Indoor Localization using Deep Reinforcement Learning.

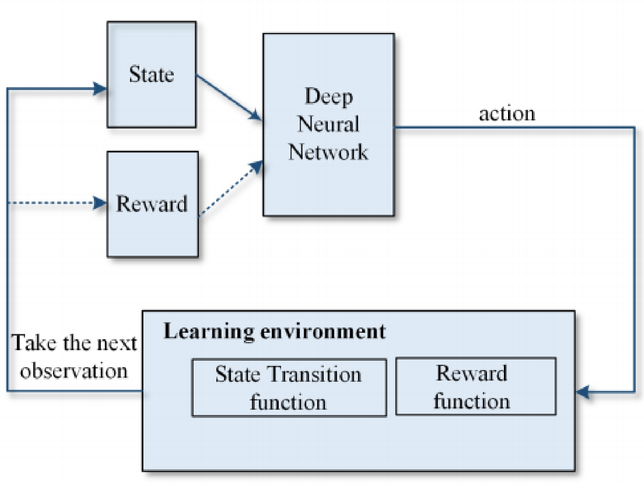
Mohamed EL-KADDOURY, Abdelhak MAHMOUDI and Mohammed Majid HIMMI
mh.kadouri@gmail.com, abdelhak.mahmoudi@um5.ac.ma, himmi.fsr@gmail.com
LIMIARF, Faculty of Sciences, Mohammed V University, Rabat, Morocco

Introduction

Nowadays, there is an increasing requirement for indoor positioning and navigation with Location Based Services(LBSs). Many applications on smartphones exploit different techniques and inputs for positioning. Most of the indoor wireless positioning systems rely on Received Signal Strength Indicator (RSSI) from indoor wireless emitting devices. However, the accuracy of indoor position is easily affected by servals signal interference. In this work, we propose a LBS system using Deep Reinforcement Learning with iBeacon RSSI (RSSI is usually represented by a negative number between 0 and -100), and hope to achieve higher accuracy for indoor positioning.

Deep Reinforcement Learning

To adopt a deep reinforcement learning approach, we need to define the following elements:



(a) Deep reinforcement learning agent

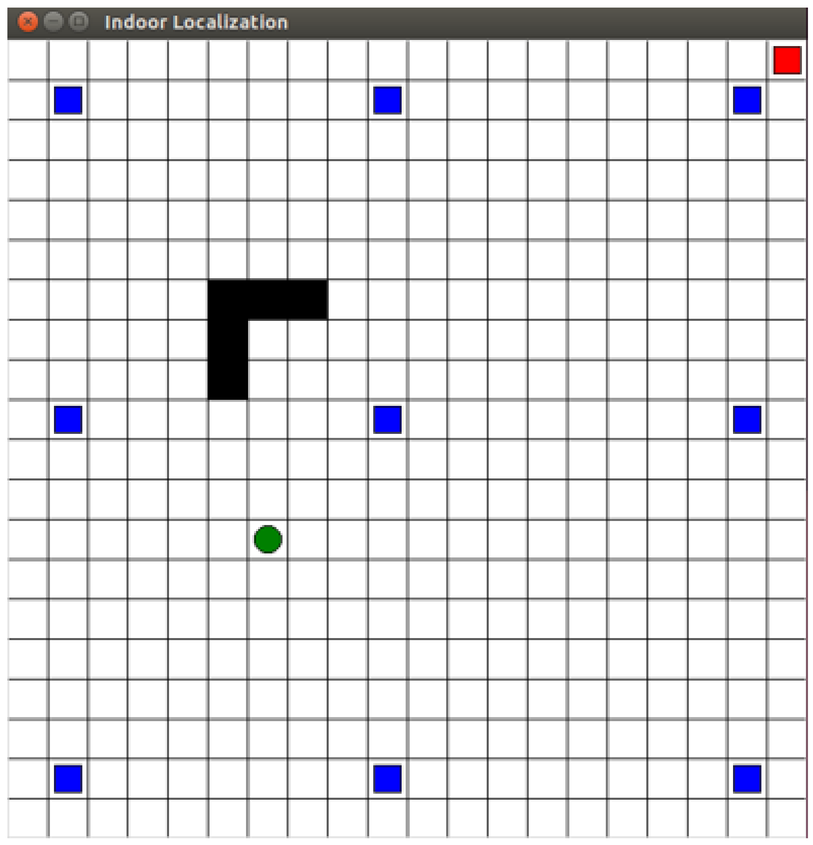
- **Environement:** see the figure (experiments part).
- **Agent:** The positioning algorithm itself is represented as an agent. The agent interacts with the environment over time.
- **States:** The state of the agent is represented as a tuple of these observations:
  1- **a vector of RSSI values,**
  2- **current location (identified by row and column numbers), and**
  3- **distance to the target.**
- **Actions:** The action is to move to one of the neighboring cells in a direction of North, East, West, South. The first action chooses a random state in the grid.
- **Reward function:** the reward function is the reciprocal of the distance error. The reward function has a positive value if the distance to the target point is less than a threshold (δ). Otherwise, the agent receives a negative reward. Whenever the agent is close to the target, it gains more rewards. On the other hand, if the agent wanders away from the target and its distance is larger than a threshold (δ), it gains a negative reward. The reward function is represented as follows:

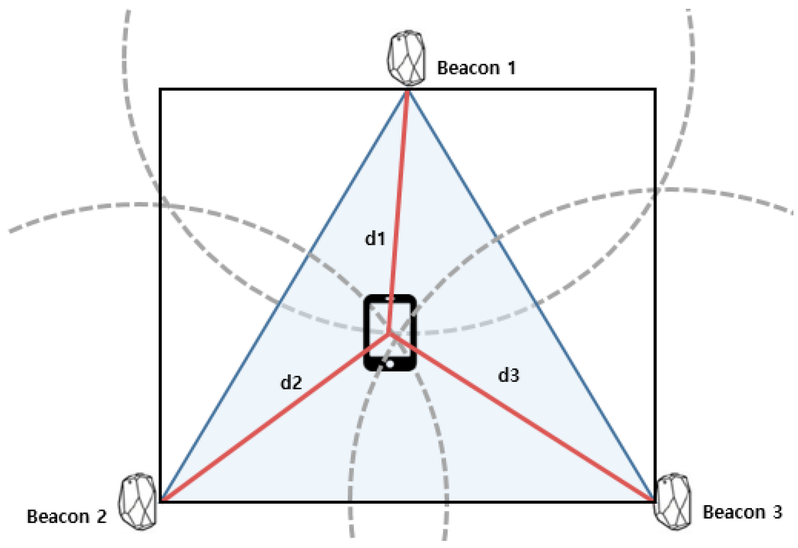
$$r_t = \begin{cases} \frac{1}{\|O_t - S_t\|} & \text{if } 0 < \|O_t - S_t\| \leq \delta \\ -\|O_t - S_t\| & otherwise. \end{cases}$$

in which O_t is the observed location and S_t is the target location.

Experiments

The environment is represented as a set of positions that are labeled by row and column numbers. Each position is also associated with the set of RSSI (distance between the agent and iBeacons) values from the iBeacons (blue square). The agent observes the environment by receiving RSSI values at each time. Our design requires the agent to take action based on the three RSSI observations. The agent (red square) can choose one of the allowed 4 actions to move in different directions. In turn, the agent obtains a positive or negative reward according to its proximity to the right point (green circle) and avoid obstacles (black square). The goal of the agent is to approximate the position of the device that has received the RSSI values from the environment by moving in different directions.





Results

The reward plot shows that the algorithm could rapidly find good solutions, reaching an average score of 0. The algorithm converged pretty rapidly. The reward passed from negative (in the first 400 episodes) to zero.