Environment Mapping with Mobile Robot Guided by a Markers Vision System

Juan Pablo Ángel-López and Santiago Murillo-Rendón(✉)

Universidad Autónoma de Manizales, Manizales, Colombia
jangel@autonoma.edu.co, smurillo@autonoma.edu.co

Abstract. A Markers vision system is a set of infrared cameras capable to distinguish little spheres made of reflective material. These systems usually have been employed to study bio-mechanical processes, and as a tool for animating characters in virtual worlds, for instance in animated movies and video games. In this work, this tool is used as a positioning system for a mobile robot in a closed room. The robot is a prototype made over Arduino®, it consist of a two motor control provided with a proximity sensor for monitoring the space around. The control of this robot was designed to be automatic, in other words it is conditioned to walk in a delimited space and to report the objects found in its way. To control the mobile robot, an algorithmic framework was built in Matlab®. The idea with this framework is to receive the data obtained by the motion capture system to locate the robot in the delimited area. Otherwise, the framework is used to build the virtual map of the place and to produce the control orders to guide the robot. The aim of this work is to evaluate a set of strategies for environment mapping and the performance of the robot in the identification of the hidden space and its reconstruction in a virtual representation. In the other hand, this work is used to teach techniques of artificial vision, algorithm writing, digital signal processing and control to undergraduate students. Furthermore, this project is proposed to identify new lines in mobile robots research and in the development of automatic machines that works in cooperative schemes.

Keywords: Robotics · Motion capture · Environment mapping · Spatial coordinates

1 Introduction

Mobile robots (MR) are systems designed to work in an autonomus way, it is, a MR itself has a set of systems that provide it with energy, locomotive properties, environment recognition, positioning and specific tools to do the task for which it was designed. This kind of robots have been developed to resolve a lot of tasks as: disperse fertilizer [1], collecting plantation crops [2], locate and distribute products in warehouses [3], among many others applications. In the first example, the robot has to rove the plantations providing fertilizer, in its way the robot must avoid the obstacles to optimize its work [1]. The second example corresponds with a harvester robot, this MR is capable to identify apples and collect them, it is accomplished by means of a system of artificial vision (AV), the robot captures an image of the place and segments the apples' shapes, then, an algorithm locates each apple in a matrix. The MR is guided by this matrix to the apples' position and a robotic manipulator harvest it. Other example is the Amazon® robots, used by this enterprise to trace the customer orders. The Amazon® warehouses are distinguished for their big dimensions, so the MRs use have optimized the tracing procedure and have reduced the operation cost over 20% [3]. The positioning ability in mobile robots is provided by different systems as: gps, proximity sensors, stereoscopic vision cameras, motion capture technology and others. Most of these are based in AV [4]. The AV consist in programming an electronic system to recognize a scene and its elements. In mobile systems can also allow the robot to identify and navigate around its environment. In general an AV system consist of five stages: the data acquisition, preprocessing, segmentation, feature extraction and classification, but depending of the complexity of the application some of these stages can't be included or modified [5]. An example of positioning is showed in [6]. Here the target is to place a disk in a hole. Given that the disk position related to the hole is unknown, it is necessary to use an illumination set to project points to the disk surface, then the lights are detected by the control cameras and the disk location is estimated. The algorithm makes a triangulation process to determine the disk disposition. Other application of MR is the exploration of hostile places like the ruins after an earthquake, places with chemical contamination or in the extraterrestrial exploration in the same way as Curiosity in Mars [7]. The previous scenario is possible to achieve in most cases thanks to the incorporation of environment mapping and route planning systems and strategies. First, the environment mapping consist in building a surroundings virtual representation [8], this representation enables the robot to identify obstacles for its navigation. In [8] the desing of a 3D sensing system for a Unmanned Ground Vehicle (UGV) is described. The autors reviewed different technologies for mapping, including Ultrasonic based systems [9], RADAR (Radio Detecting And Ranging) systems [10], systems based on stereoscopic cameras [7] and LIDAR systems (Light Detection And Ranging). After this review, the authors described the UGV designed, and the technology used for mapping is a LIDAR System. The work showed by [11] uses a stereoscopic vision system to obtain a 3D matrix, the dimension of this matrix is $264 \times 264 \times 64$ in order to build the representation of a space with 6 by 6 by 2 m. The crayfish algorithm allows one to define camera 3D rays, through this rays it is possible to found displacements that corresponds with object location. This work allows the representation of the environment with full definition and it is useful for high perception robots. In [12] the authors present the development of a MR under the Open Unified Process methodology. The robot works with a stereoscopic vision system to identify the place around, whit this the robot identifies obstacles and determines routes from 2D images. Other strategies to route planing are the use: of potential fields, Brownian particles and Dijkstra algorithm for multi-target routes. In general, the route planning strategy depends of the localization and mapping system. In this work, a MR developed over an Arduino® board is showed. This robot uses a localization system based on motion caption and is provided with an infrared distance sensor which permits the environment mapping. The control system is centralized in a PC, where the algorithms to calculate orientation of the robot are implemented in Matlab®. The Matlab® implementation also supports the algorithms for calculating the best route obtained after the surroundings mapping, and the virtual map is built too. The communication protocol was implemented with WiFi technology and the robot mechanical behaviour corresponds to a differential architecture. Forward, the article is organized in this way: First, in Sect. 2 the equipments and methods are presented, then in Sect. 3 the results are showed, finally in Sect. 4 there will be the conclusions of this research.

