# Docking autonomous robots in passive docks with Infrared sensors and QR codes

Roberto Quilez*, Adriaan Zeeman†, Nathalie Mitton*, Julien Vandaele*
* Inria Lille-Nord Europe, France. e-mail: firstname.lastname@inria.fr
† Stellenbosch University, South Africa

***Abstract*—In this paper, we describe an inexpensive and easy to deploy docking solution in passive charging docks for autonomous mobile robots. The objective is to achieve long-term autonomous robots within an experiment test-bed. We propose to combine the use of QR codes as landmarks and Infrared distance sensors. The relative size of the lateral edges of the visual pattern is used to position the robot in relation with the dock. Infrared distance sensors are then used to perform different approaching strategies depending on the distance. Experiments show that the proposed solution is fully operational and robust. Not to rely exclusively on visual pattern recognition avoids potential errors induced by camera calibration. Additionally, as a positive side effect, the use of Infrared sensors allows the robot to avoid obstacles while docking. The finality of such an approach is to integrate these robots into the FIT IoT Lab experimental testbed which allows any experimenter to book wireless resources such as wireless sensors remotely and to test their own code. Wifibots holding wireless sensors will be integrated as additional reservable resources of the platform to enlarge the set of possible experimentations with mobile entities.**

## I. Introduction

Experimental validations are particularly important in wireless sensor networks and multi-robot systems research. The differences between models and real-world conditions are amplified because of the large number of nodes and robots, interactions between them, and the effects of mobility, asynchronous and distributed control, sensing, and actuation. As part of the FIT [1] federation, the FIT IoT-LAB [2] testbed is an evolution of SensLAB [3] platform, enhancing a very large scale open wireless sensor network platforms with new material architectures and new mobile nodes based on autonomous robots. Two types of robots (with wireless sensor nodes embedded) will be provided by the FIT IoT-LAB platform to the user to control the mobile patterns in their experiments: Turtlebots 2 (with a Kobuki base [4]) and Wifibots [5]. Autonomous mobile robots are constrained in their long-term functionality due to a limited on-board power supply.Recharging the batteries of a mobile robot in an autonomous way is a difficult and disruptive event. The homing action is commonly only achievable when the robot is within line of sight of a recharge station's beacon. The robot can either move towards the beacon using its navigation system, or wonder around within a described region while searching for the beacon signal. A mechanism is finally needed to secure the robot to the charger. Automated docking with Kobuki is already implemented [6]. The objective of this work is to enable Wifibot robots to achieve long-term autonomy and provide a similar functionality as for Kobuki.

This paper presents a simple and inexpensive docking method utilizing an on-board camera, infrared (IR) sensors and passive base stations. The method benefits from two key concepts: The first is image processing using a camera module for decoding QR codes and defining a four point square. This allows the robot to determine its position relative to the dock. The second concept is the use of infrared sensors to avoid obstacles and perform alignment with the dock. We show that this is wining combination that allows efficient docking.

The finality of such an approach is to integrate these robots into the FIT IoT Lab[1] experimental testbed which allows any experimenter to book wireless resources such as wireless sensor remotely and to test their own code. Wifibots holding wireless sensors will be integrated as additional reservable resources of the platform to enlarge the set of possible experimentations with mobile entities.

The paper is organized as follows. Section II explains the motivation and purpose of this work. Section III introduces some already existing robot docking solutions and works on visual patterns from which we inspire. Then we give an overview of our docking problem (Section IV). Section V describes the proposed solution. And finally we discuss some results and conclude with some considerations about related future work.

## II. Motivation and Objective

The main purpose of this work is to provide the FIT IoT-Lab [2] testbed with a set of autonomous robots. **FIT IoT-LAB** testbed offers a first class facility to evaluate and experiment very large scale wireless IoT applications from low level protocols to advanced services integrated with Internet, accelerating the deployment of advanced networking technologies for the future IoT. IoT-LAB provides wireless nodes and robots to test and improve the impact of IoT devices' mobility. FIT IoT-LAB's main and most important goal is to offer an accurate open access multi-user scientific tool to support the design, development, tuning, and experimentation of real complex large-scale sensor, IoT node and robot network[2] applications by any IoT developer. Robots to be provided in

This work has been partially supported by a grant from the CPER Nord Pas de Calais CIA and the EquipEx FIT project.

[1] http://iot-lab.info/
[2] https://www.iot-lab.info/hardware/

TRIDENTCOM 2015, June 24-25, Vancouver, Canada

DOI 10.4108/icst.tridentcom.2015.259673

the platforms are of two kinds: Wifibots (See sec. IV-1) and Kabuki TurtleBots[3].

The objective of this work is to allow Wifibot robots to achieve long-term autonomy by implementing automated docking procedure. The solution's functionality should be a similar to the Kobuki's robot.

## III. Related work

Methods used for docking and charging robots vary greatly in both effectiveness and application. Examples of methods used for robot docking include the use of radio signals, dead reckoning, ultrasonic beams, infrared beams coupled with radio signals, etc.

