AE465 Introduction to Aerial Robotics

Final Report

SPRING 2019/2020

Student Number – Surname, Name: 201616263 – EMRE, Mustafa Can

Project Title: Scenario of an Aircraft Landing on a Moving Landing Ground

Contents

1. Step 15 1
   1.1. Adding The Necessary Files Using Windows Explorer 1
   1.2. Creating The World with Atom Text Editor 1
   1.3. Phase 1 Design of the Attitude and Altitude Controllers 1
   1.4. Phase 2 Design of The Position Controller 5
   1.5. Phase 3 Determination of the Object Position In the World Coordinates 9
   1.6. Results of the Step 15 15
2. Scenario of Aircraft Landing on a Moving Landing Ground 16
   2.1. Adding the Necessary Files Usign Windows Explorer 16
   2.2. Opening the New World File and Checking the Models 16
   2.3. Changing the Simulink File for Desired Scenario 17
   2.4. Results of the Scenario 20

1. Step 15

1.1. Adding The Necessary Files Using Windows Explorer

File opened from Ubuntu with the following code:

$ explorer.exe .

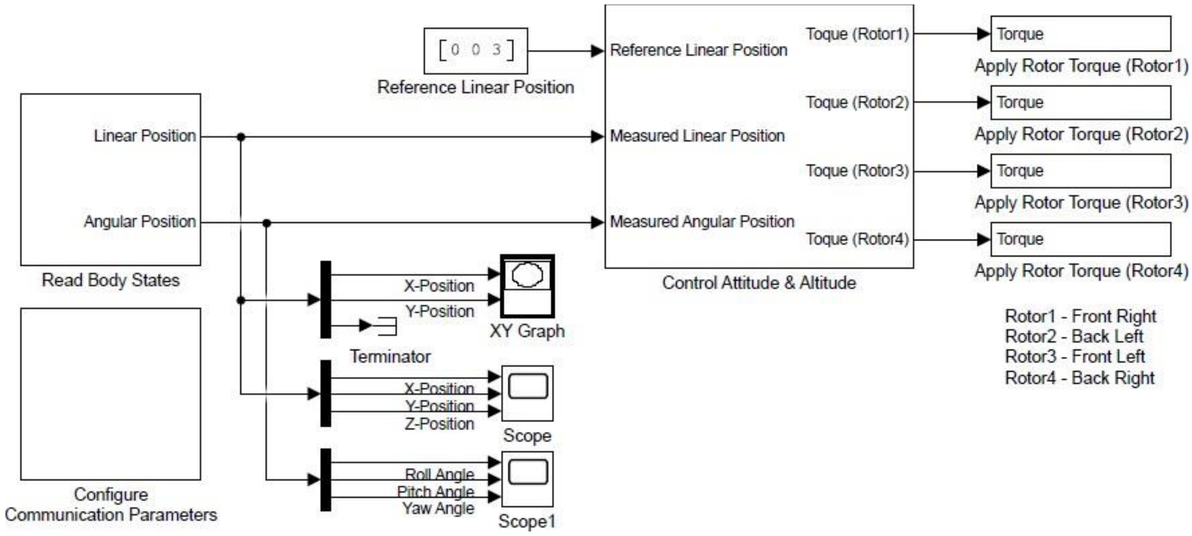
And the files copy pasted to \\wsl$\Ubuntu-18.04\home

1.2. Creating The World with Atom Text Editor

“myiris.world” is created from atom text editor which is opened from Ubuntu.

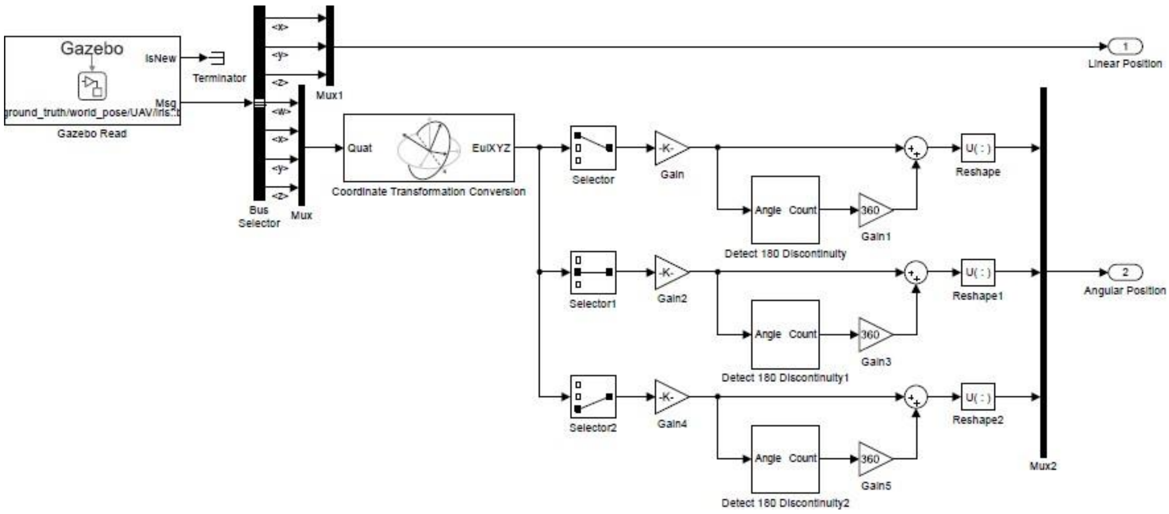
1.3. Phase 1 Design of the Attitude and Altitude Controllers

In this phase, a Simulink file will be created to control the attitude and altitude of the Iris quadcopter model. Simulink model is given below:

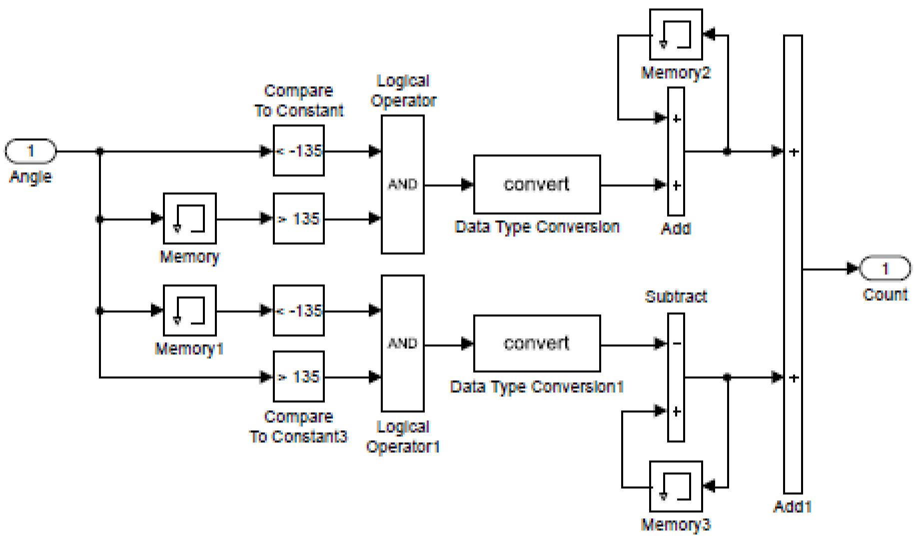


Simulink model for attitude and altitude controllers.

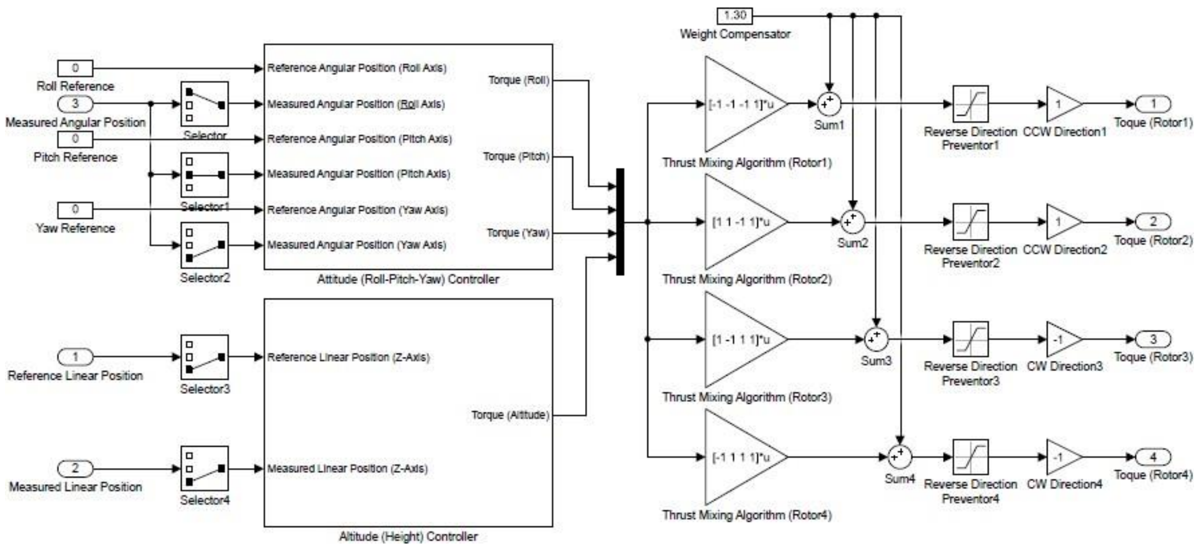
Subsystems:



