**COMP0128 Coursework 2**

**Jian Zhou**

**Q1:**

There are two files containing code, “ and . Drone.m is a class file, which defines the properties and methods a specific class has. While the other file is a script that helps to instantiate the class and makes the whole code work.

The following picture shows the main part in file . It calls method in class to iterate until time reaches to 8 seconds.

Graphical user interface, text, application, email

Description automatically generated

Figure1: quadcopter.m

**“**

Graphical user interface, text, application, email

Description automatically generated

Figure 2: update function

Instance Drone will response to the call of function . Firstly, it will add 1 to the timer. Then, for this task, it will set the inputs to default vector, which is [0.49;0.49;0.49;0.49]. The number of 0.49 is the square of angular velocity of a single propeller and it is the parameter that we can access to. As is shown in figure 2, the inputs are increased by 15% when timer is at the range of (2,4) seconds. For the next four seconds, the power we apply to is removed.

In the end, we call to update the state of the model. The details of kinematics are shown below in figure 3.

Table

Description automatically generated

Figure 3: Tool functions and kinematics function

Firstly, as we only have gyro to observe the drone, we can only access to angular velocity . We assume the sensor is precise enough and hence can calculate theta. Basing on thetadot and theta, we can therefore calculate acceleration and omega. Using linear information, the actual position can be calculated by discrete integral. All other tool functions are shown below in figure 4 and the specific explanation can be found in [1].

Graphical user interface, application

Description automatically generated

Figure 4: All other tool functions

In the end, the results are shown below.

Chart

Description automatically generatedGraphical user interface

Description automatically generated with medium confidence

Figure 5 figure 6

From figure 5 and figure 6, we can find that, before the 4th second, the orientation of the drone didn’t change as the inputs of it hadn’t changed. After we removed the power of the third propellers, the drone started to rotate and eventually fell.

**Q2:**

1. To find the non-linear dynamics of the quadcopter in state-space representation. We could firstly define multiple symbols in MATLAB and then using MATLAB to perform the partial derivatives, instead of performing by hand (as is shown in figure 7).

Graphical user interface, text

Description automatically generated

Figure 7:

Following the state definition in [1], where x1 is the position, is the linear velocity, is the angular and is the angular velocity vector. The derivatives of them are shown in figure 8.

Diagram, schematic

Description automatically generated

Figure 8: State

After we get all state and statedot, we can then use MATLAB built-in function Jacobian() to get parameters of LTI system. Basis and ] are used as they are the input and state of the state form LTI system.

Now we have already digitalized the system. Then, substitute the initial state into the system and then use MATLAB function () to digitalize the system.

Text

Description automatically generated

Figure 9: digitalize the system

1. The core idea of state-space representation is using the formula

Text

Description automatically generated

Figure 10: kinematics of state-space representation.

In figure 10, using the parameter A and B we can calculate the current state by apply the state of the last iteration. Note that we should update the parameter A and B in each iteration with up-to-date data.

Chart

Description automatically generated Graphical user interface

Description automatically generated with low confidence

Figure 11 Figure 12

The comparison of results from Q1 and Q2 are shown in figure 11 and 12. The difference between trajectories are noticeable. And thus, roll, pitch and yaw all have difference non-negligible numeral difference. This error may be generated when we linearize the model. As we used Tylor series to expand the origin formula and only kept the first two terms for calculation. In this case, the abandoned terms cause noticeable error.

**Q3**

1. Two control a quadcopter basing on the model in [1], we can apply two controllers to the model. The first one is used to control the altitude of the aircraft. For a quadcopter with four propellers, the thrust on z axis should compensate the gravity and friction. Fortunately, we can increase and decrease the power supply of the propellers to control the quadcopter. Therefore, the first PID controller has been created in figure 13.

