Documentation for Robofish Experiments

Fritz Francisco

September 15, 2022

Full repository containing all required information can be found on GitHub:

https://github.com/RoboFishLab/Robofish.io

Installation

Currently only Linux is supported

- WLAN Router: Network Name: RoboFish; Password: vhr0b0134js
- RoboFish ID 135: Fixed IP: 192.168.0.4
- RoboTracker: Entire folder should be downloaded to Desktop or other reliable location
- .AppImage should be executed once, then closed. This creates config.ini file! File supplied here should be used as template
- Search for `config.ini` file (somewhere in a `FU Berlin/RoboTracker/` directory). Change `Plugin` directory at the bottom of the file, to point towards the correct plugin file to be used in the `RoboTracker` folder
- Install `v4l-utils` and `guvcview`:
  find instructions here: https://davejansen.com/logitech-streamcam-on-linux/
- `guvcview` can used to set the camera parameters, which can be found in the provided template file. Load `oldenburg_guvcview_settings.gpfl` for experiments.
- Start `RoboTracker` by clicking on the .AppImage or running `START_ROBOTRACKER.sh`
- Supply robot with fully charged battery and turn on before usage
- If Robot was turned on before `RoboTracker` the robot should be found automatically. Otherwise go to **Search Network for Robot** in the top right in `RoboTracker`

Setup

1. Make sure battery packs for the robot are charged, prior to all experiments
2. Open Baser Video Recording Software
3. Start `guvcview` by pressing the Windows key and typing `guvcview` to find it or by opening a terminal and typing `guvcview` and pressing Enter. Load `oldenburg_guvcview_settings.gpfl` (**Settings > Load Profile**)
4. Start `RoboTracker` by clicking on the .AppImage file or by clicking on `START_ROBOTRACKER.sh`

5. If Robot was turned on before `RoboTracker` the robot should be found automatically. Otherwise go to **Search Network for Robot** in the top right in `RoboTracker`
6. Start tracking by clicking on **Start Tracking** and make sure the detection indicator (thin line) is between the two LED spots and shows the correct RobotID
7. If the robot is turned on and connected a window should appear with RobotID as title. There, you can select whether you want to manually control the robot using the A,W,S and D keys on a keyboard (**Manual AWSD control**) or have it follow a predefined trajectory (**Trajectory**). Once selected click on **Activate** to start the system.
   NOTE: Do not touch the controller and speed settings in `RoboTracker`, since there is currently a bug when running in trajectory mode and using a non-default speed setting.

Pump Mechanism

The pump consists of the following parts:

- 12V DC Peristaltic Motor
- ESP32 WROOM-32 Microcontroller
- TB6612 Dual DC/Stepper Motor Driver
- LED
- Switch
- 5V External Power Source (USB)
- 12V External Power Source

The general mechanism is as follows. The pump is powered by the external 12V power source and controlled by the microcontroller through the TB6612 motor driver. To do so, three pins are used. Two pins supply the polarity of the motor (rotation clockwise or counterclockwise), while one pulse width modulated pin gives the duty cycle, or amount of power which the motor receives.
A microcontroller is a unit which can control certain processes and electrical circuits, and is commonly used for home automation at low voltage (3-5V). The idea is that the controller can store a execution script on it's internal memory and executed whenever supplied with power (commonly through the USB port). In this case the microcontroller can be set up, for the motor to be controlled by the switch and the LED to illuminate whenever the motor is running. However, the controller can do much more.
By using Arduino IDE, a script can be uploaded to the microcontroller which is executed on startup during which the `setup` function is always executed first and only once, followed by the `loop` function, which is executed iteratively with a sub-second frequency.
In order to upload a script to the microcontroller, a USB cable capable of transmitting data information (which not all cables can do) and a computer with Arduino IDE installed, is all which is required.

To establish a connection between the controller and the computer the follow these steps:

1. Connect USB to computer and microcontroller
2. Start Arduino IDE
3. If you have a freshly installed version of Arduino IDE, please make sure to install all dependencies for the ESP32 WROOM-32. A good introduction can be found here: https://randomnerdtutorials.com/installing-the-esp32-board-in-arduino-ide-windows-instructions/

4. In Arduino IDE, go to **Tools** and set the following settings:

Board:	ESP32 Dev Module
Upload Speed:	921200
CPU Frequency:	240 MHz (WiFi/BT)
Flash Frequency:	90MHz
Flash Mode:	QIO
Flash Size:	4MB (32Mb)
Partition Scheme:	Default 4MB with spiffs (1.2MB APP/1.5MB SPIFFS)
Core Debug Level:	None
PSRAM:	Disabled

5. Further, make sure **Port** shows a path (for example `/dev/ttyUSB0`), as this should point to the USB port which is connected to the controller. If no port is shown, the cable is faulty or there is no connection with the controller.

6. Once a connection is established and all settings have been set correctly you can verify the script by clicking on the **Verify** button with the ‘check’ symbol in the top left corner. This will test the script by compiling it, as if it were uploaded.

7. If the test was sufficient and didn’t return any error, the script can be uploaded by clicking on the **Upload** button next to the test button. This can take a few seconds depending on how large the script is.

8. Once the script was uploaded the microcontroller will execute it as wished and you can monitor output defined in the script with `Serial.println()` or `Serial.print()` by using the **Tools > Serial Monitor** or **Tools > Serial Plotter**. These two options are only available if the controller is connected to the computer running Arduino IDE.