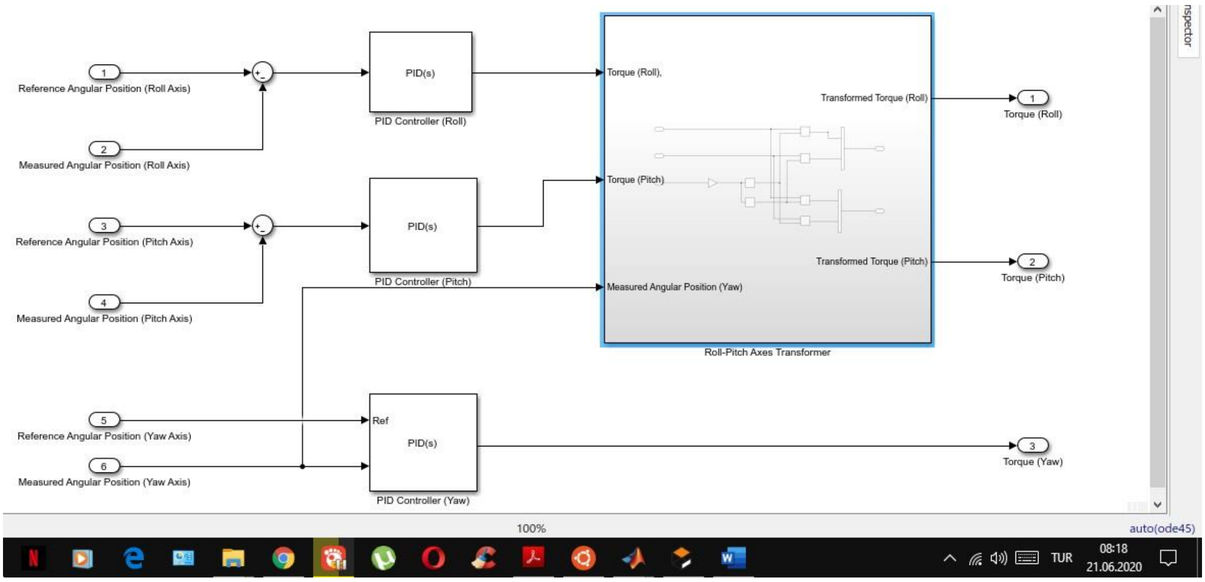
Real Body States



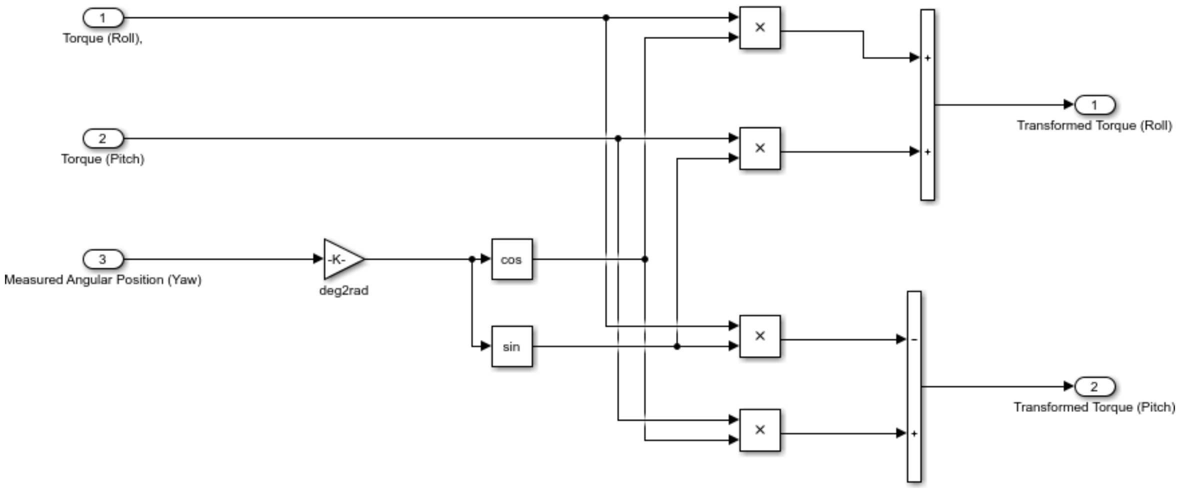
Detect 180 Discontinuity, Detect 180 Discontinuity1, Detect 180 Discontinuity2.



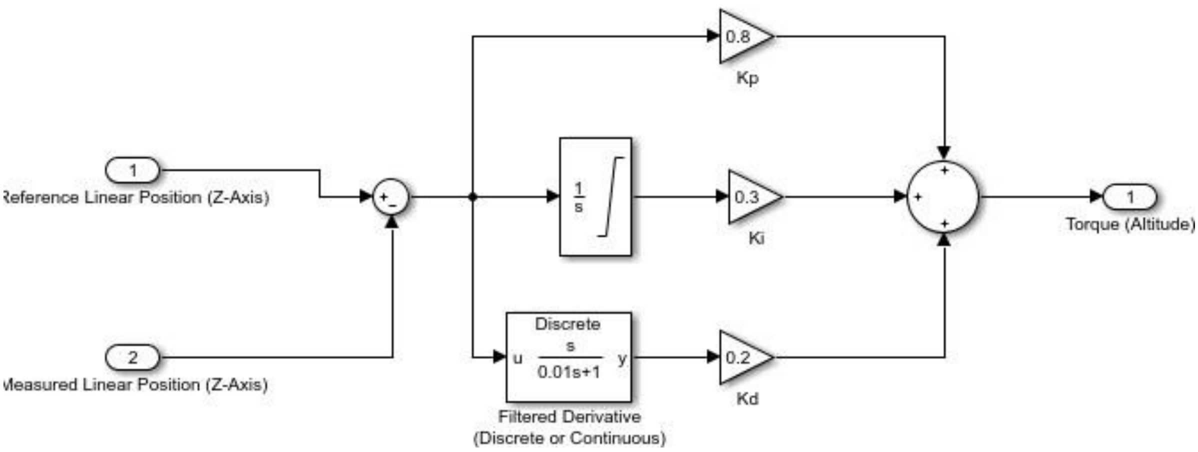
Control Attitude and Altitude.



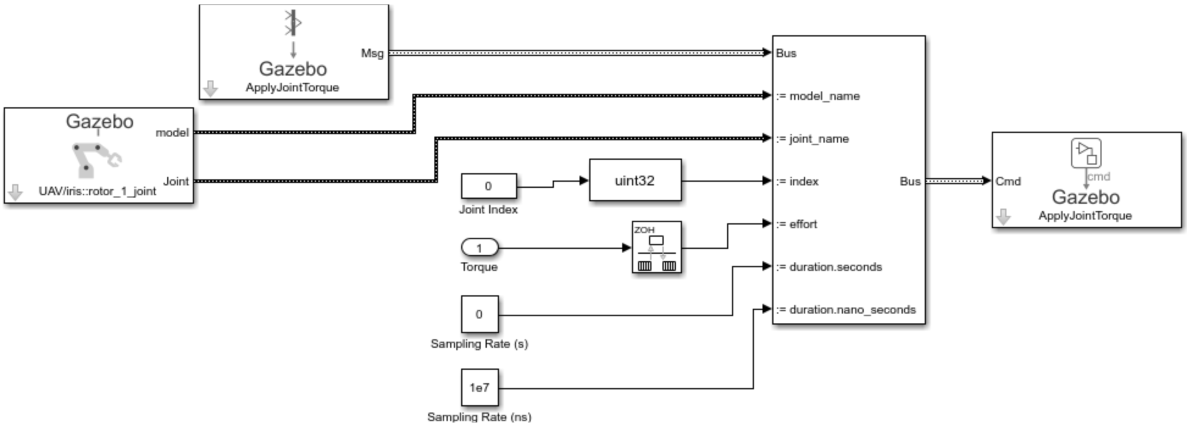
Altitude Controller.



Roll-Pitch Axes Transformer.



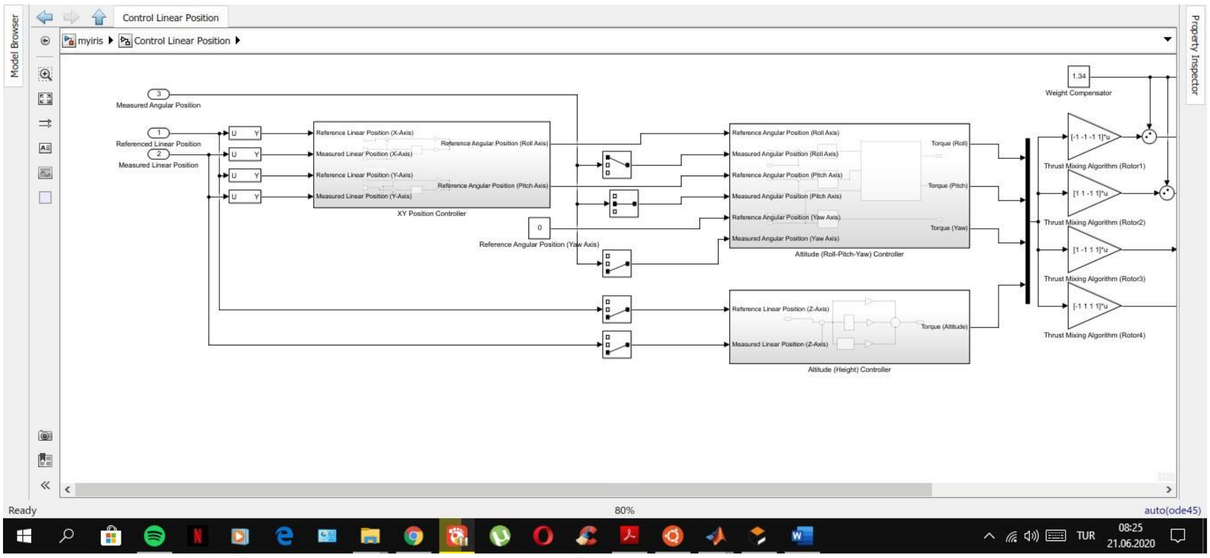
Altitude Controller.



Apply Rotor Torque(Rotor 1), Apply Rotor Torque(Rotor2), Apply Rotor Torque(Rotor 3), Apply Rotor Torque(Rotor 4) subsystems.

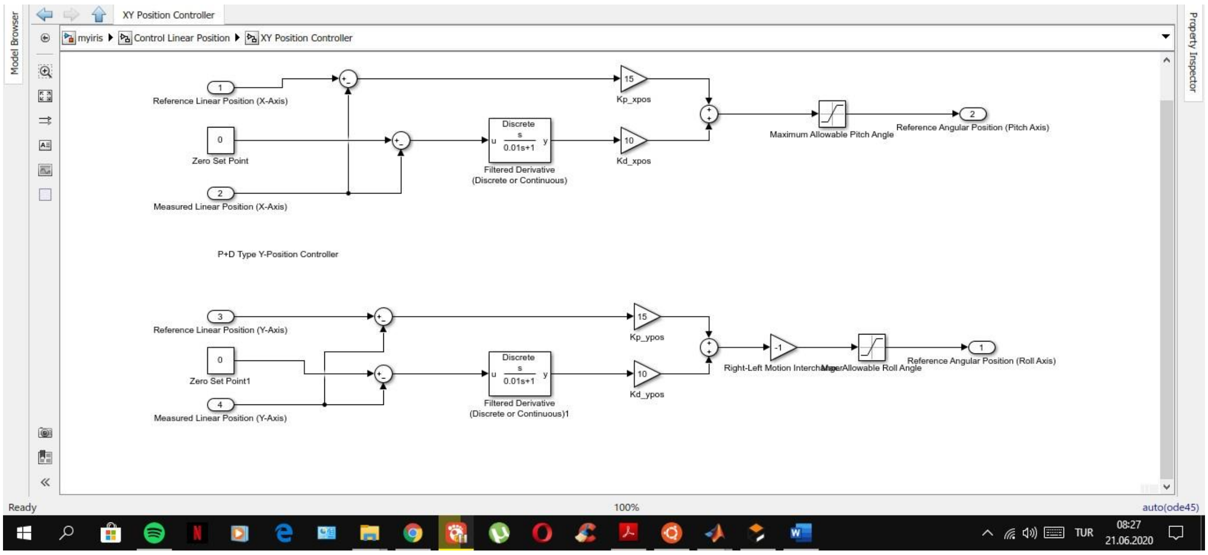
1.4. Phase 2 Design of The Position Controller

We need position controller because even if the designed controllers seems to be working well for a short period of time it seems problematic for longer periods. In the phase 1 aircraft shifts.



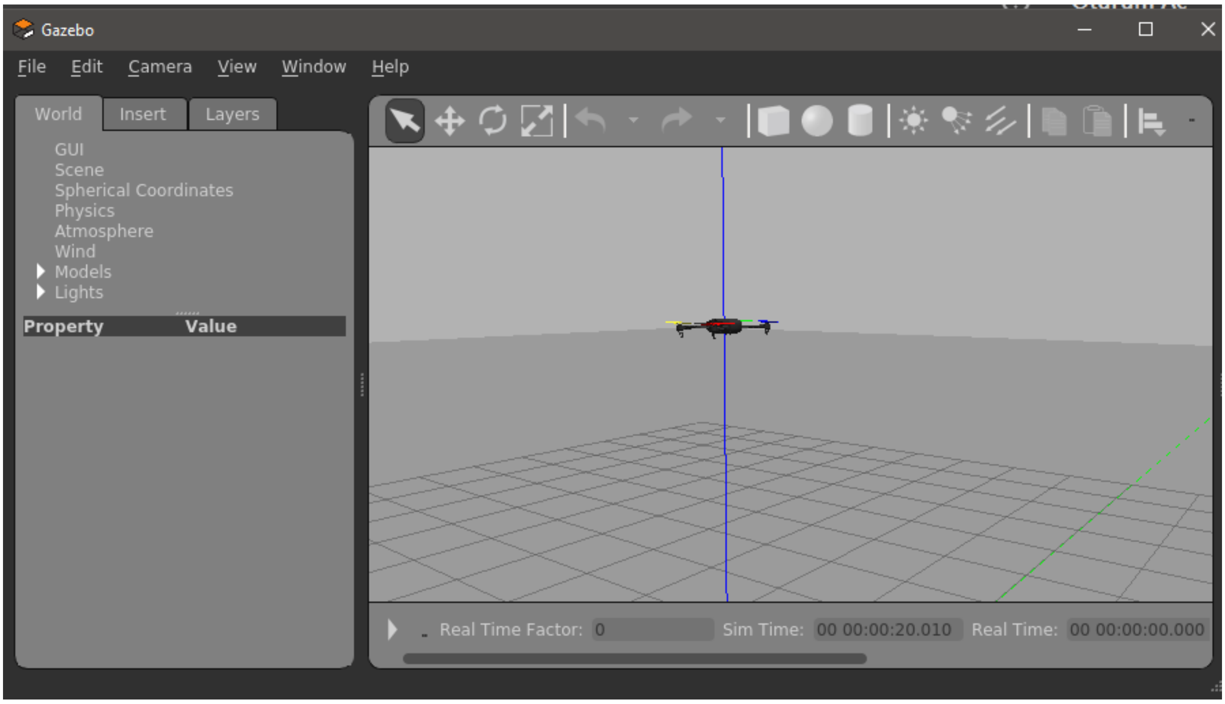
Changed part of the Control Attitude & Altitude(now it is called Control Linear Position).

