Design of the Life-ring Drone Delivery System (LDDS) for Rip Current Rescue

Andrew Hardy
Mohammed Rajeh
Lahari Venuthurupalli
Gang Xiang

Sponsor:
CATSR - Lance Sherry

Support:
Expert Drones - Brett Velicovich, Brian Yi
# TABLE OF CONTENTS

1.0 Context Analysis ........................................................................................................6
  1.1 Beach Analysis ........................................................................................................6
  1.2 Beach Rescues and Fatalities ..............................................................................6
    1.2.1 Beach Rescues ..............................................................................................7
    1.2.2 Beach Fatalities ...........................................................................................9
  1.3 Rip Currents ........................................................................................................11
  1.4 Beach Layout .......................................................................................................12
  1.4 Drowning Process ...............................................................................................14
  1.5 Lifeguard Rescue Process ..................................................................................14
  1.6 Escape Methods ..................................................................................................15

2.0 Stakeholder Analysis ............................................................................................16
  2.1 Liability Issues ....................................................................................................17
  2.2 Stakeholder Tensions ..........................................................................................17
  2.3 Win-Win .............................................................................................................18

3.0 Problem and Need ...............................................................................................19
  3.1 Performance Gap ................................................................................................19
  3.2 Problem Statement ..............................................................................................19
  3.3 Need Statement ..................................................................................................19
  3.4 System Scope ......................................................................................................20
  3.5 Proposed Design Solution ..................................................................................20

4.0 Requirements .........................................................................................................21
  4.1 Project Requirement ............................................................................................21
  4.2 Mission Requirement ..........................................................................................21
  4.3 Functional Requirements ...................................................................................21
  4.4 Usability Requirements .......................................................................................22
  4.5 Availability Requirements ..................................................................................22
7.4.4 Position/Velocity to Euler Angle Error ................................................................. 50
7.4.5 Main Simulation Model ......................................................................................... 51

8.0 Results and Recommendations ............................................................................. 52
8.1 Floatation Device AHP Results .............................................................................. 52
8.1 Case Study ............................................................................................................... 54
8.2 Design of Experiment 1 - Best Location/Path/Operation Range ............................... 55
8.3 Design of Experiment 2 - Best Battery .................................................................... 58
8.3 Design of Experiment 3 - LDDS rescues compared with lifeguard rescue ............... 62
  8.3.1 Cost Model ......................................................................................................... 64
  8.3.2 Benefit Versus Cost ........................................................................................... 67

9.0 Business Case .......................................................................................................... 68
  9.1 Competitors ......................................................................................................... 68
  9.2 Revenue and Business Case .................................................................................. 69

10.0 Compare LDDS with Jet Skis ............................................................................... 71
  10.1 Cost Model of Jet Ski ......................................................................................... 72
  10.2 Break Even Analysis between Jet Skis and the LDDS ........................................... 73
  10.3 Results ................................................................................................................. 74

11.0 Combining Rescue Alternatives ............................................................................ 77

12.0 Sensitivity Analysis ............................................................................................... 78
  12.1 Distance Between Towers .................................................................................. 78
  12.2 Pilot Delay and Path .......................................................................................... 79
  12.3 Jet Ski Speed Equivalence .................................................................................. 81

13.0 Statement of Work ............................................................................................... 82
  13.2 Schedule/Milestones ......................................................................................... 82
  13.3 Payment Schedule ............................................................................................. 83
  13.4 Work Breakdown ............................................................................................... 83
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>13.4.2 Critical Tasks</td>
<td>84</td>
</tr>
<tr>
<td>13.4.3 Schedule</td>
<td>85</td>
</tr>
<tr>
<td>13.4.4 SPI, CPI, SV, and CV</td>
<td>88</td>
</tr>
<tr>
<td>13.4.5 Earned Value</td>
<td>89</td>
</tr>
<tr>
<td>Appendix</td>
<td>91</td>
</tr>
<tr>
<td>Appendix A: Final Octocopter Parameters</td>
<td>91</td>
</tr>
<tr>
<td>Appendix B: Matlab Modules</td>
<td>92</td>
</tr>
<tr>
<td>Victim Model</td>
<td>92</td>
</tr>
<tr>
<td>Lifeguard Model</td>
<td>94</td>
</tr>
<tr>
<td>Linear Dynamics</td>
<td>95</td>
</tr>
<tr>
<td>Rotational Dynamics</td>
<td>99</td>
</tr>
<tr>
<td>Main Model</td>
<td>105</td>
</tr>
<tr>
<td>Design of Experiments</td>
<td>107</td>
</tr>
<tr>
<td>Sensitivity Analysis</td>
<td>126</td>
</tr>
<tr>
<td>References</td>
<td>131</td>
</tr>
</tbody>
</table>
1.0 CONTEXT ANALYSIS

1.1 BEACH ANALYSIS

In the United States, approximately 42% of the adult population visits the beach every year [1]. Beach goers visit the beach for activities such as swimming, surfing, scuba, and attending attractions. While on beaches, they pay for beach services which provide revenue for local businesses and the beach owners. There are approximately 6,200 beaches in the United States [2]. These beaches are a profitable business for the U.S.; they generate more than $320 billion annual revenue [3]. The costs to manage and maintain a beach in the United States are less than 4% of $2.65 billion annual park service budgets. For example, beach patrol at Ocean City, Maryland operates on a budget $2.3 million [4]. On any sizable beach, there are lifeguards that keep the beaches safe for beach goers.

1.2 BEACH RESCUES AND FATALITIES

There are different incidents that lifeguards respond to every year. Some incidents can lead to fatalities. The following statistical data on rescues and fatalities are from the USLA between 2003 and 2013 [5]. That data includes rip currents rescues and total rescues. That data include surf zone fatalities total, rip current drowning deaths and total drowning deaths. For rip current drowning deaths and total drowning deaths, the situation when there was a lifeguard and when there was not a lifeguard is considered.

Please note, some beach agencies only report totals for fatalities and rescues, but not the specific causes of rescues. For example, just because a graph says there are only two rip current rescues that year, does not mean there were only two rip current rescues that year for all agencies. It means only two rescues from agencies that do report that subcategory. These graphs are to reveal the general trends and minimum numbers regarding rescues and fatalities.
Rip currents are the primary cause of rescue in the United States since they account for nearly 330,000 or nearly 80%, of rescues between 2003 and 2012 [6]. The rescues from surf, swiftwater, and scuba were 18%, 0%, and 1% respectively.
Figure 3a and 3b shows the amount of rip current rescues from 2003 to 2013, according to the USLA [5]. Figure 3a shows the total rescues for all causes between 2003 and 2013. The total includes rip currents with swift, surf and scuba rescues. By looking at the trend line, total rescues are increasing over the years by an increasing rate of about 1160 people per year. When comparing rip current rescues with the total rescues, rip current rescues are almost half of the total rescues. For example, in 2011 the rescues from rip currents were 32,867 while the total rescues were 63,909 people rescued. Remember that some agencies report totals and not rescues with specific causes, so the 32,867 is an estimate of the minimum number of rip current rescues that year.
Figure 3 shows the fatalities in 2014 in the United States according to the National Oceanic and Atmospheric Administration (NOAA) [7]. Rip current drowning accounts for 79% of all beach fatalities, more than the other causes combined. In addition, according to NOAA, the 10-year average of annual rip current fatalities is 51 people per year [8]. However, other reporting agencies report different numbers. For example, the United States Life Saving Association (USLA) believes rip current fatalities exceed 100 people every year [9].
Figure 4a and 4b. Unguarded deaths according to the USLA

Figure 4a and 4b shows unguarded (no lifeguards present) deaths between 2003 and 2013, and unguarded drowning deaths from rip currents from 2003 to 2013 [5].

Figure 5a and 5b. Guarded deaths according to the USLA

Figure 5a and 5b show guarded drowning deaths in total and for rip currents according to the USLA from 2003 to 2013 [5].
1.3 RIP CURRENTS

Rip currents are strong powerful currents of water that flow away from shore. They form when waves break near the shoreline, meaning it can occur at nearly every beach. Beaches usually have multiple rip currents spread across the shoreline. Beach goers who swim in the waters have to be careful of rip currents. Agencies like the USLA and NOAA are actively spreading knowledge about the dangers of rip currents [10].

When waves break near the shoreline, they generate feeder currents that move along shore [11]. Once this feeder current is deflected offshore, it forms the rip current. This rip current has a neck and a head area. The neck is the area where the rip current’s speed and strength is highest. Most drowning deaths happen in the neck area of the rip current. The head of the rip current is the area where the rip current’s speed and strength starts to weaken.
Shown in Figure 6, rip currents can also form around jetties. Additionally, when waves travel through sandbars, the water level increases and this is also form rip currents. As the water level increases, the pressure increases and this would form faster and stronger rip currents. Some rip currents last for many days or months, while others some last hours or days. The table below shows the ranges for certain rip current properties [12].

<table>
<thead>
<tr>
<th>Rip Current Measurements</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Width</td>
<td>[10,200] feet</td>
</tr>
<tr>
<td>Length</td>
<td>~[100,1000] feet</td>
</tr>
<tr>
<td>Speed</td>
<td>[1,8] feet/second</td>
</tr>
</tbody>
</table>

When the rip current’s speed is below four feet per second, it is not considered to be dangerous for strong swimmers, but it is dangerous for the weak swimmers [9]. However, if the speed exceeds four feet per second, this is considered to be dangerous for even the strongest of swimmers.

### 1.4 BEACH LAYOUT

Beach lifeguards have usually three classes of buildings, consisting of the headquarters, sub-quarters and towers. The headquarters is where the electricity and main offices are located. There is typically one headquarters for the beach. The sub quarters are where most lifeguards start their day to take attendance; there are typically a few of them spread throughout the length of the beach. The towers are what the lifeguards sit on to monitor the safety of their specific section. There is a designated swimming area of the beach. It extends out from the shoreline about 150 yards. The orange dotted line in the figure below indicates the end of the area. The beach is divided into multiple sections with one tower per section. The figure below shows 3 beach sections and their corresponding towers.

The distance between towers varies from beach to beach. 68 beaches report the average distance between their lifeguard towers to USLA [5]. 5 of the beaches have an average
distance between the towers that is greater than 1000 yards, 5 more beaches report an average distance of between 500 and 1000 yards. 16 beaches report an average distance between lifeguard towers of 100 yards or less. The mean of this average distance is 352.6 yards, the standard deviation is 500.6 yards. The histogram below summarizes the average distance between towers. The bins each contain a range of 100 yards (i.e. 0-100, 101-200, etc.)

![Histogram of Avg. Distance Between Towers](image)

**Figure 9. Distance between towers**

These same beaches also report their average number of towers that are staffed. This is split between weekdays and weekends. The mean for weekdays is 15.7 and standard deviation is 15.8. The mean for weekends is 17.6 and standard deviation is 17.3. The histograms below summarize the data.

![Histogram of Avg. Weekday Towers Staffed](image)

**Figure 10a and 10b. Towers Staffed per Day**
Drowning is a very common and deadly experience. After the victim is caught in a rip current, their first reaction might be to fight against the current. The main reason for a victim to drown is either because they are exhausted, dehydrated, or they do not know how to swim. Hence, they start to panic. The body accumulates carbon dioxide which leads to dry-drowning. Dry drowning is when the water reaches the airway (when the victim coughs or ends up swallowing water), this causes the throat spasms and the lungs start to seal. The water then starts to accumulate in the stomach. Dry drowning leads to large amounts of panic and resultant rapid movements. When the victim becomes unconscious they enter secondary drowning, or pulmonary edema, where the throat relaxes and water flows into the lung. The buildup of carbon dioxide leads the body to automatically attempt to breathe which only worsens the situation. With the resultant accumulation of carbon dioxide, the victim has only minutes before death. This process is detailed in the figure below.

![Rip Current Drowning Diagram](image)

**Figure 11. Rip Current Drowning. The Total Time Varies Person to Person**

**1.5 LIFEGUARD RESCUE PROCESS**

A lifeguard notices a drowning victim when the victim is panicking. When a drowning victim is spotted in the section, the lifeguard will radio the control room, whistle at the adjacent lifeguards to cover their sections, and point in the direction of the victim [13]. The lifeguard then starts running towards the shoreline with either their right hand up in the air, which means assistance is needed, or pointing and tapping on top of their head, which means assistance is not needed. The lifeguard swims to the victim, rescues them, guides them back to the shoreline, and then drags them out of the water. Medical care is provided as needed afterwards. This entire lifeguard rescue process takes a max of 4 minutes [14]. This process is shown in the diagram depicted below.
1.6 ESCAPE METHODS

There are three escape methods – fighting the current, floating with the current, and swimming perpendicular to the rip current until they are out of it. The recommended method is to swim perpendicular to rip current then parallel to the rip current toward the shore. However, most victims float because they are unable to swim. Floating is recommended if the victim is unable to swim perpendicular to current due to reasons such as avoiding exhaustion. Fighting against the current is not recommended as it leads to exhaustion quickly without gaining any distance towards shore. The rip current is sometimes stronger than the victim, which leads to the victim getting exhausted while still being pulled away from shore.

Distributions for survival time was approximated with the help of lifeguard interviews [13] and Pia’s video analysis on drowning [15][16]. The time a victim can survive when they are unable to keep a float is between 20 to 60 seconds (3rd to 6th stage in Figure 11). Rip current victims usually still have the strength to swim when they are first caught by the rip current, thus assume 20 to 60 seconds more of the ability to swim (First two stages in the victim drowning process in Figure 11). Assuming this total of 40 to 120 seconds is a normal distribution that captures 95% of victims, the normal distribution for survival when fighting the current is made in Figure 13. The survival time of victims floating and swimming parallel were assumed to be 50 and 100 seconds longer respectively.
2.0 STAKEHOLDER ANALYSIS

There are six main stakeholders: 1) lifeguarding associations, 2) lifeguards, 3) beach goers, 4) manufacturers, 5) municipalities and 6) businesses.

1. Lifeguarding Associations

Lifeguarding associations are professional lifesaving associations that train beach lifeguards and open water rescuers. These associations certify lifeguards and make sure they are up to date with these four requirements: 1) yearly licensing, 2) four hours of in-service training, 3) monthly training meetings and, 4) red ball test. If the individual meets all these requirements each year they will be certified as a lifeguard.

2. Lifeguards

Lifeguards are strong swimmers who supervise the safety and rescue of swimmers, surfers, and other water sports participants. Three certified lifeguards were interviewed to help with research on rip currents, rescue process, and liability issues [13][14][17].

3. Beach Goers

Beach goers are people who go to the beach and use its services. They want the least restrictions on their activities, thus instructions given by lifeguards are of no concern. However, they want the lifeguards to be 100% effective at their jobs. In other words, the beach goers want the lifeguards to be always alert, well-trained, and to save everyone in time.

4. Manufacturers

Manufacturers are companies that produce equipment for the lifeguards. Some major companies include Swimoutlet and Marine Rescue Products. A lifeguarding association can also manufacture rescue products. For an example, the Jeff Ellis & Associates, an international
lifeguard training company, builds their own products such as a rescue board, a lifeguard buoy, ring buoys (life rings) and life jackets.

5. Municipalities

Municipality is broken down into owner and operator. The owner can be the county, city, or state government that owns the beach property. The operator documents the lifeguards current status of certification and requirements met. They are the ones who pay the lifeguarding associations, certified agencies (like Jeff Ellis), and lifeguards. Both the owner and operator are usually the same. However there are some operator businesses that concentrate on the documentation and financial aspect of lifeguarding that owners may hire.

6. Businesses

The businesses are where the municipalities get money to provide safe and clean beaches for the beach goers. Some of these businesses include hotels, restaurants, and different stalls. Providing the best services and attractions are the municipalities’ main goal.

2.1 LIABILITY ISSUES

Between these five stakeholders there are liability issues. Municipalities that own beaches can be broken down into the owner and operator. The operator holds the lifeguard certification history, updates documentation, and monitors the financial status. The lifeguard associations are certified lifeguard agencies. These associations train lifeguards who are hired by the owner to supervise the safety of the beach goers.

Lifeguards and certification agencies are always liable if a drowning victim sues a lifeguard for injuries and fatalities during the rescue process. However, if the lifeguards use only the designated equipment the agencies provided and the agencies have documents on the lifeguard's training and license, there is a very low chance of a successful lawsuit. Most cases end in settlements out of court [13].

2.2 STAKEHOLDER TENSIONS

Figure 14 below is a summary of all the interactions, liabilities, and tensions among the relevant stakeholders. The lifeguards protect the beach goers in case of any emergency. If the beach goers are injured during the rescue process they can sue the lifeguards. However, at the end of the day, the municipality, specifically beach owner, is always responsible for any lawsuits against the lifeguards. The beach owner is protected under the catastrophic umbrella insurance which is an extra layer of liability protection over and above the beach property in case the owner is sued because of an accident. If the owners lose the lawsuit, the taxpayers pay that umbrella insurance.

The beach goers provide the revenue for the operator who then provides clean and safe beaches for the owner. The Lifeguarding associations train and certify professional lifeguards
who are hired by the operator. These associations also provide legal assistance for the owner, operator, and lifeguards. Also, the associations, operator, and lifeguards provide feedback of the equipment the manufactures produce. The associations are looking for the most reliable and effective equipment whereas the operator is looking for something that is affordable.

The businesses provide services (hotels, restaurants, kiosks and shops) to the beachgoers, in return the businesses get money and the taxes they pay to the municipalities provide clean beaches for the goers. Lifeguards provide safe beaches and the major negative tension is that the beachgoers do not obey the lifeguards’ rules and regulations. Rip currents are silent and hard to spot, thus beachgoers would not believe the warnings, rules, and regulations from lifeguards about rip currents. Another major tension is the competition between municipalities for hiring the best lifeguards and providing safest environment and attractions.

Figure 14. Stakeholder Tension Diagram

2.3 WIN-WIN

Assume a solution exists that would decrease the amount of rip current fatalities. Since there is less fatalities, beaches with this solution would be considered safer, thus more beachgoers would come, which is a win for municipalities, businesses, and for beachgoers. The solution could be equipped with any reasonably weighted flotation device, so it is unlikely any
lifeguarding agencies need to buy equipment from a different manufacturer. Thus, manufacturers should not be affected by the system. There is no stakeholder that is against our Life-Ring Drone Delivery System. All stakeholders will receive some benefits with this successful system.

3.0 PROBLEM AND NEED

3.1 PERFORMANCE GAP

![Graphical Representation of the time gap](image)

Figure 15. Graphical Representation of the time gap

There is a gap between victim survival time and lifeguard rescue time. A lifeguard can reach a victim in a maximum of 92 seconds; however some victims have survival times as low as 60 seconds [13]. Shown in the figure, victims who can survive more than 92 seconds can survive long enough to be rescued (shaded green under the curve). Victims that cannot survive until the lifeguard reaches them are in risk of drowning (shaded in red). 60 seconds marks the estimated survival time for a weak swimmer attempting to fight against the rip current.

3.2 PROBLEM STATEMENT

Rip currents are the cause of 81% of annual beach rescues, 79% of annual beach fatalities, and cause an average of 51 deaths per year [6][8][18]. Lifeguards are very good at their job as they have a 95 to 100% chance of successful rescue, and can reach a victim caught in a rip current in a max of 93 seconds [13]. However some victims cannot survive this long, as some have survival times as low as 60 seconds.

3.3 NEED STATEMENT

There is a need for a system that can reach and assist at least 99% of victims in order to increase the victim's survival time. By increasing the speed of delivery of the flotation device, drowning deaths can be decreased.
3.4 SYSTEM SCOPE

In order to meet this need, a drone delivery system was designed. Design choices on the drone platform, battery, flotation device, location/path/operation range were made for the best possible drone system for rescue. A tether holding and releasing system concept was made to allow the drone to keep the flotation device close to the victim. A drone system design, business case, cost model, and case study for a specific beach was developed.

3.5 PROPOSED DESIGN SOLUTION

The proposed design to delivery flotation devices is a delivery drone. It will incorporate a separately designed flotation device holder and tether system. There will be 2 cameras mounted to the drone, one which will be forward facing and provide the pilot with a video feed of the direction the drone is flying. The other will be downward facing and provide the pilot with a view over the drowning victim to allow for accurate delivery. The pilot controller will be an off the shelf commercial controller. The drone will be provided with eight interchangeable batteries for the drone. The buyers of the LDDS drone will provide the flotation device. Many beaches would already have their own flotation devices and can equip the drone with one that fits their needs.

The tether system is the key to the drone system. The tether line is attached to both the drone and the flotation device. This allows for the drone to maintain control of the flotation device in the water to ensure that the device remains within the victim's grasp and that the victim has sufficient time to grasp it. Once the victim has secured the flotation device, the tether line can be detached from the drone to allow the drone to fly back to shore to restock on another flotation device.

![Figure 16. Model of the Drone Delivery System](image)
4.0 REQUIREMENTS

4.1 PROJECT REQUIREMENT

The overall goal is to save lives, more lives than lifeguards can currently save.

P.1. The system shall reduce the annual number of rip current deaths by a minimum of 95%.

4.2 MISSION REQUIREMENT

As has been stated, the goal of our system is to increase the number of victims saved.

M.1. The system shall increase the percent of saved victims to at least 99%.

4.3 FUNCTIONAL REQUIREMENTS

The following requirements are derived from discussions with lifeguards, from FAA regulations, and our own engineering judgement.

F.1. The system shall get a flotation device to a victim faster than a human lifeguard can on average 90% of the time.

   F.1.1. The system shall utilize a drone to deliver a lifesaving device to the victim before the lifeguard reaches them during the rescue process.

      F.1.1.1. The drone shall be able to maintain its altitude.

         F.1.1.1.1. The system shall not fly above 400 feet.

         F.1.1.1.2. The system shall be able to fly at minimum altitude of 10 feet.

