# ROBIN: Activity Based Robot Management System

Adriana G Wilde, Pascal Bruegger, Benjamin Hadorn, Béat Hirsbrunner
Pervasive and Artificial Intelligence Group
University of Fribourg
Fribourg, Switzerland
{adrianagabriela.wilde, pascal.bruegger, benjamin.hadorn, beat.hirsbrunner}@unifr.ch
http://diuf.unifr.ch/pai

*Abstract*— **This paper examines the design and implementation of a system to support firefighting exercises based on tools and ideas of pervasive computing. Through this development a range of technologies and conceptual tools were used, such as sensors, wireless communication, and the programming framework *uMove*; as well as context-awareness, situation management, and activity detection. In this development, the notion of implicit human-computer interaction was of particular relevance. The use of a programming framework for interaction through motion was central, and this implementation is one of its proofs of concept, as it presented opportunities to improve it and recommendations to adapt it for future applications.**

***Keywords- Pervasive computing; sensors; motion-aware systems; activity-based computing.***

## I. INTRODUCTION

Firefighting involves the work of skilled personnel who are regularly required to take important decisions based on rapidly changing situations. They must take these decisions while performing strenuous physical activities using heavy equipment and uncomfortable protective clothing under life threatening conditions. From 1995 to 2007 there were 1,345 on-duty fire fighter fatalities in the United States alone [1], making this job one of the most dangerous of our times. Furthermore, research shows that the highest mortality factor in the profession is over-exertion or stress leading to on-the-job heart attacks, which account for 45% of the total deaths of US firefighters, many of which occur while performing non-emergency duties [2].

Among the many hazards firefighters face which might account for such elevated levels of stress, are chemical exposure, thermal injury and trauma, all of which potentially interfere with the assessment of the rapidly changing situations encountered during search and rescue operations, generally conducted in low-visibility and high-heat conditions. Primary search operations require moving as quickly as possible through the structure while being thorough, and although there is a variety of equipment available to firefighters to perform their duties, these may hinder rather than facilitate the gathering of information, as handling such equipment can be difficult because of its weight and volume.

Semi-autonomous robots can help the work of firefighters in collecting data. Their use in urban search and rescue operations is not novel. There is a wealth of research in this area, even more so after the terrorist attacks of September 11th in the United States [3], [4], [5]. This is the rationale and motivation for the ROBIN project.

The main goal of this work was the development of a context-aware application able to detect activity through motion. The application would interpret such activities by sensing the motions of firefighters, and would use semi-autonomous robots as tools to collect environmental data that could inform the relevant person about their situation.

In principle, and depending on the firefighters' movements within a building, the system would possibly control robots sent ahead of the team to gather information such as the state of the site (for example, temperature, the presence of smoke and/or dangerous gases in rooms explored) that might represent a potential physical danger for the rescue team. For example, consider a scenario where a fire fighter might be aware of an injured person trapped in a room in fire. By deploying a robot in advance to the room in question, the system has enough data to determine whether firefighters can proceed safely or whether there are dangers to be taken into account. This would prevent them from finding themselves in a critical situation unexpectedly, and by doing so, it would increase the knowledge about the incident, reducing the levels of stress and allowing them to perform the appropriate rescue operation.

Fig. 1 illustrates the various aspects taken into consideration in the development of this work: firstly, the flow of information related to the detection of motion, via sensors, to ascertain the activity that the firefighter performs at any given time. Secondly, the information flow carrying contextual data (such as temperature, the presence of smoke, gas, etc.), which will, together with the activity information, eventually inform the fireman about his current situation (in dashed lines) via a portable device. Thirdly, the control of the robots according to the current situation and the firefighters' activities. The layered system into which this information is either fed or produced, is the KUI (Kinetic User Interface) model, which the *uMove* architecture implements [6].

This document is organised as follows: Section II explains the division of work into modular tasks that could be developed independently. Section III deals with important concepts in pervasive computing which are the theoretical background of the project. Section IV describes each of the system components. Under section V, various implementation aspects are considered, including a description of the technologies that were used, and some technical considerations. Finally, in section VI, future work recommendations are offered with the conclusions of this work.

This work was partially supported by centeractive a.g., Bernstrasse 35, CH-3072 Ostermundigen and the SNF grants n° 116355 and 130095.