Subsystems:

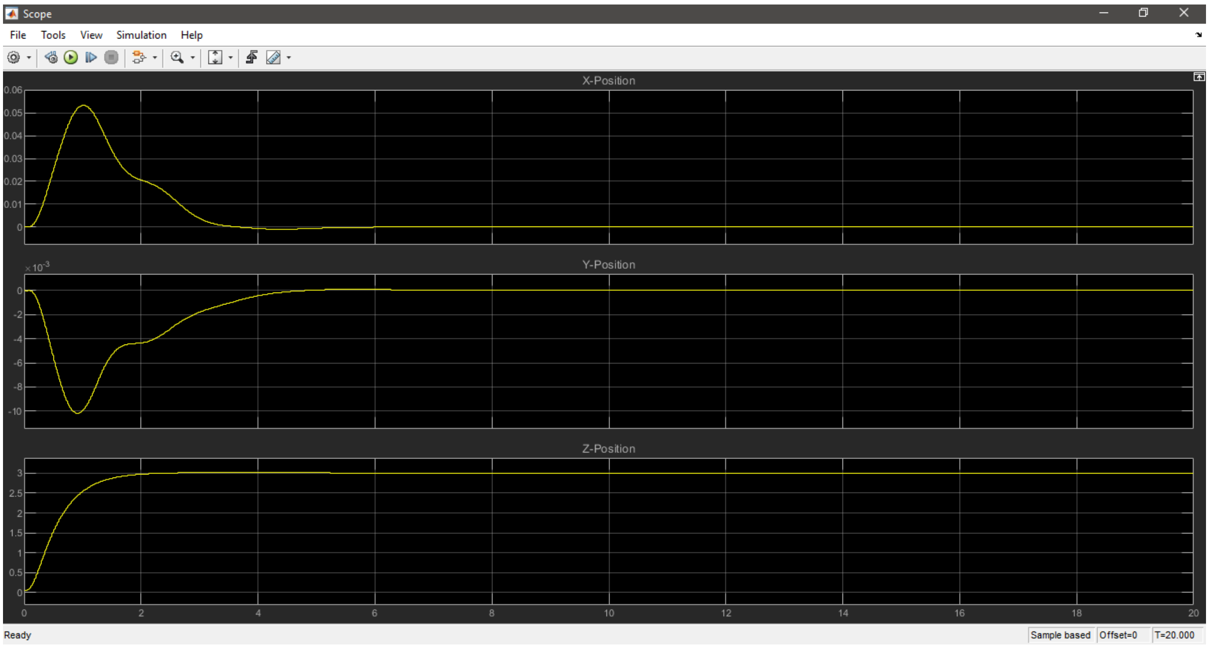


XY Position Controller.

Results for Reference Linear Position = [0, 0, 3] :



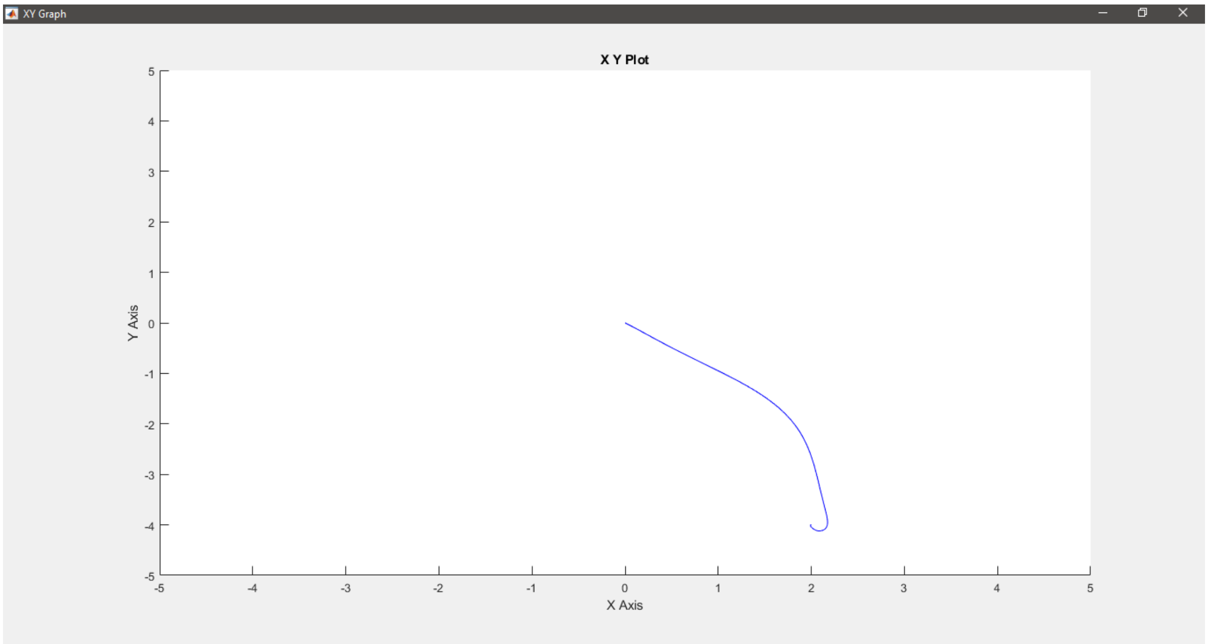
Position of the Aircraft after 20 seconds.



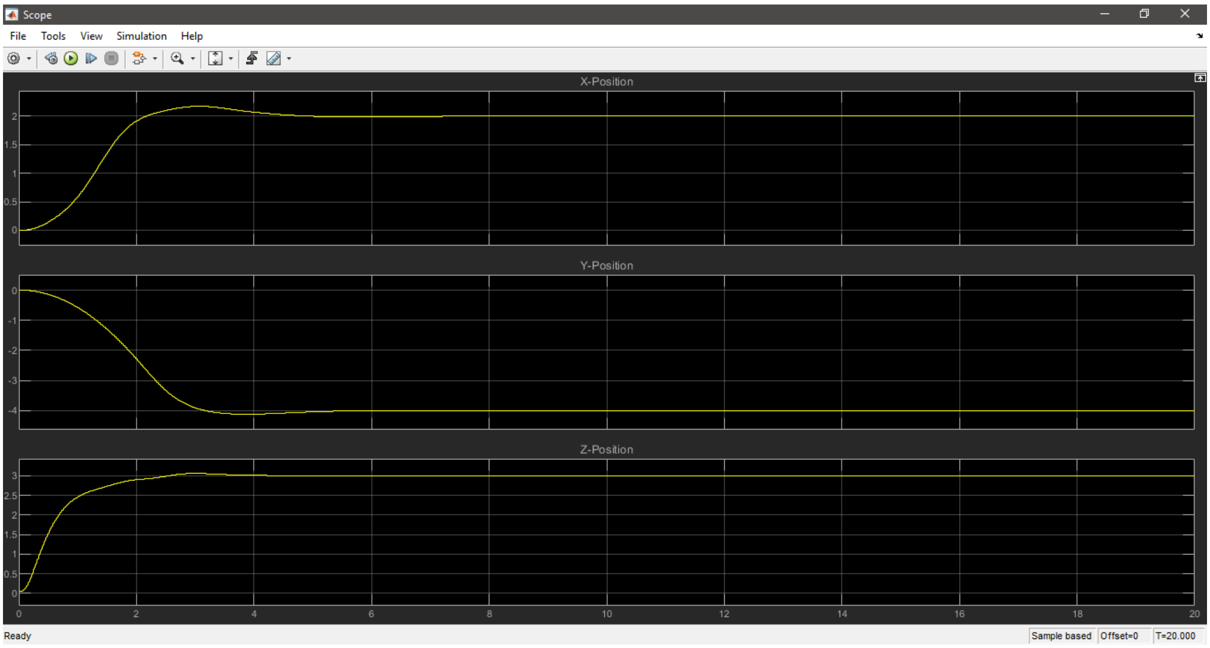
X-Y-Z Positions-time Plots.

Roll-Pitch-Yaw Angle-time Plots.

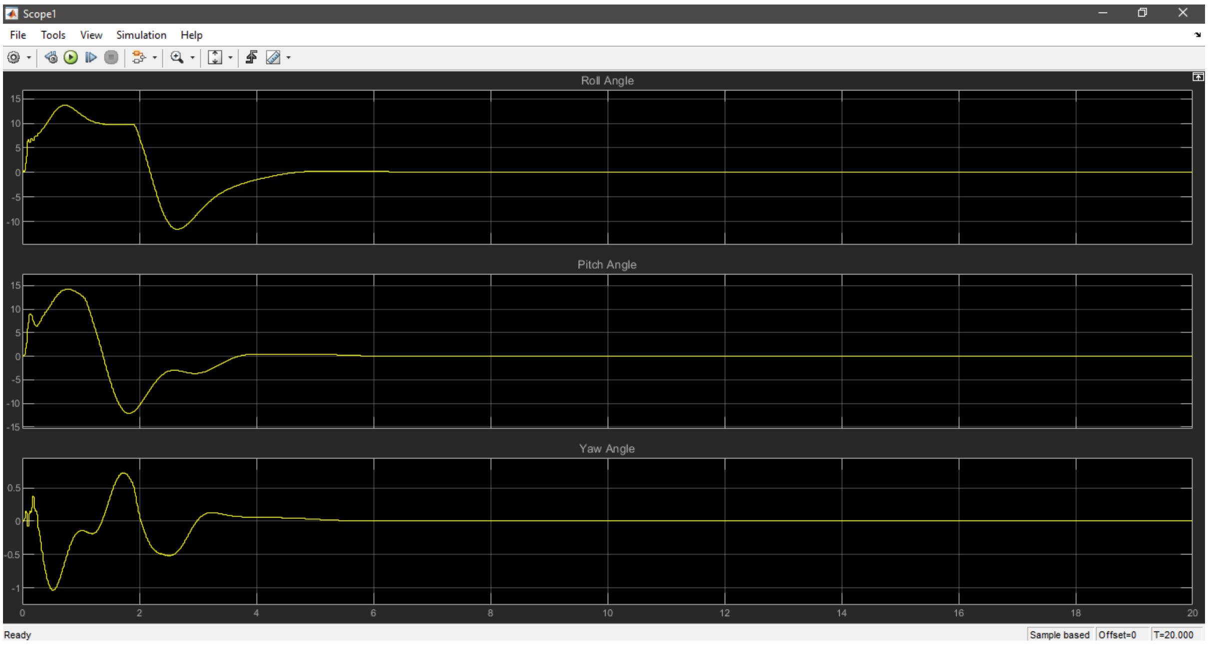
Results for For Reference Linear Position = [2, -4, 3] :



X-Y Plot.



X-Y-Z Positions-time Plot.



