Cleanbot 3000

Justin Nghiem, Freddy Nunez, Arin Barouni, Harut Ketsoyan, Cristian Miramontes, Eros Aldin Lachica, Gary Chiv, Vicente Zavala, Lilit Aghanes, Samuel Ochoa, Shayan Reza, Joshua Garner
Advisor: Professor James Flynn
California State University Northridge
November 10, 2020

Abstract – The primary focus for this paper is to present the current progress for the Cleanbot3000 senior design team during the Fall 2020 semester term. The Cleanbot 3000 project is a Jet Propulsion Laboratory (JPL) funded project that is advised by Professor James Flynn. The main objective is to design and assemble an autonomous vehicle to navigate through and sanitize the floors of JPL's cleanrooms, whilst adhering to JPL's and general cleanroom standards. Due to the extensive background of robotics, the senior design team is divided into three separate groups: the mapping navigation, marker localization, and power teams. Each of these teams will be expanding the research that they have performed, comparing the finalized models with their alternatives, and discussing possible future goals and considerations for the project.

1. Introduction

The primary focus for Cleanbot 3000, or Cleanbot for later references, will be to disable the microorganisms that exist within Jet Propulsion Laboratory (JPL) cleanrooms. Microorganisms and microbials pose a threat to the terrestrial ecology of Earth's upper atmosphere if they are transited on JPL's flight hardware [1]. Not only can these organisms invade and contaminate the environment, but receiving false positives of a foreign life source in outer space is another plausible issue. In order to combat this, Cleanbot is tasked to eliminate the remaining microbials while also minimizing human interaction, as humans are the primary source for contamination [2]. Although JPL laboratory workers wear protective gear to prevent contamination, the less the human interaction, the better.

Cleanbot has been a senior design group since Spring 2019 and has undergone several changes in their infrastructure and plan of approach as the specifications of the project are ever changing. The general scope of Cleanbot is that it is an autonomous robot that must navigate through JPL's cleanroom and sanitize as much floor surface area using ultraviolet-C (UVC) LEDs. In order to prevent any damage to the flight hardware, it must remain one meter away from any designated, restricted zones. Cleanbot must also adhere to the cleanroom standards by limiting any methods of outgassing.

The Cleanbot team comprises three different subgroups: Mapping Navigation, Marker Localization, and Power. Each of these groups has been continuing the progress made from the past semesters and research on an optimal design for further tests and improvement. By taking into consideration all of the possible scenarios and cross communicating through each of the groups, Cleanbot 3000 may ultimately be assembled and operating.

The navigation group is responsible for implementing a SLAM (Self Localization and Mapping) algorithm to test mapping in an unknown environment. They are also working with the lidar sensor and PiCar to solve the mapping solution and track the robots pose.

The marker localization team is responsible for finding the location of the Cleanbot in the clean room at any given time through the use of pre-placed markers.

The power group is responsible for creating a power system that is capable of providing enough power to each of the robot's subsystems. They have also provided updates to the sanitization subsystem and worked on the robot's mechanical design.

2. Mapping Navigation

2.1 Previous Work

The previous semester had acquired a Raspberry Pi 3B+ and SunFounder PiCar to be used as our main testing platform for navigation. The PiCar had been fully assembled and the Raspberry Pi had been booted with a Raspbian operating system. The PiCar software was installed on the Raspbian OS. The previous semester students also acquired a Scanse Sweep 360° Lidar from a different senior design team. ROS Kinetic was attempted to be installed on the Raspbian operating system on the Raspberry Pi. However, due to memory exhaustion issues and the lack of support of Raspbian by ROS Kinetic, the ROS operating system was never successfully installed. The SDK files for the Scanse Lidar were successfully installed, however only raw laser range and angle data were able to be extracted since the launch files for a visual and SLAM integration were dependent on ROS.

2.2 Plan of Action

The navigation team's main objective was to successfully integrate Hector SLAM's creation of maps with the PiCar by manually moving the car around a room. The navigation team also planned to implement obstacle avoidance through SLAM and the PiCar's ultrasonic sensors. Some of the future work will include using saved maps and the Rplidar to move the robot autonomously.

2.3 Robot Operating System (ROS)

ROS provides the computer's default Linux operating system the building blocks it needs to create and compile ROS packages. This includes dependencies and libraries that allow ROS packages such as the Rplidar, Hector SLAM, and Rviz, a user interface tool for visualization, to run. ROS packages can be made up of anything that makes it a useful module such as configuration files, libraries, or datasets. This module is then launched as a node which publishes topics that either send or receive information to accomplish tasks. For example the Rplidar node when launched publishes a "scan" topic which sends out universal laser range data which Google Cartographer can subscribe to, and uses the information to create a map in real time. Google Cartographer then publishes a "map" topic, when opened from Rviz will show the map being generated in real time [3].

