POLITECNICO DI TORINO

Master's Degree in Mechatronic engineering

Title: Deep Reinforcement Learning and Ultra-Wideband for autonomous navigation in service robotic applications
Candidate: Enrico Sutera [s253364]
Supervisor: Prof. Marcello Chiaberge

Abstract

Autonomous navigation for service robotics is one the greatest challenges and there's a huge effort from scientific community. This work is born at PIC4SeR (PoliTo Interdepartmental Centre for Service Robotics) with the idea of facing the aforementioned challenge merging rediscovered and promising technologies and techniques: Deep Reinforcement Learning and Ultra-Wideband technology.
Over few past years the world has seen a huge advance in the field of Artificial Intelligence, especially thanks to Machine Learning techniques. The latter include a branch called Deep Reinforcement Learning (DRL) that involves the training of Artificial Neural Network (ANN) from experience, i.e. without the need of huge starting datasets. Here DRL has been used to train an agent able to perform goal reaching and obstacle avoidance.
Ultra-wideband (UWB) is an emerging technology that can be used for short-range data transmission and localization. It can be used in GPS-denied environments, such as indoor ones. In this work UWB has been used for localization purposes. UWB is supposed to be a key technology in future: many giant companies are involved and Apple has already inserted an UWB chip in its latest product.
It has been used a differential drive robot as implementation platform. The robot is controlled by an ANN (which has robot pose information, lidar information and goal information as input and linear and angular speeds as outputs) using ROS (Robot Operating System). The ANN is trained using a DRL algorithm called Deep Deterministic Policy Gradient (DDPG) in a simulated environment. The UWB has been used in testing phase only.
The overall system has been tested in a real environment and compared with human performances, showing that it is able - in some tasks - to match or even outdo them. There have been satisfying results and it is believed that, although there are strong limitations given by the difficulty of the challenge, the system complies with expectations and constitutes a good baseline for future work.

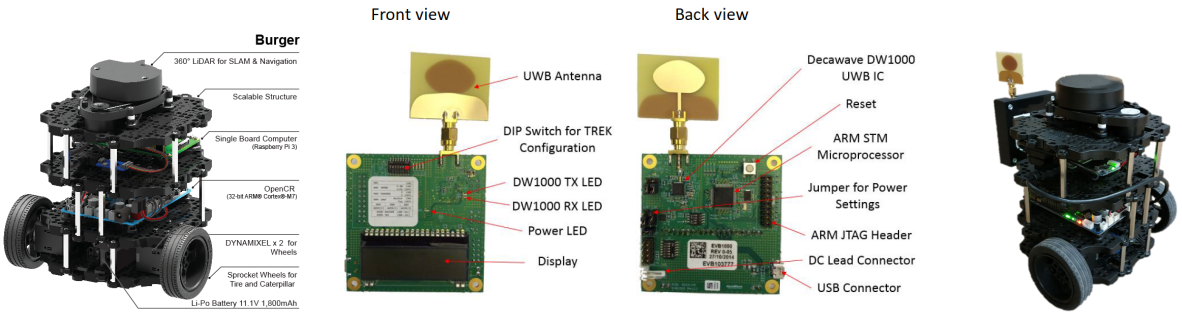
1 Introduction

Recently there's have been a great attention to service robotics, whose focus is to assist human beings, generally performing dull, repetitive or dangerous tasks, as well as household chores. Many of them require the robot to navigate in a environment, almost always non static or even unknown. The centre of attention of this thesis is to tackle the *navigation problem*, especially in indoor environment, by using a novel approach, in order to achieve more flexibility without the need for expensive high-performance hardware. The system includes a TurtleBot3 burger controlled by a neural network able to make it reach a given destination while performing obstacle avoidance, by using lidar, odometry and Ultra-wideband information. The neural network is trained using Deep Reinforcement Learning. The purpose of this thesis is hence to provide a general baseline that may be included in several service robotic specific applications which require autonomous navigation.

2 Hardware and implementation

The robotic platform was TurtleBot3 *burger* model (see figure 1a) which is a differential driver mobile robot equipped many sensors, such as 3 axis accelerometer, gyroscope and magnetometer, lidar and motors built-in magnetic encoders. TurtleBot3 provides ROS packages to handle the robot at high-level, e.g. by performing control in velocity.
Concerning the Ultra-wideband technology, EVB1000 units (see figure 1b) were used as both receivers and transmitters. One of them is the target, while four are used as fixed anchors. What they provide

are the distances from the latter to the formers, which then can be used to compute the position of the target, hence of the robot, in a fixed reference frame. An EVB1000 unit was mounted on the burger (see figure 1c).



(a) *TurtleBot3 burger model.*

(b) *EVB1000 Ultra-wideband unit device.*

(c) *TurtleBot3 burger model with the EVB1000 unit mounted on board.*

Figure 1: Overview on hardware

The robot is controlled by a network which receives as **input** (see scheme in figure 2):

- **Lidar data**: though it provides 359 measurements, only 60 are used, to simplify the learning process and the network complexity. Basically the complete 360 ° sector is divided in 60 parts and for each of them the minimum value is taken, discarding outliers. This allows the controller to be aware of narrow objects using less inputs, that otherwise may not be detected.
- **Goal data**: TurtleBot3 burger gives access to its pose, estimated via odometry. However only orientation is taken among all these data (it is computed using magnetometer), while the position is estimated via the ultra-wideband measurements, which do not suffer from cumulative error. Nonetheless the controller is given as input just the distance and the angle with respect to the goal, instead of the coordinates of both of them (and the orientation of the robot).