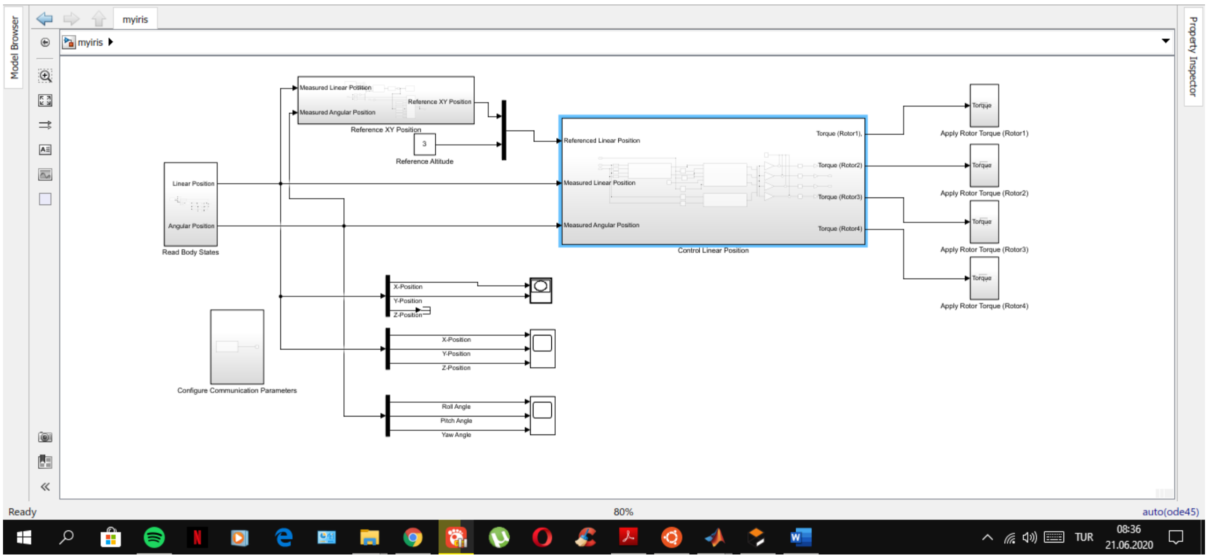
Roll-Pitch-Yaw Angle-time Plots.

1.5. Phase 3 Determination of the Object Position In the World Coordinates

In the previous phase, the reference values of linear position were given arbitrarily by using a constant block. In this phase, the x and y coordinates of this vector will be calculated automatically.

The following commands are merged and added to “myiris.world” using Atom Text Editor opened from Ubuntu:

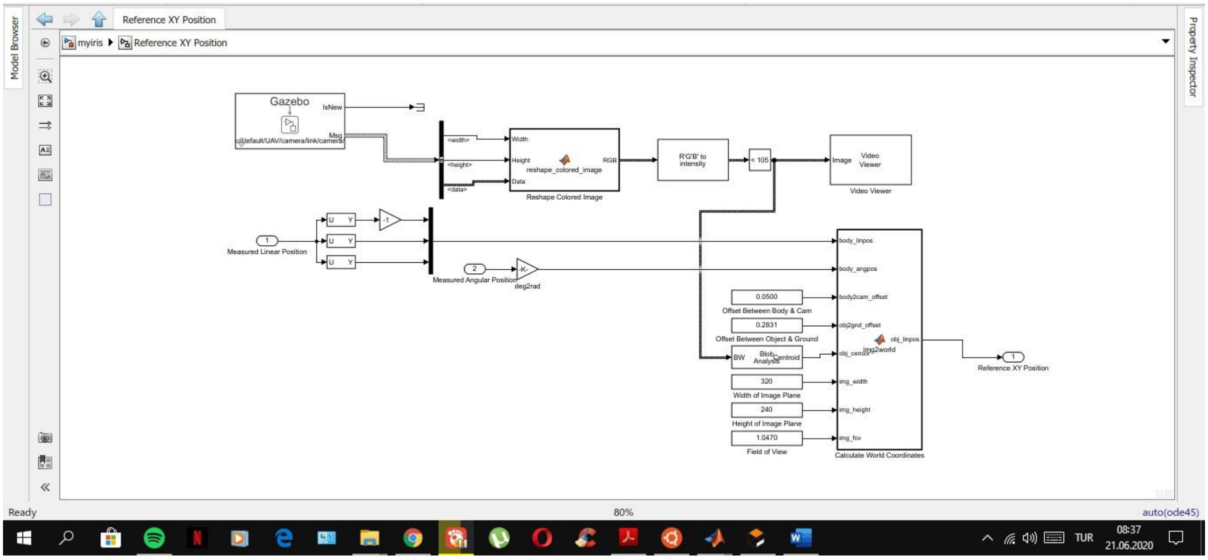
```
<!-- UAV: Unmanned Aerial Vehicle -->
<model name="UAV">
<include>
<uri>model://my_iris</uri>
<static>false</static>
</include>
<include>
<uri>model://my_camera</uri>
<static>false</static>
<pose>0 0 0.014 0 1.5708 0</pose>
</include>
<joint name="camera_joint" type="fixed">
<child>UAV::camera::link</child>
<parent>UAV::iris::base_link</parent>
</joint>
<pose>0 0 0.3 0 0 0</pose>
</model>
 <!-- Object -->
<model name="tote1">
<include>
<uri>model://grey_tote</uri>
<static>true</static>
</include>
<pose>0 0 0 0 0 1.5708</pose>
</model>
```



Final Look of the Simulink Model.

Weight Compensator block value is changed to 1.34.

Subsystems:



Reference XY Position Subsystem.

Reshape Colored Image block has the following code which is written from Matlab Command Panel:

```
function RGB = reshape_colored_image(Width,Height,Data)
channum = 3; % number of channels, R,G,B
RGB = zeros(Height, Width, channum, 'uint8');
if ~isempty(Data)&& ~isscalar(Data)
index = reshape(1:Width*Height*channum,[channum,Width*Height])';
RGB = reshape(Data(index), Width, Height, channum);
end
```

```
% result will be a height x width x channum image matrix
RGB = permute(RGB, [2 1 3]);
```

Calculate World Coordinates block has the following code which is written from Matlab Command Panel:

```
function obj_linpos = img2world(body_linpos,body_angpos,body2cam_offset,obj2gnd_offset,obj_cencor,img_width,img_height,img_fov)
% This function calculates world coordinates of object from its image coordinates
%
% INPUTS
% body_linpos: linear position of vehicle body
% body_angpos: angular position of vehicle body
% body2cam_offset: distance between body center of gravity and camera focal point
% obj2gnd_offset: distance between object top surface and ground surface
% obj_cencor: centroid coordinates of the objects in image plane
% img_width: width of image plane
% img_height: height of image plane
% img_fov: field of view
% Focal_Length = Width/(2*tan(Field_of_View/2))
%
% OUTPUTS
% obj_linpos: xy position of object in world coordinates
% calculate focal length
img_fl= img_width/(2*tan(img_fov/2));
% initialize object world coordinates
obj_linpos = zeros(2,1);
if ~(obj_cencor(1) == -1)
theta = -atan((obj_cencor(1)-0.5*img_width)/img_fl);
alfa = body_angpos(1) + theta;
obj_linpos(2) = body_linpos(2) + (abs(body_linpos(1))-body2cam_offset-obj2gnd_offset)*tan(alfa);
end
if ~(obj_cencor(2) == -1)
```

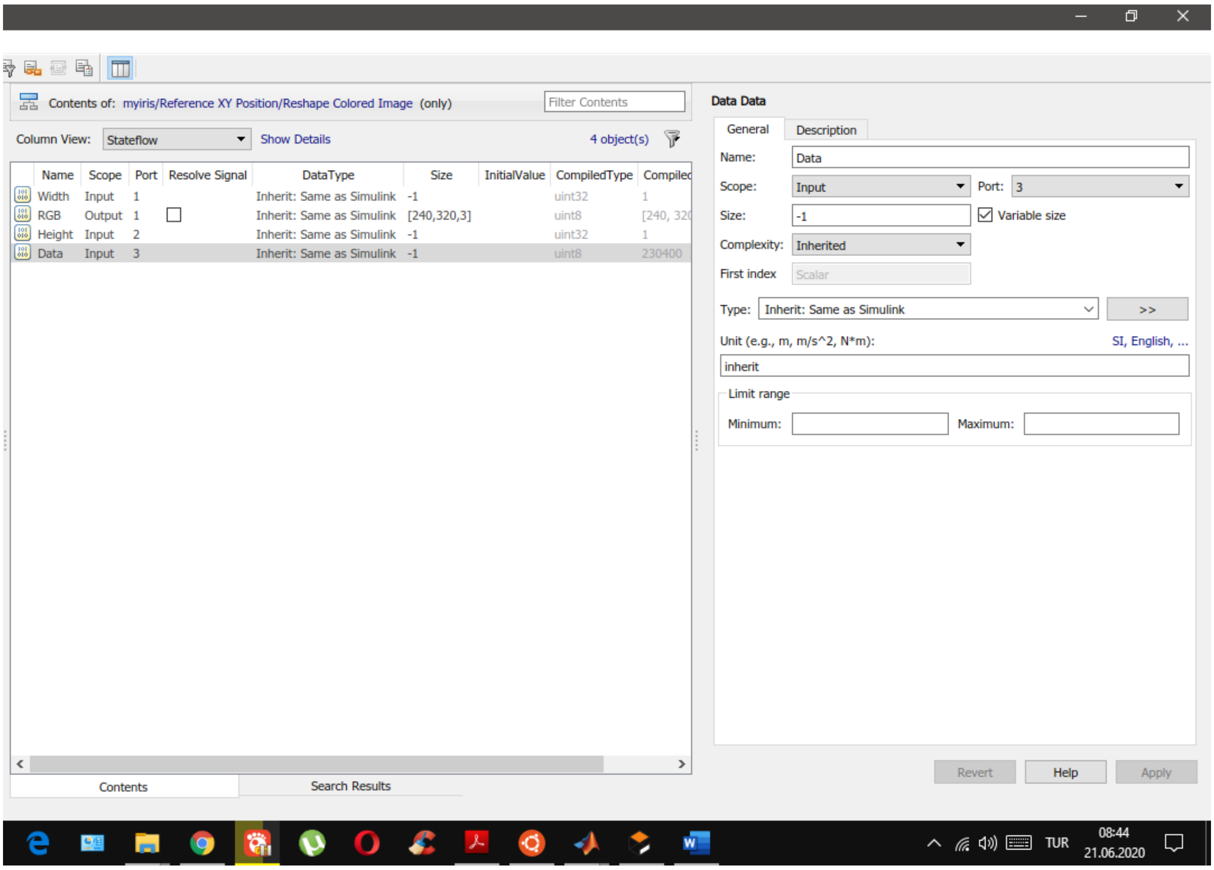
*beta = -atan((obj_cencor(2)-0.5*img_height)/img_fl);*

gamma = -body_angpos(2) + beta;

