Hackaton II: A Brief

Anoop M. Namboodiri
IIIT Hyderabad

Topics

1. Autonomous Navigation
   - The Problem
   - Sensors for ANS
   - Key Problems in ANS
2. Hackathon – II
   - The Challenge
   - Preparatory Work
   - Tasks for the Hackathon
   - Evaluation Criteria

The Problem of Autonomous Navigation

Given a target location, find the precise sequence of action you need to take to move from your current location to the target location

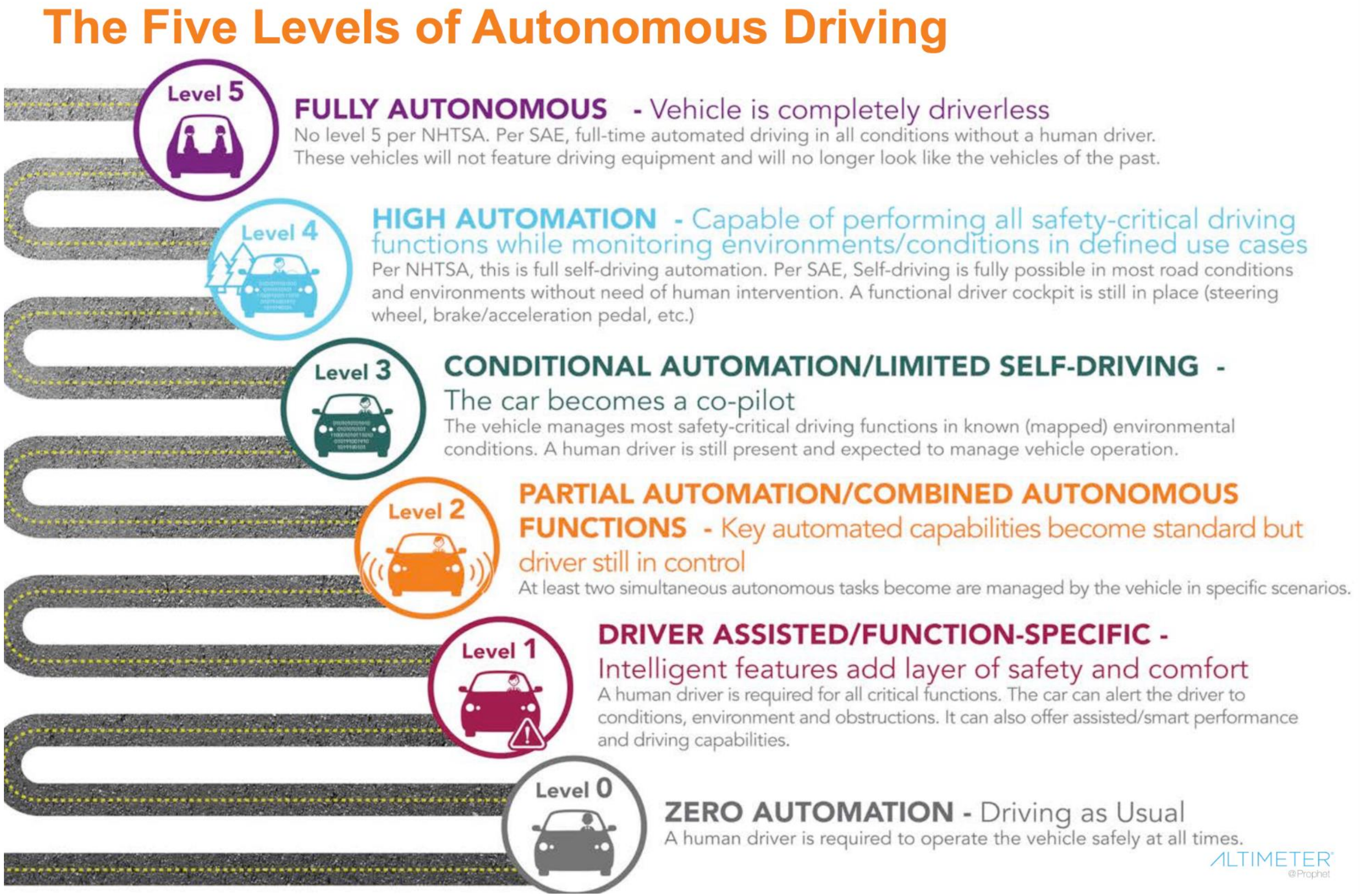
- By the optimal path: Time, Distance, Cost, Comfort
- Under uncertain conditions: Map, Location
- With other active participants on the way
- Within the capabilities of your vehicle

- A Multi-objective optimization problem, under uncertainty

Levels of Autonomy

Depending on:

- Environmental Conditions
- Amount of human intervention



Definition vary slightly:
NHTSA* vs SAE†

* National Highway Traffic Safety Administration

† Society of Automotive Engineers

Automation: A Historical Perspective

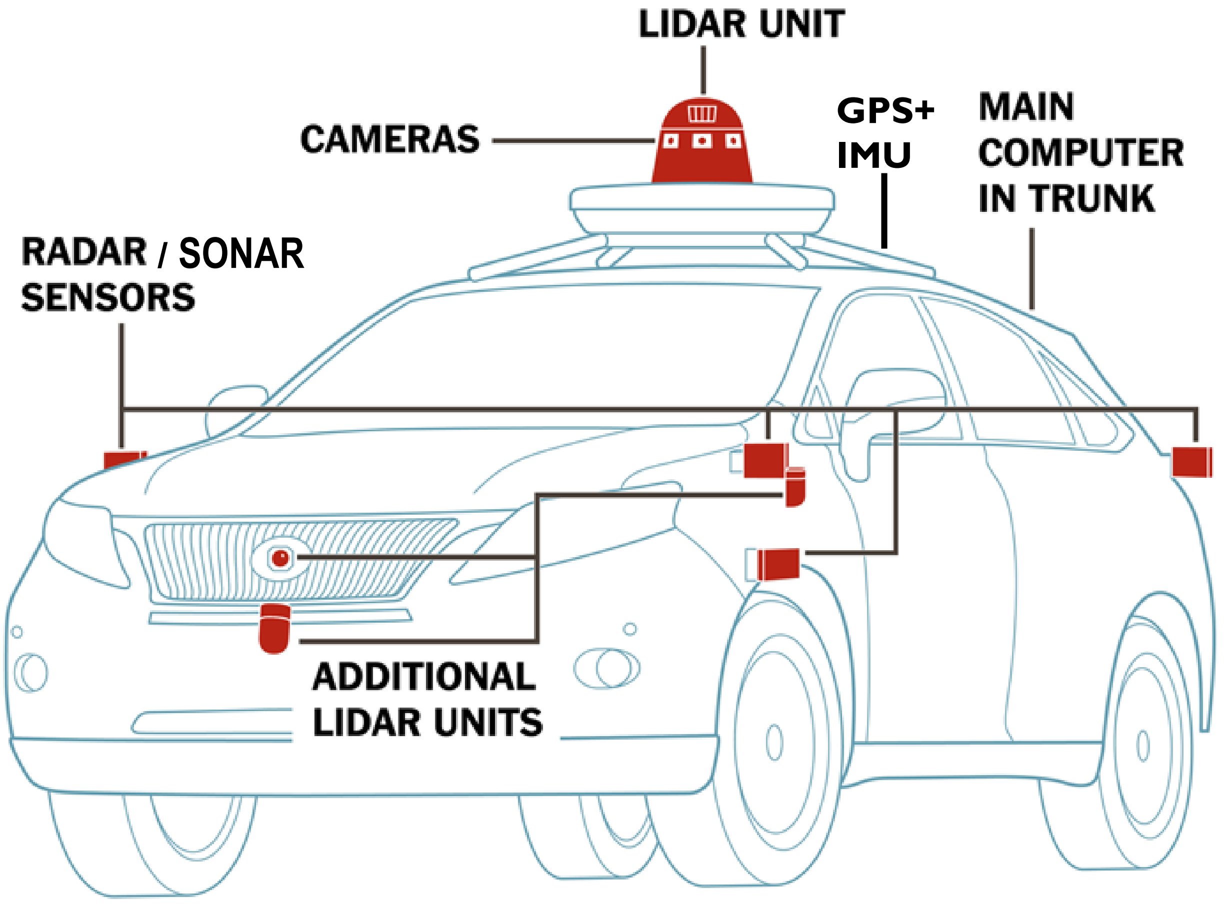
- 1948 Oldsmobile: Hydra-Matic transmission
- 1958 Imperial: Cruise Control
- 1992-95 Mitsubishi: Lidar based Distance Control (L1)
- 2001 Nissan: Lane-Keeping Support
- 2013 Mercedes: Distance with Steering Assist (stereo)
- 2014 Tesla Model S: Advanced Lane Assistance
  - Speed limit recognition. Level-2 Automation
- 2018 Audi A8: Level 3 autonomy (in slow traffic)
  - Up to 60kmph

Topics

1. Autonomous Navigation
    - The Problem
    - Sensors for ANS
    - Key Problems in ANS
2. Hackathon – II
    - The Challenge
    - Preparatory Work
    - Tasks for the Hackathon
    - Evaluation Criteria

