

Object Hunt

Final Presentation

Manuel Audran

Tim Mennicken

Robert Rose

University of Applied Sciences Cologne

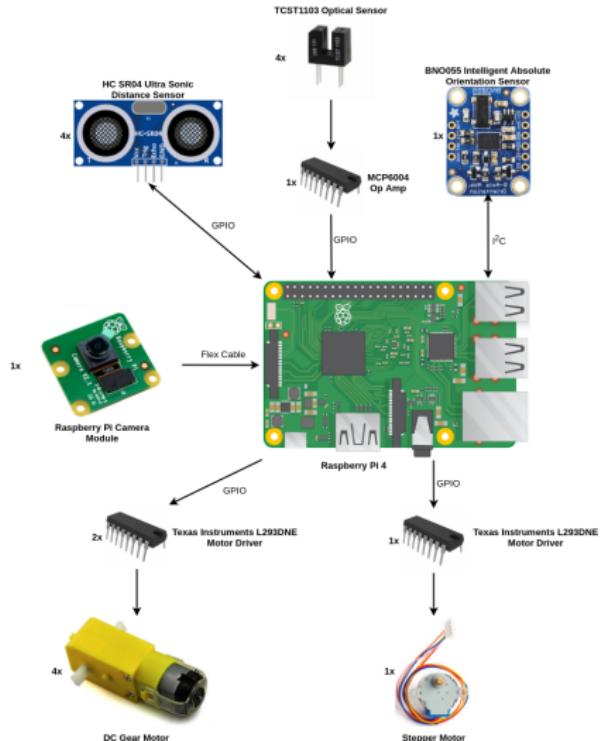
Wednesday the 29th of January, 2020

Table of Contents

- Hardware
 - PCB Board
 - Assembly
- Software
 - Inter-process communication
 - Internal processes
- Image processing
 - Software setup
 - Framework
 - Streaming
 - Model
- Mapping & Routing
- References

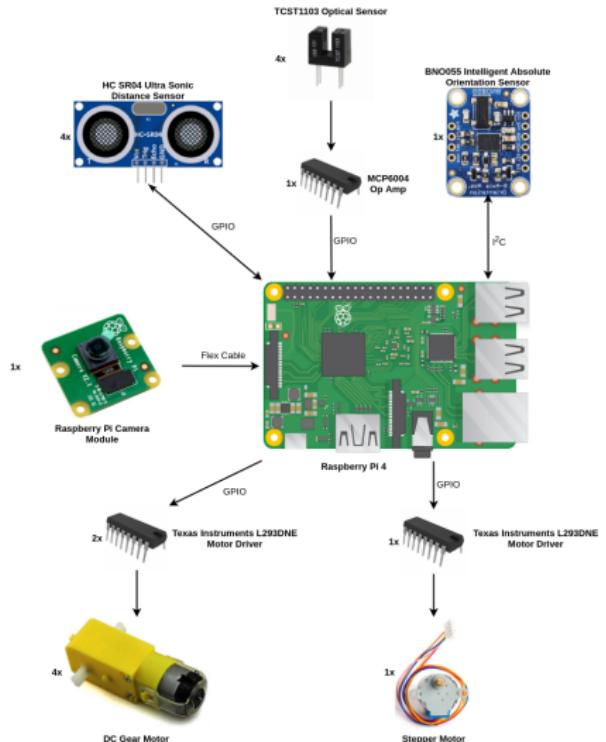
Hardware Overview

- ▶ Ultra sonic distance sensor
- ▶ Infrared revolution sensor
- ▶ Smart movement sensor
- ▶ Camera module

