

# Mechatronics Project

# Autonomous Guided Vehicle

Supervised by - Prof. Dr.-Ing. Jörg Wollert, Victor Chávez-Bermúdez

06. April 2022

-Aadithya Ramamurthy, Michael Knoll, Rachana Kodilla, Sameer Tuteja &  
Vedhashruthi Harinath

# Outline

---

## Motivation & Starting Point

## System Design

- > ROS
- > PLC
- > Power Supply
- > Motor Controller

## Conclusion & Outlook

# Motivation & Starting Point

## Introduction

---

- Great demand to connect the workstations in the Industry 4.0 model factory at the AMA.
- Task to design, build and test an autonomously driving shop floor delivery vehicle.
  - Carry a load of 100kg
  - Costs a maximum of 4000€
  - Being industry compliant

# Motivation & Starting Point

## Introduction

---

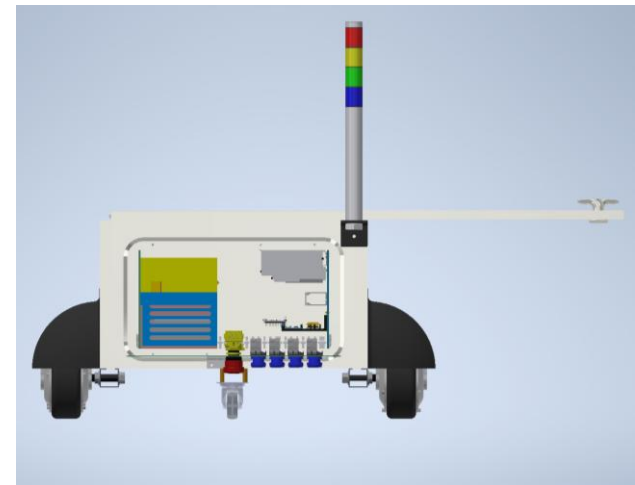
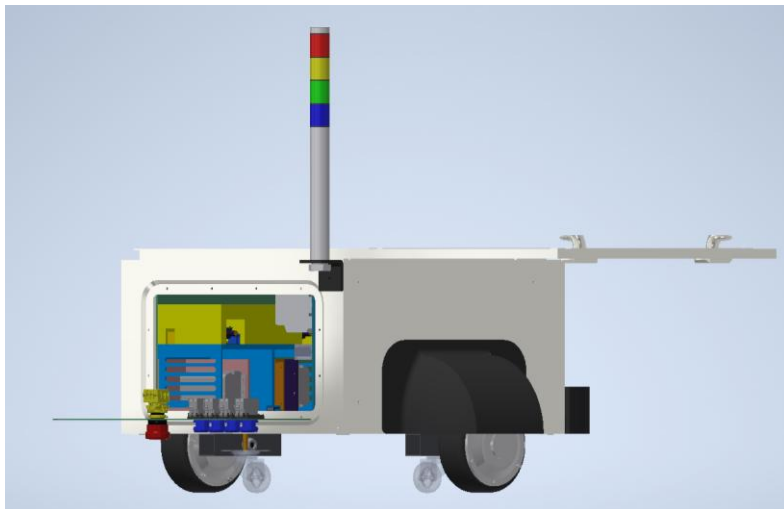
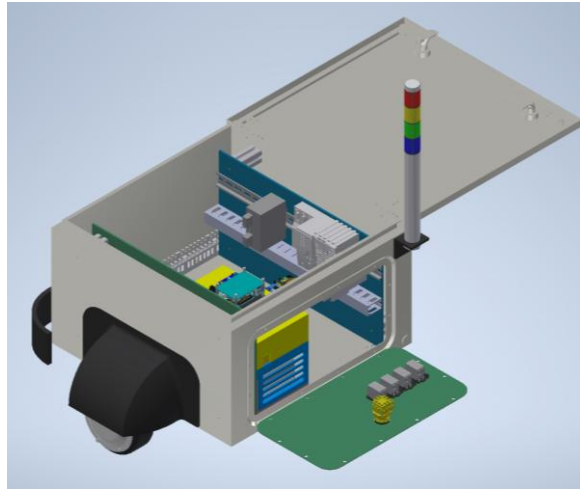
- Most Components already selected and purchased.
- System Design made.

### ➤ Requirements

- Obstacle detection and step detection
- Line following and station detection (Stage 1)
- Localisation and autonomous movement (Stage 2)
- ROS Communication

