

Multi-Robot Exploration for Search and Rescue Missions — A Report on Map Building in RoboCupRescue 2009 —

Keiji Nagatani, Yoshito Okada, Naoki Tokunaga, Seiga Kiribayashi, Kazuya Yoshida, *
Dept. of Aerospace Engineering
Graduate School of Engineering, Tohoku University
6-6-01, Aramaki-Aoba, Aoba-ku, Sendai, Miyagi 980-8579, Japan
{keiji, yoshito, tokunaga, kiribayashi, yoshida}@astro.mech.tohoku.ac.jp

Kazunori Ohno, Eijiro Takeuchi, Satoshi Tadokoro,
Dept. of Bioengineering and Robotics
Graduate School of Engineering, Tohoku University
6-6-01, Aramaki-Aoba, Aoba-ku, Sendai, Miyagi 980-8579, Japan
{kazunori, takeuchi, tadokoro}@rm.is.tohoku.ac.jp

Hidehisa Akiyama, Itsuki Noda,
National Institute of Advanced Industrial Science
and Technology (AIST)
Tsukuba Central 2,
1-1-1 Umezono, Tsukuba,
Ibaraki 305-8568, Japan
{hidehisa.akiyama, I.Noda}@aist.go.jp

Tomoaki Yoshida, Eiji Koyanagi
Chiba Institute of Technology
2-17-1, Tsudanuma, Narashino, 275-0016, Japan
{yoshida, koyanagi}@furo.org

Abstract

In the future, mobile robots may be able to assist rescue crews in search and rescue missions that take place in the dangerous environments that result from natural or man-made disasters. In 2006, we launched a research project to develop mobile robots that can rapidly collect information in the initial stages of a disaster. One of our important objectives is 3D mapping, which can be a very useful tool for assisting rescue crews in strategizing rescue missions. To realize this 3D mapping, we identified five issues that we needed to address: (1) Autonomous traversal of uneven terrain, (2) Development of a system for the continuous acquisition of 3D data of the environment, (3) Coverage path planning, (4) Centralization of map data obtained by multiple robots, and (5) Fusion of map data obtained by multiple robots. We solved each problem through our joint research. Each research institute in our group took charge of solving one of the above issues according to its area of expertise. We integrated these solutions to perform 3D mapping using our tracked vehicle, Kenaf. To validate our integrated autonomous 3D mapping system, we participated in RoboCupRescue 2009 and demonstrated our system using multiple robots on the RoboCupRescue field. In

*Webpage: <http://www.astro.mech.tohoku.ac.jp/>

this paper, we introduce our mapping system and report the mapping results obtained at the RoboCupRescue event.

1 Introduction

1.1 Background: Rescue robots

In the future, mobile robots may be able to assist rescue crews in search and rescue missions in dangerous environments after natural or man-made disasters such as the Hanshin earthquake or the sarin gas attack on the Tokyo subway. To promote the development of search and rescue robots for rescue missions (called rescue robots), we launched the Project for Strategic Development of Advanced Robotics Elemental Technologies, Area of Special Environment Robots, RT System to Travel within Disaster-affected Buildings (Yoshida et al., 2009). The project has been supported by the Ministry of Economy, Trade and Industry of Japan (METI) and the New Energy and Industrial Technology Development Organization (NEDO).

One of the goals of this research project is to develop rescue robots that can rapidly collect information in the first stage of a disaster in order to eliminate the risk of a possible secondary disaster that would harm human responders. In this situation, 3D mapping is one of the important technologies that would assist rescue crews in strategizing rescue missions. Furthermore, autonomous rescue robots will be able to perform rapid mapping in a large-scale disaster environment. Autonomy is important because, generally, there are not many operators who can control tele-operated rescue robots skillfully. In this context, we began the development of rescue robots that can perform autonomous mapping along with other tele-operated rescue robots.

1.2 Background: RoboCupRescue

RoboCup was started in 1997 as an annual international competition for the promotion of robotic technologies. Its primary objective was to develop a robotic football team that can beat a human team. Because of RoboCup’s concrete objective and competitive format, rapid progress was expected in many areas of robotic technologies, such as mechanisms, control, sensors, and artificial intelligence.

In 2001, the RoboCupRescue league (a real robot league and a simulation league) was added to RoboCup in order to encourage research in the field of search and rescue robotics. (Jacoff et al., 2003) and (Kitano et al., 2002) reported in detail on RoboCupRescue league topics.

One activity of this league was a competition regarding the number of victims and the accuracy of the maps that robots could obtain in a simulated disaster environment within certain time limits.

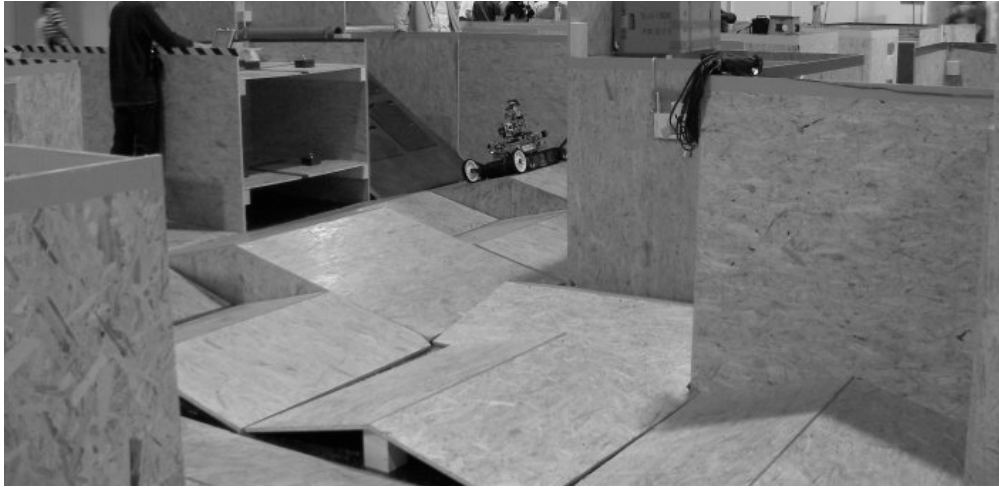
The target field in this competition is basically divided into three areas: a yellow area, an orange area, and a red area.

The yellow area contains a path bounded by wooden walls that has small steps and gentle slopes that are designed for autonomous robots. The orange area is designed to have the same structure as the yellow area, but with steeper slopes and larger step heights typically. Additionally, stairs are included in the orange area. The red area, which was originally designed for tele-operated robots, is the most challenging to traverse. It contains steep slopes, stairs, and a rugged terrain. The rugged terrain is prepared by assembling prismatic vertical wooden poles of different lengths in a lattice-like arrangement. This rugged terrain is called the “step field.”

A number of simulated victims are embedded in these fields, and they must be found by the robots. The victims are characterized by temperature, recorded voices, motions (such as hand waving), CO2 gas emission,



(a) Overview of target field



(b) Orange area in target field

Figure 1: Target field in RoboCupRescue 2009

and ID tags. When a robot finds a victim, the operator reports the victim’s location and other information to the referees, and the score is calculated according to a weighted sum of the information. For example, in an actual rescue mission, identifying the victim’s location on a map is essential, so a higher score is assigned to success in this activity.

The deployment point of the robots is not announced beforehand; the participants locate their robots at the deployment point at the beginning of the trial.

Each team can deploy any number of robots in a trial. There are two categories of robots: autonomous robots and tele-operated robots. The operator of the robots can use a 5GHz wireless LAN to communicate with robots in the field. Autonomous robots explore the field and search for victims without any prior access to maps. When a robot finds a possible victim, it presents its collected information (a photo, a CO₂ sensor reading, a thermal image, etc.) to the operator. The operator then decides if it should be on the final map and sends “accept” or “reject” command to the robot. There are only three commands that operators are allowed to send to autonomous robots: “start mission”, “accept victim” and “reject victim.” The tele-operated robots have no such restrictions on communications.

1.3 Research objectives and issues to be solved

The RoboCupRescue field may appear a toy-like version of the real world. However, if we are not able to perform mapping in such a toy world, mapping in an actual disaster environment remains a distant dream.

We, therefore, made mapping the RoboCupRescue field the first objective of our research. Our second objective was to map the field using multiple robots.

