User Manual

CrazyTrain

2022–12–16

Version 1.1

Status

Reviewed	Anton Bossen	2022-12-15
Approved	FiFilipe Barbosal	2022-12-15

Project Identity

Group E-mail:	antbo569@liu.se
Homepage:	http://www.isy.liu.se/tsrt10/group
Orderer:	Filipe Barbosa, LiU, ISY Phone: - E-mail: filipe.barbosa@liu.se
Customer:	Linköpings universitet (LiU), ISY Phone: - E-mail: -
Supervisor:	Daniel Arnström, LiU, ISY Phone: - E-mail: daniel.arnstrom@liu.se
Course Responsible:	Daniel Axehill, LiU, ISY Phone: +46 13 28 40 42 E-mail: daniel.axehill@liu.se

Participants of the group

Name	Responsibility	E-mail
Morteza Akbari	Hardware (HW)	morak752@student.liu.se
Erik Axelsson	Documentation (DOC)	eriax970@student.liu.se
Anton Bossen	Project leader (PL)	antbo569@student.liu.se
Oskar Grönlund	Simulation (SIM)	oskgr783@student.liu.se
Anton Håkansson	Design (DES)	antha842@student.liu.se
Lukas Jonsson	Software (SOF)	lukjo147@student.liu.se
Patrik Lindström	Testing (TEST)	patli821@student.liu.se
Emil Lydell	File management (FM)	emily553@student.liu.se

CONTENTS

DOCUMENT HISTORY

Version	Date	Changes made	Changes by	Reviewer
0.1	2022-11-08	First draft.	Project group	Anton Bossen
1.0	2022-12-09	First Version.	Project group	Anton Bossen
1.1	2022-12-09	Revised from comments by Orderer	Anton Bossen	Anton Bossen

1 INTRODUCTION

This is a user manual for a system created in the project Crazytrain in the course TSRT10, where software for controlling multiple drones have been developed. This document aims to provide the user with basic information about the system components for a safe and efficient usage of the entire system. A description of the installation and usage of every subsystem is included.

2 REQUIRED COMPONENTS

In this section, the required components needed in order to reproduce the project are listed.

- **Computer**: Running Ubuntu and ROS Noetic.
- **Crazyflie**: The drones used for the project are Crazyflie 2.1. Crazyflies are nano-quadcopters that are developed by Bitcraze and are open source [1]. The nano-quadcopters have four rotors, an internal measurement unit (IMU) and, for this project, an active marker deck.
- **Crazyradio**: Radio developed by Bitcraze which connect the drones to the computer.
- **Gamepad**: The gamepad is used to control a Crazyflie manually.
- **Qualisys Mocap System**: Camera system which uses motion capture and is used to get the position measurements of the Crazyflies. The system can detect both active or passive markers to track objects.

3 INSTALLATION

This section of the manual will describe how to properly install the system to be able to use it. The installation will be separated into different parts since the system in its entirety depends on multiple different software sources. To ensure proper functionality, please follow the steps in chronological order. Some of the steps may be subject to change with eventual updates of software which the system depends on.

3.1 Ubuntu

To run the system, it is required to use a Linux-based OS. Furthermore, it is highly recommended to use the Linux distribution Ubuntu 20.04 since the system is confirmed to work on this version. Other Linux distributions and versions may work, although this is not guaranteed. Worth noting is that the system **will not run** on the Windows Subsystem for Linux (or WSL for short), instead the Linux OS should be installed either directly on the computer or onto a Virtual Machine.

3.2 Python & PIP

The system consists of multiple different scripts, mainly written in the programming language Python. To ensure that all python based scripts are runnable, it is required to have Python3 installed. Furthermore, it is worth to consider using the Python3 version 3.7 since the system has been confirmed to work with this version. The system may be able to run with different versions of Python3, however this is not guaranteed.

When python is installed on the computer, it is also heavily suggested to install pip to make it easier to install python packages later on. Find a suitable pip version that coincides with the python version active on the computer and install it accordingly.

3.3 GIT

To be able to work with the CrazyTrain system, it is easiest to make a clone of the GIT Repository where all the project files are stored. Hence, it is required that your computer has GIT installed, fortunately most computers running Linux has GIT pre-installed.