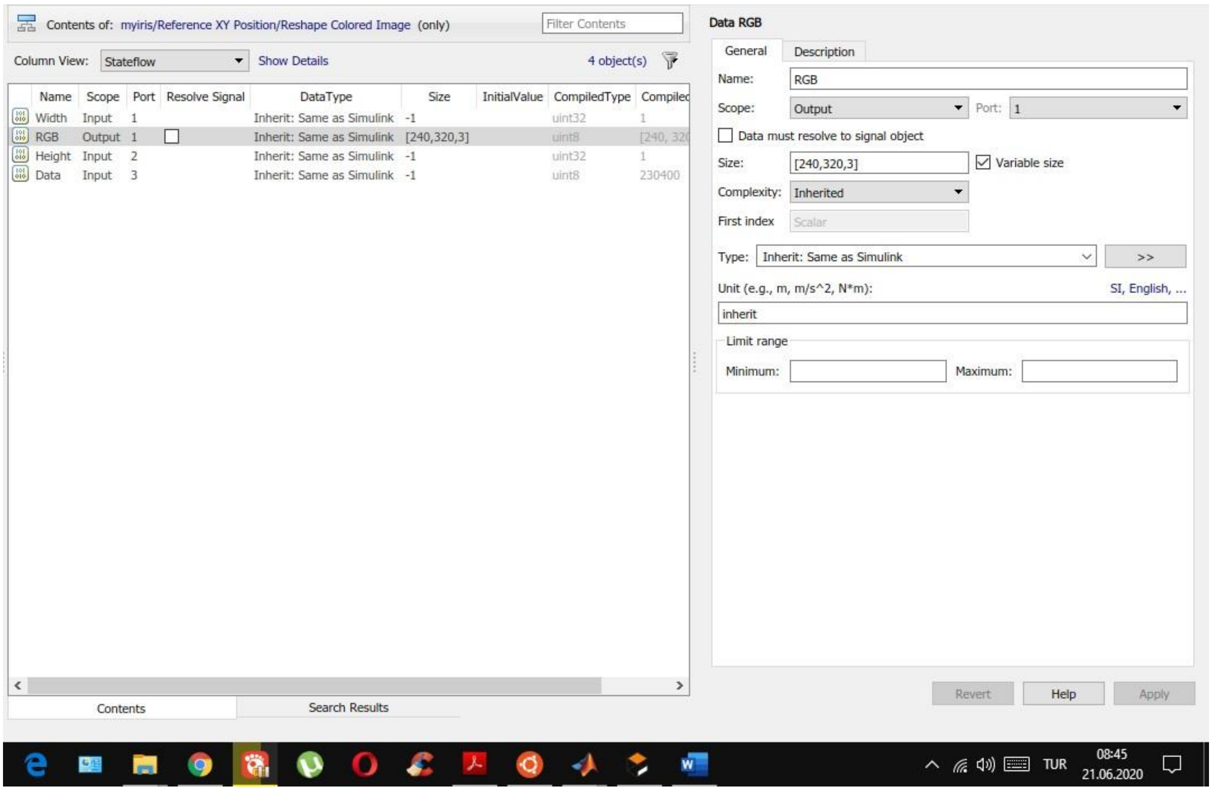
*obj_linpos(1) = body_linpos(3) + (abs(body_linpos(1))-body2cam_offset-obj2gnd_offset)*tan(gamma);*

end

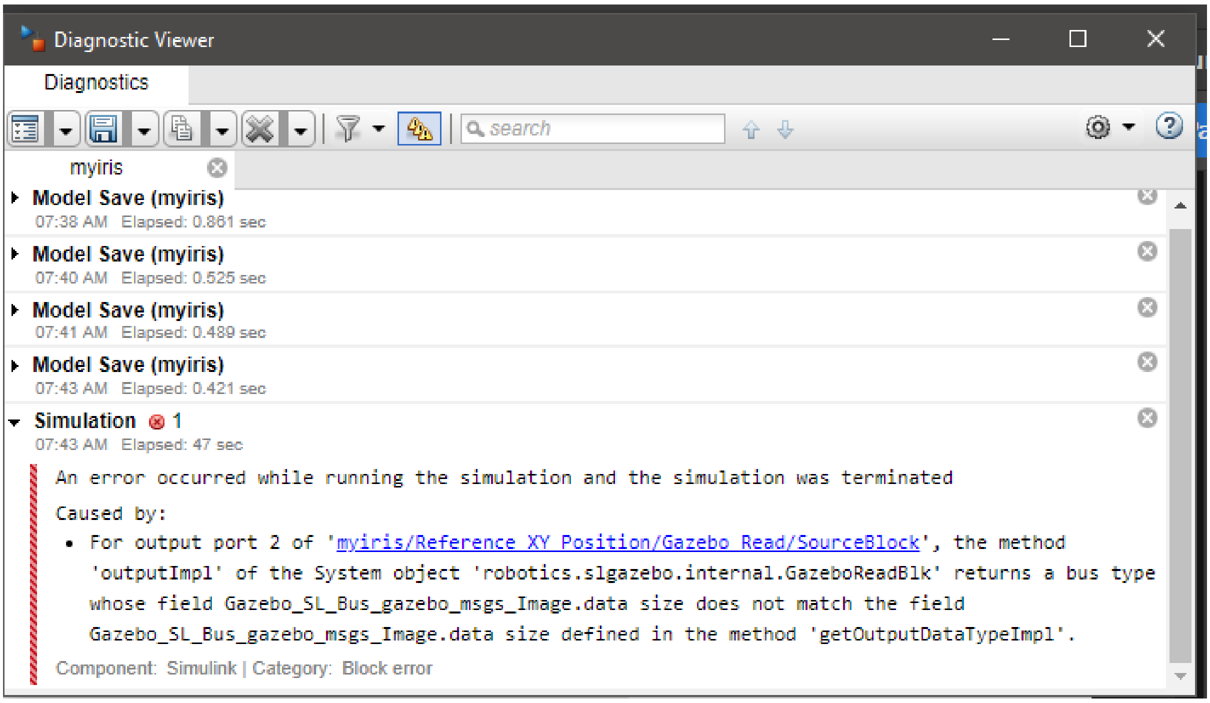
Final Settings of the Third Phase:



Changes In the Model Explorer.



Changes In the Model Explorer.



Error While Reading the Data From Camera.

This error is fixed by the following command written on the Matlab Command Panel:

>> Gazebo_SL_Bus_gazebo_msgs_Image

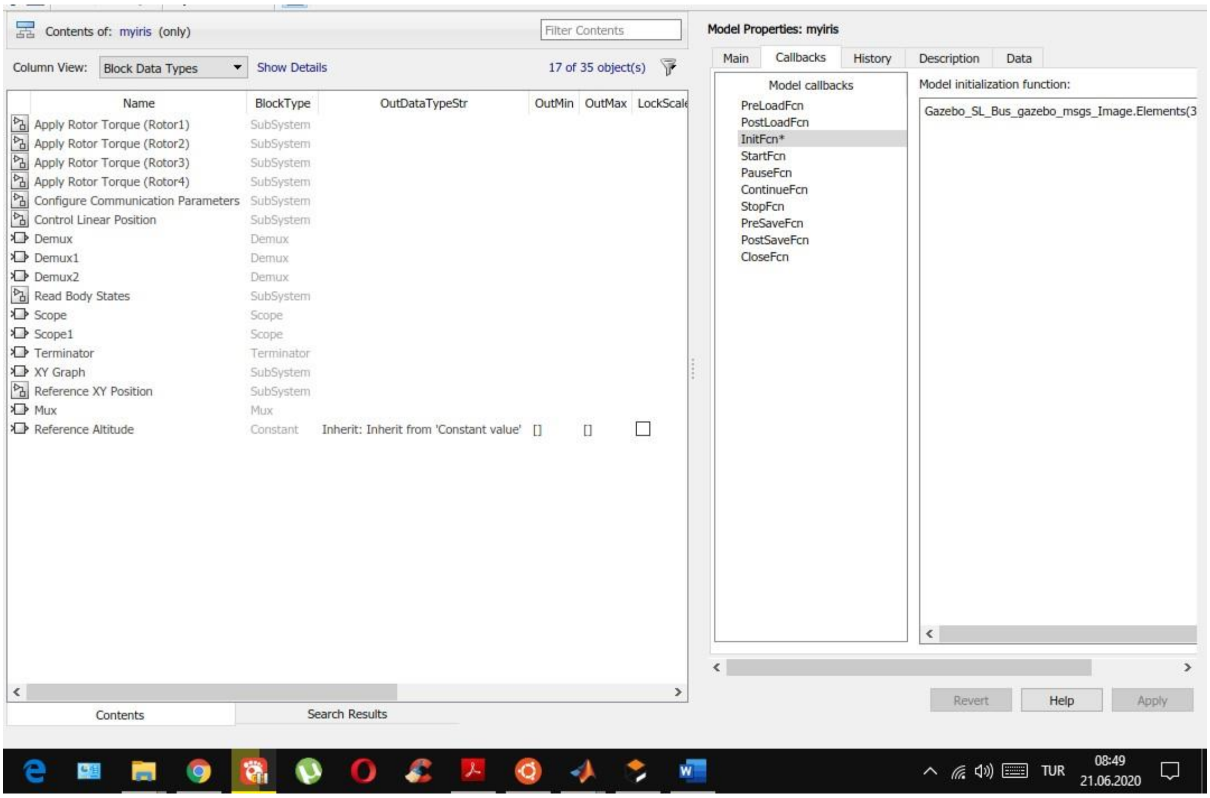
>> Gazebo_SL_Bus_gazebo_msgs_Image.Elements.Name

>> Gazebo_SL_Bus_gazebo_msgs_Image.Elements(3)

>> Gazebo_SL_Bus_gazebo_msgs_Image.Elements(3).Dimensions = 230400;

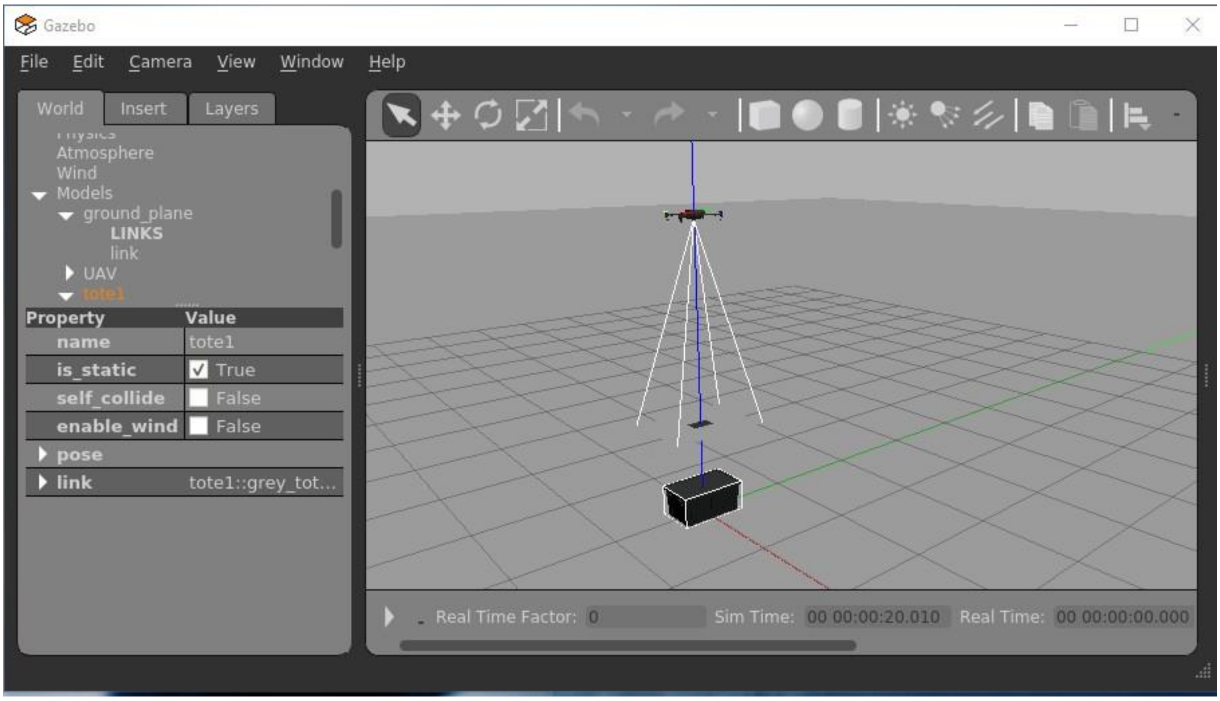
>> Gazebo_SL_Bus_gazebo_msgs_Image.Elements(3).DataType = 'uint8';

