EE324: Control Systems Lab
Experiment 4: Noise Cancellation in Headphones
Group 1 - Thursday

Harsh S Roniyar 22B3942 | Pranav Prakash 22B3945 | Aman Verma 22B3929

November 1, 2024

1 Objective

Design and implement an analog circuit for noise cancellation in headphones.

The specific objectives were:

- To achieve an attenuation of 20 dB (not necessarily exact 20, small attenuation also works if the system is stable) when a noise of 100 Hz frequency is applied.
- To design an analog compensator to stabilize the system, i.e., loop shaping of the loop transfer function.

2 Methodology and Design

We initially examined the headphone setup to understand its open-loop output transfer characteristics. Subsequently, we employed `MATLAB` tools to model the compensated stable system using a second-order approximation based on the gathered data. Following this, our attention turned to identifying suitable values of resistances and capacitors in the compensator, labeled as $C(s)$.
The specific aim was to choose such a compensator that, at a frequency of 100 Hz, achieves a 20 dB (or similar) attenuation in the output-to-noise transfer function. It's crucial to emphasize that this analysis is centered on noise cancellation headphones, with a primary emphasis on enhancing the system's efficiency in minimizing undesired noise.

2.1 Procedure

Frequency response analysis is performed on the headphone setup with sinusoidal waves given as input, from the function generator. Both the input and output are observed in the DSO, and then the magnitude and phase bode plots are plotted versus frequency. This is the open-loop characteristic of headphones.

2.1.1 Compensator Transfer Function

The compensator $C(s)$ is of second order and is given by the equation:

$$C(s) = \frac{as^2 + bs + c}{ds^2 + es + f}$$

Parameter	Value
a	
b	
c	
d	
e	
f	

Table 1: Values for the parameters.

3 Challenges Faced and Solutions

Problem 1: **Issue with Initial PID Parameters:** The code did not perform as expected with the initially set values of k_p, k_i, and k_d, resulting in unsatisfactory motion on the track.
Solution: After multiple iterations of tuning these parameters, we were able to achieve smooth turning around the curves.

Problem 2: **Unbounded Control Signals:** Initially, control signals were not bounded, which caused no visible variations even when we changed parameters.
Solution: We solved this by bounding the rotate signal between -1 and 1. Additionally, introducing checks to limit PWM values to 255 resolved the issue.

Problem 3: **Failure on Discontinuous Track Segments:** The system failed to handle discontinuous square blocks in the track.
Solution: By varying the threshold values (e.g., 50, 120) through multiple iterations, we were able to handle this track segment more effectively.

4 Results

The bot successfully completed the track in 26 seconds, consistently staying well under the 30-second mark. After incorporating the PID controller, the bot executed smooth turns throughout the course, as opposed to the sharp and jerky turns observed with the initial thresholding-based approach.
The final PID controller constants used were:

- k_p = 1.05
- k_i = 2.8 $\times$ 10^{-7}
- k_d = 0.49

5 Observations and Inference

- The implementation of the PID controller significantly improved the bot's ability to handle curves on the track, leading to smoother turns compared to the initial thresholding method.
- Fine-tuning of the PID constants (k_p, k_i, k_d) allowed for an optimal balance between stability and responsiveness, leading to faster track completion times.
- The marginal contribution of k_i (2.8 $\times$ 10^{-7}) indicates that integral control had minimal impact, likely due to the relatively short duration of the course and the real-time nature of the control adjustments.
- The overall improvement in performance demonstrates that the PID approach is far more efficient and adaptable for handling dynamic changes in the track.