3.3.1 *Cloning CrazyTrain Repository*

Open a terminal and navigate to a new folder where you want to place the project. Clone the CrazyTrain repository by entering the following command.

```
$ git clone git@github.com:CrazyTrain2022/CrazyTrain2022.git
```

Since the repository contains the project *crazyswarm* as a submodule, this has to be recursively updated. To do this, use the command:

```
$ git submodule update --init --recursive
```

3.4 ROS

The communication between the drones within the system is handled by the Robotic Operating System software, or ROS for short. Therefore, it is required to have ROS installed, more specifically the distribution ROS Noetic should be used. For a detailed guide on how the installation of ROS is carried out, please see the official installation guide for ROS: http://wiki.ros.org/ROS/Installation

The installation guide should be followed until the Tutorial part, if the user is not well-traversed in ROS and wants to gather increased knowledge one can continue with the Tutorial part of the installation.

3.5 Configure & Build

After the CrazyTrain Repository has been retrieved, it needs to be installed. The CrazyTrain repository is based on the Crazyswarm project (courtesy of USC-ACTLab, https://github.com/USCACTLab/crazyswarm) but also includes submodules called NLopt (courtesy of GitHub user stevengj, https://github.com/stevengj/nlopt) and UAV-trajectories (courtesy of GitHub user *whoenig*, https://github.com/whoenig/uav_trajectories . Installation of all of these modules will be explained in this section.

3.5.1 *Building Submodules*

For convenience purposes, the building of the submodules has been collected within a bash script. To run the script, make sure that you are located within the CrazyTrain2022 folder and proceed to the terminal and run the command:

```
$ bash ./setup/build_nlopt_uav.sh
```

If any unexpected error occur, the user can traverse to the setup folder and look at the build file directly and try to run the commands included separately.

3.5.2 *Configuring Crazyswarm*

To configure the Crazyswarm submodule, go to their installation guide at: https://crazyswarm.readthedocs.io/en/latest/installation.html and follow the steps. **Avoid the step where their repository is cloned** as this is already included in the CrazyTrain repository. Make sure to mark the tab **"Physical Robots and Simulation"** in order to fly the real drones. Use the visualiser alternative "matplotlib" as that is what has been implemented in this project.

3.6 Additional dependencies

For full functionality of the system, there are a few additional installations that have to be made. These will be described in this section.

3.6.1 *Gamepad drivers*

In order to fly a drone manually with an XBOX-controller, additional steps are required. A full guide for this can be found in the file *Manual_control_installation.txt* located in /CrazyTrain2022/setup.

3.6.2 *Tkinter*

Tkinter is a python package that is used to create our Graphical User Interface, hence it is needed to achieve full functionality. The Tkinter package is most easily installed via the use of pip by running the following command in the terminal:

```
$ python3 -m pip install tk
```

4 SETUP PROCEDURES

When the required software have been downloaded and set up according to section 3 it is time to configure the hardware. In this section, the setup procedure for Qualisys and connection to the drones are explained in detail.

4.1 Qualisys system

Following is the start-up procedure for the Qualisys motion capture system in Visionen. These steps need to be done before every session in Visionen. The procedure is generally straightforward but can be troublesome on occasion. This is further explained in section 4.3.

1. Roll down the curtains so the environment in Visionen becomes dark.
2. Start the Qualisys cameras using the controller. The cameras should now light up.
3. Start Qualisys on the computer, and open *"LiU New setup"*
4. Press the red record button and let all available cameras connect to the computer, see figure 1 for reference.

