

## BACHELOR

### Installation E-Puck Setup

van de Weijer, Job A.

*Award date:*  
2022

[Link to publication](#)

#### **Disclaimer**

This document contains a student thesis (bachelor's or master's), as authored by a student at Eindhoven University of Technology. Student theses are made available in the TU/e repository upon obtaining the required degree. The grade received is not published on the document as presented in the repository. The required complexity or quality of research of student theses may vary by program, and the required minimum study period may vary in duration.

#### **General rights**

Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain

DEPARTMENT OF MECHANICAL ENGINEERING

2021-2022 QUARTILE 2 & 3

---

**Installation E-Puck Setup**  
Bachelor End Project

---

*Eindhoven, April 24, 2022*

**Project coordinators:**

A.A.J. Lefebber  
A. Bayuwindra

**Authors:**

J.A. van de Weijer      1371517

# Contents

<b>1</b>	<b>Intro</b>	<b>1</b>
<b>2</b>	<b>Overview of the setup</b>	<b>2</b>
2.1	The e-puck . . . . .	2
2.2	The camera . . . . .	3
<b>3</b>	<b>Problems with the old PC</b>	<b>4</b>
3.0.1	Mex files . . . . .	4
3.1	Later information about the error . . . . .	4
<b>4</b>	<b>Manual for making Bluetooth connection</b>	<b>5</b>
4.1	Windows . . . . .	5
4.1.1	Problems with Bluetooth on Windows . . . . .	6
4.2	Linux . . . . .	6
4.2.1	Getting permission . . . . .	7
4.2.2	Libraries for running the shells . . . . .	7
4.2.3	e-puck pairing . . . . .	8
<b>5</b>	<b>Setting up the camera and its problems</b>	<b>10</b>
5.1	Setting up the files . . . . .	10
5.1.1	Structure of the files . . . . .	10
5.2	The camera under Windows & Linux . . . . .	10
5.3	Missing required modules . . . . .	10
5.4	Marker detection . . . . .	11
<b>6</b>	<b>Big problems during the project</b>	<b>12</b>
6.1	No Profile with UUID: . . . . .	12
6.2	Missing libraries . . . . .	12
6.3	Arrival delay of the new computer . . . . .	13
<b>7</b>	<b>Recommendations</b>	<b>14</b>
7.1	The use of a newer camera . . . . .	14
7.2	The use of Robot Operating System . . . . .	14
7.2.1	ROS for complex systems . . . . .	14
7.2.2	New camera and software for ROS . . . . .	15
7.2.3	Long time support . . . . .	15
<b>8</b>	<b>Reflection</b>	<b>16</b>
8.1	Lessons learned for the work field . . . . .	16

# 1 Intro

Nowadays, there are a lot of technical challenges concerning mobile robots. For example, the rapid increase of the use of self-driving cars, robots used in warehouses for order picking, intelligent navigation and coordination of traffic in urban areas, and so on. These technologies are thoroughly tested before they are put into practice[1][2][3]. Simulations and experiments are often done with complex and expensive setups. In the beginning phase of these experiments, it can be beneficial to first test the ideology on simpler and cheaper setups. On the Eindhoven University of Technology (TU/e) there is such a simple experimental setup with small mobile robots (figure 1.1).



Figure 1.1: The experimental setup

The experimental setup allows small robots to perform movement patterns. These robots are recorded by a camera to observe their behavior.

Currently, this setup is running on an old computer with outdated software. This is not optimal for long time support with the rapid improvement of the software and technologies. The goal of this project is to make the setup also run on a newer computer. This paper takes a look at how the different aspects of the experimental setup are structured and in what way they can be transferred from one computer to the other.

## 2 Overview of the setup

The experimental setup mainly consists of three components: the computer, the camera, and 4 small mobile robots. The computer instructs the robots, the robots do their movement, the camera detects what is happening and gives back the information to the computer. The computer processes this information and can analyze this data. The computer uses Linux Ubuntu as its operating system.

### 2.1 The e-puck

The small robots that are used in this setup are the e-puck robots. The robot contains two step motors to control the wheels separately, and contains several sensors to check if there are objects nearby. The e-puck robot is a robot mainly used for educational purposes. The robot is made with the ideology to make a simple and cheap robot. With this robot, experiments can be tested before stepping up the experiment with bigger and more complex robots[4].



Figure 2.1: The e-puck robot

Figure 2.1 shows the first version of the e-puck released at the beginning of 2006. The developers of the e-puck released the second version of the e-puck, the e-puck2 in 2018[5]. To make it possible for the camera to detect where the e-pucks are located within the box, a marker is mounted on top of the e-puck (figure 2.2). The camera detects the shape of the marker and through software the computer can identify which e-puck is which.

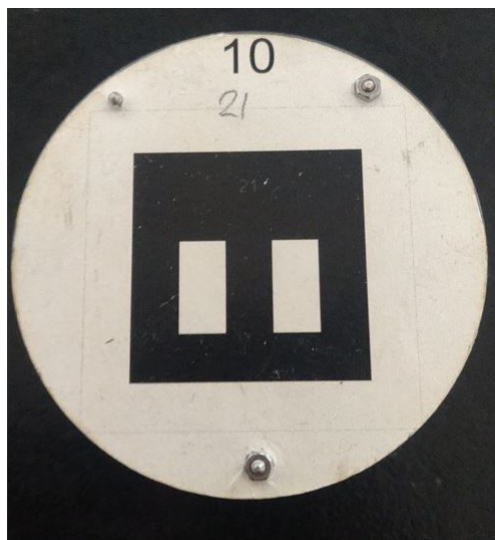


Figure 2.2: The marker on top of the e-puck

## 2.2 The camera

The camera used for this setup is the Allied Vision(AVT) Guppy F-080b, which is a camera that uses the 1394a firewire interface. The Guppy series of Allied Vision focuses on ultra-compact, small housing cameras. These are a perfect fit for all applications with space constraints. The camera was released at the beginning of 2007. In the meantime, Allied Vision released its successor in the Guppy PRO series[6].

