PETR GONDEK (ID: 0745011)
10.10.2018

# Mobile Robots
# First checkpoint protocol

## INTRODUCTION

For the first iteration we have to do:

- Install OS on Raspberry PI and configure it (WiFi, SSH)
- Install ROS and Arduino on personal computer
- Install ROS and Arduino on Raspberry PI
- Code testing program

## SOLUTION

### INSTALLATION OS AND CONFIGURATION

Anyone from our team does not have an external monitor. What we needed first was to make RPI able automatically connect to our WiFi and set up SSH server.

We downloaded the Ubuntu for RPI and we installed it. Then we set up SSH server and WiFi how was described in manual.

After first reboot the auto-update started so we were forbidden to use apt-get for a while. We turned the auto-updates off.

### INSTALLING ROS AND ARDUINO

We added new repositories, updated package references and started installation as was described in manual. On our personal computer we used the Ubuntu for Windows. Main problem was that I installed the 18.02 version and ROS is working only on 16.04. It took a time to figure it out.

During the installation we lost shh connection few times. That was a problem because we lost screen with running apt-get and the installation somehow froze. I think it was waiting for user input. We killed it, started apt-get with auto-repair procedure. Then we installed application Screen so we were able to attach the screen next time.

After everything was installed we created ROS workspace. Then we continued with workspace manual. That was without problem.

Because we are using only ssh to access RPI we needed to broadcast GUI from application (mainly from Arduino) to our computers. We installed the XServer on our computers and did configuration on RPI. RPI detects user's IP (who connected via SSH) and sets the environment variables for XServer so connected user can immediately see the GUI output.

After all these steps we made backup image.

## CODING THE TESTING PROGRAM

We installed GIT and read introduction to how to use GIT with ROS packages. We created GitHub repo for multiplying package.

We created ROS package by command from manual. Then we copied the example code from tutorial and changed it little bit. We created two queues: /number where user input belongs and /multiplied where is stored output from Arduino. RPI subscribed to /multiplied and publish to /numbers. Arduino did it the other way, subscribed to /numbers and published to /multiplied.

When we sent the message to /numbers we set a flag (message_send) and we were trying to read a message from /multiplied until some occurred. When we received a message from /multiplied we printed the message and took flag away. After that user could type another input.

Arduino did it in easier way. It was trying to read from /numbers until input occurred. Then number was multiplied by 2 and send to /multiplied.

We handled subscriber and publisher in one file – in one application.

# CONCLUSION

Everything is installed correctly. RPI can connect to hotspot from my laptop automatically so we can easily connect via SSH. Software for developing is installed on both team member's personal computers and also on RPI.

Testing program is working correctly. RPI reads the input and sends it to Arduino. Arduino does the math logic and returns the output to RPI. Then RPI shows the output.

Everything is prepared for next checkpoint, so we can start code movements of our robots.