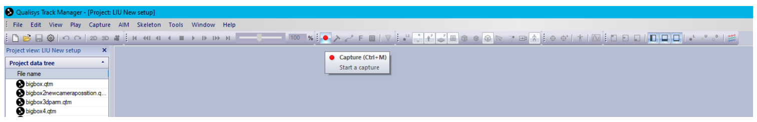
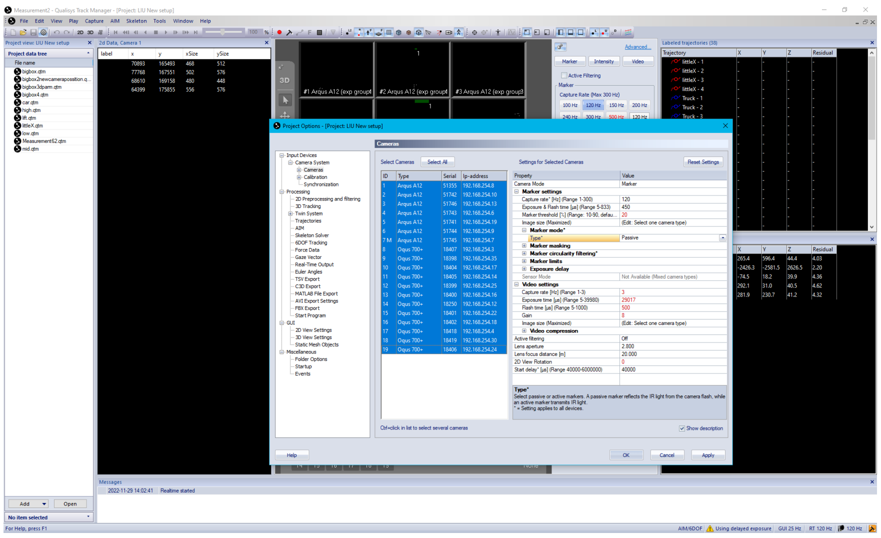


Figure 1: Red capture button

4.1.1 *Calibration*

When Qualisys has successfully been opened, the next step is to calibrate. The calibration can be a problem, and often the calibration needs to be done multiple times.

1. Ensure that the Qualisys system is measuring passive markers. See figure 2 for reference.
2. Place the calibration origin in the desired spot. See figure 3 for reference.
3. Press the calibration button and specify how long the calibration should take (preferred time is 240 seconds). See figure 4 for reference.
4. Walk around slowly in Visionen with the calibration stick for the duration of the calibration. Wave the stick slowly up and down, and focus on calibrating origo best. See figure 5 for reference.
5. If everything has gone well, the calibration should be successful and Qualisys should now be ready. Otherwise, the calibration has to be redone. Try to change the exposure time to 200 μs under the tab *2D*, see figure 4 for reference, and try again.
6. If the calibration still fails, restart Qualisys both on the computer and the camera system and restart from step 3 in section 4.1.



Project Options - [Project: LIU New setup]

Cameras

Select Cameras | Select All

ID	Type	Serial	Ip-address
1	Arqus A12	51355	192.168.254.8
2	Arqus A12	51742	192.168.254.10
3	Arqus A12	51746	192.168.254.13
4	Arqus A12	51743	192.168.254.6
5	Arqus A12	51741	192.168.254.19
6	Arqus A12	51744	192.168.254.9
7 M	Arqus A12	51745	192.168.254.7
8	Oqus 700+	18407	192.168.254.3
9	Oqus 700+	18398	192.168.254.35
10	Oqus 700+	18404	192.168.254.17
11	Oqus 700+	18405	192.168.254.14
12	Oqus 700+	18399	192.168.254.25
13	Oqus 700+	18400	192.168.254.16
14	Oqus 700+	18250	192.168.254.12
15	Oqus 700+	18401	192.168.254.22
16	Oqus 700+	18402	192.168.254.18
17	Oqus 700+	18418	192.168.254.4
18	Oqus 700+	18419	192.168.254.30
19	Oqus 700+	18406	192.168.254.24

Ctrl+click in list to select several cameras

Settings for Selected Cameras | Reset Settings

Property	Value
Camera Mode	Marker
Marker settings	
Capture rate* [Hz] (Range 1-300)	120
Exposure & Flash time [µs] (Range 5-833)	450
Marker threshold [%] (Range: 10-90, defau...	20
Image size (Maximized)	(Edit: Select one camera type)
Marker mode*	
Type*	Passive
Marker masking	
Marker circularity filtering*	
Marker limits	
Exposure delay	
Sensor Mode	Not Available (Mixed camera types)
Video settings	
Capture rate [Hz] (Range 1-3)	3
Exposure time [µs] (Range 5-39980)	29017
Flash time [µs] (Range 5-1000)	500
Gain	8
Image size (Maximized)	(Edit: Select one camera type)
Video compression	
Active filtering	Off
Lens aperture	2.800
Lens focus distance [m]	20.000
2D View Rotation	0
Start delay* [µs] (Range 40000-6000000)	40000

Type*
Select passive or active markers. A passive marker reflects the IR light from the camera flash, while an active marker transmits IR light.
* = Setting applies to all devices.

