University of Technology in Wroclaw
Electronics department

Intermediate project

Two wheeled robot controlled by remote system

Author:
Mateusz Kupczyk

Course instructor:
Witold Paluszyski, Ph.D.

February 1, 2019

0.1 Introduction

Guiding mobile robot in workspace is not easy task, especial if global position of the robot is not known. This project is an approach of building global localization system which will be able to track robot using video image of the workspace. Project consists of two main parts which are robot and computer software, data between these two is transferred over WiFi network.

0.2 Mobile robot

0.2.1 Connectivity

First step of the project was building robot which will be able to communicate over WiFi network with remote computer. To give this possibility wireless module ESP-12E[4] was used, idea of using Bluetooth was abandoned because of two main reasons.The first reason is paring process which in case of Bluetooth requires additional time on connection start and second is required additional data serialization due to fact that most available solutions are wireless uart implementations. ESP module gives ability to connect any device that supports TCP connections to the robot. Thanks to this feature robot based on such module can be controlled by system which is located in considerable distance, connection through internet is also possible. What is more it is possible to connect few robots to the same network what gives possibility of future extension of the project what basic Bluetooth based system cannot provide.
To speed up development ESP module was configured in AP mode. Since ESP is programmable, Micropython[5] image was loaded into its memory. Loading micro python gives ability to execute python scripts from file saved in module memory, in this project written script opens TCP socket port and listens for data from connected device. Received data is parsed and send over SPI bus to central unit of the robot.

0.2.2 Main control unit and Sensors

The heart of the robot is micro-controller STM32F103[1] which handles motor control and processes sensor data. Micro-controller is connected with ESP module by SPI bus in order to receive data necessary to navigate robot. In current version of the robot, sensor such ultrasonic range finder HC-SR04[2] and encoders mounted on shafts of robots motors are used. Ultrasonic rangefinder is used to stop robot if any object appears in front of it. Wide angle of taken measurements gives possibility to use single sensor to detect all obstacles on the front of the robot.

0.2.3 Written software

Main loop waits for tasks which are ready to be completed. Tasks handled by the robot are:

- Checking if new data arrived and processing them,
- Periodically called motor update task which allows to control motor with use of implemented regulator,
- Sonar task which triggers new measurements and computes distance to object when data is available,
- Battery monitor task which monitors level of battery and shuts down robot when voltage is too low to avid battery damage,
- Controller task which is responsible for moving robot to the target,

Motors are controlled with use of H-bridge[3], their speed is controlled by duty of generated PWM signal. In order to avoid drifts when speed of wheels is changed PI regulator was implemented for each of two wheels.

Robot receives from remote system information about its global position and global target position. Path to the target is simple line, that's why at first, implemented algorithm try to minimize orientation error which is defined as difference between robots orientation and angle between line and global x axis. Second step is linear speed adjustment which is higher when orientation error is small enough.
Currently implemented target capturing algorithm have low performance and need to be tuned up.

In term of wireless data transfer robot handles three data functions which are:

- Set coordinates of current target,
- set current robots global position,
- set current controls(remote control of robots drive).

