2014 IEEE International Conference on Systems, Man, and Cybernetics
October 5-8, 2014, San Diego, CA, USA

Logic-probabilistic model for event recognition in a robotic search and rescue scenario

José A. Gurzoni Jr and Fabio G. Cozman
Escola Politécnica
Universidade de São Paulo
S. Paulo - SP, Brazil

Murilo F. Martins and Paulo E. Santos
Dept. of Electrical Engineering,
FEI University Centre
São Bernardo do Campo - SP, Brazil 09850-901.

Abstract—**This paper presents initial results towards the development of a logic-based probabilistic event recognition system capable of learning and inferring high-level joint actions from simultaneous task execution demonstrations on a search and rescue scenario.**

We adopt a probabilistic extension of the Event Calculus defined over Markov Logic Networks (MLN-EC). This formalism was applied to learn and infer the actions of human operators teleoperating robots in a real-world robotic search and rescue task.

Experimental results in both simulation and real robots show that the probabilistic event logic can recognise the actions taken by the human teleoperators in real world domains containing two collaborating robots, even with uncertain and noisy data.

I. Introduction

Robotic urban search and rescue (USAR) is a challenging research area due to the complexity of exploring and actuating in highly cluttered and unstructured disaster scenarios. For this reason, the majority of robots deployed in USAR situations today are teleoperated [1], and while there is an increasing interest in the robotics community for improving the autonomy of robots in USAR situations, little attention has been paid to the issue of using the amount of information contained in the teleoperated actions for learning two-robot joint action plans from demonstration [2]. The present paper describes initial results towards the development of an event recognition system capable of recognising high-level joint actions from simultaneous task execution demonstrations on a search and rescue scenario. This work assumes the standpoint of logic-based event recognition systems, whereby joint actions are interpreted from a knowledge-base of events represented on a Markov Logic Network (MLN) [3] extended with a formal ontology for time points, actions and change.

Event recognition systems based on probabilistic logic have been receiving considerable attention lately due to their capability of representing relational knowledge while providing rigorous probabilistic inference tools [4]. Closely related to the purposes of the present work, the task of event recognition from image sequences using MLN was first investigated in [5] where domain commonsense knowledge is represented using first-order logic sentences, while the uncertainty inherent to the event recognition is represented by the weights in the MLN formulae. These initial proposals, however, did not have a clear ontology of time. This issue was tackled in [6], where event recognition is accomplished using an MLN version of the Event Calculus [7], [8], which is a logical theory for representing and reasoning about actions and change. In all these approaches, events were observed from a single, static, camera. In this work we build upon these ideas to develop an event recognition system capable of interpreting two-robot actions from task execution demonstrations.

This paper starts by briefly presenting related works that also applied temporal relations to real reasoning problems in robotics (Section II). Section III describes the Event Calculus formalism, and Section IV introduces its probabilistic extension (the MLN-EC). The search and rescue task used in this work is then presented on Section V, and Section VI describes our Probabilistic Event Calculus implementation with Markov Logic Networks for recognizing two-robot joint actions.

II. Related work

The use of theories that define temporal relations to model events in reasoning under uncertainty problems appears in a number of existing works in the literature. [9] extend Allen's Interval Algebra [10] with their probabilistic graphical representation P-CASE and apply the resulting systems to activity detection in video scenes. [11] use a probabilistic version of Allen's Algebra to create a relational structure that captures spatio-temporal relations in video scenes and use it to classify and detect events. Whereas [12] combine Allen's Algebra formalism to Markov Logic Networks to automatically discover high-level events that happened in a basketball game, from noisy visual data. [13], [14] present a probabilistic extension of the Event Logic, called Probabilistic Event Logic (PEL). As PEL has been specifically created to deal with temporal relations, its inference mechanism features optimization methods that only add to the grounded graph events where the corresponding logical formula in the knowledge base (KB) is true for the queried time interval, thus avoiding the computational effort of grounding all events on every time interval. In [15], the authors present a probabilistic variant of the GoLog language (which is itself based on the Situation Calculus) capable to recognize multi-agent action plans, and apply it to a car traffic domain. [16] use the Linear Temporal Logic (LTL) formalism to model the processes of a warehouse where a robot must operate. To handle the uncertainty due to noise in observations, the authors use clustering algorithms on the sensor data, which is then transformed into qualitative observations that are inputs of the LTL model.

