Active Cooperative Perception in Network Robot Systems Using POMDPs

Matthijs T.J. Spaan, Tiago S. Veiga and Pedro U. Lima

Abstract— **Network robot systems (NRS) provide many scientific and technological challenges, given that robots interact with each other as well as with sensors present in the environment to accomplish certain tasks. In this work, we consider an essential problem in NRS, namely how to perform task planning given the limitations both in on-board sensing as well as in the environment's sensors. Partially observable Markov decisions processes (POMDPs) form an attractive framework to address planning in the uncertain environments that typify NRS. We show how to model a typical cooperative perception task in a NRS, namely tracking and classifying people, and we present experiments that show how the proposed approach results in an effective interplay between robot and environment sensors.**

I. Introduction

The recently growing interest in Network Robot Systems (NRS) is driven by contributions from the sensor networks and the cooperative robotics communities. A NRS is usually thought to be a distributed system which consists of a multitude of networked environment sensors and actuators, including robots and possibly people. They cooperate among themselves to perform given tasks, and are capable of interacting with the environment through the use of perception and action [1]. In this work we present a formal approach for planning for robots in NRS, explicitly modeling the limitations and noise for each of the environment's sensors as well as the robots' on-board sensors. In particular, we will consider a tracking and classification task, in which a robot classifies a target in cooperation with a set of fixed cameras.

Sensor networks consist of a (usually large) number of sensor devices which measure their environment with the goal of obtaining an integrated view of an area larger than can be covered by a single sensor. In sensor networks typically no actuation is involved, while in NRS decision-making is explicitly handled, so as to select actions to be performed by network devices, robots, and/or people. Decision-making in NRS may range from dispatching mobile robots to detected events (for instance, a fire or an intruder) so as to help handling them, to improving the accuracy or resolution of the system's perception of a given event or activity. The latter is an instance of Active Cooperative Perception (ACP) [2], i.e., a scenario in which a (set of) robot(s) actively selects actions taking into account their effects on its sensors, in particular to improve system performance. Depending on the system's task, performance can for instance be expressed as accuracy or confidence degree in detection of an event.

In this paper we propose to handle ACP in NRS using decision-theoretic methods, in particular partially observable Markov decision processes (POMDPs) [3]. Such methods provide a means to quantify and predict results from probabilistic models of decision-making under uncertainty. In particular, POMDPs model imperfect sensors as well as uncertainty in action effects, both of which are important features when planning for robots in real-world scenarios. POMDPs provide a solid mathematical framework which can model different ACP problems within NRS, such as optimization of target tracking accuracy or optimal event detection (given event priority and detection uncertainty).

In previous work, we considered a POMDP-based approach to sensor selection in sensor networks [4], another type of decision-making under uncertainty problem. Here we focus on optimizing detection and classification of people, by combining a network of surveillance cameras with robot-mounted sensors. We discuss how POMDPs can be applied in such scenarios, and present a POMDP-based framework, which can serve as an example for many ACP tasks in NRS. The key benefit is that we can reason about beliefs over certain features in the environment, without leaving the classic POMDP framework. We present experiments in scenarios in which a robot has to classify one or more people, and which show that the proposed approach results in an effective interplay between environment and robot sensors.

Applications of decision-theoretic techniques to active sensing have been reported, although not always explicitly modeled as POMDPs, including methods for active robot localization using information gain [5], multi-modal sensor scheduling [6], and locating objects in large images of office environments [7]. Other related work using POMDPs has been presented: in [8] a POMDP framework for active sensing is described, in which the actions are using a particular sensor (with an associated cost) or outputting a classification label. In [9] a related setting is considered, but coupled with the training of Hidden Markov Model classifiers. However, only myopic solutions are under consideration. Our work applies non-myopic planning techniques, and considers real-world scenarios backed up by NRS experiments.

The paper is organized as follows: first Section II presents some background on POMDPs. In Section III we discuss POMDP models for active perception, and in Section IV we introduce the framework for tracking and classification. Results from several experiments are presented in Section V while Section VI concludes and discusses future work.

This work was funded by Fundação para a Ciência e a Tecnologia (ISR/IST pluriannual funding) through the PIDDAC Program funds as well as by project PTDC/EEA-ACR/73266/2006.

