CS 467
Fall 2021

Greenhouse HyperRail: Project Plan

Project Owner/Mentors
Jorian Bruslind
Chet Udell

Team Members
Kelley Reynolds, Eric Sanchez, Lando Shepherd, Peter Wright

Introduction

This project contributes to the software package which will operate the motion and data collection modules of a HyperRail linear motion control system. This system is intended for use in a plant cultivation study at Oregon State University and will be used to collect humidity, light, carbon dioxide, and spectral imaging data to log environmental and fertilizer uptake data in a test plot. Our project is to develop Robot Operating System (ROS) nodes which control the x-y motion of the sensor end effector, data collection through the sensor end effector, and the graphical user interface (GUI) for controlling the motion and data collection either manually or automatically. The HyperRail portion of the cultivation study is designed to automate specific actions which will improve consistency of data collection, provide automated data collection, and reduce the amount of human interaction required for the study.

User's Perspective

Our software will allow the end user to interact with the HyperRail System by controlling the motion and data collection of the instrumentation module. This includes specifying the path of travel for the end effector, data collection through environmental and multispectral sensors, and specifying a data collection schedule. The end user will be able to specify a path for the end effector to travel, the type of data to collect, and an automation schedule. The interface will also allow the user to review historical data collected by the system.

Initial Project Plan

The major portions of this project consist of the motion control node for the HyperRail system, the data collection node for the sensor end effector, and the GUI for interacting with the system and the collected data. Motion control will be developed as a ROS node which will send G-code instructions to an ESP32 microcontroller. Movement of the sensor end effector will be defined using G-code instructions which will be sent from the host microcontroller (Nvidia Jetson Nano) to the ESP32 which controls the stepper motors for motion along the X and Y axes of the HyperRail.

Sensor data collection will also be developed as a ROS node which will collect data as specified by the end user as it moves along the X and Y axes. This will include collecting data from a specified point at a specified time, or the user may input a specified path from which to collect data. Data may be collected from a one-time point or on an automated schedule.

The user interface will be developed for use on a web browser using Javascript, HTML, and CSS. The interface will allow the user to input the HyperRail path of travel, the type of data to be collected, and the option to collect data at specified intervals.

Tools and Technologies

ROS - Motion control and sensor data collection
Rvis, RQT - 3D visuals of robot or diagramming/organization of ROS packages
SQLite
HTML/CSS/Javascript, Ruby - User Interface

Initial Tasks (implementation subject to change)

1. User Interface
    - Basic Login page → Provide credentials before access to main user interface
        - Login → directs to User Interface
    - User Interface Home
        - Display relevant data about system (HTML)
            - Sensor data (TBD → moisture, temp, C02)
            - Location Data (XY)
            - System info (Build info, voltage / current , PCB / system OS)
            - Display work area (picture stitched)
        - Control elements of the system
            - Send manual movement commands
            - Download SQL data
        - Path Planning display (HTML)
            - Display the current work area
                - Start path planning (provide a cursor to draw paths)
            - Control options
                - Start, Stop, Reset
                - Each button should modify visuals on work areas (Track position)

        Notes-

                - Multicolor points -
                - Picture, movement
                - Pause and take picture

2. ROS Motion Tracking packaging (not sure about this one)

    - Motion tracking ROS → ROS Visualization (sends data)

- ESP32/Microcontroller USB output ROS → Motion Tracking ROS (receives data)

3. ROS Visualization

- ROS Motion Tracking → ROS Visualization (receives data)
- ROS Visualization → directs to User Interface (sending data)
- Function:
    - Display work area (picture stitched) with robot position overlay
        - Inputs: GRBL Status info
        - Outputs: .png or another picture format of work area with robot picture

4. ROS Sensor Interpretation
- Sensor Output C++ → USB OUTPUT ROS → ROS Sensor Interpretation (receiving data)
- ROS Sensor Interpretation → User SQL Database (sending data)
- ROS Sensor Interpretation → User Interface Home (sending data)
- Function:
    - Take data from the sensor end effector and formulate SQL queries to store the data in the database
        - Input: Parsed Sensor data
        - Output: SQL queries to input data into the database

5. ROS Path Drawing

- Path planning feature (from UI Home) → ROS Path Drawing (receiving data)
- ROS Path Drawing → ROS Motion Tracking (sending data)
- Function:
    - Node to determine Gcode commands for User path planning
        - Inputs: User drawn data + Workspace grid / info
        - Outputs: Gcode formatted data for location commands

