life.augmented

Robot Navigation

Giuseppe Spampinato – STMicroelectronics

UniCT, May 2024

Agenda

1 **Introduction**

2 Bug Algorithms

3 Self Localization

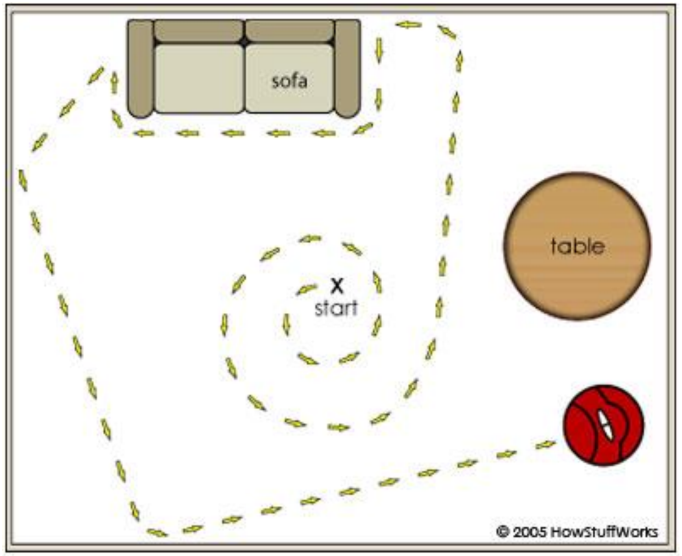
4 Neuronal Network Investigation

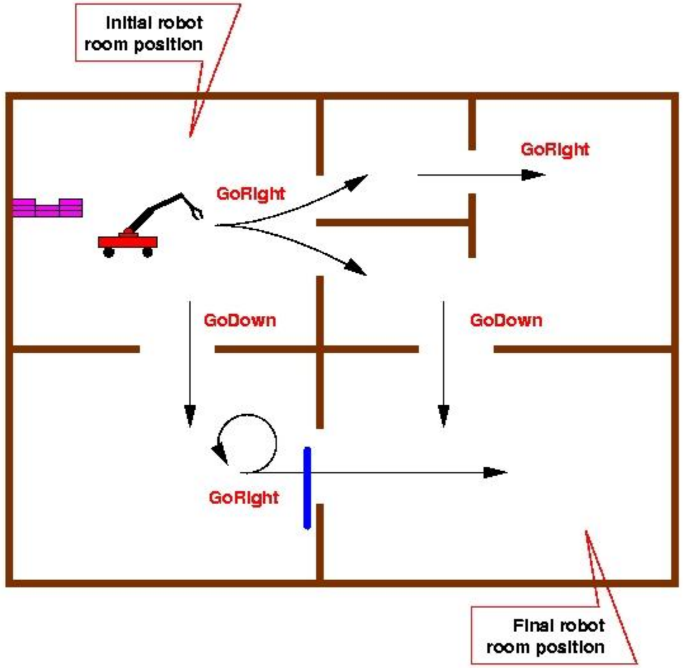
5 Final Remarks

life.augmented

Navigation Main Tasks

Robot Navigation can be summarized into two main tasks:



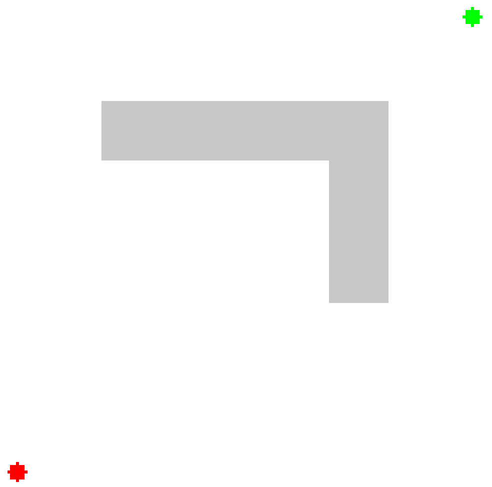


1. Obstacle Avoidance (OA) 2. Point to Point (P2P) Navigation

1. Robot goes around until manual stop avoiding to hit obstacles; map reconstruction and map coverage can be considered as add-on features (e.g. vacuum cleaner).
2. Robot moves towards a target point and it stops when target is reached. Map reconstruction is mandatory (e.g. crane).

Point to Point Navigation

P2P Navigation algorithms can be divided into two main categories:



A. Global Navigation System

B. Local Navigation System

A. Environment information should be already known (map, grid, cells, …). The ideal path is searched at the beginning and the robot moves in the specified environment (e.g. A*).
B. Environment information can be unknown, since the robot must construct the path to reach the goal step by step. (e.g. Bug Algorithms).