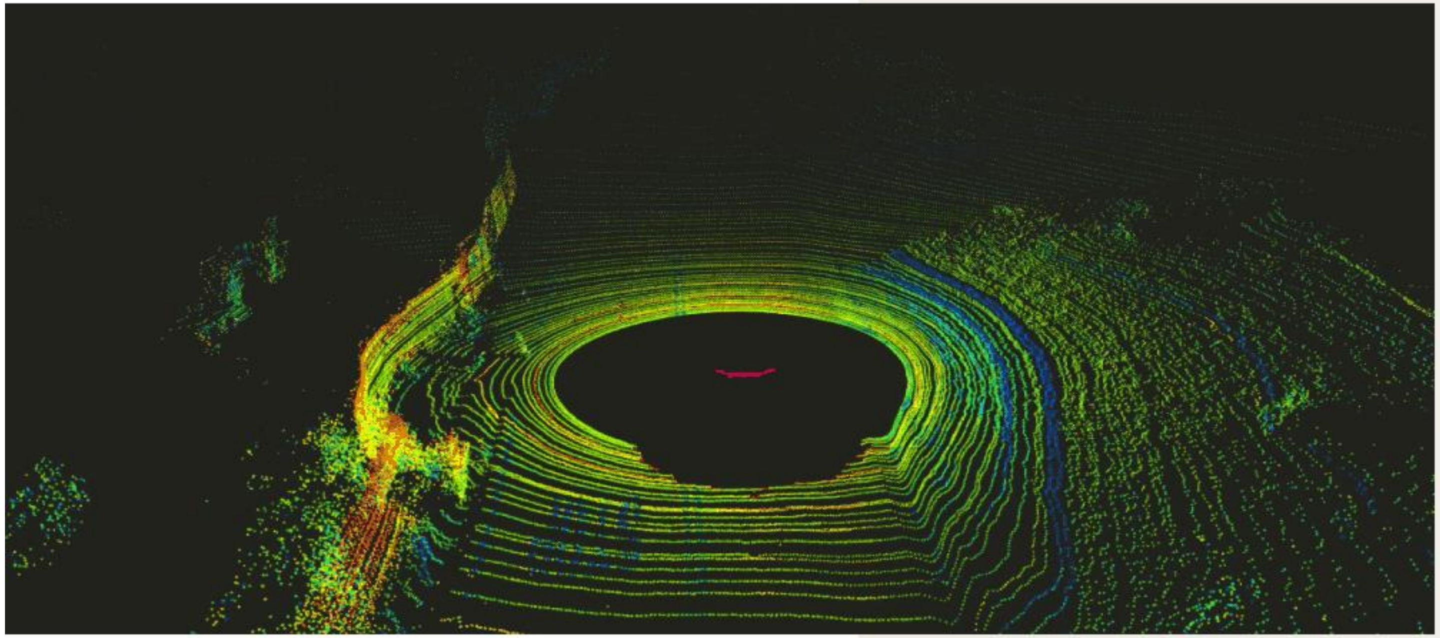
Sensors for Autonomous Driving

- Sensing Local Environment
  - LIDAR
  - Cameras
  - Radar
  - Sonar
- Global Position
  - GNSS
  - Compass
- IMU

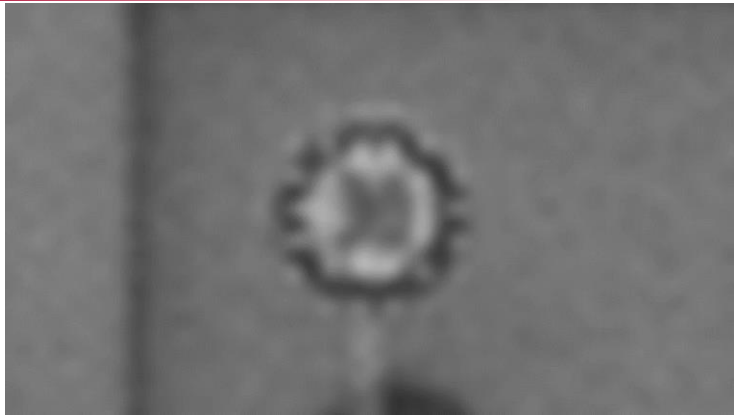


LIDAR

- Laser ToF Sensor: Can give 2cm accuracy
- Capture: Spinning emitter-receiver pairs
- E.g.: Velodyne, Quanergy, Luminar (solid state)
- Laser power and safety
  - 900nm (40m range limited due to power)
  - 1550nm (200m range)



Camera



- General purpose sensor
  - Object detection
  - Distance
  - Traffic sign/signal
  - Road condition
  - Adverse weather scenarios
- Resolution
  - 50 pixels at max range
- Focal Length vs. field of view
- Stereo configurations
- Data Rate
- Shutter: Global vs. Rolling

How do they compare?

- Single Camera
  - Simplest generic sensor
  - Academic Interest
  - Mobileye, NEC
- Multi-camera (calibrated)
  - Better data quality
  - More processing, calibration, synchronization, data rate, MTBF
  - Tesla, Mercedes
- LiDAR
  - High quality depth data
  - Reliable in adverse weather
  - Cost
  - (Waymo, Ford, Uber)

Radar, Sonar

- Primarily 1D sensors
- Radar
  - Very reliable in adverse weather
  - Works in long ranges
- Sonar
  - Inexpensive
  - Primarily at short ranges

GNSS, IMU, Vehicle Info.

- GNSS (GPS)
  - Can do most of driving (DARPA challenge)
  - Can work even in zero visibility
  - Satellite signal obstruction
- Inertial Meas. Unit + Distance Meas. Instrument
  - Complements GPS
  - Accelerometer + Gyroscope
  - Drift in integration

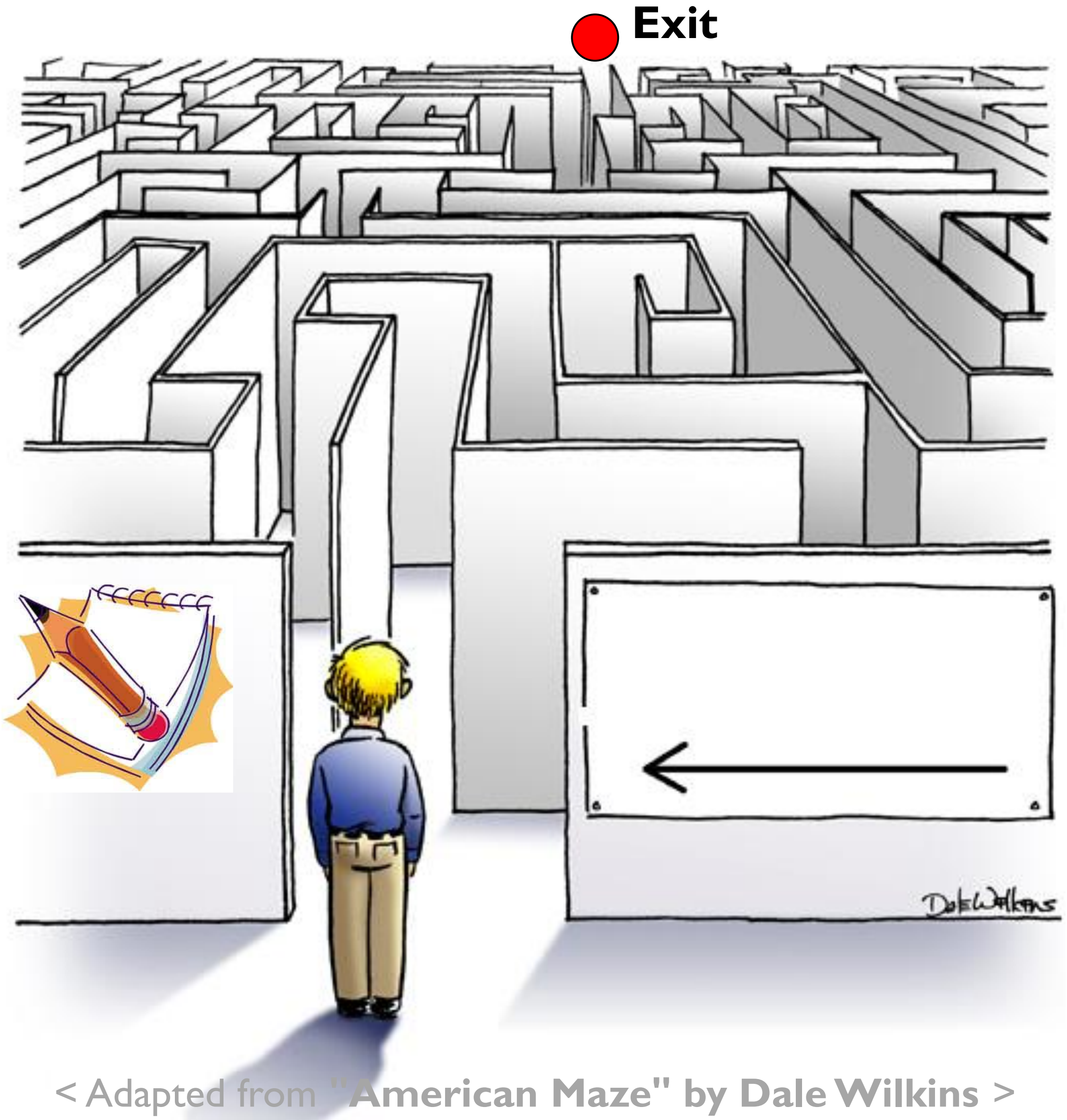
Topics

1. Autonomous Navigation
   - The Problem
   - Sensors for ANS
   - Key Problems in ANS

2. Hackathon – II
   - The Challenge
   - Preparatory Work
   - Tasks for the Hackathon
   - Evaluation Criteria

P1: Localization and Mapping

- Mapping
  - Create a 2D/3D map of the world
  - Often an offline process
- Localization
  - Your location within the map
  - Approximate / Precise
- Exploratory Settings
  - What if we do not know the map?