Due to the pre-existing complication of installing ROS Kinetic on Raspbian OS , it was decided to flash a completely new OS, one with ROS Kinetic already installed. This was Ubiquity Robotics Ubuntu 16.04 ROS Kinetic operating system, last updated in December of 2019. Once the new OS was installed, the Scanse Lidar SDK was easily installed and ROS integrated launch files from Github were built. Hector SLAM was also successfully built and was able to launch.

After testing the three SLAM algorithms, it was found that Hector SLAM and Gmapping ran best with the ROS Kinetic distribution. However, Google Cartographer worked only on the newer ROS Melodic distribution. Thus, currently two linux operating systems are being tested on separate memory micro SD cards. Rather than ROS constantly updating its libraries which would cause certain packages to break, the software releases distributions every few years to decrease the amount of changes and bugs ROS packages have to deal with.

2.4 Linux Operating System

The Linux operating systems were initialized with many features and settings were changed to best fit the Raspberry Pi and navigation software. Some packages such as Hector SLAM required additional dependencies to be manually upgraded. One was qt4 which was upgraded to qt5. Qt is a software that creates graphical user interfaces, mainly in the case of Rviz. Other adjustments included initializing an SSH (secure shell host) to allow team members to have remote access to the raspberry pi from personal computers.

Another necessary step was to create a swap file on the Raspberry Pi that would resolve issues of memory exhaustion. Oftentimes when building large packages such as Hector SLAM, the Raspberry Pi would crash. Adding a swap file allows the operating system to use more memory from the micro SD in the Raspberry Pi for executions such as compiling files. Since operating systems were frequently swapped with the Raspberry Pi to test for functionality of several programs, backup images files of the currently used operating systems were made. SD cards would become reusable and if a switch back to Ubuntu 16.04 ROS Kinetic was needed for example, booting up using a backup image file that already contains Hector Slam, lidar sdk files, shh, etc. would be possible.

2.5 PiCar

Sunfounder PiCar-S models which are DIY (Do It Yourself) robot kits that include ultrasonic sensor, light and line follower sensors, as well as all the parts required to build a functioning robot, were acquired for navigation testing. This PiCar model is a rear wheeled operated robot with the front two wheels being servo driven.

The PiCar-S from Sunfounder has a github repository that is documented in its included manual in order to set up the drivers and software that will allow for movement of the PiCar. The scripts included in the git allows us to install the servo motor and test the wheels of the PiCar. There are several parameters in the configuration files that can allow for modification of servo adjustment and speed of the motors.

A ROS implementation for the PiCar was found which allows control of the PiCar through other ROS packages such as teleop keyboard control and autonomous movement through the ROS navigation package. The ROS PiCar package is a set of scripts that translate the built in PiCar movement methods into ROS library software constructs. This creates a PiCar node that publishes the topic "cmd_vel" which other ROS packages can subscribe to in order to move the robot. A ROS topic is a universal method of transferring information from one node to another. Work is currently being done with the PiCar node to create an odometry frame and publish an "odom" topic which will broadcast the PiCar's velocity and odometry information.

2.6 Sensors

2.6.1 Scanse Sweep Lidar

The Scanse Sweep Lidar was designed as a low cost lidar to retrieve laser range and angle data. However, it was discontinued on May 16, 2018, and the Scanse website and forum were shut down. The Cleanbot team received a Scanse sweep lidar in January 2020 from engineers of another project.

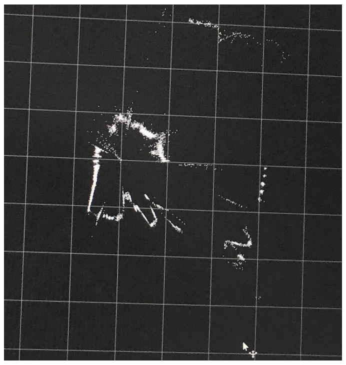
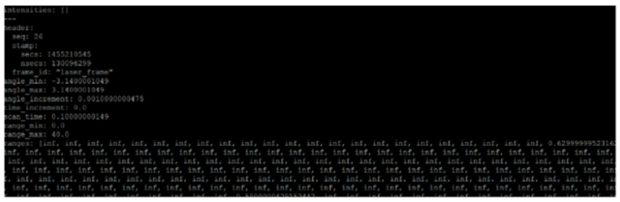


Figure 2.1: Scanse Lidar Scan



```
ntensities: []
--
eader:
 seq: 26
 stamp:
   secs: 1455210545
   nsecs: 130096299
 frame_id: "laser_frame"
ngle_min: -3.1400001049
ngle_max: 3.1400001049
ngle_increment: 0.0010000000475
ime_increment: 0.0
can_time: 0.10000000149
ange_min: 0.0
ange_max: 40.0
anges: [inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, 0.6299999952316
nf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf,
 inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf
, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, in
f, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, i
, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, in
 inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, 0.5800000429153442, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf, inf,
```

