# 3D laser from RGBD projections in robot local navigation

Luis V. Calderita[1], Juan P. Bandera[2], Luis J. Manso[1] and Ricardo Vázquez-Martín[3]

*Abstract*—**Social robots are required to work in daily life environments. The navigation algorithms they need to safely move through these environments require reliable sensor data. We present a novel approach to increase the obstacle-avoidance abilities of robots by mounting several sensors and fusing all their data into a single representation. In particular, we fuse data from multiple RGBD cameras into a single emulated two-dimensional laser reading of up to 360 degrees. While the output of this virtual laser is two-dimensional, it integrates the obstacles detected at any height, so it can be safely used as input for regular two-dimensional navigation algorithms (both VFH* and $\Re$-ORM have been tested). Experiments conducted on real scenarios demonstrate the usefulness and efficiency of the proposed solution, which allows the robot to reach goals while avoiding static and dynamic obstacles.**

*Index Terms*—**Mobile robots, Reactive navigation, Sensor arrays, RGBD.**

## I. Introduction

Social robots have to interact with people in daily life environments, and navigation is one of the main requisites for social robots to successfully accomplish their tasks. These environments are unpredictable and populated by many different moving objects, which make moving through these environments a challenging task.

There are two main types of navigation algorithms. *Deliberative* algorithms find clear paths towards the objective using a model of the environment. They are not well-suited for dynamic environments because the models used to compute paths get outdated quickly. *Reactive* navigation algorithms, on the other hand, use the latest sensory inputs to detect obstacles in the local environment of the robot and propose a safe direction towards the desired objective. Despite reactive algorithms can react to unpredicted situations, they lack of a complete model of the environment, and may lead the mobile agent to traps or local minima where it can be bogged down. These issues are usually solved by adding higher-level (deliberative) path planners to lower-level reactive navigation systems (i.e., using *hybrid* systems) [14].

Classical reactive navigation algorithms are based on potential fields [6]. This algorithm models repulsive and attractive forces that makes the robot moves away from obstacles, and towards the objective. Despite its theoretical usefulness, this method has strong issues: it tends to halt in traps or local minima, produces nodding trajectories in corridors and cannot navigate between near obstacles. Thus, many alternatives have been proposed in recent years. Vector Field Histogram (VFH) [2] employs local histograms to reduce sensory inputs to a set of motion alternatives. Dynamic Window Approach (DWA) [4] minimizes a

[1]Luis Calderita and L. J. Manso are with Computer and Communication Technology Department, Universidad de Extremadura, Av. de la Universidad sn, 10071, Cáceres, Spain. `lvcalderita@unex.es`

[2]Juan Pedro Bandera is with Electronic Technology Department, ETSI Telecomunicación, Campus Teatinos s/n, 29071, Málaga, Spain.

[3]Ricardo Vázquez-Martín is with Centro Andaluz de Innovación y Tecnologías de la Información y las Comunicaciones (CITIC), Edificio OTIC, Marie Curie 6, Parque Tecnológico de Andalucía, 29590, Málaga, Spain.

cost function to avoid obstacles. ND Nearness Diagram [10] and Obstacle Restriction Method (ORM) [9] follow a "divide and conquer" strategy and decompose the problem of reaching a certain goal into a set of sub-problems. Each of these sub-problems focuses on reaching a certain sub-goal while avoiding obstacles.

All these pure reactive navigation systems can make a robot reach a certain objective defined in local coordinates. However, the time consumed to reach that objective may be much longer than the optimal. Besides, reactive navigation uses to produce odd motion patterns with respect to human natural motion. Social robots, that have to interact with people in daily life environments, should avoid these behaviours to increase efficiency, but also to avoid rejection [11].

Perceptual limitations are among the key reasons for these issues of reactive navigation systems. Reachable objectives are usually set in the perceived area of the environment [3], and thus limited perception constraints the efficiency of the navigation system. In example, a robot equipped with a 180 front laser will not be able to detect obstacles behind it (unless the system incorporates some type of memory about previously perceived obstacles). Thus, it will avoid setting navigation objectives there.

Sensor fusion is a powerful mechanism to increase the perceptual abilities of a robot [1]. Instead of using a single expensive sensor to acquire data from the environment, the sensor fusion approach relies on multiple sensors, which data are fused to conform a single representation. In other words, using sensor fusion several constrained sensors can be fused to emulate a sensor that has a wider range, and that is more accurate. An emulated sensor may have very different characteristics. However, most reactive navigation algorithms are adapted to work with laser data [7]. Thus, for navigation in flat, daily life environments, making these emulated sensors behave as laser range finders is the most practical choice.