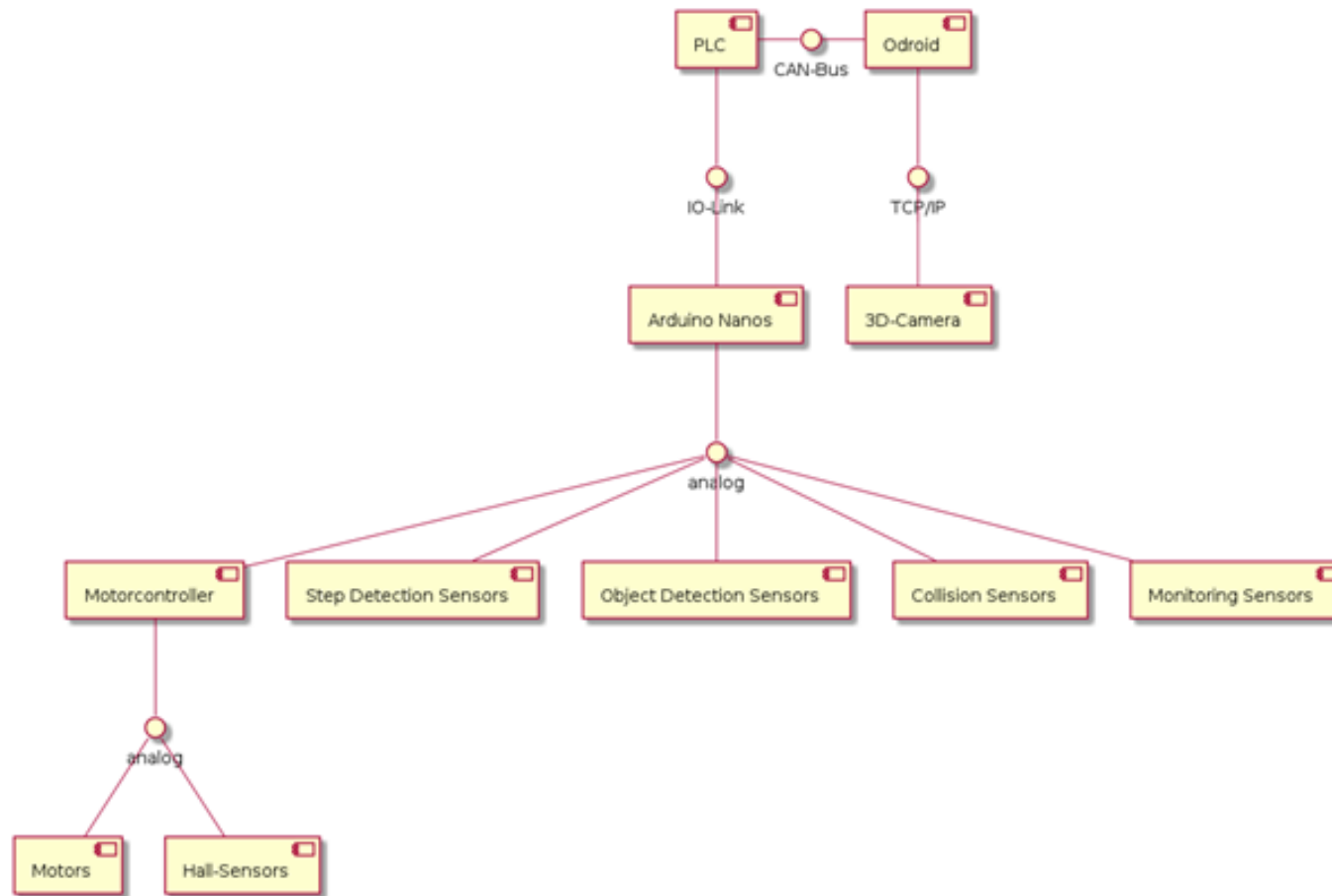
# Motivation & Starting Point

## 3D Model



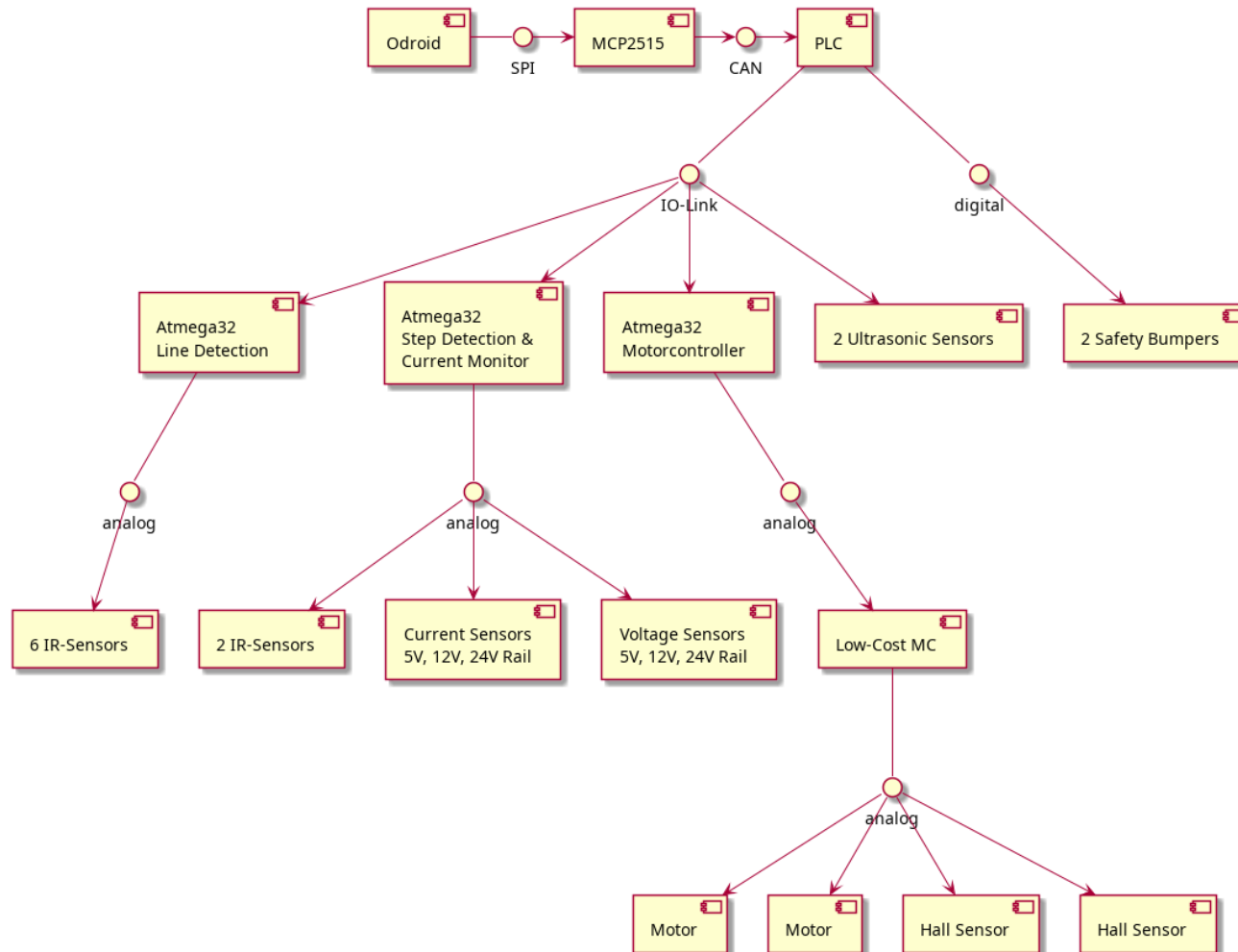
# Motivation & Starting Point

## Initial System Diagram



# Motivation & Starting Point

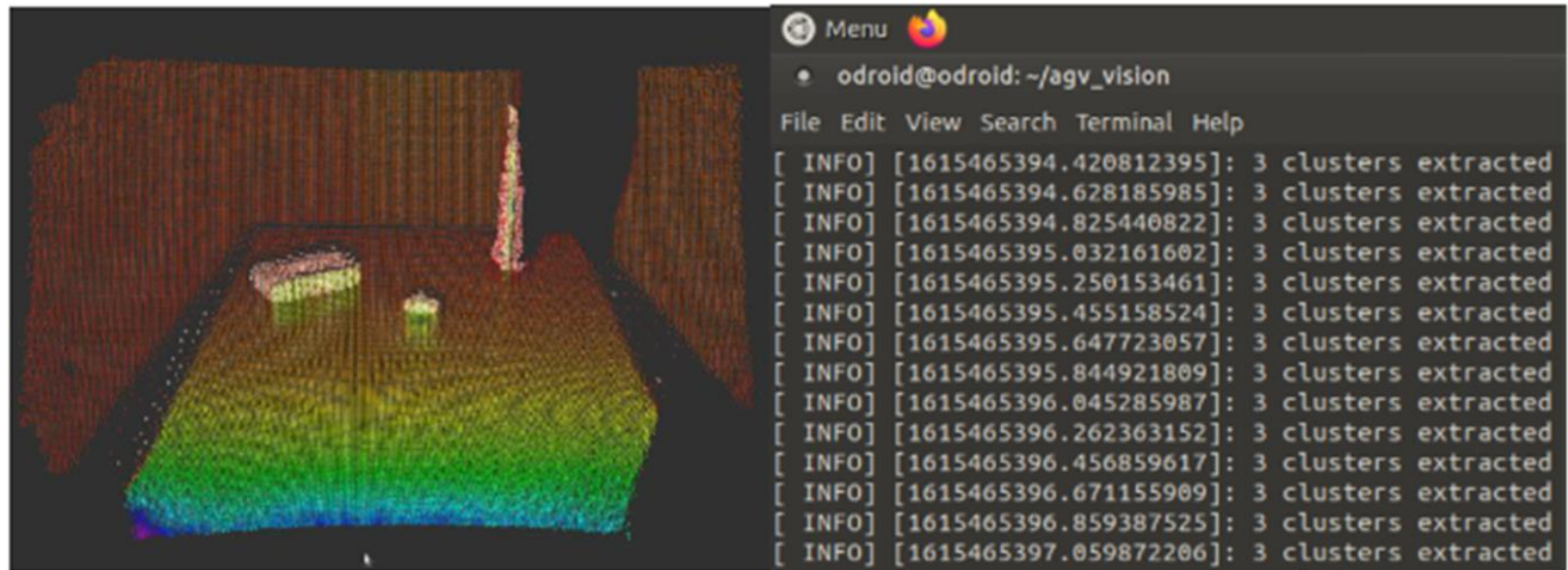
## System Diagram Reviewed



# ROS

## Obstacle detection

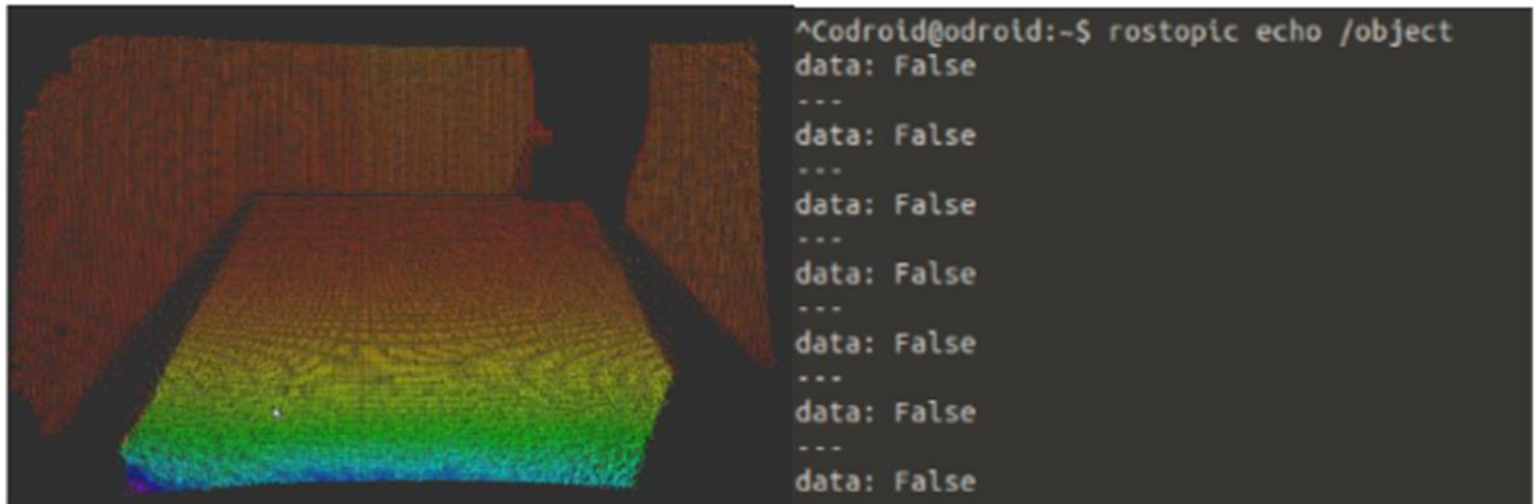
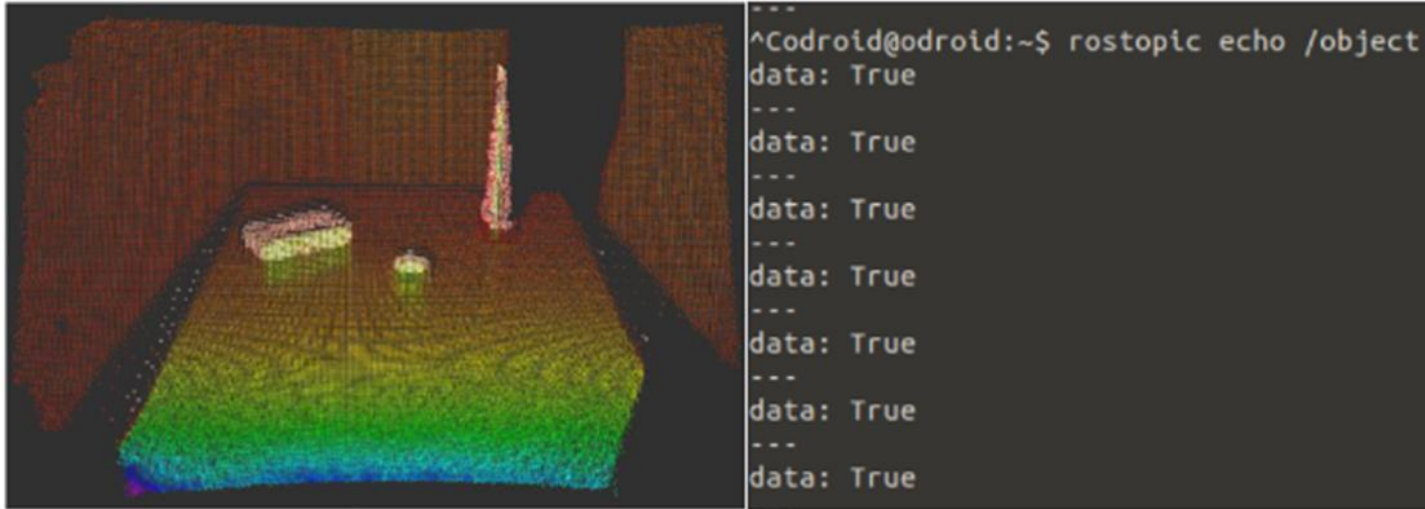
- Obstacle detection using IFM 3D Camera was tested which was done by the previous groups.
- Planar segmentation and Euclidianclustering was done on the point cloud image to detect the objects as clusters.





# ROS

## Obstacle detection



# ROS

## Obstacle detection

---

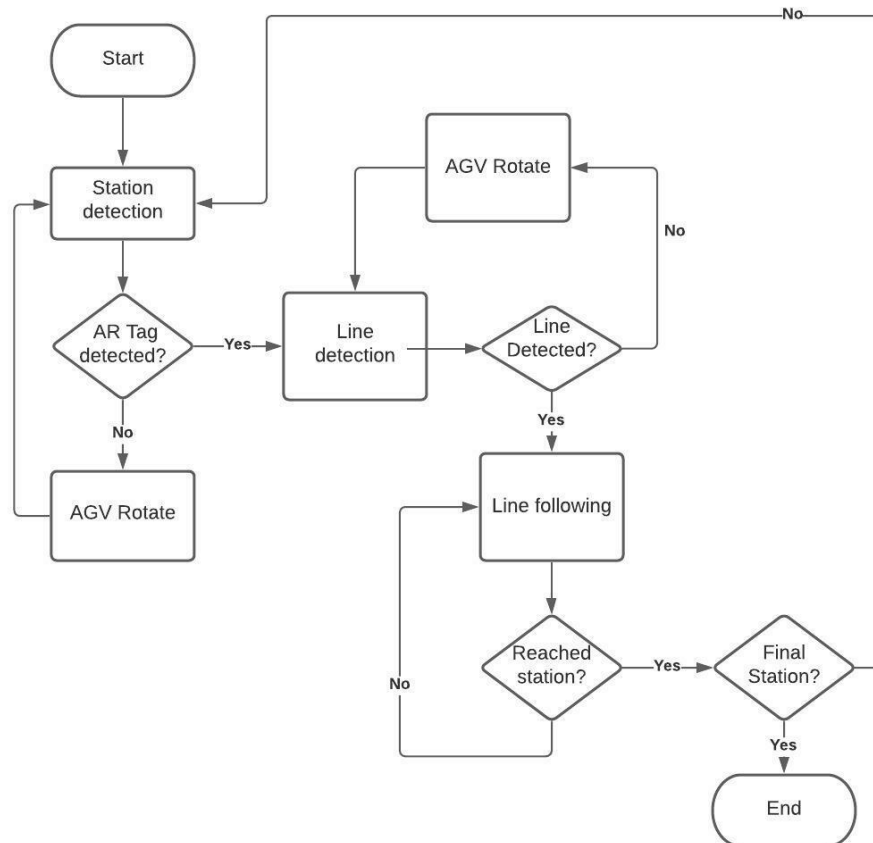
- A Boolean variable is used and set to false if number of clusters are zero and true if its greater than zero.
- This was published on a topic and a subscriber to this topic was used to send messages regarding the presence of obstacles

