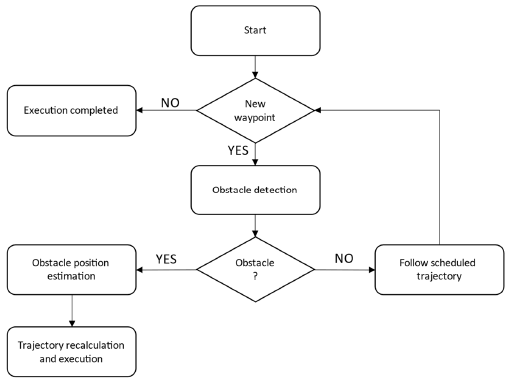
Student Name :

Student ID :

**Introduction:**

Obstacle avoidance is a safety feature found in more advanced drones, including many DJI drones. The feature enables drones to scan the environment and detect any obstacle along their route in real-time. Once your DJI aircraft detects the obstacle, it reacts immediately by avoiding it. The program is made up of 1 drone and 1 obstacles which successfully following the obstacles avoidance algorithm.



**Figure :** Decision tree of the obstacle avoidance algorithm

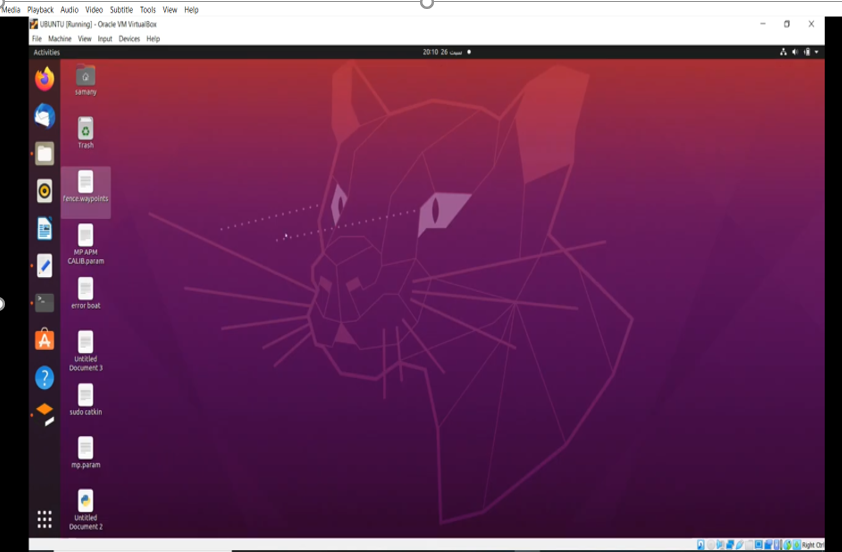
**Environment and Tools :**

**Linus**

Linux s a family of [open-source](https://en.wikipedia.org/wiki/Free_and_open-source_software) [Unix-like](https://en.wikipedia.org/wiki/Unix-like) [operating systems](https://en.wikipedia.org/wiki/Operating_system) based on the [Linux kernel](https://en.wikipedia.org/wiki/Linux_kernel), an [operating system kernel](https://en.wikipedia.org/wiki/Kernel_(operating_system)) first released on September 17, 1991, by [Linus Torvalds](https://en.wikipedia.org/wiki/Linus_Torvalds). Linux is typically [packaged](https://en.wikipedia.org/wiki/Package_manager) in a [Linux distribution](https://en.wikipedia.org/wiki/Linux_distribution).

Distributions include the Linux kernel and supporting [system software](https://en.wikipedia.org/wiki/System_software) and [libraries](https://en.wikipedia.org/wiki/Library_(computer_science)), many of which are provided by the [GNU Project](https://en.wikipedia.org/wiki/GNU_Project). Many Linux distributions use the word "Linux" in their name, but the [Free Software Foundation](https://en.wikipedia.org/wiki/Free_Software_Foundation) uses the name "GNU/Linux" to emphasize the importance of [GNU](https://en.wikipedia.org/wiki/GNU) software.

We use Linux Ubuntu.



**Gazebo 11**

Gazebo brings a fresh approach to simulation with a complete toolbox of development libraries and cloud services to make simulation easy. Iterate fast on your new physical designs in realistic environments with high fidelity sensors streams. Test control strategies in safety, and take advantage of simulation in continuous integration tests.

Gazebo is an [open-source](https://en.wikipedia.org/wiki/Open-source) 3D [robotics simulator](https://en.wikipedia.org/wiki/Robotics_simulator). It integrated the [ODE](https://en.wikipedia.org/wiki/Open_Dynamics_Engine) physics engine, [OpenGL](https://en.wikipedia.org/wiki/OpenGL) rendering, and support code for sensor simulation and actuator control.

