Proactive Intention Recognition for Joint Human-Robot Search and Rescue Missions through Monte-Carlo Planning in POMDP Environments

Dimitri Ognibene[1] $^{[0000-0002-9454-680X]}$, Lorenzo Mirante[1], and Letizia Marchegiani[2] $^{[000-0001-6782-6657]}$

[1] School of Computer Science and Electronic Engineering, University of Essex, Colchester, UK
dimitri.ognibene@essex.ac.uk

[2] Department of Electronic Systems, Aalborg Univeristy, Aalborg, DK
lm@es.aau.dk

Abstract. Proactively perceiving others' intentions is a crucial skill to effectively interact in unstructured, dynamic and novel environments. This work proposes a first step towards embedding this skill in support robots for search and rescue missions. Predicting the responders' intentions, indeed, will enable exploration approaches which will identify and prioritise areas that are more relevant for the responder and, thus, for the task, leading to the development of safer, more robust and efficient joint exploration strategies. More specifically, this paper presents an active intention recognition paradigm to perceive, even under sensory constraints, not only the target's position but also the first responder's movements, which can provide information on his/her intentions (e.g. reaching the position where he/she expects the target to be). This mechanism is implemented by employing an extension of Monte-Carlo-based planning techniques for partially observable environments, where the reward function is augmented with an entropy reduction bonus. We test in simulation several configurations of reward augmentation, both information theoretic and not, as well as belief state approximations and obtain substantial improvements over the basic approach.

Keywords: Active Vision · Active Perception · Active Intention Recognition.

1 INTRODUCTION

Humans are endowed with sophisticated social interactions skills that provide invaluable advantages. These are realised by implicit intention reading, through the interpretation of partners' sensorimotor coupling with the environment, or explicit communication [22, 20]. The former modality can be advantageous for the 'acting' partners as it requires a lower cognitive load and allows focusing on the task at hand. However, the perceiving partner must gather and integrate information on several important factors like the presence, identity and configuration of any relevant affordances, as well as the pose of the partner and the trajectories of its effectors [18]. In complex and unstructured environments these skills have to heavily rely on proactive perception capabilities, as not all these elements may be immediately and simultaneously observable (*e.g.* being in another room, or occluded, out of the field of view or simply unattended), and require

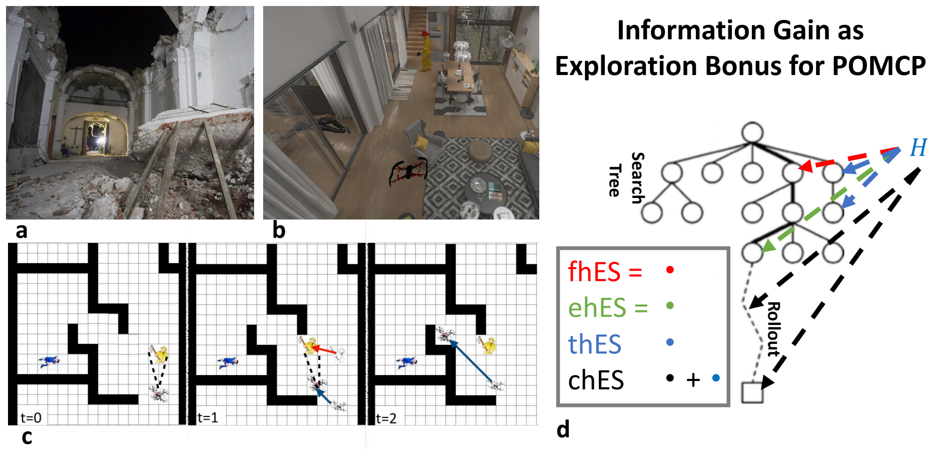

Fig. 1: **a)** A real disaster condition. **b)** A simulated environment view generated by AIRSIM [25] for testing our algorithm. The figure shows the drone, the responder and the target (*e.g.* a survivor in a disaster). The responder is supposed to be able to easily guess a good location where the target might be and slowly approach it. The drone enters the disaster zone from a different position, has initial knowledge of the entrance location of the responder and of a set of candidate areas where the target could be located. The drone, due to the complexity of the environment (*e.g.* occlusions), can see the target only when it is close to it. It must employ a policy that can take advantage of the presence of the responder (whose original position is known) to find the target as fast as possible. **c)** A cartoon-like illustration of an example scenario. *t=0)* the drone finds the first responder; *t=1)* the drone observes the responder moving towards the room where the survivor is; *t=2)* the drone anticipates the first responder and approaches the survivor room. **d)** Information gain as exploration bonus for POMCP. The arrows represent where in the MCTS search three is computed the entropy used for the exploration bonus.