Figure 2.2: Sweep SDK Range Data

The navigation team decided to attempt to test the lidar and use it for SLAM. The functionality of the lidar seemed satisfactory as seen from **Figure 2.1**. The visual laser data in **Figure 2.1** shows the laser scans of a room through the ROS visual launch package. However **Figure 2.2** shows a different ROS launch file that extracts raw laser range data for SLAM but does not work properly. It was quickly noticed that our scanse lidar firmware is not up to date to properly run with the ROS package and with SLAM. Since the Scanse lidar website was taken down, there was no way to update our firmware. Thus due to lack of resources the navigation team obsoleted the Scanse lidar for the Cleanbot project and acquired a Slamtec RPLIDAR A1.

2.6.2 RPLIDAR A1

The Slamtec RPLIDAR A1 is the lidar currently being used by the Cleanbot. The RPLIDAR A1 is a low cost 360 degree laser range scanner that measures distances up to 12 meters at a sample rate of 8000 times per second. The navigation team is using the RPLIDAR for SLAM algorithms such as Hector Mapping, Gmapping, and Google Cartographer.

For the purposes of navigation, the transition from the Scanse Sweep Lidar to the RPLIDAR was necessary as most of the libraries, firmware and resources that are associated with the Scanse Lidar are obsolete and do not work with recent updates of ROS. The RPLIDAR was chosen due to the abundance of resources online that could be used to implement autonomy for the Cleanbot. The main goal of the RPLIDAR is to be able to create accurate maps through ROS and use the various SLAM algorithms to create maps that consider odometry, localization, and obstacle avoidance.

2.7 SLAM (Simultaneous Localization and Mapping)

2.7.1 Hector SLAM

Simultaneous Localization and Mapping is a computational problem of building a map of an unknown environment while keeping track of the robot's location as it moves. Cleanbot had begun its first stages of mapping testing with Hector SLAM. Hector SLAM is an open source algorithm that uses laser range data to create an updating map of its surroundings. Once a lidar is set up, Hector SLAM can be downloaded and compiled. Modifications were made to the code to operate with the Rplidar and Picar. This includes editing launch files, changing parameters, adding missing packages, and creating necessary ROS transformations. Once completed it was ready for testing.

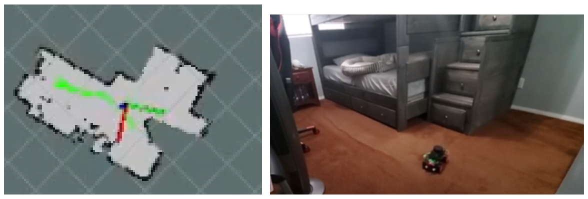


Figure 2.3: Hector SLAM Test in Enclosed Area

While testing, it was noticed that Hector SLAM does well with recognizing small corners such as the space in between the nightstand and bed in **figure 2.3**. However it struggled recognizing smaller objects such as the legs of chairs. The algorithm also became erratic when encountering a mirror, mapping out of bounds values shown in **figure 2.4**. This is however perceivable since the laser from the lidar reflects off of mirrors causing inaccurate readings.

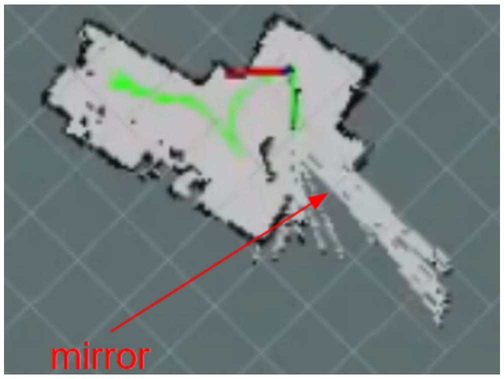


Figure 2.4: Hector SLAM Encountering Mirror

2.7.2 Gmapping

The navigation team is currently testing and working with Gmapping. Gmapping is a much more complex open source algorithm that uses laser range data and odometry to create an updating map of its surroundings. Gmapping is a SLAM approach that uses particles as individual maps and Rao-Blackwellized particle filters to learn grip maps. The algorithm not only takes into account the robots movement, but also the most recent observation for the best pose accuracy. Once Gmapping is compiled, heavy modifications are needed to operate with its given peripherals (i.e. Rplidar and Picar). This includes creating launch files, changing parameters, adding missing packages, and creating necessary ROS transformation nodes.