< Adapted from "American Maze" by Dale Wilkins >

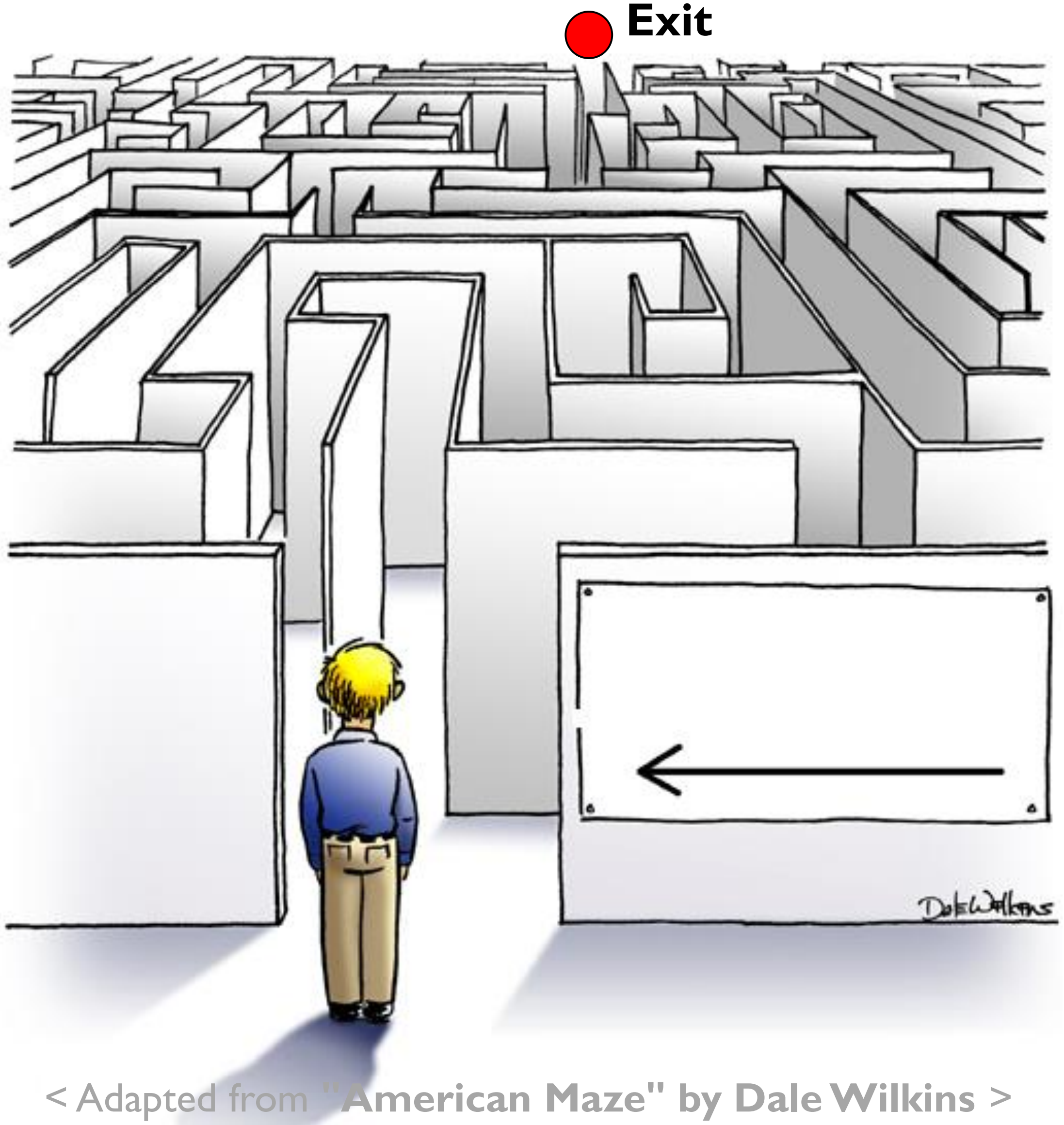
The SLAM Problem

SLAM is the process by which a robot **builds a map** of the environment and, at the same time, uses this map to **compute its location**

- **Localization:** inferring location given a map
- **Mapping:** inferring a map given a location
- **SLAM:** learning a map and locating the robot simultaneously

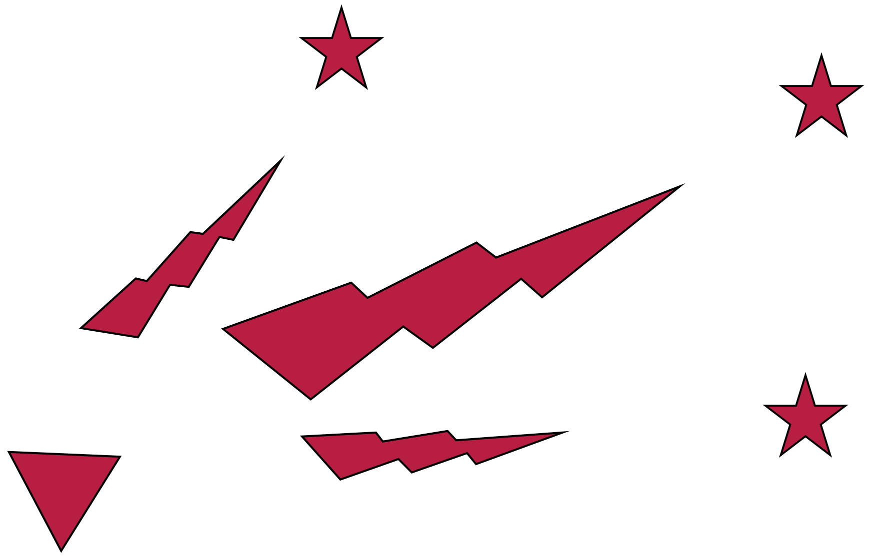
The SLAM Problem

- The problem has 2 stages: Mapping and Localization
- The paradox:
  - To build a map, we must know our position
  - To determine our position, we need a map!
- SLAM is like the chicken-egg problem
- Solution is to alternate between the two steps.