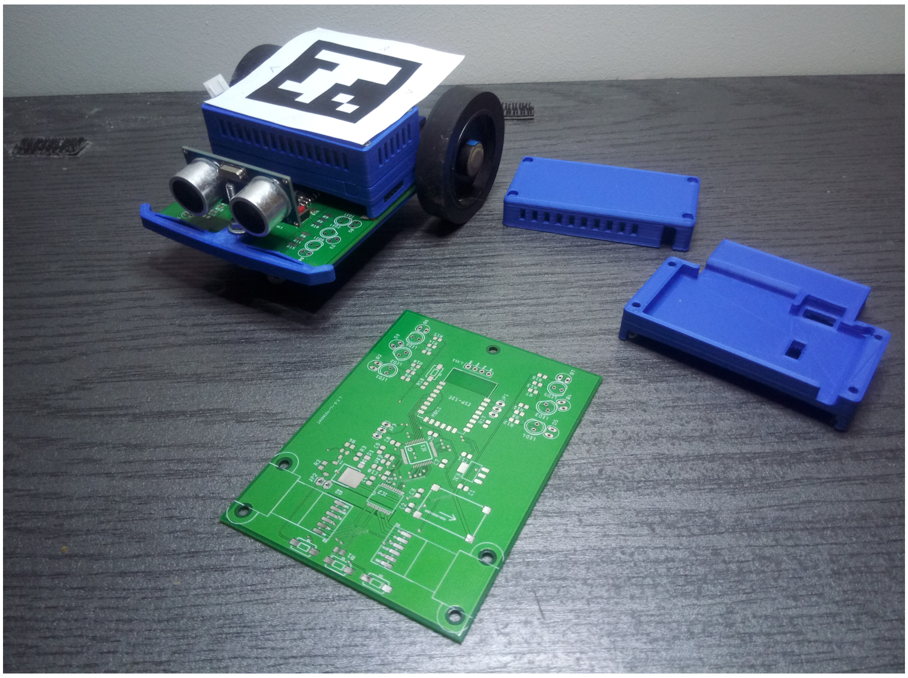

Figure 1: Robot build in as a part of the project. On front - PCB board and 3d printed parts on the left

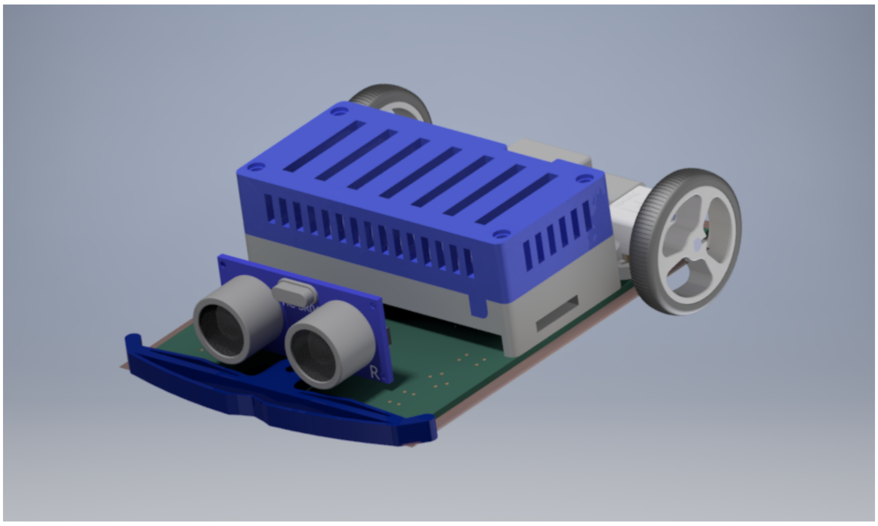

Figure 2: Mobile robot 3d model in Autodesk Inventor

To connect all electronic parts PCB board was designed and ordered. To cover electronics special covers were designed using Autodesk Inventor and next printed on 3d printer. Custom made 3d parts not only gives robot nice look but also protects electronics and keeps robots 2-cell LiPoli battery in place.

0.3 Computer Software

Computer software was written with use of ROS[7] package which allows to build complicated projects in form of package of nodes which are able to communicate with each other and run independently. Additionally written software utilizes libraries such as:

- OpenCv[6] - image processing library,
- Aruco[8] - library which allows to detect special generated markers,
- Qt[9] - library used to build user interface.

Moving back to ROS, whole software have form of three nodes which are:

- Network node which handles communication with robot,
- User Interface node which provides user interface,
- Video node processing image.