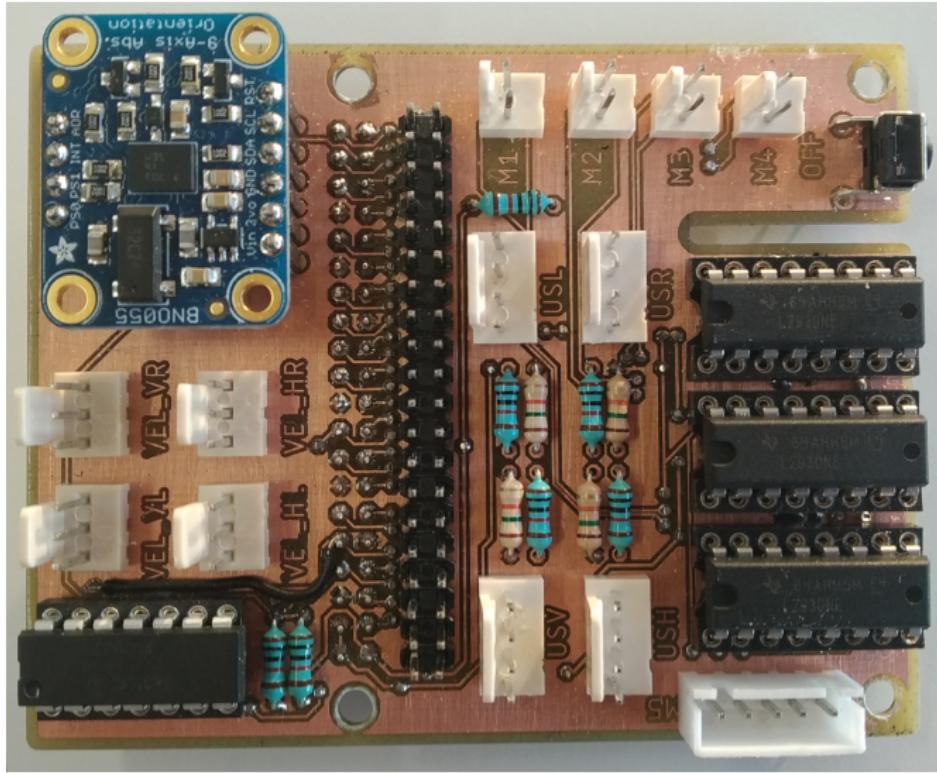


Hardware Overview

- ▶ Ultra sonic distance sensor
- ▶ Infrared revolution sensor
- ▶ Smart movement sensor
- ▶ Camera module
- ▶ DC gear motor
- ▶ Stepper motor



PCB Board

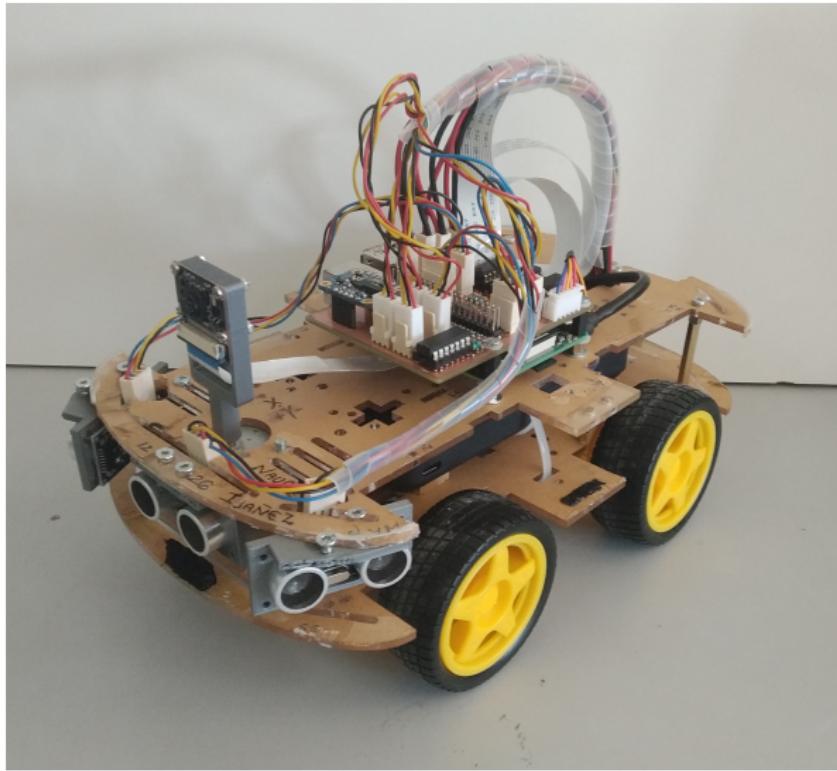


Object Hunt

Manuel Audran, Tim Mennicken, Robert Rose
29.01.2020

Technology
Arts Sciences
TH Köln

Assembly



Object Hunt

Manuel Audran, Tim Mennicken, Robert Rose
29.01.2020

Slide 4 of 27

Software Principles

- ▶ Communication
 - ▶ Via TCP/ IP
 - ▶ Independent from other devices
 - ▶ Beyond system border