### Guppy F-080

Interface: IEEE 1394a

Resolution: 0.8MP (1032 (H) × 778 (V))

Sensor model: Sony ICX204

Frame rate: 30 fps



Figure 2.3: The camera and its main specifications [7]

### 3 Problems with the old PC

At the beginning of the project, the experimental setup was located in the Gemini building. After a few weeks, the setup was moved to the Fenix building. After the setup was moved from the Gemini building to the Fenix building, there were some problems getting the setup working on the old computer. The first problem encountered was the failure to connect the old computer to the internet. The PC is old and was blocked by the ICT to the internet service of the TU/e. This leads to the problem that the system could not update any additional packages anymore. In addition, a license issue arose, as Matlab was not able to access the license through the internet. However, it was possible to get the Matlab working on the old PC via an offline Matlab license.

Unfortunately, this license was only a temporary solution. It worked the first time and the system was running and working with it. In an attempt to start Matlab for the second time, it was not able to initiate the program, as the interface window is automatically shut down after a short amount of time. Several attempts are made to fix this problem by refreshing the offline license, looking at the faillog (var/log/faillog/) and reinstalling Matlab, unfortunately without any success. Matlab 2010b was the Matlab version that did not start anymore. To get a working Matlab version, the 2011b version was installed. However, this did not get the setup working since certain mex files are in the Matlab script and are compiled on the 2010b version. A search for the source codes of these mex files did not directly lead to any results.

#### 3.0.1 Mex files

The mex files cannot be copied from one version of Matlab to the other. It has to be compiled for each Matlab version. At the beginning of the project, the original codes of these mex files were not found. Later on during the project, these files were discovered. Since at the time of discovery it was not the priority to get the mex files, it was not tested for another Matlab version. The files that were found are displayed in table 3.1:

Table 3.1: Mex files

Filename	Location
mexShared_matlab2010	/jurjen/src
mexGetRefPoints.new	/jurjen/src/newepuckAlt
mexGetPosesShared	/jurjen/src/newepuckAlt
mexShared	/jurjen/src/newepuckAlt
mexReacti	/jurjen/src/newepuckAlt

#### 3.1 Later information about the error

At the start of writing this paper, it was checked again if the source of this problem could be found. Matlab was searching for the Matlab 2010b startup file in the desktop folder. This file was not present. This can be because Matlab 2010b was reinstalled to see if that could fix the error. After trying to copy the startup file to the desktop folder, the main error was that the script came from an unknown source. After some trying, the error occurred that the disk of the computer was full. At this point, no attempt was made to resolve the issue due to the late nature of the findings.

## 4 Manual for making Bluetooth connection

This chapter provides a manual, which describes the steps necessary to establish the Bluetooth connection between the computer and the e-pucks. This is done for Linux as well as Windows.

### 4.1 Windows

Windows has fewer steps than Linux to establish a connection. All that is necessary is to select **Start > Settings > Devices > Bluetooth other devices > Add Bluetooth or other device > Bluetooth > e-puck\_XXXX**. At that moment a pin code is requested. This pin code corresponds to the XXXX of the e-puck. So for instance, if a connection is wanted with e-puck\_1963 the pin code is 1963. In most instances the e-pucks are arranged at "other devices" and it is stated that the computer is paired to the e-pucks but not connected, which is no problem.

After this is done, the Bluetooth connection can be established via the Matlab tool ePic2[8]. This is a tool developed by the makers of the e-puck and can be downloaded from their site[9]. For this tool, it is important that the Instrument Control Toolbox is installed on the Matlab that is used. The ePic2 tool can be started by selecting the ePic directory as the current Matlab directory and running "main" in the command window. The interface of figure 4.1 should pop up:

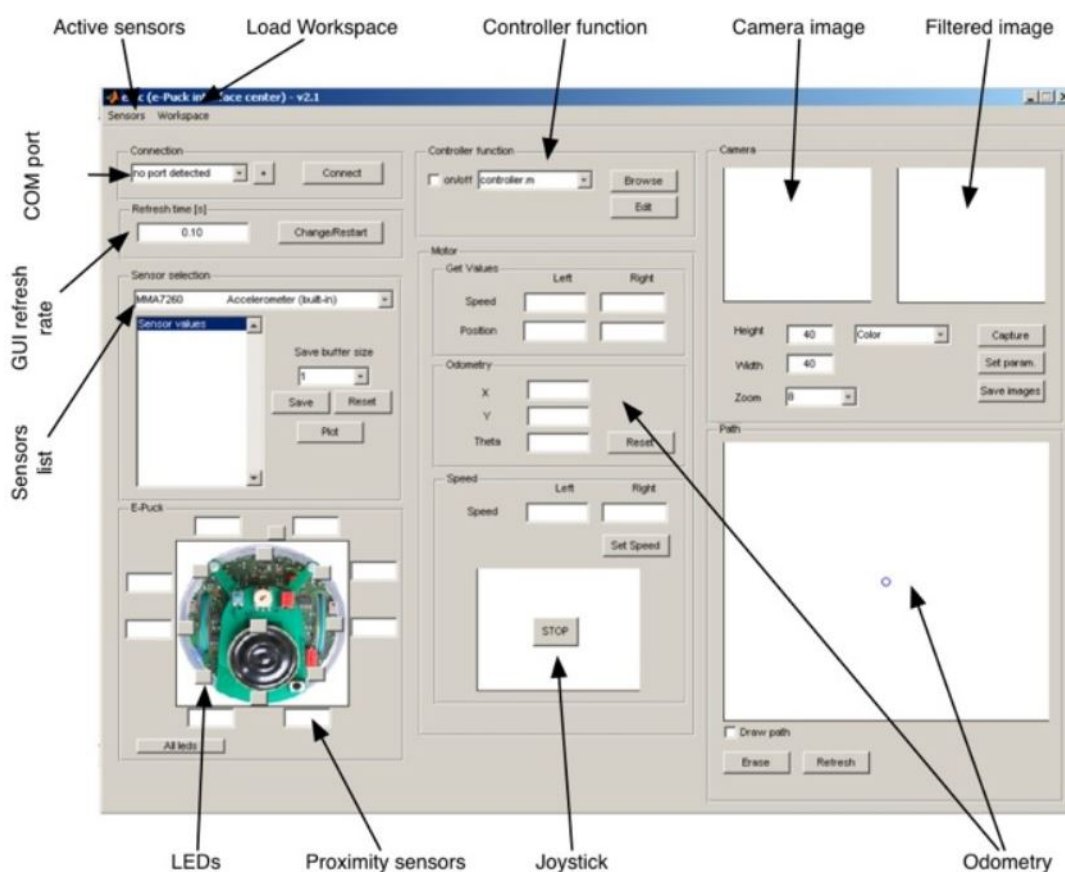


Figure 4.1: Interface ePic2 tool

When the ePic2 tool is started, the Bluetooth connection can be made. This is done by selecting the corresponding COM port of the e-puck and clicking on connect. The corresponding COM port of the e-puck can be found in **Control Panel > Hardware and Sound > View devices and printers > e-puck\_XXXX > Right mouse click > Properties > Services**. In the tab (figure 4.2) it can be found which COM port is used:



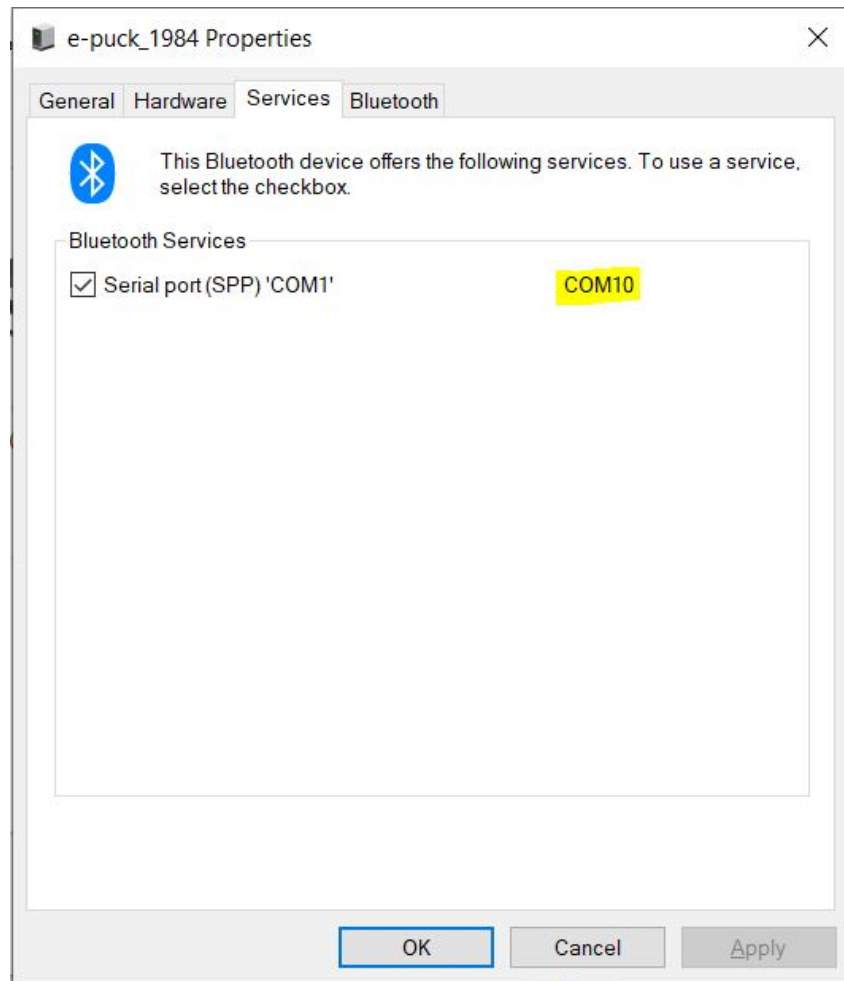


Figure 4.2: Identification of COM port

As the corresponding COM port is found for the e-puck, the e-puck can be connected to the computer by selecting the COM port in the top left and click on connect. Establishing this connection can take a while, especially the first time. When the e-puck is connected, the values of the sensors are visualized and the motor speed can be adjusted manually. This can also be done in the command window with coding. The code required to do this can be found in the open documentation of the ePic2 tool.

#### 4.1.1 Problems with Bluetooth on Windows

The open documentation states it is not possible to connect with multiple e-pucks at the same time, due to Matlab shortcomings. Nevertheless, a manual connection has been established to multiple robots manually. So, it is possible to connect with more than one e-puck manually, which is needed because the setup uses 4 robots. However, the open documentation does not advise to make a controller manually as a lot of information is required about the inner programming. Also with simple coding, it was not achieved to move multiple robots simultaneously. Therefore, Linux is considered to be the best option of the two as an operating system.

## 4.2 Linux

The Linux variant has a bit more steps to it. The following shells and executables need to be placed at the places that are showed in table 4.1:

Table 4.1: Files that need to be copied

File Name	Place
startEpuck	/usr/local/bin/
epuckProxy	/usr/local/bin/
startAllEpucks4bayu.sh	/usr/local/bin/
epuck_bdaddr	/etc/bluetooth/
rfcomm.conf	/etc/bluetooth/
bind_epuck.sh	/etc/bluetooth/

#### 4.2.1 Getting permission

The file epuckProxy is an executable, copying an executable can cause problems with the shared libraries among other things. For this executable, there are no complications that arise when it is copied, it is also possible to make this executable from the source code. The source code can be found in the map (/home/jurjen/src/newepuck/epuckProxy). When it is decided to copy and paste the executable, one change has to be made. The file has to be set as an executable.

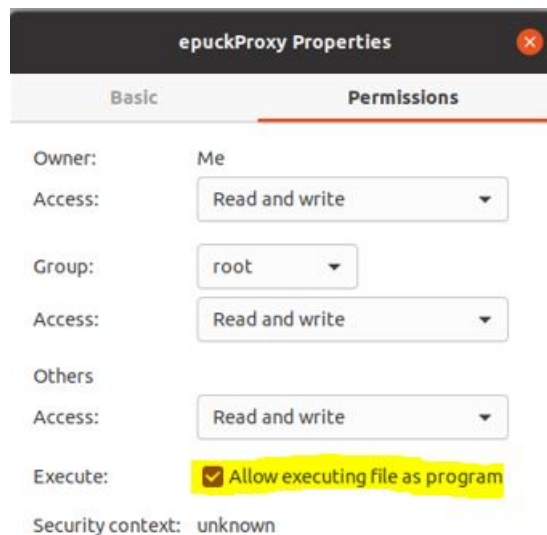


Figure 4.3: Permission should be selected

This can be done by changing the owner of the executable with the command:

```
sudo chown -R <username> epuckProxy
```

In which <username> is the name of the user that you want to have access to the file. After this, in the permissions page (figure 4.3), it has to be selected as an executable.