### A. Docking

The first autonomous mobile robots were created by Grey Walter in the late 1940's [7]. These robots followed a light to find their way into a hut with a battery charger. Systems available today are not that much different, using sensors to identify the recharging station and assisting with docking. They could also integrate localization and navigation capabilities to find the dock.

In [8], infrared sensors and a reflective type on the floor is used to accurately position the robot for connecting to a docking base. Landmarks can also be placed above the docking station [9]. The robot is able to visually align itself during docking thanks to the landmarks.

A more complex docking system is proposed in [10]. The robot approaches the docking station using a long infrared beacon and a sonar. Once the robot is in proximity of the station, a Sick Laser Measurement Systems (LMS) is used to align the robot to a target. The target consists in a grid with a pattern designed to distinguish it from the surrounding environment. An interesting docking system is also explained in [11], where a particular mechanism allows a high angular and displacement error during the docking process. A combination of vision and laser beacons are then deployed to perform the autonomous recharging.

Commercially available robots for household use (e.g. cleaning), with auto-docking capabilities, are becoming increasingly popular [12] [13] [14] [15]. Low-cost robotic platforms, such as the Turtlebot 2 (built upon the Kobuki robot base), are derived from these household robots, and become a popular research and educational tool. For these robots, cheap Infrared sensors are usually adopted for use with a homing system that allows the robot to dock at the docking station. They require the robot to be placed in an open space of at least 2 by 5 $m^2$ for automatic docking to run properly. There should be no obstacles between the robot and the docking station.

### B. Visual patterns

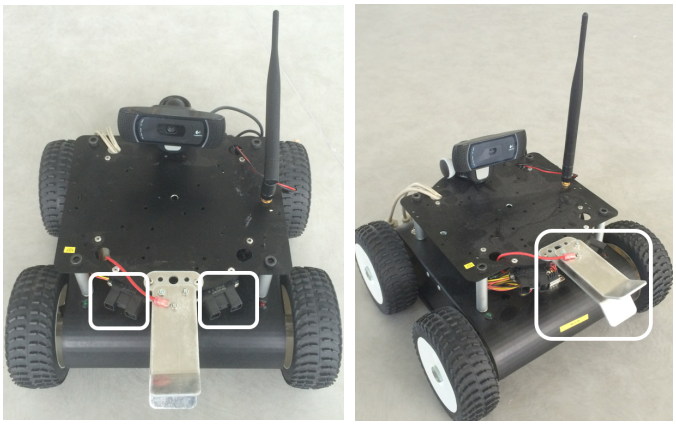
Advances in camera technology have dramatically reduced the cost of cameras making them the sensor of choice for robotics and automation. Visual data from a single sensor can provide a variety of cues about the environment (motion, color, shape, etc.). Visual pattern recognition algorithms solve some fundamental problems in computer vision: correspondence, pose estimation, and structure from motion. Therefore, visual pattern recognition can be considered an important, cost-efficient primitive for robotics and automation systems [16]. For navigational purposes, robots can recognize landmarks to acquire the angle, the XY coordinates and the information encoded in the QR code [17]. 2D bar-code landmarks could also enable self-localization of mobile robots. Some authors have proposed self-containedness, i.e. encoding everything needed for localization in the landmark. These landmarks can therefore be placed arbitrary in an environment, and used by the robot for navigation [18]. These kinds of landmarks have also been used in combination with other localization techniques to improve position estimation [19]. Other proposed QR code implementations have no direct relation between the bar code data and its position in an environment. [20]. However, relying entirely on camera calibration for position estimation is processor intensive and yields potential errors [18] besides the burden of the calibration task [20].

Referred as visual servo control, computer vision data can also be used in the servo loop to control the motion of a robot [21]. The aim of this vision-based control schemes is to minimize an error. This error is typically defined by the difference between a set of relevant parameters from image measurements (e.g. the image coordinates of interest points) and the desired values of the features. Nevertheless, visual serving schemes are local feedback control solutions. Vision feedback control loop come up against difficulties when the initial and the desired positions of the robot are distant [22]. They thus require the definition of intermediate sub-goals in the sensor space at the task planning level. In our configuration, the camera is mounted directly on a mobile robot. Motion of the robot induces camera motion. The trajectory to reach the docking station may require the robot to turn significantly. According to the degrees of freedom of the Wifibot movements, turning may increase the predefined error function even pushing the target out of the camera field. More complex position and trajectory estimation have to be implemented. We therefore propose the use of a combination of sensors (sensor fusion) for a robot to estimate its relative position in an environment. This is accomplished by combining a camera, for reading QR codes, and an IR sensor pair, for distance and angle estimation. The information encoded in the QR code is kept to a minimum. It only includes an identifier to inform the robot which dock it is currently seeing. This allows the QR code image to be smaller and also improves readability from longer distances. IR sensors are utilized for distance estimation, rather than relying exclusively on computer vision. This allows a more robust and easier to deploy solution.

