Local Planners with Deep Reinforcement Learning for Indoor Autonomous Navigation

2021 I-RIM Conference
October 8-10, Rome, Italy
ISBN: 9788894580525
DOI:10.5281/zenodo.6367976

Local Planners with Deep Reinforcement Learning for Indoor Autonomous Navigation

1st Mauro Martini
DET, Politecnico di Torino
PIC4SeR, Politecnico di Torino
Turin, Italy
0000-0002-6204-3845

2nd Vittorio Mazzia
DET, Politecnico di Torino
PIC4SeR, Politecnico di Torino
SmartData@Polito, Politecnico di Torino
Turin, Italy
0000-0002-7624-1850

3rd Simone Angarano
DET, Politecnico di Torino
PIC4SeR, Politecnico di Torino
Turin, Italy
0000-0002-4445-9783

4th Dario Gandini
DET, Politecnico di Torino
PIC4SeR, Politecnico di Torino
Turin, Italy
0000-0002-5013-7561

5th Marcello Chiaberge
DET, Politecnico di Torino
PIC4SeR, Politecnico di Torino
Turin, Italy
0000-0002-1921-0126

***Abstract*—Autonomous indoor navigation requires an elaborated and accurate algorithmic stack, able to guide robots through cluttered, unstructured, and dynamic environments. Global and local path planning, mapping, localization, and decision making are only some of the required layers that undergo heavy research from the scientific community to achieve the requirements for fully functional autonomous navigation. In the last years, Deep Reinforcement Learning (DRL) has proven to be a competitive short-range guidance system solution for power-efficient and low computational cost point-to-point local planners. One of the main strengths of this approach is the possibility to train a DRL agent in a simulated environment that encapsulates robot dynamics and task constraints and then deploy its learned point-to-point navigation policy in a real setting. However, despite DRL easily integrates complex mechanical dynamics and multimodal signals into a single model, the effect of different sensor data on navigation performance has not been investigated yet. In this paper, we compare two different DRL navigation solutions that leverage LiDAR and depth camera information, respectively. The agents are trained in the same simulated environment and tested on a common benchmark to highlight the strengths and criticalities of each technique.**

***Index Terms*—Indoor Autonomous Navigation, Autonomous Agents, Deep Reinforcement Learning**

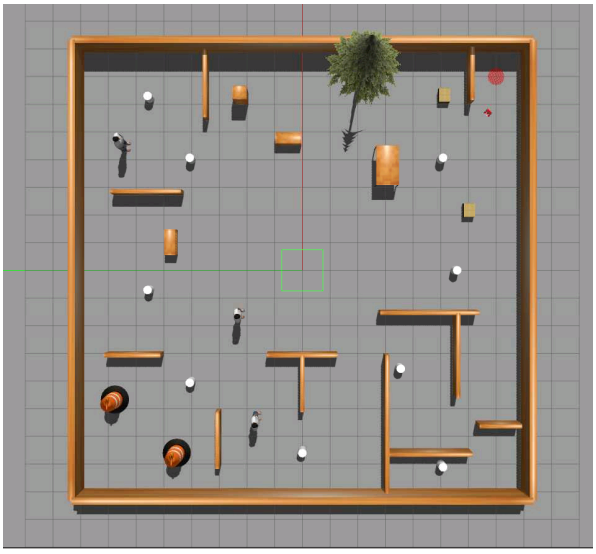

Fig. 1. Representation of a common simulated environment for training an agent. If robot dynamics and task constraints are correctly encapsulated into the simulation, it is possible to deploy on the real setting with minimal loss of performance.

I. Introduction

Autonomous navigation is a challenging task in the research area of robotics, which has been tackled with numerous contributions and different approaches. Indeed, global and local path planning [1], [2], localization and mapping [3] are only some of the required tools that each year undergo heavy research from the scientific community to achieve necessary requirements for full automation. Among them, learning methods have been investigated in recent years due to the successful spreading of Deep Learning (DL). In a Reinforcement Learning (RL) framework, an agent learns by experience through the interaction with the environment where it is placed, avoiding the need for a huge dataset for the training process. [4] RL has shown very promising results in fields as diverse as video games [5], energy usage optimization [6], and with Chiang et al [7] deep reinforcement learning has been firstly used to obtain a robust agent able to map noisy low-level 2-D LiDAR observations to robot linear and angular velocities. The policy obtained through a plain and fast simulation process is a lightweight, power-efficient local motion planning system that can be deployed at the edge on very low-cost hardware with limited computational capabilities. Several implementations of RL agents for autonomous robots are already available in literature [8], [9]. However, a detailed study which aims at comparing the behaviour and the performance of RL local

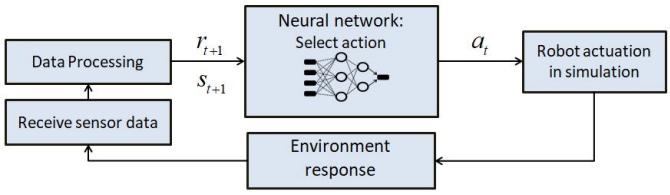

Fig. 2. Schematic representation of the working principle of the system.

planners trained with sensor data of different nature can be a helpful resource for robotics researchers.