multiple steps to be sensed. This is one of the main issues impeding human robot collaboration in such environments as dense sensorisation of the environment necessary for passive perception is not viable, and current active perception strategies are still limited to single step 'myopic' proactive perception strategies [19, 15]. An important application domain for this skill is search-and-rescue missions where both risk for the first responders and time spent in localising a survivor must be optimized. Yet, fully automating even the riskiest and earliest phases of these tasks is still hindered by limited autonomous exploration algorithms. Human responders, indeed, rely not only on the flexibility of their perceptual skills, but also on non-verbal task knowledge gained by exploring multiple environments, which is difficult to transfer to a robot.

In this work we propose the use of active intention recognition (IR) paradigms to allow the robot to indirectly and naturally exploit first responders' knowledge and, thus, to improve its exploration performance. The intuition is that the responders may be able to more accurately guess the position of the target and start moving to reach it. Their initial movements, together with some partial information on the map (*e.g.* information on regions of interest), can be enough to estimate where the responders are directed and, thus, anticipate them to test the location and secure the path. We carry out our analysis in simulation assuming the use of a drone as the search-and-rescue robot of reference.

2 Background

Search and rescue Robot disaster responders are an important research target [13]. They aim to quickly locate and extract survivors. This has driven the study of the non-verbal interaction strategies adopted by human responders [2] and the development of collaborative navigation algorithms for groups of robots [4]. In this line of research we try to bridge these problems by enabling robots to more naturally understand and account for responders' intentions while finding computationally affordable solutions.

Planning in partial observable conditions Planning tasks are traditionally represented and addressed relying on *Markov Decision Processes* (MDP), as being a framework which allows for a neat problem formulation, while also accounting for the presence of uncertainty and control noise. Most commonly, in MDPs, the task is defined as finding a *policy* that maximizes the *expected cumulative discounted reward* given a representation of the environment in terms of (i) a set of states and (ii) actions, (iii) a transition function describing the probability of reaching a certain state once an action is performed in a state, and a (iv) reward function describing the reward expected once an action is performed in a state. More realistic setups require to model noisy or missing sensory information. These problems are described extending MDPs to *Partially Observable MDP* (POMDP) [12], by providing an (v) observation function which describes the probability of obtaining a certain sensory input when the environment and the agent are in certain configurations. POMDP are particularly suitable to describe search-and-rescue environments, where the positions of the targets and the one of the responder are not necessarily known and they are detectable only when the robot is in their proximity. In this work, to approach these issues, we extend POMCPs, a Monte Carlo Tree Search (MCTS) algorithm for POMDP, which has been proposed in [26].

Intention Recognition (IR) IR supports natural, proactive collaboration between robots. Several algorithms have been proposed in the literature: some based on the idea of using pre-baked libraries of actions which suit a specific condition; others relying on the idea of inverse planning and the possibility of predicting the intentions of other agents, given a model of the environment which can be used to evaluate the effectiveness of a sequence of actions to achieve any possible goals [3, 24]. This kind of approach leads to a more precise and flexible recognition process, but has a higher computational cost. Other approaches, particularly suitable for robotics applications, instead, employ the idea of IR as an inverse control task, where multiple parametrizable controllers are compared to explain others' behaviours [19, 8].

Active Perception and Active Vision Active Perception is the problem of actively controlling sensors to improve the speed and accuracy in the estimation of behaviourally relevant variables [17, 27]. Its applications are numerous, spanning from inspection and localization, to object recognition and autonomous driving, and not only limited to robotics. Indeed, relevant efforts can be also be found in the neuroscience and cognitive science communities [10]. Different approaches have been proposed in the literature:

e.g. methods based on neural models often trained by Reinforcement Learning (RL) [17, 21, 27], evolutionary algorithms [7], and information theoretic approaches [9, 19].

