Getting started with the Arlo Robot

Kim S. Pedersen

September 5, 2021

1 Introduction

This note explains how to get started on using and programming the Arlo robot.

The Arlo robot [5] is an open source and open hardware mobile robotics platform consisting of a base containing two wheels with motors and two support wheels. The robot is equiped with four sonar ping distance sensors (capable of measuring distance to objects in the range of 2cm – 3m from the sensor) and a HD color Raspberry Pi camera. Motors are controlled by a DHB-10 dual H-bridge motor controller which is connected to an Arduino Uno board with the Board of Education (BOE) shield attached to it. The Arduino Uno and BOE shield controls sonar sensors and the motor controller. In order to allow for high level programming, the Arduino board is connected to a Raspberry Pi 3 Model B computer-on-a-board via a USB cable. The Raspberry Pi camera is attached to the Raspberry Pi board.

The DHB-10, Arduino and Raspberry Pi boards are powered by two 12V lead batteries that can be charged with a 12V lead battery charger. The Arlo robot is equiped with a power distribution board which includes a connector for the charger. On top of the power distribution board you find two switches that light red when on.

To switch on the Arduino and Raspberry Pi flick the switch called `Main`. This automatically powers up the Raspberry Pi and the Arduino microcontroller and starts running the
`serialRobotArlo` command interpreter on the Arduino. The Raspberry Pi boots the Raspberry Pi OS into user login mode.

To switch on the DHB-10 motor controller you have to flick the switch called `Motors`.

Please respect these rules of operations:

1. NEVER power up the robot motors (`Motors` switch on) while having it on a table. The robot may drive off the edge and fall down! The robot should be on the floor at all times when the `Motors` switch is on!

2. Start by only flicking the `Main` switch on. Once the Raspberry Pi is up, you may flick on the `Motors` switch.

3. It is your responsibility to avoid driving into humans, other robots, objects and walls. Be aware of your surroundings. Also be aware of the other robots driving around you.

4. If the robot runs amok, you can stop it by flicking the `Motors` switch off.

5. Remember to attach the robot to the charger when you return the robot to the robot laboratory.

1.1 Arduino Uno

The Arduino board runs a program called `serialRobotArlo` (the source file is called `serialRobotArlo.ino`) which interprets simple commands send over the USB connection which allows for control of the motors and reading the sensors. This program uses the `ArloRobot` library. The source code for both of these are available in Absalon under `Material`.

If you want to make changes to the `serialRobotArlo` program you first need to install the Arduino IDE software by following the instructions on [1]. Next you have to install the `ArloRobot` library by following the instructions on [6]. Finally, to upload a new program connect your computer with the Arduino using a USB cable and follow the instructions for the Arduino IDE software.

1.2 Raspberry Pi 3 Model B and Raspberry Pi Camera

The Raspberry Pi 3 Model B is a computer on a single board based on an ARM CPU and it runs a special Linux distribution called Raspberry Pi OS. See the technical details of the Raspberry Pi computer and camera at [9] and [3]. There are also a large amount of online help resources for the Raspberry Pi available at [4] and other places on the internet. As persistent storage, the Raspberry Pi has a micro SD flash ram card which contains the Linux installation as well as user data.

It is possible to connect a USB keyboard and mouse to the Raspberry Pi boards USB ports and a HDMI screen via the HDMI connector on the side of the board. WARNING: Do NOT bend or break the Raspberry Pi camera cable strip while attaching USB devices or HDMI screen (or in any other situation for that matter)!

However, we recommend that you access your robots Raspberry Pi via SSH (see Section 1.3 for details).

When the Raspberry Pi powers on, it boots the Linux distribution into GUI desktop mode. The board is setup with one user called `pi` with password `DIKU4Ever`. This user has super-user rights and can use the `sudo` command.

It is possible to switch between desktop and command line mode and if you want to restart the graphical interface (X-Windows system based) enter the command `startx`.

In order not to loose files it is important to power off the Pi board properly. This can be done from the windows system or from the command line by issuing the command:

```
sudo poweroff
```

You may have to enter the password of the `pi` user. Notice that the Raspberry Pi is shutdown when the green diode on the board (near the power socket just besides the red power indicator) switches off.

If you just want to reboot the Raspberry Pi then just issue the command:

```
sudo reboot
```

The Pi board comes complete with an installation of Python (both version 2.7 and 3.5). We recommend using Python 3 by issuing the command `python3`. We have also installed the OpenCV library on the board [2].

You may find various example programs in Absalon under `Material`.

1.3 Network

The Raspberry Pi can connect to the internet via ethernet or the build-in WIFI controller. If you attach an ethernet cable to the Raspberry Pi ethernet port while it boots the Pi connects automatically to the internet using the ethernet port.

Because of security restrictions in Eduroam we cannot use this network. Instead we have two options: Robo network or personnel wifi hotspot.

1.3.1 A note on SSH

You need an ssh client installed on your laptop to connect to the robot. On Linux and MacOS this is part of the default installation and is available as the command line command, `ssh`.