This paper proposes to fuse the data collected by an array of different sensors, mounted on a social robot, to make it navigate through a dynamic, unconstrained environment. The robot will use only a reactive navigation algorithm. In order to meet the stringent requirements of robotics software (robustness, code reuse, scalability, distribution, hardware independence, etc.) all software will be developed using the open-source Robotics Framework *RoboComp*[1] [8].

The paper firstly explains, in Section II, the overall structure of the proposed system. Then, Section III describes how different sensor readings have been fused to create a synthetic laser reading. It focuses on the particular case of fusing data obtained from different RGBD cameras. These devices are cheap, robust, accurate, and able to detect obstacles in a volume (instead of a plane, as most laser range finders). Their main limitation is their constrained field of view, but sensor fusion can solve this issue. Thus, they become the perfect candidate to test the performance of the proposed method. Section IV provides a brief description of the two reactive algorithms employed in this paper. Section V details the experimental set-up used to test the system and analyses obtained results. Finally, Section VI concludes the paper highlighting the main advantages and drawbacks of the proposes approach, and the future work that will be conducted in further related researches.

## II. Proposed architecture

As detailed above, most reactive navigation algorithms are able to process data provided by laser range finders. Thus, representing fused perceptual data as laser readings increases the usefulness, efficiency and adaptability of the proposed approach. Of course, it also allows using a real laser range finder instead of the emulated one.

Fig. 1 shows the proposed system architecture. Each blob represents a software

[1]This framework is available at www.robocomp.org

component in the *RoboComp* framework [8]. As depicted, several heterogeneous perceptual systems are fused to create an emulated laser sensor (a *virtual laser*). Data provided by this virtual sensor is employed to feed the reactive navigation algorithm. This paper tests two different algorithms: The well-known VFH* (Vector Field Histogram *) [13] and the $\Re$-ORM (Relaxed Obstacle Restriction Method), a modification of the ORM algorithm [9].

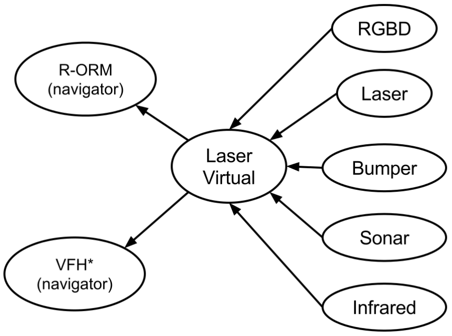

Fig. 1. Reactive navigator: System overview

## III. Sensor fusion: Generation of synthetic laser readings by fusing different sensory inputs

As explained above, a group of RGBD sensors may become a cheap, robust and more useful alternative to traditional laser range finders, if their data is correctly fused. In this paper, laserRGBDComp, a new component for the RoboComp framework, has been developed. This software component allows fusing information provided by one or more sensors (RGBD or not) into an emulated laser projection. This eases testing different navigation algorithms, as most of them require laser readings. Thanks to *RoboComp* framework, the only requisite for laserRGBDComp is to implement the same interface than a component dedicated to obtain real laser readings.

### A. *laserRGBDComp*

It is a special component, that has to connect to other components (via *proxies*) but that does not predict *a priori* the number or types of required connections. This allows the component to dynamically include new sensors in the sensor fusion process. On the other hand, this characteristic makes the design of laserRGDComp more complex, to the point that it cannot be self-generated using the component generation tool (RoboCompDSL) provided by the *RoboComp* framework [5]. Thus, the component had to be manually designed and implemented from scratch.

