Teknologiprosjekt: Øving 2

Planning report

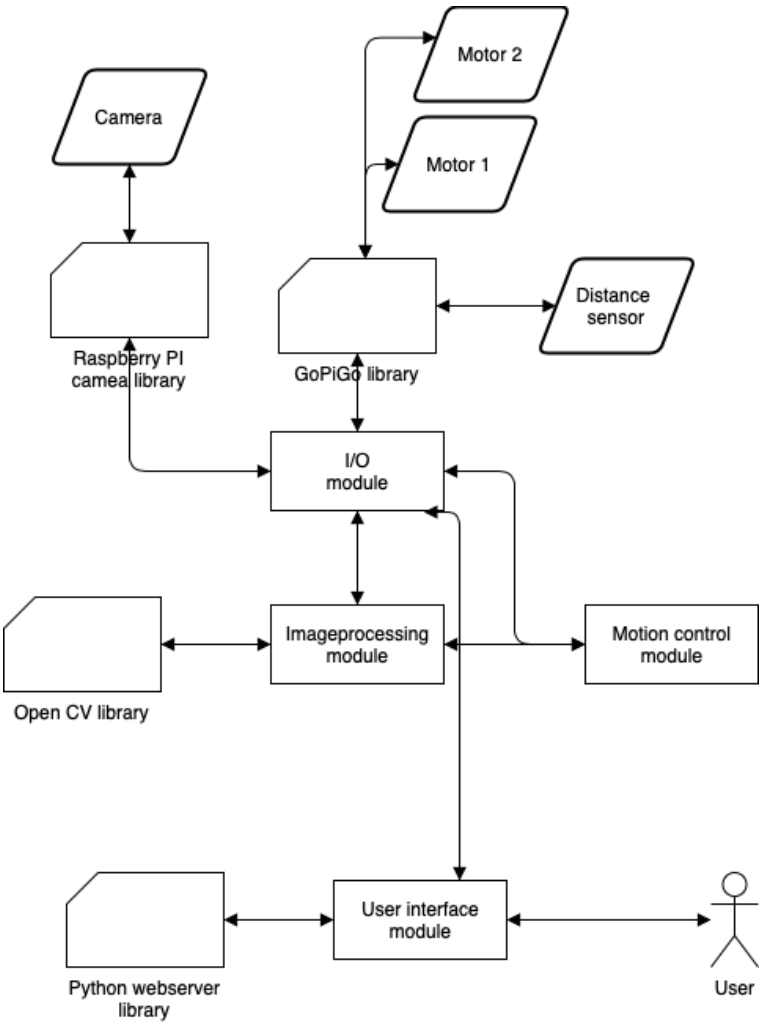
The project fits for industrial warehouses, hospitals and anywhere robots are suited to move things around automatically. There is a lack of cheap options in the current state of the art.
Our expected outcome is to make a robot follow a painted line, using a camera and detect objects in its path using an ultrasonic range finder. The robot should be able to move on any track and make decisions in real time. The overall goal is to create a cost-efficient line-following robot that can do simple tasks such as carrying cargo from A to B. The robot should be able work in different scenarios (harbor, warehouse, hospitals etc).

Requirements

- Following any track with specific requirements.
- Regulate angular and linear velocity.
- Detect obstacles and respond to them.
- Carry small items as cargo.
- Control and get values / status in a web interface.

Limitations

- The robot will stop if it detects an obstacle instead of trying to move past it.
- Can't carry too heavy/big objects.
- The lines in the track must be in a constant color, it must be a contrast between the background and the track.



Motion control algorithm

1. Sample input data from the distance sensor and stop the robot if the distance is within a certain value.
2. Check data from the image processing if the track exists in the image.
   - On track: Goto point 3.
   - Not on track: Drive around in circles until the track is found.
3. Check if the robot is near a crossroad.
   - On crossroad: Move either left. right or forward. The robot can be hardcoded to always move in one direction or take a random direction.
   - Not on crossroad: Go to point 4.
4. Find the shortest distance and angle to the middle of the track. This will be the general direction / the next destination.
5. Calculate angular and linear velocity according to the data sampled in point 3.

6. Check if it has reached it's final destination or goal. If that is not the case, go to point 1.

I/O module
This module contains code for every sensor on the robot, including output to the motors. This module will be responsible for connecting every other module together, it will be the core code of the robot.

Camera module
The project includes a Raspberry Pi camera. We will use the standard raspberry Pi camera library to receive images from the camera. Then we will use OpenCV to convert the image to an array of pixels.

Distance sensor
The project includes a distance sensor. The GoPiGo library contains a function that returns the distance to the obstacles in millimetres which we can use further in the code.

Motor I/O
A function to control the wheels of the robot. The function should take two parameters, one for the speed of the wheel and another that defines which wheel to set the speed.

Image processing module

- Get the image as an array of pixels from the I/O module.
- Convert the colors in the array to black/white.
- Use the Hough transform algorithm to find the edges of the track.
- Find the middle of the road.
- Return the middle of the road and the robots offset to the middle.