On Windows, you need to install a ssh client. We recommend using PuTTY available from [7].

1.3.2 The Robo WIFI network

In room 4-0-17 (the old DIKU library) the WIFI access points reveals a network with SSID `robo` (the network is available only in this room). It is a WPA2 network with password `R2-D2andC3PO`. The Pi boards on the Arlo robots are setup to connect to this network. You can also connect your laptop to the network and by using SSH you can then connect to the robot (the robot and your laptop must be on the same network).

You can now use ssh from your laptop to login to the Raspberry Pi by issuing this command (as an example lets assume that the robot is ArloUno which has IP address `10.69.252.12`):

```
ssh pi@10.69.252.12
```

and give the password `DIKU4Ever`. Remember to change the IP address to the address of your robot – see table 1 for other robots.

1.3.3 Personal WIFI hotspot network using your smartphone

If the above network fails you can use your smartphone to setup a local network (a WIFI hotspot) that you can connect both the robot and your laptop to. In this way you can remote access the robot from your laptop without the need for a keyboard and screen.

The Pi board can be setup to connect to a WIFI network using the on-board WIFI from the command line. You need to enter your security credentials into

Table 1: Static IP addresses of robots on the Robo network

ProtoArlo	10.69.252.11
ArloUno	10.69.252.12
ArloDue	10.69.252.13
ArloTre	10.69.252.14
ArloQuattro	10.69.252.15
ArloCinque	10.69.252.16
ArloSei	10.69.252.17
ArloSette	10.69.252.18
ArloOtto	10.69.252.19
ArloNove	10.69.252.20

`/etc/wpa_supplicant/wpa_supplicant.conf` under the relevant network block and set `disabled=0`.

Open the config file in an editor such as

```
sudo nano /etc/wpa_supplicant/wpa_supplicant.conf
```

Add before the robo network entry, your phones network SSID and password, e.g.:

```
network={
        ssid="MyPhoneNetwork"
        psk="MyPassword"
        key_mgmt=WPA-PSK
        disabled=0
}
```

Change the `ssid` and password in `psk` to match the SSID name and password of your phone network.

After rebooting the robot raspberry pi (`sudo reboot`), the robot will connect to your phones network (remember to switch on the hotspot on your phone) using the DHCP protocol and will receive a dynamic IP address. You can see the range of IP addresses either on your phone or by connecting your laptop to the phone network and checking what IP address it receives (how, depends on your OS).

You might want to fix the robot IP address to be static (the same whenever it connects to the network) by choosing an arbitrary IP address in the range of your phone network (below I choose 192.168.252.14). You then need to add the following lines in `/etc/dhcpcd.conf` (remember to `sudo nano /etc/dhcpcd.conf`),

```
# Added by Kim S. Pedersen, Aug. 2020
# Static IP on phone network
ssid MyPhoneNetwork
static ip_address=192.168.252.14/24
static routers=192.168.252.1
```

Change `ssid`, `ip_address`, and `routers` to match your phone network and your chosen IP address for the robot.

When you reboot the robot it will attempt to connect to your phone network using the static IP address. This is a hack, and will work most of the time, but it can happen that DHCP assigns the robot IP address to another device on the network, causing problems. If your phone allow you to configure DHCP to reserve some IP addresses then add the robot IP address to the list of reserved addresses.

If configured correctly, the robot will choose the robo network if your phone network is not available.

1.4 Using GUI remotely on the robot

1.4.1 Using VNC

You can use Virtual Network Computing (VNC) protocol to remotely access the desktop of the robot Raspberry Pi. The Pi is running a VNC server so the only thing you need is a VNC client installed on your laptop and that your laptop is on the same network as the robot Pi. We recommend using the VNC Viewer from RealVNC (free of charge) available at [11] or TigerVNC available at [10]. Both are available for all major operating systems. Other VNC clients exists and will work as well. Install the client and connect to your robot (with the IP address from above).

1.4.2 Using X11 forwarding in SSH

You can establish a SSH connection to the robot that allows you to open graphical windows on your laptop using X-windows by using the `-Y` option on the `ssh` command, e.g.:

```
ssh -o "Compression yes" -o "TCPKeepAlive yes" -Y pi@172.20.10.5
```

This allows you to run a X-windows GUI program on the robot that can open a graphical window on your laptop.

However, this requires that you run an X-windows server on your laptop. This should work out of the box for Linux users. On MacOS you need to install XQuartz (free from Apple) and run this program every time you want to establish a GUI connection. For Windows users, there are X-server implementations out there, but you are on your own.

This approach is very very slow because it requires a lot of network traffic, so we do not recommend this approach if VNC works.

1.5 A simple program

In Absalon under `Material` you find the Python script `simple_arlo.py` and the module `robot.py` which shows how to communicate from the Raspberry Pi to the Arduino and issuing simple commands for driving the robot and reading the sensors. Worth noticing is that it is necessary to wait a second (or less) after opening the serial port connection before starting communication with the robot. This is in order

to allow the Arduino board to finish setting up the communication. Also notice that communication may fail if you send commands too fast to the robot.

