB webcams are UVC compatible and support MJPG encoding. This means that the streamer does not even need to parse the image. It only has to host it and broadcast it on localhost. The stream can reach 30 FPS and only use 0.5% of a single processor on the ODROID. Access rate from localhost exceed 80 FPS while access rate from the Command Center can reach 30 FPS is a good router is used. Synchronization is only necessary in certain critical sections in worker processes. This is handled automatically so the streamer does not need to worry about keeping a history of frames.

Logging is handled using FIFO queues. Traditional Python logging is incapable of handling multiple processes. However, the proxy manager allows logging from all modules to be asynchronously passed to a queue. The event loop will send them asynchronously by using a separate thread to handle the log parsing.



CRASH RECOVERY AND MANUAL CONTROL

While it is highly unlikely that any of the processes will crash, such an event can definitely occur. Eclipse Technologies has created preventative measures to ensure that if any processes crash, the robot will be able to quickly recover autonomously.

The event loop is started at system boot up. It creates a PID file in its own folder. Once it connects to the server, it will initialize its worker processes when the command is given. All of the workers are spawned and detached, so that even if the event loop crashes, the workers can continue to operate.

```
def spawn_python(file):
    """
    Spawn a detached Python process using the same interpreter.
    :param file: The Python file name relative to this file's path.
    :return: A Popen object representing the opened process.
    """

    return Popen([sys.executable, file])
```

Workers spawned with the above command automatically expose their PID. The event loop collects their PIDs and writes them to a separate file which contains the PIDs of all workers and the manager. Every few seconds, the event loop will asynchronously check for the status of its workers. If it detects that any workers have crashed, it will respawn them.

If the event loop itself crashes, the operating system will handle the recovery. A cron job runs once every 30 seconds to check if the process in the master PID file is alive. If the process is not, it will respawn the event loop. When the event loop restarts, it modifies its own PID file to reflect the new process. It will then check the file of its workers to ensure that all are still alive. Again, it is able to spawn any workers or the manager if they are dead.

For a majority of these operations, the Python library *psutils* is used. The library allows platform independent checks for PID as well as data for memory, disk, and CPU usage. This allows the event loop to efficiently report this information to the command center and automatically balance load during low-resource conditions.

In the extremely rare case that something messes up in the entire framework, manual SSH access can be obtained. The command user is able to directly modify any part of the robot's operating system while on the mission field.

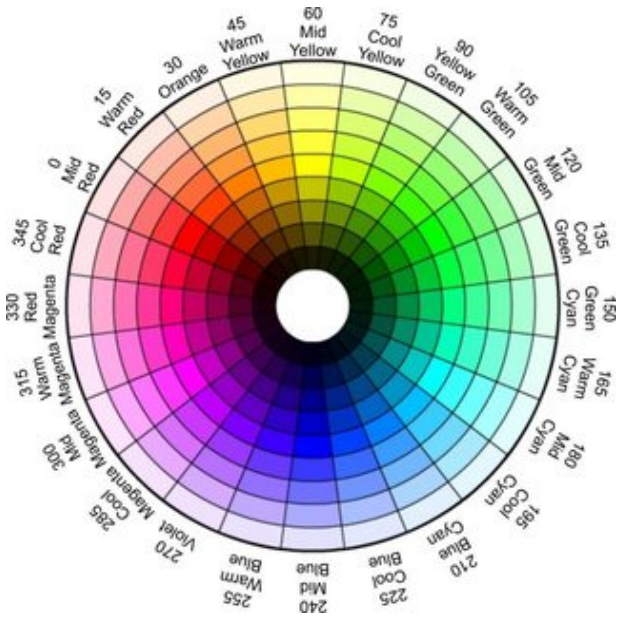
All robots are also configured to reboot automatically almost immediately after a kernel panic (BSOD or system crash). The OS also throttles the CPU automatically when it hits 95 degrees Celsius and will automatically reboot system when the CPU temperature reaches 100 degrees Celsius.

VISION SYSTEM

IDENTIFYING COLOR

In the beginning of the mission, the PackBOTs are tasked with identifying a colored sand pile and digging out the ball from underneath the sand. There are three piles to choose from -- purple, orange, and green. The desired color will be input at the command center and transmitted to the PackBOTs.

Color identification, which is part of the Theia module, relies primarily on the HSV color space. Unlike RGB, HSV is more uniform and more closely models the human perception of color. The hue component of HSV is given as an angle measurement. Each bin of hue represents a human-determined color. Saturation and value are often affected by lighting conditions. However, only hue needs to be analyzed to determine the color. The color spectrum can be seen as a wheel when only incorporating hue and saturation.[1]



From the diagram, the hue values for purple (warm blue - magenta) has a range of 240 to 285. The hue values for orange and green are 10 to 50 and 80 to 150 respectively. These constants have been arbitrarily determined based on human perception. Eclipse technologies has developed four different color identification solutions which are detailed below.

[1] "Color Picker and Converter (RGB HSL HSB/HSV CMYK HEX LAB)." Colorizer. N.p., n.d. Web. 28 Dec. 2015.

PROPOSED METHODS

METHOD 1: MEAN HUE + HUE INFERENCE

It is possible to take the mean hue of all pixels in an image and map the average hue to a color. This method is computationally efficient, but does not take into account the saturation and value. This prevents hue inference from detecting colors like gray, which has no specific hue values. Another issue is that taking the mean hue doesn't really guarantee accuracy. If an image has two dominant colors that were distant from each other, the mean algorithm would return a hue value that is in the middle, which matches neither of the dominant colors.

A given 8-bit input image is converted to the HSV color space. After the mean 8-bit hue is taken, it is converted to a 32-bit hue and mapped to a color.

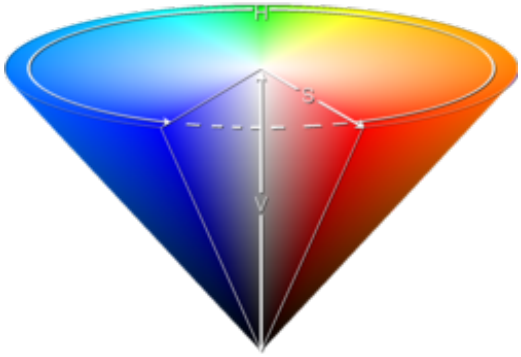
METHOD 2: K-MEANS RGB CLUSTERING + HUE INFERENCE

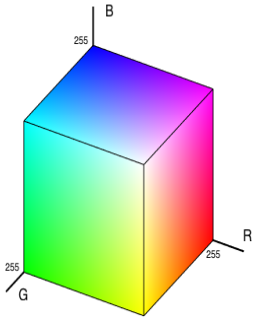
K-means is a clustering algorithm designed to mitigate the issue of multiple dominant colors in an image. Whereas mean HSV's accuracy degrades significantly with more than one dominant color, the k-means algorithm is capable of finding k dominant colors in an image at the cost of performance.[2]

The pixels in an image can be represented as an (x, y, z) coordinate pair in Euclidean space, whether it be RGB or HSV. However, as shown in the HSV color space below on the left, the space is not very uniform nor suitable for clustering. Consider hue, which is given as an angle. On a circle, 359 degrees is equally close to 0 degrees as 1 degree would be to 0. K-means

[2] "K-Means Clustering in OpenCV." OpenCV. N.p., 10 Nov. 2014. Web. 28 Dec. 2015.

does not compensate for the circular nature of the data set. Thus, the RGB color space, shown on the bottom right, is more suitable as it more accurately represents a 3D Euclidean space.





Clustering computation is fairly simple. First, the constant k is set to determine the number of centroids the k-means algorithm should find. 3 is a good constant for accuracy and performance tradeoff. Next, the 8-bit RGB centroids are converted to 32-bit HSV. Finally, the hue of each HSV centroid is used to determine the color. The proposed method also includes a percentage system, where each dominant color is attached to a percent indicating the percentage of points associated with that centroid. For most applications, the greatest percentage is chosen as the true color.

METHOD 3: MEAN LAB + CIELAB DELTA E INFERENCE

Hue inference is a very relative measure of color, as it cannot distinguish between colors in the same hue, such as dark blue, light blue, black, and silver blue. Eclipse Technologies also proposes a different method of color detection known as the Delta E inference. Delta E inference relies on the CIELAB color space, a color space that is designed to model the human perception of color variations.

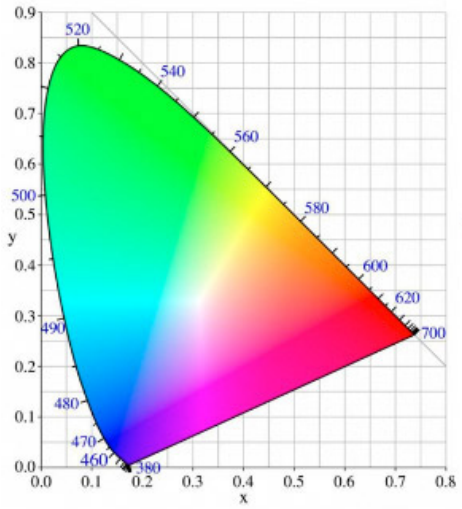
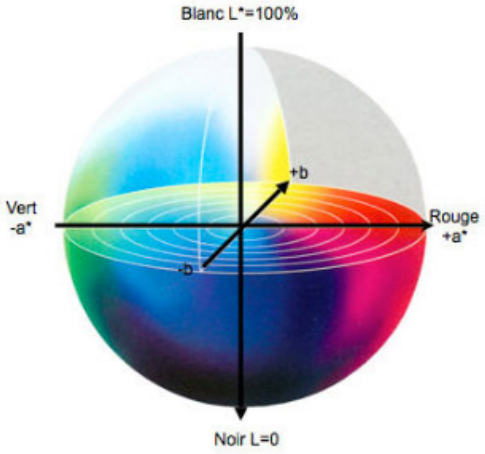


Diagram of chromaticity



Lab color space

As shown in the diagrams above, CIELAB is much more uniform that HSV and RGB, allowing a relatively simple measure of color difference[3]. The CIE defines in the CIE76 that the algorithm used to compare two points of (L*, a*, b*) is simply the Euclidean distance[4]. While this does not create a perfectly uniform computation, its efficiency and accuracy for use in this field of robotics is more than sufficient.

A predefined number of RGB colors is preset and converted to CIELAB. The input 8-bit image converted to a 32-bit RGB image, which is then converted to 32-bit LAB. The average LAB is taken. The result is compared to every predefined colors using CIE76. The lowest value of the array of outputs is taken as the closest “true” color.

[3] "CIELAB - Color Models - Technical Guides." CIELAB - Color Models - Technical Guides. Adobe, n.d. Web. 28 Dec. 2015.

[4] "Color Difference." *Wikipedia*. Wikimedia Foundation, n.d. Web. 28 Dec. 2015.

METHOD 4: K-MEANS RGB CLUSTERING + CIELAB DELTA E INFERENCE

Method 4 utilizes the k-means clustering algorithm defined in method 2 and passes the output to the CIELAB Delta E inference algorithm defined in method 3. However, unlike method 2, the output of 8-bit RGB centroids are converted to 32-bit LAB instead of 32-bit HSV. This is theoretically the most computationally expensive method of the four.

TESTING METHODOLOGY

For the purpose of testing, the methods have been named M1 to M4 respectively. Testing machine and software specifications are listed below.

Each method is defined as a single function. Tests only account for mission-related scenarios. All methods will be passed the same image. The color that should be detected will always be the most dominant of the three (orange, green, and purple). However, it may not be the globally dominant color. This emulates a simple ROI capture by the DOGs on mission date. All k-means containing algorithms use k=3, which means that the color of the most percentage will be used. The procedure is detailed below.

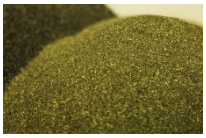
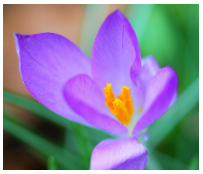
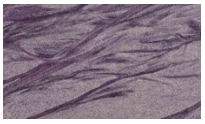
1. A human identifies the color in an image. This is considered the touchstone color.
2. The image is loaded into the memory.
3. Timing begins.
4. The function is passed the image as an argument.
5. The function returns a color.
6. Timing stops.
7. The time taken and the returned color are recorded.

8. Repeat steps 1-7 for all method functions.

All LAB-based algorithms are given the colors below as a means of identifying the closest match. They have standard RGB values as defined by CSS3. These RGB values are pre-converted into LAB values before execution.

Purple, RGB (128, 0, 128)	Orange, RGB (255, 165, 0)	Green, RGB (0, 128, 0)

DATASET

No.	Image	Resolution	Color	Comments
1		200 x 200	Orange	Simple solid color. Light gradient change across image.
2		385 x 257	Green	Green sand with slight background color variations.
3		500 x 374	Orange	Difficult full scenery image. Predominantly orange with blue sky and dark brown rocks.
4		540 x 565	Purple	Solid color with slight lighting differences. Contains a white background.
5		325 x 244	Purple	Difficult image with various colors. Purple is only marginally dominant. Green also exists as a secondary dominant color.
6		500 x 424	Purple	Difficult image with all three colors. Purple is only marginally dominant. Green and orange are also present are relatively dominant.
7		796 x 448	Purple	A realistic example where the purple color exists, but is not dominant.
8		525 x 394	Green	Another realistic example where the green is not very prominent. A lot of background interference exists.

RESULTS

Green indicates a successful detection and red indicates a failed detection. The number indicates the time taken (in milliseconds) by the algorithm over an average of 10 runs. The best algorithm is the one which scores the highest in terms of accuracy. In the case of a tie between two algorithms, the winner is chosen based on speed.

Image	M1	M2	M3	M4
1	3.002	71.45	3.402	74.85
2	3.603	161.8	5.154	168.0
3	4.103	198.3	7.306	182.5
4	4.603	480.3	11.41	481.4
5	3.702	147.0	7.005	131.9
6	3.903	312.6	9.607	312.7
7	4.802	569.0	13.51	592.7
8	4.303	307.0	9.207	291.6

It is evident from testing that methods 3 and 4 are superior to methods 1 and 2 in terms of accuracy. However, what is surprising is that method 3 significantly outperforms method 4 in terms of speed without sacrificing accuracy. The reason that CIELAB Delta E outperforms hue inference is that it is a relative algorithm, which is superior to the absolute hue inference algorithm for this mission.

CONCLUSION

Given the testing and mission parameters, method 3 -- Mean LAB + CIELAB Delta E Inference -- is the best algorithm for sand color detection and will be implemented. Method 4 is more suitable for detection of multiple colors.

DETECTING PREYBOTS

DOG-1E5 and DOG-4S1's main task will be to locate and capture the PreyBOTs. No features or color descriptions of the desired targets will be given prior to mission day. The DOGs must be able to first identify the PreyBOTs before they can be tracked.

One property of PreyBOTs that makes them easier to detect is that they move around in random patterns. This assumption makes it possible to distinguish the PreyBOTs from their background. However, the DOGs are not given a background to subtract out prior to the mission. On mission day, it is possible that no background devoid of PreyBOTs will be given for observation. Thus, Eclipse Technologies relies on background/foreground segmentation algorithms provided by OpenCV.

