Robocup 2010-Rescue Simulation League Virtual Robot Competition KavehGlass

Majid Yasari, Majid Sadeghi Alavigeh, S. Ali Zahiri, Morteza Sabetraftar, Saeed Shariati, Mohamad Taghi mir Mohamad Rezaee, Nasser Mozayani

IUST Robotic Association Lab
Computer Engineering Department
Center of Student Scientific Association
Iran University of Science & Technology, Tehran, Iran
majid_yasari@comp.iust.ac.ir, {majid.sadeghi.alavijeh, sa.zahiri, morteza.sabetraftar, saeed.shariati, mmrezaie}@gmail.com, mozayani@iust.ac.ir

http://robotic.iust.ac.ir

Abstract. This paper is a quick overview of Kaveh 2010 rescue virtual robot team which describes our improvements and works in relation with robotic science and AI in order to participate in Singapore 2010 Robocup Competitions. The most our effort in last year progress was finding some ways to optimize and reduce SLAM process. This year we are going to use some new concepts to dissolve some problems such as obstacle avoidance and inspiring from nature in order to multi-robot coordination and disaster recovery when an agent missed in multi-robot systems which is called Stem Cell Theory. More, we are going to give a short description of our robot and techniques that are used in order to deal with several challenging problem in Rescue Virtual Robot such as 6D SLAM, Autonomous Exploration, Obstacle Avoidance and Victim Detection via camera.

1 Introduction

Every year in all around the world, thousands of people die or get injured under collapsed buildings or other unreachable areas because of explosion, earthquake, flood and other natural or unnatural disasters. Delay in rescue operations due to these dangerous situations, vast locations, massive casualties leads us to employ high technologies to aid rescue group. Designing, implementing and using rescue robots not only makes explorations much more accurate by seeking for vital signs of victims, but also brings much more safety to rescue teams. Because of mentioned problems, shortage of equipments, cost overheads and also to prevent doing several experiments repeatedly which takes a lot of time, nowadays simulated environment are commonly used. Urban Search and Rescue Simulation (USARSim) is an environment which is used in robotics and artificial intelligence methods. Since 2002, RoboCup Rescue Competitions are held as a part of the annual RoboCup World Championships. It is the purpose of RoboCup Rescue, to promote research and development in this socially significant domain in order to ultimately acquire solutions that can be used by USAR (Urban Search and Rescue) teams under real emergency situations.

In this paper, the most problem which cannot be solved by single robot and a team of heterogeneous robot that dynamically combines individual capabilities and cooperatively solves the task is needed [13]. So, we present a system of heterogeneous team of semi-autonomous robots, which explores an unknown environment. The tasks are performed cooperatively to ensure maximum information extraction of the environment, in order to assist first responders to plan rescue operations. Our agent architecture is designed to use intelligent function up to fully autonomous [13]. This includes works on autonomous multi-robot systems. For examples, mapping a vast area with multiple robots [14] and research on exploration under constraint of wireless networking. In order to improve simultaneous localization and mapping we decided to migrate from ordinary SLAM to 6D SLAM. More, last year in Graz competitions we had used compound of Bug algorithm and Vector Field Histogram (VFH) in order to preventing to strike any obstacles and barriers which maybe are between our agent and target but in this competitions we are going to use camera in order to avoid obstacles too, in order to have better performance and lower faults. More, it is good to mention that we use QT development framework, which is C++ cross-platform library.