#### 4.2.2 Libraries for running the shells

To make the connection with the e-pucks, the shell "startAllEpucks4bayu.sh" should be run. In order to run the shell the following three libraries have to be installed:

- libSHARED.so.2
- libserial.so.1
- liblog.so.4

Again, these libraries can be copied from the old computer (which is not ideal, but does not give complications in this case) or from source code. When copy and paste are used, the original libraries should be copied and symbolic links to those libraries need to be made:

- libSHARED.so.2 → libSHARED.so.2.1.0
- libserial.so.1 → libserial.so.1.0.0
- liblog.so.4 → liblog.so.4.2.0

### 4.2.3 e-puck pairing

The e-pucks should be paired with the computer. This can be performed by taking the following steps:

1. Editing the rfcomm.conf file. With the copied files, (see table 4.1) it should already be done correctly. The rfcomm.conf file can be found in the (/etc/bluetooth) map and should have the layout of figure 4.4:

```
rfcomm0 {
bind yes;
device 08:00:17:2C:E0:88;
channel 1;
comment "e-puck_0006";
}
```

Figure 4.4: Rfcomm.conf file layout

The device MAC address can be found in the bluetooth settings by clicking on the device information. There is also a document where all the MAC addresses of the e-pucks are stated. This is the "epuck\_bdaddr" file (figure 4.5) that can be found in (/etc/bluetooth):

Line	MAC Address	e-puck ID
1	10:00:E8:52:B4:BB	e-puck_1404
2	10:00:E8:6C:A2:E5	e-puck_1841
3	10:00:E8:6C:A1:EA	e-puck_1910
4	10:00:E8:6C:A1:E4	e-puck_1914
5	10:00:E8:6C:A1:C1	e-puck_1925
6	10:00:E8:6C:A1:A3	e-puck_1929
7	10:00:E8:6C:A2:D9	e-puck_1933
8	10:00:E8:6C:A1:E6	e-puck_1935
9	10:00:E8:6C:A1:F8	e-puck_1942
10	10:00:E8:6C:A2:00	e-puck_1963
11	10:00:E8:6C:A2:FB	e-puck_1971
12	10:00:E8:6C:A2:C1	e-puck_1981
13	10:00:E8:6C:A1:DA	e-puck_1984
14		
15		
16	10:00:E8:52:B4:C2	e-puck_1314
17	10:00:E8:52:B4:BB	e-puck_1404
18	10:00:E8:52:CA:CC	e-puck_1309
19	10:00:E8:6C:D7:B8	e-puck_1592

Figure 4.5: All the MAC-addresses of the e-pucks

2. The MAC address should be connected with their corresponding rfcomm port. This is done with the following command:

```
sudo rfcomm bind 0 10:00:E8:6C:A1:F8 1
```

In which the 0 before the MAC address should be the same number as the rfcomm port that was in the structure of rfcomm.conf. (figure 4.4). The 1 at the end is for which channel the device uses. This should be kept on channel 1. With this command, a (/dev/rfcommXX) file should be made automatically.

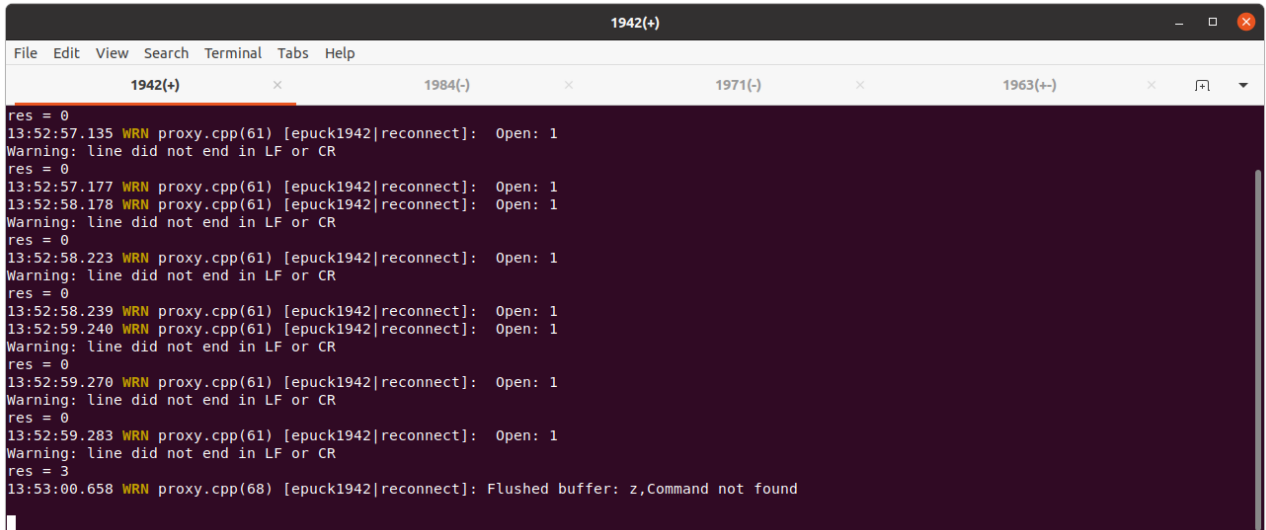
3. The program "epuckProxy" searches for a (/opt/epuckXXX) link to connect with the e-puck. This link should be made manually and should link to the (/dev/rfcomm) file that was made in the previous step. The link can be made with the following command:

```
sudo ln -s /opt/epuckXXX /dev/rfcommXX
```

- An e-puck bootloader needs to be installed. This bootloader can be found at: <https://www.gctronic.com/doc/index.php/E-Puck> at the section bootloader. In addition, a package should be installed. This is done by the command:

```
sudo apt-get install libbluetooth-dev
```

