2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 25-29, 2020, Las Vegas, NV, USA (Virtual)

Toward Detecting Anomalies in Activities for Daily Living with a Mobile Robot using Plan Recognition

Jean Massardi[1] and Éric Beaudry[2]

***Abstract*— In order to provide help to elderly people with cognitive impairment, a robot assistant must understand two things, what the person wants to perform, and if the person requires cognitive help to perform it. Approaches from Plan Recognition can be options to address both of these problems. In this article, we propose to use the particle filter approach of plan recognition over plan libraries to detect anomalies in activities. These anomalies can be used to detect cognitive distress. We use the Expected Next Observation compute by this approach to determine how likely the observation of an action is. If this likelihood is beneath a threshold we consider that this is an anomaly. Empirical results on simulated problems show that this approach is effective in case of perfect observability or low level of noisy observations, but the approach is still too sensitive to noisy observation for real-life applications on mobile robots.**

I. INTRODUCTION

In most western countries, the aging of the population has become an important healthcare concern. There is an increasing need to develop new solutions to help elderly people to stay at home with more autonomy to both improve their quality of life and to reduce pressure on caregivers [1]. Several mobile robots designed to help at home have been developed in this intent like ACCOMPANY [2], Giraff [3] or SAM [4]. Rather than proposing physical assistance, some of these robots like Giraff and SAM focus on two aspects : (1) provide telepresence support for the elderly and (2) provide some forms of cognitive assistance.

A way to provide cognitive assistance is to help persons in achieving their Activities for Daily Living (ADL) by monitoring their actions and by ensuring every one of them has been done correctly. In order to do that, the robot needs to understand what the person is currently doing and what the person aims to achieve in a foreseeable future. These problems correspond to an activity and a goal recognition problem. Plan recognition is the opposite task of planning. In planning, the objective is to produce a plan composed of low-level actions in order to achieve a goal. With plan recognition, the objective is to infer the plan or the goal of an observed agent based on the observation of its actions [5]. Plan recognition have already been proposed and effectively used to help in assisting elderly people [6] [7]

Plan and activity recognition are prerequisites to obtain a cognitive assistant. A second step would be to detect when a person is doing an unexpected action. These unexpected actions are good indicators that a person needs cognitive assistance. This second problem is called anomaly detection [8]. While there is a lot of work on anomaly detection using approaches from activity recognition, there are only a few from a plan recognition point of view. Rather than focusing on the execution of single actions, a plan recognition approach would focus on the sequence of actions. In most cases, cognitive distress impacts the sequence of actions, for example when someone forgets his current objective, therefore, the use of the plan recognition offers interesting perspectives for anomaly detection for elderly care. Recent advances in plan recognition techniques, such as PARC (Plan and Activity Recognition Component) [9], can already be used on mobile robots for real-time plan recognition. Interestingly, PARC provides the expected next actions, which can be used both to detect the anomalies and to provide help by proposing the most probable goal with expected next actions.

In this article we propose a way to adapt the particle filter approach on plan libraries which is used in PARC to detect anomalies in ADL by using the prior probability of an action, also call Expected Next Observations (ENO). If an observed agent performs an action which the probability is lower than a predetermined threshold, the system considers this action unlikely and will suppose it is an anomaly.

We show that this approach is effective on simulated environments in case of perfect observability settings and in low noise settings, but the accuracy drops significantly in case of low to medium noise settings ($<$20 % of noisy observations). This drop makes the approach not suitable for real-life applications in knowing the state of the art of activity recognition on mobile robots, but give interesting perspectives if the accuracy of low-level activity recognition improves in the near future.

This paper is organized as follows. Section II describes previous approaches on anomalies detection in a sequence of actions. Section III describes PARC and the particle filter approach of library based plan recognition. Section IV presents how PARC can be adapted to detect anomalies. Section V shows experimental results. A conclusion follows.

II. RELATED WORKS

Anomalies detection can be described as the following : knowing a sequence of observations $O = \{o_1, o_2, ..., o_{t-1}\}$, does the observation o_t is normal, i.e was expected, or not. This can be expressed by the following equation.

$$P(o_t|O) < \sigma \quad (1)$$

*This work was not supported by AGEWELL-NCE

[1]Jean Massardi is with the Faculty of Sciences, University of Québec at Montreal, 201 Avenue du Président-Kennedy, Montréal, Canada massardi.jean@courrier.uqam.ca

[2]Éric Beaudry is with the Faculty of Sciences, University of Québec at Montreal, 201 Avenue du Président-Kennedy, Montréal, Canada beaudry.eric@uqam.ca

Here σ is the threshold that corresponds to the limit where an observation is considered as an anomaly.