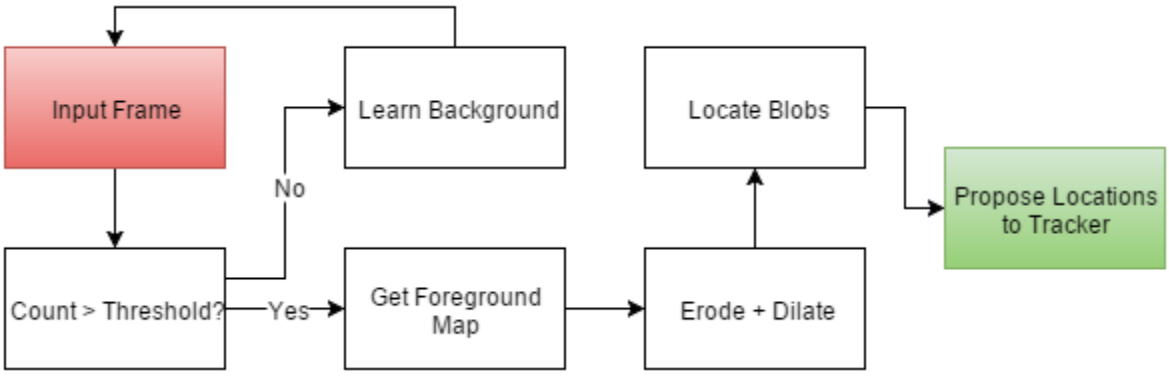
Before entering the field, the PackBOTs will remain stationary and observe the field for a few seconds. Each frame collected during that time is passed to the background/foreground segmentation algorithm. After observation is completed, the next frame will use the output background mask to locate the PreyBOTs. The frame is preprocessed with a Gaussian blur and post-processed with erosion and dilation. The resultant blobs are bounded and used as locations for tracking analysis.

PROPOSED METHOD

Two algorithms are used for segmentation -- Mixture of Gaussians v1 (MOG) and Gaussian Average (GA). Both were selected based on their performance and accuracy.[5] They

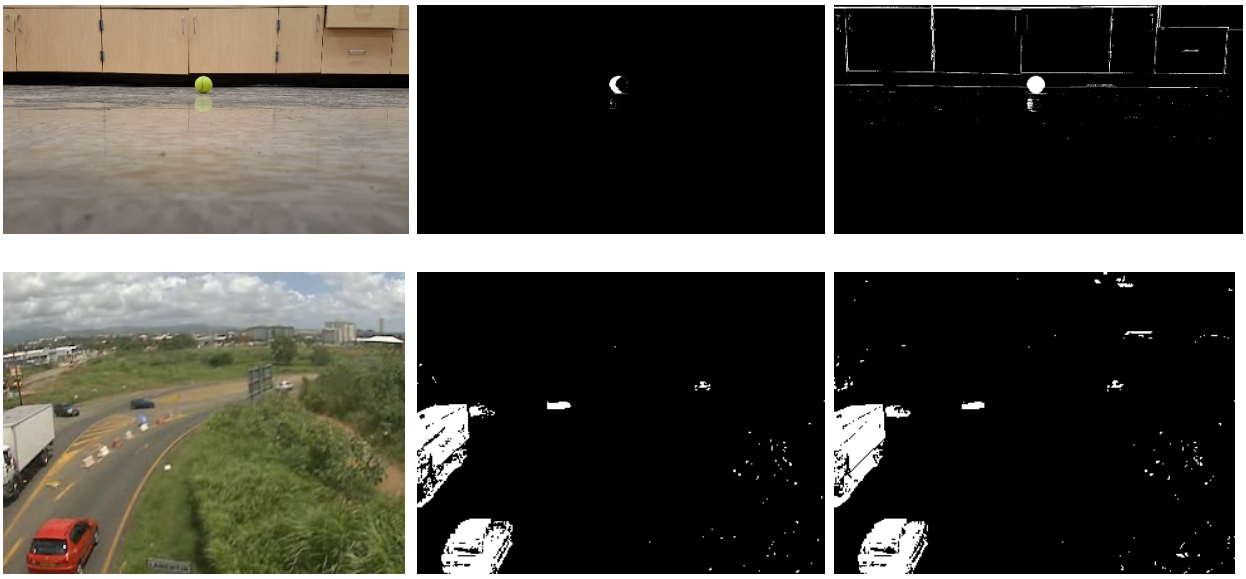
[5] Vacavant, Antoine, and Andrews Sobral. Research Gate. Computer Vision and Image Understanding, n.d. Web. 23 Dec. 2015.

are the only foreground/background segmentation algorithms that can theoretically perform in real-time on the ODROID-C1+. Both algorithms will be given a certain amount of frames to learn the background. After which, the next frame will be used to generate a binary foreground mask. The mask is post-processed with erosions and dilations. Finally, blob detection is used to locate regions of potential trackable targets.



TESTING METHODOLOGY

Multiple videos will be sent to the algorithm chain for processing. The FPS will also be measured to ensure that performance is feasible. All tests are performed on the Windows configuration.



The first row is a tennis image by Eclipse Technologies. Its resolution is 1920 x 1080. The first column is the image frame, the second column is the foreground mask by MOG, and the final row is the foreground mask by GA. MOG in this sequence ran at 16 FPS while GA ran at 12 FPS. The specific frame was chosen because the ball was moving slowly. As shown, MOG begins failing while GA successfully is able to detect the ball.

The second row is a traffic sequence found on the Internet. The difference between this one and the tennis sequence is that the first frame of the traffic sequence contained cars. As shown, both MOG and GA perform very well. MOG ran at over 500 FPS on the 320 x 240 sequence while GA ran at over 350 FPS.

CONCLUSION

Due to the fact that the velocities of the PreyBOTs will be unknown, it will be much safer to use GA, as MOG begins failing when objects move slowly. However, using MOG should not create a significant difference, as the blob bounding done after post-processing are almost identical for GA and MOG results.

TRACKING PREYBOTS

Tracking is very complex, considering that no prior details of the PreyBOTs are given. Eclipse Technologies proposes a novel tracking method that outperforms nearly all current state-of-the art trackers in terms of accuracy, robustness, and performance.

CONSTRAINTS

What makes tracking so difficult in this scenario is the limited hardware and PreyBOT information. An ODROID-C1+ has a 1.5 Ghz quad-core ARMv7 processor. The 1 GB memory is limited to 792 Mhz. This is much slower than the 3.0+ Ghz Intel processors with multiple GBs of ram at 1300+ Mhz. In addition, Intel processors come with hyperthreading and 8 logical processors, which allow them to be much more performant than the embedded devices Eclipse Technologies is using. The camera is also another constraint. Given the capacity of the device, the maximum capture rate is 30 FPS and the resolution must be limited to 640x480 for performance. Grains and camera lighting adjustments can significantly reduce tracking capacities.

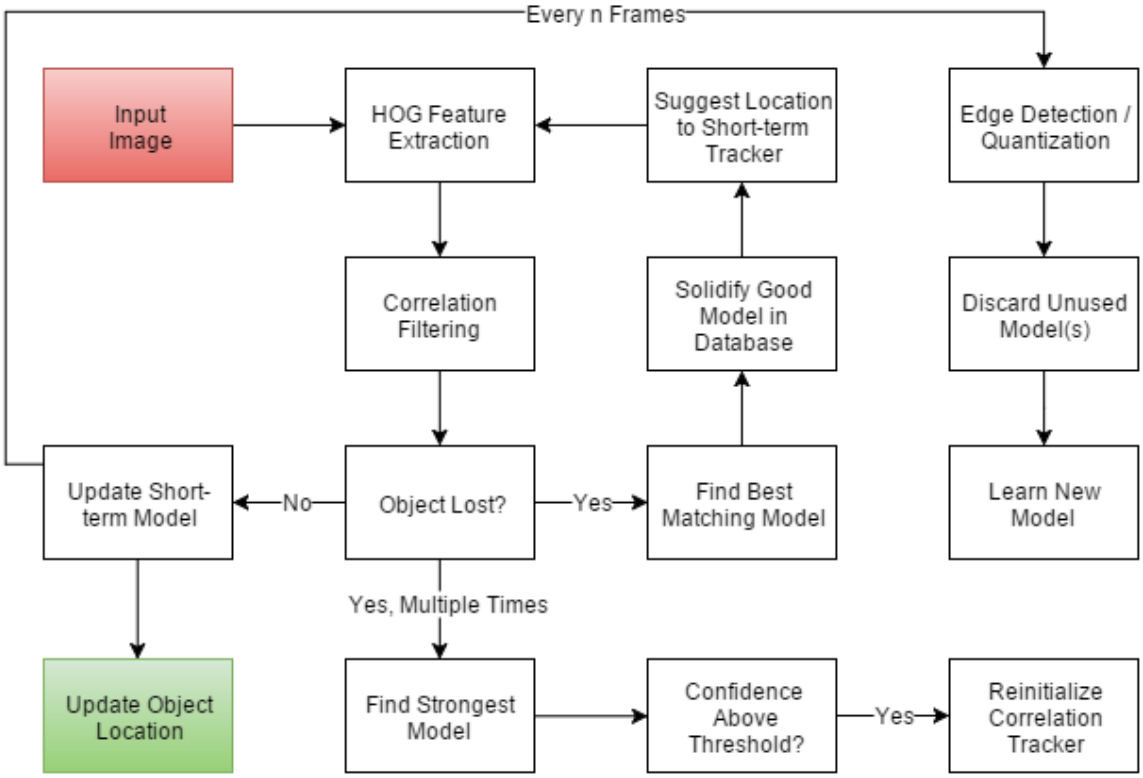
The mission scenario presents one of the most difficult tracking scenarios ever to exist. Once an ROI is identified, the algorithm must be able to track the object from then on. PreyBOTs are presumed to have fast motion, and the PackBOTs are in motion, making the camera shake. It is also highly likely that blur and occlusion can occur. PreyBOTs can escape the FOV of the DOGs, making tracker recovery a necessity. Finally, the PreyBOTs are constantly changing in appearance, as they can turn or flip. They may even have drastic rotation and size changes during the mission.

A proposed tracker must be able to account for all of the below and operate at real time (10+ FPS) using at most 2 processors of the ODROID-C1+.

A) Tracking from single ROI.	**B)** Low resolution.
C) Lighting adjustments.	**D)** Grainy and blurry images.
E) Rotation and scale invariance.	**F)** Real time.
G) Different appearances of entity.	**I)** Out of FOV recovery.
J) Fast motion.	**K)** Minimal trackable features.

PROPOSED ALGORITHM

Eclipse Technologies proposes a combination tracker based on the human memory and response to external stimuli. The system incorporates a novel and sharable learning system.



The algorithm consists of two main parts -- a short-term tracker using HOG features and a long-term tracker using machine learning, edge quantization, and model database. The trackers are described in more detail below.

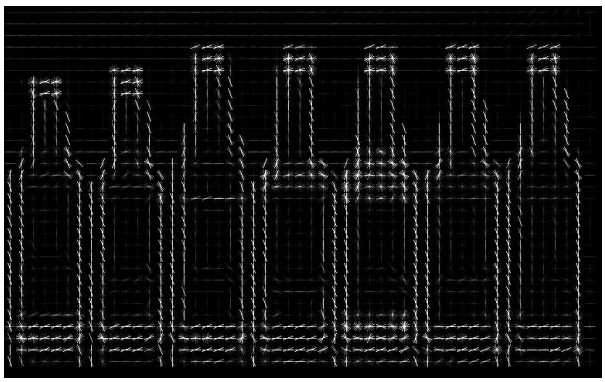
SHORT-TERM TRACKING: DSST ON HOG

A short-term must be extremely good at tracking an object for a short period of time. Research led to two possible candidates -- DSST and KCF. According to the VOT 2014 results, DSST is the best tracker and is superior to KCF in terms of rotation and scaling.[6] However, it was shown in the report to be much slower (3x) than KCF. However, it still produced more than sufficient real-time results of 30+ frames per second in its Matlab implementation.[7]

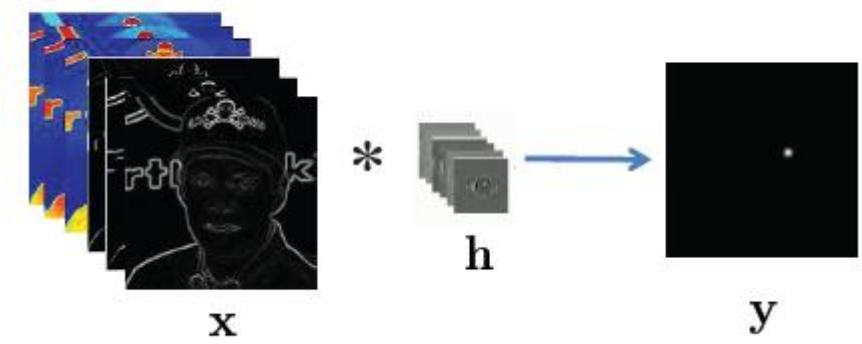
The accuracy of the tracker comes from HOG. The HOG descriptor takes an image and separates it into cells. It computes the gradient magnitude and direction for each cell. These descriptors are then used for matching. The image of bottles on the right is the HOG descriptor of the image on the left. The fact that HOG uses color gradients satisfies constraints C, D, J, and K. The segmentation into boxes satisfies constraint B as not every pixel is counted individually.

[6] "The Visual Object Tracking VOT2014 Challenge Results." (n.d.): n. pag. EPICS. University of Paderborn, Germany. Web. 20 Dec. 2015.

[7] Danelljan, Martin, Gustav Häger, Fahad Shahbaz Khan, and Michael Felsberg. "Integrated Object Models for Robust Visual Tracking." Robust Vision for Vision-Based Control of Motion (2009): n. pag. Li.u. Linköpings Universitet. Web. 22 Dec. 2015.



DSST was designed with the tracking of a single ROI in mind. This automatically satisfies constraint A. The tracker itself keeps a well-trained model of the object HOG descriptor. Over time, this model is updated based on new object appearances, satisfying constraint G for short-term tracking.



The principle of tracking is relatively simple. A certain input image along with a trained filter is multiplied to get a response matrix. The brightest location in the response matrix

indicates the position of the object.[8] DSST utilizes additional filtering techniques to adapt the size of the window, thereby satisfying constraint E.

The tracker must also be able to detect when it has lost an object. The original implementation by the authors of DSST does not address this issue. Eclipse Technologies uses the Peak-to-Sidelobe Ratio (PSR) in DSST's predecessor MOSSE to evaluate the tracking quality.[9] If the computed PSR value is less than the threshold, it indicates that the object has been lost. This satisfies part of constraint I. As with all other aspects of the tracker, the PSR confidence value gets more stable (higher) over time as the tracker learns the model.

LONG-TERM TRACKING: EDGE QUANTIZATION AND TEMPLATE MATCHING

Eclipse Technologies' tracker also has the ability to learn multiple different models or templates of the tracked object over time. This is particularly useful for recovery and rapid changes to the object. Every 30 or so frames, the long-term tracker initializes in a separate process or thread. If DSST reports that it has a good track of the object, the long-term tracker will learn a new template for the object using the reported ROI.

[8] Danelljan, Martin, Gustav Häger, Fahad Shahbaz Khan, and Michael Felsberg. "Integrated Object Models for Robust Visual Tracking." Robust Vision for Vision-Based Control of Motion (2009): n. pag. Li.u. Linköpings Universitet. Web. 22 Dec. 2015.

[9] Bolme, Dav, J. Ross Beveridge, Bruce A. Draper, and Yui Man Lui. "Visual Object Tracking Using Adaptive Correlation Filters." 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (2010): n. pag. Colorado State University. Computer Science Department. Web. 20 Dec. 2015.

The image above shows 20 templates determined by DSST to be good ROIs. Matching is done by iterating through the templates and computing response maps for an image. Template matching is done using OpenCV's TM_CCOEFF_NORMED algorithm, which will return a response map with floats from 0 to 1. The brightest point in the response map is the most likely location of the template.

Edge detection is done using the canny edge detector, which also relies on gradients. Thus, it satisfies constraints B, C, D, and J. The number of different templates satisfies constraint E and G.

RECOVERY

When the short-term DSST tracker reports that it has lost the object, it will no longer update its HOG descriptor model. The tracker has a built in padding value, which allows it to converge on the object if it comes within a certain search radius of the last known location. However, this cannot be assumed. When DSST fails, the long-term tracker is activated, finding the best template and the best location for that template.

The long-term tracker will suggest the location to the short-term tracker. However, it is up to DSST whether or not to reject that suggestion. If it is accepted, tracking continues as normal. If it is rejected, DSST tries to converge on the object's last known position and the long-term tracker is activated again on the next track.

