RoboCup2004 – US Open
Rescue Robot League Competition
New Orleans, LA, USA
April 24 – 27, 2004
www.cs.uno.edu/~sheila/americanopen04

RoboCupRescue - Robot League Team
Team Corky, United States

Authors listed alphabetically: Steve Burion[3], Michael Coblenz[1], Jeffrey Gennari[2], Mary Koes[1], Michael Lewis[2], Tomasz Loboda[2], Michael Manjlovich[2], Illah Nourbakhsh[1], Kevin Oishi[1], Jumpol Polvichai[2], Katia Sycara[1], Jijun Wang[2], Mark Yong[1]

[1] Carnegie Mellon University
The Robotics Institute
5000 Forbes Ave.
Pittsburgh, PA USA 15213
usar-list@cs.cmu.edu
http://www.cs.cmu.edu/~softagents/project_grants_NSF.html

[2] University of Pittsburgh
Computer Science Department
ml@sis.pitt.edu
http://www.pitt.edu/~cmlewis/

[3] Ecole Polytechnique Fédérale de Lausanne (EPFL)
VRAI Group
IPR-LSRO
Lausanne, Switzerland
steve.burion@epfl.ch
http://imtsg7.epfl.ch

Abstract. Heterogeneous teams of robots, cyber agents, and people working can work together to search an urban disaster site much more safely, efficiently, and quickly than human rescue workers alone. As a step towards realizing this goal, our RoboCup 2004 U.S. Open Rescue Robot team consists of two robots with different capabilities, a human operator, and an interface agent designed to facilitate interaction between robot and human teammates. Both of the robots, Corky and PER, were designed to be portable, robust, and comparably inexpensive. Corky is large enough to carry an advanced sensor suite for victim detection while the PER is able to navigate smaller spaces. Both robots have limited autonomy in order to facilitate the task of the human operator. This paper details the robot mechanics, sensing algorithms, user interface, mapping algorithms, and operator training that enable Team Corky to locate victims in the competition disaster arena.

Introduction

In the event of a natural or man made disaster in an urban environment, rescue workers must race against time to search buildings and rubble for victims. This task is currently undertaken primarily by humans and trained dogs. To minimize the risk of human life due to chemical or biological agents, fires, and additional collapses, robots could be used to explore the space, look for victims, and determine whether or not an area is safe for human rescue workers. As a step towards this goal, competitions like this RoboCup US Open Rescue Robot Competition allow researchers to test their robots, sensors, and software in a representative but controlled environment. Team Corky has built two wheeled robots with different locomotive and sensing capabilities, Corky and the PER, and a user interface designed to allow a single operator to guide both robots on their mission of mapping the environment and locating victims. This paper describes our team strategy, the mechanics and electronics of both robots, our user interface, and algorithms for victim identification, localization and mapping.

1. Team Members and Their Contributions

• Katia Sycara	Advisor
• Illah Nourbakhsh	Advisor
• Michael Lewis	Advisor
• Steve Burion	Human detection sensor development
• Michael Coblenz	Localization/Map building
• Jeff Gennari	Interface support
• Mary Koes	Systems engineer
• Tomek Loboda	Human interface design
• Joseph Manjlovich	Localization/Map building
• Kevin Oishi	Software development
• Jumo Polvichai	Robot software
• Jijun Wang	Simulator
• Mark Yong	Mechanical and control development
• National Science Foundation	Sponsor
• Intel Corporation	Sponsor

2. Operator Station Set-up and Break-Down (10 minutes)

While our robots are designed to be broken down and transported as checked luggage on a commercial aircraft and could be likewise carried in parts to the disaster site, we plan to have our robots fully assembled before deployment. The smaller robot, PER, can be easily carried by a single person to the site entrance while the larger robot, Corky, can be rolled like a dolly by using the tail as a handle. The operator will run a quick test to confirm that both robots are operational and then return to the operator station where we will us a table and chair for the main control station, a laptop computer.

3. Communications

We currently use 802.11b wireless Ethernet on a peer-to-peer network which does not require access points. We also use an analog transmitter and receiver in the 900MHz range for the transmission of infrared images. Based on the constraints of this competition, we will make every effort to switch to 802.11a Ethernet.

Effective communication is one of the most important aspects of search and rescue robotics. In order to facilitate communication between different systems, allow dynamic team coordination on a large scale, and manage information in a principled way, we use the RETSINA [8] multi-agent system architecture (see figure 1). Agents can locate other agents by name using the Agent Name Service (ANS) or by capability using Matchmaker [9]. Interagent communication in RETSINA uses the KQML agent communication language. To improve efficiency, we have extended the RETSINA architecture with backchannels [1], direct simplex connections between agents, for low level information, i.e. video, state information, teleoperation imperatives. Although the autonomy limitations of the robots currently limits the extent to which we can draw on the power of the RETSINA architecture, we feel it is important to adopt this technology from the beginning in order to facilitate future work on teamwork and coordination.