Agenda

1 Introduction

2 **Bug Algorithms**

3 Self Localization

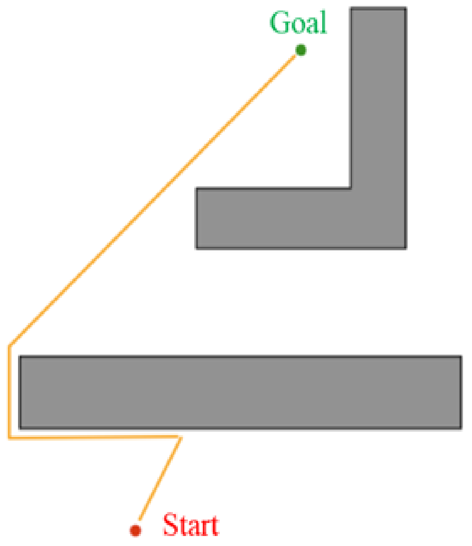
4 Neuronal Network Investigation

5 Final Remarks

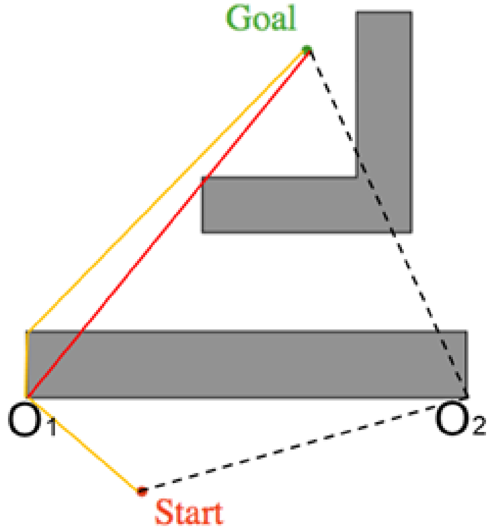
life.augmented

Bug Algorithms

- Several P2P algorithms exist in literature, among them Bug Algorithms are the simplest:
  - Bug-0, aka "common sense algorithm" (Lumelsky et al., 1987)
    - Two steps algorithm: Towards and Circumnavigate
      (circumnavigate is not a simple task, e.g., in case of irregular shapes)
    - Contact algorithm: it needs to be as close to the obstacle as possible
    - there is no precise indication about the direction to take (in the circumnavigation step).
  - Tangent Bug (Kamon et al, 1998)
    - The robot moves following the path which minimize the local tangent distances, defined as the sum of two distances: between robot position and border O_i and between border O_i and *Goal.* Then the robot circumnavigates the obstacle until it can move towards *Goal*.
    - It gives precise indication about which direction to take.
    - Once border O_i is reached, a "follow boundary" function is needed to overcome object.



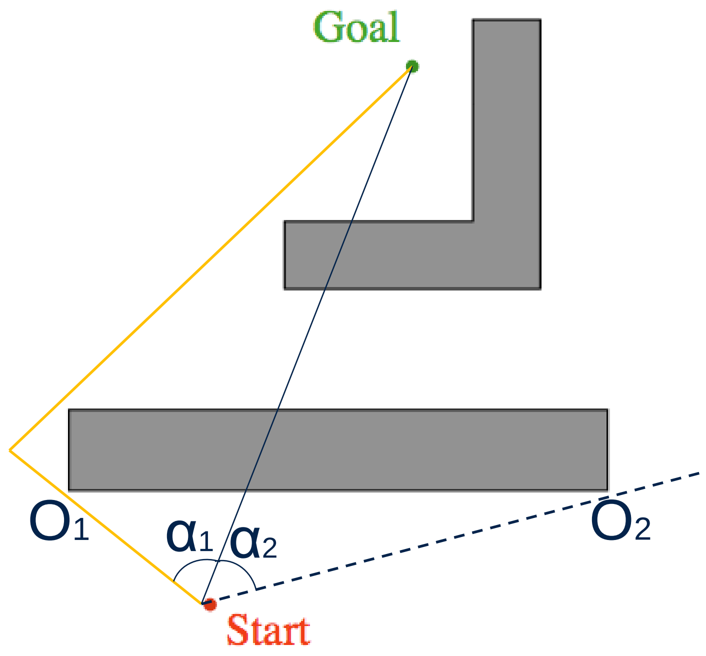
Bug-0



Tangent Bug

life.augmented

Proposed solution: Towards and Tangent (1/2)



Proposed Algorithm (basic idea)

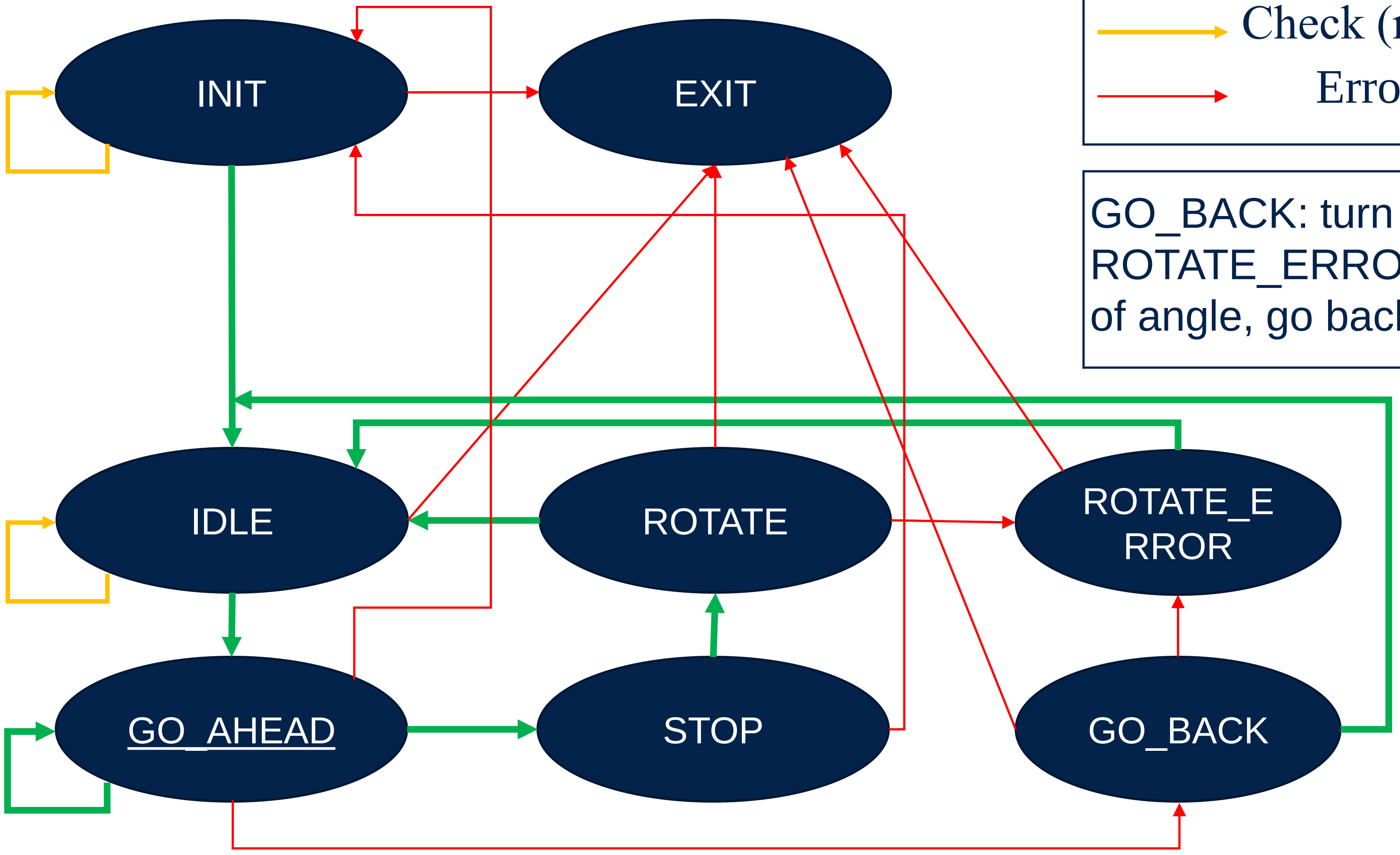
1) [Head towards goal];
2) When obstacle is in the path, choose the direction with lower angle.
Follow the chosen direction, [until you can move towards goal again (leave point)];
3) continue from (1)