To realize these objectives, we needed to solve the following five issues: to solve the following five issues:

1. Autonomous traversal of uneven terrain
2. Development of a system for the continuous acquisition of 3D data of the environment
3. Coverage path planning
4. Centralization of map data obtained by multiple robots
5. Fusion of map data obtained by multiple robots

To address the first issue, our approach was to use a system of four subtracks for our target tracked vehicle and to implement sensor-based control of the subtracks using two laser range sensors. This topic is discussed in section 3. To address the second issue, we implemented a continuous acquisition system and ICP-matching-based map correction method for 3D-LIDAR-equipped all-terrain tracked vehicles to acquire 3D range data. Details are provided in (Nagatani et al., 2008) and an overview of this topic is provided in section 4. For the third issue, we implemented frontier-based coverage path planning, discussed in section 5. The fourth issue is very important as a research infrastructure that will enable exploration of disaster areas using multiple robots. Therefore, we implemented a GIS system for rescue robots, discussed in section 6. For the fifth issue, we adopted a map fusion method based on graph-based SLAM, discussed in section 7. Finally, conclusions and future works are described in section 8.

2 Related works

This work is related to demonstrations (or field reports) and some research components. The first successful demonstration of 3D mapping in RoboCupRescue was in (Nuchter et al., 2005). In their work, clear 3D maps of the RoboCupRescue field were autonomously created by a wheeled robot. Recently, the RoboCupRescue field for autonomous robots became more difficult to traverse, and 3D mapping became a challenging task for wheeled robots. We believe that our work in this paper represents the first successful demonstration of 3D mapping in such a difficult environment using such small tracked vehicles. Regarding the research components that are related with this work, we categorize the topics as follows.

Autonomous traversal on rough terrain

The traversability of the robot on rough terrain depends on its size and its mechanism. For planetary exploration robots, Rocker-bogie suspension is popular for increasing the locomotion ability of wheeled robots on rough terrain without actuators (Harrington and Voorhees, 2004)(Bartlett et al., 2008). An all-terrain hex-limbed robot, called Athlete, developed by NASA JPL, has a capability of rolling mobility on moderate terrain and walking mobility on extreme terrain (Wilcox et al., 2007). The above robots are typical wheeled-robots on rough terrain.

Recently, a mechanism for tracked vehicles that includes subtracks has become very popular because it can negotiate bumpy terrain, particularly in RoboCupRescue. However, the operator’s workload increases when the number of actuators is increased. Consequently, some methods for the autonomous control of subtracks were proposed. One method was to use a measurement of the contact force of the subtracks (Ohno et al., 2007). In this research, two short-range LIDARs located at both sides of the robot were used to obtain information about the ground. Details about this method were reported in (Y.Okada et al., 2009). Another autonomous controller of subtracks was proposed by Birk and implemented on their tracked vehicle,

named Rugbot. A fuzzy controller is used for control of rear subtracks to enable autonomous stair climbing (Chonnaparamutt and Birk, 2008). One method that did not involve subtracks used an actuated tail to maintain stability (Guarnieri et al., 2009).

Coverage path planning

Yamauchi proposed frontier-based exploration (Yamauchi, 1997). This is a very simple and powerful method in a 2D environment. Actually, our autonomous traversal function worked well, and the RoboCupRescue field was assumed to be a 2D environment. Therefore, the frontier-based approach was used in our exploration strategy and worked well, as shown in section 5. Yamauchi also proposed cooperative frontier-based exploration in (Yamauchi, 1998). This may be useful for our cooperative exploration strategy. However, we did not apply it because our system included a tele-operated robot, not a fully autonomous system.

Graph-based SLAM

The research area of SLAM is vast, and a basic approach of SLAM is clearly introduced in (Thrun et al., 2005). Here, we would like to discuss some works related to graph-based SLAM. One of the popular algorithms involves the use of a probabilistic method, such as FastSLAM2.0 (Montemerlo et al., 2003). This is known to be a very robust and powerful method. However, in the case of map building with multiple robots, the number of particles increases along with the number of robots, which can lead to the problem of excessive computing cost. (F.Lu and E.Milios, 1997), Olson et al. (Olson et al., 2006), and Takeuchi and Tsubouchi (E.Takeuchi and T.Tsubouchi, 2008) proposed node-splitting methods, which are referred to as graph-based SLAM. Grisetti et al, improved the optimization algorithm of graph-based SLAM (Grisetti et al., 2009) (Konolige et al., 2010). Such methods are robust against an increase in the number of robots because they only require additional nodes and constraint conditions. One of the applications of graph-based SLAM was proposed by Kleiner and Sun (Kleiner and Sun, 2007); this application was used for estimating trajectories of firemen in rescue missions. In RoboCupRescue 2009, we adopted the above graph-based approach in order to fuse the map data obtained by multiple-robots.

Cooperative mapping

From the point of view of research topics in cooperative mapping, various algorithms have been proposed. Burgard proposed a coordinated multi-robot exploration method that takes into account the cost of reaching a target point and the robots' utility (Burgard et al., 2005). Andreas uses pose graph map representation for communicating map updates (Pfingsthorn and Birk, 2008). Our approach is based on the graph-based SLAM (E.Takeuchi and T.Tsubouchi, 2008), because it is very easy to extend for Multi-robots' localization and mapping.

3 Autonomous traversal of uneven terrain

Our objective is to realize autonomous mapping in disaster environments. In the case of a flat surface, conventional wheeled robots are sufficient for this purpose. The target environment shown in section 1.2, however, was constructed to resemble a disaster environment that includes slopes, bumpy surfaces, and step fields. Therefore, an autonomous system for traversal of an uneven terrain is required in order for our target robot to achieve the first goal we proposed.

In this competition, we used our tracked vehicle named "Kenaf 22" (Fig.2), which was equipped with four actuated subtracks that could change their mounting angles to maintain the stability of the body. Though Kenaf was originally designed for tele-operation, we installed an autonomous control system of actuated subtracks. The control algorithm of the subtracks is shown in Fig.3. A brief summary of the

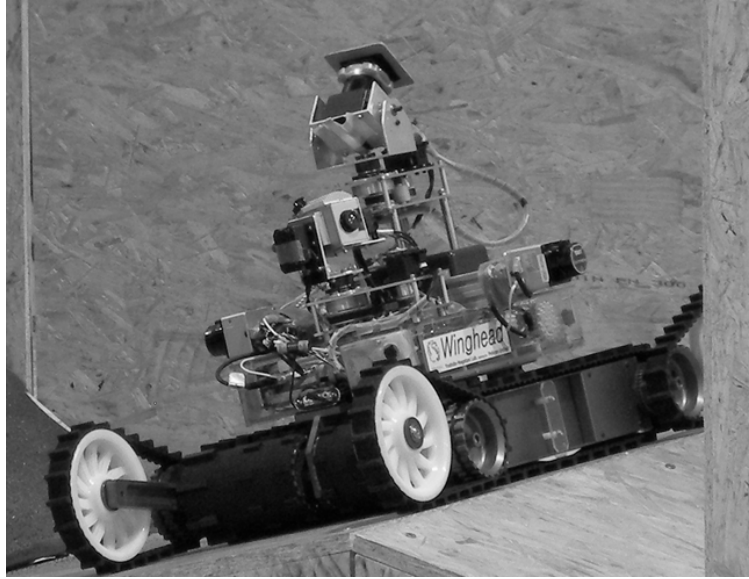


Figure 2: Tracked vehicle Kenaf 22 with 3D LIDAR (on top)

system is given below, and details of the control system were introduced in (Y.Okada et al., 2009).

1. The ground surface along both sides of the vehicle is first detected by two LIDARs (Laser Imaging Detection and Ranging), one attached on each side of the robot, and a target area of the detected surface is determined.
2. The desired attitude of the robot is calculated based on the detected ground surface. The surface is usually not flat. Therefore, the desired attitude of the robot is designed to fit the least-squares plane of the surface.
3. Using the desired attitude of the robot, the desired positions of the flippers are also determined by a rigorous geometric calculation.
4. A stability criterion is calculated.
5. If the stability criterion does not exceed a threshold value, the desired attitude is stabilized and steps 3 – 5 are repeated.

In the target environment in RoboCupRescue 2009, this system worked rather well. Indeed, while the organizer of the competition assumed that autonomous robots would function only in the yellow field that included gentle inclines and small steps, our system performed autonomous navigation not only in the yellow field, but also in the orange field that included large slopes and steps (with a maximum gap of approximately 30cm). As can be seen in Fig.4, our robot autonomously traversed a set of challenging steps. We attempted traversal in the red field that included step fields, but the robot almost tipped over a number of times. Despite this result, the robot has the definite potential to surmount step fields because the tele-operated version has done so successfully. From this, we understood that local path planning with consideration of stability is quite important for such applications.

