Cart-O-matic project* : autonomous and collaborative multi-robot localization, exploration and mapping.

Antoine Bautin[1], Philippe Lucidarme[2], Remy Guyonneau[2], Olivier Simonin[1], Sebastien Lagrange[2], Nicolas Delanoue[2] and Francois Charpillet[1]

***Abstract*— The aim of the Cart-O-matic project was to design and build a multi-robot system able to autonomously map an unknown building. This work has been done in the framework of a French robotics contest called Defi CAROTTE organized by the General Delegation for Armaments (DGA) and the French National Research Agency (ANR). The scientific issues of this project deal with Simultaneous Localization And Mapping (SLAM), multi-robot collaboration and object recognition. In this paper, we will mainly focussed on the two first topics : after a general introduction, we will briefly describe the innovative simultaneous localization and mapping algorithm used during the competition. We will next explain how this algorithm can deal with multi-robots systems and 3D mapping. The next part of the paper will be dedicated to the multi-robot path-planning and exploration strategy. The last section will illustrate the results with 2D and 3D maps, collaborative exploration strategies and example of planned trajectories.**

I. INTRODUCTION

Localization and mapping become the basis of many mobile robotics systems. Vacuum cleaners and mowers are great illustrations in tune with the times. Such systems may also be of prime interest for defence, military applications and rescue [9]. In 2008, the french research agency (ANR) and the General Delegation for Armaments (DGA) launched a robotics challenge called CAROTTE (CArtographie par un ROboT d'un TEritoire - Autonomous mapping of an area with a robot). Five teams ([15], [5] and [12]) have been selected and founded to participate in this challenge organized as a robotics competition similar to [13]. Each team had to design and build an autonomous grounded robotics system able to map a planar stage of a building in less than 30 minutes. The system must output at the end of the run the following data :

- a 2D map of the building,
- a 3D map of the building,
- a topological map of the building,
- location and type of walls,
- location and classification of objects.

Three events were organized in 2010, 2011 and 2012 and the results of the five teams were scientifically measured and compared. Unfortunately the results of the comparison stay confidential but the rank of each team was published. As the reader probably understood, we were one of the team engaged in the competition. Our system reached the first overall rank during the last evaluation (2012, June) and the aim of this paper is to present and share our solution. Each selected team was specialized in a given topic and the characteristic of our team was to proposed a multi-robot solution (the reader can refer to [8] for previous works). The philosophy behind this approach is the reliability (if a robot encounters a failure, it does not compromise the mission) and the speed improvement of the mission (sharing the area between several robots decreases the exploration time).
The first part of this paper is mainly focussed on localization and mapping, the next section will present the multi-robot exploration strategy. An overview of the experimental sets and results will be described and a general conclusion ends the paper.

*This work was partially supported by the French National Research Agency (ANR) and General Delegation for Armaments (DGA) through the Cart-O-matic project in the CAROTTE challenge.

[1]MAIA Group, INRIA Lorraine, LORIA, Campus scientifique, BP 239, 54506 Vandoeuvre-les-Nancy Cedex, France. firstname.lastname@loria.fr

[2]LISA - University of Angers, 62 avenue Notre Dame du Lac, 49000 Angers , France firstname.lastname@univ-angers.fr

II. SLAM-O-MATIC

For such exploration missions, localization and mapping are clearly key items of the development of the architecture. We proposed a novel SLAM algorithm based on scan matching called Slam-O-matic [11]. This algorithm is odometry-free and only requires LIDAR data. It is based on scan matching: the key idea is to find the transformation (two translations (Δ_y and Δ_Ψ) and one rotation (Δ_Ψ) for 2D SLAM) that offers the best matching between the LIDAR data and the known map. The principle of Slam-O-matic is similar to the one used for Hector Slam [10] which is based on the computation of the map derivatives and use the Gauss-Newton algorithm to maximize the matching between scan data and the map. Slam-O-matic does not require the computation of the derivatives and uses the Nelder and Mead algorithm for minimizing the distance between scans and known map (C_d on Equation 1). Nelder and Mead is a derivative-free optimization algorithm.
SLAM is thus reformulated as an optimization problem where Δ_x, Δ_y and Δ_Ψ are the parameters to optimize. The objective function is the sum of absolute distances between each end-point of the scan and each obstacle of the map is the scalar to minimize:

$$Cd(\Delta_x, \Delta_y, \Delta_\Psi) = \sum_{i=1}^{n} \sqrt{(X_s^i - X_m^i)^2 + (Y_s^i - Y_m^i)^2)} \quad (1)$$

where :

- X_s^i, Y_s^i are the coordinates of the i^{st} point of the scan data according to $\Delta_x, \Delta_y, \Delta_\Psi$.
- X_m^i, Y_m^i are the coordinates of the closest occupied cell of the map in regard with the i^{st} point of the scan data.

The environment is represented as a grip map [16], i.e. a value associated to each cell of the map is representing the current estimation of chances of having an obstacle. When the occupancy reaches a given threshold (half of the maximum value in practice) the cell is considered as an obstacle otherwise it is considered as a free space. For each cell of the map considered as an obstacle, the distance to the closest occupied cell is computed as shown on Figure 1. Unlike existing algorithm [1] where the distance is approximated, the Euclidean distance is here pre-computed thanks to a Look-Up-Table.

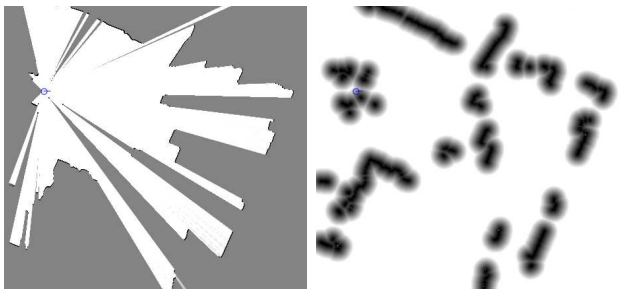

Fig. 1. Occupancy grid map (Left) and associated map with locally computed distances (Right)

When a new LIDAR scan is available the location of each end-point of the scan is located in the maps (occupancy and distances map) based on the assumption that the the previous estimated pose is the best known. It becomes thus very simple and fast to compute the cumulated distance between the new scan and the known map (Figure 2). This principle allows a fast estimation of the cumulated distances for any given transformation (Δ_x, Δ_y and Δ_Ψ). In other words, this provides a quick numerical evaluation of the mathematical function $Cd(\Delta_x, \Delta_y, \Delta_\Psi)$.

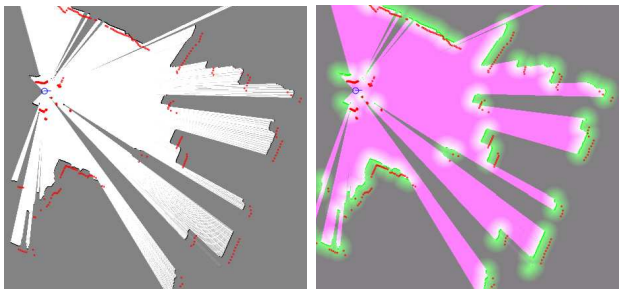

Fig. 2. New scan located in the map (Left) and surimposed maps: occupancy, distances and new scan (Right)

The first intuitive idea is to take advantage of the distance map (that can be seen as a gradient) to find the attractive direction of the transformation as it is done in [1]. Unfortunately, such gradient descent approach needs parameters tuning (size of the steps for example) that may prevent the algorithm to converge and may be sensitive to inital conditions. We preferred the Nelder and Mead algorithm (also called downhill simplex method) that is based on iterative transformations of a simplex defined in the search space. A map illustration is presented on Figure 3.

