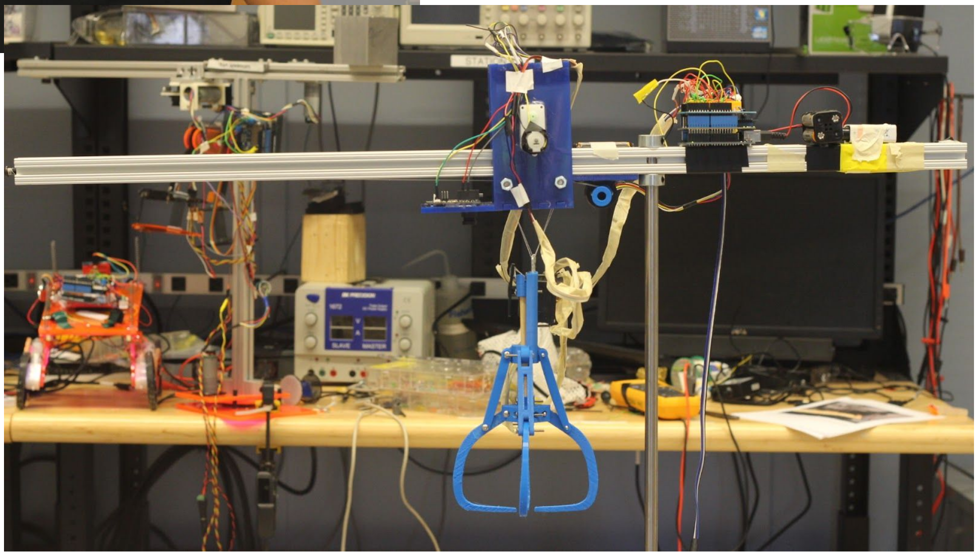
Autonomous Emergency Responders: Final Report



Team Jan Michael Vincent
Camden Jacinto-Schreeder
Calvin Pollard-Vincent
Benjamin Hoertnagl-Pereira

EN.530.421 Mechatronics
May 17th, 2017

1. Introduction

The goal of the final project is to implement either a one or two system solution to search and rescue an accident victim. We are required to have one system be an ambulance that traverses a city, avoiding obstacles on its way to the accident site - in our case, on top of a cliff. Once the ambulance has reached the base of the cliff, we can start a second system to search the rough terrain of the cliff for the victim - the second system being either an off-road responder vehicle, or a crane. Once the victim is found, it must be transported to the waiting ambulance, which will then move towards the hospital. All of this must be completed as quickly as possible, while ensuring the victim's health during transportation, of course.

Initial Concept

We will be using the one system design: one ambulance and one crane. After some initial analysis, we found that the main components that were most important to the success of the project was: speed, reliability, software complexity, mechanical complexity, and size. We focused on the distinction between the off-road responder and the crane since the ambulance was common to both of these options. We agreed that when trading off software/mechanical complexity, we would rather choose mechanical, and so we weight this field higher. The following is an initial trades study based on expectations for the mentioned solutions.

	Speed	**Reliability**	**Soft. Complex**	**Mech. Complex**	**Size**
Responder	🟥	🟩	🟥	🟩	🟩
Crane	🟨	🟩	🟨	🟨	🟥

Figure 1) trade study comparing responder (2 system) and crane (1 system)

2. Approach

Systems Overview

The ambulance will navigate the city at ground level, moving into position at the bottom of the cliff to receive the load of the crane, and then exit with the victim to the hospital. The basic design will feature two front motor wheels, with a rear caster ball to distribute weight. In addition, the ambulance will require the smarts for avoiding substances, most simply done with distance sensors. Finally, the ambulance will include some basket structure to hold the victim within.

The crane will be positioned at the edge of the cliff, and will include a boom that is long enough to reach any section of the search space (20cm radially). There will be a moveable carriage fitted with a Pixy camera that will complete a sweeping search for the victim, and move the claw over its position.