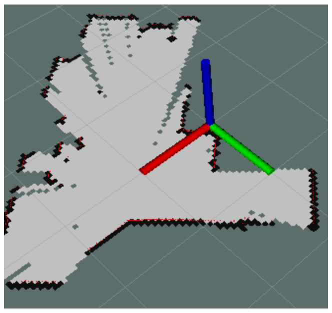


Figure 2.5: Gmapping Initial Scan

Testing so far has consisted without the odometry aspect of the algorithm. **Figure 2.5** portrays initial scans of maps (without further movement). When moving the Rplidar, the maps generated begin to overlap. This is because the Gmapping algorithm has not been optimized for the Rplidar, and parameterized given its max and min range data readings, operating frequency, and accuracy. Another factor that plays into the issue is that no odometry frame has been created for the PiCar. An odometry frame is a software script that tracks the velocity of PiCar given the rotation speed of its rear motorized wheels, and direction of its front wheels. Gmapping uses this frame to estimate an accurate pose and location of the PiCar as it moves. An odometry frame is currently being created by the navigation team. Further work with SLAM is being done with Google Cartographer and operates much more accurately than Gmapping since an optimized version with the Rplidar was found.

2.7.3 Cartographer

Cartographer is an open source system provided by Google that is described as being a system that has the capability to provide real-time simultaneous localization and mapping (SLAM) in 2D and 3D across multiple platforms and sensor configurations. The advantage in using Google Cartographer is that it provides real-time mapping and is configurable with many types of sensors, including the RPLIDAR.

In conjunction with the RPLIDAR, Google Cartographer is configured to use the laser scan data from the lidar for map building and position estimation. The maps created by Cartographer can allow a creation of a map of the environment and also shows where the robot is located by referencing the map. Since Cartographer has a ROS integration package (cartographer_ros), it became suitable for the requirements of navigation.

In configuring the Cartographer package to work with ROS, maps could be created. In the process of creating a map, the RPLIDAR was mounted onto the PiCar and the robot was moved manually around a room to try to recreate the environment. With initial scans from the RPLIDAR, Cartographer only builds very small portions of the map. By having the lidar move around the room the map starts to become more detailed with increasing contrast that shows the edges of the map. As the Cartographer continues to process laser scans, the confidence level from lidar scans start to increase and the Cartographer's uncertainty about the environment decreases. Periodic scans are used in order to create maps of the environment which is important as real-time mapping will be required for obstacle detection and avoidance. Cartographer also provides 2D odometry from laser scans which allows the map to consider trajectory and positioning of the robot as it moves around a room [4].

The 2D odometry provided by Cartographer is important as localization is needed when requiring the Cleanbot to autonomously navigate a map. The 2D odometry provided in maps using Cartographer were observed to be much better implemented than the two previous SLAM algorithms of Hector SLAM and Gmapping.

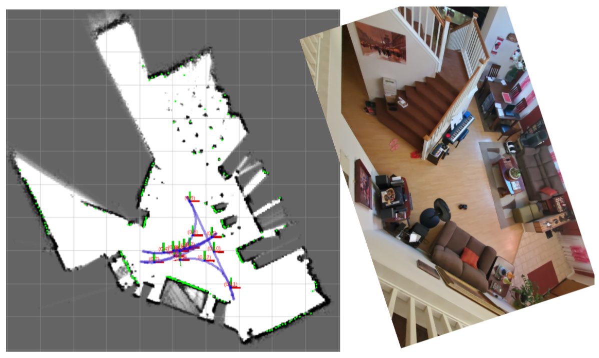


Figure 2.5: Google Cartographer

Figure 2.5 portrays the Google Cartographer algorithm mapping a large room with many obstacles. The algorithm was able to accurately map the small objects such as the chair legs at the top right corner. The navigation team is currently researching and moving forward with Cartographer as it is the most hopeful solution in the creation of maps that take in account the robot's trajectory and position.

3. Fiducial Marker Localization

A fiducial marker is a fixed identifiable marker or object that is used as a reference in order to find the location of an object with reference to the given marker. The Cleanbot 3000 will be stationed in a clean room which is subject to change. A series of markers would be placed on the walls of the clean room to assist the Cleanbot in calculating its current location.

3.1 ArUco

The ArUco markers are similar to QR codes. They are small 2D, binary square fiducial markers that make use of the open source, augmented reality ArUco library. Each ArUco marker has a unique binary number that is encoded into a small black and white grid of pixels. The first content of the markers are a border of pixels that are all black, translating to all 0s. This is to indicate whether it is a marker or not. After the identification of a marker being present, the combination of the inner pixels determine which dictionary the marker corresponds to. The grid of pixels is sized as an NxN matrix which differs in size depending on the corresponding dictionary. There are 25 predefined dictionaries that range from 4x4,5x5,6x6 and 7x7 matrices. These dictionaries also have a fixed number of IDs that range from 50 to 1000. The purpose of these markers is to use them for camera pose estimation. The marker also can give its 3D coordinates relative to the camera [5].