There is a lot of work on anomaly detection [8] with several applications such as intrusion detection, security systems, or anomaly in ADL detection [10]. Most of these works focus on an activity recognition perspective. They use techniques directly from this domain to address the problem [11] [10]. One of the most effective techniques is the use of Deep-learning on the observation of sequences of actions. An efficient way to characterize action to discover anomalies is to use a temporal description of the actions. These temporal descriptions are linked to two key elements : (1) the duration of an action and (2) the position of an action in a sequence of other actions. In their works [11], the authors go further and use the context of an action. In this context, the position of an action in the sequence is a key element. This particular subproblem of the position in the sequence can be addressed using a plan recognition perspective since this domain focus on the plan, i.e the sequence of actions, and not on the action by themselves.

One of the early works using plan recognition to address an anomaly detection problem comes from [12], where the authors applied plan recognition to the goal abandonment problem. The goal abandonment problem consists in inferring if an observed agent is still following its goal or if its goal has changed. This approach is based on plan libraries, a collection of goals, actions, and rules [13]. The authors use these plan libraries to compute the probability of an action being part of the current plan. If this probability is below a threshold, the system infers an abandonment of the current goal. This approach uses several hypotheses, some of the most important being an equiprobability of the possible actions given a goal, perfect observability of the actions, and the certainty of the goal prior to the goal abandonment detection.

Another work linking plan recognition to anomaly detection is the commitment abandonment detection [14]. In their work, the authors detect if an action is sub-optimal to reach a goal. If the action is, then the system infers that the observed agent as shifted from its goal. This approach uses planning domains as a way to infer plans. It is also designed mostly for cooperative tasks, but can still be applied in other contexts.

Another approach from plan recognition which is not anomaly detection but is still linked to our work comes from Bouchard [7] then Roy [15]. In their studies, the authors use plan recognition to infer the goal of an elderly with Alzheimer in a smart-home environment. To do so, the authors propose to use an extensive plan library which contains all the possible valid plans as well as all the possible invalid plans, i.e which does not lead to an intended goal. Using this approach, the system is able to identify the goal and the state of mind of a person in case of cognitive distress.

III. PARC and the Particle Filter Approach of Plan Recognition

By definition, an anomaly is something that is not expected. In our case, it is the execution of unexpected actions or the absence of expected ones. Using plan recognition, it would mean either create a subset of unexpected actions, or a subset of expected ones. Creating the subset of expected actions or unexpected actions based on their probability of appearance is a complex task, especially with noisy observations [13] [16]. The particle filter approach of plan recognition over a plan library is a good candidate to deal with this problem for three reasons: (1) it can give results in real-time and can deal with noisy observations, (2) in order to determine the goal of an observed agent, the system computes an estimator of the most probable next observations, which can be used as a set of expected actions, (3) this approach has been embedded in PARC, which can be used for real-life plan recognition on mobile robots.

A. Plan Libraries Representation

The particle filter approach relies on plan libraries. Plan libraries are collections of goals, actions, and rules describing how an observed agent will interact with his environment in order to perform certain tasks. This approach uses more specifically the formalism proposed by Geib and Goldman [13]. This formalism describes plan libraries as plan tree grammars and plans as partially ordered AND/OR trees. In these trees, actions are represented by eaves and goals as roots.

Formally speaking, it follows definition 1.

Definition 1. *A Plan Library (PL) is a tuple* $PL = (A, NT, G, R)$ *where :*

- A *is a finite set of terminal symbols;*
- NT *is a finite set of non-terminal symbols;*
- $G \subset NT$ *is the set of goals;*
- R *is a set of production rules in the form* $\alpha \rightarrow S, \sigma$*, with* $\alpha \in NT$*,* S *a set of symbols from* $A \cup NT$ *and* σ *a partial order of* S*.*

Figure 1 represents a small plan library with 2 goals *Tea Making* and *Hot chocolate making*, a sub goal with *Make hot water* and 7 low level actions. Arrows indicate the partial order, in this case that all other actions have to be performed before *Fill mug* and *Fill with water*.

