Towards Remote Control of Mobile Robots to Help Dependent People

Yvon Autret, Jean Vareille, David Espes, Valérie Marc and Philippe Le Parc
Université Européenne de Bretagne, Université de Brest
Laboratoire en Sciences et Techniques de l'Information (Lab-STICC UMR CNRS 6285)
France
Email: {yvon.autret, jean.vareille, david.espes, valerie.marc, philippe.le-parc}@univ-brest.fr

Abstract—**In this paper, we focus on a Web-controlled mobile robot for home monitoring, in the context of Ambient Assisted Living. The key point is low-cost and the robot is built from standard components. We use a few sensors to allow the robot to estimate its position, its direction and the obstacles in front of it. An Ultra Wide Band system is used to estimate the position of the robot. A distant user controls the robot by using a map in the user interface. The result is a small robot that can be used inside or outside the house.**

Keywords-Home monitoring; Web control; UWB positioning.

I. INTRODUCTION

In 1898, Nikola Tesla demonstrated a remote-controlled boat [1]. It was based on the radioconduction discovered by French physicist Edouard Branly in 1890. One century later, the emergence of the Web technology provided new opportunities. The first Web controlled robot was developed at the University of Western Australia by Kenneth Taylor in 1995 [2]. At the beginning of the 2000's, Web development has led to the emergence of Service Robotics [3].

However, Web-controlled robots have rather remained unused until now, especially for Ambient Assisted Living (AAL) applications. A typical application consists of helping persons with diminishing mental or physical ability to stay at home as long as possible. When picking up the phone becomes too difficult, a mobile robot usable as a phone could be useful. In the same way, care helpers or relatives cannot spend all their time with a person. Devices that would be able to monitor what is going on in a house, and send the information to the care helpers could be of great interest. Cameras could be installed in every room. Such systems exist but they are not really acceptable because they are too intrusive. Thus, we think that a mobile robot could be more easily accepted. The robot can look like an animal. It can move in the house, and only one camera is required in the house. If the camera is considered too intrusive, it can be replaced by a lidar to analyze movements in the house.

Such robots are easy to build at affordable cost. Some of them are even commercially available. However, almost nobody uses them in real-world environments, such as AAL. The Romo example is typical [4]. The robot was launched in 2012 by the Romotive company. It is a mobile robot that uses a smartphone to control the motors. It can be remotely controlled from anywhere by using the smartphone connectivity. As soon as 2013, one Romotive co-founder wanted to move in the direction of making a robot that could solve real-world problems. After years of aimless decisions, Romotive's Website was shut down in 2016. Beyond disputes that have led Romotive to its fall, one key point appears. It is possible to build and sell toy robots, but nobody knows whether it is possible to build and sell at affordable prices, robots that can be used in the real world, especially in an AAL environment. In this paper, we will ask why. We will review the main criteria required to make an AAL mobile robot truly usable.

A. *The cost*

The cost must be kept as low as possible because it will probably be used by elderly people who often have tight budgets. It is inconceivable to rent a satellite channel to control the robot. In the same way, it is neither possible to use components, such as those found in military weapons, for example a €50000 inertial unit. From our point of view, the cost of an AAL robot should not exceed €1000. The price of a TV or a high-tech smartphone is also a good estimate.

B. *Performance of the network*

When a command is sent to a robot through a network, if an acknowledgment is received back in less than 200 ms, there is no perceptible lag between the triggering of the action and the visual result [5]. A guaranteed 200 ms round-trip-time (RTT) allows secured remote command of mechanical devices. In the case of AAL robots, a 300-500 ms RTT remains acceptable if the speed of the robot is low (1 km/h). When the RTT is beyond 500 ms, the operator feels something uncertain.

C. *Security of the system*

If a server is installed on or near the robot, it can cause serious security problems in the house. A server is never 100% secure. Even if techniques, such as traffic analysis are implemented, and if a problem is detected, who will handle the problem? It is not the role of the robot users.