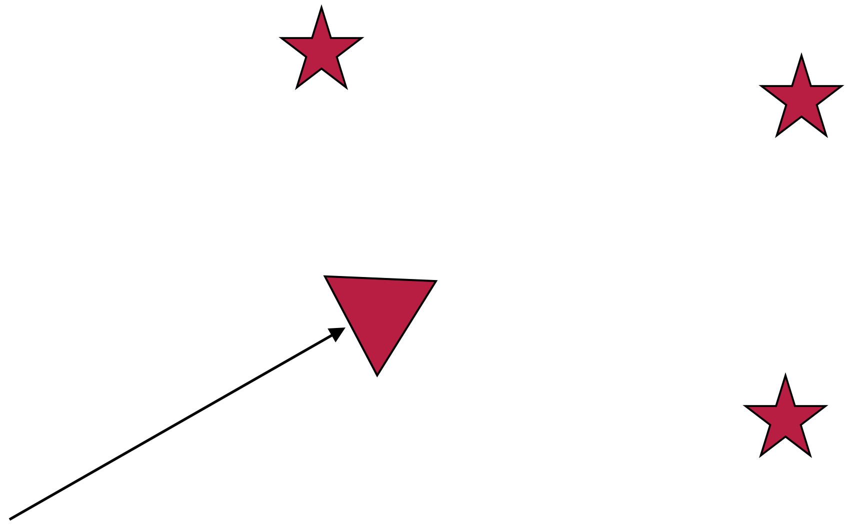


< Adapted from "American Maze" by Dale Wilkins >

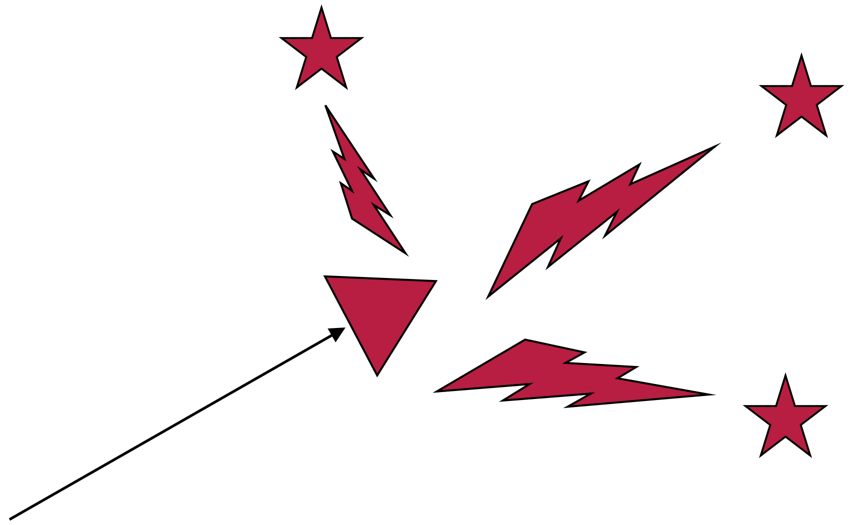
Sense the World, Localize and Map



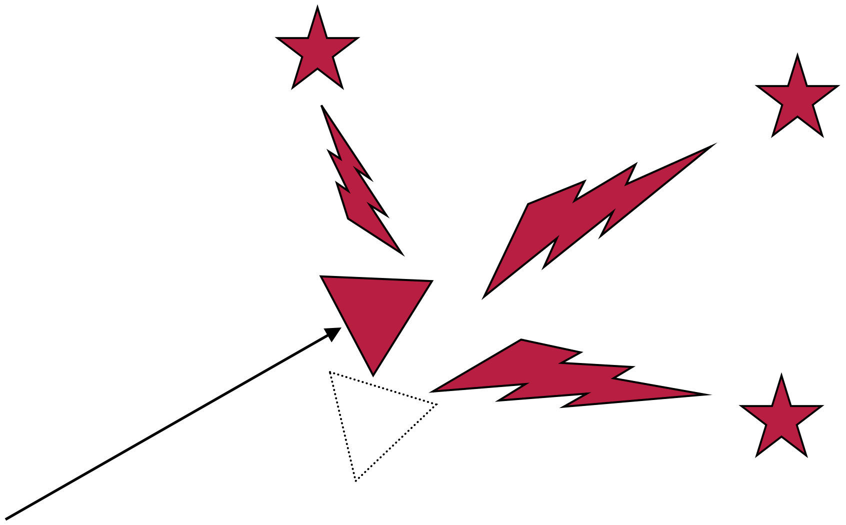
Move: Get Odometry



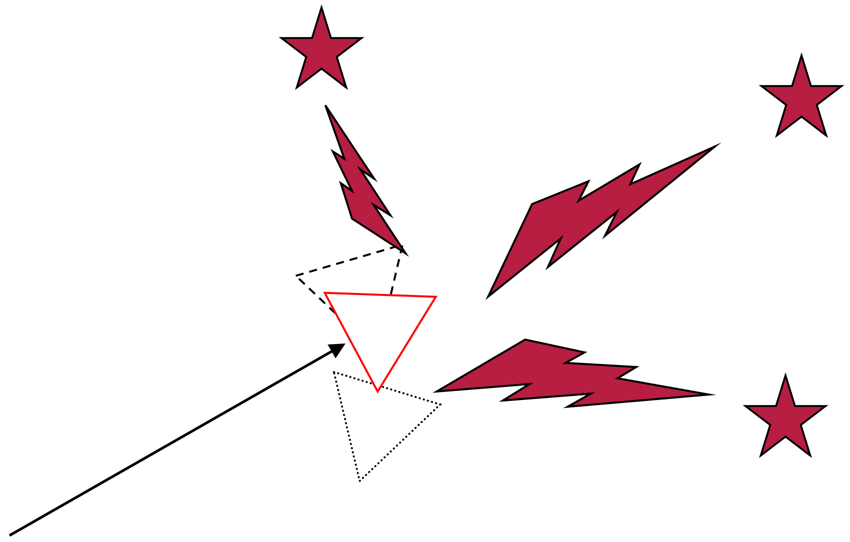
Sense the world again



Localize from Previous Map



Update Location and Map

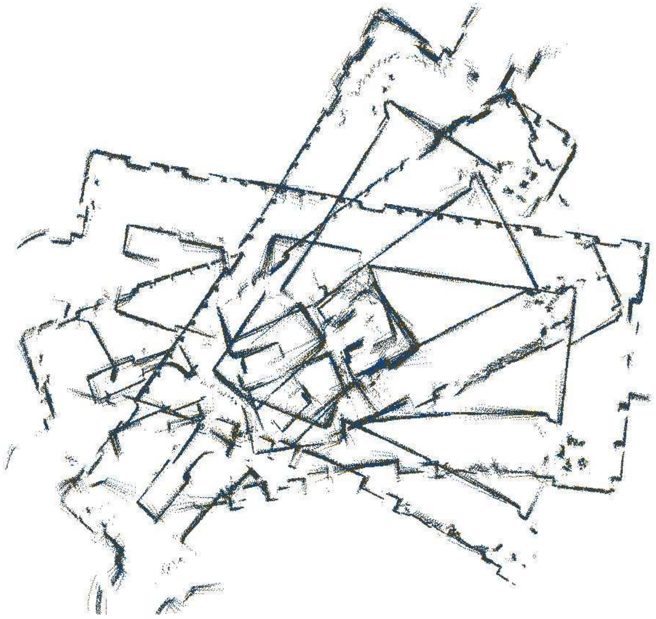


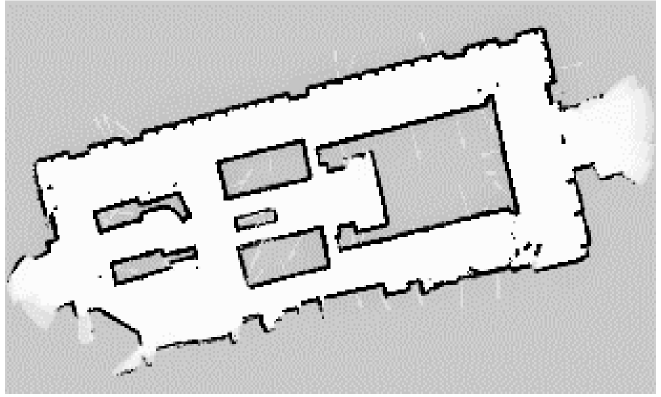
Simultaneous Localization and Mapping

- Use Tracking to have an estimate of your position. Can use odometry data.
- Use the pose estimate to update the map
- Use the updated map to refine your position
- Can be in two modes:
  - Online SLAM
  - Full SLAM
- Multiple approaches:
  - Extended Kalman Filter
  - Particle Filter (FastSLAM)
  - Graph based

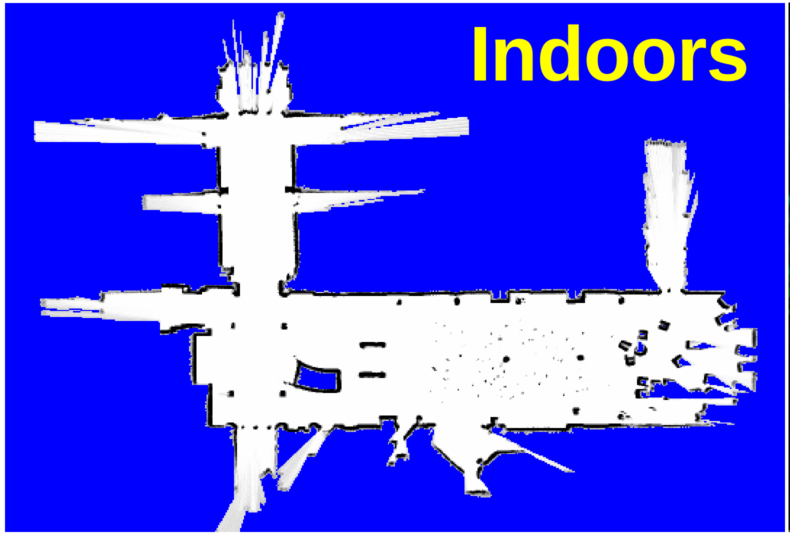
Issues in SLAM

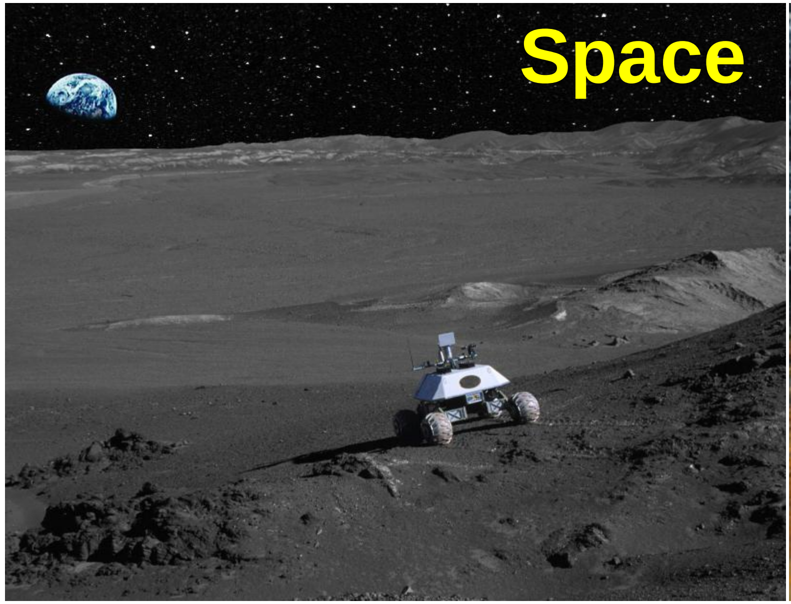
- Reliable Landmarks (RANSAC, Spikes)
- Data Association: Registering two scans
  - Landmarks vary between scans
  - Incorrect association
- Synchronizing odometry and scan
- Moving objects (landmarks)
- Loop Closure