In [17] some of the MLN-EC authors show another possible version of a probabilistic Event Calculus implementation, written in the ProbLog language [18]. The ProbLog language offers a more straight-forward manner to visualise the probabilities associated with the formulas and terms than the MLN. We also plan to investigate this possibility in a subsequent work.

III. Event Calculus

This section succinctly describes the Event Calculus (EC). For a more complete understanding, we recommend the reading of [19].

The EC is a first-order formalisation of time points, events (including actions)[1] and fluents, whose axioms express the *commonsense law of inertia* that constrain the fluents to be affected only by the actions explicitly related to them. In this work, EC guarantees a compact representation of high-level actions, while also providing a seamless solution to the frame problem. Informally, the EC axioms constrain a fluent F to hold at a time T in the following base situations[2]: F *holds* initially and is not *terminated* up till time T; F *holds* if it was *initiated* by some event that *happened* at time T_0, and no other event *terminated* it in the interval $[T_0, T]$ (EC includes analogous axioms for the cases when a fluent does not hold). The commonsense law of inertia in EC solves the frame problem by imposing a circumscriptive policy over the domain, constraining its interpretations to the intended models of its actions and fluents.

The following are the predicates of the Event Calculus and their meanings:

- $HoldsAt(f,t)$ - fluent f holds at time t;
- $Initially(f)$ - fluent f holds from time 0;
- $Happens(a,t)$ - event a happens at time t;
- $Initiates(a,f,t)$ - fluent f holds after the occurrence of an event a at time t;
- $Terminates(a,f,t)$ - fluent f does not hold after the occurrence of an event a at time t;
- $Releases(a,f,t)$ - fluent f is not subject to the commonsense law of inertia after event a at time t;

Two defined notions are also introduced:

- $Clipped(t_1,f,t_2)$ - means that the fluent f ceased to hold (or has been released from the commonsense law of inertia, as will be discussed next) in some time point in $[t_1,t_2]$;
- $Declipped(t_1,f,t_2)$ - means that the fluent f started to hold (or has been released from the commonsense law of inertia) in some time point in $[t_1,t_2]$.

The predicates described above are constrained by Axioms 1, 2 and 3 below.

[1]In this paper the terms *actions* and *events* are used interchangeably.

[2]The emphasized terms are predicates in the EC ontology

$$HoldsAt(f,t) \leftarrow \quad (1)$$
$$Initially(f) \wedge \neg Clipped(0,f,t).$$
$$HoldsAt(f,t_2) \leftarrow \quad (2)$$
$$Happens(a,t_1) \wedge Initiates(a,f,t_1) \wedge t_1 < t_2 \wedge$$
$$\neg Clipped(t_1,f,t_2).$$
$$\neg HoldsAt(f,t_2) \leftarrow \quad (3)$$
$$Happens(a,t_1) \wedge Terminates(a,f,t_1) \wedge$$
$$t_1 < t_2 \wedge \neg Declipped(t_1,f,t_2).$$

$Clipped/3$ and $Declipped/3$ are defined by Formulae 4 and 5.

$$Clipped(t_1,f,t_2) \leftrightarrow \quad (4)$$
$$\exists a,t[Happens(a,t) \wedge [Terminates(a,f,t) \vee$$
$$Releases(a,f,t)] \wedge t_1 < t \wedge t < t_2]$$
$$Declipped(t_1,f,t_2) \leftrightarrow \quad (5)$$
$$\exists a,t[Happens(a,t) \wedge [Initiates(a,f,t) \vee$$
$$Releases(a,f,t)] \wedge t_1 < t \wedge t < t_2]$$