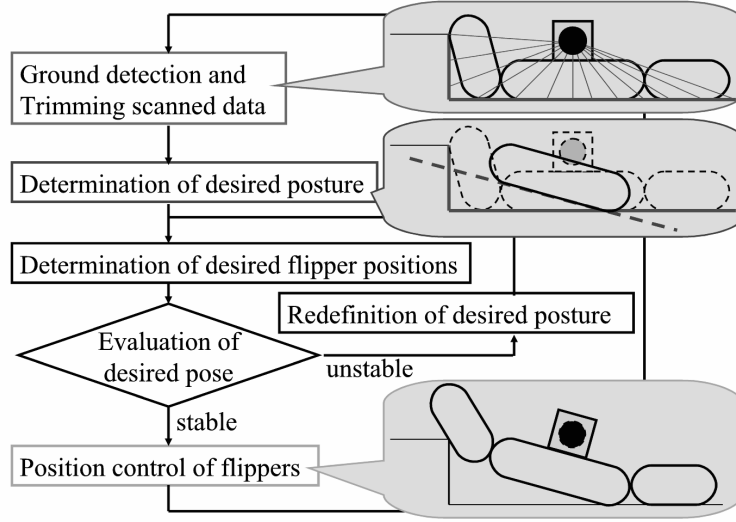


Figure 3: Algorithm for autonomous controller for subtracks

4 System for continuous acquisition of 3D environment data

To realize non stop navigation with the continuous acquisition of 3D environment data, we proposed — at an international workshop on Safety, Security, and Rescue Robotics (SSRR 2008) (Nagatani et al., 2008) — a continuous acquisition system for 3D LIDAR equipped all-terrain robots that acquire 3D range data. In RoboCupRescue 2009, we used this system for mapping in simulated disaster environments. Furthermore, we employed an ICP (Iterative Closest Point) algorithm (Besl and McKay, 1992) (Rusinkiewicz and Levoy, 2001) to correct the irregularity of the data for the walls and to construct global 3D environment data online.

4.1 Creation of 3D environment data

In order for continuous 3D LIDAR to function while a robot is moving on uneven terrain, the robot must know its changes in position and attitude in 3D. However, because the sensor data for the robot’s position and attitude are not synchronized with the sampling period of the LIDAR data, the robot cannot integrate both sets of data. One approach for resolving this problem is Ueda’s method of synchronization (Ueda et al., 2006), which uses a synchronous signal output by a LIDAR system (URG-04LX, manufactured by HOKUYO AUTOMATIC). This method can synchronize the robot’s position and attitude data with the data from the LIDAR, and it can thus obtain 3D environment information while the robot is moving. Because this approach requires additional hardware, our implementation adopted a software synchronization method that uses timestamps.

This method of software synchronization is shown in Fig.5. When a robot calculates its position with a timestamp (the exact time value) by 3D odometry and receives scanning data with a timestamp from a LIDAR, the robot obtains exact correspondence of the data, and it does not need to halt for data collection. The data coming from our 2D LIDAR (UTM-30LX, manufactured by HOKUYO AUTOMATIC) includes timestamp information in 25 ms intervals, and we can get comparatively accurate 3D position and attitude data with the timestamp from the odometry system (which includes a gyroscope module) every 20 ms. Currently, we do not consider slippage of the tracks. Through synchronization of the odometry and the LIDAR, the robot can integrate both sets of data.

Based on the above strategy, we successfully implemented a continuous 3D LIDAR system on our tracked vehicle, so that the robot does not need to stop while the range data are being scanned. We have described

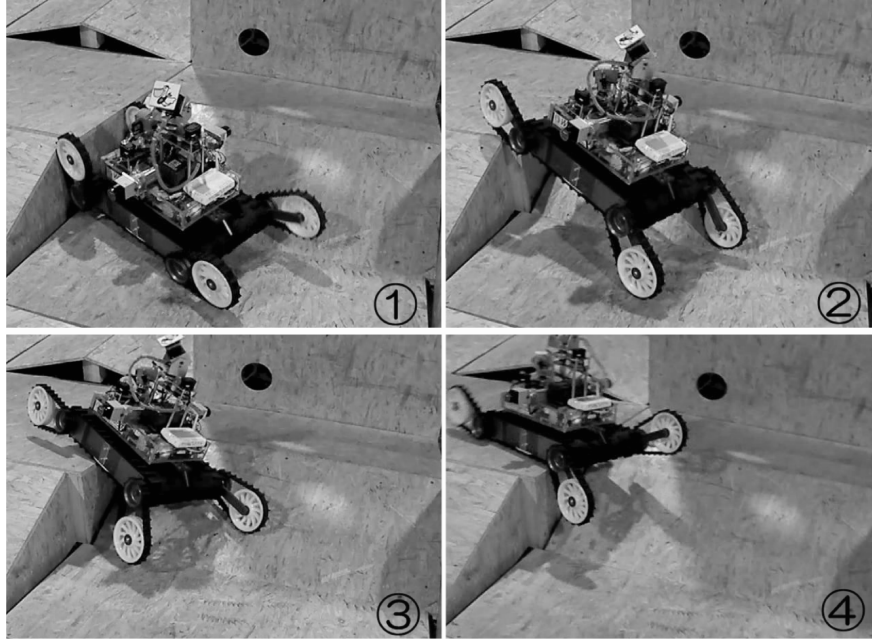


Figure 4: Maintaining stability through autonomous control of subtracks: In the above case, the goal of the planned path was set in the left side, and there was an upward step (about 20 cm-high) in front of the robot. It used the front subtracks to carry up its body (1-2), and it used the rear subtracks to prevent itself from tipping over (2-3); all this was done autonomously.

the method and performance tests on rough terrain in detail in (Nagatani et al., 2008).

In the target environment of RoboCupRescue 2009, this system worked quite well. It obtained accurate 3D data on the basis of the odometry. An example of the mapped 3D data results is shown in Fig.6. Because of the track slippage caused by the slopes and bumpy surfaces, long-distance navigation included an odometry error that caused a mapping error; this is evident in some irregularities in the data for the walls in Figs.6-(a) and 6-(b). However, a short-range map drawn using only odometry information is trustworthy.

4.2 DEM matching to create a large-scale environment

Because of the large size of the obtained data, we used the DEM (Digital Elevation Map) representation for position adjustment of the robot (Fong et al., 2003). However, doing so has a disadvantage in that the DEM cannot represent the space under objects because only the highest scan point is effective. Some ideas have been presented to extend the DEM representation, such as in (Pfaff et al., 2007). In our case, the robot was not capable of climbing up to the second floor, so we set the height threshold to obtain environment data for entering tunnels that could be free spaces. In other words, the world for robots was under 1 m. Thus, in our approach to the RoboCupRescue field, this disadvantage did not present a problem. While the robot was in motion, a local DEM ($5\text{m} \times 5\text{m}$) was generated every 5 s.

For each creation of a local DEM, scan-matching was executed using the ICP algorithm (Besl and McKay, 1992) (Rusinkiewicz and Levoy, 2001) to correct the irregularity of the data for the walls and to construct a global DEM (both performed online). Each calculation between the latest local DEM and the global DEM was conducted within 1.5 s. In our implementation, we did not consider the problem of loop closing. Because there were many environmental signatures, the matching did not fail in the target environment. A representation of point-cloud mapping without ICP is shown in Fig.6-(a), and the result with ICP is shown in Fig.6-(c). In addition, magnifications of the right-hand parts of the point-cloud maps in Fig.6-(a) and

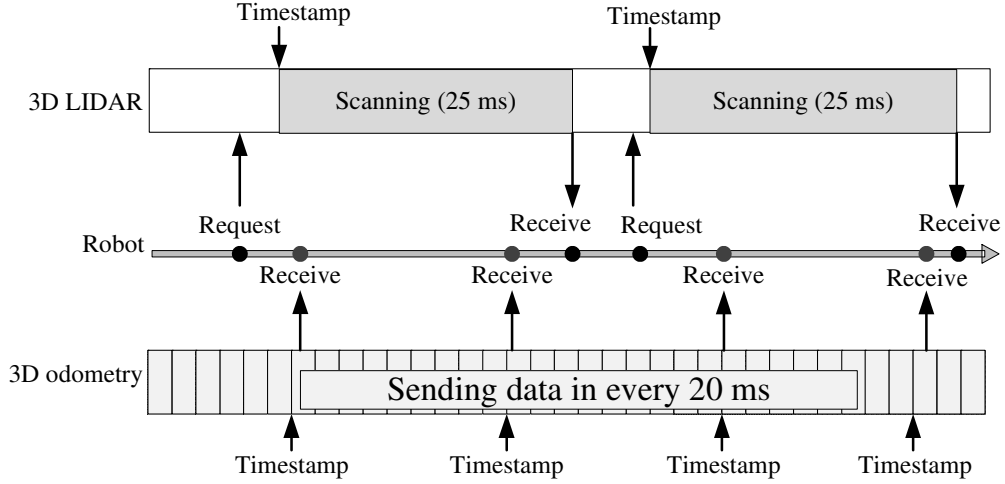


Figure 5: Continuous acquisition system for 3D LIDAR based on timestamps

6-(c) are shown in Fig.6-(b) and 6-(d). As is obvious in these images, the standard ICP algorithm worked well in our case.