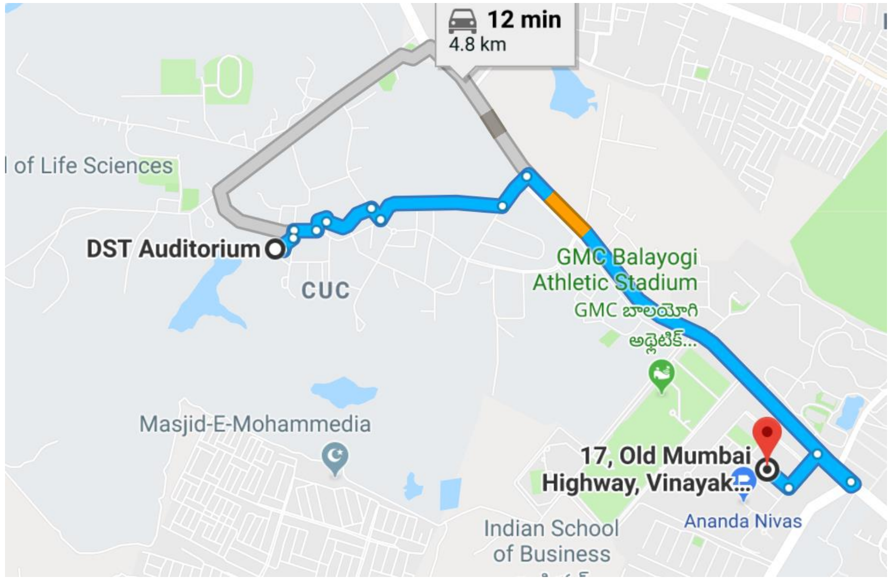
SLAM Applications





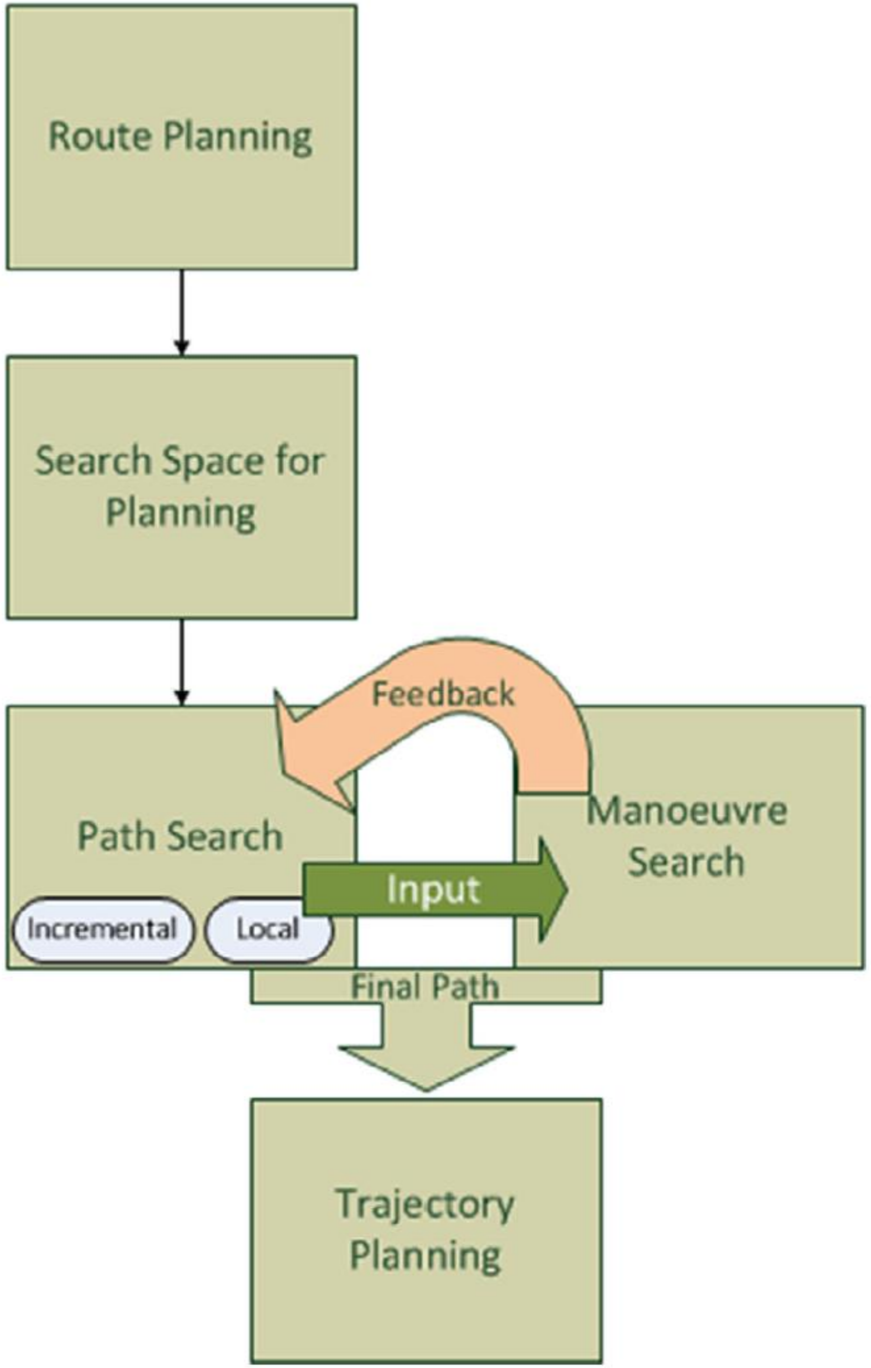
P2: Planning

- Want to go from DST Auditorium to Himalaya 105 (assume we know the map, roads and conditions)
- Divide the problem into smaller instances (nodes)
  - Route Planning
- Each node is then expanded to plan further details
  - Trajectory Planning (Path + Manoeuvre)
- Can use any path finding algorithm: Graph Search
- Need to update the planned paths based on contingencies



Planning

- Multi-resolution planning
- May use different approaches for each stage
- Need different inputs for each stage



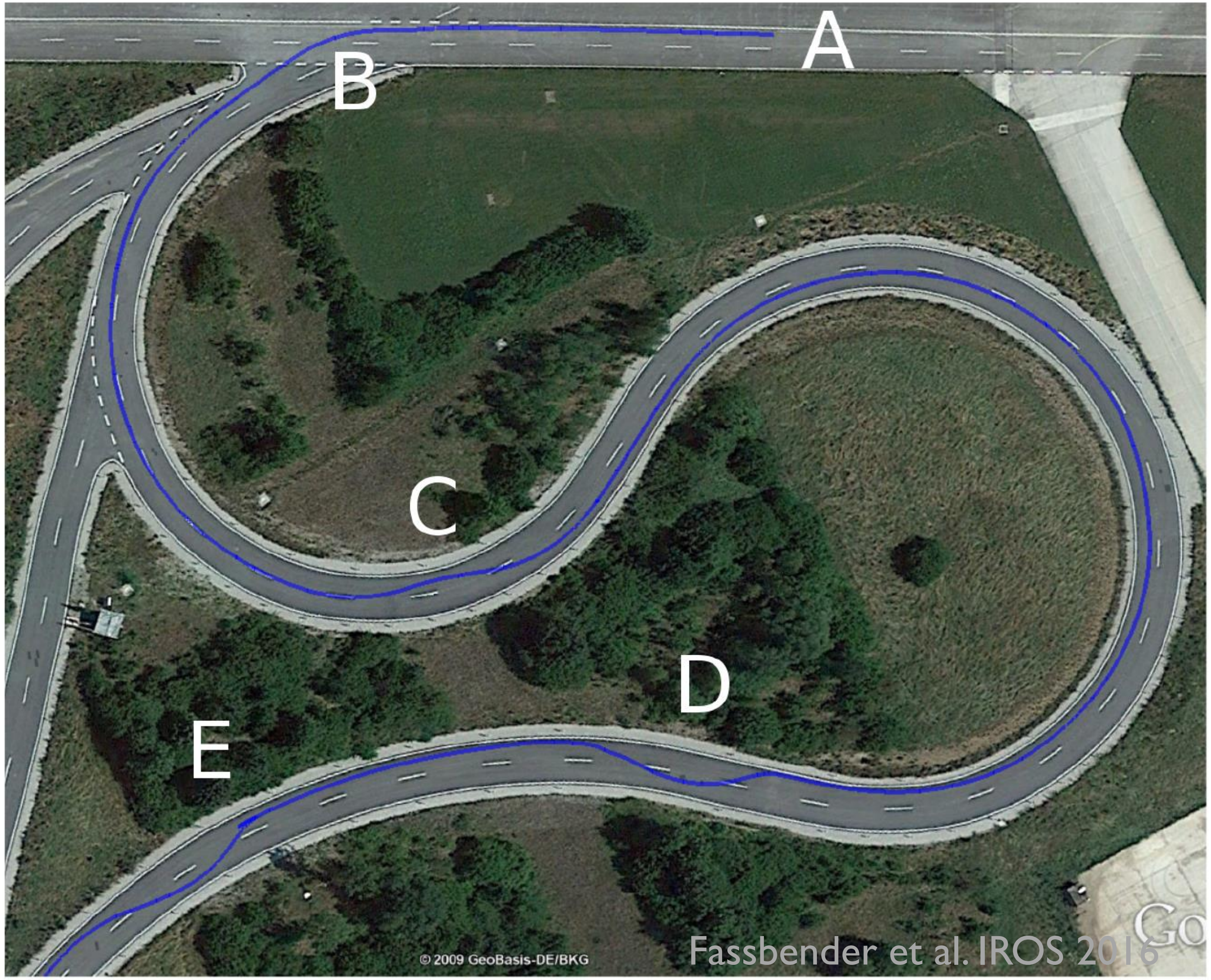
Why is Planning Challenging

- Unexpected behaviour of participants
- Ambiguity in position estimation
- Lack of features
- Changes in map (road closures, delays)
- Want the path to be smooth
- The grid cells are dynamic
- Need to plan for effective, efficient, smooth and safe driving
- Planning for a set of vehicles

Planning vs Execution

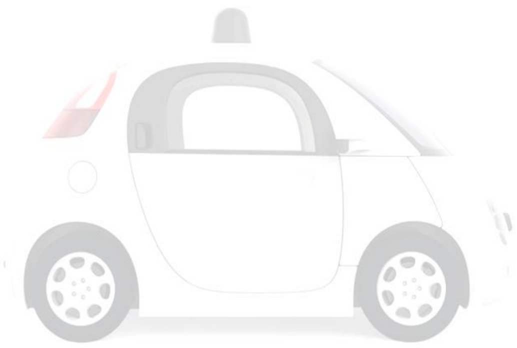
- The final path that you take may vary significantly in detail compared to what you planned

- The devil is in the details (and so is the use of AI)



Questions?

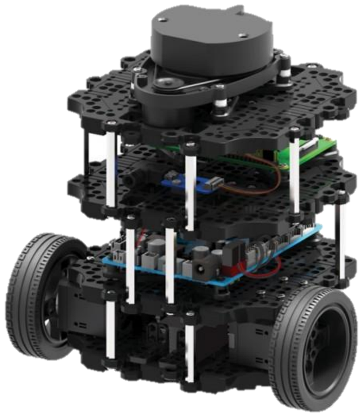
Topics

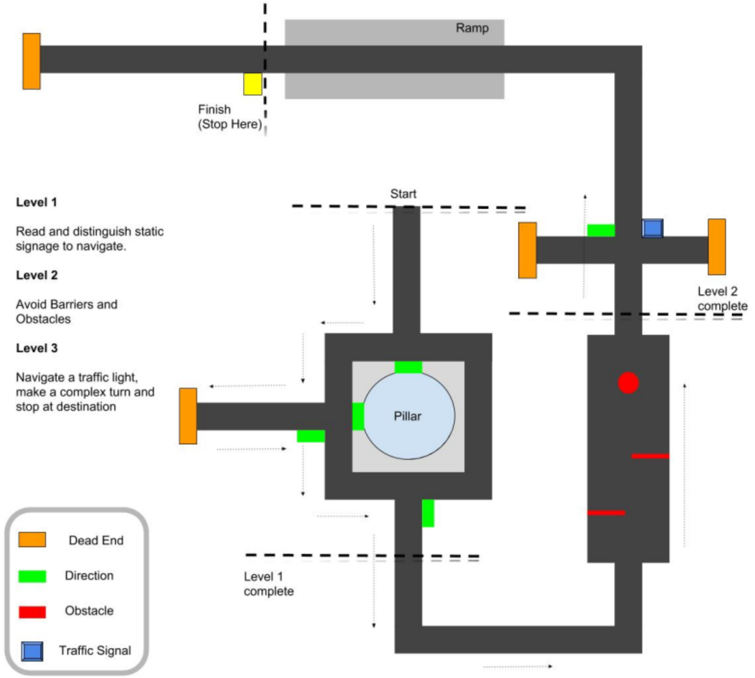


1. Autonomous Navigation
   - The Problem
   - Sensors for ANS
   - Key Problems in ANS
2. Hackathon – II
   - The Challenge
   - Preparatory Work
   - Tasks for the Hackathon
   - Evaluation Criteria

The Hackathon Challenge

- Your goal is to develop:
  - A fully autonomous vehicle (a robot)
  - that will traverse a specific path,
  - avoiding obstacles,
  - in minimal time,
  - using Camera and LiDAR inputs.
- Three increasing levels of difficulty
- Bonus tasks



