Lessons Learned:

Throughout the ECE 4961/4971 Capstone Courses, the Autonomous Crawl Space Inspection Robot Team has experienced many difficulties, which allowed for positive reinforcement of best practices and some interesting learning opportunities.

The mechanical movement subsystem experienced a setback with utilizing real-time interrupts (RTI) on the Arduino microcontroller. By interfacing with speed sensing, RTI, and direction controls on one microcontroller, the RTI often caused the program to stop and become stuck within the interrupt functions. To solve this problem, team members discovered a way to detach and reattach the interrupts within the Arduino software, slowing down the program in the process. While the current use of RTI within the motor code works, the team members believe this program could be made much more efficient by using more than one microcontroller for controlling interrupts, speed, and changes in direction.

For the environmental sensing subsystem, building the wood moisture probe required significant research into different methods for measuring resistance. Thanks to this research, the sensor worked well with the first iteration while providing a lesson in designing a circuit. On the other hand, one of the most significant issues with this subsystem came from the PCB design. The first iteration of the boards had many problems including incorrect footprints, plugs that were hard to access, and misaligned parts. These problems provided invaluable lessons in PCB design but unfortunately came after the parts had already been ordered.

The navigation subsystem experienced one specific setback/learning opportunity during the building phase of the capstone project. One evening, during a test of the power draw on the robot, the LiDAR sensor contacted the ammeter probes and was shorted through the USB ports of the Raspberry Pi 4B and back to the battery. The short caused damage to the RPi and the LiDAR sensor, both of which had to be replaced. Additionally, the microSD card connected to the RPi was corrupted. After many hours, the team was able to recover/recreate the ubuntu system and navigation algorithms. While this was disheartening at the time, the team members learned to be more careful with measuring current and voltage on a moving system as well as the importance of regularly backing up code.

The navigation subsystem also experienced issues with implementing common autonomy packages and previously simulated navigation programs into ROS. While the Hector SLAM algorithm is currently working with ROS to generate detailed maps, the team members were not able to get 2D navigation goals to work with the current iteration of the robot, instead relying on a custom python navigation program. If time permitted, the team feels confident that a custom ROS package, based on the official ROS wiki tutorials, could be created for the robot which would allow for Autonomous SLAM to be implemented.