2 Equipment and Methods

The main goal of this research is to develop an environment mapping by tracking the position and orientation of a mobile robot inside a controlled area. For the detection of position and orientation of the robot a (Motion Capture) MoCap System is used, which broadcasts the current status of the vehicle to a main control script written under the Matlab® Environment. The mobile robot is communicated with the main PC via WiFi in order to constantly send and receive data.

2.1 MoCap System

This research takes place thanks to a MoCap System composed by twelve infrared Flex 3 cameras and the software Motive©, both manufactured by OptiTrack™. The cameras are placed on a rectangular frame of 3.8 by 7, 3.5 m above the ground. Figure 1 shows the laboratory setup, where it is possible to appreciate the rectangular frame and the capture area where the cameras are aiming to. This capture area is 2 by 3 m approx (Fig. 2).

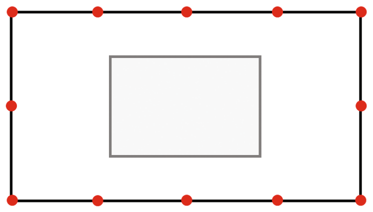


Fig. 1. The external *filled circles* represent the cameras and the *inner rectangular* area represents the capture volume

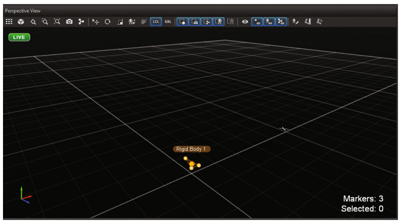


Fig. 2. Motive© live capture interface and rigid body creation

The cameras used for this study have a capture rate of 100 *FPS*, a video resolution of 640 × 480 and a maximum range of detection of 11 m [13]. On the other hand, the software which the cameras where connected to, gathers the information sent by the cameras to create a 3D data set. Motive© builds a tridimensional matrix that contains the *xyz* coordinates from each one of the markers placed inside the capture volume [13]. In order to track the robot in “Real Time”, Motive© broadcasts the 3D data of each rigid body and the markers attached to it, to an interface named *NatNet* [14] written in Matlab®, provided by OptiTrack™. The original *NatNet* script was modified so it could send to another PC the raw data needed to calculate position and orientation of the vehicle. For the detection of the vehicle, three markers are placed on its top, forming an isosceles triangle aiming to the front of the robot. The reason of this setup is that the MoCap software needs at least three markers for crating a rigid body, and a rigid body is needed for broadcasting data via *NatNet*. As a result of the creation of the rigid body, Motive© creates a virtual marker placed at the geometrical center of the object, which is also transmitted by the *NatNet* interface.