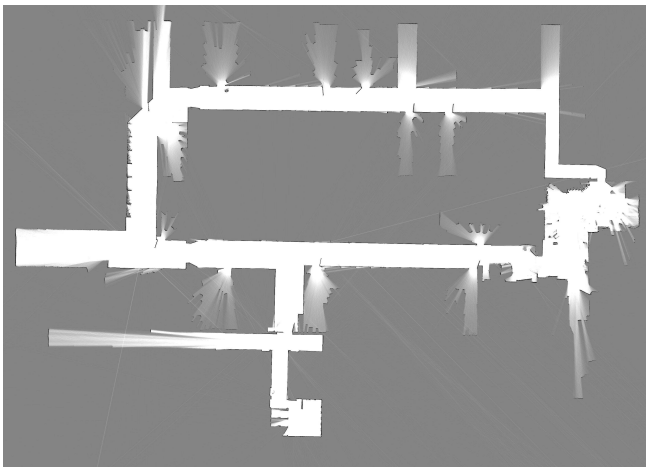

Fig. 3. Illustration of a map (63m long building) created without loop closure.

III. Multi-robot SLAM

As explained in the introduction our objective is multi-robot exploration that necessarily involves multi-robot SLAM. In 2011, we experimented the following strategy : each robot locally computes the best location for its own current scan data and send to the other robots the result of the optimization (best computed location and scan data). The other robots have no more computation to perform while the optimization has been previously done by the involved robot. They just have to update their maps with the received information. This strategy takes advantage of the multi-robot to perform distributed computation. It also ensures (if we assume there is no communication failure) that all the robots have the same map. Unfortunately, in practice, this solution appeared to be unsatisfactory due to communication problems. With a complete sharing of the information of each robot, the wireless bandwidth quickly saturated due to the amount of sent data.

In 2012, based on our experience, the global map was not computed on line: each robot computes it own local map and stores scan data in memory. At the end of the mission, raw data were sent via a wired network to a central laptop for computing a single map for the whole system. This acts exactly as the solution used in 2011, except that scan data are gathered on a central computer to avoid communication failure.

IV. RGB-3D Mapping

As explain in the introduction, a 3D map of the building was requested. Each robot was equipped with a RGB-D sensor that provide RGB images and letter ”D” stands for depth image. This combination makes the acquisition of a 3D

colored image as illustrated in Figure 4. A popular solution for realization of such a device is the KINECT™.

Fig. 4. Raw output of the RGB-D camera, and illustration of 3D image forged by combination of depth and color information.

We knew, according to the rules, that the floor was planar in the explored building. Once the robot are located in the 2D map it becomes easy to build a 3D map since the RGB-D sensor of each robot is calibrated before the mission. Calibration consists in estimating the transformation between the LIDAR and the KINECT. First step consists in computing roll and pitch while the robot is resting on a planar floor (ground is used as a reference). Next step consists in estimating yaw in front of a wall : the wall is simultaneously observed by the LIDAR and the RGB-D sensor. A line and a plane are extracted respectively from the scan and 3D data that provides the yaw angle between the LIDAR and the RGB-D sensor. Similar operations are performed for estimating translation parameters. Once sensors are calibrated, 3D data can easily be located according to the 2D map. Figure 13 shows an illustration of the 3D map.

V. Multi-robot exploration

Our multi-robot exploration strategy is frontier-based [17] i.e. the targets assigned to robots are borders between known and unknown cells. The problem consists in assigning a frontier to each robot during the exploration process. The originality of the approach is to favour the distribution of robots among the frontier *directions*. For this purpose, we do not only take into account the distance between robots and frontiers, but we also consider the notion of rank of a robot towards a frontier, by counting how many robots are closer to the frontier than the considered one. By reasoning on ranks instead of distances, two close robots will be assigned on frontiers having distinct directions where they will be in first position whatever the distances. Such an approach tends to separate robots on different directions favouring a well balanced assignation on frontiers.

To cooperate, each robot broadcasts periodically its location and a sub-sampled map of the environment. Each robot autonomously decides its next target when it has reached the previous one. This decision is based on the robot current available information.