## IV. Models and hypotheses

As mentioned, the main purpose of this work is to provide the FIT IoT-Lab [2] testbed with a set of autonomous wifibot robots. This section details the robot features and assumptions made in this work to better understand our implementation

[3]http://turtlebot.com



(a) IRs distance sensors (b) Dock connector

Fig. 1: Wifibot robot

TABLE I: Wifibot description

| | |
|---|---|
| Sensors | 4x Hall encoders 12ppr (336 tics/Wheel turn)<br>Battery level<br>Current Level<br>4 IR distance sensors (SHARP 2Y0A02) |
| Speed Control | 1 X DSPIC 33ep (4 x PID) |
| Motors | 4x Brushless motors 12V 16W<br>28:1<br>12Kg/cm<br>136 rpm |
| Dimensions | $32 \times 37 \times 15$ cm$^3$<br>Weight : 3.8Kg |
| Batteries | 12.8V LIFEPO4<br>10 AH<br>Charger included |
| Control Bus | RS232 / Ethernet |
| CPU | x86 SBC Intel Duo Core Atom 1.8Ghz |

choices. The main goal of this work is to allow Wifibot robots to achieve long-term autonomy by implementing automated docking procedure. The solution's functionality should be a similar to the Kobuki's robot.

*1) The Wifi robot:* Wifibot robots are convenient for those who are interested in an affordable open mobile platform for developing and learning robotics. The base system is described in Table I. The two front IR sensors are fixed with an angle as shown in figure 1a. They can accurately measure distances between 20 cm. and 150 cm. The web-cam is a Logitech HD Pro Webcam C920. The robot is customized with a front metallic bar with two contact points allowing the robot to autonomously plug into the docking station (figure 1b).

*2) The dock:* The dock consists on an 80 cm. metallic bar with two contact surfaces providing 18V and 8A, and is able to dock 2 robots next two each other.

### A. Objective and hypotheses

- The dock station must be supported by a wall or heavy object to prevent the robot from displacing it.
- The robot uses a navigation system (not discussed in the paper) that takes it to an area close to the docking station.
- Thereafter, it must be able to dock autonomously.

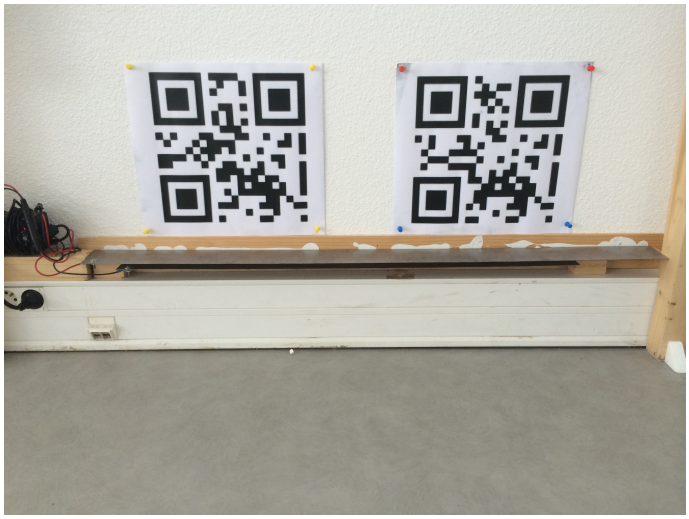


Fig. 2: The Wifibot's dock

Additionally, for an inexpensive, easy implementable and deployable solution, we name the following requirements:

- The docking base station should be preferably passive. Active beacons at the dock should be avoided, i.e., no active infrastructure other than the robot power supply should be required.
- No extra hardware (sensors) on the robot other than the on board camera module and IR sensor pair is desirable.
- The number of robots could be important. In order to optimize space, the solution should enable two robots docked close to each other.

## V. Implementation

Visual pattern recognition systems provide robust and reliable recognition of visual landmarks [23]. It allows for a variety of tasks such as object and target recognition and navigation. For our application, we will define a target (the dock station) and place a visual pattern above it with with a unique identifier. Choosing the proper type of landmark is important for image-based landmark recognition. While several landmark types are feasible, we choose to use QR (Quick Response) codes for the following reasons.

- They are more than an easily findable sign,landmarks of this kind are much easy for robots to detect and read.
- They are easy to detect and read.
- Cost effectiveness. Just a printed piece of paper.
- Ease of creation; it is easy to produce.
- They include error correction functionality.
- They can store a considerable amount of information.
- High-speed reading is possible for QR code from every direction.
- High availability of libraries to decode them.

However, the docking range for the solution is limited by the maximum distance the camera can detect and read the QR code. The recognition rate falls with an increase in distance between the QR code and the camera. The size of the QR code

is therefore determined from the camera's capabilities and the desired docking range.

### A. Infrastructure

We use a Wifibot Lab v4 robot with a web-cam and front IR distance sensors (See Fig. 1). The base system is described in Table I. The two front IR sensors are fixed with an angle as shown in Figure 1a. They can accurately measure distances between 20 cm. and 150 cm. The web-cam is a Logitech HD Pro Webcam C920. The robot is customized with a front metallic bar with two contact points allowing the robot to autonomously plug into the docking recharging station (Figure 1b). The robot is running ROS Hydro Medusa on the Ubuntu 12.04 LTS (Precise) release. There is an existing "roswifibot" package [24] in order to control the robot and get its status and sensors information.