☑ Show description

Figure 2: Menu where marker type is chosen. The menu is accesed by the setting button in the right corner in Figure 1.

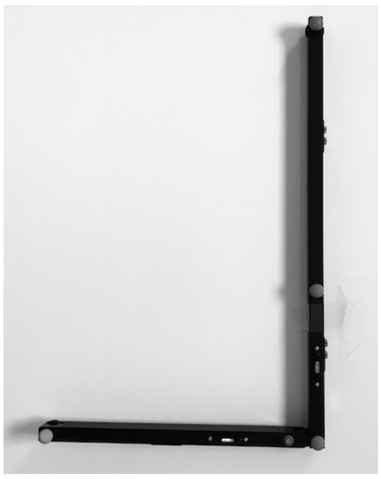
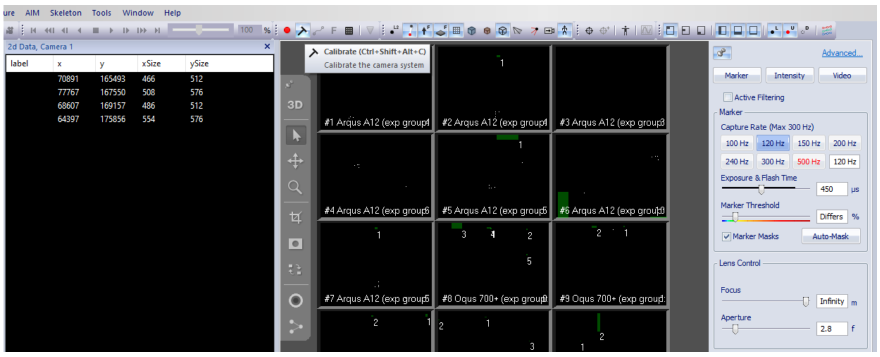


Figure 3: Calibration Origo


2d Data, Camera 1

label	x	y	xSize	ySize
	70891	165493	466	512
	77767	167550	508	576
	68607	169157	486	512
	64397	175856	554	576

Calibrate (Ctrl+Shift+Alt+C)
Calibrate the camera system

Capture Rate (Max 300 Hz): 120 Hz
Exposure & Flash Time: 450 µs
Marker Threshold: Differs %
Marker Masks ✓
Focus: Infinity m
Aperture: 2.8 f

Figure 4: Calibration button

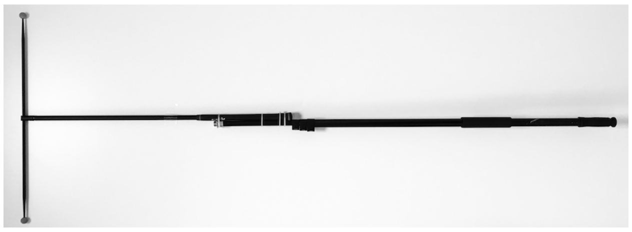


Figure 5: Calibration Stick

4.2 Setup of the drones

The first time the drones are flown, they need to be configured and connected with the sensor system.

4.2.1 *Connect drone(s) to radio and assign ID*

Before the drones can be flown, they have to be assigned individual ID's in the *crazyflie-client*. This is a setup step that is only required the first time the drones are flown.

1. Connect the Crazyradio to a USB port on the computer and turn on the drone(s) by pressing the small power button, located on the front of the Crazyflie. Connect the Crazyflie to the computer via a USB cable.

2. Open a new terminal and navigate to the path */crazyflie-clients-python/bin* and the start the crazyflie client by typing *cfclient*.

