**Hands-on Lab:**

**LabVIEW – Simulink PID**

Proportional-Integral-Derivative (PID) control is the most used method to make a system perform as desired. In a previous lab, Simulink was introduced to simulate the LEGO Damped Compound Pendulum (DCP). That lab concluded: (1) the theoretical simulated response matched those of the DCP’s experimental results; (2) the DCP’s step response showed the pendulum oscillating but eventually reached a steady-state angle, but took a long time. This lab applies and simulates PID control of the DCP.

**Preamble:** **PID and DCP Block Diagram**

For reference, as illustrated in the previous lab, Figure A, B, and C:

**Figure A**: DCP free body diagram (left) and both physically measured and calculated values (right)

thrustTesterFbd012905

|  |  |  |
| --- | --- | --- |
|  | Pendulum Length | 0.265 [m] |
|  | Pivot to CG distance | 0.1325 [m] |
|  | Mass of Pendulum | 0.0137 [kg] |
|  | Damping ratio | 0.033 |
|  | Natural frequency | 7.32 [rad/s] |
|  | Moment of Inertia | 0.00033 [ |
|  | Coefficient of Friction | 0.00016 [] |
|  | Steady-state angle | at 50% motor power |

**Figure B:** Block diagram relating the motor power input, to the DCP’s angle output



**Figure C**: Transfer function relating the motor input (as a percentage) and angle output (in radians)



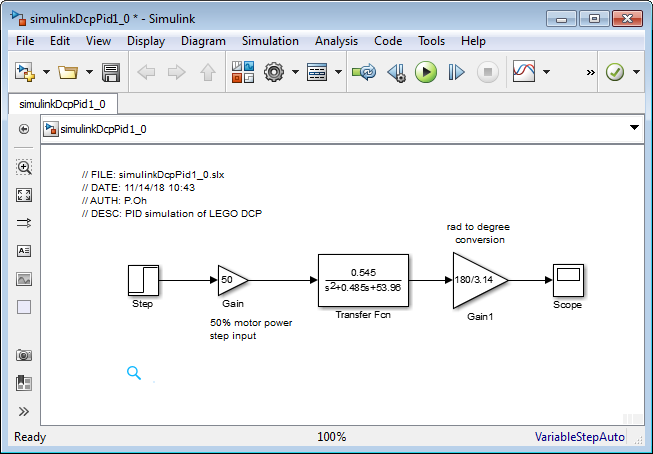
The box in **Figure C** is called the open-loop transfer function (OLTF) and relates the input (motor power command in units of %) to the output (DCP’s angle in units of radians). The specific values in the box’s numerator and denominator were specific to the DCP used in the previous lab. These values may differ for another DCP, depending the mass, pendulum length, and motor-prop properties. As such, one would replace the values with obtained from system identification, into **Figure B** to yield **Figure C**.

