LQR with Integral Feedback on a Parrot Minidrone

Michael F. Everett *

* *MIT, Cambridge, MA 02139 USA (e-mail: mfe@mit.edu).*

Abstract: Integral feedback was added to LQR on a Parrot minidrone to improve altitude tracking and steady-state error. Low steady-state error is critical in navigation and avoiding obstacles in flight. A Simulink model from the 16.31 Toolbox was modified to include the integral term, and new LQR gains were calculated with a new model to include the integral of altidude error. The introduction of integral feedback reduced steady-state altidude error from over 40cm to less than 1cm in real flight data, although its simulated performance was slightly worse. Therefore, integral feedback is a viable extension to LQR in situations where low steady-state error is required.

1. INTRODUCTION

LQR is an algorithm to design an optimal controller based on a cost function of what the designer deems most important. For example, if correct quadcopter orientation is more task-critical than x,y-position, the algorithm will consider states with incorrect orientation to be more costly than states with incorrect position, and the resulting controller will follow this logic.

While a stable hover is often the paramount characteristic of a quadcopter controller, hovering at the correct height is also incredibly important, especially in the context of drones delivering packages to a doorstep. Imagine a drone comes to a stable hover above a front porch and drops a new laptop from a few feet instead of a few inches — that steady-state error just became very expensive.

2. BACKGROUND

2.1 Previous Lab Experiments

During the semester. several controllers have been implemented on the Parrot minidrone, including PID, Pole Placement, and LQR. All of these methods have acheived a stable, hovering drone. LQR has the most physical intuition in its design process. Instead of choosing eigenvalue locations or gains and somehow linking them to a time response (not easy for a 12-state system), a matrix of weights is chosen depending on relative importance of control effort and state error for the application. Therefore, it would be nice to continue using LQR.

However, as seen in Lab 3, LQR has no guarantees on speed of response or steady-state error; it only generates the optimal controller for the particular cost function and model supplied, which are both often flexible. So a controller that meets tight specs for steady-state error was not possible using typical LQR, as seen in the drone's altitude estimates.

2.2 Integral Feedback

In order to improve steady-state error, the regular LQR structure is modified to include feedback on the integral of the error. In this project, only altitude integral feedback is implemented, though the same structure would apply for any of the states.

Integral feedback is a common technique in classical controls (the "I" in PID). In SISO systems, it can guarantee zero steady-state error. It is not so obvious how well LQR + Integral feedback will work, because the drone system is extremely complex (12 states, estimator, noisy sensors, linearized model, etc.).

Still, it makes sense that an integral term would help reduce steady-state error, because the integral of the error would continue to grow in magnitude as long as error persists.

Using the same model of the drone as in previous labs, the drone dynamics are linearized about a hover position. That provides a state-space model of the form:

$$\frac{d\mathbf{x}}{dt} = A\mathbf{x} + B\mathbf{u} \tag{1}$$

$$y = C\mathbf{x} \tag{2}$$

where $\mathbf{x}$ is a 12-state vector of position, attitude, and their derivatives.

A new system is proposed with a 13^{th} state, namely z_{integral}. While the output equation stays the same, that makes the new state evolution equation:

$$\frac{d}{dt}\begin{bmatrix}\mathbf{x}\\ z_{\text{integral}}\end{bmatrix} = \begin{bmatrix}A & 0\\ E & 0\end{bmatrix}\begin{bmatrix}\mathbf{x}\\ z_{\text{integral}}\end{bmatrix} + \begin{bmatrix}B\\ 0\end{bmatrix}u \tag{3}$$

$$E = [\,0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\,] \tag{4}$$

because the derivative of z_{integral} is z, which is already the 3^{rd} element of $\mathbf{x}$. This derivation differs slightly from [1] because there are Simulink blocks already in the toolbox to account for changing between state estimate, $\mathbf{x}$ and error, $\mathbf{e} = \mathbf{x}$ - reference.

