Quadcopter Trajectory Planning and Control Simulation Performance Report

beileiz@cs.cmu.edu

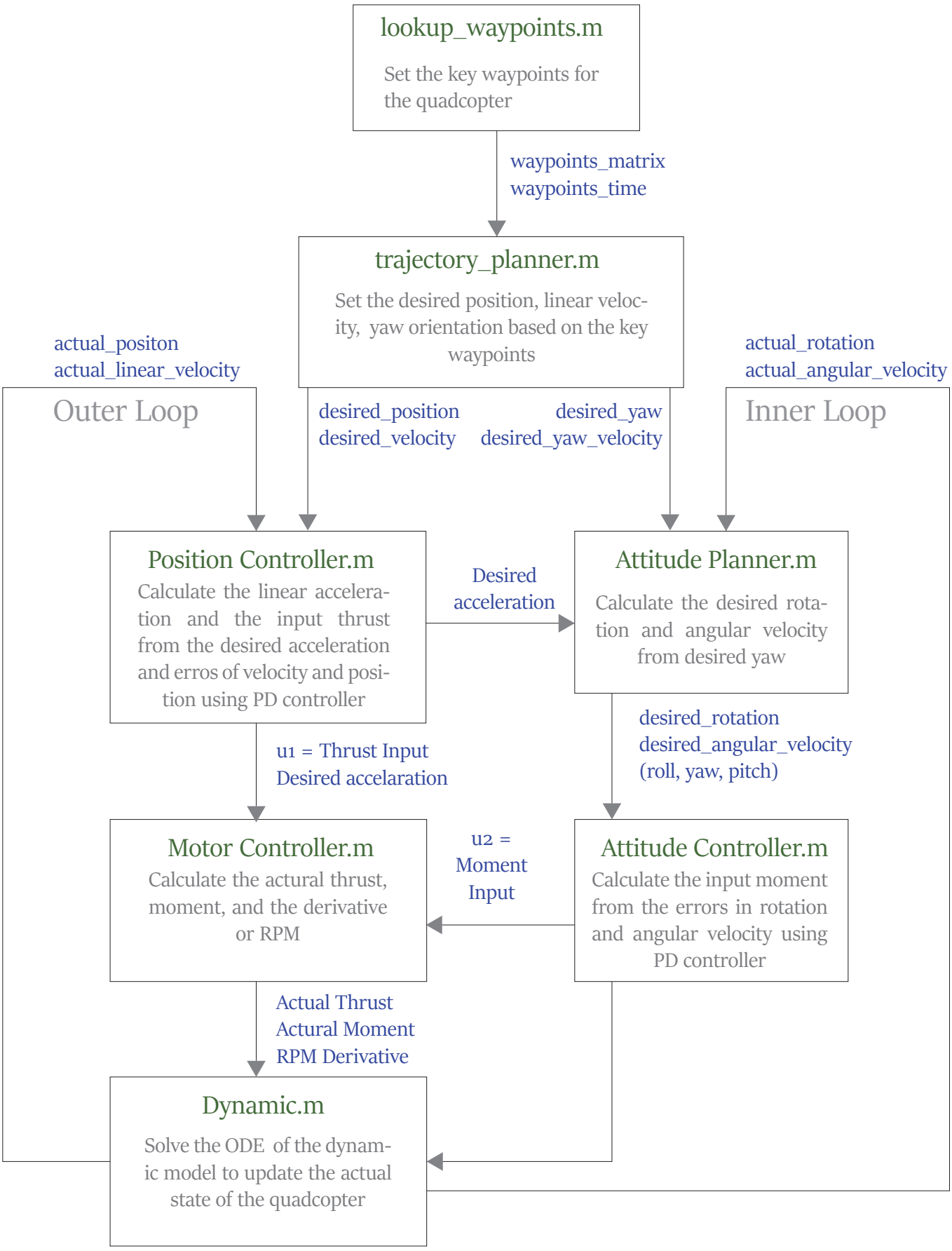
Contents

This report is about the performance of the similutation of various trajectory planning and different control methods of the quadcopter. The simulation is all done in Matlab.

Software Pipeline 2
Hover Performance with PD controller 3
Line Tracking Performanci with PD controller 8
State Machine with PD controller 13
Gain Selection and Turning 15
LQR Controller Design and Evaluation 22
Boundary Acceleration Tracking Generation 26
Elliptical Trajectory 30

Part 1 - Software Pipeline (PD controller)

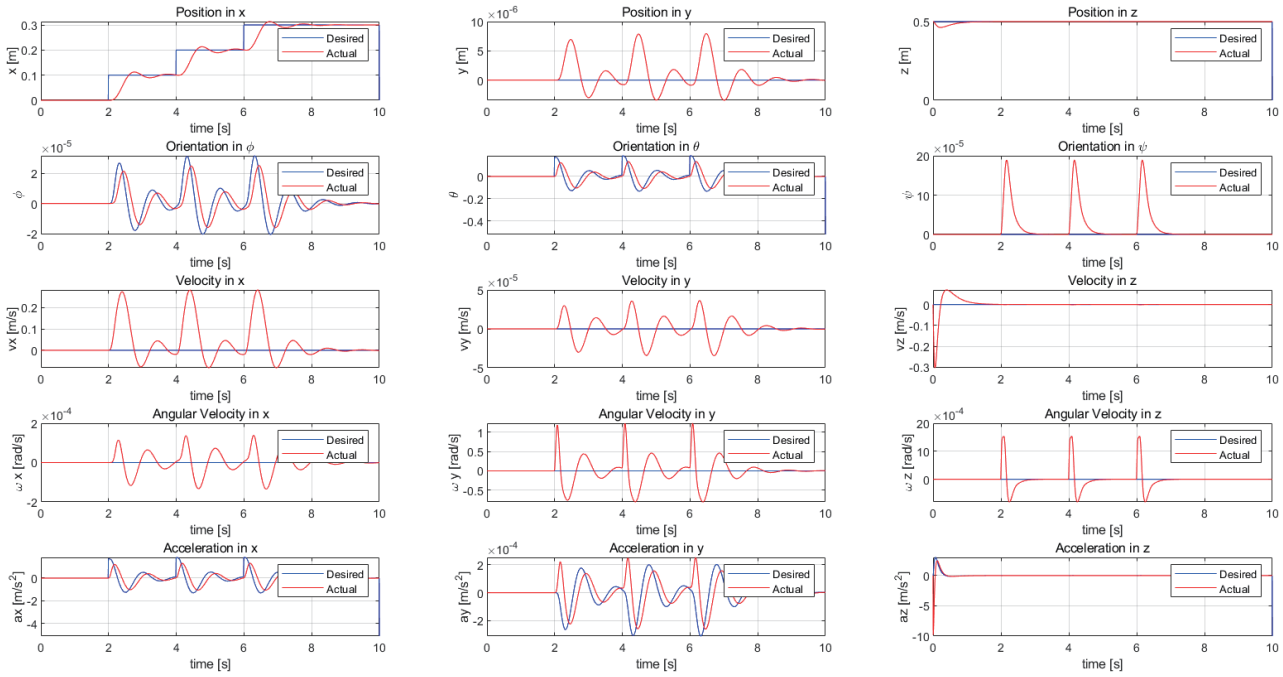
* Function Name in matlab * Data stream * Description



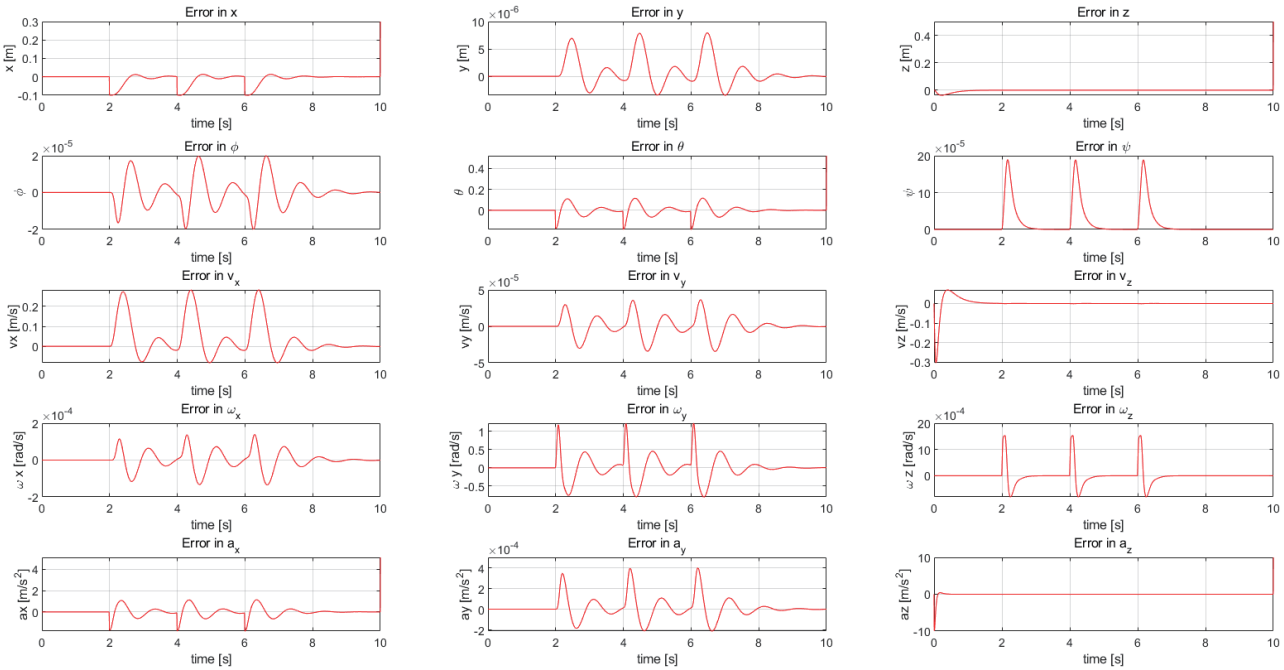
Part 2 - Hover Performance with PD controller

Implement a PD feedback controller to enable the robot to hover at a desired location (z = 0.5 m). Create a simulation scenario where the robot transitions between multiple way-points along the x-axis by specifying goals that increment by 10 cm in the x direction.

Kp1 = 17; Kd1 = 6.6; Kp2 = 17; Kd2 = 6.6; Kp3 = 20; Kd3 = 9;
Kpphi = 190; Kdphi = 30; Kptheta = 198; Kdtheta = 30; Kppsi = 80; Kdpsi = 17.88;

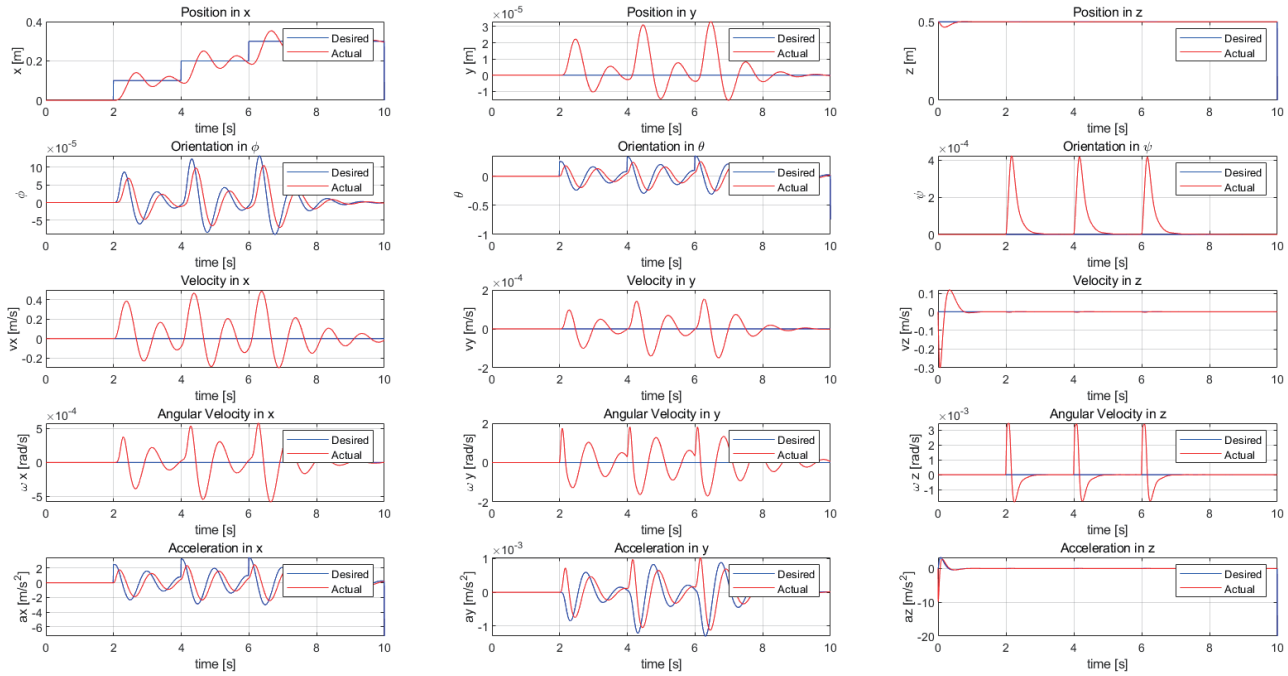


(Figure 1.1) The desired and actual state of the quadcopter in 0 to 10 seconds

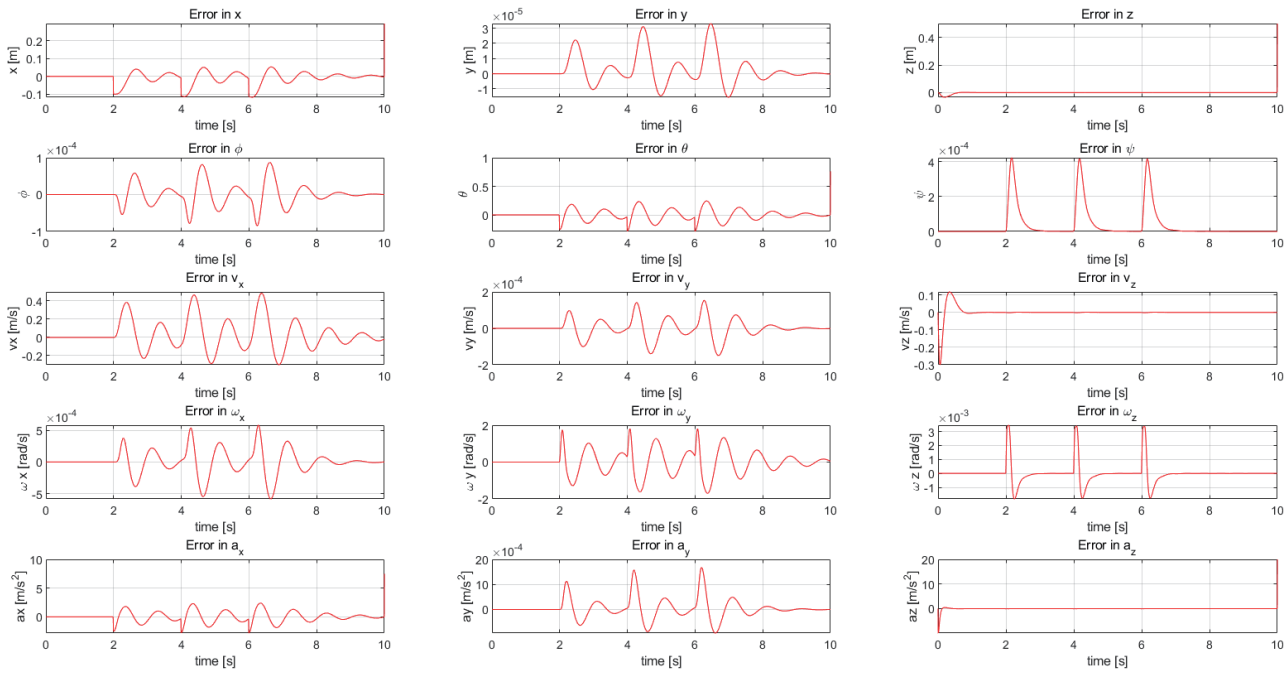


(Figure 1.2) The error between desired and actual state of the quadcopter in 0 to 10 seconds

Kp1 = 25; Kd1 = 6.6; Kp2 = 17; Kd2 = 6.6; Kp3 = 20; Kd3 = 9;
Kpphi = 190; Kdphi = 30; Kptheta = 198; Kdtheta = 30; Kppsi = 80; Kdpsi = 17.88;