This plan library definition is interesting to represent complex ADL, especially at home. Most of the ADL can be expressed as a sequence of lower-level actions which can be linked to a goal like the one presented in Figure 1. Furthermore, this approach does not rely on plan domains like some other plan recognition techniques [17]. It can be considered as an advantage since people usually don't plan their actions and work more with habits. Habits seem easier to represent with plan libraries since it is easier to add non-necessary actions, part of the habit, in the action sequence. One of the inconvenient is that it is necessary to create an explicit plan library. This problem exists also with a planning-based approach, where an explicit planning domain is required.

In order to describe the preferences of an observed agent, these plan libraries are linked to a Decision Model. Decision Models are usually probability distributions linked to the

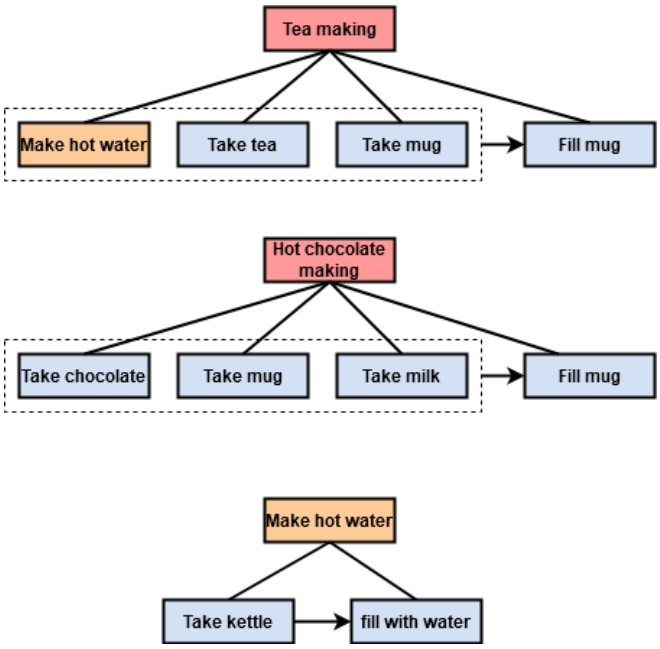

Fig. 1: Plan Library Example

production rules described in the plan libraries. We use the following definition.

Definition 2. *A Decision Model DM over a Plan Library PL is a probability distribution in the form $DM : NT \times P \rightarrow [0,1]$, with the following constraint $\forall nt \in NT, \sum_{p \in P} DM(nt,p) = 1$.*

The Decision Model aims to describe 2 kinds of preferences : (1) the preferences over the intentions of an observed agent, for example, someone might prefer tea to hot chocolate, therefore the probabilities linked to the goal *Tea making* would be higher than the probabilities linked to *Hot chocolate making*, (2) the preferences over the way a goal can be achieved if there several rules that can achieve the same goal/sub-goal.

Since in real-life applications the environment is generally not fully observable and the sensors tend to get noisy observations, the system needs to have a description of these noises. In the particle filter approach, the noise is described as a probability distribution of the set of actions over a power set of himself, as described in Definition 3.

Definition 3. *A noise function N over a Plan Library PL is a probability distribution in the form $N : A \times (A+\emptyset)^p \rightarrow [0,1]$, with the following constraints $\forall a \in A, \sum_{o \in (A+\emptyset)^p} N(a,o) = 1$ and $p \in \mathbb{N}$.*

This probability distribution is very similar to the observation function which can be found in planning domains. It also can be interpreted as a confusion matrix. This representation can express the 3 kinds of noises which can be found in plan recognition : (1) mislabeled observation, when an action is mixed with another one, (2) missing observation, when an action is not observed, (3) extraneous action, when an action that has not occurred as been observed in addition of an another observation.

B. Plan Recognition & Activity Prediction

Particle filters are a class of sampling algorithms used to compute probability distributions over Markov processes [18]. Most of the particle filter approaches follow the same steps :

1) Create a population of particles, each particle being a single simulation.
2) For each particle, generate an outcome using the simulation.
3) Sort the particles by the outcomes.
4) Once a new observation is received, keep the particles where the outcomes match the observation.
5) Compute the probability distribution over the Markov processes by counting surviving particles for each possible Markov process.
6) Increase the population by copying some of the remaining particles.
7) Go back at step 2.

Applied to plan recognition using plan libraries, this approach consists in considering partial plan trees as particles. A partial plan tree is a plan tree where all the sub-goals and actions are not specified yet. In order to create the initial population, this approach randomly selects goals, which are the partial plan trees root, using the decision model. Usually, the population size is fixed and determined before the initialization. Previous works [19] [9] have shown that even a relatively small population, over 500 particles, can give a high degree of certainty.