# ROS

## Station detection

### AR Tags

FLOW CHART FOR STATION DETECTION USING AR TAGS AND LINE FOLLOWING OF AGV



# ROS

## Station detection

---

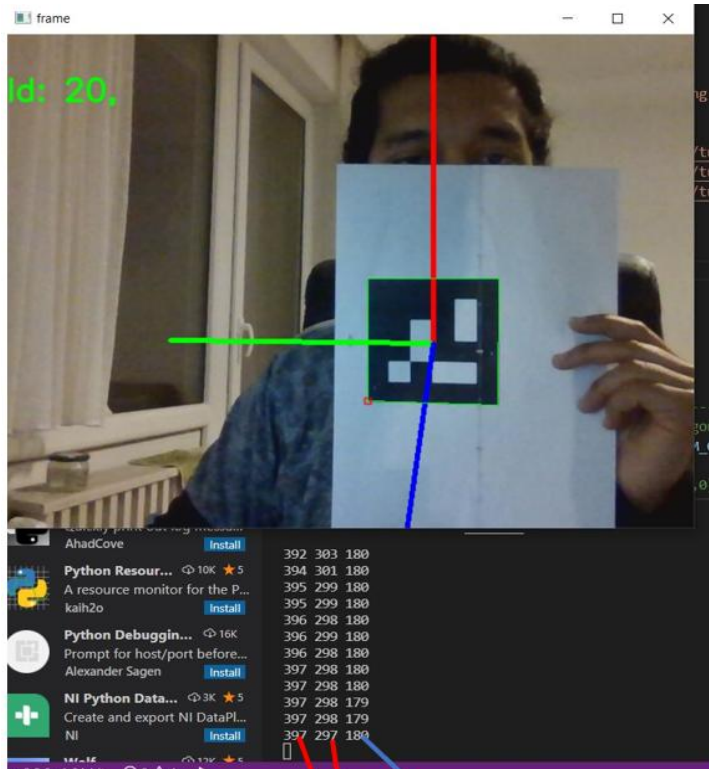
### AR Tags

- For the testing of AR Tag detection, ArUco markers were used.
- Once the code was run and executed, the marker ID was detected, the location of its center point was obtained in terms of X and Y coordinates and its orientation was shown when the marker was placed in front of the camera.

# ROS

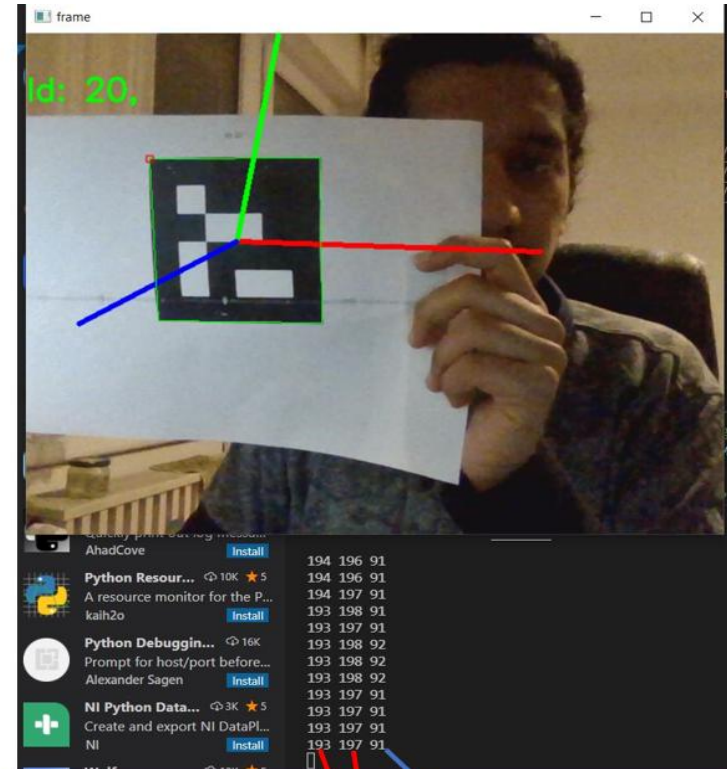
## Station detection

## AR Tags



Position of the marker

Orientation of the marker



Position of the marker

Orientation of the marker

# ROS

## Station detection

---

### AR Tags – Further system development

- AR Tag detection testing was done using the laptop web camera. It must be tested on a USB camera and must be implemented on ROS on Odroid.

# ROS

## Station detection and localisation

---

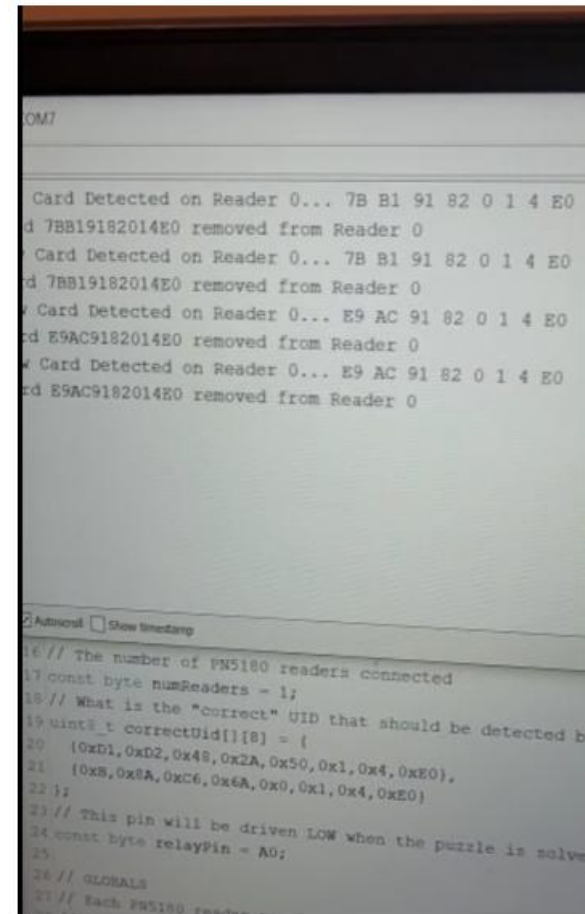
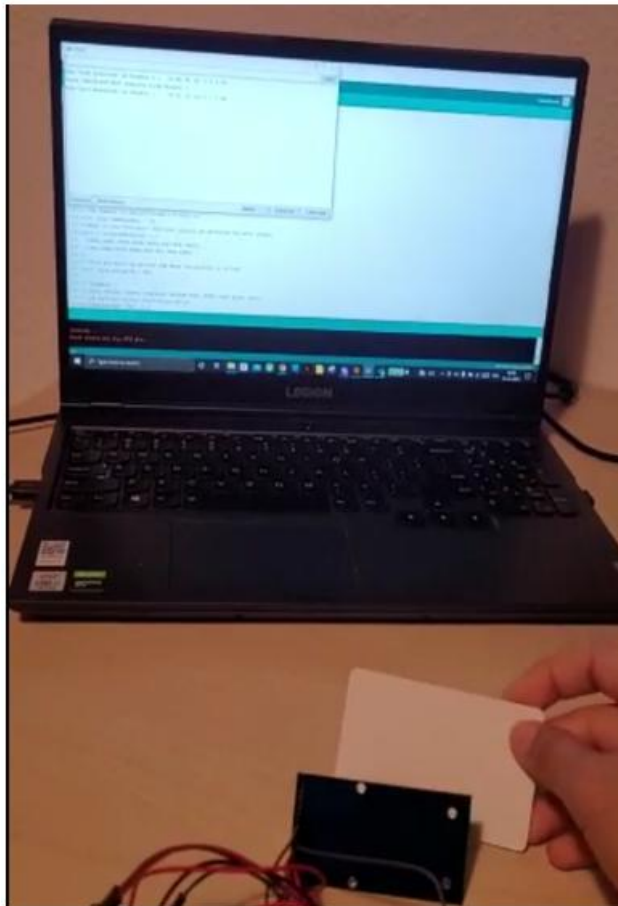
### RFID Tags

- An alternate way to detect stations with line following, can also be as a redundant system with AR Tags. Further, it can be used for localisation of the bot.
- RFID Tag detection was done which were detected by PN5180 -NFC Reader.
- A NodeMCU ESP32 was used to test the PN5180 reader

# ROS

## Station detection and localisation

## RFID Tags





# ROS

## Station detection and localisation

---

- The detection of RFID tags using the NFC Reader on nodeMCU ESP32 worked on SPI communication interface. To make it work on droid, the SPI interface had to be enabled on the Odroid.

# ROS

## Station detection and localisation

---

### SPI on Odroid

- Device Tree Overlay was used to enable the GPIO functions.
- Enabling the communication protocols SPI, I2C, UART and PWM using device tree overlay.
- Enabling the SPI feature on the board.
- To check the details of the GPIO of the Odroid, Wiringpi was installed and used.

# ROS

## Station detection and localisation

### SPI on Odroid

- Yellow indicates that the SPI MOSI(19) and MISO(21) pins are connected using a jump cable and the SPI communication is taking place between them and red indicates no connection.

```
root@odroid:~# ./spidev_test -D /dev/spidev* -s 1000000 -b 8
spi mode: 0
bits per word: 8
max speed: 1000000 Hz (1000 KHz)
01 02 03 04
root@odroid:~# ./spidev_test -D /dev/spidev0.0
spi mode: 0
bits per word: 8
max speed: 500000 Hz (500 KHz)
FF FF FF FF
```

# ROS

## Station detection and localisation

---

### Further system development

- RFID Tags detection was tested using the PN5180 NFC reader using a NodeMCU ESP32. It works on SPI communication.
- SPI communication on Odroid N2+ was enabled and further the code for RFID Tag detection has to be tested on Odroid N2+ using SPI Communication
- It has to be implemented on ROS for an alternative way to detect stations and further for localisation and autonomous movement of the bot.