If the short-term tracker rejects too many proposals of the long-term tracker, which can happen if it has been learning the wrong object for the past few frames with a high learning rate, the long-term tracker can reinitialize DSST. After template matching completes and the response point has a high confidence value, the long-term tracker will reinitialize DSST at the new location.

LEARNING AND DISTRIBUTION

The single-model learning of DSST is not modified externally in any way by the system. The long-term multi-template system can be impacted directly. The long-term tracker database is modeled like the human memory system. It has a certain capacity, which in the previous case was 20 templates. When a template is matched or when it assists with a recovery, it is solidified or promoted in the memory. Over time, unused models are removed to make space for newer models that can assist in tracker.

The exception to this system is the first template, which comes from the foreground/background segmentation algorithm. This ground truth template is always kept using a high initial bias. By keeping this template, the robot can always ensure that it has at least one definite description of the tracked object. Recovery satisfies constraints G and I completely.

The templates are extremely small, averaging 500 bytes in uncompressed format. Good templates are sent to the server to be exchanged with other robots. This is particularly useful in the beginning of the mission, as one robot can learn the appearance of the PreyBOTs and

quickly teach other robots how to identify them without waiting for a full detection and track using foreground/background segmentation.

OPTIMIZATION

Constraint F is solved through heavy optimization. The slowest part of the algorithm rests in computing the HOG descriptor. Felzenszwalb's method of computing HOG descriptors (FHOG) can reach over 125 FPS easily.[10] This implementation is used by the algorithm. However, ARM processors do not support SSE2 intrinsics, which are required for vectorization on an assembly level. ARMv7 has its own version of intrinsics -- NEON. The SSE2NEON header along with hand-written NEON code was used to give FHOG compatibility with ARMv7-A and other NEON supporting processors.

Multithreading as well as separate processes for short-term and long-term tracking are also used. A shared memory spaces allows both processes to communicate. Compilation options include the -O3 flag to maximize optimizations as well as the usage of the hardware FPU on the ARM processor.

TESTING

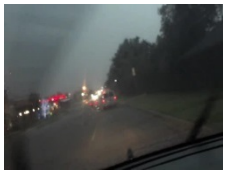
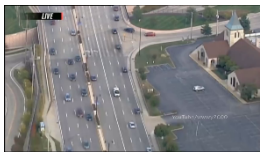
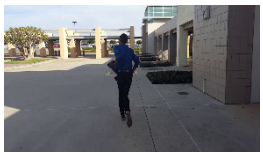
DSST has already shown very good results in VOT 2014. However, the scenarios of detection and tracking are very different from that of hunting down PreyBOTs. Thus, Eclipse Technologies has created a set of sequences to push the combination tracker to its limits.

[10] "Piotr's Matlab Toolbox." Piotr's Matlab Toolbox. N.p., n.d. Web. 23 Dec. 2015.

The tracker is evaluated on the following:

1) Accuracy - how well the tracker locates the center and size of a given object.
2) Robustness - how well the tracker identifies the presence or absence of the object.
3) Recovery - how well the tracker relocates the object after occlusion or movement off the FOV.
4) Performance - the CPU and RAM usage as well as the FPS.

DATASET

N	Sequence	Resolution	Name	Comments
1		640 x 480	Tiger	Fast motion. Small object. Partial Occlusion. Distinct Object. Sequence from VOT 2015.
2		640 x 480	Wiper	Lighting Changes. Unclear Image. Constant occlusion. Fast speed. Shaking. Sequence from VOT 2015.
3		640 x 360	Tunnel	Very small object. Drastic size changes. Bad lighting. No features on object. Sequence from VOT 2015.
4		1280 x 720	Chase	No features. Extreme motion blur. Partial and full occlusion. Extreme size and appearance change. Similar objects. Sequence from YouTube.
5		1280 x 720	Kenneth	Full occlusion. Drastic shaking, lighting changes, scale changes, and appearance changes. Similar objects. Sequence by Eclipse Technologies.

RESULTS

The dataset is in the order of easiest to most difficult. Accuracy is rated as percentage of frames where the center of the frames deviates less than 10 pixels from the actual center as determined by a human. Robustness is rated as a percentage of frames where the tracker correctly identifies the presence or absence of the object. Recovery is a percentage of successful recoveries by the tracker after occlusion or target loss. Performance is rated as two numbers, A/B, both of which are in FPS. A is the FPS of core 1 running the short-term tracker, and B is the FPS of core 2 running the long-term tracker.

No.	Accuracy	Robustness	Recovery	FPS
1	100%	99%	100%	240 / 220
2	100%	99%	100%	235 / 133
3	100%	90%	100%	263 / 200
4	100%	88%	100%	159 / 46
5	82%	75%	93%	222 / 40

On the most difficult sequence, Kenneth, the algorithms makes one wrong suggestion which causes the program to mis-converge on a trash can rather than Kenneth. As shown in the image on the left, the pink box is the suggestion, matching a template at the bottom which is also shown in purple. The green box indicates that the short-term tracker has accepted the suggestion. A correct convergence is shown on the right. The long-term tracker proposes a location after DSST loses the object due to a heavy camera shake. DSST successfully locates the object in the next frame.

Performance is very fast. On the ODROID, sequences average 19 FPS using DSST and 10 FPS using the long-term tracker on all 640 x 480 sequences. This performance satisfies constraint F fully, making this tracker the optimal choice for Eclipse Technologies.

DETECTING ENTITIES

In order to move efficiently and autonomously, both PackBOTs and Wolfgang must be able to locate and distinguish different environmental objects as well as locations. Stringent requirements included that the detection system must be able detect multiple classes of objects all in real time on a single core, perform real-time learning, understand different appearances of the same class of objects, as well as have a efficient serialization system for saving of memorized data. Eclipse Technologies proposes a novel template-based detection system based on precomputation of response maps and accelerated using NEON.

CURRENT DEFICIENCIES

The most prevalent and mature detection framework relies on the Viola/Jones Detector. However, accurate detection of a single class of objects, such as the human face, requires a few thousand positive images and a few million for training before detection accuracy can exceed 90%.[11] Training one class can take weeks if high accuracy is required. An alternative method that surfaced in recent years is the use HOG descriptors with a scanning window. This method is much faster in terms of training, but can still take upwards of hours to train a single object. One disadvantage this method has is its speed. In order to perform detection, the algorithm must compute the HOG descriptor for an entire image.[12] This is highly time consuming

[11] Viola, Paul. *The Viola/Jones Face Detector* (n.d.): n. pag. *University of British Columbia* . Department of Computer Science. Web. 18 Dec. 2015.

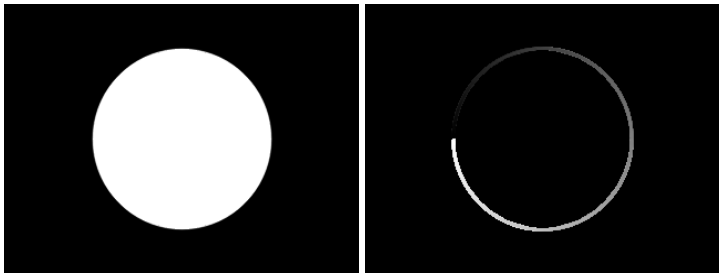
[12] "Dlib C++ Library: Dlib 18.6 Released: Make Your Own Object Detector!"*Blogger*. N.p., 3 Feb. 2014. Web. 28 Dec. 2015.

procedure. In addition, every class of object relies on a separate iteration of scanning window, drastically reducing performance.

Finally, both Viola/Jones and HOG Scanning Window are deficient for use by Eclipse Technologies as they both rely on Black Box Methods. Training images are fed to a SVM, which acts like a human brain. Once the preset neurons are trained, firing it takes very little time. However, it is impossible to distinguish objects once they enter the SVM. Bad training templates which can negatively impact performance are kept. In addition, it is impossible to serialize only one part of an SVM -- the entire SVM must be transferred to replace another. Eclipse Technologies' proposed method solves these problems.

GRADIENT ORIENTATION

It is usually sufficient to determine an object from its gradient stiletto. Stilettos allow for detection of featureless objects.



As shown in the image on the left, a featureless circle can be detected based on its gradient orientation. The gradient orientation can be thought of as the directional flow of the color gradient. This can be computed efficiently using two Sobel convolutions to compute the x and y derivatives of a given image. After thresholding is applied, the magnitude output is

ignored, as specified by [11][13]. Additionally, the gradient orientation is categorized into 8 bins from [0, 180] degrees. A final 3x3 neighborhood voting cycle confirms the accuracy of a given bin at a pixel. One difference from the original paper is that Eclipse Technologies' algorithm uses a 7x7 Gaussian blur to filter out image noise and decrease inaccuracies.

PRECOMPUTATION

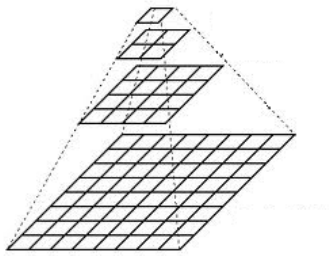
After computation of bins, the image is binarily encoded and a response map is precomputed[12]. This response map is shared between all templates, making matching a very efficient process of simply adding up numbers to compute the Steger similarity value.[14] Templates, which can be added at any time, are already precomputed with a gradient orientation map and placed in the memory for fast matching.

[13] Hinterstoisser, S., C. Cagniart, S. Ilic, P. Sturm, N. Navab, P. Fua, and V. Lepetit. "Gradient Response Maps for Real-Time Detection of Textureless Objects." *IEEE Transactions on Pattern Analysis and Machine Intelligence IEEE Trans. Pattern Anal. Mach. Intell.* 34.5 (2012): 876-88. Technische Universitat Munchen. Web. 23 Dec. 2015.

[14] Steger, Carsten. OCCLUSION, CLUTTER, AND ILLUMINATION INVARIANT OBJECT RECOGNITION (n.d.): n. pag. International Society for Photogrammetry and Remote Sensing. MVTec Software GmbH. Web. 20 Dec. 2015.

OPTIMIZATION

The speed of the algorithm comes from the 8 bins. The binary descriptor is limited to a maximum of 255.[15] During the final similarity value computation, the memory for the input image can be linearized, allowing intrinsics like SSE and NEON to perform vector additions for 16 different cells in one operation.

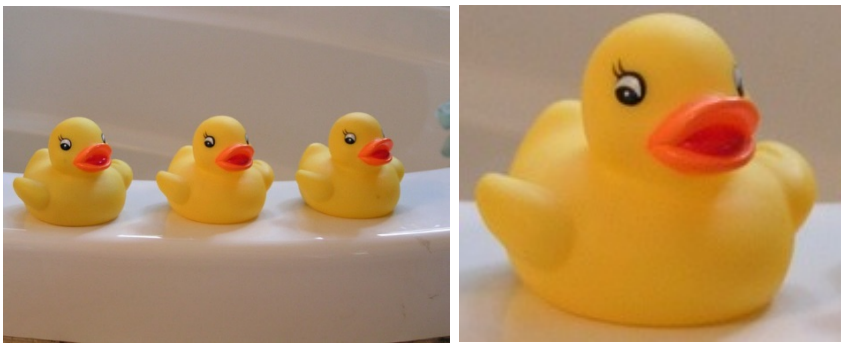


Further optimization, which is not detailed in the research, is to use a Gaussian pyramid. Precomputed templates and the input image are all placed in a pyramid with an arbitrarily determined number of layers. Matching starts from the top layer down to the bottom layer, which is the original image or template. At each layer, matched locations specify the ROI for the next layer, drastically reducing computation time. In addition, at each layer, only areas above a certain threshold are considered, furthering algorithm performance.

[15] Hinterstoisser, S., C. Cagniart, S. Ilic, P. Sturm, N. Navab, P. Fua, and V. Lepetit. "Gradient Response Maps for Real-Time Detection of Textureless Objects." *IEEE Transactions on Pattern Analysis and Machine Intelligence IEEE Trans. Pattern Anal. Mach. Intell.* 34.5 (2012): 876-88. Technische Universitat Munchen. Web. 23 Dec. 2015.

SPEED AND MEMORY COMPARISONS

The proposed method written in unoptimized C++ with SSE2 (“Ours”) will be compared to OpenCV’s state-of-the-art FFT template matching algorithm written in optimized C++ and SSE2 . The OpenCV algorithm will be used with two different template types -- RGB (“RGB”) and Canny edge (“Canny”). This test compares speed and memory consumption for storing templates. The following image and ROI is used.



OPENCV TEMPLATE MATCHING CONFIGURATION:

Canny: Threshold 1 = 50, Threshold 2 = 100.

Matching done by TM_CCOEFF_NORMED.

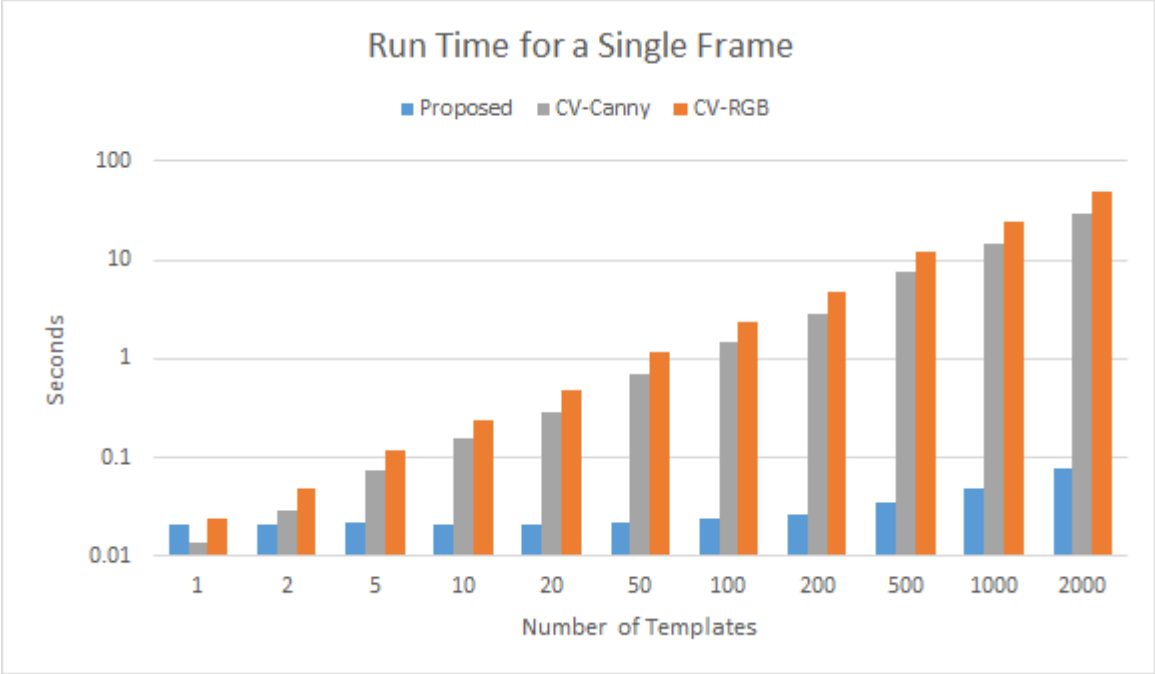
Max located using minMaxLoc().