Active Intention Recognition (AIR) Intention Recognition (IR) is a particularly challenging problem, as it needs to take into account the interactions among multiple elements (*e.g.* agents, affordances and effectors) which may not be known or observed simultaneously (*e.g.* while watching only the mouse running, we may not see the cheese, or not notice that a cat is running after the mouse). In unstructured and not instrumented environments, such as construction or disaster sites, or in general, when only few sensors with a limited receptive field are available, IR can be impossible to achieve if those sensors are not efficiently directed, so that they can perceive and track other agents, their effectors and the affordances they are interacting with. Active intention recognition (AIR) has been recently introduced as the problem of recognising the intentions of other agents, either humans or robots, when the sensing process can actively be controlled to deal with missing observations [19, 18] or limited bandwith [15]. AIR is formally defined as selecting the policy of sensor control which minimizes the final expected entropy on the state of the observed agent. Ognibene & Demiris in [19] presented an implementation of the concept on a humanoid robot, where IR is approached as an inverse control problem. The formulation used a mixture of Kalman filters to represent the robot's possible movements, assuming they were generated by one of multiple parametrizable linear controllers, and using a single-step myopic strategy based on a novel approximation of the information gain on the mixture selector variable. In [15] a similar method was employed to simultaneously recognize the complex activities of multiple actors, represented using Probabilistic Context Free Grammars.

In this work we extend [19] by: a) closing the loop between social perception and interaction through the integration of AIR and action execution, aimed at the shared goal of finding the target (survivor); b) relaxing the myopic constraint and considering multi-step policies.

3 Methods

Environment The problem is described according to the POMDP formalism through transition, observation and reward functions. A termination function is also used, as adopted in other POMDP problems with Monte Carlo (MC) methods.

Transition function The environment is static apart from the responder's movements. The transition function is, thus, defined by using a simple, deterministic, model of the drone's movements and a probabilistic model of the movements of the responder when searching and approaching the target. The state is composed of a 6 dimensional tuple, containing the position of the drone, the position of the responder and the one of the target. The responder either stays still with probability equal to p_{still} (to simulate a slower speed compared to the drone) either moves one step in the direction of the survivor location with $0.95(1 - p_{still})$ or in random direction.

Observation model The observation model always provides the position of the drone without noise. The positions of the target and of the responder, instead, are available when they are in the same position of the drone.

Reward function The main objective for the system is to find the goal, which is the same objective of the responder. A reward equal to 1 is given when the drone finds the goal. AIR is auxiliary to the main collaborative objective and, thus, rewarding social interaction directly is not part of the problem description. Yet, as explained in the algorithm subsection, several forms of intrinsic reward [16] are employed to address several algorithmic limits in exploring large environments and in exploiting the information provided by the responder's movements.

End function End function stops both the real world simulation and the simulation inside the MC sampling algorithm when the drone is near the target.

Planner The controller derives from the POMCP algorithm [26]. It integrates the planning with the state estimation processes by using the Monte Carlo simulation sampling to generate samples for the particle filter. POMCP has improved state-of-the-art performance in several tasks, and it has been already applied to social robotics in several instances [11]. POMCP uses a black-box generative model of the environment to estimate the action-observation sequences' values through repeated sampling. Each sampling iteration starts from a root node associated with the current sequence of observations h integrated in the belief state $b(h)$ with the initial 'prior' belief b_0. Then, POMCP builds a limited portion of the tree of the future actions and observations through the following steps: 1) an initial state is sampled from the current belief state $b(h)$; 2) an action a is selected according to an action-selection strategy that balances exploration and exploitation; 3) an observation o is obtained defining a new history $h' = [h, a, o]$, the corresponding node is evaluated recursively if it is already in the search tree (going back to step 2), or is inserted into the search tree after getting a first estimate of its value by continuing the simulation using a predefined 'rollout' action selection policy until a certain depth d; and 4) update the statistics of the tree nodes by back-propagating the simulation results up to the root.