      F.1.1.2. The system shall be able to hover within a horizontal distance of 3 ft from the target.

      F.1.1.3. The system shall be operable within 1 mile of the home point.

      F.1.1.4. The system shall be able to hold payloads of 5 pounds or under.

      F.1.1.5. The system shall have a minimum max flight speed of 12.5 m/s.

F.1.2. The system shall have a tethering device that can hold and deploy the floatation device.

   F.1.2.1. The floatation device shall be able to fit underneath the drone.

      F.1.2.1.1. The system shall support a standard SOLAS certified lifebuoy.

      F.1.2.2. The system shall be able to have the floatation device restocked.

      F.1.2.3. The system shall have a tether line that attaches the floatation device to the drone.
F.1.2.3.1. The system shall be able to release the tether line.

F.1.2.4. The system shall capture and transmit video data.

4.4 USABILITY REQUIREMENTS

Ensure the system would be usable in the process it is intended to supplement and by the people who will be involved in its operation. Requirements were derived by stakeholder interviews and FAA regulations.

U.1. The system shall be used by licensed pilot.
    U.1.2. The system shall be usable by a person that has less than 12 hours of training.

U.2. The system shall use a commercial controller.
    U.2.1. The system shall provide a video feed output device to pilot.

U.3. The Flotation device shall be usable by the drowning victim.
    U.3.1. The flotation device shall be deemed effective by lifeguards.

4.5 AVAILABILITY REQUIREMENTS

A.1. The system shall be available to at least any beach on U.S territory.
    A1.1. The system shall comply with all federal drone regulations (FAR).

A.2. The system shall be available for rescues over 95% of the time.

A.3. The system shall be usable in minimum 2 rescues a day.

A.4. The system shall have a minimum lifetime of 2 years.

4.6 DESIGN REQUIREMENTS

D.1. The drone must weigh less than 55 pounds.

D.2. The system shall have a flotation device holder and a mount for the camera and tether line mechanism.
    D.2.1. The flotation device holder shall have a mount for a camera that faces downward.
    D.2.2. The flotation device holder release mechanism that is activated from the pilot’s control device.

D.3. The tether line shall attach to the tether line release mechanism and the flotation device.
    D.3.1. The system shall have the tether line release mechanism that is activated from pilots control device.
D.3.2. The tether line release mechanism shall have a device to attach to the tether line to the drone.

4.7 FAA REGULATIONS

Currently there are the FAA advisories for drone operation, what they refer to as small unmanned aircraft systems (UAS). The FAA limits the max altitude the drone can fly at to 400 ft and the max weight of the drone to 55 pounds. The drone operator or a spotter (can only spot for one drone at a time) must maintain a visual line of sight of the drone for all times it is operating. The most important advisories are: 1) the drone controller should be a licensed aircraft pilot, and 2) the done cannot be operated directly overhead of people who are not involved in the operation of the drone. [19] [20]

The FAA has also released a set of draft rules for the operation of drones. These are similar to current advisories. It also adds other restrictions such as: 1) the drone must be flown under a max speed of 100 mph, 2) the drone must be registered and available to the FAA for inspection, 3) the pilot must inspect the drone prior to flight, 4) maintain and operate drone when there is a 3 mile visibility from control station. [20][21][22]

A Certificate of Waiver or Authorization (COA) exemption can be granted for drone usage that are for public (government) use. These public COAs involve an agreement between the agency operating the drone and the FAA to allow the loosening of some restrictions while maintaining safety and aircraft avoidance, often it's applied to operation in remote areas. The FAA lists some example uses as search and rescue, firefighting, and border patrol. [23]

Due to the COA and no concrete regulations on drones, this project will consider the situation with current advisories and the situation where some rules are loosened, by new regulation or by a COA.

4.8 SOLAS REQUIREMENTS

The US Coast Guard has requirements for the specification for lifebuoys (SOLAS). The SOLAS guidelines outline requirements such as minimum inside diameter of 400 mm & maximum outside diameter of 800 mm and the device must contain reflective material [24] [25]. The tether hold and release system is designed to hold at least a SOLAS certified flotation device, however the tether system shall also be able to hold any life ring the lifeguards want to use that is under five pounds and is reasonable in dimensions.
The lifeguard process and the drone system are independent thus the lifeguard rescue process is outside of our system boundary. Thus, the lifeguard rescue process is unaffected by the addition of the LDDS. If the lifeguard is not able to reach the victim in time, the drone will reach the victim before them. The drone process will start and simultaneously take place in the 90 seconds the lifeguard has to swim to the drowning victim. The controller receives what section the victim is and finds the exact coordinates the victim is. The controller will then command the drone to lift off. If the drone reached the victim in time, it will continue the process of dropping the flotation device near the drowning victim and releasing the tether when the victim grabs it, and then the drone returns to its home point. This will modify the victim’s survival time. However, if the lifeguard reaches the victim first, the drone has the option to return back to the homepoint.

The orange line in Figure 17 indicates when the victim is no longer in danger, meaning the drone has delivered the flotation device to the victim and the victim already has grabbed the device. Notice how all this happens before the lifeguard swims to the victim. The green line indicates when the victim is rescued by the lifeguard. By closing this gap survival time for victims can be increased, which leads to an increase in successful rescues.

Due to the nature of our drone, our system can only assist the victim when they are still conscious, in a state known as active drowning. Active drowning is when a victim is still conscious, they’re still moving and struggling, head thrown back and face upward in an attempt to keep it above water and breathe [26]. When a victim is unconscious, known as passive
drowning, (recognizable by a lack of motion and floating face down) the drone will be unable to render assistance to the drowning individual.

5.2 OPERATIONAL SCENARIO

**Precondition:** Lifeguard has identified a drowning victim. Lifeguard is prepared for rescue process. Lifeguard radios control room of the section # or victim’s general direction. Victim is located somewhere on the rip current and is attempting an escape method. Drone is ready to deploy. Flotation device is stocked on drone.

**Primary:** Stage 1
- Controller is informed by lifeguard of general area of the victim
- Controller takes off drone
  - Confirm victim location by eyesight if near tower
- The system takes off to a height of 10 meters.
- The system accelerates to 10 m/s towards the section given.
- Controller confirms specific location in that section through camera or eyesight.
  - Relative to Drone
- The system maintains 10 m/s towards the victim's location.
- Once the system is within 5 m of the victim’s location, system shall decelerate to victim’s speed and position. At the same time, the system will reduce height until the flotation device is just above the water (confirmed by controller).

**Primary:** Stage 2
- The system drops the flotation device and positions it by the victim
- Once the victim is about to grab the flotation device, system detaches the tether.
- The system maintains a 2 m hover over victim until lifeguard has reached the victim.
  - Controller uses camera to visually determine victim state (active or passive)
  - If necessary Controller inform medical personnel of victim status

**Primary:** Return
- Once lifeguard has reached the victim, or drone has been determined to be of no further use, or drone has reached critical battery charge, system will be flown back to the home point.
  - System lands on home point.

**Post-Condition:** Lifeguard is enacting the rest of rescue process starting with rescuing the victim. Drone has landed back at the home point and awaits restocking of life ring. Victim is being helped by the lifeguard.

Exceptions to the operational scenario are also considered below.

- Lifeguard reaches first
- System goes back to home
- Victim underwater but visible to drone
  - Drone continues rescue process
- Drone cannot see victim in section
  - Drone searches area
  - IF cannot find them, look for lifeguard and go in the same direction
- Victim grabs tether system or drone
  - Release tether system. Continue rescue process
- Drone runs on low battery mid-rescue
  - Drone automatically comes back to home, or continue rescue process by pilot’s discretion
- Tether system is not releasing
  - Drone hovers, lower altitude to prevent pulling too hard
- Tether system is not dropping
  - Drone keeps an eye on victim. Flies back when lifeguard reaches them
- Tether is dropped too early
  - Drone goes back home point. If there are spare devices, continue rescue process.
- Weather is too hazardous
  - Pilot chooses whether to continue or not

6.0 DESIGN DECISIONS

There are three sets of design decisions that must be considered to create the best possible delivery drone for rescue.

6.1 SET A: FLOTATION DEVICE

The best flotation device to be carried and delivered by the drone is considered. Four reasonable alternatives are listed in the table below [27][28][29]. The quantitative factors being compared are weight, dimensions, buoyancy, and cost [30]. The effectiveness and usability are subjective ratings set by a professional lifeguard. An Analytical Hierarchy Process analysis was used to determine the best flotation device. Usability is the weighting of how easy it is for the drowning person to grab and utilize the flotation device. A high usability indicates an easy to use flotation device. Effectiveness is a rating of, when the victim secured the device, how well can it keep up the victim’s survival.
<table>
<thead>
<tr>
<th>Flotation Device</th>
<th>Cost</th>
<th>Weight</th>
<th>Dimensions</th>
<th>Buoyance</th>
<th>Effectiveness (5 Star)</th>
<th>Usability (5 Star)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ring Buoy (Jimbuoy JBW-20)</td>
<td>$85.98</td>
<td>3 lbs.</td>
<td>20 in</td>
<td>16.5 lbs.</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>Rescue Can (Jimbuoy model 8t)</td>
<td>$139.99</td>
<td>4 lbs.</td>
<td>29.5x9.5 in</td>
<td>18.2 lbs.</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>Lifejacket (First Mate - Stearns flotation)</td>
<td>$74.99</td>
<td>1.5 lbs.</td>
<td>24x12x3 in</td>
<td>15.5 lbs.</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>Ultra 3000 (Auto inflating life jacket)</td>
<td>$204.99</td>
<td>3 lbs.</td>
<td>30x52 in</td>
<td>37.7 lbs.</td>
<td>3</td>
<td>3</td>
</tr>
</tbody>
</table>

Figure 18. Objective Values of the Flotation Device Alternatives

![Life Ring](image1.png) ![Rescue Can](image2.png) ![Life Vest](image3.png) ![Ultra 3000](image4.png)

Figure 19. The Flotation Devices Considered

6.2 SET B: DRONE STATION, OPERATION RANGE, & FLIGHT PATH

The best location for the drone homepoint, best path for the drone to fly, and the best operation range for the drone to cover are considered.

There are two locations considered to be the homepoint of the drone. The first is at the lifeguard towers. Stationing it here might allow the pilot to see the drowning victim. It provides easy access to the lifeguards who are monitoring for drowning victims as well. The second location is on a boat out in the water. This would place the drone potentially closer to the drowning victims, and avoid overhead flight over other beach goers. However boats are scarce and must service several sections. The figure below shows the potential locations of the drone homepoint.

Operation range is the number of lifeguard sections the drone will help rescues in. If a drone covers more sections, a smaller number of drones is used, thus reducing costs. However, covering more sections mean the rescues are more dispersed, thus drones might take longer to
reach victims that are far away from the drone homepoint. The range of 1 lifeguard section, 3 lifeguard section, and 5 lifeguard sections is considered.

There are two paths that the drone can fly to the drowning victim. The first is to fly directly towards the drowning victim. The second path would be to first fly out to sea through a designated path to avoid flying overhead of beachgoers and swimmers. This must be considered to ensure the system can meet FAA regulations. However, this path leads to a slower time for the drone to reach the victim. The figure below shows these alternatives considered.

![Figure 20: Location, Path, and Operation Range](image)

### 6.3 SET C: BATTERY

For the design of the LDDS drone system, the properties of the battery are important. The larger the battery the more battery capacity the drone will have. However, with the increase in size comes an increase in weight. An increase in weight leads to an increase in thrust required. An increase in thrust required leads to an increase in power required. More power means a larger battery, and so on. The lightest battery possible while still ensuring enough flight time and power is necessary.

![Figure 21: Positive Feedback Between Weight, Thrust, and Power](image)
7.0 METHOD OF ANALYSIS

7.1 SIMULATION GOAL

In order to determine the best design decisions and to verify that the system meets the mission requirement of increasing the percent of victims rescued, a simulation was developed.

7.2 EQUATIONS

The equations used by the simulation is detailed in this section. The equations are based off of Gibansky’s work [31] and the work of an 2015 senior design team that also used equations for drones [32].

Drones, or Unmanned Aerial Vehicles, have an even number of motors (four, six, or eight). Half the motors spin counterclockwise and the other half spin clockwise. By having motors that spin in different directions, the drone can control rotation about its axes. For example, the figure below shows the top view of the drone. Clockwise rotation is stronger, thus the drone body must be rotating counterclockwise.

![Figure 22. How rotation works on drones](image)

There are two important frames of reference considered. the drone's reference points (body-frame) and the controller's reference point (inertia-frame).
Motor spin direction is defined as in Figure 3. The simulation will focus on hexacopters and octocopters. The front side is in the direction of positive X-axis. The left side is the direction of the Y-axis. The top side of the drone is the positive Z-axis, which is pointing out of the page in Figure 3.

The position of the drone is defined as X-Y-Z coordinates. The orientation of the drone will be in the Euler angles of φ-θ-ψ, or roll-pitch-yaw. There is three axis of rotations. They can rotate around their X-axis (roll), the Y-axis (pitch), and the Z-axis (yaw) as shown in

In order to transform body-frame forces into inertia-frame forces, transformation matrices are used. There are respective transforms for φ-θ-ψ (roll-pitch-yaw) shown as (1), (2), and (3). The important equation is the product of the three rotational matrices in order to a full transform using the three values of φ-θ-ψ (4). The following four transformation matrices convert body-frame forces to inertia-frame forces.
Additionally a transform matrix to convert derivatives of φ-θ-ψ to inertia-frame angular velocities is shown below.

\[
\begin{bmatrix}
\omega_x \\
\omega_y \\
\omega_z
\end{bmatrix} = \begin{bmatrix}
1 & 0 & -\sin \theta \\
0 & \cos \phi & \sin \phi \cos \theta \\
0 & -\sin \phi & \cos \phi \cos \theta
\end{bmatrix}
\begin{bmatrix}
\dot{\phi} \\
\dot{\theta} \\
\dot{\psi}
\end{bmatrix} 
\]  

(5)

### 7.2.1 POWER

To determine power, assume that the propeller motor torque is proportional to the current difference. Shown below:

\[
\tau = K_t (I - I_0) 
\]

(6)

Where \( \tau \) is motor torque, \( I \) is the current, \( I_0 \) is the no-load current when the motor is not spinning, and \( K_t \) as the proportionality constant. Assume \( I_0 \) is negligible. Assume that the voltage difference is the sum of the voltage drop across internal resistance and a term that involves the angular velocity of the propellers because the faster the propeller spins, the more voltage it uses.

\[
V = IR_m + K_v \omega 
\]

(7)

Where \( V \) is voltage, \( I \) is the current, \( R_m \) is the internal resistance, \( K_v \) is the proportionality constant, and \( \omega \) is the angular velocity of the motor's propeller. For the simulation, assume \( R_m \) is
negligible. Also note that motor manufacturers list $K_v$ as the inverse of the $K_v$ in (7). Voltage and current are used in the two equations before (6) (7), thus the power equation is found.

$$P \approx \frac{K_v}{K_t} \tau \omega$$  \hspace{1cm} (8)

Now assume that motor torque is proportional to the motor's thrust, thus,

$$\tau = K_t T$$  \hspace{1cm} (9)

$$P \approx \frac{K_v K_t}{K_t} T \omega$$  \hspace{1cm} (10)

Where $\tau$ is the motor torque, $K_t$ is proportionality constant, $T$ is thrust, $P$ is power generated, $K_v$ and $K_t$ are proportionality constants. The table below summarizes all the proportionality constants stated previously.

<table>
<thead>
<tr>
<th>Variable 1</th>
<th>Relation</th>
<th>Variable 2</th>
<th>Proportionality Coefficient</th>
<th>Equation</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\tau$</td>
<td>$\propto$</td>
<td>$I$</td>
<td>$K_t$</td>
<td>$\tau = K_t \ast I$</td>
</tr>
<tr>
<td>$\tau$</td>
<td>$\propto$</td>
<td>$T$</td>
<td>$K_t$</td>
<td>$\tau = K_t \ast T$</td>
</tr>
<tr>
<td>$V$</td>
<td>$\propto$</td>
<td>$\omega$</td>
<td>$K_v$</td>
<td>$V = K_v \ast \omega$</td>
</tr>
</tbody>
</table>

### 7.2.2 THRUST

The thrust equation used is shown below.

$$T = \left[ \frac{\pi}{2} D^2 \rho P^2 \right]^{\frac{1}{3}}$$  \hspace{1cm} (11)

Where $T$ is thrust, $D$ is diameter of the propeller, $\rho$ is the air density, and $P$ is power. Substituting (10) into the power term in (11), the new thrust equation is listed below.

$$T = \frac{\pi}{2} \rho D^2 \left( \frac{K_v K_t}{K_t} \right) \omega^2$$  \hspace{1cm} (12)
Where $K_v$, $K_\tau$, and $K_t$ are proportionality constants. In the body frame, the drone only provides thrust upwards. Moving is possible when the drone is tilted in the inertia frame. Thus, the total thrust in the body-frame the summation of thrusts due to the motors.

$$\begin{align*}
T_{body} &= k * \left[ \begin{array}{c}
0 \\
0 \\
\sum \omega_i^2
\end{array} \right] \\
k &= \frac{\pi}{2} \rho D^2 \left( \frac{K_v K_\tau}{K_t} \right)
\end{align*}$$

(13)

(14)

Where $\omega_i$ is the ith motor.

### 7.2.3 DRAG MODEL

The following drag equation is used.

$$F_D = \frac{1}{2} C_D \rho v^2 A$$

(15)

Where $F_D$ is force due to drag, $C_D$ is the coefficient of drag of the drone, $\rho$ is the air density, $v$ is the velocity of the drone, and $A$ is the cross sectional area of the drone. Since there are three surfaces normal to the X-Y-Z axes, there are three components to the force of drag.

$$F_D = \frac{1}{2} \rho \left[ \begin{array}{ccc}
C_{Dx} A x & 0 & 0 \\
0 & C_{Dy} A y & 0 \\
0 & 0 & C_{Dz} A z
\end{array} \right] \begin{bmatrix} v_x^2 \\ v_y^2 \\ v_z^2 \end{bmatrix}$$

(16)

### 7.2.4 LIFE RING MODEL

The force of the life ring is decomposed into the life ring’s weight force and drag force, which also uses the same drag equation (16).
7.2.5 MAIN FORCE MODEL

With the force of gravity, force of thrust (13), force of drag(16), and force due to the life ring, net force is found from the following.

\[ ma = \sum F_i \]  

(17)

Where \( m \) is the mass of the drone system, \( a \) is the acceleration, and \( F_i \) is the various forces acting on the drone. Expanding on this equation shows the four forces considered in the simulation.

\[
\begin{bmatrix}
\ddot{x} \\
\ddot{y} \\
\ddot{z}
\end{bmatrix} =
\begin{bmatrix}
0 \\
0 \\
-mg
\end{bmatrix} + RT_{body} + F_D + F_{lifering}
\]

(18)

Where \( x,y,z \) is the coordinate position of the drone in the inertia frame, \( g \) is the gravitational constant (9.81 m/s/s), \( R \) is the transformational matrix (4), \( T_{body} \) is the thrust in body-frame, \( F_D \) is the force of drag (16), and \( F_{lifering} \) is the force due to the life ring. A free body diagram illustrates these forces.
Motors provide torque through drag force from air or by a thrust difference between opposite pairs of motors.

\[
\tau_D = \frac{1}{2} r \rho C_{DP} A(r)^2 (\omega)^2 \tag{19}
\]

\[
b = \frac{1}{2} r \rho C_{DP} A(r)^2 \tag{20}
\]

\[
\tau_z = b \omega^2 + I_z \dot{\omega} \tag{21}
\]

Where \(\tau_D\) is the propellers torque due to drag on air, \(C_{DP}\) is the propellers coefficient of drag, \(A\) is the cross-sectional area of the propeller, \(\tau_z\) is the torque around the body Z-axis due to one motor, \(I_z\) is the propellers moment of inertia about its Z-axis, and \(r\) is the radius of the propeller. Counterclockwise rotation denotes a positive torque about the body-frame Z-axis. Assume \(\dot{\omega}\), angular acceleration of the propeller, is negligible.

For roll torque, the torque is equal to the thrust force multiplied by distance. The distance is the distance from the motor to the center of mass. Assume the drone is symmetrical, thus the distance is the same for all motors. When rolling, drones usually increase the thrust of one motor while decreasing the other motor in the pair in order to maintain the same body-frame thrust while also creating a torque due to the difference of thrust. Thus, the torque from one motor minus the torque from the opposite motor creates the body’s rolling torque. The pitch torque is similar to the roll torque.