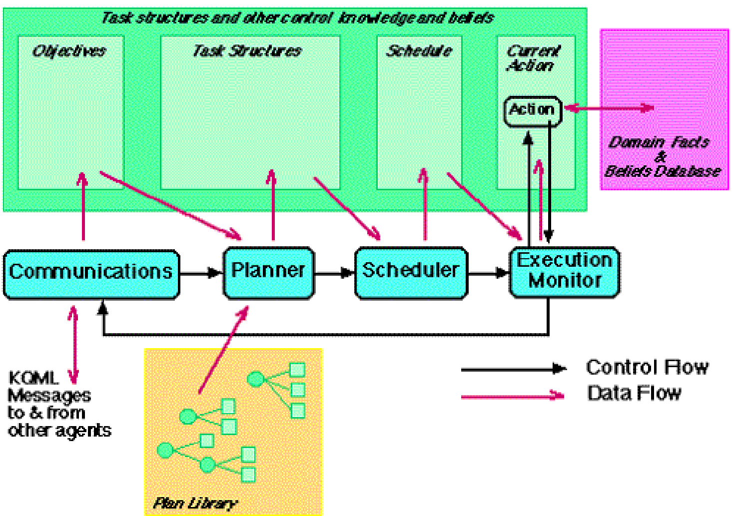


Fig. 1. The RETSINA Agent Architecture. The current system uses only the communications module.

4. Control Method and Human-Robot Interface

Our team consists of two heterogeneous robots with limited autonomy controlled by a single human operator. We have identified the following taxonomy of control for our system.

4.1 Operator Tasks The operator will be controlling two robots at the same time. Operator's tasks will include:

- *On-line Teleoperation*
  The operator will use a control device of their choice (mouse, joystick, keyboard) to give robot immediate instructions on low level actions to be taken, e.g. move, rotate, rotate camera etc.

- *Quick Teleoperation*
  The operator will give a robot an instruction which robot will then execute without operator's attention needed. An example of such instruction is "move forward for 4 meters".

- *Triggering robot-specific actions*
  The operator will trigger short- and long-lasting operations, set of which will be different among robots. Examples include taking a panoramic picture or searching for life in the nearest environment (gathering and analyzing information from the immediate environment with in context of life signs).

- *Creating and running Robot Control Scripts*
  The operator will be able to write scripts consisting of a set of atomic commands. For example, a script can instruct the robot to (1) move forward for 5 meters (to reach the T-intersection of two corridors), (2) check if the right corridor is longer than 2 meters, (3a) stop if it is, or (3b) move another 5 meters otherwise.

- *Maintaining the Environment Map*
  The operator will be assisting robots in generating their environment map (see Section 5).

The operator may cancel a robot's task at any time.

4.2 Robots Tasks Our robot team consists of Corky and PER, two robots with different capabilities. Since the robots are not homogeneous, they share some tasks but not others:

- *Performing Low Level Actions*
  The robot will react to immediate operator instructions requesting movement, rotation, camera rotation etc.

- *Performing Medium Level Actions*
  The robot will perform longer actions autonomously. An example of such action would be moving forward for 4 meters while avoiding obstacles.

- *Performing Robot Specific Actions*
  The robot will perform a set of actions autonomously, like taking a panoramic picture or searching for life in its nearest environment.

Upon completion of a task a robot notifies the operator. During long-lasting tasks the operator receives a series of feedback data from the robot, containing information on the current progress. For example, the robot will report every 50cm while traveling 4 meters. If the robot needs an operator's assistance it halts its current task and sends an appropriate message to the operator. We distinguish two levels of urgency here: (1) critical assistance request and (2) non-critical assistance request. The first communication occurs when the robot found life signs while performing the current task, such as spotting a human head while taking a panoramic picture. The second communication can happen when the robot was instructed to take a specific position, but it is unable to reach the designated destination because of an obstacle.

4.3 User Interface
The user interface will be constructed form the following elements:

- *Robot ID Area*
  Since more then one robot will be operated at the same time it is essential for the operator to always be sure which one of them is currently being controlled.

- *Robot Selection Bar*
  This bar provides a means for selecting a robot to be controlled. Every robot will have its selector here, e.g. a button. Every selector will have a robot state indicator. This way the operator will see which robots are waiting for instructions, which need assistance etc.

- *Commands Toolbar*
  This toolbar lists all the tasks a robot is capable of performing, e.g. move, rotate, rotate camera etc. It also holds shortcuts to predefined Robot Control Scripts.

- *Sensors Area*
  Information from robot's sensors is displayed in the sensors area, e.g. video, infrared video, panoramic picture, distance to the object opposite to the robot etc.

- *Environment Map Area*
  This is where a map of robot's environment is displayed.

- *Interaction History List*
  This list contains the history of the whole interaction between the operator and a particular robot. All operator-robot and robot-operator messages are recorded and presented there.

- *Teleoperation Controls*
  There are two main teleoperation controls – one devoted to controlling the robot, and the other to controlling its camera. These controls enable the operator to perform both On-line and Quick teleoperation.
  An indicator of the camera orientation relative to the robot orientation is also shown here.

- *Status Bar*
  This area is used to present the robot's state information, e.g. power source capacity, network signal strength etc.