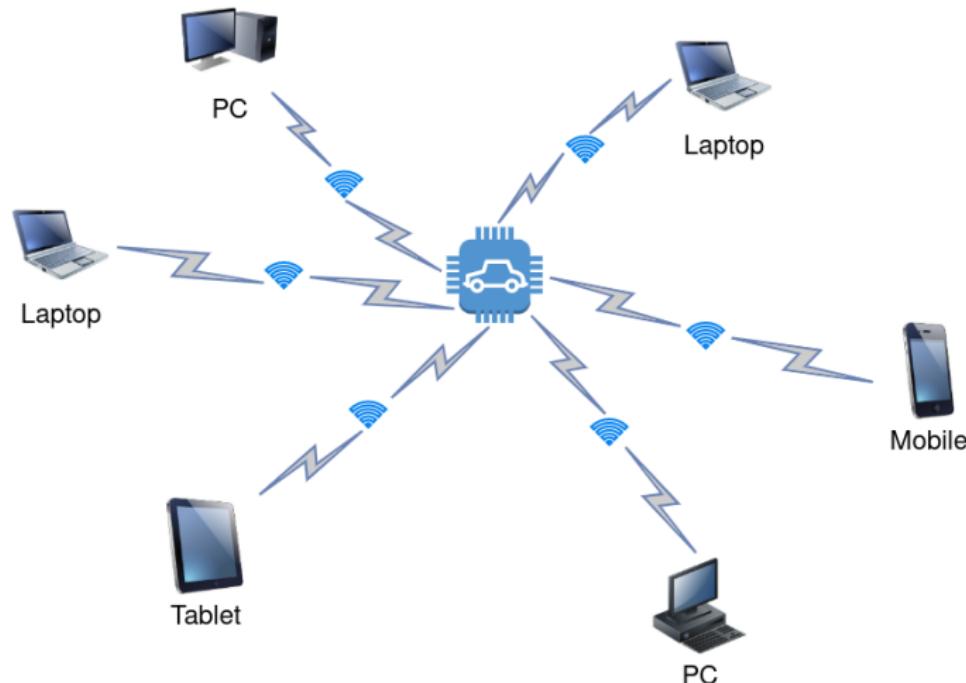
Software Principles

- ▶ Communication
 - ▶ Via TCP/ IP
 - ▶ Independent from other devices
 - ▶ Beyond system border
- ▶ Economical
 - ▶ No unnecessary CPU consume
 - ▶ Fast reaction times
 - ▶ Event driven

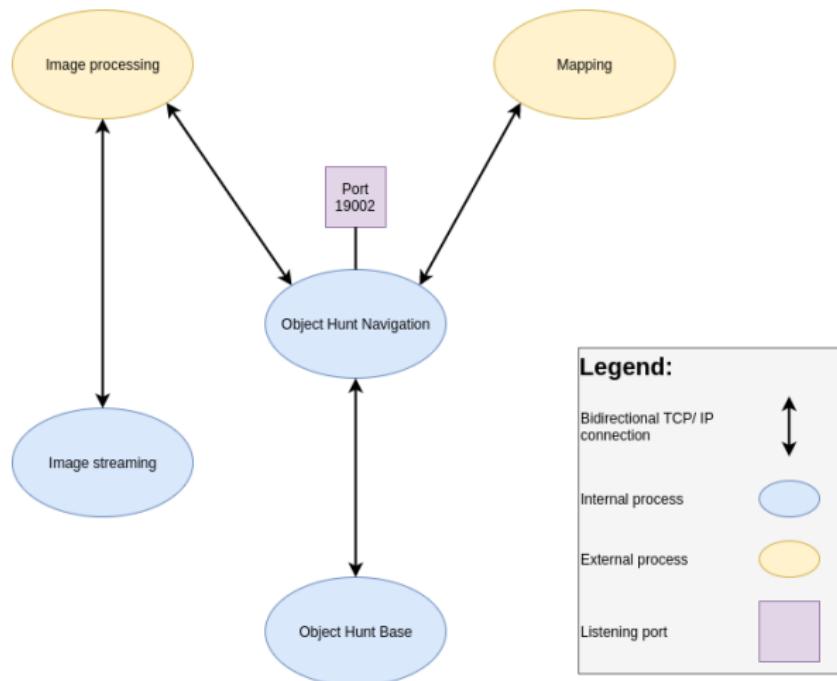
Software Principles

- ▶ Communication
 - ▶ Via TCP/ IP
 - ▶ Independent from other devices
 - ▶ Beyond system border
- ▶ Economical
 - ▶ No unnecessary CPU consume
 - ▶ Fast reaction times
 - ▶ Event driven
- ▶ Documentation
 - ▶ Complete
 - ▶ Easy accessible

Connection Overview



Connection Implementation

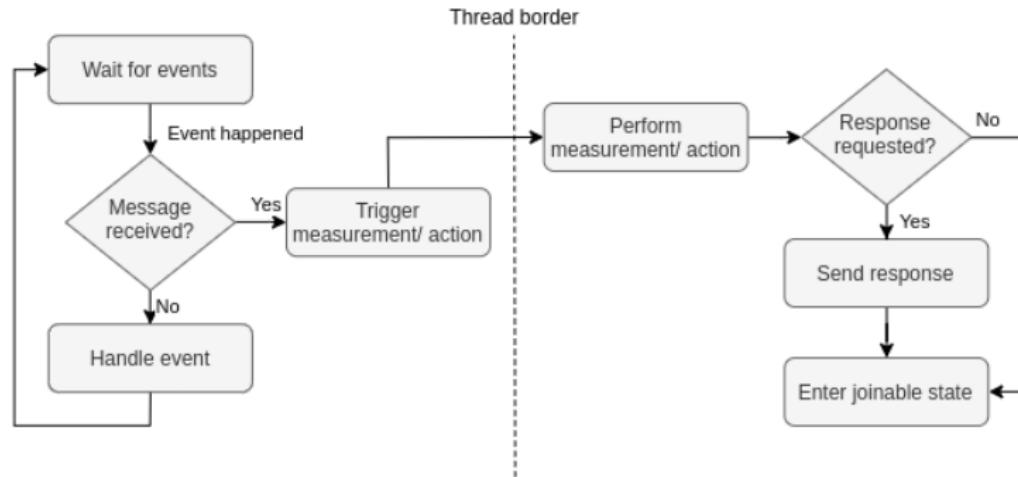


Base Process

- ▶ Exclusively accesses the hardware
- ▶ Receives commands and requests from the navigation process
- ▶ Continuous communication with the navigation process