If there is a wireless connection between a server and the robot, the radiations may cross the limit of the house and they can be captured and modified from the outside. Data will have to be encrypted but it may not be sufficient.

D. *Security of the persons and resilience*

If there is a failure, the robot may become dangerous. It may go anywhere in the house and hurt people. In any case, the speed of the robot must remain low. The robot should not exceed 1 km/h to avoid frightening the inhabitants. The resilience of the system is also very important. The robot must be able to work despite total or partial failure of one or more components. For example, if the network performance decreases, the robot should automatically reduce its speed. When a fault is detected, the robot must be able to restart, and eventually go to a fallback position. An accurate positioning system must be available.

E. *User interface*

The user interface must be designed for a semi-autonomous robot. When only using video feedback, controlling the robot is not easy. If images are not sent to the distant user for a while, the robot control may quickly get lost. The user interface must give accurate information about the robot, its position and its environment. The information must be redundant.

F. *Positioning*

Estimating the robot position is a key point. If the estimated position is not accurate, the whole system will collapse. The user interface will display wrong information, and the robot will be dangerous. Most of the previous criteria depend on the estimation of the robot position.

In this paper, Section II presents the proposed robotic system. We will show how the previous criteria have been taken into account. Section III presents the user interface. The results are shown in Section IV. The paper finishes by a conclusion and perspectives.

II. Designing a home robot for an AAL environment

A. *The mechanical base*

We use a very simple experimental mechanical base (Figure 1). There are four wheels mounted on gear motors and a wooden plate. An Arduino and a motor shield control the motors two by two. The motor shield is a 2x2A. It is based on a L298P chip. This means that the robot will slide slightly on the floor when turning. This choice reduces the cost but it will make the robot more difficult to locate. In the future, it might be necessary to have independent wheel control. The gearmotors rotate at a maximum of 84 revolutions per minute. The 120 mm wheels allow a maximum speed of 1.9 km/h. The motor torque is 1,0 kg.cm and the total mass of the robot can reach about 3 kg. This mechanical base is very reliable, especially if brushless motors are used.

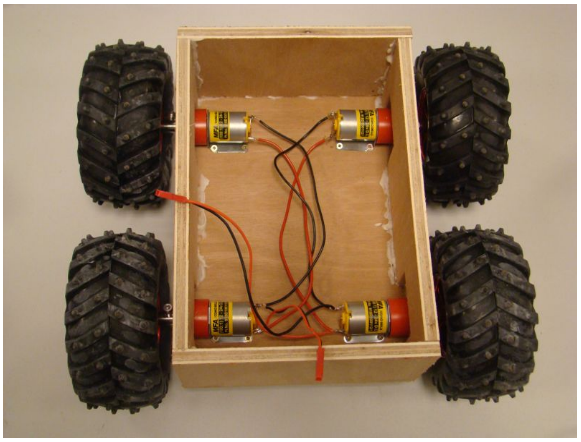


Figure 1. The mechanical base

B. *The proposed architecture*

If the mobile robot is in a house and the user in a different place, we have no choice but the Web to allow remote control. Another solution would increase the total cost too much. The remaining question is whether a thin client is preferred to a fat client. We have chosen a thin client for security reasons. A fat client would have been more powerful but the risk of security breach would have been higher. When using a thin client, we use a standard Web browser and rely on its security. The Web browser communicates with a Tomcat Web server that is fairly secure. The HTTP(S) protocol is used.

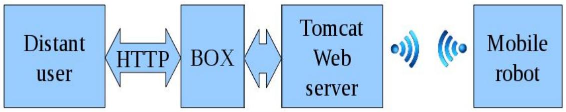


Figure 2. The proposed architecture

The architecture is shown in Figure 2. The distant user uses a Web browser to reach an Internet Box in the house, and next the Web server through an Ethernet cable. This ensures that there will be no wireless problems from the Box to the Web server.