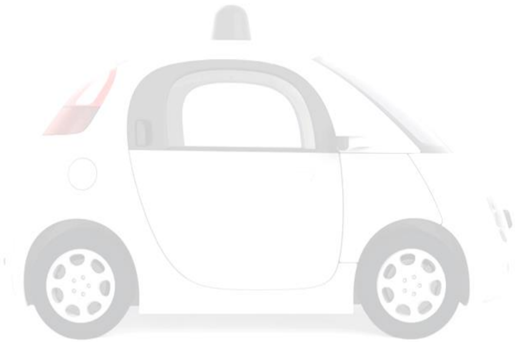


Preparatory Work

AIML

- Install Ubuntu 16.04
- Install ROS and Gazebo
- Develop a traffic sign recognizer
- Develop an obstacle sensor
- Integrate the above and run through and obstacle course

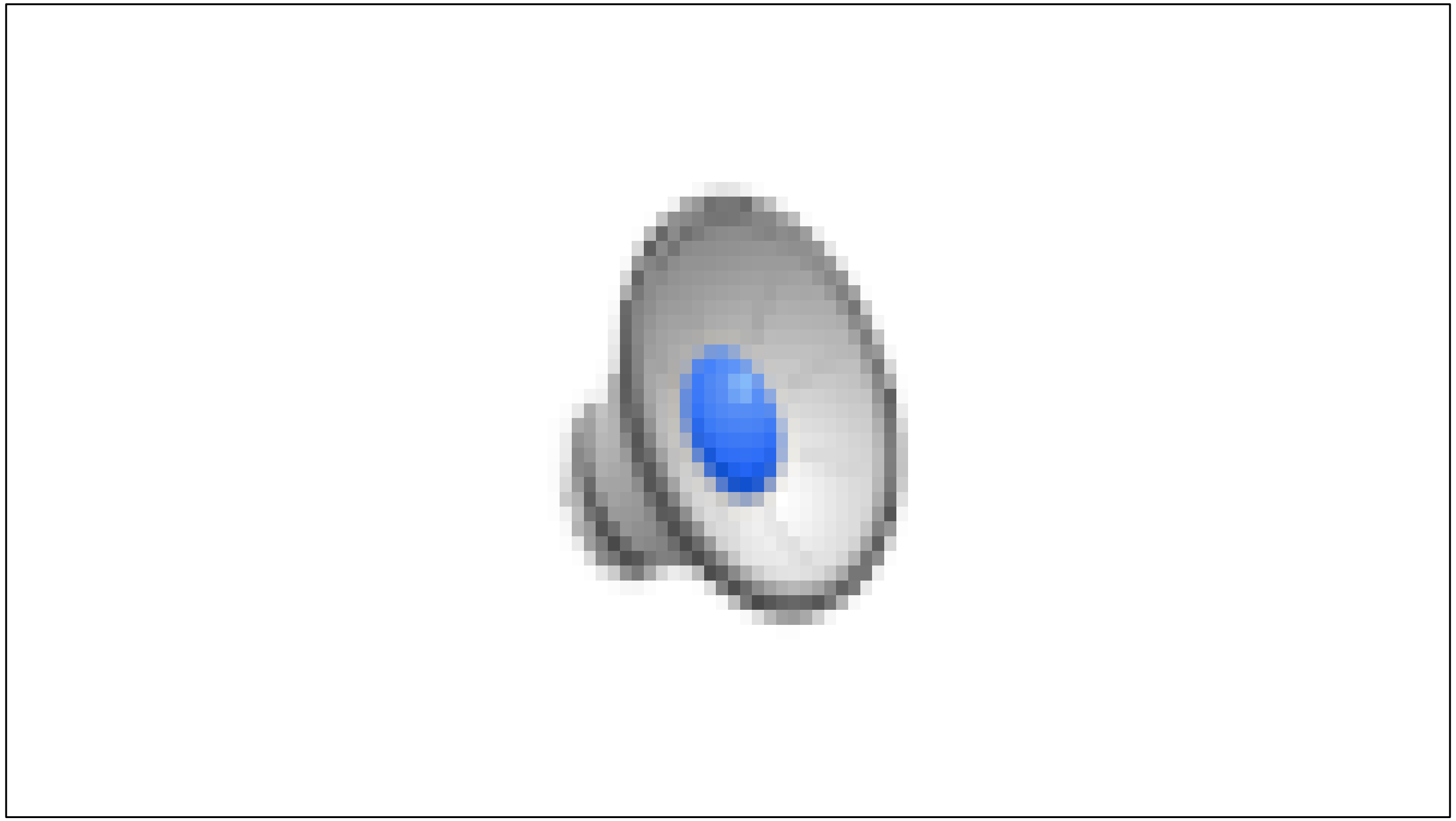
Topics



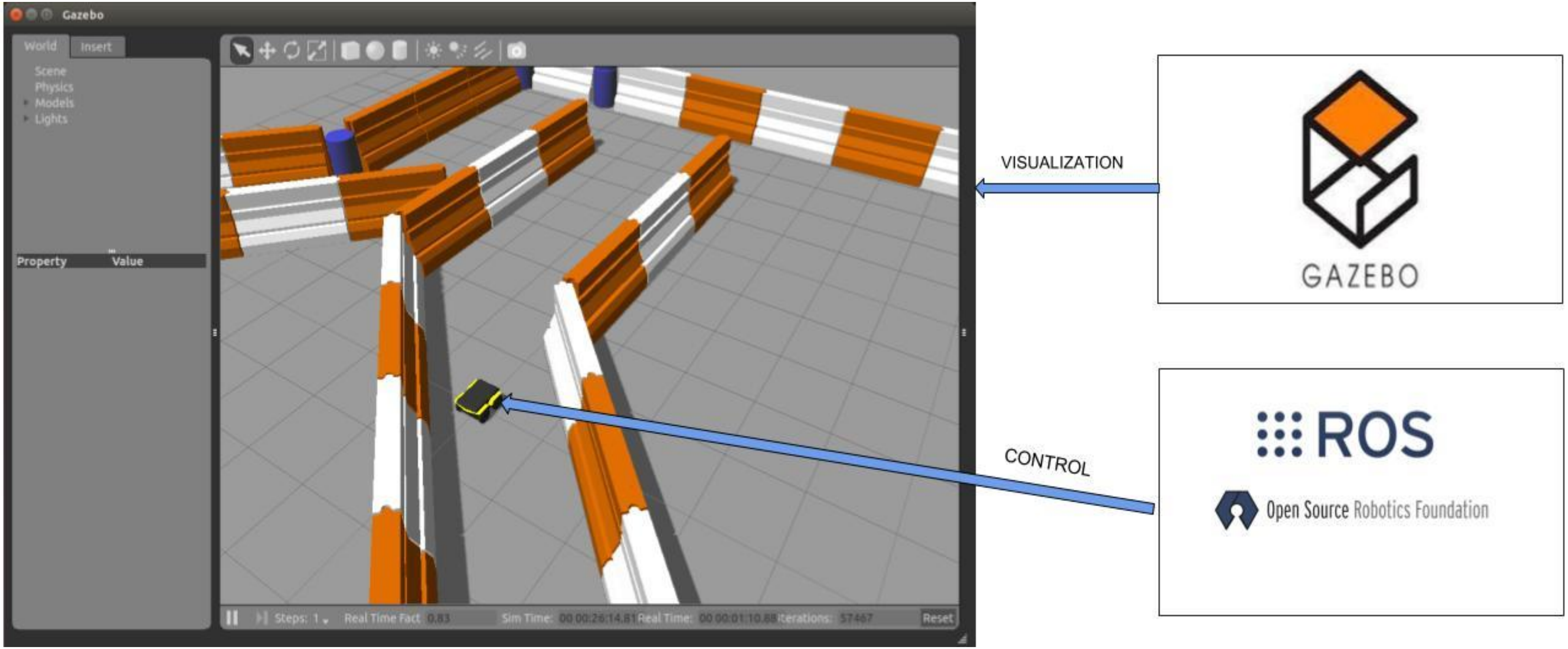
1. Autonomous Navigation
   - The Problem
   - Sensors for ANS
   - Key Problems in ANS
2. Hackathon – II
   - The Challenge
   - Preparatory Work
   - Tasks for the Hackathon
   - Evaluation Criteria

Simulation Environment

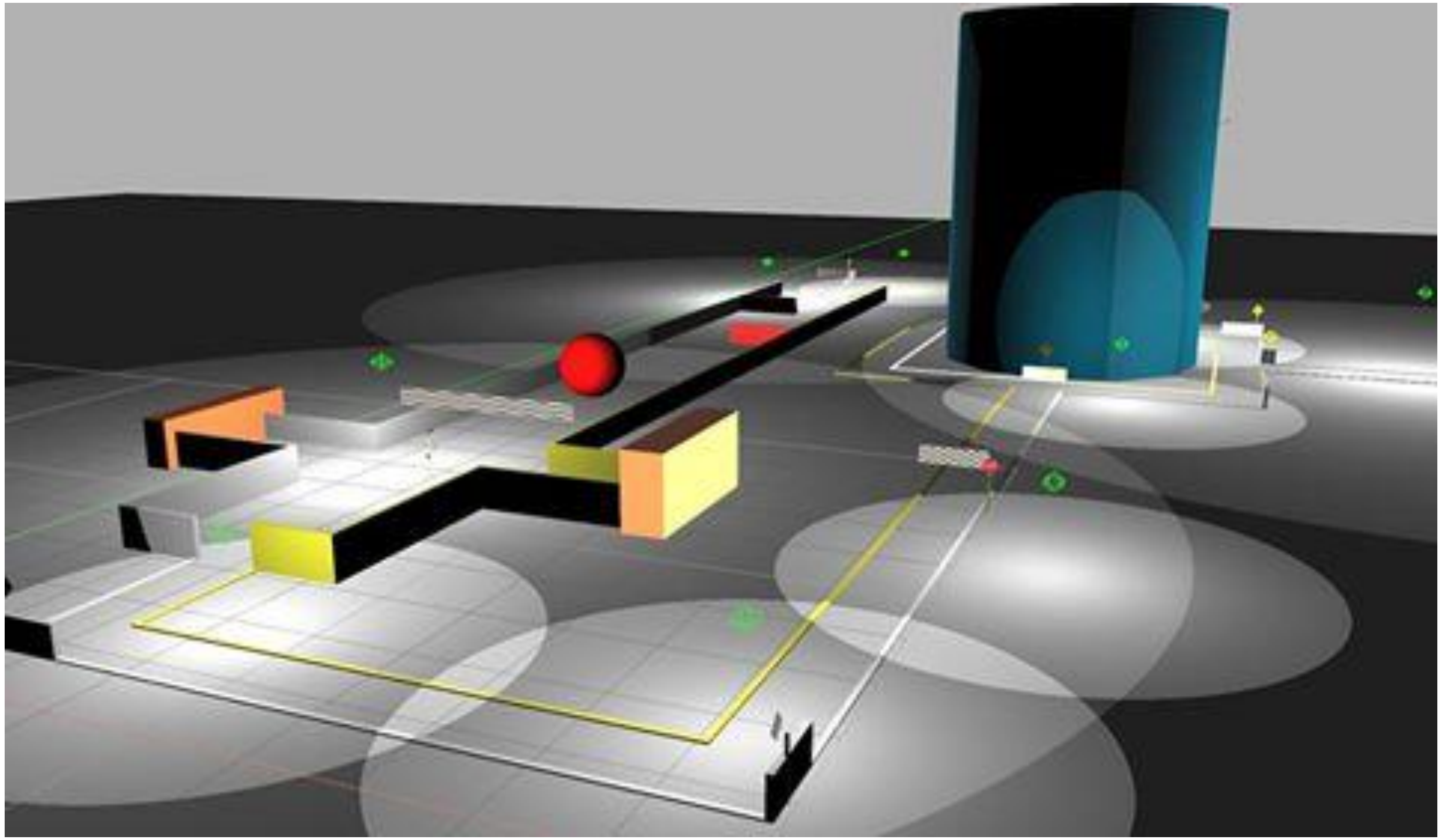
- It is dangerous to test directly on the real robot