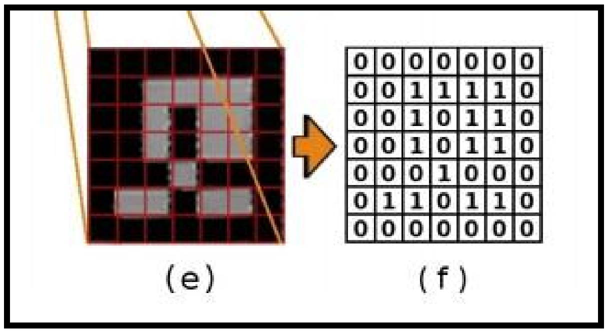


Figure 3.1: Translation of marker to matrix

3.2 Camera Calibration

For testing, the Marker Localization team are using the Raspberry Pi Camera Version 2. The ArUco detection algorithm uses the camera calibration file in order to find the distance from itself to the camera, so it is imperative that the calibration is accurate. In order to calibrate the camera, the Marker Localization team employed a 35"x 44" 8x6 checkerboard. Using the camera calibration driver included in the ROS library, a number of samples of the checkbard were taken at different distances and angles in order to calibrate the camera. At the end of the calibration process the ROS driver creates a yml file with the calibration parameters.

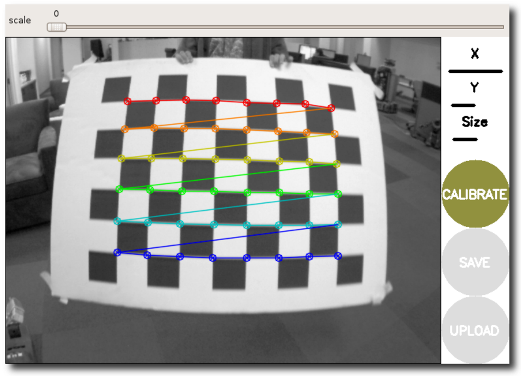


Figure 3.2: Calibration testing

3.3 Fiducial SLAM

The detection of an ArUco marker is done in a series of steps that are directed to detecting rectangles and extracting the binary code from the marker image. Using the aruco_detect node, the robot will be able to detect the markers that are within view of the camera. After this, to get an estimation of the robot's position and camera's orientation relative to the markers, the fiducial_slam node is used. Note that the placement of the ArUco marker does not matter, and only two or more markers must be visible in the camera's view.

The ArUco library has code that allows it to estimate the pose of the Cleanbot by solving a set of linear equations to find the X, Y, Z coordinates from the marker. Using these coordinates, the fiducial coordinate system is transformed into the camera coordinate system denoted as T_fid_cam. This transformation represents the pose of the marker in the camera coordinate system. Since the camera was calibrated, the pose of the camera was able to be determined, and the camera was able to find the location of the robot using T_map_fid2 = T_map_fid1 * T_cam_fid2 * T_fid1_cam.

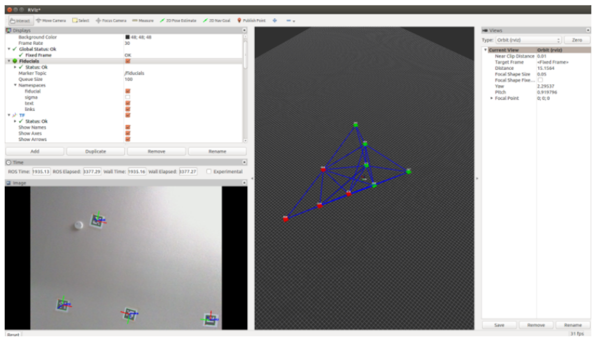


Figure 3.3: Example map using fiducial SLAM

4. Power

4.1 Block Diagram

A general visualization of the power distribution of the project is displayed in the two block diagrams below. **Figure 4.1** describes the method at which Cleanbot will be recharging itself when it is not under operation. To minimize human assistance, the robot must be able to charge itself through a docking station. The charging station will contain a power supply that will charge the battery system through a conducting metal sheet. Although the model and method at which the docking station will operate is not yet finalized, this topology will assist the team in determining how the internal connections within the robot are made so that all of the batteries are charged through one point of contact.

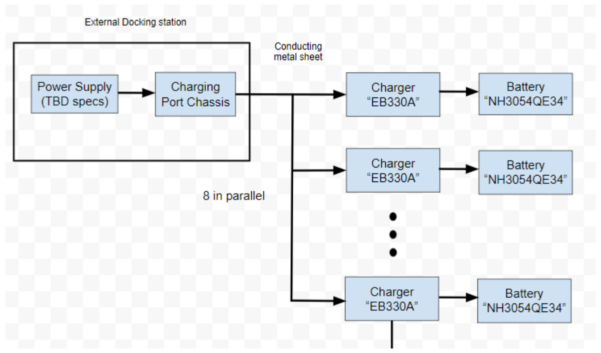


Figure 4.1: First Portion of the General Block Diagram

Figure 4.2 presents the required DC-DC converters, motor controller, and the main subsystems of the robot. These subsystems, from left to right, include the Raspberry Pi for navigation, camera and ultrasonic sensors, the UVC LED light engines, and the motors for the robot.

