

















<https://www.coursera.org/learn/robotics-flight/discussions/weeks/3/threads/E9fbliRLEeezyRLjKgAj0A>

## Tuning the gains (davidjameshall)

**Masoud Hassani**[Assignment: 2-D Quadrotor Control](https://www.coursera.org/learn/robotics-flight/item/iOWUe/discussions) · [hace 3 años](https://www.coursera.org/learn/robotics-flight/discussions/weeks/3/threads/E9fbliRLEeezyRLjKgAj0A" \t "_blank) · Editado

Hi everyone,

I have been working on the gain tuning for this assignment for a long time now. It seems like an impossible task to tune 6 parameters simultaneously. Also, even if I tune the gains for a line motion, it does not work very well for a sine motion. I have assumed that the second derivatives of phi\_c is zero and the first derivative is:

phi\_dot\_c = -( Kv\_y\*yddot\_des+ Kp\_y\*edot\_y )/g

I was wondering if someone can give me a hint!

Thanks

Publicación del davidjameshall

[D](https://www.coursera.org/learn/robotics-flight/profiles/c1f613cf70b34ec146bbcab06c55f338)

[DAVIDJAMESHALL](https://www.coursera.org/learn/robotics-flight/profiles/c1f613cf70b34ec146bbcab06c55f338)

**davidjameshall** · [hace 3 años](https://www.coursera.org/learn/robotics-flight/discussions/weeks/3/threads/E9fbliRLEeezyRLjKgAj0A/replies/kg2ZOS39EeeZ1Qo60q6J_A)

As for as specifics, I forgot about everything but the inner loop (rotation) and tuned it. Keep in mind Ixx is very small, so some big values will be required.

I got the simulated quadcopter to fall as usual, but as it fell, it actually rotated and attained the desired angle (remember that in the code commanded values are radians, while the plot is in degrees)

THEN I started playing with trying to get it to move in the other directions.

Honestly, I didn't spend any time the other four remaining constants, but chose values close to those previously used, and my final values were all well within tolerance, and I scored 50/50.

BTW, would anyone like to compare values resulting from running the submit.m code? I'm curious "how low we can go".

1. Publicación del davidjameshall

[D](https://www.coursera.org/learn/robotics-flight/profiles/c1f613cf70b34ec146bbcab06c55f338)

[DAVIDJAMESHALL](https://www.coursera.org/learn/robotics-flight/profiles/c1f613cf70b34ec146bbcab06c55f338)

**davidjameshall** · [hace 3 años](https://www.coursera.org/learn/robotics-flight/discussions/weeks/3/threads/E9fbliRLEeezyRLjKgAj0A/replies/y9OZPC38EeeqRw77WotmkA)

I used the fact that

PercentOvershoot = 100\*exp(-zeta\*pi/sqrt(1-zeta^2)) to solve for zeta,

and Ts=4/(zeta\*omegan) to solve for omegan as a function of zeta and Ts settling time (time required to reach about 2% of desired value)

Then we take the traditional controls second order Laplace equation

s^2 + 2\*zeta\*omegan\*s + (omegan)^2 = 0

and equate it to our traditional error equation

edotdot + Kv\*edot + Kp\*e = 0

and voila! We can magically find Kv and Kp fairly quickly in a closed-form manner, rather than through hours of exhaustive trial and error.

I humbly submit this , based on my background with a master's in control theory.

* 1. Publicación del Deleted Account

[DA](https://www.coursera.org/learn/robotics-flight/profiles/795f07cf55a430193c9e96a7a9042315)

[DELETED ACCOUNT](https://www.coursera.org/learn/robotics-flight/profiles/795f07cf55a430193c9e96a7a9042315)

**Deleted Account** · [hace 3 años](https://www.coursera.org/learn/robotics-flight/discussions/weeks/3/threads/E9fbliRLEeezyRLjKgAj0A/replies/y9OZPC38EeeqRw77WotmkA/comments/k6TAyC8BEeeqRw77WotmkA" \t "_blank)

How did you find two variables (kp and kv) with a single equation.

* 1. Publicación del davidjameshall

[D](https://www.coursera.org/learn/robotics-flight/profiles/c1f613cf70b34ec146bbcab06c55f338)

[DAVIDJAMESHALL](https://www.coursera.org/learn/robotics-flight/profiles/c1f613cf70b34ec146bbcab06c55f338)

**davidjameshall** · [hace 3 años](https://www.coursera.org/learn/robotics-flight/discussions/weeks/3/threads/E9fbliRLEeezyRLjKgAj0A/replies/y9OZPC38EeeqRw77WotmkA/comments/DyMpZS-TEeeuuQ7tmHEh3A)

Actually, technically I'm using two equations (equations 1 and 2 below) , and then using the fact that the form of a traditional second order equation from control theory (equation 3) should look like the error function in differential equation form used a lot in class (equation 4). Hope this helps! It should help a lot of the frustration I see going on in the forum. CAVEAT: technically, equations (1) and (2) are approximations, but for our purposes I think they work pretty well.

I used the fact that

PercentOvershoot = 100\*exp(-zeta\*pi/sqrt(1-zeta^2)) (equation 2) to solve for zeta, (For example, the percent overshoot desired might be 4 , something somewhat less than 5%)

and Ts=4/(zeta\*omegan) (equation 2) to solve for omegan as a function of zeta and Ts settling time (time required to reach about 2% of desired value) (For example, you might desire Ts to be 1 second).

Now you have zeta and omegan.

Then we take the traditional controls second order Laplace equation

s^2 + 2\*zeta\*omegan\*s + (omegan)^2 = 0 (equation 3)

and equate THE COEFFICIENTS in it to our traditional error equation (given in class)

edotdot + Kv\*edot + Kp\*e = 0 (equation 4)

and voila! We can magically find Kv and Kp fairly quickly in a CLOSED-FORM manner, rather than through HOURS of exhaustive trial and error. I hope this helps. I added a few extra explanatory notes in case the first explanation didn't make sense.