The predicate $HoldsAt(f,t)$, according to Axioms (1) and (2), says that the fluent f is true from instant zero to the time point t, according to (1); or from a time point t_1 to a time point t_2 if an event initiates f at a time t_1, according to (2). In other words, Axiom (1) states that $HoldsAt(f,t)$ is true if the fluent f holds at time zero (represented by $Initially(f)$) and no event occurred that terminates (or releases) the fluent f in the time interval $[0,t]$, this is represented by $\neg Clipped(0,f,t)$. According to (2), $HoldsAt(f,t)$ is true if an event a happens at time t_1 initiating the fluent f, and no event occurred that terminates (or releases) f during the time interval $[t_1,t_2]$. The predicate $Releases(a,f,t)$, used in Definitions (4) and (5), represents the case that action a causes indetermination in the value of fluent f. Axiom (3) states that a fluent f **does not** hold at a time point t_2 ($\neg HoldsAt(f,t_2)$) if an event a occurs at a time t_1 ($Happens(a,t_1)$) that terminates the fluent f, and no other event occurs that initiates f in the time interval $[t_1,t_2]$ ($\neg Declipped(t_1,f,t_2)$). The sentence (4) defines the predicate $Clipped(t_1,f,t_2)$ in the way that it is true if an event occurs (in the time interval $[t_1,t_2]$) such that the fluent f is terminated by the event or is released from the commonsense law of inertia ($Releases(a,f,t)$). Analogously to (4), (5) says that the predicate $Declipped(t_1,f,t_2)$ is true if an event a occurs at time point t ($t \in [t_1,t_2]$) such that a initiates the fluent f or releases it from the commonsense law of inertia.

The axioms presented so far describe which events occur and their effects on the world. However, because of the semantics of first-order logic, when an event happen that affects a fluent, one cannot ensure that the value of other (possible unaffected) fluents remain unchanged, this is called the frame problem. EC resolves that through the use of *circumscription*, which limits the extension of the domain-dependent predicates. In a nutshell, circumscription ensures that if a clause states that certain logical conditions initiate an event, then only these same logical conditions are also required for the predicate to be considered as initiated.

IV. MLN-EC

In this paper we use the dialect proposed in [6], called MLN-EC, that encodes the EC axioms into Markov Logic Networks.

Markov logic networks [3] are probabilistic relational graphical models defined by a set of weighted first-order formulae. The formulae define the topology of a Markov network that has in its vertices each possible grounding of the formulae predicates, whereby the network edges represent the logical connectives in the formulae. Thus, each formula defines a clique in the graph and the formula's weight represents the clique's potential.

More formally, a Markov logic network L is a set of formulae F_i in first-order logic with a weight w_i (real number) attached to each formula. This can be viewed as a template for constructing Markov networks $M_{L,C}$, where C is a set of constants. The probability distribution of the domain variables, over possible worlds x (interpretations of F_i), is given by:

$$P(X=x)=\frac{1}{Z}exp\left(\sum_i w_i f_i(x)\right)=\frac{1}{Z}\prod_i \phi_i(x_{\{i\}})^{n_i(x)},$$

where $n_i(x)$ is the number of true groundings of F_i in x, $x_{\{i\}}$ is the state of the i-th clique which has a corresponding feature $f_i(x) \in \{0,1\}$ and an associated weight $w_i = log\phi_i(x_{\{i\}})$. Z is a normalisation factor, also known as the partition function,

$$Z = \sum_{x\in\mathcal{X}} \prod_i \phi_i(x_{\{i\}}). \tag{6}$$

The basic idea of MLN is to soften the restrictions imposed by a first-order knowledge base (KB), i.e., the more formulae in the KB an interpretation satisfies, the more probable it is. The weight associated to each formula reflects its strength as a constraint. Formulae with infinite weights rule out worlds in which they are not satisfied, whereas lower weight formulae make these worlds less probable but not impossible. In other words, the weight w_i means that a world where n ground instances of a formula are true is $e^{w_i n}$ less probable than a world where all instances of this formula are true.