To formally define the algorithm, let's introduce the following notations :

- $\mathcal{R}$ the set of robots, $\mathcal{R} : \{\mathcal{R}_1 ... \mathcal{R}_n\}$ with $n = |\mathcal{R}|$ the total number of robots,
- $\mathcal{F}$ the set of frontiers, $\mathcal{F} : \{\mathcal{F}_1 ... \mathcal{F}_m\}$ with $m = |\mathcal{F}|$ the number of frontiers,
- $\mathcal{C}$ a cost matrix with $\mathcal{C}_{ij}$ the path distance from robot $\mathcal{R}_i$ to frontier $\mathcal{F}_j$,
- $\mathcal{A}$ an assignment matrix with $\alpha_{ij} \in [0, 1]$ defined as follows :

$$\alpha_{ij} = \begin{cases} 1 & \text{if robot } \mathcal{R}_i \text{ is assigned to } \mathcal{F}_j, \\ 0 & \text{otherwise.} \end{cases}$$

Let RK_{ij} be the rank of the robot $\mathcal{R}_i$ towards the frontier $\mathcal{F}_j$. RK_{ij} is equal to the number of robots which are closer to the frontier than the robot $\mathcal{R}_i$. Algorithm 1 formally defines the algorithm, named *MinPos*, processed by each robot for computing its assignment.

Algorithm 1: *MinPos*

Input: $\mathcal{C}$ cost matrix
Output: α_{ij} assignment of robot $\mathcal{R}_i$

foreach $\mathcal{F}_j \in \mathcal{F}$ **do**

$$RK_{ij} = \text{Card } (\tilde{\mathcal{R}}) \; with \; \tilde{\mathcal{R}} = \{\forall \mathcal{R}_k \in \mathcal{R} \mid \mathcal{C}_{kj} < \mathcal{C}_{ij}\}$$

end

$j = \underset{j|\mathcal{F}j \in \mathcal{F}}{\text{argmin}} \; RK_{ij}$ (*If several* RK_{ij} *are minimum then choose the one with lowest cost* $\mathcal{C}_{ij}$)

α_{ij} = 1

Figure 5 illustrates the exploration with 3 robots in a $35m^2$ rooms environment. The trajectories of each robot demonstrate the validity and efficiency of the proposed approach, indeed each robot explored a different part of the environment.

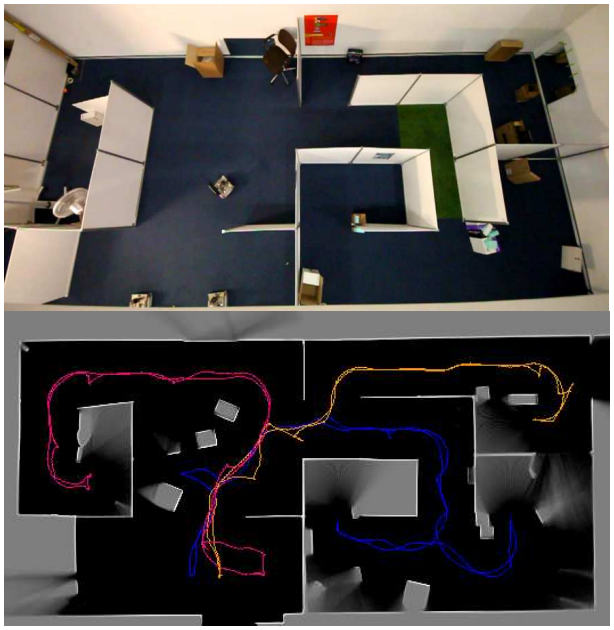

Fig. 5. Photo of the environment and map with trajectories resulting from an exploration with 3 robots.

Simulation results demonstrated that our *MinPos* algorithm outperforms the nearest frontier algorithm [18]. Depending on the environment topology and the number of robots, our algorithm outperforms or gives similar results than

utility greedy algorithm [4]. However, our approach has a lower computational complexity ($O(nm)$) than the greedy algorithm ($O(n^2m)$).