>> Gazebo_SL_Bus_gazebo_msgs_Image.Elements(3).DimensionsMode = 'Variable';

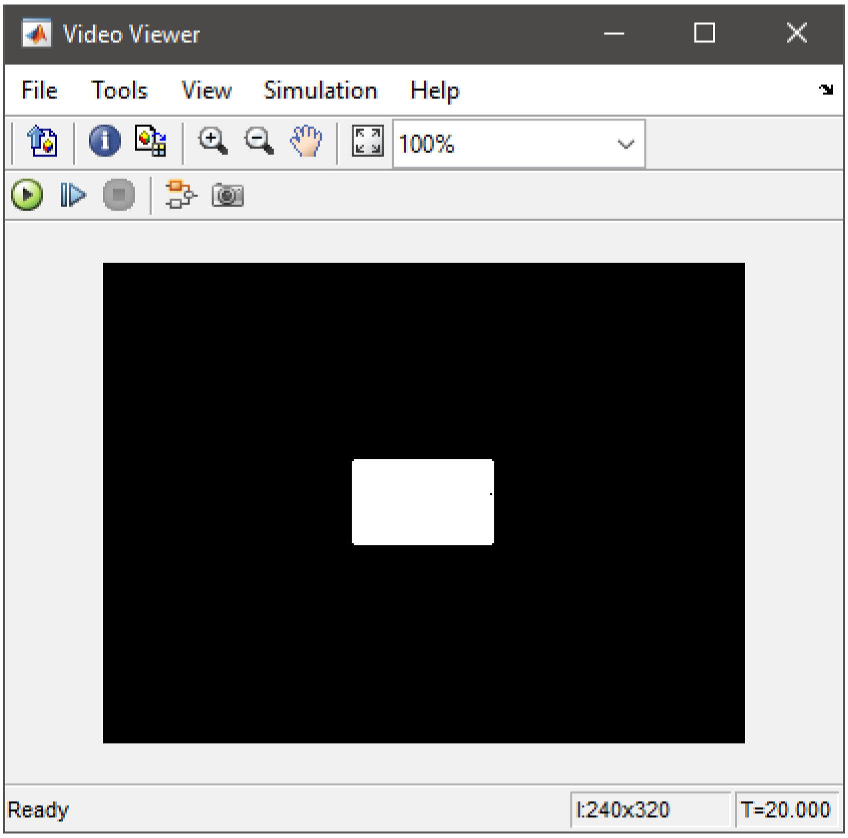


Permanent Solution for Error While Reading the Data From Camera. Deleted afterwords becaus it seems like there are some errors.

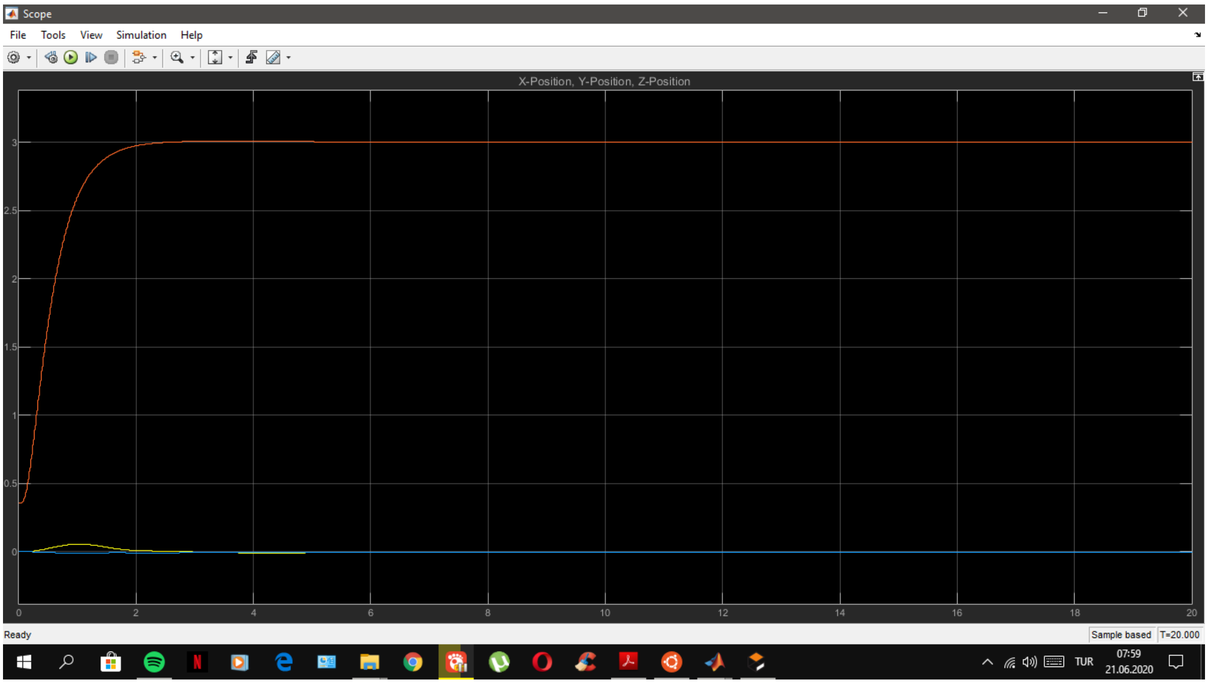
1.6. Results of the Step 15



Position of the Aircraft after 20 seconds.



Final look of the Video Viewer after 20 seconds.



X-Y-Z Positions of the Aircraft after 20 seconds.

2. Scenario of Aircraft Landing on a Moving Landing Ground

The goal of this phase is an aircraft will follow a moving landing ground for 30 seconds and after 30 seconds it will start to lower its altitude and land on landing ground.

2.1. Adding the Necessary Files Usign Windows Explorer

File opened from Ubuntu with the following code:

$ explorer.exe .

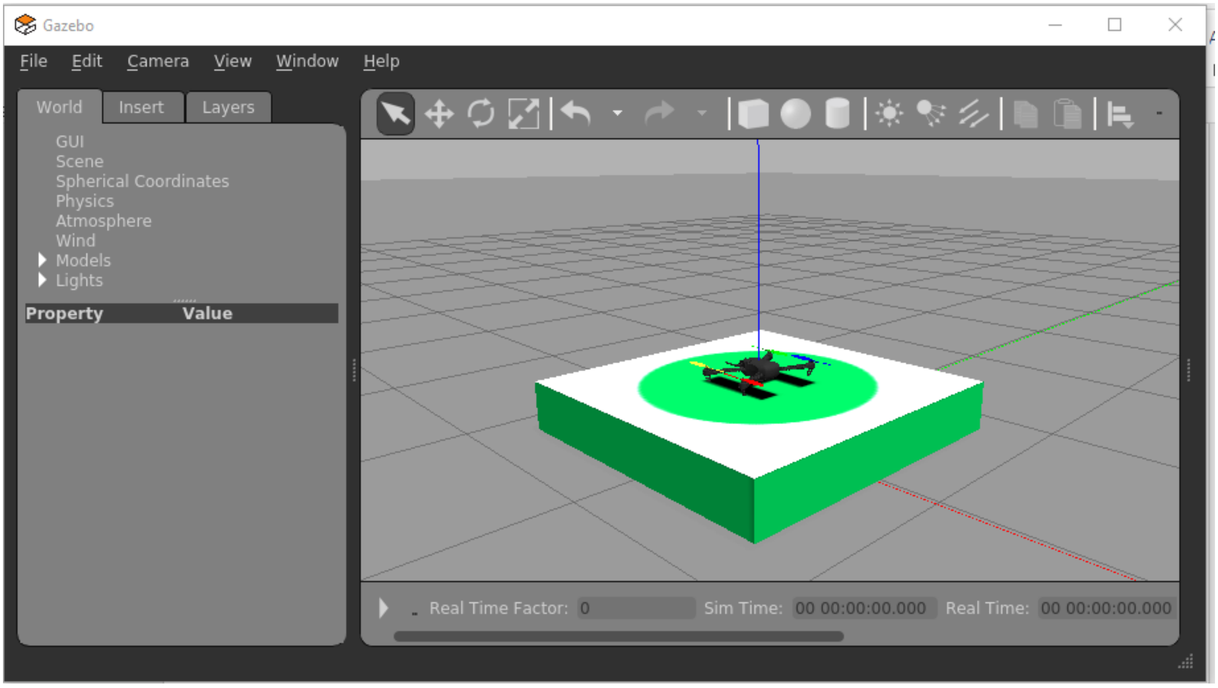
And the files pasted to \\wsl$\Ubuntu-18.04\home\

From now on "myheliport.world" world file will be used.

2.2. Opening the New World File and Checking the Models

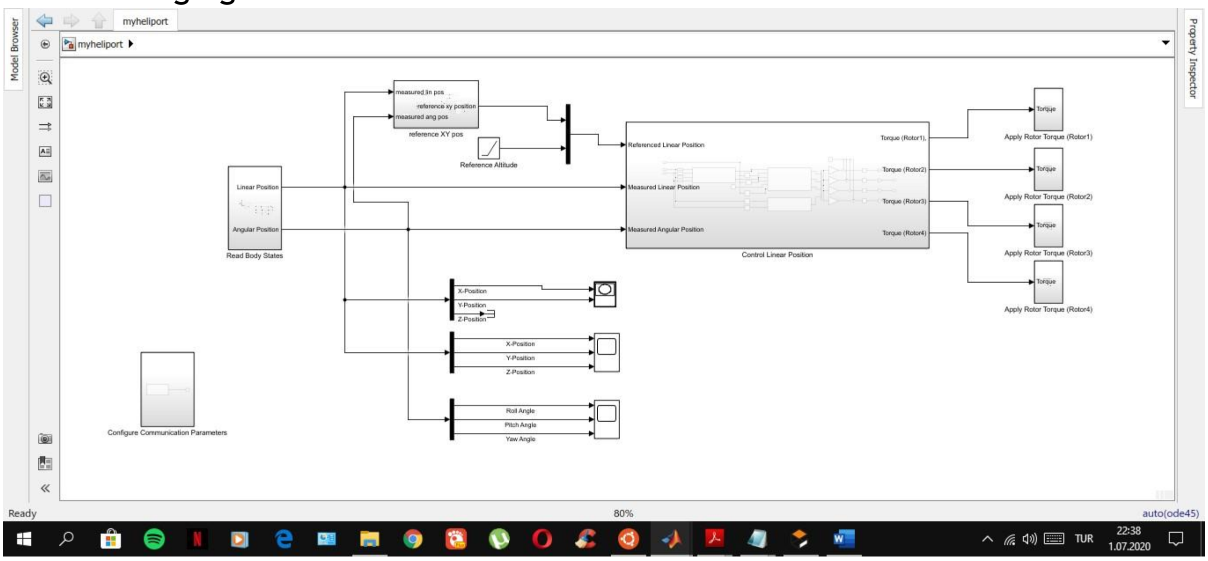
World file opened from Ubuntu using Gazebo with following code:

$ gazebo ~/.gazebo/worlds/myheliport.world --verbose



Initial Look of the World File.

2.3. Changing the Simulink File for Desired Scenario



Final look of the simulink file.

For flying at 3m altitude "Reference Altitude" constant block is deleted and a ramp block is added with following parameters.

