Author: Zhiqi Gong, Zixu Lu: Electrical and Computer Engineering, Carnegie Mellon University

Robot Suitcase

***Abstract*—Robot suitcase is a robot capable of tracking the user, following the user and liberating the users from carrying the luggage all the time. Robot suitcase could help customers improve their travel experience. We integrated depth camera and Apriltag detection component to track the user and collect data from environment. We changed from tracking the heels to tracking the entire person. Our computing unit will process the data and determine which direction we should go and which path we will take. The control module will execute the command onto the Mecanum wheels.**

***Index Terms*—Control, Robot, Computer Vision, Path Planning, April Tags, Mobile Robot, Simultaneous Localization and Mapping(SLAM)**

I. Introduction

Robot suitcase is a smart carry-on suitcase robot that tracks the user, follows the user anywhere and liberates user from carrying the luggage all the time. This project could be further expanded and commercialized for travels today. With industrialized battery design and integration, robotic suitcase could help customers improve their travel experience by liberating their hands and keeping them away from carrying luggages. Besides, we believe that a fully autonomous hands-free suitcase could also assist disabled travelers and parents traveling with small children. Similar products on the market used beacon-based localization which requires the user to carry a beacon while using the suitcase. Other competing technologies include lidar as one of their sensor which is a costly solution. Our suitcase adopted computer vision and Apriltag modules to perform the environmental perception. Compared to the competing technologies, our solution is more cost-effective and focus more on the software system.

Our goal is to create a smart luggage which tracks and follows the user. One of our main assumptions is that our user will mainly use this robot suitcase at the airport, and most of the modern airports are flat and obstacle-free. Our robot luggage should mainly deal with the tracking and path planning but not including avoiding obstacles. The preliminary requirement for the suitcase is that it could track and follow the path the user took so that it will not lose track and encounter any other obstacles. Our robot should be able to follow a moving user, which means that the robot should determine the tag within 2 meters and follow the user within the speed of 1.5 meters per second. The robot must find the shortest and the optimized path in less than 1 second. The camera should process the images at least at a rate of 3fps.

II. Design Requirements

In order to reach the requirements, we design a series of test metrics to ensure that our robot can correctly track and follow the users. We design a series of test metrics to test each of the modules individually to make sure each of them are working well.

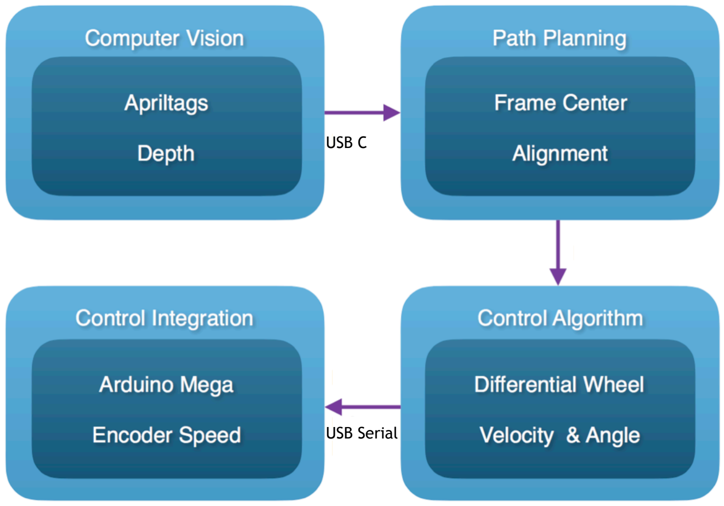
For the computer vision, first we need to make sure our detection works well. We are going to test our computer among group of 5 tags. And the robot will be able to detect the correct user among 5 different tags and track the correct user. This is designed to ensure that no one could interfere with the suitcase and confuse the suitcase with a similar apriltag. Since the robot is supposed to keep a certain distance from the user. We also need to test if the robot could determine the tag within 2 meters or 3 meters. Secondly, in order to track the user, the camera also need to determine the distance from the user. The precision should be over 90%. Based on our assumption that the robot suitcase would be mainly used at the airport, we do not require the suitcase to detect any non-human targets such as obstacles.

For motion planning, we build test metrics to test whether our robot could find the optimized and shortest path. And we could follow the user with smooth path without any sharp turns. For the test, we will neglect all the obstacles because we believe airport is supposed to be flat and obstacle-free.Therefore our path planning test metrics will not include any obstacle avoiding optimization.