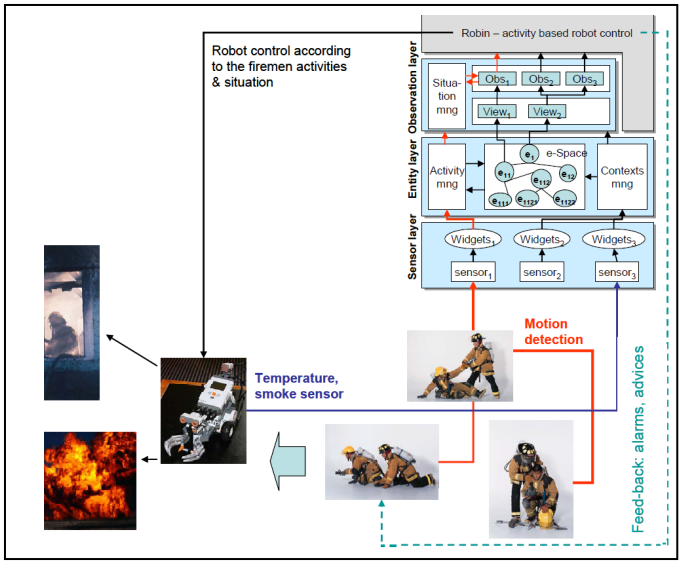


Figure 1. ROBIN information flow.

## II. PROJECT MANAGEMENT

As can be derived from the diagram in Fig. 1, this work has three different aspects, namely: motion and activity detection from sensor data, context-based situation analysis, and communication issues between robots and the application and between the application and the fireman. The work was divided in two parts. Our responsibilities included the situational analysis, that is, to define the situations that the application takes into account based on a user's detected activities and the context in which the action takes place and also the integration of this situation reasoning module within the application developed with the *uMove* framework.

Another issue was to ensure the appropriate feedback was provided to the users according to the detected situation, as well as the selection of an adequate communication protocol between the users and the application. Finally, we designed and implemented an algorithm producing a command to the robot as a function of the fireman's activity.

## III. PERVASIVE COMPUTING CONCEPTS

This section gives a brief discussion about the key definitions of pervasive computing as well as how they were applied in our work.

### A. *Entities, context, activities, situations, implicit interactions*

There are many definitions of context, as surveyed by Ensing in [7]. This diversity is justified by Dourish in [8]. We choose to conform to Dey and Abowd's definition: *"any information that can be used to characterize the situation of an entity. An entity is a person, place or object that can be considered relevant to the interaction between a user and an application, including the user and the application themselves"* [9]. For these authors, a system is context-aware *"if it uses context to provide relevant information and/or services to the user, where relevancy depends on the user's tasks."*

The concepts of activity and situation are closely interrelated with this definition, moreover, what Dey and Abowd call 'user's tasks' is the same as activities [9]. According to Li and Landay: *"An activity evolves every time it is carried out in a particular situation. A situation involves a set of tasks or actions performed under certain conditions. Activity-based ubiquitous computing affords an implicit interaction style. Interactions in activity-based computing are mainly based on input such as what people are doing as well as in what conditions (e.g., locations and times) an activity is conducted. These inputs are acquired implicitly (e.g., via sensors)"* [10].

These authors call 'conditions' what we call contextual information. Additionally, the idea of using future intentions as input for the system was appealing to us. In practical terms, we expected the robot to be able to explore the conditions ahead of the fireman, using solely implicit interactions in this process. Nevertheless, during early stages it became evident that explicit interactions were required to input the firefighters' plans. In other words, explicit commands to the robot were required at times, for example, when intending to change directions, or go into a new room. This is not necessarily a limitation, since occasional explicit interactions do not make the system less context-aware. In this regard, Li and Landay affirm: *"At the same time, activity-based computing does not exclude explicit or traditional interactions. Instead, explicit and implicit interactions can work together"* [10].