# System Design

## ROS - Communication

---

- Execution of speed test with
  - CAN interface
  - OPCUA server/client configuration

resulted in faster data rates of the CAN interface

> Toggle Digital Output by exchanging 8-bit variable

CAN	OPCUA
10 Hz	0.5 Hz

# System Design

## ROS - Communication

- Definition of AGV CAN Messages

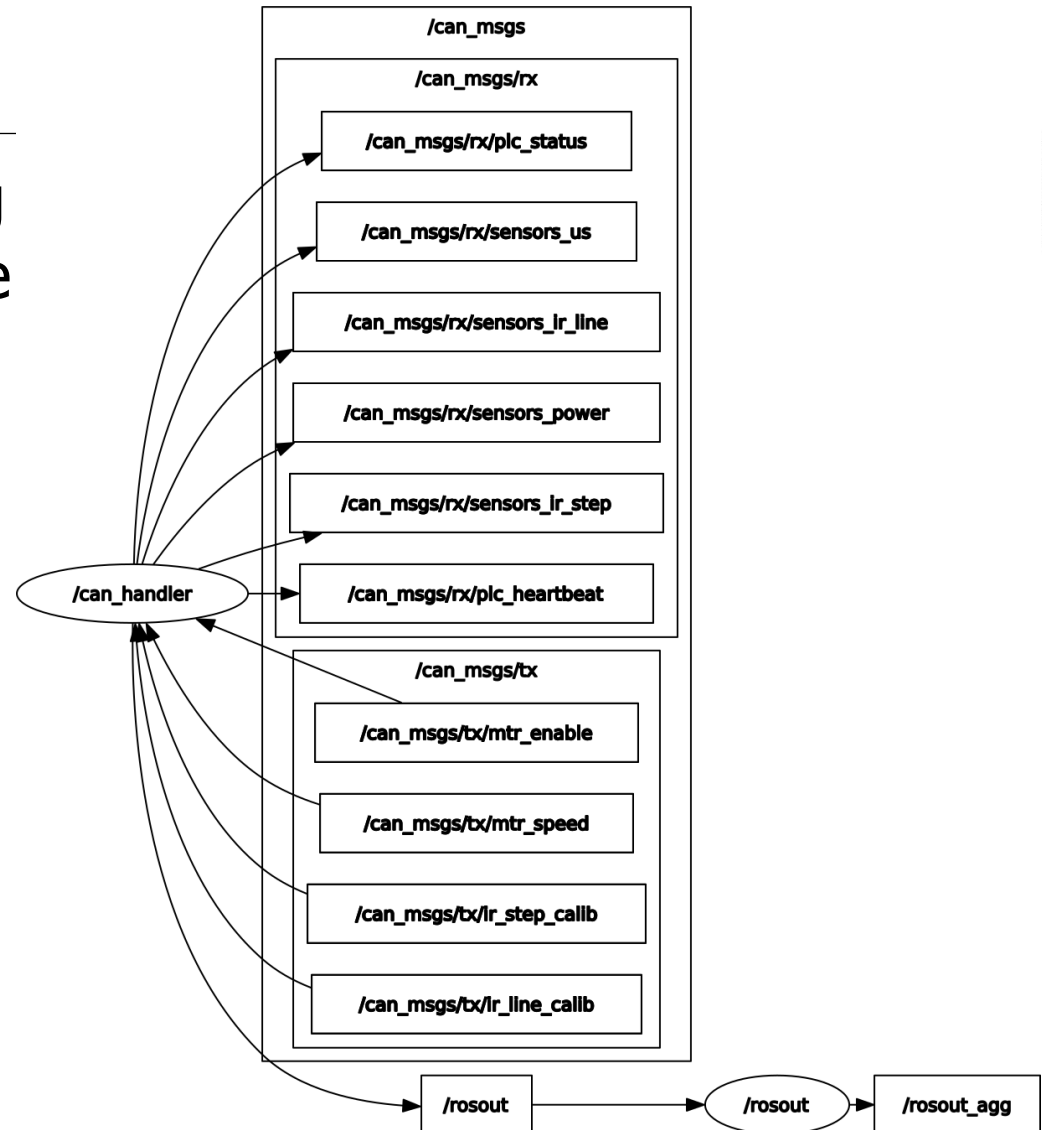
ID	Name	Sender	Signals	Start bit	Signal type	Bits	Factor	Offset	Min	Max	Unit
0x100	Heartbeat	PLC**	Heartbeat	0	UInt	8	1	0	1	1	-
0x101	IR Line Detection	Odroid*	IR_1_1	0	UInt	8	1	0	0	1	bool
			IR_1_2	8							
			IR_1_3	16							
			IR_1_4	24							
			IR_1_5	32							
			IR_1_6	40							
0x102	IR Step Detection	Odroid*	IR_2_1	0	UInt	8	1	0	0	1	bool
			IR_2_2	8							
0x103	US Object Detection	Odroid*	US_1	0	UInt	16	1	0	0	1	-
			US_2	16							
0x104	Power Monitor 24V	Odroid*	PWR_SENS_24V_V	0	UInt	16	1	0	0	65535	mV
			PWR_SENS_24V_I	16							mA

- Request & Response model

# System Design

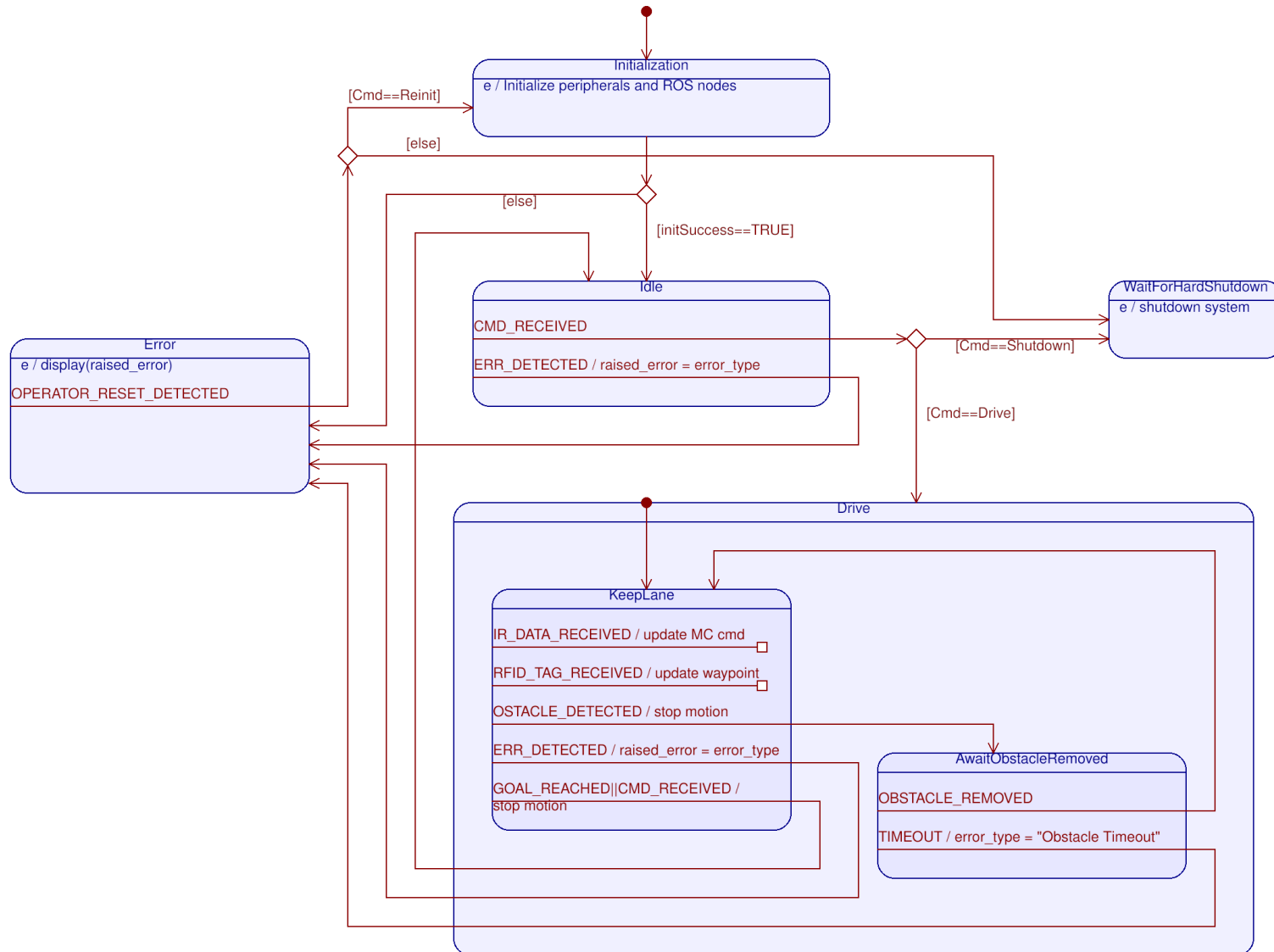
## ROS - Communication

- Decoding / Encoding using Python Module cantools and CAN database file .dbc
- Nodes subscribe to rx & tx topics to receive and send CAN messages



# System Design

## ROS – State Machine





# System Design

## ROS – State Machine

- User Control interface via ROS Topic
- Successful test with simulated PLC

```
/home/odroid/state_ws/src/line-follower/launch/line_follower.launch http://localhost:11311
File Edit View Search Terminal Help
NODES
ROS_MASTER_URI=http://localhost:11311
process[can_handler_node_py_odroid_4825_5373188606200081067-2]: started with pid [4901]
[INFO] [1648643435.906705]: User Interface initialized
[INFO] [1648643438.036695]: CANhandler initialized
[INFO] [1648643440.653140]: Launching RFID scanner
[INFO] [1648643440.657262]: Executing state IDLE
[INFO] [1648643468.880786]: Please set a valid destination first! Use topic /user_ctrl/set_goal
[INFO] [1648643480.068418]: Destination set to A
[INFO] [1648643480.179602]: Destination set to A
[INFO] [1648643480.276366]: Destination set to A
[INFO] [1648643491.369300]: Executing sub state KEEPLANE
[INFO] [1648643505.672734]: Executing sub state AWAITOBSTACLEREMOVED
[INFO] [1648643509.405590]: Executing sub state KEEPLANE
[INFO] [1648643519.058115]: Executing sub state AWAITOBSTACLEREMOVED
[INFO] [1648643549.058216]: Executing state ERROR
[ERROR] [1648643549.062305]: Raised by Obstacle Timeout
[INFO] [1648643549.066279]: Waiting for input by operator on topic /user_ctrl/cmd
[INFO] [1648643573.674763]: Executing state INIT
[INFO] [1648643573.679622]: Launching user_node
started roslaunch server http://odroid:33291/

SUMMARY
=====
PARAMETERS
* /roscpp: noetic
* /rosversion: 1.15.14

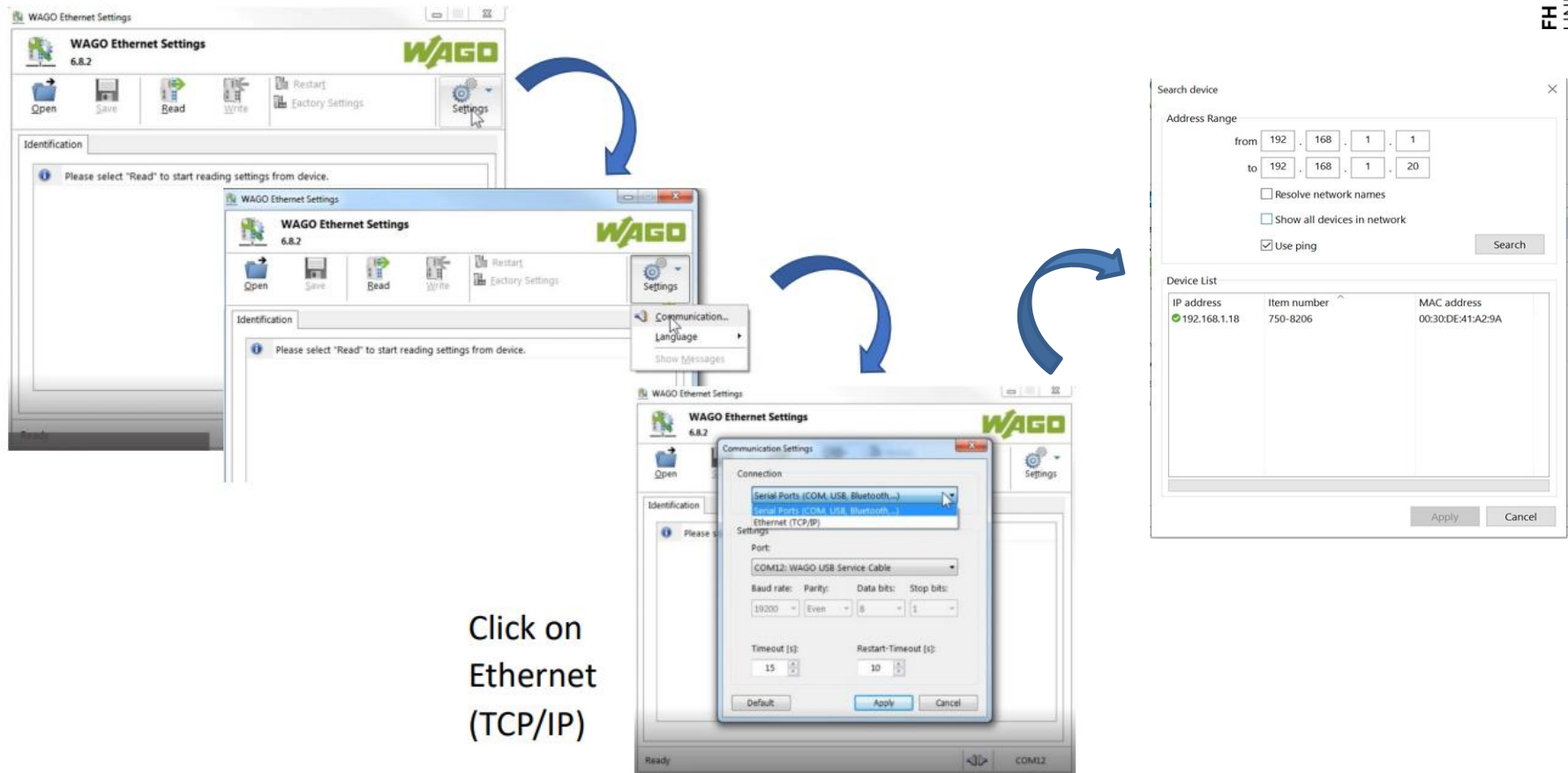
NODES
ROS_MASTER_URI=http://localhost:11311
process[user_node_py_odroid_4825_1133004756369066880-3]: started with pid [5155]
[INFO] [1648643574.225982]: Launching can_handler
started roslaunch server http://odroid:35569/

SUMMARY
=====
PARAMETERS
* /roscpp: noetic
* /rosversion: 1.15.14

NODES
ROS_MASTER_URI=http://localhost:11311
process[can_handler_node_py_odroid_4825_4959808068553748744-4]: started with pid [5171]
[INFO] [1648643575.125805]: User Interface initialized
[INFO] [1648643577.318310]: CANhandler initialized
[INFO] [1648643579.815984]: Launching RFID scanner
[INFO] [1648643579.823044]: Executing state IDLE
```