The dock consists on an 80 cm metallic bar with two contact surfaces providing 18V and 8A, and is able to dock 2 robots next two each other (see figure 2) It is marked by a QR code above it. Actually, there are two QR codes, allowing two dock slots per 80cm bar (see figure 2). The QR code is 28 cm wide and high, to be visible from a distance of 4m. In order to be visible from 4 meters distance, the QR codes will be 28cm. edge side length. There are a number of existing libraries for reading data from QR codes. We use the ZBar [25] for reading the QR codes, which has been used for similar purposes [19] and seems to be highly reliable under normal lighting conditions, and is easy to implement. The library is capable of finding and decoding multiple QR codes in a single image. For each QR code, it returns a set of four points (XY pixel coordinates) representing each corner of the QR code square. This information is then processed to obtain the relative position of the robot in relation with the dock according to the optical distortion of the QR code (Fig: 3).

**input** : A dock identifier dockID
**output**: Docked robot

```
1  docked ← False;
2  turned ← 0^o ;
3  while not docked do
4      picture ← TakePicture();
5      QRs_list ← ScanQRcodes(picture);
6      QR_dock ← FindMatchingQR(QRs_list,
       dockID);
7      ir ← ReadIR();
8      IR_list ← ir;
9      if QR_dock was found then
10         QR_dock ← AlignRobotwithQR();
11         docked ← Approach(QR_dock);
12     else
13         if turned ≥ 360^o then
14             ObstacleDetected
               ←DetectObstacle(IR_list);
15             if ObstacleDetected then
16                 GetAwayFromObstacle(IR_list);
17                 Clear(IR_list);
18                 turned ← 0^o ;
19             else
20                 Return(Error);
21             end
22         else
23             Turn(30^o);
24             turned ← turned +30^o ;
25         end
26     end
27 end
```

**Algorithm 1:** Docking algorithm overview

### B. Algorithm

The sequence in which decisions are made in the algorithm is independent of the robot's location and distance from the docking station. Nevertheless, the approaching actions performed by the robot will differ depending on how far the robot is (from the dock or an obstacle).

The basic idea depicted in Algorithm 1 is that the robot turns around itself looking for a QR code. It takes a picture, decodes it, and checks whether there is a QR code with the desired information encoded. If the QR code is not found, it turns (around $30^o$), and tries again. Once the QR code is found, the robot will align with the QR code, i.e. it will turn little by little until the symbol is centered in the middle of its image ($AlignRobotwithQR$ function, Algorithm 1 line 10). From the relative size of the lateral edges of the QR code, the robot determines whether it is in the right side, left side or centered; objects become smaller as their distance from the observer increases (see Figure 3). From the relative size of the lateral edges of the QR code, the robot determines its relative position ($side$). As shown in Fig. 3, when the robot is on the left, the right lateral side of the QR code is smaller than the left one, and the other way around. Thereafter, the robot will perform an $Approach$ action (Algorithm 1 line 11) as detailed later. If as a result, the robot is not still docked, the complete procedure restart from the beginning.

Obviously, the robot may complete a $360^o$ turn without finding the QR code with the correct dock ID. This scenario is likely to occur if the starting position of the robot is so close to an obstacle or too far from the code that it is not able to see the QR code properly from that perspective. In order to deal with these situations, we make use of the IR sensors. Every time the robot takes a picture looking for the dock, it stores the IR distance reading values from its front sensors. These values allow the robot to determine the closest obstacle, face it, and move away from it in order to gain a better perspective of the QR code. The QR code detection procedure is then repeated. If no obstacle is detected, and no QR code could be found, the robot is too far from any QR code (further than 3 m away) and thus perform a random walk till finding one.

The robot's approaching procedure towards the docking station (l 11 Algo. 1) is detailed in Algo. 2. We have defined 3 regions (see Table II) delimited by the readings of the IR

TABLE II: Docking algorithm regions

| | |
|---|---|
| VERY CLOSE | $distance < 0.75m.$ |
| CLOSE | $0.75m. < distance < 1.5m.$ |
| FAR | $1.5m. < distance$ |

sensors. The actions performed by the robot differ depending on the distance to the dock (or an obstacle) i.e the region it finds itself. If the robot is *FAR* from the dock, it approaches directly, going straight towards the dock until it leaves the *FAR* region. Once the robot is *CLOSE* to dock, it performs an *indirect approach*. It slightly turns to correct the angle of attack and heads to the dock, i.e if it is on the right side, it turns left with the purpose of being positioned perpendicular to the dock. Finally, when the robot is in the *VERY CLOSE* area, it checks whether the attacking angle is adequate. This angle estimation is done using both IR readings and the differences between the size of the lateral edges of the QR code. If the angle is $OK$, the robot advances straight to dock till it is docked. If not, a *very indirect approach* is performed, i.e the robot turns and moves forward to get a better angle of attack.

