

# The Keystone Fire Brigade 2004

Jacky Baltes, John Anderson, Shawn Schaerer, and Ryan Wegner

Autonomous Agents Laboratory  
Department of Computer Science  
University of Manitoba  
Winnipeg, Manitoba, R3T 2N2, Canada  
jacky.andersj@cs.umanitoba.ca

**Abstract.** The Keystone Fire Brigade is a robotic rescue team that has previously competed in competitions at RoboCup (Fukuoka, 2002; Padua 2003), AAAI (Edmonton, 2002), and IJCAI (Acapulco, 2003). The key elements of our approach are an emphasis on vision, a fully autonomy solution, and an implementation on inexpensive robot bases. This paper describes the version of the team that will be appearing at RoboCup-2004. We overview the hardware employed, methods used for visual processing, map-making and victim identification. We also describe the experiences we had in the test domain and offer some recommendations on future competitions.

## 1 Introduction

Robotic rescue is both a worthwhile application for artificial intelligence and a challenge problem that allows solutions to be compared in a controlled setting. Because of the extreme difficulty associated with the problem, most entries nowadays are teleoperated.

We believe that, ultimately, autonomous processing will be of great importance in robotic rescue. In rescue settings, issues such as operator fatigue, lack of situational awareness of the operator, cognitive load on the operator, and the number of individuals an operator can control in real time [1, 2] all place limitations on human control.

We also believe that a focus on autonomous processing is important from the standpoint of truly advancing artificial intelligence: the reason that most entries in the competitions are teleoperated is precisely because autonomous control is still very primitive in an areas as complex as robotic rescue, and avoiding the use of autonomous control mechanisms does not do anything to improve this technology. We believe that once autonomy has improved, teleoperation can be added to fill in the gaps where a human operator can be helpful to the situation without being overwhelmed. We have thus been focussing on autonomy as a key focal point in our work in robotic rescue.

This paper describes the Keystone Fire Brigade the University of Manitoba's entry in the RoboCup-2004 Robotic Rescue Competition. Our approach embodies several principles we believe will ultimately be important in successful robotic rescue problems: autonomy, a multiagent perspective, and parsimony.

Many problems are naturally amenable to solution through a collection of distributed agents. A multiagent approach to robotic rescue is a natural one: it is the approach that is already taken by humans in this problem; it allows the power of numbers

to deal with the geographic distribution of the problem; and the presence of multiple agents allows any individual in this domain to be considered more expendable than would otherwise be the case. Because of this, we want to design agents that are meant to operate in groups.

Parsimony is an important goal in all robotics applications. Any additional feature has a cost, both financially and in terms of computing power and other local resources, and reliability. If a feature is not necessary to solve the problem, eliminating it provides more resources to those features that are necessary. We believe, like others [3] that the addition of any component should be carefully considered. Cost must be balanced with the efficacy of the equipment to the improvement of overall system performance. Parsimony is also one of the reasons that a multi-agent approach is important - by taking the same resources and spreading them among a number of simpler agents, the interaction of these over a geographic area can deal with the problem better than a single, highly complex agent. The more parsimonious the agent design, the more expendable any individual robot can be considered as well.

The remainder of this paper details the hardware platforms employed in this year's Keystone Fire Brigade the use of optical flow for localization and mapping, and discusses our experiences in this year's competition.

## **2 Team Members and Their Contributions**

We have a variety of people who have contributed to this year's team:

- Jacky Baltes: Team leader, lead designer, programmer
- John Anderson: Design, support
- Shawn Schaerer: implementation, optical flow, mechanical design
- Ryan Wegner: support, ego motion estimation
- Terry Liu: Programming

## **3 Operator Station Set-up and Break-Down**

The Keystone Fire Brigade is an entirely autonomous team. It requires no operator. Set up includes loading code into the embedded systems, but is easily done before the allotted set-up time. The team itself has no robots larger than a breadbox, and so is easily ported to the disaster site.

