Université catholique de Louvain – Ecole Polytechnique de Louvain

LELME2732: Robotics —2023 edition
Install instructions (Docker version)

Louis Devillez, François Heremans and Renaud Ronsse

This documents shows how to use the Docker version of the LELME2732 APP simulation environment. Docker is a container platform that allows an application and its required environment to be completely independent from the executing machine. This allows your application to be crosscompatible and to be quickly deployed to new machines. The LELME2732 simulator is packed as a lightweight Linux application including all dependencies and only requires a connection to your screen in order to render the realtime animation. This document goes through the few steps needed to setup the project on your computer.

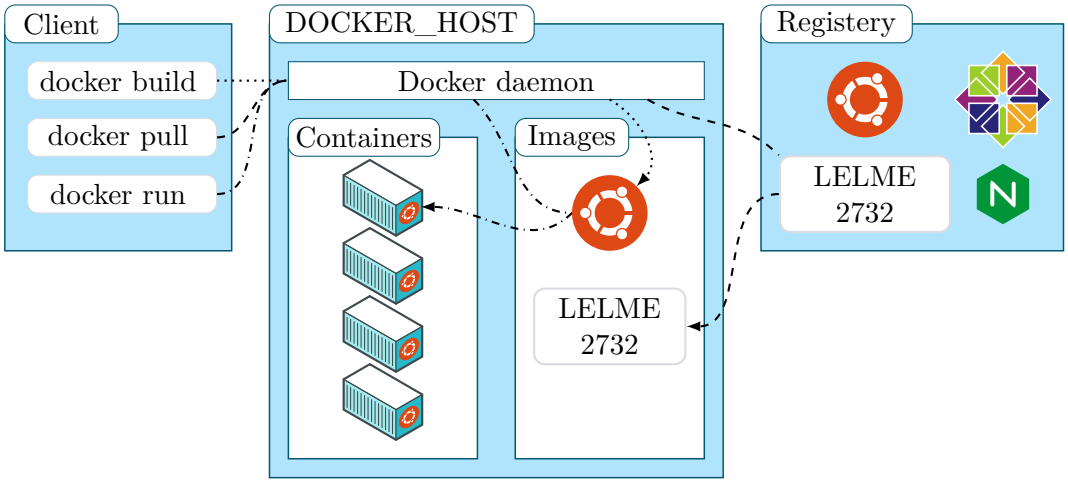


Fig. 1: Docker architecture [`docs.docker.com`]

As showned in Figure1, the simulator is bundled as an *Image* hosted in the *Registry* from where it can be downloaded to a host machine using a "*pull*" command. This *Image* acts as a template and can be used to produce *Containers* which are instances of an *Image*. A "*run*" command will check if the corresponding *Image* is available locally, if not it will be downloaded from the *Registry* and then run as a *Container*.

Contents

1 Installation

1.1 Getting Docker (Linux)

First, make sure your environment contains all required packages and keys. Open a terminal and type:

```
$ sudo apt-get install apt-transport-https ca-certificates curl gnupg-agent
software-properties-common git
```

```
$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
```

```
$ sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu
$(lsb_release -cs) stable"
```

Then install Docker

```
$ sudo apt-get update && sudo apt-get install docker-ce docker-ce-cli containerd.io
```

Check that the installation was successufl by running the hello-world example. The image will be automatically downloaded and the container will print “hello world”

```
$ sudo docker run hello-world
```