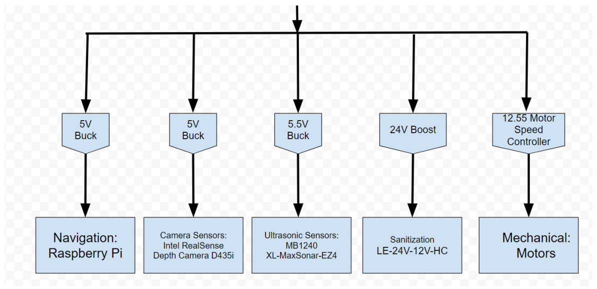

5V Buck
5V Buck
5.5V Buck
24V Boost
12.55 Motor Speed Controller
Navigation: Raspberry Pi
Camera Sensors: Intel RealSense Depth Camera D435i
Ultrasonic Sensors: MB1240 XL-MaxSonar-EZ4
Sanitization LE-24V-12V-HC
Mechanical: Motors

Figure 4.2: Second Portion of the General Block Diagram

4.2 Power Budget

The power budget is a table that details the power consumption from each of the Cleanbot components. Thus, the team is able to determine the total power consumption of the entire Cleanbot. In addition to power, the budget provides the input voltage values of the components, the typical current that they draw, and an estimate of the amount of heat they generate due to losses. This semester, the power budget was refined to account for the changes in some components. The most notable of these components are the Klaran Light Engines, as they draw significantly less current and power than the previous LEDs that had been selected.

The total power calculated from all of the components that may be considered came out to be 125.6W. Out of the 125.6W, only 115.4W was dissipated into heat, the majority of it stemming from the UVC light engines (about 87%). Although it was possible to calculate the expected heat generation of all of the components, it would still be difficult to determine accurately how it may be dissipated throughout the robot's body. Further tests will be performed when these components are on hand in order to decide the necessity for heat dissipating methods such as fans, heat sinks, body design, etc. Also, the number of batteries needed were determined based on the total power consumption.

4.3 Battery System

In the previous semester, JPL had recommended the power team to use a lithium ion smart battery from Inspired Energy, called NH2054HD34. The battery was rated at 14.4V and 8.25A with a capacity of 98Wh. Furthermore, it was non-outgassing which meets the cleanroom standards. However, the NH3054QE34 battery, a newer model than the NH2054, is now being considered. Compared to its older model, the NH3054 is higher rated, in which the discharge and charge rate is greater. The current rating in the new battery is 10.5A, allowing for greater current outputs, if necessary. In addition, it is IEC62133 certified, which allows these batteries to be safely used in "portable applications" [6]. With the calculations made in the previous section, eight batteries will be utilized in parallel to provide Cleanbot approximately three hours of normal operation.

Alongside the battery, Inspired Energy provides single-embedded chargers that can be integrated within the body of the Cleanbot. The 299EGAEB330A model is being considered as their single bay feature will be able to support higher current. They are also not a "pass through" charger since the Cleanbot is not required to be operating during its charging period, which can also increase the longevity of the batteries [7]. These chargers are involved in the charging and discharging processes of the battery and communicate to the battery through its System Management Bus (SMBus). By implementing the chargers in the battery system, it enables the Cleanbot to be charged through a single source and will not require the process of manually charging the batteries individually.

4.4 Simulink

This semester, the power subgroup decided to start working on a Simulink model composed of the Cleanbot's components. The purpose of the Simulink model is to demonstrate the power distribution system within the Cleanbot and to test the capability of the batteries in providing adequate power to each of the loads. **Figure 4.4** shows the overall Simulink model. The power source includes eight 14.4V batteries connected in parallel. The batteries have a discharge profile based on their lithium-ion cell chemistry, as shown in **Figure 4.3**. The discharge rate for each battery was assumed to be about 1 C/s or 1A, which is corresponding to the light blue curve. This results in an estimated operation time of 6.8 hours before the batteries are completely drained (based on their nominal capacity of 6.8 Ah). A sample point of 3.4V per cell at 90% discharge was selected to fill in the discharge parameters of the Simulink battery blocks. Since each battery has four cells in series, the actual sample value is 13.6V at a remaining capacity of 0.68Ah. The loads, or the components of the Cleanbot in which power is delivered to, are modeled as variable resistors to account for inrushes of current during operation. Between the batteries and the loads there are DC to DC converters, which step-up or step-down the battery voltage to the voltage required for each load. Each converter has a controller that adjusts the duty ratio as needed in order to ensure that the output voltage does not swing as the battery voltage varies. However, the controllers may or may not be practical or necessary during the actual development of the Cleanbot.