Fig.7 shows a comparison between the ground truth and the odometry-based map, and Fig.8 shows a comparison between the ground truth and the ICP-based map. These graphs indicate that the irregularity of the odometry was corrected through the ICP matching and that a more reliable map was generated.

For the purpose of quantitative evaluation of the correction by ICP, we neutrally selected distance errors from three corners of the map and summarized them in Table 1. This result clearly shows that every error was significantly reduced by ICP matching. We consider this to be a typical advantage of the use of ICP.

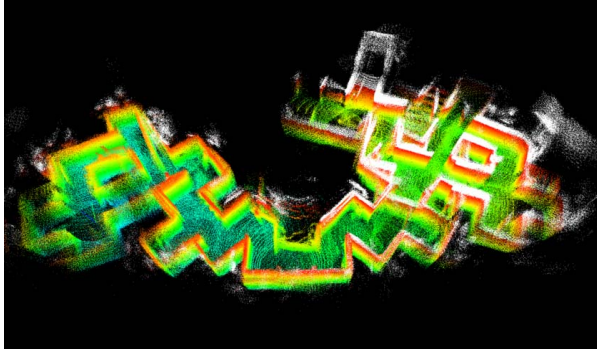
5 Coverage path planning

In the mission of the RoboCupRescue competition, the identification of the victim’s location is the most important goal. The acquisition of this information requires the additional capabilities of accurate coverage and map construction. In this paper, we focus on mapping the environment and do not address the issue of finding victims.

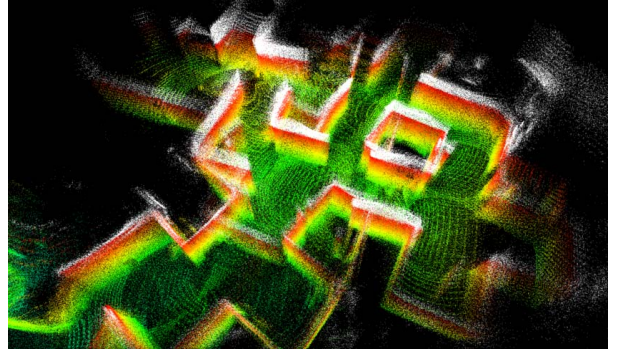
There are many approaches (described in section 2) to achieve autonomous exploration and mapping in unknown environments. If a target environment is designed with long slopes and ambiguous potential obstacles, a 2D exploration approach may fail. In this competition, our robot had an autonomous traversal function on somewhat rough terrain, and it did not have the capability to climb up onto the second floor. Therefore, we assumed that the target field was flat for planning, and we implemented frontier-based coverage path planning (Yamauchi, 1997), as described in the following procedure.

Table 1: Correction of irregularities by ICP matching

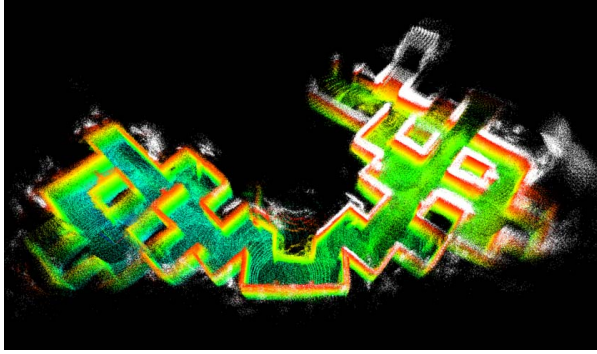
	Odometry[m]	ICP matching[m]
A-A'	2.72	0.60
B-B'	5.20	0.14
C-C'	2.76	0.57



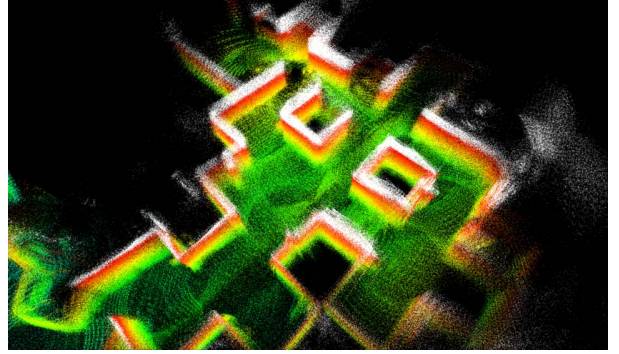
(a) Online 3D mapping based on odometry



(b) Magnification of right part of map shown in (a)



(c) Online 3D mapping based on ICP algorithm



(d) Magnification of right part of map shown in (c)

Figure 6: DEM matching results: These are point-cloud expressions. The color corresponds to the height of the objects in the environment. Green indicates low objects, such as the surface of the ground. Red and white indicate high objects, such as high obstacles or walls. A comparison between (a) and (c) indicates that the ICP algorithm worked correctly and maintained the visual consistency of the environment.

1. Segmentation of the global DEM

An obtained map based on the global DEM is segmented –on the basis of the height data of the DEM– into (1) open space (O_s), (2) obstacle space (O_b), (3) C-space (C_s) (which is at a certain distance from obstacles), and (4) unknown space (U_s). In our case, we simply segmented the DEM grids into open space and obstacle space by height with the threshold value of 30cm. Fig.9-(a) is an example of an obtained map based on a global DEM, and Fig.9-(b) shows a segmented result of that map. The green area corresponds to O_s , the black area to O_b , the yellow area to C_s , and the gray area to U_s .

2. Identifying frontier regions

In this research, we use two definitions for frontier regions (F_r) that must be explored. The first definition takes into consideration the distance from the robot's path, as follows:

$$F_r = O_s \cap C(R_p), \quad (1)$$

where $C(\cdot)$ is the complementary set in the bracket, and R_p is the space in which the distance to the path swept by the vehicle over all previous time is less than the threshold (in our case, 1.0 m). This means that, even if the robot senses that a location is open in front, the regions beyond the threshold distance are classified as open space. Fig.9-(c) provides an example of frontier regions, which are colored red, according to this definition.

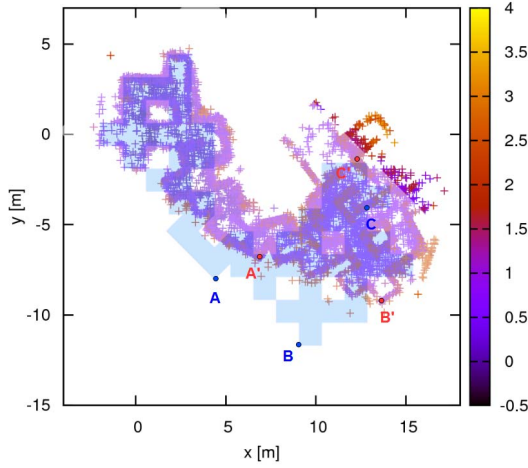


Figure 7: Comparison between ground truth and global map based on odometry

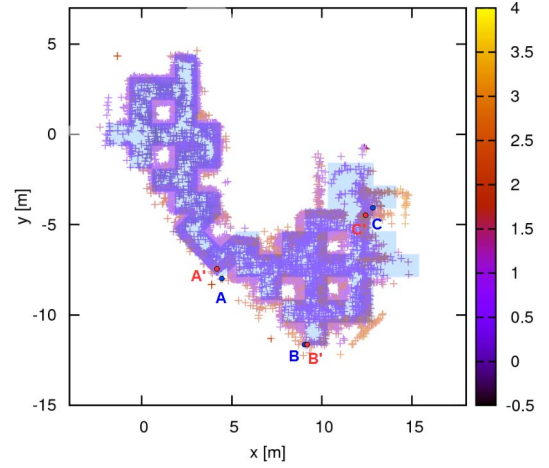


Figure 8: Comparison between ground truth and global map based on ICP matching

The second definition of frontier regions considers the “line of sight” of the LIDAR. The line of sight is defined as the border of an open space around the unknown space. Fig.9-(d) provides an example of frontier borders, which are colored red, according to this definition.