Between the Web server and the robot we use a wireless Ultra Wide Band network (UWB) [6]. It will give us positioning capabilities.

The system works as follow. The Tomcat Web server is running on a computer that can be a Raspberry PI 2 or any other computer. A second server is running on the computer. The Tomcat server communicates with the second server by the mean of sockets. The second server rejects all communications except those coming from the Tomcat server. It is used to handle an Arduino connected to the computer. The Arduino has to manage an UWB communication with the robot. Thus, 128 bytes packets can be sent from the Web server to the mobile robot. When the robot is too far from the computer, UWB relays are required. Depending on the environment, relays must be added every 5 to 30 meters. UWB is managed by a Pozyx shield. Sending one hundred bytes from an Arduino UNO to the mobile robot and receiving a response of one hundred bytes takes 75 ms when using the I2C bus on the Arduino.

C. The sensors

As defined above, a distant user could make the robot move by using basics commands, such as forward, backward, right or left. If video is available, a remote control is possible.

A webcam is available on the robot. It is managed by a Raspberry PI 2. It is a light solution to stream videos over an IP-based network. The webcam is independant from the robot. The Tomcat Web server catches the video and sends it to the distant user when required. Thus, the webcam is not directly accessible from the outside. Only the Tomcat Web server can be accessed from the outside and security is kept relatively high because distant users must be identified in order to get the video images.

However, if the mobile robot is used by caregivers who do not know the house very well, video feedback is not sufficient because the experience shows that users are quickly lost. Moreover, estimation of the position of obstacles is not easy with video only. Thus, we have two main problems, estimating the obstacle positions, and estimating the robot position in the house.

Estimating the obstacle positions can be done by using a laser telemeter (Lidar) [7]. Such devices are available since several years. However their price can easily reach €2000. We rather use a €150 Lidar-lite that can measure distances in only one direction. To scan a 180 degree field in front of the robot, we mount the Lidar-lite on a servo motor.

To make the robot go forward and follow a direction, we also use a 9-axis accelerometer/magnetometer. Experiments have shown that for our problem, a Kalman filter is required. Without the Kalman filter, the magnetometer produces many wrong values. Using an extended Kalman filter does not seem to be necessary until now. We use a €30 CMPS11 tilt compensated compass module from Robot-Electronics [8]. The module includes a processor to compute a Kalman filter. It processes the raw values produced by the gyroscope, the accelerometer and the magnetometer. The compass output is pitch, roll and heading. To give correct results, the compass must be at 30 cm above the gear motors. Only heading will be used in our case. We will use that value to make the robot follow a direction. The distance traveled by the robot could also be computed from the accelerometer data, but the errors would accumulate and the position of the robot would be incertain. We will rather use UWB to determine the distance traveled by the robot.

D. Estimating the robot position

Estimating the absolute robot position is now possible, thanks to UWB. One of the main features of UWB signals is their potential for accurate position location and ranging. UWB technologies are often described as the next generation of real time location positioning systems. Due to their fine time resolution, UWB receivers are able to accurately estimate the time of arrival (ToA) of a transmitted UWB signal. This implies that the distance between an UWB transmitter and an UWB receiver can be precisely determined.