Slope: -0.1

Start Time: 30

Initial Output: 3

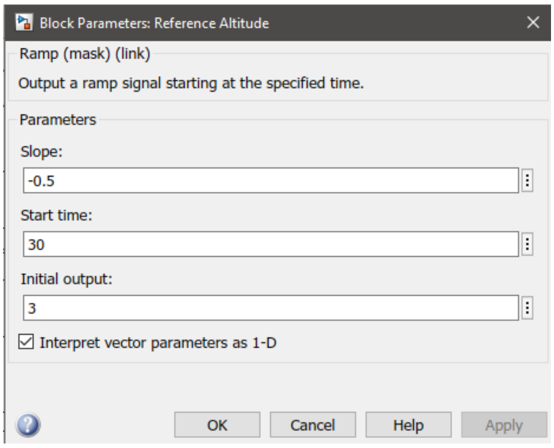
These are the initial values but aircraft lost sight of the landing ground as it get closed to the landing ground so the parameters are changed as below:

Slope: -0.5

Start Time: 30

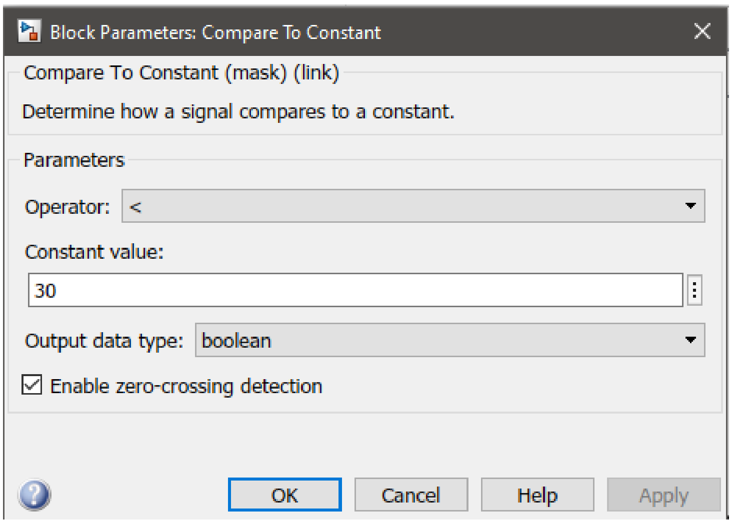
Initial Output: 3

With these parameters aircraft lands faster but makes a successful landing so these parameters are used.



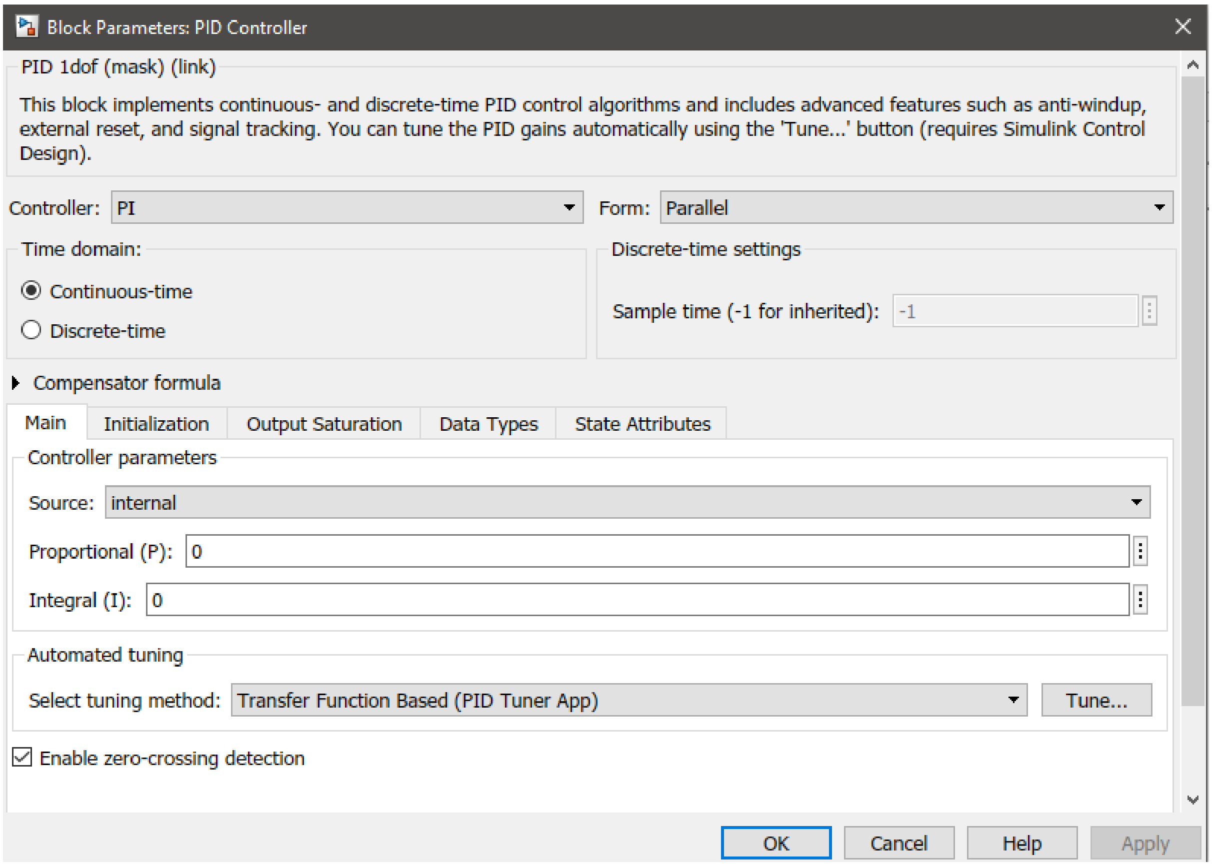
Final look of the Reference Altitude block.

The block parameter in the Reference XY Position /Compare to Constant s changed to 30 because if the aircraft falls behind to the landing ground camera senses the sides of the landing ground as the surface of the landing ground loses its stability. Initial value was 105.



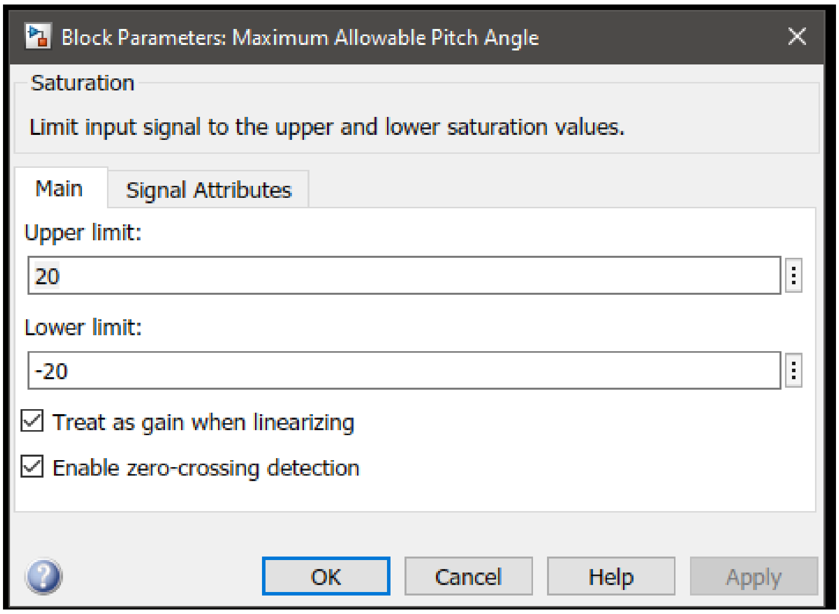
Final look of the Compare to Constant block.

2 PID Controller block are added to the Control Linear Position/XY Position Controller subsystem. Because as the aircraft gets closer to the surface of the landing ground it starts to lose sight of the landing ground so PID Controllers are added for faster response.

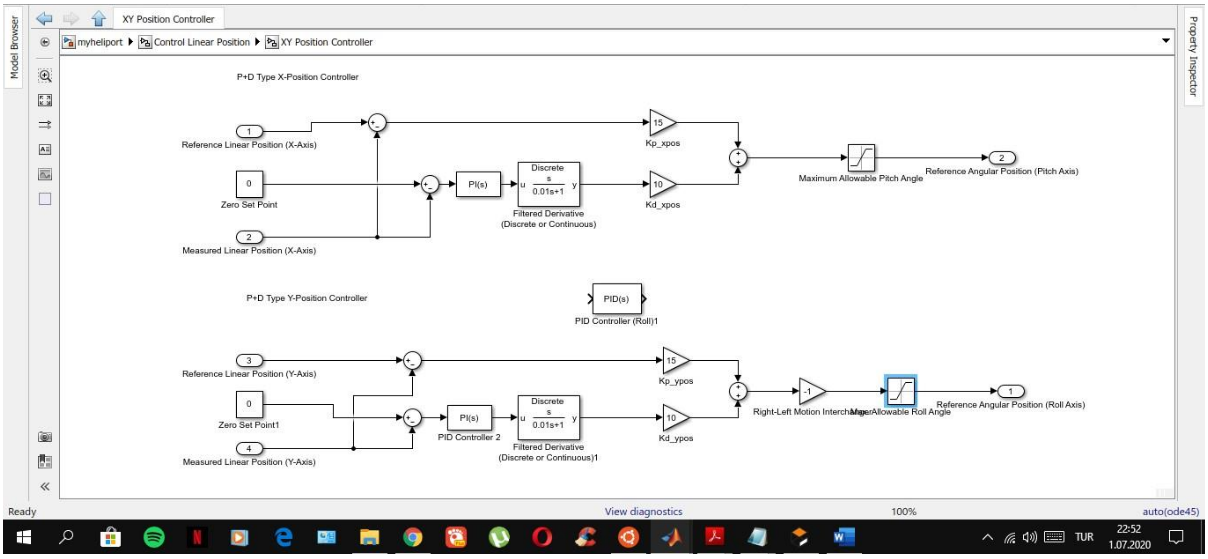


Parameters of the PID Controllers.

The parameters of the Maximum Allowable Pitch Angle and Max. Allowable Roll Angle are changed as below for better and faster reaction for aircraft.



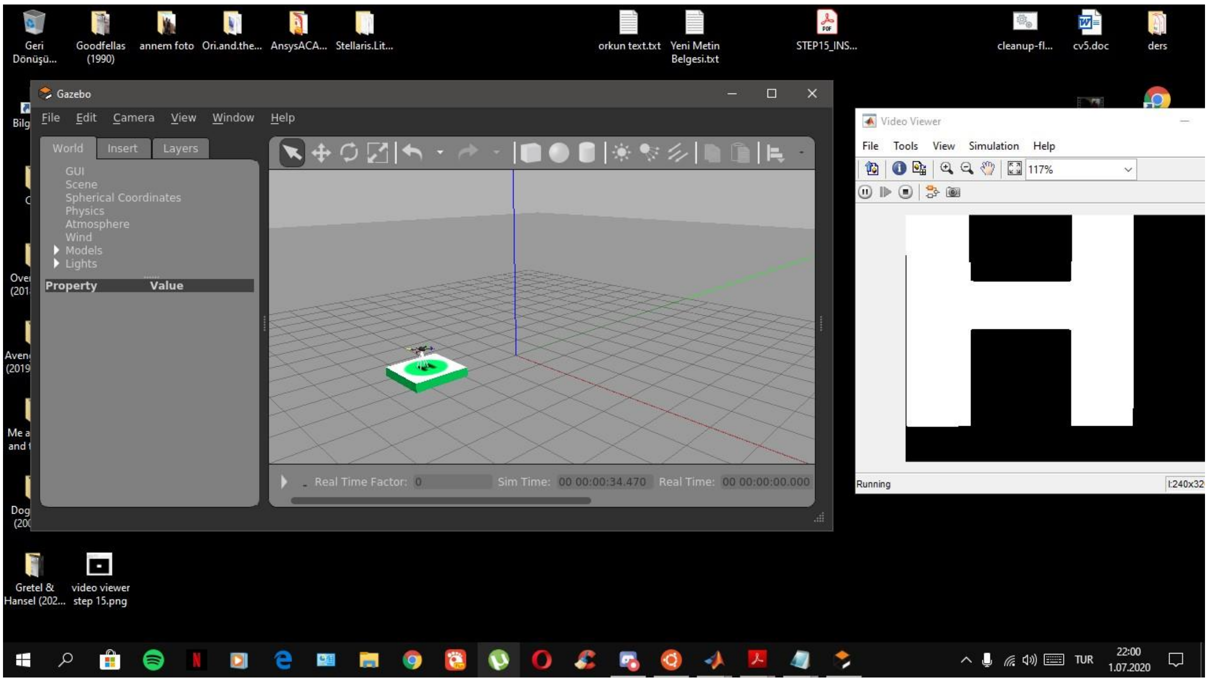
Both of blocks Maximum Allowable Pitch Angle and Max. Allowable Roll Angle blocks are set as above.



