Multi-robot Sanitization of Railway Stations Based on Deep Q-Learning

Riccardo Caccavale[1], Vincenzo Calà[2], Mirko Ermini[3], Alberto Finzi[1], Vincenzo Lippiello[1] and Fabrizio Tavano[1,2]

[1] *Università degli studi di Napoli "Federico II", Naples, Italy*
[2] *Rete Ferroviaria Italiana, Rome, Italy*
[3] *Rete Ferroviaria Italiana, Firenze Osmannoro, Florence, Italy*

Abstract

Sanitizing railway stations is a relevant issue especially due to the recent evolution of the Covid-19 pandemic. In this work, we propose a multi-robot approach to sanitize railway stations based on a distributed Deep Q-Learning technique. The framework relies on anonymous information from existing WiFi networks to localize passengers inside the station and to develop a map of possible risky areas to be sanitized. Starting from this map, a swarm of cleaning robots, each one endowed with a robot-specific convolutional neural network, learns how to on-line cooperate inside the station in order to maximize the sanitized area depending on the presence of the passengers.

Keywords

Deep Reinforcement Learning, Multi-robot Systems, Experience Replay Buffer

1. Introduction

In recent years, the spreading of diseases such as the Covid-19 has emphasized the problem of sanitizing large and crowded public environments like railway stations. In the present work, our aim is to design a solution for the sanitizing by the Deep Q-Learning technique in a real case of study of interest for Italian railway infrastructure manager RFI s.p.a., in a real environment offered by the most important italian railway station of the capital, Roma Termini. The framework relies on anonymous information from existing WiFi networks to localize passengers inside the station and to develop a map of possible risky areas to be sanitized. Starting from this map, we propose a decentralized approach where a swarm of cleaning robots, each one endowed with a robot-specific convolutional neural network, learns how to on-line cooperate inside the station in order to maximize the sanitized area depending on the presence of the passengers. In the multi-robot sanitizing system literature, the prominent approach is based on coverage path planning (CPP) [1, 2, 3, 4, 5] where the area to sanitize is divided between agents in order to cover the whole space. These approaches are suitable for cleaning and sanitizing the environment with a scalable number of robots, but prioritization issues are hardly considered. MARL frameworks are often proposed to ensure flexibility and scalability in

The 8th Italian Workshop on AI and Robotics - AIRO 2021

riccardo.caccavale@unina.it (R. Caccavale); v.cala@rfi.it (V. Calà); mi.ermini@rfi.it (M. Ermini); alberto.finzi@unina.it (A. Finzi); vincenzo.lippiello@unina.it (V. Lippiello); fabrizio.tavano@unina.it (F. Tavano)

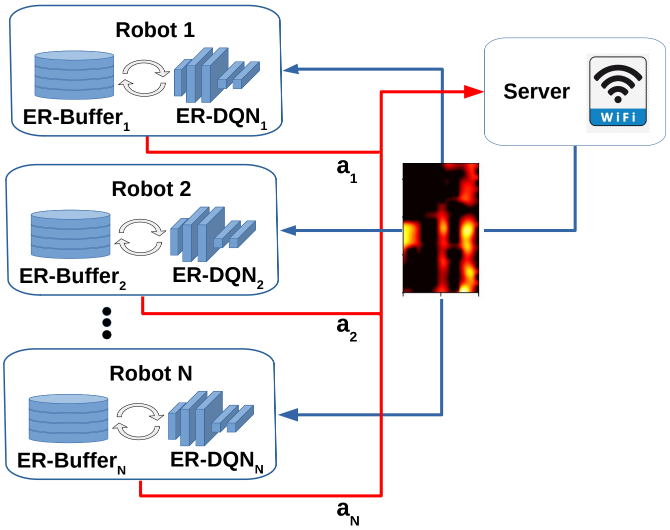

Figure 1: Graphical representation of the framework including multiple agents (left) endowed with agent-specific experience replay buffers and networks, and a single server (right) exploiting WiFi statistics to provide an heatmap of priorities (red to yellow spots) for the agents.

different applications like exploration [6], construction [7], or target-capturing [8], but also in this case priority-based cleaning issues are not commonly covered. An interesting approach is proposed by [8], where multiple agents distributedly learn a collaborative policy in a shared environment using A3C training method in order to achieve a target-capturing task. Inspired by these approaches, we propose a scalable multi-robot sanitizing framework where multiple mobile robots learns to cooperate during the execution of cleaning tasks into large crowded environments, introducing a priority-based strategy.