The first implementation of laserRGDComp fused RGBD sensors into an emulated laser reading. This fusion is based on two ideas:

- RGBD cameras provide depth maps whose pixels $(i, j)$ can be characterized by an azimuth (horizontal) angle $\alpha_{(i,j)}$, an elevation (vertical) angle $\beta_{(i,j)}$, an a z-depth value $d_{(i,j)}$. On the other hand, a laser reading $\vec{P}$ can be represented as an array of angle-distance tuples ($\sigma$, $dl$). The laserRGDComp component creates an emulated laser reading using Eq. 1.

$$\begin{aligned}\vec{P}_i(\sigma) &= \alpha_{(i,j)} \\ \vec{P}_i(dl) &= min(d_{(i,j)})\forall j\end{aligned} \quad (1)$$

  being, in RGBD depth maps, $\alpha_{(i,j1)} = \alpha_{(i,j2)}$ $\forall j1, j2$. Fig. 2 depicts how the data gathered from an RGBD sensor is employed to obtain an emulated laser reading that includes obstacles detected at different heights (*e.g.*, table tops and legs).
- Several of the previous readings can be fused into a single reading if the relative poses of each sensor are known. Thus, for a system composed of $n$ RGBD cameras, being $M_k$ the transformation matrix from camera $k$ to the local origin $(0, 0, 0)$ of the robot, Eq. 2 is employed

to fuse readings $\vec{P_k}$ into a single reading $\vec{P}$:

$$\begin{aligned} \vec{P_k'} &= M_k^{-1} \cdot \vec{P_k} \\ \vec{P_i}(\sigma) &= \alpha_{(i,j)}^{k'} && \forall k \in [1..n] \\ \vec{P_i}(dl) &= min(d_{(i,j)}^{k'}) && \forall k \in [1..n] \forall j \end{aligned} \tag{2}$$

being $\alpha_{i,j}^{k'}$ and $d_{(i,j)}^{k'}$ the azimuth values and distances for $P_k'$ pixels, respectively.

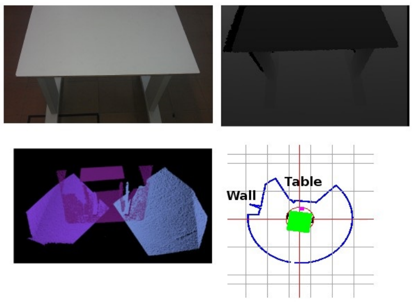

Fig. 2. Detection of obstacles using laserRGBDComp.

New sensors, such as sonar or infrared range sensors, can be easily incorporated to these fused readings. laserRGBDComp can currently fuse data provided by (i) RGBD cameras; (ii) laser sensors; (iii) any sensor data that can be represented as a sequence of 2D or 3D position-distance vectors. The components that deal with sonar sensors, infrared sensors or even bumpers in RoboComp can match these criteria, with small changes in their interfaces at most [8].

### B. *How does laserRGBDComp work?*

This subsection provides implementation details about this component. As commented above, given a number $n$ of depth images from $n$ sensors and knowing the positions of these sensors on board the robot, each transferring data is transformed to a common position. The emulated laser is located on this position. The proposed method applies Eq. 2 to obtain a synthetic laser measure, in which all collected distances are fused. If two sensors are measuring the same position, then the minimum distance is set for that position.

In order to ease its initialization, the component loads a file that describes the *InnerModel*, or specific configuration of the robot. This description consists of a kinematic hierarchical tree that describes the robot. It is essentially an internal representation of the robot. As any other kinematics chain, it allows to easily obtain the transformation matrix between any two nodes. This makes it adaptable to other robots. The file includes also additional parameters that allow to adjust it to different needs (see Table I).

laserRGBDComp requires to execute a prior calibration process in order to correctly merge data from more than one sensor on a single laser projection. The calibration process checks that the projection of each camera is coherent with its pose. The calibration is currently performed manually, through a specific "calibration mode" When in this mode, laserRGBDComp projects the point clouds of every camera according to the virtual laser position. The user manually adjust positions and orientations of each sensor to make these projections match the real distance measures. This process refines the poses initially provided by the kinematics tree. Figure 3 summarizes it at a glance. As depicted, a good distribution of sensors will have to reach a compromise between the total angle covered by the cameras, the common area and the threshold without information about the robot.