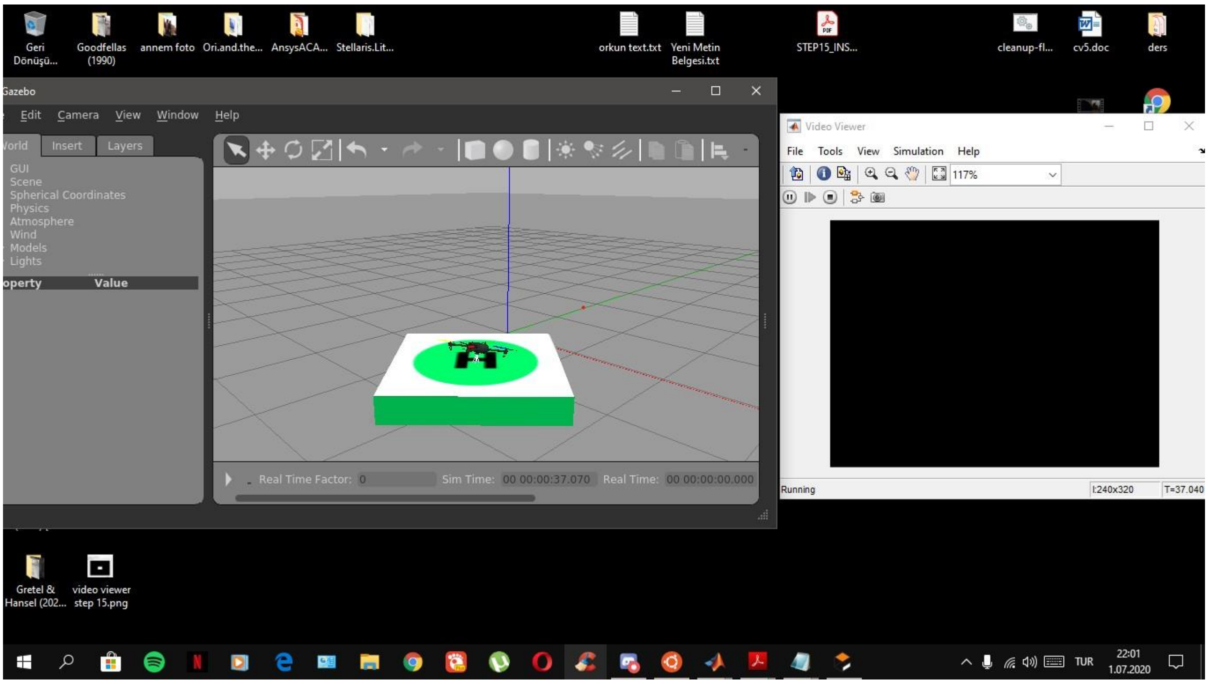
Final look of the XY Position Controller block.

2.4. Results of the Scenario

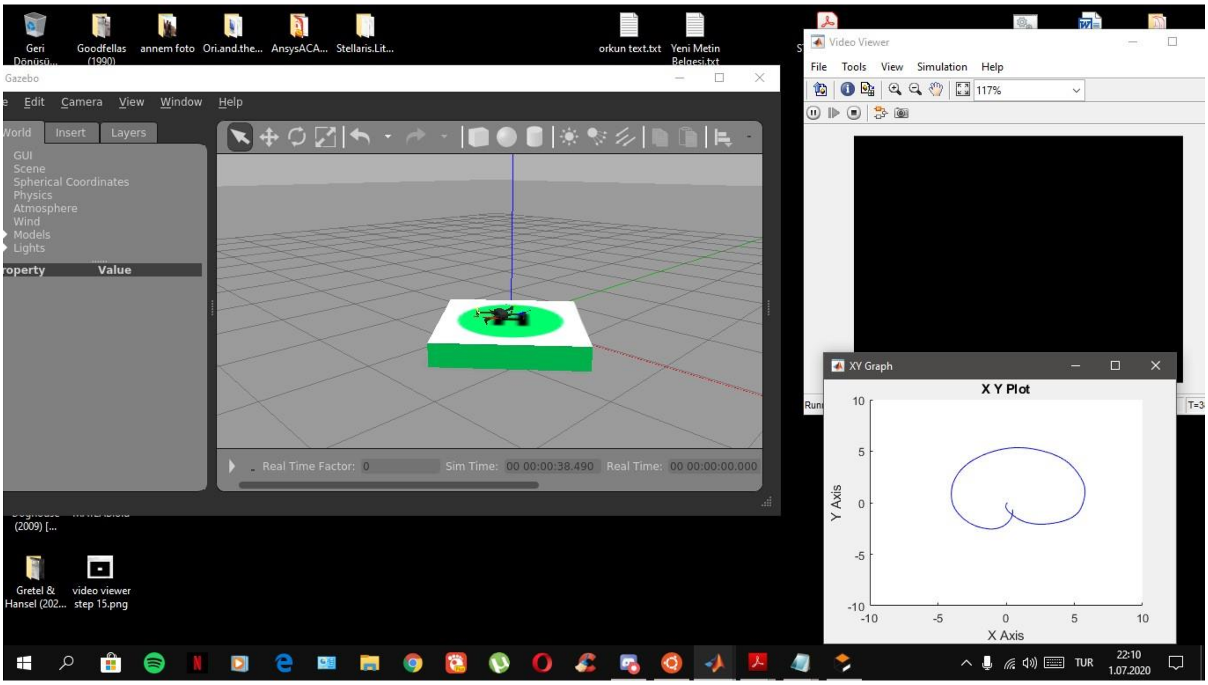
Aircraft landed on the landing ground as planned but when it touches the surface it does not stop the rotors so after 5 seconds it starts to move. So scenario is successfuly completed.



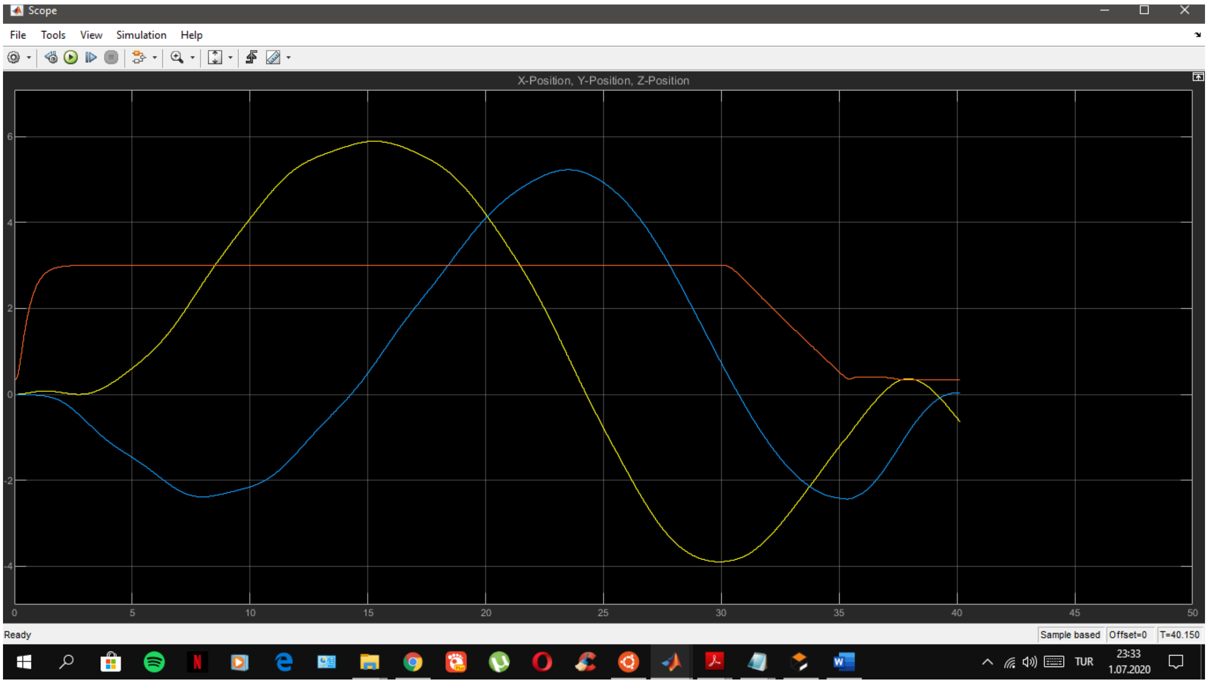
Look of the aircraft when it is close to the surface of the landing ground.



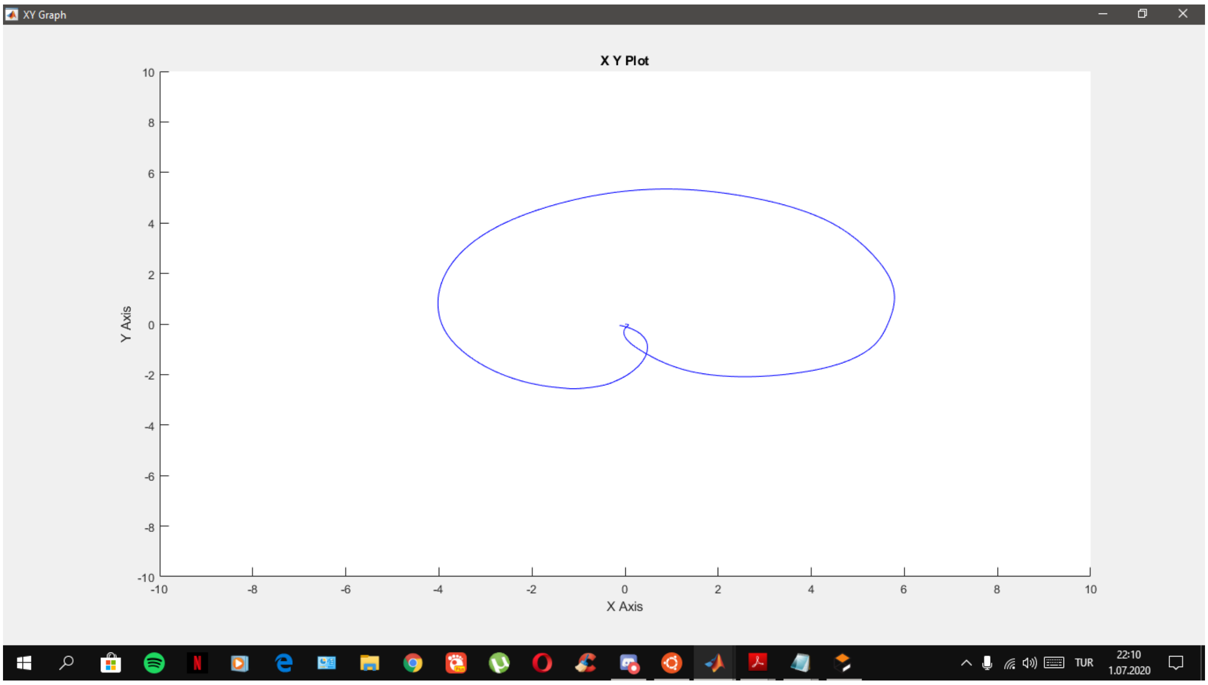
Look of the aircraft as it lands.



Look of the aircraft after it is landed.



X-Y-Z position graph of the aircraft.



X-Y position graph of the aircraft.

Roll-Pitch-Yaw angle graph of the aircraft.