### B. *Robots as entities*

The first and most important point to make is that the robot is regarded as an entity. It is defined and handled as such in the application. However, it was not always this way, as this was a very contentious issue at early stages of the design. The robot could have been seen as a mere sensor carrier, reading environment settings which are associated with the various rooms in the building, or even as an extension of the fireman to whom it is assigned. This interpretation of the robot meant that there would not be involvement of robots in the situation analysis. Eventually it became clear that the robot should be considered as an entity, even if it has a special role. The reasons were two-fold: firstly, the robot can be seen as a "special kind of fireman", who is mobile and semi-autonomous, and its role is to help gathering information about potentially dangerous places. Secondly, being an entity, its location can be determined easily and the sensor readings can be stored as contextual information for this location. This is a more elegant solution, more scalable as it is independent on the actual number of robots and firefighters. In addition, it means that, being entities like firemen, robots can be as well in a situation (such as fully operational or damaged), which therefore can be incorporated in the situational analysis. Moreover, Scholtz recommended that systems controlling robots taking part in urban search and rescue operations needs to include a frame of reference to determine the position of the robot relative to environment (and therefore provide awareness of the robot's surroundings) [11]. This information is easily supported by *uMove* for all entities. Other recommendations that Scholtz offers include: indicators of robot health/state; information from multiple sensors presented in an integrated

fashion; the ability to self-inspect the robot body for damage or entangled obstacles; and automatic presentation of contextually appropriate information.

### C. *Fire as context*

On the other hand the fire is not an entity, despite being 'mobile' while it propagates. Unfortunately, fire is too difficult to characterize, as it behaves differently depending on the type of combustible materials, the amount of air available, the humidity, existence of draughts, and many other factors. Nevertheless, its existence can be inferred in presence of high temperatures, therefore it can be characterized as contextual information of a given spatial location. We can look at fire as a value tuple:

<heat, X,Y,Z> = Heat at the coordinates (X,Y,Z)

In other words, we consider heat as a context of the location, for which (X,Y,Z) are spatial coordinates (like the centre of the room).

## IV. SYSTEMIC ANALYSIS OF THE APPLICATION

### A. *System overview*

The system consists of two major components:

#### 1) *Mobile device for the fireman.*

This device is always with the fireman and shows urgent messages to him when required.

#### 2) *System controller.*

It is outside of the building in which the incident takes place but is used to survey the site. It shows the location and status (current situation) of the firefighter and the robot.

These components are linked through a WIFI connection. Both the remote visualizer and the visualizer on the system controller receive data through the Data Interface.

### B. *Situation Manager*

To evaluate situations we had to consider two different types of sensor values.

1. The sensor values belong to the context of the entity in question, as in Fig. 2.

2. The sensor values belong to the context of the environment of entity in question, as in Fig. 3.

In the first case the value is stored within the entity A and a message is sent to the viewer (a "lens" between the observed system and the observer) and all its observers. Each observer is now able to evaluate the new situation of the entity A. In the second case the value must be propagated to the location of the entity A, because it belongs to the context of the location.

Generally, for each change of context in a location, the situation of all entities at that location must be re-evaluated. For example, if the robot and the fireman are in the same room and the robot measures dangerous levels of gas in that room, the situation of the three entities (robot, fireman and location) might change, so it must be re-evaluated.

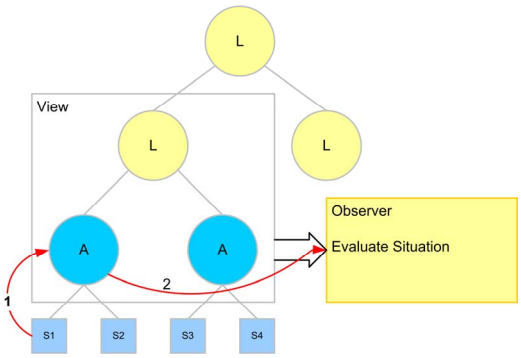


Figure 2. Sensor information belongs to the context of entity A. "L" are places (rooms, floor)

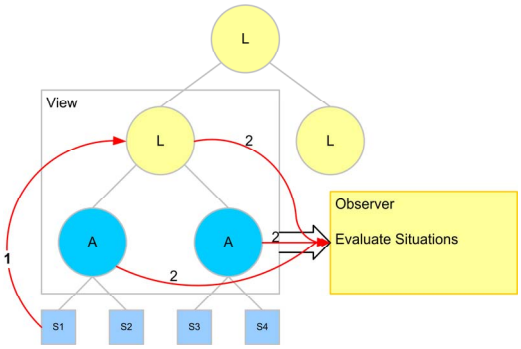


Figure 3. Sensor information belongs to the context of the environment of entity A.