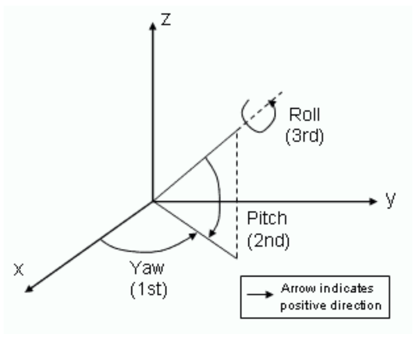
2 Autonomous Multi-Robot Exploration

In the past years, robots have been used in different places with different aims, for example in industry, military, underwater, in the space and etc. Mobile communication networks and multi-agent systems mean that it is now possible to develop robot systems for an ever wider range of tasks. A multi robot system can be highly beneficial for exploration, which is a core robotics task. Application domains include for example surveillance, reconnaissance, planetary exploration or rescue missions. When using a team of robots, the overall performance can be much faster and more robust.
Generally, there has been significant progress in exploration by single robots, but multi-robot exploration remains a new field of study having several open problems, such as how to coordinate robots, how to merge information obtained by several robots, and how to deal with limited communication [2]. After this short description about necessity of mutli-robot exploration it is time to explain our approach and solution in order to optimize and using mixture of some basis solution. First of all it is good to mention that in general, there is a popular basis exploration which is famed to the Frontier-Based exploration algorithm which introduced by Yamaychi in 1997. In Frontier-Based exploration algorithm, frontier is defined as the collection of regions on the boundary between explored and unexplored spaces of the environment. So in this approach a robot moves to the nearest frontier, which is the nearest unexplored area. By moving to the frontier, the robot explores new part of the environment. This new explored region is added to the map that is created during the exploration.

Obviously, multi-robot systems are so interesting option for exploration as they can lead to a significant speed-up and increased robustness. There are several possibilities to extent the frontier based algorithm to a multi robot version. We divide our approach in two section, first we have a communication station in our environment, in second state we don’t have a communication station, so we have a group of robots which we can name them “Robot Cluster”. According to the result from Role-Based Autonomous Multi-Robot Exploration by Julian de hoog and A.Visser. the second state has better performance and speed in exploring an environment. Here we inspired from second state, and combine it with Frontier-Based exploration algorithm and make some changes on it so we have a new approach in order to autonomous exploration of constraints wireless networking[3]. The more any robot knows about the other robots’ actions, the easier it is to efficiently coordinate the team effort. So in exploration under cover of wireless network with a communication station pointed goal will be more reachable. In our approach we should use different type of robots, and we inspired from Stem Cell of creatures. We inspired from Stem Cell and called it Stem Theory[A], here it is good to mentioned that, there are two types of Stem Cells in general, first is Embryonic Cells and second one are Adult Cells, here presented how can divide robots into two groups. At first when our robots want to start exploration or doing multi-robot tasks and when they are in same characters and they don’t have any special different in their tasks based on their specifications. This firs concept is similar to the embryonic cells which in the beginning of this type of Stem Cells life haven’t any characters and after that they can convert to the cells based on mammalian cells and body requirement. In following, in the second state when the robots tasks and positions have been determined, when a robot missed in the map or can’t do its assigned task, in this condition, we define that how we can inspire from adults Stem Cells to fix this problems that all of the system work properly as before.

3 SLAM (Simultaneous Localization & Mapping)

An important task in mobile robots is generating consistent environment maps, based on the collected sensor data. Since manual environment mapping is a tedious job, the robotic mapping problem has drawn a lot of attention in the research community. Recently, the focus shifted from planar 2D maps towards 3D mapping. 3D maps outperform 2D maps for many purposes, such as obstacle avoidance, object recognition and scene understanding. Based On [5] some groups have attempted to build 3D volumetric representations of environments with 2D laser range finders. Thru et al. [6] use two 2D laser range finders for acquiring 3D data. One laser scanner is mounted horizontally and one is mounted vertically. The latter one grabs a vertical scan line which is transformed into 3D points using the current robot pose. The horizontal scanner is used to compute the robot pose. The precision of 3D data points depends on that pose and on the precision of the scanner. 3D laser scanners have been used in [7], [8] and [9]. A 3D laser scanner generates consistent 3D data points within a single 3D scan. Recently, different groups employ rotating SICK scanners for acquiring 3D data [10] i.e. rotate scanner around the vertical axis. They acquire 3D data while moving, thus the quality of the resulting map crucially depends on the pose estimate that is given by inertial sensors, i.e., gyros. CCD-cameras that provide a view of the robot's environment has been used for collecting necessary information in order to make 3D SLAM e.g., stereo cameras and structure from motion, have difficulties providing reliable navigation and mapping information for a mobile robot in real-time. Algorithms do not consider all six degrees of freedom. Approaches that have been considered in our group were KALMAN filtering [4], and Scan Matching Based methods. In KALMAN filtering based algorithm our goal is to estimate the 6D pose of a robot equipped with any kind of sensor capable of detecting 3D landmarks in its environment [4].In this situation robot position is presented in formula 1.



$$\mathbf{x_v} = [x\ y\ z\ \phi\ \chi\ \psi]^T$$
$$\phi : \quad yaw,\ \chi : \ pitch,\ \psi : \ roll$$