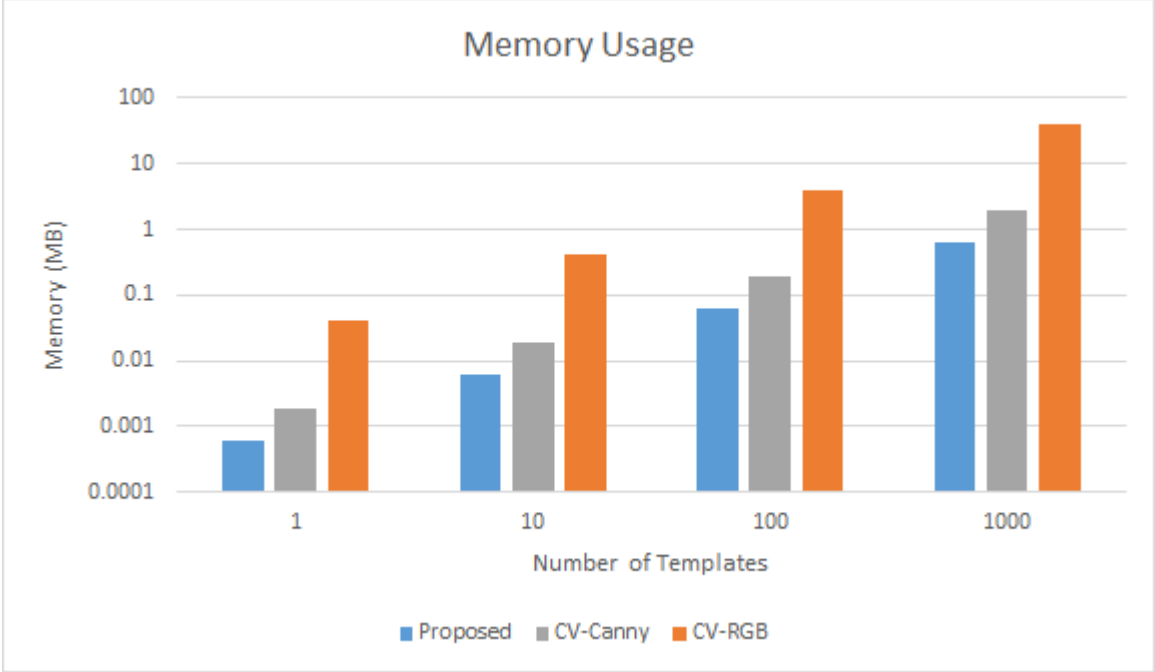
PROPOSED METHOD CONFIGURATION:

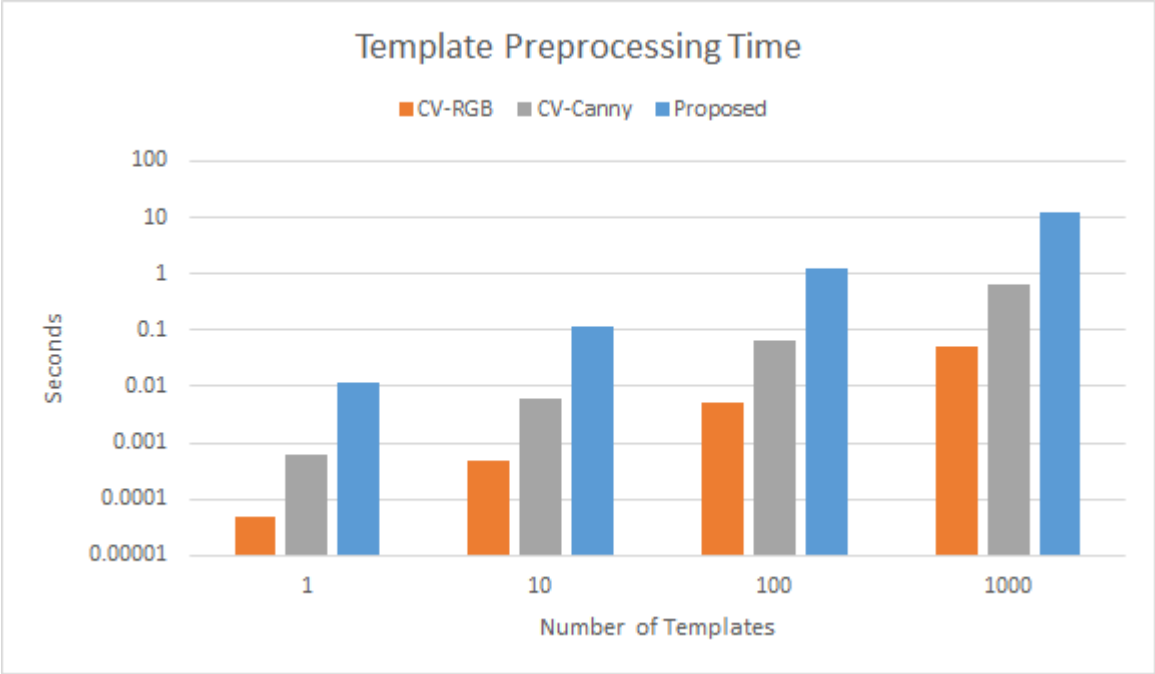
2 level pyramid with sampling steps of { 5, 8 } at each level respectively.

Locating 63 features in pyramid.

Weak Threshold = 10.0, Strong Threshold = 55.0.


Run Time for a Single Frame
Proposed
CV-Canny
CV-RGB
100
10
1
0.1
0.01
Seconds
1
2
5
10
20
50
100
200
500
1000
2000
Number of Templates


Memory Usage
100
10
1
0.1
0.01
0.001
0.0001
Memory (MB)
1
10
100
1000
Number of Templates
Proposed
CV-Canny
CV-RGB



As shown through testing, the proposed algorithm compares extremely favorably to state-of-the-art OpenCV algorithms. For larger number of templates, it is orders of magnitude faster than the RGB template matching algorithm. Note that RGB takes almost no preprocessing time, as the comparison directly uses the ROI. In terms of memory consumption, even the 2-layer pyramid takes up only around 2.3 MB of memory for 1000 templates.

ACCURACY

The accuracy of the algorithm has already shown to be superior to every other current known template matching methods.[16] More details can be seen in the research paper. The

[16] Hinterstoisser, S., C. Cagniart, S. Ilic, P. Sturm, N. Navab, P. Fua, and V. Lepetit. "Gradient Response Maps for Real-Time Detection of Textureless Objects." *IEEE Transactions on Pattern Analysis and Machine Intelligence IEEE Trans. Pattern Anal. Mach. Intell.* 34.5 (2012): 876-88. Technische Universitat Munchen. Web. 23 Dec. 2015.

following tests performed by Eclipse Technologies are simply used to verify the implementation of the algorithm.

Eclipse Technologies reverse the short-term tracking technology as a trainer. An ROI is defined. The user will rotate and scale the object of interest while the algorithm learns the appearance of the object. After learning a predefined number of templates, the trained data is exported and then used on static images to detect objects.

For example, a stapler is tracked and learned using DSST. After the long-term tracker has learned the appearance of the stapler by collecting 100 templates, it is then exported. Later, a phone image containing the same stapler is used as the input and the trained algorithm successfully detects the stapler and its bounding box.

Eclipse Technologies has also done tests using faces, bottles, books, etc. All tests were done by collecting 100 templates. After which, using a detection threshold of 80%, the algorithm successfully locates the target under different lighting conditions with over 97% accuracy and no false positives.

TEMPLATE SHARING

The power of a template system is that templates can be shared. Eclipse Technologies' algorithm supports exports and imports of individual YAML or XML formats. Each template pyramid data in its serialized state (pure text) is under 4 KB and can be easily transferred over the network. Entire classes containing hundreds or thousands of templates can also be exported this way and loaded. This allows one robot who has learned the appearance of the PreyBOTs to distribute it's learned class to other PackBOTs, reducing learning time from a minute of observation to mere seconds for data transfer. An example of the YAML file format can be seen in the image below.

```yaml
%YAML:1.0
class_id: face
modalities: [ ColorGradient ]
pyramid_levels: 2
template_pyramids:
   -
      template_id: 0
      templates:
         -
            width: 193
            height: 260
            pyramid_level: 0
            features:
               - [ 4, 123, 7 ]
               - [ 172, 144, 7 ]
               - [ 131, 5, 5 ]
               - [ 49, 250, 0 ]
               - [ 190, 260, 2 ]
               - [ 27, 20, 2 ]
               - [ 93, 92, 7 ]
               - [ 85, 175, 1 ]
               - [ 189, 63, 7 ]
```

WALKING SYSTEM

LEG KINEMATICS

FORWARD KINEMATICS

Both PackBOTs and Alpha have legs defined by three servos. However, the three servos form two different configurations, creating interesting kinematic problems.

CONFIGURATION 1: DOG

Configuration one consists of two servos at the hip. The first servo allow for rotation about the y-axis while the second servo allows for rotation about the x-axis. The third servo at the knee controls a rotation about an axis determined by the first two servos. Configuration two also has two servos at the hip. However, the second servo allow for rotation about the z-axis rather than the x-axis. Similarly, the third knee servo's axis of rotation is determined by the first two servos.

Configuration one is more used and should be analyzed first. It is possible to visualize servos or joints as points in space and ligaments as line segments between those points. One special aspect of the existence of two servos at the hip is that it can be seen as essentially one point with two degrees of freedom, simplifying the problem drastically.

A general principle for analysis is to go from the end effector to the root. However, definition must come before analysis. The root is located at $(0, 0, 0)$ with a neutral leg position along the z-axis in the negative direction. Thus, the end effector in the neutral position should be $\big(0, 0, -(l_1 + l_2)\big)$ where l_1 and l_2 are the lengths of segments one and two respectively. Defining the root as (x_1, y_1, z_1) and the end effector as (x_3, y_3, z_3), the neutral points are as follows:

$$\begin{bmatrix} x_1 \\ y_1 \\ z_1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \;\&\&\; \begin{bmatrix} x_2 \\ y_2 \\ z_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ -l_1 \end{bmatrix} \;\&\&\; \begin{bmatrix} x_3 \\ y_3 \\ z_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ -(l_1 + l_2) \end{bmatrix}$$

The location of point two points is described by two rotations – one about the x-axis and then another about the y-axis. The reversed order is due to a parent-child link effect. The y-axis rotation controls the x-axis rotation so the x-axis rotation must be done first. Let θ_2 be the angle of servo two. The first rotation is completed by performing a dot product of the coordinates of point two with an x-axis rotation matrix.

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta_2 & -\sin\theta_2 \\ 0 & \sin\theta_2 & \cos\theta_2 \end{bmatrix} \cdot \begin{bmatrix} 0 \\ 0 \\ -l_1 \end{bmatrix} = \begin{bmatrix} 0 \\ l_1 \sin\theta_2 \\ -l_1 \cos\theta_2 \end{bmatrix}$$

Taking the result of the above and performing a rotation about the y-axis:

$$\begin{bmatrix} \cos\theta_1 & 0 & \sin\theta_1 \\ 0 & 1 & 0 \\ -\sin\theta_1 & 0 & \cos\theta_1 \end{bmatrix} \cdot \begin{bmatrix} 0 \\ l_1 \sin\theta_2 \\ -l_1 \cos\theta_2 \end{bmatrix} = \begin{bmatrix} -l_1 \cos\theta_2 \sin\theta_1 \\ l_1 \sin\theta_2 \\ -l_1 \cos\theta_1 \cos\theta_2 \end{bmatrix} = P_2$$

The completed rotations yield workable functions for x_2, y_2, and z_2. They are shown below in non-matrix format.

$$x_2 = -l_1 \cos\theta_2 \sin\theta_1$$

$$y_2 = l_1 \sin\theta_2$$

$$z_3 = -l_1 \cos\theta_1 \cos\theta_2$$

The next step is to analyze the third servo. The knee servo in neutral position defines a rotation about the y-axis translated l_1 units down on the z-axis. Incorporating θ_3, the third servo's angle (relative to the z-axis), the last point's coordinates becomes the following:

$$\begin{bmatrix} x_3 \\ y_3 \\ z_3 \end{bmatrix} = \begin{bmatrix} -l_2 \sin\theta_3 \\ 0 \\ -l_1 - l_2 \cos\theta_3 \end{bmatrix} = P'_3$$

Performing the same series dot products to rotate the third point about the x-axis and then about the y-axis with rotational matrices $R_x(\theta_2)$ and $R_y(\theta_1)$:

$$R_y(\theta_1) \cdot [R_x(\theta_2) \cdot P'_3] = \begin{bmatrix} -l_2 \cos\theta_1 \sin\theta_3 + \sin\theta_1 \cos\theta_2 \, (-l_1 - l_2 \cos\theta_3) \\ \sin\theta_2 \, (l_1 + l_2 \cos\theta_3) \\ l_2 \sin\theta_1 \sin\theta_3 + \cos\theta_1 \cos\theta_2 \, (-l_1 - l_2 \cos\theta_3) \end{bmatrix}$$

Expanding the equations and simplifying by substitution of P_2:

$$x_3 = x_2 - l_2 \cos\theta_1 \sin\theta_3 - l_2 \cos\theta_2 \cos\theta_3 \sin\theta_1$$

$$y_3 = y_2 + l_2 \cos\theta_3 \sin\theta_2$$

$$z_3 = z_2 + l_2 \sin\theta_1 \sin\theta_3 - l_2 \cos\theta_1 \cos\theta_2 \cos\theta_3$$

The forward kinematics equations for the end effector were tested against the original quaternion-based kinematics system. Tests of all valid points of θ_1, θ_2, and θ_3 return a match.

```
assert(forward(lengths, angles) == quaternion(lengths, angles))
```

CONFIGURATION 2: ALPHA

Configuration two is exclusively for the back legs of Alpha. The only difference from configuration one is that configuration two has servo two rotating about the z-axis rather than the x-axis. Skipping all the previous steps, the resultant points for the end points would be:

$$R_y(\theta_1) \cdot [R_z(\theta_2) \cdot P'_3] = \begin{bmatrix} -l_2 \sin\theta_3 \cos\theta_1 \cos\theta_2 + \sin\theta_1 (-l_1 - l_2 \cos\theta_3) \\ -l_2 \sin\theta_2 \sin\theta_3 \\ l_2 \sin\theta_1 \sin\theta_3 \cos\theta_2 + \cos\theta_1 (-l_1 - l_2 \cos\theta_3) \end{bmatrix}$$

Similarly, by ignoring θ_3, it is possible to obtain the locations of P_2 for the second leg configuration.

$$R_y(\theta_1) \cdot [R_z(\theta_2) \cdot P_1] = \begin{bmatrix} -l_1 \sin\theta_1 \\ 0 \\ -l_1 \cos\theta_1 \end{bmatrix}$$

Thus, the forward kinematics for the end effector can be described using three equations like configuration one.

$$x_3 = x_2 - l_2 \sin\theta_3 \cos\theta_1 \cos\theta_2 - l_2 \sin\theta_1 \cos\theta_3$$

$$y_3 = y_2 - l_2 \sin\theta_2 \sin\theta_3$$

$$z_3 = z_2 - l_2 \sin\theta_1 \sin\theta_3 \cos\theta_2 - l_2 \cos\theta_1 \cos\theta_3$$

INVERSE KINEMATICS

Inverse Kinematics is taking the “inverse” of the forward kinematics equations. However, this is not as simple as it may sound. It is nearly impossible to deduce from pure mathematics the necessary equations that would work.

IK: CONFIGURATION ONE

Configuration one can be analyzed from end effector to root. The knee joint is uniplanar in rotation. A triangle is formed by the end effector and the root with θ_3 being the only know angle. The sides of the triangle – l_1, l_2, and the distance between end effector and the origin – are all known. Thus, it is possible to use the law of cosines to determine the value of θ_3 given the desired target coordinates.

$$\theta_3 = \cos^{-1}\left(\frac{{l_1}^2 + {l_2}^2 - {x_3}^2 - {y_3}^2 - {z_3}^2}{2\, l_1\, l_2}\right) - \pi$$

Given θ_3, it is possible to find θ_2 by plugging it into the original forward kinematics equation, as y_3 from the forward kinematics equation only relies on θ_3 and θ_2

$$\theta_2 = \sin^{-1}\left(\frac{y_3}{l_! + l_2 \cos\theta_3}\right)$$

θ_1 is much more mathematically complex to derive. However, given θ_3 and θ_2, geometric analysis shows that θ_1 affects the end effector in that it rotates it in a circle. Given the initial angle, it is possible to derive the final angle through inverse tangent.

$$\theta_1 = \tan^{-1}\left(\frac{z_3}{-x_3}\right) + \tan^{-1}\left(\frac{l_1 \cos\theta_2 + l_2 \cos\theta_2 \cos\theta_3}{l_2 \sin\theta_3}\right)$$

When creating the inverse kinematics function using code, it is necessary to utilize the *arctan2* function, as the function will determine from the y and x values in which quadrant the end effector lies.