2. The architecture

Our multi-robot sanitizing problem can be described as follows. Starting from a gridmap M representing the environment to be sanitized, we define S as the set of possible heatmaps (i.e., priority distributions) on the map M, and X as the set possible free-obstacle positions of M. In this setting, we assume k agents, tasked to sanitize the environment M, each one endowed with a set of single-agent actions A. Our aim is to find a set of agent-specific strategies $(\pi_1, \ldots, \pi_k)$ such that each $\pi_i : S \times X \rightarrow A$ drives an agent towards prioritized areas, in coordination with the other agents, in order to maximize the global cleaning effect. This distributed approach is mainly designed to support the scalability: we adopt a client-server approach, where each agent (client) learns a decoupled agent-specific strategy by communicating with a central system (server).

A representation of the overall architecture is depicted in Figure 1. The framework is composed of a set of intelligent agents, representing mobile cleaning robots, each one communicating with the central server. The role of the server (server-side) is to merge the outcomes of the agents activities with (anonymized) data about people positions in order to produce a heatmap for the risky areas to be sterilized. The role of each agent (agent-side) is to elaborate the heatmap by means of an agent-specific Deep Q-Network (DQN) and to update the local strategy π_i considering the environmental settings and the different priorities in the map. In this framework, the cleaning priority can be defined as a heatmap, whose hot/cold points are high/low priority areas to be sanitized. Following this perspective, a state-position couple $(s, x) \in S \times X$ is defined as a 2 channel matrix $m \times n \times 2$ where m and n are the width and the height of the environment, respectively. The first channel s of the matrix represents the cleaning-priority on the environment, whose elements are real numbers in the interval $[0, 1]$, where 1 is the maximum priority and 0 means that no cleaning is needed. The second channel x is a binary matrix representing the position and size of the cleaning area of the robot, which is 1 for the portions of the environment that are in the range of the robot cleaning effect, and 0 otherwise. This matrix can be shown as a heatmap (see map in Figure 1), where black pixels have 0 priority, while colors from red to yellow are for increasingly higher priorities.

In our framework, the update of priorities is performed by the server, which collects the outputs of the single agents, and integrates them considering the position of people and obstacles. More specifically, the cleaning priority is computed from the position of clusters of people by modeling possible spreading of viruses or bacteria. In our setting, we exploit the periodic convolution of a Gaussian filter $\mathcal{N}(\mu, \sigma^2)$ every ψ steps, where μ, σ^2 and ψ are suitable parameters that can be regulated depending on the meters/pixels ratio, the timestep, and the considered typology of spreading (in this work we assume a setting inspired to the aerial diffusion of the Covid-19 [9]). Here, starting from a set of randomly generated clusters, the probability distribution evolves through the iterative convolution of the Gaussian filter. The convolution process acts at every step by incrementally reducing the magnitude of the elements of the heatmap matrix, while distributing the priority on a wider area. Convolution is here exploited to simulate the effects of the attenuation and the spreading of the contamination process over time. We have chosen the parameters of the Gaussian function in order to have a radius of the area, interested by the infection, of 5 meters (μ =0, σ = 0.9). This value is selected considering that we know the position of a cluster of people with an WiFi average positioning error of accuracy of about 3 meters as described in [10] and we consider also that the distance of safety is about of 2 meters between peoples that make use of the indicated surgery masks during the actual period of emergency caused by the Covid-19 diffusion [9]. In the map (see Figure 2, right) there are several black areas (0 priority) that are regions of space associated with the static obstacles of the environment (shops, rooms and walls inside the station). These areas are assumed to be always clean, hence unattractive for the robots. When an agent moves into the environment with an action $a_i \in A$, the region in the neighborhood of the newly reached position is cleaned by the server, which sets to 0 the associated priority level. In our framework, we propose a simple multi-agent variation of the experience replay method proposed in [11]. Following this approach, each of the k agents is endowed with a specific replay buffer, along with specific *target* and *main* DQNs, that are synchronously updated with respect to the position of the agent and to the shared environment provided by the server (see Figure 1). The local *reward* function r_i is designed to drive the agents toward prioritized areas of the environment (hot points), while avoiding obstacles and already visited areas (cold points). In this direction, we firstly introduce a cumulative priority function cp_i that summarizes the importance of a cleaned area:

$$cp_i = \sum_{(j,l)} s_i(j,l)x_i(j,l) \tag{1}$$

as the sum of the element-wise priorities from matrix s_i in the area sterilized by the agent i (i.e. where $x_i(j,l) = 1$). The value in Equation 1 is then exploited to define the reward r_i for the agent i:

$$r_i = \begin{cases} cp_i & \text{if } cp_i > 0; \\ penalty & \text{otherwise.} \end{cases} \tag{2}$$

