**2nd Prototype Test Plan**

To: Professor Pisano

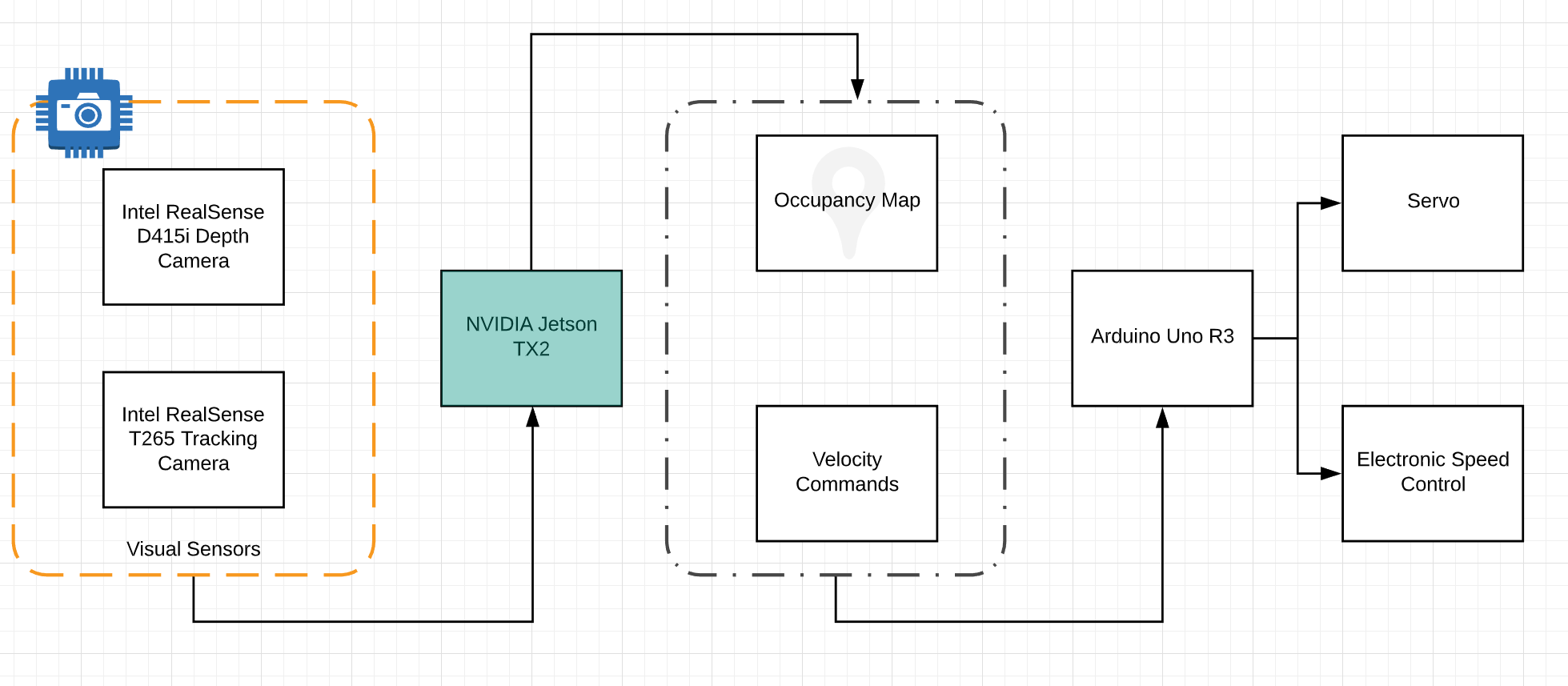
From: Mars Rover Autonomous Navigation

Team: 28

Date: 2/27/20

Subject: Critical Design Review

1. **Required Materials**
   1. Hardware
      1. Nvidia Jetson TX2 developer kit
      2. Intel Realsense T265 Tracking Camera
      3. Intel Realsense D415 Depth Camera
      4. Arduino UNO
      5. L298N Motor Driver
      6. Inland Powered USB hub
      7. Vantec USB PCIe Host card
      8. Servo(9KG)(Waterproof) 51C00-SP9002
      9. Electronic Speed Controller (Waterproof) 98120
      10. Motor (RC540) W/ Gear (12T) 03012 x 2
      11. Batteries:
          1. Battery(7.2V,2000mAh) 51C00-03200 for Drive Train
          2. PowerAdd Pilot Pro 2 (5-19V, 24000mAh) for Jetson and USB hub
      12. Cables:
          1. USB A - USB B cable for Arduino-Jetson Connection
          2. USB A - USB C cable for D415 - Jetson
          3. USB A - USB Micro B cable for T265 - Jetson
          4. Jumper Wires for Arduino connections
      13. Non Electronic Components:
          1. 3D-printed camera mount
          2. Rover chassis
          3. 16” x 6” x ⅛” Acryllic x2 (+mounting screws)
          4. ~10 strips of velcro tape
   2. Software
      1. Installs:
         1. Jetpack 4.3 with CUDA
         2. OpenCV 4.0
         3. ROS Melodic
         4. Intel Librealsense 2
         5. Python 2.7
         6. C++
2. **Setup**
   1. The setup can be incorporated into three processes: obtaining the Jetson IP address
3. **Pre-testing Setup Procedure**
   1. First, place the upper level (consisting of the Jetson, USB hub, and cameras) on the rover, if it’s not already.
   2. Then, connect the USB hub and Jetson to each other and the battery. Connect also the HDMI cable, keyboard, mouse and Arduino to the Jetson.
   3. Turn on the Jetson and run ifconfig to retrieve its IP address.
   4. Connect the D415 and T265 to the Jetson.[[1]](#footnote-0)
   5. At this phase, perform the occupancy map test.
   6. Disconnect the HDMI, keyboard, and mouse.
   7. Flip the drivetrain battery kill switch to ON. The rover is good to go.
   8. On another computer, enable ssh connectivity with the command *ssh team28@<ip address> -X*
   9. In separate ssh sessions, do the following in this order:
      1. *source ~/catkin\_ws/devel/setup.bash*
      2. *roslaunch occupancy occupancy\_live\_rviz.launch*
         1. When the two gdb prompts show up, run them in the order they show up.[[2]](#footnote-1)
      3. rosrun python rosserial