1.2 Getting Docker (MacOS)

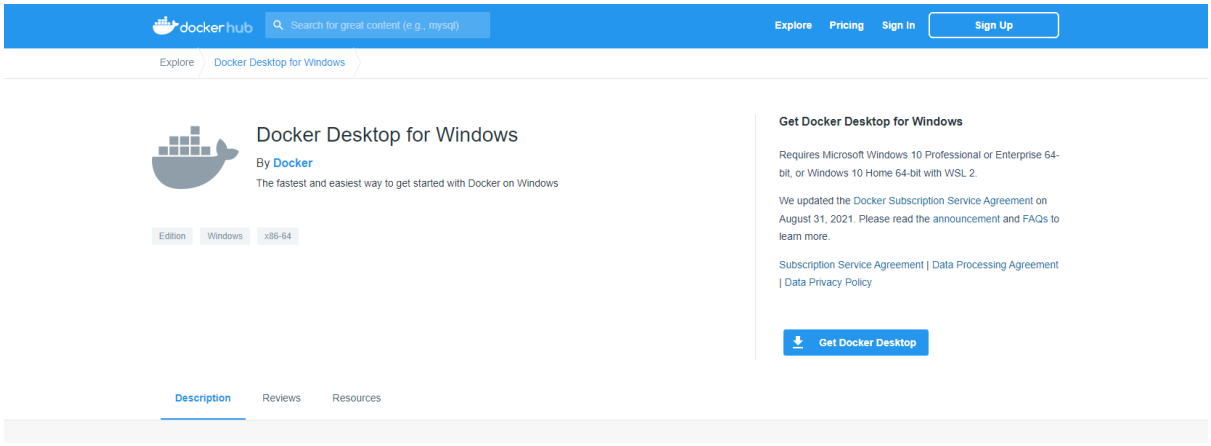
Start by downloading and installing Docker. Then install Git for Mac.

Check that the installation was successful by running the hello-world example. The image will be automatically downloaded and the container will print "hello world". Open a terminal and type:

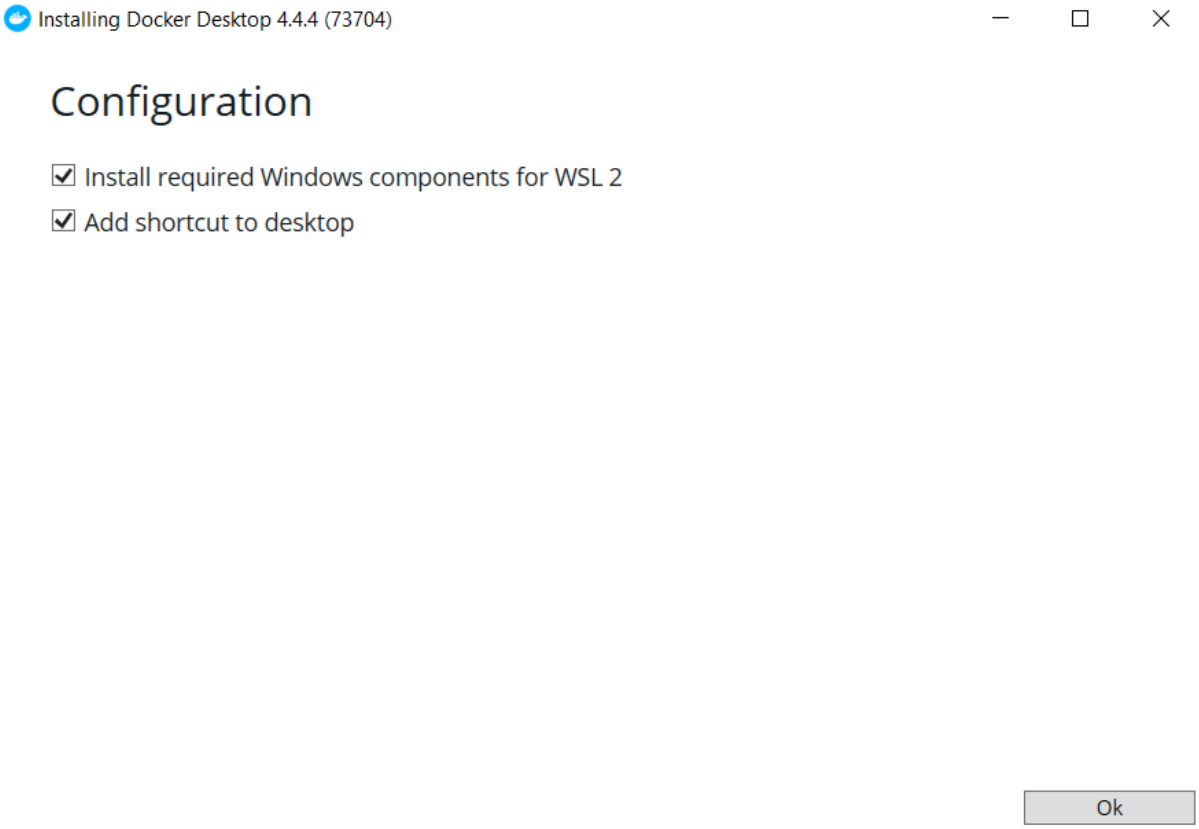
```
$ sudo docker run hello-world
```

1.3 Getting Docker (Windows)

1. Download Docker Desktop for Windows from the official website

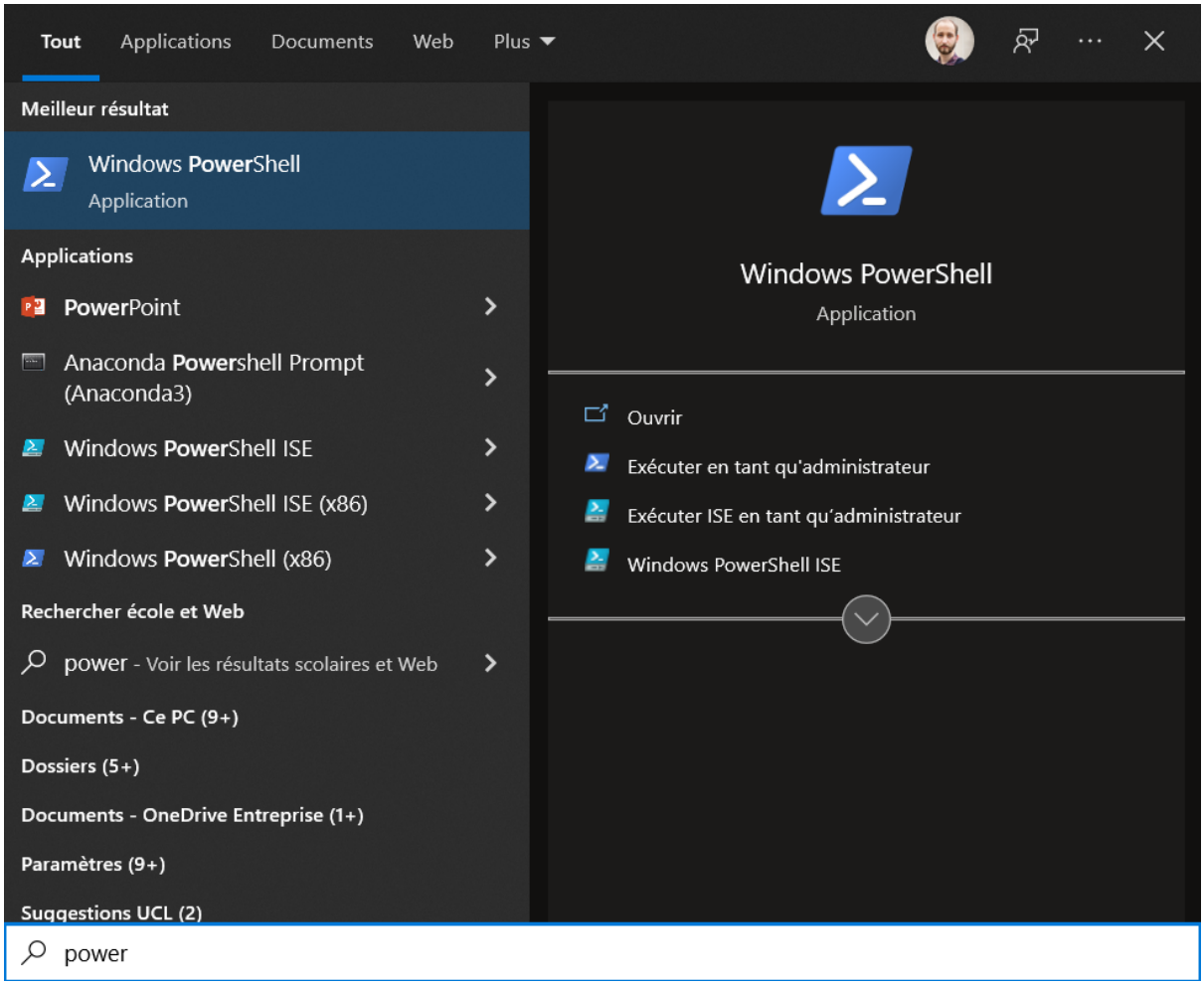


2. Install docker on your computer



3. Install Windows Subsystem for Linux (WSL) 2 following the manual install procedure.

   (a) Start a Powershell as administrator



   (b) Enable WSL

```
$ dism.exe /online /enable-feature /featurename:Microsoft-Windows-Subsystem-Linux
/all /norestart
```

   (c) Enable the virtual machine feature

```
$ dism.exe /online /enable-feature /featurename:VirtualMachinePlatform /all
/norestart