Simulation Environment

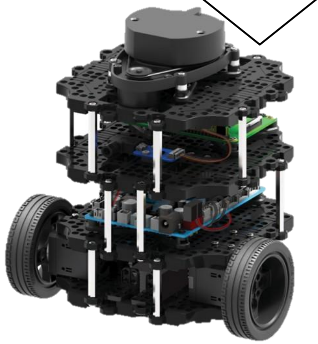


Task



Hey :D !!, This is Jeeves

The map looks scary
 I need help to complete this map



Level 0 – Manual Control

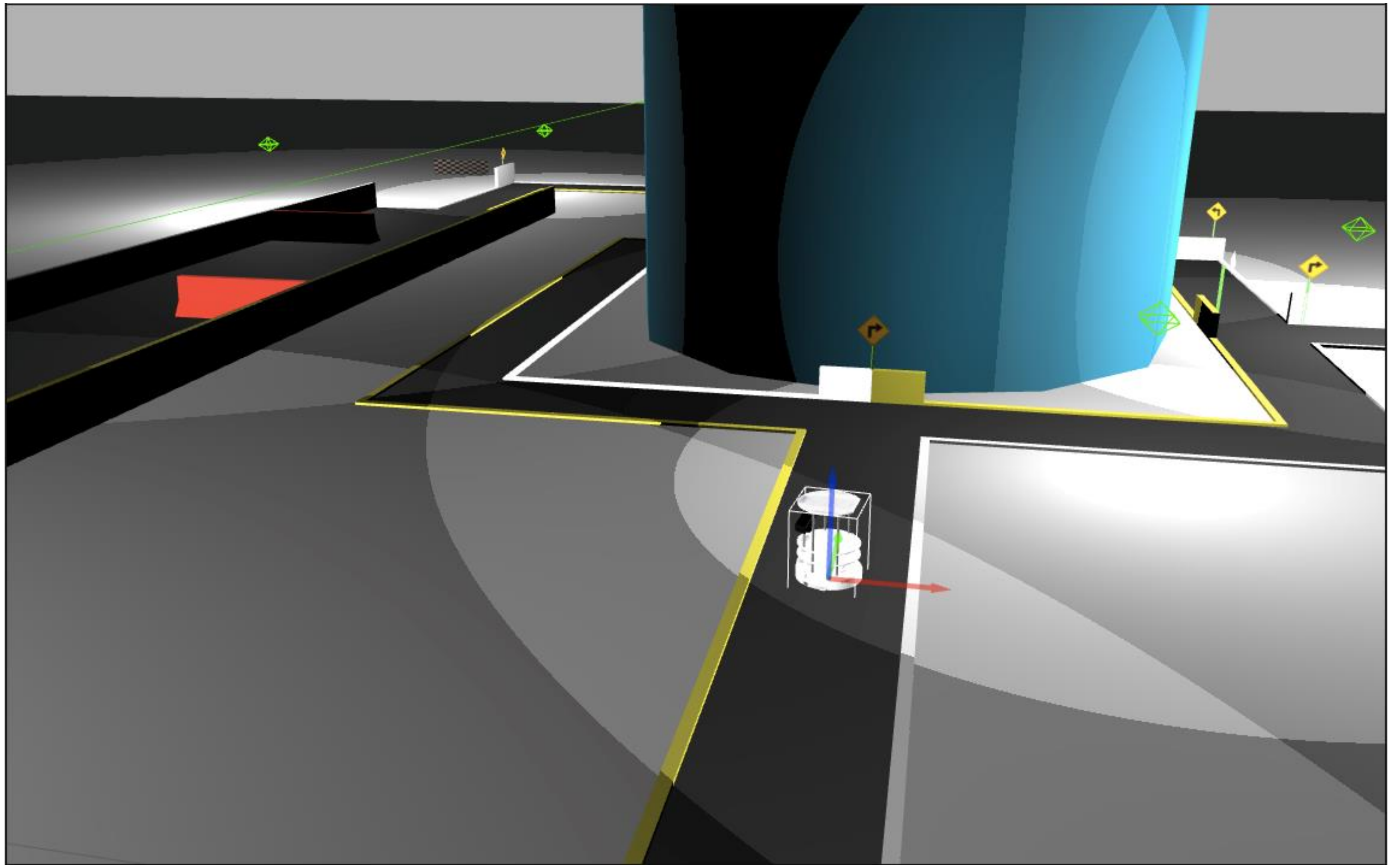
AIML

Tasks -

1. Load the map and turtle-bot onto Gazebo
2. Tele-operate/control it using your keyboard
3. Play around with the Robot in Gazebo !!

How –

Follow the instructions for the Level 0



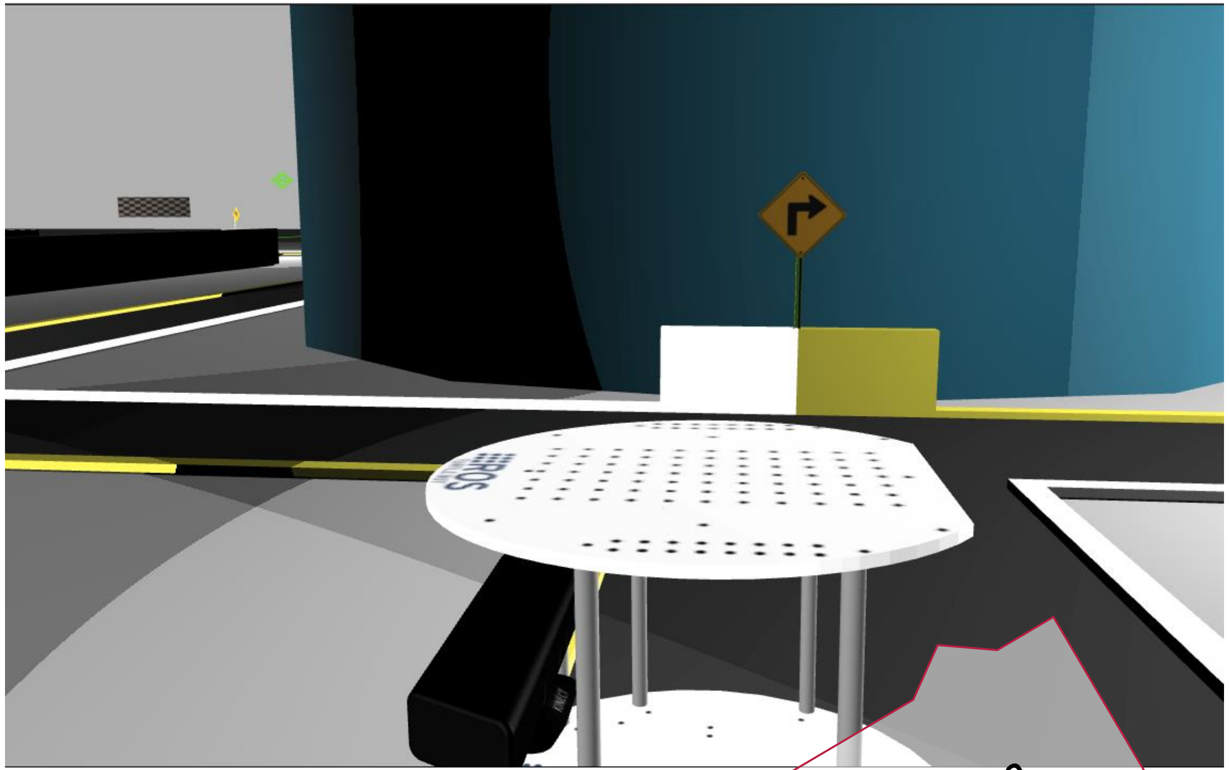
Level 1 – Visual Navigation

Tasks -

1. Given the **color Image** of the first person view of the robot
2. The Robot needs to recognize sign boards and takes actions

How –

1. Build a Image Recognition Module.
2. Integrate the module based on the instructions given for Level 1



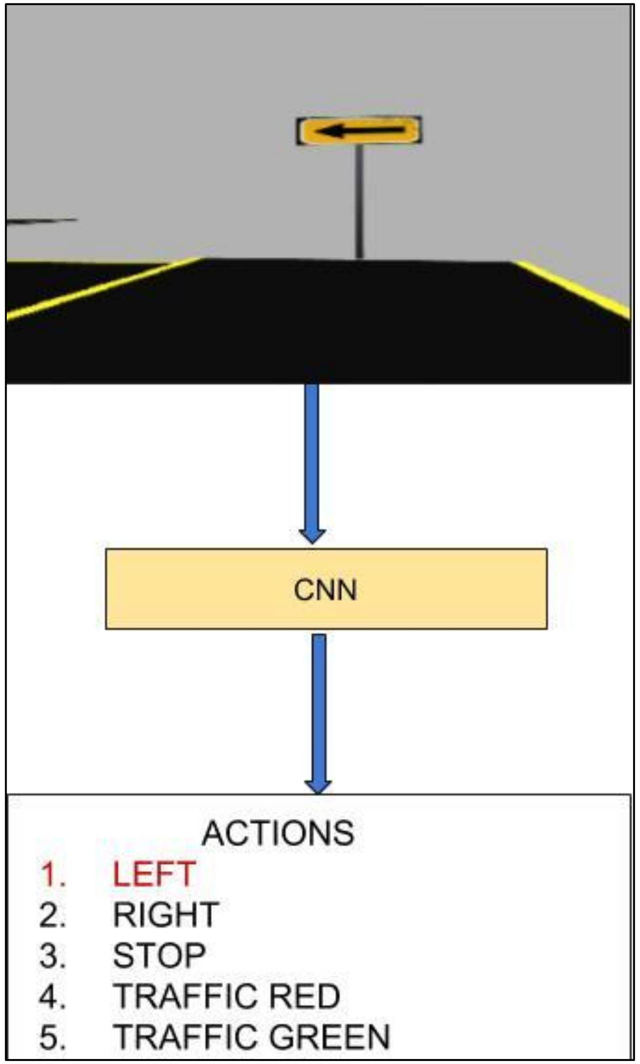
Oh! There is a right turn in front.