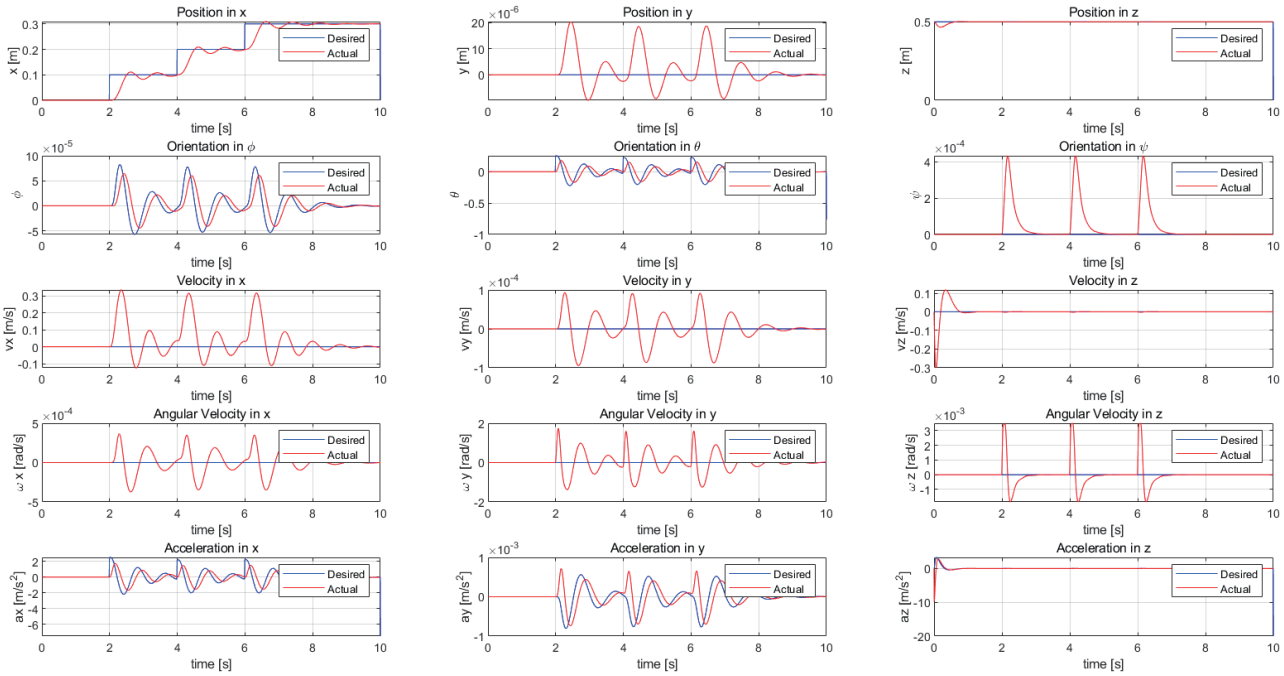
(Figure 1.3) The desired and actual state of the quadcopter in 0 to 10 seconds



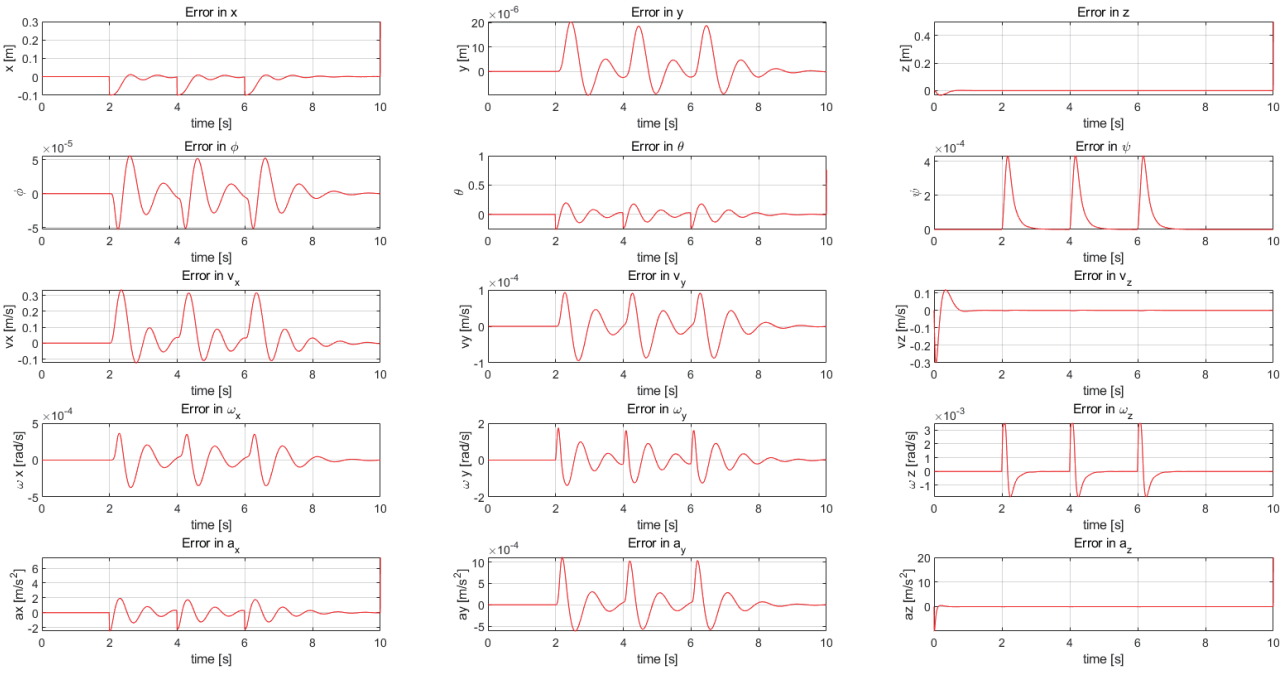
(Figure 1.4) The desired and actual state of the quadcopter in 0 to 10 seconds

From figure 1.1 and 1.2 we can see the quadcopter oscillates around the waypoints. Now we do an experiment of increasing the Kp1 from 17 to 25. We can see from the Figure 1.3 and Figure 1.4 that the rise time is a bit shorter. However, the overshoot is bigger and the settling time is longer. Now we adjust the Derivative controller to solve the problem

Kp1 = 25; Kd1 = 12; Kp2 = 17; Kd2 = 6.6; Kp3 = 20; Kd3 = 9;
Kpphi = 190; Kdphi = 30; Kptheta = 198; Kdtheta = 30; Kppsi = 80; Kdpsi = 17.88;



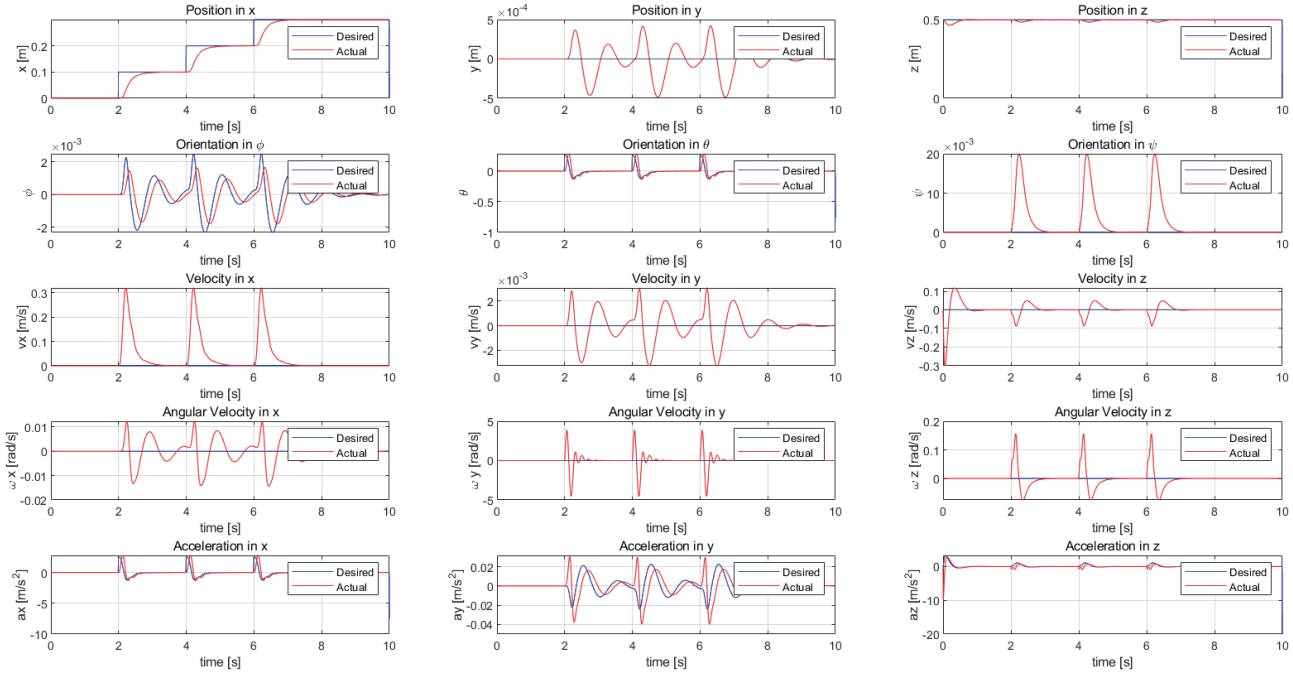
(Figure 1.5) The desired and actual state of the quadcopter in 0 to 10 seconds



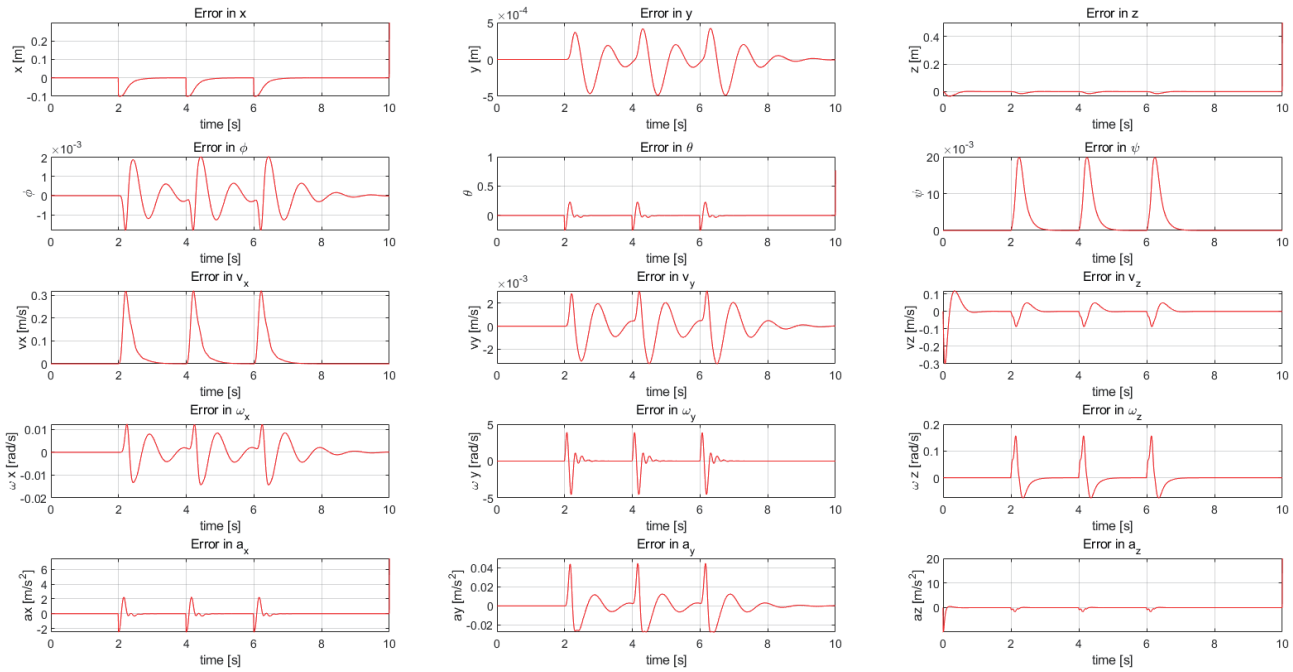
(Figure 1.6) The desired and actual state of the quadcopter in 0 to 10 seconds

To decrease the overshoot and shorten the settling time, we increase Kd1 from 6.6 to 12. It seems at Kp1 = 25, Kd1 =12, though we increase the rise time slightly, we lost the settling time and increase the overshoot slightly, it will depend on the situation that which one is more important.

Kp1 = 25; Kd1 = 12; Kp2 = 17; Kd2 = 6.6; Kp3 = 20; Kd3 = 9;
Kpphi = 190; Kdphi = 30; Kptheta = 600; Kdtheta = 40; Kppsi = 80; Kdpsi = 17.88;



(Figure 1.7) The desired and actual state of the quadcopter in 0 to 10 seconds



(Figure 1.8) The desired and actual state of the quadcopter in 0 to 10 seconds

From Figure 1.5 and 1.6 we can see a bad performance in theta orientation, to solve this problem, we increase both Kptheta to 600 and Kdtheta to 40. From the Figure 1.7 and 1.8 we can see it decreases the overshoot and settling time, as well as increases the rise time. As a side effect, it decreases the overshoot and settling time of x postion and slightly increase the error in z position.

Part 2 - Hover Performance with PD controller Qualitative Conclusion

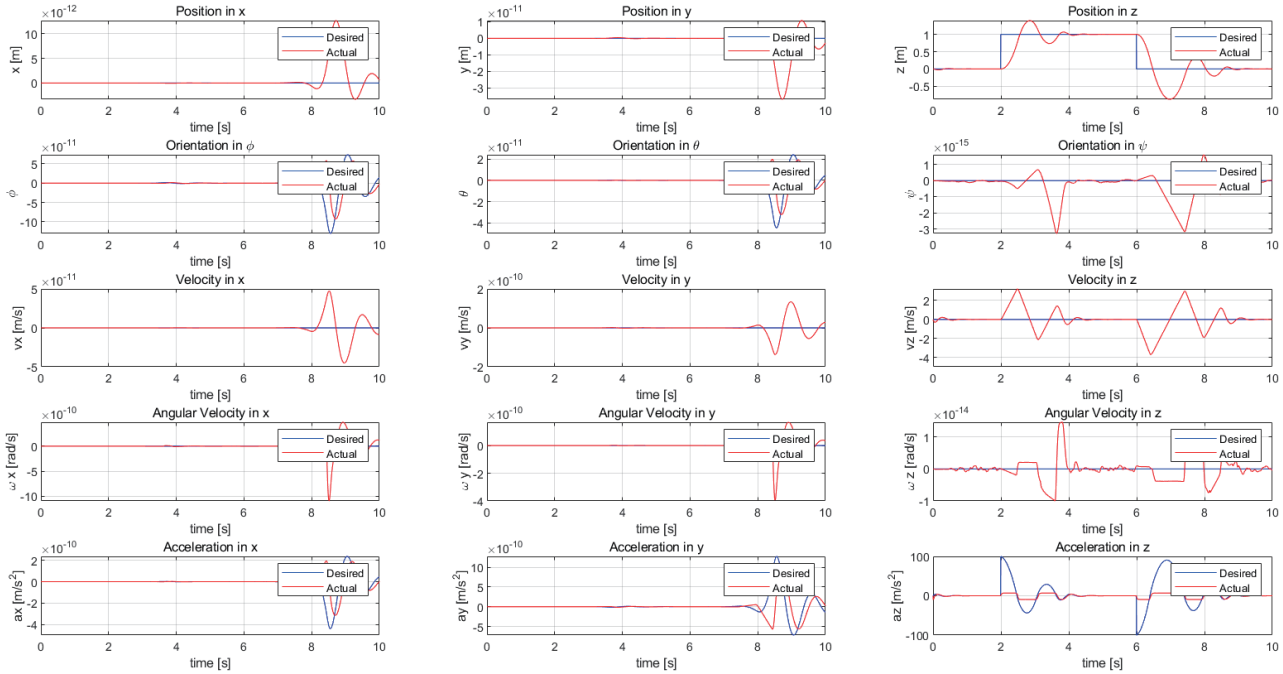
Kp1 = 25, Kd1 = 6.6 is great gain for the hover and moving task. Increasing Kp1 and Kd1 can help slightly increase the risetime while hurt the settling time and overshoot.