Formula 1

And observed landmarks be shown as

$$\mathbf{y_i} = [x_i\ y_i\ z_i]^T$$

Formula 2

As common in the SLAM literature, we model Poses Matrix in (6+3*k)*1 Matrix (Formula 3)

$$\mathbf{x_v} = [\overbrace{x\ y\ z\ \phi\ \chi\ \psi}^{Robot\ Position}\ \underbrace{x_1\ y_1\ z_1}_{Landmark\ 1}\ \ldots\ \underbrace{x_i\ y_i\ z_i}_{Landmark\ i}]^T$$

Formula 3

Summary of algorithm presented in below,

$$\begin{cases} x_k = x_{k-1} + R_{11}x_u + R_{12}y_u + R_{13}z_u \\ y_k = y_{k-1} + R_{21}x_u + R_{22}y_u + R_{23}z_u \\ z_k = z_{k-1} + R_{31}x_u + R_{32}y_u + R_{33}z_u \\ \phi_k = \phi_{k-1} + \phi_u \\ \chi_k = \chi_{k-1} + \chi_u \\ \psi_k = \psi_{k-1} + \psi_u \end{cases}$$

Formula 4

1. Predict new robot pose x^vk|k−1 using (4). O(1)

$$\begin{aligned} P_{k|k-1} &= \frac{\partial f}{\partial x} P_{k-1|k-1} \frac{\partial f}{\partial x}^T + Q_k \\ &= \left(\begin{array}{c|ccc} \frac{\partial f_v}{\partial x_v} P_{xx} \frac{\partial f_v}{\partial x_v}^T & \frac{\partial f_v}{\partial x_v} P_{xy1} & \ldots & \frac{\partial f_v}{\partial x_v} P_{xyL} \\ \hline P_{y1x} \frac{\partial f_v}{\partial x_v}^T & & & \\ \ldots & & Unmodified & \\ P_{yLx} \frac{\partial f_v}{\partial x_v}^T & & & \end{array} \right) + Q_k \end{aligned}$$

Formula 5

2. Modify the covariance following (5). O(N)

$$z_i = h_i(\mathbf{x_v}, \mathbf{y_i})$$

$$z_k = \begin{pmatrix} r \\ \alpha \\ \beta \end{pmatrix} \quad \rightarrow \begin{cases} r : \text{Range (distance) to the landmark} \\ \alpha : \text{Yaw (azimuth) to the landmark} \\ \beta : \text{Pitch (elevation) to the landmark} \end{cases}$$

$$\begin{aligned} r &= \sqrt{\bar{x}_i^2 + \bar{y}_i^2 + \bar{z}_i^2} \\ \alpha &= \arctan \frac{\bar{y}_i}{\bar{x}_i} \\ \beta &= -\arctan \frac{\bar{z}_i}{\sqrt{\bar{x}_i^2 + \bar{y}_i^2}} \end{aligned}$$

Formula 6 , 7 , 8

3. Predict observations hi (6) and Ja-cobins of hi. O(N)

$$S_k = \frac{\partial h}{\partial x} P_{k|k-1} \frac{\partial h}{\partial x}^T + R_k$$

Formula 9

4. Compute the whole matrix S (9). O(N3)

$$K_k = P_{k|k-1} \frac{\partial h}{\partial x}^T S_k^{-1}$$

Formula 10

5. Inverse of S and computation of K_k (10). O(N3)

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k \tilde{y}_k$$

Formula 11

6. Update the filter state vector x^vk|k using (11). O(N2)

$$P_{k|k} = (1 - K_k \frac{\partial h}{\partial x}) P_{k|k-1}$$

Formula 12

7. Update the filter covariance Pk|k using (12). O(N3)

Scan Matching based approaches usually use Iterative Closest Point (ICP)[12] in order to solve most important sub problems which involves finding transformation consists of a rotation R and a translation t which minimize the following cost function (13)for the acquired sets of 3D points, M and D.

$$E(\mathbf{R}, \mathbf{t}) = \sum_{i=1}^{N_m} \sum_{j=1}^{N_d} w_{i,j} \left|\left| \mathbf{m}_i - (\mathbf{R}\mathbf{d}_j + \mathbf{t}) \right|\right|^2$$

Formula 13

Presented Approaches in [5], [11] is based on Scan Matching which includes following steps in summarized:

- Odometry extrapolation

  The Odometry is extrapolated to 6 degrees of freedom using previous registration (To create a correct and consistent model, the scans have to be merged into one coordinate system. This process is called registration) matrices.