For yaw torque, add up all the motor’s torques (21). The following torque equations are for the hexacopter and octocopter respectively,

\[
\tau_B = \begin{bmatrix}
\tau_{\text{roll}} \\
\tau_{\text{pitch}} \\
\tau_{\text{yaw}}
\end{bmatrix} = \begin{bmatrix}
Lk(\omega_1^2 - \omega_4^2) \\
Lk(\omega_2^2 + \omega_3^2 - \omega_5^2 - \omega_6^2) \\
b(\omega_1^2 - \omega_2^2 + \omega_3^2 - \omega_4^2 + \omega_5^2 - \omega_6^2)
\end{bmatrix} \tag{22}
\]

\[
\tau_B = \begin{bmatrix}
\tau_{\text{roll}} \\
\tau_{\text{pitch}} \\
\tau_{\text{yaw}}
\end{bmatrix} = \begin{bmatrix}
Lk(\omega_1^2 + \omega_2^2 - \omega_5^2 - \omega_6^2) \\
Lk(\omega_3^2 + \omega_4^2 - \omega_7^2 - \omega_8^2) \\
b(\omega_1^2 - \omega_2^2 + \omega_3^2 - \omega_4^2 + \omega_5^2 - \omega_6^2 + \omega_7^2 - \omega_8^2)
\end{bmatrix} \tag{23}
\]

Where \(L\) is the distance from the propeller to the drone’s center of mass, \(b\) is a coefficient defined by (21), and \(k\) is a coefficient defined by (13). Torque is then used with the following equation to get angular acceleration.
\[ \tau = I\dot{\omega} + \omega \times (I\omega) \quad (24) \]

Where \( \tau \) is the drone’s torque about the 3 axes, \( I \) is moment of inertia, and \( \omega \) is the angular velocity of the drone. This equation does not directly give us roll, pitch, or yaw, thus it is expanded. Since the drone is symmetric, the moment of inertia about the X-Y-Z axes are independent of each other. The equation (26) was solved for angular acceleration.

\[
I = \begin{bmatrix} I_{XX} & 0 & 0 \\ 0 & I_{YY} & 0 \\ 0 & 0 & I_{ZZ} \end{bmatrix} \quad (25)
\]

\[
\dot{\omega} = \begin{bmatrix} \dot{\omega}_x \\ \dot{\omega}_y \\ \dot{\omega}_z \end{bmatrix} = I^{-1}(\tau - \omega \times (I\omega)) \quad (26)
\]

\[
\dot{\omega} = \begin{bmatrix} \tau_\phi I_{xx}^{-1} \\ \tau_\theta I_{yy}^{-1} \\ \tau_\psi I_{zz}^{-1} \end{bmatrix} - \begin{bmatrix} I_{yy} - I_{zz} & \omega_y & \omega_z \\ I_{xx} & I_{xx} - I_{yy} & \omega_x & \omega_z \\ I_{yy} - I_{xx} & \omega_x & \omega_y \end{bmatrix} \quad (27)
\]

Where \( \tau_\phi, \tau_\theta, \) and \( \tau_\psi \) are the torques in their respective rotations (22)(23), \( I_{xx}, I_{yy}, I_{zz} \) is the moment of inertia about their respective axes, and \( \omega_x, \omega_y, \omega_z \) are the angular velocities about their respective axes. Once angular accelerations are found, the transformation matrix (5) to get the \( \phi-\theta-\psi \) derivatives can be used. Integrating the derivatives give values of roll-pitch-yaw.
7.3 CAD MODEL

The DJI S900 and S1000+ are the models of our hexacopter and octocopter. To determine the drag coefficient and moment of inertia of the drones, a 3D model of each drone was made with Autodesk inventor. Drone owners provided some measures of the system components [33][34][35][36].

The major components of the drone were constructed as parts:

1. The top of the body for the S900 and the S1000+

![Figure 24. Hexacopter and Octocopter Base Respectively](image)

2. The arms of both the S900 and the S1000+

![Figure 25. Arms for the Hexacopter and Octocopter Respectively](image)

The S1000+ arm is a bit longer. The rest of the parts for both drones are modeled as using the same parts.

3. The motor of the drones and its rotor

![Figure 26. Motor and Propeller Blade Model](image)
4. The lower body of the drone

Figure 27. Lower Body of the Drone

5. One of the landing gear

Figure 28. Landing Gear when Tether System not Attached

6. The battery and tray were each modeled as individual parts.

Figure 29. Battery and Battery Holder
7. A mocked up tether system was built as a few parts and the life ring as 1 part.

From here there were for models of the drone that were created as an assembly from the above parts.

7.3.1 MOMENT OF INERTIA

From these assemblies, the drones' iProperties are examined to find the moment of inertia as well as the drones' mass. The picture below shows an example of what the iProperties window shows for the S900 with a ring buoy attached, showing both the principle and global moment of inertias.
This table summarizes the moment of inertias found by the models.

<table>
<thead>
<tr>
<th>Description</th>
<th>I1</th>
<th>I2</th>
<th>I3</th>
</tr>
</thead>
<tbody>
<tr>
<td>S900 w/o Device</td>
<td>0.295 kg m^2</td>
<td>0.290 kg m^2</td>
<td>0.522 kg m^2</td>
</tr>
<tr>
<td>S1000+ w/o Device</td>
<td>0.450 kg m^2</td>
<td>0.445 kg m^2</td>
<td>0.825 kg m^2</td>
</tr>
</tbody>
</table>
7.3.2 DRAG

To find the Drag, flow design was opened up as an environment in inventor. When the environment was loaded the simulation was set at 200% resolution and the wind speed was adjusted as needed in the simulation settings. The wind tunnel was sized so that there were about 2 lengths of the drone in front of the drone and 4 lengths behind the drone. The wind tunnels width and height were both set to the same length of about 2 times the size of the drone. This is the recommended setup as stated in the flow design tutorials. [37] Once the wind tunnel was established, the simulation was let to run. When the status of the simulation reached “stabilized” the value of drag was recorded from the drag plot. The following graph shows an example of the drag plot.

Many tests with different drone configurations and simulation settings were done to gain an understanding of how they affected the drag. The table below shows the numbers that resulted from those tests. Wind speed was found to have a negligible effect on drag. The difference between both the horizontal components was also negligible. The vertical drag was larger than the horizontal drag.

<table>
<thead>
<tr>
<th>Drone</th>
<th>Wind Direction</th>
<th>wind</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>5 m/s</td>
</tr>
<tr>
<td>S900 with buoy</td>
<td>X (horizontal)</td>
<td>Coefficient of Drag</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Force of Drag</td>
</tr>
<tr>
<td></td>
<td>Z (horizontal)</td>
<td>Coefficient of Drag</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Force of Drag</td>
</tr>
<tr>
<td>-30 degree rotation x</td>
<td>Coefficient of Drag</td>
<td>0.61</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Force of Drag</td>
</tr>
<tr>
<td></td>
<td>Coefficient of Drag</td>
<td></td>
</tr>
<tr>
<td>----------------</td>
<td>---------------------</td>
<td>-------</td>
</tr>
<tr>
<td><strong>Y (vertical)</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(top to bottom)</td>
<td>Coefficient of Drag</td>
<td>1</td>
</tr>
<tr>
<td>Force of Drag</td>
<td>31.2 N</td>
<td>233.458 N</td>
</tr>
<tr>
<td><strong>Y (vertical)</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(bottom to top)</td>
<td>Coefficient of Drag</td>
<td>1.05</td>
</tr>
<tr>
<td>Force of Drag</td>
<td>33 N</td>
<td>197 N</td>
</tr>
<tr>
<td><strong>X (horizontal)</strong></td>
<td>Coefficient of Drag</td>
<td>0.863</td>
</tr>
<tr>
<td>Force of Drag</td>
<td>1.033 N</td>
<td>4.050 N</td>
</tr>
<tr>
<td><strong>Z (horizontal)</strong></td>
<td>Coefficient of Drag</td>
<td>0.707</td>
</tr>
<tr>
<td>Force of Drag</td>
<td>.962 N</td>
<td>3.870 N</td>
</tr>
<tr>
<td><strong>S900 with NO buoy</strong></td>
<td>Coefficient of Drag</td>
<td>0.6</td>
</tr>
<tr>
<td>-30 degree rotation x</td>
<td>Force of Drag</td>
<td>1.476 N</td>
</tr>
<tr>
<td><strong>Y (vertical)</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(top to bottom)</td>
<td>Coefficient of Drag</td>
<td>0.885</td>
</tr>
<tr>
<td>Force of Drag</td>
<td>2.653 N</td>
<td>64.998 N</td>
</tr>
<tr>
<td><strong>Y (vertical)</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(bottom to top)</td>
<td>Coefficient of Drag</td>
<td>1.06</td>
</tr>
<tr>
<td>Force of Drag</td>
<td>3.267 N</td>
<td>19.728 N</td>
</tr>
<tr>
<td><strong>S1000+ with NO buoy</strong></td>
<td>Coefficient of Drag</td>
<td>0.795</td>
</tr>
<tr>
<td>Force of Drag</td>
<td>1.045 N</td>
<td>6.420 N</td>
</tr>
<tr>
<td></td>
<td>Z (horizontal)</td>
<td>Coefficient of Drag</td>
</tr>
</tbody>
</table>

The below tests done at 200% resolution. Below are the results.
<table>
<thead>
<tr>
<th>Drone</th>
<th>Wind Direction</th>
<th>Wind Speed</th>
<th>10 m/s</th>
</tr>
</thead>
<tbody>
<tr>
<td>Buoy Only</td>
<td>horizontal</td>
<td>Coefficient of Drag</td>
<td>0.324</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Force of Drag</td>
<td>1.805</td>
</tr>
<tr>
<td></td>
<td>-30 degree rotation</td>
<td>Coefficient of Drag</td>
<td>0.526</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Force of Drag</td>
<td>7.404 N</td>
</tr>
<tr>
<td></td>
<td>vertical</td>
<td>Coefficient of Drag</td>
<td>1.192</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Force of Drag</td>
<td>27.387 N</td>
</tr>
<tr>
<td>S900 with NO buoy</td>
<td>X (horizontal)</td>
<td>Coefficient of Drag</td>
<td>0.897</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Force of Drag</td>
<td>4.294 N</td>
</tr>
<tr>
<td></td>
<td>Z (horizontal)</td>
<td>Coefficient of Drag</td>
<td>0.75</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Force of Drag</td>
<td>4.083 N</td>
</tr>
<tr>
<td></td>
<td>-30 degree rotation x</td>
<td>Coefficient of Drag</td>
<td>0.6</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Force of Drag</td>
<td>5.899 N</td>
</tr>
<tr>
<td></td>
<td>Y (vertical)</td>
<td>Coefficient of Drag</td>
<td>1.082</td>
</tr>
<tr>
<td></td>
<td>(top to bottom)</td>
<td>Force of Drag</td>
<td>13.084 N</td>
</tr>
<tr>
<td></td>
<td>Y (vertical)</td>
<td>Coefficient of Drag</td>
<td>1.038</td>
</tr>
<tr>
<td></td>
<td>(bottom to top)</td>
<td>Force of Drag</td>
<td>12.438 N</td>
</tr>
<tr>
<td>S1000+ with NO buoy</td>
<td>X (horizontal)</td>
<td>Coefficient of Drag</td>
<td>0.854</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Force of Drag</td>
<td>4.490 N</td>
</tr>
</tbody>
</table>
The drag in the x direction was done with 0 degrees of rotation and not the drag at a 30 degree rotation (simulating the drone in a flying position). The largest drag values were recorded. The last table here shows the drag coefficients found.

<table>
<thead>
<tr>
<th>Device</th>
<th>Drag Coefficient</th>
</tr>
</thead>
<tbody>
<tr>
<td>S900 w/o Device</td>
<td></td>
</tr>
<tr>
<td>X: Horizontal</td>
<td>0.9</td>
</tr>
<tr>
<td>Y: Horizontal</td>
<td>0.8</td>
</tr>
<tr>
<td>Z: Vertical</td>
<td>1.1</td>
</tr>
<tr>
<td>S1000+ w/o Device</td>
<td></td>
</tr>
<tr>
<td>X: Horizontal</td>
<td>0.85</td>
</tr>
<tr>
<td>Y: Horizontal</td>
<td>0.9</td>
</tr>
<tr>
<td>Z: Vertical</td>
<td>1.2</td>
</tr>
<tr>
<td>Ring buoy</td>
<td></td>
</tr>
<tr>
<td>Horizontal</td>
<td>0.5</td>
</tr>
<tr>
<td>Vertical</td>
<td>1.2</td>
</tr>
</tbody>
</table>
7.3.3 SURFACE AREA

Surface areas normal to the body-frame X-Y-Z axes were approximated by assuming the drone components were simple shapes and objects. The life ring surface area was also approximated similarly. Results are shown in the table below.

<table>
<thead>
<tr>
<th>Area (m²)</th>
<th>S900</th>
<th>S1000</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>0.0481</td>
<td>0.0517</td>
</tr>
<tr>
<td>Y</td>
<td>0.0481</td>
<td>0.0517</td>
</tr>
<tr>
<td>Z</td>
<td>0.1001</td>
<td>0.1587</td>
</tr>
</tbody>
</table>
7.4 SIMULATION

In order to verify that the system meets our requirements (reducing fatalities, increasing the percent of victims saved), and to determine the best design choices out of our alternatives, a simulation was built in Matlab and Simulink.

The three processes simulated are:
1. Victim Model
2. Drone Dynamics Model
3. Lifeguard Model

7.4.1 VICTIM BEHAVIOR PROCESS

Victim position over time is calculated by the above block. Assume only one escape method is used, constant swim velocity, and victims are identified 10m away from shore.

The inputs are chosen escape method and rip current properties. The state variable is victim velocity and position. There are three methods of escape methods (float, swim parallel to shore, swim toward shore). Rip current properties are the length, width, velocity, and distance away from tower of the rip current. Rip current distance away from tower is used as the initial position of the victim.

The rip current's velocity and victim's velocity is summed to get the inertia frame velocity of the victim. This velocity is integrated once to get position over time. The position of the victim over time will act as a waypoint for the lifeguard and drone.
7.4.2 LIFEGUARD RESCUE PROCESS

Lifeguard position over time is calculated by the above model. Assume all velocities are constant, and lifeguards always run along shore to the base of the rip current and then jump into the rip current to chase the victim.

Inputs are the victim position and the rip current properties. Similar to the victim model, the lifeguard’s swimming velocity is the sum of the base swimming velocity and rip current velocity. Lifeguards also have a base running speed. The rescue process after the lifeguard reaches the victim is outside the scope of the simulation.
7.4.3 DRONE DYNAMICS MODEL

Drone dynamics is simulated in the following model. The input is the victim's position. The output is the drone's position. The state variables are the roll, pitch, yaw, and X-Y-Z position of the drone. The main output is the drone's position over time.

Victim position comes into the controller block, which outputs motor voltages. Voltage goes into the linear dynamics block to get out acceleration, position, and velocity of the drone. The rotational dynamics block outputs roll-pitch-yaw. The angles, position, and velocity go into a proportional-integral-derivative (PID) control which compares values to the victim's position and velocity. The errors are fed back into the controller, which adjusts the voltage. Rotational Dynamics and Linear Dynamics blocks are block representations of the equations (27) and (18) respectively.

Figure 36. Drone Dynamics Model

Figure 37. Rotational Dynamics
7.4.4 POSITION/VELOCITY TO EULER ANGLE ERROR

The controller will be simulated as GPS flight. This means that the drone autonomously hovers when there is no input, there is a maximum velocity and tilt, and that it maintains altitude while in horizontal movement. The controller is programmed to use Euler angle (roll-pitch-yaw) errors to adjust the voltage, thus a relationship between angle error and position/velocity error is used. The velocity and position error could be any value in the interval \([-\infty, \infty]\), and the wanted angle can be any value that is within the tilt limits set by the drone’s controller (assume this limit is \([-\pi/6, \pi/6]\). Thus there needs to be a relation between the velocity domain and wanted angle range. The relation used is the following.

\[
E = \frac{a(1-e^{bx})}{(1+e^{bx})}
\]

Where \(E\) is the angle error, \(a\) is the tilt angle limit in radians, \(x\) is the position or velocity error, and \(b\) is an arbitrary coefficient. The higher the value of \(b\) is, the more sensitivity angle error is to position/velocity errors.
7.4.5 MAIN SIMULATION MODEL

The previous models are used to create the main simulation model. The inputs are victim escape method (fight, float, or swim parallel) and rip current properties (length, width, velocity, and distance from tower). Both inputs are random variables. The inputs are used to generate the victim position during rescue. The victim’s position acts as a waypoint to calculate the lifeguard’s position and drone position during the rescue. The lifeguards position and drone position is inputted into the Data Analysis block, which outputs the time it took for the lifeguard and drone to reach the victim and whether the victim survive the rescue process.

A sample output of the main simulation model is shown in Figure 40. The blue dot is the victim being pulled by a rip current. The red dot is the lifeguard that is swimming after the victim. The green dot is the drone which has lifted off and is following the victim. Notice that the drone is reaching the victim faster than the lifeguard can.
### 8.0 RESULTS AND RECOMMENDATIONS

#### 8.1 FLOATATION DEVICE AHP RESULTS

The subjective attributes of usability and effectiveness were also considered. Values of the attributes and the weights were found by interviewing a lifeguard [14].

<table>
<thead>
<tr>
<th>Flotation Device</th>
<th>Weight</th>
<th>Dimensions</th>
<th>Buoyance</th>
<th>Effectiveness</th>
<th>Usability</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ring Buoy (Jimbuoy JBW-20)</td>
<td>6</td>
<td>2</td>
<td>4</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>Rescue Can (Jimbuoy model 8t)</td>
<td>8</td>
<td>6</td>
<td>6</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>Lifejacket (First Mate – Stearns flotation)</td>
<td>2</td>
<td>4</td>
<td>2</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td>Ultra 3000 (Auto inflating life jacket)</td>
<td>4</td>
<td>8</td>
<td>8</td>
<td>5</td>
<td>5</td>
</tr>
</tbody>
</table>

Importance between attributes was evaluated. Time of Delivery (which is directly related to weight) and buoyancy are the most important attributes.

<table>
<thead>
<tr>
<th></th>
<th>Buoyance</th>
<th>Time of Delivery</th>
<th>Effectiveness</th>
<th>Usability</th>
<th>Dimensions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Buoyance</td>
<td>1</td>
<td>1/2</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>Time of Delivery</td>
<td>2</td>
<td>1</td>
<td>7</td>
<td>7</td>
<td>7</td>
</tr>
<tr>
<td>Effectiveness</td>
<td>1/5</td>
<td>1/7</td>
<td>1</td>
<td>1/2</td>
<td>1</td>
</tr>
<tr>
<td>Usability</td>
<td>1/5</td>
<td>1/7</td>
<td>2</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>Dimensions</td>
<td>1/5</td>
<td>1/7</td>
<td>1</td>
<td>1/3</td>
<td>1</td>
</tr>
</tbody>
</table>

Logical Decision was used to determine the weights of the attributes and the utility of each alternative, shown in Figure 41. The ring buoy is the best alternative because it has the highest utility of 6.4 (weight is out of 10). The ring buoy is rated lowest in buoyancy however the buoyancy only matters when it hits the water surface, then the effectiveness and usability is what comes in play and is important. As you can see below, both these factors are the greatest in comparison to the other alternatives.
Figure 41. AHP Analysis Results

The five graphs below show how sensitive these alternatives are among one another. The result is very sensitivity with the measure of buoyancy, effectiveness, and usability. The Ultra 3000 is a very close alternative; however the life ring is picked because further interviews with lifeguards show high recommendations for the life ring. Additionally it was found that life jackets are difficult to put on in the water and are sometimes slippery.

Figure 42. Sensitivity Analysis on AHP Results
8.1 CASE STUDY

For further experiments and design choices, a case study was considered. Galveston Island Beach, TX was chosen due to the USLA record of 62 out of 65 successful rip current rescues in 2014 (3 fatalities) [6]. Galveston Island has 6 miles of guarded beach, an average of 400 yards between lifeguard towers (approximately 25 lifeguard towers), and 5 jet skis.

Figure 43. Galveston Island on Google Maps
8.2 DESIGN OF EXPERIMENT 1 - BEST LOCATION/PATH/OPERATION RANGE

**Objective**: Find the best location, best path, and best operation range for the LDDS system.

To determine the best location-path-range combination, each combination was put into a rip current rescue and time was recorded. The velocity of the drone was also varied in order to determine which combinations had more time flexibility (5, 7.5, 10, 12.5 m/s). A Monte Carlo simulation was run with 100 victims. The following distributions were used. Assume the drone moves at constant velocity, takes off in 10 seconds, and the boat option covers 5 lifeguard sections.

<table>
<thead>
<tr>
<th>Property</th>
<th>Distribution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rip Current Length</td>
<td>Unif(30.48 – 90.5 m)</td>
</tr>
<tr>
<td>Rip Current Speed</td>
<td>Unif(0.308 - 2.438 m/s)</td>
</tr>
<tr>
<td>Rip Current Width</td>
<td>Unif(3.048 - 60.96 m)</td>
</tr>
<tr>
<td>Victim Escape Method (0,1,2)</td>
<td>(0.4, 0.25, 0.35)</td>
</tr>
<tr>
<td>Victim Swim Speed (not float)</td>
<td>Norm(1.161 m/s, 0.267)</td>
</tr>
<tr>
<td>Boat Location</td>
<td>(150, Unif(0,-1000))</td>
</tr>
<tr>
<td>Tower Location</td>
<td>(0,0)</td>
</tr>
<tr>
<td>Max distance away from Drone</td>
<td>200, 600, 1000m</td>
</tr>
</tbody>
</table>

Results are shown below.
<table>
<thead>
<tr>
<th>Location</th>
<th>Path</th>
<th>Operation Range</th>
<th>Velocity (m/s)</th>
<th>% reached under 60 sec</th>
</tr>
</thead>
<tbody>
<tr>
<td>At tower</td>
<td>Straight</td>
<td>1-section</td>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>7.5</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>10</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>12.5</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3-section</td>
<td>5</td>
<td>0.98</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>7.5</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>10</td>
<td>0.99</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>12.5</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5-section</td>
<td>5</td>
<td>0.23</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>7.5</td>
<td>0.45</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>10</td>
<td>0.78</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>12.5</td>
<td>0.93</td>
</tr>
<tr>
<td>Around</td>
<td>1-section</td>
<td>1-section</td>
<td>5</td>
<td>0.13</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>7.5</td>
<td>0.33</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>10</td>
<td>0.63</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>12.5</td>
<td>0.88</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3-section</td>
<td>5</td>
<td>0.20</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>7.5</td>
<td>0.28</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>10</td>
<td>0.37</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>12.5</td>
<td>0.66</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5-section</td>
<td>5</td>
<td>0.10</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>7.5</td>
<td>0.22</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>10</td>
<td>0.30</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>12.5</td>
<td>0.56</td>
</tr>
<tr>
<td>On Boat</td>
<td>Straight</td>
<td>5-section</td>
<td>5</td>
<td>0.23</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>7.5</td>
<td>0.40</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>10</td>
<td>0.53</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>12.5</td>
<td>0.74</td>
</tr>
<tr>
<td></td>
<td>5-section</td>
<td>5-section</td>
<td>5</td>
<td>0.13</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>7.5</td>
<td>0.30</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>10</td>
<td>0.44</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>12.5</td>
<td>0.63</td>
</tr>
</tbody>
</table>
Figure 45. DoE2 Results

The y-axis is the percent of victims reached in less than 60 seconds. The x-axis is the location-path-range combinations (2 locations, 2 paths, 3 operation ranges). Feasible location-path-range combinations must meet the set threshold of at least 80% of victims reached in less than 60 seconds.