For control and integration, we would like to test if our robot could follow a pedestrian. The average walking speed of a person is around 1.5 meters per second.Our robot do not need to go as fast as human. If human suddenly halt, our robot need to slow down or even stop. We will also test if our speed increase and decrease is linear. Moving slightly slower the human and then catch up when the user stop is the optimized way to deal this issue. The test metrics for the control is to move around or slightly lower than 1.5 meters per second. We would also test if the robot can do a wide turn around 120 degrees. This is a normal range of the turn of the walking pedestrians. For braking part, we will test if the control part could brake with trapezoidal speed reduction.

For middleware, we will need to test this before we run the final test. The middleware is the subsystem used to communicate data. Our data are fed in from the camera to the Jetson, and from Jetson to the Arduino Microcontroller where PID is adopted. We will make sure the image the Jetson gets reaches at least 3 fps so that the Jetson could give a continuous series of outputs to the Arduino. And by listening to the serial port, the Arduino should receive all the results from the Arduino. This must reach 100% correctness.

Overall testing will be performed once we passed all the test metrics. Overall testing will include following the moving human, reaching a speed of 1.5 m/s, braking when user is breaking, turning when the user is turning and differentiating the right tags among a group of tags.



i. System picture.

III. Architecture and/or Principle of Operation

Our entire system is embedded into a normal size carry-on luggage. We mount the Intel Real-sense depth camera on the front edge of the luggage facing the back of the user. We change the normal wheels from the luggage to two Mecanum wheels with certain amount of angle (about 30 degrees) to create omni-direction control, and use another omni-wheel to balance the robot. We will separate a thin layer inside the luggage to fit in our circuit and control part. We will mount the Jetson TX2 board inside and connect input signals to it. We will find the best angle to compensate its initial error, and then install it in that layer. We are using a large output power bank to power our electronic part, so the power bank will also be mounted inside the layer. The user part will just be a printed April tag sticked to the back of the user.

Upon the start of the following, the sensors will first collect data respectively and send data to the center controller - the Jetson TX2 board. Then the board will run an algorithm of detecting the April tag and then determine the distance from the camera to the tag using stereo vision. With information and data being analyzed and processed, the jetson board will issue a new task to compute the path to follow using cubic spiral path planning algorithm. The output will then be process with differential driving algorithm. The differential driving system will output the speed of the two motors respectively. Then the motor will drive the robot suitcase to the desired location with PID control and trapezoidal speed control.

IV. Design Trade Studies

A. Design Trade of Computerv Vision I

The first trade in our project is the speed that the luggage can travel to follow people versus the speed of computation after collecting the data. The ideal range of image processing is about seven to fifteen image frames per second. The higher the image frames processed, the more computational power is needed. However, if the rate is higher, the controlled driving part will be much smoother, so the speed can be higher. However, it is hard to maintain the computation power at a high level, since multiple processing will cause drop in the computational power, and we are aiming at about 12 images per second processing rate.

B. Design Trade of Computerv Vision II

The second trade in our project is whether to use YOLO to detect the user. By using YOLO, we are mostly likely to be guaranteed a high percentage accuracy. However, it takes time to train and might not be quickly applicable to the first time use if the user is in a hurry. Instead, we can stick an April tag to the back of the user and use apriltags api in openCV to find the apriltag and find the distance and direction afterwards. However, the accuracy is not always guaranteed if the user decides to do quick turns or starts to walk faster or wobbly.

We decided to adopt April tags.