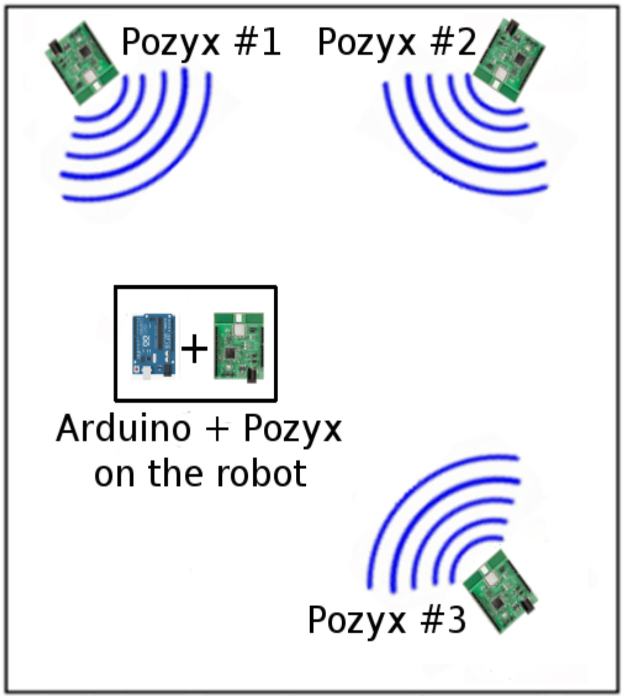


Figure 3. The positioning system

This feature of high localization accuracy makes the UWB an attractive technology for diverse ranging and indoor localization applications. It really allows 10-30 cm accuracy in ranging and promises the realization of low-power and low-cost communication systems [6].

We already have one UWB Pozyx module on the robot to ensure communication with the Web server. Three other modules will be added in the house to allow positioning. We will use the trilateration technique to estimate the position of the robot. Three Pozyx modules are positioned in the house (Figure 3).

The Arduino on the robot is connected to a Pozyx. It computes the distance from the robot to the three other Pozyxs. When the signal received from the reference nodes is noisy, the system is non-linear and cannot be solved. An estimation method has to be used. To get a satisfying approximated position of the mobile robot, we use the Newton-Raphson method [9]. This method attempts to find a solution in the non-linear least squares sense. The main idea of the Newton-Raphson algorithm is to use multiple iterations to find a final position based on an initial guess (for example, the center of the room), that would fit into a specific margin of error.

The first results of our experiments show that distance values are not constant due to multipath components. Hence, the precision of our system is about 30-50 centimeters. Such a precision is sufficient to know where the robot is in a room, but insufficient to pass through a door or something narrow.

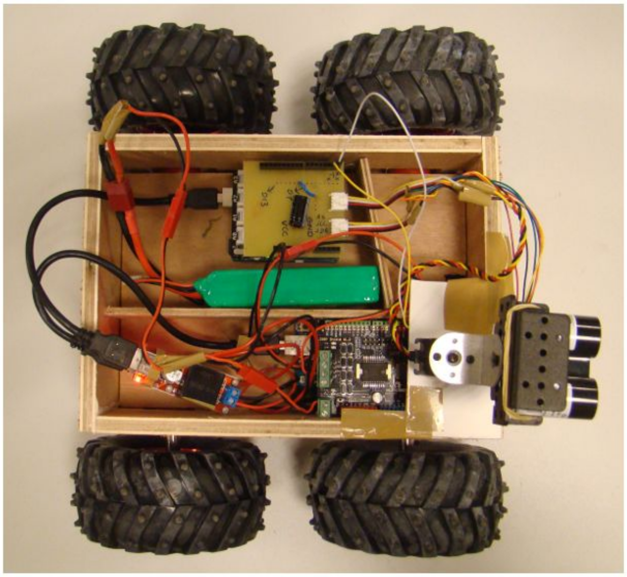


Figure 4. A part of the robot (compass and webcam not shown)

After the addition of sensors and UWB positioning, the mobile robot architecture is as follows. The robot includes several sensors that are managed by two Arduinos communicating through a 9600 baud serial link. The first Arduino manages the motors, the Lidar-lite laser telemeter, and the compass. It is able to make the robot move, stop if there is an obstacle, and follow a direction. It communicates with a second Arduino that estimates the robot position. The second Arduino periodically sends the estimated position to the first one. It can also send orders, such as stop, change the heading, or move forward in the current direction over a certain distance. To estimate its position, the second Arduino computes the distance between itself and the Pozyx modules. To compute the position, the Arduino sends the measured distances to the distant computer that processes the Newton-Raphson algorithm. Results are obtained faster if the computer has efficient floating point capabilities.