In the MLN-EC, the authors use a simplified version of the Discrete Event Calculus (DEC) with four domain-independent axioms, the two shown below and their equivalent for the $\neg HoldsAt$. With this simplification, there are no more existentially quantified variables, what reduces significantly the number of ground clauses in the generated network.

$$\begin{aligned}&HoldsAt(F,T+1)) \leftarrow\\ &\quad InitiatedAt(F,T)\end{aligned}$$

$$\begin{aligned}&HoldsAt(F,T+1)) \leftarrow\\ &\quad HoldsAt(F,T)\wedge\\ &\quad\quad \neg TerminatedAt(F,T)\end{aligned}$$

Along with these domain-independent axioms, the domain-dependent rules can be defined. These formulae have the general form shown below, where X and Y are domain objects, T is a time point and *Conditions* impose restrictions on the formula definition.

$$\begin{aligned}&InitiatedAt(fluent_i(X,Y,T)) \leftarrow\\ &\quad Happens(event_i(X,Y),T) \wedge \ldots \wedge Conditions[X,Y,T]\end{aligned}$$

The domain-independent axioms are defined as hard constraints (i.e. infinite weights, which are used to express deterministic rules) whereas the domain-dependent rules, which represent the specific fluents and actions of a particular task, are represented as probabilistic formulae. This policy guarantees the consistency of any action specification (forced by the domain-independent hard constraints) while also allowing for an effective treatment of uncertainty in the action interpretation process.

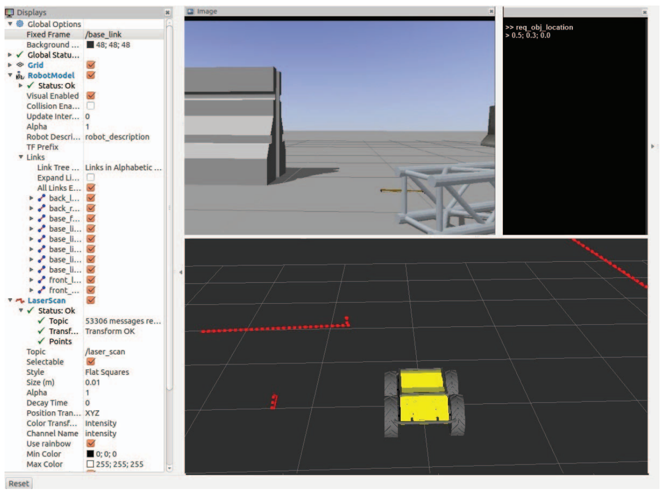
V. Urban Search and Rescue Task

A search and rescue task scenario was created for tests. The task consisted of a search for hazardous objects in an unknown location, which the robots had to find, retrieve and carry to an isolation container while avoiding obstacles and dead-ends formed by walls and debris in the scene. Two robots were used in the task: one was a terrestrial wheeled robot, equipped with a PTZ (Pan, Tilt and Zoom) camera and a front-mounted 2D laser range-finder. The other was an aerial vehicle, a quadrotor, equipped with a downward camera capable of transmitting live images to the pilot on the ground. Both robots also had GPS and compass units for localisation aid. During the task execution, the robots were teleoperated, each of them controlled from a remote dashboard (as shown in Figure 1) that allowed the operator to control the robot via joystick as well as to view its camera, sensors and GPS coordinates. In the case of the terrestrial robot, the operator could also rotate and zoom the camera, and to see the laser scanner reading indicating the position and shape of nearby obstacles.

The dashboard registers all the commands issued by the operators, including when they access sensor or image data. It also features a module for communication between the operators via text exchange, using pre-defined messages like *request_object_location* or *request_gps_location_of_object*. The use of pre-defined messages was devised as a way to capture the interactions between the operators, while the recording of actions serves to capture the individual actions taken by them. The dashboard information, with the registered commands, images from cameras, laser range data and GPS information, is all recorded, to be used as input evidence by the event recognition system.

