SENSITIVITY AND DISTURBANCE REJECTION

LAB EXPERIMENT 15

Harel Ko
BS Electronics Engineering, Samar State University (of Aff.)
Institute of Electronics Engineers of the Philippines (of Aff.)
Catbalogan City, Philippines
Harelko999@gmail.com

Luis Nacional
BS Electronics Engineering, Samar State University (of Aff.)
Institute of Electronics Engineers of the Philippines (of Aff.)
Catbalogan City, Philippines
Obeywes7@gmail.com

***Abstract*—In control systems, maintaining performance in the face of disturbances is essential. This experiment aimed to investigate sensitivity and disturbance rejection using simulated systems, since physical components such as a DC motor and oscilloscope were unavailable. Through Python simulations, we modeled a second-order system and observed its response to disturbances both before and after applying a compensator. The sensitivity function was analyzed in both scenarios. Without compensation, the system exhibited poor disturbance rejection, with a high steady-state output. After implementing a proportional-integral (PI) controller, we observed significantly improved disturbance rejection, validating the role of compensation in reducing sensitivity to external inputs. The experiment demonstrates how compensators can shape the sensitivity function to improve robustness in control systems.**

I. RATIONALE

Sensitivity and disturbance rejection are crucial for maintaining system performance in the presence of external disturbances. This experiment explores the sensitivity function and disturbance rejection in feedback systems.

II. OBJECTIVES

- Calculate the sensitivity function for a given system.
- Measure the impact of a disturbance on the system's output and quantify the disturbance rejection.
- Implement a compensator to improve disturbance rejection and measure its effectiveness.

III. MATERIALS AND SOFTWARE

- Software: Python (with control library)

IV. PROCEDURES

1) Set up a system with a feedback loop (e.g., DC motor).
2) Apply a disturbance (e.g., a sudden change in load) to the system and measure its effect on the output.
3) Implement a compensator (e.g., PI controller) to improve disturbance rejection.
4) Measure the improvement in disturbance rejection after implementing the compensator.

V. OBSERVATION AND DATA COLLECTION

DATA COLLECTION:

https://drive.google.com/drive/folders/186e51rzYsN2s3K2mbOya_esmihDJAIP4

Click here to open the Drive

VI. DATA ANALYSIS

The plant system was modeled as:

$$G(s) = \frac{1}{s(s+2)}$$

This represents a typical second-order system, commonly used to approximate the dynamics of a DC motor. The unity feedback configuration was used, and the sensitivity function was defined as:

$$S(s) = \frac{1}{1+G(s)}$$

Before compensation, the magnitude of the sensitivity function was relatively high at low frequencies, indicating that the system was quite sensitive to low-frequency disturbances. A unit step disturbance introduced at the output showed a significant steady-state response, meaning the disturbance passed through with minimal rejection.

After implementing a PI controller:

$$C(s) = 10 + \frac{5}{s}$$

the sensitivity function flattened at low frequencies, which corresponds to better attenuation of persistent (low-frequency) disturbances. The disturbance response showed that the output quickly stabilized near zero, indicating a strong rejection capability compared to the uncompensated system.

Both the Bode magnitude plots of the sensitivity function and the time-domain step responses were used to quantify and visualize the disturbance rejection.

VII. DISCUSSION AND INTERPRETATIONS

From the simulation, it was evident that the uncompensated system struggled to reject disturbances. This behavior is explained by the shape of the sensitivity function, which showed poor attenuation in the low-frequency range. A control system with such characteristics is prone to being influenced by disturbances that vary slowly over time—a typical issue in real-world physical systems like motors under varying loads.

Upon implementing the PI compensator, we effectively increased the low-frequency gain of the controller. This had the dual effect of improving the system's tracking accuracy and reducing its sensitivity to steady disturbances. The reduced gain in the sensitivity function's Bode plot at low frequencies confirms this behavior. In essence, the compensator reshaped the system's frequency response to improve robustness and maintain output stability.

This experiment also underscores an important trade-off in control design: while we aim to minimize sensitivity to disturbances, we must be cautious not to introduce instability by excessively boosting controller gain.

VIII. CONCLUSION

Through this simulation-based experiment, we were able to demonstrate the significance of sensitivity and disturbance rejection in control systems. We learned that an uncompensated system is highly vulnerable to external disturbances, which can drastically affect its output. However, by designing and implementing a compensator—specifically a PI controller—we achieved improved disturbance rejection, as evident in both the sensitivity function and the system's time response.

Even without physical hardware, simulation tools like Python allowed us to deeply understand how feedback and compensation shape the behavior of control systems in the presence of disturbances. This reinforces the theoretical concepts and provides a practical foundation for future real-world applications.