Level 1 – Visual Navigation

Data set collection –
1. Data is collected by tele-operating the robot in the map and taking Images.
2. Dataset will be given to you.

Image Recognition Module –
1. Use the dataset and train a simple CNN to understand the traffic sign.
2. Given a first person view of the robot, the model should predict the sign the Robot is Looking at.

Movement of the bot –
1. The robot will move till it reaches a fixed distance from the sign and stops**(This behavior will be given)**
2. The robot will look at the signs and wait for the image recognition



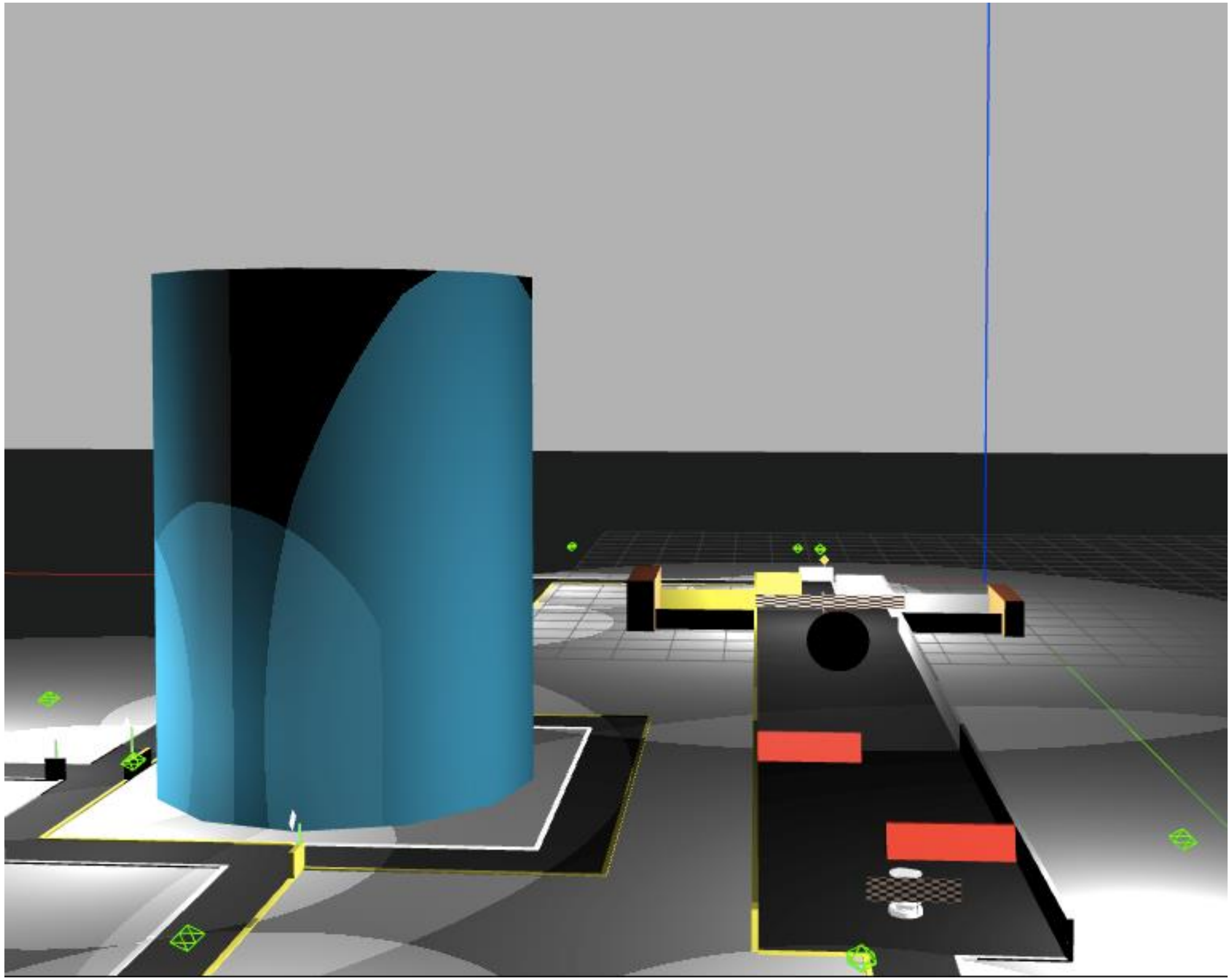
Level 2 – Avoid Obstacles

Tasks -

1. Given Color and Depth Image
2. Control the robot and avoid the obstacles

How –

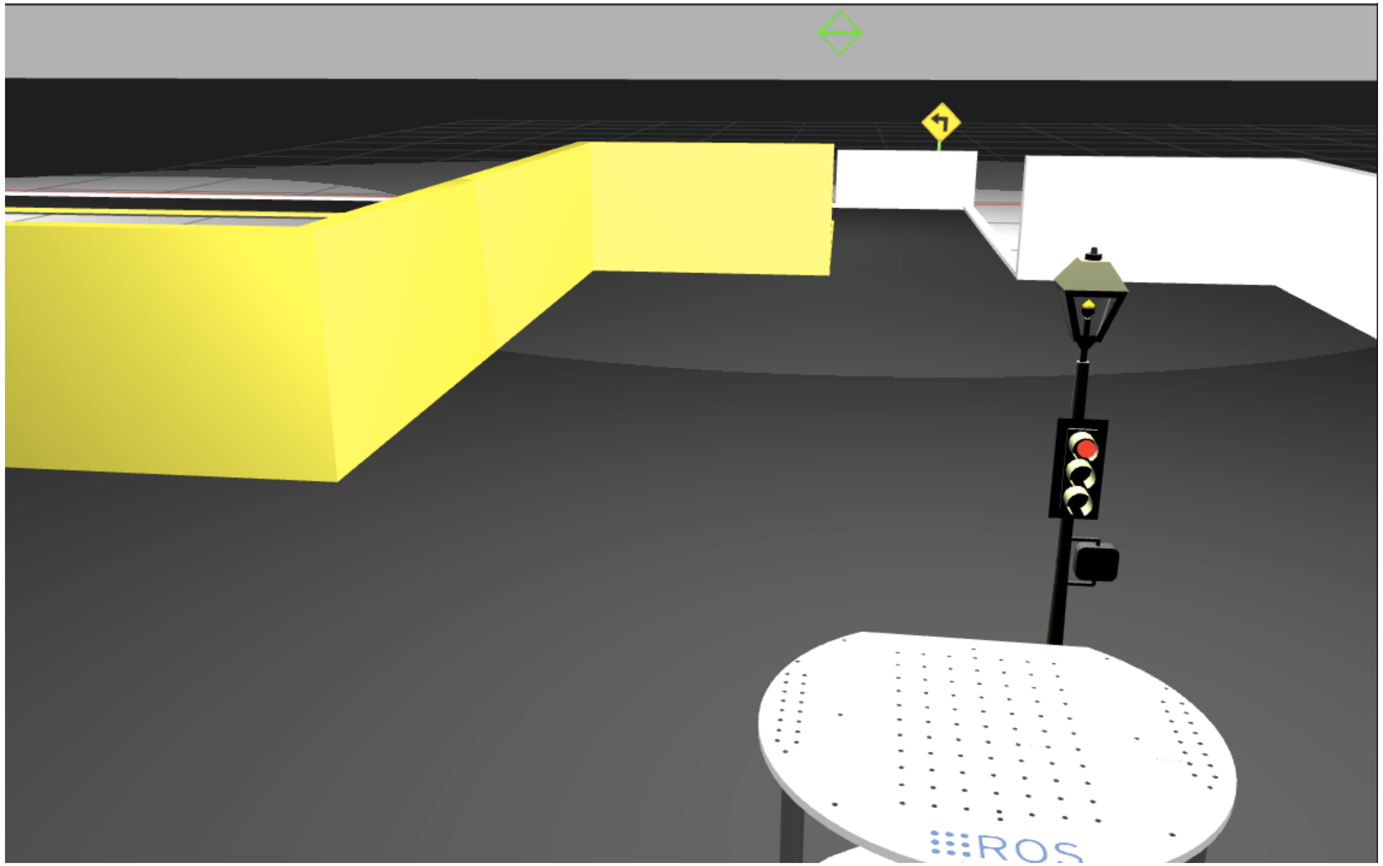
1. Build a Control Module based on the depth images.
2. Integrate the module based on the instructions given for Level 2



Tasks – Follow the Rules

Tasks –
1. Given the color and depth images
2. The Robot recognizes waits till the signal is red and moves when it turns green.

How –
1. Use image recognition module and the control module



Evaluation

- Level 0 – 60 Points
- Level 1 – 50 Points
- Level 2 – 50 Points
- Level 3 – 40 Points

Note: Participants crossing Level 3 will get a chance to deploy their code on the real robot

Demo

Questions?