C. Design Trade of Control System

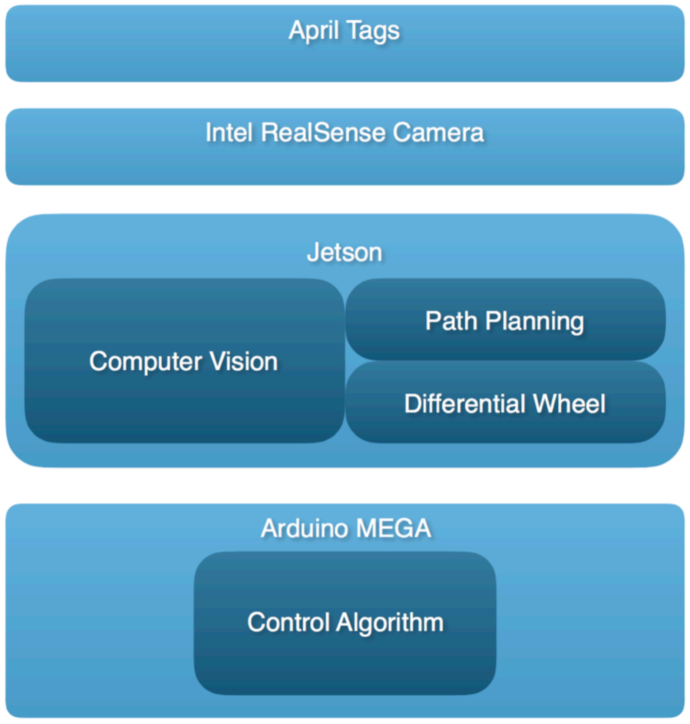
The third trade is when we tried to integrate the batteries into the suitcase. We underestimated the influence caused by the batteries. In order to drive the the suitcase, we need to instal mid/high-power torque motors which consume more power and require more voltage than we expected. Due to the budget, we cannot buy light-weight high output battery.

Instead, our system now contains two heavy batteries. This has caused several issues for the wheels and PID. Previously he wheels and the motor were mounted directly on the bottom of the suitcase. However, when we added more loads to the suitcase, we have to redesign a shaft. We designed a wood shaft and support which prevent the motor from being crushed. Another issue is that adding more loads causes the PID to compensate more power onto the motor. However, at this point, we have reached the limitation of the motor, and therefore decrease the overall speed of the robot.

D. Design Trade of Differential Drive

Last but not lease, we need to make the suitcase move and turn. Unlike vehicles, mobile robots do not have wheels they cannot turn their front wheels so that the rear wheels will follow the path it created. After doing some research, we decided to adopt Differential Drive. It requires that we can control two of the three wheels. And the third wheel will become a free wheel. The turn angle could be determined by the differential speed of two wheels. We believe that this will be the best solution for our projects. It could help fully control our robot, reach the design requirement and mostly importantly save budget for other parts.

i. **Robot Suitcase.**



i. **System Layer**

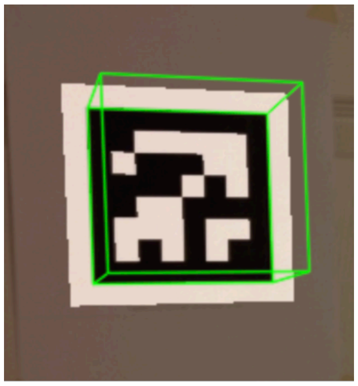
V. System Description

The overall system layer is shown in the picture above. We will take a look at each of them in this section.

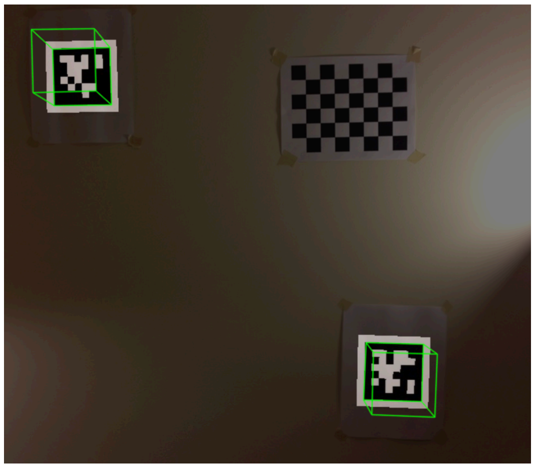
A. Computer Vision

First, we get a video stream from the camera, and then we capture one image per five frames, to both save computational power and extend the use of the battery. We locate the tags stuck to the back of the user by Apriltag detection. It can give us the center of the tag located in the image frame, so we can use trigonometry and depth information to find the actual x and y location in the world frame. We extend the detection to detect multiple tags per image, so we can make sure that we are never going to lose track of the user. We also calculate the angles of the turn by tags' distortion with the homography matrix of the tags, so we can predict any turning movements by the user. Once we predict a wide turning movement by the user, the suitcase turns in advance to prevent losing the sight of the user.