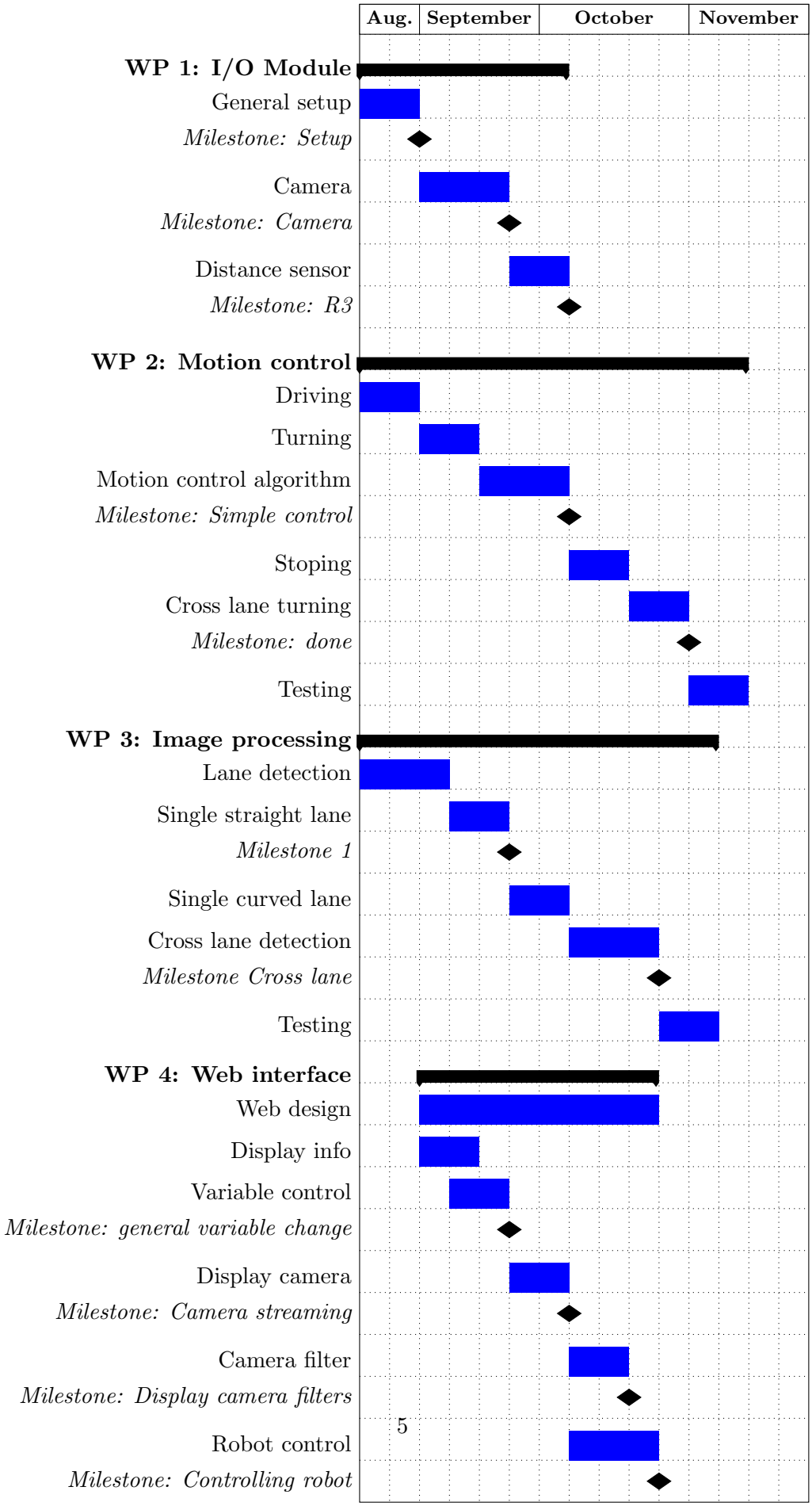
User interface
The user interface is a web page that is locally hosted on the raspberry pi. The webserver is running within the python program. The web page is using API calls to collect data / give command to the python program. All of this is happening asynchronous with the rest of the python code.