Base Process

- ▶ Exclusively accesses the hardware
- ▶ Receives commands and requests from the navigation process
- ▶ Continuous communication with the navigation process

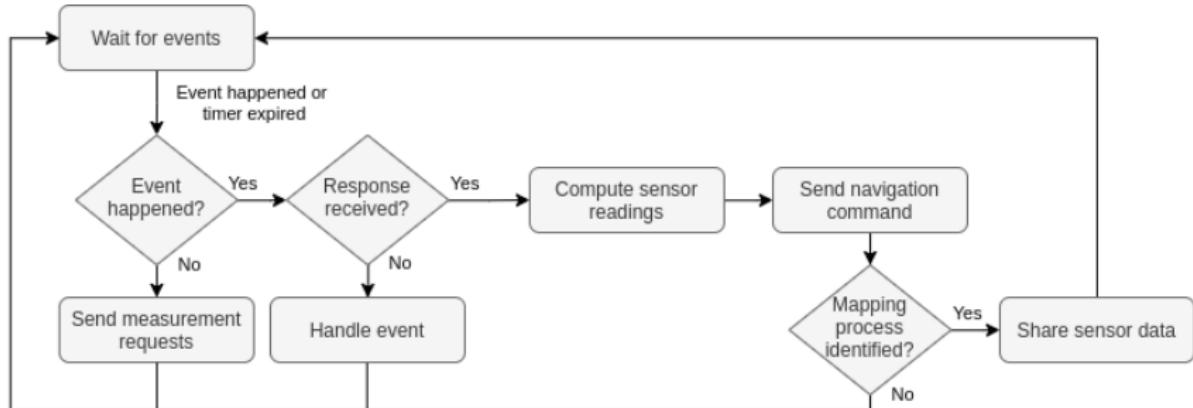


Navigation Process

- ▶ Requests periodically sensor readings
- ▶ Sends navigation commands
- ▶ Handles connections with further processes

Navigation Process

- ▶ Requests periodically sensor readings
- ▶ Sends navigation commands
- ▶ Handles connections with further processes

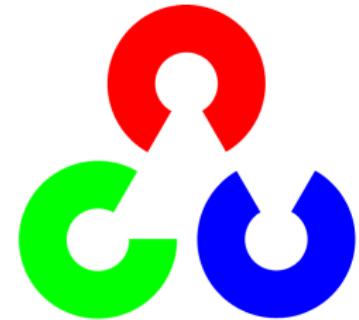


Software Setup

- ▶ Python 3.7.3
- ▶ OpenCV 4.1.2
- ▶ ImageZMQ (Python wrapper for ZMQ functionalities)
- ▶ MobileNet v2 SSD
- ▶ COCO dataset

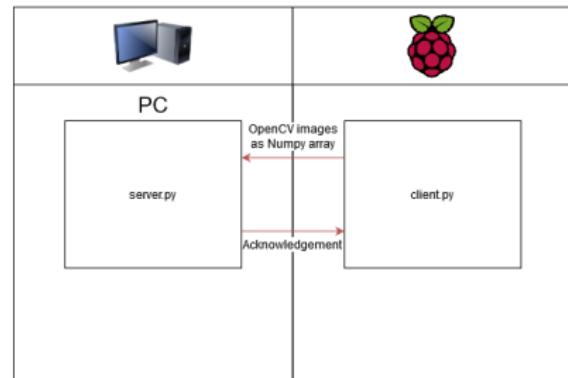
Framework - Open Source Computer Vision Library

- ▶ Platform independent library written in C++
- ▶ Available as Python pip package
- ▶ DNN module to utilize pretrained Tensorflow models
- ▶ Offers utilization methods to present the images



Streaming

- ▶ First analysis shows Raspberry Pi is not capable of real-time object detection
- ▶ Solution ⇒ stream the video and process it on the PC
- ▶ ImageZMQ is based on ZeroMQ, made for transporting OpenCV images over network
- ▶ Results in two processes, connected via sockets



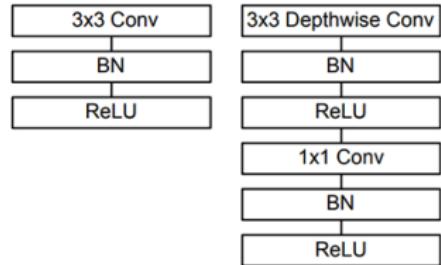
Requirements:

- ▶ Object detection (tennis ball as defined object)
- ▶ Real-time processing
- ▶ Sufficient accuracy

⇒ MobileNet in combination with SSD offers an efficient solution for embedded systems with real-time object detection
⇒ COCO dataset includes 90 different classes

Model - Mobilnet Architecture

- ▶ Main difference: Depthwise separable convolution
- ▶ Splits up convolution in two parts
 - ▶ 3x3 depthwise Conv
 - ▶ 1x1 pointwise Conv
- ▶ Reduces the number of trainable parameters remarkably
- ▶ Trade-off is less accuracy