#### 1) *Logic of Situation Manager*

We designed the situation manager by establishing logical constructs which would determine, via predicates based on contextual information and activities performed by the entities, in which situation any given entity is, in a manner similar to that proposed by Loke [12]. Even though Loke proposes situation programs based on sensor predicates, this does not exclude the consideration of activities into the predicate logic. In fact, although the terms 'activity' and 'situation' are distinct, we consider activity as a type of contextual information which can be used to characterize the situation of a person. Thus we can easily generalize to all entities capable of performing activities, i.e. including robots but not locations.

Based on our understanding of activities performed by firemen which could be identified with a simple motion detection system (such as the ones in [13] and [14]), we proposed originally four situations, each of which was a logical consequence of the evaluation of Boolean values of three different origins: sensorial information, an activity being performed, or even relationships between entities :

##### a) *Normal Situation*

Actions associated: None. Conditions of the situation: No other situation has been identified.

##### b) *Danger Awareness*

Actions associated: Inform firemen about probable danger in location L, in which currently there are no firemen. Proceed with caution. Localize danger with precision.

Conditions of the situation: (too_hot(L) v too_cold(L) v gas(L) v smoke(L)) ^ NOT_in(L).

NOT_in(L) becomes true when the location of the entity (say a fireman) is different from L

*c) In Danger Now*

Actions associated: The actor A (which could be a fireman or a robot) is in a location L in which a danger has been identified. Warn firemen about the type of danger, indicating time left before the situation becomes critical. Backtrack if it is a robot.

Conditions of the situation: (too_hot(L) v too_cold(L) v gas(L) v smoke(L)) ^ (NOT_too_long_in(A, L))

*d) Critical_Situation*

Actions associated: The actor A (which could be a fireman or a robot) is in a location L in which a danger has been identified. Warn firemen about the type of danger, indicating time left before the situation becomes critical. Backtrack.

Conditions of the situation: (too_hot(L) v too_cold(L) v gas(L) v smoke(L)) ^ (too_long_in(A, L))

A difficulty that arose was related to the determination of the spatial relationship between entities (whether robot and fireman are in a given location), which was required for the "Danger Awareness" situation. It became technically difficult to define a function of the distance between the fireman (for whom the situation applies) and the robot (sensing the danger in any of its presentations). Furthermore, this particular Boolean value was critical in differentiating the situations "Danger Awareness" and "In Danger Now". We have considered that all that is required to trigger "Danger Awareness" is to know whether there exists a location L for which any sensor reading has an abnormal value (and there are no firemen in L), despite knowing that this oversimplification renders the system less usable. In the case of there being a fireman F in L, he would be "In Danger Now". This left us with three situations, which we renamed: normal situation, potentially dangerous situation, and critical situation.

## C. Entity Tracker

The entity tracker is used to receive situation alerts and activities. It sends the situation alerts to the system controller and to the mobile device of the fireman. The three different situation states are handled differently, as shown in Table 1:

TABLE I. SIGNALS FOR THE SYSTEM COMPONENTS

| Component | Situations | | |
|---|---|---|---|
| | ***Normal*** | ***Potentially dangerous*** | ***Critical*** |
| System controller | Level blue | Level orange | Level red |
| Mobile device | Green (no message) | Warning message | Critical alert |

We kept the interface as simple and clear as possible in order to avoid the firemen wasting time to interpret the received information.

## D. ROBIN controller

The ROBIN controller manages the robot activities. Since the robot is fully controlled by the system control software, it is unnecessary to observe the activity of the robot entity by sensors in the way it is for the firemen's activities. The way the controller operates is by generating commands to the robot, or interactions, which could be either implicit or explicit, in compliance with Li and Landay's definition of activity based systems as presented in section III.