In the rest of the paper, we discuss a LiDAR-based solution and a depth camera configuration, highlighting the strengths and weaknesses of both techniques. Nevertheless, both resulting systems successfully solve the complex local path planning problem with limited computational requests and can be easily integrated into a complete navigation stack.

II. Methodology

Deep Deterministic Policy Gradient (DDPG) [10] is the specific Deep Reinforcement Learning (DRL) algorithm used to train an agent in a custom virtual environment realized in Gazebo. An Artificial Neural Network (ANN) is used to directly select suitable actions for the robot, expressed in terms of linear and angular velocity (ANN output). Input information is composed of the robot's orientation and distance from the goal, in addition to the obstacle's distance data collected by a LiDAR or by a depth camera. LiDAR points are 1-D measurements of all-around distances from obstacles. Depth images are single-channel grey-scale images that provide distance information. A significant focus is also devoted to minimizing the computational cost of the model, looking forward to a future hardware implementation. The performance of the navigation system when using depth images is compared to solutions based on LiDAR points to highlight the strength and weaknesses of the two sensor data. On the one hand, a camera offers a piece of rich depth information in a restricted Field of View (FOV). On the other hand, a simple 2D LiDAR is able to cover a wider angle, limiting the depth information to a horizontal plane. The two agents are trained in virtual environment with obstacles and tested in a different challenging simulated world.

III. Virtual Environment framework

ROSbot 2.0 Pro is the standard robotic platform used for the work, which allows to carry out the whole development in simulation thanks to its virtual model. Gazebo is used for modeling virtual indoor scenes where the simulated interaction between the robot and the environment takes place. Standard ROS packages are used to control the robot actuators at a high level, directly acting on the velocity. Simulations are structured in episodes in which the robot has to reach a specific target point. Each episode is composed of a maximum of 600 temporal steps and it terminates with three possible outcomes: *Goal*, *Collision* or *Timeout*. The system working principle can be summarized as follow (see also Figure 2). At a generic time instant, an artificial neural network selects an action a_t for the robot, i.e. a velocity command, according to the state s_t, and it is executed in the virtual world. The environment samples a score r_{t+1}, then it processes the sensor data received, and it sends back a new state s_{t+1}.

The neural network receives as **input** the following data:

- **Goal information**: the pose of the robot is known thanks to the odometry data. They are processed to compute the actual distance from the goal and the robot orientation with respect to the goal (heading angle).
- **LiDAR points**: LiDAR-based navigation relies on LiDAR distance measurements. They are collected and filtered from 359 to 36 all-around points, selecting the minimum distance values for each angular range of $10°$, to guarantee the perception of nearest obstacles.
- **Depth image**: for the visual-based navigation, a Orbbec Astra camera captures raw depth images with a resolution of 480x640, which is reduced to 60x80 to minimize the computational resources required for training. For each pixel a maximum cutoff distance value of $8m$ is accepted.

The **output** is a velocity command composed of a linear velocity in the range $[0, 0.6]m/s$ and an angular velocity in the interval $[-1.5, 1.5]rad/s$.

IV. Training

Reinforcement Learning enables multiple advantages to tackle the navigation task. In fact, the agent is trained without the need for a dataset and results in being a versatile motion planner for mapless navigation. The Deep Deterministic Policy Gradient is the actor-critic algorithm chosen to train the agent. This particular architecture exploits two different neural networks. The **actor** network aims to approximate a deterministic policy to select actions, and it is the one controlling the robot. The **critic** network is responsible for evaluating the actions chosen by the actor, by approximating the action-value function $Q(s, a)$.

For the LiDAR-based local planner, the actor neural network is composed of three fully connected layers with 256 units. Differently, the actor Convolutional Neural Network for visual-based navigation is composed of a stack of convolutional layers to extract features from depth images. Features are concatenated with the goal's information and forwarded to fully connected layers. In both cases, two distinct output neurons, having a *sigmoid* and a *tanh* activation function, respectively map the linear and angular velocity command. A vital element of the training process with reinforcement learning is the reward function. The reward is a numerical score assigned at each time step to the agent to evaluate its behavior. The reward function can be shaped with basic numerical contribution, for example assigning $+1000$ when the goal is reached and -100 in case of a collision with an obstacle. More complex reward functions have been studied with a specific focus on desired behaviors (behavioral rewards). In this study we make use of a behavioral reward function with the aim of let the agent minimize the distance from the target goal while keeping

a satisfying orientation during the navigation. Rewards have also been modulated along the series of training episodes to progressively learn challenging behaviors of the agent. According to the DDPG algorithm, the reward is needed to compute the target values and the critic's gradients. Finally, the actor's gradients are computed, and the ADAM optimizer updates the weights of the networks.