```

   (d) Restart your computer and then reopen a powershell

   (e) Download and install the Linux kernel update package.

   (f) Set WSL 2 as your default version

```
$ wsl --set-default-version 2
```

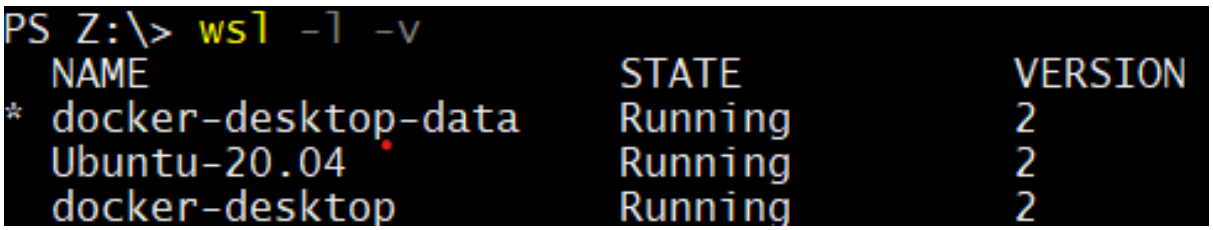
(g) Install the Ubuntu distro

```
$ wsl --install -d Ubuntu-20.04
```

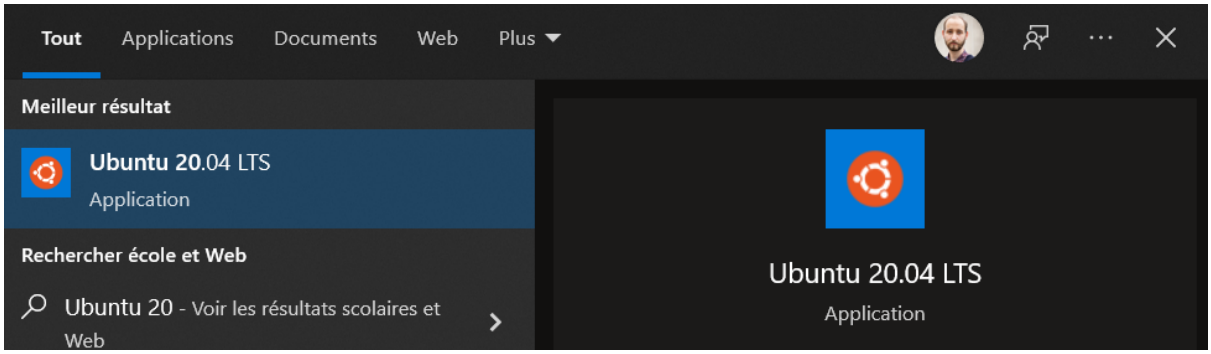
(h) A new window should have opened. Wait the end of the installation and then enter a UNIX username and a password (you can choose whatever you want but do not forget them).

(i) Check if everything is set correctly in the powershell. You should see your new ubuntu distribution.