## **4 Communications**

Because the Keystone Fire Brigade is an entirely autonomous team, we use no communication between the robots and a human operator. While it is entirely possible to have communication ensue between robots, we currently do not implement this facility. There are thus no issues in communication compatibility or interference with other teams in the competition.

## 5 Robot Hardware and Locomotion

We have two primary motivations for hardware design in rescue robots. The first of these is reliance on extremely simple robotic platforms. Ultimately, the task of robotic rescue will benefit from implementation on inexpensive platforms, since larger teams can be afforded and individual robots can be viewed as being expendable in a dangerous environment. Our motivation in using simple hardware, however, is to force reliance on more robust and versatile control methodologies. A system relying heavily on accurate odometry, for example, is severely crippled under conditions where odometry is inaccurate. A system that does not assume the availability of accurate odometry readings, however, will still operate under such conditions, as well as in conditions where odometry can be useful.

The second major design factor is an emphasis on vision. Each of our robots employs simple low-power CMOS cameras or webcams, and has enough local processing power for vision and robot control. Vision is the only sense employed by all of our robots.

In the 2004 version of the Keystone Fire Brigade the robots will be based on hand-constructed corrugated posterboard bases (as illustrated in Figure 1) which we first employed at AAI-03 in Acapulco. Two servos attach to drive the left and right wheels, and a webcam is integrated for vision. The servos were modified by cutting out the motion stop and thus provide relatively nice velocity control. To support autonomous processing, platform carries a VIA mini-ITX microcontroller board with a 533 MHz Eden (x86 compatible) processor, 256 MB RAM, and a 256 MB flash card. We developed a mini Linux distribution (based on the Debian Linux distribution), which fits into this small space.

## 6 Sensors for Navigation and Localization

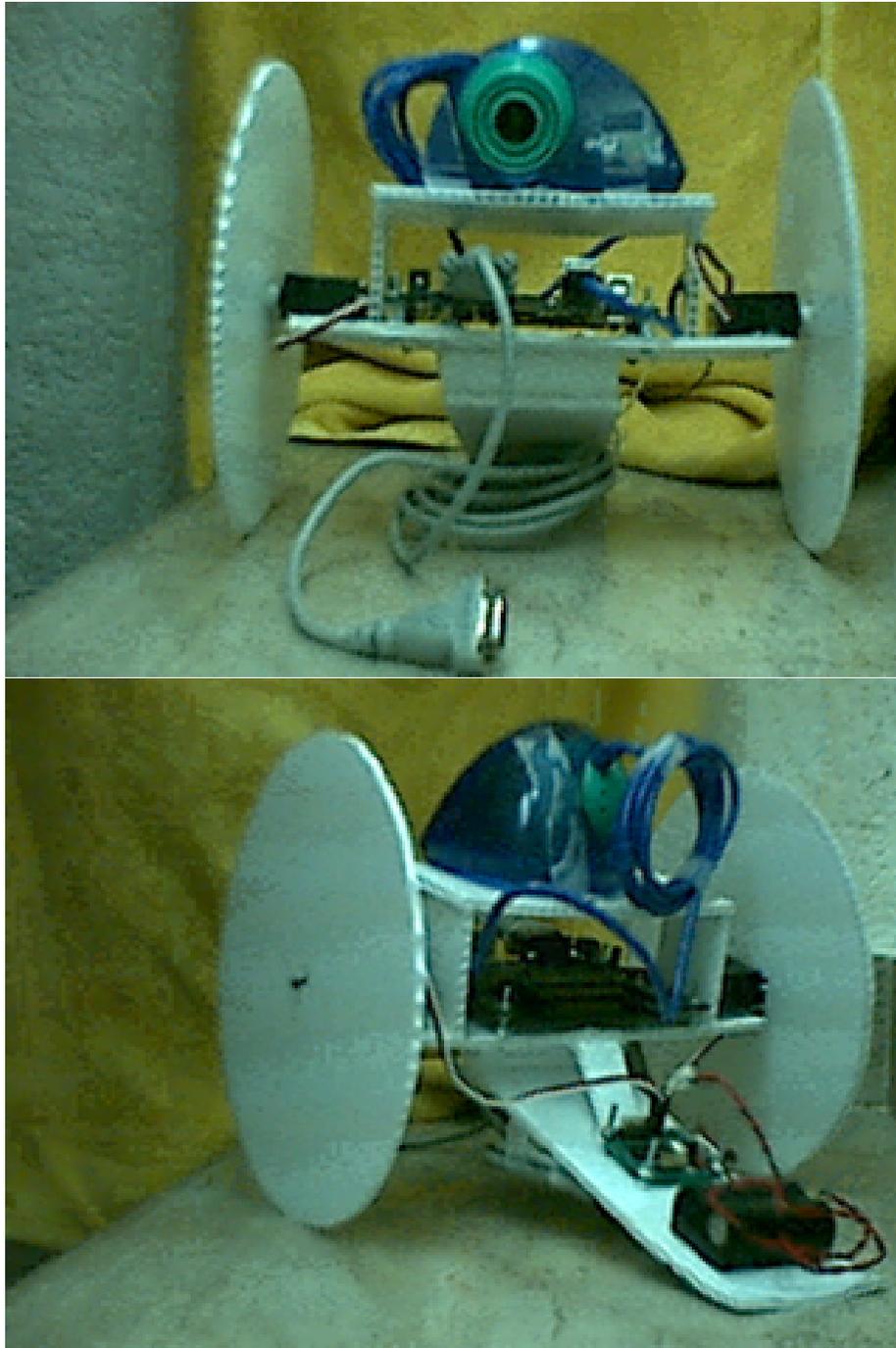
The greatest challenge on platforms such as those employed by the Keystone Fire Brigade is the design and implementation of pragmatic algorithms for intelligent visual processing, and the adaptation of these to the low frame rates that are achievable using the embedded systems driving the robots. This is the main contribution of our team.