All authors are with the Institute for Systems and Robotics, Instituto Superior Técnico, Lisbon, Portugal. The work was performed while Pedro Lima was at Universidad Carlos III de Madrid, Leganés, Spain.

{mtjspaan, tsveiga, pal}@isr.ist.utl.pt

II. Background on POMDPs

We will briefly introduce the POMDP model, while a more elaborate description is provided in [3], for instance. A POMDP models the interaction of an agent with a stochastic and partially observable environment, and it provides a rich framework for acting optimally in such environments.

A POMDP assumes that at any time step the environment is in a state $s \in S$, the agent takes an action $a \in A$ and receives a reward $r(s, a)$ from the environment as a result of this action, while the environment switches to a new state s' according to a known stochastic transition model $p(s'|s, a)$. After transitioning to a new state, the agent perceives an observation $o \in O$, that may be conditional on its action, which provides information about the state s' through a known stochastic observation model $p(o|s', a)$. The agent's task is defined by the reward it receives at each time step t and its goal is to maximize its expected long-term reward $E[\sum_{t=0}^{h} \gamma^t r(s_t, a_t)]$, where h is the planning horizon, and γ is a discount rate, $0 \leq \gamma < 1$.

Given the transition and observation model the POMDP can be transformed to a belief-state MDP: the agent summarizes all information about its past using a belief vector $b(s)$. The initial state of the system is drawn from the initial belief b_0, and every time the agent takes an action a and observes o, its belief is updated by Bayes' rule:

$$b_a^o(s') = \frac{p(o|s', a)}{p(o|a, b)} \sum_{s \in S} p(s'|s, a) b(s), \tag{1}$$

where $p(o|a, b) = \sum_{s' \in S} p(o|s', a) \sum_{s \in S} p(s'|s, a) b(s)$ is a normalizing constant. Solving POMDPs optimally is hard, and thus algorithms that compute approximate solutions are used. Recent years have seen much progress in approximate POMDP solving which can be used in this paper, see for instance [10], [11], [12]. Furthermore, if a value function has been computed off-line, the on-line execution of the policy it implements is computationally cheap.

III. POMDP models for active perception

A belief update scheme is the backbone of many robot localization techniques, in which case the state is the robot's position. In our case however, the state will also be used to describe the location of people or events in the environment, as well as some of their properties. From each sensor we will need to extract a probabilistic sensor model to be plugged in the observation model. Furthermore, we need to construct the transition model based on the robot's available actions. Both models can either be defined by hand, or can be obtained using machine learning techniques, for instance [5].

From the perspective of active perception, as the belief is a probability distribution over the state space, it is natural to define the quality of information based on it. We could use the belief to define a measurement of the expected information gain when executing an action. For instance, a common technique is to compare the entropy of a belief b_t at time step t with the entropy of future beliefs, for instance at $t + 1$. If the entropy of a future belief b_{t+1} is lower than b_t, the robot has less uncertainty regarding the true state of the environment. Assuming that the observation models are correct (unbiased etc), this would mean we gained information. Given the models, we can predict the set of beliefs $\{b_{t+1}\}$ we could have at $t + 1$, conditional on the robot's action a. If we adjust the POMDP model to allow for reward models that define rewards based on beliefs instead of states, i.e., $r(b, a)$, we can define a reward model based on the belief entropy.

However, a reward model defined over beliefs significantly raises the complexity of planning, as the value function will no longer be piecewise linear and convex. Such a compact representation is being exploited by many optimal and approximate POMDP solvers. Instead, we opt to add classification actions to the problem definition, which allow for rewarding the system to obtain a certain level of knowledge regarding particular features of the environment. In this way, we achieve a similar objective as defining reward functions over belief entropy, while remaining in the classic POMDP framework.

IV. Tracking and classification with POMDPs

First we will detail our problem definition, followed by the POMDP models we use to tackle the problem.