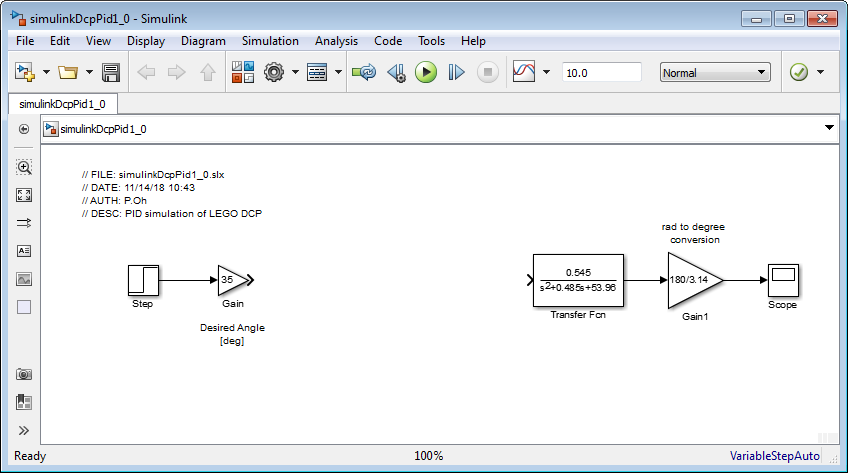
**Concept 1:** Simulink Block Diagram

**Step 1:** Open Simulink and Create the Open-loop Transfer Function (OLTF)

The previous lab created simulinkDcpStepInput1\_0.slx (**Figure 1-1A left**) which had the OLTF. Save this Simulink file as simulinkDcpPid1\_0.slx and delete the wire between the 50% gain block and the OLTF (**Figure 1-1A right**). For this simulation, let us set the desired DCP angle for. Thus, double-click the 50% gain, and under Block Parameters, replace 50 with 35. Edit the text box, to underscore that this gain represents a desired DCP angle of.

**Figure 1-1A:** The former Step input Simulink program simulinkDcpStepInput1\_0.slx already created the OLTF (left). Rename the 50% gain block, edit comments and save as simulinkDcpPid1\_0.slx (right).

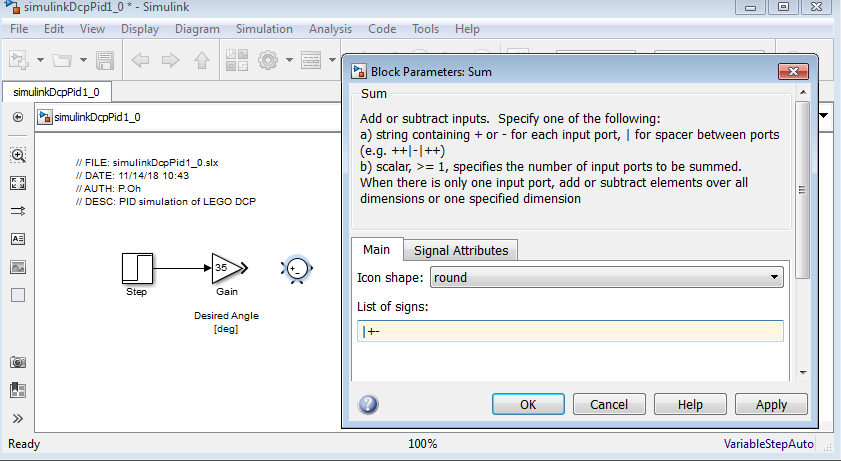


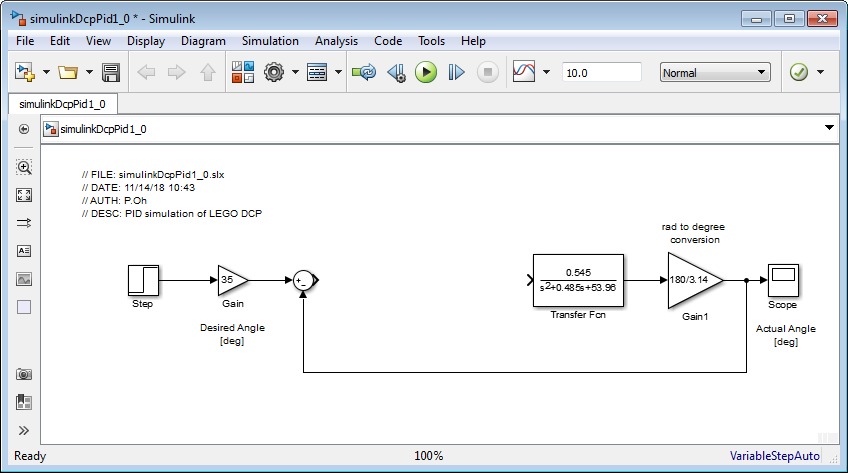


**Step 2:** Add the Error Signal

PID control operates by performing functions on the error signal. This signal is the difference between the desired and actual angles. Under the Library Browser, select Commonly Used Blocks and click-and-drag the Sum block into the Simulink canvas. By default, the Sum block adds 2 inputs. Double-click the Sum block and in Block Parameters change the last addition to a subtraction (**Figure 1-2A left**) and wire blocks (**Figure 1-2A right**).