3. Start by pressing the *Scan* button in the top left corner. An alternative to connecting via USB will show up in the neighboring scroll bar. Press the USB-alternative and click on *Connect*. If a connection is successful, the orientation of the drone should be displayed in the mainframe.

4. Click on *configure* under the *Connect* tab to set the radio channel and Crazyflie address. The channel should be set to ch. 80 and the Crazyflie radio address should be assigned to 0xE7E7E7E7<X>, where <X> is replaced with the desired drone ID.

5. It is important to update the drone firmware to the latest edition by pressing the *bootloader* under the *Connect* tab. Check the latest firmware releases and update if a new version is available. Do not touch or restart the crazyflie until all flashing is done, and you see "status: idle" at the bottom

6. The next step is to open the *parameters* tab and make sure that each drone has unique active marker ID:s. (If the tab is not present, it can be enabled from the *view* tab). Open the *activeMarker* section and check the marker values. Make sure that these values are individual for each arm on the drone and that all drones have a different set of ID:s. Preferably, drone 1 is labeled with values 1-4, drone 2 with values 5-8, and so on, for simplicity. This step is crucial since it makes sure that the Qualisys system can keep track of each drone separately.

7. Close the *Crazyflie-client* and continue to the next section.

4.2.2 *Define drone(s) to Qualisys*

After connecting the drones to the radio and assigning an ID to each drone, it is time to define the drones in Qualisys.

1. Ensure that the Qualisys system is measuring active markers and that the exposure time is set to 450 μs. See Figure 4 for reference.

2. Place the drones at their initial position. Make sure that the front arrow on the marker deck is pointing in the same direction as your global positive x-axis.

3. On the Qualisys computer, four white dots for each drone should now be displayed. For one drone, mark all four dots by ctrl+left-clicking each dot.

4. Right-click the drone on the computer and choose *Define rigid-body*. Enter the name cf1, cf2, cf3, or cf4 for which drone you are trying to define.

5. Repeat step 3 and 4 until all drones are defined in Qualisys.

6. Try to move around the drones and cover the active markers to see if the drones have been defined the right way. If the names on the bodies in Qualisys shift place, open the *crazyflie-client* and check that the marker ID:s are unique for each drone.

4.2.3 *Connect drone(s) and Qualisys to the GUI*

The last step in the setup is connecting the drones and Qualisys to the GUI.

1. In the directory */CrazyTrain/CrazyTrain2022/crazyswarm/ros_ws* run *catkin_make*.

2. In the directory */CrazyTrain/CrazyTrain2022/* open the GUI by running *python3 GUI/gui.py*.

4.3 Common problems with setup

- Qualisys is unreliable when setting up. Sometimes when everything is done in the right way, it still does not work. The reason for this is quite unclear and could have a number of causes. Keep repeating the configuration steps until it works, or try to restart the system.

- Limited memory available on the computer running Qualisys in Visionen. This problem has made it impossible to save a project i Qualisys and thus, the setup stages described in section 4.2.2 must be redone before every new session.
- Check that the drones are defined the right way in Qualisys and that their local coordinate systems are oriented in the same way as the global coordinates. If not, it will affect the flight performance of the drones.
- Calibration of Qualisys. The calibration often returns as not successful. Try calibrating again, if the calibrating returns not successful again, and it is the same camera that is problematic. Try to turn the system off and on again.

5 OPERATING THE SYSTEM USING THE GUI

When the system has been configured properly and the drones are ready to fly, everything can be operated from the Graphical User Interface which connects the whole system. An overview of the GUI can be seen in figure 6.