If we want to decrease the error in theta orientation. We need to make both Kptheata and Kdtheata bigger. The experienment shows Kptheta = 600, Kdtheta =40 is a great choice. As a side effect, it decreases the overshoot and settling time of x postion and slightly increase the error in z position.

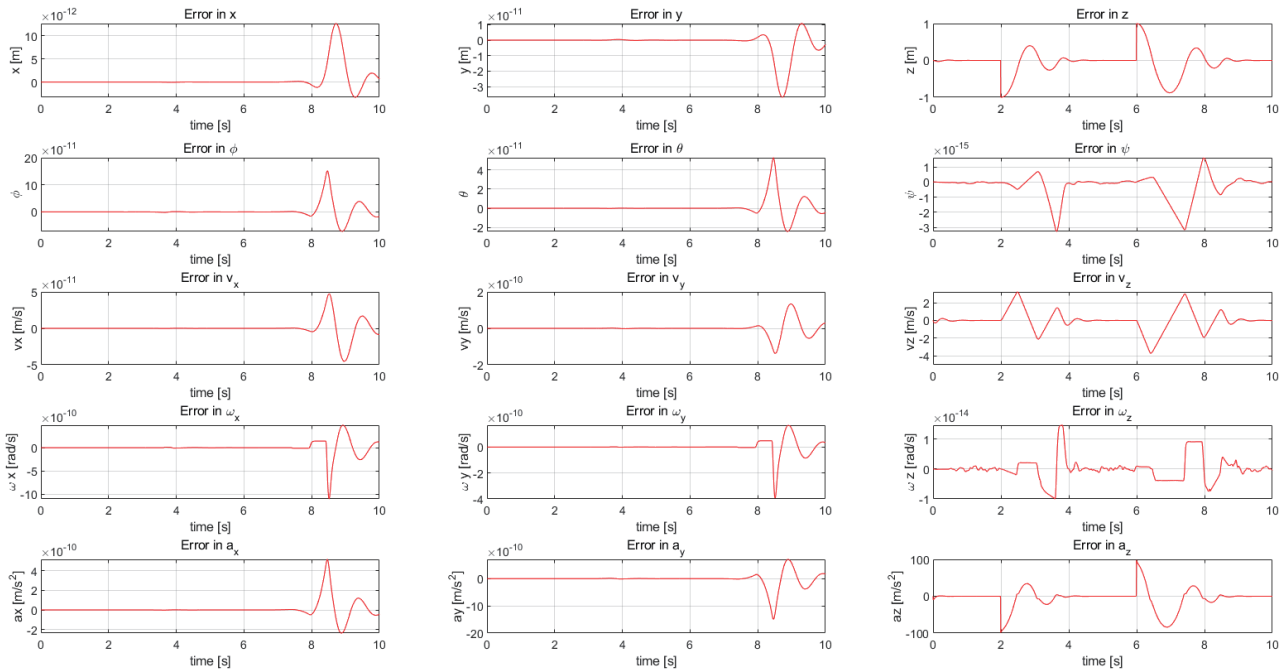
Part 3 - Line-tracking Performance with PD controller

Implement a PD line tracking controller to enable the robot to take-off from a starting location, go to a fixed height of 1.0 m, and return to the ground.

Kp1 = 17; Kd1 = 6.6; Kp2 = 17; Kd2 = 6.6; Kp3 = 100; Kd3 = 9;
Kpphi = 190; Kdphi = 30; Kptheta = 198; Kdtheta = 30; Kppsi = 80; Kdpsi = 17.88;



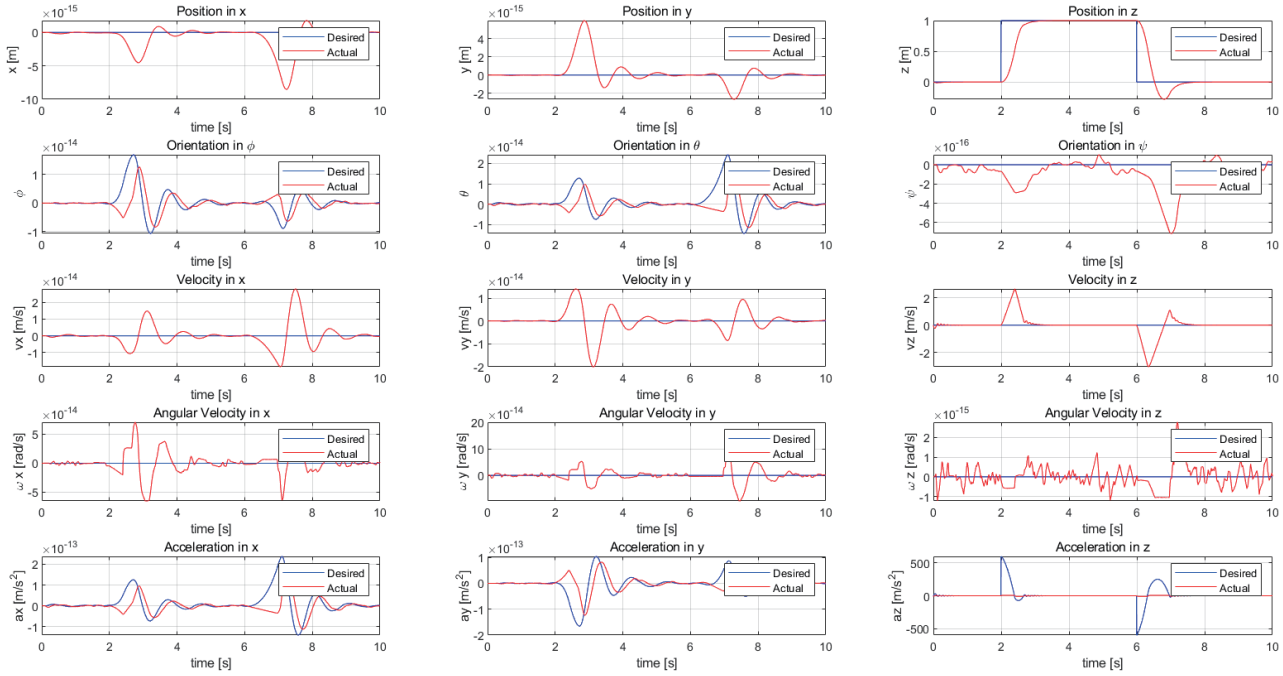
(Figure 2.2) The desired and actual state of the quadcopter in 0 to 10 seconds



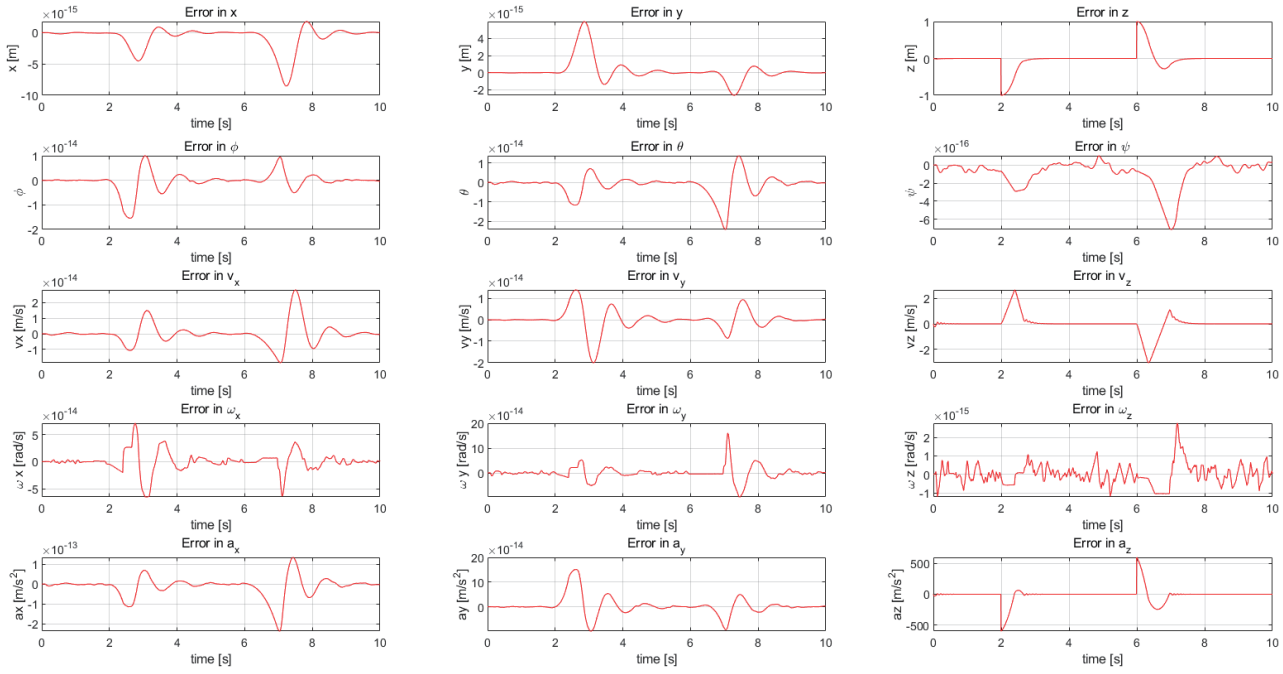
(Figure 2.2) The error between desired and actual state of the quadcopter in 0 to 10 seconds

From figure 2.1 and 2.2 we can see the quadcopter oscillates around the waypoints. There is large over shoot and settling time with Kp3 = 100, Kd3 = 9

Kp1 = 17; Kd1 = 6.6; Kp2 = 17; Kd2 = 6.6; Kp3 = 600; Kd3 = 120;
Kpphi = 190; Kdphi = 30; Kptheta = 198; Kdtheta = 30; Kppsi = 80; Kdpsi = 17.88;



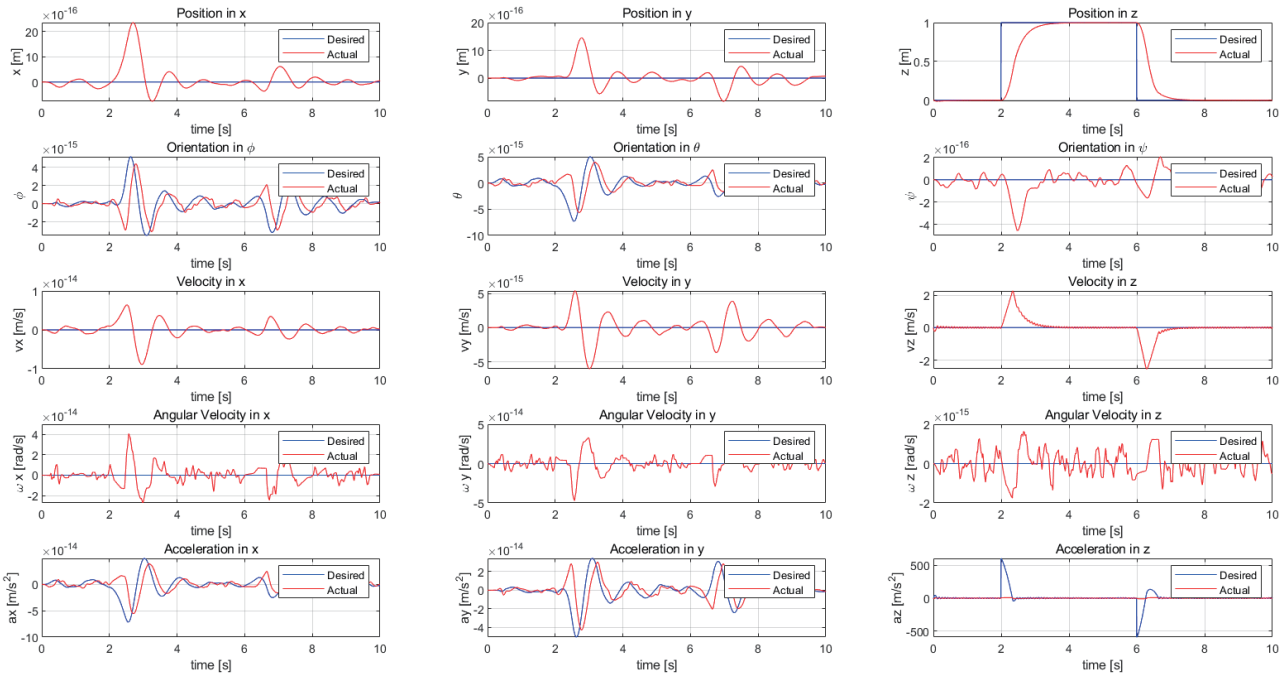
(Figure 2.3) The desired and actual state of the quadcopter in 0 to 10 seconds



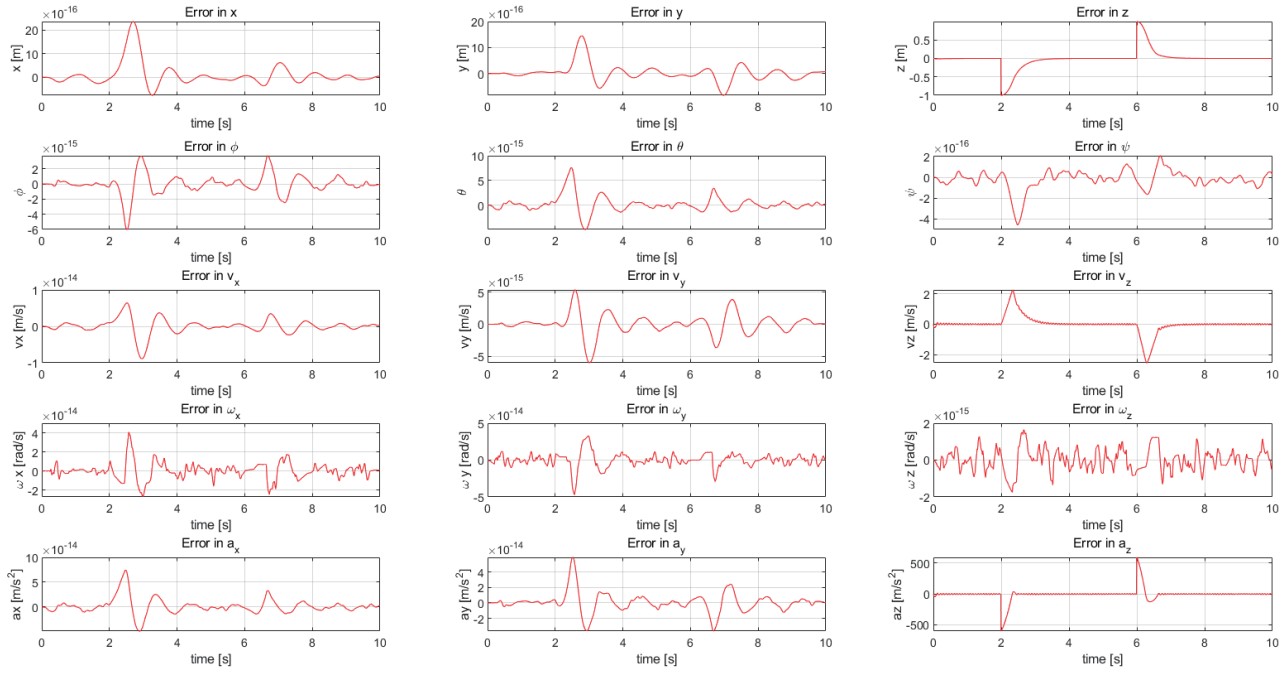
(Figure 2.4) The error between desired and actual state of the quadcopter in 0 to 10 seconds

From figure 2.3 and 2.4 we can see, by turn on the Kp3 to 600, Kd3 = 200, we effectively reduce the oscillation around the way points

Kp1 = 17; Kd1 = 6.6; Kp2 = 17; Kd2 = 6.6; Kp3 = 600; Kd3 = 180;
Kpphi = 190; Kdphi = 30; Kptheta = 198; Kdtheta = 30; Kppsi = 80; Kdpsi = 17.88;



(Figure 1.1) The desired and actual state of the quadcopter in 0 to 10 seconds



(Figure 2.2) The error between desired and actual state of the quadcopter in 0 to 10 seconds

From figure 2.5 and 2.6 we can see, by increase Kd3 to 180, we succesfully eliminate the overshoot of z position and let the robot converge to the end point smoothly