A part of the obtained robot is shown in Figure 4. A single LiPo 3s battery powers the robot. DC-DC converters are used to power the two Arduinos. One Arduino manages both the Lidar and the compass, another Arduino manages the Pozyx modules.

The robot is now able to estimate its position by using UWB Pozyxs. It is also able to communicate with a remote server installed in the house, to detect obstacles by using a Lidar-lite, and to follow a direction by using a compass. We must now propose a user interface to make all those features available to a distant user.

III. THE USER INTERFACE

A. *Using a map*

The main item of the user interface will be a map. We will try to show the robot moving on the map in real time. To build the map, we have chosen to extend an available solution: OpenStreetMap [10]. In France, most of the buildings, including the individual houses, are shown by OpenStreetMap. Thus, we can use these basics plans that show the edges of the buildings. We will superimpose a detailed plan on the basic OpenStreetMap plan. To build the detailed plan, we provide a tool that allows to draw on the basic OpenStreeMap. It is implemented by using the OpenLayers V3 (or V4) standard library [11]. Details such as furniture or door openings can be shown. The direction of the exterior walls relative to magnetic north is shown by OpenStreetMap, and all other elements can be placed on the map accordingly (Figure 5). More sophisticated solutions, such as Lidar analysis have not been experiment yet to automatically produce maps. Although limited, the current solution is easy to use and makes it easy to produce a relatively detailed plan.

When zoomed in, a room of a house can be seen in full screen. The robot position is shown by the letter “R”. The direction of the robot is shown by the direction of the letter. For example, if the letter is inverted on the map, the robot goes south.

To make positioning work, we must hang three Pozyxs on the walls. Our algorithm requires that they must be at the same height which can be different from that of the robot. In order to simplify configuration, the three Pozyxs must form a right angle triangle (Figure 6). Thus, in the user interface, there is something to indicate the position of the #1 Pozyx (P1), the position of the #2 Pozyx (P2), the distance between the #1 and #2 Pozyx (P1-P2), and the distance between #1 and #3 (P1-P3). The system deduces the position of the Pozyx #3 and there is no need to indicate directly its position. Pozyx configuration is very easy because walls of a house are very often perpendicular. The distant user must click twice on the map, the first click to indicate where the #1 Pozyx will be positioned, the second one to indicate where the #2 Pozyx will be positioned. Using a perpendicular axis for the Newton-Raphson algorithm we use in position estimation, can lead to problems because zero divisions can occur. In fact, experiments have shown that it is not a problem. If one position estimation can not be computed, the next one almost always can be computed. Even if the robot is stopped, the Pozyxs continuously produce distance values.

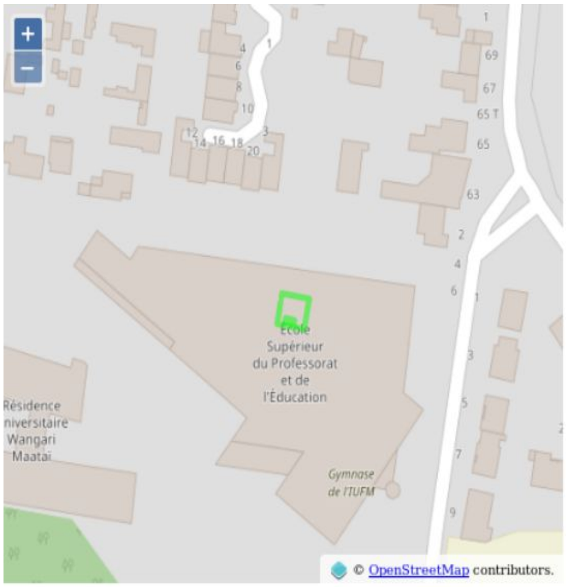


Figure 5. Example of OpenStreetMap plan with overlay

As soon as the Pozyxs are configured in the user interface, the robot position is displayed. The user interface shows the estimated distances between the robot and the Pozyxs by means of three circles. Those circles were used for debug at the beginning. We keep them in the user interface because they show a living system. The circles oscillate slightly continuously and the distant user can see if the system is working or not, and if there is no network problem. As seen above, the robot position is shown by the letter “R”. It should be at the intersection of the three circles.