- Strenght:
  - It is really simple;
  - No need to execute “follow boundary” instruction;
  - Precise indication about which direction to take.

G. Spampinato, A. Bruna, D. Giacalone, G. Messina
"Low Cost Point to Point Navigation System"
ICARA 2021, Prague, Czech Republic, February 2021

Proposed solution: Towards and Tangent (2/2)

State Machine for Obstacle Avoidance



Regular Behavior
Check (motors, laser, …)
Error Handling

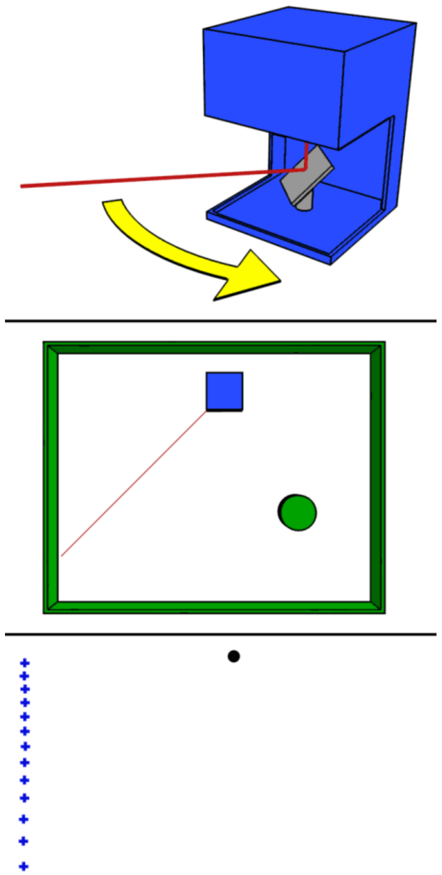
GO_BACK: turn 180º and go back
ROTATE_ERROR: rotate by opposite of angle, go back, repeat angle

Measurement Performance

- For Model A2M7/A2M8 Only

Item	Unit	Min	Typical	Max	Comments
Distance Range	Meter(m)	0.15	-	12	Based om white objects with 70% reflectivity
Angular Range	Degree	-	0-360	-	-
Distance Resolution	mm	-	<0.5	-	<1.5 meters
			<1% of the distance		All distance range*
Angular Resolution	Degree	0.45	0.9	1.35	10Hz scan rate
Sample Duration	Millisecond(ms)	-	0.25	-	-
Sample Frequency	Hz	2000	4000	8000	
Scan Rate	Hz	5	10	15	The rate is for a round of scan. The typical value is measured when RPLIDAR takes 400 samples per scan

Figure 2-1 RPLIDAR Performance



LIDAR (*Laser Imaging Detection And Ranging)* is a method to determine distances by targeting an object with a laser and measuring the time for the reflected light to return to the receiver

Obstacle Avoidance Example (1/2)

The Robot starts going forward up to the distance limit of 40 cm, then turn left or right (choosing the direction with less rotation to make) and then continues up to it is stopped.

life.augmented

Obstacle Avoidance Example (2/2)

Obstacles are lower than the LIDAR, so the Robot cannot avoid obstacles.
In this case, it recognizes the problem and go back.

life.augmented

Agenda

life.augmented

Self Localization: TinySlam

- TinySlam is based on the probabilistic method called Monte Carlo localization.