Fig. 2. Sample user interface design displaying results from motion detection sensor and IR camera. Results of the last panorama performed may be brought up from the bottom panel.

5. Map generation/printing

Urban search and rescue environments pose a significant challenge for localization and map generation. While there are many different techniques for simultaneous localization and mapping (SLAM), they are typically computationally expensive, ill suited for very large and unstructured environments, and require relatively expensive and bulky laser range finders to achieve acceptable results. Dust, rubble, and apparent lack of structure make feature extraction for vision based SLAM algorithms. Mirrors, glass, mesh, and ceiling panels confuse range sensors and rubble causes significant slippage distorting odometry readings. Yet these challenges must be overcome as it is critical that robots not only navigate treacherous terrain, but that they create accurate maps of the environment with the victims' location clearly marked to facilitate later rescue.

There are many different mapping algorithms including Kalman filters, Expectation Maximization algorithms, occupancy grid maps, object maps, and hybrid approaches [10]. We are particularly interested in occupancy grid mapping methods, which divide the world up into discrete cells and attempt to determine whether or not a cell is occupied, and object mapping methods, which assume that environments are composed of geometric shapes. Occupancy grids have already been applied with some success to the USAR robotics domain [12] are a relatively simple mapping technique and can better handle unstructured environments while object mapping allows us to exploit knowledge of structure in the environment such as building blueprints or rectilinearity assumptions.

The National Institute of Standards and Technology (NIST) provides standard test arenas for USAR robotics. Three arenas are available, in increasing level of difficulty for the robots. The first, the yellow arena, is a two-dimensional maze with an office setting [5]. We believe that object maps can represent this environment very well. The next level is the orange arena, a three-dimensional maze with stairs, ramps and holes. While not as more cluttered with rubble than the yellow arena, the orange arena possesses a distinct structure as well, and capturing this structure with object maps makes it easier for a human to follow the resulting map. The final arena, the red arena, has no remaining structure, but is simply rubble. One failing of object maps is that they do not work in unstructured environments. We feel that occupancy grids are the best approach for mapping the red arena. Our team strategy, however, is to focus on the yellow and orange arenas as the red arena poses not only a significant mapping challenge, but also a significant navigational challenge.

Robots capabilities are largely restricted by their sensors. As explained in the following section, our robots possess relatively limited sensing capabilities for navigation and mapping. Both robots are equipped with very coarse range finders and odometry feedback. We are collaborating with the developers of the laser line striper [4], a low cost range finder with no moving parts, centimeter accuracy, that uses a single laser and a camera, but as this sensor is still under development, there is no guarantee that this sensor will be integrated into our system before the competition. We must develop algorithms given the limited abilities of our robots and the assumptions we can make about the environment structure.

Given the constraints, we feel that a combination occupancy grid and object mapping approach will have the best results. Our algorithm divides the arena into 1m x 1m grid cells, a good size for our range sensors. Within each cell, we note the presence or absence of geometric barriers, in accordance with the object mapping algorithm. Assumptions of rectilinearity can be used to refine the map and correct errors in odometry. We also plan to allow the human operator to inject knowledge into the system should the robots become irretrievably lost, though the system will be designed to minimize the situational awareness requirements on the operator.

One advantage to object maps is that images taken by the robot while gathering range data can be superimposed on the map to provide additional data to a human rescue worker sent in to retrieve victims. We plan to generate a traditional overhead map of the arena that can be printed out but also a virtual map of the arena with image fingerprints at each grid cell. Many approaches to mapping focus only on generating an accurate mesh of the world, yet visual clues are of enormous benefit for humans following the map. A map with street signs and landmarks is much more intuitive for humans than a map drawn to millimeter accuracy without any such indicators. Similarly, associating visual landmarks with locations may allow rescue workers to overcome minor discrepancies in the map.

Although the proposed algorithm has several shortcomings including discretizing the environment and requiring assumptions about the structure of the environment, we believe that, given the limited sensor data of our robots, it will produce the best possible results. We will continue our search for better sensors and algorithms for localization and mapping.

6. Sensors for Navigation and Localization

Both robots use vision, range sensors, and odometry data for navigation and localization, though to varying degrees. Each robot has a USB web cam mounted on a pan tilt head. The resolution of the cameras and processing power constraints of the onboard computers make it unfeasible to run sophisticated vision algorithms to aid autonomous navigation but the video is transmitted to the human operator who can use this data to guide the robot.

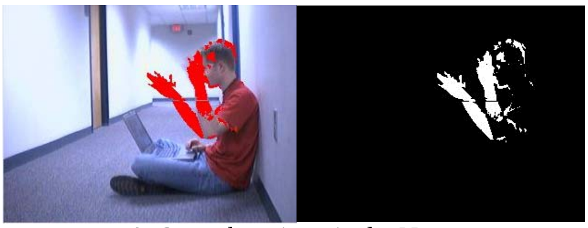
Corky has five Sharp range sensors placed at strategic positions to determine the proximity of obstacles. The PER has only one range sensor, mounted on the pan tilt head. Both robots use these sensors to avoid obstacles while traversing the arena and to determine the presence of walls for mapping.