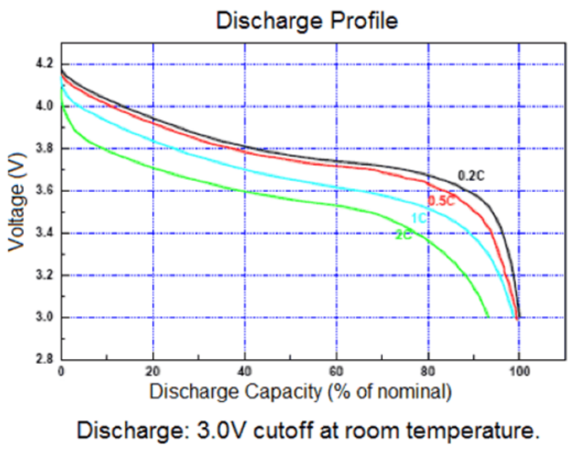


Figure 4.3: Discharge Profile of a Lithium-Ion Battery [10]

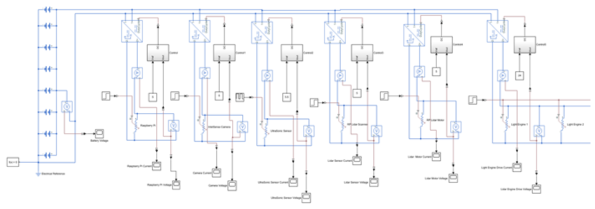


Figure 4.4: Simulink Model of Cleanbot 3000 Components

5. Mechanical Design

The wheel design that will be utilized in the robot will consist of a four wheel, two motor configuration. The front two wheels will be the motor driven wheels and will be utilizing 4” diameter polyurethane wheels. Polyurethane is a cleanroom standard material for wheels as it is non marking. The rear wheels will be two omnidirectional wheels that have a 4” diameter and 10 rubber rollers. The omnidirectional wheels are a great fit for the robot because they have the advantage of being able to roll freely and perpendicular to the drive direction. The 10 rubber rollers provide excellent traction to accelerate in the forward direction.

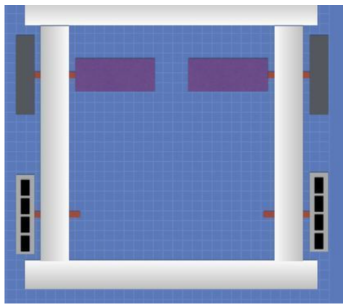


Figure 5.1: Wheel Model Configuration [8]

The previous design that was considered in the previous semesters was the Ackerman steering configuration. This design would operate similar to a car, with the front two wheels steering the car by turning in the same angle [9]. However, this configuration would involve more skidding on the floor with the wheels and would be less compatible with the ROS systems. The current design with normal wheels in the front and omnidirectional wheels in the back will function by turning the front wheels at different speeds or directions. The omnidirectional wheels allow turning in the same position, which can help Cleanbot to maneuver in tight corners.

This semester, the power team continued the ongoing research on what type of motors would be ideal for moving the CleanBot. Last spring, two 12V DC motors, one of 5 RPM and the other of 10 RPM, were tested to see if their performance would meet the rest of the desired specifications. Unfortunately, these motors are no longer available, so the team needed to find alternatives. The RobotShop Drive Motor Sizing Tool was an excellent resource in helping the team to obtain various options for potential motors. After entering a variety of input parameters, output parameters relating to the motor were obtained, as shown below in **Fig 5.2**.

Input Parameter	Value
Robot Mass	20lbs
# of Motors	2
Wheel Radius	2 in
Robot Velocity	4 cm/s
Desired Acceleration	2 cm/s^2
Maximum Incline	10 degrees
Supply Voltage	12V
Desired Operating Time	8 hrs
Total Efficiency	70%

Output Parameter	Value
Angular Velocity	7.5 rpm
Torque	5.78 kg.cm
Total Power	0.45 W
Maximum Current	0.037 A
Battery Capacity	0.59 Ah

Figure 5.2: Table for the Input and Output Parameters for Motors

6. Sanitization

Following the previous work done last semester, the team is continuing to use UVC LEDs to destroy microorganisms along the floor of the JPL cleanrooms. The team will be using the Klaran UVC light engines this semester as they are stronger, more efficient and the LEDs are already mounted on a PCB. The light engines draw 29W and have 12 LEDs each outputting 50mW of UVC doubling our efficiency to roughly 2%.

Characteristic	Unit	Min	Typical	Max
Power Input Voltage (VCC)	V	22.8	24	25.2
Power Consumption (LED ON)	W	-	29	-

Figure 6.1: Table of the 12LED 24V Light Engine Power draw from Klaran's data sheet provided

These features allow the Cleanbot to move faster and have a longer life span. Subsequently, the team was able to focus more time on other systems of the robot. JPL currently holds the light engines and will begin testing once a mount has been created. With the actual light engine in hand next semester, the team will be able to track the actual heat generated, power draw, and sanitization capabilities.