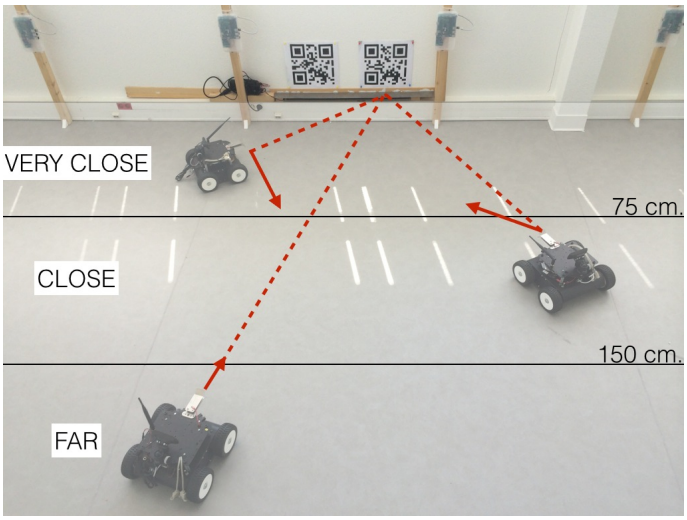



Fig. 4: Sample approaching behavior depending on the region.

As this procedure is based in the front IR readings, it does not discriminate between any objects (a wall or another robot). This allows the robot to avoid obstacles. An obstacle will force the robot to be in the *CLOSE* or *VERY CLOSE* region. The robot will therefore first turn and then advance forward, resulting a new starting point. The purpose of this work is not to implement a obstacle avoidance algorithm but a docking procedure in a relatively controlled environment. Nevertheless, this procedure allows the robot to avoid other robots already docked and dock in close proximity with thereof.

## VI. Results and Evaluation

To evaluate our approach, we carried out some experiments in the definitive platform environment as can be shown on Figures 5, 6 and 7. The objective is to dock a Wifibot (*Robot-2*) in a base station (*Dock-2*) signaled with a QR code above it. An identifier (*dock:2*) encoded in the goal QR code will allow the robot to distinguish this dock from others.

We have selected some relevant starting points. In each case two representative paths are displayed. The dotted line indicates the path followed when there is another robot (obstacle) present in the neighbor dock ( i.e. *Robot-1* is already in *Dock-1*). The solid line shows the case there is no any other robot in the docking area.

The *Point A* (Fig: 5) is placed 170 cm. left to the base

**input** : QR_dock: 4 points coordinates of the dock QR to determine the relative position of the robot.
**output**: Approaching action to dock. Returns $True$ when docked.

```
1  The robot has seen the Dock QR code;
2  The robot is aligned with Dock QR code;
3  region ←GetRegionFromIR();
   /* region: FAR/CLOSE/VERY CLOSE        */
4  side ← GetRelativeSideFromDock(QR_dock);
   /* side: Left/Right/Center             */
5  switch region do
6      case FAR
7          while region = FAR do
8              GoStraight();
9              region ← GetRegionFromIR();
10         end
11     end
12     case CLOSE
13         if angle_with_dock is SMALL then
14             while region ≠ VERY CLOSE do
15                 GoStraight();
16                 region ← GetRegionFromIR();
17             end
18         else
19             Turn(30°, not side);
20             GoStraight();
21         end
22     end
23     case VERY CLOSE
24         if angle_with_dock is SMALL then
25             while not docked do
26                 GoStraight();
27                 docked ← CheckDocked();
28             end
29             Return True ;
               /* Robot docked                */
30         else
31             Turn(90°, not side);
32             GoStraight();
33         end
34     end
35 endsw
36 Return False;
```

**Algorithm 2:** Detailed approaching procedure

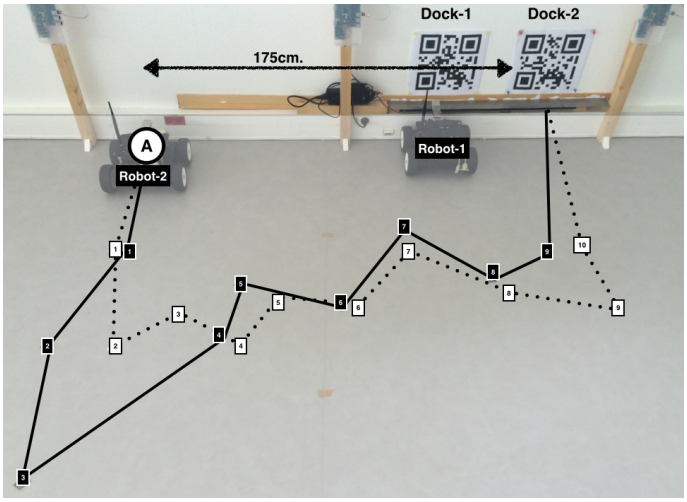



Fig. 5: Sample docking paths from point A to dock 2. Dotted and solid lines represent whenever Robot-1 is present in dock-1 or not respectively.

station. From that position the robot is not able to see the *Dock-2*. After a complete turn without seeing the QR code, it gets away from the wall detected thanks to the IR readings. This procedure is applied iteratively until the robot reaches a position from which dock can be discovered (Fig 5: point 3 in the solid line and point 2 in the dotted one) until it reaches a position from which it can decode properly the QR code signaling the docking station. We can observe as well that the the fact of having another robot present in the in a dock close to the goal, forces the robot to go farther before addressing the final docking procedure (point 9 in dotted line, Fig: 5). In both cases, the time the robot took to dock was around 4 minutes.