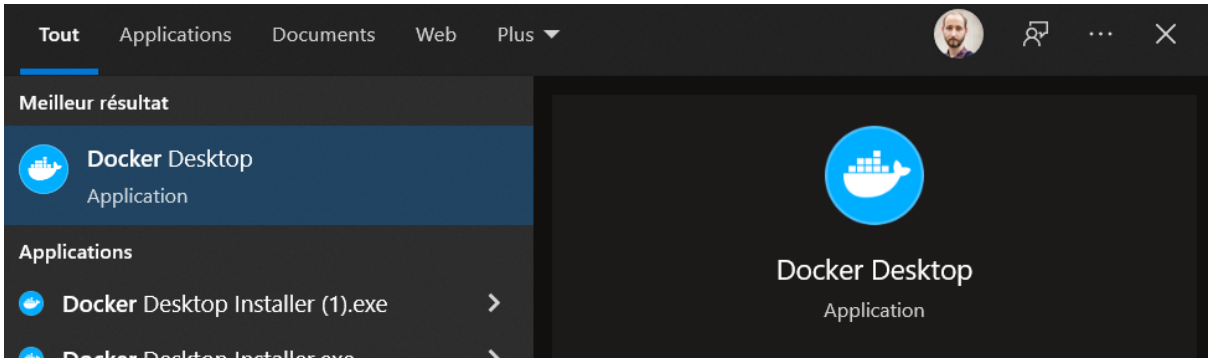
```
$ wsl -l -v
```



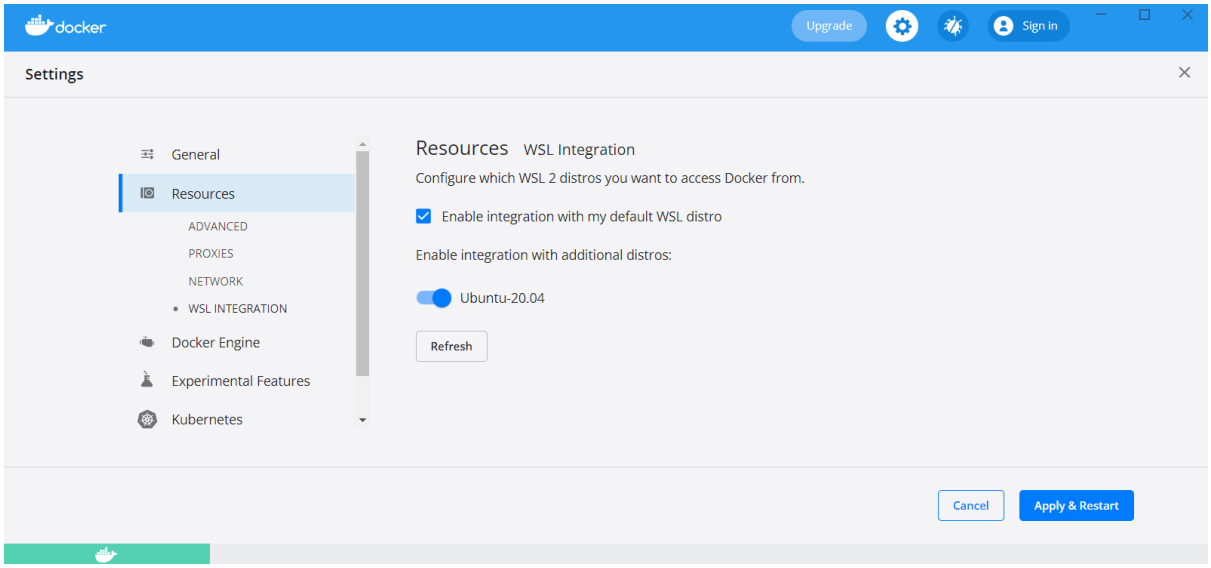
4. Launch your ubuntu distribution (if it is not already the case)



5. Start docker (you will maybe need to add your users to the [docker-users](https://icij.gitbook.io/datashare errors/you-are-not-allowed-to-use-docker-you-must-be-in-the-docker-users-group-.-what-should-i-do))



