**Getting Started with the Sliding Controlled Tower of Hanoi**

The Tower of Hanoi GUI allows its user to adjust the parameters of a robot arm controller that attempts to make the robot arm move in a path compatible with the Tower of Hanoi game. The robot arm should make a single move by picking up a piece from the left-hand tower and moving it to the right-hand tower. In its default configuration, the arm is very good at making this move – the default estimates of the arm’s parameters are accurate to the actual parameters. The user can adjust these estimates to make the arm less effective, and then add sliding control to correct for these mistaken parameters.

When the user first starts the Controlled Hanoi GUI, they are greeted with the screen shown in Figure 1. The user can enter estimates of the system’s parameter in the system parameter panel in the top-left. The initial estimates are equal to the actual system’s values, so any changes will reduce the performance of the arm. The user can change the controller’s estimates of the load mass (mL) and the joint friction coefficient (b). The user can also set the target move time in this panel. This adjusts how quickly the arm will attempt to move between target points.

Below the system parameter panel is the control parameter panel. This panel allows the user to configure the sliding control algorithm used to compensate for unknown parameters. By default, the sliding control is rather weak, with a gain of 0.1. Increasing this gain to 10 or 20 causes the system to more quickly compensate for mistaken system parameters. This panel also allows the user to set the value of epsilon, the parameter that limits the output of the sliding control for small error values. Increasing epsilon increases the system’s resistance to chatter, but reduces the performance of the sliding control.

When the user is satisfied with their choice of parameters, the “Plot!” button runs a simulation of the robot arm with the chosen parameters. The results of this simulation are shown in several ways. The upper-left plot shows an animation of the robot arm moving through 2d space. The blue path represents the arm’s trajectory, while the green path represents the ideal trajectory. The upper-right plot shows the error between the ideal trajectory and the actual trajectory. The lower-left plot shows the torque applied at each joint versus time, and the lower-right plot shows the angle of each joint versus time.

In addition to these plots, the system tabulates the completion time, integral square position tracking error, integral square applied control torque, and maximum position deviation during moves 2 and 4 (the vertical moves). An example set of outputs is shown in Figure 2.