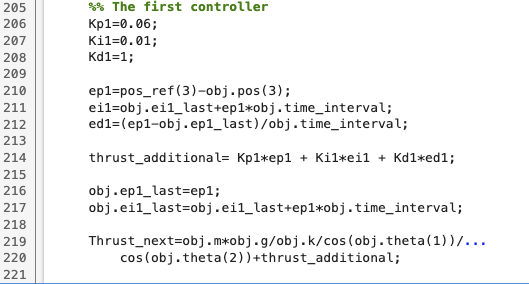


Figure 13: the first controller

The input of the controller is distance vector between target and quadcopter and the output is the additional thrust that will be add to total thrust in the end before kinematics step.

The other controller is responded to control the angular of the quadcopter to make the aircraft move in x-y plane. Code is shown below in figure 14. Graphical user interface, text, application

Description automatically generated

Figure 14: The second controller

The input of the controller is the angular of quadcopter and the output is the error which is proportional to the differences between desired trajectory and observed trajectory and its derivatives. However, we can only access to the angular velocity as we just own a gyro. Therefore, we have to use to use the relationship below, which assume the gyro can precisely record angular velocity.

A picture containing text, receipt

Description automatically generated

Then, as the output of the PID is proportional to , which means we can directly add our desired gesture information into the system with the . Therefore, the rotation between the desired gesture and current gesture can represented by the rotation between current acceleration and desired gesture. Desired gesture can be calculated by differences to target and current acceleration can be found as is shown in Q1. The corresponding code is shown below.

Graphical user interface, text, application

Description automatically generated

Figure 15: The second controller

Beside controller, the is determined by logic control. Several tasks’ functions are created to control the main thread of the code. I defined some flags to indicate the progress of the code. And the five tasks have been combined into three.

Text

Description automatically generatedText

Description automatically generated

Figure 16 Figure 17

Figure 16 shows how the program know the progress of itself according to the flags we set. And Figure 17 ,18 and 19 are the code for each task. It’s safe for them to use same timer because we are using single thread and single progress and the tasks are performed sequentially.

Graphical user interface, text, application, email

Description automatically generatedText, application

Description automatically generated

Figure 18 Figure 19

For both task D and task E, we divide the whole desired trajectory into several checkpoints.

Chart

Description automatically generatedChart

Description automatically generated

Figure 20 Figure 21

Graphical user interface, application

Description automatically generated

Figure 22

The results are shown above. The basic tasks are completed successfully. However, it is noticeable that the circle is not smooth and has many vibrations in it. One of the possible reasons is that the parameters of the PID controller are not tuned to an optimization, if I had enough time, I could use automatic PID tuning method to help find the best PID parameters. Another possible reason could be that the error calculation model for the pid is wrong, I refer exactly to the method in [1], although I still have doubts about it. I think it would be more appropriate to use the pos obtained by theta calculation as the error. But for time reasons, I will have to do it later when I have the chance.

**b)**

Since we only has gyro that can only observe angular velocity, therefore, we just simply need to add a gaussian noise to dot before each iteration. The code is shown below and figures with different means and variances are shown in table 1.

Text

Description automatically generatedGraphical user interface, text, application

Description automatically generated

Figure: 23 Figure 24

|  |  |  |  |
| --- | --- | --- | --- |
| Table 1 : Multiple combinations of mu and sigma | | | |
|  | Mu | Sigma | Figure |
| 1 | 0.001 | 0.001 |  |
| 2 | 0.001 | 0.1 |  |
| 3 | 0.001 | 1 |  |
| 4 | 0.01 | 0.001 |  |
| 5 | 0.01 | 0.1 |  |
| 6 | 0.01 | 1 |  |
| 7 | 0.1 | 0.001 |  |
| 8 | 0.1 | 0.1 |  |
| 9 | 0.1 | 1 |  |
| 10 | 1 | 0.001 |  |
| 11 | 1 | 0.1 |  |
| 12 | 1 | 1 |  |

Basing on the figures in table 1, we can find that the model can be easily affect by noise. It is also possible that the noise of the setting is not within a reasonable range. As for the answer to this question, more in-depth research is needed

**Reference**

[1] A. Gibiansky, Quadcopter Dynamics, Simulation, and Control. An approximated dynamic model of a quadcopter and some potential control strategies are described in the following document.