The implicit interactions are automatically executed by the software depending on the situation of the robot, and the activity of the fireman. The decision rules devised are detailed in Table II:

TABLE II. DECISION RULES FOR IMPLICIT INTERACTIONS WITHIN THE ROBIN CONTROLLER

| **Fireman activity** | ***Robot Situation*** | ***Command to robot*** |
|---|---|---|
| Being still (not moving) | normal<br>potentially dangerous<br>critical | stop<br>stop<br>backtrack |
| Crouching | normal<br>potentially dangerous<br>critical | go forward<br>go forward<br>backtrack |
| Running | normal<br>potentially dangerous<br>critical | explore<br>backtrack<br>backtrack |
| Walking | normal<br>potentially dangerous<br>critical | go forward<br>go forward<br>backtrack |

Since the robot control can be used also for explicit commands there is a priority handling between both kinds of interactions, allowing the fireman to control the robot even if the system fails to give the appropriate implicit commands to the robot for any reason.

# V. IMPLEMENTATION

ROBIN uses the programming framework *uMove*, sensors (SunSPOTs), robots (Lego MindStorm NXT), and portable devices (Glofiish X500). In the following subsections some of these will be considered in more detail.

## A. uMove

*uMove* is a JAVA based implementation of concepts of interaction through motion for pervasive systems as represented in the KUI model ([6],[15]) offering to programmers a standard platform to develop KUI enabled applications. As shown in Fig. 5, the framework is separated into layers in order to have a clear separation of concerns. The sensor layer contains all the *senget* representing the logical abstraction of the sensors connected to the system. The entity layer contains the logical representation of the physical users, places or objects being observed. The activity manager aggregates the motion events into activities and makes them available to the observer. Context changes received from the sensor layer are also reported to the observation layer. The observation layer analyses the current situation of the entity taking the current activity and the contexts of the entity.

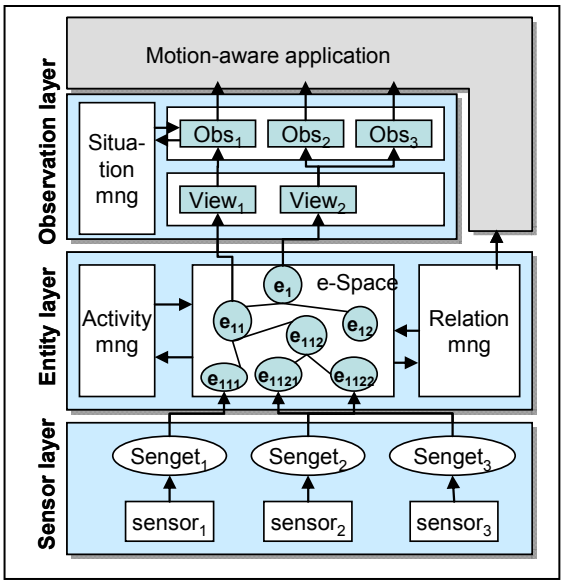


Figure 4. *uMove* framework

### B. Sensors

For motion detection, we used tri-axial accelerometers, as their use has proven useful in minimizing human intervention in ubiquitous computing applications [14]. Specifically, we used SunSPOTs (Sun Small Programmable Object Technology, [16]), which are small, wireless, battery-powered devices developed at Sun Labs. Powered by a specially designed small-footprint Java virtual machine, called Squawk, it can host multiple applications concurrently, requiring no underlying operating system. Stackable boards include application-specific sensors and actuators such as accelerometers, light detectors, temperature sensors, LEDs, push buttons and general I/O pins. We considered the following questions: how many Sun SPOTS should be placed on the fireman's body to detect and discriminate his motion? Where should they be placed? How fine should the sampling be in order to identify the type of activity the fireman is performing? Is there a particular type of activity that could be difficult to identify?