There are multiple solutions to the kinematics equations. θ_3 can be a relative angle which is either positive or negative. θ_2, being an arcsine, can also be $\pi - \theta_2$. This provides the system with four valid solutions:

$$(\theta_1, \theta_2, \theta_3)\ ||\ (\theta_1, \theta_2, -\theta_3)\ ||\ (\theta_1, \pi - \theta_2, \theta_3)\ ||\ (\theta_1, \pi - \theta_2, -\theta_3)$$

IK: CONFIGURATION TWO

It is not necessary to perform a full 3D kinematic analysis on configuration two, as the leg does not really need to do anything except move in the 2D xz-plane and rotate about the z-axis to turn when it is standing up. Thus, only a 2D inverse kinematics problem has to be solved.

Similar to configuration one, the third servo's angle can be solved using the law of cosines:

$$\theta_3 = \cos^{-1}\left(\frac{{l_1}^2 + {l_2}^2 - {x_3}^2 - {y_3}^2 - {z_3}^2}{2\, l_1\, l_2}\right) - \pi$$

θ_1 is again just a rotation to the right point once θ_3 has been determined. The equation is almost exactly like configuration one.

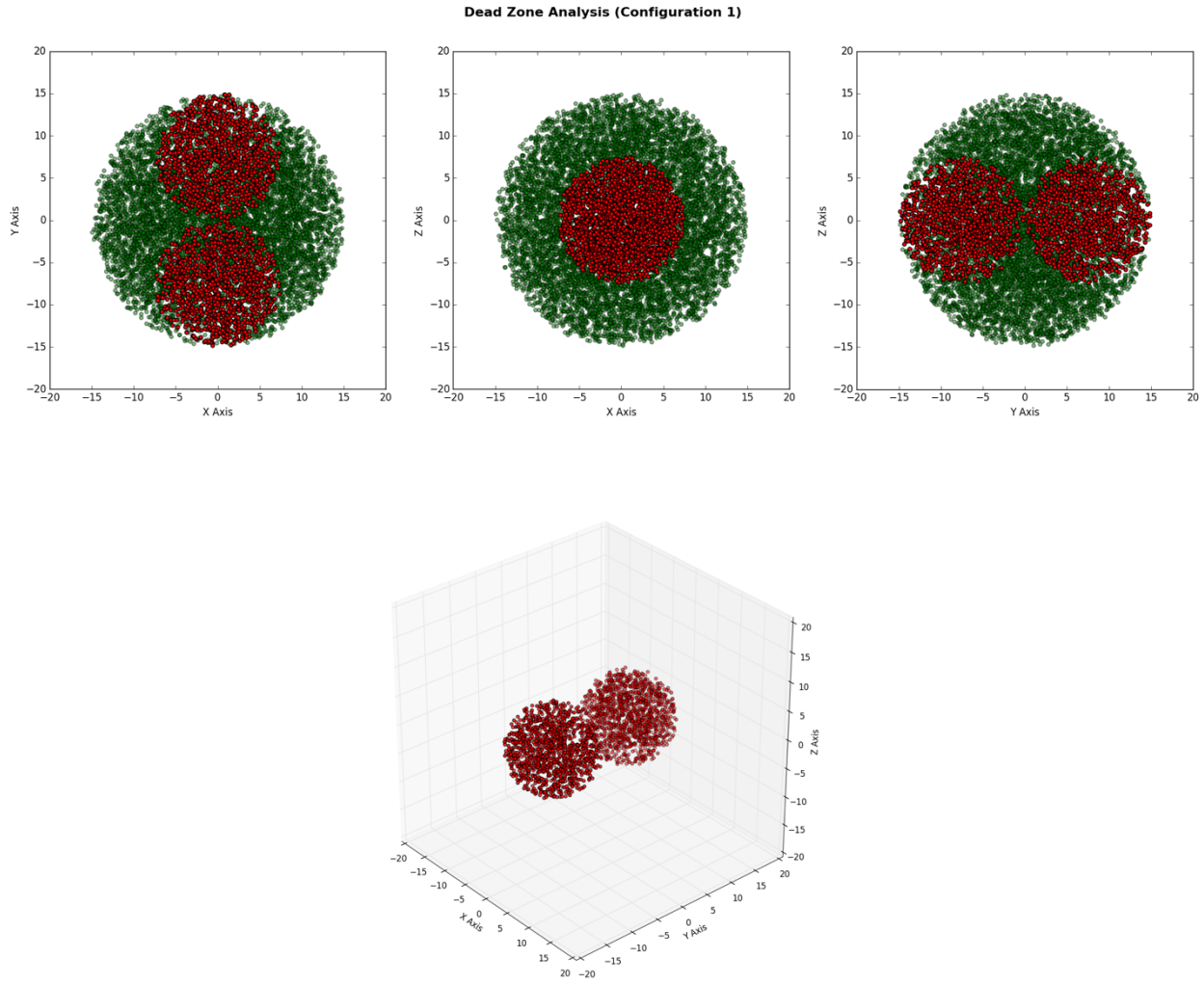
$$\theta_1 = \tan^{-1}\left(\frac{z_3}{-x_3}\right) + \tan^{-1}\left(\frac{l_1 + l_2 \cos\theta_3}{l_2 \sin\theta_3}\right)$$

ALGORITHM SPEED TESTING AND OPTIMIZATION

Solving for the inverse kinematics and obtaining functional equations allows the system to operate extremely fast. Optimization is done through factoring trigonometric operations out of the functions as much as possible as trig operations are comparative slow. In the optimized Python version, the solver can produce over 100,000 solutions per second, which allows for real-time computation of IK.

DEAD ZONE TESTING, ANALYSIS, AND OPTIMIZATION

Every kinematic system is not perfect. There are certain dead zones for each configuration, or regions where the leg cannot reach. Configuration two does not need to be analyzed, as it will be the same as the kinematic analysis of configuration one in the xz-plane.



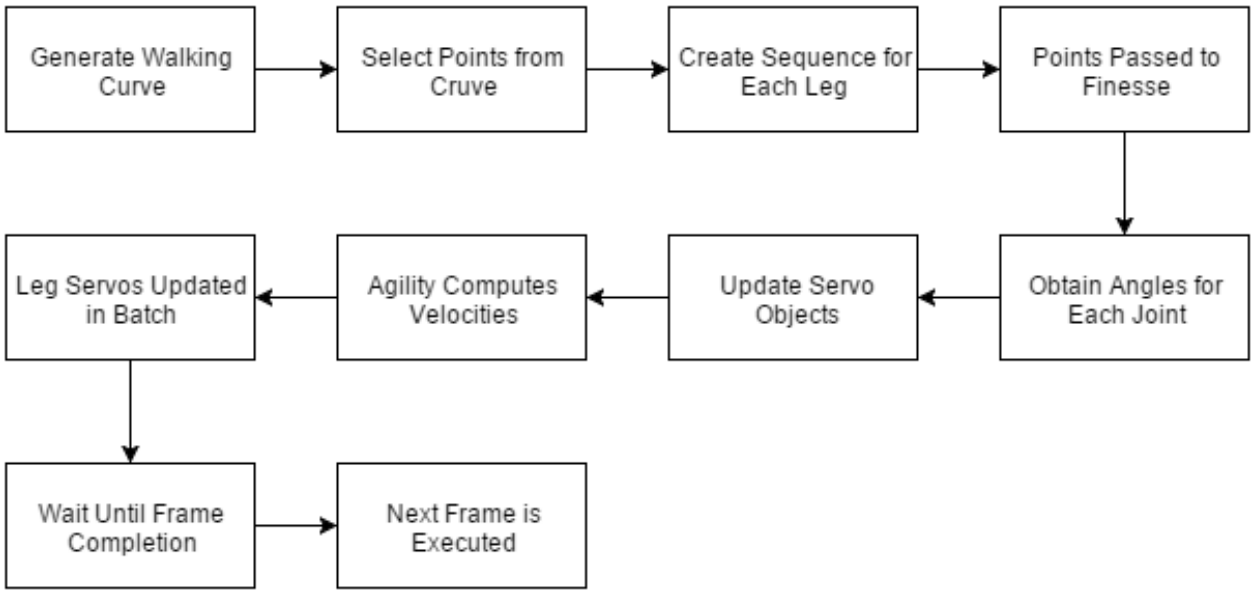
The images above shows the dead zone analysis for configuration one. 10,000 uniformly chosen random points are used. The green circles indicate the regions that the robotic leg can reach, while the red circles indicate points where the robotic leg cannot reach. Red points can

exist either from a domain error by the inverse kinematics function or when the result returned by that function is more than 0.1 mm away from the true point.

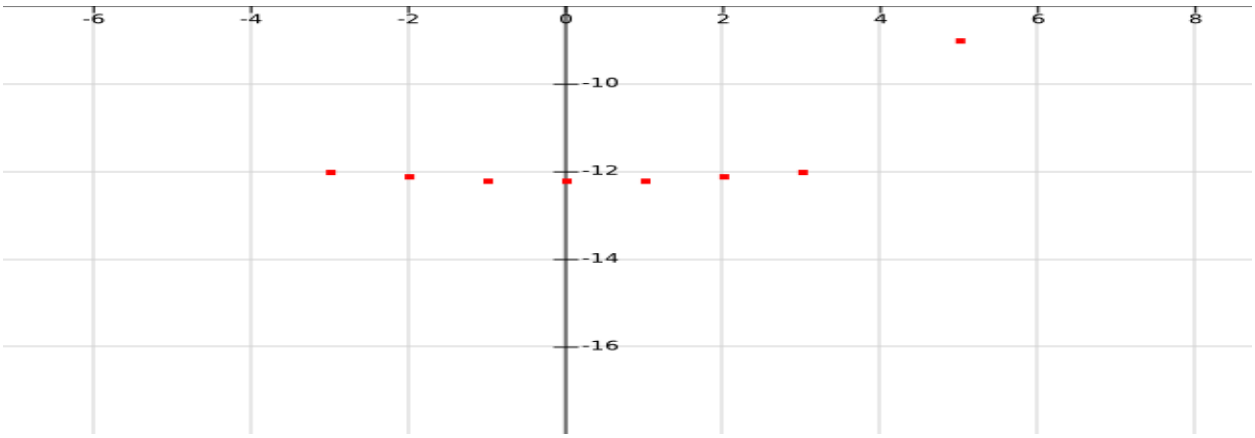
There is a tradeoff between torque and dead zone volume. If the last segment l_2 is made shorter, the dead zone area would decrease. Eclipse Technologies chose to have l_2 and l_1 be the same length. The dead zone analysis for the 7.5 cm leg segments is shown in the images.

GAIT GENERATION AND EXECUTION

Gait generation and execution is done through a combination of functions and relies heavily on the Finesse Module. The actual generation and execution procedures are embedded inside Agility. A diagram of a basic iteration of gait can be seen below.



WALKING CURVE AND POINT SELECTION

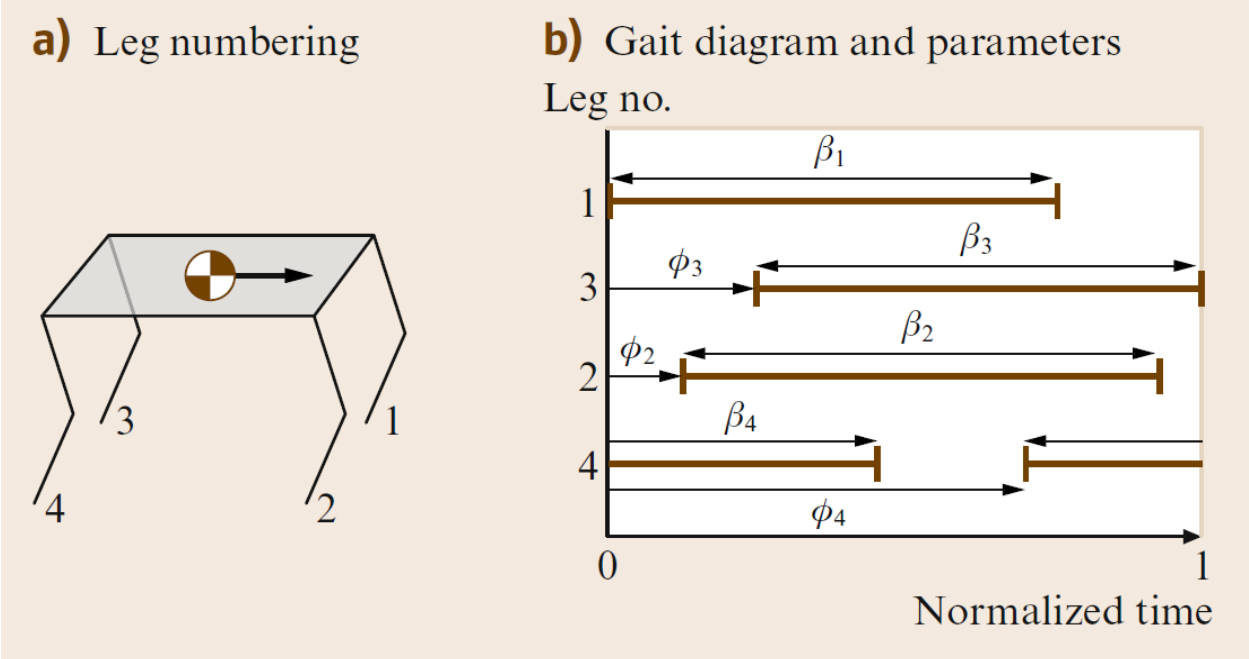


Walking curves are often parabolas or manually calibrated points. For parabolas, the vertex is generally chosen as a point while others can be arbitrarily and symmetrically selected. The crawl gait manual sequence is shown below.

The points are only located in the xz-axis, as no leg tilting needs to be done. The highest point is the top of the descent. The seven points in the shape of a parabola is the drag sequence. Finally, the leftmost point to the highest point is the return.

SEQUENCE CREATION

A crawl gait sequence is shown below. The beta value is the amount of time a leg is on the ground. According to springer, the optimal time for beta is 0.75 of the total frame time [1]. Beta can be greater than 0.75 but should not be less for optimal stability [1].



Trot is similar to crawl, except legs 14 and 23 move together. Beta in this case would be exactly 0.5.

SYNCHRONIZATION

A difficult aspect that must be considered is that the returned angles are different for every servo. However, all servos must be synchronized and arrive at their destination together. Eclipse Technologies has developed an algorithm that leverages the velocity controls of the Mini Maestro.

A velocity control is available as an inherent part of the Maestro. It has units of $1\ pwm\ /\ 10\ ms$. Using angular velocity, the velocities can be set before the positions are executed.

$$\theta = \theta_i + (k\omega)t$$

Given that ω is $1\ pwm\ /\ 10\ ms$ and that t is a known constant of time, k can be solved for. k is quantities of ω which controls the velocity in the servo controller.

$$k = 10\frac{\Delta\theta}{t}$$

The velocities of all servos for each frame are set before they are instructed to move to their respective targets. The Maestro will handle the velocity change. Because servo objects keep their maximum velocity, it is possible for the shortest t for a frame to be computed through:

$$t_{min} = \max\left(10\frac{\Delta\theta}{vel_{max}}\right)$$

SPEED TESTS

Servos run at 50 Hz, meaning that the computation for an entire sequence should not exceed 20 ms in order to catch the servo's next update. Tests are done on each individual function.