# PLC

## Connecting to the Wago Ethernet Settings



# PLC

## Reset the PLC

750-8206 - WAGO Ethernet Settings

750-8206  
WAGO 750-8206 PFC200 CS 2ETH RS CAN DPS

Open Save Read Write Restart Factory Settings Settings

Identification Network PLC Status

Parameter	Edit	Currently used
Address Source	Static Configuration	Static Configuration
IP address	192.168.1.18	192.168.1.17
Subnet Mask	255.255.255.0	255.255.255.0
Gateway	0.0.0.0	0.0.0.0
Preferred DNS-Server	0.0.0.0	0.0.0.0
Alternative DNS-Server	0.0.0.0	0.0.0.0
Time server	0.0.0.0	not available
Hostname		PFC200-41A29A
Domain name	localdomain.lan	localdomain.lan

Interface X1  
Interface X2  
Run WBM  
Interfaces  
☒ Switched  
☐ Separated

Ready 192.168.1.17

192.168.1.17/wbm/#/configuration/networking/tcpip

WAGO

Information Configuration Fieldbus Security Diagnostic

PLC Runtime

Networking

TCP/IP Configuration

Ethernet Configuration

Host-/Domain Name

Routing

Clock

Administration

Package Server

13.10.2021 19:50:45 STOP SYS RUN IO MS NS CAN BF DIA U1 U2 U3 U4

TCP/IP Configuration

If the IP source of a network interface is 'external', it is likely that an application active in the system has adopted the IP configuration for this interface. Changing the source would probably affect the functionality of this application.

Network Details Bridge 1 (br0)

Current IP Address 192.168.1.17

Current Subnet Mask 255.255.255.0

IP Source Static IP

Static IP Address 192.168.1.18

Subnet Mask 255.255.255.0

Submit

# PLC

## Configure in IO-Check 3

The screenshot displays the WAGO IO-Link software interface. The main window is titled "WAGO IO-Link" and shows the configuration for a "0750-0657 4 Port IO-Link Master [Position 3] 01.01.58(08)". The interface includes a menu bar (File, View, Node, I/O module, Settings, Help) and a toolbar with icons for Exit, Open, Save, Connect, Read, Write, Data Frame, Events, Options, and Help.

The "Master Configuration" tab is active, showing the following settings:

- Diagnosis settings:**
  - Enable diagnosis per acyclic channel: ☐
  - Enable diagnosis per KBUS status byte: ☒
- Segmentation Mode:**
  - Mode Port 1: Not fragmented
  - Mode Port 2: Not fragmented
  - Mode Port 3: Not fragmented
  - Mode Port 4: Not fragmented
- Segmentation:**
  - Auto Calc: ☐
  - SIO Byte: Offset 5, Length 1
- Input (Upstream):**
  - Offset Port 1: 6, Length Port 1: 12
  - Offset Port 2: 14, Length Port 2: 2
  - Offset Port 3: 16, Length Port 3: 2
  - Offset Port 4: 20, Length Port 4: 10
- Output (Downstream):**
  - Offset Port 1: 6, Length Port 1: 0
  - Offset Port 2: 14, Length Port 2: 0
  - Offset Port 3: 14, Length Port 3: 0
  - Offset Port 4: 14, Length Port 4: 0

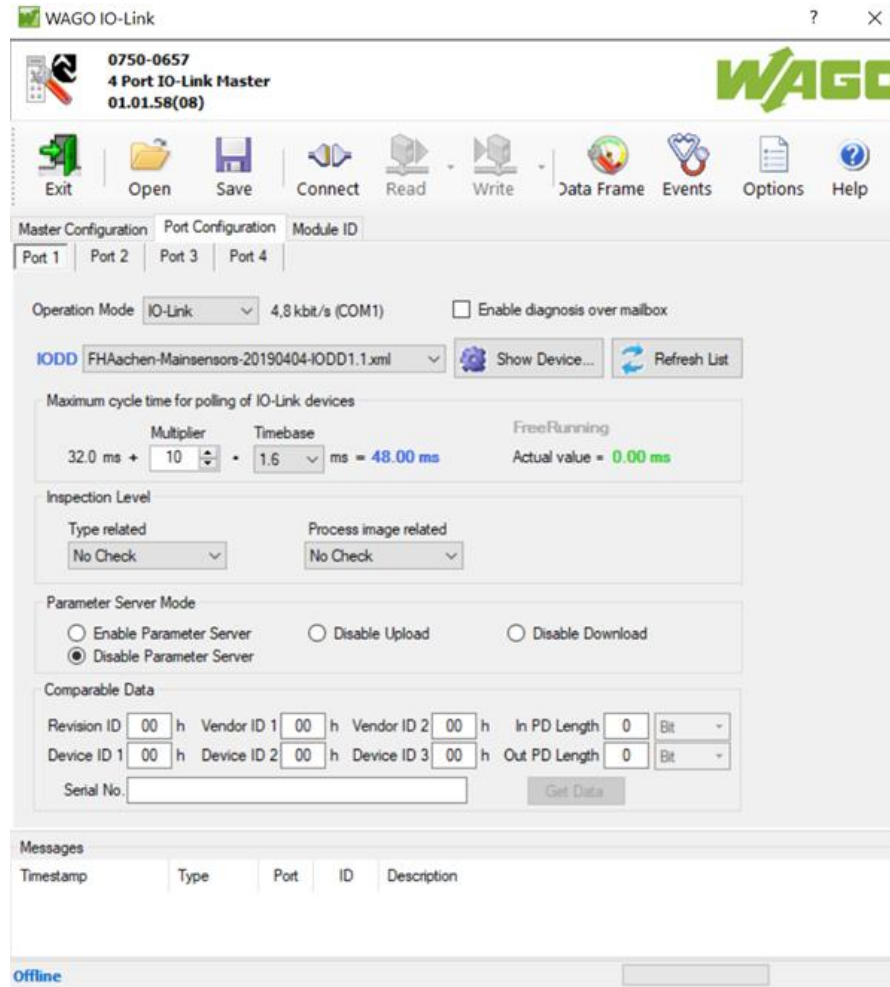
The "Messages" table at the bottom shows a log of events:

Timestamp	Type	Port	ID	Description
05.04.22 - 18:54:48	Error	Mas...	0x1102	

The status bar at the bottom indicates "Offline" and "Mailbox Error 1". The bottom right corner shows the IP address "192.168.1.18" and the parameter "PARAM".

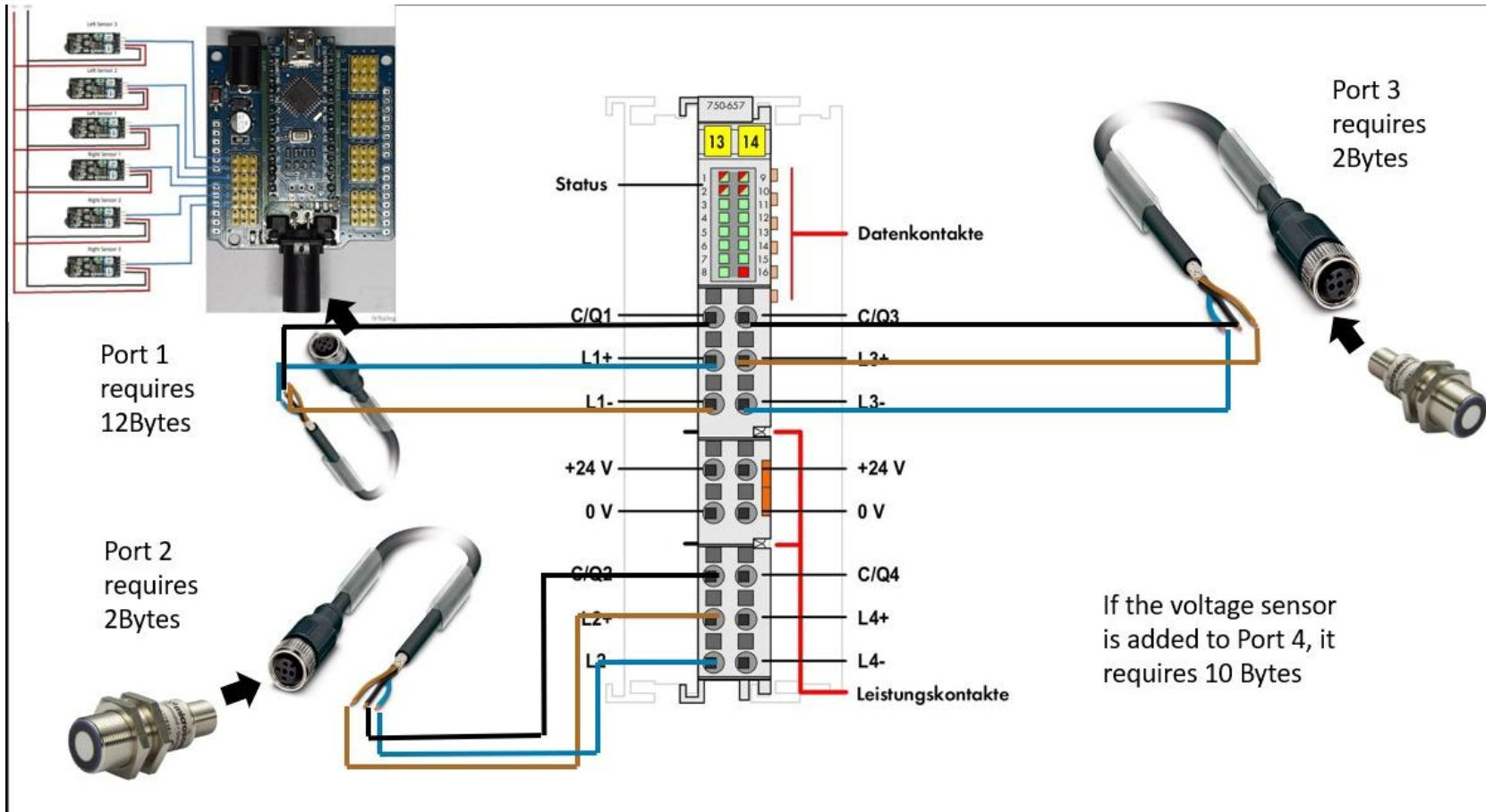
# PLC

## Configure in IO-Check 3



# PLC

## Connecting the sensors to the PLC





# PLC

## IO Link Port Function Blocks

---

- Port 1: IO shield connecting to 6 IR sensors
- Port 2: Ultrasonic Sensor
- Port 3: Ultrasonic Sensor
- Port 4: IO shield connecting to Voltage sensor