The first approach was used for the main competition, because the robot should close to within 1m of victims according to RoboCupRescue regulations. The second approach was much faster than the first approach, so we used the latter approach in “the autonomous challenge in best in class,” which required only mapping, not finding victims.

3. Path planning and execution

Once the frontier regions are defined, the region closest to the robot is chosen as the subgoal. To find the shortest path, we applied Dijkstra’s algorithm. This was a very simple algorithm to implement, and its performance in terms of speed was adequate for our application. Finally, a grid-based path to the subgoal is generated. To allow for smoother travel to the subgoal, we chose a waypoint-tracking method. Waypoints are selected out of the grids on the path; the first waypoint is set to the grid on which the robot is currently located, and the n-th waypoint is the farthest grid that is reachable by linear travel from the (n-1)-th waypoint. To track the waypoints sequentially, we employed a conventional line-following method and a steering method for typical two-wheeled mobile robots. Fig.9-(e) provides an example of a planned path to one of the frontier regions based on the first definition. Fig.9-(f) provides an example of a planned path to one of the frontier borders based on the second definition.

4. Repetition

The algorithm repeats steps 1-3 until there are no remaining frontier regions.

On the field in RoboCupRescue 2009, the abovementioned algorithm worked well in terms of exploring the environment. The first frontier definition was applied to the typical competition of searching for victims. It took a very long time for the robot to explore the target environment. In the end, because of its poor victim search function, the robot did not find any victims in the preliminary stages. The algorithm did work well, however, in terms of coverage of the target environment. The second frontier definition was applied to the autonomous challenge in best in class (a separate competition to validate the functions of rescue robots). This challenge did not require a robot to search for victims, but rather, to obtain a map of the entire environment. It took our robot only about 20 min to map three quarters of the entire environment, as shown in Fig.6-(c).

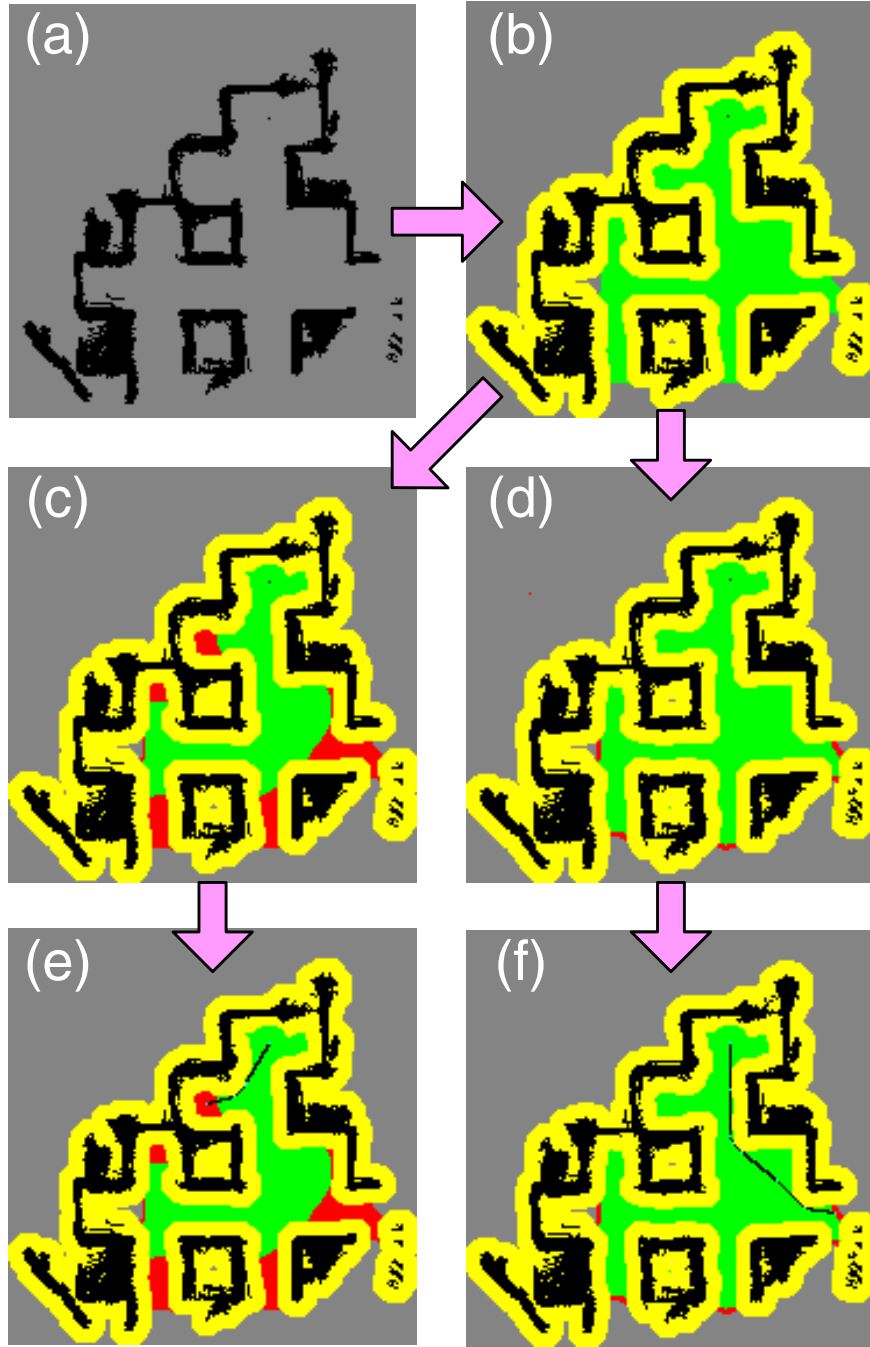


Figure 9: Frontier-based coverage path planning

(a) Original map: This was obtained by 3D LIDAR and filtered by the ICP algorithm. The robot, represented by the small black dot, was located at the center of the top rectangle area.

(b) Segmentation map: The black areas represent obstacles, the yellow areas represent C-obstacles, the green areas represent free space, and the gray area is unknown area.

(c) Frontier regions: The red areas indicate frontier regions based on the first definition.

(d) Frontier regions: The red areas indicate frontier regions based on the second definition.

(e) Planned path: The path generated from the location of the robot to the closest frontier region obtained by the first definition.

(f) Planned path: The path generated from the location of the robot to the closest frontier region obtained by the second definition.

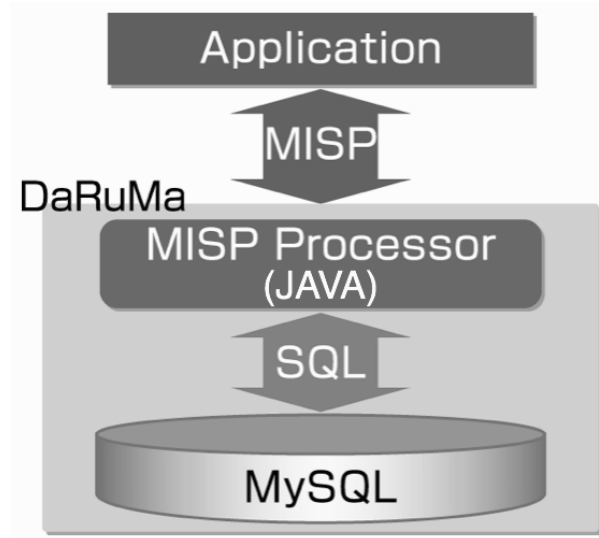


Figure 10: Structure of DaRuMa: DaRuMa consists of Java middleware and an SQL server. The Java middleware interprets MISP messages and stores and manages the data in an SQL server. Currently, MySQL is used as a backend SQL server.

6 Centralization of map data obtained by multiple robots

The exploration of disaster area using multiple robots is effective for search missions that are conducted under time constraints. In RoboCupRescue, cooperative map building using tele-operated robots and autonomous robots is preferred because autonomous robots still find it difficult to navigate the orange and red areas in the challenge environment. In this competition, we therefore sought to achieve the integration of data through the use of a database.

To centralize map data obtained by distributed robots, the robots and GIS database must share a common communication protocol. Furthermore, for rescue missions, the protocol should handle not only numeric values but also geometry data, time, non-text data such as images, and structured data in the disaster environment. An additional requirement is that the database should not stop functioning while a rescue mission is being conducted.