Figure 6 compares the exploration times given in simulation steps of different methods, while varying the number of robots. The methods compared are the nearest frontier algorithm [18], the Burgard et al. greedy-based algorithm [4] and our *MinPos* algorithm, on an hospital section environment. Results shown are an average of 60 runs of each algorithm with a given robot count. We observe that the Burgard et al. and *MinPos* algorithms are more efficient improving by 13% on average the number exploration steps required to fully explore the environment. We improve the greedy approach when the number of robots is low, as *MinPos* forces a well balanced spatial distribution. Details can be found in [3].

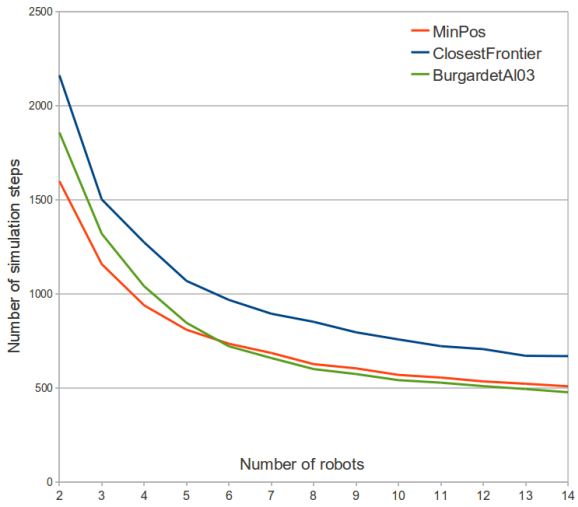

Fig. 6. Results from the exploration of the hospital environment when varying the number of robots

The computation of this novel rank criteria depends on the information of the cost matrix. To compute this matrix, distances are evaluated using a wavefront propagation algorithm [2] on a discrete environment representation. A wavefront computes path distances incrementally around a source. Here, the idea is to propagate a wavefront from each frontier. For a robot assignment, the propagation of a wavefront is stopped when it encounters its location. Thus, it gives the shortest paths (on the grid) to the frontier from all points closer than the robot's location. This is sufficient to know the distances from the other robot's location useful to compute the robot rank. This approach is computationally efficient especially when the number of robot is large, compared to computing the path distance using an A* algorithm from the frontier to every robot. Such a wavefront propagation is illustrated in Figure 7.

VI. Trajectory-planning

The wavefront propagation used to compute the robots assignment also provides paths that could be used for the robot navigation. However, it does not take into account the dynamic and nonholonomic constraints.

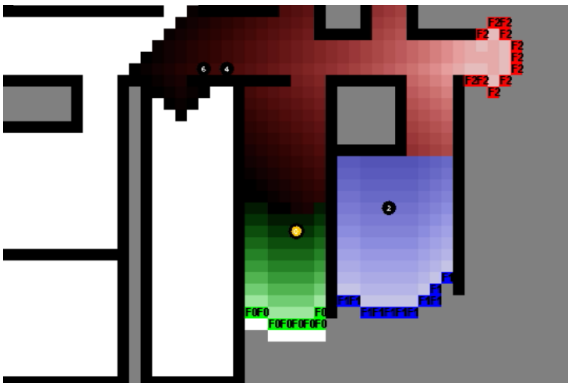

Fig. 7. Illustration . Wavefronts are stopped on the encounter of the yellow robot computing its assignment. Color code : white=explored, gray=unknown, black=walls, red-green-blue=frontier and gradient wavefront propagation result, only the wavefront closest to the frontier is shown where waves are superimposed.

To tackle this issue, we perform an A* algorithm in 4D (x, y, orientation, speed) using the wavefront distances, previously computed, as heuristics. The computational time of A* depends on the heuristics. Using the euclidean distance, as heuristic, is computationally costly. The originality of our trajectory planning is that we use the already computed 2D wavefront propagation as heuristic (introduced in section V).