Function Block Library: WagoAppIOLink

Function Block: FbIOL\_PortData

```
//Access the IOLink Process Data
IOLProcessData( xEnable:=bEnable,
    bIO_LinkPort:=byIOLinkPort,
    I_Port:=IO_Link_Master, //Name of the IO-Link Master in the "Device Structure" Tab of Ecookit
    pTxBuffer:=ADR(typIOLink_VisuData.abypDOut), //Process data output
    uDiTxnbytes:=SIZEOF(typIOLink_Device_Visu.abypDOut), //Length of the data to send through PD Out
    pRxBuffer:=ADR(typIOLink_VisuData.abypDIn), //Process data input
    uDirxBufferSize:=SIZEOF(typIOLink_VisuData.abypDIn), //Size of the buffer used for recieving PDIn
    xTxTrigger:=PDOutTrigger, //Trigger to send proccess output data
    xCommunicationReset:=xCommReset, //Reset communication
    xError=>typIOLink_VisuData.bStatusInvalid,
    xValid=>typIOLink_VisuData.bStatusValid,
    uDirxNBytes=>typIOLink_VisuData.iPDInSize //Total recieved bytes (Process data input)
);
IOLProcessData.oStatus.ShowResult(sDescription=>typIOLink_VisuData.sDeviceInfo); //Get a String error from the FB
```

# PLC

## Why Microsonic Pico+35?

---

- Variant with 90° angled head
- IO-Link interface —> for support of the new industry standard
- Automatic synchronisation and multiplex operation —> for simultaneous operation of up to ten sensors in close quarters
- UL Listed to Canadian and US safety standards
- Improved temperature compensation —> adjustment to working conditions within 120 seconds
- Smart Sensor Profiles —> more transparency between IO-Link Devices



# PLC

## IR Sensor for Line following

---

### **WHICH?**

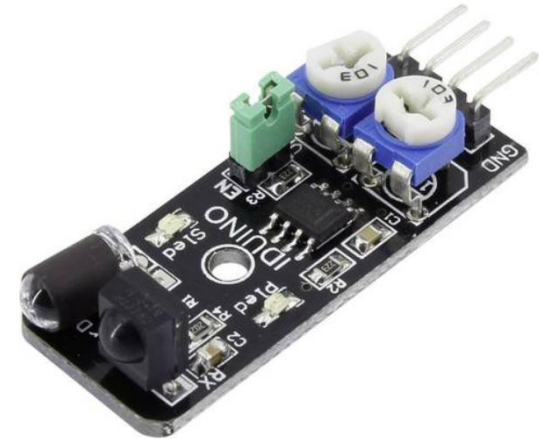
IDUINO IR SENSOR

### **WHY?**

- detection distance :2-40cm
- Cost effective
- Working voltage: DC 3.3V-5V

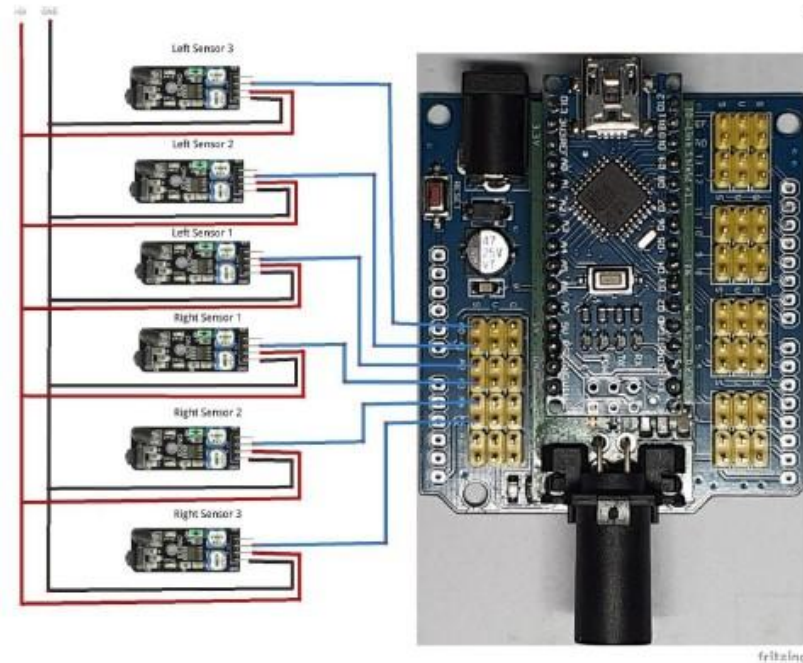
### **HOW MANY?**

6 IR Sensors



# PLC

## IR sensor to Arduino Nano schematic diagram



# PLC

## Softwares used

---

- WAGO IO CHECK 3  
To check the IO Link Communication
- WAGO e!COCKPIT  
To program the PLC
- ARDUINO IDE  
To program the Arduino
- PYTHON 3.10  
To communicate with the IFM IO Link Master

# PLC

## IR sensor values

6 IR sensors.ecp - eiCOCKPIT

FILE HOME VIEW NETWORK PROGRAM DEBUG

Disconnect Simulate Application Start Stop Program Download Multiple Download Boot Application Download Edit Offline Compare applications Online Change Rebuild Generate Code Clean Clean All Output cross references CODESYS V3 PLCopen XML Import Import Export Export Input Assistant... Auto Declare... Refactoring... Advanced... Go To Source Position

Library Manager Could not open library 'WagoSysVirtualBuffer'. (Reason: The placeholder library 'WagoSysVirtualBuffer' could not be resolved.) Display message view

Program Structure Network/Devices IOLink\_Demo

Project Library (POUs) GlobalTextList Library Manager Applications Application (PFC200\_2ETH\_RS\_CA) typIOLink\_Device\_Visu Visualizations Library Manager BytesToFloat Config IOLink\_Demo IOLink\_PD\_Access Task Configuration Visualization Manager

Expression	Type	Value	Prepared value	Address	Comment
IOLink_SensorPort	BYTE	1			
sSensorName	STRING	'Arduino'			Display Name for ... visualization d...
IR_L3	REAL	1023			Temperature obtained from custom...
IR_L2	REAL	47			
IR_L1	REAL	9			
IR_R3	REAL	1023			
IR_R2	REAL	255			
IR_R1	REAL	40			
oIOLinkSensor	IOLink_PD_Acc...				
IR_array	ARRAY [0..5] O...				

```

1 oIOLinkSensor(I_Port:=IO_Link_Master,bEnable:=TRUE,sName:=sSensorName);
2 IF oIOLinkSensor.info.bStatusValid THEN
3
4   IR_R3[1023] := oIOLinkSensor.info.abvPDIn[0]255 + SHL(WORD(oIOLinkSensor.info.abvPDIn[1]3), 0);
5
6   //Bits_array[index] := Bits;
7   //index := index +1;
8   //Convert RAW Bytes to readable data in REAL format
9
10  ELSE
11    //If the IO-Link Status for the IO-Link Arduino port is not valid
12    IR_R3[1023] :=0.0;
13  END_IF
14
15  IF oIOLinkSensor.info.bStatusValid THEN
16
17    IR_R2[255] := oIOLinkSensor.info.abvPDIn[2]255 + SHL(WORD(oIOLinkSensor.info.abvPDIn[3]0), 0);
18
19  ELSE
20    //If the IO-Link Status for the IO-Link Arduino port is not valid
21    IR_R2[255] :=0.0;

```

Product Catalog

- Operation and Monitoring
- Controllers
- I/O Systems
- Infrastructure
- Basic Network Functions
- Other communicating parties
- Discontinued

Device Structure Program Structure

Product Catalog Settings

# Power Supply Distribution

---

1. 24 V (without DC/DC converter) – wire for connection used 6mm<sup>2</sup> (uninterrupted current flow)
  - Motor Driver
  - Motors
  - Voltage Sensor (voltage measurement)
  - Fan
  
2. 24V (with DC/DC converter) – wire for connection 0.75 mm<sup>2</sup>
  - PLC
  - IFM
  - 24V Sensors

# Power Supply Distribution

---

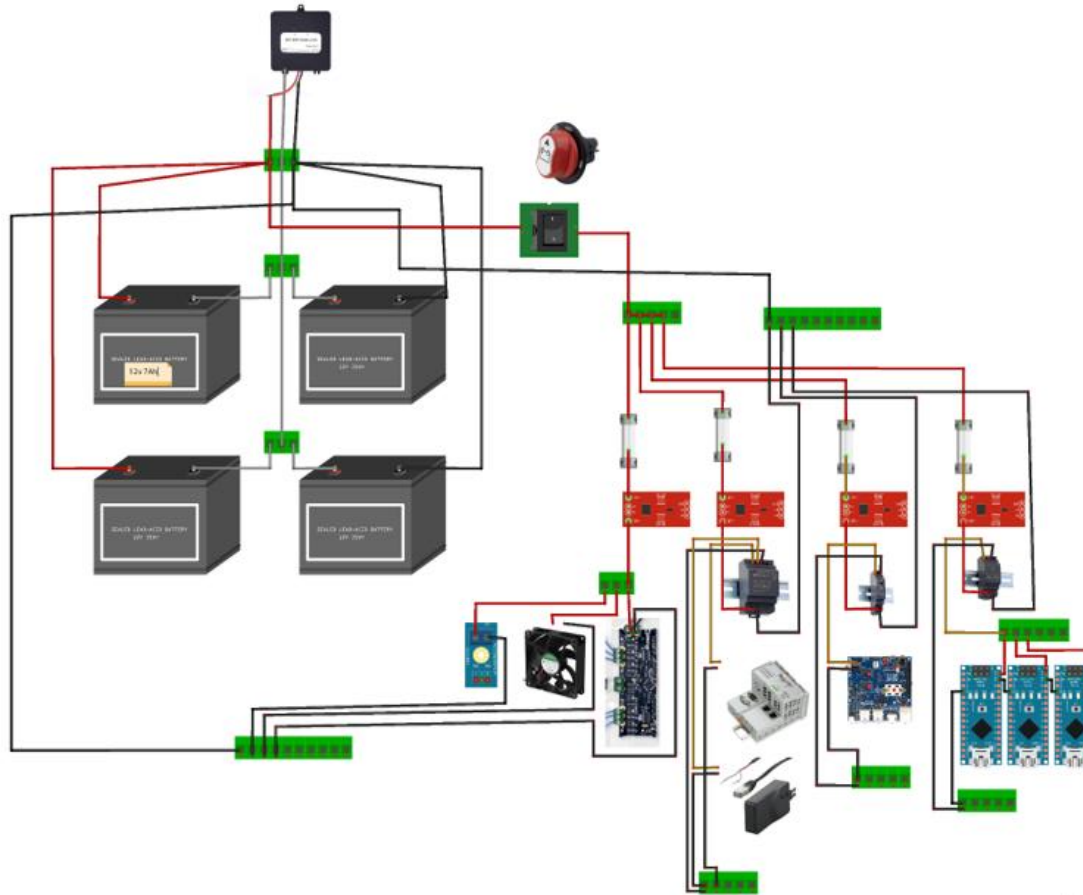
3. 12V (with DC/DC converter) – wire for connection 0.75 mm<sup>2</sup>