The use of vision as the only form of sensing requires that vision not only be used to identify victims, which is the primary use of vision for most teams, but also to allow the robot to localize and map the environment. The following subsections describe our methods for dealing with each of these elements.

### 6.1 Ego Motion Estimation

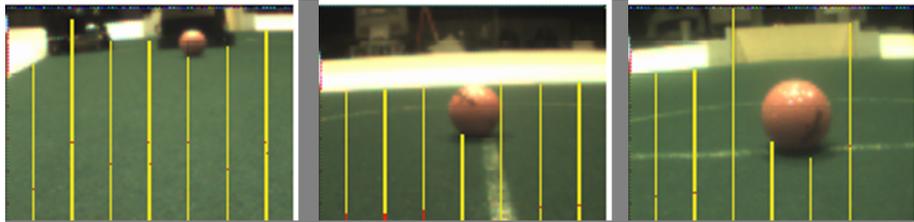
In order for a robot using only vision to map an environment, its progress through the environment must be measured by vision rather than by odometry, sonar, laser, or a combination thereof. This is the problem of ego motion estimation, which requires the robot to estimate its movement in terms of distance and angle by examining the differences between visual frames. This is an extremely difficult problem, mainly because of

**Fig. 1.** Robot platform using a simple corrugated board base (front and rear views)



the amount of confusing or conflicting information in visual frames. Furthermore, in the rescue environment itself, there is significantly less structure than that available in other common robotic environments. In soccer for example, the detection of walls can be done by plotting lines from the bottom of the image up to a white colour change (figure 2), and then calculating the wall's likely position based on the end points of the plotted lines [4]. While some consideration has to be made for the portions of the wall that are not visible (i.e. obscured by the ball or other players), there is generally enough information in most frames to plot the wall's position quickly. In more general environments such as the rescue domain, however, the structure that can be assumed in simpler environments does not exist. Instead we must employ more sophisticated edge detection algorithms (upper part of Figure 3) followed by a line detection algorithm (Hough transform), illustrated in the lower part of Figure 3.

**Fig. 2.** Detecting Walls in Soccer



While every recognized line in the rescue domain is not a wall, we can still employ these regular features to do ego-motion estimation. Our approach uses the optical flow between images to estimate the motion of the robot (Figure 4), and also to indicate a lack of motion on the part of the robot (i.e. detecting when the robot is stuck).