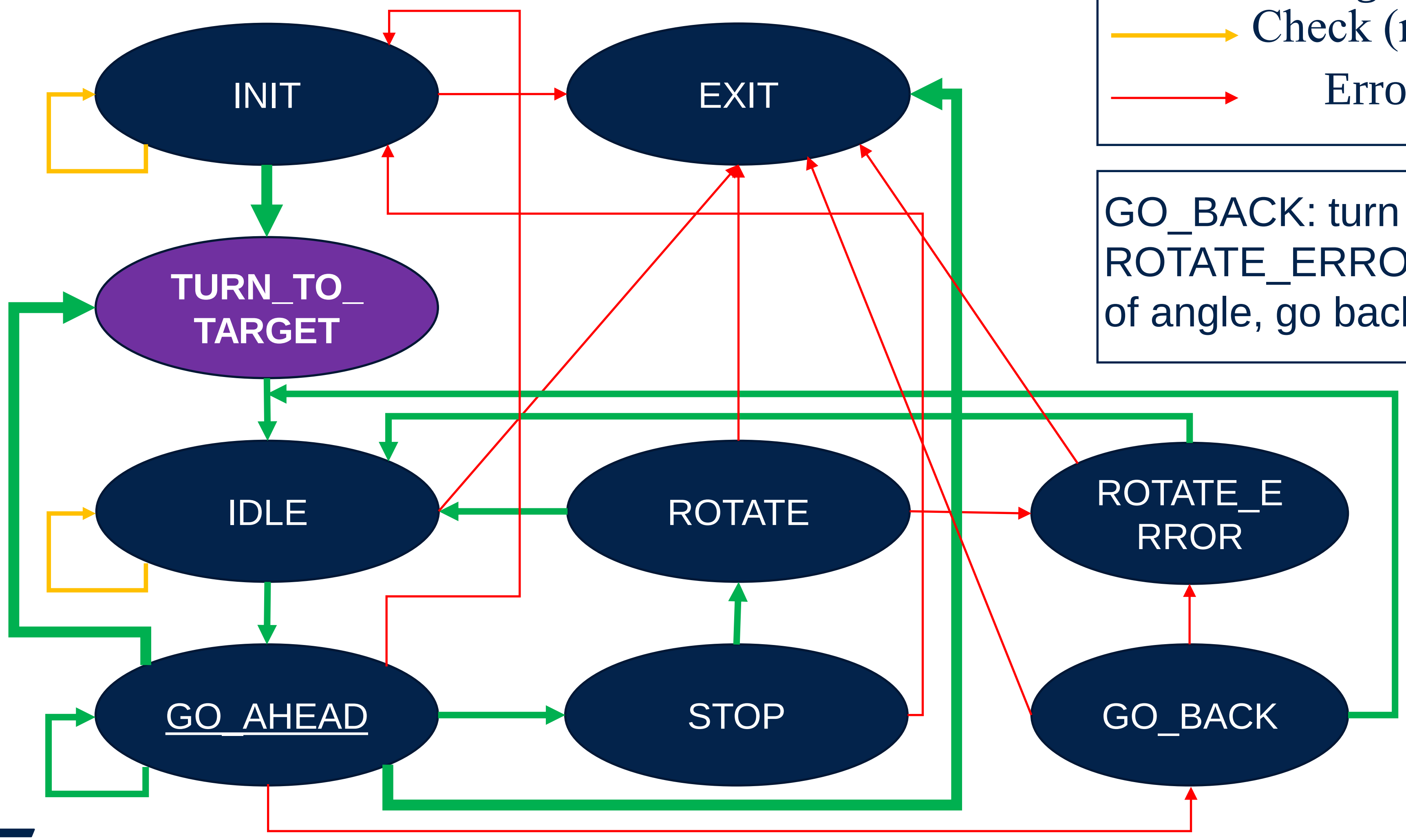
- Monte Carlo localization is an algorithm that approximates a robot’s position on a map using “particles”, where each particle is a hypothesis for the position and bearing of the robot. Using the most recent map and the new odometry, it guesses the best new position for the robot and updates the map accordingly.

- Hypothesis are taken random in each try, according to the ziggurat approach, which covers the target density with the union of a collection of sets from which it is easy to choose uniform points.

- Pro: It is really simple, it uses low memory, so it is suitable for low-power applications.