A first study investigating performance of recognition algorithms is the one by Bao and Intille [13], with multiple, wire-free accelerometers on a range of activities. In this work, five accelerometers were used simultaneously to discriminate motion. These were biaxial, and even so, 'the recognition performance dropped only slightly' when using just two, placed in thigh and wrist. The extra cost of providing more than double the amount of resources did not enrich the data significantly. Ravi, Dandekar, Mysore and Littman [14] place only one tri-axial accelerometer (the CDXL04M3, by Crossbow Technologies), choosing to place it near the pelvic region. This idea seemed to be also applicable to ROBIN, given that placing it on a wrist, as considered at one stage, could introduce unwanted noise to the data (e.g. jerky movements with the hand might be interpreted as 'running' when in fact the fireman might be walking or just standing still). However, we had considered placing it on the wrist and we even discussed the possibility of introducing explicit gestures as motion commands even though it would not be ideal. Furthermore, Ravi, Dandekar, Mysore and Littman imply that 'short activities', such as arm gestures might be recognised using accelerometer data but with greater difficulty than general activities spanning over longer periods [14]. Therefore, this solution, in addition to not being elegant, poses potential technical difficulties when it comes to interpret the data, so it was our decision to emulate Ravi et al, placing one Sun SPOT near the pelvic area of the fireman. The downfall of placing the accelerometer near the pelvis of the fireman is that activities that are limited to the movements of hands will be not recognised. Activities such as standing, walking, and running are well classified even under suboptimal settings, but it is reasonable to expect that activities such as lying down or crouching were more difficult to discriminate against standing, because of the nature of the sampling process. Acceleration in all axes would be zero after the first few seconds, which Ravi, Dandekar, Mysore and Littman discard anyway 'to minimize mislabelling', and is precisely within these few seconds when the change of state occurs (from standing to lying down or crouching, and vice versa).

However, the Sun SPOT has an accelerometer capable of a much finer sampling than the CDXL04M3, so it was reasonable to expect that the classification problems that Ravi, Dandekar, Mysore and Littman found with the mentioned activities were resolved by increasing the sampling rate.

### C. Portable devices

The Glofiish X500 is used as a mobile device. The application runs on a Java Virtual Machine designed for Windows CE called Mysaifu. Since the JVM of Mysaifu has some limitations on the graphic and communication libraries, we selected AWT for drawing a simple graphic user interface showing the alerts. The communication is performed via Wi-Fi (Layer 1) and IP/UDP as Layer 3 and 4. Since UDP (User Datagram Protocol) is a message oriented communication protocol each re-evaluation of the fireman-situation is sent to the device. This ensures that package loss does not lead to wrong system behaviour, in case of a missed situation alert, which is likely, as this very simple protocol does not guarantee data integrity. Each situation level is translated into a simple UDP datagram which is then sent.

### D. System controller and simulator

The system controller shows all entities, represented as coloured dots in the GUI, within the correct locations. The colour of the entities shows their current situation:

- normal situation: blue
- potentially dangerous situation: orange
- critical situation: red

### E. Tests and evaluation

During the development of the project, we were able to test each component under development. Two formal opportunities were created to perform evaluation of the whole project: the first evaluation was Lo-Fi as described extensively in [15], allowing each team of developers the chance to subject their

algorithms to a test scenario, enabling the developers to detect some logical errors, potential problems, and also to think about issues not yet considered, such as the sensitivity of the system to appropriate thresholds. One problem of this testing mechanism is that it required a certain amount of time, as well as concentration, because the probability of introducing human errors was very high.

The second evaluation was Hi-Fi, after both teams completed their developments and were ready for an integration of the systems. These evaluations allowed some adaptation of the *uMove* framework. The following changes were made:

*1) Definition of a new role: Sensor Carrier*

An entity can be a sensor carrier, if the mounted sensors are observing the environment (location). It implies that the sensors are not necessarily attached to the location.

*2) What is observed, is it the environment or the entity itself?*

The distinction has to be made in order to set the context values to the correct entity. If a temperature change is recognized (by a sensor mounted on the robot) the context must be stored with the location (environment).

*3) Data propagation.*

If a room temperature has changed, the situation of all sub-entities might change (robot and fireman situated in this location). In order to do that for each entity the situation has to be re-calculated.

## VI. CONCLUSIONS AND FUTURE WORK

In this paper, we presented a project that can help firefighters during their activities to take decision in rapidly changing environment. We proposed a context-aware system in which a semi-autonomous robot sent ahead the firemen team could gather useful information (temperature, smoke, gas) through its sensors. The information is analysed by the situation manager which evaluates the situation of the firemen according to his current activities and contexts. We described the types of situation that were taken into consideration and we discussed the technical difficulties of detecting the event of a fireman with the technology we used. We also presented the different software and hardware components of the project and technically described them and motivated our choices. Finally, we presented two different evaluations which we carried out. The first one was a Lo-Fi evaluation which consisted of regrouping the development teams around a table for a session and let them play the algorithms designed for each component. This test allowed us to correct the design and see if the inter-component flow of information was correct. The second evaluation (Hi-Fi) was the physical test running the robot and the software modules.