There are 2 steps to generate the outcomes. The first step consists in expanding the partial plan tree until a low-level action is reached. To do that the system chooses randomly, based on the decision model, a new action or subgoal from all the possible next actions or subgoals. This expansion of the plan tree is done recursively until a low-level action has been added to the partial plan tree.

The second step consists in generating an Expected Next Observation (ENO). The ENO corresponds to what would be observed if the particle describes accurately the plan followed by an observed agent. Since the model supports noise, the idea behind the ENO is that rather than using the last low-level action of a particle to sort them, the noise function is applied to the last low-level action. This can generate either the real action or another action, i.e a noisy observation. By using the ENO as the sorting outcome, the filtering can be done directly, even with noisy observation, and without further computation.

Figure 2 shows the generation of ENO on a single particle using the top-down approach. At initialization, the partial plan tree contains only the goal (particle root). At step 1, one of the possible actions or subgoals is randomly selected, here it is *Take tea* which is a low-level action. The noise function is applied to this action. In this case, we could imagine that *Take tea* and *Take milk* can be confused, which could lead the ENO not corresponding to the actual action. If this particle

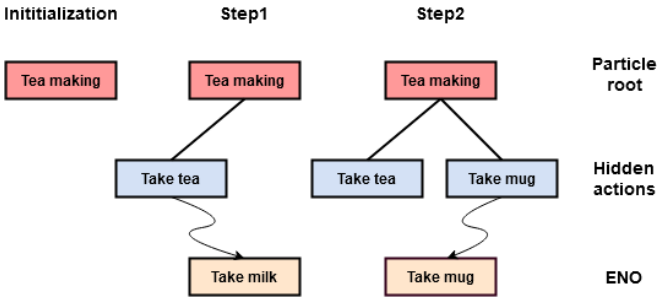

Fig. 2: Top-down generation of ENO

survives the filtering, it would lead to step 2, where a new action is randomly selected, here *Take mug*, and a new ENO is emitted, here the ENO corresponds to the actual action.

Anytime, an estimator of the probability of each goal can be found using the following equation :

$$\widehat{P}(G|O) = \frac{1}{|Pop|} \sum_{p \in Pop} \delta_p(p.G = G) \tag{2}$$

In this equation, Pop is the total number of particles and $\delta_p()$ is the Dirac operator over a particle p, i.e it returns 1 if the condition in the parenthesis is true, 0 otherwise. This equation consists in using the normalization of the particle population as a probability estimator, like for most other particle filter techniques.

In order to refill the particle population after filtering, a good strategy is to take the surviving particles and clone them randomly until the initial population size is reached.

IV. ANOMALIES DETECTION USING THE PARTICLE FILTER

We propose to use the particle filter already used for plan and recognition to detect anomalies. Since we have a set of particles, and these particles compute the ENO, we can use these particles in order to estimate the probability of an observation knowing the previous observation using the same logic used to estimate the probability of goal.

The new equation to describe this estimator is the following

$$\widehat{P}(O_{t+1}|O_t) = \frac{1}{|Pop|} \sum_{p \in Pop} \delta_p(p.ENO = O_{t+1}) \tag{3}$$

In the same idea as in Equation (2) the probability is estimated by counting the number of particle with a specific ENO.

Like for Geib and Goldman goal abandonment or for some non neural-network based approaches of anomalies detection, we propose to use the previous estimator with a threshold σ to determine if an action is normal knowing a plan library and the previous observations or if it is an anomaly.

$$\begin{cases} anomaly, & \text{if } \widehat{P}(O_{t+1}|O_t) < \sigma \\ normal, & \text{otherwise} \end{cases} \tag{4}$$

One of the main advantages of this approach is that it does not require any new calculus, except a check on the threshold. Therefore, this approach has the same features as the underlying plan recognition technique, which means it is real-time and should at least be partially noise tolerant. These 2 features are essential in order to perform plan recognition and anomaly detection on a mobile robot.