After this, all the settings are set correctly. The e-pucks can now be connected by running the "startAllEpucks4bayu.sh" file that can be found in (/usr/local/bin). The screen of figure 4.6 should pop up:



```
1942(+)
```

```
File Edit View Search Terminal Tabs Help
```

```
1942(+)
```

```
res = 0
```

```
13:52:57.135 WRN proxy.cpp(61) [epuck1942|reconnect]: Open: 1
```

```
Warning: line did not end in LF or CR
```

```
res = 0
```

```
13:52:57.177 WRN proxy.cpp(61) [epuck1942|reconnect]: Open: 1
```

```
13:52:58.178 WRN proxy.cpp(61) [epuck1942|reconnect]: Open: 1
```

```
Warning: line did not end in LF or CR
```

```
res = 0
```

```
13:52:58.223 WRN proxy.cpp(61) [epuck1942|reconnect]: Open: 1
```

```
Warning: line did not end in LF or CR
```

```
res = 0
```

```
13:52:58.239 WRN proxy.cpp(61) [epuck1942|reconnect]: Open: 1
```

```
13:52:59.240 WRN proxy.cpp(61) [epuck1942|reconnect]: Open: 1
```

```
Warning: line did not end in LF or CR
```

```
res = 0
```

```
13:52:59.270 WRN proxy.cpp(61) [epuck1942|reconnect]: Open: 1
```

```
Warning: line did not end in LF or CR
```

```
res = 0
```

```
13:52:59.283 WRN proxy.cpp(61) [epuck1942|reconnect]: Open: 1
```

```
Warning: line did not end in LF or CR
```

```
res = 3
```

```
13:53:00.658 WRN proxy.cpp(68) [epuck1942|reconnect]: Flushed buffer: z,Command not found
```

Figure 4.6: e-puck connection window

The + at the top indicates that the e-puck is connected, the +- indicates it is currently connecting, and the - sign indicates it is not connected and detected. If the e-puck is connected it shows an orange light as shown in figure 4.7:

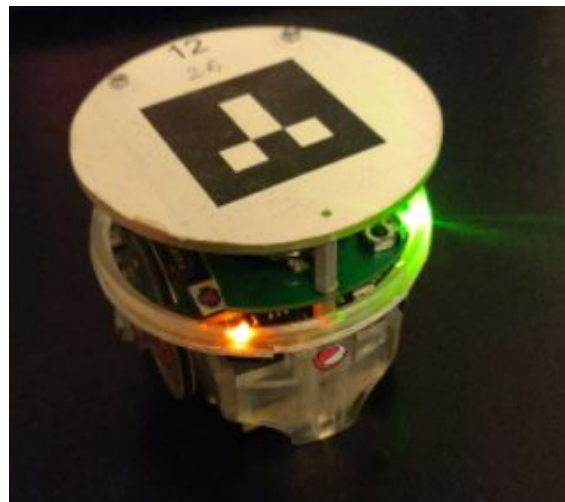


Figure 4.7: Orange light that the e-puck is connected

## 5 Setting up the camera and its problems

Currently, the setup uses the AVT Guppy camera (see figure 2.3). It is not achieved to make this camera and its additional marker recognition work on the new computer. These problems are discussed in this section.

### 5.1 Setting up the files

The files required to make the camera work were copied to the computer. Some of these files are executables. It was tried to remake the executables from the source code to avoid any further complications that come from copying, but the source code was not found. The important files are displayed in table 5.1:

Table 5.1: Files for the camera

Filename	Location
startCameras	/usr/local/bin
setShutter.sh	/opt
CameraProxy	/usr/local/bin
cameraProxy	/usr/local/bin
Stream1	/usr/local/bin
Stream2	/usr/local/bin
startCamPose	/usr/local/bin
startCamPose1	/usr/local/bin
startCamPose2	/usr/local/bin
Markerpose	/etc
firepackage installer	<a href="https://www.alliedvision.com/en/support/software-downloads/">https://www.alliedvision.com/en/support/software-downloads/</a>
Camera_pose_ID_reactivation	jurjen/src/positioning
Camera_pose_ID_reactivation_SHARED	jurjen/src/positioning

The last two items are folders, it is advised to copy the entire folder since it contains multiple documents which are needed and they are already structured in the right position. The symbolic links that connect these files should be added as well.

#### 5.1.1 Structure of the files

The camera needs to be activated, which can be done by running the shell "startCameras". After this is done successfully, the camera needs to be processed. This can be done by running "startCamPose". This is also the step where it goes wrong at the moment. For the shell "startCamPose", there are a few programs run from the Camera\_pose\_ID\_reactivation\_SHARED folder. These files give problems with their shared libraries. Especially the library libdc1394. This library helps with support for the IEEE1394 camera type. If the library is copied from the old computer, it displays the message that there is no device list found. Installing the library from the source code does not seem to help either since the library is not added to the path, even though the folder where the library is located is added to the library path.

### 5.2 The camera under Windows & Linux

The camera under the Windows operating system works fine. After installing the firepackage, the camera can be monitored. In Linux, it is also possible to install the firepackage. The application "wine windows loader" needs to be installed in Linux[10], before it is possible to run such a .exe file. After the firepackage is installed in Linux, it shows that there is no 1394 card available even though the computer recognizes a 1394 card is mounted to the computer.

### 5.3 Missing required modules

It was found that there is a Linux program that directly monitors a 1394 camera. This program is called coriander 2. Some underlying software of this coriander program is supported till Ubuntu 18.04, this means that Ubuntu 20.04, which is currently used on the new computer, is not supported. Therefore,

Ubuntu 18.04 was also installed on the computer. After trying to run coriander on Ubuntu 18.04 the error of figure 5.1 showed:

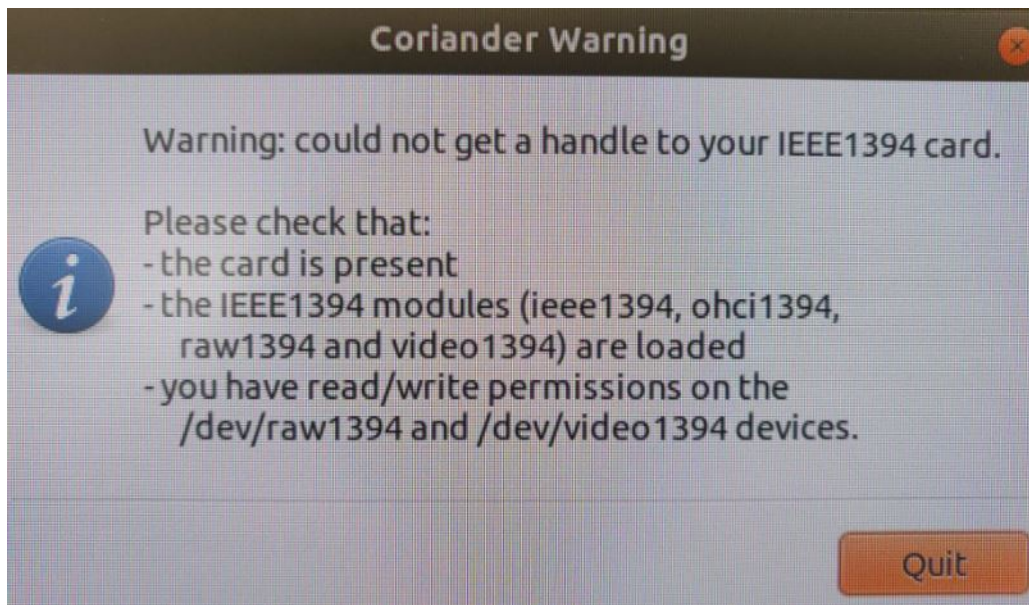


Figure 5.1: Error for coriander program

It was found that the problem comes from the modules. The coriander and the guppy camera use the classic IEEE 1394 modules. These modules were updated halfway through 2008 to the newer drive stack Juju which now provides the 1394 kernel support[11]. On the old computer, this Juju stack was blacklisted, which means that they are deactivated, and the classic IEEE modules were used. The same strategy was tried on the new computer. Unfortunately, it failed to install the classic modules of the 1394 support on the new computer. This software is old and hard to find. An attempt was made to install an old Ubuntu version (Ubuntu 09.04 or older) that would contain a kernel that would have these modules. It turned out that the versions were so old that it was not automatically supported by the computer. Even tweaking the settings in the BIOS menu to simulate an old computer did not result in the possibility to install an old version of Ubuntu. Via a CD-rom it is possible to install an older version of Ubuntu. The installation with the CD-rom was not tried anymore, only the simulation of an old computer. Installing an older version of Ubuntu is not the best idea regarding safety issues and such, but it was tried to check if the camera would work.

## 5.4 Marker detection

Since the camera is not working, the part of marker identification is not yet tested. It was checked which files are required for the marker identification. Most of the files for the marker identification are already in the files required for the camera. The additional files are displayed in table 5.2:

Table 5.2: Files for marker identification

Filename	Location
markerID	/opt
marker_pos	/usr/local/bin
markers.d	/jurjen/src/positioning/campos_SHARED_orig
markers.o	/jurjen/src/positioning/campos_SHARED_orig

## 6 Big problems during the project

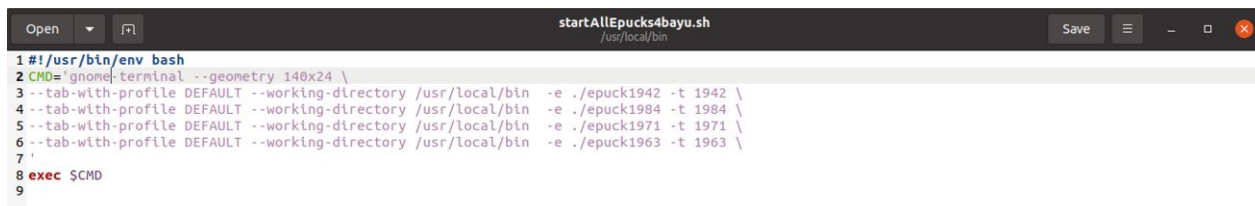
As it can be seen, there are not many results from the project. There were some big problems encountered during the project that took a while to solve. Some were solved others not. If this project has a follow-up, chances are that these problems are encountered. Therefore, the issues are listed below, along with the steps that were attempted to resolve them and whether they worked.

### 6.1 No Profile with UUID:

```
Felixl@felixl-HP-EliteDesk-800-G6-Tower-PC:/usr/local/bin$ ./startCameras
# Failed to parse arguments: No profile with UUID or name "DEFAULT" exists
```

Figure 6.1: No profile with UUID error

The error (figure 6.1) is not common among Ubuntu users and therefore there is not much information available on the internet about this problem. It has a good reason for it, the problem of the error is not about wrong settings in the Ubuntu system but in the layout of the shell that is run. All these shells have a layout similar to figure 6.2:



```
Open startAllEpucks4bayu.sh /usr/local/bin Save
1 #!/usr/bin/env bash
2 CMD="gnome-terminal --geometry 140x24 \
3 --tab-with-profile DEFAULT --working-directory /usr/local/bin -e ./epuck1942 -t 1942 \
4 --tab-with-profile DEFAULT --working-directory /usr/local/bin -e ./epuck1984 -t 1984 \
5 --tab-with-profile DEFAULT --working-directory /usr/local/bin -e ./epuck1971 -t 1971 \
6 --tab-with-profile DEFAULT --working-directory /usr/local/bin -e ./epuck1963 -t 1963 \
7 '
8 exec $CMD
9
```

Figure 6.2: Wrong shell layout

The problem is with the first part of lines 3 till 6. The "--tab-with-profile DEFAULT" part makes the computer search for a profile with the name "DEFAULT". This profile cannot be found on the old and new computer. On the old computer the terminal shows:

```
No such profile "DEFAULT", using default profile
```

This default profile that is used can be made in the terminal settings. However, after doing this on the new computer the error still occurs. There are two solutions to solve this problem. The first solution is to change the layout of the shell, this is done by modifying "--tab-with-profile DEFAULT" to "--tab". The second solution is to change the use of the gnome-terminal (line 2) to the use of the mate terminal, so modify "gnome-terminal" to "mate-terminal". Therefore, the mate-terminal needs to be installed. From experience, the second solution gives some troubleshooting with certain cells so the easiest is to use solution 1.

### 6.2 Missing libraries

Another big error during this project was the missing shared libraries. These libraries are needed as support to run shells. These libraries can be installed from source code or copied from the old computer. Copying the libraries is not advised as copying can lead to the shell not getting the right information from the library.

However, with installing the libraries from source code there was also some trouble. If a library was installed from source code, it was often not added to the library path. This means that the shell does not recognize that the library is installed. The two main solutions to this problem are adding the directory of the library to the (/etc/ld.so.conf) file and adding it to \$LD\_LIBRARY\_PATH. Most of the time it did not work, even though it was recognized that the directory was added to \$LD\_LIBRARY\_PATH. Moving the library is not an option, the libraries are installed in a directory for a reason and if they are moved, the Linux system crashes. In general, there is still no solution for the library problem on the new computer.