Trajectory planning is quite efficient. However, as it uses the almost-shortest path the robot tends to graze obstacles. We therefore added a penalty to nodes close to obstacle with a value inversely proportional to its distance to the closest obstacle. This generates smooth and safe trajectories. Figure 8 illustrates such a planned trajectory.

To evaluate our technique we randomly draw 500 points and compute a trajectory passing by all these points in the order they were generated in. On average, the trajectory planning in an office environment (14 rooms along a corridor in a 1 million pixels image) takes :

- 264.2 ms with no heuristics,
- 20.6 ms with euclidean distance heuristics,
- 14.6 ms with the potential field heuristics (11.7 ms for the A* and 2.8 ms for the wavefront propagation).

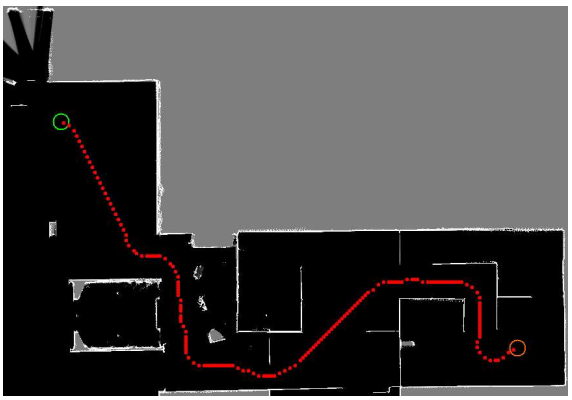

Fig. 8. Exemple of a planned trajectory keeping away from obstacles

VII. Results

A. MiniRex

MiniRex (MINI Robot for EXploration) is a robot, dedicated to the project, designed and build in our laboratory. The main specification was to design a low cost, reliable and small robot. The robot has a square shape (0.25 x 0.25m width) by 0.5m height. It is a tracked robot actuated by two DC geared motors (Faulhaber 2657 012 CR). A PC (Kontron pITX-SP - Intel Atom Z530 1.6GHz) and a real-time dedicated processor (ATmega2560) are embedded and powered by two Lithium-Polymer batteries (22.2V 3300mAh). Several sensors provide internal and external information : ultrasonic ranging and proximity sensors, voltage battery sensors, inclinometer, track encoders, KINECT™and an actuated LIDAR (Hokuyo UTM-30LX). Figures 9 and 10 show details and illustration of the robot.

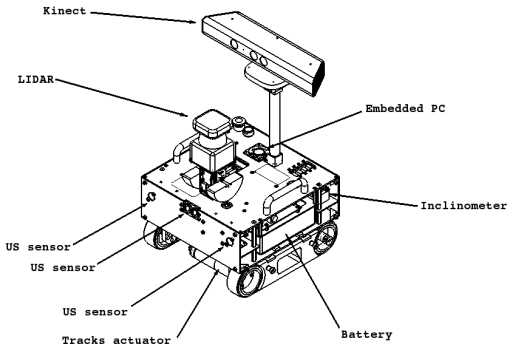

Fig. 9. Specifications of the MiniRex robot.

Fig. 10. Minirex exploring a test area in our laboratory.

Seven robots were built. According to the area size (about $120m^2$), we decided to engage only five robots in the exploration to prevent congestion during the exploration. The sixth robot was keep as a spare robot and the seventh for spare parts.

B. Results

This section presents the results of the final run during the last year competition. Figure 11 shows the global map and trajectories of the robots. The exploration task was clearly distributed between the robots and each area (not to say each room) was explored by a robot. During the mission, one robot got stuck in the gravel and was not able to reach its starting point (on the bottom of the area located in the left of the map). Despite the fact that a failure occurred during the mission, the other robots successfully end their task and the failure didn't compromise the whole mission. Figure 13 shows the 3D map built thanks to the RGB-D sensor.

Fig. 11. Map and trajectories of the robots.