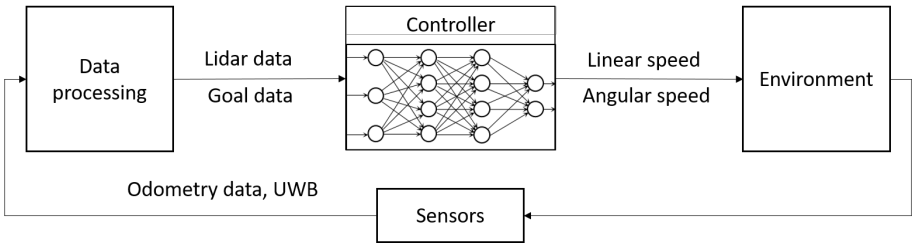


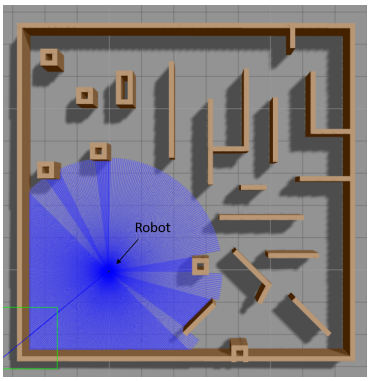
Figure 2: System implementation scheme.

Concerning the **outputs**, the robot controller acts on velocities, sending command for linear (in range $0 \div 0.22\ m/s$) and angular speed ($-1 \div 1\ rad/s$).

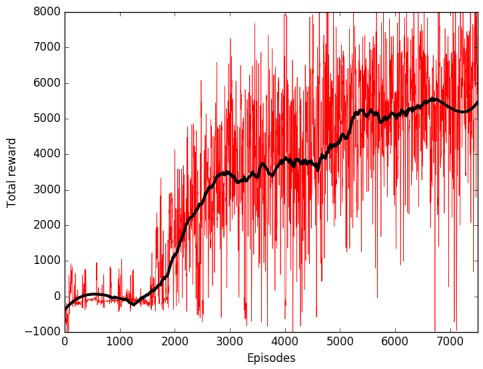
3 Training

The most important part of this work was certainly the training of the artificial neural network. It was done using Deep Reinforcement Learning (DRL), which today is considered the form of automatic learning that more matches the biological learning. Indeed, it does not require a starting dataset, but just raw experiences, in the same way as trial and error processes. The used algorithm is called Deep Deterministic Policy Gradient (DDPG), which falls into the category of actor-critic algorithms. During the training indeed there are two networks involved: the *critic* learns to estimate the value of making actions(combination of linear and angular speed) in given state (representation of the state of the robot, including the environment around), while the *actor* learns a policy, that is the choice of the action in a given state. During the training the agent (the robot) receives a reward at each iteration. This is a

numerical value given by a function which rewards achievement of the goal (+1000), approaching (depends on the state and action made) and penalizes collision(−200). This value is than used as target to perform the training of the critic network, using an optimizer called Adam. Moreover the obtained critic gradient is used to perform the training of the actor network, which is then trained indirectly. During the training, only the odometry is used, since it is trustful for short paths and the robot is re-spawned when colliding. The environment in which the training took place is shown in figure 3a, while the learning curve in figure 3b. The latter shows how the robot gets better performances over the episodes (a episodes ends when the robot collides or after a certain amount of time).



(a) *Training environment. Possible goals to be reached are not shown, however they were distributed in all the map.*



(b) *Learning curve of the agent during the training. The huge noise is due to the difference in difficulty of the goals, in terms of distance and obstacles to be avoided*

Figure 3

4 Test and results

The overall system was finally tested along with some human beings, in a real environment. Indeed, it is not unusual to compare human performance and Deep Reinforcement Learning ones. Four different environment were set-up. Human agents had the chance to control discretely the velocities by using a keyboard and monitor interface and having slightly more information than the neural network. The results obtained evidence that:

- in terms of path optimality, the DRL agent showed to being able to match human performances, indeed both time spent and path travelled have similar values (see *path* and *time spent* in table 1;
- for what concerns the smoothness, the agent could not achieve better result than humans, thought it is influenced by both the control frequency and the continuous action space, indeed, jerk values are higher (table 1).
- the algorithm is quite robust with respect to noise. Indeed data provided by UWB are quite disturbed, though they do not suffer from cumulative error.
- in general the system is satisfactory but it requires the addition of a memory, to escape local maxima.

Agents	p 1	p 2	p 3	p 4	p avg	DRL agent
Total path length (m)	36	34	42	43	39	40
Total time spent (s)	268	229	360	328	296	258
Average linear jerk $(m \cdot s^{-3})$	7.1	6.7	7.8	8.1	7.4	14
Average angular jerk $(rad \cdot s^{-3})$	39.7	35.4	39.6	45.2	40.0	73.8

Table 1: Some data about the test performed. *p 1* stands for *person 1*, while *p avg* refers to the average of all people.