### **6.3 Arrival delay of the new computer**

It took one and a half months for the computer to be delivered. This was not much of a problem, since getting to learn Linux was the priority at the beginning. However, from experience, it was found that learning Linux went much better with the new computer since the underlying links and software came to light with errors. This was not the case on the old computer since the links were already set correctly and needed to be searched for, which led to overlooking links easily. With the delay of the computer came also the problem that one of the project coordinators was away for some time. This again was not a big issue, but it was a bit more difficult to communicate about the old PC errors and the delivery of the new computer.



## 7 Recommendations

As can be concluded from this paper, the first steps towards a working setup have been set. However, it is not yet achieved to get the entire setup working. If there were to be a follow-up of this project, some recommendations are given.

### 7.1 The use of a newer camera

As discussed in the Camera problem section 5.3, the camera is outdated and requires old software to run. The simple solution to solve this problem is to get a newer camera that is able to run with newer software. The main problem is that there needs to be written an exclusive script again for the marker identification on the new camera. However, this can be circumvented with a certain camera, which is discussed in section 7.2.

### 7.2 The use of Robot Operating System

A second option is to use the Robot Operating System (ROS). ROS is a free and open-source software. ROS helps to easily connect different components of the system. These components do not need to be on the same hardware or operating system[12]. ROS makes it possible to connect the image of a camera working on Linux with the control system which runs on Windows. Since, in this case, the camera works fine on Windows, the Bluetooth connection and marker identification works with Linux, it can be combined with the use of ROS to make the system run. ROS supports the use of a 1394 camera. However, since the camera is old and has caused multiple unexpected problems in the past, it is a bold statement to assume that it will work using this method.

ROS comes with a big disadvantage, there needs to be additional hardware, like a personal laptop, with a running operating system to be able to make ROS work. It is also possible to have the old computer as the second hardware, then the camera also works on Linux but the internet issue that is discussed in chapter 2 should be fixed. Additionally, it is not ideal with long time support in mind, since the software is old.

#### 7.2.1 ROS for complex systems

As described in the introduction, this experimental setup is a step up for more complex and bigger simulations. One of these bigger more complex setups is also present at the TU/e. This setup is a simulation about trucks and a distribution center. This system uses ROS and one of the PhD'ers working on the setup advised to use ROS for the e-puck setup. The layout of the ROS system for the truck setup can be seen in figure 7.1:

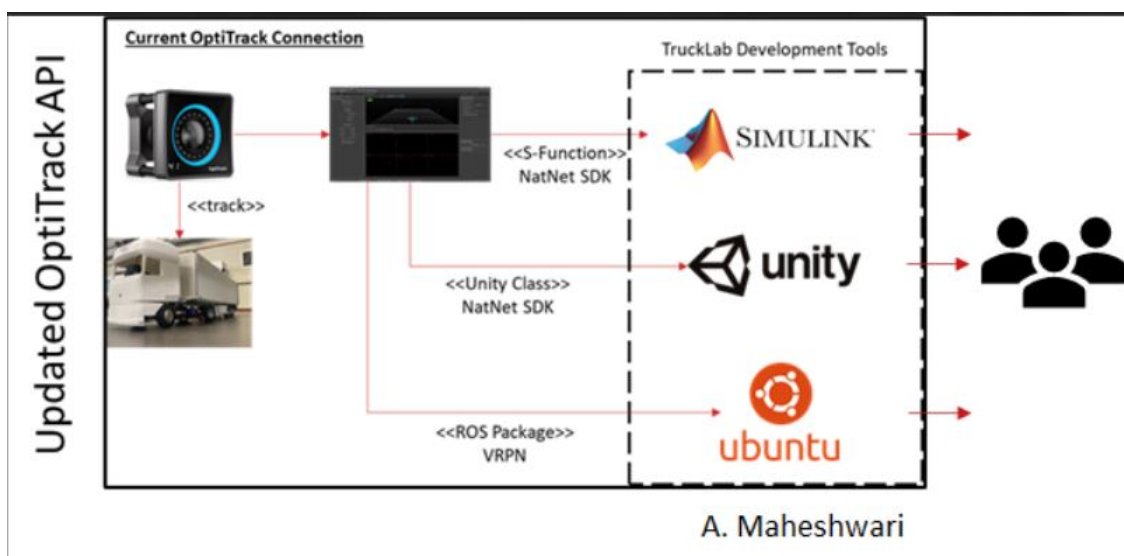


Figure 7.1: ROS setup

For the e-puck setup, the truck is replaced with the e-puck and only the route to Ubuntu and Matlab is taken. In this example, Matlab is substituted by Simulink. As the Matlab controller is already written for the e-puck setup, the Simulink model for the e-pucks can be easily made.

### 7.2.2 New camera and software for ROS

If the setup is changed to the ROS system, a few things need to be changed. Changing the camera is one of these things, the new recommended camera is an Optitrack camera. This camera is especially specialized in optical motion capturing[13]. Optitrack cameras are also used for the truck setup. Pricewise, these cameras are at the higher end of the spectrum. Although, there is one camera that is made with affordability in mind. This camera is the flex 3 camera and can be seen in figure 7.2:



Figure 7.2: The flex 3 camera[14]

The camera comes with knots which can be put on the e-pucks. The software can detect the movement of these knots and therefore the e-puck. These knots are alternatives for the markers, which would help in case the marker identification script does not work with the new camera. The only problem with the motion detectors is that they cannot distinguish the e-pucks from each other like the marker identification could with the different shapes.

For the movement knots a new software package should be installed. The software used to detect the movement knots is NatNet SDK. This is the same software used in the truck setup and works extremely well with the ROS system and Matlab.

### 7.2.3 Long time support

The ROS system is recommended in particular with long time support in mind. Most setups are currently converted to the ROS system and it gives a lot of perspective for the future. In addition, it is convenient to have the same operating system for the e-puck and truck setups. That way someone can make a quick switch between the two setups. The PhD'er that recommended ROS also suggested that he could install the setup within a few days since he already has a lot of experience with the system. This would save a lot of time for continuing with this project.