Part 3 - Line-tracking Performance with PD controller Qualitative Conclusion

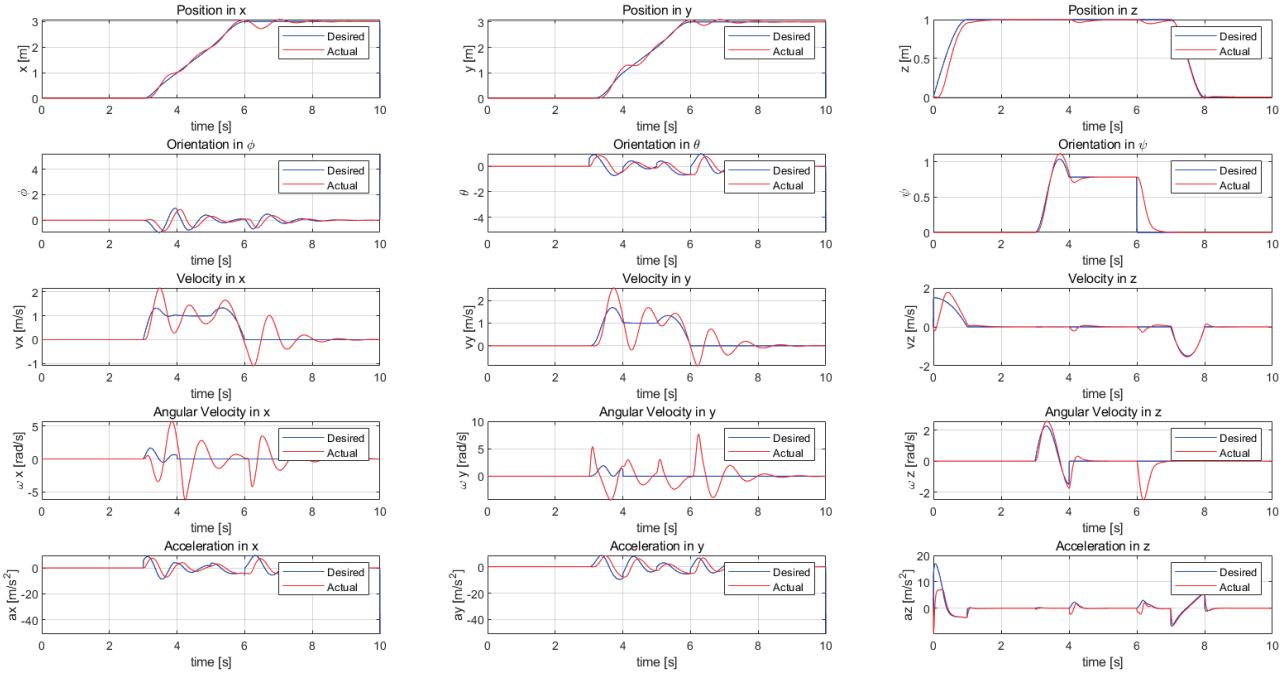
Kp3 = 600, Kd3 =180 is great gain for simple taking off and landing task. For similar task, we can make Both Kp3 and Kd3 large to ensure the performance, it won't hurt the performance of other postion and orientation

Part 4 - State Machine with PD controller

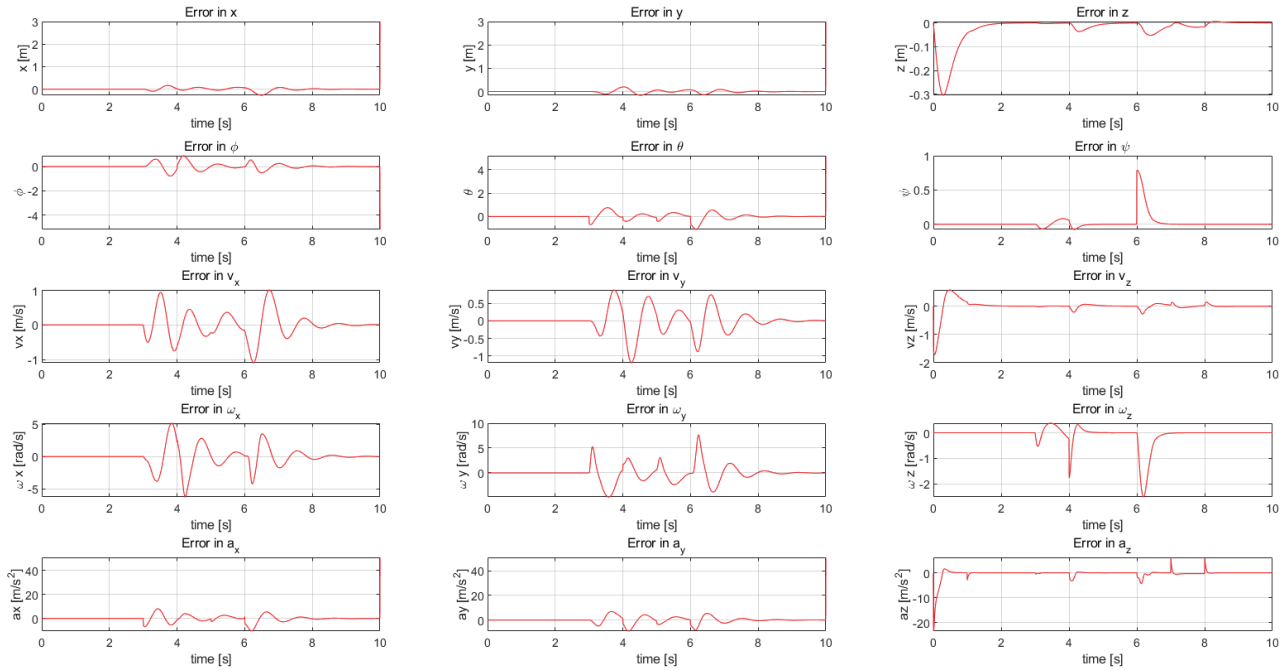
A detailed state machine that enables the platform to takeoff to a pre-specified height, hover, track trajectories, and land.

• The system begins in an idle state generating no (or null) control inputs.
• A simulation trial begins by transitioning into a takeoff state that consists of a trajectory tracking controller from the current robot state to a desired hover state (fixed pose). When the robot approaches the desired hover state (based on the error between current and desired pose), the robot transitions into a hover mode.
• The platform remains in hover mode for a small amount of time (e.g., 5 s) before transitioning into a tracking mode.
• The platform tracks a specified trajectory and upon completion transitions into the hover mode.
• After remaining in hover mode for a brief period of time, the robot transitions into land mode and begins a descent to the ground.

Kp1 = 17; Kd1 = 6.6; Kp2 = 17; Kd2 = 6.6; Kp3 = 20; Kd3 = 9;
Kpphi = 190; Kdphi = 30; Kptheta = 198; Kdtheta = 30; Kppsi = 80; Kdpsi = 17.88;



(Figure 3.1) The desired and actual state of the quadcopter in 0 to 10 seconds



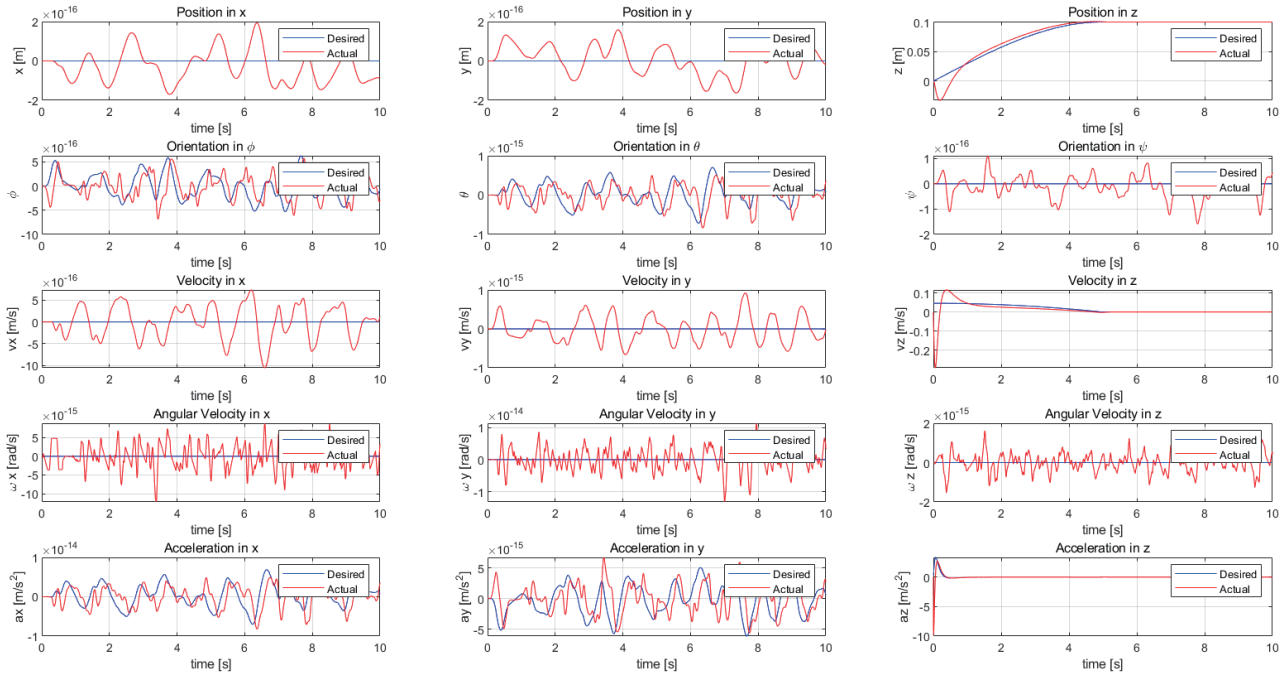
(Figure 3.2) The error between desired and actual state of the quadcopter in 0 to 10 seconds

From Figure 3.1 we can see the quadcopter taking off at 0s, hovering in the air in 0.5s-2 When it receives the command to move at 2s, it stays in the hover model for another 1s and starts to move at 3s. The linear velocity of both x and y is 1m/s. After arrival at (3m, 3m) at 6 s, the quadcopter switches back to the hover model for another 1s. It starts landing at 7s and successfully lands at 9s. It stays idle for the another 2s. Figure 3.2 shows the corresponding errors of this state machine.

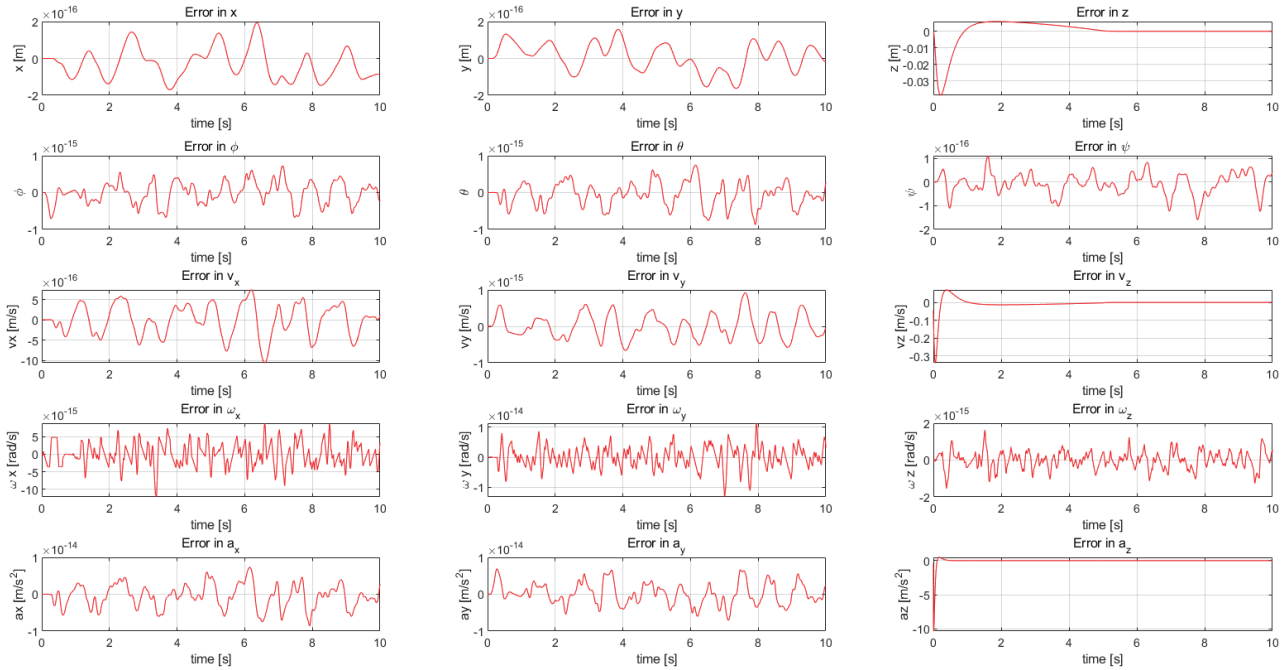
Part 5 - Gain Selection and Turning

After commanding the robot to takeoff and hover, let it tracks a single waypoint at [0, 0, 0.1] m (zero velocity). Choose the best gains for this movement and repeat it with another condition of 15 degree heading in yaw to see different performance.

Kp1 = 17; Kd1 = 6.6; Kp2 = 17; Kd2 = 6.6; Kp3 = 20; Kd3 = 9;
Kpphi = 190; Kdphi = 30; Kptheta = 198; Kdtheta = 30; Kppsi = 80; Kdpsi = 17.88;



(Figure 4.1) The desired and actual state of the quadcopter in 0 to 10 seconds



(Figure 4.2) The error between desired and actual state of the quadcopter in 0 to 10 seconds

From Figure 4.1 we can see the steady state value of z position is 0.1m. The steady stae value of z velocity is 0m/s

Kp1 = 17; Kd1 = 6.6; Kp2 = 17; Kd2 = 6.6; Kp3 = 20; Kd3 = 9;
Kpphi = 190; Kdphi = 30; Kptheta = 198; Kdtheta = 30; Kppsi = 80; Kdpsi = 17.88;

z position performace:		z velocity performace:	
90%RiseTime:	3.3437	90%RiseTime:	2.8073e-13
5%SettlingTime:	4.8472	5%SettlingTime:	5.0737
SettlingMin:	0.0900	SettlingMin:	-0.2930
SettlingMax:	0.1007	SettlingMax:	0.1155
Overshoot:	0.7428	Overshoot:	9.5795e+12
Peak:	0.1007	Peak:	0.2930
PeakTime:	4.7350	PeakTime:	0.0550

(Figure 4.3) The per formance of selected gains in 0 degree yaw

From Figure 4.3 we can see the bad performance in z velocity of the current gains. Though the risetime is short, the overshoot is huge while the settling time is to long

Kp1 = 17; Kd1 = 6.6; Kp2 = 17; Kd2 = 6.6; Kp3 = 20; Kd3 = 50;
Kpphi = 190; Kdphi = 30; Kptheta = 198; Kdtheta = 30; Kppsi = 80; Kdpsi = 17.88;

z position performace:		z velocity performace:	
90%RiseTime:	2.7698	90%RiseTime:	6.6330e-05
5%SettlingTime:	9.3068	5%SettlingTime:	7.6657
SettlingMin:	0.0916	SettlingMin:	-0.2267
SettlingMax:	0.1148	SettlingMax:	0.1152
Overshoot:	12.7889	Overshoot:	3.1262e+04
Peak:	0.1148	Peak:	0.2267
PeakTime:	4.5800	PeakTime:	0.0400

(Figure 4.4) The per formance of selected gains in 0 degree yaw

From Figure 4.4 we can see that by increasing Kd3, the settling time of both z postion and z velocity increase while the overshoot of z velocity decrease a half.