Model – Single Shot Detection

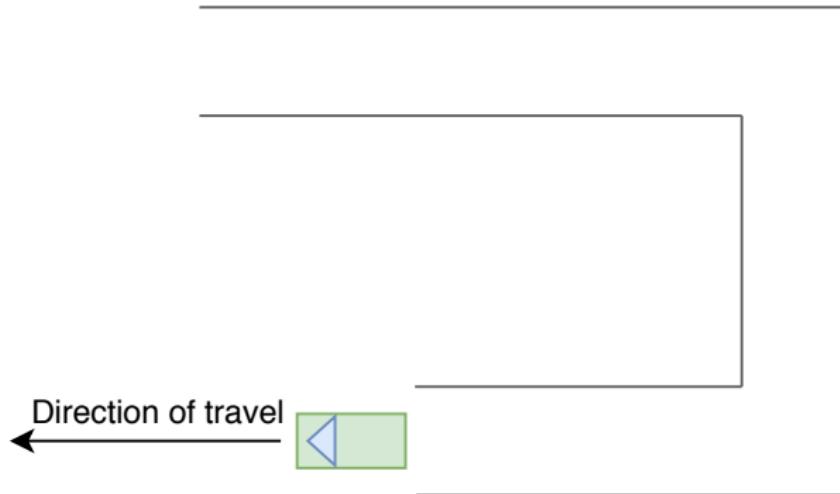
- ▶ Defines default boxes
- ▶ Generates scores for each category and default box
- ▶ Adjusts the box to match the object shape
- ▶ Three commonly used object detection methods:
 - ▶ Faster R-CNN
 - ▶ YOLO
 - ▶ SSD



Method	mAP	FPS	batch size	# Boxes	Input resolution
Faster R-CNN (VGG16)	73.2	7	1	~ 6000	~ 1000 × 600
Fast YOLO	52.7	155	1	98	448 × 448
YOLO (VGG16)	66.4	21	1	98	448 × 448
SSD300	74.3	46	1	8732	300 × 300
SSD512	76.8	19	1	24564	512 × 512
SSD300	74.3	59	8	8732	300 × 300
SSD512	76.8	22	8	24564	512 × 512