2.2 Mobile Robot

The robot used in this research has a differential architecture (Fig. 3), which means that its movement is generated by two independent wheels placed in the rear of the car, each one attached to a DC motor, and a free wheel in the front. The vehicle is controlled wirelessly by an ESP8266 WiFi module attached to an Arduino® board, from a PC using Matlab®. The Arduino® script receives orders from the PC like going forward, going backwards, rotating clockwise (CW), rotating counter-clockwise (CCW) or stopping, which results in the control of the direction of rotation of the actuated wheels. The vehicle is also equipped with an infrared sensor placed at its front, in order to detect and avoid obstacles ahead of it. In the case and obstacle is detected, the robot stops and sends data about the distance between itself and the obstacle to the PC so the main algorithm could be able to build a mapping image of the environment.

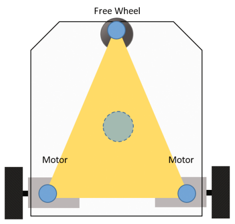


Fig. 3. The schematic representation of architecture used for the mobile robot and the location of the markers

2.3 Robot Tracking and Environment Mapping

The origin $(0, 0)$ of the controlled area's coordinate system is placed at the exact geometrical center of that area so all the coordinates and vectors are measured relative to that point. The *NatNet* interface sends the 3D coordinates of the geometrical center and the front marker of the robot, relative to coordinate system mentioned before, to the main Matlab® script and then, with that information, an algorithm calculates the position and orientation of the mobile robot. The main task of the mobile robot consist in defining a target xy coordinate to achieve, enclosed in a certain area. When the destination is determined, two vectors are created in order to determine the orientation of the vehicle relative to its target. The first action is to align the robot with the desired destination by rotating clockwise or counter-clockwise and the next action is to achieve the target by moving straight forward. Here, the problem is to determine what kind of rotation (CW or CCW) must be done by the vehicle. To do so, a coordinate system transformation is needed. The coordinates of the target are transformed to a local coordinate system centered at the geometrical center of the robot, whose $X\ axis$ is the vector created between the center of the car and the front marker (Fig. 4). If the target is placed in the first or second quadrant of the local coordinate system, the robot must rotate CCW, and in the other case, the robot must rotate CW. To determine the quadrant where the target is placed, it is only necessary to calculate the y (vertical) coordinate according to the car, as shown in Eq. 1, where β is the angle between the vehicle's system and the reference system, θ is the angle between the vehicle and its destination and r is

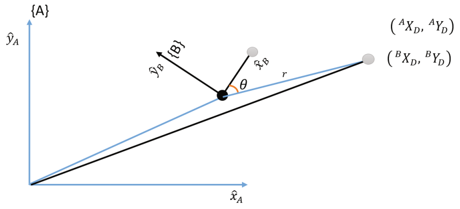


Fig. 4. Local coordinate system determination for the robot rotation

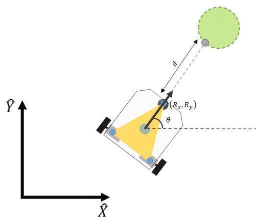


Fig. 5. Environment mapping representation

the distance between the center of the car and the target.

$$^{B}Y_D = r\sin\theta\cos\beta - r\cos\theta\sin\beta \tag{1}$$

When the robot finds an obstacle on its path, it stops by a certain distance before the object. The obstacle and that distance are detected and measured by an infrared distance sensor placed under the front marker, so the detected object is aligned with the direction vector of the vehicle. The main tracking and control PC is able to determine if the robot has stopped, in which case it is asked to the vehicle for the measured distance. The xy coordinate of the obstacle is calculated with the measure given by the robot and its position and orientation, detected with the MoCap system (Fig. 5). A monochrome image is built for representing the controlled area and the objects detected inside it, where every black pixel corresponds to an obstacle or a point from the frontier of the navigation area, and its resolution equivalent is one pixel per squared centimeter. Every time the robot sends a distance, a coordinate is calculated and then its corresponding pixel of the image is filled black.