- Calculating Heuristic Initial Estimations for ICP[12] Scan Matching

  For the given two sets M and D of 3D scan points stemming from the 3D scans, our heuristic computes two octrees based on these point clouds. The octree's rigid transformations are applied to the second octree, until the number of overlapping cubes has reached its maximum

- Scan Registration

> *Given two independently acquired sets of 3D points, M (model set, |M| = Nm) and D (data set, |D| = Nd) which correspond to a single shape, we want to find the transformation consisting of a rotation R and a translation t that satisfy by formula 13.*

Because of high time complexity and high memory leakages in KALMAN Filter, we decided to develop proposed approach in [5], in order to implement 6D SLAM.

4 Explorations

Exploration is one of the most effective tasks that each robot in robotic teams must do in the best way. Last year we had implemented just Sick laser and sonar sensor in order to avoid obstacles. In that algorithm first we used Bug Algorithm for avoiding any obstacles. Then we worked on Vector Field Histogram (VFH); therefore, we reached to this conclusion that it is better that we apply a compound of that two methods and algorithms. Therefore, in our compound algorithm we define a Cost function which use from sonar data for selecting the best trajectory. But in this year in our improved algorithm, exploration has three levels; Obstacle avoidance, Path planning and Victim detection that we have integrated them with each other and then called them OPV. In order to participate in Singapore 2010 we indicate same OPV algorithms for two rounds of competitions. So we describe our OPV algorithm in this part:

In Obstacle avoidance we use Image processing and SICK laser to detect all kind of obstacles together. Before running the main core of Obstacle Avoidance algorithm, we must realize which kind of environment our agents are?

In order to use this pattern we choose one of the following methods:

1) If the floor of our agent's environment is single color or colors of it almost are the same, first technique will be used.
2) If the floor of our agent's environment is multi colors, second technique will be used.

To recognize this problem we will need to understand that the floor plane contains more than one pixel color. While we could detect both colors and perhaps merged them in some way an easier approach is to make an assumption that the immediate foreground of the robot is obstacle free. If we were to sample the colors in the lowest part of the image which is the immediate space in front of the robot we could use these color samples and find them in the rest of the image. By searching for all pixels who share the same or similar color to those pixels in this sample space we can theorize that those pixels are also part of the floor plane. After this method we will be able to use one of the following techniques.

4.1 Edge reorganization

As shown in figure 1, robot is located in the flat floor with four obstacles.

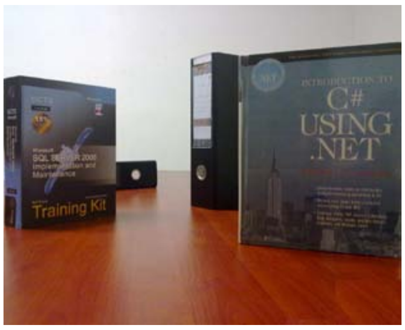


Fig.1- Robot sight

The following algorithms will refer to aspects of this image and exploit attributes are common in obstacle avoidance scenarios. By looking at this image we can see that the floor is more or less a single color with the obstacles being different in many ways than the ground plane. First of all we use Canny technique to derivation edges of the obstacle to produce an edge only version of the main picture. Now it is possible to realizing obstacles somewhat outlined by the edge detection routine. This helps to identify the objects but still does not give us a correct bearing on what direction to go in order to avoid the obstacles. After that we must understand which obstacles would be hit first if the robot moved forward. For beginning this step we use the Side Fill Module (this module which fills the black area of an image starting from the specified side and proceeds until a non-black pixel is found). In the case of filling from the top side to the bottom of the image you can think of water drops starting from the top of the image and stopping when they encounter a non-black pixel. As the water fills the image the dark area on top becomes white. This function is very useful to create a silhouette outline of an edge boundary used for skyline or ground plane detection. To fill in the empty space at the bottom of the image as long as an edge is not encountered. This works by starting at the bottom of the image and proceeding vertically pixel by pixel filling each empty black pixel until a non-black pixel is seen. The filling then stops that vertical column and proceeds with the next.