6. ROS Image Capture

Positive y direction

- Camera Interface → ROS Image Capture (receiving data)
- ROS image capture (image file) → Path planning feature (from UI Home)
- Function:
    - Take an image from the camera and distribute the image to the required "nodes"
    - See MicaSense API
        - Input: Camera Interface action
        - Output: Camera image

7. ROS Sensor End Effector Parser

- Parse end effector output into a more useable format to be used elsewhere
- Need to find updated code for the sensor end effector (or write new code)

Team Member Tasks

Kelley Reynolds

Task	**Time Estimate (hrs)**
(HTML) Basic site (login, framework, database	**10**
(HTML) Display system data	**10**
(HTML) Control elements of system	**10**
(HTML) Path planning interface	**30**
ROS Sensor Interpretation	**20**

Eric Sanchez

Task	**Time Estimate (hrs)**
Set up development environment, familiarize with ROS concepts	**10**
Familiarize with G-codes/GRBL	**10**
Path Planning - ROS, create node to handle sending G-codes to ESP32	**20**
ROS Image Capture - familiarize with MicaSense API/integration with ROS node	**10**
Work on API interaction - image collection along path of travel	**20**

Work on image stitching with OpenCV	20

Lando Shepherd

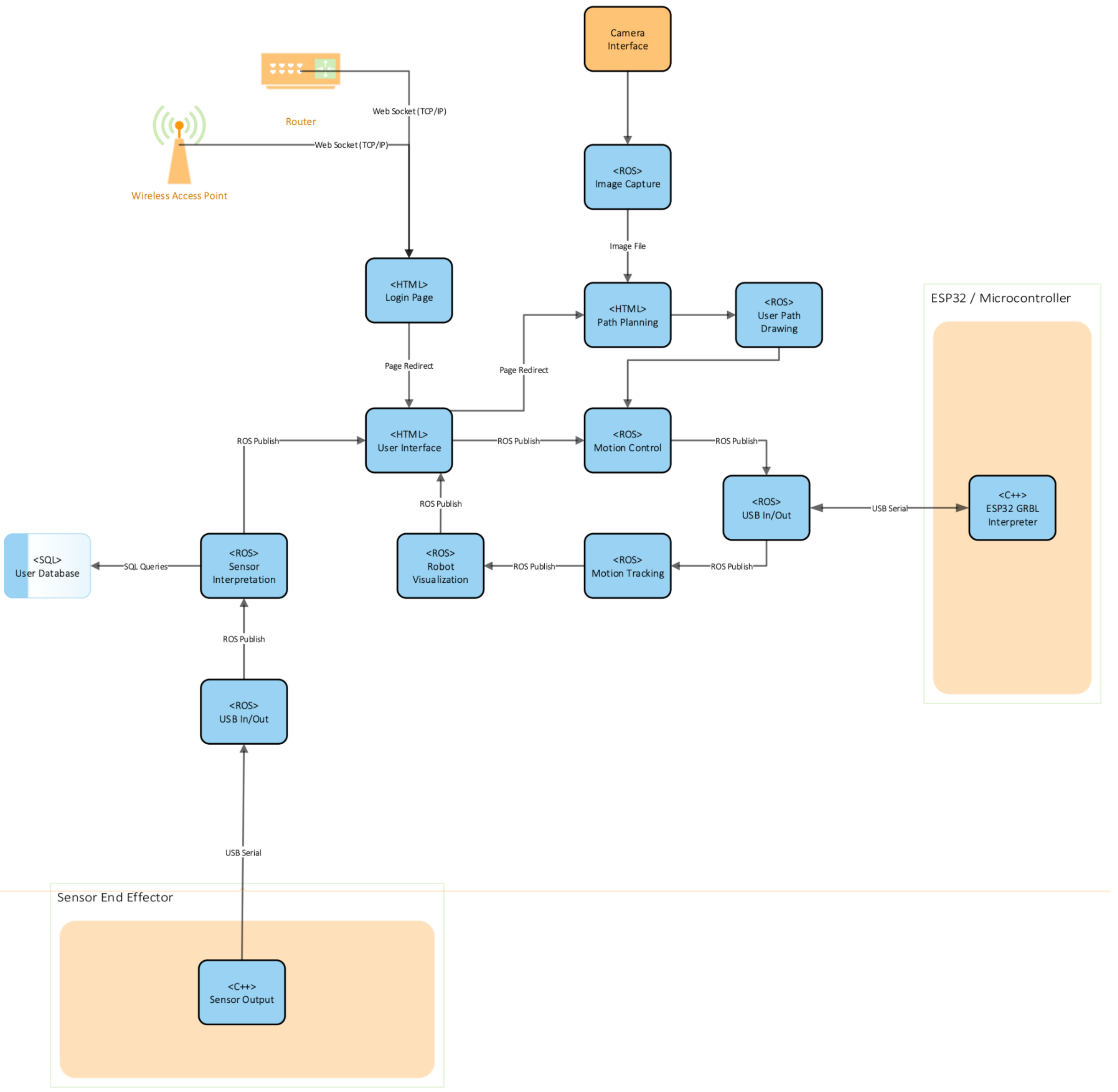
Task	Time Estimate (hrs)
- Set up a development environment. - Collaborate with the team to complete the project plan.	10
- ROS Sensor Interpretation research and design planning	20
- Implement ROS Sensor Interpretation	15
ROS Sensor End Effector Parser research and design planning	20
- Implement ROS Sensor End Effector Parser	15