3 Results

As a result of this research, a differential mobile robot was built and two Matlab® Graphical User's Interfaces (GUIs) were created in order to control the robot via WiFi connection (TCP/UDP). One GUI performs a manual control by push buttons and keyboard entries (Fig. 6) and the other GUI performs an automated control by giving a desired destination xy coordinate, while making a live tracking of the vehicle via *NatNet* (Fig. 7).

The manual control GUI has five different movement actions: straight forward, straight backwards, rotate CW, rotate CCW and to stop. There are also manual increasing and decreasing speed buttons for both motors independently. All these previous actions were also mapped with the keyboard, following this equivalence:

- W: Straight forward
- S: Straight backwards
- A: Rotate CCW
- D: Rotate CW

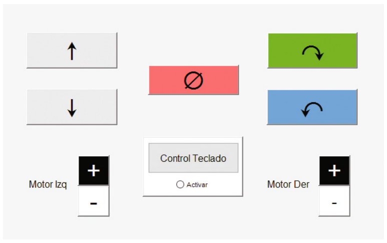
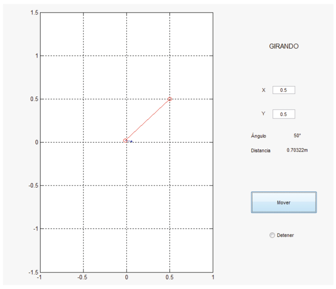


Fig. 6. Manual control GUI



GIRANDO

X 0.5

Y 0.5

Ángulo 50°

Distancia 0.70322m

Detener

Fig. 7. Live tracking control GUI

- Q: Stop
- I: Increase Left Motor speed
- K: Decrease Left Motor speed
- O: Increase Right Motor speed
- L: Decrease Right Motor speed

The live tracking control GUI asks the user to enter the target xy coordinates for the vehicle to achieve, if the user gives a point outside the defined area, the GUI will ask for another pair of coordinates. The mobile robot's movement control is a simple iterative proportional approach with hysteresis. When the vehicle is not aligned with its target, the control algorithm decides the direction of rotation (CW or CCW) in order to place the robot aiming to the destination and then proceeds to move straight forward. This process is not made in this two basic steps given that the robot is not able to move exactly in a straight line. The vehicle has to recalculate and to correct its trajectory several times before arriving to the destination point. According to the environment mapping, the vehicle stops at the presence of an obstacle placed in front of it. The main PC detects this state of the robot by calculating the standard deviation between the last positions of the car, and when this parameter is close to a minimum value, it will be considered that the robot has stopped. After this detection, it will be asked for the car to send via WiFi the value of the distance between itself and the detected object. With this information and with the position and orientation of the robot, the exact coordinate of the object is calculated and then,

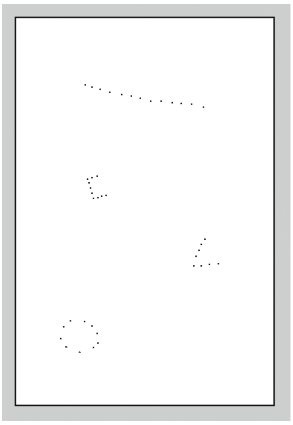


Fig. 8. Example of constructed map for several objects

this coordinate is filled in the monochrome map of the controlled environment (Fig. 8).