Each bar represents the percent reached with that constant velocity. Minimum velocity is defined as the velocity needed to reach at least 80% of the victims in time. A lower minimum velocity is better because it means there’s more flexibility in rescue time for the drone, which gives more time for the pilot to react to the rescue and more time to deploy and maneuver the flotation device to the victim. A high minimum velocity means the rescue requires more time to reach the victim due to the location or path used and a smaller window for the pilot to react and deploy the flotation device.

<table>
<thead>
<tr>
<th>Acceptable Location-Path-Range Combinations</th>
<th>Minimum Velocity (m/s)</th>
<th>% Reached under 60 seconds</th>
<th>Average Time to Reach (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tower, 1-Section, Straight Path</td>
<td>5</td>
<td>100</td>
<td>31.58</td>
</tr>
<tr>
<td>Tower, 1-Section, Around Path</td>
<td>5</td>
<td>100</td>
<td>33.21</td>
</tr>
<tr>
<td>Tower, 3-Section, Straight Path</td>
<td>12.5</td>
<td>93</td>
<td>38.31</td>
</tr>
<tr>
<td>Tower, 3-Section, Around Path</td>
<td>12.5</td>
<td>88</td>
<td>43.80</td>
</tr>
</tbody>
</table>

As a result, for the case study, the best location is at the lifeguard tower, the best path is any path, and the best operation range is 3-sections or less. The four location-path-range combinations will be used in Design of Experiment 3 to determine the best design. Sensitivity analysis on how the avoiding overhead path affects results are done in section 12.2.
Objective: Determine the best battery under the design space.

The drone’s payload weight will be varying as it performs three maneuvers, and the power required to maintain these maneuvers at steady state will be recorded. The maneuvers are: hover at 4m, constant velocity level-flight at a height of 4m and a speed of 5 m/s, and accelerating velocity level-flight at a height of 4m and an acceleration of 1 m/s². A standard from the manufacturer of a hexacopter and octocopter were used to standardize power for the right amount. DJI lists their hexacopter S900 as using 1000W to hover with a weight of 6.8kg. The S1000+ octocopter uses 1500W to hover with a weight of 9.5kg [36]. Assume steady state is 180 seconds after launch. The results are shown below.

<table>
<thead>
<tr>
<th>DoE 2 – Inputs</th>
<th>Hexacopter</th>
<th>Total Weight (kg)</th>
<th>Maneuver</th>
<th>Total Power Output (W)</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.8</td>
<td>Hover</td>
<td>422.5</td>
<td>Const. Velocity</td>
<td>422.9</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Accelerating</td>
<td>524.4</td>
</tr>
<tr>
<td>4.3</td>
<td>Hover</td>
<td>508.6</td>
<td>Const. Velocity</td>
<td>509.0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Accelerating</td>
<td>631.0</td>
</tr>
<tr>
<td>4.8</td>
<td>Hover</td>
<td>599.8</td>
<td>Const. Velocity</td>
<td>600.2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Accelerating</td>
<td>744.3</td>
</tr>
<tr>
<td>5.3</td>
<td>Hover</td>
<td>696.0</td>
<td>Const. Velocity</td>
<td>696.3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Accelerating</td>
<td>863.5</td>
</tr>
<tr>
<td>5.8</td>
<td>Hover</td>
<td>796.7</td>
<td>Const. Velocity</td>
<td>797.0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Accelerating</td>
<td>988.6</td>
</tr>
<tr>
<td>6.3</td>
<td>Hover</td>
<td>901.9</td>
<td>Const. Velocity</td>
<td>902.2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Accelerating</td>
<td>1119.1</td>
</tr>
<tr>
<td>6.8</td>
<td>Hover</td>
<td>1011.4</td>
<td>Const. Velocity</td>
<td>1011.7</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Accelerating</td>
<td>1254.9</td>
</tr>
<tr>
<td>7.3</td>
<td>Hover</td>
<td>1125.0</td>
<td>Const. Velocity</td>
<td>1125.3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Accelerating</td>
<td>1395.9</td>
</tr>
<tr>
<td>7.8</td>
<td>Hover</td>
<td>1242.5</td>
<td>Const. Velocity</td>
<td>1242.8</td>
</tr>
<tr>
<td>Octocopter</td>
<td>Total Weight (kg)</td>
<td>Maneuver</td>
<td>Total Power Output (W)</td>
<td></td>
</tr>
<tr>
<td>------------</td>
<td>------------------</td>
<td>------------</td>
<td>------------------------</td>
<td></td>
</tr>
<tr>
<td>4.9</td>
<td>Hover</td>
<td>524.3</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>524.6</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Accelerating</td>
<td>650.5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5.4</td>
<td>Hover</td>
<td>606.5</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>606.8</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Accelerating</td>
<td>752.5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5.9</td>
<td>Hover</td>
<td>692.6</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>692.9</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Accelerating</td>
<td>859.3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6.4</td>
<td>Hover</td>
<td>782.5</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>782.8</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Accelerating</td>
<td>970.8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6.9</td>
<td>Hover</td>
<td>875.9</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>876.2</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Accelerating</td>
<td>1086.7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7.4</td>
<td>Hover</td>
<td>972.7</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>973.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Accelerating</td>
<td>1206.9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7.9</td>
<td>Hover</td>
<td>1072.9</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1073.2</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Accelerating</td>
<td>1331.2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8.4</td>
<td>Hover</td>
<td>1176.4</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1176.6</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Accelerating</td>
<td>1459.5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8.9</td>
<td>Hover</td>
<td>1282.9</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1283.1</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Accelerating</td>
<td>1591.7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9.4</td>
<td>Hover</td>
<td>1392.5</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1392.7</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Accelerating</td>
<td>1727.6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9.9</td>
<td>Hover</td>
<td>1505.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1505.2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Weight</td>
<td>Octocopter</td>
<td>Hexacopter</td>
<td></td>
<td></td>
</tr>
<tr>
<td>--------</td>
<td>------------</td>
<td>------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10.4</td>
<td>Accelerating 1867.2</td>
<td>Accelerating 2010.4</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Hover 1620.4</td>
<td>Hover 1738.6</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Const. Velocity 1620.6</td>
<td>Const. Velocity 1738.8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10.9</td>
<td>Accelerating 2010.4</td>
<td>Accelerating 2157.1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Thus, in graphical form, the required power output for each maneuver is shown below.

Figure 46a and 45b. The power required for each maneuver at the following weights
Additional, a list of batteries were gathered online, and their power outputs were plotted alongside the required output for acceleration. The weights of the batteries were added to drone base weight, and the following three graphs show feasible and infeasible options of batteries.

Any batteries that are in the red zone are not feasible for the drone system due to the high total weight. An additional criteria not show in Figure 47 is that the battery shall have at least 10,000mAh of capacity in order for the drone to performance at least two rescues per battery. Shown by the graphs above, the LDDS cannot support two life rings due to the high weight of the system. The batteries that do meet the weight criteria do not fit the battery capacity criteria. The following batteries were chosen as the best among the battery alternatives found online.

<table>
<thead>
<tr>
<th>Drone + # LR</th>
<th>Battery Name</th>
<th>Cost</th>
<th>Energy Output(&gt;10000mAh)</th>
<th>Watts</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hexacopter + 1LR</td>
<td>Tattu 12000</td>
<td>248.00</td>
<td>12,000</td>
<td>5328</td>
</tr>
<tr>
<td>Octocopter + 1LR</td>
<td>Power 20000</td>
<td>347.74</td>
<td>20,000</td>
<td>11100</td>
</tr>
<tr>
<td>Octocopter + 2LR</td>
<td>NaN</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 47. Graphs showing the weight vs. power of each battery option on the LDDS system
Objective: Find the best method to rescue victims, between the LDDS system and the baseline of lifeguard rescue.

The hexacopter and octocopter is used with the location and operation range alternatives found in DoE 1 will be simulated in rip current rescues alongside a lifeguard, and their performances will be compared. The path used is straight to the victim. Sensitivity analysis on how the other path affects results is done in section 12.2. Assume no crowds, rescue starts with an already identified victim, no obstruction from waves, and a victim survives after being reached by one of the rescue methods. The five alternatives being considered are the following.

<table>
<thead>
<tr>
<th>Alternatives</th>
<th>Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lifeguard Only (Baseline)</td>
<td>Run Speed = 2.2 m/s</td>
</tr>
<tr>
<td></td>
<td>Swim Speed = 1.8 m/s</td>
</tr>
<tr>
<td>Hexacopter, Tower, 1-Section, Straight</td>
<td>Optimal Battery and Location/Path/Range</td>
</tr>
<tr>
<td>Octocopter, Tower, 1-Section, Straight</td>
<td>Optimal Battery and Location/Path/Range</td>
</tr>
<tr>
<td>Hexacopter, Tower, 3-Section, Straight</td>
<td>Optimal Battery and Location/Path/Range</td>
</tr>
<tr>
<td>Octocopter, Tower, 3-Section, Straight</td>
<td>Optimal Battery and Location/Path/Range</td>
</tr>
</tbody>
</table>

Our main simulation model will run in Monte Carlo mode with 325 random victims (five years' worth of victims at Galveston Island). Each alternative will attempt rescue. Then the time difference (survival time minus time to reach victim) will be taken in the end of each rescue.

\[ \Delta t = t_{\text{survival}} - t_{\text{reach}} \]
Results are shown below. The bolded options are the LDDS alternatives that are considered the best options over lifeguard rescue. ARENA Input Analyzer was used to test time to reach results. All distributions passed the Kolmogorov-Smirnov test with p-values greater than 0.15.

<table>
<thead>
<tr>
<th>Alternatives</th>
<th>Probability of reaching victim in time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Galveston Island, TX 2014</td>
<td>62/65 rescues = 0.954</td>
</tr>
<tr>
<td>Lifeguard Rescue</td>
<td>0.923</td>
</tr>
<tr>
<td>1-Section Hexacopter</td>
<td>0.994</td>
</tr>
<tr>
<td>1-Section Octocopter</td>
<td>1.000</td>
</tr>
<tr>
<td>3-Section Hexacopter</td>
<td>0.883</td>
</tr>
<tr>
<td>3-Section Octocopter</td>
<td>0.908</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Distribution of Time to Reach</th>
<th>Mean (sec)</th>
<th>Stdev</th>
</tr>
</thead>
<tbody>
<tr>
<td>Victim Survival</td>
<td>102.8</td>
<td>30.4</td>
</tr>
<tr>
<td>Lifeguard Rescue</td>
<td>48.3</td>
<td>29.5</td>
</tr>
<tr>
<td>1-Section Hexacopter</td>
<td>32.2</td>
<td>13.0</td>
</tr>
<tr>
<td><strong>1-Section Octocopter</strong></td>
<td><strong>30.8</strong></td>
<td><strong>10.9</strong></td>
</tr>
<tr>
<td>3-Section Hexacopter</td>
<td>62.6</td>
<td>22.0</td>
</tr>
<tr>
<td>3-Section Octocopter</td>
<td>59.1</td>
<td>20.5</td>
</tr>
</tbody>
</table>

Figure 49. Time to reach for each alternative plotted as normal distributions.
The first table shows that the simulated lifeguard rescue has an error of 3% compared with the real rescue results from 2014. The 1-section drones were able to beat lifeguard rescue in percent saved. Thus, the best operation range is 1-section. In the second table, the octocopter has a smaller average time to reach and a smaller standard deviation compared with the hexacopter, thus the octocopter is the best drone platform to use.

Thus the best LDDS design found is an octocopter with a weight of 10.2kg, holding a life ring, with a Power 20000 battery, launched near a lifeguard tower, covering 1-section per drone, and on any path.

8.3.1 COST MODEL

For the benefit vs. cost analysis, cost must be considered. The costs considered are the cost of life, cost of the LDDS, and the cost of a licensed pilot. The situation where the FAA can loosen their restrictions is considered by the case of sending a lifeguard to flight school to get a student or private license.

8.3.1.1 COST OF LIFE

The cost of life is the cost the lifeguards will see when a rescue has failed. The cost used is the cost of settlement due to lawsuits against lifeguards. Looking at seven settlement cases against lifeguards, the average is $120,000 per failed rescue, ranging between $10,000 and $400,000. There is also a probability of suing with each failed rescue as not 100% of people sue lifeguards when something goes wrong. Thus, assume a 25% chance of lawsuit for each failed rescue. The expected cost of each failed rescue is then $30,000.
8.3.1.2 DRONE COSTS

The drone cost model is shown below.

Figure 50. Cost Model of the Octocopter

<table>
<thead>
<tr>
<th>Acquisition</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>Drone Platform (S1000+)</td>
<td>$2,670</td>
</tr>
<tr>
<td>Two Cameras</td>
<td>$399</td>
</tr>
<tr>
<td>Battery (8 batteries)</td>
<td>$2,781</td>
</tr>
<tr>
<td>Battery Charger</td>
<td>$126</td>
</tr>
<tr>
<td>Location Setup</td>
<td>$500</td>
</tr>
<tr>
<td>User Training (8hr)</td>
<td>$2,392</td>
</tr>
<tr>
<td>Tether System</td>
<td>$500</td>
</tr>
<tr>
<td>Total</td>
<td>$9,369</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Operation &amp; Support (Yearly)</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>Annual Repair Cost</td>
<td>$500</td>
</tr>
<tr>
<td>Annual Battery Recharging Cost</td>
<td>$51</td>
</tr>
<tr>
<td>Tether 25ft</td>
<td>$42</td>
</tr>
<tr>
<td>Total</td>
<td>$593</td>
</tr>
</tbody>
</table>
The drone platform cost is the cost of buying the DJI S1000+ drone. The two cameras are based on the cost of a GoPro Hero 3. The assumed cost of setting up the drone is $500. The controller is expected to have zero experience with flying a drone, thus the cost of user training is considered as the per hour cost of ExpertDrone’s drone training [38]. Assumed eight hours of training with a cost of $2,392. The assumed cost of the tether system would be $500. In the end, the total base acquisition cost of the system is $9,369.

The annual repair cost of the drone is assumed at $500. This cost involves replacing the propellers of the drone, repairing motors, or any other repairs related to the drone [39]. The annual recharging cost of the eight batteries is $51 based off of Virginia’s electricity bill rate. For the tether release system, assumed twenty five feet of tether will be used every year. In the end, the total yearly base operation and support cost is $593.

### 8.3.1.3 PILOT COSTS

Every drone has a controller. Current FAA advisories recommend a licensed aircraft pilot be the controller due to their ability to read aeronautical charts. However, due to the COA and future possible regulations, the case where a lifeguard is sent to flight school to get a student or private license is considered.

<table>
<thead>
<tr>
<th>Pilot</th>
<th>Cost</th>
<th>Total Acquisition Cost</th>
<th>Total Operation &amp; Support Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>Entry Level Pilot</td>
<td>$34,000</td>
<td>$9,369</td>
<td>$34,593</td>
</tr>
<tr>
<td>Student Pilot</td>
<td>$5,000</td>
<td>$14,369</td>
<td>$593</td>
</tr>
<tr>
<td>Private Pilot</td>
<td>$10,000</td>
<td>$19,369</td>
<td>$593</td>
</tr>
</tbody>
</table>

Assume an entry-level aircraft pilot has a salary of $34,000 per year. Having an entry-level pilot would not change the total acquisition cost of the system, but the total yearly operation and support cost would increase to $34,593. In addition, when considering a student pilot, the cost to send a person to flight school is $5,000 [40]. This would increase the total acquisition cost to $14,369, but the yearly total operation and support cost would remain the same, assuming the person in flight school has a constant salary. Finally, the cost to become a private pilot is $10,000. This would increase the total acquisition cost to $19,369, but the yearly total operation and support cost would remain the same.
8.3.2 BENEFIT VERSUS COST

Benefit is defined as the utility based on the attributes of probability of successful rescue, average time to reach, and the time to reach standard deviation. However, after some analysis, the probability of successful rescue was a dominating attribute. Because the probability of successful rescue is the major attribute to compare the alternatives with, the best measure of benefit is probability of successful rescue. Thus, the benefit versus cost is shown below.

<table>
<thead>
<tr>
<th>Alternatives</th>
<th># Drones</th>
<th>Total Cost 5 year</th>
<th>Cost of Life for 5 years</th>
<th>Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>LG</td>
<td></td>
<td>0</td>
<td>750,000</td>
<td>0.923</td>
</tr>
<tr>
<td>Octo 3-S</td>
<td>8</td>
<td>1,400,000</td>
<td>900,000</td>
<td>0.908</td>
</tr>
<tr>
<td>Octo 1-S</td>
<td>25</td>
<td>4,500,000</td>
<td>0</td>
<td>1.000</td>
</tr>
</tbody>
</table>

### Student Pilot Scenarios

<table>
<thead>
<tr>
<th>Alternatives</th>
<th># Drones</th>
<th>Total Cost 5 year</th>
<th>Cost of Life for 5 years</th>
<th>Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>Octo 3-S</td>
<td>8</td>
<td>150,000</td>
<td>900,000</td>
<td>0.908</td>
</tr>
<tr>
<td>Octo 1-S</td>
<td>25</td>
<td>500,000</td>
<td>0</td>
<td>1.000</td>
</tr>
</tbody>
</table>

Assuming the case of requiring a licensed pilot, the LDDS system is very costly at $4,500,000 (more than the budget of Ocean City found in Section 1.1). The 3-section alternative has a lower probability of successful rescue compared with lifeguards and for a higher cost, thus is not a viable alternative. Considering the case of a student pilot, the LDDS 1-section not only has the highest probability of successful rescue, but also a lower cost.

In conclusion, with the FAA restriction on having licensed pilots, Galveston should remain with lifeguard rescue as the LDDS system cannot be recommended due to the high salary of licensed pilots. If the FAA were to loosen restrictions, the LDDS 1-section would be the best alternative due to its lower cost and higher performance.
Right now, 40% of Americans visit the beach each year, and lifeguards have always been in charge of keeping beach goers safe. Lifeguards perform more than 40,000 rescues per year, but the number one problem for all beaches is rip currents, which are narrow currents that pull people away from shore. USLA and NOAA statistics show that rip currents cause 80% of beach rescues, 80% of beach fatalities, and an average of 51 deaths per year.

A fatality on the beach is a cost on the beach patrols because of lawsuits. The estimated annual cost to lifeguards from failed rip current rescues amounts to three million dollars a year. This is not including the impact on the victims’ families, the lost business from people that think the beach is unsafe, and the mental burden on lifeguards.

Thus, the LDDS is introduced that will reach a victim faster than a lifeguard can, and delivery a life ring to the victims so they can survive long enough to be rescued by lifeguards. The value created is through improving beach safety, faster rescues, reducing fatalities, and being cheaper to operate and maintain than jet skis.

With about 6,000 beaches in the United States, assuming 25% are large enough to warrant the use of a drone, and assuming 5 drones per beach, the estimated market size is about 156 million dollars. By the 2nd year, we expect to capture 3 beaches as the first few customers. By the 5th year, we hope to capture 20 beaches, or 3 million dollars of the market.

There are many direct competitors of the LDDS. Jet skis are one indirect competitor. Although jet skis have a higher top speed than the LDDS, the jet ski is also harder to maneuver around breaking waves and crowds. It is also more costly to maintain due to the jet ski needing oil changes, gasoline, and repairs.

The direct competitors are the other rescue drone prototypes developed around the world. However, they have two problems. The first problem is inaccurate delivery – many of the rescue drones do not have cameras and can easily overshoot the victim’s location. The second problem is that these drones cannot control the life ring around deployment. If the drone drops the life ring, and the victim swats the life ring away or the life ring floats away, then the delivery was a failure. The LDDS is better because it is equipped with two cameras to pinpoint the victim’s location, and the LDDS has our designed tether hold and release system, which allows the LDDS to drag the life ring while it's in the water, keeping the life ring close to the victim until they can grab it.
9.2 REVENUE AND BUSINESS CASE

Our revenue stream will be through unit sales of the drone, selling at a price of $14,000 (50% overhead) per drone. The business plan is to spend the first year prototyping the drone and selling it to the first few beach patrol customers for a discount. We will monitor the performance of the drone on these beaches and gather testimonials. The documented performance and testimonials will be used to develop further sales leads at other beaches. By the end of the second year, we plan to be fully operational. Miami Beach and Ocean City have both expressed interest in the prototypes of our system.

Step 1. Sell to lead customers for a discount

Step 2. Monitor performance of the LDDS on those beaches

Step 3. Gather testimonials

Step 4. Use performance and testimonials to develop further sales leads

Figure 52. Business Plan for the LDDS Startup

With the salary of an FAA regulation specialist, salary for a sales representative for municipalities, and the cost of building the drones, the startup cost is estimated at 250,000 with an ongoing cost of $400,000. Assuming 35 drones will be sold per year, the business will break even at the end of 5 years, with a return on investment of 22.5% per year.

