An Open Source Robotic Platform for Ambient Assisted Living

Marco Carraro, Morris Antonello, Luca Tonin, and Emanuele Menegatti

Department of Information Engineering, University of Padova
Via Ognissanti 72, 35129, Padova, Italy
{marco.carraro,morris.antonello,luca.tonin,emg}@dei.unipd.it
http://robotics.dei.unipd.it

1 Introduction

Last years have seen a worldwide lengthening of life expectancy [1] and, as a consequence, an increment of advanced assistive solutions for integrated care models. In this context, ICT can enhance home assistance services for elderly people and thus, the economical burden for health-care institutions. Indeed, recent studies report that 89% wish to stay at home for sentimental reasons or because they cannot afford nursing homes [2]. In addition, the shortage of professional caregivers is a well-known issue and relatives or friends must face with emotional distress negatively impacting also their productivity at work [3]. In this perspective, home robots will play a crucial role. Not only they will keep their house safe by monitoring and detecting anomalies or sources of hazards but they can also be companions able to enhance their social life, e.g. by better connecting them with their relatives and friends. Examples of recent projects that have tried to develop such a kind of systems are Hobbit [4, 5], Astro [6, 7] and Giraffplus [8]. The difficulties in creating a robust and reliable prototype and the encountered challenges in computer vision and autonomous robotics have been well-explained in [5]. This work aims at presenting an open-source[1] and practical solution for an autonomous robotic platform for home care. The final goal is to develop a set of artificial intelligence services for indoor autonomous and safe navigation, fall detection, people recognition, speech interaction and telepresence. In Section 2 and 3 respectively, the prototype and the tasks currently implemented are described. We have developed these functions in *ROS: Robot Operating System* [9] which provides many algorithms and a standard communication framework. The need for standards and open solutions have been already pointed out in [10]. Future works are reported in Section 4: this project enables research in many fields like scene understanding, human robot interaction, socially assistive/intelligent robotics and sensor integration.

[1] https://bitbucket.org/iaslab-unipd/orobot

2 Hardware Configuration

Our prototype of home robot, shown in Fig. 1a, is built on top of a commercial open-source mobile platform, the Turtlebot 2[2], which is already equipped with odometry, a gyroscope, bumpers, cliff sensors, wheel drop sensors and a docking IR receiver. Furthermore, this platform is a smart choice also because of the available API to interface with ROS and the existence of a worldwide community working with it. For our purposes, we have added an Hokuyo URG-04LX-UG01 2D scanning laser rangefinder and a Microsoft Kinect v2. The former is placed at a height of 15 cm so as to easily detect low obstacles; the latter at a height of almost 115 cm which is the best trade-off for the people detection at difference distances. The robot control is accomplished by the Lenovo Y50-70 laptop with Ubuntu 14.04 and ROS Indigo.

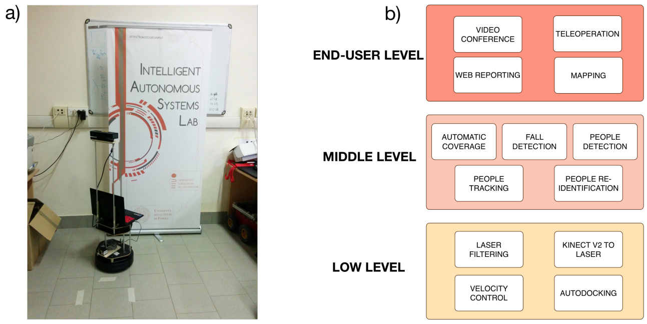

Fig. 1. a) An open source robotic platform for home care. b) The layered software infrastructure and its main components.

3 Software and Testing

The software of our robot is entirely based on ROS. The standard robot's task is the complete visit of the house while registering all events around it. Such a task has a low priority and can be preempted whenever any other web request like a teleoperation one or a video-conference call occurs. The robot has also the capability of autodocking, i.e. to autonomously return to the docking station for recharging. The software architecture is characterized by the layers in Fig. 1b. The end-user level, the highest, provides human-robot functionalities while the middle level implements navigation and people monitoring functionalities. At the lowest level, ROS nodes control motors, sensors and low-level behaviours like auto-docking. Algorithm parameters can be modified online thanks to the *dynamic reconfigure* offered by ROS. Each implemented functionality will be described in the following sections.