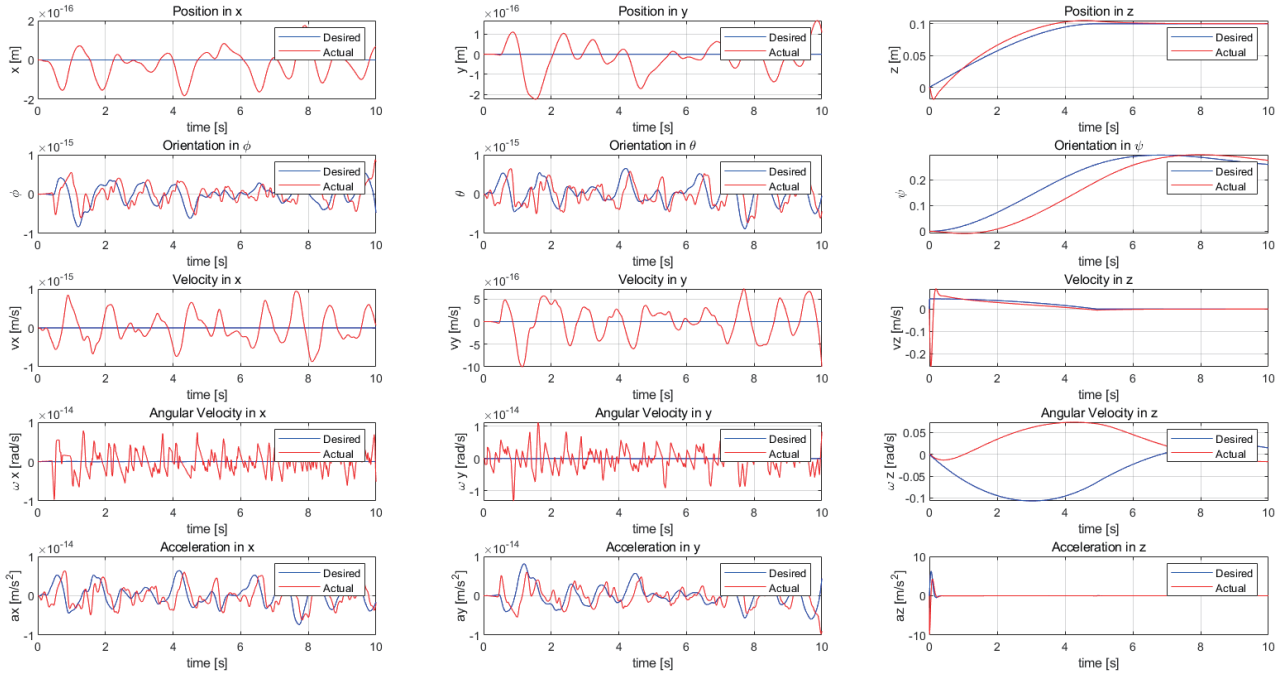
Kp1 = 17; Kd1 = 6.6; Kp2 = 17; Kd2 = 6.6; Kp3 = 20; Kd3 = 192;
Kpphi = 190; Kdphi = 30; Kptheta = 198; Kdtheta = 30; Kppsi = 80; Kdpsi = 17.88;

z position performace:		z velocity performace:	
90%RiseTime:	3.2536	90%RiseTime:	1.2333e-04
5%SettlingTime:	9.5179	5%SettlingTime:	7.3160
SettlingMin:	0.1009	SettlingMin:	-0.2143
SettlingMax:	0.1206	SettlingMax:	0.1150
Overshoot:	7.6497	Overshoot:	1.5851e+04
Peak:	0.1206	Peak:	0.2143
PeakTime:	4.8400	PeakTime:	0.0400

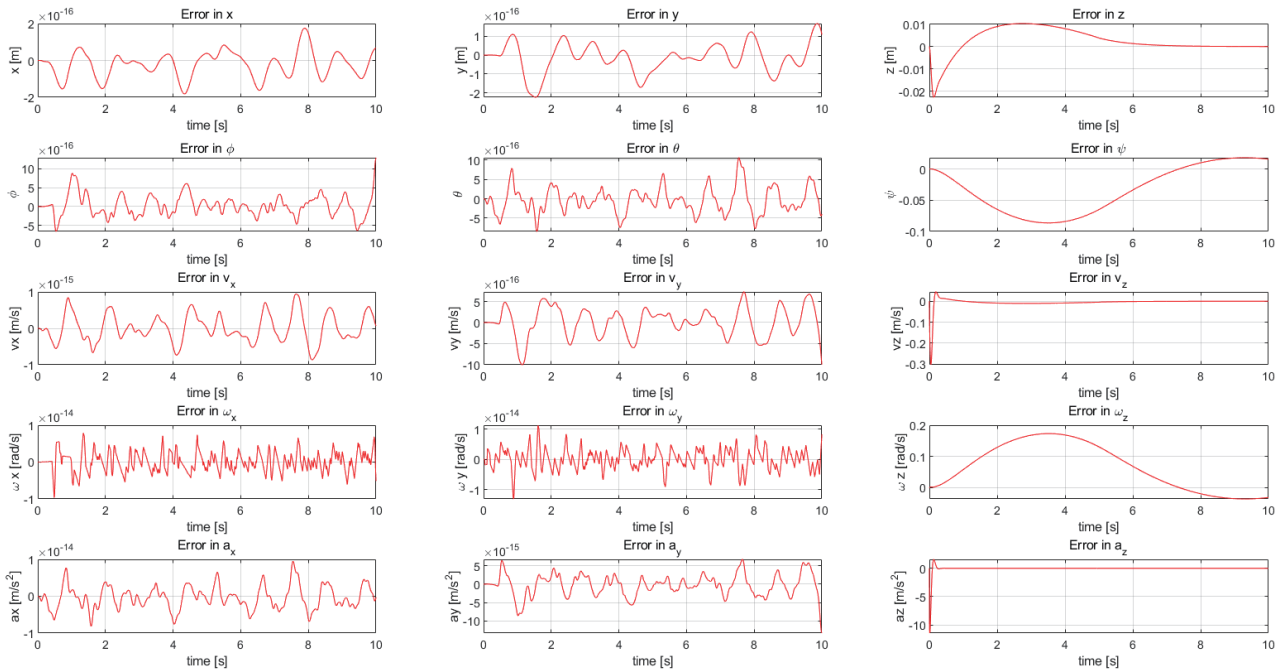
(Figure 4.5) The per formance of selected gains in 0 degree yaw

From Figure 4.5 we can see that by keep increasing Kd3, the overshoot decrease a half. However, it won't decrease exponentially no matter how we increase our Kd3

Kp1 = 17; Kd1 = 6.6; Kp2 = 17; Kd2 = 6.6; Kp3 = 20; Kd3 = 50;
Kpphi = 190; Kdphi = 30; Kptheta = 198; Kdtheta = 30; Kppsi = 80; Kdpsi = 17.88;



(Figure 4.6) The desired and actual state of the quadcopter in 0 to 10 seconds



(Figure 4.7) The error between desired and actual state of the quadcopter in 0 to 10 seconds

Figure 4.6 and 4.7 show the performance of the quadcopter taking off. At the same time it slowly turns to 15 degree in yaw. The steady state value of z_position, z_velocity and yaw angle is 0.1m, 0s/m and 15 degree

Kp1 = 17; Kd1 = 6.6; Kp2 = 17; Kd2 = 6.6; Kp3 = 20; Kd3 = 50;
Kpphi = 190; Kdphi = 30; Kptheta = 198; Kdtheta = 30; Kppsi = 80; Kdpsi = 17.88;

z position performace:

90%RiseTime:	3.0044
5%SettlingTime:	6.7390
SettlingMin:	0.0900
SettlingMax:	0.1050
Overshoot:	4.9519
Peak:	0.1050
PeakTime:	4.5850

z velocity performace:

90%RiseTime:	1.9100e-06
5%SettlingTime:	6.0695
SettlingMin:	-0.2578
SettlingMax:	0.0898
Overshoot:	1.2384e+06
Peak:	0.2578
PeakTime:	0.0450

yaw angle performace:

90%RiseTime:	5.8105
5%SettlingTime:	9.9142
SettlingMin:	0.2490
SettlingMax:	0.2985
Overshoot:	7.9904
Peak:	0.2985
PeakTime:	7.9800

(Figure 4.8) The per formance of selected gains in 15 degree yaw

By comparing Figure4.4 and Figure 4.8 we can see that adding turning task to the take-off task will :

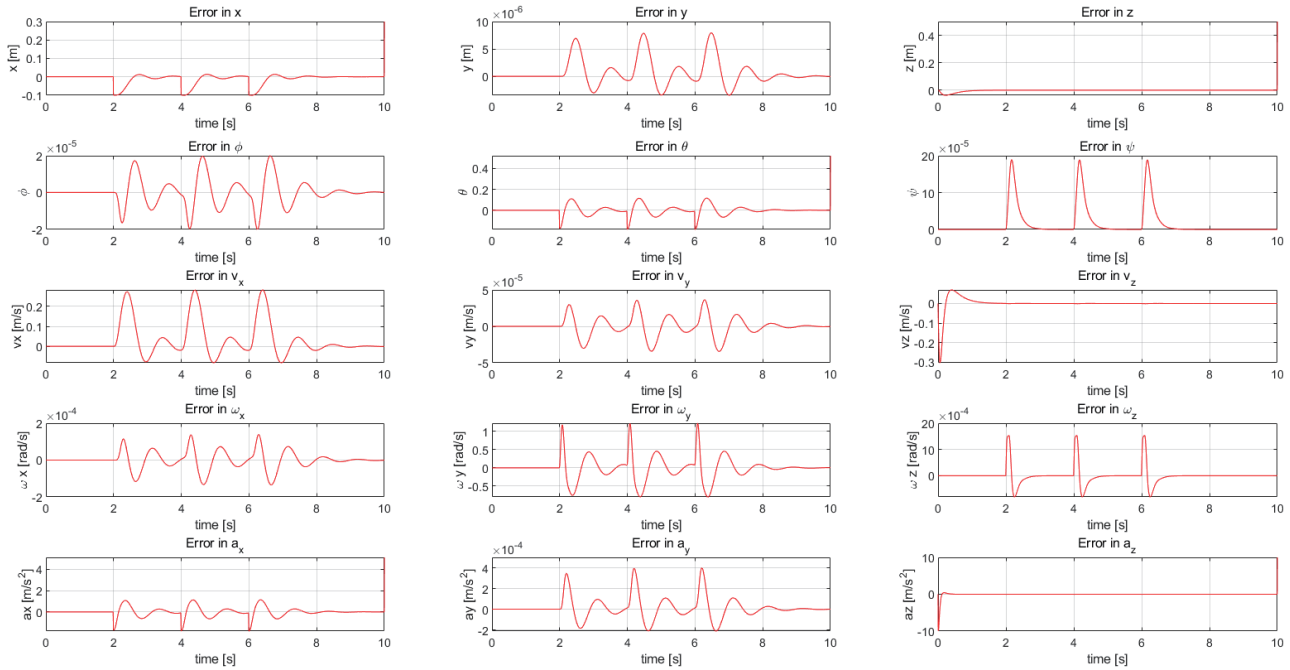
1. Reduce the rise time of z position, increase the rise time of z velocity
2. Increase the settling time of z position and z velocity
3. Increase the overshoot of z position and z velocity

Part 5 - Gain Selection and Turning Conclusion

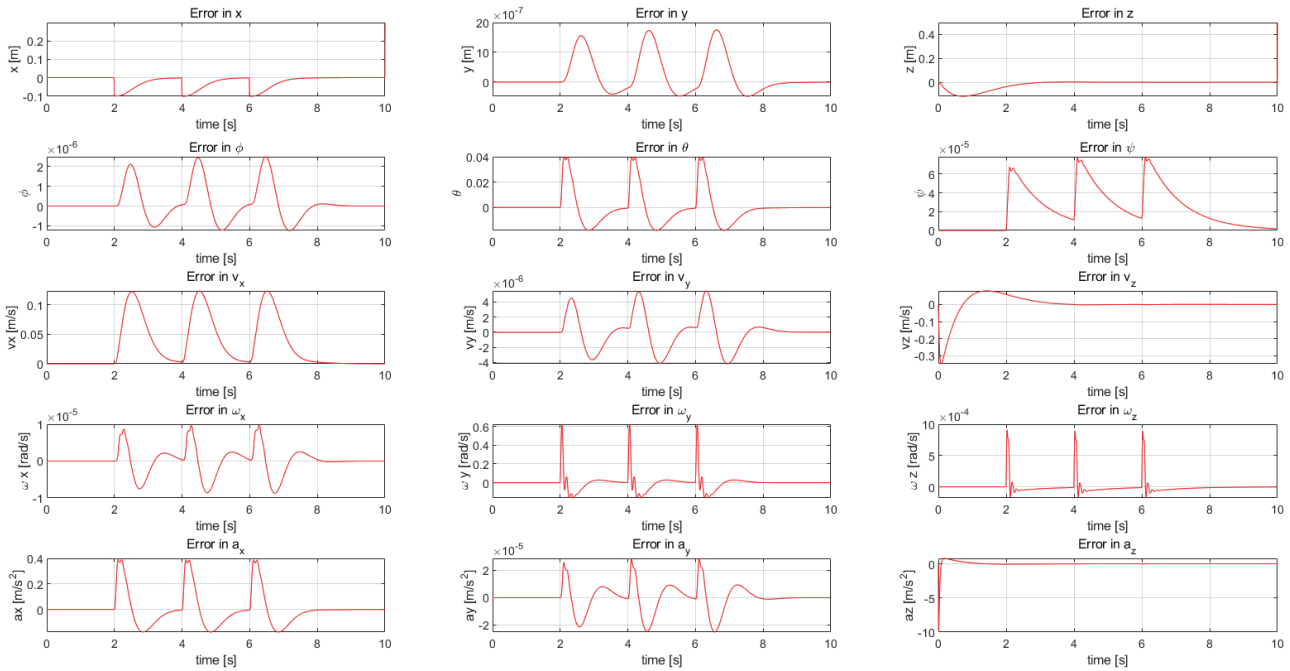
For taking-off and hover task, the overshoot of velocity is always large and hard to control. However, the position convergence is easy to control. Adding turning task to the take-off task will hurt the performance of z position convergence and z velocity convergence

Part 6 - LQR Controller Design and Evaluation

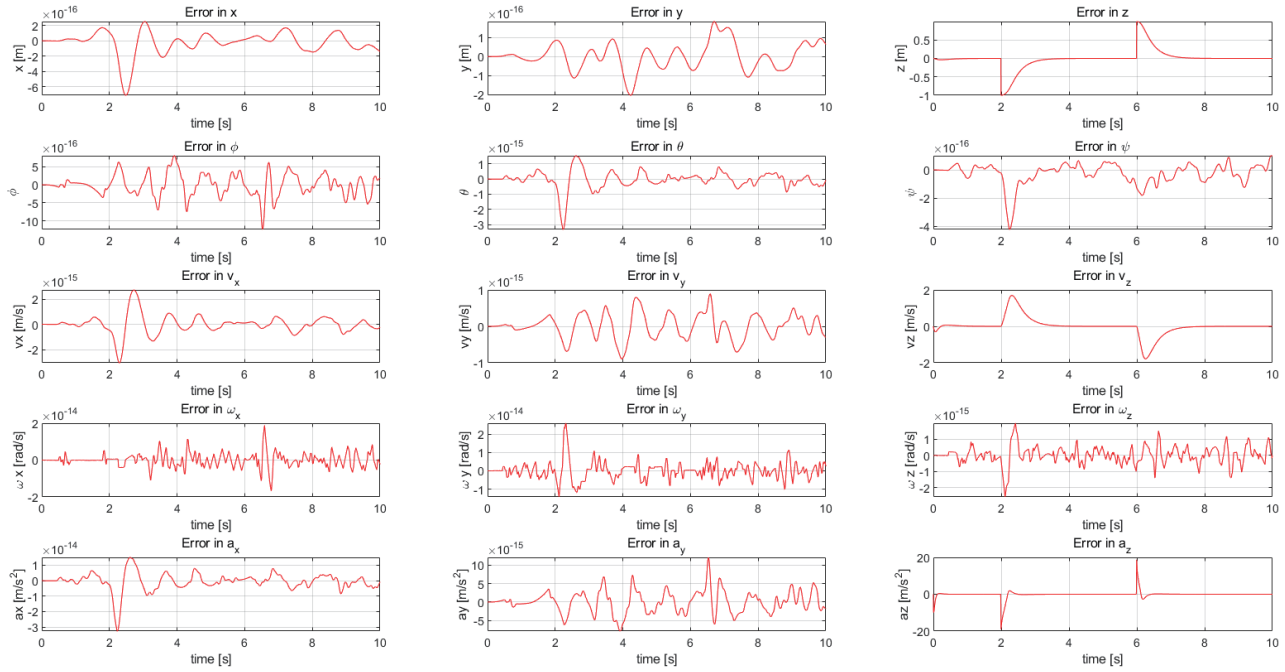
In this part, we redo the task 2, task 3 and task 5 using LQR controller instead of PD controller and compare their performance