Process Overview

1. Ambulance navigates city with obstacle avoidance scheme
    a. Move into starting position inside city, over first ramp
    b. Try red path, keeping track of direction and distance moved
        i. If blocked - move to starting position, transition to c.
        ii. If not blocked - transition to d.
    c. Take blue path, guaranteed to not be blocked
    d. Move forward util wall detected, then choose exit 2
    e. Upon reaching the base of the cliff, ambulance will broadcast signal to crane via XBee to begin search of victim

   f. Crane sweeps maximum radius and searches for color code matching that of victim and/or stretcher
   g. If not found, attempt second pass smaller searching radius
   h. Repeat until found
2. Once found, crane will grab stretcher with claw, move to where ambulance should be located, lower victim based on color code in basket of ambulance
   a. Lower claw to object, continually checking that object is in center of Pixy frame
   b. Once preset distance is lowered, pull up claw and allow grip of stretcher
   c. Move claw over side of cliff to directly on top of ambulance based on Pixy frame reading
   d. Lower claw until victim in basket, trigger release wire
3. Ambulance moves off to the hospital

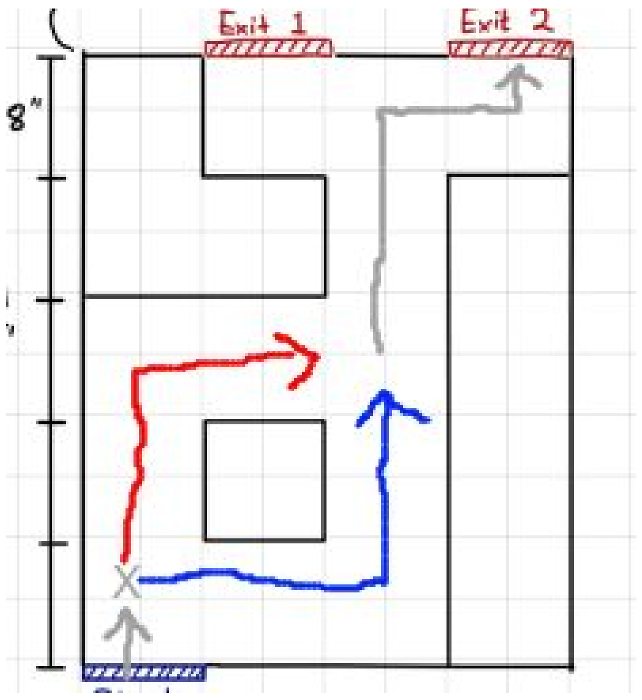


Figure 2) map of city with potential paths: red indicates the first path option, blue indicates second path option, gray indicates the path that will be traveled regardless of intermediate path option

3. Design

Ambulance Design

In designing our ambulance, we focused on 5 core components that needed to be included: chassis, movement, avoidance, transportation, and communication. For the chasses, we opted for a laser cut acrylic base, given its strength and ease of manufacturing. For movement, we decided to use two DC Pololu motors with encoders, so that we could keep exact positioning of our robot for any point of the city navigation. While the encoders were sufficient for forward and backward tracking, they lacked in their consistency for turning. To improve this, we used an IMU for precision turns and forward direction correcting to make sure we drove exactly square in all directions.

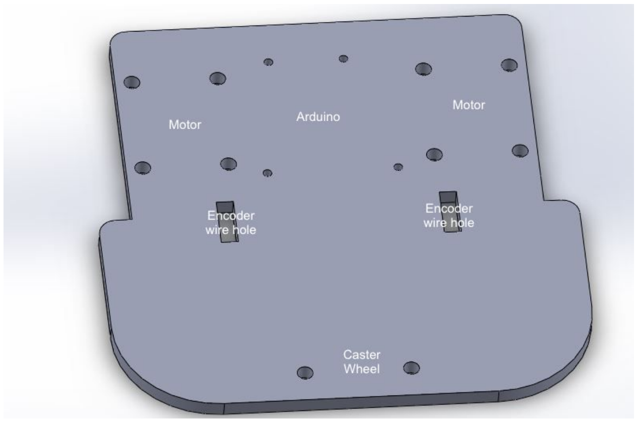


Figure 3) ambulance chassis model

The smarts required for the obstacle avoidance included 5 IR sensors, 3 of which pointed forward and 2 of which were orthogonal to the forward direction. These were housed on an acrylic front panel. For transportation, we laser cut a simple box that would act as a cradle for the victim and the stretcher. Later, we added a protrusions to catch any slight deviations from the center of the basket. Also, we positioned a color label in the center, which would be used for location by the crane. Finally, in order to allow communication between the ambulance and the robot, we included an Xbee and Arduino Xbee shield.