VI. MLN-EC applied to USAR Event Recognition

In order to build the two-robot plans, it is necessary to recognise the events involved in the task execution, that is, the actions performed by the teleoperators. This is the purpose of the implemented system in this work.



```
Global Options
  Fixed Frame        /base_link
  Background ...     48; 48; 48
Global Statu...
Grid                 ✓
RobotModel           ✓
  Status: Ok
  Visual Enabled     ✓
  Collision Ena...   ☐
  Update Inter...    0
  Alpha              1
  Robot Descri...    robot_description
  TF Prefix
  Links
    Link Tree ...    Links in Alphabetic ...
    Expand Li...     ☐
    All Links E...   ✓
      back_l...      ✓
      back_r...      ✓
      base_f...      ✓
      base_li...     ✓
      base_li...     ✓
      base_li...     ✓
      base_li...     ✓
      base_li...     ✓
      front_l...     ✓
      front_...      ✓
LaserScan            ✓
  Status: Ok
    Topic            53306 messages re...
    Transf...        Transform OK
    Points
  Topic              /laser_scan
  Selectable         ✓
  Style              Flat Squares
  Size (m)           0.01
  Alpha              1
  Decay Time         0
  Position Tran...   XYZ
  Color Transf...    Intensity
  Channel Name       intensity
  Use rainbow        ✓
  Min Color          0; 0; 0
  Max Color          255; 255; 255
```

```
>> req_obj_location
> 0.5; 0.3; 0.0
```

Figure 1. Operator's dashboard. The upper centre window of the screen shows the robot's camera image. The bottom centre window shows the laser ranger sensor, with obstacle lines in red. The upper right window shows the communication messages. The dashboard was implemented with ROS RViz.

The event recognition system, implemented with the MLN-EC dialect from [6], consists of a probabilistic knowledge base (KB) representing the high-level actions (events) which the operators may perform during the course of the task. The occurrence of these events change relevant properties of the world, the *fluents*, and by doing so allow the recognition of the plan devised by the human operators.

In the MLN-EC, MLN formulae are written using the EC logical structure, with clauses indicating the occurrence of events that change fluents through the use of the domain-dependent predicates *Happens*, *InitiatedAt* and *TerminatedAt*. Other predicates, these domain-independent, are responsible for the EC *engine* axiomatization, and ensure the common-sense law of inertia application. *Maximum a Posteriori* (MAP) queries to the KB are made through the domain-independent predicate *HoldsAt*, which is responsible for evaluating if the fluent is valid or not at a given time.

The input evidence for the MLN-EC predicates is given by low-level event detectors, which, at each time step of the system, processes the sensory data. These are basically algorithms that parse data from the operator dashboard, such as images, sensor and control readings and extract from it a specific feature. For instance, one detector, the *target_defined*, reads the dashboard's log and outputs whether an object has been marked as the target by the operator at that time step. One other detector defines a qualitative relation called *approaching_object* as true or false based on the GPS data of the robot and the position of the objects. This input evidence is injected into the MLN-EC event recognition system, and the response is retrieved by querying the KB, which returns the probability of the ground *HoldsAt* predicates for each fluent actually holding at that given time.

The logical clauses describing the operators' sub-tasks for the USAR scenario described earlier were written. The three main fluents are: *searchObj*, indicating that the operator is searching for objects; *fetchObj* indicating he aims to fetch a specific object; and *explore* for obstacle avoidance and path finding. Initiation and termination rules were defined. Some of the *initiation* and *termination* rules are shown below. In these clauses, R is the ego-robot, R_p is the robot's partner, O is an object and T denotes time point:

$$\begin{aligned}&InitiatedAt(searchObj(R, R_p, O), T) \leftarrow \\ &\quad [happens(request\_object\_location(R_p, O), T) \vee \\ &\quad happens(camera\_tilting(R), T)] \wedge \\ &\quad \neg target\_defined(R, T) \wedge \neg has\_load(R, T).\end{aligned} \tag{7}$$

The rule above can be interpreted as: The fluent searchObj for the robot R and object O is considered initiated on the time point T if the operator is requesting an object location to his partner robot, or the operator is moving the robot's onboard camera view while a target is not defined and the robot is unloaded.