[2] http://www.turtlebot.com/

3.1 Laser Filtering

The raw data acquired with the laser scanner is subject to many inaccuracies due to the technology it is based on. There are issues with transparent and black surfaces producing frequent outliers. Nonetheless, the laser scanner is the core sensor for the navigation so the less inaccuracies occur, the better the navigation algorithm performs. For this reason, we have adopted an interpolation filter from the ROS packet *laser_filters*[3]. The laser streams data to the *raw_scan* ROS topic; the filter reads, processes and finally publishes the filtered data to the *scan* ROS topic. This one is subscribed by the robot functionalities. In Fig. 2a the comparison of the same map built with the *raw_scan* topic (left) and the filtered one (right) is shown. In the unfiltered map we can see how the outliers can result in extremely noisy borders. In contrast, the interpolated map shows significant improvements.

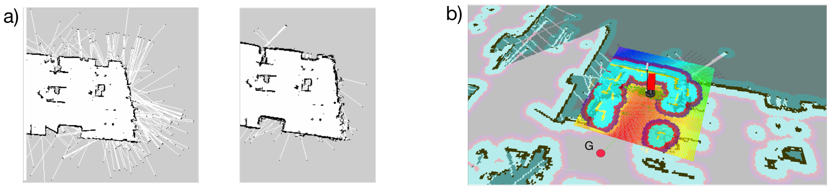

Fig. 2. a) Comparison of the map built with the raw data from the laser scanner (left) and the data filtered with an interpolation filter (right). b) View of the static and dynamic maps while the robot is navigating to the goal G.

3.2 Mapping and Navigation

Building a map is a core functionality in mobile robotics. In literature this problem is known as *Simultaneous Localization And Mapping (SLAM)* and there exist several implementations in ROS. For this purpose, we are exploiting the well-known technique described in [11] and implemented in the *GMapping*[4] ROS package. We have tested it successfully in several natural environments (e.g., office, corridors, homes). Once we have a static map representing the environment, a method for reaching a point maintaining the localization within the map during the robot movement is of concern. The technique used is based on *AMCL* [12] *(Adaptive Monte Carlo Localization)* ROS packet. This algorithm is based on two maps: the static one and the dynamic one which is computed on-line. For reaching a goal G (Fig. 2b), it computes a static plan from its initial position to the goal basing the decision on the static map. After this phase, the control of the movements of the robot passes to the dynamic map which is computed

[3] `http://wiki.ros.org/laser_filters`

[4] `http://wiki.ros.org/gmapping`

in real-time. With this map, AMCL algorithm checks whether the initial plan can be performed without collisions. In this case, the robot makes its moves maintaining the localization inside the static map with the *GMapping* package. Otherwise, the plan is modified to avoid it. If there are no possible new plans, the robot tries to rotate around itself to see if other paths are available given the dynamic map, otherwise it fails to reach the goal with an error which can be caught and managed. For instance, in the coverage algorithm (see Section 3.3), there is no guarantee that the various computed goals will be reachable without collisions because they are computed only on the static map. For each unreachable goal, the coverage algorithm catches the error and the robot is sent to the next goal in the coverage array, or to the docking station if it is the last one.

3.3 Coverage

Finding a map coverage (i.e. to visit all the map) is a sensitive problem for different kind of robots such as automatic lawn-mower, cleaning robots and so on. For our purposes, we use the coverage for monitoring all significant events happening in the house. The most used algorithm in literature is the one in [13]. This algorithm decomposes the map in square cells, it computes its minimum spanning tree and makes the robot circumnavigate it. This approach is perfect from a theoretical point of view, but in our case the robot does not know exactly where the obstacles are until it is close to them. Our artificial intelligence method is based on a simpler approach and ensures a good coverage which gives an open view of the scene to the Kinect v2. The algorithm consists of three steps:

- cell decomposition of the map: a grid is built on the map;
- marking phase: only the empty cells are kept in the matrix;
- zig-zag fashion array filling: we divide the empty cells in rows and we fill an array with a visit in a zig-zag fashion of them. In this way, the cells of each row are all in the coverage path, but the robot makes a smart visit of them, by minimizing the space to be traversed.