## IV. Reactive navigation

Two reactive algorithms have been employed in this paper to perform navigation. Both of them use emulated laser data as their only input. This Section briefly describes them. It also explains the modifications and improvements added to these algorithms to make them work in long-term experiments, performed in dynamic real environments.

```
#laserRGBD.conf Config file
# provided laser interface
laserRGBComp.Endpoints=tcp -p 11176
# Cinematic tree described in a kind of file called
Innermodel.
# contains the relative position of the RGBDs
relative to the position of the virtual laser we
simulate.
InnerModelPath =
/home/robocomp/robocomp/Components/Adapta/Files/
gualzruMRGBD.xml
# Number of RGBD
RGBDNumber=3
# Identifier of the node of the cinematic tree
where it is the virtual laser
LaserBaseID = base
# Minimum height to the obstacles. Useful to
discard the floor
MinHeight = 200.
# Maximum height of the obstacles. Useful to
discard the ceiling or lamps.
MaxHeight = 2000.
#Minimum height to gaps. Useful to avoid holes
or steps.
MinHeightNeg = -5000.
# number of laser measures
LaserSize = 100 #628
# Minimun range of the laser. Distance in
millimetres.
MinRange = 50.
# Maximun range of the laser. Distance in
millimetres.
MaxRange = 1500.
# Field of View of the laser in radians
FOV = 3.14
#proxy to rgbds to generate the 3D laser projection
RGBDProxy1 = rgbdbus:tcp -h gualzru3.local -p
10229
RGBDID1 = 1208240087
RGBDProxy2 = rgbdbus:tcp -h gualzru3.local -p
10229
RGBDID2 = 1103010046
RGBDProxy3 = rgbd:tcp -h robonuc.local -p
100096
RGBDID3 = 1105150157
```

TABLE I
LASERRGBD CONFIG FILE.

### A. VFH*

The Vector Field Histogram (VFH) navigation algorithm uses an stochastic representation of the environment, based on an histogram grid. This representation helps reducing the uncertainty derived from sensor errors and modelling. It considers the dynamics and shape of the robot, and provides motion commands that are specific for the employed agent.

VFH [2] and its improved version, VFH+ [12], are affected by the typical issue of reactive navigation algorithms: dealing only with local data it is not possible to guarantee global optimal behaviour. VFH* [13] algorithm addresses this issue, by checking the validity and quality of provided speed and turning commands. A* algorithm is used for this purpose. VFH* has demonstrated that these *forward* -predictive- checks allow solving scenarios in which VFH and VFH+ fail. Besides, VFH* generates faster but also smoother trajectories.

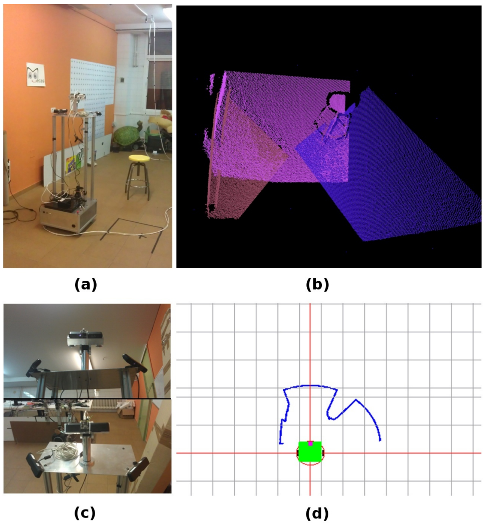

Fig. 3. (a) Current configuration of the robot; (b) Point clouds projected to laser position; (c) Detail of the pose of the RGBD cameras (Asus Xtion); and (d) Generated virtual laser.

### B. $\Re$-ORM

The $\Re$-ORM algorithm is an evolution of the Obstacle Restriction Method (ORM) proposed by Chamorro and Vázquez-Martín [3]. This algorithm divides the problem of reaching a certain objective, avoiding collisions, into a set of sub-problems. Each of these sub-problems consists on reaching a certain sub-objective. The perceived obstacle distribution around the robot defines the positions where these sub-objectives are placed. The method generates speed and turning commands that allow the robot reaching the final objective by navigating through a sequence of sub-objectives.

The vanilla representation of the $\Re$-ORM has several issues when applied to real robots employed in long-term experiments and dynamic environments. The following modifications have been incorporated to the $\Re$-ORM algorithm to solve these issues:

- Once $\Re$-ORM sets a sub-objective, it has to reach it before computing a new one. In real experiments, this decision causes very inefficient situations: a temporal occlusion of the final objective forces the selection of distant sub-objectives that attract the robot even when the objective is visible again. This issue has been solved by making the robot immediately forget the current sub-objective, and target the final objective when it is perceived as reachable.
- In order to reduce nodding produced by fast changes in the current sub-objective, an *inertia* factor has been included. Once a new sub-objective is selected, it is kept as the current goal for a certain time, that depends on the *inertia* value.
- Although $\Re$-ORM is quite robust in general, it can lead the robot to a halt position, in which it is unable to move towards the sub-objective nor to set a new one. A *patience* parameter has also been included in the code to deal with these situations. If the robot stands in the same position for a time longer than this *patience* value (and it has reached no objective yet), it spins for a certain time until a new sub-objective is set.