Modifications of POMCP ***I. Exploration Reward*** The default exploration strategy (**dES**) of POMCP algorithm [26] relies on the Upper Confidence Bound (UCB) exploration bonus [1] in action selection. In environments where most of the rewarding/punishing states are deeper in the search tree than maximum search depth applying MCTS planning based only on UCB for exploration can be problematic. In these contexts, MCTS algorithms may fail to estimate the value of observing informative cues and even prefer useless actions, as staying still at the entrance, because they never simulate long enough to reach the reward and, thus, fail to plan and observe such cues. In this specific task, the informative cues are the responders' movements, which inform on their intentions and goals [19, 24, 3] but can be noisy and expensive to acquire. Trying to identify those intentions requires finding and following the responder, which can be quite inefficient in several cases, *e.g.* when the drone is certain of the goal position or near to a candidate position, and approaching it directly would be faster than relying on the responder. Thus, while in previous works involving AIR [19, 15] the expected information gain on the others' intentions was driving the robot behaviour, here *dealing with the trade-off between acquiring information about the responder and exploring autonomously* becomes an important part of the task. To address these issues, five other specific exploration strategies are tested that use an ‘intrinsic’ reward [16, 5] to facilitate not only visiting new useful positions, but also the extraction of information from the human rescuer’s presence: **1.** the **r**esponde**r** observation reward (**rrES**), where a reward of 0.1 is added when the robot observes the responder; **2. c**omplete entropy (**chES**), at each step (both tree search and rollout) an intrinsic reward is added equal to $-0.2 \cdot H$ where H is the entropy (see fig. 1.d); **3.** search **t**ree entropy (**thES**), like chES, but entropy is added only inside the search tree but not in the rollout avoiding the update of the filter; **4. e**nd search tree entropy (**ehES**), like shES, but entropy bonus is added only at the last step of the tree search, before the rollout phase. This makes the amount of bonus added independent from the number of search steps, factor that biases the other approaches; **5. f**irst step search tree entropy (**fhES**), where the bonus is added only after the first step of search in the tree. The effect of computing entropy only on the goal location (**gH**) like in [19], or on all the complete belief state (**bH**) is also tested for all the entropy based exploration strategies. This last variation exploits the structured form of the belief space containing the drone, the rescuer and the goal’s location. This is motivated by the low variability of goals compared with the trajectories followed by the human responder to achieve them. Such variability induces a high entropy that may affect the planning algorithm. Entropy is directly computed from the belief state. Inside the search tree, filter and entropy are cached to speed up the algorithm.

We must note that using information gain instead of the reward for helper observation may be more computationally expensive but it avoids suboptimal behaviours, such as following the slower responder after the target location is known. Furthermore, the complex computations for information gain used in [19] are not necessary anymore as an expectation of the information gain will be provided by the backup phase of MCTS.

II. Rollout Policy Together with a basic random rollout policy (**rRS**) that simply randomly samples actions during the rollout phase, five rollout stategies are developed which exploit additional task knowledge to reach a possible survivor position during the rollout. They differ in the way this position is selected: 1) **s**ample **t**arget **r**ollout **s**trategy (**stRS**), which selects the survivor position of the current Monte-Carlo sample state [3] as the target position for the robot movement during the rollout; 2) **d**eterministic **n**ear target (**dnRS**), which selects the nearest survivor position between the not visited ones as the objective of the robot’s movement in the rollout; 3) **s**tochastic **n**ear target (**snRS**), which selects one of the not visited targets as the target of the robot’s movement with a probability inversely proportional to the distance from the drone; 4) **d**eterministic most **p**robable target (**dpRS**), which selects the not visited target which has the highest probability of containing the goal. This is based on the filter estimation computed during the simulation phase; 5) **s**tochastic most **p**robable target (**spRS**), which acts similarly to dpRS, but samples stochastically a not visited survivor location according to its probability of being the goal location. For all these five strategies, a new target is selected when the previous one is reached without finding the goal. Two policies for the actual *action selection* are adopted in combination with the above strategies for target selection: I. best rollout action (**bRA**), which deterministically select the best action to reach the selected target, and II. stochastic rollout action (**sRA**), which selects an action stochastically with a probability inversely proportional to the distance between the

[3] POMCP samples a state from the initial belief state at the beginning of each iteration

reached state and the selected target. To implement efficiently these modifications, the best actions to reach each target are precomputed.