The implementation has been done by using Javascript [12], Ajax [13], jQuery[14] and OpenLayers V3 [11]. An Ajax request is sent to the Tomcat Web server, the position is computed as seen above, and the result is sent back to the distant user, and shown on the user interface. As soon as the result is available, another Ajax request is sent and another position estimation expected. We have measured a round trip time (RTT) close to 500 ms when the distant user is in the same town as the robot. It takes about 100 ms to compute a distance from one Pozyx to another. As there are three distances to compute, we have a 300 ms duration. The results must furthermore be sent to the Tomcat Web server, and we have a RTT close to 500 ms to communicate between the distant user and the robot.

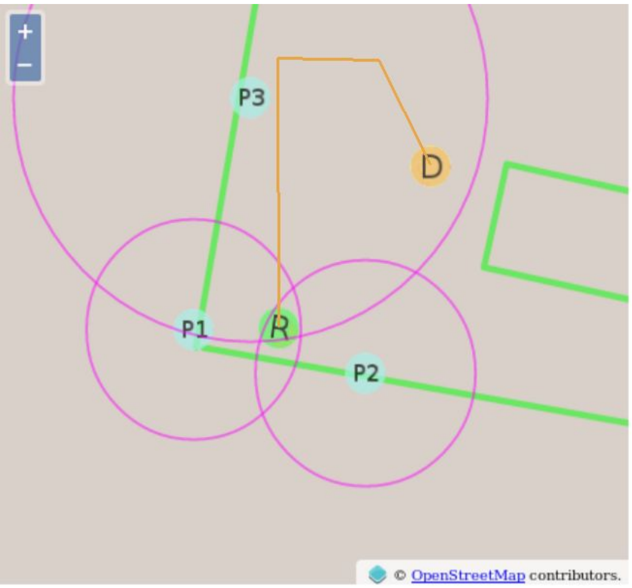


Figure 6. The user interface map

The RTT is also used on the robot. When the RTT increases, the robot automatically reduces its speed, or stops, or goes to a fallback position. Thus, if the robot does not receive commands from the Tomcat Web server, it stops.

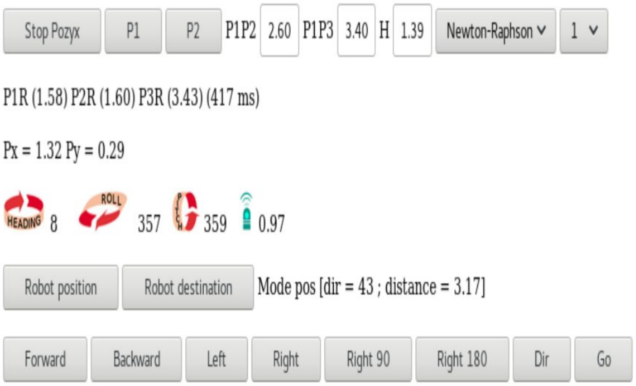
B. *Making the robot move*

To make the robot move, the distant user must indicate a destination position on the map by clicking once or more. In Figure 6, there is an orange stroke that can be split into three segments. To draw such a stroke, the distant user must click three times. The last click corresponds to the desired robot destination.

To make the robot reach that destination, the user interface will automatically send a set of commands to the robot. The three segments will be processed one by one, as follows:

- Computation of the direction of the segment (almost north for the first segment in Figure 6)
- Alignment of the robot in that direction
- Computation of the segment length
- Sending a command to the robot to make it move by the desired distance in the current direction
- Stopping the robot for two seconds to have a better robot position estimation
- Verification of the current position of the robot and adjustment (adjustment can be automatic or performed by the distant user)

We finally obtain a system that allows semi-automatic robot remote control. In addition to the map, the distant user has a control panel to monitor the robot (Figure 7).