## V. Experimental results

Two sets of experiments were conducted to check the validity of the proposed fused sensor for navigation purposes. The first set of test involved simulated environments. These tests compared VFH* and $\Re$-ORM. Its results allowed both to select the most adequate navigation algorithm for the employed scenarios and sensors, and to refine its performance.

The second set of experiments was conducted in real scenarios. Long-term experiments in dynamic environments were conducted, and their results were used to refine employed algorithms. Final results validate the proposed sensor fusion approach for real robots, working in daily life environments.

### A. Simulation tests

Simulation tests are executed in complex virtual scenarios, that include randomly moving objects (Fig. 4). The first set of tests conducted on these scenarios checks the validity of the proposed system, and compare VFH* and $\Re$-ORM algorithms. In these tests, random goals appear for the robot, and it has to approach them closer than a certain distance threshold $d_g$. If the robot manages to approach the goal before a certain time $t_{out}$ lasts, the goal is marked as reached. Otherwise, it is marked as failed.

Fig. 4 depicts a screen shot of the simulator, taken while running one experiment. Each experiment lasted for more than eight hours, in order to guarantee the stability and robustness of the developed software. As depicted, both reached and non-reached goals are marked.

Both algorithms obtained fairly good results in these tests. Most unreachable goals were located too near to the obstacles (no restriction was imposed to the goal generator). However, $\Re$-ORM demonstrated a better ability to avoid being bogged in local minima, and reached the goals in shorter time. Thus, it was selected as the reactive navigation algorithm to be employed in the robot. Further tests demonstrated that $\Re$-ORM, in its vanilla version, had several issues that should be corrected before conducting experiments in real robots. These modifications have been detailed in Section IV. The rest of the experiments described in the Section use the $\Re$-ORM algorithm, including these modifications.

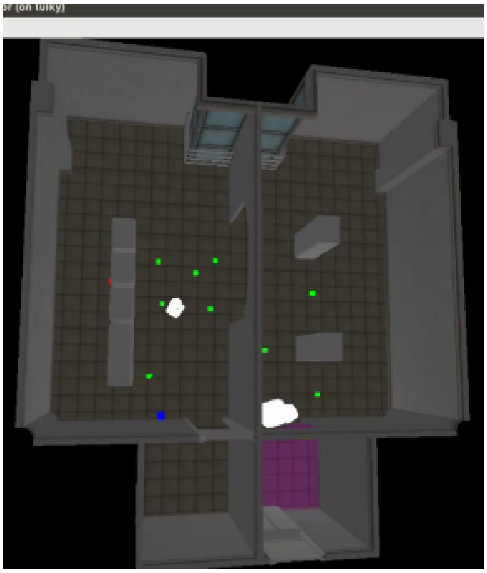

Fig. 4. Employed simulator: Green boxes represent objectives that have been already reached by the robot. The blue box is the next objective. The red box is an objective that was not reached before time out.

One of the first detected issues was related to robot speed, specially to robot rotation speed. In the first executed simulations, the robot was able to move at very high speeds. This ability allows it to aim the goal perfectly and in a very short time. However, (i) real robots have constrained advance and rotation speeds; and (ii) constrained speeds are required for safety reasons, for a robot interacting with people in daily life environments.

Thus, the speed limits of the simulated robot were set to realistic values in a second iteration[2] This increases the time required to reach navigation goals. Besides, it usually forces the robot to describe odd trajectories, as the space required to perform turning motions grows. Figure 5 depicts this behaviour, showing the path followed by the robot to reach a new goal from its pose in the previous frame.

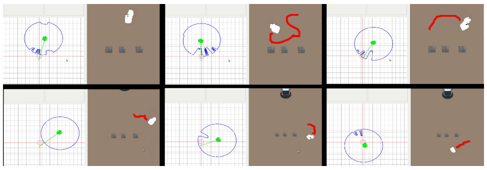

Fig. 5. Effects of a constrained rotation speed in the followed paths.