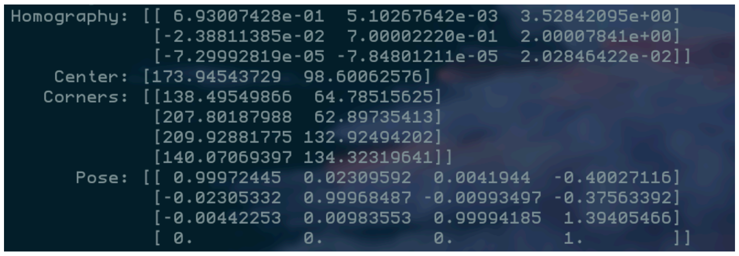
After detection is done in Jetson board, it calculates the frameshift and tries to find the best cubic spiral path to align the user in the center of the frame. After it found the best path, it also calculates the distances each wheel needs to travel based on the radius of the wheels and the distance between the two wheels. Since it sends signals to the ROS communication system 5 times per minute, it calculates the speed to input to wheels using time-distance equation. After these calculations(done within one millisecond) from Jetson board, we will use ROS to communicate with Arduino to input speed values to the wheels and do PID from Arduino.



i. **Apriltag Detection with Homography**



i. **Apriltag Detection with Noise Control**



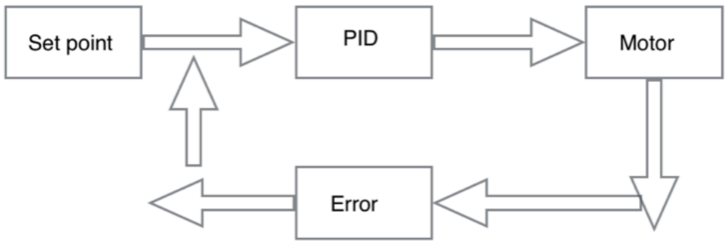
```
Homography: [[ 6.93007428e-01  5.10267642e-03  3.52842095e+00]
             [-2.38811385e-02  7.00002220e-01  2.00007841e+00]
             [-7.29992819e-05 -7.84801211e-05  2.02846422e-02]]
    Center: [173.94543729  98.60062576]
   Corners: [[138.49549866  64.78515625]
             [207.80187988  62.89735413]
             [209.92881775 132.92494202]
             [140.07069397 134.32319641]]
      Pose: [[ 0.99972445  0.02309592  0.0041944  -0.40027116]
             [-0.02305332  0.99968487 -0.00993497 -0.37563392]
             [-0.00442253  0.00983553  0.99994185  1.39405466]
             [ 0.          0.          0.          1.        ]]
```

i. **Homography Matrix**

B.Control System

Our implementation of the controller for the wheels is PID control. PID controller is a control loop feedback mechanism that takes in set point and reads data from sensors to adjust itself in order to make the output closer to the set point. We believe that PID control is a perfect use case in this low-speed obstacle-free mobile robot.

The keys to this PID control are set point and the feedback. We use this PID module as a control system for our torque output and speed control.



i. PID Block Diagram

We also take certain types of sensors data into consideration because we need to give it back to the controller so that controller could adjust itself based on the disturbance. Since the controller is driving two wheels, the set point will always be a wanted speed. The speed command then goes to the motor. With the help of the encoded motors, we read the data from the speed sensor integrated into the motor. The difference between the actual speed and the wanted speed will be the error of the system. The is error then goes back to the controller as a feedback. The control loop will compensate this error and keep the actual speed close to the set point.

PID controller consists of three parts: Proportional controller, Integral controller and Derivative controller. Staring from the proportional controller, we set the gain of the control system, C(s), to be a single value. We then witness high overshoot and steady-state error from the system. Theoretically, it appears that none of our design requirements can be met with a simple proportional controller. This is the start point where we want to lower certain errors based on the feedback. Then we added Integral controller and Derivative controller to lower the error.

$$C(s) = K_p + \frac{K_i}{s} + K_d s = \frac{K_d s^2 + K_p s + K_i}{s}$$

After adding several loads to the suitcase, we spent hours retuning the parameters in order to make sure that this works well with the loads.

C.Differential Drive

Differential Drive is a perfect solution for turning and speed in this low-speed obstacle-free mobile robot. It controls the direction of the robot by differentiating the speed of the two wheels.