Our avoidance scheme was greatly inspired by Lab 5. Essentially, we repurposed our code so that we could specify a given distance to move forward while avoiding obstacles. We also added a condition that if we cannot avoid an obstacle by going left or right, then there was a complete blockage and we must backtrack. Our obstacle avoidance function takes in an integer to determine the distance to move forward, and increases a counter while it does, so in the case of a complete blockage, we can recurse exactly the distance that we moved forward, allowing for a robust state machine. The details of the individual obstacle avoidance is discussed next.

Obstacle Avoidance

We chose to approach the control issue of obstacle avoidance by modeling a finite state machine. We begin in state 0, the forward moving state. From here, we check the forward facing sensors at uniform time increments to check if an object is in the path of the robot. If no object is seen, we remain in state 0; else, we move into state 1. State 1 chooses which direction to turn the robot based on readings from the side sensors - if the left sensor read a larger distance, that turn will be made because it is more likely that an opening is available. Of course, this is not a perfect reasoning to choose direction, but we figured that it was only useful criterion we could judge and would be better than simply choosing a direction at random. After the sensors are read, we move into the corresponding state 1a or 1b depending, which physically turns the robot 90 degrees based on the encoders on our motors. Once the turn is completed, we move into state 2, which moves "forward" (at this point, either left or right with respect to the final target), while the side sensors detect an object. As soon as the sides of the robot are free, the robot must move a preset distance farther to ensure that the back of the robot can easily clear the object, it is ready to turn back in the direction of the final destination. If the robot detects an object in front of it at any point while in state 2, it will make a 180 degree turn and continue in state 2. This allows for the robot to avoid getting stuck after an initial direction choice. Assuming that we will never encounter a situation in which our robot will be boxed in, the ability to turn around and continue in state 2 ensures that any obstacle setup can be cleared. Once the object is cleared, we can enter state 3, which just turns 90 degrees in the opposite direction that was turned in state 1a or 1b, and finally transitions back to state 0 in anticipation of another object.