The robot was still able to navigate correctly despite these drawbacks. However, the causes of these odd behaviours were analysed in order to improve obtained motion in terms of speed and naturalness (from a human perspective). This analysis revealed some parameters that can be adjusted to reduce turning radius without increasing maximum rotation speeds. More precisely, (i) the effects of close obstacles in this speed were tuned; (ii) the speed reduction rate, when reaching the goal, was also adjusted; and (iii) the sub-goal selection policy was also changed as detailed in Section IV. These changes improved obtained trajectories (Fig. 6) while preserving safety.

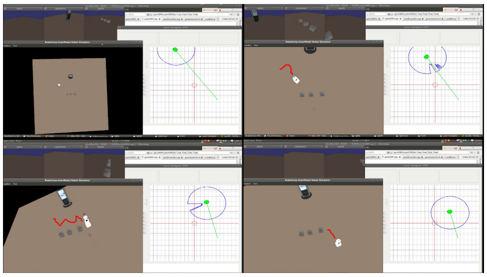

Fig. 6. Examples of robot paths after improvement.

Fig. 7 shows an example of robot navigation in a complex environment. This environment includes three boxes strategically placed, so that the robot can easily be stalled between them (VFH* was not able to move through these boxes in most cases. On the contrary, $\Re$-ORM can set a path through them if required). Additional obstacles have been added to increase complexity. The perimeter was limited to a square grid

[2]It is worthy to mention that a simulation set-up using an 'ideal' robot (in terms of speed) has been kept in the research group. This simulator is used to test higher level, symbolic decision algorithms without lower level influences.

of 5x5 meters. The robot has to navigate without collisions in these scenario, that has little room for maneuver. If the robot is able to navigate in long-term tests in this environment, then proposed algorithms could be worthy to be tested in real environments.

Figure 7 shows four frames, from left to right and top to bottom, of one of the performed tests, in which the robot has to get close to the simulated person. The two areas marked with orange ellipses in the first two frames clearly show the concept of "memory" in laser projections: In the first frame, the robot, that was firstly looking to the panel, turns 180 degrees and navigates to the opposite section of the test area, as it finds no obstacles in its path. The robot has no direct line of sight towards the goal (i.e. the person) while executing this motion, thus its sub-objective is not modified. The third frame shows that, once the robot has to set a new sub-goal, it selects the one closer to the current goal, thus it turns left and not right when reaching the wall. Then, the robot finally sets a free path towards its goal (third frame) and it finally reaches it (fourth frame).

Figure 7 also shows the fused sensor data computed for each frame. Three synthetic RGDB sensors have been employed in the simulator, to conform a 360 degree emulated laser reading. The colour image obtained for these three sensors is also depicted at the top of each frame (left-center-right sensors). Figure 7 shows that the obstacles perceived for each sensor as well as the "laser memory" are adequately represented in the laser reading.

### B. Tests in real scenarios

Figure 3 show the robot that was used in this study. The robot is named "Gualzru" and navigates using a differential base. This mobile base is made of aluminium, with dimensions of 50 x 50 x 22 cm. It has a differential drive configuration with two front wheels and two freely rotating wheels to stabilize the robot. There are four columns on the base, and over them a platform with two RGBD sensors and a pan-tilt head with a third RGBD sensor. The data collected by these three sensors have been fused in an emulated laser using the proposed method. Gualzru has two laptop on board. One of them captures data from two RGBD sensors, and the other one captures data from the last RGBD sensor, and controls the base. The robot has been completely designed and built by RoboLab research group[3].

The sequence of images in Figure 8 shows how the robot avoids a person located in its direct path towards the goal. The robot describes a gentle curve towards its target to avoid colliding with the person.

Fig. 9 shows a situation in which a person repeatedly moves into the path of the robot, blocking its trajectory towards the target. The robot reacts to this dynamic obstacle by changing current navigation sub-goals. These reactions make the robot exhibit a 'polite' behaviour, in which it avoids colliding with the person while trying to reach the target.

The modifications to $\Re$-ORM algorithm, detailed in Section IV, produce smoother motion and reduce oscillations. However, even after including these updates, the robot exhibited sometimes an odd nodding behaviour. The evaluation of these situations led to the adjustment of the execution period for the navigation algorithm. While fast periods seemed the optimal solution, they produce slight oscillations when the robot executes rectilinear motion. These oscillations are caused by repeated contradictory commands (e.g. turn left - turn right cycles). The use of slower execution periods helps reducing them.