Operation	Number of Iterations	Average per Iteration (ms)
Check Moving State	10,000	0.191
Target Euclidean (1 leg)	10,000	0.016
Target + Set (3x) + Flush	10,000	0.260
Flush (12 bytes)	10,000	0.001
Update Servo Positions (12x)	10,000	0.271
Sync and Execute (12x)	1,000	0.732

Tests indicate that a total execution time for everything should not exceed 3 ms even in the works case scenario. This is far below the 20 ms threshold, allowing slower processors like the ODROID to process the same data without missing a servo loop.

RECOVERY

The PreyBOTs and Alpha have the ability to recover whenever it falls on its back or becomes unbalanced. These robots are equipped with a gyroscope, a device that determines the orientation and direction of a certain object. Our team has realized that there might be a possibility that the robot will flip over or get on its back. In the case that the robots do fall off into their backs, both DOGs are equipped with eight wide angle servos. If the IMU detects that the robot has fallen, Agility will generate a simple recovery sequence by flattening all joints to be parallel to the body and then rotating two selected joints quickly by 180 degrees to flip the robot back to it's original positions. The joints can then reorient themselves to their pre-fall state.

SERVOS

Servos are small devices that rotate a shaft in a certain amount of degree. These small devices are essential to the success of the mission. Without servos, the robots would not be able to move. These servos are responsible for creating movement for the robots. Think of the servos as muscles and bones for a human body. The body uses muscles and bones to do several actions. These servos would have the same purpose in the robots when it tries to do the actions.

SERVO ABILITY

Eclipse Technologies has chosen servos as the primary source of mechanical power for the robots. The company has to be certain that these servos are capable of handling the amount of work that the robot needs in order to move swiftly and efficiently.

The company has decided to test the ability of these servos in order to ensure the success of the mission. Without testing the servos, there might be a great chance of failure. The purpose of the servos degree of Freedom Test is to identify the ability of the servos to have at least 270 degrees of freedom.

MATERIAL

- Paper
- Sharpie
- 5 servos
- USB
- Raspberry Pi
- Computer
- Tape tags
- 6 Volt battery

PROCEDURE

Connect the servos to the the Raspberry Pi. Then connect the Raspberry Pi to the 6V battery and the computer.

- Set the servo to zero. Then, make a mark with a pencil, where it's placed at zero.
- Max out the servo. While doing so, hold onto the paper and the servo.

- Mark where the servo ends and mark.
- Measure angle made by marks.

Servo's Degrees of Freedom Test Results (November 19, 2015)

All the Servo's were supposing to have at least 270 degrees of freedom. We were successful with 4 of the 5 servos. Servo 1,2,3 and 5 had more than 270 degrees of freedom. Meanwhile, servo number 4, had below 270 degrees of freedom.

Servo Identification Number	Degrees of Freedom
1	287.0
2	281.0
3	276.5
4	268.5
5	271.5

Even though. the degrees of freedom are not really accurate, nor precise. We will compensate the degrees of freedom by using a computer program.

TRANSFORMATION SYSTEM

Alpha requires a stimulus to trigger the transformation from a four legged state into a two legged state due to ultraviolet light. Thus, an Ultraviolet Sensor will be stimulated by an ultraviolet light which will be located in the morphing chamber. The test ensures that the UV Sensor activates exclusively to ultraviolet light and not other sources such as regular light bulbs and sunlight. It will also test whether Alpha will transform due to the stimulus of ultraviolet light.

ULTRAVIOLET (UV) SENSOR

MATERIALS

- Classroom light bulbs
- Ultraviolet Sensor
- Computer
- Engineering Notebook
- Pen
- Ultraviolet light
- Aluminum sheet

PROCEDURE

1. Connect the chip to the computer and then place an aluminum sheet on top, zero it.
2. Expose the UV light sensor to the Classroom lights and then zero it.
3. Expose the UV light sensor to the Sunlight and zero it.

4. Place the UV light sensor inside the Morphing chamber, exposing it to the UV light. The sensor must activate. If it doesn't, program sensor to do so.

RESULTS

UV Sensor Data	
Covered with box	0.004 - 0.007 mW/cm^2
Ambient room	0.516 - 1.152 mW/cm^2
Ambient room with sunlight through window	1.365 - 1.945 mW/cm^2
Outside on a sunny day	1.365 - 1.945 mW/cm^2

CONCLUSION

The UV sensor is able to detect the difference between ambient light in a room and light from the sun. This proves that the sensor will be able to detect if it is in an enclosed room filled with UV light.

ALPHA - AUDIO

In addition to the ultraviolet stimuli, Alpha requires an additional Audio Sensor in order for it to transform from a four legged state into a two legged state. The Audio Sensor will be activated by the G-note in the song "Werewolves of London", which will be playing in the chamber. This test ensures that the the Audio Sensor will only be triggered when the G note during the song is played.

BACKGROUND TEST

MATERIALS

- Audio Sensor
- Engineering Notebook
- Pen
- Youtube - Computer

PROCEDURE

1. Place the audio sensor outside the chamber room.
2. Observe if the background noise activates the sensor. If it does, zero the background sound.
3. Place the audio sensor inside the chamber room.
4. Annotate whether or not the sensor turns on. It should turn on. In the event that it doesn't, program the audio sensor again to turn on when the G note is played.
5. Play the song at different volumes.
6. Measure the frequencies of the howls and make sure they fall inside the margin of error.

RESULTS

The program was able to detect the g note only when an adequate amount of background noise was present, but with large amounts of background noise the program can not detect the note.

ACCURACY TEST

MATERIALS

- Audio Sensor
- Engineering Notebook
- Pen
- Computer Program
- Piano App

PROCEDURES

1. Connect the sensor to the computer and run the program. Open the piano app on a phone.
2. Press a note on the piano and check if the program detects the correct music note that is playing.
3. Repeat step 2 with all 7 music notes. If program can not detect the right note then reprogram.
4. Annotate findings in the engineering notebook.

RESULTS

The program was able to detect all of the notes, (A, B, C, D, E, F, and G), when played on the piano.

ACCURACY TEST #2

MATERIALS

- Audio Sensor
- Engineering Notebook
- Pen
- Computer Program
- Werewolves of London

PROCEDURES

1. Open the computer program and run it. Start playing the Werewolves of London
2. While the song is playing check to see if the program detects the g note in the howl. If it can not detect the howl then reprogram.
3. Annotate findings in the engineering notebook.

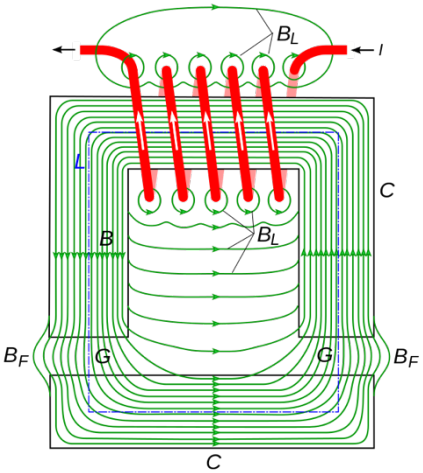
CONCLUSION

The program was able to detect the G note in Werewolves of London in 10 trials, thus making the test successful.

BALL MANIPULATION SYSTEM

ELECTROMAGNET

The Electromagnet is a crucial part in the mission as it is necessary in the manipulation of the steel ball. The ball must be picked up and accurately placed into a cup which will initiate the rest of the mission. While the ball is not guaranteed to be magnetic, the magnet will guarantee the most efficient mode of manipulation in the case that it is. The ball is guaranteed to be conductive, which usually goes hand in hand with magnetism, with exclusions of course. There are two main aspects of the electromagnet that must be tested to make sure that the magnet can complete the mission. The break weight of the magnet is necessary as we must figure just how much the magnet can hold. We know that the magnet will be able to handle a steel ball with ease, but the worry is that the object's inertia will allow the ball to come off when the Alpha sets in motion. The pull distance is also necessary so that calibrations can be done in Alpha's approach to the steel ball. A certain fear that even a slight misstep by Alpha will cause it to trip over the ball, possibly damaging Alpha beyond recovery.



ELECTROMAGNET BREAK WEIGHT

The magnet is set horizontally on a table and turned on. A one-inch steel ball is placed onto the magnetic surface and pull using a spring scale measuring when and at what force the ball comes off of the magnet. This is to ensure that the magnet will be able to hold onto the ball no matter the situation.

MATERIALS

- 1-inch steel ball
- Spring Scale
- Electromagnet
- Engineering Notebook

STEP BY STEP PROCEDURE

1. Attach the steel ball to the spring scale, without hindering the steel (covering the steel where the magnet will be placed)
2. After turning the electromagnet on, stick the ball to the magnet
3. Have the magnet turned sideways, and pull horizontally on the scale
4. Observe the strength required to break the ball free of the magnet and record
5. Repeat for accuracy

RESULTS

Trial	Break in Kg
1	4.5
2	4.2
3	4
4	4.7
5	5

CONCLUSION

The magnet is rated to hold 5Kg when metal is placed on the magnet, covering its full surface area.

The Steel ball does not cover the full surface area of the magnet and therefore cannot be expected to be held at the rate 5kg. The 5Kg break mass means that it can withstand 41 Newtons of force before being removed from the magnet, and the magnet holding the steel ball has an average 4.48Kg break mass meaning a 43.9 Newton brake force is necessary to remove the steel ball from the magnet.

A theoretical weight of a 1-inch steel ball is 44 grams or .044Kg, meaning the weight of the ball would be .43 Newtons. The magnet is more than sufficient in the manipulation of the steel ball as the force required to remove the ball from the magnet is 100 times greater than the weight of the ball itself.

ELECTROMAGNET PULL DISTANCE

Using a lightweight, coin like steel, the magnetic field is tested to see how close the magnet must be to pick up the ball. The magnet is set horizontally on the table next to a ruler. The metal piece is pushed closer and closer to the magnet until the magnet itself pulls the object closer. This process is repeated for accuracy

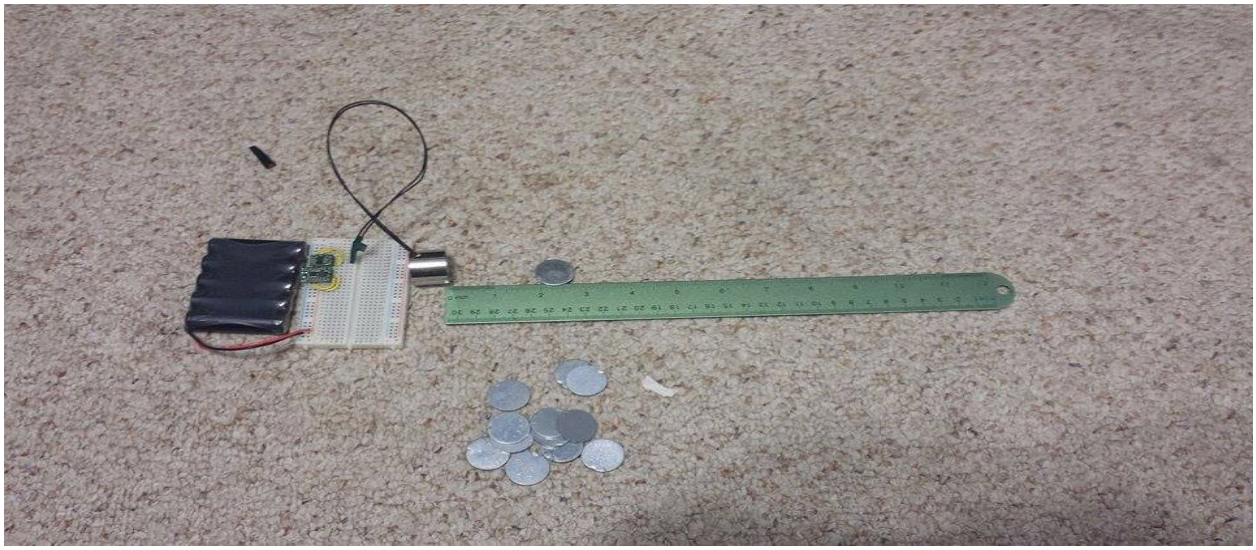
MATERIAL

- Magnetic Coin (Quarter Size)
- Ruler
- Electromagnets
- Engineering Notebook
- Pen

PROCEDURE

1. Place the electromagnetic 10 cm away from the magnetic coin.
2. The magnet should not have an attraction at this point

3. Push the coin closer and closer to the magnet until the coin is pulled to the magnet by the magnetic force
4. Observe the distance at which this happens
5. Repeat for accuracy with multiple different coins



RESULTS

Trial	Distance (in)
1	.5
2	.6
3	.4
4	.6
5	.6
6	.7
7	.6
8	.8

CONCLUSION

The magnetic pull of the electromagnet is quite small. Its pull distance is at an average of .6215 inches away from the magnet. This is an ideal distance as Alpha will attempt to step on the steel ball. Alpha will be calibrated so that the plate of the magnet will actually make contact with the steel ball before being actuated

LEG MOVEMENT

In order to move the steel ball into the cup, Alpha will utilize the three degrees of freedom in its front legs. Alpha will bend over to move its arms within range so it will be attracted to the electromagnet. An algorithm will then determine the quickest way to get the ball to the cup. The servos powering the leg need to move in a range of at least 267 degrees in order to reach the zones where the cup will be. Tests will be run to determine if the servos can move in the correct range.

SERVO DEGREE OF FREEDOM TEST PROCEDURE

Each servo will be connected and powered by the Raspberry Pi. After the servo is zeroed, marks are made at each of the servo's limits. The marks are then measured with a protractor to calculate the degrees at which the servo can rotate.

MATERIAL

- Paper
- Sharpie

- 5 servos
- USB
- Raspberry Pi
- Computer
- tape tags
- 6 Volt battery

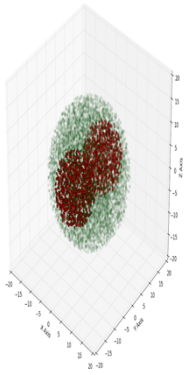
PROCEDURE

1. Connect the servos to the the Raspberry Pi. Then connect the Raspberry Pi to the 6V battery and the computer.
2. Set the servo to zero. Then, make a mark with a pencil, where it's placed at zero.
3. Max out the servo. While doing so, hold onto the paper and the servo.
4. Mark where the servo ends and mark.
5. Measure angle made by marks.

RESULTS

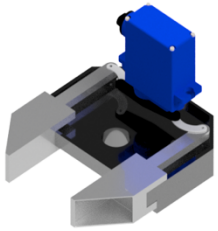
Servo Identification Number	Degrees of Freedom
1	287.0
2	281.0
3	276.5
4	268.5
5	271.5

CONCLUSION



Each servo was able to move in the target range. This would allow Alpha's legs to utilize the necessary zone to reach the cup as seen in the diagrams below.

MECHANICAL PICKUP



In case that the steel ball cannot be grabbed by the magnet, one of the legs will be equipped with a paw that can mechanically grab onto the 1-inch ball. Powered by a servo, two claws will slide together to firmly grasp the ball. The servo is attached to a servo horn and the servo horn has a curved piece on each side of it. When the servo turns one way, each part of the claw will slide out and when the servo turns the other way, the claws will slide into each other.

TRAPPING MECHANISM

IDENTIFICATION OF TARGET PREY

RFID stands for Radio Frequency Identification, and ishow the target will be identified. RFID functions by the scanner sending out a radio pulse to the chip, this pulse may provide power to the chip or the chip may be self powered. After receiving the pulse, the chip will pulse back with a multi digit ID code. The PackBOTs will have a camera which will be able to identify and track moving objects such as the PreyBOTs. Mother Goose, Jaw and BOP will immobilize the PreyBOTS so that a PackBOTs could approach it and scan the RFID tag. The RFID scanners will be inside the PackBOTs jaw, and will be able to identify and capture the correct PreyBOT. The RFID tag is going to be located on the head of the PreyBOT. Each chip has a unique code which will be scanned by the RFID scanner. That code is what determines which PreyBOT is the correct robot to capture.