V. Test and results

The agent is tested in virtual environments with obstacles of different shapes. Among them people, tables, walls and columns have been chosen to represent a common indoor scenario with complex obstacles. The ability of the robot to navigate efficiently in an unknown scenario is evaluated with some basic metrics: outcome of the testing episode, total travel time in seconds, total path length in meters. The test is performed by giving 15 target points to be reached in different areas of the environment. Each of them offers a peculiar challenge for the obstacle avoidance task. The robot always starts from a measured point inside the scene. If it collides with an obstacle, the test is considered failed. The same test is carried out with both the LiDAR-based and the depth image-based implementations. Both the agents have shown the ability to reach target points in the presence of static obstacles, without colliding with any entity in the chosen path. The peculiar advantages and limitations of the local motion planner when based on LiDAR points and depth images emerge in the tests. A brief summary of the obtained results is reported in table I. LiDAR-agent shows great robustness and generalization ability, reaching all 15 goals, while the visual agent misses 4 of them. Nonetheless, it demonstrates a faster navigation. The following considerations can be done for a qualitative comparison:

- Both the agents offers a collision-free navigation.
- The two agents choose different paths to avoid obstacles and reach goals. In particular, the visual-agent often avoid narrow passages and prefer to circumnavigate walls.
- The camera-driven agent is not always able to find a suitable path in test world. This is due to the weaker generalization ability of a visual approach. This critical issue can be improved by increasing the visual randomization of the training scenarios. Moreover, depth cameras data suffer the gap between simulated and real sensors. The transfer of knowledge has to be tackled carefully, introducing noise in the simulation to cover the gap with real images.
- LiDAR provides a robust solution for embedding an all around information about obstacles in the agent's state. Generalization is effective with the LiDAR, also when transferring the agent to real world. However, more complex obstacles with narrow or peculiar shape can still represent a limit for this sensor.
- An hybrid solution camera-LiDAR may be a promising configuration.
- More recent DRL algorithm for continuous control can be implemented for a faster and robust convergence of the training in simulation.

TABLE I
SUMMARY OF TEST RESULTS: VISUAL-BASED AND LIDAR BASED AGENT ARE COMPARED ON A POINT-TO-POINT NAVIGATION CHALLENGE COMPOSED OF 15 GOALS TO REACH. RESULTS AVERAGED OVER 3 RUNS.

Agent-Sensor	Goals reached	Total time [s]	Total path length [m]
Depth Camera	11/15	16.02	8.00
LiDAR	15/15	19.16	8.55

VI. Conclusion and Future Work

We briefly discussed how a power-efficient and low computational cost point-to-point local planner learned with RL can constitute a robust short-range guidance system solution. We introduced and compare two alternative architectural implementations based on depth images and LiDAR points, highlighting their strengths and weaknesses with respect to obstacle avoidance. Perception is crucial for local planners, and DRL agents show peculiar behaviours and abilities when trained with different sensor data.

References

[1] Vittorio Mazzia, Francesco Salvetti, Diego Aghi, and Marcello Chiaberge. Deepway: A deep learning waypoint estimator for global path generation. *Computers and Electronics in Agriculture*, 184:106091, 2021.

[2] Enrico Sutera, Vittorio Mazzia, Francesco Salvetti, Giovanni Fantin, and Marcello Chiaberge. Indoor point-to-point navigation with deep reinforcement learning and ultra-wideband, 2020.

[3] Sajad Saeedi, Bruno Bodin, Harry Wagstaff, Andy Nisbet, Luigi Nardi, John Mawer, Nicolas Melot, Oscar Palomar, Emanuele Vespa, Tom Spink, et al. Navigating the landscape for real-time localization and mapping for robotics and virtual and augmented reality. *Proceedings of the IEEE*, 106(11):2020–2039, 2018.

[4] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[5] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529, 2015.

[6] Elena Mocanu, Decebal Constantin Mocanu, Phuong H Nguyen, Antonio Liotta, Michael E Webber, Madeleine Gibescu, and Johannes G Slootweg. On-line building energy optimization using deep reinforcement learning. *IEEE Transactions on Smart Grid*, 2018.

[7] Hao-Tien Lewis Chiang, Aleksandra Faust, Marek Fiser, and Anthony Francis. Learning navigation behaviors end-to-end with autorl. *IEEE Robotics and Automation Letters*, 4(2):2007–2014, 2019.

[8] Francisco Leiva, Kenzo Lobos-Tsunekawa, and Javier Ruiz-del Solar. Collision avoidance for indoor service robots through multimodal deep reinforcement learning. In *Robot World Cup*, pages 140–153. Springer, 2019.

[9] Reinis Cimurs, Jin Han Lee, and Il Hong Suh. Goal-oriented obstacle avoidance with deep reinforcement learning in continuous action space. *Electronics*, 9(3):411, 2020.

[10] Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning, 2019.