**Figure 1-1A**: Red circle shows where to configure the Sum block to subtract signals (left). Wiring (right) now reflects an error signal that is the difference between Desired and Actual angles [deg]

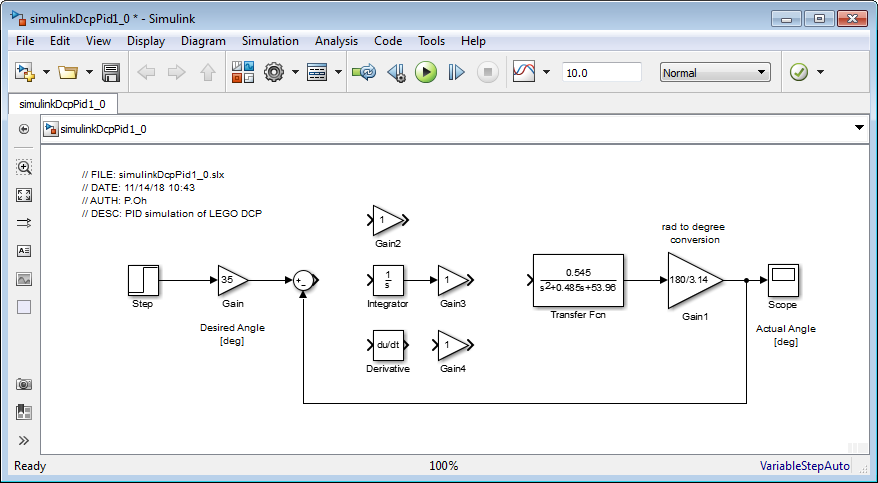




**Step 3:** Create the Proportional, Integral and Derivative Blocks

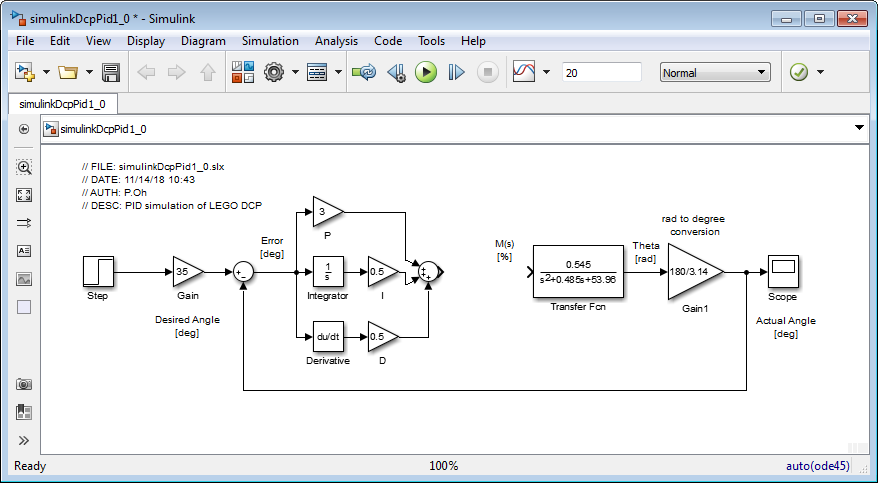
PID respectively multiplies, integrates and differentiates the error signal. Under the Library Browser – Continuous, click-and-drag the Derivative and Integrator blocks into the Simulink canvas. Under Commonly Used Blocks, click-and-drag the three Gain blocks. Rearrange blocks to resemble **Figure 1-3A**.

**Figure 1-3A:** Proportional simply multiples the error with a constant, hence a Gain block is used. Integral performs a scaled integration on the error, hence the Integrator and Gain blocks. Lastly, Derivative performs scaled differentiation on the error, hence the Derivative and Gain blocks.