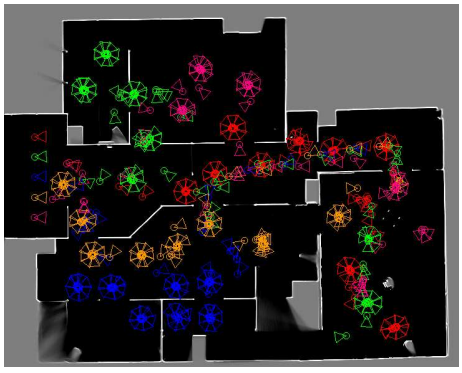

Fig. 12. Map with the location of the RGB-D captures.

Fig. 13. 3D map of the building.

For a global overview of the mission, object recognition has been illustrated on Figure 14 although this topic is not the aim of the present paper. For more information about object recognition, the reader is referred to [14].

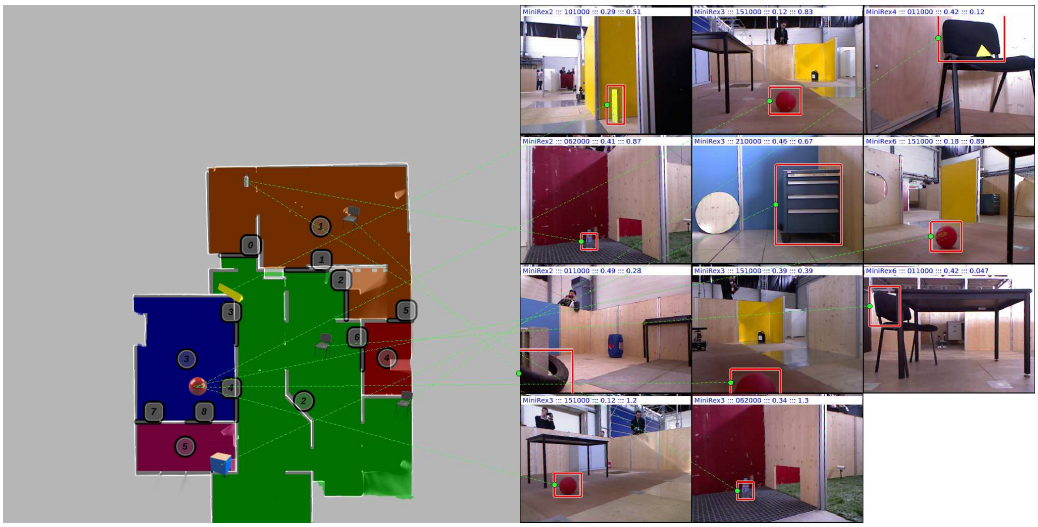

Fig. 14. Object recognition and localization.

VIII. CONCLUSIONS

This paper described an overview of the software and hardware architecture of the project Cart-O-matic. We presented a SLAM algorithm called Slam-O-matic with a quick overview of the performances. During the competition, a comparison of the map produced by each team has been performed. Detailed results are confidential, but we know that Slam-O-matic reached the first rank in term of accuracy. However, a comparison with similar algorithm (Gmapping [6], Hector Slam [10], ICP [7] ...) would be interesting to compare computation time, memory space and reliability.

We presented the *MinPos* algorithm for multi-robot exploration strategy, which uses a novel criteria ensuring a well balanced distribution of robots among different directions. Results in simulation and with MiniRex robots demonstrated the efficiency in exploration time of this algorithm. More generally, our multi-robot approach showed good robustness and efficiency during the French national robotics challenge 'Carotte', that we won in 2012. We now aim to extend this work to the exploration and mapping of dynamic environments.

REFERENCES

[1] Steux B. and H.E. Oussama. tinyslam: A slam algorithm in less than 200 lines c-language program. In *ICARCV*, pages 1975–1979, 2010.