6.1 MISP: Mitigation Information Sharing Protocol

In order to satisfy the requirements presented in the previous subsection, we designed the Mitigation Information Sharing Protocol (MISP) (NIED and AIST, 2006) to serve as a communication protocol among robots and databases. The MISP is an XML protocol based on GML (Geography Markup Language) (OGC, 2007). The MISP provides the functions of search, modification, and deletion as a standard database. In addition to the usual database functions, the MISP has an additional command, **RegisterFeatureType**, to define a new data type and its XML structure. Using this command, users can add a new data type without stopping and redesigning the whole system. This kind of flexibility is especially important for a rescue system because it is difficult to define everything before a disaster occurs. The **RegisterFeatureType** command provides the capability of connecting new systems and handling new types of information in emergency situations.

Furthermore, MISP supports any number of local coordinate system and has a function to define the relationship between coordinate systems. These specifications enable us to handle several local coordinate systems generated by multiple robots flexibly.

6.2 Implementation of GIS: DaRuMa

To realize a MISP server, we developed DAtabase for Rescue Utility MAnagement (DaRuMa). The DaRuMa is an open source project that can be downloaded at (I.Noda and H.Akiyama, 2009).

The structure of DaRuMa is shown in Fig.10. It consists of Java middleware and a SQL server. The former supports transactions of communication protocols, and stores the data in a SQL server. For our SQL server, we chose MySQL, which has proven stability in large scale databases and is commonly utilized as a standard database. The above system is open source software.

In RoboCupRescue, each robot sends MISP messages that represent victim data, 3D data of the target environment with a timestamp, and the robot's position via wireless communication to DaRuMa in order to centralize the obtained data.

6.3 Coordination between DaRuMa and SLAM program

The size of raw environment data obtained by 3D LIDAR as described in section 4 (point clouds data) is considerably large. We therefore adopted a DEM for registration of the environment data in the database, instead of point clouds data.

Once the environment data (DEM maps) obtained by multiple robots has been collected in DaRuMa, a fused map is then required. A support program of DaRuMa, called the viewer program, has the function of generating a visualized map that consists of a number of superimposed DEM maps. The database itself, however, does not have an adjustment function to account for the relative positions of robots. Then, a mismatched map is obtained, typically.

To avoid this situation, we proposed a graph-based map correction method, which is presented in the next section. The method, which is coded as an independent support program of the viewer program, is called a SLAM program, and it communicates with the viewer program to generate transformation and rotation matrices of the robots' relative positions.

An overview of the structure of this dynamic relationship is shown in Fig.11. The procedure to generate an integrated map is as follows:

1. When one of the robots obtains environment data locally, it sends its position data and a DEM map to DaRuMa.
2. The viewer program receives the DEM maps and the data on the robots' positions from DaRuMa, and sends them to the SLAM program.
3. The SLAM program generates a transformation matrix of the relative positions of the robots, and sends the matrix back to the viewer program.
4. The viewer program generates a fused map with the adjusted relative position data of the robots.

In all missions in RoboCupRescue 2009, this system functioned smoothly for the collection of data obtained by up to three robots.

Currently, the tentative SLAM program directly communicates with the viewer program to generate the transformation and rotation matrices of the robots' positions. Under ordinary circumstances, however, the SLAM program should communicate with the GIS program to generate these matrices. We have already implemented such a framework in the DaRuMa that handles the transformation and rotation of coordinates. In the near future, we will implement a new SLAM program that communicates with the DaRuMa directly.

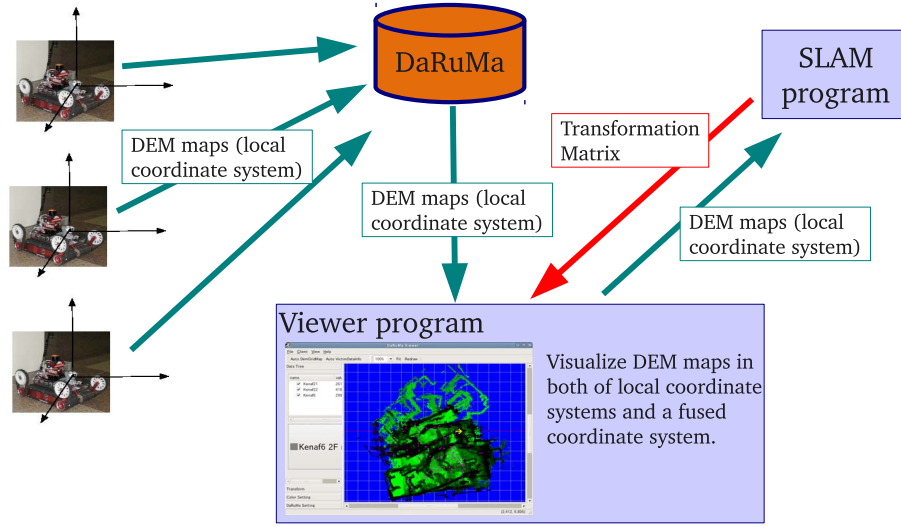


Figure 11: Map matching using DaRuMa : Environment data obtained by robots are gathered into DaRuMa. The viewer program retrieves registered data represented by local coordinate systems. In order to generate a fused map, the viewer program cooperates with the SLAM program.

7 Fusion of map data obtained by multiple robots

As mentioned in the previous section, a mismatched map may be generated if there is no adjustment of the relative positions obtained by odometry. Furthermore, each local map obtained by each robot may include its own unique distortion. For these reasons, we applied a graph-based map correction method for our fusion of map data obtained by multiple robots. The method is described in detail in (E.Takeuchi and T.Tsubouchi, 2008), and a summary of the method is given as follows.

7.1 Strategy for fusion of map data

If there are no positioning errors or sensing errors, the obtained map data are integrated in DaRuMa on the basis of the odometry-based position data. We must consider, however, that there is a positioning error for each robot.

A schematic outline of our method is shown in Fig.12, and the procedure of the method is as follows:

- (a) Generate DEMs defined as “nodes”
- (b) Produce constraint relationships between neighboring nodes from the odometry trajectory of each robot
- (c) Produce constraint relationships based on ICP scan matching (Besl and McKay, 1992) of two DEMs in two nodes
- (d) Optimize the nodes’ positions (trajectories of robots) in order to maximize the joint probabilities of possible constraint relationships

In (c), we constrain not all node-pairs but only neighboring nodes. Furthermore, to prevent mismatching, we only choose very high scores from the ICP matching. Of course, the number of matching pairs is decreasing. However, through iteration of the ICP matching (c) and the graph modification (d), the whole graph structure is gradually revised, and the successful ICP matching pair can be expanded from the root of the graph to its boundaries.

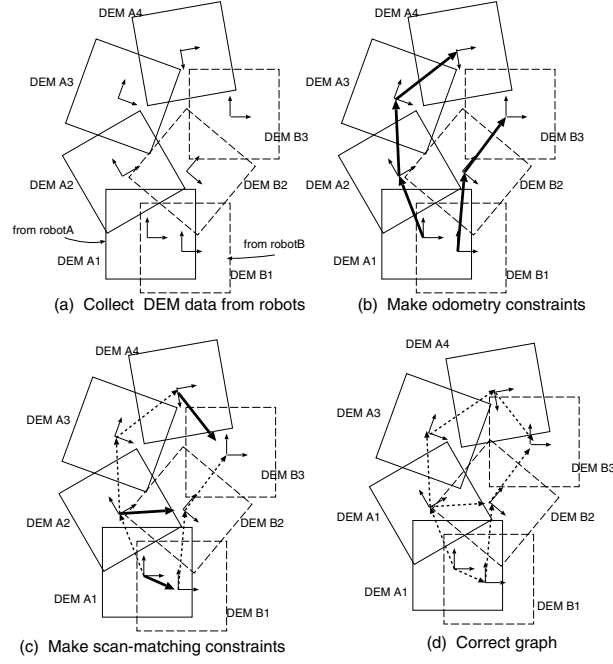


Figure 12: Multi-robot SLAM based on graph SLAM :

- (a) Local maps obtained by two robots (solid line areas were obtained by robot A and dashed line areas by robot B).
- (b) Producing constraint relationships between neighboring nodes from the odometry trajectory of each robot
- (c) Producing constraint relationships from the ICP matching.
- (d) Positions of nodes optimized by graph-based SLAM.