6. Update Docker settings to use the new distribution, click Apply & Restart



2 Deploying the container

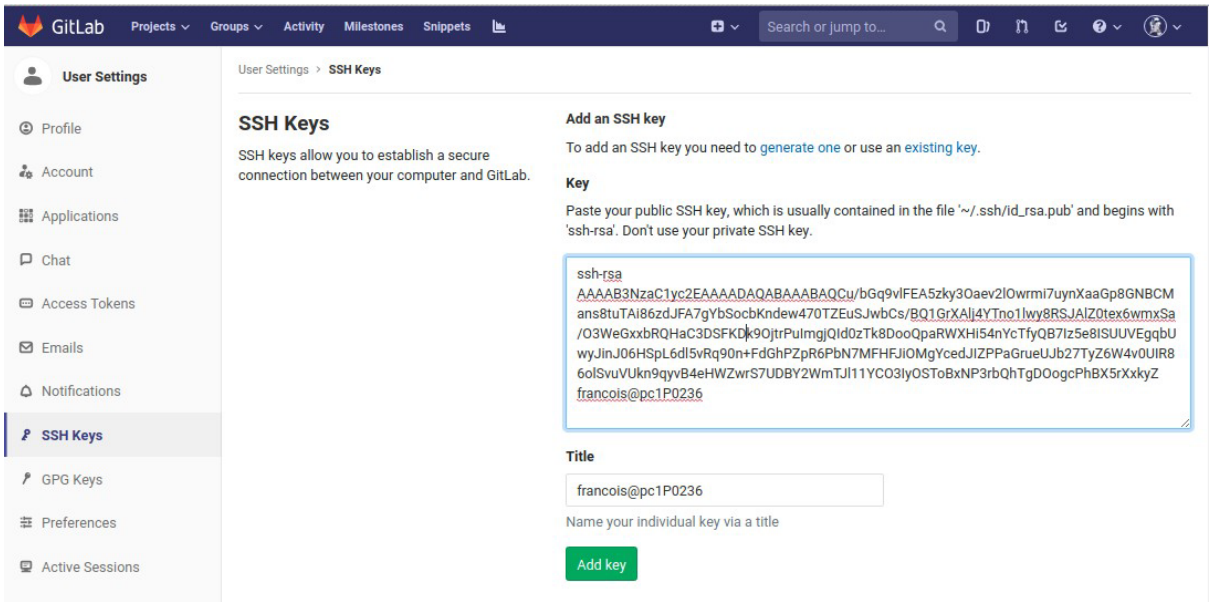
The first part consists in generating a secure ssh key to communication with the Git server where the simulator code is hosted. Type the following command (for windows it is in the ubuntu terminal) then the Enter key until the terminal returns.

```
$ ssh-keygen
```

Then, show your public key and copy it

```
$ cat ~/.ssh/id_rsa.pub
```

Paste the text and create a key in the setting of your account on gitlab



Then you will be able to clone the project belonging to your group. Don't forget to replace the X with your group number. If you are indeed a member of that group, the download will

proceed. The location of the project is not important, you can clone it wherever you want. Just avoid any space or special character in the path and remain below the user's root (`/home/you`, `/c/Users/you`).

```
$ git clone git@git.immc.ucl.ac.be:LELME2732.GrX/eurobot_2023_students.git && cd
eurobot_2023_students
```

You will now be able to get your copy of the project's Docker container. The first time you run the following command, the whole image has to be downloaded which will take some time. On subsequent calls, the container will run directly. Every time you want to access the simulator, you will need to type this same command to enter inside the container.

```
(eurobot_2023_students) $ sudo bash run.sh
```

Upon completion of this command you enter in the container's environment with:

```
$ cd /project
```

This folder is a shared volume corresponding to the `eurobot_2023_students` folder. You can modify the files in this folder outside the container with any software you like (ex: SublimeText). The changes will then be visible inside the container.

3 Building and Running the code

Building the code is straightforward as all librairies you need are already included in the container. Navigate to the workR directory and create a build folder.

```
[/project] $ mkdir -p workR/build && cd workR/build
```

Then run the configuration procedure. This must be done the first time and everytime you add or remove files from the project. You don't need to do it at every change in the code itself.

```
[/project/workR/build] $ cmake ..
```

Then compile the project. This has to be done every time you change the code.

```
[/project/workR/build] $ make -j8
```

Finally, run the created executable and enjoy the simulator.

```
[/project/workR/build] $ ./exe_projectRobotran
```

4 Accessing the realtime visualization

In order to interact with the simulation, you can use the embedded webserver. Simply navigate with your browser (preferably chrome) to the address *localhost:5000*. Click on the menu on the top right and then connect. The project should be loaded. In the *Animation* view you should be able to launche the simulation with the start button. In this page, you can use the keyboard arrows to interact with the robots. A realtime plot is also available to inspect signals in your controllers. Finally, you can slow down, accelerate or pause the simulation.