The complete list of robot commands can be found in the source file `robot.py`. In `robot.py` you find a class called `Robot` which contains a collection of methods for communicating with the robot. Fell free to change or add to this API.

1.6 Tips and Tricks

When you start developing programs for your robot you will need a good development workflow. We recommend using a Git repository - see how below - and develop code on your laptop.

You should AVOID using file synchronization e.g. such as sshfs, Dropbox, OneDrive, or similar, since this will eat up bandwidth on the Robo network which will cause problems for all groups later in the course.

1.6.1 Using a Git repository

Git [8] is a revision control system that allows you to keep track of changes to your code and share with your team members. Git is implemented both as a command line tool with the command `git` and as various GUI / Desktop tools (the service provider usually provides some recommendations) and many IDE's integrates and supports Git.

Start by creating a Git repository at a free service such as `https://github.com`, `https://gitlab.com`, or similar. For this to work, all team members should have an account and should be added as contributors to the repository (how to do this depends on your choice of service).

Once you have created the repository at your choice of service, you should make a local copy on your laptop and on the robot using the `git clone` command (for this you need a URL for your repository provided by your service provider). Now each team member can develop code on his/her laptop and put local changes into revision control using `git commit`, e.g.

```
git commit -a -m "I have made changes to bla bla"
```

Before you move to the next step make sure that you have made a commit of your local changes.

Once you are ready to share with your team members and test on the robot you should do the following set of commands in this order:
Make sure you have the latest version of all files - your team members might have made some changes you need and it is your responsibility to check that everything works before you share your files with the others,

```
git pull
```

Now you are ready to share the changes you have committed to your local copy of the repository. Do this by

```
git push
```

On the robot you just need to do

```
git pull
```

in the directory where you store your code. This should give you the latest version of the code in your repository and you can now test your code on the robot.

1.6.2 Suggestions for editors and IDE's

For Python development you can use any text editor you like or you can use an Integrated Development Environment IDE.

Currently, these are good choices of IDE's for Python:

Visual Studio Code: A free IDE by Microsoft that are available for most OS. Support Git repositories inside the IDE.
`https://code.visualstudio.com/`

PyCharm Community Edition: A free IDE by JetBrains that are available for most OS. Support Git repositories inside the IDE.
`https://www.jetbrains.com/pycharm/`

Spyder: A free and open-source IDE that are available for most OS.
`https://www.spyder-ide.org/`

And there are other choices - ask your fellow students are share with each other.

1.6.3 Troubleshooting

Robot not driving straight: The robot does not drive straight ahead when you apply the same speed on both motors. This is normal behaviour and you need to program your way around this. The problem is that one of the motors delivers a little less power than the other, so to compensate for this you can reduce the power on the stronger motor by some factor you find by experiment. For most cases this factor is a constant, but in principle it might change when the motor speed is changing.

Robot does not move: The motors need a minimum amount of power in order to be able to turn the wheels (due to friction between wheels and floor). If your program runs for a longer period of time it might blow one of the motor fuses. Try to increase the speed in your program and see if this works. If not, then get help from Kim.

Robot freeze / crash: Sometimes you might experience that the robot freezes or crashes. If you still have network access to the robot, you can try to quit your program (Ctrl-C or using the `kill` command). Then switch off `Motors`, count to ten, and switch `Motors` on again. Check if you program runs again.

Robot freezes when using VNC: This is a known problem caused by a combination of Raspberry Pi OS implementation of the VNC protocol and the Robo network configuration. If this continues to be a problem, the only work-around is to switch to a mobile phone hotspot - see section 1.3.3.

References

[1] Getting started with arduino. https://www.arduino.cc/en/Guide/HomePage, August 19 2015.

[2] Opencv.org. http://opencv.org/, August 19 2015.

[3] Raspberry Pi Camera Module. https://www.raspberrypi.org/products/camera-module/, August 19 2015.

[4] raspberrypi.org. https://www.raspberrypi.org/, August 19 2015.

[5] Arlo robotic platform system. https://www.parallax.com/product/arlo-robotic-platform-system, August 31 2017.

[6] Arlorobot arduino library. http://learn.parallax.com/tutorials/robot/arlo/arlo-arduino-uno-boe-shield-brain/arlorobot-arduino-library, August 31 2017.

[7] PuTTY. https://www.putty.org/, August 2020.

[8] Git. https://git-scm.com/, September 5 2021.

[9] Raspberry Pi 3 Model B. https://www.raspberrypi.org/products/raspberry-pi-3-model-b/, September 5 2021.

[10] TigerVNC. https://tigervnc.org/, September 5 2021.

[11] RealVNC. VNC Viewer from RealVNC. https://www.realvnc.com/en/connect/download/viewer/, August 2020.