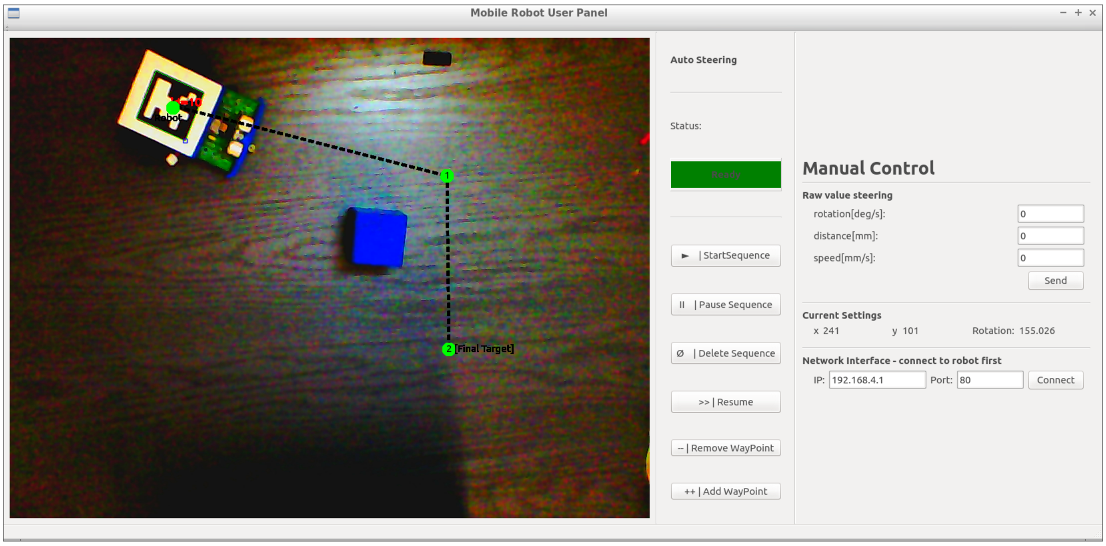

Figure 3: User interface of computer program

Main features which computer program provides through user panel:

- Manual control of the robot, it is possible to set parameters such as ω and v of the robot.
- Auto steering which allows to enter set of points which robot have to follow. Executing of the path follow can be stopped any time.

In order to localize robot Aruco library was used. This library allows to detect placement of special marker on the image. In this project only information such as coordinates and orientation is used but library allows to obtain whole information about marker placement in 3d space. Used Marker can be seen on Figure 1 where is placed on top of the robot and Figure 3 where detected marker is marked with first green dot(labeled as 'Robot') in target queue.

Figure 4: Test place of the system

0.4 Conclusions

- Project is large, consists of many different parts, requires developing software on many different platforms which have to cooperate with each other. Finding mistake and fixing it is time demanding, often change in single component requires change in others. Building good project architecture takes a lot of time on beginning, later there are only benefits,
- Well prepared workplace speeds up project development,
- Determining robot position basing on image expensive in terms of computation power,
- Ultrasonic range finders are good to detect obstacles,

- High frame rate of camera is a key in such project. Image of fast moving robot taken with cheap camera is blurry and features such as marker cannot be detected,
- Delay between capturing image, finding position and sending it to the robot causes robot to obtain inaccurate information about its global localization. Knowing kinematics of robot and time of global position sample it is possible to obtain more accurate position information,
- Systems like this should be used to periodically correct robots local position measurements when the highest accuracy is possible(robot stops in place), relying only on feedback from remote system is ineffective but possible(could be used in case of mobile robots internal measurement units failure).

Bibliography

[1] STM32F103 Reference manual
https://www.st.com/content/ccc/resource/technical/document/reference_manual
/59/b9/ba/7f/11/af/43/d5/CD00171190.pdf/files/CD00171190.pdf/
jcr:content/translations/en.CD00171190.pdf

[2] Ultrasonic rangefinder HC-SR04 datasheet,
https://www.mouser.com/ds/2/813/HCSR04-1022824.pdf

[3] TB6612FNG H-bridge datasheet,
https://www.sparkfun.com/datasheets/Robotics/TB6612FNG.pdf

[4] ESP-12E Datasheet,
http://wiki.ai-thinker.com/$_{m}edia/esp8266/a014ps01.pdf$

[5] Micropython, Quick reference to ESP8266,
https://docs.micropython.org/en/latest/esp8266/quickref.html

[6] OpenCv home page with available to download documentation,
https://opencv.org/

[7] ROS home page with available documentation and tutorials,
http://www.ros.org/

[8] Aruco library home page with available to download documentation,
https://www.uco.es/investiga/grupos/ava/node/26

[9] Aruco library home page with available to download documentation,
https://www.qt.io