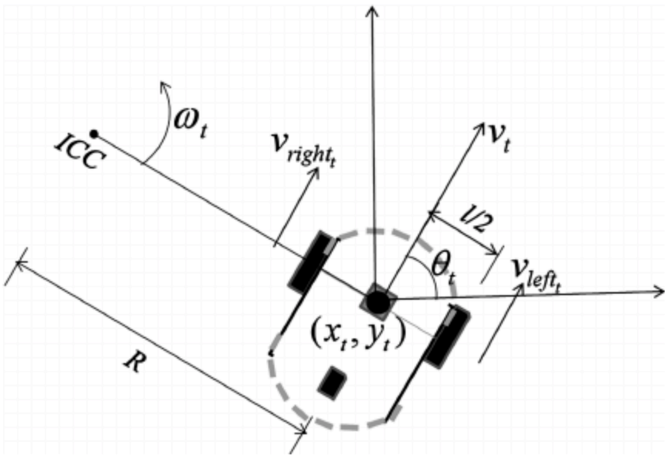
After receiving the X and Y coordinates of the target from the path planning module, we can calculate the lineal velocity VL and VR of the wheels based on the radius of the wheels. By knowing the length between two wheels, we can find out the speed of two wheels respectively. V is the speed and a is the orientation. Vx and Vy are speed of the left and right wheels.

$$V = \frac{V_L + V_R}{2}$$

$$\alpha = r\frac{V_R - V_L}{l}$$

$$\dot{V}_x = V\cos(\alpha)$$

$$\dot{V}_y = V\sin(\alpha)$$



i. Differential Drive Robot.

D. Hardware

The hardware we used for computer vision is intel Real Sense Depth camera which not only captures the image but also tells us the depth.

The second layer are executed on Nvidia Jetson TX2. This is am embedded hardware platform designed for robots, autonomous driving vehicles and machine learning. It is more powerful compared to raspberry pi. We chose this because of its high performance. But on the other hand it is more complicated than we expected. Intel real sense did not work really well on this platform because of compatibility issues.

The third layer is the control-integration system. The Jetson will communicate the wheel speed with the integration system. The data will be processed and executed on Arduino. In this layer, we used Arduino Mega 2560 and Pololu Dual TB9051FTG Motor Driver Shield to drive the motor. The motor we bought is also from Pololu, and it has bulit-in hall effect encoders. By counting the ticks of the encoders, we could know the current speed of the wheels. The encoder input is fed through digital interrupt input on Arduino. We need more than two interrupt pins for the encoder. This is the reason we chose Arduino Mega 2560 instead of UNO. The shield is designed for dual motor control especially for mid/high-power motor in our case. Its corresponding Arduino library makes it easy to control. We also adopt MyPID library which provides us robot PID modules. We can define the parameter and the library will calculate the output and error on its own.

i. Nvidia Jetson TX2.

ii. Pololu Dual TB9051FTG Motor Driver Shield(Above)

E. Middleware / Communication

We implement our communication system with the python interface and we wired up Jetson and Arduino. We use the Jetson as the master server, and all speed information is packed up in a string. After packing up all information, Jetson writes in the serial port to the Arduino, where the string got parsed into speed information. Arduino will never try to communicate back to the Jetson board since its only function is to run the speed given on the wheels and using PID to control the robot within the desired speed range.

VI. Project Management

A.Schedule

See Attachment

B.Team Member Responsibilities

Zixu Lu's primary responsibilities are computer vision and path planning. He is working on the camera integration, camera depth and tracking systems. Then he will be working on the path planning algorithms. His secondary responsibilities are middleware design and implementation and testing.

Zhiqi Gong's primary responsibilities is PID control. He will be working on calibration and testing. He is also the mechanic guy who designs and integrates all the parts including wheels, motors and arduino. Then he will be working on designing and tuning the PID controller. His secondary responsibilities include communication between modules.

C.Budget

See attachment

D.Risk Management

One of the risks we have encountered is the battery issue. Our original plan was to use a 12v 24k mAh power bank to drive the whole system. However, it may cause burden for the power bank to drive the jetson board and the motor at the same time. When the motor output the torque, it will drain large amount of voltage from the power bank and bring down the voltage for the jetson board. Once the jetson board sensed the drop in the voltage, it will shut down immediately. We have noticed this issue came, and we tried several power bank and hope to find the optimized solution. Due to the limitation of the budget, we cannot buy industrial battery. The most cost efficient solution is a power bank with higher output.