V. EXPERIMENTS

We perform our experiment on 2 sets of simulated plan libraries inspired by the plan libraries generally used to test plan based plan recognition performances [19] [20] [21] as well as PARC for ADL recognition [9]. Both sets of simulated plan libraries consist in a randomly generated domain with 5 high-level goals, a depth of 4, a branching factor of 3 for AND rules and a branching factor of 2 for OR rules. We consider equiprobability for every rule. We also add a 33 % probability of ordering constraint between any pair of symbols in all the production rules. This leads to plans of a length of 9 actions. For the first set, we use 10 low-level actions. Since we wanted to assess the precision with lower ambiguity, we propose a second set of plan libraries with 100 low-level actions.

To represent anomalies in the plan execution, we randomly chose a value t representing the instant when an observed agent stops to follow the plan. All actions in the plan performed after this instant t are randomly chosen over the set of actions. We perform two sets of experiments. In the first one, all plans contain anomalies, on the second set, all plans are done correctly. We test our approach with 1000 particles, for 3 levels of threshold, at 1%, 3% and 5% and for 4 levels of noise at 0%, 10%, 20% and 30%.

For these experiments, we measure the accuracy of our approach, i.e the percentage of true positive, the false positive, the false negative, the percentage of goal found before inferring an anomaly, which indicates the certainty of the system over goal recognition before an anomaly is detected, and the latency. The latency corresponds to the number of observations needed before inferring an anomaly after the first anomaly occurs.

Table 1 shows the results, ie the set of values Accuracy (the percentage of anomalies detected accurately), the ratio of false positive, the ratio of false negatives, the percentage of cases when the goal was accurately found before declaring the anomaly, and the Latency, the mean observations required before detecting an anomaly after its start, when there are anomalies in every plans and table 2 presents the results when there are no anomalies to detect, in the case of the 10 low-level actions domain (1st half of the tables) and the 100 low-level actions domain (2nd half of the tables), for several noise values (0, 10%, 20% and 30%) and testing 3 thresholds (1%, 3% and 5%).

The system is able to detect anomalies accurately when they are present with high precision (in more than 90% percent of the tests) with a threshold of 1% and no noise. With this threshold, adding noise decreases Accuracy, but even with maximal noise of 30% , the system still detects anomalies accurately 2 out of 3 times (see table 1, 1st half, Accuracy). When there are no anomalies to detect, the system

TABLE I: Results in case of anomalies in the plan execution

	Noise	Accuracy	False Positive	False Negative	Goal Found	Latency (avg)
10 actions domain						
precision = 1%	0	0.929	0.037	0.034	0.826	1.41
	0.1	0.802	0.175	0.023	0.76	1.546
	0.2	0.657	0.323	0.02	0.694	1.715
	0.3	0.612	0.369	0.019	0.643	1.794
precision = 3%	0	0.819	0.145	0.036	0.748	1.056
	0.1	0.687	0.302	0.011	0.676	1.223
	0.2	0.55	0.441	0.009	0.669	1.38
	0.3	0.517	0.475	0.008	0.59	1.358
precision = 5%	0	0.67	0.31	0.02	0.627	0.9478
	0.1	0.513	0.474	0.013	0.605	1.027
	0.2	0.484	0.514	0.002	0.567	0.9463
	0.3	0.403	0.594	0.003	0.538	1.127
100 actions domain						
precision = 1%	0	0.846	0.15	0.004	0.765	0.1111
	0.1	0.619	0.381	0	0.719	0.1179
	0.2	0.452	0.548	0	0.649	0.1704
	0.3	0.326	0.674	0	0.624	0.1933
precision = 3%	0	0.379	0.621	0	0.424	0.06069
	0.1	0.26	0.74	0	0.385	0.05769
	0.2	0.218	0.782	0	0.393	0.06881
	0.3	0.185	0.815	0	0.322	0.07027
precision = 5%	0	0.185	0.815	0	0.243	0.03784
	0.1	0.162	0.838	0	0.241	0.02469
	0.2	0.14	0.86	0	0.249	0.03571
	0.3	0.139	0.861	0	0.234	0.02158