Finally, both robots use odometry data for localization. The PER has a 332:1 gear ratio that allow the wheels to turn at a near constant velocity independent of load. The motors are controlled through the Cerebellum board, a fast, low cost, PIC based microcontroller. Open loop position estimation works well experimentally and is very simple and inexpensive. The larger robot, Corky, has custom mounted encoders on the motors. The PID control of the motors is handled by Acroname's BrainStem Moto Module, an off the shelf board. We built custom libraries for the ARM architecture to send commands to the Moto board based on the protocol specified in Acroname's BrainStem Documentation. These libraries promote rapid development through modularity and reusability, creating high-level primitives such as setting the wheel velocities, reading encoder values, and reading actual wheel velocities.

7. Sensors for Victim Identification

In order to detect as many different properties of victims as possible, we are using four different complementary sensors: USB cameras, microphone, pyroelectric sensor and infrared camera. The PER has a relatively limited sensor suite for victim identification of only the Creative WebCam Pro USB camera. With this sensor, it can detect motion and identify victims with the help of the human operator. Corky has an extensive sensor suite designed and tested for victim identification including the Philips ToUCam Pro which has a built-in microphone, a Murata pyroelectric sensor with a Fresnel lens, and a Raytheon Infrared camera. This robot can survey its environment for motion, heat, and sound, fuse the data together, and suggest places for further exploration to the human operator who can efficiently examine the scene for shapes and colors that indicate human presence.

The algorithm for motion detection requires the simple subtraction of two successive images taken while the robot was at rest. With assumptions on the distance to the moving object, the number of disparate pixels has empirically been shown to correspond to the probability that the motion detected was actually caused by a victim (see Appendix) [2]. Although more advanced motion detection algorithms are certainly available, this algorithm can be performed in real time on the onboard computer and the results are relatively accurate as shown in figure 3.



At 3m: changing pixels: N = 3.2%

Fig. 3. Results of motion detection algorithm. Pixels that changed between the images are on the right with overlay onto the original image on the left.

We also employ a Murata IMD-B101-01 pyroelectric sensor, designed to detect moving heat sources with wavelengths in the range of 5-14 μm (heat produced by humans has a wavelength of approximately 7 μm) [2]. A Fresnel lens extends the range of this sensor to 5 meters. Since the pyroelectric sensor only works when the heat source is moving, the sensor is mounted on the pan tilt head and scans the environment when the head moves.

Our sound detection assumes that the calls for help of trapped victims are louder than ambient noise. Initial calibration of the system suppresses the ambient noise.

Sound detection is then performed in the time domain. Our algorithm detects sustained high amplitude sound and returns the duration of the sound and the mean amplitude. This information can be used as input to an empirically determined function of probability that the noise detected was generated by a victim (see Appendix) [2]. This algorithm for voice detection lacks the ability to distinguish between human and non-human sources of noise but can be run in real time onboard the robot's computer. We plan to add the capability to record interesting sound as a wave file and transmit it back to the human operator for analysis. Sample results of the voice detection algorithm are shown in figure 4.

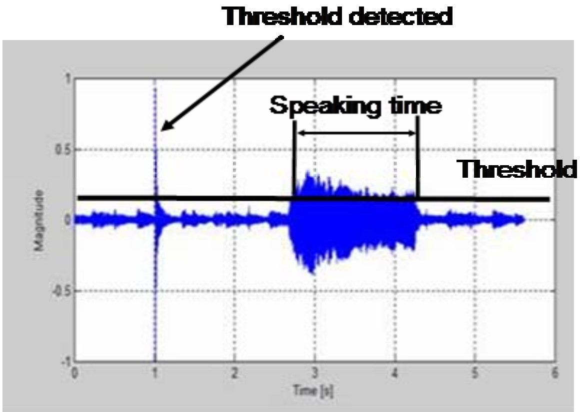


Fig. 4. Sample voice detection results illustrating that short noises that are unlikely to be human voices are filtered

Our final and most powerful sensor for human detection is the Raytheon Infrared camera 2000b. This camera provides images of the environmental heat in gray scale where white indicates hot objects and black indicates cold objects. The spectral range is 7- 14 μm. We use the analog output of the infrared camera and transmit the image via an analog transmitter to a frame grabber on an external computer. The image is analyzed to determine the location of the largest brightest section of the image, which is returned as the most likely location of the victim. The image is currently also analyzed for motion as a moving hot spot is probably a victim in a real disaster environment. This analysis, which uses the tLib library [3] is computationally expensive and is performed on an external computer. For the competition, we plan to remove this step. A sample image from the infrared camera is shown in figure 5 with the head of the victim correctly identified.