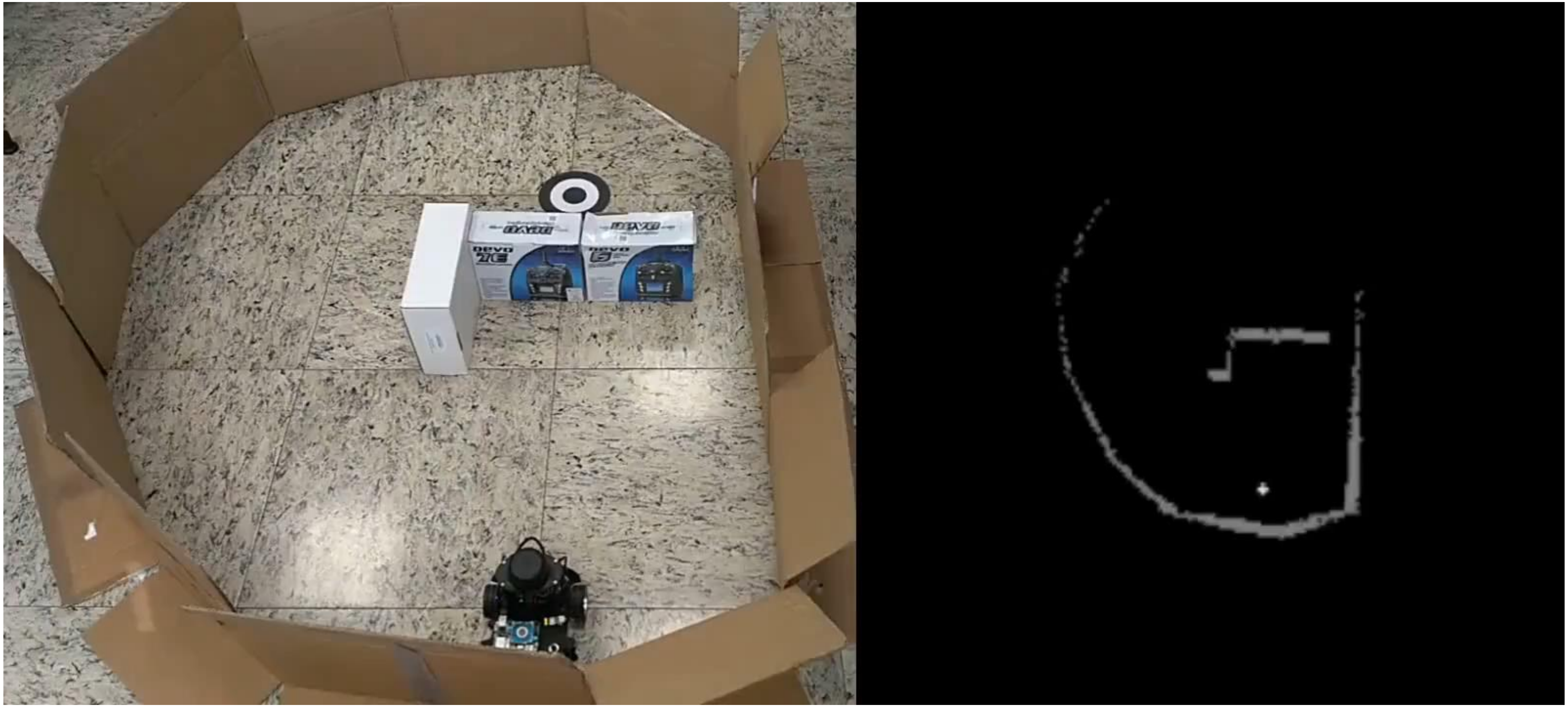
Proposed solution: Towards and Tangent

State Machine for Point to Point Navigation

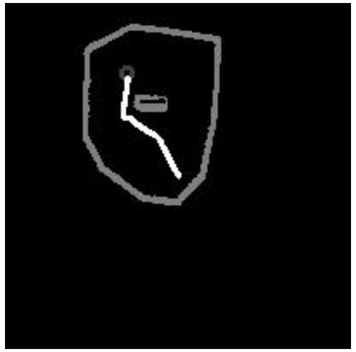


GO_BACK: turn 180º and go back
ROTATE_ERROR: rotate by opposite of angle, go back, repeat angle

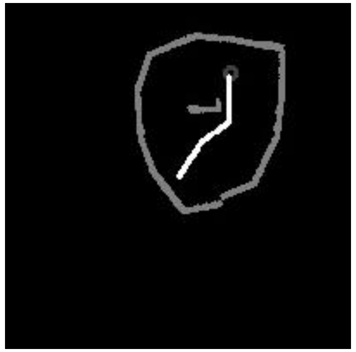
Point To Point Example



Precision Test



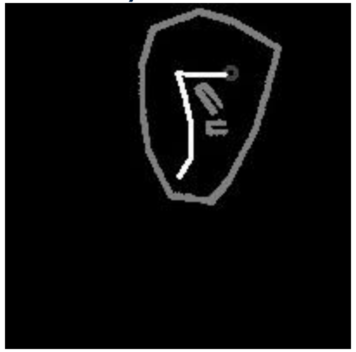
1 obstacle (left)
P=1482; ER=1.01%



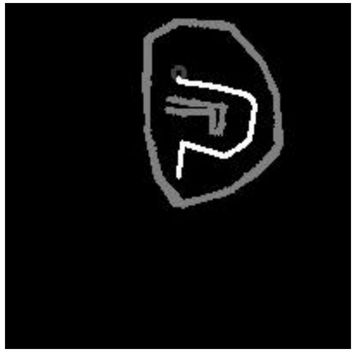
1 obstacle (right)
P=1457; ER=1.16%

2 obstacles (left)
P=2010; ER=1.24%

P = path length (in mm)
ER = Estimation Error (%)