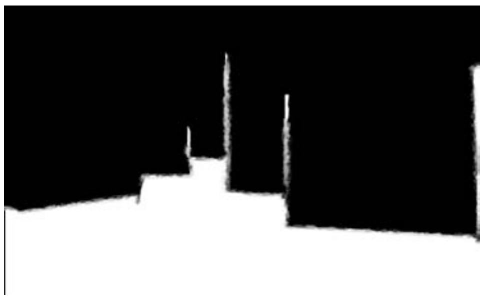


Fig. 2 - Using Side Fill to Fill from Below

Single width vertical lines that appear in the image are caused by holes where the edge detection routine failed. As they specify potential paths that are too thin for most any robot we want to remove them as possible candidates for available robot paths. We do this by using the Erode Module (this module performs an erosion routine. Objects that are connected with other objects will become separated. Objects that are too thin may disappear entirely. This module is useful for removing noise from an image. Eroding an image by a large amount will remove all small objects and cause remaining larger ones to have smoother boundaries) and just eroding or shrinking the current image horizontally by an amount large enough such that the resulting white areas would be large enough for the robot to pass without hitting any obstacle.

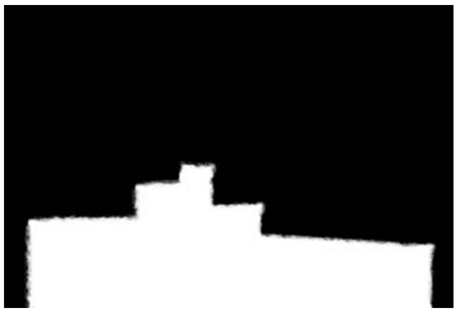


Fig. 3 - Using Erode Module (Creating Horizontal Eroded)

After it, we now need to identify the highest point in this structure which represents the most distant goal that the robot could head towards without hitting an obstacle. Based on the X location of this point with respect to the center of the screen you would then decide if your robot should move left, straight, or right to reach that goal point. To identify that location we use the Point Location Module (this module provides a quick way to identify specific coordinates within the image based on their location. For example, you may want to know the position of the highest point of any non-black pixel within the image to identify a maximum peak. Note that this module operates on a binary image. The module is often used after numerous morphological operations that have binaried and changed the image shape. Note that the north, south, east, west points are calculated relative to the object's Center of Gravity point.) and request the highest point which is identified by a gray square.

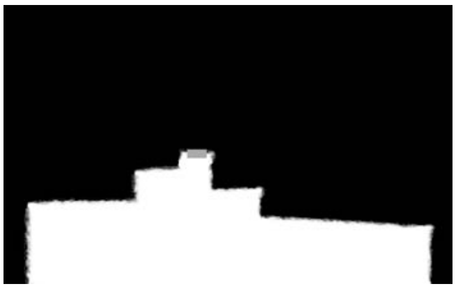


Fig. 4 - Using Point Location Module (For Finding Final Goal Point)

Finally just for viewing purposes we merge the current point back into the original image to help us gauge if that location appears to be a reasonable result.

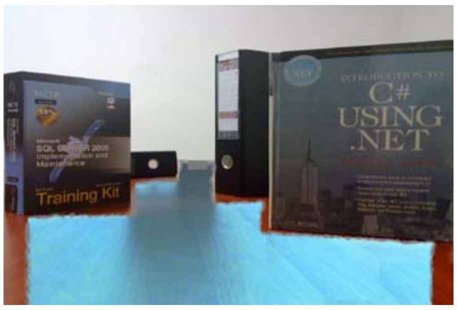


Fig. 5 – Final result which acquired from first image

This works reasonable well as long as the floor is a single color. But this is not the only way that we use to solve obstacle avoidance problem. If the floor has multi color, we use another technique to recognize the obstacles as follow:

4.2 Blob Recognition

The second technique exploits the fact that the floor is a single large object. Thus starting with the original image we can segment the image into a smaller number of colors in order to connect pixels into blobs that we can process. This grouping can use either the Flood Fill Module or the Segment Colors Module.

The next step is to isolate the largest blob in the image which is assumed to be the floor. This is done using the Blob Size Module which is set to just return the single largest blob in the image.

We then dilate this image by 2 pixels using the Dliate to close all the small holes in the floor blob.

Then we negate this image and use the same Side Fill module as before to determine the possible vertical routes the robot could take. We need to negate the image prior to this module as the Side Fill Module only fills black pixels. In the above image the object to be filled is white and thus when negated will become black.