TABLE II: Results in case of plan execution without anomalies

	Noise	Accuracy	False Positive	Goal Found
10 actions domain				
precision = 1%	0	0.899	0.101	0.961
	0.1	0.379	0.621	0.851
	0.2	0.166	0.834	0.763
	0.3	0.068	0.932	0.678
precision = 3%	0	0.698	0.302	0.881
	0.1	0.311	0.689	0.769
	0.2	0.116	0.884	0.703
	0.3	0.06	0.94	0.66
precision = 5%	0	0.451	0.549	0.777
	0.1	0.201	0.799	0.696
	0.2	0.068	0.932	0.627
	0.3	0.037	0.963	0.578
100 actions domain				
precision = 1%	0	0.846	0.154	0.883
	0.1	0.329	0.671	0.8
	0.2	0.107	0.893	0.739
	0.3	0.032	0.968	0.657
precision = 3%	0	0.261	0.739	0.423
	0.1	0.093	0.907	0.412
	0.2	0.045	0.955	0.388
	0.3	0.009	0.991	0.345
precision = 5%	0	0.063	0.937	0.257
	0.1	0.028	0.972	0.266
	0.2	0.011	0.989	0.253
	0.3	0.005	0.995	0.267

still functions properly with a threshold of 1% and no noise but the performance quickly decreases with noise addition (see table 1, 1st half, Accuracy). If there is an anomaly to detect the system is still functional with 10% of noise, Accuracy is still above 80% however it is to note that in these conditions the system has a high rate of false-positive for anomalies detection when there are no anomalies to detect (see table 2, 1st half, Accuracy). Interestingly, we can see that the results are consistently better (higher accuracy values, lower false positive and false negative values) for the domain with 10 actions than for the domain with 100 actions. This can be explained by the fact that in the 100 actions domain each action has a low percentage of occurrence, therefore the system is more prone to call an anomaly for actions that are not anomalies but are only rare actions. This is consistent with the fact that the 100 actions domain has higher false positive values than the 10 actions domain for the same noise and threshold values. Another interesting result is that setting the threshold value at 1% gives better results than, 3% or 5% regardless of the number of actions in the domain with or

without the presence of anomalies. Increasing the threshold value, increases the false positive and false negative rates thus decreasing accuracy. The system is able to consistently infer the current goal when detecting an anomaly, the value Goal found is above 70% with the 1% threshold and noise up to 10%. This is interesting because it shows that this system could propose the next actions required to accomplish the goal of a person in cognitive distress, which is an interesting feature for assisting robots. Latency results are also worth mentioning (table I, Latency), when there is an anomaly to detect, higher threshold values decrease the latency, the system is less precise but more sensitive. An assisting robot could react faster in case of anomaly detection if using higher threshold values but it would be at the cost of precision. Increasing the noise decreases the Accuracy and Goal Found values while increasing the rate of false positive and false negative and the latency increases. At 10% of noise, the system still performs well, however the performance drops considerably in all cases when the noise is at 20% or higher. This is an issue for real-life applications, in PARC, the noise is fairly constant around 30%, this anomaly detection system is not precise enough yet to be embedded in a mobile robot.

VI. CONCLUSION

In this paper, we propose to address the anomaly detection problem using an approach from the plan recognition domain, specifically the particle filter approach over a plan library. This approach computes the prior probability of possible observations. With this information and a predetermined threshold, the system can infer is an observation of a person is normal or if its an anomaly knowing the previous observations.

This approach offers several advantages : (1) it can be used in real-time settings, (2) since it computes the probable goal and next observations an observed agent it does not need to explicitly know what the goal is to determine if there is an anomaly, (3) the fact that the system can compute the most probable goal and the most probable next actions means it can use these results to offer cognitive help to someone in cognitive distress.

While this approach can handle a few noisy observations, the accuracy drops significantly when the percentage of noisy observations is over 20%. PARC, one of the systems using the particle filter approach to perform plan recognition on a mobile robot for ADL recognition, works with around 30% of noisy observations. This gap means that our approach of anomaly detection can not be embedded in a mobile robot without external sensors due to the noise occurring in the low-level activity recognition. That being said, this approach seems promising. It could work by decreasing the observation noise, either by adding external sensors or by improving the activity recognition onboard the mobile robot.

In future works, we would like to focus on improving the activity recognition by using Active Goal Recognition approaches [22]. Active Goal Recognition consists in using predictions over the goal of an observed agent in order to improve the observations. By improving these observations this way, we may enable the use of our anomaly detection approach to be used on mobile robots. A second focus would be on how to construct plan libraries. As we are aware of, there is no way to atomically construct these plan libraries which means a lot of work to construct them apriori.

ACKNOWLEDGMENT

We want to address a special thanks to César Ombredane, Mathieu Gravel, and Amandine Laffitte, without whom this work would not have been possible.

REFERENCES

[1] M. E. Pollack, "Intelligent technology for an aging population: The use of AI to assist elders with cognitive impairment," *AI magazine*, vol. 26, no. 2, p. 9, 2005.