Finally, it is important to highlight the differences that exist between a conventional laser and the emulated one employed in these tests. The $\Re$-ORM algorithm has always been used with a conventional laser, which usually has a range from four to thirty meters. On the other hand, the emulated laser,

[3]http://robolab.unex.es

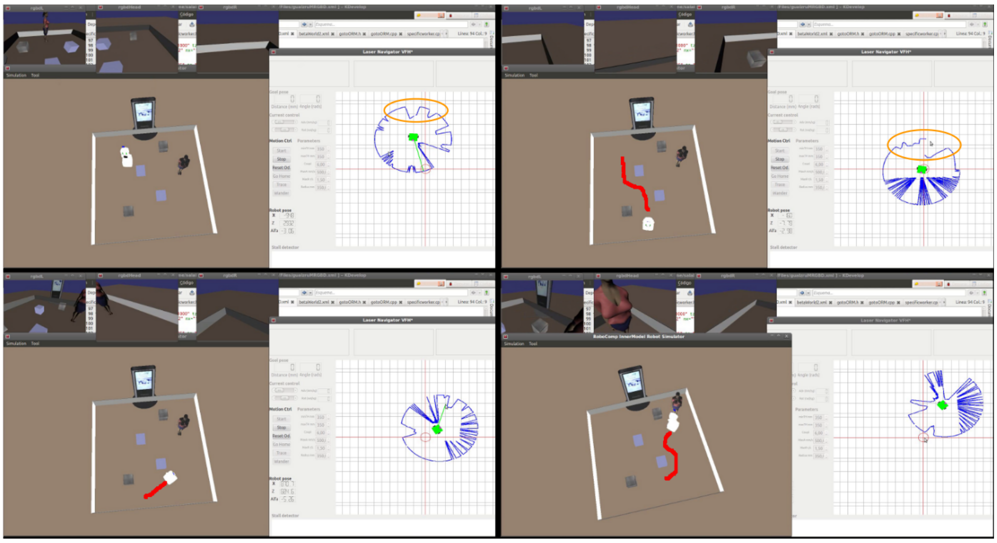

Fig. 7. Navigation results in a complex scenario.

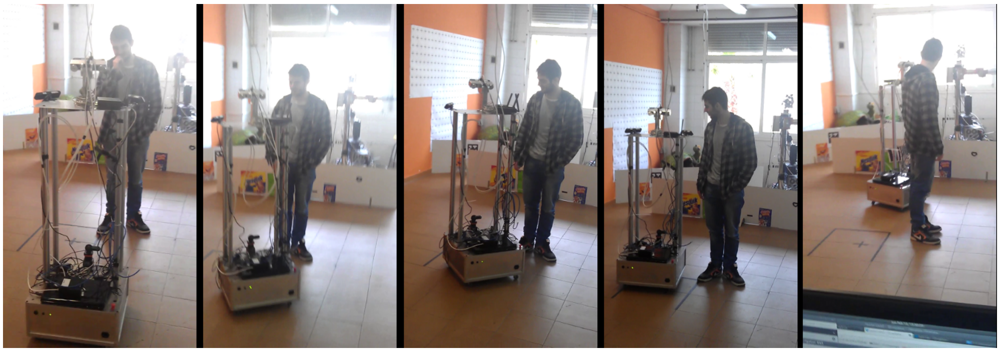

Fig. 8. The robot moves towards its goal while avoiding a standing person.

created from RGBD data, has a short maximum range, of about two meters. This forces the ℜ-ORM algorithm to put sub-targets near the current position of the robot. Despite this, the robot describes smooth paths towards these objectives, using gentle curves. The use of these near sub-targets does not interfere with the ability to avoid obstacles nor to reach desired targets.

## VI. Conclusions

Experimental results show that the proposed sensor fusion system is able to be used by a social robot working in daily life environments. It is robust and accurate enough for social interactions. The system successfully merges data obtained by a set of sensors, building an emulated laser reading from them. It is easily scalable, and can fuse data obtained by different types of sensors. The particular implementation described in this paper fuses data provided by three RGBD cameras. These data allow detecting obstacles at different heights while using algorithms designed to work with 2D laser scans.