Mapping & Routing

Example: Tunnel

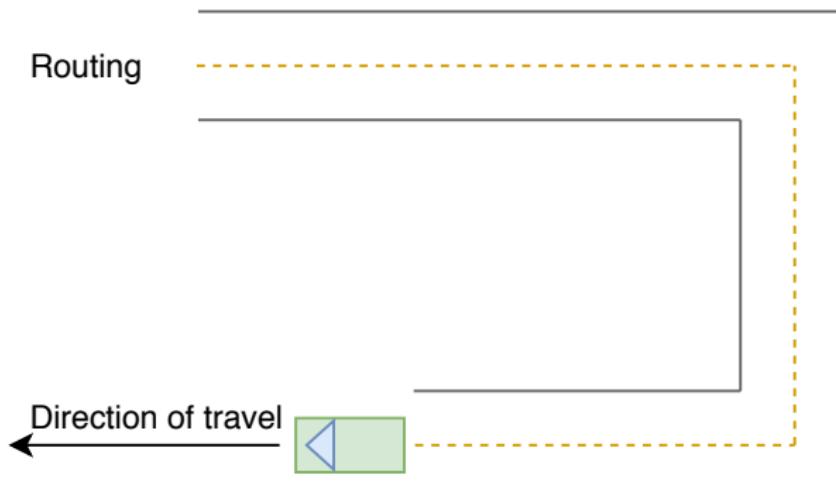


Object Hunt

Manuel Audran, Tim Mennicken, Robert Rose
29.01.2020

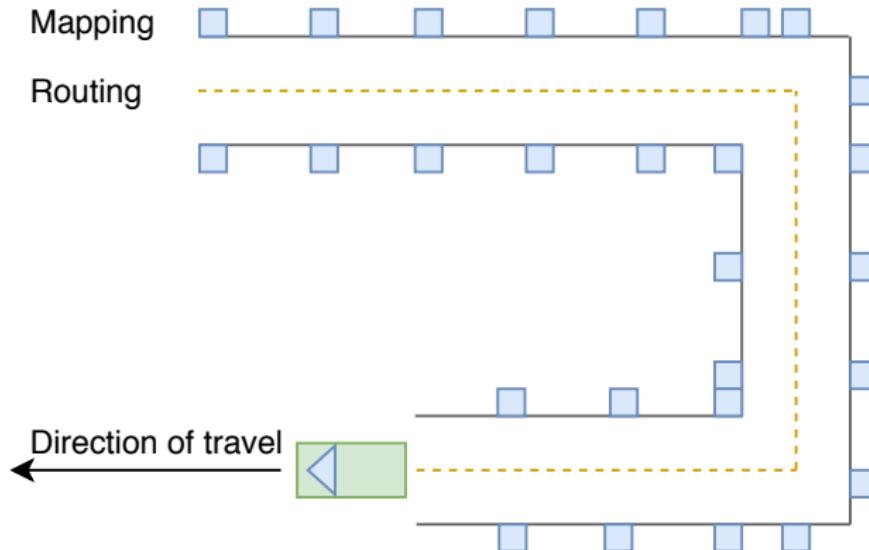
Mapping & Routing

Example: Tunnel

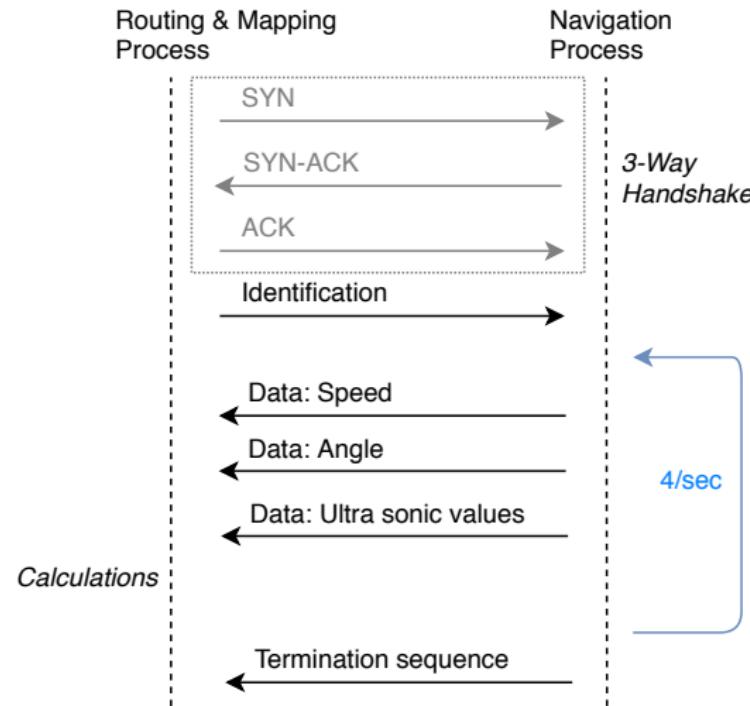


Mapping & Routing

Example: Tunnel



Communitaion



Routing

- ▶ Revolution Sensor
- ▶ Smart movement sensor
- ▶ Start point [0,0] (x,y)
- ▶ Logged position coordinates
 - ▶ Calculated with sensor data