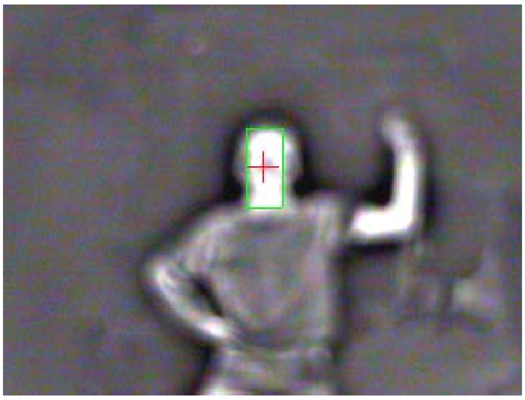


Fig. 5. A sample image from the infrared camera with the victim's head correctly identified in green

The results from each sensor are fused together by weighting each probability with confidence values. These confidence values can either be empirically determined, set at calibration time, or set by the operator. For more information on the sensor fusion, refer to [2] and Appendix.

8. Robot Locomotion

8.1 Corky

Corky is a two-wheeled differential drive robot. It has a Kevlar tail which provides stability as a caster which can be lifted up in the air for zero-radius turns.

Fig. 6. Side view (left) and rear view (right) of Corky supported by a wooden stand

This elegant design is highly maneuverable, enjoying all the benefits of a diff-drive robot with 9” ground clearance. It is mechanically very simple, yet we have achieved some success with stair-climbing using specially modified wheels.

The tail can help un-wedge the robot from a difficult situation; should a wheel get caught in debris, the tail can be brought around and used as a lever to extract the wheel. The Kevlar material is strong, lightweight and highly resistant to abrasion while the shape ensures rigidity.

Fig. 7. View of Corky’s Kevlar wheel caster

We chose to use off-the-shelf 26” diameter mountain bike wheels with Panaracer FireXCPro 2.1 tires. These tires are 2.1” wide and are known for their durability and traction. The distance between wheel centerlines is 12.5”.

Each wheel is independently driven by a Huafeng Electrical wheelchair motor rated at 3.6A at 24V with a no-load speed of 110 RPM. The maximum rated torque on the motors is 120 in-lb at 94 RPM with a current draw of 13.2A. Each wheel is coupled to a motor using a custom machined aluminum hub which mates directly to the bicycle wheel spokes (see figure 8). This allows easy disassembly of the wheels for transportation and the ability to change the dynamics of the robot by mounting different wheels as long as they have a standard bicycle wheel hub.

Fig. 8. Wheel mounting hubs (left) and wheel mounted on motor (right)

The fully assembled robot weighs approximately 40lb. Two 12V 8Ah lead-acid batteries power the robot motors. Power for the electronics comes from two 7.2V 1800mAh NiCad batteries. In this configuration, we expect a runtime of several hours on flat ground.

Our robot does have some limitations. Clearly, it is fairly large. Corky was designed to be narrow in order to easily fit through doorways; however our desire for useful ground clearance limited how much we could reduce the vertical dimension. While our robot can pass through doorways, it cannot fit into small, cramped spaces. Also, the robot may have problems with objects getting stuck between its wheel spokes. The problem has been solved reasonably well with the addition of wheel covers.

Another concern has is that although Corky has a reasonable tread width and a low center of gravity, it is possible to flip the robot on its side in extreme terrain or at high speeds. In this arrangement it is not always possible to reorient the robot. In addition to implementing careful control algorithms that mitigate this possibility, one possible solution to this problem is to install lightweight pneumatic lifts on the robot's sides.

Fig. 9. Corky searches for victims in the disaster arena

8.2 The PER

The Personal Exploration Rover or PER was developed at Carnegie Mellon University as part of a larger project to develop low-cost robots for education, science museums, and homes. Forty PER robots were built and deployed at science centers around the country for educational use in "Mars Yards", environments designed to imitate the environment on Mars.

The PER uses a rocker-bogie mobility system and has a differential axle that serves as the attachment point for the left and right wheel structures and the main rover body. The differential ensures that the main body angle always averages the left and right wheel rocker angles.

The PER has independent front and rear wheel steering, powered by four motors each with a 332:1 gear ratio that allows them to maintain a near constant velocity whether or not they are under load with open loop control. Each motor uses 16 volts and is small enough to fit inside the robot's custom wheels. Each wheel is connected to a servo allowing the wheels to be individually steered. This allows the robot to move sideways, although current control methods do not fully utilize the power of this drive system. The wheels were custom-manufactured for the PER, designed to mate with the Hsiang drive motors. The middle wheels are omnidirectional, enabling the rover to move sideways. [6,7]

The robot measures .4 meters tall, .2 meters wide, and .3 meters long. It is easily transported by a single person at 4.5 kilograms (10 pounds). It has a maximum speed of 4 cm (1.6 in) per second can last for over 5 hours on a fully charged battery pack.