On the other hand, two different reactive navigation algorithm have been tested in this

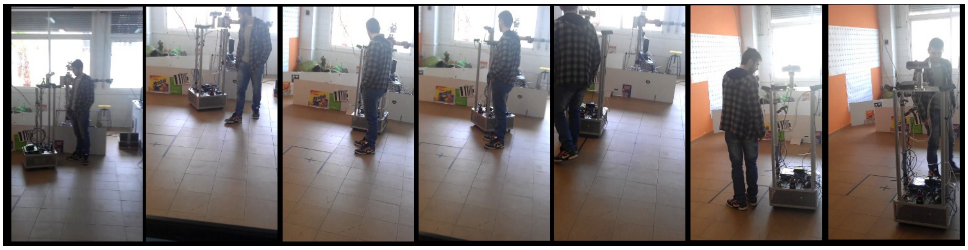

Fig. 9. The robot moves towards its goal while avoiding a moving person.

paper. Both of them offer good results, being $\Re$-ORM slightly better than VFH* for the daily life scenarios employed in the experiments.

The main issue of the current proposal is the necessity of a manual sensor calibration. Projected point clouds are useful to help aligning different sensor readings. However, the calibration process is tedious when three RGBD devices are involved.

Automatic calibration is a key characteristic to be developed for the proposed system. Future work will focus on matching point clouds (e.g. using ICP-like algorithms) to correctly estimate sensor poses. The possibility of including motors to automatically adjust sensor positions will also be evaluated.

## ACKNOWLEDGMENT

This work has been partially supported by the Spanish Ministerio de Economía y Competitividad, Project TIN2012-38079-C03-(01-02-03) and by the Extremadura Government Project GR10144.

## REFERENCES

[1] J. R. Asensio, J. M M Montiel, and L. Montano. Goal directed reactive robot navigation with relocation using laser and vision. In *Robotics and Automation, 1999. Proceedings. 1999 IEEE International Conference on*, volume 4, pages 2905–2910 vol.4, 1999.

[2] J. Borenstein and Y. Koren. The vector field histogram-fast obstacle avoidance for mobile robots. *Robotics and Automation, IEEE Transactions on*, 7(3):278–288, Jun 1991.

[3] David Chamorro and Ricardo Vázquez-Martín. R-orm: relajación en el método de evitar colisiones basado en restricciones. In *X Workshop de Agentes Físicos, Cáceres, España*, 2009.

[4] D. Fox, W. Burgard, and S. Thrun. The dynamic window approach to collision avoidance. *Robotics Automation Magazine, IEEE*, 4(1):23–33, Mar 1997.

[5] Marco Antonio Gutierrez Giraldo. Progress in robocomp. *Journal of Physical Agents*, 7(1):38–47, 2013.

[6] O Khatib. Real-time obstacle avoidance for manipulators and mobile robots. *Int. J. Rob. Res.*, 5(1):90–98, April 1986.

[7] J. Larsson, M. Broxvall, and A. Saffiotti. Laser based intersection detection for reactive navigation in an underground mine. In *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*, pages 2222–2227, Sept 2008.

[8] L. J. Manso, P. Bachiller, P. Bustos, P. Nez, R. Cintas, and L. Calderita. RoboComp: a Tool-based Robotics Framework. In *Int. Conf. on Simulation, Modeling and Programming for Autonomous Robots (SIMPAR)*, pages 251–262, 2010.

[9] J. Minguez. The obstacle-restriction method for robot obstacle avoidance in difficult environments. In *Intelligent Robots and Systems, 2005. (IROS 2005). 2005 IEEE/RSJ International Conference on*, pages 2284–2290, Aug

2005.

[10] J. Minguez and L. Montano. Nearness diagram (nd) navigation: collision avoidance in troublesome scenarios. *Robotics and Automation, IEEE Transactions on*, 20(1):45–59, Feb 2004.

[11] M. Mori, K.F. MacDorman, and N. Kageki. The uncanny valley [from the field]. *Robotics Automation Magazine, IEEE*, 19(2):98–100, June 2012.

[12] I. Ulrich and J. Borenstein. Vfh+: reliable obstacle avoidance for fast mobile robots. In *Robotics and Automation, 1998. Proceedings. 1998 IEEE International Conference on*, volume 2, pages 1572–1577 vol.2, May 1998.

[13] I. Ulrich and J. Borenstein. Vfh*: local obstacle avoidance with look-ahead verification. In *Robotics and Automation, 2000. Proceedings. ICRA '00. IEEE International Conference on*, volume 3, pages 2505–2511 vol.3, 2000.

[14] R. Vázquez-Martín. *onLine Environment Segmentation based on Spectral Mapping (LESS-Mapping)*. Ph.d. dissertation, ETSI de Telecomunicación, Málaga, Spain, October 2009.