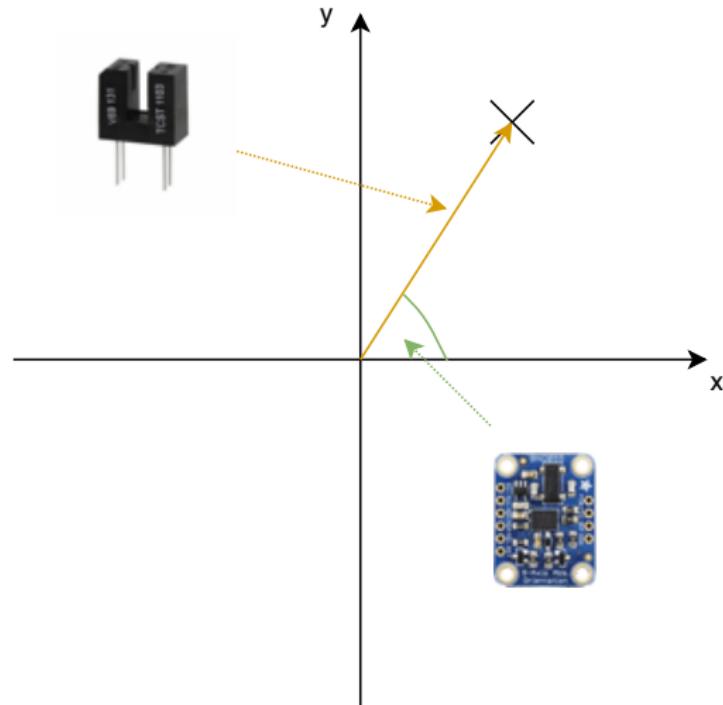


Direction of travel
Angle changes

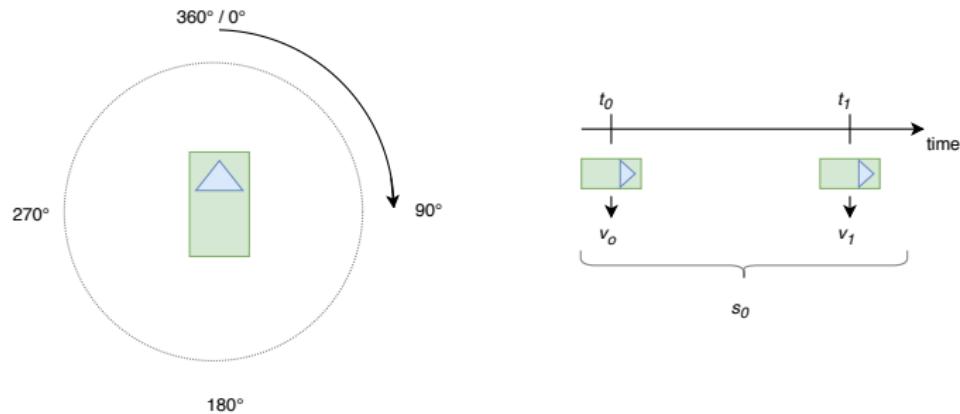


Travel speed

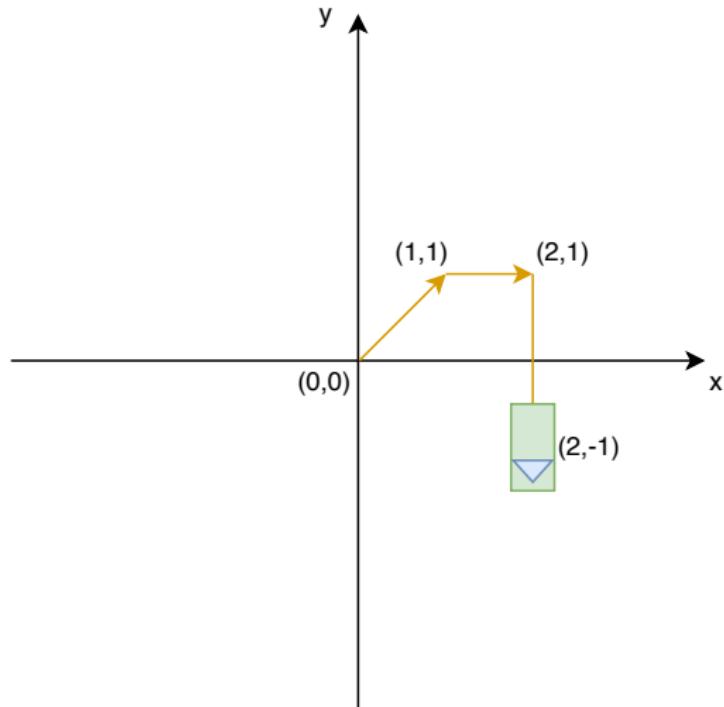
Routing



Routing



Routing



Mapping

- ▶ Ultra sonic sensor
- ▶ Smart movement sensor
- ▶ Related to actual vehicle position and direction of travel
- ▶ Logged coordinates of positions of objects
 - ▶ Calculated with sensor data

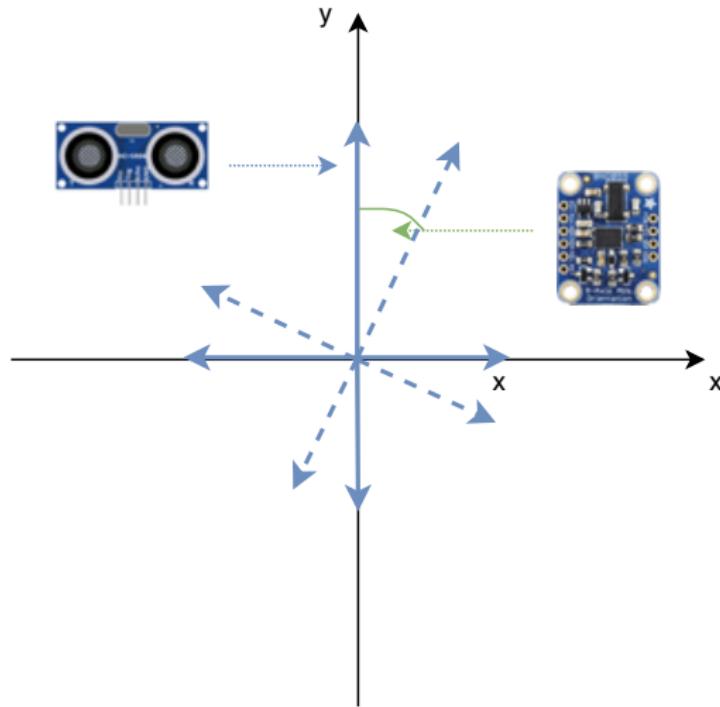


Direction of travel
Angle changes

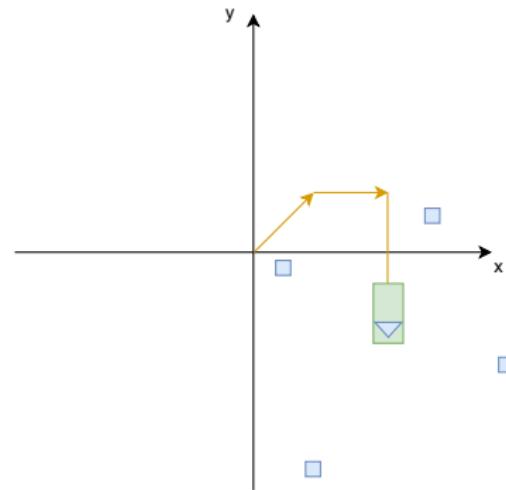
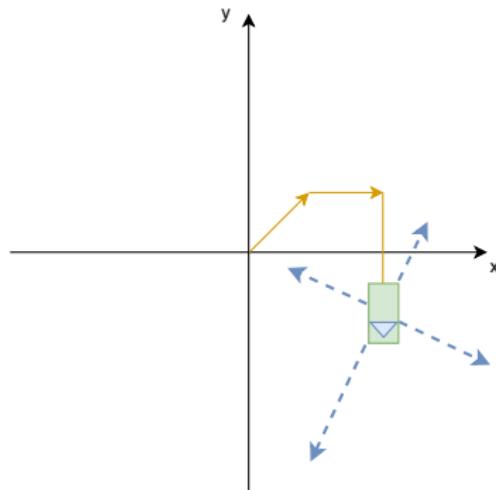


