Multi-Robot Simultaneous Coverage and Mapping of Complex Scene
Laëtitia Matignon, Olivier Simonin

To cite this version:

HAL Id: hal-01774799
https://hal.science/hal-01774799
Submitted on 14 Sep 2018

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers. L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
Multi-Robot Simultaneous Coverage and Mapping of Complex Scene

Laetitia Matignon\textsuperscript{1,2} and Olivier Simonin\textsuperscript{1,3}

Abstract—In this demonstration, participants will explore a system for multi-robot observation of a complex scene involving the activity of a person. Mobile robots have to cooperate to find a position around the scene maximizing its coverage, i.e. allowing a complete view of the human skeleton. Simultaneously, they have to map the unknown environment around the scene.

We developed a simulator presented in this paper that allows to generate an environment, a scene, and to simulate robots’ observations and motion. During the demonstration, users will be able to test our simulator, including setting up a scenario and a demonstration workflow and its interactive aspects with the audience. The video accompanying our demonstration is available at https://liris.cnrs.fr/crome/wiki/doku.php?id=demoaamas2018.

In this demonstration paper, after a brief presentation of the proposed approach and techniques involved, we will focus on describing our simulator, the accompanying publication to this demonstration [5].

We also developed a simulator that allows to generate an environment and a scene, and to simulate robots’ observations and motion. By using this simulator, we were able to compare different algorithms for the simultaneous exploration of the environment and coverage of the scene. These results are detailed in the accompanying publication to this demonstration [5].

II. OUR APPROACH

Coverage task: To formalize the coverage task of a complex scene observation problem, we use the CMOMMT framework [1]. It aims to dynamically position robots to maximize the coverage of mobile targets. In our problem scenario, robots have to cover a human pose (skeleton) defined as a set of joints (targets). The observation vector of a robot is then defined as a binary vector, where each element is 1 if the robot is tracking the target or 0 otherwise. The individual (resp. joint) observation quality made by a robot (resp. a team) is the average number of joints tracked by the robot (resp. the team). To quantify the individual contribution of a robot to the joint observation, the contribution of a robot is the part of the observation that it is the only one to see.

Mapping task and circular topology: As the environment is unknown, robots must also build a map to locate themselves and learn observation qualities from different

\textsuperscript{1}Cooperative Multi-robot Observation of Multiple Moving Targets

\textsuperscript{2}Here it is defined as the number of targets under observation and the duration of observation of each target.
locations. So we extend the CMOMMT framework with a simultaneous mapping task. The map uses cells as a discrete representation of the robots’ positions. These cells arise from sectors and concentric circles around the scene where the robots are moving (cf. Fig. 1). This circular navigation topology is adapted to the continuous observation of a scene.

Incremental mapping: One issue of our coverage and mapping problem is the trade-off between exploitation and exploration, that is moving to optimize the coverage versus exploring the environment to find new interesting observation positions. To master the complexity of the state space and the time to explore it, we propose an incremental mapping based on a quadtree structure. The idea is to refine the discretization by splitting cells only in interesting areas i.e. where the quality of observation is promising. Fig. 2(a) shows a quadtree map constructed by robots. Two views are given to visualize the different map data: coverage data on the top map with cell qualities in shades of green (the greener the cell, the better is the quality from this cell; white is for cells where the scene is not visible, dark for obstacle cells); obstacle probabilities in shades of grey on the bottom map.

Decision algorithms: We proposed different algorithms to guide the robots exploration of the state space. They rely on local versus global information and lead to solutions with different computational and memory costs. Approaches based on local information and meta-heuristic optimizations obtained better results than the exhaustive exploration algorithm. For further details on this approach and its evaluation, you should refer to the accompanying publication [5].

III. SIMULATOR

We developed from scratch a simulator (cf. Fig. 2) with two goals: (i) to run a large quantity of experiments in order to compare our different algorithms, (ii) be realistic enough to properly model key features of real mobile robots, environments and scene. First we consider that robots’ motion around the scenes is perfect, i.e. robots can move along circles without trajectory errors, and robots are equipped with sensors allowing them to remotely detect nearby obstacles. Communications between robots are also supposed to be instant and errorless. We use a reference environment to simulate obstacles and observation of the scene from each cell. The robots do not have access to this environment. They build the quadtree map (cf. Fig. 2(a)) during their exploration. To generate this reference environment, one can choose where to put obstacles by loading a file or random ones according to a probability. To simulate a scene and assign observations to each cell, one can choose random observations or real skeleton data. They have been obtained from human pose captures with Kinect sensor and OpenNI/NITE skeletal tracking library [6]. These data have been imported in the simulator to generate the observation vectors (composed of 15 body joints) for each sector. Fig. 2(b) shows a reference environment with obstacles (black cells), occluded cells (white), and cells with different observation qualities (shades of green)\(^3\). We add noise to the perception from a cell to simulate camera sensor noise and occlusions by other robots\(^4\). So the observation perceived by a robot from a cell may vary from the reference environment. One important feature of our simulator is that its main parts (data structures (e.g. cell, quadtree), decision algorithms, interface) are used both in simulation and for real robots experimentation (with Turtlebot2 robots and ROS middleware) [4], [5].

IV. DEMONSTRATION

During the demonstration, participants of the conference will be given the opportunity to configure and observe examples of how a team of robots simultaneously map and cover a human pose. The sequence of the demonstration will be divided into two parts: the configuration of the scenario and the monitoring of the execution. First, the user can choose via the setup panel (cf. Fig. 2(d)) an environment, a human pose, the number of robots and their initial position, the noise degree, the algorithm and its parameters, ... Then, he can launch and control the robots’ execution via the control panel (cf. Fig. 2(d)). During this phase, the user can monitor the movement of the robots and the map they are building (cf. Fig. 2(a)), and have access to information specific to each robot (its current observation vector and its contribution) (cf. Fig. 2(c)). Some statistics concerning the team are also displayed (current joint quality, best joint quality found so far, ...). Finally, a video of our experiments with real robots (Turtlebot2) will be shown.

REFERENCES


\(^3\)The greener the cell, the better is the observation quality.

\(^4\)A noise parameter defines the probability for each value of an observation vector to be flipped compared to the reference values.