If a recognizable pattern (a set of lines, intersections between lines, etc.) can be recognized in two different frames, we can compute the change in angle and distance on the part of the robot that produced that change in visual reference point. Note that we assume that the line is at a constant height (e.g., a line on the floor).

Figure 5 shows the geometry of the situation. Assuming that the robot can determine the angle between itself and a line, then the change in orientation  $\delta\theta$  can be easily computed by the difference in angle.

In the case of differential drive robot, this allows one to calculate the *difference* between the right and left wheel velocities (assuming the width of the robot is known). In the case of a rear-wheel or front wheel drive car, the steering angle can be computed (assuming the axle distance of the robot is known).

The change in angle of the line does not allow one to solve for right and left wheel velocities (in the case of a differential drive robot), or the linear velocity (in case of a car-like robot). However, given that the robot can also determine the distance between the robot and the wall, solutions to the kinematic equations can be found and the motion can be recovered. The geometry and solution is shown in Fig. 6.

Fig. 3. Detecting Lines in the Rescue Environment

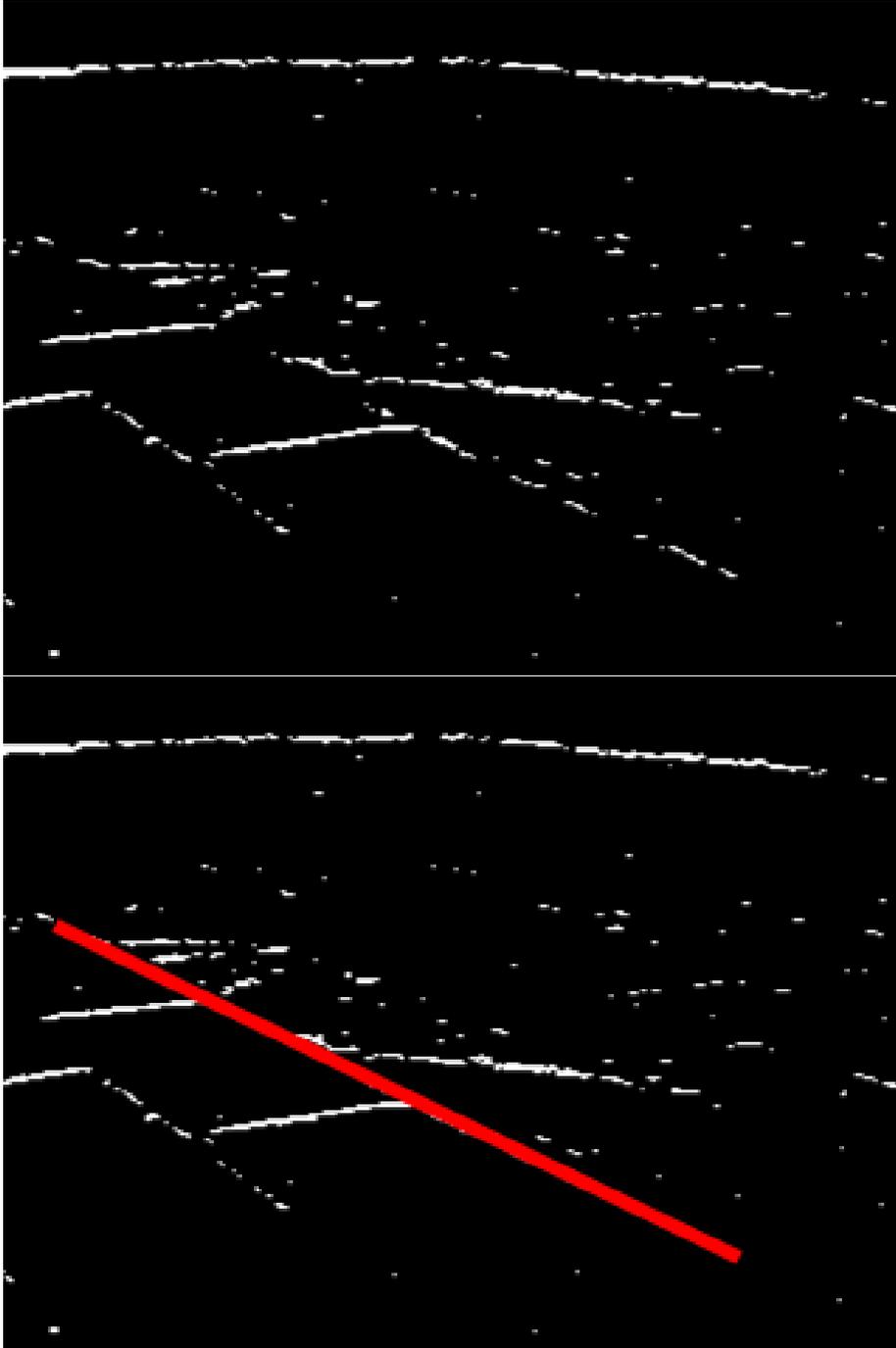
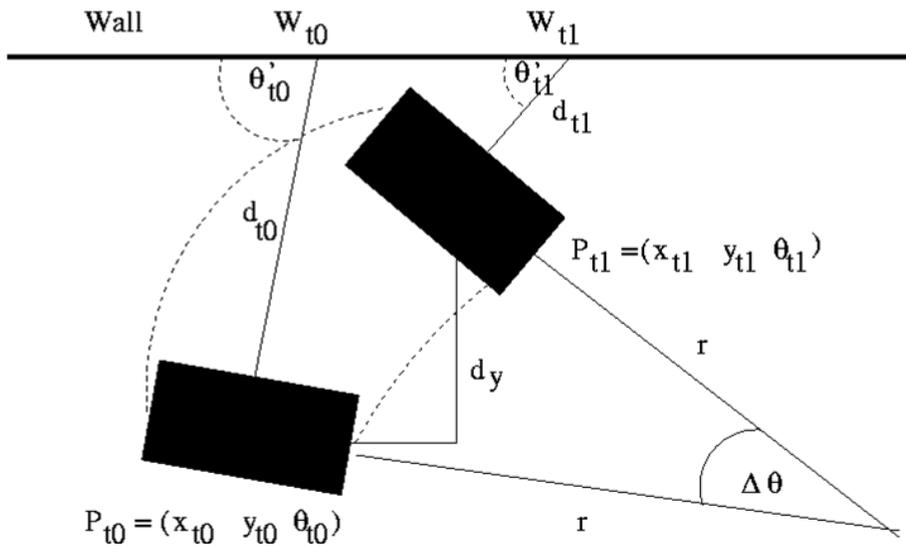