A. Problem definition

We define our problem as creating a classifier system which decides whether a detected event is one of interest. For instance, in a surveillance task, an event of interest might be to detect a particular person using visual sensors, by running face detection algorithms. We consider an environment with mobile and fixed sensors, where mobile sensors increase the observability of the system at a particular location but, on the other hand, they have an associated cost of moving. Furthermore, robot-mounted sensors will take time to arrive a particular destination, and are a scarce resource: given a limited number of robots available, the system needs to choose carefully when to send a robot where. Given these circumstances, in order to successfully optimize its behavior, the system needs to consider the sequential decision making problem, while at the same time trading off benefits and costs. POMDP planners provide these features, allowing the system to plan ahead and anticipate the occurrence of future events.

B. Models

The POMDP model is represented as a two-stage dynamic Bayesian network, which allows us to use solvers that exploit (context-specific) independence between variables, for instance [11]. We will consider a scenario in which a NRS classifies visual features of a person, though our model is general, and can be adapted to other situations with the proper changes in the models. We consider discretized time and take decisions at predefined intervals. Fig. 1 depicts a graphical representation of the model for time steps t and $t + 1$.

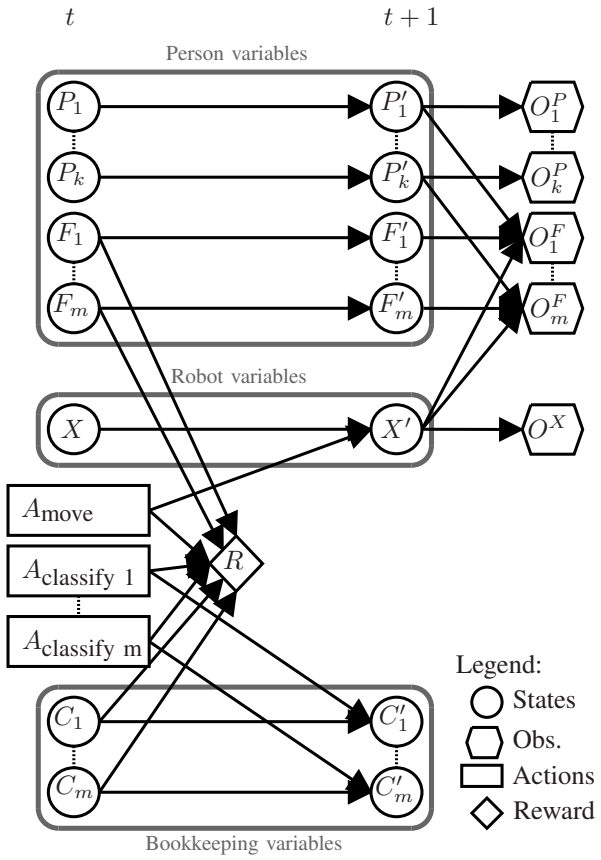


Fig. 1: Two-stage dynamic Bayesian network representation of the proposed POMDP model. In this figure for simplicity we assume $m \equiv k$, i.e., each person has only one feature.

1) States and transitions: To tackle this problem, we encode the environment in several state variables, depending on how many people and features the system needs to handle. The locations of people and robot are represented by a discretization of the environment, for instance a topological map. Graphs can be used to represent topological maps with stochastic transitions. Hence, we represent the variables describing the robot location X, k people locations P_1 through P_k with a set of nodes. Besides a person's location, we represent a set of m features, and each feature f is associated with a person, for instance whether it matches a visual feature, or another characteristic. We assume each person has at least one feature, hence $m \geq k$, and features are represented by variables F_1 through F_m. The feature variables intend to keep track of the belief over an event of interest being detected in the environment. They form an important part of the model as the belief over these variables gives the certainty of an event of interest to be happening. Finally, the state includes m bookkeeping variables, keeping track of which features have already been classified or not.

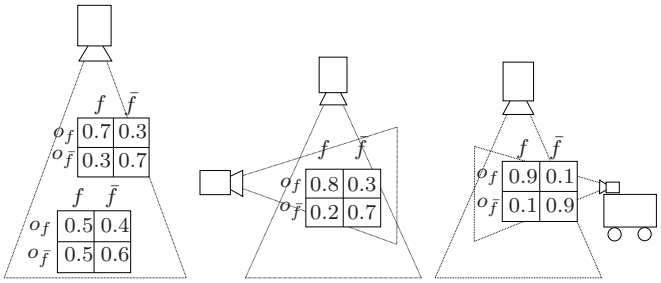
We assume a random motion pattern for each person, as we do not have prior knowledge on the person's intended path, in which the person can either stay in its current node, or move to neighboring ones. Such a representation allows us to model movement constraints posed by the environment (for instance, corridors, walls or other obstacles), which constrain a person's possible paths. We also need to take into account the uncertainty in the robot movement due to possible errors during navigation or some unexpected obstacles present in the environment. The value of the feature nodes $F_1 \dots F_m$ have a low probability to change, as it is unlikely that a particular person's characteristic changes. For the remainder, we assume binary features, without loss of generality.