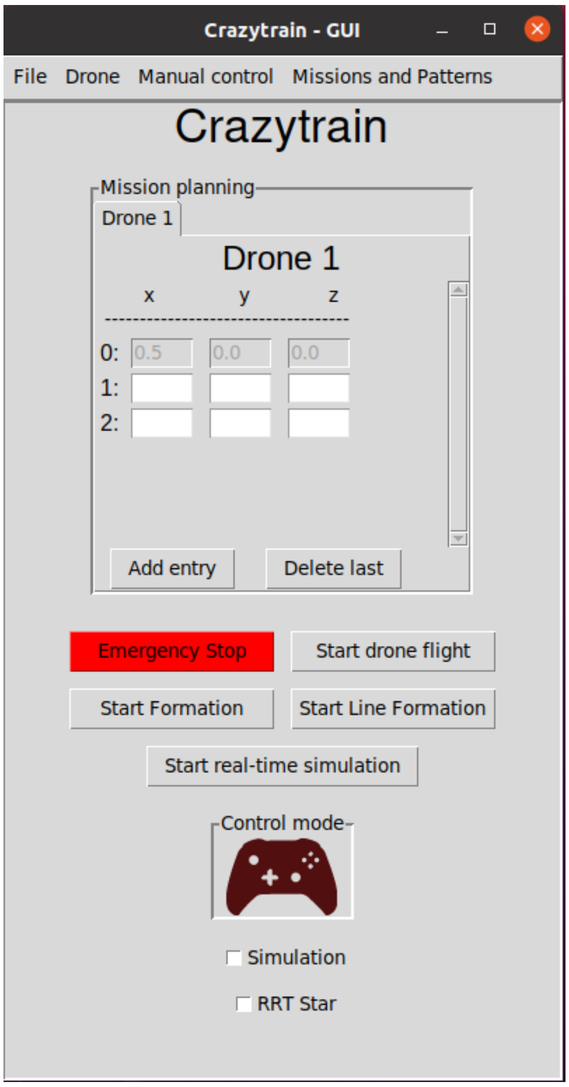


Figure 6: GUI Overview

5.1 Initiate the system

To initiate the system, follow the steps below:

1. Open a terminal and navigate to the folder CrazyTrain2022.

2. Open GUI by typing *python3 GUI/gui.py*. A GUI window will open, which looks according to fig 6.
3. Initiate drones by clicking *Select drones* in the Drone tab in the GUI. An interface will pop up (presented in figure 7) where the drones can be selected. Choose which drones to use and click on the battery button to verify that the drones are connected. If a drone is connected to the radio, it will display its battery level, otherwise, an error message will occur. When drones have been selected, the chooser window can be closed.

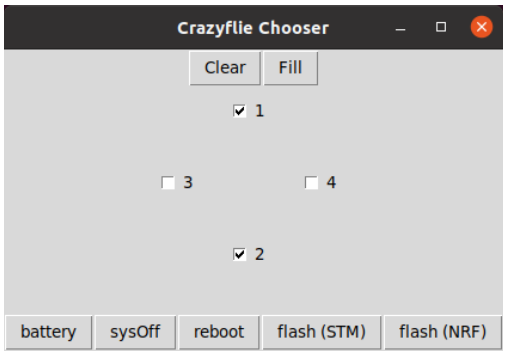


Figure 7: Crazyflie chooser interface

4. Initiate the ROS server, hover swarm, and the Qualisys-node by clicking *start all preflight scipts* in the Drone tab. This will open several new ROS terminals and a visualisation of the drones in Visionen, similar to what you see in Qualisys. **Do not close these windows**.

5.2 Manual control

1. Initiate the system by following step 1–4 in Section 5.1 if you have not already.
2. Go to the Manual Control tab in GUI and select "Take manual control" and select drone 1. In the GUI, the Xbox controller logo should turn green.
3. Click the *Start flight* button on the GUI. A new terminal window will show up asking for a password. In our project, this password is "crazytrain123". Do not close this terminal window.
4. Click the *Start flight* button on the GUI again. A new terminal will show up asking for a password. You do not have to write the password again. You should be able to control drone 1 now manually.

Move the right analog stick up/down to make the drone fly up/down. Move the right analog stick left/right to make the drone turn left/right. Move the left analog stick up/down to make the drone go forward/backward. Move the left analog stick left/right to make the drone go left/right. The drone will land by pressing the B button.

5.3 Create flight mission

Flight missions can either be created by putting in coordinates directly in the GUI (waypoints) and then by pressing *Start drone flight*, or by going to *Missions and Patterns* to choose an already defined mission or pattern. Two missions are also predefined in the main frame of the GUI, *Start Formation* and *Start Line Formation*.