Fig. 4. Ego Motion Detection from Visual Frames

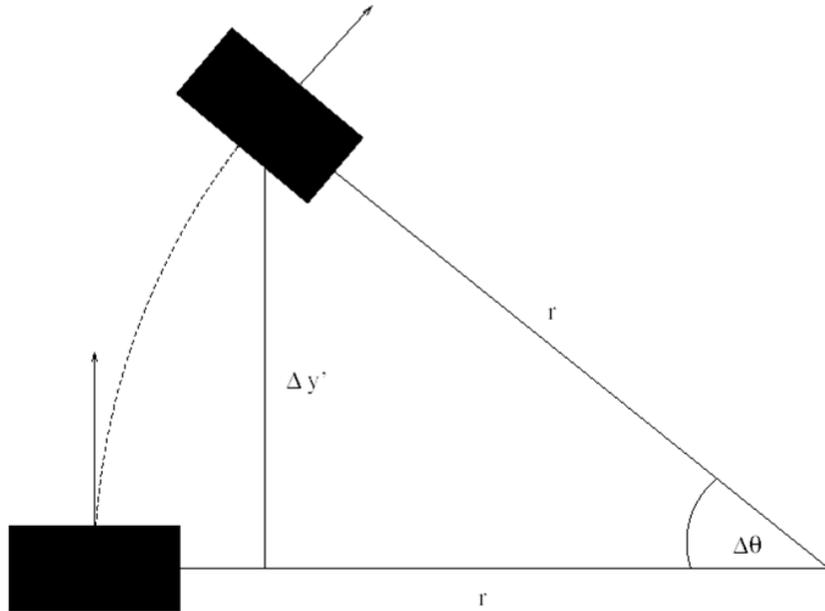


Fig. 5. Determining the change in angle from two visual reference points



To determine if the robot is blocked or otherwise stuck in one position, the image is broken up into 16 equal sized sub-images. Of these, only the bottom 8 sub-images need to be considered - everything else is further away and would not be likely to provide useful feedback regarding the motion of the robot. The system then computes the differences between the current and the previous image for each quadrant. The colour difference is defined as the sum of the absolute value of the differences in the red, green, and blue channels. If the difference in a sub-image is above a threshold, the quadrant is marked. If there are more than eight marked sub-images and the motors were turned on in the previous time step, than the system signals that the robot is stuck or blocked. We break the image into sub-images to allow for localized changes due to the motion of some other agent or other external motion in the image, to try to limit the number of false positives.

Fig. 6. Determining the distance travelled from two visual reference points



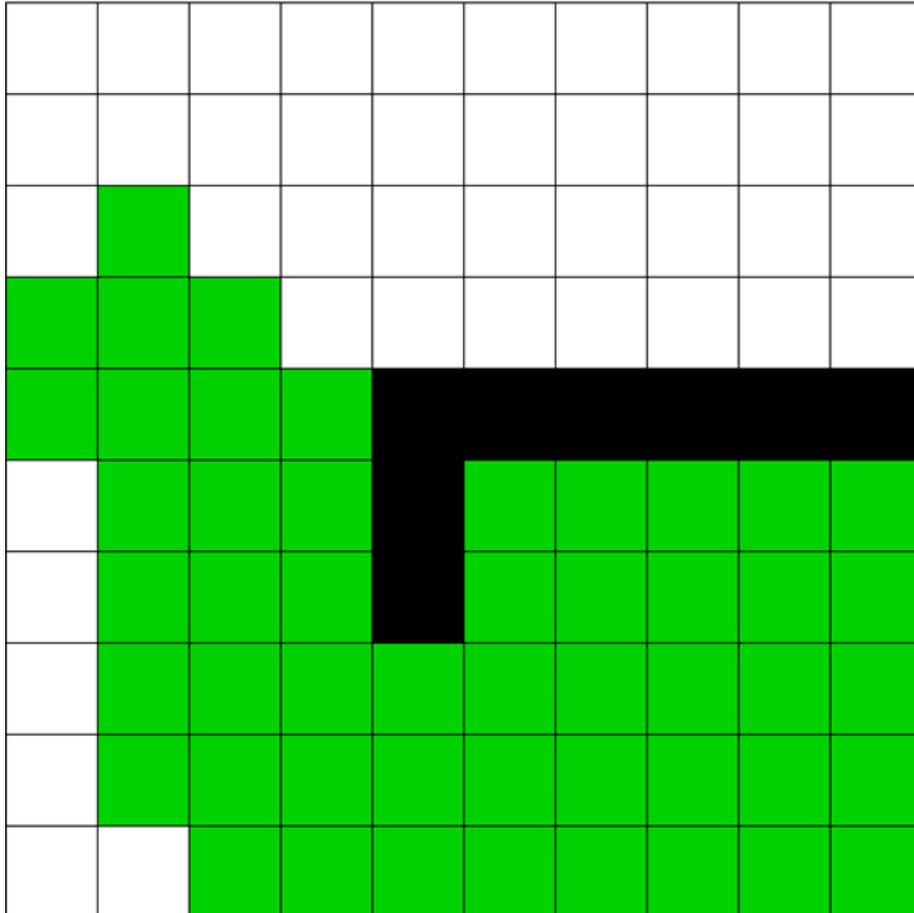
## 7 Map Generation

Just as we rely solely on vision for localization through ego-motion detection, we also rely on vision for constructing a map while localizing through optical flow. This results in a chicken-and-egg problem: While localization becomes easier as maps are created, we must begin out of necessity with no valid initial map, making localization difficult, which in turn complicates the process of constructing an accurate map.