2 obstacles (right)
P=1960; ER=1.17%



1 big obstacle (left)
P=2826; ER=1.38%

1 big obstacle (right)
P=2448; ER=1.55%

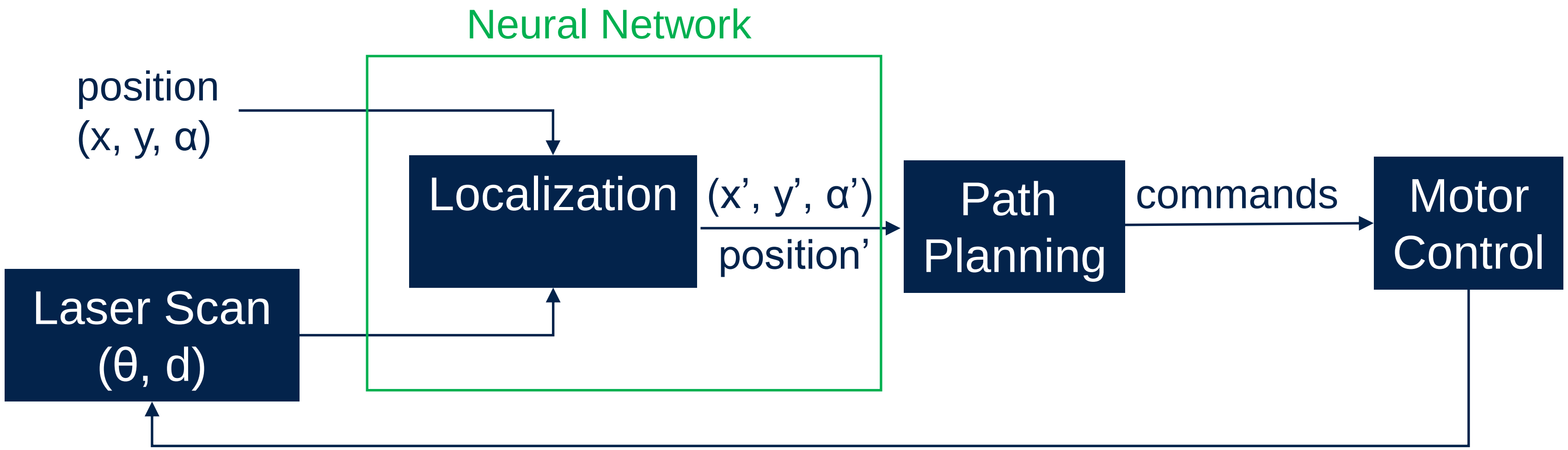
Average Error: 1.25%

life.augmented

Agenda

life.augmented

Point to Point NN Schema

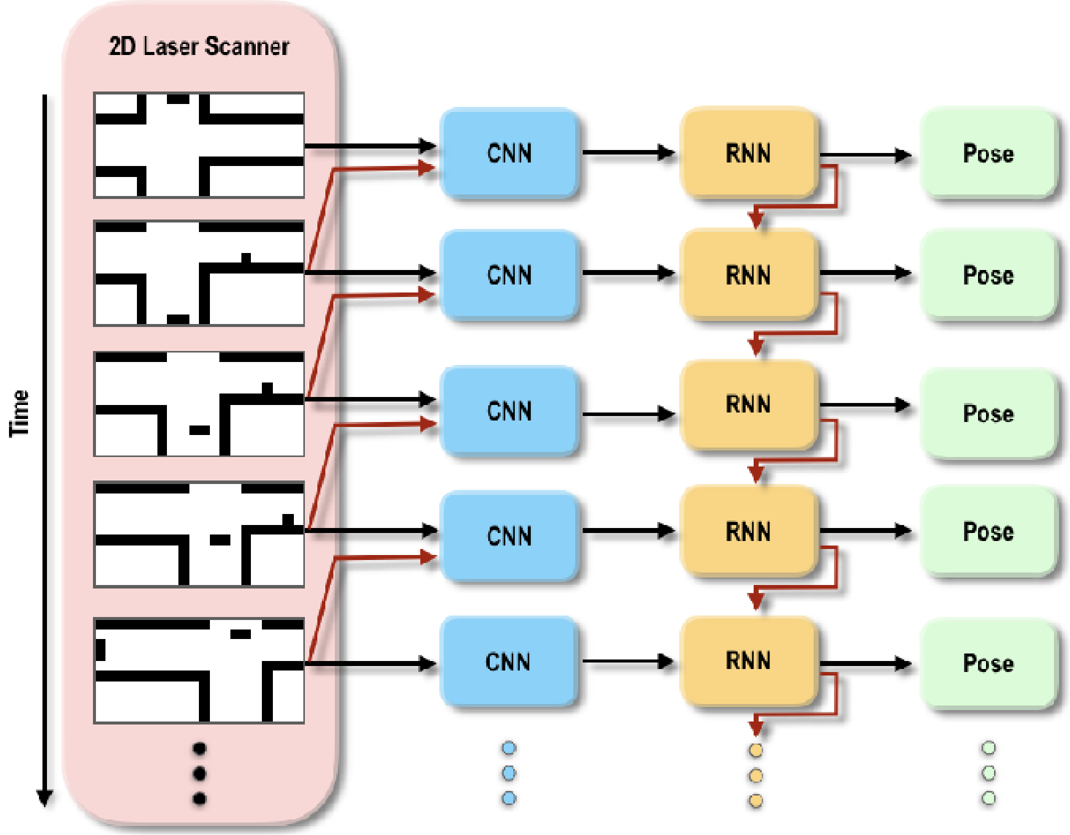


- M. Valente, C. Joly, A. de La Fortelle, "An LSTM Network for Real-Time Odometry Estimation", IEEE Intelligent Vehicles Symposium (IV), 2019.

"The application of deep learning techniques for this purpose using laser scanners is still considered as a new challenge and only few papers have addressed it" (2019).

Potential problems: sparse data (uniformity), noise, accumulation errors, complexity.

Prior Art

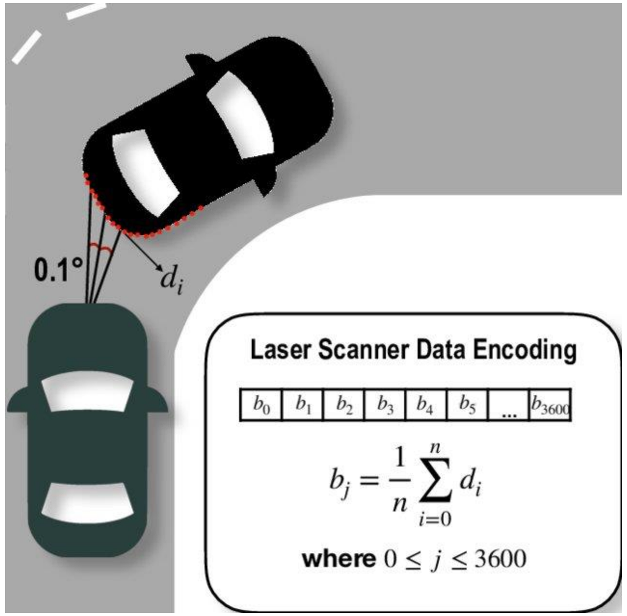


Laser Scan (θ, d)

Feature Extraction

Localization

position’ (x’, y’, α’)



Laser scanner raw data is separated into 0.1° bins. Average depth of all the points in a specific bin is stored. The result is a 3601 size vector.