<table>
<thead>
<tr>
<th>Non-recurring Costs</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial Prototyping (R&amp;D, building the drones)</td>
<td>$50,000</td>
</tr>
<tr>
<td>Office and Warehouse acquisition</td>
<td>$80,000</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>$130,000</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Recurring Annual Costs</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Building maintenance, taxes, costs</td>
<td>$20,000</td>
</tr>
<tr>
<td>Price of drone parts (35 drones/year)</td>
<td>$300,000</td>
</tr>
<tr>
<td>Salary of FAA Lawyer</td>
<td>$80,000</td>
</tr>
<tr>
<td><strong>Total per Year</strong></td>
<td><strong>$400,000</strong></td>
</tr>
</tbody>
</table>
Figure 53. Break Even for the LDDS Company

Fast, Accurate, Drone Rescue
10.0 COMPARE LDDS WITH JET SKIS

The jet ski is the closest substitute to the LDDS, as both are devices that help lifeguards in the rescue process. But the LDDS is not a direct competition to jet skis as there are some situations where the jet ski is a better device to use. The jet ski and drone are different tools in the lifeguard’s tool box; they both are best in their own situations. This section will analyze and compare their performances.

<table>
<thead>
<tr>
<th>LDDS</th>
<th>Jet skis</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Pros</strong></td>
<td><strong>Pros</strong></td>
</tr>
<tr>
<td>1. Higher line of sight</td>
<td>1. Only lifeguard is needed in rescue</td>
</tr>
<tr>
<td>2. Better maneuverability</td>
<td>2. Usable in all kinds of weather</td>
</tr>
<tr>
<td>3. No oil changes, gasoline, or expensive repairs</td>
<td></td>
</tr>
<tr>
<td>4. Runs only on rechargeable batteries</td>
<td></td>
</tr>
<tr>
<td>5. Easily replaceable parts</td>
<td></td>
</tr>
<tr>
<td><strong>Cons</strong></td>
<td><strong>Cons</strong></td>
</tr>
<tr>
<td>1. Adds a person (pilot) into rescue process</td>
<td>1. Hard to maneuver around waves and crowds</td>
</tr>
<tr>
<td>2. Cannot be used in very strong winds &gt; 20 knots</td>
<td>2. Expensive to maintain and repair</td>
</tr>
<tr>
<td>3. Cannot be used in rainy weather</td>
<td></td>
</tr>
</tbody>
</table>
10.1 COST MODEL OF JET SKI

The cost model is shown below.

Figure 54. Cost Model of the Jet Ski

<table>
<thead>
<tr>
<th>Acquisition</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jet Ski</td>
<td>$9,699</td>
</tr>
<tr>
<td>Battery</td>
<td>[$60, $115]</td>
</tr>
<tr>
<td>Trailer</td>
<td>$1,038</td>
</tr>
<tr>
<td>Total</td>
<td>[$10,797, $10,852]</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Operation &amp; Support (Yearly)</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>Annual Gas</td>
<td>$4,126</td>
</tr>
<tr>
<td>Annual Maintenance</td>
<td>[$150, $200]</td>
</tr>
<tr>
<td>Annual Insurance</td>
<td>[$150, $500]</td>
</tr>
<tr>
<td>Two Years Registration</td>
<td>$32</td>
</tr>
<tr>
<td>Total</td>
<td>[$4,458, $4,858]</td>
</tr>
</tbody>
</table>
The assumed cost of the jet ski is at $9,699 [41]. The battery cost of the jet ski ranges from $60 to $115. In addition, when buying a jet ski there is a need for a trailer to help moving the jet ski from one place to another. The cost of the trailer is $1,038. Total acquisition cost ranges from $10,797 to $10,852.

The tank size of 16.4 gallons was considered and can be used for 4 to 5 hours on a full tank. Assume a full tank is refilled every two days. Therefore, the annual gas cost is $4,126. There is an annual maintenance cost that ranges from $150 to $200 [42]. In addition, the annual insurance of the jet ski ranges from $150 to $500. There is a two-year registration for $32; this cost will be added every two years [43]. The total yearly cost ranges from $4,458 to $4,858.

10.2 BREAK EVEN ANALYSIS BETWEEN JET SKIS AND THE LDDS

![Drone (S1000+) vs. Jet Ski](image)

**Figure 55. LDDS Drone with a Licensed Pilot Never Breaks Even**

Figure 55 shows the NPV for jet skis and the LDDS octocopter drone with an entry-level pilot (licensed pilot) over eight years. Looking at the graph, the LDDS with an entry-level pilot for the drone will always be more expensive than a jet ski. This is due to the high salary of the pilot. The case of of a student pilot, private pilot, and no pilot (lifeguard uses the drone) is considered below.
Figure 56. The LDDS is Cheaper After Some Years

The case of no pilot breaks even at 0 years. The case of a student pilot breaks even after 1 year. The case of a private pilot breaks even after 2 years. In these cases the LDDS drone is cheaper than the jet ski after some years. This is because of the relatively high annual cost of gasoline compared with the drone’s ability to recharge.

10.3 RESULTS

The comparison of jet skis with the LDDS is a repeat of Design of Experiment 3, except with a jet ski as an alternative. Since Galveston Island has 5 jet skis and about 25 towers, each jet ski is assumed to cover 5 sections each. Another alternative of just buying more jet skis (jet skis covers 3 sections each) is also considered. Assume the jet skis are constantly patrolling 150m from shore along the sections.

<table>
<thead>
<tr>
<th>Alternatives</th>
<th>Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lifeguard Only (Baseline)</td>
<td>Run Speed = 2.2 m/s</td>
</tr>
<tr>
<td></td>
<td>Swim Speed = 1.8 m/s</td>
</tr>
<tr>
<td>5-section Jet Ski (Baseline)</td>
<td>Assumed Velocity = 20mph</td>
</tr>
<tr>
<td>1-section Hexacopter</td>
<td>Optimal Battery and Location</td>
</tr>
<tr>
<td>1-section Octocopter</td>
<td>Optimal Battery and Location</td>
</tr>
<tr>
<td>3-section Hexacopter</td>
<td>Optimal Battery and Location</td>
</tr>
<tr>
<td>3-section Octocopter</td>
<td>Optimal Battery and Location</td>
</tr>
<tr>
<td>3-section Jet Ski</td>
<td>Assumed Velocity = 20mph</td>
</tr>
</tbody>
</table>

The results are shown below.
<table>
<thead>
<tr>
<th>Alternatives</th>
<th>Probability of reaching victim in time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Galveston Island, TX 2014</td>
<td>62/65 rescues = 0.954</td>
</tr>
<tr>
<td>Lifeguard Rescue</td>
<td>0.923</td>
</tr>
<tr>
<td>5-Section Jet Ski</td>
<td>0.833</td>
</tr>
<tr>
<td><strong>1-Section Hexacopter</strong></td>
<td><strong>0.994</strong></td>
</tr>
<tr>
<td><strong>1-Section Octocopter</strong></td>
<td><strong>1</strong></td>
</tr>
<tr>
<td>3-Section Hexacopter</td>
<td>0.883</td>
</tr>
<tr>
<td>3-Section Octocopter</td>
<td>0.908</td>
</tr>
<tr>
<td><strong>3-Section Jet Ski</strong></td>
<td><strong>0.942</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Distribution</th>
<th>Mean (sec)</th>
<th>Stdev</th>
</tr>
</thead>
<tbody>
<tr>
<td>Victim Survival</td>
<td>102.8</td>
<td>30.4</td>
</tr>
<tr>
<td>Lifeguard Rescue</td>
<td>48.3</td>
<td>29.5</td>
</tr>
<tr>
<td>5-Section Jet ski</td>
<td>74.4</td>
<td>18.3</td>
</tr>
<tr>
<td>1-Section Hexacopter</td>
<td>32.2</td>
<td>13.0</td>
</tr>
<tr>
<td><strong>1-Section Octocopter</strong></td>
<td><strong>30.8</strong></td>
<td><strong>10.9</strong></td>
</tr>
<tr>
<td>3-Section Hexacopter</td>
<td>62.6</td>
<td>22.0</td>
</tr>
<tr>
<td>3-Section Octocopter</td>
<td>59.1</td>
<td>20.5</td>
</tr>
<tr>
<td><strong>3-Section Jet Ski</strong></td>
<td><strong>47.4</strong></td>
<td><strong>15.6</strong></td>
</tr>
</tbody>
</table>

Figure 57. Time to Reach for All Alternatives
Shown in Figure 57, the jet ski mostly reduces the standard deviation of rescue. Buying more jet skis does increase the probability of successful rescue; however the LDDS 1-section still has the highest benefit.

<table>
<thead>
<tr>
<th>Alternatives</th>
<th># Drones</th>
<th>Total 5-year Cost</th>
<th>Total 5-year Cost of Life</th>
<th>Benefit</th>
</tr>
</thead>
<tbody>
<tr>
<td>LG</td>
<td></td>
<td>0</td>
<td>750,000</td>
<td>0.923</td>
</tr>
<tr>
<td>Octo 3-S</td>
<td>8</td>
<td>1,400,000</td>
<td>900,000</td>
<td>0.908</td>
</tr>
<tr>
<td>Octo 1-S</td>
<td>25</td>
<td>4,500,000</td>
<td>0</td>
<td>1.000</td>
</tr>
<tr>
<td>Student Pilot Scenarios</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Octo 3-S</td>
<td>8</td>
<td>150,000</td>
<td>900,000</td>
<td>0.908</td>
</tr>
<tr>
<td>Octo 1-S</td>
<td>25</td>
<td>500,000</td>
<td>0</td>
<td>1.000</td>
</tr>
<tr>
<td>Jet Ski 5-S</td>
<td>5</td>
<td>170,000</td>
<td>1,620,000</td>
<td>0.833</td>
</tr>
<tr>
<td>Jet Ski 3-S</td>
<td>8</td>
<td>220,000</td>
<td>570,000</td>
<td>0.942</td>
</tr>
</tbody>
</table>

The LDDS drone is lower cost and higher performance than a 20mph jet ski. Sensitivity Analysis on what average speed a jet ski needs to be higher performance than the LDDS system is done in Section 12.3.

In conclusion, with FAA restrictions, buying more jet skis is the recommended choice due to increasing percent saved from 92% to 94%, with only a 5% increase in cost. The LDDS drone cost will be always more expensive than a jet ski due to salary of a licensed pilot. With loosened restrictions, the LDDS will be cheaper than jet skis after 1 to 2 years. The LDDS 1-section will also have a higher benefit compared with jet skis.
COMBINING RESCUE ALTERNATIVES

The probabilities of successful rescues in the last section show that many alternatives have a lower probability of successful rescue than the baseline lifeguard rescue. However, in reality, if the jet ski or drone cannot make it to the victim, a lifeguard still can reach them in time. Thus the case of a combined rescue was considered. For every victim, if the drone does not reach them, but the lifeguard did, then the victim is considered safe.

The lowest probability is the lifeguard baseline rescue, and all other alternatives are built upon lifeguard rescue so that the alternatives are evaluated on reaching the victims the lifeguards cannot. Thus, probability of successful rescue has been modified in the table below.

<table>
<thead>
<tr>
<th>Alternatives</th>
<th># Drones</th>
<th>Total 5-year Cost</th>
<th>Total 5-year Cost of Life</th>
<th>Benefit</th>
</tr>
</thead>
<tbody>
<tr>
<td>LG</td>
<td>0</td>
<td>0</td>
<td>750,000</td>
<td>0.923</td>
</tr>
<tr>
<td>Octo 3-S</td>
<td>8</td>
<td>1,400,000</td>
<td>450,000</td>
<td>0.954</td>
</tr>
<tr>
<td>Octo 1-S</td>
<td>25</td>
<td>4,500,000</td>
<td>0</td>
<td>1.000</td>
</tr>
<tr>
<td>Student Pilot Scenarios</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Octo 3-S</td>
<td>8</td>
<td>150,000</td>
<td>450,000</td>
<td>0.954</td>
</tr>
<tr>
<td>Oct 1-S</td>
<td>25</td>
<td>500,000</td>
<td>0</td>
<td>1.000</td>
</tr>
<tr>
<td>Jet Ski Alternatives</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Jet Ski 5-S</td>
<td>5</td>
<td>170,000</td>
<td>450,000</td>
<td>0.954</td>
</tr>
<tr>
<td>Jet Ski 3-S</td>
<td>8</td>
<td>220,000</td>
<td>270,000</td>
<td>0.972</td>
</tr>
</tbody>
</table>
Galveston Island uses jet skis in rescue and the error between the real rescue probability in 2014 and the probability with jet ski 5-S is 0%. The best alternatives above the baseline are the Octo 1-S Student Pilot, Octo 3-S Student Pilot, Jet Ski 3-S, and Jet Ski 5-S. If FAA restrictions are considered, the best alternative for Galveston is to purchase more jet skis, as it has a lower cost (due to less lawsuits) and a higher probability of successful rescue than lifeguard only rescues. If FAA restrictions were loosened, the LDDS 1 section student pilot is the far superior alternative due to being the lowest cost and highest benefit.

### 12.0 SENSITIVITY ANALYSIS

#### 12.1 DISTANCE BETWEEN TOWERS

**Objective:** Determine how sensitive the results are to distance between towers.

Distance between towers is the radius of coverage the drone provides. A larger radius means rescues become more dispersed distance wise. Distance between towers of 400, 600, 800, ... 2000m were considered with 100 random victims for each distance. The percent saved and time to reach was recorded for each distance between towers. The results are shown below with a 95% CI around each point.

![Distance Between Towers vs. Victims Saved (alpha = 0.95)](image)

*Figure 60. How Distance Between Towers Affect Percent Saved*

\[ y = -0.0417x + 1.035 \]

\[ R^2 = 0.9351 \]
As the distance increases, the CI increases due to the higher variance from dispersed rescue locations. From the results, for every 200m added to the distance between towers, time to reach increases by 5 seconds, and percent saved decreases by 4%. There is a 95% confidence that the drone system will save less than 99% of victims when distance between towers is 800m or more. From section 1.4, 90% of the 68 recorded beaches have an average distance between towers under 800m. Thus the LDDS 1-section drone is a good alternative for 90% of beaches register with the USLA.

**12.2 PILOT DELAY AND PATH**

**Objective:** Determine how sensitive the results are to delays due to pilot skill or pathing.

The drone rescue process is path independent with respect to time. If a drone has two paths and both paths lead to the victim located at the same position, then both paths took the same amount of time for the drone to go through. Thus any path can be modeled or looked at as the drone flying straight to the victim, and then delaying the flotation device drop so the drone is following the victim but the rescue is not over yet. Furthermore, the added time of delaying the flotation device is called the delay time, while the time to reach is called the Reach Time. While varying the delay time, the 325 victims generated in DoE3 were added a varying number of delay times, and the percent saved is plotted. The results are shown below.
As shown in Figure 63, the LDDS can afford 5 to 10 seconds of delay and is still able to save 99% of victims. From DoE1, the around path was about 3 seconds slower than the straight path, thus the around path should still save 99% of victims for the 1-section LDDS alternative. The maximum delay the pilot is able to take is 30 seconds before the drone becomes worse than the simulated baseline lifeguard rescue.
12.3 JET SKI SPEED EQUIVALENCE

**Objective:** to find the equivalent jet ski velocity the drone is equal to.

One victim was generated and the jet ski was timed to reach this victim with a 10, 20, 30, 40, 50, and 60mph constant speed. Results are shown in Figure 64. The drone is approximately equivalent to a 20mph jet ski. If a beach timed their jet skis during rescue and found their average speed to be less than 20mph (distance from victim divided by the time it took the jet ski to reach the victim), then the drone is a better choice performance wise.

![Figure 64. Time to Reach the Victim for a Jet Ski](image-url)
## 13.2 SCHEDULE/MILESTONES

<table>
<thead>
<tr>
<th>Milestones</th>
<th>Dates</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fall Brief 1</td>
<td>9/21/2015</td>
</tr>
<tr>
<td>Fall Brief 2</td>
<td>10/5/2015</td>
</tr>
<tr>
<td>Preliminary Project Plan</td>
<td>10/21/2015</td>
</tr>
<tr>
<td>Fall Brief 3</td>
<td>10/26/2015</td>
</tr>
<tr>
<td>Fall Brief 4</td>
<td>11/9/2015</td>
</tr>
<tr>
<td>Faculty Presentation</td>
<td>11/20/2015</td>
</tr>
<tr>
<td>Proposal Final Reports</td>
<td>12/9/2015</td>
</tr>
<tr>
<td>Proposal Final Report Slides</td>
<td></td>
</tr>
<tr>
<td>Draft Conference Paper</td>
<td></td>
</tr>
<tr>
<td>Draft Poster</td>
<td></td>
</tr>
<tr>
<td>Milestones (SYST 495 Spring 2016)</td>
<td></td>
</tr>
<tr>
<td>Spring Brief 1</td>
<td>1/25/2016</td>
</tr>
<tr>
<td>Spring Brief 2</td>
<td>2/8/2016</td>
</tr>
<tr>
<td>SIEDS Abstract Due</td>
<td>2/9/2016</td>
</tr>
<tr>
<td>SIEDS Notification</td>
<td>3/4/2016</td>
</tr>
<tr>
<td>Westpoint abstract</td>
<td>3/11/2016</td>
</tr>
<tr>
<td>Spring Brief 3</td>
<td>3/14/2016</td>
</tr>
<tr>
<td>Dean business plan registration</td>
<td>3/19/2016</td>
</tr>
<tr>
<td>Pitch IT registration</td>
<td>3/20/2016</td>
</tr>
<tr>
<td>Spring Faculty presentation dry run</td>
<td>3/28/2016</td>
</tr>
<tr>
<td>Pitch IT</td>
<td>4/2/2016</td>
</tr>
<tr>
<td>Final westpoint abstract</td>
<td>4/2/2016</td>
</tr>
<tr>
<td>SIEDS Manuscript Due</td>
<td>4/4/2016</td>
</tr>
<tr>
<td>Spring Faculty Presentation</td>
<td>4/15/2016</td>
</tr>
<tr>
<td>Final deliverables</td>
<td>4/20/2016</td>
</tr>
</tbody>
</table>
13.3 PAYMENT SCHEDULE

The average salary of an Entry-Level Systems Engineer in Fairfax is 63,000 per year [21]. Assuming 50 work weeks and 40 hours a week, this is 27.50/hour. We will round this up to $30 for simplicity. We will assume an overhead multiplier of 2.00. Thus, the total charge is $60.00 per hour per person [21] [22]. The total project cost is $110,660.

13.4 WORK BREAKDOWN

13.4.1 WORK BREAKDOWN STRUCTURE


Figure 65
13.4.2 CRITICAL TASKS

We are considering a task as a critical task if it has a slack less than 2 days. This is a margin of error as in that time an unplanned for homework assignment or test could lead to slippage of the task for our project.

<table>
<thead>
<tr>
<th>#</th>
<th>WBS</th>
<th>TASK NAME</th>
<th>START</th>
<th>FINISH</th>
</tr>
</thead>
<tbody>
<tr>
<td>18</td>
<td>3.2.3</td>
<td>RESCUE PROCESS DIAGRAMS</td>
<td>10/8/15</td>
<td>1/23/16</td>
</tr>
<tr>
<td>21</td>
<td>4.2</td>
<td>FUNCTIONAL REQUIREMENTS</td>
<td>10/30/15</td>
<td>11/27/15</td>
</tr>
<tr>
<td>22</td>
<td>4.3</td>
<td>DESIGN REQUIREMENTS</td>
<td>10/31/15</td>
<td>11/27/15</td>
</tr>
<tr>
<td>29</td>
<td>5.1.4</td>
<td>EVALUATE DRONE BATTERY ALTERNATIVES</td>
<td>3/31/16</td>
<td>4/2/16</td>
</tr>
<tr>
<td>31</td>
<td>5.1.6</td>
<td>EVALUATE DRONE MOTOR ALTERNATIVES</td>
<td>3/17/16</td>
<td>3/19/16</td>
</tr>
<tr>
<td>33</td>
<td>5.1.8</td>
<td>EVALUATE DRONE ROTOR ALTERNATIVES</td>
<td>3/19/16</td>
<td>3/23/16</td>
</tr>
<tr>
<td>35</td>
<td>5.1.10</td>
<td>EVALUATE DRONE LOCATION ALTERNATIVES</td>
<td>3/14/16</td>
<td>3/17/16</td>
</tr>
<tr>
<td>36</td>
<td>5.1.11</td>
<td>RESEARCH DRONE CONFIGURATION ALTERNATIVES</td>
<td>10/1/15</td>
<td>10/3/15</td>
</tr>
<tr>
<td>37</td>
<td>5.1.12</td>
<td>EVALUATE DRONE CONFIGURATION ALTERNATIVES</td>
<td>10/3/15</td>
<td>10/27/15</td>
</tr>
<tr>
<td>40</td>
<td>5.2.2</td>
<td>EVALUATE FLOTATION DEVICE ALTERNATIVES</td>
<td>10/15/15</td>
<td>4/28/16</td>
</tr>
<tr>
<td>52</td>
<td>6.3</td>
<td>SENSITIVITY ANALYSIS</td>
<td>3/25/16</td>
<td>4/3/16</td>
</tr>
<tr>
<td>56</td>
<td>6.4.3</td>
<td>SIMULATION RISK ANALYSIS</td>
<td>11/27/15</td>
<td>12/5/15</td>
</tr>
<tr>
<td>77</td>
<td>8.1.7</td>
<td>CREATE RIP CURRENT MODEL</td>
<td>2/13/16</td>
<td>2/27/16</td>
</tr>
<tr>
<td>79</td>
<td>8.1.9</td>
<td>CREATE GUI</td>
<td>2/27/16</td>
<td>3/14/16</td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>80</td>
<td>8.2</td>
<td>SIMULATION TESTING</td>
<td>3/14/16</td>
<td>3/25/16</td>
</tr>
<tr>
<td>81</td>
<td>8.3</td>
<td>PERFORM SENSITIVITY TESTS</td>
<td>3/25/16</td>
<td>4/3/16</td>
</tr>
<tr>
<td>89</td>
<td>9.2.1</td>
<td>FACULTY PRESENTATION CREATION</td>
<td>11/14/15</td>
<td>11/19/15</td>
</tr>
<tr>
<td>90</td>
<td>9.2.2</td>
<td>FACULTY PRESENTATION</td>
<td>11/20/15</td>
<td>11/20/15</td>
</tr>
<tr>
<td>97</td>
<td>9.4.3</td>
<td>SPRING BRIEF 3</td>
<td>3/14/16</td>
<td>3/14/16</td>
</tr>
<tr>
<td>102</td>
<td>10.1.2</td>
<td>PROPOSAL FINAL REPORT</td>
<td>12/9/15</td>
<td>12/9/15</td>
</tr>
<tr>
<td>104</td>
<td>10.2.1</td>
<td>DRAFT CONFERENCE PAPER</td>
<td>12/9/15</td>
<td>12/9/15</td>
</tr>
<tr>
<td>105</td>
<td>10.2.2</td>
<td>DRAFT POSTER</td>
<td>12/9/15</td>
<td>12/9/15</td>
</tr>
<tr>
<td>106</td>
<td>10.2.3</td>
<td>FINAL REPORTS AND STUFF</td>
<td>11/23/15</td>
<td>12/8/15</td>
</tr>
</tbody>
</table>

### 13.4.3 SCHEDULE
13.4.4 SPI, CPI, SV, AND CV
CPI and SPI have been overall positive. CV and SV have been very positive due to the predicted cost was less than predicted.