Our approach to building a map involves the construction of sets of local two-dimensional maps. The robot makes a map of the immediate area around itself (1m x 1m), storing the map as an occupancy grid such as that shown in Figure 7. In this map, the robot has plotted an obstacle (in black) and an open area (in green), while the white areas represent unexplored areas. These local maps are separated by longer traversals (a random walk in a particular direction) and are linked together as topological maps. The distance and length of a traversal serves as a link between maps, but as new features are detected earlier maps are studied for these features, allowing local maps to overlap. The smaller size of the local maps allows the local area to be explored quickly, and the traversals between allow the robot to map different areas without errors in one map compounding to cause problems in later maps.

We plan in the future to extend this work to include a case-based reasoning system that employs typical sensor readings (especially “typical” images of the area) to identify areas and to connect them via topological paths.

Fig. 7. Local Map



### 7.1 Detection of Victims

While the system for awarding points is strongly oriented toward the use of multiple forms of sensing, most victims in the NIST testbed are reasonably easily identified visually. While there is much current work on the visual detection of victims, we are attempting to work with a reasonably simple, pragmatic approach to this difficult problem that is computationally viable for small embedded systems.

Our victim detection approach uses both colour as well as shape information. Flesh colored spots are marked as possible victim locations (these algorithms were trained beforehand on flesh patches of team members in the lighting used in the test arena).

We have developed a 12 parameter colour model which uses Red, Green, and Blue as well as three difference channels: Red - Green, Red - Blue, and Green - Blue. The

differences are included in this parameter model because of their tendency to remain relatively constant in different views of objects of the same colour despite of lighting variations over a field. This approach is the same used in our Doraemon vision server [5] and has proven itself in years of robotic soccer competition.

Currently, we use a simple blob detection scheme. The system signals that it has found a victim by calculating the apparent size and aspect ratio of a skin coloured blob. If these parameters are within limits, the system signals the detection of the victim by performing a series of 360 degree turns at the current location. It then continues to search the environment.

## **8 Human Factors**

One of the advantages of an autonomous system is ability to avoid many issues of human-computer interaction. Our system requires no training to actually operate, since there is no human operator.

## **9 Practical Application to Real Disaster Sites**

While this base is extremely cheap and portable, this is not robust enough to employ in a real disaster setting. However, it is sufficient to demonstrate the approaches we believe are useful for future robotic rescue work, and can navigate within the yellow zone of the NIST testbed [6]. Our intent is to demonstrate the use of vision and autonomous control this task, and to further applied research in these areas in robotic rescue, rather than to tackle the less stable terrains in the orange and red arenas.

## **References**

1. Casper, J.: Human-robot interactions during the robot-assisted urban search and rescue response at the world trade center. Master's thesis, University of South Florida (2002)
2. Casper, J., Murphy, R.: Workflow study on human-robot interaction in usar. In: Proceedings of the IEEE International Conference on Robotics and Automation. Volume 2., Washington, DC, IEEE (2002) 1997–2003
3. Balch, T., Arkin, R.C.: Communication in reactive multiagent robotic systems. *Autonomous Robots* **1** (1994) 27–52
4. Baltes, J.: Localization for mobile robots using straight lines. In: Proceedings of Seventh International Conference on Control, Automation, Robotics and Vision. (2002) to appear.
5. Anderson, J., Baltes, J.: The Doraemon User's Guide. Department of Computer Science, University of Manitoba, Winnipeg, Canada. (2002) <http://robocup-video.sourceforge.net>.
6. Jacoff, A., Messina, E., Evans, J.: Experiences in deploying test arenas for autonomous mobile robots. In Messina, E., Meystel, A., eds.: Proceedings of the 2nd Performance Measures for Intelligent Systems Workshop (PerMIS). NIST Special Publication 982, National Institute of Standards and Technology (2001) 87–95