[2] S. Jenkins and H. Draper, "Care, monitoring, and companionship: views on care robots from older people and their carers," *International Journal of Social Robotics*, vol. 7, no. 5, pp. 673–683, 2015.

[3] F. Palumbo, J. Ullberg, A. Štimec, F. Furfari, L. Karlsson, and S. Coradeschi, "Sensor network infrastructure for a home care monitoring system," *Sensors*, vol. 14, no. 3, pp. 3833–3860, 2014.

[4] S. Laniel, D. Létourneau, M. Labbé, F. Grondin, and F. Michaud, "Enhancing a beam+ telepresence robot for remote home care applications," in *2017 International Conference on Virtual Rehabilitation (ICVR)*. IEEE, 2017, pp. 1–2.

[5] G. Sukthankar, C. Geib, H. H. Bui, D. Pynadath, and R. P. Goldman, *Plan, activity, and intent recognition: Theory and practice*. Newnes, 2014.

[6] C. W. Geib, "Problems with intent recognition for elder care," in *AAAI Workshop Automation as Caregiver*, 2002, pp. 13–17.

[7] B. Bouchard, S. Giroux, and A. Bouzouane, "A keyhole plan recognition model for alzheimer's patients: First results," *Applied Artificial Intelligence*, vol. 21, no. 7, pp. 623–658, 2007.

[8] A. A. Sodemann, M. P. Ross, and B. J. Borghetti, "A review of anomaly detection in automated surveillance," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 42, no. 6, pp. 1257–1272, 2012.

[9] J. Massardi, M. Gravel, and E. Beaudry, "Parc: a plan and activity recognition component for assistive robots," in *International Conference on Robotics and Automation (ICRA)*, 2020.

[10] S. W. Yahaya, A. Lotfi, and M. Mahmud, "A consensus novelty detection ensemble approach for anomaly detection in activities of daily living," *Applied Soft Computing*, vol. 83, p. 105613, 2019.

[11] A. Revathi and D. Kumar, "An efficient system for anomaly detection using deep learning classifier," *Signal, Image and Video Processing*, vol. 11, no. 2, pp. 291–299, 2017.

[12] C. W. Geib and R. P. Goldman, "Recognizing plan/goal abandonment," in *International Joint Conferences on Artificial Intelligence (IJCAI)*, 2003, pp. 1515–1517.

[13] ——, "A probabilistic plan recognition algorithm based on plan tree grammars," *Artificial Intelligence*, vol. 173, no. 11, pp. 1101 – 1132, 2009.

[14] R. F. Pereira, N. Oren, and F. Meneguzzi, "Using sub-optimal plan detection to identify commitment abandonment in discrete environments," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 11, no. 2, pp. 1–26, 2020.

[15] P. Roy, B. Bouchard, A. Bouzouane, and S. Giroux, "A hybrid plan recognition model for alzheimer's patients: interleaved-erroneous dilemma," *Web Intelligence and Agent Systems: An International Journal*, vol. 7, no. 4, pp. 375–397, 2009.

[16] G. Behnke, D. Höller, and S. Biundo, "On the complexity of htn plan verification and its implications for plan recognition," in *International Conference on Automated Planning and Scheduling (ICAPS)*, 2015.

[17] M. Ramırez and H. Geffner, "Plan recognition as planning," in *International Joint Conferences on Artificial Intelligence (IJCAI)*, 2009, pp. 1778–1783.

[18] O. Cappé, S. J. Godsill, and E. Moulines, "An overview of existing methods and recent advances in sequential monte carlo," *Proceedings of the IEEE*, vol. 95, no. 5, pp. 899–924, 2007.

[19] J. Massardi, M. Gravel, and E. Beaudry, "Error-tolerant anytime approach to plan recognition using a particle filter," in *International Conference on Automated Planning and Scheduling (ICAPS)*, vol. 29, no. 1, 2019, pp. 284–291.
[20] F. Kabanza, J. Filion, A. R. Benaskeur, and H. Irandoust, "Controlling the hypothesis space in probabilistic plan recognition." in *International Joint Conferences on Artificial Intelligence (IJCAI)*, 2013, pp. 2306–2312.
[21] R. Mirsky *et al.*, "Slim: Semi-lazy inference mechanism for plan recognition," *arXiv preprint arXiv:1703.00838*, 2017.
[22] M. Shvo and S. A. McIlraith, "Active goal recognition." 2020.