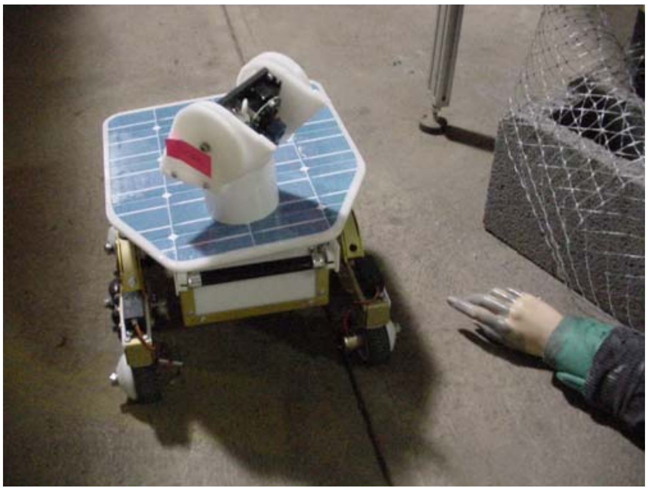


Fig. 10. PER robot explores the arena

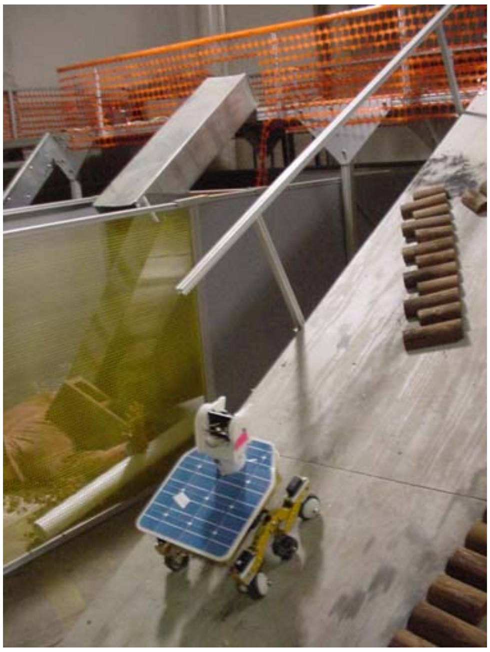


Fig. 11. Drive motors with a 332:1 gear ratio allow the PER to spin at a near constant velocity whether or not they are under load. The PER easily navigates this ramp

9. Team Training for Operation (Human Factors)

We intend to use our simulations for initial operator training. Desirable operator skills are likely to involve the perceptual problems of identifying victims and other features from low resolution video, shifting attention between robots, shifting between automated navigation and teleoperation, and maintaining situational awareness. We are attempting to make these tasks easier by limiting the number of independent displays and providing as much information as possible through HUD-like overlays on the video display. We have not yet integrated panoramas into our interface and do not know what impact this may have on training. Further operator training will occur with the real robots in the orange level disaster arena developed by the National Institute of Standards and Technology (NIST).

USARSim [11] is the simulation we built to help people test and evaluate the design and performance of remotely controlled robots in urban search and rescue environments. The simulator uses the Unreal engine on which Unreal Tournament and other video games are based to provide a high fidelity interactive environment that includes 3D environment simulation of the yellow, orange, and red NIST arenas that include textures, glass, and mirrors, a rigid-body physics engine to build robot models to simulate mechanical robots, and accurate noisy robot sensors.

10. Possibility for Practical Application to Real Disaster Site

Practical application to a real disaster site requires several basic issues in robotics to be fully addressed. The robot must have the power autonomy to function for a long enough duration to improve the state of information and allow slow and deliberate exploration; the robot must be reproducible and low-cost enough to warrant construction of a number of units, for both redundancy in the face of failure and so that the rescue team can benefit from the advantage of greatly increased numbers of eyes and ears in the disaster site; the robot must truly have the locomotive means to travel robustly through at least a subset of true disasters, where its particular array of sensors and/or effectors is of value; finally the robot must provide out-of-the-box functionality that is truly desirable to the current players at the scene of a disaster.
Our robot makes use of a power-efficient architecture composed of two main processors, a low-level PIC based processor for motor and sensor control, and an embedded single-board PC Stayton board designed by Intel Corporation for low-power robotic computation. This architecture demands less than 300 milliamps of power draw, providing the potential for hours of use at a disaster site. The PER smaller robot has a total energy autonomy of 12 hours under continuous motion and uses the same computer architecture as Corky. Furthermore the PER has been mass-manufactured in quantity 20, demonstrating that a construction run is possible when careful attention is paid to using high-volume, off-the-shelf actuators and sensors throughout the robot design. Locomotion is a remaining challenge, demanding extremely differing levels of functionality in different disaster or emergency scenarios. Corky has the ability to climb four inch obstacles with little or no trouble, enabling straightforward navigation in lightly obstacle-filled environments. PER has a rocker-bogie suspension system similar to that of the Mars rovers, and as such can climb obstacles exceeding the diameter of its wheels. We believe the final, true test of practicability involves the degree to which emergency rescuers would find the interface and functionality of such robots to be sufficiently valuable to deem the technological hurdle to be worthwhile surmounting. This is the critical axis on which the jury remains out and which we plan to measure through human factors-based testing of our robots and interfaces.

11. System Cost

Please use this section to total the costs of key system components and the overall expense of your system. In this section, we're looking for particular information that other teams could use to replicate your successful approaches. So part numbers, prices and websites are much appreciated. This information will only be available to the other teams (and the public) when it is published in the Springer Book long after the competition is complete.