Next, add a second Sum block. Double-click it and under Block Parameters, and a + so that the Sum block allows the addition of 3 signals. Add text comments for blocks and wires to identify signals and units (**Figure 1-3B**).

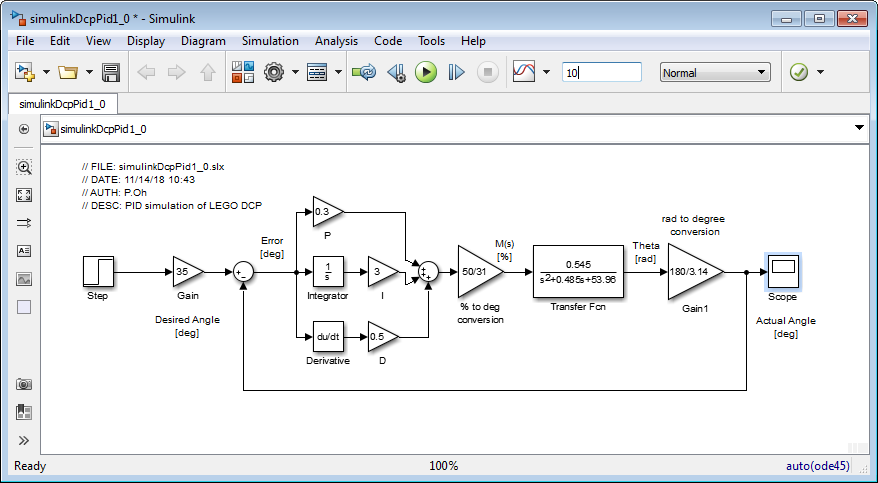
**Figure 1-3B**: Add and configure a second Sum Block so that it can add the 3 signals from the P, I, and D blocks (red arrow). Add text comments to clarify each signal and units (e.g. M(s) and [%]) in the appropriate areas (red circles)



As **Figure 1-3B** shows, the output of the PID summation block will be in degrees. However, the input into the OLTF is a percentage (i.e. motor power). Recall in previous derivations, a 50% motor power input yielded a steady-state angle of 31 degrees. Assuming linearity, these values can be used to convert degrees to percentage, namely .

Add a Gain block between the PID summation and the OLTF and configure it be 50/31 (**Figure 1-3C**). Save the resulting program (i.e. simulinkDcpPid1\_0.slx).

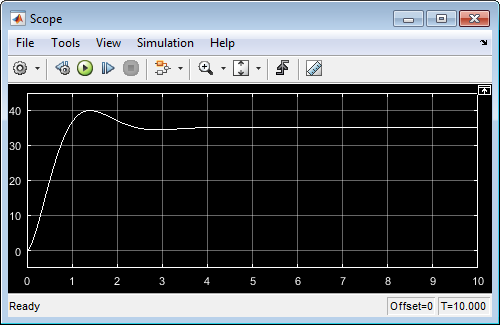
**Figure 1-3C:** Red arrow points to the Gain block used to convert the PID control signal (in degrees) to a motor power input in percentage. Simulation time (red circle) was set to 10 seconds.



**Step 4:** Execute PID

**Figure 1-3C** also shows gains values to scale the proportional, integration, and differentiation. These values are respectively. The figure also sets the simulation to 10 seconds. Saving and executing the program results in some overshoot and convergence to the desired steady-state angle (**Figure 1-4A**).

Figure 1-4A: PID output for. One sees that after about 3 seconds,



**Congratulations: LEGO DCP PID Simulation Completed**

**Exercise 1:**

* 1. What PID gain values will result in no overshoot i.e. Show your resulting figure
  2. Add an additional scope to display the value of in percentage for the gain values you chose in 1.1. If this re-do 1.1 with PID gains such that the motor power command remains between 0 and 100%
  3. What PID gain values will result in a reduced settling time i.e. seconds. Show your resulting figure.
  4. If this re-do 1.3 with PID gains such that the motor power command remains between 0 and 100%