2) Actions: Two types of actions are available to the system. First, actions with an effect in the physical world, that for instance move the robot (node A_{move} in Fig. 1) or adjust its sensors to help the system identify some event at a particular location. The second type of actions do not influence the environment, but instead announce that an event f has been classified (nodes $A_{\text{classify 1}}$ through $A_{\text{classify m}}$). Besides a reward signal, their only effect is to switch the corresponding C_f bookkeeping variable to *classified*. Initially, each C_f is initialized to *not yet classified*.

3) Observations and sensor models: For each state variable except the bookkeeping ones we define a set of observations. Each observation $o^p_1 \in O^P_1$ through $o^p_k \in O^P_k$ and $o^x \in O^X$ indicates an observation of a person or robot close to a corresponding $p_k \in P_K$ resp. $x \in X$. The observations $o^f \in O^F$ indicate whether a particular feature is observed.

The key for cooperative perception in this model lies in the observation model for detecting features. The false negative and false positive rates are different at each location, depending on conditions such as the position of sensors, their field of view, lighting, etc. For detecting certain features, mobile sensors have a higher accuracy than fixed sensors, although with a smaller field of view. Therefore, the observation model differs with respect to person and robot location. Typically uncertainty on observations is lower when a person is closer to the sensor (Fig. 2a), or when there are multiple sensors looking at its location, see Fig. 2b. In particular, if a person is observed by the robot we might have good observability (Fig. 2c), and the probability of false negatives $P(O^F = o^{\bar{f}} | F = f)$ or false positives $P(O^F = o^f | F = \bar{f})$ decreases in these cases. This is an important issue in decision making, as the presence of a mobile sensor will be more valuable in areas where fixed sensors cannot provide high accuracy.

4) Reward model: The reward model should encode the general goal for the problem, which is to classify an event as being of interest or not. A POMDP approach provides an appropriate way to tackle this. Since we include a state variable to track whether a feature F_f has a particular value, for instance whether it is present or not, the system maintains a belief $b_t(F_f)$ for this variable indicating how sure it is that the feature is present. This belief is updated upon observations, which depend on robot and person location. Our reward model is encoded in the *classify* f actions, one for each feature f. It depends on the corresponding F_f variable, assigning a reward $r_f > 0$ when it indicates an event of interest and a penalty $r_{\bar{f}} < 0$ otherwise. This guides the system to get a high certainty with respect to detected



(a) With 1 camera. (b) With 2 cameras. (c) Camera and robot.

Fig. 2: Observation model examples. Each matrix shows the observation probability for a single feature: $p(O^F|F)$.

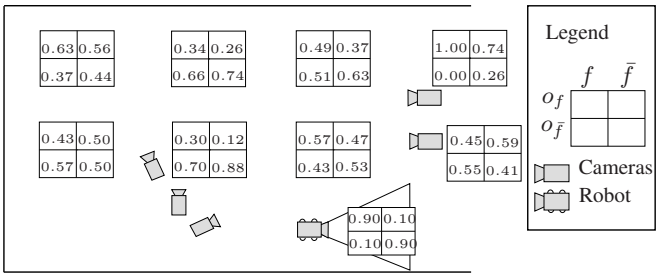


Fig. 3: Learned observation model for fixed cameras. Each matrix represents $p(O^F|F)$ and is positioned at a node in the map, except the one for the mobile camera, which shows the observation at any location if $X = P_i, 1 \leq i \leq k$.

features. Furthermore, we only reward the robot one time for classifying a feature, by using the bookkeeping variable C_f. Positive reward for classification is awarded only when C_f still has value *not yet classified*. This directs the robot to be sure about a feature before trying to classify it, as it can only do it once. How sure it should be before classifying is task specific, and is defined by the values assigned to r_f and $r_{\bar{f}}$. Also, different features can be valued differently in this way.