5.4 Formation flight

Formation flight works by adding the desired waypoints to the leader drone. The leader drone will always be the one with the lowest ID of the drones that are being used. So if drone #1, #2, #3 and #4 are flying the leader is drone #1 and if drone #2, #3 and #4 are flying, drone #2 is now the leader. You also need to add a waypoint to all non-leader drones, this can be for example the waypoint [0, 0, 1]. These waypoints will not be used in the flight, but are needed because of the way the system reads waypoints from the GUI. Now the formation flight can be started by pressing either *Start Formation* or *Start Line Formation*.

5.5 Change controller and filter parameters

The controller, the controller parameters, and the filter parameters can be changed by pressing *Change parameters* under *File* which will open up the respective file in which those things can be changed. After the change, it is important to do a *catkin_make* which can either be done in a terminal or by pressing the *catkin_make* button in *File*.

5.6 Save and Load

Missions which has been defined in the GUI or patterns can be saved. This is done by pressing *Save mission* under *File*. Those files can also be loaded by pressing *Load mission* under *File*.

5.7 Emergency Stop

The button *Emergency stop* can be pressed at any times during a flight. This calls the land function for the drones and then shuts everything down.

5.8 RRT

A rapidly-exploring random tree can be used when flying or simulating with waypoints by just marking the *RRT star* checkbox. The RRT checks if there are any obstacles defined in the workspace and calculates a route to avoid these. Obstacles can be added by going to *Add obstacles* under *File* and then by specifying the obstacle coordinates. The obstacles are added by defining a minimum and maximum value of the obstacle border in each dimension.

5.9 Simulation environment

Missions, patterns, and waypoints can all be simulated by just doing the same procedure as explained in chapter 5.3 and by also marking the simulation box. The simulation does not require ROS or connection to any drone.

5.9.1 *Simulation GUI*

The simulation mode behaves differently when following waypoints (Drone Flight) rather than simulating a preset mission. When simulating drone flight, after the waypoints have been determined (with or without the RRT*-planner activated) the user will be presented with a Simulation GUI according to figure 8.

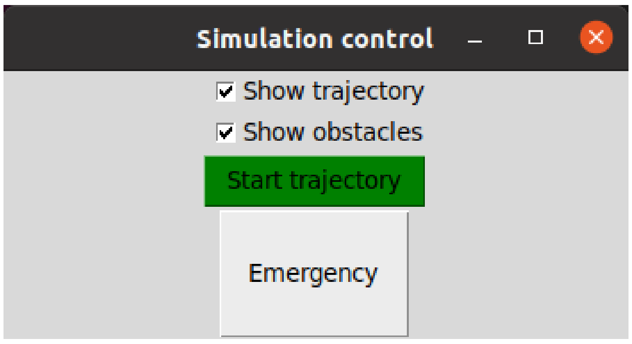


Figure 8: Graphical User Interface for the simulation

In the simulation GUI, there are two buttons along with two optional graphical features. The first button presented will start the simulation of the chosen trajectory and the second button gives the user an option to simulate the Emergency Land feature, which is described in section 5.6. The graphical options that are given will activate/deactivate whether the drone trajectory along with obstacles shall be shown in the simulation. Obstacles for example are beneficial to show when using the RRT*-planner to determine the trajectory.

When the user is finished with the simulation, it can be closed by pressing the red close button in the top-right corner (which can be seen in figure 8).

5.9.2 *Matplotlib*

Once the simulation has been started, an animated Matplot3D (which is a python package used for presenting a visual of the simulation) graph will be presented. An example simulation can be seen in figure 9.