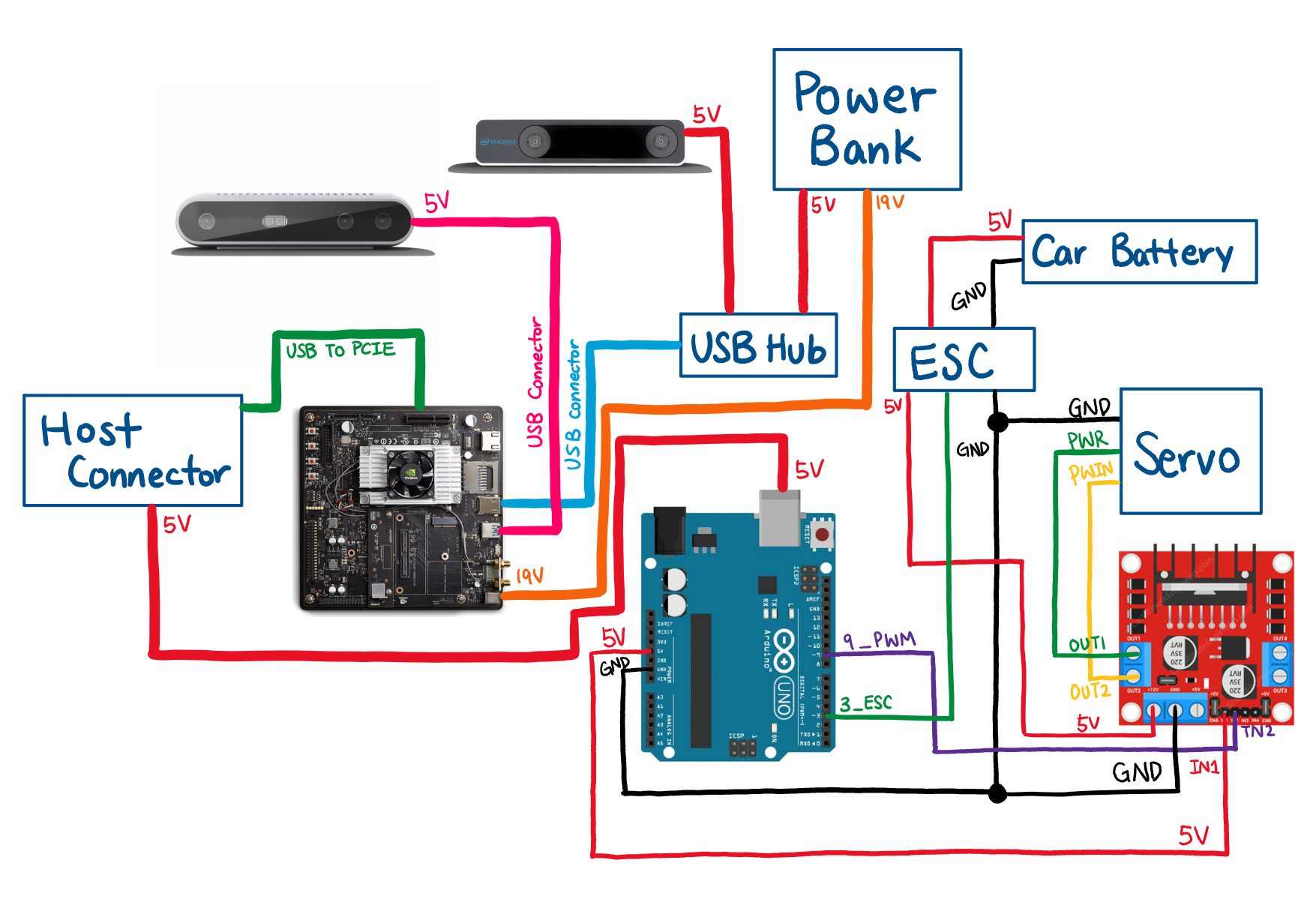


System Design Overview

1. **Testing Procedure**
   1. Test that the rover is building an occupancy map. For display purposes, this will have to happen with the rover connected to the monitor, since 3D modelling GPU commands cannot be forwarded through ssh like other graphics.
   2. Next, after placing the rover down in the test site, test manual control. Using rostopic pub commands, tell the rover to move and turn in the ssh shell.
   3. The Arduino node will read these commands and move and turn with minimal delay.
   4. Then, test automatic control. Place an april tag within 5-7 m of the front camera. The cameras will detect the tag, and without user input, navigate towards the tag. While navigating, use the rostopic echo command to see if the rover has “locked” onto the tag.
2. **Measurable Criteria**
   1. The rover should be able to display build and display a map while connected to the monitor.
   2. When receiving movement commands, the rover should arm (calibrate) the ESC correctly. This will be indicated by a beeping sequence from the ESC.
   3. The rover should turn and move as prompted by the user during the manual control test.
   4. When under automatic control, the rover should detect the location of the tag, and publish the tag coordinates to the appropriate topic (this will be confirmed by running *rostopic echo* in the ssh session). The /tag\_detections topic will output the X,Y,Z Position coordinates of the AprilTag in the camera frame, as well as the X, Y, Z Orientation coordinates of the AprilTag. The 6DoF measurements are effectively known as the 3D Pose of the tag.