- Odroid N2+

4. 5V (with DC/DC converter) – wire for connection 0.75 mm<sup>2</sup>

- Arduino Nano
- 5V Sensors

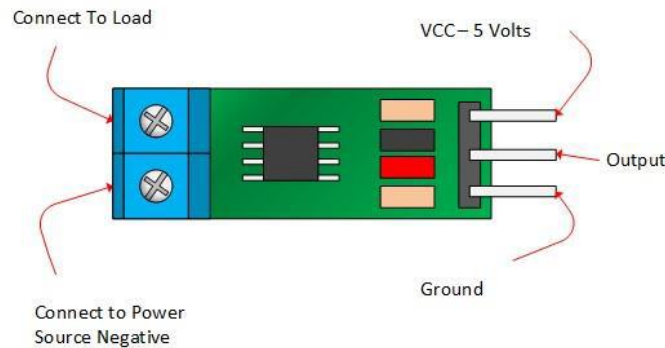
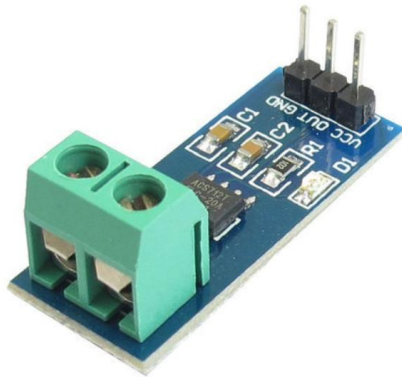
# Power Supply Distribution



# Power Supply

## Power monitoring sensors

- Current sensor



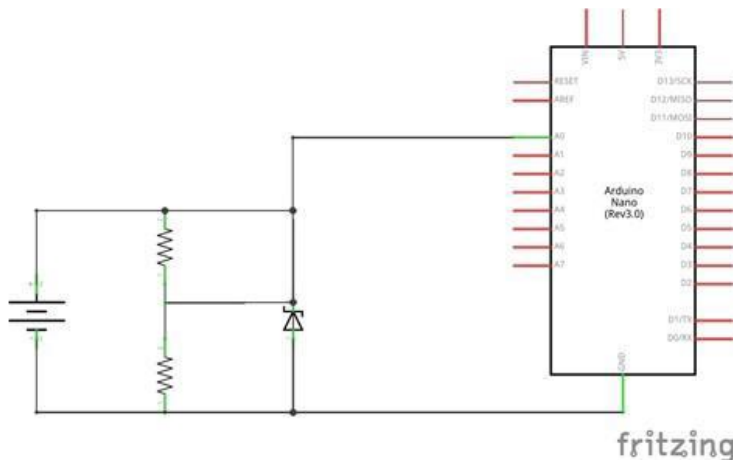
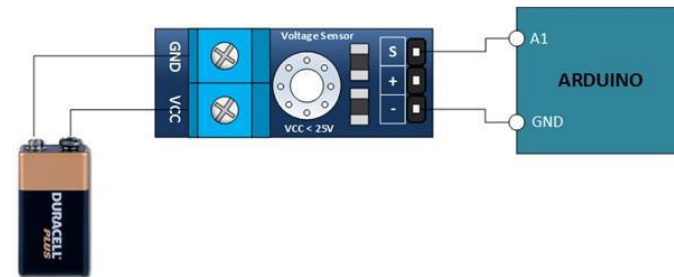
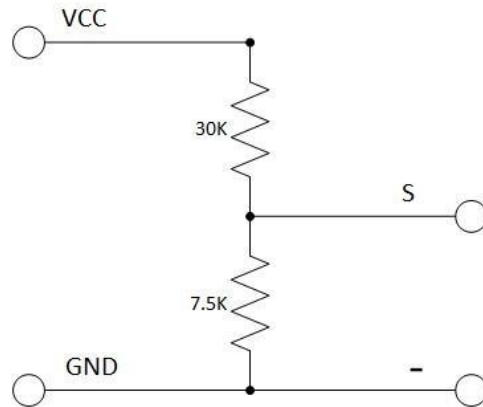
CurrentSensor
<ul style="list-style-type: none"> <li>- uint8_t sensorPin</li> <li>- const int sensitivity</li> <li>- const int offsetVoltage</li> <li>- int adcValue</li> <li>- uint16_t adcVoltage</li> <li>- uint16_t currentValue</li> <li>- uint16_t adValue</li> <li>- int inputVoltageStandard</li> </ul>
<ul style="list-style-type: none"> <li>+ CurrentSensor(uint8_t pin)</li> <li>+ uint16_t CurrentValue()</li> <li>+ uint8_t CurrentMapValue()</li> <li>+ outData CurrentMapValuesBytes()</li> </ul>



# Power Supply

## Power monitoring sensors

- Voltage sensor



VoltageSensor
<ul style="list-style-type: none"> <li>- uint8_t sensorPin</li> <li>- uint16_t adcVoltage</li> <li>- uint16_t inVoltage</li> <li>- uint16_t R1</li> <li>- uint16_t R2</li> <li>- uint16_t refVoltage</li> <li>- int adcValue</li> <li>- uint16_t adValue</li> </ul>
<ul style="list-style-type: none"> <li>+ VoltageSensor(uint8_t pin)</li> <li>+ uint16_t VolatgeValue()</li> <li>+ uint8_t VoltageMapValue()</li> <li>+ outData VoltageMapValues Bytes()</li> </ul>

# Motor controller

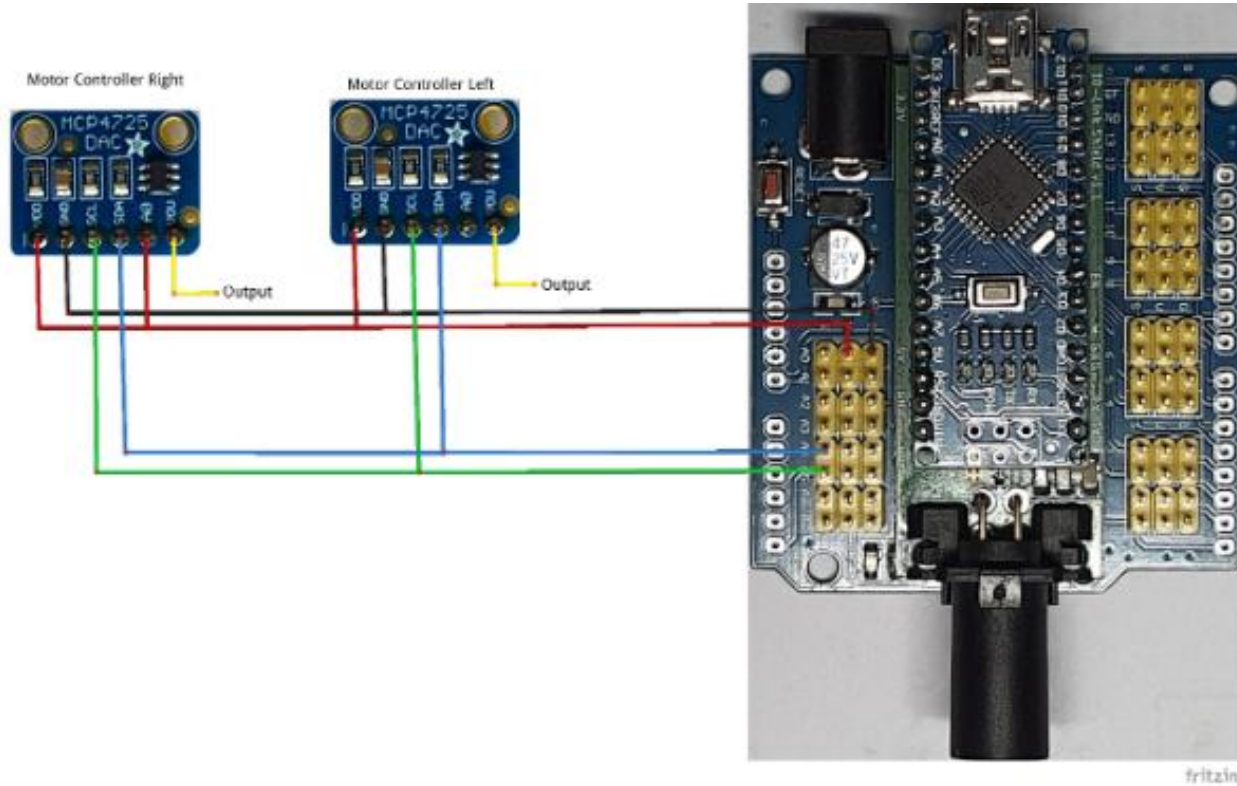
## Ebike controller

---

- Ebike controller is controlled using analog signal. The controller responds to analog signal between the range 1.2v to 3.8v.
- To supply analog output, DAC MCP4725 is used. It is connected to Arduino nano using I2C communication.
- For the project's application, the MCP4725 receives commands from PLC via IO-Link Shield. The shield mounts Arduino nano to which the MCP4725 module is connected. The output (VOU pin) is connected to the respective motor controller.