Gazebo can use multiple high-performance physics engines, such as [ODE](https://en.wikipedia.org/wiki/Open_Dynamics_Engine), [Bullet](https://en.wikipedia.org/wiki/Bullet_(software)), etc. (the default is ODE). It provides realistic rendering of environments including high-quality lighting, shadows, and textures. It can model sensors that "see" the simulated environment, such as [laser range finders](https://en.wikipedia.org/wiki/Laser_range_finder), cameras (including wide-angle), [Kinect](https://en.wikipedia.org/wiki/Kinect) style sensors, etc

**Mavros**

ArduPilot capabilities can be extended with [ROS](http://www.ros.org/) (aka Robot Operating System).

[ROS](http://www.ros.org/) provides libraries, tools, hardware abstraction, device drivers, visualizers, message-passing, package management, and more to help software developers create robot applications. In the future, we expect ROS will be replaced by [ROS2](http://design.ros2.org/articles/why_ros2.html)

[MAVROS](http://wiki.ros.org/mavros) is a ROS “node” that can convert between ROS topics and [MAVLink messages](https://github.com/ArduPilot/mavlink) allowing ArduPilot vehicles to communicate with ROS.

<http://wiki.ros.org/mavros>

<https://ardupilot.org/dev/docs/ros.html>

**Connecting with ROS**

Before attempting to connect with ROS, ensure that the physical serial cable connection and ArduPilot side parameters have been set as described on the appropriate companion computer page ([RPi3](https://ardupilot.org/dev/docs/raspberry-pi-via-mavlink.html#raspberry-pi-via-mavlink), [NVidia TX1](https://ardupilot.org/dev/docs/companion-computer-nvidia-tx1.html#companion-computer-nvidia-tx1), [NVidia TX2](https://ardupilot.org/dev/docs/companion-computer-nvidia-tx2.html#companion-computer-nvidia-tx2) or [Intel Edison](https://ardupilot.org/dev/docs/intel-edison.html#intel-edison)).

The connection between ArduPilot running on the flight controller and ROS/MAVROS can be established with the following commands on the companion computer:

* open up a terminal and type, “roscore”
* open another terminal and start mavros as described [here on the ros/mavros wiki](http://wiki.ros.org/mavros#Usage).

roslaunch mavros apm.launch fcu\_url:=udp://:14855@

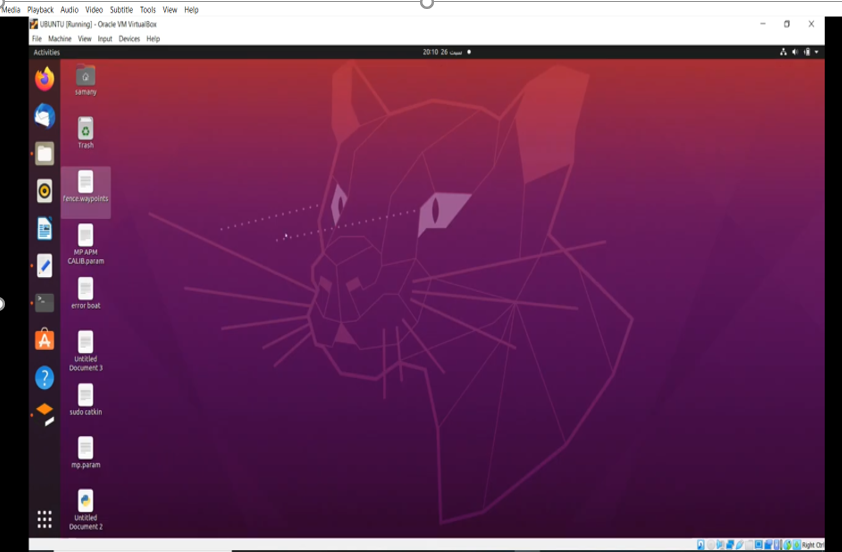
rosrun mavros mavsys mode **-**c 0 (sets the vehicle to mode "0")

rosrun mavros mavsafety arm (to arm the vehicle)

**Oracle Virtual Box:**

VirtualBox is a powerful x86 and AMD64/Intel64 [virtualization](https://www.virtualbox.org/wiki/Virtualization) product for enterprise as well as home use. Not only is VirtualBox an extremely feature rich, high performance product for enterprise customers, it is also the only professional solution that is freely available as Open Source Software under the terms of the GNU General Public License (GPL) version 2. See "[About VirtualBox](https://www.virtualbox.org/wiki/VirtualBox)" for an introduction.

Presently, VirtualBox runs on Windows, Linux, Macintosh, and Solaris hosts and supports a large number of [guest operating systems](https://www.virtualbox.org/wiki/Guest_OSes) including but not limited to Windows (NT 4.0, 2000, XP, Server 2003, Vista, Windows 7, Windows 8, Windows 10), DOS/Windows 3.x, Linux (2.4, 2.6, 3.x and 4.x), Solaris and OpenSolaris, OS/2, and OpenBSD.