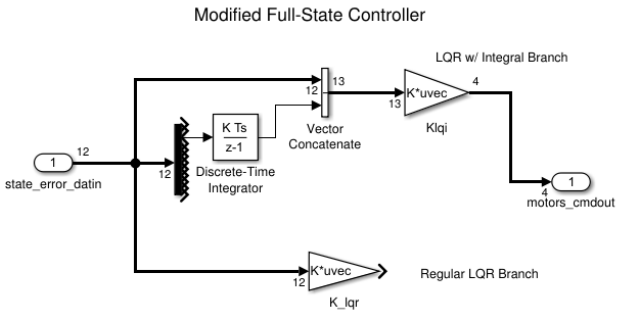

Fig. 1. Modified Full-State Controller: The bottom branch is standard LQR. The top branch includes the z integral error term and the LQI gain matrix.

With the new state-space model, it is easy to calculate the gain matrix, K_{LQI} using the MATLAB command `lqr` of the new system matrices. The new state weight matrix, Q_{new} contains the same weights for the original 12 states, Q, plus a designed weight w to control the weight of integrated altitude error.

$$Q_{\text{new}} = \begin{bmatrix} Q & 0 \\ 0 & w \end{bmatrix} \tag{5}$$

The R matrix stays the same, since the new state does not affect the control effort component directly.

The Simulink implementation of the new LQI can be seen in Figure 1 above.

The MATLAB code to generate gain matrices using `lqr` is attached in Appendix A.

3. EXPERIMENTAL DESIGN

3.1 Controller Choices

Five different controllers were used in this experiment.

- Regular LQR with state and control weights
- Regular LQR with z weight doubled
- Modified LQR with 3 different z integral weights

The LQR weights of the first controller were carried through for the other four (except where indicated).

The controller with doubled z weight was used to determine if integral feedback was more or less helpful than simply raising weight and keeping the regular structure.

The three integral weights were 0.1, 1, and 10: a wide range of weights.

3.2 Data Collection

The systems were simulated by Simulink for 20 seconds, with a step input of 1 meters. This was enough time for the altitude to reach a steady-state value for each of the controllers implemented.

After simulation, the compensator block was built and uploaded to the drone to measure performance on the real system. The drone was flown for just under 20 seconds, which again was enough to draw conclusions on settling and error to a 1.1 meter step.

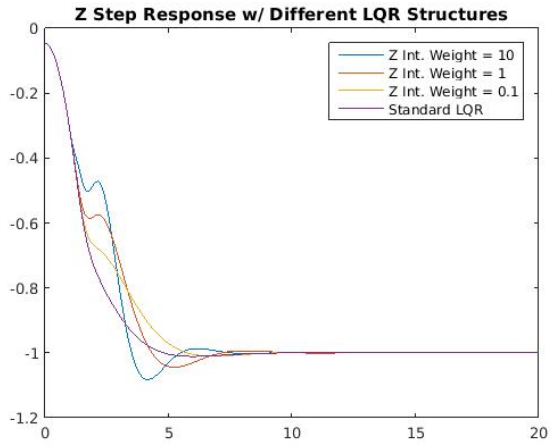

Fig. 2. Simulated Results: Standard LQR takes the most direct trajectory to -1. As Z integral weight increases, weird behavior occurs around -0.5 meters. From this view, it is hard to see a difference in steady-state value.

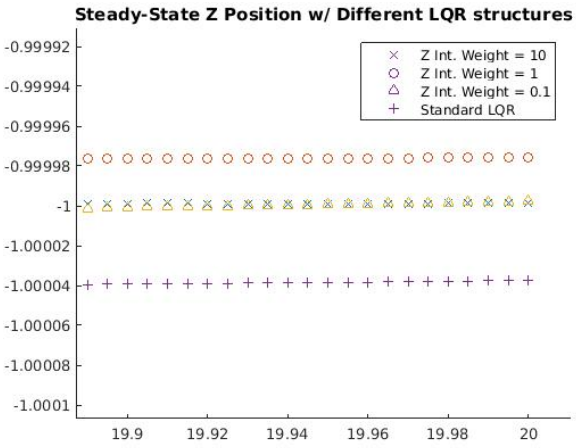

Fig. 3. Simulated Results (zoomed in): Now it is clear that the LQI controllers have lower steady-state error in simulation. However, the difference is micrometers which is orders of magnitude smaller than any reasonable spec for a quadcopter.