Another problem that this system can model, is the case when there are different priorities for different areas in the environment. Then, even if an area has a lower event detection uncertainty, the system might want the robot to check there first. This is modeled in the reward model by giving a higher reward for those areas of high importance. In this case, the reward model will depend explicitly also on the person location.

V. Experiments

The proposed model is applied to a real scenario, considering a surveillance task in which the event of interest is to detect people wearing a red shirt. We implement and test the system at our lab [13].

A. Experimental setup

The map of the area we use is discretized into a 8-node topological map, represented as a graph. The system is composed of a camera network mounted on the ceiling of our lab and a mobile robot (Pioneer 3-AT) with on-board camera and laser range finder. Each ceiling camera runs an adaptive background subtraction method for person localization and a simple color detection algorithm for shirt color. The robot-mounted camera runs only the color detection, and although it has a high accuracy, it only has a low range. Robot location is observed using a laser range finder, through an off-the-shelf implementation of Monte Carlo Localization.

Besides the classification actions, the robot four movement actions has available: *moveLeft, moveUp, moveRight, moveDown*, whose intended outcome of each action is for the robot to move towards the closest node in the respective direction. In this model, for each person we have a feature called *color*, with two possible values: *red* or *other*. The observation model for color is obtained by off-line tests in which we collect detection data of a person with a known shirt color walking randomly in the environment. This data is processed and summarized per node, giving the error rate and the uncertainty in color detection for each node. The resulting observation model is shown in Fig. 3, along position and direction of cameras available at the environment, and each matrix is positioned at one of the 8 nodes.

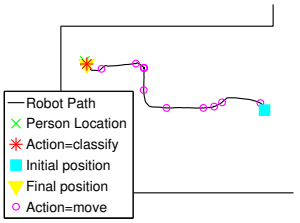
The POMDP controllers are computed using Symbolic Perseus [11], which is an approximate point-based POMDP solver, based on Perseus [10], but instead of using a flat POMDP representation, it exploits an Algebraic Decision Diagram (ADD) representation to tackle large factored POMDPs. We set a high discount rate, $\gamma = 0.99$. A reward $r_f = 10$ is given when the person is correctly classified, and $r_{\bar{f}} = -10$ if the classification is incorrect. Each movement is penalized with a reward of -0.1.

B. Experimental results

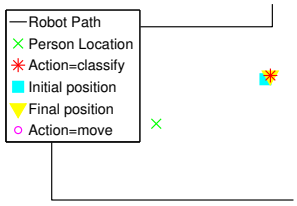
First, we consider an environment with only one person, and the belief is initially set to a uniform distribution. In Experiment A, the person is wearing a red shirt and is detected in a region where the uncertainty of red detection is high, see Fig. 3. Fig. 4 (left column) plots the robot's path until the person is classified (a), the cumulative reward awarded (c), and the belief $b_t(F_1 = \textit{red})$ over the color feature (e). However, the POMDP maintains a belief over all the state variables, and decides based on this joint belief.

Initially the robot is far from the person, but it moves to check on the person's shirt color, which is consistent with the evolution of belief. As the uncertainty is high, the belief increases slowly over time and the cumulative reward is decreasing as the robot is moving, which incurs a negative reward. Note the quick increase of the belief at time step 20, when the robot meets the person and has it in its field of view. Due to the low error in the robot's camera detections, $b_t(F_1 = \textit{red})$ increases rapidly, and the system classifies the detected person as in fact wearing a red shirt. After classification the system enters in an absorbing state, i.e., bookkeeping variable C_1 switches to *classified*. The robot will not move further since there is nothing left to classify in the environment and the cumulative reward stabilizes.

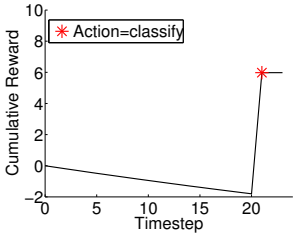
In Experiment B a person is detected in a low uncertainty area, on the other hand, see Fig. 4 (right column). In this



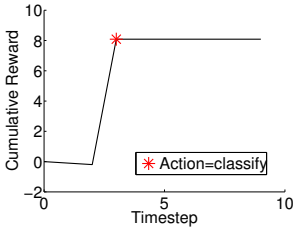
(a) Exp. A: Robot's path.