Table 1. Cost and availability of major parts used

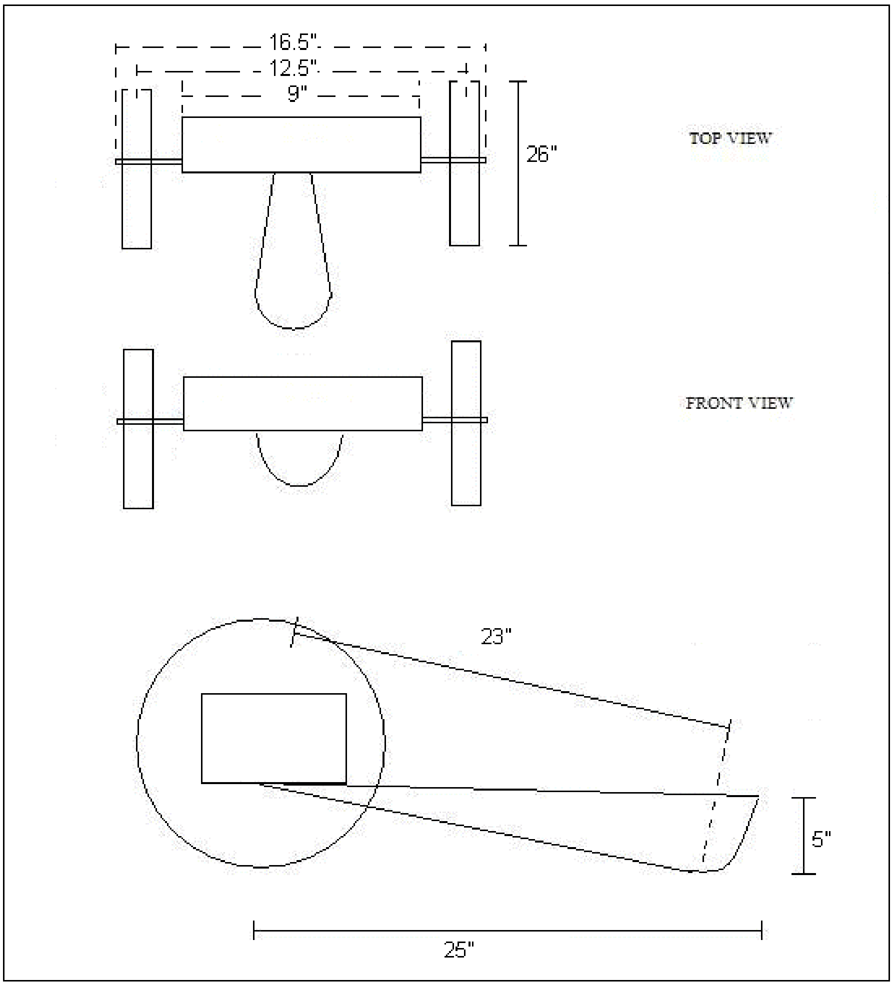
Part	Cost	Supplier
Hitec 225MG (Futaba J type)	$30	www.towerhobbies.com

tilt servo (HS-1225MGj) GWServo S11 HP/2BB MG		
Drive motors for PER: part number 170641		www.jameco.com
IR camera: Raytheon infrared ControlIR 2000B with 18 mm lens	$7500	Must be ordered through a dealer (for dealers): http://hurleyir.com/prod-ucts/2000B.php
Corky web camera: Philips ToUCam Pro (Model PCVC740)	$70	http://www.pc-cameras.philips.com/manuals/english/win/pcvc720k40_730_740k/index.html
PER web camera: Creative WebCam Pro	---	No longer in production
Pyro Sensor: Murata Pyroelectric Infrared Sensor Module IMD-B101-01	$30	http://www.murata.com
Sharp GP2Y0A02YK IR Sensor: 1 on PER, 5 on Corky	$15x6	www.acroname.com
Orinoco Gold wireless Ethernet cards: PER, Corky, controlling laptop	$90x3	www.nextag.com
Windows laptop for user interface	varies	
Frame grabber:		
Analog transmitter:		
Stayton board: main computer on Corky and PER	---	Made by Intel, not commercially available: similar to stargate board at www.Xbow.com
PER Power board	---	Botrics
Cerebellum board on Corky and PER	---	Botrics
Moto board for encoders and PID control on Corky	$70	www.acroname.com

