Edward Venator

EECS 304 Spring 2012

Helicopter Competition Project Report

## Solution

numG = [125 50 35]

denG = [(1/90) 1 0]

## Methodology

My approach to this problem was to use a PID controller combined with a low-pass filter. The PID controller provides the control signal, and the low-pass filter accounts for sensor noise and quantization effects by removing high frequencies from the error signal. I chose not to implement a prefilter on the reference path, as I did not find it to have any appreciable effect on the results.

Although I originally considered using a low-Q notch filter centered in the 1 to 1.5 rad/sec range to remove the oscillatory behavior of the system, I found that the low frequency oscillations of the system were not very strong, and were only apparent when the system’s damping parameter was at its minimum. Based on this, I decided that the notch filter added unnecessary complexity to the controller.

After deciding on a PID controller, I began experimentally tuning the controller, validating my gains by testing them in the Simulink model. I began by setting the proportional gain and the filter corner frequency. In order to remove steady state error, I added integral gain. I then increased derivative and proportional gain to get as fast a controller as I could without it going unstable or being so oscillatory that it was in danger of crashing.

## Results

The figures and table below depict simulated results of the controller. For the most part, the controller yields a cost between 3.4 and 8. It appears that the gain of the system, k, has the greatest effect on the performance of the controller, with low gain systems consistently out performing high gain systems. This is explained by the oscillations that are introduced at higher gains. I could have reduced this problem by reducing my overall gain, but at the cost of increased tracking error. There is one worst-case condition where the cost is over 14. As one might expect, this is one of the high gain, low damping, long time delay cases. It is surprising how much this case departs from the overall performance of the controller, and I hope that the actual system does not operate in this region.

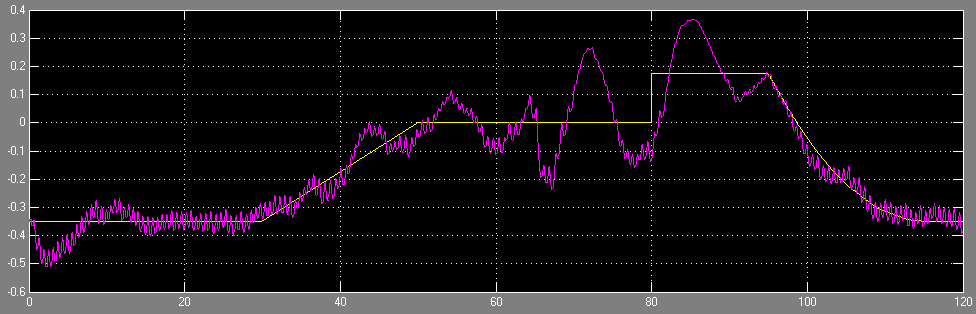


Figure 1: Worst case controller performance

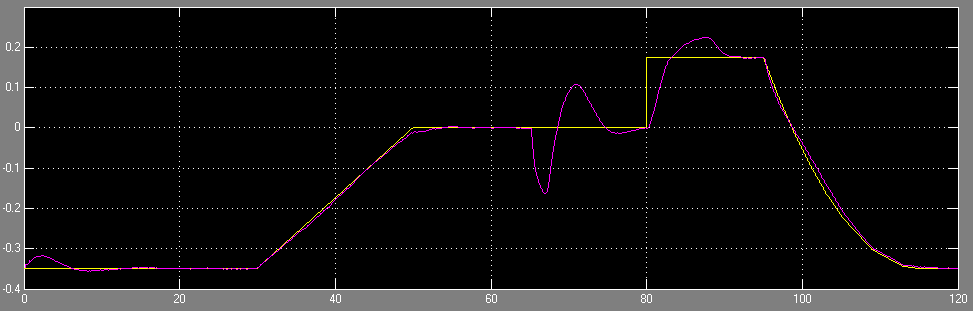


Figure 2: Best case controller performance

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **T** | **ωn** | **ζ** | **k** | **J** |
| .09 | 1.0 | .03 | .1 | 3.4049 |
| .09 | 1.0 | .03 | .15 | 3.9813 |
| .09 | 1.0 | .05 | .1 | 3.3939 |
| .09 | 1.0 | .05 | .15 | 3.9247 |
| .09 | 1.5 | .03 | .1 | 5.2782 |
| .09 | 1.5 | .03 | .15 | 8.2649 |
| .09 | 1.5 | .05 | .1 | 5.4248 |
| .09 | 1.5 | .05 | .15 | 7.0073 |
| .11 | 1.0 | .03 | .1 | 3.4804 |
| .11 | 1.0 | .03 | .15 | 4.7634 |
| .11 | 1.0 | .05 | .1 | 3.5423 |
| .11 | 1.0 | .05 | .15 | 4.4361 |
| .11 | 1.5 | .03 | .1 | 6.3267 |
| .11 | 1.5 | .03 | .15 | 14.4299 |
| .11 | 1.5 | .05 | .1 | 7.1175 |
| .11 | 1.5 | .05 | .15 | 9.8917 |

*Table 1: Simulated costs at boundary conditions*

## Conclusions

I could have approached this problem differently and used a different controller architecture, but I chose not to for a few reasons. First, PID controllers are intuitively easy to understand. This makes them easy to tune quickly if an accurate simulation is available. Second, the PID controller is relatively robust with respect to inaccuracies in the simulation. More advanced techniques like pole-zero cancelation and use of notch filters to cancel the effects of system dynamics are highly dependent on accurate knowledge of the system characteristics. Since I have had no access to the system itself, I cannot verify that the simulation is accurate. Even if the simulation is accurate, it has several parameters that can vary. If, for example, the natural frequency of the system varies from 1.0 to 1.5 rad/sec, it would break any controller that attempts to cancel out the system dynamics. A PID controller can be tuned reasonably well, even if the system parameters change slightly.

One thing that I would change if I were to implement this controller again is that I would revisit the idea of a notch filter to attenuate frequencies in the 1 to 1.5 rad/sec range. These frequencies are particularly visible in the worst case response, and attenuating them with an appropriately designed notch filter would improve the controller’s tracking ability and greatly reduce the “fuel use” of the vehicle.

Although my results are not perfect, I think they are nearly as good as can be expected from a PID controller. If the parameters of the system were known perfectly, it would be possible to optimally tune the controller, and rltool has a function to do so. However, the optimally-tuned PID controller operates right at the edge of stability. As a result, it is highly dependent on the accuracy of the system parameters. Varying the system parameters causes the optimally-tuned PID controller to go unstable. Given the range over which the parameters can vary, I am pleased with my controller.