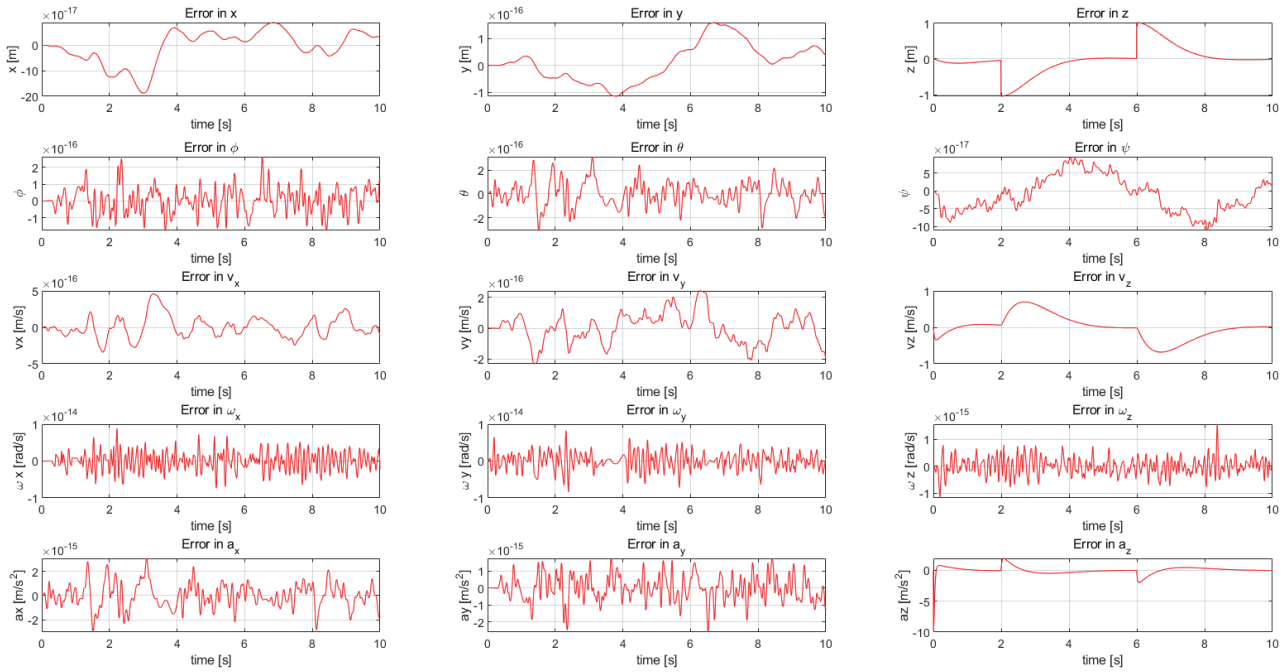
(Figure 6.1) The error of part-2 hover task using PID controller



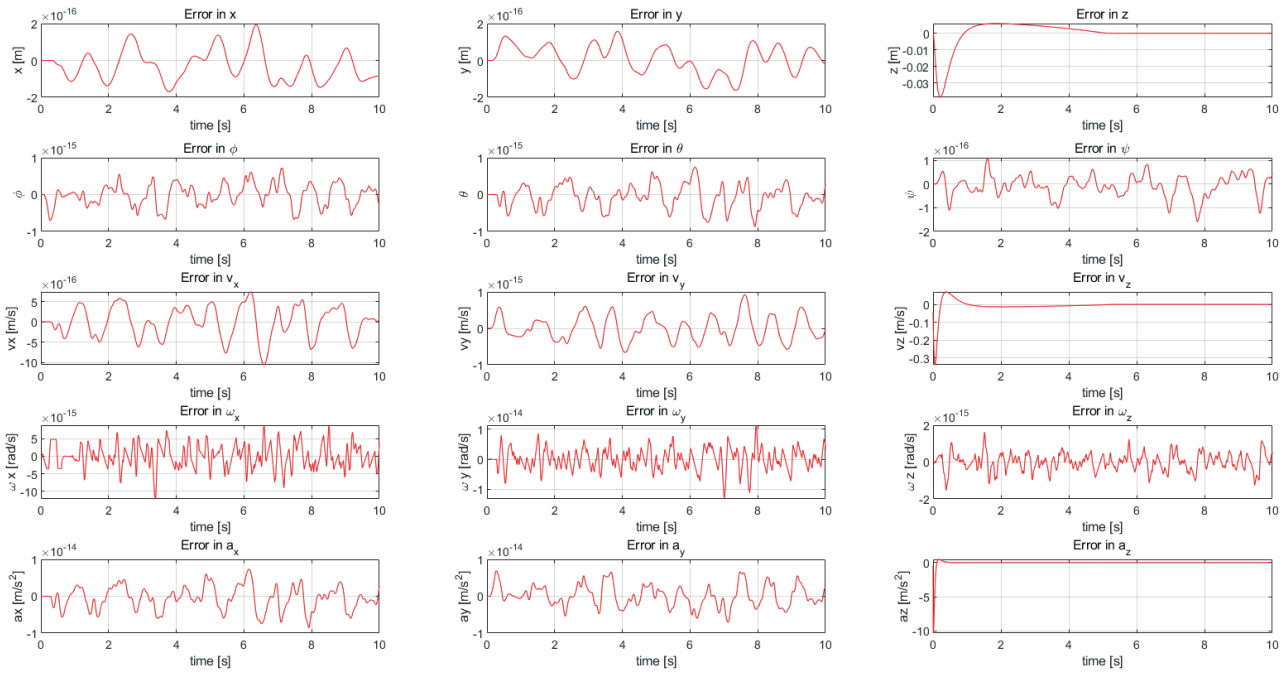
(Figure 6.2) The error of part-2 hover task using LQR controller



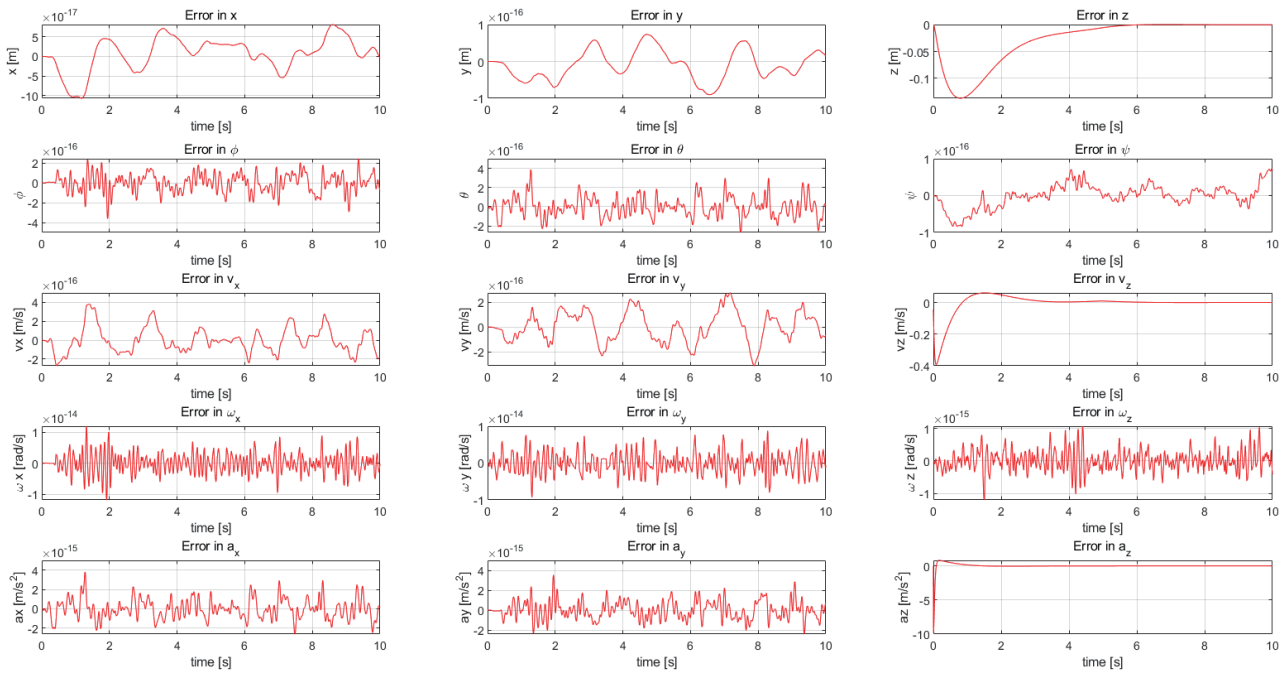
(Figure 6.3) The error of part-3 Line-tracking task using PID controller



(Figure 6.4) The error of part-3 Line-tracking using LQR controller



(Figure 6.1) The error of part-5 take-off and hover task using PID controller

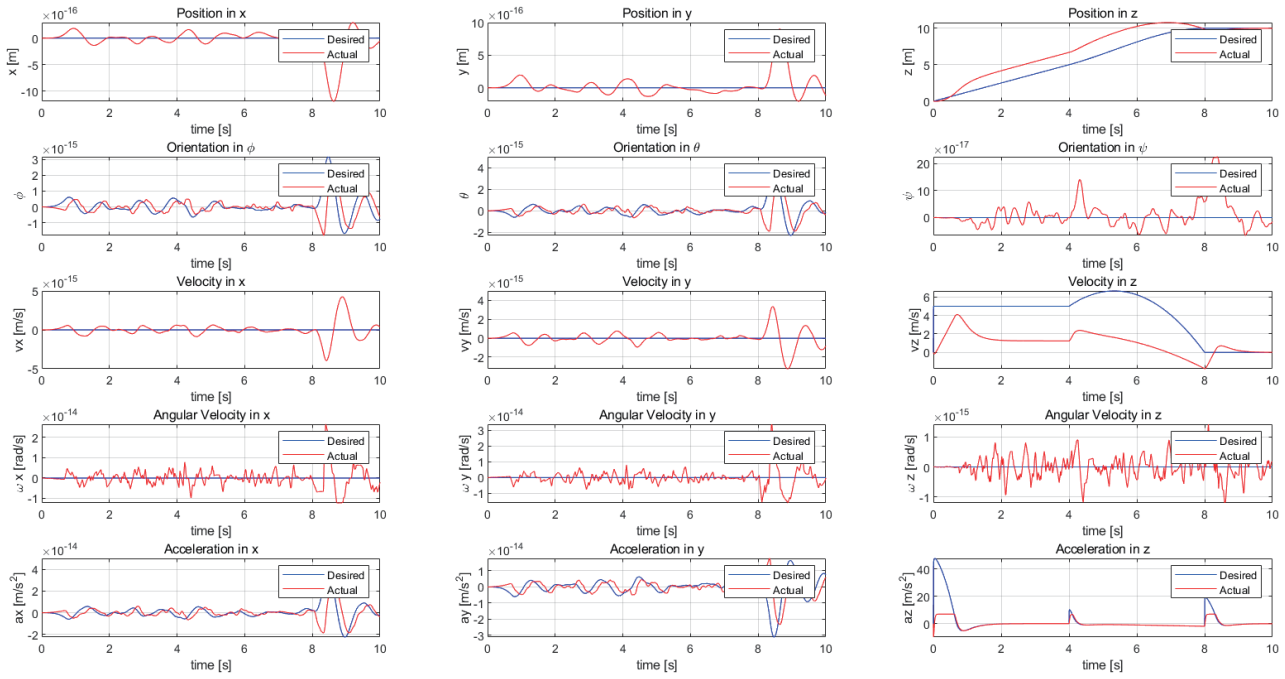


(Figure 6.2) The error of part-5 take-off and task using LQR controller

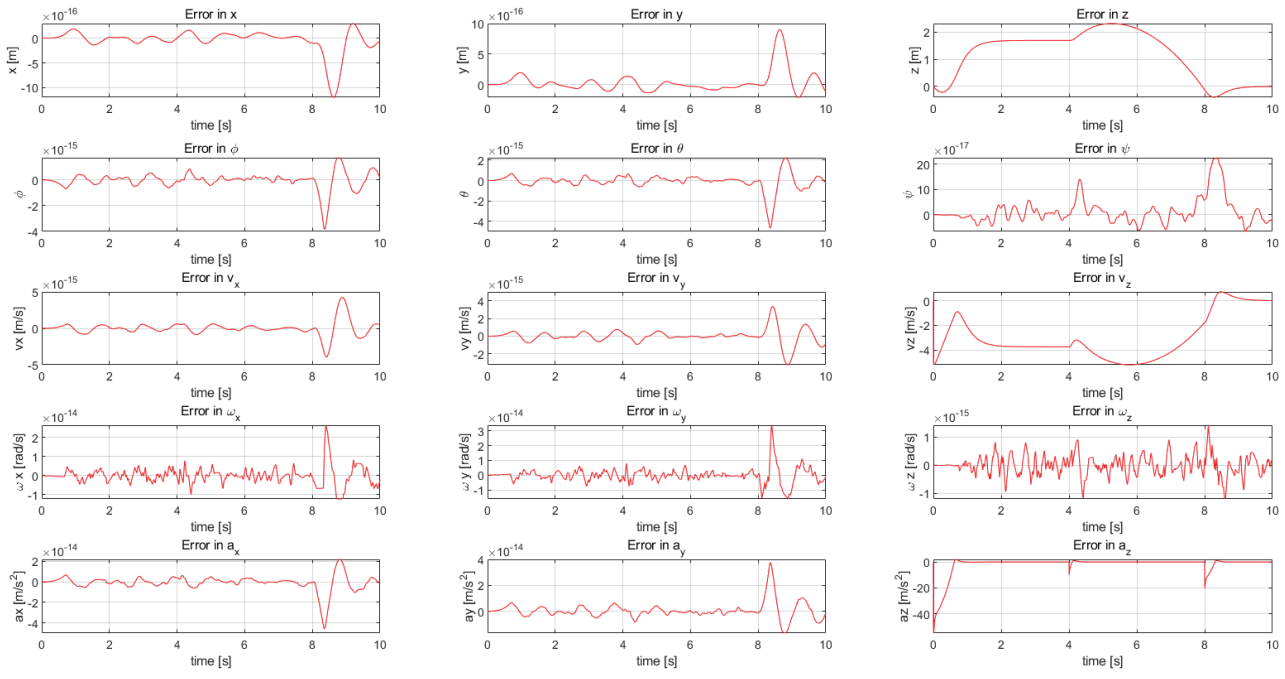
Part 7 - Boundry Acceleration Trajectory Generation

A straight line time-parameterized polynomial trajectory with initial and final endpoint constraints ([0, 0, 1] and [0, 0, 10], respectively; higher-order terms zero). Choose a time-scaling that ensures a bounded (maximum/minimum) acceleration less than 3 m/s2. Generate a set of error plots that depict the performance of the platform when tracking the trajectory (using PD control) including the error in the pose and linear/angular velocities.Increasing the motor gain to decrease the error in tracking acceleration

According to the formula S = acceleration * sqr(t) / 2, we know that to control the acceleration under 3m/s/s in moving 9m, the take-off time should be longer than sqrt(6) = 2.4495, in the following experiment we set the rise time to 8 s to see the performance.