4 Conclusions and Future Works

In this work we have developed an open-source, autonomous robot for elder care assistance. Currently, the robot can navigate autonomously with both static and dynamic obstacle avoidance and has been tested in different environments like a laboratory, a home, an office and many classrooms in our department. Furthermore, it can also perform additional tasks such as auto-docking and automatic coverage of known maps. In future, our research will be focused on the human-robot interaction to implement on-board people detection, tracking and re-identification modules via vision algorithms based on Kinect v2. Afterwards, our work will be more end-user oriented developing the web reporting application, the video-conference system and, in general, all the web-based features of the robot.

Acknowledgments. This work is supported by Omitech Srl

References

1. United Nations. Department of Economic, World population ageing, 1950-2050. No. 207, New York: United Nations, 2002.
2. L. Jeannotte, M. J. Moore, et al., The State of aging and health in America 2007. Merck Company Foundation, 2007.
3. P. Rashidi and A. Mihailidis, "A survey on ambient-assisted living tools for older adults," Biomedical and Health Informatics, IEEE Journal of, vol. 17, no. 3, pp. 579–590, 2013.
4. D. Fischinger, P. Einramhof, W. Wohlkinger, K. Papoutsakis, P. Mayer, P. Panek, T. Koertner, S. Hofmann, A. Argyros, M. Vincze, et al., "Hobbit-the mutual care robot," in Workshop on Assistance and Service Robotics in a Human Environment Workshop in conjunction with IEEE/RSJ International Conference on Intelligent Robots and Systems, vol. 2013, 2013.
5. D. Fischinger, P. Einramhof, K. Papoutsakis, W. Wohlkinger, P. Mayer, P. Panek, S. Hofmann, T. Koertner, A. Weiss, A. Argyros, et al., "Hobbit, a care robot supporting independent living at home: First prototype and lessons learned," Robotics and Autonomous Systems, 2014.
6. F. Cavallo, M. Aquilano, M. Bonaccorsi, I. Mannari, M. Carrozza, and P. Dario, "Multidisciplinary approach for developing a new robotic system for domiciliary assistance to elderly people," in Engineering in Medicine and Biology Society, EMBC, 2011 Annual International Conference of the IEEE, pp. 5327–5330, IEEE, 2011.
7. F. Cavallo, M. Aquilano, M. Bonaccorsi, R. Limosani, A. Manzi, M. Carrozza, and P. Dario, "On the design, development and experimentation of the astro assistive robot integrated in smart environments," in Robotics and Automation (ICRA), 2013 IEEE International Conference on, pp. 4310–4315, IEEE, 2013.
8. S. Coradeschi, A. Cesta, G. Cortellessa, L. Coraci, J. Gonzalez, L. Karlsson, F. Furfari, A. Loutfi, A. Orlandini, F. Palumbo, et al., "Giraffplus: Combining social interaction and long term monitoring for promoting independent living," in Human System Interaction (HSI), 2013 The 6th International Conference on, pp. 578–585, IEEE, 2013.
9. M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "Ros: an open-source robot operating system," in ICRA workshop on open source software, vol. 3, p. 5, 2009.
10. M. Memon, S. R. Wagner, C. F. Pedersen, F. H. A. Beevi, and F. O. Hansen, "Ambient assisted living healthcare frameworks, platforms, standards, and quality attributes," Sensors, vol. 14, no. 3, pp. 4312–4341, 2014.
11. G. Grisetti, C. Stachniss, and W. Burgard, "Improving grid-based slam with rao-blackwellized particle filters by adaptive proposals and selective resampling," in Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on, pp. 2432–2437, IEEE, 2005.
12. D. Fox, W. Burgard, F. Dellaert, and S. Thrun, "Monte carlo localization: Efficient position estimation for mobile robots," AAAI/IAAI, vol. 1999, pp. 343–349, 1999.
13. Y. Gabriely and E. Rimon, "Spanning-tree based coverage of continuous areas by a mobile robot," Annals of Mathematics and Artificial Intelligence, vol. 31, no. 1-4, pp. 77–98, 2001.