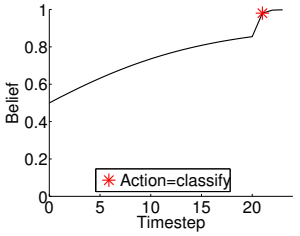
(b) Exp. B: Robot's path.



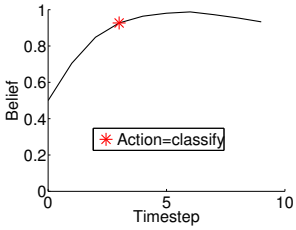
(c) Exp. A: $\sum_t \gamma^t r_t$.



(d) Exp. B: $\sum_t \gamma^t r_t$.

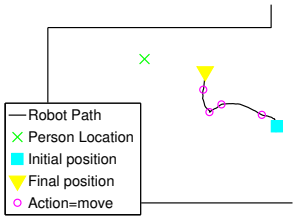


(e) Exp. A: Belief $b_t(F_1 = red)$.

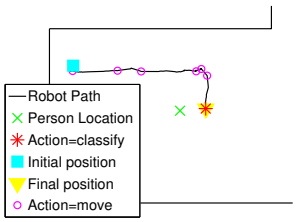


(f) Exp. B: Belief $b_t(F_1 = red)$.

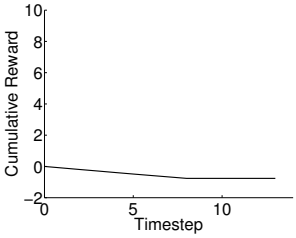
Fig. 4: Experiments with 1 person wearing red shirt.



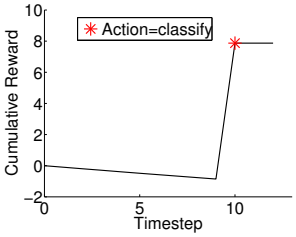
(a) Exp. C: Robot's path.



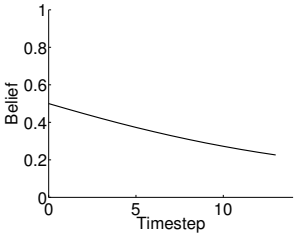
(b) Exp. D: Robot's path.



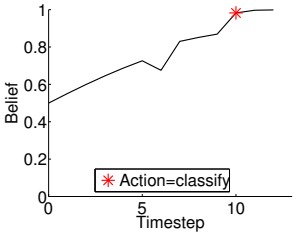
(c) Exp. C: $\sum_t \gamma^t r_t$.



(d) Exp. D: $\sum_t \gamma^t r_t$.



(e) Exp. C: Belief $b_t(F_1 = red)$.



(f) Exp. D: Belief $b_t(F_1 = red)$.

Fig. 5: Experiments with 1 person.

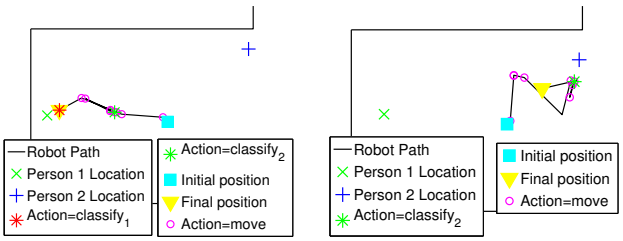
case, there is no need for the robot to move to check on the detected event. When considering the cumulative reward and belief figures, $b_t(F_1 = red)$ increases more rapidly than in Experiment A, which leads to a quicker classification and less collection of negative rewards, and consequently a higher cumulative reward. The differences between these first two experiments show the purpose of our classifier system. A mobile sensor (the robot in this case) will only spend its resources if the system really needs its information to make an informed decision whether to classify the event or not.