**Procedure:**

**# gazebo\_apm\_mavros**

it is (gazebo models and worlds + ardupilot + my\_mavros\_api) in order to communicate and control drone to accomplish a specific missions.

**## install gazebo 11 from source**

http://gazebosim.org/tutorials?tut=install\_from\_source&cat=install

**## install ardupilot from source**

https://ardupilot.org/dev/docs/building-setup-linux.html

**## install ros from source**

http://wiki.ros.org/noetic/Installation/Ubuntu

**## install mavros/mavlink from source**

https://github.com/mavlink/mavros/blob/master/mavros/README.md#installation

**## download reposity**

1- open catkin\_ws/src on terminal

2- cd catkin\_ws

3- catkin build

4- gedit ~/.bashrc

5- at the end add these lines

\*\* ros path

  export ROS\_PACKAGE\_PATH=/catkin\_ws/src/gazebo\_apm\_mavros/:$ROS\_PACKAGE\_PATH

  export ROS\_PACKAGE\_PATH=/catkin\_ws/src/mavros/mavros:$ROS\_PACKAGE\_PATH

\*\* gazebo path

  export GAZEBO\_RESOURCE\_PATH=~/catkin\_ws/src/gazebo\_apm\_mavros/worlds:${GAZEBO\_RESOURCE\_PATH}

  export GAZEBO\_MODEL\_PATH=~/catkin\_ws/src/gazebo\_apm\_mavros/models:${GAZEBO\_MODEL\_PATH}

  export GAZEBO\_PLUGIN\_PATH=~/catkin\_ws/src/gazebo\_apm\_mavros/build:${GAZEBO\_PLUGIN\_PATH}

  export GAZEBO\_PLUGIN\_PATH=~/catkin\_ws/src/gazebo\_apm\_mavros/src:${GAZEBO\_PLUGIN\_PATH}

7- ctrl+s and close it

8- source ~/.bashrc

9- open scripts file on terminal

10- chmod +x obstacle\_avoidance.py             # to give it permission to run you will make this for all python modules #

11- tip:- open apm.launch file from mavros and change FCU\_URL connection to udp like "udp: //: 14550 @ 192.168.1.130@5760"

