Software Engineering for Cyberphysical Devices: Design Sheet

Pex Jan-Mathijs, Theys Ferre and Vandenberk Seppe

System Requirements

The system needs to be capable of moving around an industrial facility to survey the area and measure various data elements. This means that the device should be able to move under user control as well as being directed to specific points within a room. The device doesn't require the ability to carry more than its own weight. Power for the device should come from the robot itself, necessitating a battery-operated solution without the need for cables. The company intends to control the robot through a web browser, meaning its operational range is constrained by network coverage.

The industrial facility may feature multiple floor types, including the possibility of wet and dusty floors, as the robot's primary purpose is to detect errors. Although the rooms won't have stairs, the potential for ramps exists.

Ultimately, the robot should be capable of performing inspection, repair, and maintenance tasks.

Mechanical Design

- Tracks are more versatile and stable then wheels. They also provide more traction across multiple floor types. Possible addition of weight won't be much of a problem, because of the weight distribution and the higher payload capacity compared to regular wheels. Finally, wheeled robots may have more maneuverbility in tight spaces, tracks can still navigate efficiently enough through industrial environments.
- The idea is to build a low profile bot, with a telescopic arm, so it fits better in to tight spaces. The use of a skid plate will prevent the spraying water from entering the system.

Electrical Design

- **Raspberry Pi**: The Raspberry Pi serves as the brain of the robot, controlling its behavior and components. It processes sensor data, executes algorithms, and sends data over the cloud to our interfacing software.
- **Sensors**: Our robot is equipped with an array of sensors for perception and environmental awareness. These sensors include:
  - **Distance Awareness**: Our system will use a Lidar sensor for distance perception. This in combination with a GPS sensor, can help determine the robots location.
  - **Inertial Measurement Unit (IMU)**: Provides data on orientation, acceleration, and angular velocity for navigation and motion control.
  - **Camera(s)**: Enables visual perception and object recognition for tasks such as navigation and object manipulation.
  - **Environmental Sensors**: Measure parameters such as temperature, humidity, and atmospheric pressure for environmental monitoring and adaptation.
  - **Safety Sensors**: A safety sensor, such as a touch sensor can prevent the robot from doing unwanted behavior.

- **Actuators**:
  - **DC Motors**: Used for driving mechanisms.
  - **Servo Motors**: Provide precise control over angular position for tasks such as camera orientation.
- **Power Supply**: The power supply system based on batteries, provides electrical energy to the robot's components. It includes voltage regulators and power distribution circuits to ensure stable and reliable operation.

Software Design

- Python will be used to program the robot, it is integrated in webots and is there for one of the easiest methods to program in webots.
- For the robot autonomous drive we are going to use the D* lite algorithm. D* lite is a modified version of A* that saves the previous paths and uses them for future pathfinding. It's light to run on less sofisticated hardware. It has also proven itself in many occations that it is one of the best pathfinding algorithms for autonomous robots.
- D* lite will run based on a map of the environment the robot will make with its lidar sensor. The data the lidar returns is used to make a point cloud in which surface reconstruction algorithms can be used to make straight walls to complete the 2D environment map.
- A waypoint system will be added to improve the user efficiency. Most waypoints will be located at important locations, such as its charger of measuring interfaces.
- In the event the D* lite algorithm fails to safely pathfind the robot to its destination there will be the pressure sensor that triggers a piece of code that retraces the last step the robot took and goes back. After the robot is back in its previous known good position the D* lite algorithm will run a second time to recalculate its path to set destination.
- There will also be basic safety functionality. A stop and emergency stop will be implemented to stop or even exit the code so the robot stops running.
- Interaction with the robot goes as mentioned via a webinterfac, which is written in html. The webinterface will be running an MQTT connection in the background to communicate with the robot itself and there will a websockets server build in to receive the live video feed the robot will be getting from the camera.

Safety Systems

To make sure our robot stays safe and nothing bad happens several safety measures are put in place, these include:

- **Killswitch**: To ensure that when our robot does make a mistake or something unexpected happens an emergency stop is present on the webinterface and on the robot itself that will disable any robot functions.
- **Manual override**: This will give the users the possibility to stop all robot movement and take direct control of the movements remotely.
- **PID control**: PID controllers will monitor how big the error is and will adapt the system so that the error becomes smaller.
- **Touch sensors**: If other sensors fail and the robot hits a wall or an object a touch sensor can give feedback that the route it is currently following is not possible so the robot can make adjustments.
- **Continuous map updates**: If while driving an object appears on the lidar and gets added to the map the route will be periodically recalculated to ensure no collisions happen.
- **Latency monitoring**: Ping will be displayed on the webinterface.

Human-Machine Interaction (HMI)

The robot is controlled with a webinterface. The webinterface will contain 4 control buttons to move the robot manually, the buttons will also be mapped to keys on the keyboard of the used machine. The user can pick the method they want to control the robot. User input for move commands will be send to our robot with MQTT. There will also be a live video feet that is directly streamed from the robot camera to the webinterface. The video streaming will be setup with websockets. And at last there will some kind of method to start the autonomous drive on the robot with some kind of button to start te process. There will also be a stop and emergency stop button build in the website so the user can stop the running code in an event of a failure.

Environmental Assumptions

To develop a good simulation of the system, it is crucial to asses the environmental factors that could influence the efficiency and durability of the robot. These factors encompass a range of elements, including terrain, weather conditions, power requirements, and sustainability considerations.

When it comes to terrain, it is essential to make some assumptions about the surrounding of the robot. Whether it will be navigating indoor spaces with smooth surfaces or outdoor terrains with rough surfaces and obstacles, the selection of mobility mechanisms like wheels or tracks becomes vital. For this design, the robot will be limited to an area that is covered by a wireless network. So the robot will be designed for indoor use, but the floors can be wet and dusty due to it being an industrial facility and the main task of the robot to surveil and detect any problems.

Weather conditions don't significantly impact the operational efficiency and longevity of the robot. Because of its indoor use. Therefore, there will be no direct action taken to prevent wheather damage to the robot. The robot we design will be splash proof, to prevent spraying water from the tracks from entering the system. Since it is an indoor industrial facility, the assumption is made that the temperature will not exceed the working temperatures of the equiment

Power considerations are fundamental in determining the autonomy and dependability of the robot. Because of the intended use the robot should be able to function on a battery and will be able to charge itself in the facility. Most of todays industrial facility are not to big to drive around in one singular battery charge, so this is also an assumption we take for this project.

Maintenance

Our robot utilises a webinterface and thus is not bound to any specific software or operating systems. Because of this updates and maintenance on the software can simply be performed by opening the webinterface and editing the necessary code. Updates to our code or design will happen on github where users can pull the most recent version to remain up to date.