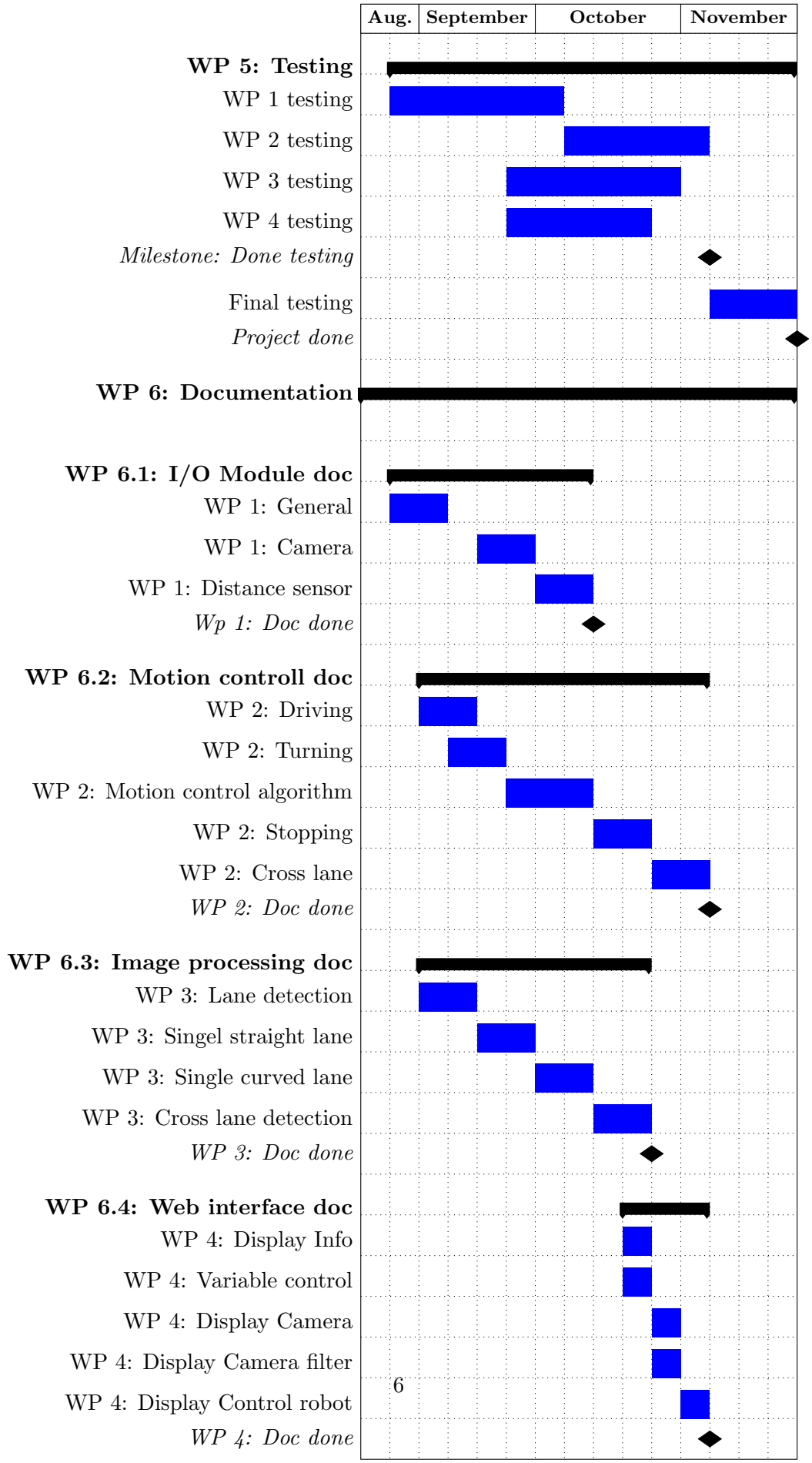
Installation and deployment of the solution (Implementation)

1. Installing the raspbian OS on a microSD card.
2. Setting up the raspberry pi to host a WIFI network.
3. Creating users for each group member on the linux system.
4. Setting up SSH.
5. Installing the GoPiGo libraries from github.
6. Opening port 80 on the firewall for the web interface.
7. Setting up a service in systemctl to run the python code when linux is booting.

List of responsibilities:

Work package (WP)	**Responsibility**
I/O module	x person
Motion control module	x person
Image processing module	x person
Web interface	x person
Testing	x person
Documentation	x person


Aug.
September
October
November
WP 1: I/O Module
General setup
Milestone: Setup
Camera
Milestone: Camera
Distance sensor
Milestone: R3
WP 2: Motion control
Driving
Turning
Motion control algorithm
Milestone: Simple control
Stoping
Cross lane turning
Milestone: done
Testing
WP 3: Image processing
Lane detection
Single straight lane
Milestone 1
Single curved lane
Cross lane detection
Milestone Cross lane
Testing
WP 4: Web interface
Web design
Display info
Variable control
Milestone: general variable change
Display camera
Milestone: Camera streaming
Camera filter
Milestone: Display camera filters
Robot control
Milestone: Controlling robot


Aug.
September
October
November
WP 5: Testing
WP 1 testing
WP 2 testing
WP 3 testing
WP 4 testing
Milestone: Done testing
Final testing
Project done
WP 6: Documentation
WP 6.1: I/O Module doc
WP 1: General
WP 1: Camera
WP 1: Distance sensor
Wp 1: Doc done
WP 6.2: Motion controll doc
WP 2: Driving
WP 2: Turning
WP 2: Motion control algorithm
WP 2: Stopping
WP 2: Cross lane
WP 2: Doc done
WP 6.3: Image processing doc
WP 3: Lane detection
WP 3: Singel straight lane
WP 3: Single curved lane
WP 3: Cross lane detection
WP 3: Doc done
WP 6.4: Web interface doc
WP 4: Display Info
WP 4: Variable control
WP 4: Display Camera
WP 4: Display Camera filter
WP 4: Display Control robot
WP 4: Doc done