$$\begin{aligned}&InitiatedAt(fetchObj(R), T) \leftarrow \\ &\quad [happens(approaching(R, O), T) \vee \\ &\quad happens(manoeuvring(R), T)] \wedge \\ &\quad target\_defined(R, T) \wedge \\ &\quad \neg has\_load(R, T) \vee (stopped(R, T) \wedge loading(R, T)) \vee \\ &\quad (stopped(R, T) \wedge happens(req\_obj\_location(R_p, O), T))\end{aligned} \tag{8}$$

$$\begin{aligned}&TerminatedAt(fetchObj(R), T) \leftarrow \\ &\quad happens(manoeuvring(R), T) \vee \\ &\quad target\_defined(R, T) \vee has\_load(R, T)\end{aligned} \tag{9}$$

VII. Experimental Results

To test the event recognition system, an experiment was performed on the search and rescue scenario described. It consisted on having the teleoperators control the robots from their remote dashboards until all the objects in scene had been taken to the isolation container. During the execution of the task, as described in Section V, sensory data was collected at each time step, what was empirically set to 2 seconds. Once the task was accomplished, all data (operator commands, images, gps positions) was recorded and the execution was deemed complete. Four of these task executions were conducted in the simulated environment, in the Gazebo simulator, and 1 was performed with real robots. The positioning of robots, objects and obstacles was randomly changed for each execution, to create different scenarios. An image showing one of the sessions can be seen in Figure 2, where the two robots are cooperating to locate an object.

When data from all executions had been collected, supervised learning took place, to define the weights of the MLN formulae in the KB. Data collected in one of the executions was used as training dataset, and Alchemy's learning algorithm was executed, using discriminative learning with the MC-SAT algorithm [20]. Once the weights were learnt, their values were used for inference, also with the MC-SAT algorithm, on a test dataset retrieved from a different execution. This allowed us to cross-validate the experiment, and average the results. During

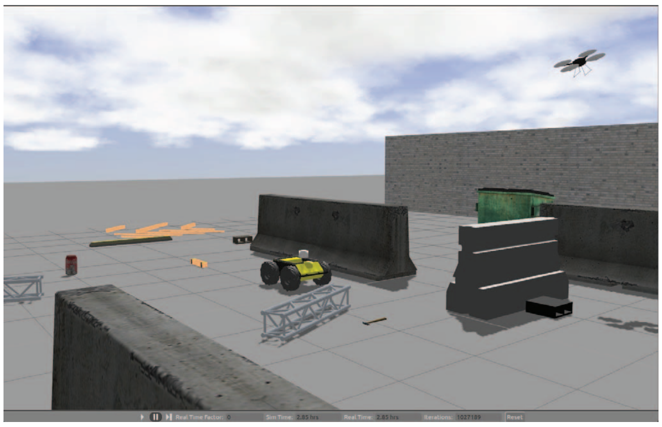


Figure 2. Image from the simulated environment, showing the two robots cooperating to retrieve the blocks. ROS Gazebo simulator was used.

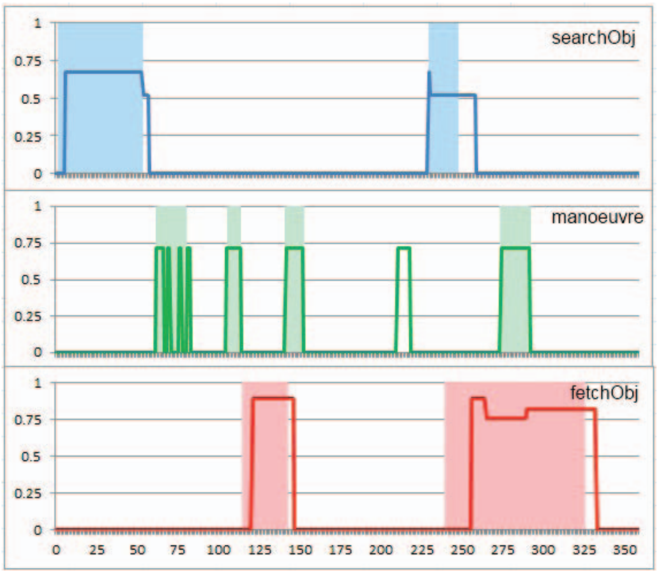


Figure 3. Experimental results. Lines represent the confidence levels of the fluents holding (the MLN outputs), the coloured areas on the graph show ground truth, and the horizontal axis is time. Blue colour represents the fluent searchObj, green for manoeuvre and red for the fetchObj.

all executions, the time step for sensory data logging was kept to 2 seconds.