   5. Finally, the rover should move to within 2m of the apriltag autonomously.
3. **Score Sheet**

|  |  |  |
| --- | --- | --- |
| **Test** | **Result** | **Success?** (**Y/N)** |
| Occupancy Map | able to generate map according to what the rover sees | **Y** |
| ESC calibration | able to calibrate properly | **Y** |
| Manual Control test | able to control the rover manually | **Y** |
| April Tag detection | able to detect april tag and print coordinates | **Y** |
| Navigation to April Tag | able to navigate towards april tag | **Y** |

1. **Hardware Pinout**
   1. Hardware Pinout Graph

****

7.2 Arduino Pinout Chart

|  |  |
| --- | --- |
| **Pin Number** | **Usage/Description** |
| 9 | Connect to L298 Motor Driver |
| 3 | Connect to ESC |

7.3 L298 Motor Driver Pinout Chart

|  |  |
| --- | --- |
| **Pin Number** | **Usage/Description** |
| IN\_2 | PWM from Pin 9 of Arduino |
| IN\_1 | 5V from 5V Pin of Arduino |
| GND | GND from Arduino, ESC, and Servo |
| OUT\_1 | To PWR of Servo |
| OUT\_2 | To PWIN of Servo |

7.4 NVIDIA Jetson TX2 Pinout Chart

|  |  |
| --- | --- |
| **Pin Number** | **Usage/Description** |
| USB | Connect Intel Realsense D413 |
| USB | USB Hub to connect Intel Realsense T265 |
| USB to PCIE | Connect to Host Connector that provides power to Arduino |
| Power | Power connected to Power Bank |

1. If the T265 is connected to the USB hub at boot, the Jetson will not recognise it. This is a known issue, and the only fix is to plug in the hub first, and then the camera to the hub. [↑](#footnote-ref-0)
2. If the nodes for both cameras are launched simultaneously, they will both crash. This is also a known issue, and our fix manually delays the launches. [↑](#footnote-ref-1)