III. State estimation and reconstruction While the algorithm employed is inspired by the POMCP, it does not use a particle filter, instead it uses a **c**omplete discrete Bayesian **f**ilter **(cF)**. This helps deal with the limited amount of information the robot has access to during the task and which may strongly affect the particle filter performance. The cF allows also for effective entropy computation as well, but still presents some limitations (for more details, see [14, 6]). To decrease its computational cost we implement a truncated filter (**aF**), which, after each update, only maintains the $N_s = 20$ states with the highest probability and zeroes the others[4]. Given that the movements are localized, *i.e.* only adjacent cells can be accessed from another cell, the filter update cost is limited. However, this approximation may lead to observation conditions which are wrongly estimated as impossible, *i.e.* to have zero probability (similarly to the particles impoverishment in particle filters). This happens when the robot either *i.* focuses all the probability on solely one target and finds it empty, or *ii.* observes the first responder in an unexpected position. In these conditions, a new belief state is re-sampled as follows:

I. when all the probability is focused on one false target, the belief is regenerated by simulating the first responder's behaviour starting from its last observation at time t_o and moving towards each of the non visited target locations l_g. The same probability $p_g(t_o)$, which was assigned to the corresponding targets at time t_o, is then assigned to each of these generated states x_g . If the first responder was never observed, then the behaviour is simulated from each possible starting condition l_s.

II. When the first responder is observed in a non-expected position l^*, a new element of the state x_g is generated for each of the non explored targets l_g. In this case the probability associated with that state is computed according to how efficient it is to visit l^* for the first responder from its previous location l_{old} before reaching the target location l_g, versus travelling directly towards l_g [23, 24]. Thus $p(x_g) = p_g(t_o) \cdot c(l_{old}, l_g)/[c(l_{old}, l^*) + c(l^*, l_g)]$ where the $c(a, b)$ represents the cost of reaching b from a while following the optimal trajectory and its value is the same used to drive the rollout process.

Current implementation uses Python 3.6 and will be soon available online.

4 Results

Environments We consider four environments: a small and a big squared environment, a cross-like environment, a builiding like maze as shown in Fig. 1 and big randomly generated environments of size 64x64 containing 6 rooms connected by corridors. The small environment **[SE]** is a 5×5 grid. The drone is initially positioned in the center and the responder is in one of 2 adjacent cells, but the drone does not know in which one of them. The target is positioned in one of the 4 corners. In total 8 equally probable initial conditions are possible, representing the initial belief state of the robot in this environment. The large environment **[LE]** is an 11×11 grid. Again the drone is initially positioned in the center and the responder is in one of 4 adjacent cells, but the drone is

[4] Entropy is computed before this zeroing process

not informed of which of them. The target is positioned in one of the 16 cells in the 4 corners, 4 cell in square per corner. In total 64 equally probable initial conditions are possible, representing the initial belief state of the robot in this environment. The cross-like environment **[CE]** is not convex and, thus, more demanding. It is an 11×11 grid with the arms of the cross 5 cells wide where the drone and the responder start from same location, as in condition LE. The goals are located in the corners of the ends of the cross arms. For the building-like environment we used a grid 15×15 with 4 rooms, each containing two possible target locations.

The result for the small environment with 1000 samples and max-depth-search=14 (average on 100 trials) are reported in Table 1.

Table 1: Performance with 1000 and 100 MC samples.

	1000 Samples				100 Samples			
	chES	rrES	thES	dfES	chES	rrES	thES	dfES
Success	0.85	0.88	0.54	**0.95**	**0.8**	0.63	0.25	0.63
Steps	7.96	8.43	1.033	**7.91**	**8.31**	9.12	11.58	9.74
RObs	31	37	148	10	20	24	29	13

Average behavior over 100 trials with 1000 MC samples for each step. **Success** is the average number of times the drone found the target before 16 steps. In the table, **Steps** is the average number of steps necessary to the drone to find the target (or 16 if it fails). RObs total number of times the drone met the responder before finding the target. Tests performed on the following variations of the architecture: **rrES** reward was given also when observing the responder (0.1) other than the target (1); **chES** reward contained an entropy bonus (-H*0.2); **thES** reward contained an entropy bonus only during the search not during rollout (-H*0.2); **dfES** Default, reward was given only reaching the target.