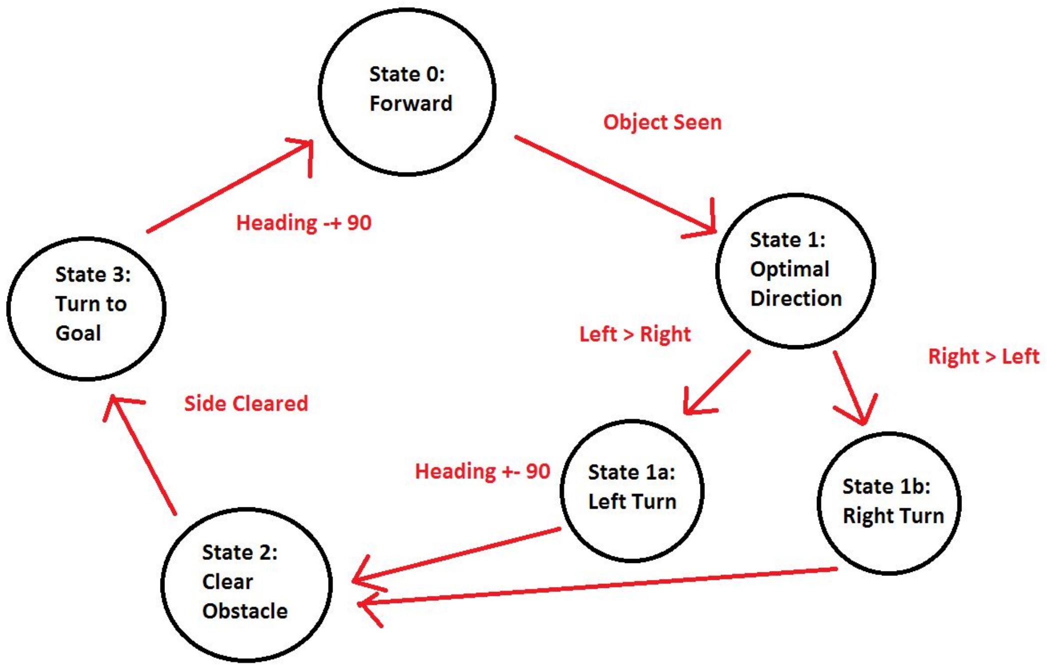


Figure 4) obstacle avoidance finite state machine

Crane Design

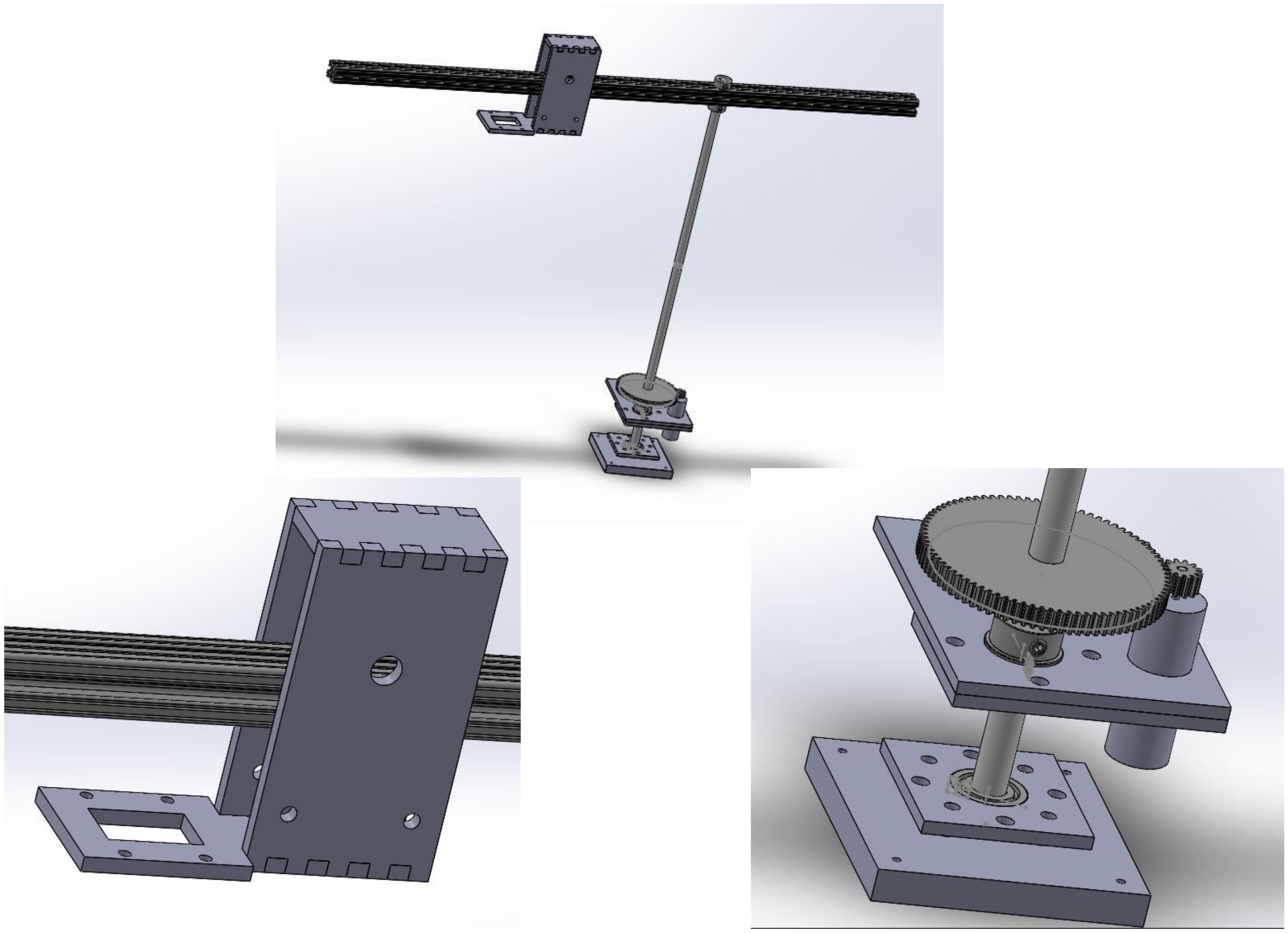


Figure 5) crane CAD models of complete (top), carriage(left), and base (right)

The mechanical design is above. The overall photo shows the 80-20 Al beam and the ½" vertical shaft. The bottom right picture shows the mountings for the bearings that allow the beam to rotate freely as well as the DC motor's gearing and mounting. What is not pictured are the bolts that go vertically through the holes or the bolts that mount to the bottom plate. The bottom left photo is of the carriage that the pixy is mounted to as well as the dc motor and wheel that drove the claw forward and back. Many of the parts were cut out of acrylic because of the ease of cutting and since we didn't need to meet too many mechanical requirements. We designed a claw that precisely picked up the custom stretcher we created to hold on tightly and keep it level. The pixy was set up on a separate power supply and used I2C for communication because of digital pin scarcity and the power draw of the pixy. The shaft was lengthened from the initial 1 foot to allow the pixy to see the full field of view.