It is important to verify that the system is also accurate when detecting that no event took place. Therefore, Experiment C shows what happens when a person not wearing a red shirt is detected, see Fig. 5 (left column). A person has been detected in an area with some uncertainty, and as such, the belief decreases slowly as the system keeps receiving observations at every time step. In the meantime, the robot will follow a path to approach the person until it can observe the respective color or the observations received are consistent enough to decide on the classification.

Experiment D (Fig. 5, right column) shows the effects of noisy observations. This is a situation similar to Experiment A: a person detected in an area with high uncertainty, such that the robot needs to check it. Note, however, that the belief does not increase monotonously, since some false negative detections are received along with correct detections. This does not influence the system behavior, although reduces the belief at that particular time step, the robot continues to check at the person's location, and verifies that indeed, it should be classified.

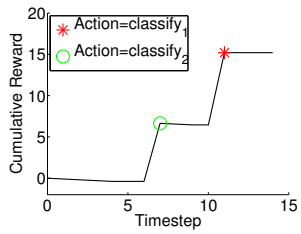
We also consider more complex scenarios, namely when instead of a single person we consider to have 2 persons. The system must reason whether to classify each one, and if the uncertainty is high and the robot needs to check on them, in which order to do so. In Experiment E (Fig. 6, left column) one person is detected in a low uncertainty area and the other one in a region where uncertainty is higher. Note that $b_t(F_2 = red)$ increases faster than $b_t(F_1 = red)$. Therefore, it is not necessary to check on person 2, but rather on person 1, and the robot navigates in its direction and uncertainty decreases when robot confirms the event at that location. In the meantime, the system has received enough information to also classify the other person.

So far, we have assumed that all locations and people have equal priorities. However, another interesting scenario is when we consider priority areas, that is, when one or more areas in the environment require special attention, and hence are more important. In Experiment F of Fig. 6 (right column), persons are detected in the same areas as Experiment E but each location has a different priority, as encoded in the reward function: $r_1 = 10$ and $r_2 = 20$. Although person 2 is in a low uncertainty area, the robot goes to check on it, as it is a more important area, and therefore will receive a higher reward.

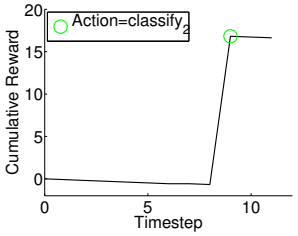


(a) Exp. E: Robot's path.

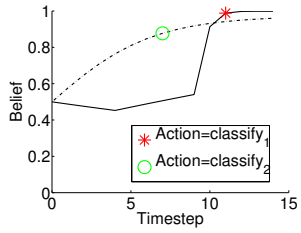
(b) Exp. F: Robot's path.



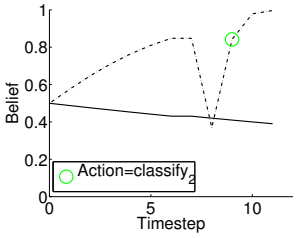
(c) Exp. E: $\sum_t \gamma^t r_t$.



(d) Exp. F: $\sum_t \gamma^t r_t$.



(e) Exp. E: $b_t(F_1 = red)$ (solid) and $b_t(F_2 = red)$ (dashed).



(f) Exp. F: $b_t(F_1 = red)$ (solid) and $b_t(F_2 = red)$ (dashed).

Fig. 6: Experiments with 2 persons.

VI. Conclusions

In this paper, we presented a POMDP approach to active cooperative perception in network robot systems. In NRS, decision making based on incomplete and noisy perception is often crucial for successful task completion. POMDPs offer a strong mathematical framework for sequential decision making under uncertainty, explicitly modeling the imperfect sensing and actuation capabilities of the overall system. In particular, we considered the problem of how a robot should act in order to track and classify a particular target, considering both its local sensors as well as sensors present in the environment. We tackled this problem by modeling movement as well as classification actions.

Classification actions allow us to consider reward functions that are linear in the belief state, and not for instance based on the entropy of the belief state. In the latter case, the POMDP is nonstandard, and the optimal value function is no longer linear. By defining a reward function over states, we remain in the standard POMDP setting, for which many results are known and successful approximate algorithms have been developed. Also, considering a discretized model allows us to compute closed-loop non-myopic solutions. The difficulty of solving continuous-state POMDPs in closed form has obstructed their solution, leading for instance to open-loop feedback controllers [14], or requiring additional model assumptions [15]. The experimental results illustrate that we can successfully trade off the costs of moving a robot vs. the desired level of confidence regarding an event. MDP-based heuristic solutions will not work in our scenarios, as they do not reason about future belief states, which is crucial for ACP. POMDPs provide a principled approach to integrating value of information with other costs or rewards, optimizing task performance directly.