The datasets contained each, on average, data from 700 time steps.

Figure 3 shows the results of holdsAt queries compared with manually annotated ground truth. The results indicate that the action sequence correctly starts with the operator searching for objects. Although with some noise, overall the system detects the actions reasonably well. Then, from time step 100 to 125 the fetching of the object starts, followed by a new manoeuvre, to move to the destination point. Between time steps 150 and 220, the graph does not show activity because taking the cargo to the destination has not been modelled. The only clear error happens at the time step 200, with a false positive.

The precision and recall rates for each of the fluents was obtained, as shown in Tab. I. These rates are the average over all 5 trials and the two robots. For the calculation of the precision and recall results, a fluent was considered to hold when its probability exceeded the threshold level set as 60%.

Table I. PRECISION AND RECALL RESULTS, WITH STANDARD DEVIATION, FOR THE DETECTION OF WHETHER FLUENTS HOLD OR NOT AT A GIVEN TIME.

Predicate	**Precision**	**Recall**
searchObj	70.1 ± 3.1	93.1 ± 2.0
fetchObj	59.8 ± 4.6	88.2 ± 2.9
manoueuvring	65.9 ± 6.5	89.7 ± 7.0

VIII. DISCUSSION

During the implementation and experiments for this work, some interesting questions arose. One of them is whether the multiple initiations and terminations of events at different time points are a good feature or a problem. Because of the iterative nature of the domain-independent axioms that propagate the law of inertia across time points in the EC, the predicate probability increases as more initiations occur, and, likewise, decreases with terminations. On one hand, this may provide immunity to noisy low-level detections and other mistakes, but, on another hand, may cause a cumulative effect on the probability values that are not correct. A more in-depth analysis of this issue is in order, although during the experiments performed, the values did not get clamped to maxima or minima.

Another question is the ramification problem. Basically, the ramification problem states that events may have indirect, potentially non-modelled, effects (also known as side-effects). These effects can substantially change the world state, and even lead to domino effects. The open question is how and to what extent the effects of events should be modelled to prevent the problem, or how exactly to deal with it. Although in this paper nothing in this regarded was studied, the topic is relevant and needs consideration, as the chances of one robot causing effects that need to be considered by the reasoning systems of others increase with the number of robotic agents involved.

During the experiments, the question about how to set the threshold for when the probability of a predicate should be considered True or not also appeared. In this work, the threshold values were manually, empirically, set. However, as more events and fluents are written to the KB, and more robots participate, it will be harder to define thresholds manually.

IX. CONCLUSIONS AND FUTURE WORKS

In this work, the focus was to apply the probabilistic Event Calculus for the recognition of actions (events) in a real-world robotic task. This has been accomplished and the results are positive, indicating that probabilistic event logics can be used to infer real-world, two-robot, actions and are capable to handle noisy data and incomplete logic modelling.

As future works, one interesting line of work would be to extend the knowledge base, incorporating more tasks from the search and rescue domain, and verifying how the system would scale. A related line of work would also be to verify the possibility to execute inferences on-line, in real-time.

In this paper we only dealt with scenarios with two robots. Extending this work to multirobot domains is an ongoing work.

Also as future works, we believe that once events can be recognised at single-and multi-robot level, these sequences, which actually form the task, can be extracted for each robot as in [2], and modelled as Markov Chains. Confidence-based approaches such as [21] could also allow for incremental autonomy of teleoperated robots, resulting in a reduction in the task load of human operators.

Last, but not least, the solution proposed in this work is not designed only to search and rescue scenarios. It can be potentially applied to other scenarios in which multiple agents are remotely controlled, such as in war zones or space exploration.

X. Acknowledgments

This work receives financial support from FAPESP, proc. 2012/04089-3. Paulo E. Santos is partially supported by CNPq. Murilo F. Martins is supported by FAPESP, proc. 2012/12640-1.

References

[1] Y. Liu and G. Nejat, "Robotic urban search and rescue: A survey from the control perspective," *J. of Intel. & Rob. Sys.*, pp. 1–19, 2013.