P1P2 2.60 P1P3 3.40 H 1.39

P1R (1.58) P2R (1.60) P3R (3.43) (417 ms)

Px = 1.32 Py = 0.29

8 357 359 0.97

Mode pos [dir = 43 ; distance = 3.17]

Figure 7. Elements of the user interface

The current user interface is experimental. It shows the distances measured from the Pozyxs (P1R, P2R, and P3R), the Round Trip Time (417 ms in Figure 7), the position of the robot on the orthogonal axis defined by P1, P2 and P3 (1.32 m from P1 on the X-axis defined by P1-P2, 0.29 m from P1 on the Y-axis defined by P1-P3).

The user interface also shows the heading of the robot in degrees (8 degrees, almost north, in Figure 7), and also the unused pitch and roll values. The distance from the closest obstacle to the robot is also shown (0.97 m in Figure 7). There is also a set of buttons to define a new robot destination and make the robot move.

In the next section, we will show the results and review the criteria exposed in the introduction.

IV. RESULTS

A. *The total cost*

In the introduction, we said that the total cost should not exceed €1000. If there were no Pozyx, the total cost would be lower. The mechanical base costs about €100, the Lidar-lite about €200 [15], the compass about €30 [8], and the webcam about €100 including Raspberry PI 2 (Figure 8). We must still add the price of a computer that supports the Tomcat web server (from €50 to €500 depending on the model). We reach a maximum €900 total cost, Pozyx excluded.

One Pozyx is about €150 [16] and we need at least five. However, we think that it is not a problem. The very first Pozyxs were sold by the end of 2015 and the price will probably fall. The Decawave DW1000 chip used on the Pozyx module costs about one euro. The DWM1000 version that includes an antenna is now sold per unit for €30. We can expect UWB boards much cheaper in the near future. If a €50 UWB board was available, the cost criteria would be met.

Figure 8. The experimental robot

B. *Performance of the external network*

We have been testing Web performance for a decade. Tests have been done from Brest (France) to Auckland (New-Zealand). It is the longest distance possible in the world. Results are shown in Figure 9.

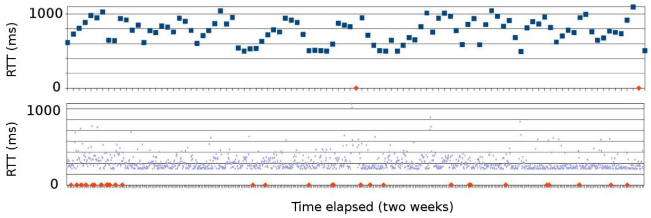


Figure 9. Web performance 2005-2015

The top diagram shows the measures taken in 2005 overs two weeks (horizontal axis in Figure 9). We have measured the Round Trip Time (RTT) between two computers, one located at the University of Brest (France) the other at the University of Auckland (NZ). We have obtained values from 495 to 1093 ms (vertical axis in top diagram in Figure 9). The average RTT is 768 ms. Exactly ten years later, the average RTT is 415 ms and most values are close to this average (bottom diagram in Figure 9). The minimum was 295 ms. The measures were performed between one Wi-Fi connected computer, located in a hotel in Auckland (NZ), and another computer located at the University of Brest (France).

This means that the Web can be used for remote control all over the world. However, we still have numerous RTT values greater than 500 ms. A RTT prediction system would be of great interest.

In fact, the problem comes from the local UWB network. The positioning process is very slow because communication between a Pozyx and an Arduino UNO is slow. One reason seems to be the use of the I2C Arduino bus. The Decawave chip on the Pozyx board uses the SPI bus (Serial Peripheral Interface Bus). The SPI bus must be converted to an I2C bus. Faster Arduinos or equivalent could improve communications. Direct connections to the Decawave chip by using the SPI bus could also produce improvements. That remains to be tested.

C. *Security of the system*

The security of the system is that of a distant user communicating with a remote Tomcat Web server through the encrypted HTTPS protocol.