When computational resources are available the **dfES** default algorithm, not using any exploration bonus, gave the best performance. In this case, we must note that it performs better the trivial solution of navigating through the four corners which would have an expected number of steps (2+7+11+15)/4=8.75. This is true also for the chES and rrES. chES and rrES also observe the responder much more often than dfES; yet dfES performs better than them¿. This implies that dfES makes good use of the observations of the responder (both when it finds the responder in a cell and when it doesn't). We observe, without a statistical analysis, that the policy usually chosen by the dfES robot often consisted in (a) reaching a corner, (b) going back to the center if no target is observed, and (c) going to the next corner. This strategy may maximize the probability of encountering the responder, while not meeting it would decrease the probability of the target to be in that direction. Unexpectedly, the thES version of the system presents lower performance. We attribute this to the dependence of the exploration bonus on the length of the search tree branch.

Table 1 reports also the results when only MC 100 samples per step are performed. In this case the chES system is the most effective one, followed by rrES and dfES. chES is the least affected by the reduced number of samples. This suggest that using an entropy-based bonus may support exploration of environments when not enough time is available for construction of deep search tree.

With the big environment and 100 samples per action the best performance are achieved by the chES, which, in 50% of the trials, found the target in less than 40 steps (maximum steps allowed). thES achieved (28%), rrES 27 % and dfES 18%.

In Table 2 the results for the building like environment are reported when simulations are run for 100 MC samples. The results consistently favour the **ehES**, while the

thES consistently show the worst performance. The performance of the best **dfES** are reported and are less effective than the configurations using **ehES**. The approximation does not appear to affect the performance, while yielding a substantial speed-up in the computation. Similar results have been obtained with the randomly generated bigger environments, but could not be reported for space reasons.

Table 2: Performance with 100 MC samples in building like environment.

Approx	Rollout Pol	Rollout Act	Expl Strategy	H	steps	time	reward	reward/time
FALSE	snRS	bRA	ehES	gH	1959	1483	76	0.051
TRUE	spRS	bRA	ehES	bH	1962	258	74	0.287
TRUE	dpRS	sRA	ehES	bH	2029	291	74	0.254
TRUE	dpRS	sRA	ehES	bH	2120	307	74	0.241
FALSE	dpRS	sRA	ehES	gH	2058	1583	74	0.047
FALSE	stRS	sRA	dfES	-	2148	331	69	0.208
TRUE	stRS	bRA	thES	bH	2431	344	42	0.122
TRUE	dnRS	sRA	thES	bH	2422	298	41	0.138
FALSE	stRS	bRA	thES	bH	2557	2006	40	0.020
FALSE	dnRS	sRA	thES	bH	2650	2026	34	0.017

Performance over 100 trials on the cross-like environment of the first best five, the most performing system without using entropy and the worst four configurations. No data collected for the **chES** due to its computational demands. **Approx** column define if the approximated belief state **aF** (True) or the complete **cF** one are tracked. **Rollout Pol** shows the rollout policy used. **Rollout Act** shows the Rollout action selection strategy used. **Expl Strategy** shows the Exploration Strategy employed **H** shows which entropy function was used to compute the exploration bonus, **steps** reports the number of steps in 100 trials, **time** computation time, **reward** total reward obtained and **reward/time** total reward over time, representing computational efficiency.

5 Conclusions and future work

Current result suggested that the idea of coupling a robot with a more expert but constrained human partner in the task of joint search for a target in an unknown environment may be promising. Future work will focus on optimizing the methods proposed, by integrating them with flexible SLAM algorithms, devising and testing alternative exploration bonus, employing realistic responders' models, considering additional collaborative tasks, such as verifying that the surrounding of the responder are safe.

References

1. Auer, P., Cesa-Bianchi, N., Fischer, P.: Finite-time analysis of the multiarmed bandit problem. Machine learning **47**(2-3), 235–256 (2002)
2. Bacim, F., Ragan, E.D., Stinson, C., et al.: Collaborative navigation in virtual search and rescue. In: 3D User Interfaces (3DUI), 2012 IEEE Symposium on. pp. 187–188. IEEE (2012)
3. Baker, C.L., Jara-Ettinger, J., Saxe, R., Tenenbaum, J.B.: Rational quantitative attribution of beliefs, desires and percepts in human mentalizing. Nature Human Behaviour (2017)
4. Beck, Z., Teacy, L., Rogers, A., Jennings, N.R.: Online planning for collaborative search and rescue by heterogeneous robot teams. In: AAMAS 2016 (2016)
5. Bellemare, M.G., Srinivasan, S., Ostrovski, G., Schaul, T., Saxton, D., Munos, R.: Unifying count-based exploration and intrinsic motivation. arXiv preprint arXiv:1606.01868 (Jun 2016)