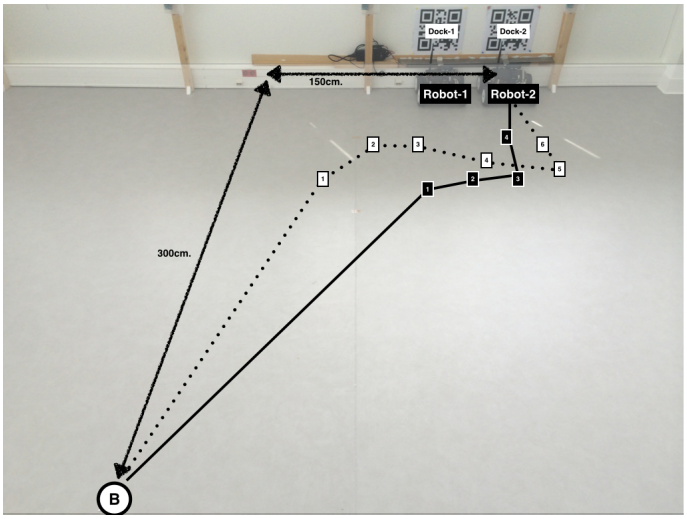



Fig. 6: Sample docking paths from point B to dock 2. Dotted and solid lines represent whenever Robot-1 is present in dock-1 or not respectively.

Another starting point (*Point B* , Fig: 6) is placed 150 cm. left of the base station and 300 cm far from the wall. We can observe how the robot goes straight to the dock until it reaches the *CLOSE* region (point 1 in both paths, Fig: 6). Again, the robot must go more to the right when another robot is docked to its left. There was no significant difference in the docking time for each case: around 1 min and 50s respectively.

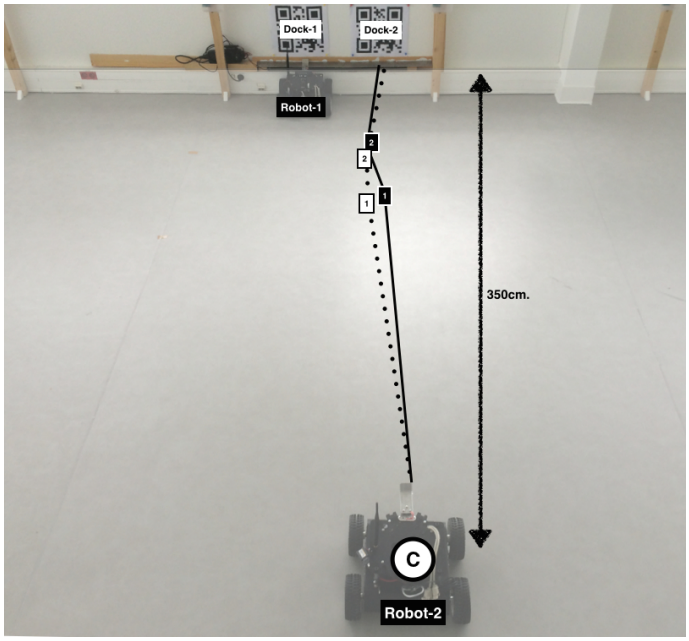



Fig. 7: Sample docking paths from point C to dock 2. Dotted and solid lines represent whenever *Robot-1* is present in dock-1 or not respectively.

A more favorable scenario is represented by the *Point C* (Fig: 7). The robot is placed 350 cm. far from the wall, just in front of the dock, From that point the robot took around 40 seconds to dock even when the other robot was docked.

Obviously, the time the robot takes to dock depends on the starting point. The most disadvantageous positions to start are those from where the robot is not able to see the dock (*Point A* , Fig: 5). In that case the robot has to complete an entire turn before getting away from the wall and trying again. Additionally, one process that slows down the docking time is the *Center* procedure (Algo: 1, line 10). The robot performs small turns to place the QR code just in the middle of its image before deciding whether it is in the right side or the left one (decision based on the size of the lateral edges of the QR codes). Here, not only a threshold has to be defined, but also how much will the robot turn to correct its position (depending on the distance to the goal) and the maximum number of attempts before going on with the next step. All those variables impact the time to dock and its reliability. Similarly, the parameters used to determine whether the angle is appropriate to dock affects the docking results. That is, the difference between the IR readings and the difference between the lateral edges of the QR code again.