- M. Valente, C. Joly, A. de La Fortelle, “An LSTM Network for Real-Time Odometry Estimation”, IEEE Intelligent Vehicles Symposium (IV), 2019.

Data Pre-processing (1/2)

A pre-processing step was required to handle data non-uniformity

LIDAR angles LIDAR ranges	→	b_0	b_1	b_2	…	b_{N-1}

Laser scanner raw data is separated into β° bins, for a total of N=360°/β° bins

β	N	Article	Year
0.10	3600	"An LSTM Network for Real-Time Odometry Estimation"	2019
0.25	1440	"Deep Learning for Laser Based Odometry" "Deep Learning for 2D Scan Matching and Loop Closure"	2016 2017

life.augmented

Data Pre-processing for NN (Mat2NN) (2/2)

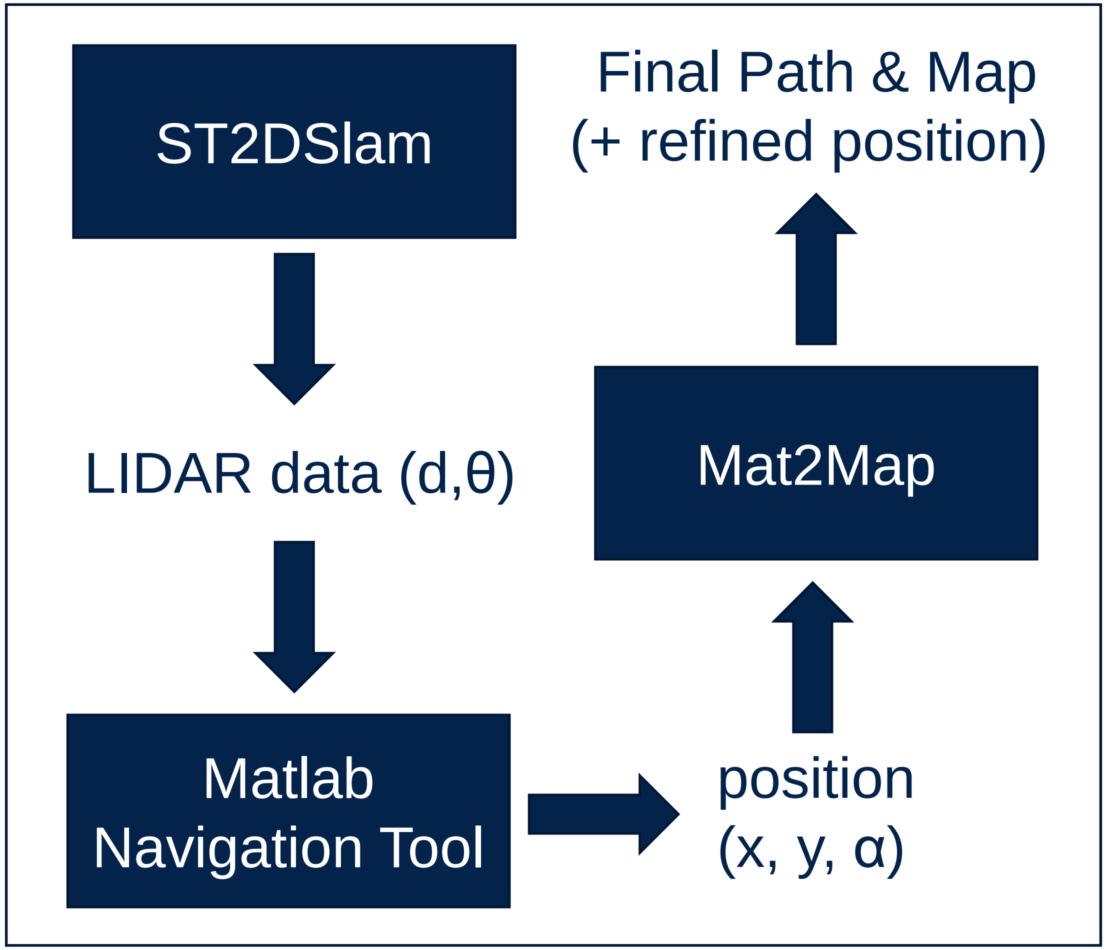
β	N	Collisions per frame	Final Array Density
0.10	3600	0.05	9%
0.25	1440	0.13	22%
0.50	720	0.48	46%
1.0	360	2.54	89%

Results obtained with one sequence of 410 scans just for example
Collisions = number of ranges in the same bin
Density = number of non-zero bins

- How to solve collisions?
  - Valente et alii: just take the average.
  - Our choice: take the minimum (if different from 0)
    to not introduce false object in the scene,
    since lower ranges are more precise.

life.augmented

Ground truth generation



- Matlab's Navigation Tool is based on Google Cartographer:
    - really slow: more then one minute to elaborate on PC a couple of LIDAR data;
    - enough robust to be used as reference;