Distance to object

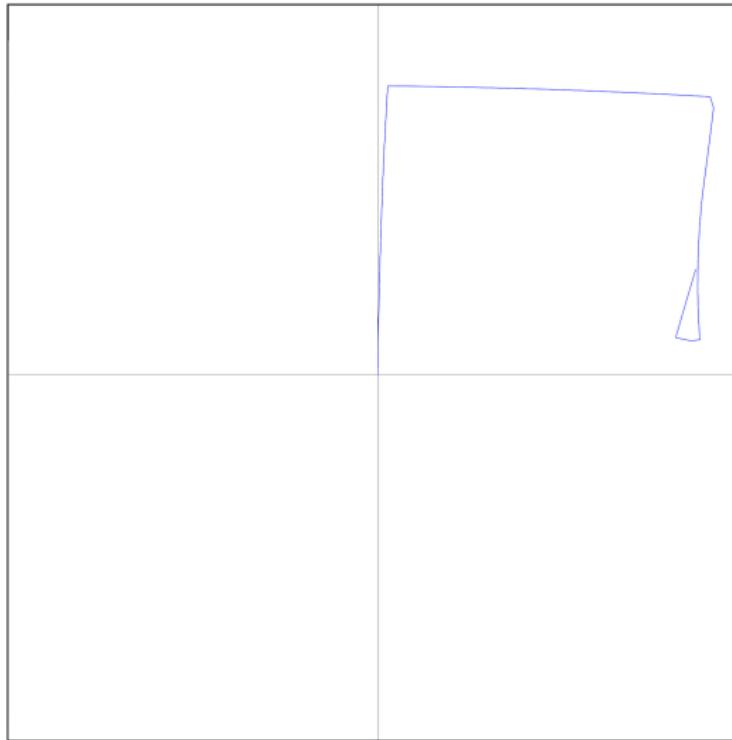
Mapping



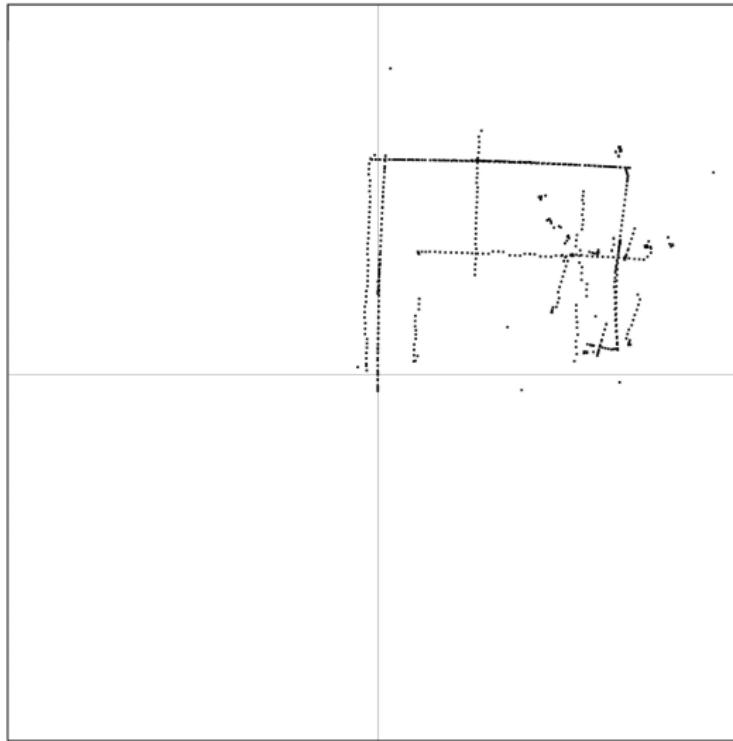
Mapping



Example



Example



Object Hunt

Manuel Audran, Tim Mennicken, Robert Rose
29.01.2020

References

- ▶ ImageZMQ - <https://github.com/jeffbass/imagezmq>
- ▶ Image Processing setup -
<https://www.pyimagesearch.com/2019/04/15/live-video-streaming-over-network-with-opencv-and-imagezmq/>
- ▶ Tensorflow Modelzoo -
https://github.com/tensorflow/models/blob/master/research/object_detection/g3doc/detection_model_zoo.md
- ▶ SSD - <https://arxiv.org/pdf/1512.02325.pdf>
- ▶ MobileNet - <https://arxiv.org/pdf/1704.04861.pdf>
- ▶ OpenCV - <https://github.com/opencv/opencv/wiki>

For further information please see the *documentation*