Finally, although not the main goal of this work, we decide observe the behavior of the robot with obstacles. We place

another robot in the middle of the way to the dock (Fig: 8). When *Robot-2* gets close (point 1 in Fig: 8) to *Robot-1*, from the IR readings, it estimates it is in the *VERY CLOSE* region so it will perform a big turn that will allow to avoid the obstacle and restart the docking procedure from more convenient point (point 2 in Fig: 8) as we have seen previously. Even though this procedure works quite well in general, we have encountered some problems when the robot goes straight to a small obstacle. As the IR front sensors are disposed in angle, they may not detect a frontal object in that case. Another IR sensor aiming directly to the front should solve this situation.

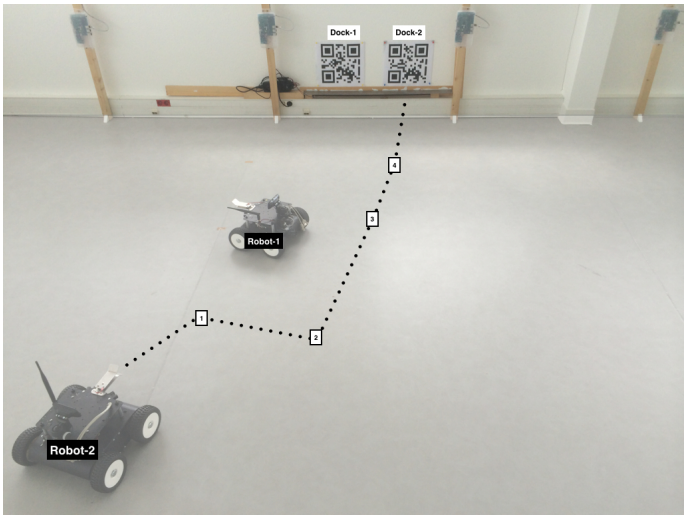



Fig. 8: Sample docking path with obstacle.

## VII. Conclusion

In this paper we propose to use QR code landmarks and Infrared distance sensors to enable autonomous docking. Experimental results obtained from different starting points show that the proposed solution is fully operational and robust. Additionally, the approaching procedure allows the robot to avoid obstacles to reach its final destination (the dock). This could be used for navigation proposes with the help of a localization system. There is also room to speed up the process and improve its performance with a more complex calculation for the relative position estimation. For the sake of simplicity and to avoid potential errors yield by calibration, we use just the relative size of the lateral edges to determine the relative side of the robot to the dock. Other parameters such as the surface of the QR in the picture could be used to estimate the distance. Nevertheless, the use of IR distance sensors for distance estimation rather than relying exclusively on computer vision methods results in a more robust and easier to deploy solution.

## VIII. Discussion and Future work

Future work will focus on study the reliability of the solution. In a near future we expect to integrate the docking solution within the testbed platform [2] to prove the robots long term autonomy. Tests with a large number of robots sharing the platform have be done to explore the limitations of the obstacle avoidance procedure and its real potential.

In order to speed up the docking procedure, visual servo control and techniques from image processing, computer vision and control theory merit to be explored.

Working with real time video processing (instead of pictures) while moving merits to be explored. Even though video decoding is supported by the ZBar [25] library, we chose to take pictures with the robot completely stopped every time we wanted to discover a landmark. Video decoding appears to exceed the computation capabilities of the robot adding a non tolerable delay. This delay prevented the robot to do the right movement at the right moment. The use of more powerful machines, lower resolution images or simpler landmarks could address this situation.

We will further investigate how to improve the landmark detection during movements and in different light conditions. More testing is also necessary to tune the different parameters involved in the decision making of the robot (i.e: thresholds to determine id the landmark is centered, the relative side the robot is or the angle to the dock) and the actions performed (ex. how long does the robot have to turn or go straight depending on its situation).

Finally, in a pre-production environment, tests with a large number of robots sharing the platform has be done to explore the limitations of the obstacle avoidance procedure and its real potential to be used within a navigation stack to go from a point to another with the help of a localization system.

Next steps also include the integration in the FIT IoT Lab experimental testbed to enlarge the set of possible experimentations with mobile entities.

## Acknowledgment

This work has been partially supporters by a grant from CPER Nord-Pas-de-Calais/FEDER Campus Intelligence Ambiante and the LIRIMA PREDNET project.

## References

[1] FIT-Equipex. http://fit-equipex.fr.
[2] FIT-IoT-Lab. https://www.iot-lab.info.
[3] *Using SensLAB as a First Class Scientific Tool for Large Scale Wireless Sensor Network Experiments*, (Valencia, Spain), 2011.
[4] Turtlebot. http://kobuki.yujinrobot.com/home-en/.
[5] Wifibot. http://www.wifibot.com.
[6] K. docking. http://wiki.ros.org/kobuki/Tutorials/Testing%20Automatic%20Docking.
[7] W. Walter, *The living brain.* WW Norton, 1953.
[8] S. Yuta and Y. Hada, "Long term activity of the autonomous robot-proposal of a bench-mark problem for the autonomy," in *Proc. of Intelligent Robots and Systems (IROS)*, (Victoria, Canada), 1998.
[9] I. R. Nourbakhsh, "The failures of a self-reliant tour robot with no planner," Tech. Rep. 998, Carnegie Mellon University, Robotic Institute, 1998.
[10] S. Oh, A. Zelinsky, and K. Taylor, "Autonomous battery recharging for indoor mobile robots," in *Proc. of the australian conference on robotics and automation (ACRA)*, (Sydney, Australia), 2000.
[11] M. C. Silverman, D. Nies, B. Jung, and G. Sukhatme, "Staying alive: A docking station for autonomous robot recharging," in *Proc. of Int. Conf. on Robotics and Automation (ICRA)*, (Washington DC, USA), pp. 1050–1055, 2002.