13.4.5 EARNED VALUE
Again the earned value just reinforces that we were pessimistic about scheduling our tasks and that the pessimism and resultant spare time helped us to cover for the unforeseen tasks.

According to Figure 70, our earned value (BCWP) was always higher than our predicted cost and actual cost. The earned value flat lined near the end due to tasks not listed in the project file. In the end, our predicted cost (BCWS) was $113,781.72, our actual cost (ACWP) is $110,660.23, and our earned value (BCWP) is $111,664.08
## APPENDIX A: FINAL OCTOCOPTER PARAMETERS

<table>
<thead>
<tr>
<th>Variable</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total Drone Weight</td>
<td>m</td>
</tr>
<tr>
<td>Kv</td>
<td>Kv</td>
</tr>
<tr>
<td>Ktau/Kt</td>
<td>Kτ/Kτ</td>
</tr>
<tr>
<td>Moment of Inertias</td>
<td>Ixx, Iyy, Izz</td>
</tr>
<tr>
<td>Coefficients of Drag</td>
<td>C_{DX}, C_{DY}, C_{DZ}</td>
</tr>
<tr>
<td>Surface Area for Drone</td>
<td>Ax, Ay, Az</td>
</tr>
<tr>
<td>Surface Area for Life Ring</td>
<td>Ax, Ay, Az</td>
</tr>
<tr>
<td>Diameter of Propeller</td>
<td>D</td>
</tr>
<tr>
<td>Air density</td>
<td>ρ</td>
</tr>
<tr>
<td>Distance from Motor to Center</td>
<td>L</td>
</tr>
<tr>
<td>Coefficient of Drag for Propeller</td>
<td>C_{DP}</td>
</tr>
<tr>
<td>Propeller Cross Sectional Area</td>
<td>A</td>
</tr>
<tr>
<td>Propeller Moment of Inertia</td>
<td>Iz</td>
</tr>
<tr>
<td>Angle PID gains</td>
<td>Roll {0.125, 0.375, 0}</td>
</tr>
<tr>
<td></td>
<td>Pitch {0.125, 0.375, 0}</td>
</tr>
<tr>
<td></td>
<td>Yaw {0.01, 0.01, 0}</td>
</tr>
<tr>
<td>Velocity PID gains</td>
<td>Xdot {0.1, 0.3, 0}</td>
</tr>
<tr>
<td></td>
<td>Ydot {0.1, 0.3, 0}</td>
</tr>
<tr>
<td></td>
<td>Zdot {0, 0, 0}</td>
</tr>
<tr>
<td>Position PID gains</td>
<td>X {0.2, 1.2, 0}</td>
</tr>
<tr>
<td></td>
<td>Y {0.2, 1.2, 0}</td>
</tr>
<tr>
<td></td>
<td>Z {0.2, 0.6, 0}</td>
</tr>
</tbody>
</table>
Calculate Velocity

```matlab
function [VveloX,VveloY] = CalcVictimVelo(escape, ripProp, currPosi, Swimspeed)
%#codegen

%escape : chosen escape method.
%escape = 0 if method is fighting against current (-x velocity)
%escape = 1 if method is floating (x velocity is 0)
%escape = 2 if method is swimming parallel to shore (-x and -y velocity)
%Adjust Swimspeed to make sure not 0 and not above olympic record
SS = Swimspeed(1);
if (SS<0.1) SS = 0.1; end
if (SS>2.4) SS = 2.4; end
%calculate Vvelo based on escape method
Vvelo = [0;0]; [%xvelo; yvelo] in meters per second
switch (escape)
    case 0 %fight against
        Vvelo(1) = -SS; % [-2, 0]
    case 1 %float
        %Vvelo = [0;0]; do nothing
    case 2 %swimming parallel
        Vvelo(2) = -SS; % [0, -2]
end
```
% rip current properties
% [length, width, speed, midpoint in y axis]
% assumes rip current begins at x = 0
len = ripProp(1);
width = ripProp(2);
Vrip = ripProp(3);
mid = ripProp(4);

if (escape ~= 2) % not swim parallel
    Vvelo = Vvelo + [Vrip;0];
    if ((abs(currPos(1)-len) < 1))
        Vvelo(1) = 0;
    end
else % else they are swimming parallel then we need to do two stages
    % SecVelo = velocity when out of rip current
    SecVelo = [-SS;0];
    if (abs(currPos(2)-mid)>width-1) % if outside the width of rip current
        Vvelo = SecVelo;
    else
        Vvelo = Vvelo + [Vrip;0];
        if (currPos(1)>len) % if outside of rip vertically, cant go vertical anymore
            Vvelo(1) = 0;
        end
    end
end

VveloX = Vvelo(1);
VveloY = Vvelo(2);

if (currPos(1)<0) || (currPos(1)>len) % if they reach the shore, dont go farther!
    VveloX=0;
end
Calculate Lifeguard Velocity

```matlab
function [VveloX, VveloY, meetYet] = CalcLGVelo(Waypoint, currPosi, ripProp, LGland, LGswim, jetski)
    % victim position
    vx = Waypoint(1);
    vy = Waypoint(2);

    % current position
    LGx = currPosi(1);
    LGy = currPosi(2);

    % rip current properties
    % [length, width, speed, midpoint in y axis]
    % assumes rip current begins at x = 0
    len = ripProp(1);
    width = ripProp(2);
    Vrip = ripProp(3);
    mid = ripProp(4);

    LGlandspeed = LGland;
    LGswimspeed = LGswim;

    VveloX = 0;
    % jet ski is 0 or 1, 1 meaning yes
    if (jetski == 0)
        VveloY = LGlandspeed;
    else
        VveloY = LGswimspeed;
    end
```
%if horizontal distance is close enough, go vertical
if (abs(LGy - vy) < 1)
    VveloY = 0;%stop horizontal movement
if (abs(vx-LGx)>1)
    if (jetski==0)
        VveloX = LGswimspeed + Vrip;
    else if ((vx-LGx)>0)
        %victim is ahead of lifeguard
        VveloX = LGswimspeed + Vrip;
    else %victim is not ahead
        VveloX = -LGswimspeed + Vrip;
    end
end
end

%-----------------------------------------------
%Did the LG meet the victim yet?
if (abs(LGx-vx)<1.1)
    meetYet = 1;
else
    meetYet = 0;
end

%-----------------------------------------------

LINEAR DYNAMICS

What comes into the system is a voltage vector ([v1, v2, v3, v4,v5,v6,v7,v8]). They will be inputted
into volt2thrust in order to get a value for the thrust in body form. In the linear motion model, we will use the equation above to calculate the acceleration. Note that mass of drone is separated outside, because it is varied during the DoEs. After integrating the acceleration for position and velocity, these values will be used to calculate various other forces. Velocity will be used to calculate the drag forces on the drone and on the life vest.

Voltage to Thrust Conversion (volt2thrust)

```matlab
function [THRUST, power1] = volt2thrust(VOLTS)
%#codegen
%Function to convert voltage into the thrust force vector in inertia frame
%All units are metric unless specified

%constants
Kv = 1/400; %proportionality constant between propeller voltage and angular velocity
Ktau = 28; %proportionality constant between propeller torque and current
KT = 1; %proportionality constant between propeller thrust and torque
D = 0.0322; %propeller diameter. Adjusted by propeller properties
rho = 1.225; %density of air (humid or dry). Probably need to be a function

% get voltage out of VOLTS
v1 = VOLTS(1);
v2 = VOLTS(2);
v3 = VOLTS(3);
v4 = VOLTS(4);
v5 = VOLTS(5);
v6 = VOLTS(6);
v7 = VOLTS(7);
v8 = VOLTS(8);

% V = Kv * w assuming internal resistance = 0
w1 = v1/Kv;
w2 = v2/Kv;
w3 = v3/Kv;
w4 = v4/Kv;
w5 = v5/Kv;
w6 = v6/Kv;
w7 = v7/Kv;
w8 = v8/Kv;

W = [0; 0; (w1^2+w2^2+w3^2+w4^2+w5^2+w6^2+w7^2+w8^2)];
k = pi/2*rho*D^2*(Kv*Ktau/KT);
%ThrustBody = thrust vector in body frame
thrustBody = k * W;
%THRUST = thrust vector in inertia frame
power = Kv*Ktau/KT*k*[w1^3,w2^3,w3^3,w4^3,w5^3,w6^3,w7^2,w8^2];
power1 = power;

THRUST = thrustBody;
```
**Linear Motion Model (DroneState)**

```matlab
function [ax,ay,az] = DroneState(Fthrust,Fdrag,Flifevest,EULER,massDrone)
%#codegen

%rotation matrix for converting body-frame thrust to inertia-frame thrust
roll=EULER(1);
pitch=EULER(2);
yaw=EULER(3);
Ryaw = [cos(yaw) sin(yaw) 0; -1*sin(yaw) cos(yaw) 0; 0 0 1];
Rpitch = [cos(pitch) 0 -1*sin(pitch); 0 1 0; sin(pitch) 0 cos(pitch)];
Rroll = [1 0 0; 0 cos(roll) sin(roll); 0 -1*sin(roll) cos(roll)];

RotMat = Ryaw*Rpitch*Rroll;

THRUSt = RotMat*Fthrust;
%-------------------------------------------------

Fx = 0 + THRUST(1,1) + Fdrag(1,1) + Flifevest(1,1);
Fy = 0 + THRUST(2,1) + Fdrag(2,1) + Flifevest(2,1);
Fz = -massDrone*9.81 + THRUST(3,1) + Fdrag(3,1) + Flifevest(3,1);

ax = Fx/massDrone;
ay = Fy/massDrone;
az = Fz/massDrone;
```

**Force of Drag (Fdrag)**

```matlab
function Fdrag = Fdrag(vx,vy,vz)
%#codegen

%force of drag depending on its air resistance
%pulls drone BACKWARDS
Cd = [0.9,0.9,1.2]; %coefficient of drag for the lifevest
rho = 1.225; %air density
X=0.0517;%area normal to X-axis
Y=0.0517;%area normal to Y-axis
Z=0.1587;%area normal to Z-axis
CdA = [X*Cd(1) 0 0; 0 Y*Cd(2) 0; 0 0 Z*Cd(3)]; %cross sectional area of lifevest
%backwards = negative of current velocity vector
Vsq = sqrt((vx^2)+(vy^2)+(vz^2)); %total velocity magnitude
%unit vector direction of velocity
mag = sqrt(vx^2+vy^2+vz^2); %magnitude of total velocity
if mag==0
    mag=1,
end
unitV = [vx;vy;vz]*-1/mag;

Fmag = CdA*rho*Vsq^2; %magnitude of drag force
Fdrag = Fmag*unitV;
```
% Fdrag=[0;0;0];

**Force of Lifevest (LifevestState)**

```matlab
function Flife = LifevestState(vx,vy,vz)
    %#codegen
    %force of drag depending on its air resistance
    %pulls drone BACKWARDS
    Cd = 1.2; %coefficient of drag for the lifevest
    rho = 1.225; %air density
    A = [0.0608 0 0; 0 0.0608 0; 0 0 0.315]; %cross sectional area of lifevest
    %backwards = negative of current velocity vector
    Vsqr = sqrt((vx^2)+(vy^2)+(vz^2)); %total velocity magnitude
    %unit vector direction of velocity
    mag = sqrt(vx^2+vy^2+vz^2); %magnitude of total velocity
    if mag==0
        mag=1;
    end
    unitV = [vx;vy;vz]*-1/mag;
    Fdrag = Cd*rho*A*Vsqr^2;
    FDRAG = Fdrag*unitV;
    %force of weight depending on mass
    vestMass = 2.268; %mass of the lifevest
    Fgrav = [0;0;vestMass*-9.81]; %pulls drone DOWN
    Flife = FDRAG + Fgrav;
    % Flife = [0;0;0];
```
Similar to the linear motion model, we take in four voltages (v1, v2, v3, v4) to calculate the angular velocities of each rotor. The velocities and the derivative (acceleration) will be used to calculate the torques in the main rotational model (21). Since we need the Euler angles instead of angular velocity, we need to convert angular velocity to Euler angle derivatives. Then we integrate the Euler angles into roll, pitch, and yaw which we can use for the rotation matrices in the linear motion model (4)(17).

**voltage to motor angular acceleration and velocity**

```matlab
function Wmotor = volt2motorW(v1,v2,v3,v4)
%#codegen
Kv = 1; %proportionality constant between voltage and motor angular velocity

w1=v1/Kv;
w2=v2/Kv;
w3=v3/Kv;
w4=v4/Kv;

Wmotor = [w1,w2,w3,w4];
```

**motor angular velocity to torque in body frame**
function [Troll,Tpitch,Tyaw] = w2tau(W,Wdot)
    %#codegen
    Kv=1/400; %proportionality constant between motor voltage to angular velocity
    Ktau = 1; %proportionality constant between propeller torque and current
    KT = 1; %proportionality constant between propeller thrust and torque
    L=0.5225; %distance between motor and center of body frame.
    D=0.0322; %diameter of propeller
    rho=1.225; %density of air
    Cd=0.2; %coefficient of drag for propeller wings
    A=0.00081433; %propeller swipe area.
    r=D/2; %radius of propeller. Do not change
    Iz = 0.00002; %moment of inertia for propellers

    %equations for constants we will need for tau
    k = pi/2*rho*D^2*(Kv*Ktau/KT);
    b = 0.5*r*rho*Cd*A*r^2;
    %assume w1 and w3 are right and left motors respectively.
    % assume w2 and w4 are back and front motors respectively.
    %grab angular motor velocities
    w1 = W(1); w2 = W(2); w3 = W(3); w4 = W(4); w5 = W(5); w6 = W(6); w7 = W(7); w8 = W(8);
    %tau of roll pitch and yaw based on angular velocities
    Troll = sin(3*pi/8)*L*k*(w1^2+w2^2-w5^2-w6^2);
    Tpitch = sin(3*pi/8)*L*k*(w3^2+w4^2-w7^2-w8^2);
    Tyaw = b*(w1^2-w2^2+w3^2-w4^2+w5^2-w6^2+w7^2-w8^2)+Iz*(Wdot(1)-Wdot(2)+Wdot(3)-Wdot(4)+Wdot(5)-Wdot(6));

Rotational Motion Model

function [wdotx,wdoty,wdotz] = RotationMotion(Troll,Tpitch,Tyaw,eulerDot,euler)
    %#codegen

    %moment of inertias in the X/Y/Z-axis of the drone body frame
    Ixx = 0.4447;
    Iyy = 0.4447; %or 0.4504
    Izz = 0.8246;
    I = [Ixx 0 0; 0 Iyy 0; 0 0 Izz];

    roll = euler(1);
    pitch = euler(2);
    yaw = euler(3);

    rollDot = eulerDot(1);
pitchDot = eulerDot(2);
yawDot = eulerDot(3);

tau = [Troll;Tpitch;Tyaw];
%First generate rotation matrix for angular velocity and thetaDot
eulerDot2Omega = [1 0 -sin(pitch);
0 cos(roll) sin(roll)*cos(pitch)
0 -sin(roll) cos(roll)*cos(pitch)];

%convert the euler Dots to angular velocities in inertial frame
Omega = eulerDot2Omega*[rollDot;pitchDot;yawDot];
W = [Omega(1);Omega(2);Omega(3)];

%use rotational motion equations now
Wdot = inv(I)*(tau - cross(W, I*W));
wdotx=Wdot(1);
wdoty=Wdot(2);
wdotz=Wdot(3);

Convert body angular velocity to Euler angle velocity

function [eulerDotRoll,eulerDotPitch,eulerDotYaw] = w2eulerDot(wx,wy,wz, euler)
%codegen

roll = euler(1);
pitch = euler(2);
yaw = euler(3);

%rotational matrix for EULERDOT to W. need to inverse it to get %the relevant transformation matrix
E2W = [1 0 -sin(pitch);
0 cos(roll) sin(roll)*cos(pitch);
0 -sin(roll) cos(roll)*cos(pitch)];
eulerDot = E2W*[wx;wy;wz];
eulerDotRoll = eulerDot(1);
eulerDotPitch = eulerDot(2);
eulerDotYaw = eulerDot(3);

**Euler Angle PID Controller**

![Diagram of Euler Angle PID Controller]

Takes position or velocity error (depending on how close the victim is), and convert it into an angle error for the Controller to adjust voltage with.

**Get Wanted Euler Angle**

```matlab
function wantedAngles = GetWantedAngle(Ep,Ev,posiDiff)
%#codegen
%grab all the errors for position and then velocity
Ex = Ep(1);
Ey = Ep(2);
Ez = Ep(3);

%These velocity errors are MAX velocity errors
Evx = Ev(1);
Evy = Ev(2);
Evz = Ev(3);
```
xDif = posiDiff(1);
yDif = posiDiff(2);
zDif = posiDiff(3);

% ---------------------------
% the angle is want is between pi/6 and -pi/6
% if vy is positive, it means we need to go left. To go left, we need to
% roll positive
% if vx is positive, it means we need to go forward. To go forward, we need
% to pitch negative

wantedRoll = -pi/6 * (1-exp(0.5*Ev))/((1+exp(0.5*Ev)));
% wantedRoll=0;
wantedPitch = pi/6 * (1-exp(0.5*Ev))/((1+exp(0.5*Ev)));
wantedYaw = 0;

% DoE3 PID only, turn below off when not in DoE3
% wantedPitch = pi/6 * (1-exp(0.5*Ex))/((1+exp(0.5*Ex)));
% wantedRoll = -pi/6 * (1-exp(0.5*Ey))/((1+exp(0.5*Ey)));
%-------------------

% STOPPING PHASE
% If we get within a certain distance of the target, we start to slow down
xrad = 25;
yrad = 25;
if (abs(xDif)<=xrad)
    wantedPitch = pi/6 * (1-exp(0.5*Ex))/((1+exp(0.5*Ex)));
end
if (abs(yDif)<yрад)
    wantedRoll = -pi/6 * (1-exp(0.5*Ey))/((1+exp(0.5*Ey)));
end
% comment above out when in DoE 3
wantedAngles = [wantedRoll; wantedPitch; wantedYaw];
Position PID Controller

Max Velocity PID Error
MAIN MODEL
The model will take in the waypoint position and position are used to calculate the PID errors. Position and velocity is used to build a Euler angle error. For example, if we want to accelerate in the X-direction and we want to go at max velocity, we need to a large magnitude of clockwise pitch. On the other hand, if we need to slow down, we need to pitch in the opposite direction (counterclockwise or positive pitch). We assume the drone will have a max angle of roll-pitch-yaw. Many controls on drones set the max angles at 45 or 30 degrees. We chose 30 degrees as it is more common of a limit.

After calculating the wanted angle, we calculate the angle PID error. Both errors will be used in the controller, which converts the error readings into voltage. We simply added the errors to the hover voltage (voltage needed to hover, stay in the same position), in order to get the next time step’s voltage. We must adjust the PID gains in order for voltage to make sense of the errors.