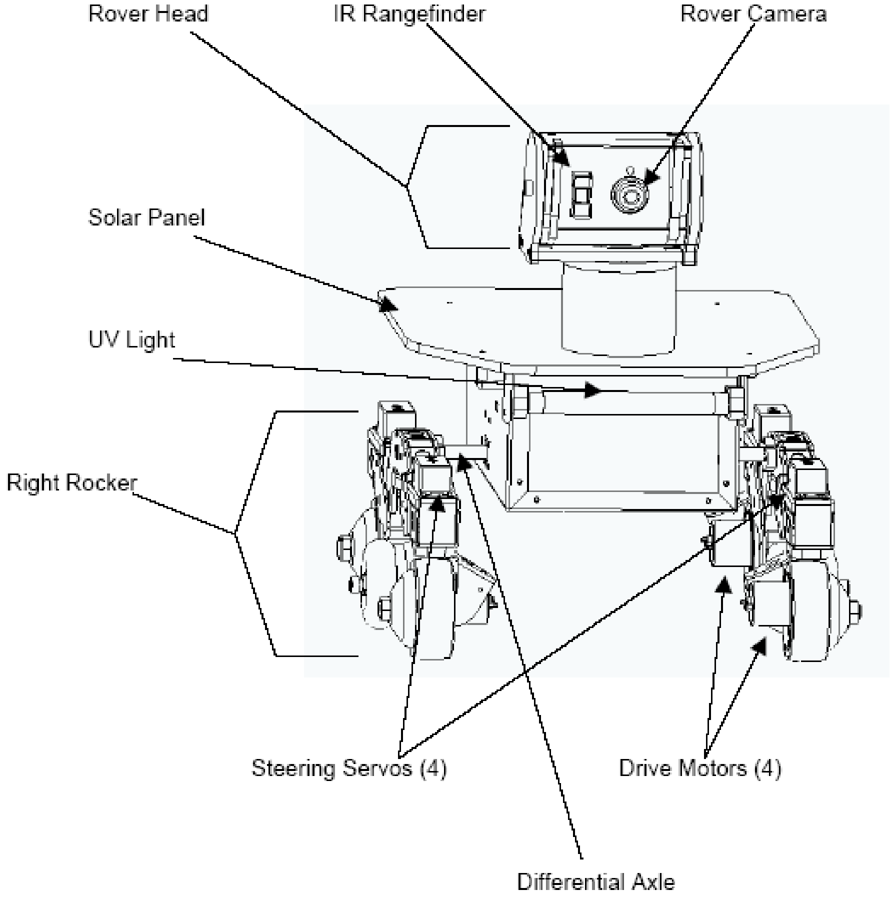
References

1. Berna-Koes, M., Nourbakhsh, I., Sycara, K.: Communication Efficiency in Multi-Agent Systems. To appear in the Proceedings of IEEE 2004 International Conference on Robotics and Automation. (2004)
2. Burion, S.: Human Detection for Robotic Urban Search and Rescue. Available Online: http://www-2.cs.cmu.edu/afs/cs/project/retsina-31/www/Report/Final%20Report.pdf (2004)
3. Grange, S., Fong, T., Baur, C.: TLIB: a real-time computer vision library for HCI. VRAI-Group, EPFL, Lausanne (2003)
4. Mertz, C., Kozar, J., Miller, J.R., Thorpe, C.: Eye-safe Laser Line Striper for Outside Use. IV 2002 (June 2002)
5. National Institute of Standards and Technology (NIST): Performance Metrics and Test Arenas for Autonomous Mobile Robots. [Online] Available: http://www.isd.mel.nist.gov/projects/USAR. (2004)
6. National Science Center: Mars Rover Project Educational Guide. [Online] Available: http://www.nationalsciencecenter.org/FortDiscovery/Events/documents/MarsEducationGuide.pdf. (2004)
7. Nourbakhsh, I.: The Personal Exploration Rover. [Online] Available: http://www-2.cs.cmu.edu/~personalrover/PER/ (2004)
8. Sycara, K., Paolucci, M., van Velsen, M., Giampapa, J.: The RETSINA MAS Infrastructure. Special Joint Sigue of Autonomous Agents and MAS. Vol. 7, No. 1 and 2, (July 2003)
9. Sycara, K., Widoff, S., Klusch, M., Lu, J.: LARKS: Dynamic Matchmaking Among Heterogeneous Software Agents in Cyberspace. Autonomous Agents and Multi-Agent Systems. Vol. 5, 173-203 (2002)
10. Thrun, S.: Robotic Mapping: A survey. In G. Lakemeyer and B. Nebe, editors, Exploring Artificial Intelligence in the New Millenium, Morgan Kaufmann, (2002)
11. Wang, J., Lewis, M., Gennari, J.: A Game Engine Based simulation of the NIST USAR Arenas. Proceedings of the 2003 Winter Simulation Conference (2003)
12. Zunino, G., Christensen, H.: Simultaneous Localization and Mapping for Navigation in Realistic Environments. In Proc. IEEE Conference on Multisensor Fusion and Integration for Intelligent Systems MFI’2001, pages 67-72, (August 2001)

APPENDIX



Drawing of Corky robot



Drawing of PER robot

	Human	No human
Human detected	**95.2 %**	46.7 %
Nothing detected	4.8 %	**53.3 %**

Accuracy of Pyroelectric Sensor

$$p_f = \frac{c_1\ f_1(x_1) + c_2\ f_2(x_2) + \ \dots\ + c_n\ f_n(x_n)}{c_1 \max(f_1(x_1)) + c_2 \max(f_2(x_2)) + \dots + c_n \max(f_n(x_n))} = \frac{\sum_{i=0}^{i=n} c_i\ p_i}{\sum_{i=0}^{i=n} c_i \max(p_i)}$$

Formula for sensor fusion with:
c = confidence value of each sensor,
f(x) = function which give a probability to have a human for each sensor
pf = final probability

For the complete description of sensor models used on robot, please refer to [2]