The weakness is again in the local UWB network. Future studies will focus on the security of the local UWB network.

D. *Security of the persons and resilience*

The robot is able to detect any problem on the network and stop if required. Its low speed should make it safe for people. Experiments have shown the positioning system is accurate in the range between 30 and 50 cm. Perfect positioning is not available but it seems sufficient in a current AAL environment. The main remaining problem is door crossing. A better use of the Lidar could be the solution.

Moreover, we have no automatic charging dock yet. This is another key point that needs to be addressed. We assume that reliable standard charging docks will be available soon.

E. *User interface*

On the user interface, we can follow the robot on a map. As first experiments have shown that the Pozyx positioning system seems to be reliable, we have a control system based on standard components, such as OpenStreetMap. The time required to configure the system and make it work is very short.

F. *Positioning*

Even if the 30-50 cm obtained precision does not allow to make the robot go everywhere in house, it allows the robot to follow predefined paths. These paths must only be carefully chosen because the Pozyx signal may be easily stopped. The signal is very weak (about -40 dBm) and has shown to be very sensitive to metal obstacles, even if they are small.

V. CONCLUSION

The aim of this paper was to present a mobile home robot that could be helpful for old and/or dependent persons, and easily used by caregivers or relatives. Proposing a low cost solution, using high tech components, promoting simplicity were some of the key ideas that conducted this project.

This has been achieved by the use of a positioning system based on UWB Pozyx modules. Combined to a map in the user interface, it seems to be a promising technique.

However, several key points must be improved. Our knowledge of the security of such a system is very weak and must be improved. The accuracy of the positioning system must be also be improved to allow at least door crossing.

REFERENCES

[1] Nikola Tesla. [Online]. Available from: https://en.wikipedia.org/wiki/Nikola_Tesla 2017.07.03

[2] K. Taylor and J. Trevelyan, “A telerobot on the world wide web,” 1995 National Conference of the Australian Robot Association, 1995 July 5-7.

[3] “Robots With Their Heads in the Clouds,“ IEEE Spectrum, March 2011.

[4] Why Romotive shut down. [Online]. Available from: http://www.simplebotics.com/2016/02/the-rise-and-fall-of-robot-startup-romotive.html 2017.07.03

[5] F. De Natale and S. Pupolin, “Multimedia Communications,“ Springer Science & Business Media, 2012.

[6] U. Mengali, “Receiver architectures and ranging algorithms for UWB sensor networks,” 2012. [Online]. Available from: http://www.iet.unipi.it/dottinformazione/Formazione/OffForm2011/Mengali/SoloTesto.html 2017.07.03

[7] Lidar. [Online]. Available from: https://en.wikipedia.org/wiki/Lidar 2017.07.03

[8] CMPS11 - Tilt Compensated Compass Module. [Online]. Available from: https://www.robot-electronics.co.uk/htm/cmps11doc.htm 2017.07.03

[9] D. Espes, A. Daher, Y. Autret, E. Radoi, and P. Le Parc, "Ultra-wideband positioning for assistance robots for elderly," 10th IASTED (SPPRA 2013), Feb. 2013, Austria.

[10] OpenStreetMap. [Online]. Available from: https://en.wikipedia.org/wiki/OpenStreetMap 2017.07.03

[11] OpenLayers. [Online]. Available from: https://openlayers.org 2017.07.03

[12] Javascript. [Online]. Available from: https://en.wikipedia.org/wiki/JavaScript 2017.07.03

[13] Ajax. [Online]. Available from: https://en.wikipedia.org/wiki/Ajax_(programming) 2017.07.03

[14] jQuery. [Online]. Available from: http://jquery.com 2017.07.03

[15] LIDAR-Lite V3. [Online]. Available from: https://www.sparkfun.com/products/14032 2017.07.03

[16] Pozyx. [Online]. Available from: https://www.pozyx.io 2017.07.03