Controller

```matlab
function VOLTS = Controller(Eposi,Evelo,posiDiff, mass, EmaxV, EDeuler)
 %#codegen

%---------------------------------------------------------------
% grab all errors first
Ex = Eposi(1);
Ey = Eposi(2);
Ez = Eposi(3);

Evx = Evelo(1);
Evy = Evelo(2);
Evz = Evelo(3);

Xdiff = posiDiff(1);
Ydiff = posiDiff(2);
Zdiff = posiDiff(3);

Emvx = EmaxV(1);
Emvx = EmaxV(2);
Emvx = EmaxV(3);

EDroll = EDeuler(1);
EDpitch = EDeuler(2);
EDyaw = EDeuler(3);
%---------------------------------------------------------------
% create the voltages
v1=0;
v2=0;
v3=0;
v4=0;
v5=0;
v6=0;
v7=0;
v8=0;
%---------------------------------------------------------------
% amount of thrust needed to hover
% how much thrust is needed to hover? (assuming 0 angles)
```
Kv = 1/400; % proportionality constant between propeller voltage and angular velocity  
Ktau = 28; % proportionality constant between propeller torque and current  
KT = 1; % proportionality constant between propeller thrust and torque  
D = 0.0322; % propeller diameter. Adjusted by propeller properties  
rho = 1.225; % density of air (humid or dry). Probably need to be a function  

k = pi/2*rho*D^2*(Kv*Ktau/ KT);  

vHover = Kv*sqrt(mass*9.81/(8*k));  
% set a max voltage the motor can be  
maxVoltage = 10;  
% --------------------------------------  
% What is thrust of 1 rotor?  
V0 = vHover + Ez; V0 = min(V0, maxVoltage); V0 = max(V0, 0); % checks to make sure thrust is within range  
% thrust is all the same currently  
v1 = V0; % v1 = vHover + Ez  
v2 = V0; % v2 = vHover + Ez  
v3 = V0;  
v4 = V0;  
% -------------------------  
% What is the max voltage that can handle the same throttle?  
% Thrust of 1 pair = v1^2 + v3^2  
% Thrust of 1 pair = 2v1^2 + 0  
% Voltage of 1 rotor while other is 0 voltage = sqrt(2)*v1  
% UNLESS the 1 rotor is above the max voltage, then we must truncate  
maxPairVoltage = sqrt(2)*v1; maxPairVoltage = min(maxPairVoltage, maxVoltage);  
% -----------------------  
% To maintain throttle, v1 and v3 must increase and decrease in pairs.  
% v1 = min(v1+ EDroll+ EDyaw, maxPairVoltage); v1 = max(v1, 0);  
v2 = min(v2 + EDroll - EDyaw, maxPairVoltage); v2 = max(v2, 0);  
v3 = min(v3 + EDpitch + EDyaw, maxPairVoltage); v3 = max(v3, 0);  
v4 = min(v4 + EDpitch - EDyaw, maxPairVoltage); v4 = max(v4, 0);  
% v3 = v3 - Eroll;  
v5 = sqrt(maxPairVoltage^2 - v1^2);  
v6 = sqrt(maxPairVoltage^2 - v2^2);  
v7 = sqrt(maxPairVoltage^2 - v3^2);  
v8 = sqrt(maxPairVoltage^2 - v4^2);  
% ------------------  
% v1 = v1 + Ey + 0.01*Ev;  
v2 = v2 + Ex + 0.01*Evx;  
v3 = v3 - Ey - 0.01*Ev;  
v4 = v4 - Ex - 0.01*Evx;  
% -----------------  

VOLTS = [v1; v2; v3; v4; v5; v6; v7; v8];
%Design of Experiment 1 - Best path, Minimum Velocity, Best Location
%Purpose: To find the best path of the drone (straight to victim versus %aroundabout). To find the velocity's relationship with percent of victims %reached. To find the best location for the drone to be stationed(tower, main room, or on %ship)

%Precondition: MUST have Waypoints.mat, LGModel.m, and VictimModel.m, and ripProp.mat in %folder
%IMPORTANT: if the end time was changed in VictimModel.m or LGModel.m, change the time
%on line 131
%Output:
% simout = [escape method, simulation output, survival time, rip properties]
% simout2 = time to reach for [L1P1, L1P2, L2P1, L2P2, ........]

%Independent Variables
% 1. Paths (straight or around and in front 5m away, or just straight and left and go?)
% 2. Velocities (constant, ranging from 5m/s to 15m/s (24m/s max from FAA)
% 3. Location ((0,0,0), at control room 80m back and randomly around, at ship 150m out located y-axis by a distrubution between 0 and 500m away from LG)
% 4. rip current properties (mid, width, length, speed)
% 5. Victim escape method (See DoE1.m)
%Outputs
% 1. Percent reached under 30 seconds
% 2. Percent reached under 60 seconds
% 3. Percent reached under 90 seconds
%
%Random Distrubutions
% 1. Rip current Properties
% Rip Current Length (100 - 300ft) / Unif(30.5,91.5 m)
% Rip Current Speed (1 - 8ft/s) / (0.308 - 2.438 m/s) Uniform
% Rip Current Width (10 - 200ft) / (3.048 - 60.96 m) Uniform
% Rip Current Mid (0 to 183m) uniform based on case study
% Distance between towers (Triangle(25,53.3,400)yards) N=27 beaches
% 2. Victim Escape Methods
% P(fight)=0.4, P(float)=0.25, P(parra)=0.35
% 3. Victim Survival Time
% Survival for Floating (normal abougt 80 seconds) stdev:20sec
% Survival for Fighting (normal about 130 seconds) stdev:20sec
% Survival for Swimming Parallel (normal about 180 seconds) stdev:20sec
% 4. Victim Swim Speeds (normal 1.161m/s, stdev 0.267m/s)
% 5. Boat Position (uniform 0-500m) further things needed

%% Load the files needed
load('RipProp');
load_system('VictimModel');

%% Determine number of iterations
%we want to create victims of random rip props
iterations = 100;
simout = cell(iterations,4); %[escape, simrun, survival time, ripProps]

%% Determine victim characteristics
%Generate escapes
%P(0)=0.4
%P(1)=0.25
%P(2)=0.35
for i=1:iterations
    tempRand = rand;
    if tempRand<=0.4
        simout{i,1}=0;
    else if tempRand<=0.65
        simout{i,1}=1;
    else
        simout{i,1}=2;
    end
end

%Generate the sim runs of each victim
for i = 1:iterations
    ripProp(:,2)=rand*61+30.5;
    ripProp(:,3)=rand*57.912+3.048;
    ripProp(:,4)=rand*2.13 + 0.308;
    ripProp(:,5)=rand*183;
    simout{i,4}=ripProp;
    set_param('VictimModel/EscapeValue','Value',num2str(simout{i,1}));
    set_param('VictimModel/Swimspeed','Value',num2str(randn*0.267+1.161));
    sim('VictimModel');
    simout{i,2} = Vposi;
end

%General survival times based on escape method
for i = 1:iterations
    S = -1;
    switch simout{i,1}
        case 0 %fight
            while (S<15)
                S = randn*20+80;
            end
            simout{i,3} = S;
        case 1 %float
            while (S<15)
                S = randn*20+130;
            end
            simout{i,3} = S;
        case 2 %swim parallel
            while (S<15)
            end
            simout{i,3} = S;
end

109
S = randn*20+180;
end
end
simout{i,3} = S-mod(S,0.05);
end

%% Generate locations
simout2 = cell(iterations,8); %[L1P1, L1P2, L2P1, L2P2, .... to L4P2]
Tests = cell(iterations,1);
%IMPORTANT - Where you change velocity for the experiment------------------
velo=12.5; %the velocities we will be testing in m/s
-----------------------------
for victim = 1:iterations %go thru each victim and create the path
%we will do 1 loop for each location path combination since its
%basically the same for all combinations
victimPosi = simout{victim, 2}(:,1:2); %just x and y coordinates
pL1P1 = [0,0]; %tower, straight to victim
pL1P2 = [0,0]; %tower, go straight then to rip current and around
mainPosiY = rand*(548-victimPosi(2)); %covers 3 towers
pL2P1 = [0,-mainPosiY]; %3 towers, go straight
pL2P2 = [0,-mainPosiY]; %3 towers, go around
main2PosiY = rand*(913-victimPosi(2)); %5 towers
pL3P1 = [0,-main2PosiY]; %5 towers, go straight
pL3P2 = [0,-main2PosiY]; %5 towers, go around
boatPosiY = rand*(800-victimPosi(2));
pL4P1 = [0,-boatPosiY];
pL4P2 = [0,-boatPosiY];
repeat = [0,0,0,0,0,0,0,0]; %once drone reaches victim, we dont care to
simulate more
totalDist = [0,0,0,0,0,0,0,0]; %distance per victim with each combo

testTable = zeros(301,8);
sizel = [0,0];

for t = 20*(0:1:180)+1 %go thru each time stamp <-------- IMPORTANT: if
%the end time has been changed, change here.
pV = victimPosi(t,1:2); %just x and y coords
%L1P1------------------------------------------
if ((pdist([pL1P1;pV])>=1)&&(repeat(1)==0)) %L1P1 - if we still not
within 1m of target, keep going
angle = atan((pV(1)-pL1P1(1)) / (pV(2)-pL1P1(2)));
    totalDist(1)=totalDist(1)+velo;
    dX=(pV(1)-pL1P1(1));
    dY=(pV(2)-pL1P1(2));
    distTra = [abs(velo*sin(angle)),abs(velo*cos(angle))];

    if (dX<0) %the difference in x-position and the velocity has to be
    the same sign
        distTra(1) = max(dX,-distTra(1));
    else
        distTra(1) = min(dX,distTra(1));
    end
end

110
if (dY<0) % same for y
    distTra(2) = max(dY, -distTra(2));
else
    distTra(2) = min(dY, distTra(2));
end

if (t>=10*20+1) % wait 10 seconds for liftoff
    pL1P1 = pL1P1 + distTra;
end
if (pdist([pL1P1; pV])<1)
    repeat(1)=1;
    simout2{victim,1}=totalDist(1)/velo;
end
end

% testTable(round((t-1)/20+1),1:2)=pL1P1; % ALL GOOD!
% L1P2------------------------------------------
% it will be done thru stages, stage 1- go vertical until we past
% victim by 10m. stage 2 - go horizontal til we just over victim
% stage 3 - go straight to victim.
% stage 1 go vertical with velo m/s. stages will be counted using
% the repeat table variable [0,1,2]
if ((pdist([pL1P2; pV])>=1)&&(repeat(2)~=3))
    totalDist(2)=totalDist(2)+velo;
    dX= (pV(1)-pL1P2(1));
    dY= (pV(2)-pL1P2(2));
    switch repeat(2)
        case 0 % go vertical til 10 m above victim
            if dX<-10
                distTra=[0,0];
                repeat(2)=1; % go to next stage
            else
                distTra=[velo,0];
            end
            if (t>=10*20+1) % wait 10 seconds for liftoff
                pL1P2 = pL1P2 + distTra;
            end
            if dX<-10
                repeat(2)=1; % go to next stage
            end
        case 1 % go horizontal til x-positions match
            if abs(dY)<=1
                repeat(2)=2;
                distTra=[0,0];
            else
                distTra=[0,0];
                if (dY<0)
                    distTra(2) = max(-velo, dY);
                else
                    distTra(2) = min(velo, dY);
                end
            end
            if (t>=10*20+1) % wait 10 seconds for liftoff
                pL1P2 = pL1P2 + distTra;
            end
            if abs((pV(2)-pL1P2(2)))<1
        end
end
repeat(2)=2;
end

case 2 %normal directoning
angle = atan((pV(1)-pL1P2(1)) / (pV(2)-pL1P2(2)));
distTra = [abs(velo*sin(angle)),abs(velo*cos(angle))];
if (dX<0) %the difference in x-position and the velocity has to be the same sign
    distTra(1) = max(dX,-distTra(1));
else
    distTra(1) = min(dX,distTra(1));
end
if (dY<0) %same for y
    distTra(2) = max(dY,-distTra(2));
else
    distTra(2) = min(dY,distTra(2));
end
if (t>=10*20+1) %wait 10 seconds for liftoff
    pL1P2 = pL1P2+distTra;
end
if (pdist([pL1P2;pV])<=2)
    repeat(2)=3;
simout2{victim,2}=totalDist(2)/velo;
end
end
end
testTable(round((t-1)/20+1),1:2)=pL1P2; %ALL GOOD!

%L2P1-----------------------------------------------
if ((pdist([pL2P1;pV])>=1)&&(repeat(3)==0)) %L1P1 - if we still not within lm of target, keep going
angle = atan((pV(1)-pL2P1(1)) / (pV(2)-pL2P1(2)));
totalDist(3)=totalDist(3)+velo;
dX= (pV(1)-pL2P1(1));
dY=( pV(2)-pL2P1(2));
distTra = [abs(velo*sin(angle)),abs(velo*cos(angle))];
if (dX<0)
    distTra(1) = max(dX,-distTra(1));
else
    distTra(1) = min(dX,distTra(1));
end
if (dY<0)
    distTra(2) = max(dY,-distTra(2));
else
    distTra(2) = min(dY,distTra(2));
end
if (t>=10*20+1) %wait 10 seconds for liftoff
    pL2P1 = pL2P1+distTra;
end
if (pdist([pL2P1;pV])<2)
    repeat(3)=1;
simout2{victim,3}=totalDist(3)/velo;
end
end
testTable(round((t-1)/20+1),3:4)=pL2P1; %ALL GOOD!

if (pdist([pL2P2;pV])>=1&&(repeat(4)~=3))
totalDist(4)=totalDist(4)+velo;
dX= (pV(1)-pL2P2(1));
dY= (pV(2)-pL2P2(2));
switch repeat(4)
    case 0 %go vertical til 10 m above victim
        if dX<10
            distTra=[0,0];
            repeat(4)=1; %go to next stage
        else
            distTra=[velo,0];
        end
        if (t>=10*20+1) %wait 10 seconds for liftoff
            pL2P2 = pL2P2+distTra;
        end
    end
    case 1 %go horizontal til x-positions match
        if abs(dY)<=1
            repeat(4)=2;
            distTra=[0,0];
        else
            distTra=[0,0];
            if (dY<=0)
                distTra(2)= max(-velo,dY);
            else
                distTra(2)= min(velo,dY);
            end
        end
        if (t>=10*20+1) %wait 10 seconds for liftoff
            pL2P2 = pL2P2+distTra;
        end
    end
    case 2 %normal direction
        angle = atan((pV(1)-pL2P2(1)) / (pV(2)-pL2P2(2)));
distTra = [abs(velo*sin(angle)),abs(velo*cos(angle))];
if (dX<0) %the difference in x-position and the velocity has to be the same sign
    distTra(1) = max(dX,-distTra(1));
else
    distTra(1) = min(dX,distTra(1));
end
if (dY<0) %same for y
    distTra(2) = max(dY,-distTra(2));
else
    distTra(2) = min(dY,distTra(2));
end
if (t>=10*20+1) %wait 10 seconds for liftoff
    pL2P2 = pL2P2+distTra;
end
end
if (pdist([pL2P2;pV])<=2)
    repeat(4)=3;
    simout2{victim,4}=totalDist(4)/velo;
end
end
testTable(round((t-1)/20+1),3:4)=pL2P2; %ALL GOOD
%L3P1-----------------------------------------------
if ((pdist([pL3P1;pV])>=1)&&(repeat(5)==0)) %L1P1 - if we still not
    within 1m of target, keep going
    angle = atan((pV(1)-pL3P1(1)) / (pV(2)-pL3P1(2)));
    totalDist(5)=totalDist(5)+velo;
    dX= (pV(1)-pL3P1(1));
    dY= (pV(2)-pL3P1(2));
    distTra = [abs(velo*sin(angle)),abs(velo*cos(angle))];
    if (dX<0)
        distTra(1) = max(dX,-distTra(1));
    else
        distTra(1) = min(dX,distTra(1));
    end
    if (dY<0)
        distTra(2) = max(dY,-distTra(2));
    else
        distTra(2) = min(dY,distTra(2));
    end
    if (t>=10*20+1) %wait 10 seconds for liftoff
        pL3P1 = pL3P1+distTra;
    end
    if (pdist([pL3P1;pV])<2)
        repeat(5)=1;
        simout2{victim,5}=totalDist(5)/velo;
    end
end
% testTable(round((t-1)/20+1),5:6)=pL3P1; %ALL GOOD!
%L3P2 -----------------------------------------------
if ((pdist([pL3P2;pV])>=1)&&(repeat(6)>=3))
    totalDist(6)=totalDist(6)+velo;
    dX= (pV(1)-pL3P2(1));
    dY= (pV(2)-pL3P2(2));
    switch repeat(6)
        case 0 %go vertical til 10 m above victim
            if dX<0
                distTra=[0,0];
                repeat(6)=1;%go to next stage
            else
                distTra=[velo,0];
            end
            if (t>=10*20+1) %wait 10 seconds for liftoff
                pL3P2 = pL3P2+distTra;
            end
            if dX<0
repeat(6)=1;%go to next stage
end

\textbf{case 1} %go horizontal til x-positions match
if \abs{dY}<=1
  repeat(6)=2;
distTra=[0,0];
else
  distTra=[0,0];
  if (dY<=0)
    distTra(2)= max(-velo,dY);
  else
    distTra(2)= min(velo,dY);
end
if (t>=10*20+1 )%wait 10 seconds for liftoff
  pL3P2 = pL3P2+distTra;
end
if abs((\pV-\pL3P2(2)))<=2
  repeat(6)=2;
end

\textbf{case 2} %normal direcitoning
angle = \atan{(\pV(1)-\pL3P2(1)) / (\pV(2)-\pL3P2(2))};
distTra = [abs(velo*sin(angle)),abs(velo*cos(angle))];
if (dX<0) %the difference in x-position and the velocity has to be the same sign
  distTra(1) = max(dX,-distTra(1));
else
  distTra(1) = min(dX,distTra(1));
end
if (dY<0) %same for y
  distTra(2) = max(dY,-distTra(2));
else
  distTra(2) = min(dY,distTra(2));
end
if (t>=10*20+1 )%wait 10 seconds for liftoff
  pL3P2 = pL3P2+distTra;
end
if (pdist([pL3P2;pV])<=2)
  repeat(6)=3;
simout2{victim,6}=totalDist(6)/velo;
end

end
end

\textbf{L4P1}-----------------------------------------------
if ((pdist([\pL4P1;pV])>=1)&&(repeat(7)==0)) %L4P1 - if we still not within 1m of target, keep going
angle = \atan{(\pV(1)-\pL4P1(1)) / (\pV(2)-\pL4P1(2))};
totalDist(7)=totalDist(7)+velo;
\textbf{dX}=(\pV(1)-\pL4P1(1));
\textbf{dY}=(\pV(2)-\pL4P1(2));
distTra = [abs(velo*sin(angle)),abs(velo*cos(angle))];
if (dX<0)
  distTra(1) = max(dX,-distTra(1));
else
  distTra(1) = min(dX,distTra(1));
if (dY<0)
    distTra(2) = max(dY,-distTra(2));
else
    distTra(2) = min(dY,distTra(2));
end
if (t>=10*20+1 )%wait 10 seconds for liftoff
    pL4P1 = pL4P1+distTra;
end
if (pdist([pL4P1;pV])<2)
    repeat(7)=1;
    simout2(victim,7)=totalDist(7)/velo;
end
end
%         testTable(round((t-1)/20+1),3:4)=pL2P1; %ALL GOOD!
%L2P2----------------------------------------------------
if ((pdist([pL4P2;pV])>=1)&&(repeat(8)~=3))
    totalDist(8)=totalDist(8)+velo;
    dX= (pV(1)-pL4P2(1));
    dY= (pV(2)-pL4P2(2));
    switch repeat(8)
    case 0 %go vertical til 10 m above victim
        if dX<-10
            distTra=[0,0];
            repeat(8)=1;%go to next stage
        else
            distTra = [velo,0];
        end
        if (t>=10*20+1 )%wait 10 seconds for liftoff
            pL4P2 = pL4P2+distTra;
        end
        if dX<-10
            repeat(8)=1;%go to next stage
        end
    case 1 %go horizontal til x-positions match
        if abs(dY)<=1
            repeat(8)=2;
            distTra=[0,0];
        else
            distTra=[0,0];
            if (dY<=0)
                distTra(2)= max(-velo,dY);           
            else
                distTra(2)= min(velo,dY);
            end
            if (t>=10*20+1 )%wait 10 seconds for liftoff
                pL4P2 = pL4P2+distTra;
            end
            if abs((pV(2)-pL4P2(2)))<=2
                repeat(8)=2;
            end
    case 2 %normal direction
        angle = atan((pV(1)-pL4P2(1)) / (pV(2)-pL4P2(2)));
    end
end
distTra = [abs(velo*sin(angle)), abs(velo*cos(angle))];
if (dX<0) % the difference in x-position and the velocity has to be the same sign
    distTra(1) = max(dX, -distTra(1));
else
    distTra(1) = min(dX, distTra(1));
end
if (dY<0) % same for y
    distTra(2) = max(dY, -distTra(2));
else
    distTra(2) = min(dY, distTra(2));
end
if (t>=10*20+1) % wait 10 seconds for liftoff
    pL4P2 = pL4P2 + distTra;
end
if (pdist([pL4P2;pV])<=2)
    repeat(8)=3;
    simout2{victim,8}=totalDist(8)/velo;
end
end
end
testTable(round((t-1)/20+1),3:4)=pL4P1; % ALL GOOD
testTable(round((t-1)/20+1),5:6)=pL4P2; % ALL GOOD
% Testing purposes below
size1 = size(testTable);
if (size1(1)==301)
    Tests{victim,1}=testTable;
end
end
DOE2

%Design of Experiment 2 - Power Required for Maneuvers
%Purpose: Find the tradespace of battery-drone-weight relationship. Find
%the optimal weight and battery construction for the drone.

%Precondition: Modelv5hex and Modelv5oct are in folder
%Eff ect: Creates a matrix with steady state power values for any payload in
%a range
%Postcondition: Createn a matrix where the (i,j) value is the total power
%from battery of the ith payload and jth flight maneuver

%Independent Variables
%wantedHeight = height the drone will stay at
%wantedVelo = if drone moving, what velocity it should stay at
%wantedAccel = if drone accelerating, what acceleration
%baseWeight = weight with no battery of drone platform
%endtime = how long does the drone simulation last until in the array?
%(if simulation lasts 180 seconds, endtime = 180/0.05 + 1

%Outputs
%maxsteadypower = for all payload amounts in the rows, what is the power
needed
%to hover/stay in constant velocity/accelerate.
%Payload = 0.5, 1, 1.5,..., 5 kg (add payload to base weight of 3.3kg to
get total weight per row 3.8,4.3,4.8,...)
%[power to hover, power for constant velocity, power for
%accelerating flight]

%maxsteadypowerOCT = for all payload amounts in rows, what is the power
%needed to hover/stay in constant velocity/accelerate with a S1000+.
%Payload = 0.5, 1, 1.5,..., 6.5kg + (add payload to base weight of 4.4kg
to get total weight per row 4.9,5.4,5.9,...)