Using the LED models created in the previous semesters the team have modified and added functions to use for the overall power/heat system of cleanbot. These functions include total system power draw, power conversion efficiency using DC to DC converters, component/overall heat, etc.

7. Conclusion

Although the entire team was restricted due to the pandemic and financial limitations, there were still several improvements and progress made for the project. Even without any equipment to work on, the research performed this semester has drastically propelled the project closer to its end product.

The mapping navigation team began testing with Hector SLAM, Gmapping, and Google Cartographer. The team found the most success with Google Cartographer, and will be moving forward with that algorithm for further testing. The future goals for the team include saving a created map from SLAM, and using a ROS navigation package to begin autonomous movement with the PiCar. The team will also research path planning algorithms to move the robot in the most efficient way possible when cleaning floors.

The marker localization team used the ArUco augmented reality library to generate markers that can be used to find the pose of the Cleanbot. They were successful in finding the pose by using two markers. The future goals for this team are testing the accuracy of the marker while the Cleanbot is in motion and using multiple markers to find the pose of the Cleanbot at any given location.

The power team will continue to update their power budget whenever there are updates to any of the subsystems. Their future goals include simulating each of the subsystems and verifying their compatibility. In addition, they must also research methods of a self-docking station and the materials and design most suitable for the Cleanbot body.

From the work performed in this semester, Cleanbot has either started or will begin the prototyping and testing of its several subsystems. By dividing into different subgroups and cross communicating with each other, the project may be completed in the most efficient and organized fashion. Adhering to the various JPL and cleanroom specifications will ensure the safety of the robot in actual implementation. Completing the Cleanbot 3000 project will provide a method of autonomous sanitization for not only cleanrooms but also other facilities such as hospitals, storage rooms, etc. It pushes the standards of robotics, as it combines elements of obstacle avoidance, mapping, self localization, and ultraviolet-C LEDs. There are endless possibilities when it comes to the capabilities of the Cleanbot 3000 and the project is always susceptible to improvements.

References

[1] Smith, David. (2014). A Balloon-Based Payload for Exposing Microorganisms in the Stratosphere (E-MIST). Gravitational and Space Research. 2. 70-80.

[2] MacDougall, Raymond. “NIH Human Microbiome Project Defines Normal Bacterial Makeup of the Body.” *National Institutes of Health*, U.S. Department of Health and Human Services, 31 Aug. 2015, www.nih.gov/news-events/news-releases/nih-human-microbiome-project-defines-normal-bacterial-makeup-body.

[3] “Powering the world's robots,” ROS.org. [Online]. Available: https://www.ros.org/. [Accessed: 11-Nov-2020].

[4] Robotics Weekends, “2D Mapping using Google Cartographer and RPLidar with Raspberry Pi,” Medium, 12-Oct-2020. [Online]. Available: https://medium.com/robotics-weekends/2d-mapping-using-google-cartographer-and-rplidar-with-raspberry-pi-a94ce11e44c5. [Accessed: 11-Nov-2020].

[5] G. Jurado, “Automatic generation and detection of highly reliable fiducial markers under occlusion,” ResearchGate, Jun-2014. [Online]. Available: https://www.researchgate.net/publication/260251570_Automatic_generation_and_detection_of_highly_reliable_fiducial_markers_under_occlusion. [Accessed: 11-Nov-2020].

[6] “What's IEC 62133-2:2017 Certification of Lithium-Ion Battery?,” What's IEC62133-2:2017 Certification of Lithium-Ion Battery? [Online]. Available: https://www.lipolbattery.com/What's-IEC62133-2-2017-Certification-of-Lithium-ion-Battery.html. [Accessed: 11-Nov-2020].

[7] Usman, “What is Pass Through Charging,” Charger Harbor - Charger Reviews, 12-Jan-2017. [Online]. Available: http://www.chargerharbor.com/what-is-pass-through-charging/. [Accessed: 11-Nov-2020].

[8] DroneBot Workshop, “Build a Real Robot - Part 4 - Selecting Wheels,” DroneBot Workshop, 06-Apr-2019. [Online]. Available: https://dronebotworkshop.com/real-robot-004/. [Accessed: 11-Nov-2020].

[9] D. Thompson, “Steering Ackerman.” [Online]. Available: https://www.me.ua.edu/me364/PDF/Steering_Ackerman.pdf. [Accessed: 10-Nov-2020].

[10] L. Ada, “Li-Ion &amp; LiPoly Batteries,” Adafruit Learning System. [Online]. Available: https://learn.adafruit.com/li-ion-and-lipoly-batteries. [Accessed: 11-Nov-2020].