As we load the suitcase with heavy batteries, the PID became abnormal. We tuned several times to set a threshold for the motor output so that it is more powerful. Since the maxim torque the motors can output is fixed, we decreased the overall speed of the robot suitcase to make sure that the acceleration is linear. However this become more complicated because the tracking algorithm needs to keep the tag in the middle of the frame. The speed of the PID is not compatible with the speed of image processing. This has caused many issues to our testing. Therefore we decree the processing speed of the overall system in order to wait for the PID.

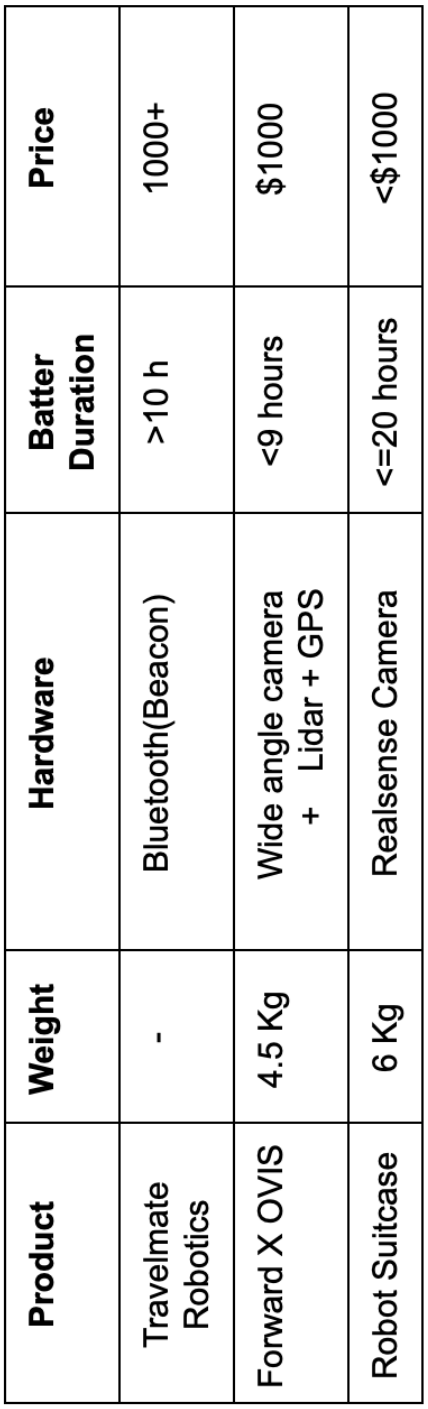
The third risk we are facing is the detection system. Before we are using the detection system and the machine learning to recognize the heels and the mark we attached to the heels. But this will bring other issues. Tags and heels are hard to detect in the image. People are moving with both heels but one of them will remain stationary while the other is moving. If the robot track one of the heels, the movement of the robot may not be smooth as it will encounter constant stop and move. Besides, if we track the human body, the accuracy is much lower than we expected. Therefore our team try to avoid these issues by adopting Apriltags.

The last risk is the overall system stability. We have experience several hardware and software failures. It may happen on any platform even Nvidia Jetson board. Our solution is that we always have a backup plan so that we can have a minimal deliverables when we do demo.

VII. Related Work

Some startups have been working on the idea of autonomous suitcase. And a few of them have successfully made the prototype and went to the market. One of the product aiming for high accuracy is using lidar, which adds cost to the development. Compared to the camera, lidar is more expensive choice of sensor. Although it is more accurate than many camera and computer vision algorithm, it also require more power from the power bank. Another product uses the beacon to locate the users which requires the user to carry extra sensor. We hope that the user of our ready-to-go product will not be worried about carrying any extra burden. Another consideration is about product system upgrade. Although our camera is programmed to determine and track the tag, we the developers could improve and invent new algorithm such as object detection and avoidance. Without installing any hardware, a software update will help our customer enjoy the latest technology. We believe that solving this problem by computer vision is the most cost-effective means and is the best replacement for existing solutions. Current price of this prototype may be higher than the others. But this is prototype. If we can commercialize the product, we can definitely have a better control of the overall cost.