[12] Mobilerobots. http://www.mobilerobots.com/Accessories/ChargingSolutions.aspx.
[13] iRobot Create. http://www.irobot.com/global/en/home.aspx.
[14] iClebo Kobuki. http://iclebo.com/english/.
[15] K. Kim, H. Choi, S. Yoon, K. Lee, H. Ryu, C. Woo, and Y. Kwak, "Development of docking system for mobile robots using cheap infrared sensors," in *Proc. of International Conference on Sensing Technology (ICST)*, (Palmerston North, New Zealand), 2005.
[16] U. Kartoun, H. Stern, Y. Edan, C. Feied, J. Handler, M. Smith, and M. Gillam, "Vision-based autonomous robot self-docking and recharging," in *World Automation Congress (WAC)*, (Budapest, Hungary), 2006.
[17] T. Suriyon, H. Keisuke, and B. Choompol, "Development of guide robot by using qr code recognition," in *Proc. of Conference on Mechanical Engineering (ASME)*, (Denver, USA), 2011.
[18] H. Kobayashi, "A new proposal for self-localization of mobile robot by self-contained 2D barcode landmark," in *Proc. of SICE*, (Akita, Japan), 2012.
[19] E. McCann, M. Medvedev, D. J. Brooks, and K. Saenko, "Off the grid: Self-contained landmarks for improved indoor probabilistic localization," in *Proc. of Technologies for Practical Robot Applications (TePRA)*, (Woburn, USA), 2013.
[20] L. George, A. Mazel, *et al.*, "Humanoid robot indoor navigation based on 2D bar codes: Application to the nao robot," in *Proc. of Conference on Humanoid Robots (Humanoids) (RAS-Humanoids)*, (Atlanta, USA), 2013.
[21] F. Chaumette and S. Hutchinson, "Visual servo control. i. basic approaches," *Robotics & Automation Magazine, IEEE*, vol. 13, no. 4, pp. 82–90, 2006.
[22] Y. Mezouar and F. Chaumette, "Path planning in image space for robust visual serving," in *Proc. of Int. Conf. on Robotics and Automation (ICRA)*, (San Francisco, USA), 2000.
[23] M. Munich, P. Pirjanian, E. Di Bernardo, L. Goncalves, N. Karlsson, and D. Lowe, "Application of visual pattern recognition to robotics and automation," *IEEE Robotics & Automation Magazine*, pp. 72–77, 2006.
[24] R. Wifibot. https://svn.code.sf.net/p/roswifibot/code/trunk/.
[25] ZBar. http://zbar.sourceforge.net.

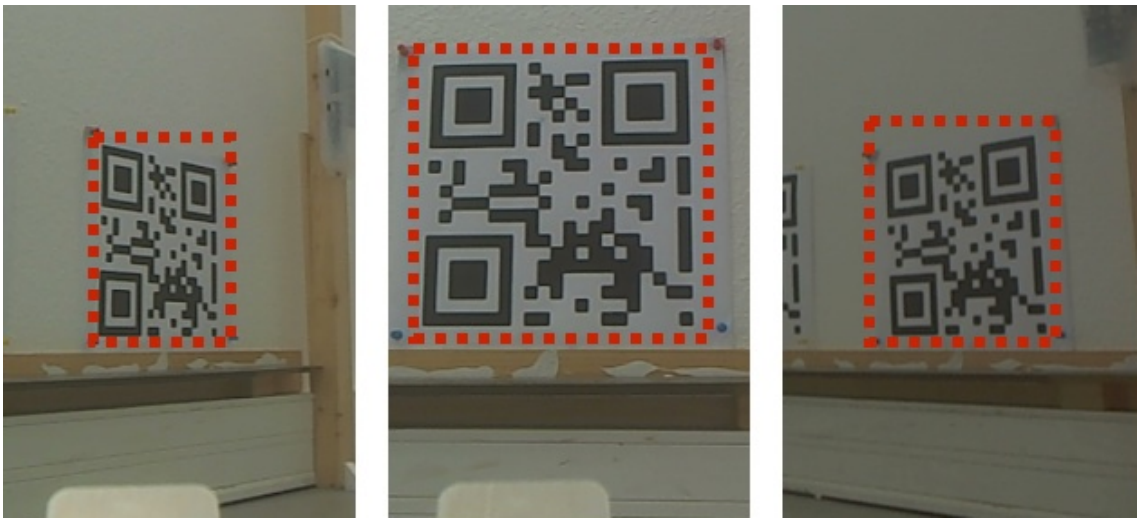


Fig. 3: Robot's views from: left, center and right.