7.2 Fusion of map data based on graph-based SLAM

Our approach to fusion of map data can be executed by obtaining the accurate trajectories of mobile robots. This can be done by finding the locations of nodes in order to maximize the joint probability:

$$p(\mathbf{x}) = \prod_{i=0}^{n-1} p(\mathbf{x}|\mathbf{c}_i), \quad (2)$$

where $\mathbf{c}_i (i = 0 \cdots n-1)$ are the constraint relationships between two nodes obtained by odometry and ICP scan matching. The nodes are defined by the centers of the DEM maps obtained by multiple robots in different places.

When we assume that the errors in the constraint relationships follow a normal distribution, equation (2) is described as follows:

$$p(\mathbf{x}) = \prod_{i=0}^{n-1} \exp\left(-\frac{1}{2} \mathbf{f}(\mathbf{x}, \mathbf{c}_i)^t \boldsymbol{\Sigma}_i^{-1} \mathbf{f}(\mathbf{x}, \mathbf{c}_i)\right), \quad (3)$$

where $\mathbf{f}(\mathbf{x}, \mathbf{c}_i)$ is the constraint function between two nodes, and $\boldsymbol{\Sigma}$ is the covariance matrix of the constraint.

The above constraint includes two conditions. One is the physical relationship constraint of the consecutive-ness of nodes. This becomes obvious when a robot continues scanning and building DEM maps sequentially. The other condition is an ICP scan-matching constraint. In our approach, ICP scan matching is adopted

for nearly every DEM map. Once the evaluated value exceeds the threshold, a new constraint is produced between the two similar nodes.

By calculation of log likelihood and linearization of vector $\mathbf{f}(\mathbf{x}, \mathbf{c}_i)$, the maximization problem of equation (2) is transformed into a solution problem for the following simultaneous equations:

$$\begin{aligned} \mathbf{A}\mathbf{u} &= \mathbf{b} \\ \mathbf{A} &= \sum_{i=0}^{n-1} (\mathbf{J}_i^t \Sigma_i^{-1} \mathbf{J}_i), \\ \mathbf{b} &= \sum_{i=0}^{n-1} \mathbf{J}_i^t \Sigma_i^{-1} \mathbf{r}_i, \end{aligned} \tag{4}$$

where $\mathbf{J}_i$ is a Jacobian matrix of $\mathbf{f}(\mathbf{x}, \mathbf{c}_i)$ at $\mathbf{u}$.

In the above simultaneous equations, matrix $\mathbf{A}$ is a large nondense matrix that includes many zero elements. Therefore, the calculation is faster when the skyline (or envelope) approach (Z.Zlatev et al., 1981) and the Cuthill-McKee method (Cuthill and McKee, 1969) are used. Details of the above are described in (E.Takeuchi and T.Tsubouchi, 2008).

7.3 Initial performance test for fusion of multiple maps

We applied the above-mentioned method to our three tracked vehicles (Kenaf 6, Kenaf 21, and Kenaf 22) to achieve fusion of map data. In this implementation, the size of the local DEM was set as a 100-grid square, with each grid size set at 5 cm $\times$ 5 cm. In two of the three robots, 3D LIDAR was mounted in order to obtain 3D environment information. In the third robot, we mounted a tilted 2D LIDAR that obtained 3D environment information by means of its motion. All robots were tele-operated for the test.

The target field, which was constructed at the Chiba Institute of Technology, Japan, was a relatively small simulation of the RoboCupRescue field.

One typical result is shown in Fig.13. In these figures, each L-shaped red mark represents a node that implies the center of the obtained DEM that was registered discretely. Each green line represents a constraint relationship between the nodes.

Fig.13-(a) shows the raw data obtained by the three robots. Fig.13-(b) represents a matching result using only the data from Kenaf 22 based on our approach. The environment was more precisely delineated, but the loop segment in the circular area A is incomplete because the robot did not explore there. Furthermore, the walls were not matched completely, as shown in circular area B. Fig.13-(c), on the other hand, represents a matching result that was obtained using data from all three robots, based on our approach. As can be seen, the problems involved in using a single robot (shown in circular areas A and B) were solved by the complementary explorations of other robots. Thus, a great advantage is offered by the integration of data obtained by multiple robots. Please keep in mind that the above fusion algorithm can be performed on the DaRuMa system discussed in the previous section. However, in our implementation, the algorithm was performed off-line. The algorithm requires an iterative calculation that took about 10 s to complete with a Pentium Core2 Duo at 1.6 GHz.

Based on this performance test, we concluded that our approach could be applied to the RoboCupRescue missions. Although we did, in fact, implement the above approach, we did not obtain a good experimental result on the field of RoboCupRescue 2009. The reasons for this were (1) the time constraints of the competition that permit only a small area of coverage by autonomous robots and (2) a rare odometry error that resulted when the robot's angular-speed exceeded the dynamic range of its gyroscope in the case of quick motions.

8 Conclusions and future works

In this paper, we reported the mapping results obtained on the field of RoboCupRescue 2009. To obtain these results, we needed to address the following five issues:

1. Autonomous traversal of uneven terrain
2. Development of a system for the continuous acquisition of 3D data of the environment
3. Coverage path planning
4. Centralization of map data obtained by multiple robots
5. Relative position adjustment among multiple robots for map generation.

To solve the first issue, we controlled actuated subtracks autonomously based on information from two LIDARs. This approach performed fairly well in the RoboCupRescue fields.

To solve the second issue, we mounted a 3D LIDAR and implemented a continuous acquisition of 3D environment data based on synchronization between 3D odometry and range sensor data. After obtaining local range data, we performed DEM matching to create a large-scale environment. In this paper, we summarized the synchronization method, and the details are available in (Nagatani et al., 2008). As a result of our DEM matching method, the inconsistency of the target environment in the RoboCupRescue field caused by odometry errors was dramatically reduced, as shown in Fig.6.

To solve the third issue, we applied a frontier-based approach. We implemented two definitions of frontier regions, one that explores an environment based on the “sweeping” and one that is based on the “line of sight.” We used the former approach for the main competition because the robot should be within 1 m of the victim according to RoboCupRescue regulations. However, the latter approach is much faster than the former approach. Therefore, we used the latter approach for “the autonomous challenge in best in class,” which required only mapping (not finding victims), and we succeeded in exploring a large area, as shown in Fig.9.

To solve the fourth issue, we proposed an MISP and implemented a GIS called “DaRuMa,” which is explained in section 6. It was effective in RoboCupRescue for gathering information obtained by multiple-robots.

To solve the fifth issue, we implemented graph-based SLAM on our robots. Unfortunately, we could not obtain good experimental results in RoboCupRescue because of the time constraints of the competition and some other factors. However, we performed one initial experimental test at the Chiba Institute of Technology as a relatively small simulation of the RoboCupRescue field. The results shown in Fig.11 provide good validation of our approach.

Based on the above approaches, we concluded that we resolved the five issues and effectively performed our mapping of the RoboCupRescue field.

In our current approach, there is no strategy for planning the sharing of exploration areas obtained by multiple robots. Developing exploration area planning for multiple robots is one of our future goals. Furthermore, the map fusion method presented in section 7 is currently implemented in ad-hoc code. To address these issues, we will extend the MISP as a common communication protocol among robots and for GIS and matching programs, and we will implement their functions in DaRuMa’s framework.

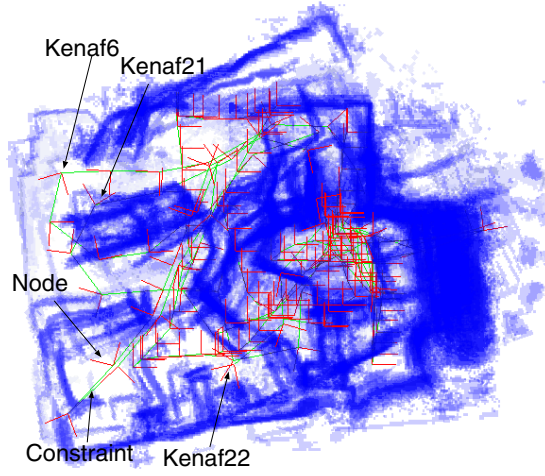
Acknowledgment

This work was supported by the Strategic Advanced Robot Technology, an R&D project of NEDO, Japan.