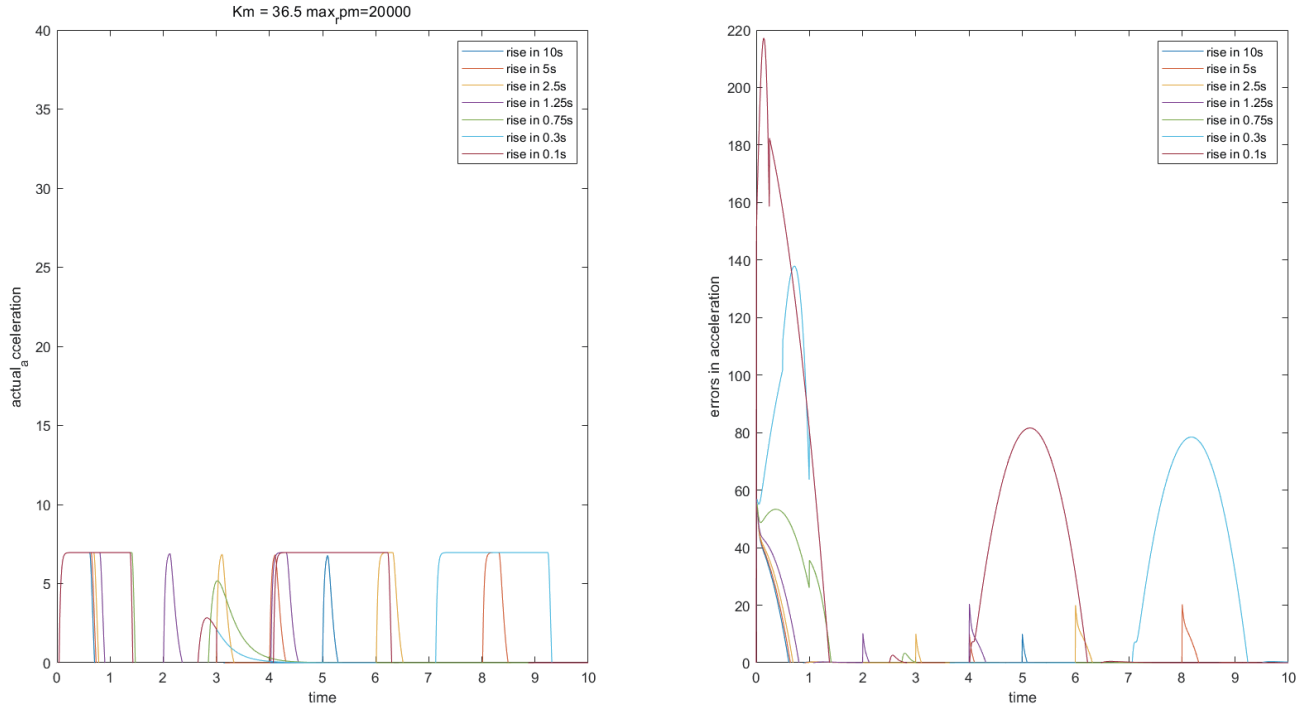


(Figure 7.1) The desired and actual state of the quadcopter in 0 to 10 seconds

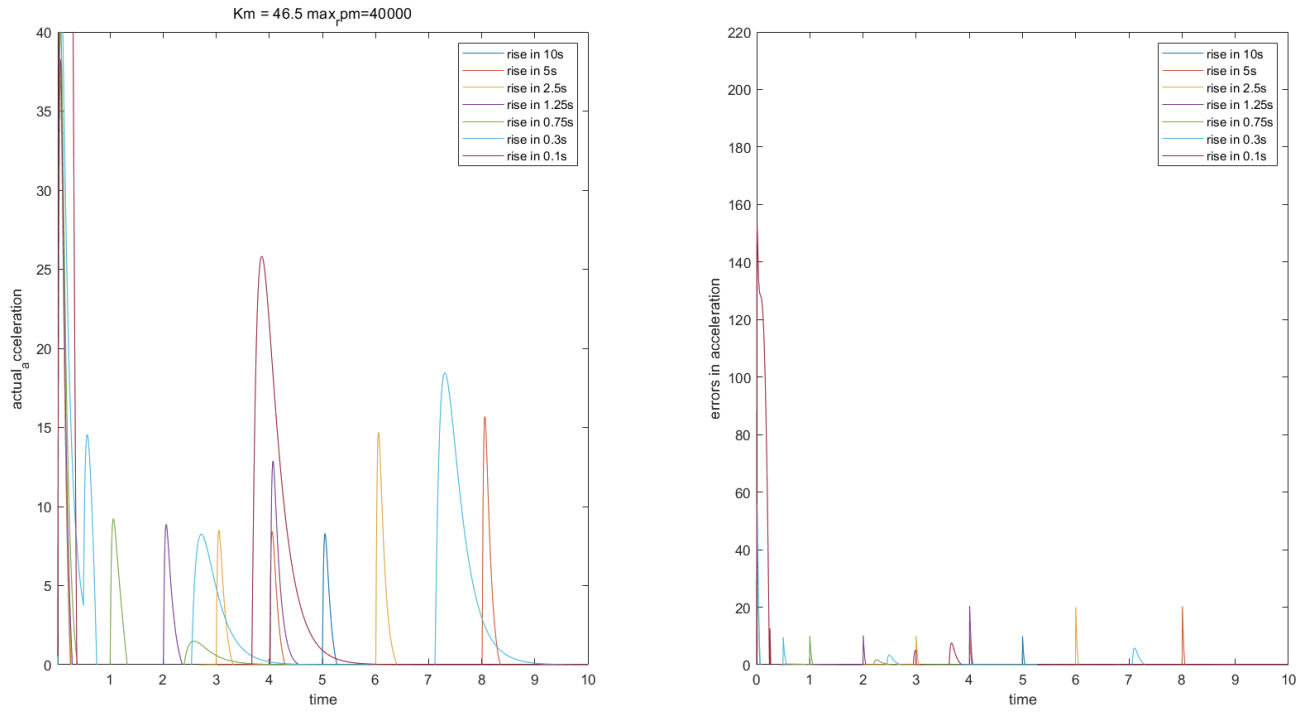


(Figure 7.2) The error of the quadcopter in 0 to 10 seconds

Here, we fail to generate a acceleration formula that control the acceleration under 3m/s/s. But we can roughly see that, the quadcopter is not able to track the large acceleration. That is because the motor gain and max RPM in motor model constrain the performance of motor, thus prevent the quadcopter to accelerate too fast



(Figure 7.3) Actual acceleration in different desired acceleration (left)
Acceleration tracking errors (right)
(Km=36.5 maxrpm =20000)



(Figure 7.4) Actual acceleration in different desired acceleration (left)
Acceleration tracking errors (right)
(Km=46.5 maxrpm =40000)

From figure 7.3 we can see that the max acceleration in Km=36.5 and maxrpm = 20000 is 6.922 m/s/s.
From the comparison between 7.3 and 7.4 we can see that by increasing the Km and maxrpm we are able to reduce acceleration errors and increase acceleration tracking performance

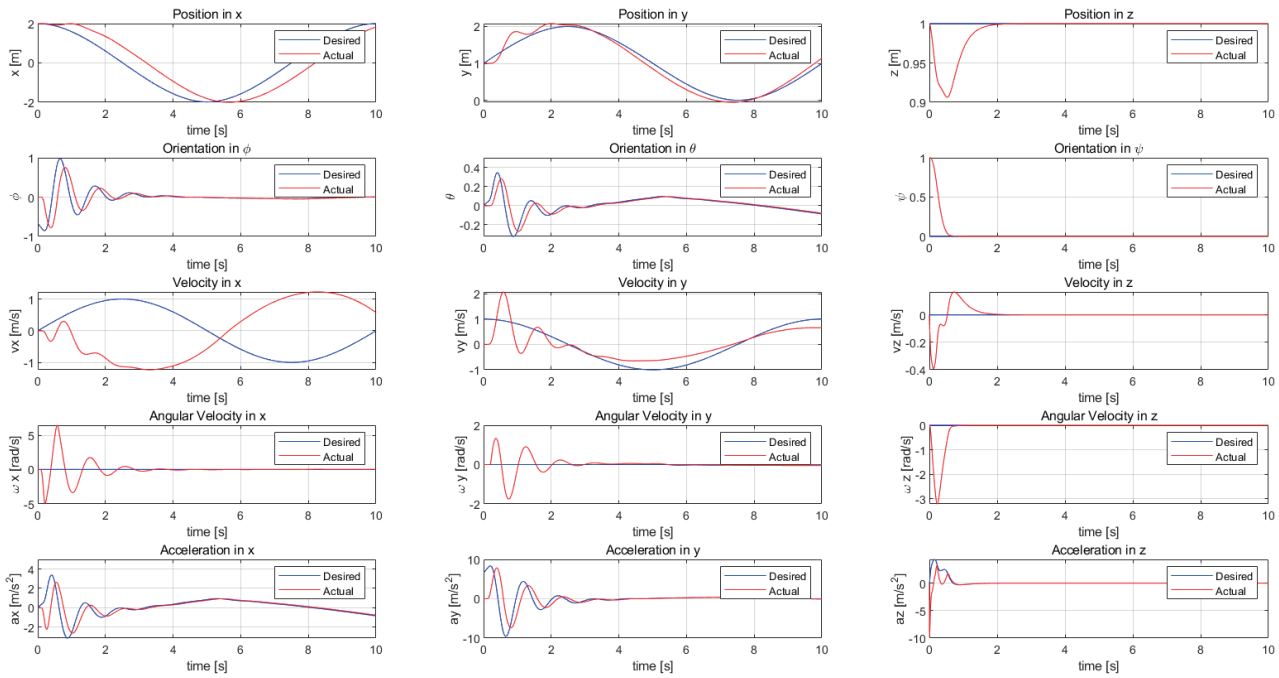
Part 7 - Boundry Acceleration Trajectory Generation Conclusion

As we see, providing certain RPM and motor gain constraint, there is some acceleration boundary that causes the quadcopter not follow the desired acceleration. This is the natural physical constrains. The only way to increase acceleration tracking performance is to replace motor with larger RPM and gains.

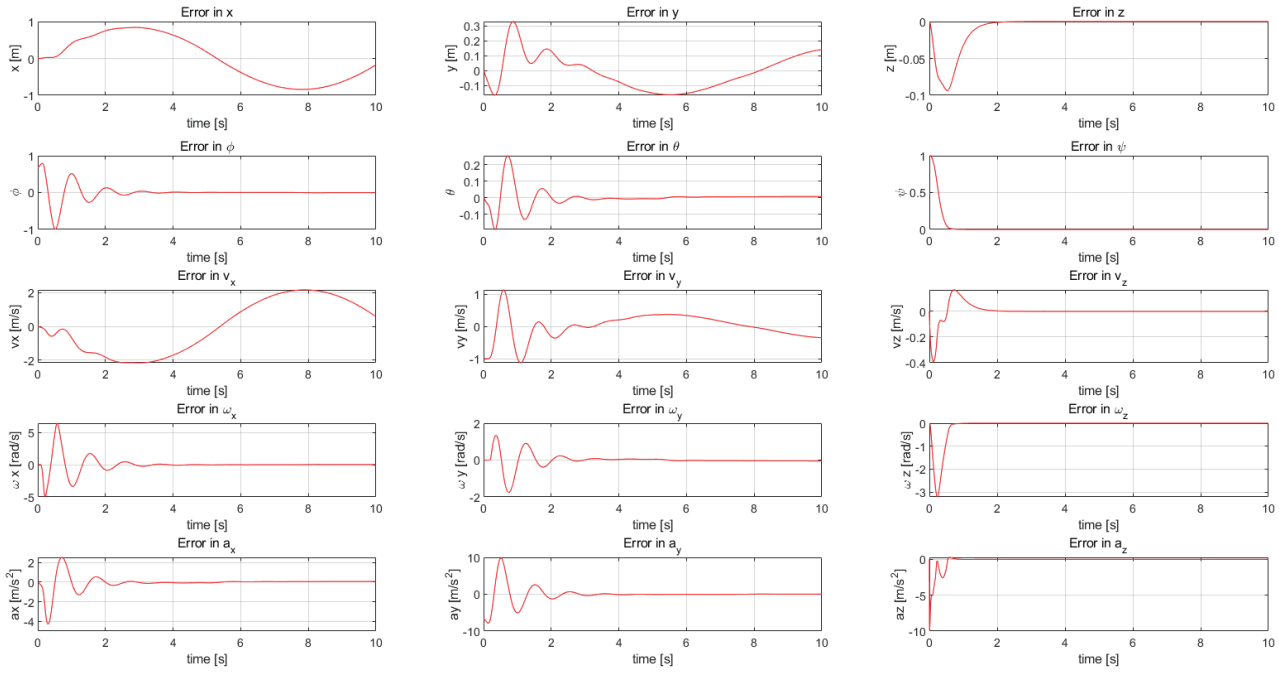
Part 89 - Elliptical Trajectory with different yaw angle and velocity

Define a piecewise continuous trajectory consisting of four waypoints: [0, 0, 1, 0], [2, 1, 1, 0], [0, 2, 1, 0], [-2, 1, 1, 0] (position, heading)The robot will start at rest, track the elliptical trajectory, and arrive at the starting location at a nonzero velocity (1 m/s). Generate a second trajectory phase given the same waypoints that starts with a 1 m/s velocity tangent to the ellipse and similar velocities at the other waypoints (all tangent to the ellipse in the direction of motion). Do experiment in different velocity and yaw angle to measure the performance

zero yaw angle elliptical with 1m/s velocity

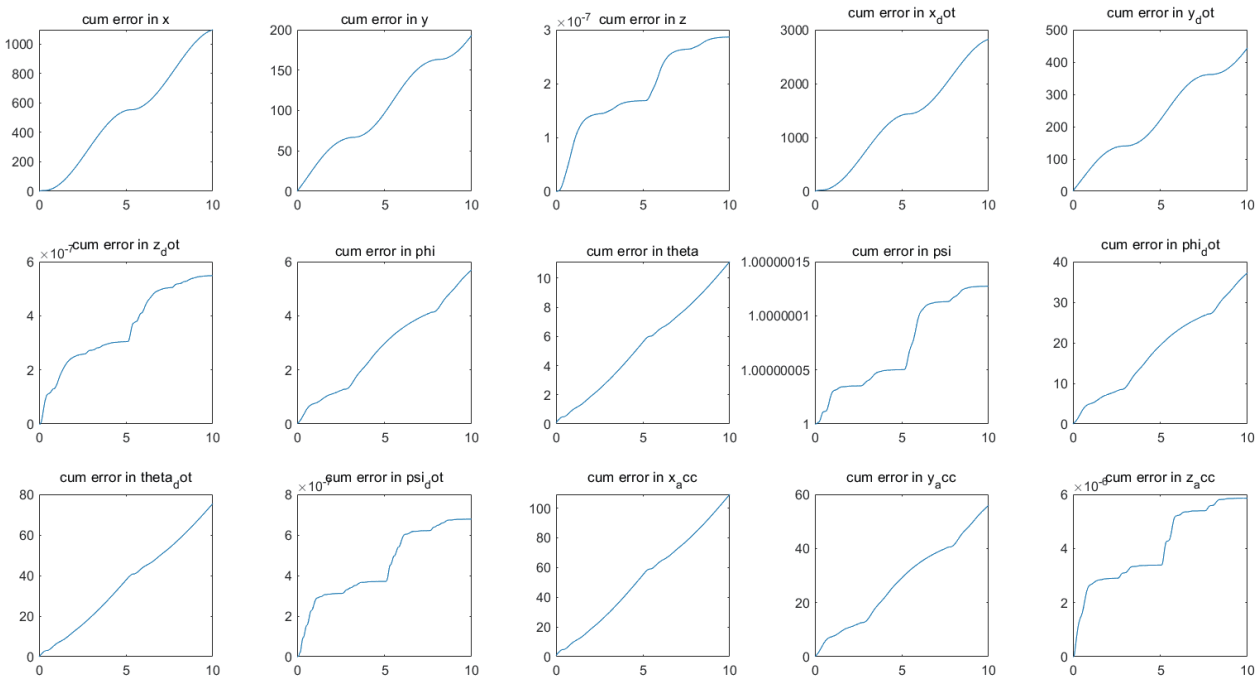


(Figure 8.1) The desired and actual state of the quadcopter in 0 to 10 seconds



(Figure 8.2) The error of the quadcopter in 0 to 10 seconds

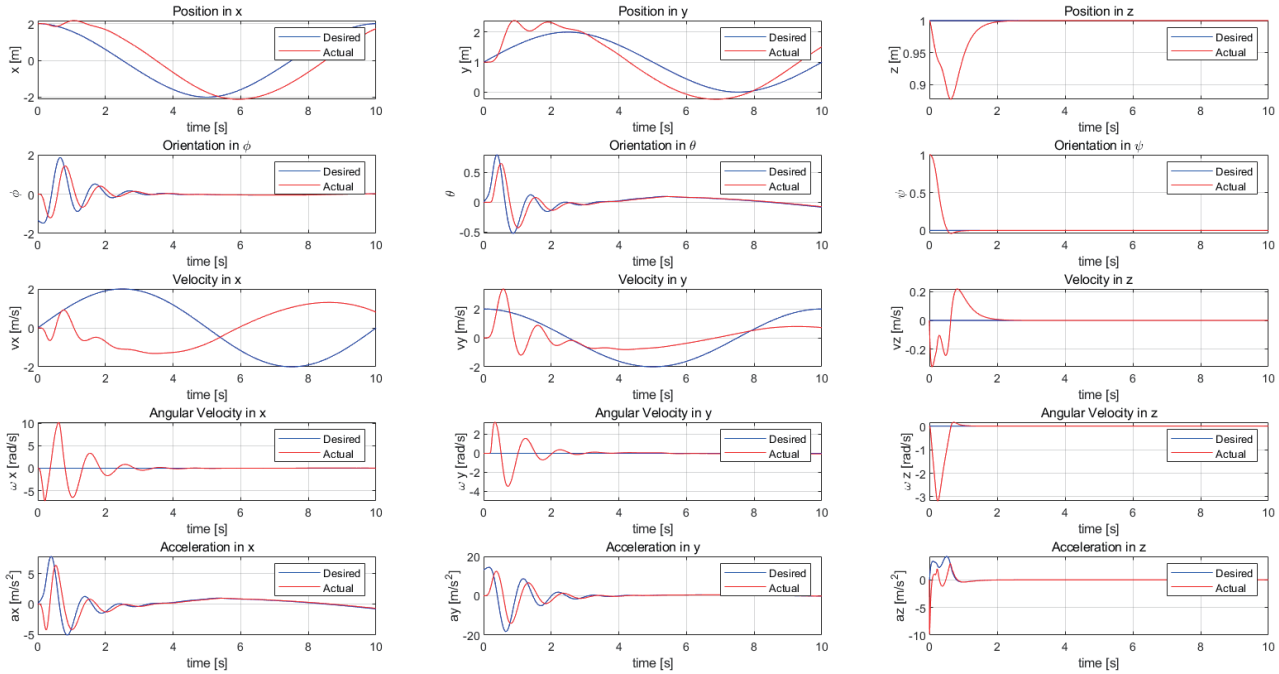
zero yaw angle elliptical with 1m/s velocity