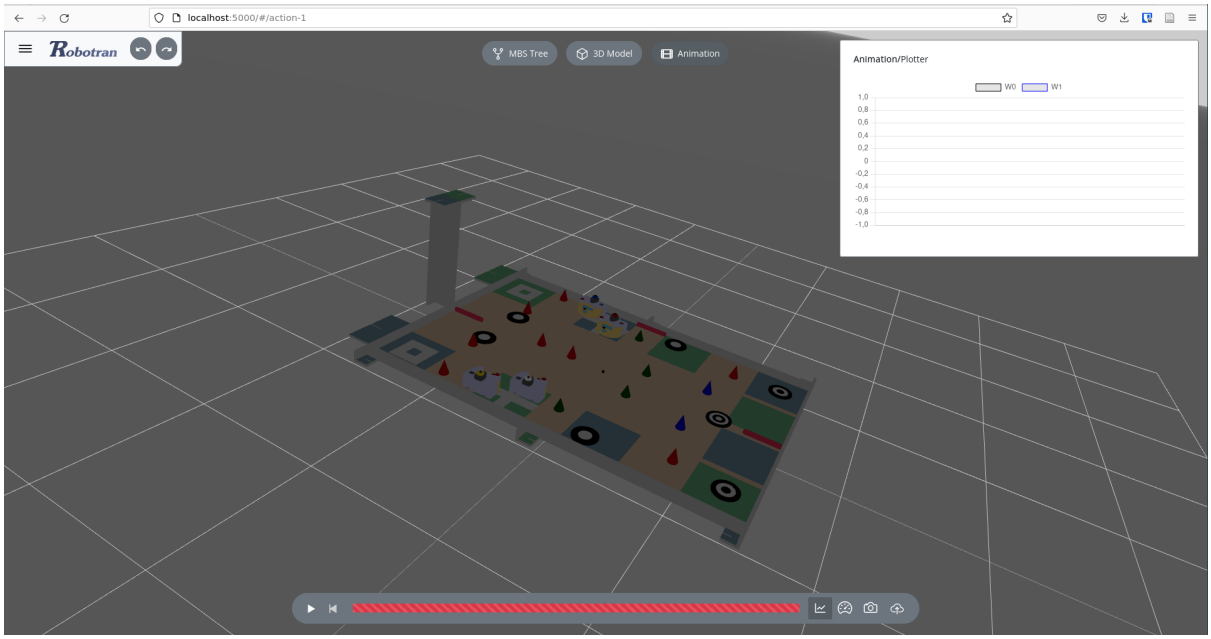
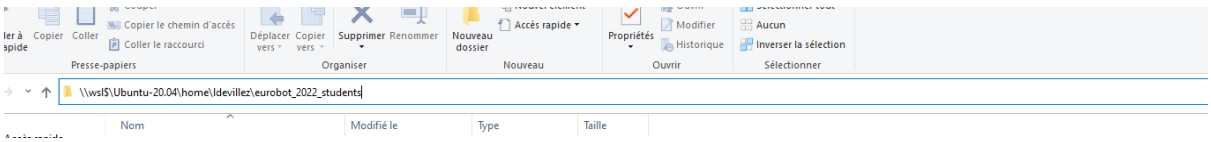


Fig. 2: Realtime visualization

5 Accessing the code (Windows)

If you followed the steps correctly the code should be accessible inside your ubuntu terminal (probably inside `\home\username\eurobot_2023_students`) but it is not showing in your windows file explorer. You can access it by taping directly its path:

```
\\wsl$\Ubuntu-20.04\home\username\eurobot_2023_students
```



So it will be easier for you to used an IDE.