# Motor controller

## Ebike controller



Motor Controller Right	Motor Controller Left	Arduino Nano
VCC	VCC	+5V
GND	GND	GND
SCL	SCL	A5
SDA	SDA	A4
A0	-	+5V

# Motor controller

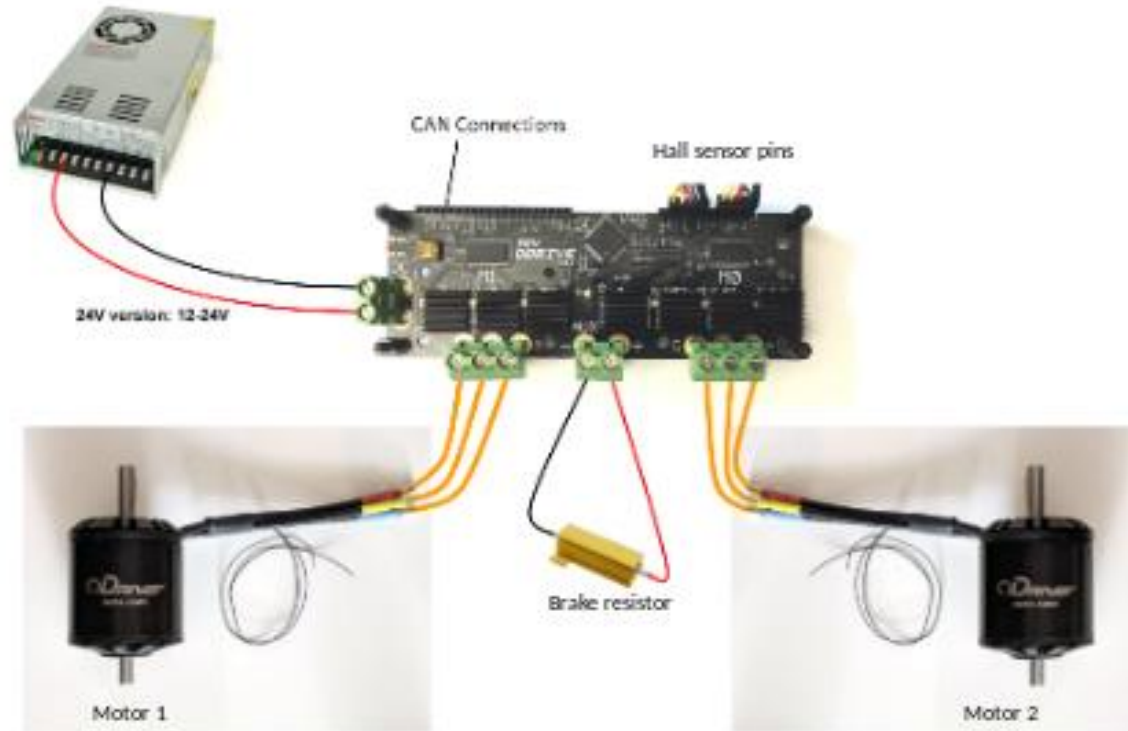
## Odrive controller

---

- The project uses hoverboard BLDC motors. Odrive motor controller allows controlling two BLDC motors simultaneously using one board.
- The commands can be sent via various communication protocols such as UART, CAN and USB. The motors will be connected in similar manner. BLDC motors consist of hall sensors which are connected to ODrive board.

# Motor controller

## Odrive controller



# Motor controller

## Current status

---

- The odrive motor controller had issues and the the controllers from the previous batch is being used now.

# Motor controller

---

The data sent from the PLC is received by the IO-Link shield. The 7 bytes of data received is structured in the following order-

MTR\_ESTOP, MTR\_1\_ENABLE, MTR\_2\_ENABLE,  
MTR\_1\_SPEED\_H, MTR\_1\_SPEED\_L,  
MTR\_2\_SPEED\_H, MTR\_2\_SPEED\_L.

# Motor controller

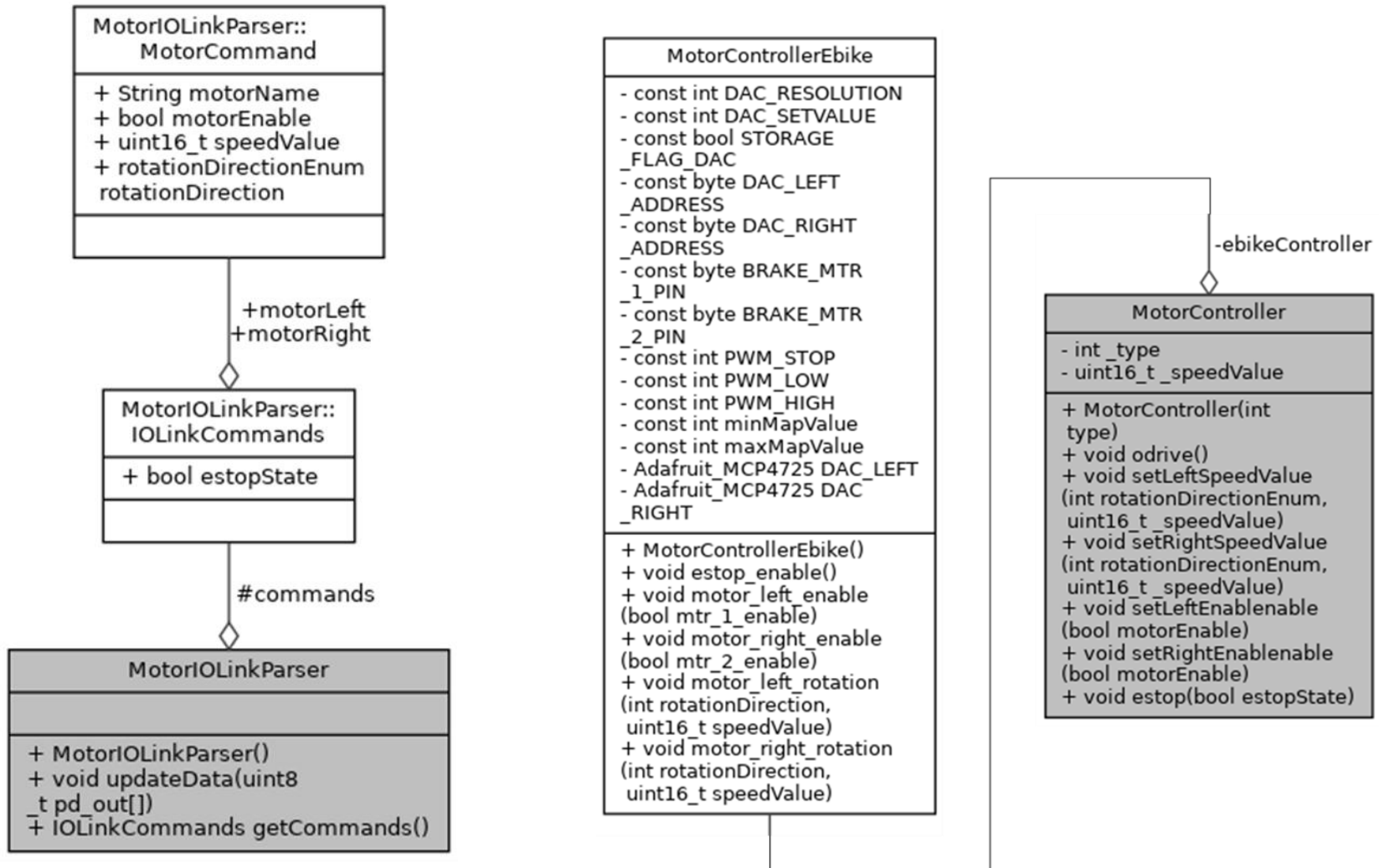
---

The Arduino nano also sends feedback to the PLC via IO-Link communication. The data is of 6 bytes structured in the following order:

MTR\_ESTOP\_STATUS,MTR\_ENABLE\_STATUS,MTR\_1  
\_SPEED\_STATUS\_H,MTR\_1\_SPEED\_STATUS\_L,  
MTR\_2\_SPEED\_STATUS\_H,  
MTR\_2\_SPEED\_STATUS\_L.



# Motor controller



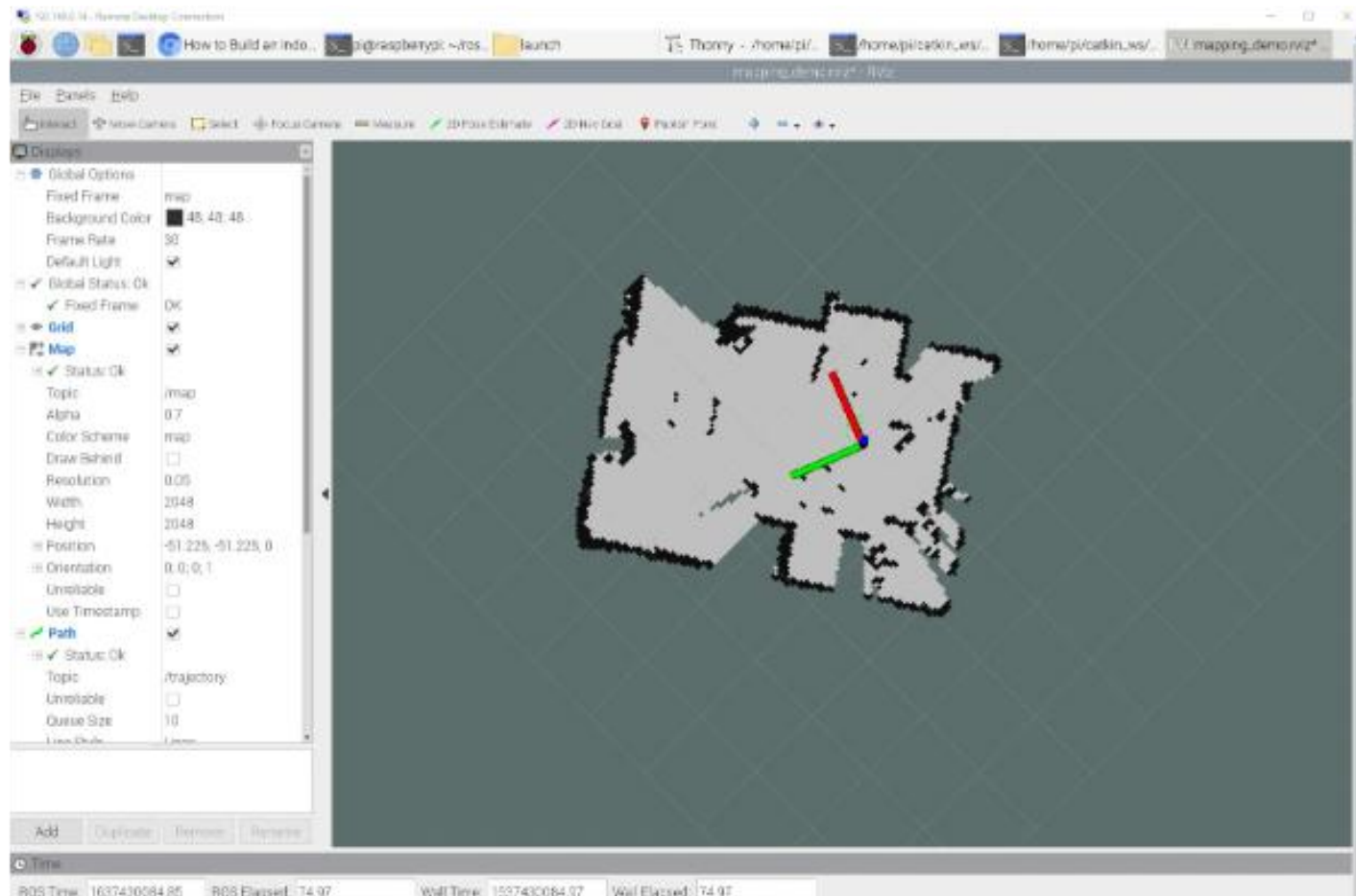
# Lidar

---

- For the low-cost project, we considered using RP LIDAR a1 by Slamtec.
- The lidar can provide a variable frequency of 5Hz and 10Hz.
- It can detect objects within the circumference with 12 meters radius

# Lidar

## Future implementation for navigation with map generation



# Conclusion & Outlook

---

- Calibration has to be done
- First Driving Tests to come
- Path Planning Concept needs to be implemented

FH Aachen  
Faculty of Mechanical Engineering and Mechatronics  
Mechatronics Project: Autonomous Guided Vehicle  
Goethestr. 1  
52064 Aachen