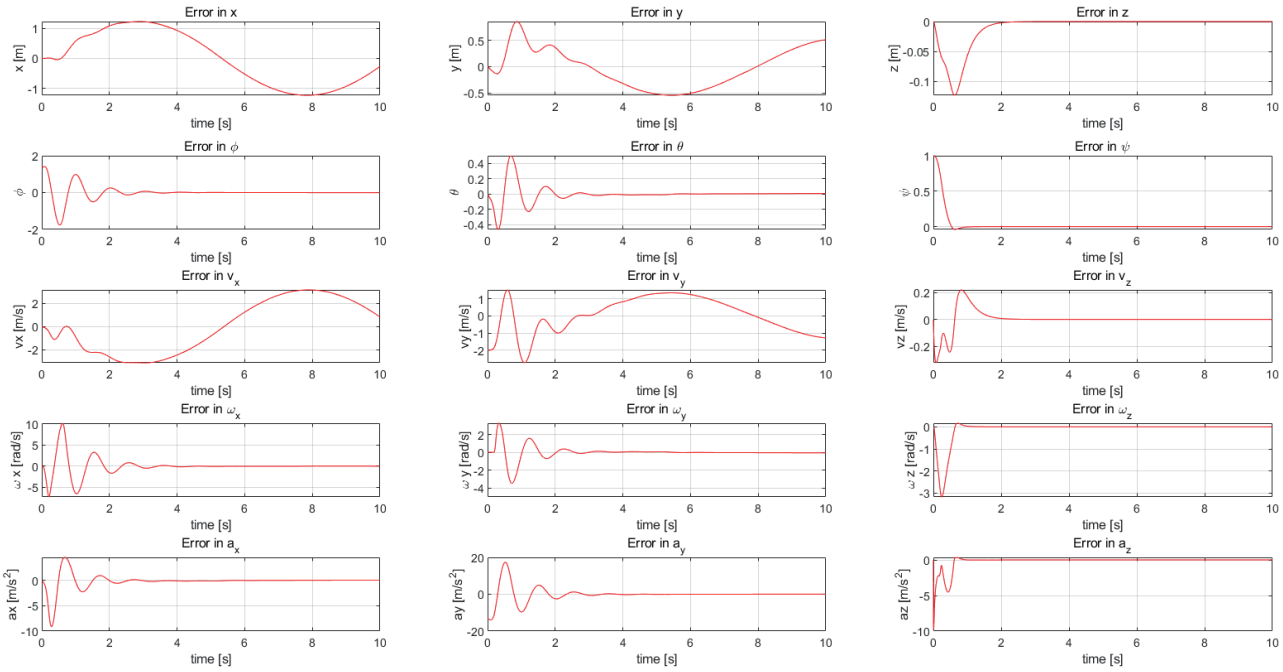
(Figure 8.3) The cumulative error of the quadcopter in 0 to 10 seconds

From Figure 8.1 Figure 8.2 Figure 8.3, we can see the performance of zero yaw angle elliptical trajectory with 1m/s velocity. Now, Let's do some experiment in different velocity

zero yaw angle elliptical with 2m/s velocity

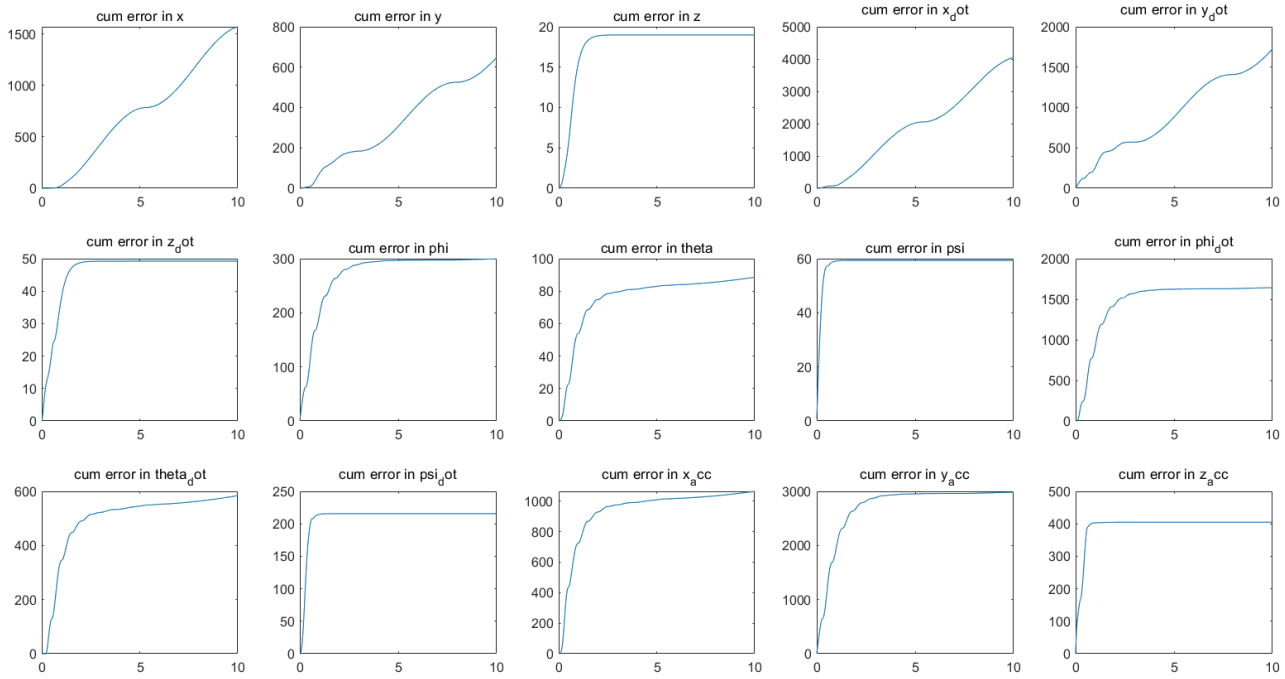


(Figure 8.4) The desired and actual state of the quadcopter in 0 to 10 seconds



(Figure 8.5) The error of the quadcopter in 0 to 10 seconds

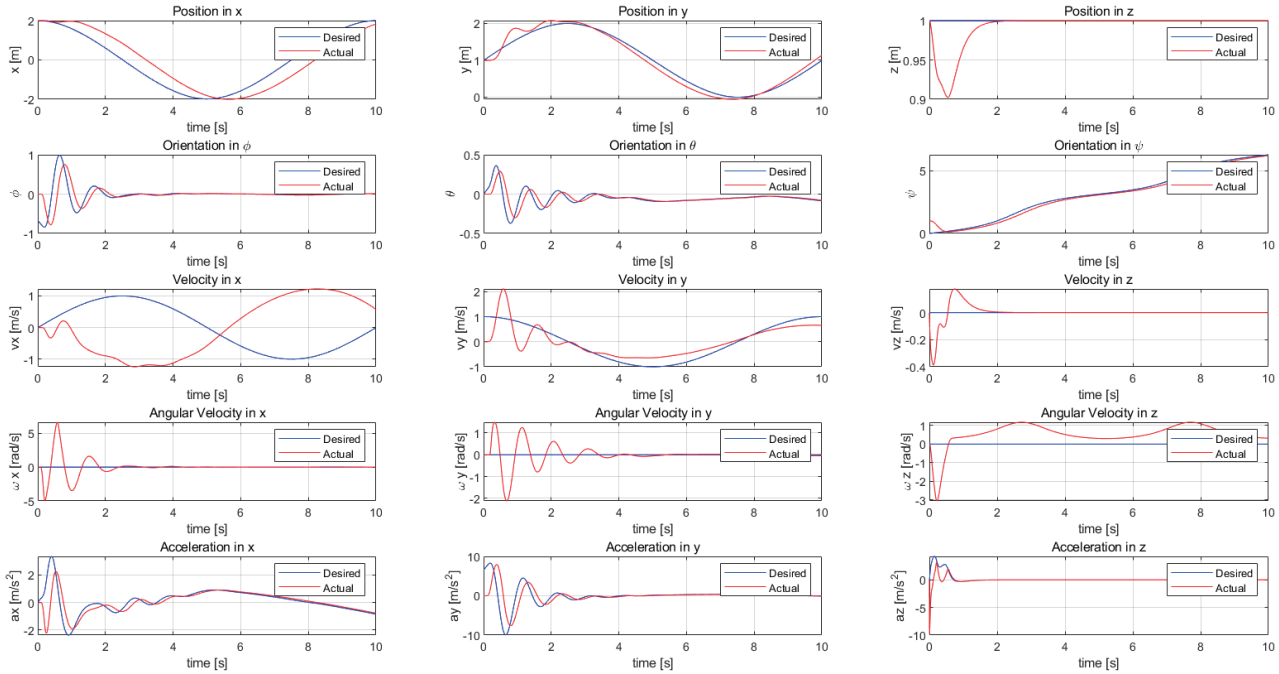
zero yaw angle elliptical with 2m/s velocity



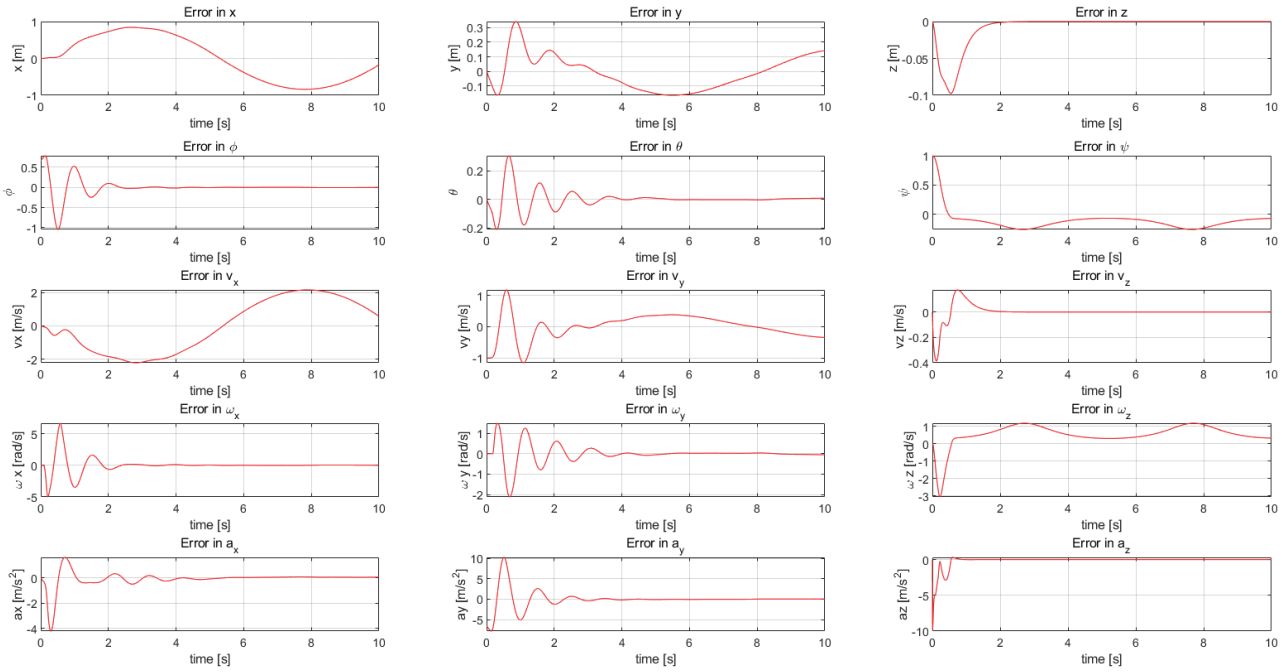
(Figure 8.6) The cumulative error of the quadcopter in 0 to 10 seconds

By comparing the figure 8.5 and figure 6 we can see, increasin the velocity cause the accumulative errors of every perspective to increase

15 degree yaw angle elliptical with 1m/s velocity

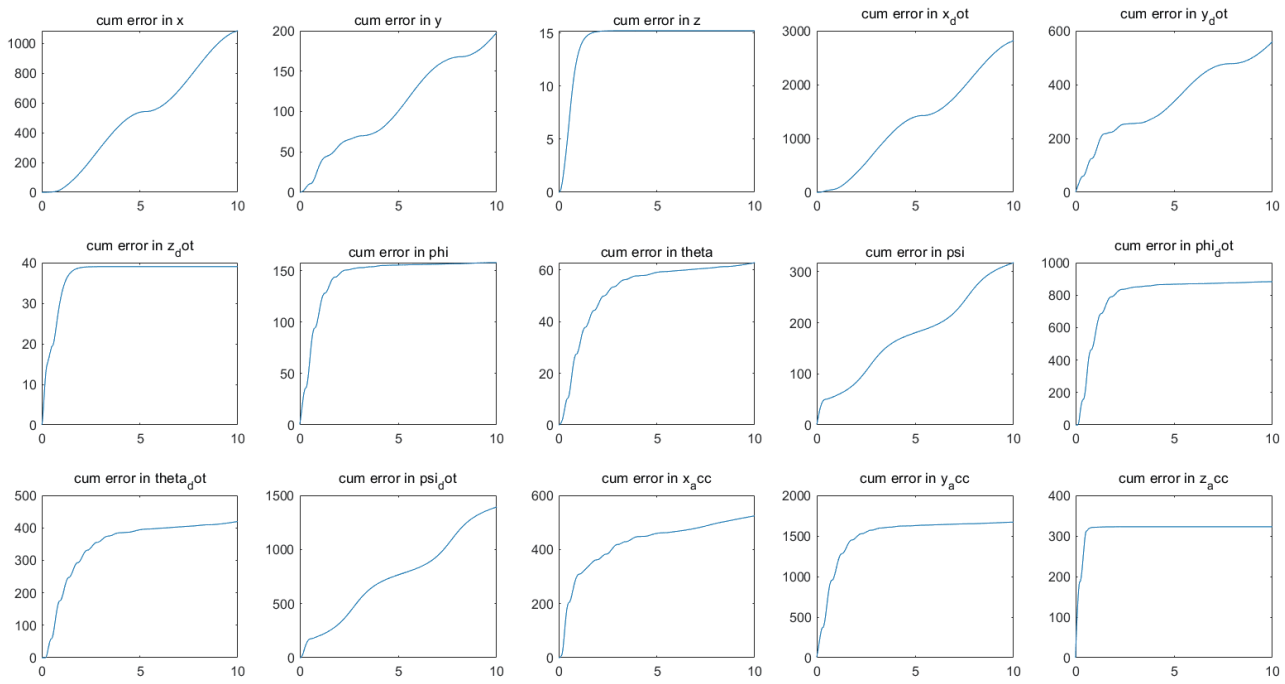


(Figure 8.7) The desired and actual state of the quadcopter in 0 to 10 seconds



(Figure 8.8) The error of the quadcopter in 0 to 10 seconds

15 degree yaw angle elliptical with 1m/s velocity



(Figure 8.9) The cumulative error of the quadcopter in 0 to 10 seconds

By comparing the figure 8.5 and figure 9 we can see, addition condition like yaw angle will hurt the performance of the quadcopter and thus cause the accumulative error to increase

Part 89 - Elliptical Trajectory with different yaw angle and velocity Conclusion

For elliptical trajectory, higher speed will cause the error to increase, as well as fixed yaw angle