RFID TAGS DETECTION

When the PackBOTs grab the PreyBOTs, the RFID scanner should be in a sufficient range to read the tag on the captured PreyBOT. Since PreyBOTs are most likely to not stay still, testing the scanner with different materials obstructing the RFID tag at different angles will give us more realistic results, helping us finalize our design.

MATERIALS

- RFID Tags
- RFID Scanner

- Computer (Command Center)
- Engineering Notebook
- Pen
- Metal parts (steel, aluminum, nylon, and plastic) with different dimensions

STEP BY STEP PROCEDURE

1. Place the RFID tag behind an aluminum sheet.
2. Place the RFID scanner in front of the aluminum sheet, with an overall distance of 2 cm.
3. Check Command Center to determine whether or not the RFID scanner was able to scan the RFID tag. Annotate the data.
4. Repeat step 1-3 with different materials between the RFID scanner and the RFID tags, such as steel, nylon, and plastic.

RESULTS

The RFID Tags were able to be read by the RFID scanner with almost any type of material in between them, with the exception of metals.

4 tags, paper plastic, steel alum, cellphone, yes to all except cellphone and metals

MATERIALS

- RFID Tags
- RFID Scanner

- Computer (Command Center)
- Engineering Notebook
- Pen
- Ruler

PROCEDURE

1. Connect the RFID scanner to the computer program and place it flat on a table.
2. Grab a ruler and place it vertically next to the RFID Scanner.
3. Place an RFID tag about 10 inches from the RFID scanner and then slowly bring the RFID tag closer to the reader.
4. Once the reader makes a “beep” sound, mark the distance between the tag and the reader it took for the reader to be able to identify the tag.
5. Repeat steps 2-4 and annotate your findings.

RESULTS

The maximum distance for the scanners to be able to read the RFID tags are as followed:

Scanner	Tag	Distance from Top	Distance from Side
AD4797240380	4900AE0CD13A	4 cm	2.5 cm
AD4797240380	4900ADF56C70	4.5 cm	2.1 cm
AD4797240380	4900ADEB656A	4.4 cm	3.0 cm
AD4797240380	4900ADE9F7FA	4.2 cm	2.5 cm

AD4797210218	4900AE0CD13A	4.4 cm	1.7 cm
AD4797210218	4900ADF56C70	4.5 cm	2.0 cm
AD4797210218	4900ADEB656A	4.5 cm	1.6 cm
AD4797210218	4900ADE9F7FA	4.2 cm	1.8 cm

RFID Tags Detection Continuation (December 15, 2015)

Scanner	Tag	Distance from Top	Distance from Side
AD4797240380	4900AE0CD13A	4.4 cm	3.2 cm
AD4797240380	4900ADF56C70	4.5 cm	3.8 cm
AD4797240380	4900ADEB656A	4..5 cm	3.6 cm
AD4797240380	4900ADE9F7FA	4.6 cm	3.8 cm
AD4797240380	4900AE0CD13A	4.5 cm	3.5 cm
AD4797240380	4900ADF56C70	4.7 cm	3.8 cm
AD4797240380	4900ADEB656A	4.5 cm	3.4 cm
AD4797240380	4900ADE9F7FA	4.4 cm	3.6 cm
AD4797240380	4900AE0CD13A	4.5 cm	3.3 cm
AD4797240380	4900ADF56C70	4.5 cm	3.9 cm
AD4797240380	4900ADEB656A	4.5 cm	3.5 cm
AD4797240380	4900ADE9F7FA	4.6 cm	3.7 cm
AD4797240380	4900AE0CD13A	5.8 cm	3.3 cm
AD4797240380	4900ADF56C70	6.1 cm	3.7 cm
AD4797240380	4900ADEB656A	6.2 cm	3.5 cm
AD4797240380	4900ADE9F7FA	6.6 cm	3.9 cm
AD4797240380	4900AE0CD13A	6.2 cm	3.4 cm

AD4797240380	4900ADF56C70	6.3 cm	3.7 cm
AD4797240380	4900ADEB656A	6.1 cm	3.5 cm
AD4797240380	4900ADE9F7FA	6.2 cm	3.5 cm
AD4797240380	4900AE0CD13A	5.4 cm	3.6 cm
AD4797240380	4900ADF56C70	6.5 cm	3.6 cm
AD4797240380	4900ADEB656A	6.4 cm	3.4 cm
AD4797240380	4900ADE9F7FA	7.0 cm	3.6 cm
AD4797210218	4900AE0CD13A	6.0 cm	3.4 cm
AD4797210218	4900ADF56C70	5.9 cm	3.7 cm
AD4797210218	4900ADEB656A	5.8 cm	3.7 cm
AD4797210218	4900ADE9F7FA	6.3 cm	3.8 cm
AD4797210218	4900AE0CD13A	5.4 cm	3.3 cm
AD4797210218	4900ADF56C70	6.0 cm	3.7 cm
AD4797210218	4900ADEB656A	5.8 cm	3.5 cm
AD4797210218	4900ADE9F7FA	6.0 cm	3.7 cm
AD4797210218	4900AE0CD13A	5.5 cm	3.6 cm
AD4797210218	4900ADF56C70	5.8 cm	3.6 cm
AD4797210218	4900ADEB656A	5.5 cm	3.4 cm
AD4797210218	4900ADE9F7FA	6.0 cm	3.5 cm
AD4797210218	4900AE0CD13A	6.0 cm	3.3 cm
AD4797210218	4900ADF56C70	6.0 cm	3.5 cm
AD4797210218	4900ADEB656A	5.8 cm	3.4 cm
AD4797210218	4900ADE9F7FA	6.2 cm	3.5 cm
AD4797210218	4900AE0CD13A	5.5 cm	3.4 cm

AD4797210218	4900ADF56C70	6.2 cm	3.4 cm
AD4797210218	4900ADEB656A	5.9 cm	3.7 cm
AD4797210218	4900ADE9F7FA	6.4 cm	3.7 cm
AD4797210218	4900AE0CD13A	5.5 cm	3.5 cm
AD4797210218	4900ADF56C70	5.9 cm	3.5 cm
AD4797210218	4900ADEB656A	5.3 cm	3.4 cm
AD4797210218	4900ADE9F7FA	6.5 cm	3.4 cm

CONCLUSION

These tests prove that the RFID scanner will work, and work reliably, within the margin we have given it, being the size of the jaw and the estimated distance from the base of the jaw to the PackBOT.

BOP

BOP is the main capture method being deployed in the mission to capture and immobilize the PreyBOTs. BOP is a sensitive mouse trap that is easily set off by the slightest of agitations. BOP must be able to be deployed without going off and be properly activated when tripped. BOP also needs to be properly handled so that the net and the newly captured PreyBOTs can be detached from the BOP mechanism.

BOP DRAG TEST PROCEDURE

Dragging BOP and turning improperly can set off the trip lever. BOP will be dragged in the same manner that the PackBOTs will. BOP will be dragged through a course with four 90 degree turns with BOP being pulled in tow. Failure of BOP staying set during the course

MATERIALS

- Assembled BOP
- Tape
- Engineering Notebook
- Pen

STEP BY STEP PROCEDURE

1. Tape down a course that BOP must maneuver through. The course should have as many as five 90 degree turns and multiple straight paths.
2. With BOP set, drag it through the track while increasing the speed by .5 mph every two times BOP completes the course until two mph is reached.
3. If BOP goes off then it must be adjusted by tweaking the holding mechanism so that there is more surface area covering the fly rod.
4. Continue and repeat this exercise until BOP can be dragged around the circuit two times in a row successfully at two miles per hour.

RESULTS

Trial	Speed (mph)	Success
1	0.5	Y
2	0.5	Y
3	1.0	Y
4	1.0	Y

5	1.5	Y
6	1.5	N
7	1.5	Y
8	1.5	Y
9	2.0	Y
10	2.0	Y

CONCLUSION

The course was set, and BOP was dragged through it. BOP went effortlessly through the course until it reached the speed of two mph. The original magnet used was doubled up to ensure that BOP would not go off unless by the intended to. From there, the tests continued until it reached the desired 2 miles per hour, where it still succeeded. There is little chance that BOP will trigger while the PackBOTs are dragging it onto the field. Now, we are confident that we will have sufficient traps on the field to capture all of the PreyBOTs and ensure the desired target is captured.

BOP NET TEST

The net must stay on during the entirety of the mission until the pullcord of the net is acted upon by a force, the PackBOT. The acceleration of the net when the trap is set off may cause the net to be removed which was observed during testing. Necessary reinforcement, via magnets, will be added until the net is secure enough to function all the way through the mission.

To ensure a PreyBOT will be captured by the net a string will be pulled to set off BOP to see if the magnetic strips hold.

MATERIALS

- Assembled BOP
- String
- Engineering Notebook
- Pen
- RC Car (resembling a PreyBot)

STEP BY STEP PROCEDURE

1. Set BOP
2. Place the Car inside of the capture area of BOP.
3. Using a string set off the BOP mechanism.
4. Observe the net as BOP goes off and the action completes.
5. If the net has fallen off, reset the net and add extra support by placing more magnetic straps on it.
6. If the net hasn't fallen off, reset the net and repeat the procedure 5 times to ensure the net is secure.

RESULTS

Trial	# of Springs	Success	Problem
1	2	N	Net completely removed
2	2	Y	-
3	2	N	Left side off
4	2	N	Right side off
5	2	N	Front off
6	2	Y	-
7	2	Y	-
8	2	Y	-
9	2	Y	-
10	2	Y	-

CONCLUSION

The net was originally attached to BOP using 20 magnet linings on each side of BOP. The failure in the trial resulted in adding four more magnets to each side of BOP. This however did not stop the net from flying off more times. An extra 2 magnets were added to each side of BOP besides the back. When the front fell off of the frame in the last failed trial, a new method of attaching the magnets was used instead of adding more magnets. By wrapping the magnets around the frame it made it so more distance would have to be covered for the magnets to remove from the frame. After applying this new method, the net passed the trials without problem.

BOP PULLCORD TEST

BOP has now captured a PreyBOT that must brought to the corral. PackBOT must be able to identify the the trap has been set off, approach the BOP, and grab the pull string. The pull string will remove the net from its frame and close it so that the PreyBOT will be stuck inside. This however is a problem as the net may not always wrap around the PreyBOT. This problem must be solved.

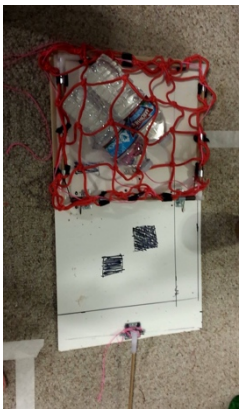
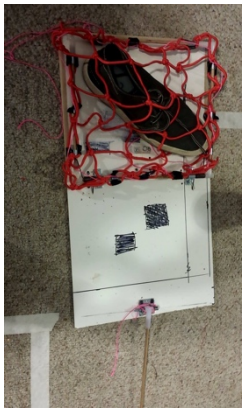
MATERIAL

- Assembled Net
- RC car (PreyBOT)
- Engineering Notebook
- Pen

STEP BY STEP PROCEDURE

1. BOP will be in the unset position with a PreyBOT inside. The PreyBOT will be multiple sizes ranging from max size all the way to minimum size.
2. Using your hands, pull the pull cord without bracing against the BOP.
3. As the net closes, observe the net surrounding the body.
4. If the Net does not fully enclose to the point that the PreyBOT can get out then it must be adjusted.
5. Observe any problems and calibrate as needed.
6. Repeat the process until the net properly closes and detaches from BOP five times successfully.

(Since a max size PreyBOT was hard to find. A variety of objects were used as subjects., Two full water bottles were chosen to be used as it simulated the size and the weight of a PreyBOT the best)


RESULTS

Trial	Success	Problem
1	N	Snagged on magnetic holder
2	N	Caught on BOT
3	Y	-
4	Y	-
5	Y	-
6	Y	-
7	Y	-

CONCLUSION

The first two trials failed. Trial one, the net snagged on the magnetic holder. To solve this from happening again the magnetic holder was carved down and sanded so that the net would more easily maneuver past it. The second trial the net got caught on the PreyBOT itself. The problem was that the pull cord is slightly raised in comparison to the PreyBOT which is on the ground. The solution was to weigh the net so that when it was removed from its frame, it could more easily be grounded when being pulled. After these two fixes, BOP's pullcord worked effortlessly.

BOP TRIP TESTS

A PreyBOT must be able to drive onto the capture platform of a BOP at any angle and set it off. If the PreyBOT sets BOP off, then it must be captured. 100+ tests will be performed driving a max size PreyBOT onto the platform from all around. The necessary adjustments must be done based off of the observation of what is wrong and should only be adjusted after every ten tests. BOP trip tests are done last as they are the final calibrations that should be done to BOP.

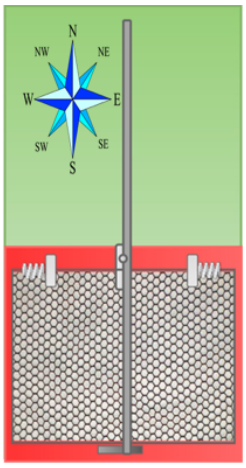
MATERIALS

- Assembled net
- Tennis Ball
- Engineering Notebook
- Pen

PROCEDURE

1. Gather equipment
2. Set BOP to its set position
3. Roll a tennis ball into the trap at a momentum consistent with that of a PreyBOTs. (A PreyBOTs max mass is one kilogram. Assuming the weight will be .75 kilograms and is traveling at 1 mph at .44 m/s, meaning the PreyBOTS momentum would be .11 KG-m/s. The tennis ball in use is .06 kilograms so it must be rolled at a speed of 1.8 m/s to match the momentum of the supposed PreyBOT).
4. Observe the mechanism to make sure that the PreyBOT is enclosed in the trap.
5. If not, make adjustments and repeat.
6. Repeat with multiple speeds and hit the trip mechanism at different angles.
7. Make adjustments accordingly and repeat until there is a 100% capture consistency in a ten trial set.

RESULTS



Trial	Direction	Success	Changes
1	N	N	Horn like tabs were added
2	N	Y	-
3	NW	Y	-
4	NW	Y	-
5	W	Y	-
6	W	Y	-
7	SW	Y	-
8	SW	Y	-
9	SE	Y	-
10	SE	Y	-
11	E	Y	-
12	E	Y	-
13	NE	Y	-
14	NE	Y	-

CONCLUSION

BOP captured the ball successfully. The ball, however, managed to slip out of the net, but there is no plausible way for the PreyBOTs to be as small as the tennis ball. The net will also be swapped out to a tighter woven and lighter version later on. When the ball was pushed straight north onto the platform the trap actually did not go off due to the fly rod needing to be pushed sideways off the magnet to trip. By adding a cuplike mechanism at the front of the fly rod it will ensure a sideways force is applied even when hit straight on from the north side.

MOTHERGOOSE

During the mission, the PackBOTs main goal is to capture a specific PreyBOT and return it to a holding cell. The PreyBOTs will most likely have an erratic movement pattern, possibly causing problems for the PackBOTs during hunting. When the PreyBOTs are captured, they may possibly struggle to be released as well. To combat these problems, a trailer, codenamed Mother Goose, will be deployed as a tool to aid capture. Mother Goose is a system designed to easily transport PreyBOTs. Mother Goose is similar to BOP in function, as in it just stays in one area and waits for a PreyBOT to wander inside. Then, it raises the ramp to enclose the PreyBOT, and scans it with an internal RFID reader. Mother Goose is on wheels to easily travel from one side of the field to the other.

ULTRASONIC RANGE FINDER TEST

The main purpose of Mother Goose is to immobilize the PreyBOTs on the field by creating a confined space. To accomplish this task, Mother Goose will need to detect whether or not something is inside of it before putting up the fourth wall/ramp using an Ultrasonic Range Finder. To test this, the following processes are used to determine if the Ultrasonic Range Finder will be able to complete its task.