[2] M. F. Martins and Y. Demiris, "Learning multirobot joint action plans from simultaneous task execution demonstrations," in *Proc. of AAMAS*, 2010, pp. 931–938.

[3] M. Richardson and P. Domingos, "Markov logic networks," *Machine Learning*, vol. 62, no. 1-2, pp. 107–136, 2006.

[4] A. Artikis, A. Skarlatidis, F. Portet, and G. Paliouras, "Logic-based event recognition," *The Knowledge Engineering Review*, vol. 27, 12 2012.

[5] S. D. Tran and L. S. Davis, "Event modeling and recognition using Markov Logic Networks," in *Proceedings of the 10th ECCV: Part II*. Springer-Verlag, 2008, pp. 610–623.

[6] A. Skarlatidis, G. Paliouras, G. A. Vouros, and A. Artikis, "Probabilistic event calculus based on markov logic networks," in *RuleML'11*, 2011.

[7] R. Kowalski and M. Sergot, "A logic-based calculus of events," *New Generation Computing*, vol. 4, no. 1, pp. 67–96, 1986.

[8] M. Shanahan, "The event calculus explained," in *Artificial Intelligence Today*, ser. Lecture Notes in Computer Science, M. Wooldridge and M. Veloso, Eds. Springer Berlin Heidelberg, 1999, vol. 1600, pp. 409–430.

[9] A. Hakeem and M. Shah, "Learning, detection and representation of multi-agent events in videos," *Artificial Intelligence*, vol. 171, pp. 586–605, 2007.

[10] J. F. Allen, "Towards a general theory of action and time," *Artificial Intelligence*, vol. 23, no. 2, pp. 123–154, 1984.

[11] M. Sridhar, A. G. Cohn, and D. C. Hogg, "Unsupervised learning of event classes from video," in *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence (AAAI)*. AAAI Press, 2010, pp. 1631–1638.

[12] V. I. Morariu and L. S. Davis, "Multi-agent event recognition in structured scenarios," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011.

[13] W. Brendel, A. Fern, and S. Todorovic, "Probabilistic event logic for interval-based event recognition," in *IEEE Computer Vision and Pattern Recognition (CVPR)*, 2011.

[14] J. Selman, M. Amer, A. Fern, and S. Todorovic, "Pel-cnf: Probabilistic event logic conjunctive normal form for video interpretation," in *IEEE International Conference on Computer Vision Workshops*, 2011, pp. 680–687.

[15] C. Schwering, D. Beck, S. Schiffer, and G. Lakemeyer, "Plan recognition by program execution in continuous temporal domains," in *Proceedings of the 35th German Conference on Artificial Intelligence (KI'2012)*, Springer. Saarbrücken: Springer, 2012.

[16] A. Kreutzmann, I. Colonius, D. Wolter, F. Dylla, L. Frommberger, and C. Freksa, "Temporal logic for process specification and recognition," *Intelligent Service Robotics*, vol. 6, no. 1, pp. 5–18, 2013. [Online]. Available: http://dx.doi.org/10.1007/s11370-012-0122-2

[17] A. Skarlatidis, A. Artikis, J. Filippou, and G. Paliouras, "A probabilistic logic programming event calculus," *Theory and Practice of Logic Programming. Special Issue on Probability, Logic and Learning*, vol. (to appear), 2013.

[18] L. De Raedt, A. Kimmig, and H. Toivonen, "ProbLog: A probabilistic prolog and its application in link discovery," in *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 2007.

[19] E. T. Muller, *Commonsense Reasoning*. Morgan Kaufmann, 2006.

[20] H. Poon and P. Domingos, "Sound and efficient inference with probabilistic and deterministic dependencies," in *Proceedings of the Twenty AAAI Conference on Artificial Intelligence (AAAI)*, 2006.

[21] S. Chernova and M. Veloso, "Confidence-based policy learning from demonstration using gaussian mixture models," in *AAMAS '07: Proceedings of the 6th international joint conference on Autonomous agents and multiagent systems*. New York, NY: ACM, 2007, pp. 1–8.