6. Boers, Y., Driessen, H., Bagchi, A., Mandal, P.: Particle filter based entropy. In: Information Fusion (FUSION), 2010 13th Conference on. pp. 1–8. IEEE (2010)
7. de Croon, G.: Adaptive Active Vision. Ph.D. thesis, Universiteit Maastricht (2008)
8. Demiris, Y.: Prediction of intent in robotics and multi-agent systems. Cog Proc **8**(3) (2007)
9. Denzler, J., Brown, C.: Information Theoretic Sensor Data Selection for Active Object Recognition and State Estimation. IEEE TPAMI **24**(2), 145–157 (2002)
10. Friston, K., Rigoli, F., Ognibene, D., Mathys, C., Fitzgerald, T., Pezzulo, G.: Active inference and epistemic value. Cognitive neuroscience **6**, 1–28 (2015)
11. Goldhoorn, A., Garrell, A., Alquezar, R., Sanfeliu, A.: Continuous real time pomcp to find-and-follow people by a humanoid service robot. In: Humanoids 2014. IEEE (2014)
12. Kaelbling, L., Littman, M., Cassandra, A.: Planning and acting in partially observable stochastic domains. Artificial Intelligence **101**(1), 99–134 (1998)
13. Kruijff, T., Linder, P., Gianni, P., Pizzoli, S., Pianese, C.: Rescue robots at earthquake-hit mirandola, italy: a field report. In: IEEE Safety, Security, and Rescue Robotics (2012)
14. Lauri, M., Ritala, R.: Planning for robotic exploration based on forward simulation. Robotics and Autonomous Systems **83**, 15–31 (2016)
15. Lee, K., Ognibene, D., Chang, H.J., Kim, T.K., Demiris, Y.: Stare: Spatio-temporal attention relocation for multiple structured activities detection. Image Processing, IEEE Transactions on **24**(12), 5916–5927 (2015)
16. Mirolli, G.B..M. (ed.): Intrinsically Motivated Learning in Natural and Artificial Systems. springer (2014)
17. Ognibene, D., Baldassare, G.: Ecological active vision: Four bioinspired principles to integrate bottom–up and adaptive top–down attention tested with a simple camera-arm robot. Autonomous Mental Development, IEEE Transactions on **7**(1), 3–25 (2015)
18. Ognibene, D., Chinellato, E., Sarabia, M., Demiris, Y.: Contextual action recognition and target localization with an active allocation of attention on a humanoid robot. Bioinspiration & biomimetics **8**(3), 035002 (2013)
19. Ognibene, D., Demiris, Y.: Towards active event recognition. In: IJCAI 2013 (2013)
20. Ognibene, D., Giglia, G., Marchegiani, L., Rudrauf, D.: Implicit perception simplicity and explicit perception complexity in sensorimotor communication. comment on" the body talks: Sensorimotor communication and its brain and kinematic signatures" by g. pezzulo et al. Physics of life reviews **28**, 36–38 (2019)
21. Paletta, L., Fritz, G., Seifert, C.: Q-learning of sequential attention for visual object recognition from informative local descriptors. In: Proc ICML 2005. p. 656 (2005)
22. Pezzulo, G., Donnarumma, F., Dindo, H., et al.: The body talks: Sensorimotor communication and its brain and kinematic signatures. Physics of life reviews **28**, 1–21 (2019)
23. Ramirez, M., Geffner, H.: Goal recognition over pomdps: Inferring the intention of a pomdp agent. In: IJCAI. Barcelona (2011)
24. Ramirez, M., Geffner, H.: Plan recognition as planning. In: IJCAI (2009)
25. Shah, S., Dey, D., Lovett, C., Kapoor, A.: Airsim: High-fidelity visual and physical simulation for autonomous vehicles. In: Field and Service Robotics (2017)
26. Silver, D., Veness, J.: Monte-carlo planning in large pomdps. In: 24th Advances in Neural Information Processing Systems,(NIPS 2010). pp. 2164–2172 (2010)
27. Sprague, N., Ballard, D.: Eye movements for reward maximization. In: Advances in Neural Information Processing Systems 16. Cambridge, MA (2004)