MATERIALS

- Assembled Mother Goose
- Ultrasonic Range Sensor
- Computer

- Engineering Notebook
- Pen
- PreyBOT prototype

PROCEDURE

1. Place a material sample in front of the Ultrasonic Range Finder at a fixed distance.
2. From the readings from the Command Center, determine whether the Ultrasonic Range Finder responds/ activates to the material sample.
3. Annotate your findings in scientific notebook.
4. Repeat above steps for the remaining material samples.
5. Set Mother Goose to have its ramp open.
6. With the PreyBOT moving, have it pass through the Ultrasonic Range Finder inside Mother Goose at different speeds.
7. Determine if the Ultrasonic Range Finder responds/ activates to each speed by closing the ramp.
8. Annotate your findings in scientific notebook.

GATE TEST

When a PreyBOT enters Mother Goose, whether from entering from its own accord or from force, the ultrasonic range finder within Mother Goose will detect the PreyBOT. Upon detection, Mother Goose will turn its servo to close its ramp, creating a fourth wall to enclose the PreyBOT. When captured, Mother Goose will broadcast a signal to the two PackBOTs to scan

the captured PreyBOT. If the captured PreyBOT is found to be incorrect, then the PackBOT will send a command to open the ramp and capture another target. The process can be repeated until the correct PreyBOT is captured.

MATERIALS

- Mother Goose
- Servo
- Spring scale
- Engineering Notebook
- Pen

PROCEDURE

1. Turn on the servo.
2. Hook a spring scale to the end of the ramp.
3. Grab the other end of the spring scale and pull, pulling the motor.
4. Keep pulling and read the measurement as the servo begins to rotate.
5. Annotate in the scientific notebook.

BILL OF MATERIALS

ELECTRICAL TEAM

Vendor	Item Number	Description	Part Number	QTY	Unit Price	Total Price
Amazon	1	Distribution Board	QAV250	3	$20.00	$60.00
Amazon	2	Electromagnet	B000701BVK	1	$15.00	$15.00
Amazon	3	16 Gauge Fuse Holder	0400ATCFH16-5	3	$7.00	$21.00
Amazon	4	Rat Trap	M201	25	$2.50	$75.00
Amazon	5	Nylon netting	KNN154	5	$10.00	$50.00
Amazon	6	Neodymium Magnets	M1414CYLSmCo	10	$11.38	$113.80
Amazon	7	Steel Ball Bearings	B004PX9KO0	40	$0.00	$0.00
BatteryJunction	8	Tenergy 18650 6600mAh 7.4VIon Battery	TENERGY-74-6600-PCB	1	$59.80	$59.80
SparkFun	9	1/4 Watt Resistors 20 pack	10969	2	$9.00	$18.00
SparkFun	10	5v Li PO battery 2000mAH	8483	1	$10.00	$10.00
Amazon	11	Ac/Dc converter	Adams rite 4603	3	$11.00	$33.00
Polulu	12	Stepup voltage regulator 12v	u3v12f12	1	$3.95	$3.95
Polulu	13	3.3V voltage regulator	s7v8f3	4	$5.95	$23.80
Amazon	14	Banana Plug Male and Female connector	M2F300	4	$5.99	$23.96
Amazon	15	Jump cables BreadBoard	BB4002	1	$5.51	$5.51
Polulu	16	5V Voltage regulator	s9v8f2	5	$4.25	$21.25
Polulu	17	12V Step-Up Voltage Regulator	s7v845	1	$3.95	$3.95
Amazon	18	5 pc Bread board Set	5PCBB	1	$5	$5

Vendor	Item Number	Description	Part Number	QTY	Unit Price	Total Price
Amazon	19	Macro Usb Cords	Vo8P8HAQ	4	$5	$20
Amazon	20	Micro Usb Cords 2 pack	1096737	2	$7	$14
Amazon	21	2200 mAH 7.4v Battery	s7v8f3	1	$12	$12
Amazon	22	Voltage steppers	6474034561	1	$80	$80
Amazon	23	6 Color Wire Kit	19452	1	$17	$17
Amazon	24	200mm Wire Dupont Cable	p480341	1	$3	$3
Amazon	25	2.54mm Dupont Kit Pinn Connector	151787391313	1	$15	$15
Amazon	26	UV Sensor Breakout	12705	1	$12.95	17.08
Amazon	27	9250 accelerometer	ACC9250	3	$37	$37
Amazon	28	Wifi Dongle Adapter	KT-RPWF	4	$7.99	$31.96
						$859.16

MANUFACTURING

Vendor	Item Number	Description	Part Number	QTY	Unit Price	Total Price
Plastics	1	Plexiglass sheet (0.125" thick by 12" by 24")	PLEXIGLASS	5	$6.22	$31.10
Lowes	2	Brass Hinges	4222351	8	$2.00	$16.00
Amazon	3	Metal Extension Washers	MW4536-SW	12	$1	$12
Amazon	4	Wheels + Axel	39002+5001	1	$10	$10
Amazon	5	Wire Coils - 16 Gauge	B008XKU0VS	2	$6.00	$12.00
Amazon	6	Wire Coils - 18 Gauge	B00XKHS3QS	1	$6.00	$6.00
Home Depot	7	1/16 inch plywood	161640	10	$3.00	$30.00
Home Depot	8	1/4" x 48" Solid White or Black Fiberglass Rod	39800	3	$5.50	$16.50
						133.60

PROGRAMMING

Vendor	Item Number	Description	Part Number	QTY	Unit Price	Total Price
element14	1	Raspberry Pi 2, Model B, 1 GB + MicroSD	RASPBERRYPI-2-MODB-1GB	4	$40.00	$160.00
Arduino Store	2	Arduino Uno, Rev 3	a000066	4	$24.95	$99.80
SparkFun	3	SparkFun RFID Reader ID-12LA (125 kHz)	11827	2	$29.95	$59.90
SparkFun	4	SparkFun UV Sensor Breakout - ML8511	ML8511	2	$12.95	$25.90
Adafruit	5	Adafruit 10-DOF IMU Breakout	1604	4	$29.95	$119.80
Amazon	6	SainSmart HC-SR04	B004U8TOE6	1	$9.98	$9.98
SparkFun	7	SparkFun RFID USB Reader	9963	2	$24.95	$49.90
Amazon	8	AmazonBasics 4-Port USB 2.0 Ultra-Mini Hub	HU2K44N3	3	$6.99	$20.97
SparkFun	9	SparkFun 9 Degrees of Freedom Breakout - MPU-9150	11486	4	$34.95	$139.80
SparkFun	10	Arduino-compatible Microcontroller SparkFun RedBoard	25K44N3	1	$19.95	$19.95
EBay	11	Mini Stylish USB 2.0 5M	HS-225MG	6	$2.45	$14.70
Servo City	12	Machine Screws, 2.6 x 8mm	HS-705MG	100	$0.25	$27.25
SparkFun	13	SparkFun RFID USB Reader	13090	1	$24.95	$24.95

Vendor	Item Number	Description	Part Number	QTY	Unit Price	Total Price
Amazon	14	Mini Computer ODROID-C1+	LP-437	1	$38.95	$38.95
Servo City	15	Ultrasonic Range Finder LV-MaxSonar-EZ1	VkvMoPmrTIU	1	$25.95	$25.95
SparkFun	16	Inertial Measurment Unit MPU-9150	9150	1	$34.95	$34.95
						$904.71

MECHCANICAL

Vendor	Item Number	Description	Part Number	QTY	Unit Price	Total Price
DFRobot	1	DFRobot 270 Degree 15KG Metal Geared Standard Servo	409557	67	$15.25	$1,021.66
AdaFruit	2	Micro servo	MM3713	1	$5.36	$5.84
Pololu	3	Mini Maestro 18-Channel USB Servo Controller	MS1354	4	$39.95	$159.80
AdaFruit	4	Sub-micro Servo - SG51R	2201	18	$5.95	$107.10
						$1,294.40

LYCANTHROPE

Electrical Team	$859.16
Manufacturing Team	133.60
Programming Team	$904.71
Mechanical Team	$1,294.40
Shipping	$100.00
Total	**$3,090.81**

FINANCIAL SUMMARY

Eclipse Technologies initially instituted a budget of $4,500.00 for the Lycanthrope Project of the CARPA Initiative 2016. Eclipse Technologies' current total cost of the budget is $3,100.00. This change is due to the fact that the bill of materials is constantly changed to adapt to the changes of the mission throughout the year. The current expenditures are $1,516.00. Eclipse Technologies' has used 26.90 % of the expenditures on electrical team, 40.60% of the expenditures were used for parts for the mechanical team, 4.20% of the expenditures were used on materials for the manufacturing team, and 28.30% of the expenditures on were also used for parts for the programming team. Parts that were bought varied because some parts were essential for testing before the proof of concepts presentation and other parts were used for the building of the walking prototype. Eclipse Technologies' financial team has gathered funds from a various local companies, newspapers, and magazines. Eclipse Technologies has currently received $2,900 from the various companies in which we reached out to. Currently Eclipse Technologies has a remaining $1,400 in the initial budget and is planning on reaching out to other companies as well, to receive more funds to be able to continue the current building of robots.

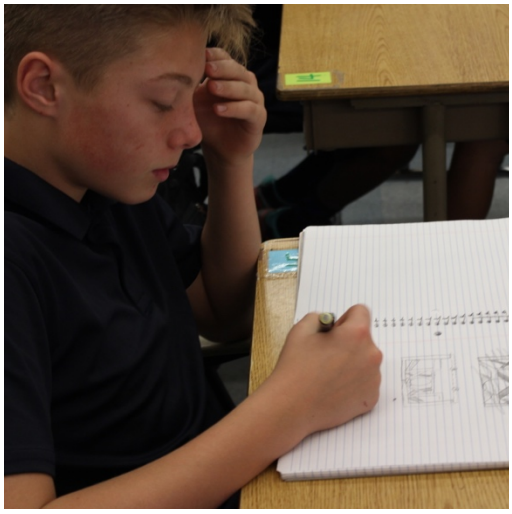
Initial Budget:	$4,500.00
Current Bill of Materials:	$3,090.81
Current Expenditures:	$1,516.00
Fundraised:	$2,900.00

OUTREACH

Eclipse Technologies has 5 core values, one being Inspire All. Staying true to this core value, one of our plans as a company is to implement an outreach program to teach middle-school students about the STEM fields. Eclipse Technologies believes that it is necessary to inspire younger generations to consider the STEM field as a career pathway because they are the future of society. Everyone at Eclipse Technologies has been inspired by someone in STEM at some point in their lives and we intend to be that inspiration for the next generation. We will partner with various middle schools in our community to provide an engaging and eye-opening experience to their students. Our outreach events include teaching them the basics of engineering and then giving them a hands-on experience applying these principles.

Eclipse Technologies' first endeavor began at Dodson Middle School in the Los Angeles Unified School District. Eclipse Technologies visited Dodson to show the robotics class how they could continue their interest in STEM throughout high school. Our presentation began

with a brief explanation of the CARPA Initiative and the Lycanthrope Project. Then, the team took the students through the engineering design process, and gave them an opportunity to apply what they had just learned to an engineering challenge of their own.



The mini mission that Eclipse Technologies invented consists of designing and constructing a structure that will be able to shelter an egg during a simulated earthquake. However, the students are constrained to using only straws, popsicle sticks, and tape. Following the design process, the students must go through phases of brainstorming, designing, and selecting an approach before they are given the materials to begin constructing their designs. Outreach events conclude with a showcase of each group's design, and a simulation to put their structures to the test.

Eclipse Technologies believes that outreach is important not only to promote interest in STEM, but especially to inspire younger students whose shoes we were in just a few years ago. It isn't enough to tell these students about the benefits of STEM and why they should pursue a career in engineering; Eclipse Technologies wants to give these students an opportunity to gain first-hand experience and see for themselves how valuable STEM truly is.

INDEX

TESTING HARDWARE SPECIFICATIONS

[Windows]
OS: Windows 10 64-bit
Processor: i7-5700 HQ
Clock Rate: 2.7 Ghz base, running at 3.5 Ghz (automatic turbo control allowed)
Number of Processors: 4 cores, 8 logical processors
RAM: 16 GB (2x 8 GB) at 1600 Mhz
Python Version: 3.5.0 (MSVC v.1900, 32bit)
OpenCV Version: 3.1 Release (+SSE2, +SSE3, +Contrib, +Python 3 Bindings)

[ODROID]
OS: Debian 8.2, "Jessie"
Processor: ARM Cortex-A5
Clock Rate: 1.5 Ghz
Number of Processors: 4 cores, 4 logical processors
RAM: 1 GB (2x 512 MB) at 792 Mhz
Python Version: 3.5.0 (GCC v4.9, 32bit)
OpenCV Version: 3.1 Release (+NEON, +Contrib, +Python 3 Bindings)

WORKS CITED

"Background Subtraction." *OpenCV*. N.p., n.d. Web. 28 Dec. 2015.

Bolme, Dav, J. Ross Beveridge, Bruce A. Draper, and Yui Man Lui. "Visual Object Tracking Using Adaptive Correlation Filters." *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*(2010): n. pag. *Colorado State University*. Computer Science Department. Web. 20 Dec. 2015.

"CIELAB - Color Models - Technical Guides." *CIELAB - Color Models - Technical Guides*. Adobe, n.d. Web. 28 Jan. 2016.

"Color Difference." *Wikipedia*. Wikimedia Foundation, n.d. Web. 28 Jan. 2016.

"Color Picker and Converter (RGB HSL HSB/HSV CMYK HEX LAB)." *Colorizer*. N.p., n.d. Web. 28 Dec. 2015.

Danelljan, Martin, Gustav Häger, Fahad Shahbaz Khan, and Michael Felsberg. "Integrated Object Models for Robust Visual Tracking." *Robust Vision for Vision-Based Control of Motion* (2009): n. pag. *Li.u*. Linköpings Universitet. Web. 22 Dec. 2015.

"Dlib C++ Library: Dlib 18.6 Released: Make Your Own Object Detector!" *Blogger*. N.p., 3 Feb. 2014. Web. 28 Dec. 2015.

Hinterstoisser, S., C. Cagniart, S. Ilic, P. Sturm, N. Navab, P. Fua, and V. Lepetit. "Gradient Response Maps for Real-Time Detection of Textureless Objects." *IEEE Transactions on Pattern Analysis and Machine Intelligence IEEE Trans. Pattern Anal. Mach. Intell.* 34.5 (2012): 876-88. Technische Universitat Munchen. Web. 23 Dec. 2015.

"K-Means Clustering in OpenCV." *OpenCV*. N.p., 10 Nov. 2014. Web. 28 Dec. 2015.

"Piotr's Matlab Toolbox." *Piotr's Matlab Toolbox*. N.p., n.d. Web. 23 Dec. 2015.

Steger, Carsten. *OCCLUSION, CLUTTER, AND ILLUMINATION INVARIANT OBJECT RECOGNITION* (n.d.): n. pag. *International Society for Photogrammetry and Remote Sensing*. MVTec Software GmbH. Web. 20 Dec. 2015.

Vacavant, Antoine, and Andrews Sobral. *Research Gate*. Computer Vision and Image Understanding, n.d. Web. 23 Dec. 2015.

Viola, Paul. *The Viola/Jones Face Detector* (n.d.): n. pag. *University of British Columbia* . Department of Computer Science. Web. 18 Dec. 2015.

"The Visual Object Tracking VOT2014 Challenge Results." (n.d.): n. pag. *EPICS*. University of Paderborn, Germany. Web. 20 Dec. 2015.

CARPA APPROVAL

Rules, conditions, and requirements are subject to change by the CARPA authority. All changes will be discussed in class, and be presented to the company teams in writing. Changes are negotiable and deadline extensions must be agreed upon by both teams.

Project Manager	Date
Deputy Project Manager	Date
Chief Scientist	Date
Systems Engineer	Date
Chief Programmer	Date
Chief Financial Officer	Date
CARPA Authority	Date