From here on the stages are the same as the previous technique. Namely Erodo to remove small pathways, smooth the resulting object and identify the top most point. The final image looks similar to the previous technique.

4.3 Victim Detection

The last step of OPV is to find victims in less time with the maximum accuracy but in new environment. So, we have some restrictions to use agents and sensors so we focus on using camera in order to use image processing same as Obstacle avoidance method .In addition we use victim sensor instead of main tool to detect victims. Our agents extract features from the environment with image processing technique and classify them into victim parts. Round objects are classified as heads; long thin objects are classified as legs etc. The victim detection module groups these parts by location. If there are enough parts at a particular location, it determines that there is a victim. In situations where there are not enough parts, the victim detection module drives the robot towards the potential victim till all the parts can be detected. In situations, when the robot reaches close to the object, and does not find enough body parts, the robot determines this to be a false positive.

5 conclusion and future works

After last year competitions in Austria 2009 we have improved our algorithms and tried to use new approaches in order to solve some major problems in robotic science that some of them has been described in modules in our TDP. More, we are going to use and test these new approaches in coming competitions. Since Hardware Capability in high computing has been increased by developing GPU as vector and stream processors, we want to migrate our time consuming method such as matrix multiplication in SLAM as a kernel into GPU and in this way, CUDA and OpenCL would be used as our framework. Developing more user friendly and applied interface that enable operator to control heterogeneous robots in efficiently way, is our next task for future competitions.

References

[1] Murphy, R., Casper, J., Micire, M., Hyams, J.: “Mixed-initiative control of multiple Heterogeneous robots for USAR” Technical report, University of South Florida (2000)

[2] D. Fox, J. Ko, K. Konolige, B. Limketkai, D. Schulz, and B. Stewart, “Distributed multirobot exploration and mapping,” Proceedings of the IEEE, vol. 94, no. 7, pp. 1325–1339, July 2006.

[3] “Multi robot exploration under the constraints of wireless networking”

[4] Bruno Siciliano · Oussama Khatib · Frans Groen. Springer Tracts in Advanced Robotics Volume 23

[5] Nuechter, A. Surmann, H. Lingemann, K. Hertzberg, J. Thrun, S. “6D SLAM with an Application in Autonomous Mine Mapping” in IEEE INTERNATIONAL CONFERENCE ON ROBOTICS AND AUTOMATION

[6] S. Thrun, D. Fox, and W. Burgard. “A real-time algorithm for mobile robot mapping with application to multi robot and 3D mapping”. In Proc. of the IEEE ICRA, USA, 2000.

[7] P. Allen, I. Stamos, A. Gueorguiev, E. Gold, and P. Blaer. AVENUE: “Automated Site Modeling in Urban Environments”. In Proc. IEEE 3DIM, Canada, 2001.

[8] M. Hebert, M. Deans, D. Huber, B. Nabbe, and N. Vandapel. “Progress in 3–D Mapping and Localization”. In Proc. SIRS, France, 2001.

[9] H. Surmann, A. N¨uchter, and J. Hertzberg. “An autonomous mobile robot with a 3D laser range finder for 3D exploration and digitalization of indoor environments. Robotics and Autonomous Systems”, 12 2003.

[10] Andreas N□uchter, Kai Lingemann, and Joachim Hertzberg ,”Extracting Driveable Surfaces in outdoor 6D SLAM”, VDI BERICHTE

[11] Bayu Slamet, Max Pfingsthorn. ManifoldSLAM: “a Multi-Agent Simultaneous Localization and Mapping System for the RoboCup Rescue Virtual Robots Competition”

[12] Paul J. Besl and Neil D. Mckay. “A method for registration of 3-D shapes. IEEE Transactions on Pattern Analysis and Machine Intelligence”, 14(2):239–256, February 1992. ISSN 0162-8828. doi: 10.1109/34.121791.

[13] Birk, A., Kenn, H.: “control architecture for a rescue robot ensuring safe semi-autonomous operation”. In Kaminka, G., U. Lima, P., Rojas, R., eds.: RoboCup02: Robot Soccer World Cup

[14] Birk, A., Carpin, S.: “Merging occupancy grid maps from multiple robots”. IEEE Proceedings, special issue on Multi-Robot Systems