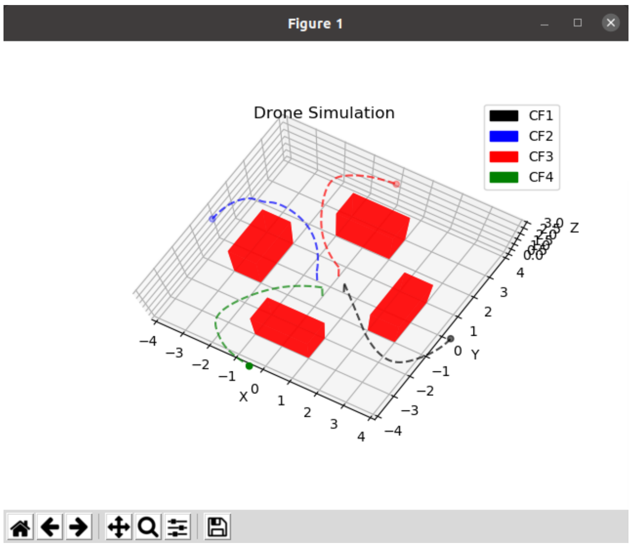


Figure 9: Example of a finished simulation with RRT* enabled.

When the animation is finished and the user is satisfied with the results, there is an option to save a picture of the Matplot3D plot. This is most easily done by just pressing "S" on the keyboard.

Other useful available options are zoom and rotation. To rotate the plot, the user holds the left-click of the mouse and can then move the mouse to rotate the graph. To enable the zooming feature, the user instead holds the right-click of the mouse and subsequently moves the mouse to zoom in on the plot.

5.10 Real Time Visualisation

To run the real-time visualisation, simply press the *Real Time Visualisation* - button in the GUI. A matplotlib 3D plot will pop up together with a control window that contains features for visualisation. To show the planned trajectories for each drone, press the "Show trajectory" - button. This will load all current trajectory files and plot them in the 3D graph. This will also start a calculation of the distances in the xy-plane between the drones' measured positions and their corresponding trajectories. The control window also gives the ability to show predefined obstacles, plot the position errors in a time plot, and save mission data points into CSV files. The saved files are stored in the *Log_files*-folder in scripts. The visualisation will also start by default when *start flight* is pressed to fly the drones in reality.

5.11 Common problems with GUI

- *Start hover swarm* can sometimes be problematic and do not connect to the drones. This can be because the computer is **not connected to Visionen's network**. If this is not the case, try to close the *hover swarm* and the terminal and open it again. Otherwise, check if Qualisys is working right.
- Occasionally, the drones spontaneously crash mid-flight for no apparent reason. This could have many different reasons, but most likely it has something to do with a loss of connection to the computer.

- If the waypoints can't be used to make a trajectory, the GUI will display a warning in the terminal, and will not continue with the desired action. The warning will display for which drone it was unable to make the trajectory, and the user can then change the waypoints. The points have to be within 5x5 meters from the centre of the xy-plane, and below 3.5m in height. Also, non-zero values have to be bigger or equal to 0.05.

6 STOP SYSTEM OPERATIONS

1. **Close GUI**: This is done by simply closing the main window. This will remove all the created waypoints- and trajectory CSV files that have not already been saved as missions. Also, close all ROS windows.
2. **Turn off the drones**: Use the same button as for startup.
3. **Qualisys system**: The Qualisys cameras should be left on. Simply shut down the Qualisys window on the computer and log out. Raise the curtains.

7 LIMITATIONS OF THE SYSTEM

7.1 Autonomous and manual control

While it is possible to both fly the drones autonomously and manually, the system is not built to fly in both modes, simultaneously. Drone #1 is also the only drone that is able to fly manually. If there are any problems with the switch between the autonomous and manual control, the best way to solve it is by closing the whole system and then open it again.

7.2 Local coordinates

A problem with how the coordinate system is defined is that the drones have locally defined origos of the global coordinate system. When flying all four drones, it will create four different coordinate systems. If you tell the drones to all go to the same waypoint, for example [1, 1, 1], they will not go to the same point since they have different coordinate systems. The origo of the coordinate system is defined from the initial position of the drone.

7.3 Obstacle avoidance

For obstacle avoidance to work the obstacles need to be defined manually in the GUI. A better way would be if the obstacles could be found by Qualisys instead through passive markers and then automatically added into the *obstacle* CSV file. The path for the drones is now also calculated before the flight which means moving obstacles does not yet work with our implementation.

7.4 Qualisys system

The system is based on the Qualisys system, which means that without Qualisys the drones will not receive its pose. This in turn will impede the drones ability to fly.