The first version of the project showed that the concept of firefighters' observation and situation analysis is interesting and useful. It showed also problems that make this project as such not applicable to a real case. For instance one problem with the system, as encountered during the Lo-Fi testing, is that it is too sensitive to the threshold values chosen, because the predicate logic for the situation manager relies on Boolean variables that take values according to real variables reaching (or not) the prefixed thresholds values. Incorporating fuzzy logic may remove or diminish this oversensitivity and make the system more robust. Also the robot is one of the major problem to solve since, in our test, it got blocked and was unable "to get back to work". As that stage, an alternative should be explored in order to assure reliable information for the situation manager and therefore guarantee a correct information transmitted to the firemen on the field.

## REFERENCES


[1] J. Brown and J. Stickford. Physiological stress associated with structural firefighting observed in professional firefighters. Tech. Rep., Indiana University, Firefighter Health & Safety Research, School of Health, Physical Education & Recreation, Dept. Kinesiology, 2008.

[2] S.N. Kales, E.S. Soteriades, C.A. Christophi, and D.C. Christiani. "Emergency duties and deaths from heart disease among firefighters in the United States". New England Journal of Medicine, 356(12):1207–1215, 2007.

[3] J.L. Burke, R.R. Murphy, M.D. Coovert, and D.L. Riddle. "Moonlight in Miami: Field study of human-robot interaction in the context of an urban search and rescue disaster response training exercise". Human-Computer Interaction, 19(1):85–116, 2004.

[4] F. Driewer, M. Sauer, and K. Schilling. "Discussion of Challenges for User Interfaces in Human-Robot Teams". In 3rd European Conference on Mobile Robots (ECMR '07), pages 1–6, Freiburg, Germany, sept 2007.

[5] J. Scholtz, J. Young, J. Drury, and H. Yanco. "Evaluation of human-robot interaction awareness in search and rescue". In IEEE International Conference on Robotics and Automation, vol.3, pages 2327–2332, 2004.

[6] P. Bruegger, B. Hirsbrunner, Kinetic User Interface: Interaction through Motion for Pervasive Computing Systems, in: Parallel session "Designing for Mobile Computing", Springer, HCII , San Diego, 2009

[7] J. Ensing. Software architecture for the support of context aware applications. Preliminary study, 2002.

[8] P. Dourish. "What we talk about when we talk about context". Personal and ubiquitous computing, 8(1):19–30, 2004.

[9] A.K. Dey and G.D. Abowd. Towards a better understanding of context and context-awareness. 1999.

[10] Y. Li and J.A. Landay. "Exploring activity-based ubiquitous computing: interaction styles, models and tool support. Computing, 11:13.

[11] J. Scholtz, J. Young, J. Drury, and H. Yanco. "Evaluation of human-robot interaction awareness in search and rescue". In IEEE International Conference on Robotics and Automation, 3:2327–2332, 2004.

[12] S.W. Loke. "Representing and reasoning with situations for context-aware pervasive computing: a logic programming perspective". The Knowledge Engineering Review, 19(03):213–233, 2005.24

[13] L. Bao and S. Intille. "Activity recognition from user-annotated acceleration data". LNCS, pages 1–17, 2004.

[14] N. Ravi, N. Dandekar, P. Mysore, and M.L. Littman. "Activity recognition from accelerometer data". In Proceedings of the Seventeenth Conference on Innovative Applications of Artificial Intelligence(IAAI), pages 1541–1546. AAAI Press, 2005.

[15] P. Bruegger, A. Lisowska, D. Lalanne and B. Hirsbrunner "Enriching the design and prototyping loop: A set of tools to support the creation of Activity-based pervasive applications" Journal of Mobile Multimedia *(in press)*, 2010

[16] R. Goldman. Using the LIS3L02AQ Accelerometer: A Sun SPOT Application Note. Sun Microsystems. Technical report, Sun Microsystems, 2007.Weiser, M. "The Computer for the 21st Century" ACM SIGMOBILE Mobile Computing and Communication Review, 3(3):3–11, 1999.