As discussed above, the electrical systems of our crane design were composed of two pololu encoder motors to control the claw height spool and the theta rotation, a standard DC motor to control the R-position of the claw, a servo motor to control the position of the claw, a pixy to provide data on object location, and an Arduino to provide overall control. Power was provided by three independent battery banks, one 9V for the Arduino logic, and two 4-AA 6V packs, one each for the pixy and the motors.

To accomplish its task, the crane code breaks sub-tasks down into methods for ease of control flow. The first is the refresh(int) method, which takes an integer of the color code and updates the x and y coordinates of the object it is trying to find. The pixy is communicated with over I2C, which requires a special connector cable, and uses the analog inputs of the Arduino. The refresh method is then used by our two most important functions, the findR and findTh functions, which locate the object passed to them in the r and theta directions, and position the crane claw to align with the object. Their control flow is easy, simply finding the position of the object and then moving the crane slightly until the object is within acceptable range. Additionally, we have a few minor supporting functions for raising and lowering the claw (distance calibrated using the encoder on the spool motor), opening and closing the claw (controlled using simple PWM for the servo), and more basic functions to move the crane arm slightly clockwise or counterclockwise, or forward and backwards.

Using these functions, our final control flow is fairly simple: Wait to receive the signal from the ambulance, and then locate the baby in the Theta and R directions. Drop the claw and pick up the baby, rotate the arm a hard coded 90 degrees to place the ambulance in the field of view. Locate the ambulance, and drop the claw to place the baby in the basket. Move the claw out of the way and signal the ambulance to move forward. The total code for the crane can be seen in the appendix.

4. Potential Improvements

Ambulance Improvements

Perhaps choosing an angled sensor scheme would have been more efficient in cutting down the total distance covered by the robot since it would not only move at right angles. This could greatly improve speed because of instead of snaking around obstacles it could just slightly adjust out of the way and continue driving forward at full speed. On the other hand, this would make monitoring exact position very difficult, since not all distances are axis oriented. Also, we could have used side sensors in detecting complete blockages. Instead of trying to avoid an obstacle left and right before determining that here is a complete blockage, we could instead simple determine if it would even be possible to pass given our lateral position in the street of the city. Since the width of our robot was about half the width of the street, we would only be able to avoid obstacles if we are positioned all the way to the left or the right of the path, which could be checked by the back sensors.

Crane

The main issue with the crane, as expected, was inconsistencies with the Pixy. Instead of relying on the ambient light in the room and needing to calibrate before every run, we could have used extra lighting, for example flood lights to ensure equal conditions for all runs. In addition, the carriage at times had issues breaking the load of the claw since it was dragging along the boom while moving. The carriage could have been better mounted to maintain levelness in the Pixy. However, the slanting didn't cause any problems because it was a consistent amount of tilt, and the wide angle of the Pixy with low focus allows a great field of view.

5. Conclusion and Discussion

Overall, our system performed very well and we are all quite pleased about it. Compared to strategies used by other teams, we feel that the the crane approach provides the smoothest transition of the victim from the cliff to the ambulance, and inflicts the least amount of damage. We wish we had considered using floodlights to help our pixy recognize color codes more easily. Despite this our system performed really well it just needed to be recalibrated frequently as the sun rose or set. Both parts were incredibly reliable (except for on the competition day) and carried out the task slowly but consistently. We definitely could have made our systems run much faster but we strived for consistency over speed.

As far as lessons learned, aside from the usual power issues, we noticed some interesting properties with our IMU. In calibrating the device, we were accidentally touching the open leads with our fingers, which would cause for occasional mis-calibrations which we assumed were due to some external interference. Advice for future teams: when calibrating an IMU keep your fingers a good distance away from the chip, if not the removal of your fingers will make the calibration unreliable. Also, when using the Pixy, we found that we got best results by keeping the field of view blurred, so we highly recommend trying different amounts of blur depending on the color signature being used.

This experience was not only great from an engineering perspective, but also from a perspective of teamwork. We felt very confident in splitting the workload over the three major components of electronics, software, and mechanical, since each one of us was specialized in one of these fields. In the end, we worked very well together, and it shows through our product.

6. Appendix

All code can be found on our github repository:

https://github.com/bpereira615/Mechatronics-Final

Our presentation video can be found on YouTube:

https://youtu.be/D43aNw1wTDE