Besides the particular task addressed in our experiments, many related NRS tasks can be cast in a similar framework. In future work, we would like to extend this work to model more types of ACP tasks, as well as scaling up to larger problems. Scalability can for instance be achieved by considering mixed-observability models [12], hierarchical models, as well as on-line optimization of a coarse off-line POMDP solution. Furthermore, instead of assuming a single robot, we could model multi-robot scenarios as well, either by including multiple robot variables in the model or assigning POMDP tasks to individual robots [16].

References

[1] A. Sanfeliu, N. Hagita, and A. Saffiotti, "Network robot systems," *Robotics and Autonomous Systems*, vol. 56, pp. 793–797, 2008.

[2] M. T. J. Spaan, "Cooperative active perception using POMDPs," in *AAAI 2008 Workshop on Advancements in POMDP Solvers*, July 2008.

[3] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra, "Planning and acting in partially observable stochastic domains," *Artificial Intelligence*, vol. 101, pp. 99–134, 1998.

[4] M. T. J. Spaan and P. U. Lima, "A decision-theoretic approach to dynamic sensor selection in camera networks," in *Int. Conf. on Automated Planning and Scheduling*, 2009, pp. 279–304.

[5] P. Stone, M. Sridharan, D. Stronger, G. Kuhlmann, N. Kohl, P. Fidelman, and N. K. Jong, "From pixels to multi-robot decision-making: A study in uncertainty," *Robotics and Autonomous Systems*, vol. 54, no. 11, pp. 933–943, 2006.

[6] S. Ji, R. Parr, and L. Carin, "Non-myopic multi-aspect sensing with partially observable Markov decision processes," *IEEE Trans. Signal Processing*, vol. 55, no. 6, pp. 2720–2730, 2007.

[7] J. Vogel and K. Murphy, "A non-myopic approach to visual search," in *Fourth Canadian Conference on Computer and Robot Vision*, 2007.

[8] A. Guo, "Decision-theoretic active sensing for autonomous agents," in *Proc. of the Int. Conf. on Computational Intelligence, Robotics and Autonomous Systems*, 2003.

[9] S. Ji and L. Carin, "Cost-sensitive feature acquisition and classification," *Pattern Recognition*, vol. 40, no. 5, pp. 1474–1485, 2007.

[10] M. T. J. Spaan and N. Vlassis, "Perseus: Randomized point-based value iteration for POMDPs," *Journal of Artificial Intelligence Research*, vol. 24, pp. 195–220, 2005.

[11] P. Poupart, "Exploiting structure to efficiently solve large scale partially observable Markov decision processes," Ph.D. dissertation, University of Toronto, 2005.

[12] D. Hsu, W. Lee, and N. Rong, "A point-based POMDP planner for target tracking," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2008.

[13] M. Barbosa, A. Bernardino, D. Figueira, J. Gaspar, N. Gonçalves, P. U. Lima, P. Moreno, A. Pahliani, J. Santos-Victor, M. T. J. Spaan, and J. Sequeira, "ISRobotNet: A testbed for sensor and robot network systems," in *Proc. of International Conference on Intelligent Robots and Systems*, 2009.

[14] S. S. Singh, N. Kantas, B.-N. Vo, A. Doucet, and R. J. Evans, "Simulation-based optimal sensor scheduling with application to observer trajectory planning," *Automatica*, vol. 43, no. 5, pp. 817–830, 2007.

[15] J. M. Porta, N. Vlassis, M. T. J. Spaan, and P. Poupart, "Point-based value iteration for continuous POMDPs," *Journal of Machine Learning Research*, vol. 7, pp. 2329–2367, Nov. 2006.

[16] M. T. J. Spaan, N. Gonçalves, and J. Sequeira, "Multirobot coordination by auctioning POMDPs," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2010.