Peter Wright

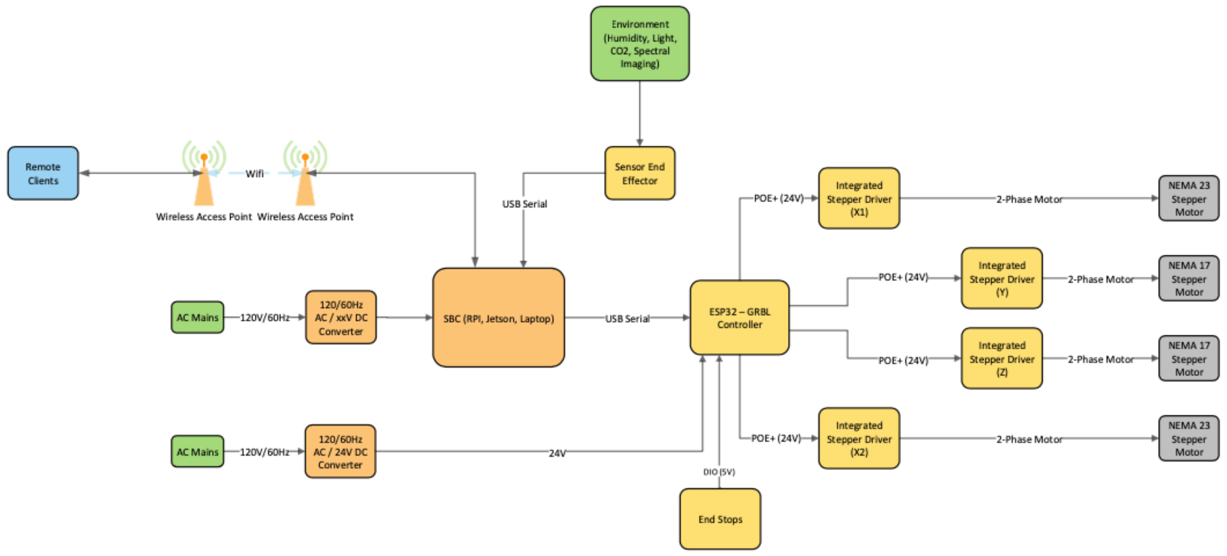
Task	Time Estimate (hrs)
-Set up the development environment. -Install ROS example practice packages. - Collaborate on a project plan. - Make initial decisions on tasks -- Research	10
Image Capture - APi MicaSense study, capture image and send file to html interface	20
User Path Drawing design - How to get inputs: User drawn data + Workspace grid / info Outputs: Gcode formatted data for location commands	20

Get acquainted with sending/receiving data with Gcode.	**10**
Work on image stitching with OpenCV	**20**

Architecture of Software



Data Flow Chart



Conclusions

This project will contribute to the software package to control the HyperRail Robotic Greenhouse. The specific modules which will be completed during the term are subject to change and will be prioritized to allow user interaction with the HyperRail, data collection, and storage. Completion of this project will include capacity for user interaction and a set of functioning ROS nodes for data collection and storage. We anticipate this project to take on the order of 320 hours to complete over the next eight weeks.