See attachment on the right for more details.



Product	Weight	Hardware	Batter Duration	Price
Travelmate Robotics	-	Bluetooth(Beacon)	>10 h	1000+
Forward X OVIS	4.5 Kg	Wide angle camera + Lidar + GPS	<9 hours	$1000
Robot Suitcase	6 Kg	Realsense Camera	<=20 hours	<$1000

VIII. SUMMARY

Our design report has been improved from our initial proposal, design review and project. We made several changes including different software and hardware solutions. We came up with several possible solutions and compare them. And the solution we discussed in this report reflected our project scope as well as how we tried to execute the plan.

A.Future work

Currently we are still working on the project. If we are going to working on this beyond the semester, we need to rethink about several aspects. First we do need a well-designed battery system. For now the power bank is not a sustaining solution. It may require the user to charge the power bank for hours and the actual battery life is short. Besides, we need to redesign the suitcase in order to minimize the weight of the suitcase itself. Our design of the suitcase neglect the product experience. At this point we do not have a well-planned wiring plan and optimization including the position of the jetson board. If the robot is going to the market, we need to deliver a well-designed product. Another thing we will be working on is the obstacle detecting and avoiding. We hope our user can bring our product out of the airport and help them whenever they are traveling. This requires the robot to be able to handle all kinds terrain and obstacles. In order to achieve this, research and development is necessary.

B.Lessons Learned

Robot Suitcase is project that combines software system (computer vision, robotics system), circuits (motors) and signal process (control). Since we are a team of two, and both of us are ECE and specifically signal processing background. It took us a long time to set the software system such as system configuration. Zixu spent two weeks to fix the compatibility issues between Jetson and Intel Real Sense. And Zhiqi spent several weeks design the hardwares and mechanical parts. This is a precious learning experience for both of us. If we can have another teammates who happen to have more experience in software, the issues will become much more trivial.

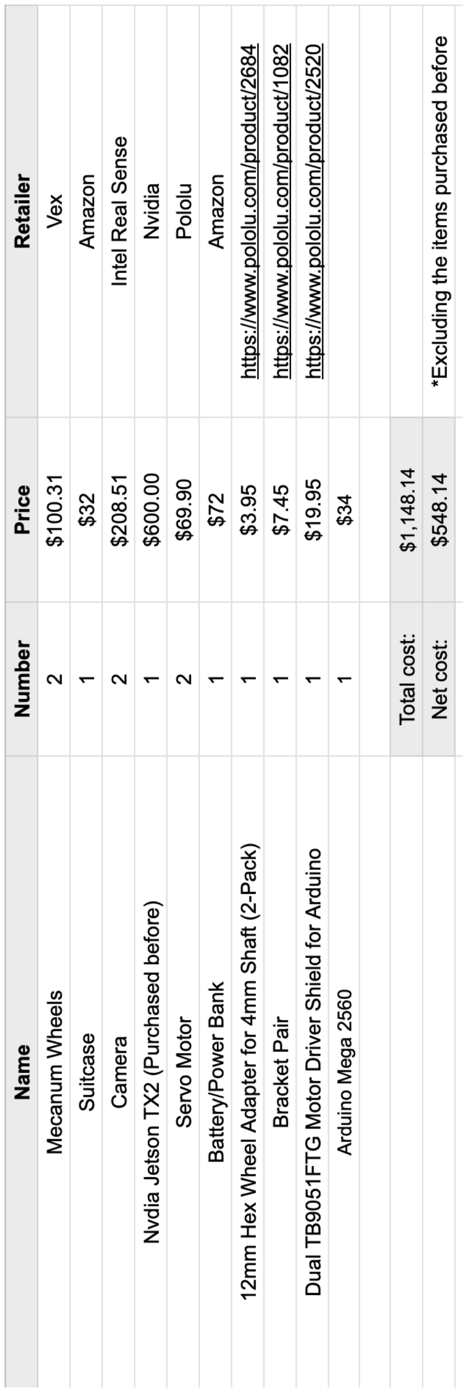
Secondly, we believe that purchasing hardware and parts from the same supplier will make our lives easier. We bought mechanic parts from different suppliers, and the size of the parts may not be compatible. Therefore we made some modification to the parts.