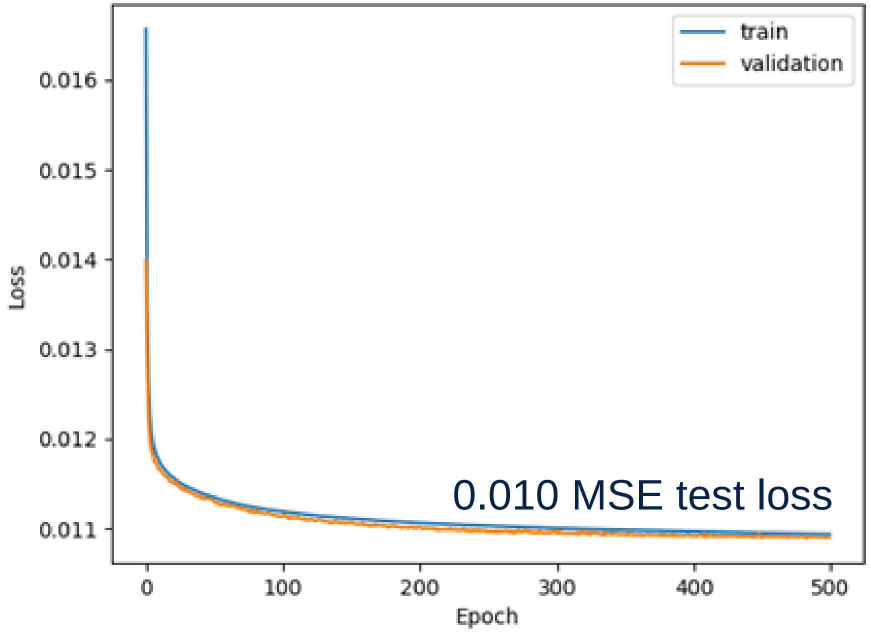
- Ground truth is composed by 51,000 samples, containing:
    - Two LIDAR scans (array of distance/angle)
    - (Δx,Δy,Δα)

life.augmented

DNN proposed approach (1/2)

- Different neural network configurations have been trained/tested, using the TensorFlow framework and Keras wrapper varying:
  - the angle resolution β, i.e., the number of bins N, which determine the input data dimension;
  - the last two layers of the neural network, using Dense layers and Long Short-Term Memory (LSTM) layers.

N	NN Model	NN Parameters
3600	CNN+LSTM	21,078,563
3600	CNN+Dense	5,343,779
1440	CNN+LSTM	15,835,683
1440	CNN+Dense	2,722,339
720	CNN+LSTM	14,262,819
720	CNN+Dense	1,935,907
360	CNN+LSTM	13,214,243
360	CNN+Dense	1,411,619



about 1 cm MSE test loss. Enough?

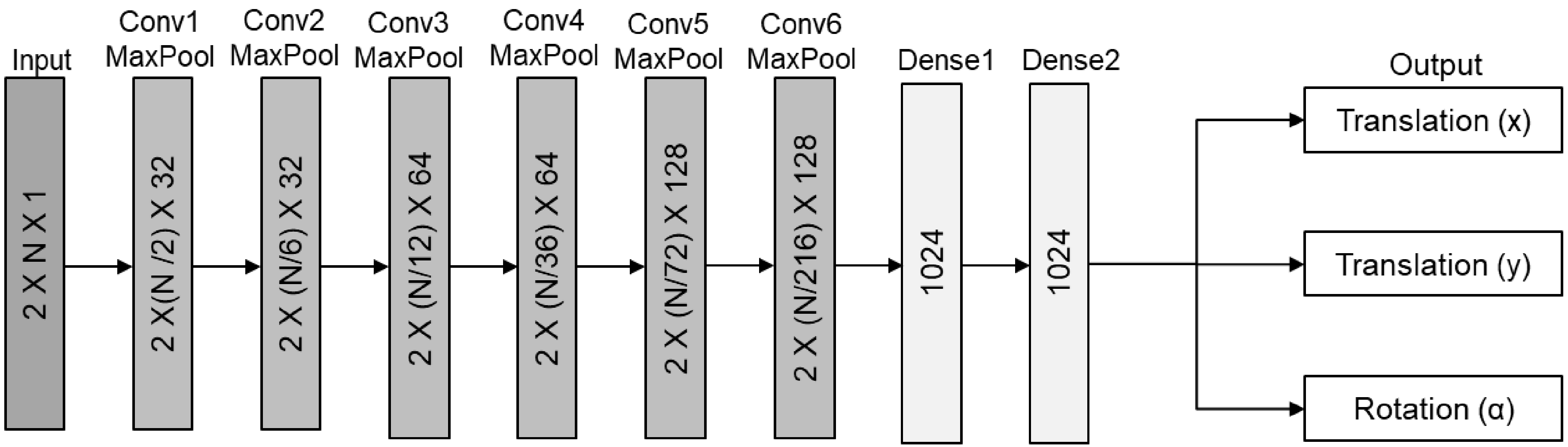
life.augmented

DNN proposed approach (2/2)

- Training details:
  - low learning rate of 0.0001 for function cost minimization;
  - 500 epochs
  - batch size: 32
  - training optimizer: Adam

G. Spampinato, A. Bruna, I. Guarneri, D. Giacalone
"Deep Learning Localization with 2D Range Scanner"
ICARA 2021, Prague, Czech Republic, February 2021

- Best results were obtained with the topology shown below:



Note: N is the number of points in the resampled scan

life.augmented

Agenda

life.augmented

Final Remarks

- An overview of different navigation tasks and categories has been shown;
- A complete obstacle avoidance and point to point navigation system with LDAR+IMU, with Towards and Tangent (TaT) proprietary algo has been presented;
- System is very simple and it is implemented in ST microcontrollers with very low CPU speed, RAM and FLASH;
- NN navigation investigation has been exposed. Numerical results obtained from the NN (about 1 cm of MAE) is acceptable, but not enough to obtain a satisfactory trajectory because:
  - Small MAE (Mean Absolute Error) value does not guarantee small absolute errors samples by samples;
  - Errors accumulated samples by sample are not at zero-mean, so the trajectory will degrade progressively.
- Area of current investigation:
  - AI for alternative navigation/SLAM strategies.

Thank you

life.augmented