%powers = for each payload amount, there is the amount of watts for each
%motor at every time step

%powersOCT = for each payload amount for the octocopter, there is the
%amount of watts for each motor at every time step

%Random Distributions: NONE

% Create the pathings
%what is your wanted height,velo,and acceleration?
wantedHeight = 4; %4m
wantedVelo = 5; %5m/s
wantedAccel = 1; %1m/s/s

load_system('DoE2PathCreate');
set_param('DoE2PathCreate/wantedheight','Value',num2str(wantedHeight));
set_param('DoE2PathCreate/wantedvelo','Value',num2str(wantedVelo));
set_param('DoE2PathCreate/wantedaccel','Value',num2str(wantedAccel));

%run the simluation once
sim('DoE2PathCreate');

%Reorganize simulation arrays! Waypoints not in good size yet!
t = zeros(6001,4);
t(:,1) = PathHover(1,1,:);
t(:,2) = PathHover(1,2,:);
t(:,3) = PathHover(1,3,:);
t(:,4) = PathHover(1,4,:);
PathHover = t;

t(:,1) = PathConst(1,1,:);
t(:,2) = PathConst(1,2,:);
t(:,3) = PathConst(1,3,:);
t(:,4) = PathConst(1,4,:);
PathConst = t;

t(:,1) = PathAccel(1,1,:);
t(:,2) = PathAccel(1,2,:);
t(:,3) = PathAccel(1,3,:);
t(:,4) = PathAccel(1,4,:);
PathAccel = t;

% Get necessary IV's

% Double check drone properties in Model5hex and Model5oct
% Create payload that will vary
load_system('Modelv5hex');
baseWeight = 3.3; %hexacopter base weight is 3.3kg without battery
powers = cell(10,3); %rows: payload amounts col: Flight manuver
endtime = 3601;
for payload = 0.5:0.5:5
    set_param('Modelv5hex/Linear Model/MassDrone','Value',num2str(3.3+payload));
    for flight = 1:1:3
        waypoint = ''; %case 1
        switch flight
            case 1
                waypoint = PathHover;
            case 2
                waypoint = PathConst;
            case 3
                waypoint = PathAccel;
            end
        sim('Modelv5hex');
        t = zeros(endtime,6);
        t(:,1) = motorpower(1,1,:);
        t(:,2) = motorpower(1,2,:);
        t(:,3) = motorpower(1,3,:);
        t(:,4) = motorpower(1,4,:);
        t(:,5) = motorpower(1,5,:);
        t(:,6) = motorpower(1,6,:);
motorpower = t;
powers{payload*2,flight} = motorpower;
end
end

% Process power

% What is the steady state power?
maxsteadypower = zeros(10,3);
for payload = 0.5:0.5:5
    for flight = 1:1:3
        sumPowers = sum(powers{payload*2,flight},2);
        currPower = max(sumPowers);
        currPower = powers{payload*2,flight}(endtime-1,:);
        if flight == 3
            sumPowers = sum(powers{payload*2,flight},2);
            currPower = max(sumPowers);
            currPower = powers{payload*2,flight}(120*20-1,:);
        end
        maxsteadypower(payload*2,flight)=sum(currPower);
    end
end

% Now the octocopter power

% Double check drone properties in Model5hex and Model5oct
% Create payload that will vary
load_system('Modelv5oct');
baseWeight = 4.4; % hexacopter base weight is 3.3kg without battery
powersOCT = cell(13,3); % rows: payload amounts col: Flight maneuver
duration = 3601;
for payload = 0.5:0.5:6.5
    set_param('Modelv5oct/Linear Model/MassDrone','Value',num2str(baseWeight+payload));
    for flight = 1:1:3
        waypoint = '';
        switch flight
            case 1
                waypoint = PathHover;
            case 2
                waypoint = PathConst;
            case 3
                waypoint = PathAccel;
        end
        sim('Modelv5oct');
        t = zeros(endtime,8);
        t(:,1) = motorpower(1,1,:);
        t(:,2) = motorpower(1,2,:);
        t(:,3) = motorpower(1,3,:);
        t(:,4) = motorpower(1,4,:);
        t(:,5) = motorpower(1,5,:);
        t(:,6) = motorpower(1,6,:);
        t(:,7) = motorpower(1,7,:);
        t(:,8) = motorpower(1,8,:);
motorpower = t;
powersOCT{payload*2,flight} = motorpower;
end
end

%% Process power

%What is the steady state power?
maxsteadypowerOCT = zeros(13,3);
for payload = 0.5:0.5:6.5
    for flight = 1:1:3
        currPower = powersOCT{payload*2,flight}(endtime,:);
        if flight == 3
            currPower = powersOCT{payload*2,flight}(120*20-1,:);
        end
        maxsteadypowerOCT(payload*2,flight)=sum(currPower);
    end
end

%Design of Experiment 3
% Purpose: Find the best beach alternative to saving rip current victims

% Precondition:
% Effect:
% Postcondition:

% Possible Independent Variables
% Rip Current Length (100 - 1000ft) / (30.48 - 304.8 m)
% Rip Current Speed (1 - 8ft/s) / (0.308 - 2.438 m/s)
% Rip Current Width (10 - 200ft) / (3.048 - 60.96 m)
% Victim Escape Method (0,1,2)
%   *Swimming speeds (0 for floating, 0.894m/s for average long open water, 2.4m/s MAX)
% More info: NYC lifeguards minimum is 1.426m/s
% Chosen distribution for fight and swim parallel:
%   normal 1.161m/s, stdev 0.267m/s
% Alternatives
% 1. Lifeguard
% 2. Lifeguard + hexacopter + 1 lifering
% 3. Lifeguard + octocopter + 1 lifering
% 4. Lifeguard + ATV
% 5. Lifeguard + Jet ski

% Possible Outputs
%  Time to reach (sec)
%  Histogram of the difference of times where the differences are:
%    (1) Difference between victim survival and lifeguard reach time
%    (2) Difference between victim survival and alternative delivery
%    (3) Find histogram and probability of losing a rescue using (1) & (2)

% Random Distributions
% Survival for Floating (normal about 80 seconds) stdev:20sec
%Survival for Fighting (normal about 130 seconds) stdev:20sec
%Survival for Swimming Parallel (normal about 180 seconds) stdev:20sec
%Distribution of escape methods: P(0)=0.4, P(1)=0.25, P(2)=0.35

%Utility Functions
% Important factors: probability of failure, reliability with redundant
% life ring

%Cost Benefit
% Cost includes buying price, annaul costs, and # of lives lost (price of
% life)

%% Setup
iterations = 10;
timeI = (0:0.05:180)';
finalSim = cell(iterations,5); %[victim run, A1, A2, A3, A4]
% A1 = altemative 1, A2 = alterantive 2, ....

%% Load and run victim models
load('RipProp');
load_system('VictimModel');

victimSim=cell(iterations,4); %[ripProp, escape, runs, survival time]

%%Generate escapes
% P(0)=0.4
% P(1)=0.25
% P(2)=0.35
for i=1:iterations
   %generate random rip properties
   len   = ripProp(1) = 75m;
   width = ripProp(2) = 30m;
   Vrip  = ripProp(3) = 2m/s;
   mid   = ripProp(4) = 25m from lifeguard;
   ripProp(:,2)=rand*61+30.5;
   ripProp(:,3)=rand*57.912+3.048;
   ripProp(:,4)=rand*2.13 + 0.308;
   ripProp(:,5)=rand*200;
   victimSim{i,1}=ripProp;
   %generate random escape methods
   tempRand = rand; %generate a random number between 0 and 1
   if tempRand<=0.4
      victimSim{i,2}=0;
   elseif tempRand<=0.65
      victimSim{i,2}=1;
   else
      victimSim{i,2}=2;
   end
   set_param('VictimModel/EscapeValue','Value',num2str(victimSim{i,2}));
   set_param('VictimModel/Swimspeed','Value',num2str(randn*0.267+1.161));
% General survival times based on escape method
for i = 1:iterations
    % will not generate times less than 15 seconds (subject to change)
    S = -1;
    switch victimSim{i,2}
    case 0 % fight
        while (S<15)
            S = randn*20+80;
        end
        victimSim{i,4} = S;
    case 1 % float
        while (S<15)
            S = randn*20+130;
        end
        victimSim{i,4} = S;
    case 2 % swim parallel
        while (S<15)
            S = randn*20+180;
        end
    end
    victimSim{i,4} = S-mod(S,0.05);
    finalSim{i,1} = victimSim{i,4};
end

% Alternative 1 - Lifeguard Only
load_system('LGModel');
% velocities of lifeguard are held constant
LGlandspeed = 2.2; % 5mph jog
LGswimspeed = 1.8; % minimum is 1.4

set_param('LGModel/LGland','Value',num2str(LGlandspeed));
set_param('LGModel/LGswim','Value',num2str(LGswimspeed));
set_param('LGModel/jetski','Value',num2str(0));
set_param('LGModel/Xposi','InitialCondition',num2str(0));
set_param('LGModel/Yposi','InitialCondition',num2str(0));
LGSim = cell(iterations, 2); % [LG runs, meetYet boolean array]
for i=1:iterations
    if (victimSim{i,2} ~= 2)
        Vposi = victimSim{i,3};
        sim('LGModel');
        LGSim{i,1} = [timeI,LGposi];
        LGSim{i,2} = meetYet;
        % Find meet time
        finalSim{i,2} = find(meetYet);
        if ~isempty(finalSim{i,2})
            finalSim{i,2} = finalSim{i,2}(1)/20; % time = (index-1)/20
        end
        finalSim{i,2} = 300;
    end
    else
end
finalSim{i,2} = -1; %no need to find time to reach cause people who fight always survive
end
end

%% Alternative 2 - Hexcopter with 1 Ring Buoy
load_system('Modelv5hex');
baseWeight = 3.3+0.18+0.82+2.26+1.62; %weight of drone + weight of cameras + weight of tether system + 1buoy
set_param('Modelv5hex/Linear Model/MassDrone','Value',num2str(baseWeight));
DroneSim=cell(iterations,2);
for i = 1:iterations
    if (victimSim{i,2} ~=2)
        waypoint = victimSim{i,3};
        waypoint(:,4) = 4;
        set_param('Modelv5hex/Linear Model/Integrator3','InitialCondition',num2str(rand*(-200+ripProp(1,5))));
        sim('Modelv5hex');
        DroneSim(i,1) = simout;
        DroneSim(i,2) = meetYet;
        finalSim{i,3} = find(meetYet);
        if (~isempty(finalSim{i,3}))
            finalSim{i,3} = finalSim{i,3}(1)/20; %time = (index-1)/20
        else
            finalSim{i,3} = 300;
        end
    else
        finalSim{i,3} = -1;
    end
end
set_param('Modelv5hex/Linear Model/Integrator3','InitialCondition',num2str(0));

%% Alternative 3 - Octocopter with 1 buoy
load_system('Modelv5oct');
baseWeight = 4.4+0.18+0.82+2.26+2.492; %weight of drone + weight of cameras + weight of tether system + 1buoy + battery
set_param('Modelv5oct/Linear Model/MassDrone','Value',num2str(baseWeight));
DroneSimOct=cell(iterations,2);
for i = 1:iterations
    if (victimSim{i,2} ~=2)
        waypoint = victimSim{i,3}; %victims position over time
        waypoint(:,4) = 4; %wanted height
        %IMPORTANT:, below is where to set operation range -200 = 400m
        %diameter in range. 3 section is -600, and 5 section is -1000m
        set_param('Modelv5oct/Linear Model/Integrator3','InitialCondition',num2str(rand*(-200+ripProp(1,5))));
        sim('Modelv5oct');
        DroneSimOct(i,1) = simout;
        DroneSimOct(i,2) = meetYet;
        finalSim{i,4} = find(meetYet); %determine the time the drone met with victim
        if (~isempty(finalSim{i,4}))
            finalSim{i,4} = finalSim{i,4}(1)/20; %time = (index-1)/20
        else
            finalSim{i,4} = 300;
end
else
    finalSim(i,4) = -1;
end
end

set_param('Modelv5oct/Linear Model/Integrator3','InitialCondition',num2str(0));

%% Alternative 4 - Jet Skis
velocities of lifeguard are held constant
LGlandspeed = 2.2; %10mph ATV , on the slow side
LGswimspeed = 11.18; %average speed of jet ski is say, 15 mph
8.94 = 20, 11.18 = 25

set_param('LGModel/LGland','Value',num2str(LGlandspeed));
set_param('LGModel/LGswim','Value',num2str(LGswimspeed));
set_param('LGModel/jetski','Value',num2str(1));
set_param('LGModel/Xposi','InitialCondition',num2str(150));

JetSim = cell(iterations, 2); [%LG simulation out, meetYet boolean array]
for i=1:iterations
    set_param('LGModel/Yposi','InitialCondition',num2str(rand*(-600+ripProp(1,5))));
    if (victimSim(i,2) ~= 2)
        Vposi = victimSim(i,3);
        sim('LGModel');
        JetSim{i,1} = [timeI,LGposi];
        JetSim{i,2} = meetYet;
        % Find meet time
        finalSim{i,5} = find(meetYet);
        if ~isempty(finalSim{i,5})
            finalSim{i,5} = finalSim{i,5}(1)/20; % time = (index-1)/20
        else
            finalSim{i,5} = 300;
        end
    else
        finalSim{i,5} = -1; % no need to find time to reach cause people who fight always survive
    end
end
SENSITIVITY ANALYSIS

JET SKI VELOCITY

%Sensitivity Analysis on Drone Weight, Pilot Delay, and Jetski Speed
%Comparison

%% Create a victim
load('RipProp');
load_system('VictimModel');

victimSimSS=cell(1,4); %[ripProp, escape, runs, survival time]

%Generate escapes
P(0)=0.4
P(1)=0.25
P(2)=0.35
%generate random rip properties
% len   = ripProp(1) = 75m;
% width = ripProp (2) = 30m;
% Vrip  = ripProp (3) = 2m/s;
% mid   = ripProp (4) = 25m from lifeguard;
ripProp(:,2)=61+30.5;
ripProp(:,3)=57.912+3.048;
ripProp(:,4)=2.13 + 0.308;
ripProp(:,5)=200;
victimSimSS{1,1} = ripProp;
victimSimSS{1,2}=1; %float

set_param('VictimModel/EscapeValue','Value',num2str(victimSim{1,2}));
set_param('VictimModel/Swimspeed','Value',num2str(randn*0.267+1.161));
sim('VictimModel');
victimSimSS{1,3} = [(0:0.05:180),',Vposi];

%General survival times based on escape method
%will not generate times less than 15 seconds (subject to change)
S = -1;
while (S<15)
    S = randn*20+130;
end

victimSimSS{1,4} = S-mod(S,0.05);

%% Part A Drone weight

%Add this to end of DoE4, pretend the victim sim is already out there
%repeat sim for drone with different weights
load_system('Modelv5oct');
OctSenAna = cell(10,4);
n = 1;
for weight = 9.3:-0.5:7.3;
    baseWeight = weight; % weight of drone + weight of cameras + weight of tether system + 1 buoy + battery
    set_param('Modelv5oct/Linear Model/MassDrone','Value',num2str(baseWeight));
    DroneSimOct=cell(1,3);
    waypoint = victimSimSS{1,3};
    waypoint(:,4) = 4;
    sim('Modelv5oct');
    DroneSimOct{1,1} = simout;
    DroneSimOct{1,2} = meetYet;
    DroneSimOct{1,3} = find(meetYet);
    if (~isempty(DroneSimOct{1,3}))
        DroneSimOct{1,3} = DroneSimOct{1,3}(1)/20; % time = (index-1)/20
    else
        DroneSimOct{1,3} = 300;
    end
    OctSenAna{n,1} = DroneSimOct;
    n=n+1;
end

% See time to meet and if victim survived
% % Part B Pilot Delay in Reaction
% % Assume after DoE4, and victimSim is present
% % Repeat sim for delays between 0,10,20,30
% % See time to reach and percent victims saved (?)
% % load_system('Modelv5oct');
% n = 1;
% for weight = 9.3:-0.5:7.3;
%     baseWeight = weight; % weight of drone + weight of cameras + weight of tether system + 1 buoy + battery
%     set_param('Modelv5oct/Linear Model/MassDrone','Value',num2str(baseWeight));
%     DroneSimOct=cell(1,3);
%     waypoint = victimSimSS{1,3};
%     waypoint(:,4) = 4;
%     sim('Modelv5oct');
%     DroneSimOct{1,1} = simout;
%     DroneSimOct{1,2} = meetYet;
%     DroneSimOct{1,3} = find(meetYet);
%     if (~isempty(DroneSimOct{1,3}))
%         DroneSimOct{1,3} = DroneSimOct{1,3}(1)/20; % time = (index-1)/20
%     else
%         DroneSimOct{1,3} = 300;
%     end
%     OctSenAna{n,2} = DroneSimOct;
%     n=n+1;
% end
% % Part C Jetski Speed Differences.
% Instead into DoE 4, Same victims, but we adjust jetski speed and repeat
% for all those victims
n=1;
for speeds = 447:447:2682
    LGswimspeed = speeds/100;
    set_param('LGModel/LGswim','Value',num2str(LGswimspeed));
    set_param('LGModel/jetski','Value',num2str(1));
    JetSenAna = cell(1, 3); %[LG ATV, meetYet boolean array]
    set_param('LGModel/Yposi','InitialCondition',num2str(0));
    Vposi = victimSimSS{1,3};
    sim('LGModel');
    JetSenAna{1,1} = [0:0.05:180]',LGposi];
    JetSenAna{1,2} = meetYet;
    if (~isempty(JetSenAna{1,3}))
        JetSenAna{1,3} = JetSenAna{1,3}(1)/20; %time = (index-1)/20
    else
        JetSenAna{1,3} = 300;
    end
    OctSenAna{n, 3} = JetSenAna;
    n=n+1;
end

DISTANCE BETWEEN TOWERS
%Sensitivity Analysis on Distance between towers
iterations = 100;
TimeReach = cell(iterations,10); %[victim run, distance 1, distance 2, ....
Survive = cell(iterations,10); %[how many survived in run 1, in run 2, ....
% Load and run victim models
load('RipProp');
load_system('VictimModel');
victimSim=cell(iterations,4); %[ripProp, escape, runs, survival time]

%Generate escapes
P(0)=0.4
P(1)=0.25
P(2)=0.35
for i=1:iterations
    %generate random rip properties
    % len   = ripProp(1) = 75m;
    % width = ripProp(2) = 30m;
    % Vrip  = ripProp(3) = 2m/s;
    % mid   = ripProp(4) = 25m from lifeguard;
    ripProp(:,2)=rand*61+30.5;
    ripProp(:,3)=rand*57.912+3.048;
    ripProp(:,4)=rand*2.13 + 0.308;
    ripProp(:,5)=rand*200;
    victimSim{i,1}=ripProp;
%generate random escape methods
tempRand = rand; %generate a random number between 0 and 1
if tempRand<=0.4
    victimSim{i,2}=0;
else if tempRand<=0.65
    victimSim{i,2}=1;
else
    victimSim{i,2}=2;
end
end

set_param('VictimModel/EscapeValue','Value',num2str(victimSim{i,2}));
set_param('VictimModel/Swimspeed','Value',num2str(randn*0.267+1.161));
sim('VictimModel');
victimSim{i,3} = [(0:0.05:180)',Vposi];
end

%General survival times based on escape method
for i = 1:iterations
    %will not generate times less than 15 seconds (subject to change)
    S = -1;
    switch victimSim{i,2}
    case 0 %fight
        while (S<15)
            S = randn*20+80;
        end
        victimSim{i,4} = S;
    case 1 %float
        while (S<15)
            S = randn*20+130;
        end
        victimSim{i,4} = S;
    case 2 %swim parallel
        while (S<15)
            S = randn*20+180;
        end
    end
    victimSim{i,4} = S-mod(S,0.05);
end

%calculate drone position for each one
for distance = 200:100:1000
    run = round((distance-100)/100); %now do it for each victim and store it for {iteration,distance-100 %divide by 100}
    %calculate time to meet and whether victim survived
    % % Alternative 3 - Octocopter with 1 buoy
    load_system('Modelv5oct');
    baseWeight = 4.4+0.18+0.82+2.26+2.492; %weight of drone + weight of cameras + weight of tether system + 1buoy + battery
    set_param('Modelv5oct/Linear Model/MassDrone','Value',num2str(baseWeight));
    % DroneSimOct=cell(iterations,2);
    for i = 1:iterations
if (victimSim{i,2} ~= 2)
    waypoint = victimSim{i,3};
    waypoint(:,4) = 4;
    set_param('Modelv5oct/Linear Model/Integrator3', 'InitialCondition', num2str(rand*(-distance+ripProp(i,5))));
    sim('Modelv5oct');
    % DroneSimOct{i,1} = simout;
    % DroneSimOct{i,2} = meetYet;
    TimeReach{i,run} = find(meetYet);
    if (~isempty(TimeReach{i,run}))
        TimeReach{i,run} = TimeReach{i,run}(1)/20; %time = (index-1)/20
    else
        TimeReach{i,run} = 300;
    end
else
    TimeReach{i,run} = -1;
end
%calculate if they survive
if (victimSim{i,4}>TimeReach{i,run}) Survive{i,run} = 1;
else Survive{i,run} = 0; end
end
set_param('Modelv5oct/Linear Model/Integrator3', 'InitialCondition', num2str(0));
end
REFERENCES


[20] A. Branch, Interviewee, "Drone Collision and Regulations", [Interview], Senior Meeting Room, GMU.