4. RESULTS

4.1 Simulation

To start, all of the controllers were simulated and their z positions over time were compared, shown in Figure 2.

In simulation, all of the controllers reach -1m within micrometers, which is incredible (and excessive) precision. If steady-state error really were the only spec, the LQI controllers performed best, as seen in Figure 3.

However, the LQI controllers have bizarre behavior around -0.5 and reverses direction, and it gets worse as z integral gain increases. It's not clear why this is. One possibility could be that something else was occuring in another state (say, pitch or yaw) during this time which had an effect on altitude, since quadcopter states are coupled.

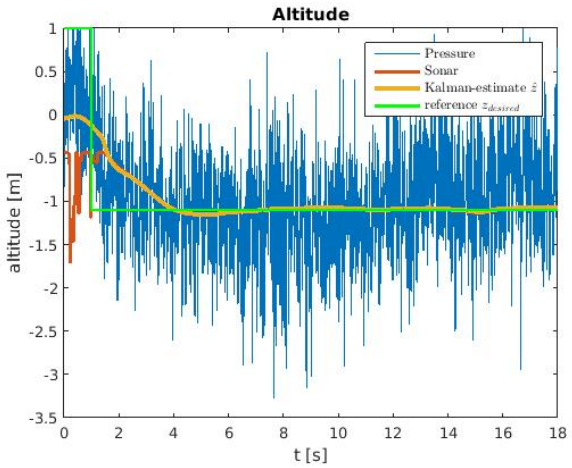

Fig. 4. Raw Altitude Data (Real Flight): The reference (green) is compared to noisy barometer data (blue), sonar (inaccurate below 30cm), and the Kalman estimate. This raw data is taken from a real LQI flight with z integral weight set to 1 (medium).

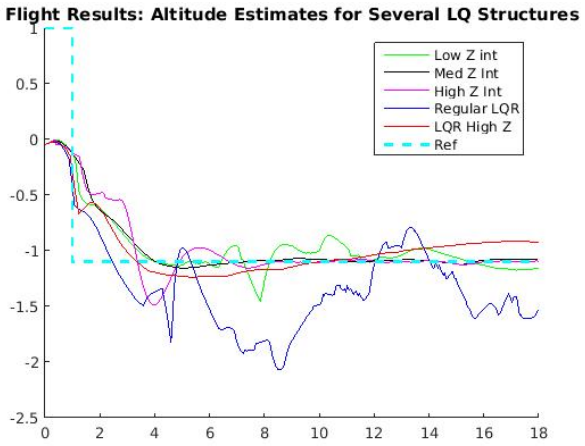

Fig. 5. Real Flight Results: Pure LQR (red and blue) settle further from the reference (cyan) than any of the LQI controllers.

The negligible improvement in steady-state error is far outweighed by the superior rise time, settling time, and continuity of the standard LQR controller, in terms of what is traditionally desired in a hover controller.

That is, in simulation, standard LQR performs the best.

4.2 Real Flight Data

The actual flights tell a different story. While the simulation is based on a complex model of the drone, there are tons of variables that are a part of real flight that aren't included in Simulink (e.g., temperature, wind, battery voltage, asymmetric motors, etc.).

The 5 controllers are compared in Figure 5. The plotted altitude estimate is the output of a Kalman filter combining sonar and barometer data. Raw data is shown in Figure 4

The regular LQR flight (blue) did not stabilize very well in altitude. Even though this is quite different from the simulated data, it is very similar to what was observed in previous labs.

When LQR z weight was increased (red), the trajectory was smoother and altitude error at the end of the flight was smaller. So, increasing z weight does improve z tracking somewhat.

The three LQI controllers (green, black, magenta) tracked the altitude reference step best. Low z integral weight was the worst of the three, which is expected because as accumulated error is multiplied by a bigger weight, the correcting control effort will be stronger.