## 8 Reflection

The project did not have the trajectory I predicted beforehand. What I expected beforehand was that getting the setup working would be a lot more convenient and less complex than it eventually turned out to be. The time predicted to get it done was around 1.5 months, this was also due to my lack of computer knowledge. I already struggled with the underlying Windows software and having to learn a brand new operating system that was far more focused on code language was difficult. I noticed during the project that I would go for the first solution that I came across more often than not. Taking the shortcuts without knowing what I was doing and not understanding the structure of certain steps came back to haunt me at the end phase of the project.

### 8.1 Lessons learned for the work field

- Not many things go as planned. During such a project many new, underlying and unexpected problems arise. The planning already took into account a margin of error for these kind of situations, however, not enough time was allocated. The lesson learned is to still incorporate a margin of error but to adjust the margin of error to the level of expertise.
- Another lesson learned regards the progression of the project. During the DBL's followed in the bachelor course, there were also problems but these were constantly quite small, and often another group member came up with a solution. This project showed another side of problem solving. If the project goes into a niche direction it is harder to find solutions provided by others. It is mentally very draining to be completely stuck on an error for several days to find a simple step as the solution at the end. After the solution was found a new error would come up in no time, as a result I was constantly moving from being stuck on an error for quite some time to being stuck on the next one for even longer. This was mentally very draining and not rewarding. I was constantly trying to find a new angle of view to find a solution, this I did not experience to this extend before. Looking back at it, it was a good exercise to train the mental aspect of such a project.
- The next lesson is about when and how to ask questions to others. Especially, during the middle fase of the project, I would ask a question when I was struggling with an error for quite a while. However, these questions did not elaborate on which ways I already tried to solve it. Most of the time the responses were solutions that I already had taken a look at. I would get frustrated with the fact that the solutions they gave me were the most basic ones and blame it on the respondents for not thinking that I did not try anything before asking the question. Later the insight came that the problem was entire on me and the way of asking questions. I was frustrated with the fact that an error could not be solved and just typed a quick question. This question did not include all the solutions that I had already tried. Therefore, the respondents could not know what I did beforehand and recommended the first step. I tried to work on formulating a question and showing what already was investigated. Taking a bit more time to formulate the question saves a lot more time for the one questioned and gives a much more valuable response.

## References

- [1] Y. Zhuang, Y. Zhou, Y. Yuan, X. Hu, and E. Hassini, “Order picking optimization with rack-moving mobile robots and multiple workstations,” *European Journal of Operational Research*, vol. 300, no. 2, pp. 527–544, 7 2022.
- [2] M. Y. Chen, Y. J. Wu, and H. He, “A novel navigation system for an autonomous mobile robot in an uncertain environment,” *Robotica*, vol. 40, no. 3, pp. 421–446, 3 2022.
- [3] M. M. Almasri, A. M. Alajlan, and K. M. Elleithy, “Trajectory Planning and Collision Avoidance Algorithm for Mobile Robotics System,” *IEEE Sensors Journal*, vol. 16, no. 12, pp. 5021–5028, 6 2016.
- [4] J. M. Allen and R. Joyce, “Swarm Intelligence: 12th International Conference, ANTS 2020, Barcelona ... - Google Boeken,” pp. 243–255, 2020. [Online]. Available: [https://books.google.nl/books?id=Oa8EEAAAQBAJ&pg=PA243&lpq=PA243&dq=epuck+released&source=bl&ots=-p4uqCgQ2w&sig=ACfU3U3x\\_LkSOhpnrA66sEixMbBizbKoLw&hl=nl&sa=X&ved=2ahUKEw3NDn5ZP3AhXoiv0HHUYODVYQ6AF6BAgUEAM#v=onepage&q=epuck%5Creleased&f=false](https://books.google.nl/books?id=Oa8EEAAAQBAJ&pg=PA243&lpq=PA243&dq=epuck+released&source=bl&ots=-p4uqCgQ2w&sig=ACfU3U3x_LkSOhpnrA66sEixMbBizbKoLw&hl=nl&sa=X&ved=2ahUKEw3NDn5ZP3AhXoiv0HHUYODVYQ6AF6BAgUEAM#v=onepage&q=epuck%5Creleased&f=false)
- [5] Administrator, “Project,” 2010. [Online]. Available: [http://www.e-puck.org/index.php?option=com\\_content&view=article&id=6&Itemid=3](http://www.e-puck.org/index.php?option=com_content&view=article&id=6&Itemid=3)
- [6] V. Allied, “Document and software downloads for Allied Vision Guppy cameras - Allied Vision.” [Online]. Available: <https://www.alliedvision.com/en/support/technical-documentation/guppy-documentation/>
- [7] —, “Guppy F-080 FireWire camera - very compact - attractive price - Sony ICX204 - Allied Vision.” [Online]. Available: <https://www.alliedvision.com/en/camera-selector/detail/guppy/f-080/>
- [8] Administrator, “Matlab Tools,” 2012. [Online]. Available: [http://www.e-puck.org/index.php?option=com\\_content&view=article&id=29&Itemid=27/](http://www.e-puck.org/index.php?option=com_content&view=article&id=29&Itemid=27/)
- [9] —, “Download,” 2010. [Online]. Available: [http://www.e-puck.org/index.php?option=com\\_phocadownload&view=category&id=12:programs&Itemid=38](http://www.e-puck.org/index.php?option=com_phocadownload&view=category&id=12:programs&Itemid=38)
- [10] J. Wallen, “How to Install and Use Wine to Run Windows Applications on Linux - Linux.com,” 2015. [Online]. Available: <https://www.linux.com/training-tutorials/how-install-and-use-wine-run-windows-applications-linux/>
- [11] “Introduction - IEEE 1394 FireWire Wiki,” 2012. [Online]. Available: [https://ieee1394.wiki.kernel.org/index.php/Introduction#IEEE\\_1394\\_support\\_in\\_old\\_kernels](https://ieee1394.wiki.kernel.org/index.php/Introduction#IEEE_1394_support_in_old_kernels)
- [12] K. Scott, “ROS: Home,” 2021. [Online]. Available: <https://www.ros.org/>
- [13] Optitrack, “OptiTrack - Motion Capture for Movement Sciences,” 2022. [Online]. Available: <https://optitrack.com/applications/movement-sciences/>
- [14] —, “OptiTrack - Flex 3 - An affordable motion capture camera,” 2022. [Online]. Available: <https://optitrack.com/cameras/flex-3/>