Specifically, when an agent i sanitizes a priority area, the reward is equal to the cumulative value cp_i; otherwise, if no priority is associated to the cleaned area (i.e., $cp_i = 0$) a negative reward $penalty < 0$ is earned (we empirically set $penalty = -2$ for our case studies). This way, agents receive a reward that is proportional to the importance of the sanitized area, while routes toward zero-priority areas, such as obstacles or clean regions, are discouraged. Notice that in this framework, when the action of an agent leads to an obstacle (collision), no motion is performed. This behavior penalizes the agent (no further cleaning are performed), thus producing an indirect drive towards collision-free paths. Moreover, as long as an agent moves through the environment it leaves a wake of cleaned space behind. This way, since the priority of already visited areas is 0, agents can indirectly observe their mutual behavior from the priority update, in so avoiding explicit communication, hence robots in our experiments are not directly aware of the position of the other agents which is indirectly estimated from their paths.

3. Case Studies

A graphical representation of the environment is shown in Figure 2. We selected a region of space of 100 × 172 meters in front of the rails, where people usually stands waiting for the incoming trains. From that region we also isolated shops, stairs and walls as obstacles to be avoided by the robot during the sanitizing process (black areas in the Figure, 2, right). Agents can move by one pixel in any direction, hence the set A includes 8 actions (4 linear and 4 diagonal) while, in case one action leads to an inconsistent location (obstacle or out of bound) the agent stays in the current location. In this setting, we propose two case studies: in the first one we assess the system performance during the learning phase considering different numbers of robots (2 to 6 robots) while, in the second case, a more realistic scenario is considered, where the cleaning performance of robots are assessed considering an increasing number of moving clusters. In this first case study, we show how the learning performance of the proposed approach scales over the number of cleaning agents. The starting point of every robot in the heatmap is set at random, because in our study we want to find a solution that is independent by this initial condition. We designed a training process where, at the beginning of each episode, a random number of clusters is selected and each cluster is randomly positioned inside the station. Specifically, each obstacle-free location of the map has a 0.02 probability of generating a cluster. Each episode ends when agents successfully clean up to the 98% of the map or until a timeout is reached (400 steps are performed). During the training process we collect the overall reward as the sum of the single agents rewards and the number of steps needed to accomplish

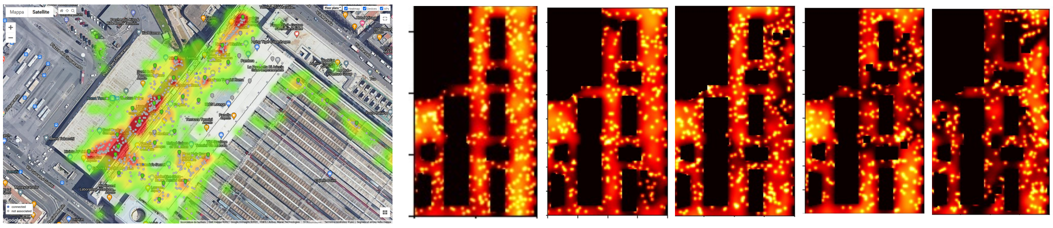

Figure 2: Example of the distribution of people inside the Termini station as retrieved from the Cisco Meraki WiFi network (left) and comparison of the 0 to 8 robot settings (5 pictures on the right) considering the simulated environment with 700 random dynamic clusters after 90 steps of execution.

the task. This setting is intentionally designed to train agents to address a generic distribution of priorities, which can be generated during daily cleaning processes. As for the execution time, the number of steps needed to accomplish the task, namely to clean the 98% of the map, decreases with the increasing number of agents. Specifically, the 2 agents configuration needs 174 steps on average to accomplish the task, while the 4, 6 and 8 agents ones need 127, 112, and 94 steps, with a time reduction of 27%, 12%, and 16%, respectively. Also in this case, the time reduction indicates that the proposed approach successfully scales to different number of robots. In order to assess the performance of the system into more realistic scenarios, we propose a different setting by considering different number of clusters and a simulated WiFi server that periodically updates the position of clusters at a specific rate (once every 15 steps). The numbers of clusters have been selected according to the average number of visitors-per-hour of the considered portion of the station (see Figure 2); moreover, during the runs, the values are designed to be randomly reduced up to the 30% in order to simulate the departure/arrival of passengers in the station.