As seen in Figure 6, the altitude estimate at the end of flight is compared for the 5 controllers to compare steady-state error in real flights.

Regular LQR controllers have steady-state error of tens of centimeters, while all three LQI controllers stabilize to within centimeters of the target altitude.

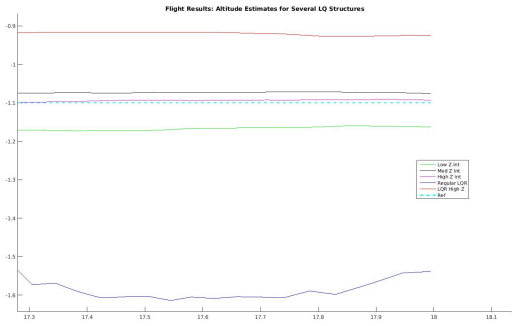

Fig. 6. Real Flight Results (zoomed in): Steady-state error is shown up close. Regular LQR is over 40cm away while LQI with high z integral gain settles within 1cm of reference.

LQI with medium z integral weight was the best controller observed in this experiment, eliminating steady-state altitude error to less than a centimeter. That is incredible accuracy considering the amount of noise in the sensors seen in Figure 4.

In real flight, adding integral feedback to LQR did significantly reduce steady-state altitude error as proposed.

Another important thing to consider is LQI's effect on other states. Similar to the logic used when evaluating the simulated controllers, it is important to compare the benefit of low steady-state altitude error versus the decrease in accuracy in other states.

As seen in Figures 7 and 8, the x and y trajectory is somewhat wobbly for both types of controller (LQR and LQI). There's approximately 50cm range of x and y values over the 20 second flight for each controller type.

While x and y are far less stable than z, it seems that LQI does not worsen other state stability much more than LQR. Similar analysis on the other 9 states could be performed to confirm this prediction.

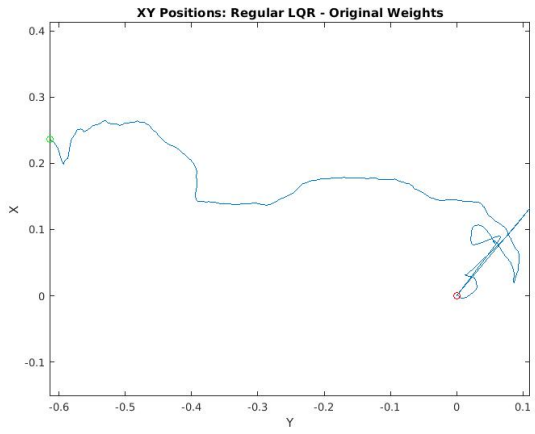

Fig. 7. XY trajectory with Standard LQR: The drone moved almost a full meter in y and 30cm in x during flight with a constant reference position. This is typical behavior seen all semester with all types of hover controllers.

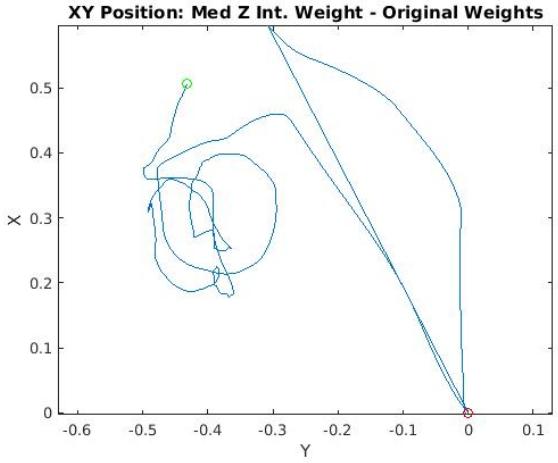

Fig. 8. XY trajectory with LQI: XY position with LQI is not expected to improve, but it also doesn't get much worse, which is important. So, LQI improves altitude accuracy and does not decrease accuracy of other states.