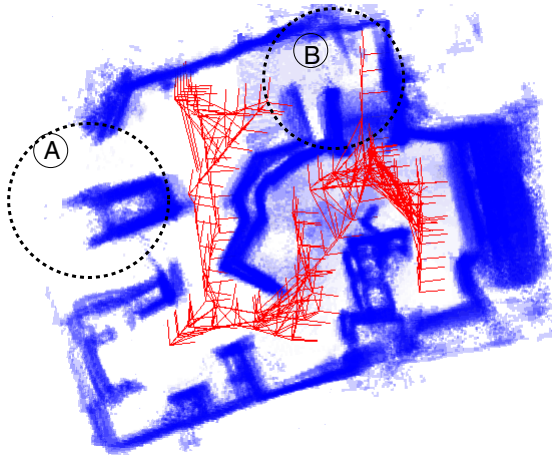
References

- Bartlett, P., Wettergreen, D., and Whittaker, W. (2008). Design of the scarab rover for mobility and drilling in the lunar cold traps. In *International Symposium on Artificial Intelligence, Robotics and Automation in Space*.
- Besl, P. J. and McKay, N. D. (1992). A method for registration of 3-d shapes. *IEEE Tran. on Pattern Analysis and Machine Intelligence*, 14(2):239–256.
- Burgard, W., Moors, M., Stachniss, C., and Schneider, F. (2005). Coordinated multi-robot exploration. *Robotics, IEEE Transactions on*, 21(3):376–386.
- Chonnaparamutt, W. and Birk, A. (2008). A fuzzy controller for autonomous negotiation of stairs by a mobile robot with adjustable tracks. *Robot Soccer World Cup XI*, page 196.
- Cuthill, E. and McKee, J. (1969). Reducing the bandwidth of sparse symmetric matrices. In *Proceedings of the 24th National Conference of the ACM*.
- E.Takeuchi and T.Tsubouchi (2008). Multi-sensor map building based on sparse linear equations solver. In *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2511–2518.
- F.Lu and E.Milios (1997). Globally consistent range scan alignment for environment mapping. *Autonomous Robots*, 4:333–349.
- Fong, E., Adams, W., Crabbe, F. L., and Schultz, A. C. (2003). Representing a 3-d environment with a 2 1/2-d map structure. In *Proc. of the IEEE/RSJ Conference on Intelligent Robots and Systems*, pages 2986–2991.
- Grisetti, G., Stachniss, C., and Burgard, W. (2009). Non-linear constraint network optimization for efficient map learning. *IEEE Transactions on Intelligent Transportation Systems*, 10:428–439.
- Guarnieri, M., Debenest, P., T.Inoh, K.Takita, H.Masuda, Kurazume, R., E.Fukushima, and S.Hirose (2009). Helios carrier: Tail-like mechanism and control algorithm for stable motion in unknown environments. In *The IEEE International Conference on Robotics and Automation*.
- Harrington, B. D. and Voorhees, C. (2004). The challenges of designing the rocker-bogie suspension for the mars exploration rover. In *37th Aerospace Mechanisms Symposium*.
- I.Noda and H.Akiyama (2009). Daruma (database for rescue utility management). <http://sourceforge.jp/projects/daruma/>.
- Jacoff, A., Messina, E., Weiss, B. A., Tadokoro, S., and Nakagawa, Y. (2003). Test arenas and performance metrics for urban search and rescue robots. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3396–3403.
- Kitano, H., Tadokoro, S., Noda, I., Matsubara, H., Takahashi, T., Shinjou, A., and Shimada, S. (2002). Robocup rescue: Search and rescue in large-scale disasters as a domain for autonomous agents research. In *IEEE International Conference on Systems, Man, and Cybernetics, 1999*, volume 6, pages 739–743. IEEE.
- Kleiner, A. and Sun, D. (2007). Decentralized slam for pedestrians without direct communication. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1461–1466. IEEE.
- Konolige, K., Grisetti, G., Kummerle, R., Burgard, W., Limketkai, B., and Vincent, R. (2010). Efficient sparse pose adjustment for 2d mapping. In *The 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*.
- Montemerlo, M., Thrun, S., Koller, D., and Wegbreit, B. (2003). Fastslam 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges. In *Proceedings of IJCAI*.
- Nagatani, K., Tokunaga, N., Okada, Y., and Yoshida, K. (2008). Continuous acquisition of three-dimensional environment information for tracked vehicles on uneven terrain. In *Proceedings of the 2008 IEEE International Workshop on Safety, Security and Rescue Robotics*, pages 25–30.
- NIED and AIST (2006). *Mitigation Information Sharing Protocol*, rev.1.00.028s edition.
- Nuchter, A., Lingemann, K., and Hertzberg, J. (2005). Mapping of rescue environments with kurt3d. In *Safety, Security and Rescue Robotics, Workshop, 2005 IEEE International*, pages 158–163.

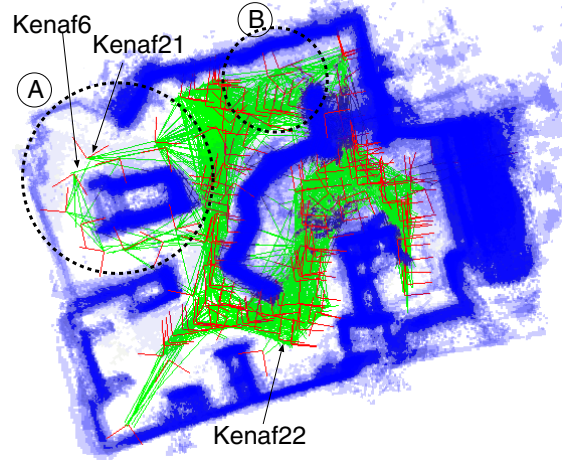
- OGC (2007). *OpenGIS Geography Markup Language (GML) Encoding Standard*, rev.3.2.1 edition. OGC 07-036.
- Ohno, K., Morimura, S., Tadokoro, S., Koyanagi, E., and Yoshida, T. (2007). Semi-autonomous control system of rescue crawler robot having flippers for getting over unknown-steps. In *Proceedings of the 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3012–3018.
- Olson, E., Leonard, J., , and Teller, S. (2006). Fast iterative alignment of pose graphs with poor initial estimates. In *IEEE International Conference on Robotics and Automation*, pages 2262–2269.
- Pfaff, P., Triebel, R., and Burgard, W. (2007). An efficient extension to elevation maps for outdoor terrain mapping and loop closing. *The International Journal of Robotics Research*, 26(2):217.
- Pfingsthorn, M. and Birk, A. (2008). Efficiently communicating map updates with the pose graph. In *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems*.
- Rusinkiewicz, S. and Levoy, M. (2001). Efficient variants of the icp algorithm. In *Proc. of the International Conference on 3-D Digital Imaging and Modeling*, pages 145–152.
- Thrun, S., Burgard, W., and Fox, D. (2005). *Probabilistic Robotics*. The MIT Press.
- Ueda, T., Kawata, H., Tomizawa, T., Ohya, A., and Yuta, S. (2006). Mobile sokuiki sensor system -accurate range data mapping system with sensor motion. In *Proceedings of the 3rd Int'l Conf. on Autonomous Robots and Agents*.
- Wilcox, B., Litwin, T., Biesiadecki, J., Matthews, J., Heverly, M., Morrison, J., Townsend, J., Ahmad, N., Sirota, A., and Cooper, B. (2007). ATHLETE: A cargo handling and manipulation robot for the moon. *Journal of Field Robotics*, 24(5):421–434.
- Yamauchi, B. (1997). A frontier-based approach for autonomous exploration. In *Proc. of the IEEE International Symposium on Computational Intelligence in Robotics and Automation*, pages 146–151.
- Yamauchi, B. (1998). Frontier-based exploration using multiple robots. In *Proceedings of the second international conference on Autonomous agents*, pages 47–53. ACM.
- Y.Okada, K.Nagatani, and K.Yoshida (2009). Semi-autonomous operation of tracked vehicles on rough terrain using autonomous control of active flippers. In *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2815–2820.
- Yoshida, T., Nagatani, K., Koyanagi, E., Hada, Y., Ohno, K., Maeyama, S., Akiyama, H., Yoshida, K., and Tadokoro, S. (2009). Field experiment on multiple mobile robots conducted in an underground mall. In *Field and Service Robotics*.
- Z.Zlatev, J.Wasniewski, and K.Schaumburg (1981). *solution of large and sparse systems of linear algebraic equations*. Springer Verlag.



(a) Raw data from three robots



(b) Fusion of map data using Kenaf22



(c) Fusion of map data using all three robots

Figure 13: Typical results of a performance test: The blue areas represent objects, the red L-marks represent positions of nodes, and the green lines represent constraints of nodes.