[2] J. Barraquand, B. Langlois, and J.-C. Latombe. Numerical potential field techniques for robot path planning. In *Advanced Robotics, 1991. 'Robots in Unstructured Environments', 91 ICAR., Fifth International Conference on*, pages 1012 –1017 vol.2, jun 1991.

[3] A. Bautin, O. Simonin, and F. Charpillet. Minpos : A novel frontier allocation algorithm for multi-robot exploration. In Chun-Yi Su, Subhash Rakheja, and Honghai Liu, editors, *Intelligent Robotics and Applications*, volume 7507 of *Lecture Notes in Computer Science*, pages 496–508. Springer Berlin Heidelberg, 2012.

[4] W. Burgard, M. Moors, C. Stachniss, and F.E. Schneider. Coordinated multi-robot exploration. *Robotics, IEEE Transactions on*, 21(3):376 – 386, june 2005.

[5] D. Filliat and al. Rgbd object recognition and visual texture classification for indoor semantic mapping. In *Proceedings of the 4th International Conference on Technologies for Practical Robot Applications (TePRA).*, pages 127–132, 2012.

[6] G. Grisetti, C. Stachniss, and W. Burgard. Improved techniques for grid mapping with rao-blackwellized particle filters. *Robotics, IEEE Transactions on*, 23(1):34–46, 2007.

[7] D. Holz and S. Behnke. Sancta simplicitas - on the efficiency and achievable results of slam using icp-based incremental registration. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pages 1380–1387, 2010.

[8] Andrew Howard, Lynne E Parker, and Gaurav S Sukhatme. Experiments with a large heterogeneous mobile robot team: Exploration, mapping, deployment and detection. *The International Journal of Robotics Research*, 25(5-6):431–447, 2006.

[9] S. Noda I. Matsubara H. Takahashi T. Shinjou A. Kitano, H. Tadokoro and S. Shimada. 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*, pages 739–746. IEEE Computer Society, 1999.

[10] S. Kohlbrecher, J. Meyer, O. von Stryk, and U. Klingauf. A flexible and scalable slam system with full 3d motion estimation. In *Proc. IEEE International Symposium on Safety, Security and Rescue Robotics (SSRR)*. IEEE, November 2011.

[11] P. Lucidarme and S. Lagrange. Slam-o-matic : Slam algorithm based on global search of local minima. Brevet num. FR1155625, June 2011.

[12] A.-I Matignon, L. Mouaddib and L. Jenapierre. Coordinated multi-robot exploration under communication constraints using decentralized markov decision processe. In *International Conference on Advanced Artificial Intelligence (AAAI)*, 2012.

[13] Edwin Olson, Johannes Strom, Ryan Morton, Andrew Richardson, Pradeep Ranganathan, Robert Goeddel, Mihai Bulic, Jacob Crossman, and Bob Marinier. Progress toward multi-robot reconnaissance and the magic 2010 competition. *Journal of Field Robotics*, 29(5):762–792, 2012.

[14] Said. Gholami Shahbandi and Philippe Lucidarme. Object recognition based on radial basis function neural networks: experiments with rgb-d camera embedded on mobile robots. In *ICSCS*, 2012.

[15] L. Thorel S. Steux, B. Bouraoui and L. Benazet. CoreBot M : Le robot de la Team CoreBots préparé pour l'édition 2011 du défi Carotte. In *6th National Conference on Control Architectures of Robots*, Grenoble, France, May 2011. INRIA Grenoble Rhône-Alpes. 3 pages.

[16] W. Thrun, S. Burgard and D. Fox. *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. The MIT Press, 2005.

[17] B. Yamauchi. A frontier-based approach for autonomous exploration. In *Computational Intelligence in Robotics and Automation, 1997. CIRA'97., Proceedings., IEEE International Symposium on*, pages 146 –151, Washington, DC, USA, jul 1997. IEEE Computer Society.

[18] B. Yamauchi. Frontier-based exploration using multiple robots. In *AGENTS '98: Proceedings of the second international conference on Autonomous agents*, pages 47–53, New York, NY, USA, 1998. ACM.