4. Conclusions

In this work we proposed a scalable multi-robot sanitizing framework based on a distributed Deep Q-Learning technique, suitable for the efficient cleaning of large and crowded indoor environment such as railways stations. The proposed simulated experiments indicate that, as expected, the cleaning performance of the framework is proportional to the number of robots and inversely proportional to the number of people in the station. To asses the performance of our framework we proposed a worst-case test, where a large number of moving people is scattered (uniformly distributed) all around the station and robots should cover a wide area to perform the task. This setting is challenging compared to a real railway station, where people are often grouped near specific areas like shops, info points or ticket offices (see example in Figure 2, left) and robots can easily converge to those areas to maximize the sanitization effect. As future research activities, we plan to extend our pilot study by testing the proposed framework in a more realistic scenario, considering more complex robotic models and daily recorded data about the real people distribution in the station.

References

[1] J. S. Oh, Y. H. Choi, J. B. Park, Y. Zheng, Complete coverage navigation of cleaning robots using triangular-cell-based map, IEEE Transactions on Industrial Electronics 51 (2004) 718–726. doi:10.1109/TIE.2004.825197.

[2] X. Miao, J. Lee, B.-Y. Kang, Scalable coverage path planning for cleaning robots using rectangular map decomposition on large environments, IEEE Access 6 (2018) 38200–38215. doi:10.1109/ACCESS.2018.2853146.

[3] T.-K. Lee, S. Baek, S.-Y. Oh, Sector-based maximal online coverage of unknown environments for cleaning robots with limited sensing, Robotics and Autonomous Systems 59 (2011) 698–710. URL: https://www.sciencedirect.com/science/article/pii/S0921889011000893. doi:https://doi.org/10.1016/j.robot.2011.05.005.

[4] T.-K. Lee, S.-H. Baek, S.-Y. Oh, Y.-H. Choi, Complete coverage algorithm based on linked smooth spiral paths for mobile robots, in: 2010 11th International Conference on Control Automation Robotics Vision, 2010, pp. 609–614. doi:10.1109/ICARCV.2010.5707264.

[5] T.-K. Lee, S.-H. Baek, Y.-H. Choi, S.-Y. Oh, Smooth coverage path planning and control of mobile robots based on high-resolution grid map representation, Robotics and Autonomous Systems 59 (2011) 801–812. URL: https://www.sciencedirect.com/science/article/pii/S0921889011000996. doi:https://doi.org/10.1016/j.robot.2011.06.002.

[6] D. Wang, H. Deng, Z. Pan, Mrcdrl: Multi-robot coordination with deep reinforcement learning, Neurocomputing 406 (2020) 68–76. URL: https://www.sciencedirect.com/science/article/pii/S0925231220305932. doi:https://doi.org/10.1016/j.neucom.2020.04.028.

[7] S. Omidshafiei, J. Pazis, C. Amato, J. P. How, J. Vian, Deep decentralized multi-task multi-agent reinforcement learning under partial observability, in: D. Precup, Y. W. Teh (Eds.), Proceedings of the 34th International Conference on Machine Learning, volume 70 of *Proceedings of Machine Learning Research*, PMLR, 2017, pp. 2681–2690. URL: http://proceedings.mlr.press/v70/omidshafiei17a.html.

[8] G. Sartoretti, Y. Wu, W. Paivine, T. K. S. Kumar, S. Koenig, H. Choset, Distributed reinforcement learning for multi-robot decentralized collective construction, in: N. Correll, M. Schwager, M. Otte (Eds.), Distributed Autonomous Robotic Systems, Springer International Publishing, Cham, 2019, pp. 35–49.

[9] L. Setti, F. Passarini, G. De Gennaro, P. Barbieri, M. G. Perrone, M. Borelli, J. Palmisani, A. Di Gilio, P. Piscitelli, A. Miani, et al., Airborne transmission route of covid-19: Why 2 meters/6 feet of inter-personal distance could not be enough, 2020. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7215485/.

[10] T. Kim Geok, K. Zar Aung, M. Sandar Aung, M. Thu Soe, A. Abdaziz, C. Pao Liew, F. Hossain, C. P. Tso, W. H. Yong, Review of indoor positioning: Radio wave technology, Applied Sciences 11 (2021). URL: https://www.mdpi.com/2076-3417/11/1/279. doi:10.3390/app11010279.

[11] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, et al., Human-level control through deep reinforcement learning, 2015. URL: https://www.nature.com/articles/nature14236#article-info.