Human movements are much complicated than we imagined: Angle and speed will cause distortion. This will make the robot believe that the human is not at the place where he or she is, and therefore send false results to the control.

Last but not least, try to find a simple solution instead of a complicated one. Many engineering issues could be solve by the most straight forward solutions.

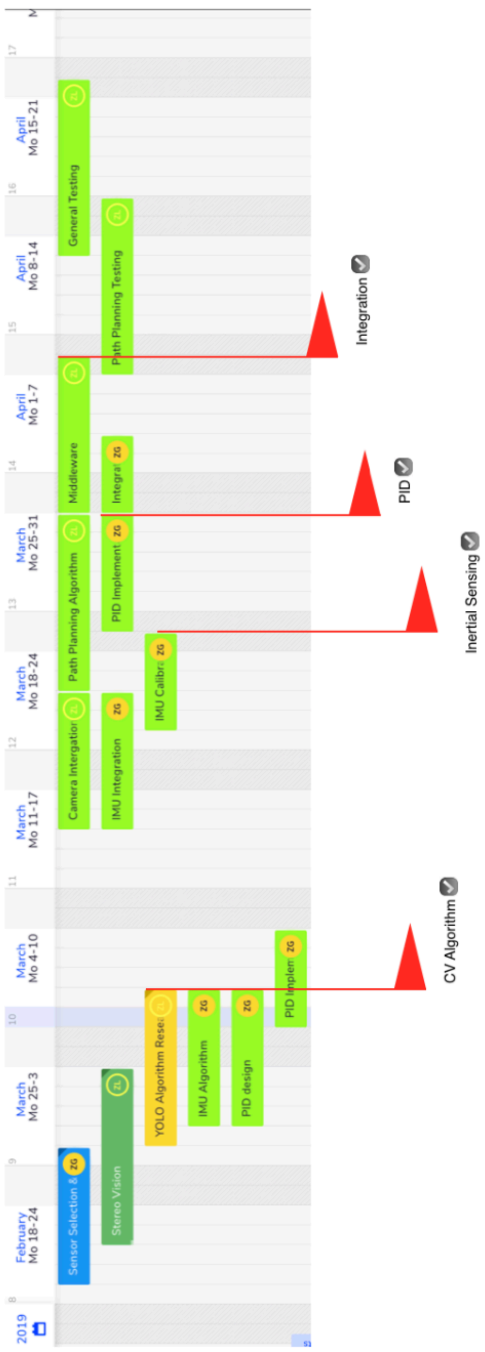
REFERENCES

1. AprilTags Visual Fiducial System,The APRIL Robotics Laboratory at the University of Michigan investigates Autonomy, Perception, Robotics, Interfaces, and Learning, https://april.eecs.umich.edu/software/apriltag.
2. Kinematic model of a differential drive robot, enesbot.me http://enesbot.me/kinematic-model-of-a-differential-drive-robot.html
3. You Only Look Once: Unified, Real-Time Object Detection, Joseph Redmon, Santosh Divvala, Ross Girshick, Ali Farhadi; The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 779-788
4. Arduino PID Library by Brett Beauregard https://playground.arduino.cc/Code/PIDLibrary/
5. Pololu Dual TB9051FTG Motor Driver Shield User's Guide https://www.pololu.com/docs/0J78/1
6. Motor Encoders with Arduino, Bot Blog, http://andrewjkramer.net/motor-encoders-arduino/



Name	Number	Price	Retailer
Mecanum Wheels	2	$100.31	Vex
Suitcase	1	$32	Amazon
Camera	2	$208.51	Intel Real Sense
Nvdia Jetson TX2 (Purchased before)	1	$600.00	Nvidia
Servo Motor	2	$69.90	Pololu
Battery/Power Bank	1	$72	Amazon
12mm Hex Wheel Adapter for 4mm Shaft (2-Pack)	1	$3.95	https://www.pololu.com/product/2684
Bracket Pair	1	$7.45	https://www.pololu.com/product/1082
Dual TB9051FTG Motor Driver Shield for Arduino	1	$19.95	https://www.pololu.com/product/2520
Arduino Mega 2560	1	$34	
	Total cost:	$1,148.14	
	Net cost:	$548.14	*Excluding the items purchased before

Team CB Budget



Gantt chart for Program Management