5. CONCLUSIONS

Overall, adding integral feedback improved performance of the LQR controller in real flight, but not in simulation. As expected, the steady-state altitude error was brought almost all the way to zero with a properly tuned LQI controller (z integral weight of 0.1) in real flight. Also as expected, there was some trade-off in performance, although it was only really seen in simulation.

LQI seems to be a viable extension to LQR in controlling a Parrot minidrone to reduce altitude steady-state error. Integral feedback had a more beneficial impact in flight than simply increasing the weight of a single state error.

More trials and analysis could be used to confirm that other state error does not suffer too much with the emphasis now on altitude error. Finally, a more comprehensive drone model would be needed to investigate why simulation and real-world flight data did not match up very well.

LQI performed better in real flights than regular LQR, and even LQR with doubled z weight. Therefore, in applications where low altitude steady-state error is imperative, integral feedback is likely a very useful method to improve on LQR.

6. ACKNOWLEDGEMENTS

Thank you to Parrot and the 16.31 course staff for providing the drones and development framework.

7. REFERENCES

[1] 16.31 Lecture #23 Notes: Fall 2014

Appendix A: MATLAB Code


```
%% LQR and LQI Gain Derivation
% Michael Everett
% Code Adapted from 16.31 toolbox

%% 1) Setup Bryson's rule

%Limits on states
pos_max     = 1;
att_max     = 0.35;
dpos_max    = 1;
datt_max    = 1;

%Limit on control input
motor_max = 500;

%Cost weights on states
pos_x_wght = 0.25/3;
pos_y_wght = 0.25/3;
pos_z_wght = 0.5/3;   % this state was doubled in one LQR controller
att_wght   = 0.175/3;
dpos_wght  = 0.175/3;
datt_wght  = 0.4  /3;

rho = 0.04;

%Pack weights and limits
weights= [pos_x_wght pos_y_wght pos_z_wght att_wght att_wght att_wght dpos_wght dpos_wght↙
dpos_wght datt_wght datt_wght datt_wght];
maxs   = [pos_max pos_max pos_max att_max att_max att_max dpos_max dpos_max dpos_max↙
datt_max datt_max datt_max];

%% 2) Compute Q and R cost matrices
Q           = diag(weights./maxs)/sum(weights);
R           = rho*diag(1./[motor_max motor_max motor_max motor_max]);

%% 3) Compute K_LQR and K_LQI, Simulate and Plot

Klqr = lqr(A,B,Q,R); % Regular LQR gain matrix

ns = size(A,1);
z = zeros(ns,1);
Qnew = Q; % Q from regular LQR
Rnew = R; % R from regular LQR still applies to LQI

Cnew = z';
Cnew(3) = 1; % Only adding 1 state (z_integral) from 12-state vector
Anew = [A z; Cnew 0]; % New A matrix
Bnew = [B;zeros(1,4);]; % New B matrix

figure;
s = 'Values: ';
l = ['x','o','^','d','>'];
for x=-1:1
    Qnew = [Q z; z' 10^(-x)]; % New Q with 13th state, z_integral
    Klqi = lqr(Anew,Bnew,Qnew,Rnew); % LQI gain matrix
    sim('sim_quadrotor.slx');
    plot(simout.Z.time,simout.Z.data);
    hold on;
```

```
end
plot(tlqr,dlqr); % Data from K_lqr simulation (can't automate)
legend('Z Int. Weight = 10','Z Int. Weight = 1','Z Int. Weight = 0.1','Standard↙
LQR');%,'mini','tiny','lqr');
hold off;

%% 4) Plot Experimental Results Together

figure;
hold on;
plot(lowz_x,lowz_y,'g',medz_x,medz_y,'k',highz_x,highz_y,'m',lqr_x,lqr_y,'b',lqrhigh_x,↙
lqrhigh_y,'r');
plot(ref_x,ref_y,'c--','LineWidth',2);
legend('Low Z int','Med Z Int','High Z Int','Regular LQR','LQR High Z','Ref')
title('Flight Results: Altitude Estimates for Several LQ Structures');
hold off;
```