Brigham Young University

AUVSI Capstone Team (Team 45)

Autopilot Tuning

ID	Rev.	Date	Description	Author	Checked By
CT-001	0.1	01-23-2019	Initial conception	Andrew Torgesen	Brandon McBride

1 Introduction

Our chosen fixed-wing autopilot for the UAV is ROSPlane, an open source implementation of the control, estimation, and path planning algorithms set forth in *Small Unmanned Aircraft: Theory and Practice* by Randal Beard and Timothy McLain. This artifact sets forth the general process for tuning the proportional, integral, and derivative (PID) gains for the autopilot, based on both theory from the text as well as personal experience.

2 Description of the Gains

Trim Gains:

- Elevator trim
- Aileron trim
- Rudder trim
- Throttle trim

These gains must be written to ROSFlight before turning on the autopilot for the first time. The directions for doing this are found at http://docs.rosflight.org/en/latest/user-guide/performance/ under the heading **RC trim (Feed-Forward Torque Calculation)**.

Longitudinal Autopilot Gains:

- Pitch_Kp
- Pitch_Kd
- Pitch_Ki
- ALT_KP
- ALT_KD
- ALT_KI
- AS_PITCH_KP
- AS_PITCH_KD
- AS_PITCH_KI
- AS_THR_KP

- AS_THR_KD
- AS_THR_KI

The suffixes Kp, Kd, and Ki refer to proportional, derivative, and integral gains, respectively. The Pitch gains control the inner loop of the longitudinal autopilot, and their tuning should correspond to a desirable convergence of current pitch to commanded pitch. The ALT gains are outer loop gains that determine the ability to hold an altitude without excessive deviation in the presence of disturbances.

In general, the AS_PITCH_ gains determine the aircraft's ability to hold an airspeed while landing, during which time pitch commands are used to control speed. Conversely, the AS_THR_ gains determine the aircraft's ability to hold an airspeed while flying, during which time throttle commands are used to control speed. The aircraft should be able to hold an airspeed even in the presence of wind and other external disturbances.

Lateral Autopilot Gains:

- Roll_Kp
- Roll_Kd
- Roll_Ki
- Course_Kp
- Course_Kd
- Course_Ki
- BETA_KP
- BETA_KD
- BETA_KI

The Roll gains control the inner loop of the lateral autopilot, and their tuning should correspond to a desirable convergence of current roll angle to commanded roll. The Course gains are outer loop gains that determine the ability to hold a commanded course angle without excessive deviation in the presence of disturbances. The BETA gains refer to slide-slip holding–in general, we don't worry too much about those, especially since we are not using rudder control.

Path Follower Gains:

- CHI_INFTY
- K_PATH

- K_ORBIT

These are gains that are tuned in accordance with the aircraft's ability to make tight turns of different radii and speeds. K_PATH determines how fast the plane approaches the straight line desired path. A high value causes the plane to fly perpendicular to the line until it reaches it while a low value causes the plane to asymptotically approach the desired line. CHI_INFTY is the course angle at which the path follower will approach a path if at an infinite distance away. K_ORBIT determines how fast a plane transitions from a straight line to an orbit. While the plane is far away from the center of the loiter locaiton, it will fly directly to the center but when the plane is close to the loiter location it flies the orbit path. A low K_ORBIT value causes the plane to have a more gradual transition from straight line to orbit.

3 Sequence of Gain Tuning (Practical Guide)

Note: The language used in this guide assumes at least a working knowledge of the Robot Operating System (ROS). ROS tutorials may be found at http://wiki.ros.org/ROS/Tutorials.

While running ROSPlane on a ROS network, the *rqt_reconfigure* plugin allows the user to view (and modify in real-time) all configurable gains for the autopilot. Figure 1 is a screenshot of the configuration user interface:

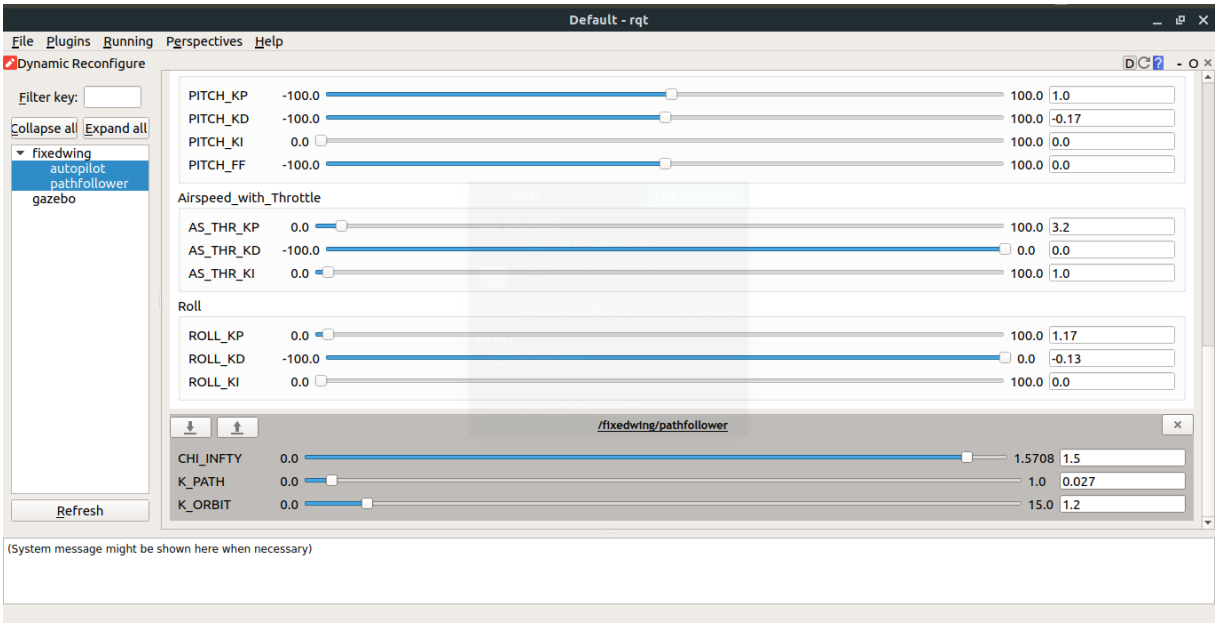

Figure 1: Dynamic gain reconfiguration user interface.

Before proceeding, keep the following heuristic for tuning PID gains in mind:

1. Increase P until the response time is satisfactory, albeit with a little bit of overshoot/oscillation.
2. Increase D until overshoot/oscillation is removed.
3. Increase I *only if* there is steady-state error.
4. **Iterate.**

3.1 Tune the Longitudinal Autopilot

Kill the path follower node with the command *rosnode kill /pathfollower*. Use the command *rqt_graph* to ensure that the node is really terminated once you've attempted to kill it. With the path manager killed and the reconfigure window open, send a command to the path follower to hold a pitch and altitude. For example, we sent the following command to the autopilot, which had the autopilot flying at an airspeed of 15 m/s, an altitude of 70 m, and a northern heading:

rostopic pub -r 50 /controller_commands rosplane_msgs/Controller_Commands – 15 70 0.0 0.0 '[0.0, 0.0, 0.0, 0.0]' False False

With this command running, tune the longitudinal gains in the following order:

1. Pitch gains
2. Altitude hold gains
3. Airspeed hold gains

Successful gain tuning should have the autopilot successfully flying at your commanded altitude at a relatively constant airspeed. Try outputing a new command with a different airspeed and altitude and see how your autopilot responds; it should transition smoothly to the new commanded values.

3.2 Tune the Lateral Autopilot

Relaunch the autopilot nodes in a subsequent flight to bring the path follower node back up. Then kill the path manager node with the command *rosnode kill /pathmanager*. Use the command *rqt_graph* to ensure that the node is really terminated once you've attempted to kill it. Now, send a command to the path follower to hold a roll and course. For example, we sent the following command to the autopilot, which had the autopilot

flying at an airspeed of 12 m/s in a clockwise orbit of radius 75 m, origin (-100 m North, -100 m East), and an altitude of 50 m:

rostopic pub -r 50 /current_path rosplane_msgs/Current_Path – 0 12 '[-1, -1, -1]' '[-1, -1, -1]' '[-100, -100, -50]' 75.0 1 False False

With this command running, tune the lateral gains in the following order:

1. Roll gains
2. Course hold gains
3. Path follower gains

Successful gain tuning should have the autopilot successfully flying at your commanded orbit at a relatively constant airspeed. Try outputing a new command with a different orbit and see how your autopilot responds; it should transition smoothly to the new commanded values.

3.3 Save the Gains

ROSPlane will not automatically save your new gain values from the reconfiguration interface window. You need to record the values that you set, put them in a .yaml file, and load those new parameters into your controller node upon every ROS launch! Likewise, the path follower gains must be loaded into the path follower node upon every ROS launch.

4 Conclusion

With satisfactorily tuned gains, the plane can now safely be used to fly waypoint paths. Try out some fun waypoint paths and see how well the autopilot handles them, even in the presence of wind and other disturbances!