          or look at http://wiki.ros.org/mavros

**## Show Time**

**### open 4 terminals add each add these lines in order and wait until being intiated**

- first : roslaunch gazbo obstacle\_avoidance.launch

- second : sim\_vehicle.py -v ArduCopter -f gazebo-iris -I0

- third : roslaunch mavros  apm.launch

- fourth : rosrun gazbo obstacle\_avoidance.py

**##                                                obstacle\_avoidance onepoint lidar simulation**

**# Move\_nesw\_localy**

- first : roslaunch gazbo grass.launch

- second : sim\_vehicle.py -v ArduCopter -f gazebo-iris -I0

- third : roslaunch mavros  apm.launch

- fourth : rosrun gazbo move\_nesw\_localy.py

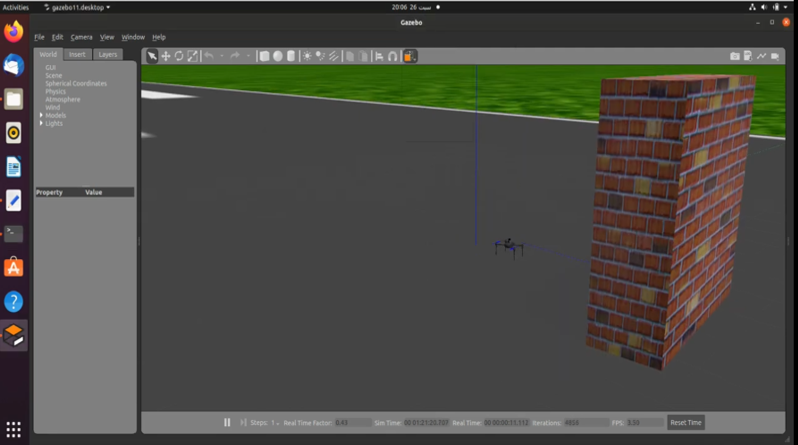


Fig : Drone Model Figure

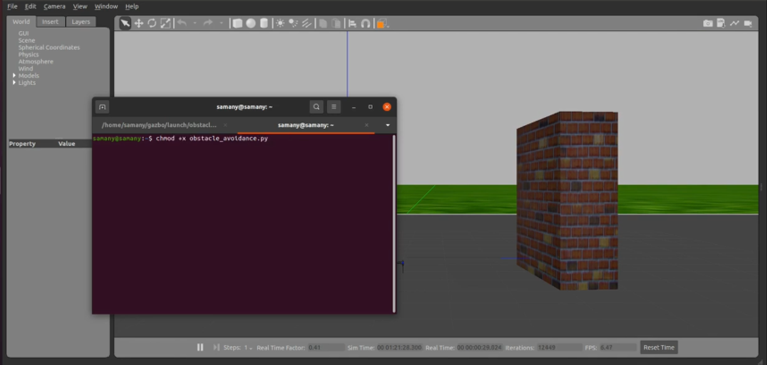


Fig : Simulation through terminal

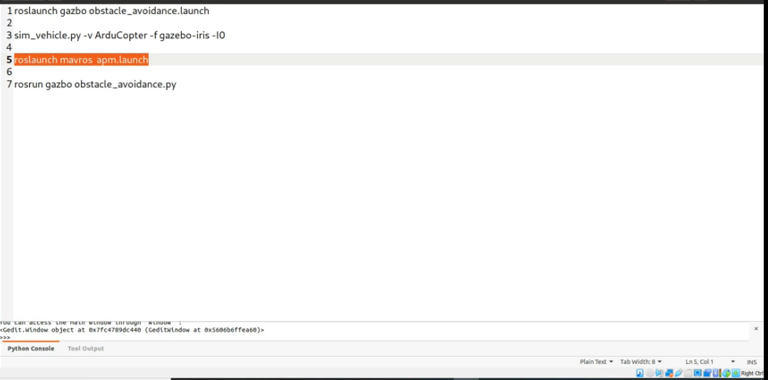


Fig : Commands

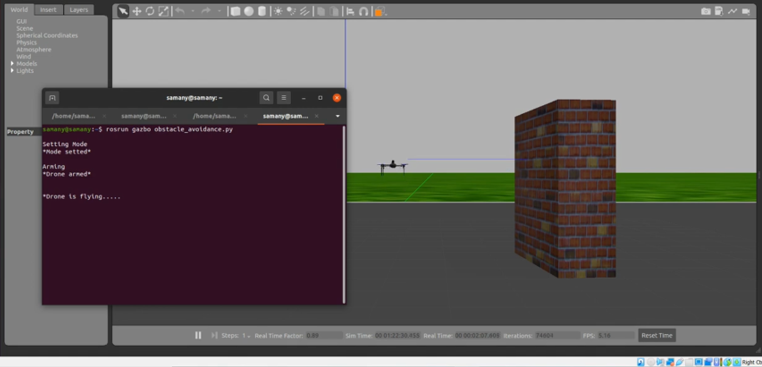


Fig : Drone Simulation through terminal

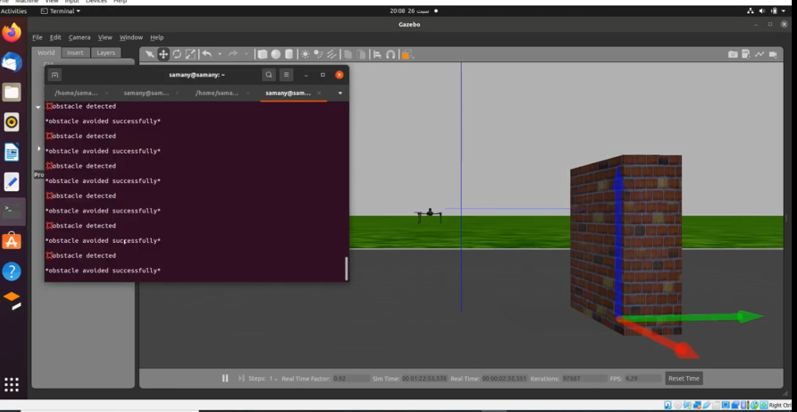


Fig : Drone Avoidance by the wall

**Note:**

We can see that drone successfully avoided the wall.

**Conclusion:**

In this program I was successfully manage to make a drone avoid the obstacles and in the terminal is shows the obstacles avoided successfully. The program can also avoid moving obstacles. In future, I would like to add more features such as avoiding using more obstacles and would more sensors that could replicate real like model. Because this is my first time using Linux and Gazebo .Therefore, I made a simple project to understand the basics and gradually with time enhance the program in future.

**Reference:**

1) <https://youtu.be/Lcc3XgO72hY>

2) <https://ardupilot.org/dev/docs/ros-connecting.html>

3) <https://www.youtube.com/watch?v=A-JaRgtljLg>

4) <https://www.youtube.com/watch?v=mKt4ZTaE2bk>

5) <https://en.wikipedia.org/wiki/Gazebo_simulator>

6) <https://classic.gazebosim.org/tutorials?tut=install_ubuntu>

7) <https://www.virtualbox.org/>