4 Conclusions

The main difficulty with the robot is to achieve a straight trajectory, since the inherent difference between both motors, in spite of having the same reference and manufacturing. To resolve it, various strategies were used: first, the evaluation of different chassis was useful to determine the most appropriate structure to accomplish the wheel proper alignment. Moreover, it was necessary to evaluate the used wheels, because the MR traction depends on the materials and dimensions. Other guideline was the use of gearboxes, facilitating the speed control. Nevertheless the gearboxes increase the current intake. The most remarkable improvement was the implementation of opto-interrupter speed controllers. This system senses each wheel spin, then by means of interruptions in the Arduino® board, it is possible to synchronize the motor speed. Even, after the described improvements, the straight movement it is still not too accurate. Taken into account the idea to build a MR with the lower cost, other relevant aspect was the communication system. In that sense, it was evaluated the option of using HC-05 and HC-06 bluetooth modules, broadly used in embedded systems. The lower transmission range and its unstable connections contributed with the exploration of other modules. The use of the ESP8266 module proved a better performance, and for this reason the communication protocol was implemented over UDP and TCP/IP standards. The current communication protocol sends to de Arduino® board a string composed by only the word "dat" followed by a three digits number. This protocol has proven to be useful but also to be so simple. It is necessary to improve this method by adding, for example, another combination of letters at the end of the sent word in order to assure a correct received string. This strategy can also be improved by ordering the board to write a certain speed (by a PWM signal) to the motors and not only to increase or to decrease such value by a predefined increment/decrement on the Arduino® algorithm. Given that the vehicle requires for a user to give it a destination coordinate so it can detect and map several obstacles, the need to implement an autonomous trajectory algorithm to navigate the environment is evident, in order to achieve a complete mapping of the obstacles placed inside the controlled area. The mobile robot's technology is as simple as it can be (not the markers system), so its performance is not perfect. The automatic positioning approach has a success rate of approximately 60% because of the poor mechanical performance of the vehicle; it is necessary to improve the motors control and speed. Another issue with this system is the required time for the robot to get to its destination. The selected control strategy requires several iterations for the robot to achieve its goal; sometimes it takes almost 5 min to travel a distance of 2 m when the robot is not properly aligned with its destination, so this could be accomplished by programming a PID (or a more complex) control strategy.

Acknowledgements. This work was granted by "Research committee, Universidad Autónoma de Manizales", under the researh project "Desarrollo de robots móviles programables controlados por tecnología de captura de movimiento".

References

1. Jiménez, F., Moreno, J., González, R., Díaz, F.R., Sánchez-Hermosilla, J.: Sistema de visión de apoyo a la navegación de un robot movil en invernaderos (2005)
2. Ji, W., Zhao, D., Cheng, F., Xu, B., Zhang, Y., Wang, J.: Automatic recognition vision system guided for apple harvesting robot. Comput. Electrical Eng. **38**(5), 1186–1195 (2012). Special issue on Recent Advances in Security and Privacy in Distributed Communications and Image processing
3. Amazon: Amazon Robottics (2016)
4. Bertozzi, M., Broggi, A., Cellario, M., Fascioli, A., Lombardi, P., Porta, M.: Artificial vision in road vehicles. Proc. IEEE **90**, 1258–1271 (2002)
5. Malpartida, E.: Sistema de Visión Artificial Para el Reconocimiento y Manipulación de Objetos Utilizando un Brazo Robot. Pontificia universidad católica del Perú, Perú (2003)
6. González Galván, E.J., Ramírez, C., Rolando, S., Durán García, H.M.: Aplicación de sensores múltiples para el posicionamiento tridimensional de robots usando visión. Interciencia, **26**(11), 541–546 (2001)
7. NASA: Seventeen Cameras on Curiosity (2016)
8. Rejas, J.-I., Sanchez, A., de Rivera, G.G., Prieto, M., Garrido, J.: Environment mapping using a 3D laser scanner for unmanned ground vehicles. Microprocess. Microsyst. **39**(8), 939–949 (2015)
9. Elvira, S., de Castro, A., Garrido, J.: Alo: an ultrasound system for localization and orientation based on angles. Microelectron. J. **44**(10), 959 – 967 (2013). Conference on Design of Circuits and Integrated System 2011
10. Cory, P., Everett, H.R., Heath-Pastore, T.A.: Radar-Based Intruder Detection for a Robotic Security System (1999)

11. Moravec, H.: Robot spatial perceptionby stereoscopic vision and 3D evidence grids. Perception, Sept 1996
12. Lancha, M.Á.F., Sanz, D.F., Plasencia, C.V., Ortíz, J.J.R.: Planificación de trayectorias para un robot móvil (2010)
13. Point, N.: Optitrack (2016)
14. Point, N.: Natnet (2016)