Control Engineering: Kentridge Inverted Pendulum Apparatus PP-300

Dean Devereaux

Dept. of Electrical and Electronic Eng.,

Cork Institute of Technology.

Email: dean.devereaux@mycit.ie

**Abstract — This report consists of the steps and methods used to design a validated model system for the Kentridge inverted pendulum apparatus. This report will go through in detail the calculations to create a model two transfer function which replicates the input of the arm and corresponding pendulum response. Both a proportional controller and posicast controller were simulated and implemented within a feedback system of the kentridge rig.**

# Introduction

The objective for this module was to create a transfer function for the Kentridge inverted pendulum apparatus (figure 1) for both open loop and closed loop model systems. A single loop controller was used with the validated closed loop system along with a posicast controller for open and closed loop. The group handling the rig had three models to examine which would create a transfer function as the end product. These models were divided between the group and examined individually. MATLAB was used for testing the rig and simulating the model systems.

The root locus tool within MATLAB was used to create a single loop controller for the system.

**Figure 1 Kentridge Rig**

# Open Loop

Using the Kentridge rig, data was gathered in an open loop MATLAB Simulink model which interfaced with the rig. This model inputs a pulse into the motor of the rig and the resulting response of the cart and pendulum gets converted to digital data. Giving the model a pulse input of 3V for 0.3 seconds gave the results seen in figure 2. There was a delay of 2 seconds before the pulse to allow no data lose at the beginning of the run.

**Figure 2 Gathered Data - Open Loop**

Three models were to be examined to create three separate transfer functions. These are:

Model One - The pulse input and the response of the cart.

Model Two - The cart movement and response of the pendulum

Model Three - Pulse input and response of the pendulum

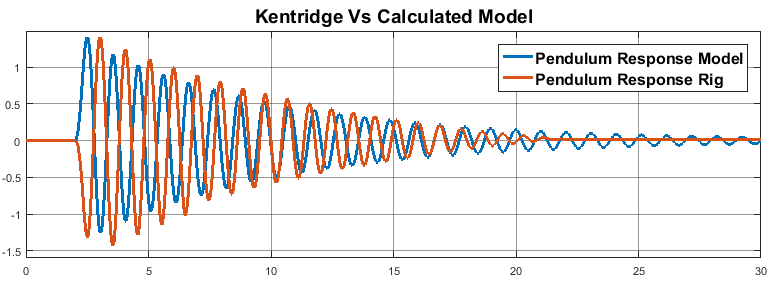
For this report a transfer function will be created to replicate the pendulum response based on the cart movement. Using the amplitude values of the pendulum in figure 3 a value for ωn and ζ was calculated by taking the first two peaks. Period Td was calculated from oscillation of the pendulum. Gain value Kp was chose after comparing model with rig. This would help with the model amplitude.

**Figure 3 Pendulum Peak Values**

X1 = 1.415, X2 = 1.245

|  |  |
| --- | --- |
| **Td** | 1.036 |
|  | 0.0206 |
| **ωn** | 6.1192 |
| **Kp** | 0.095 |

**Table 1**

Figure 4 shows the results from the model system of G(s)[1] calculated above. The s term used in the numerator cancels out the non-zero steady state of the model response as it will attempt to replicate the cart steady state value. By adding the s term, the function becomes a differentiator and cancels the effects of the integral within the cart response. In order to reverse the model response seen in figure 4 an initial negative “-s+1” was introduced to the numerator which would cause the response to dip rather than peak, replicating the rig.

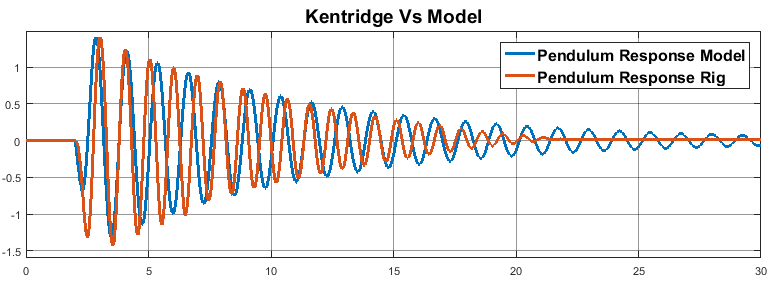
**Figure 4** **Transfer Function** **Attempt**

Kp value was altered as the initial negative would influence its role with the amplitude. ζ and were also altered through trial and error to improve the model’s replication. Results for this can be seen on figure 5 which used G(s)[2] function.

**Observation:** ζ gives the right amount of dampening to closely replicate the movement of the real pendulum. Open loop systems aren’t adequate enough to be used for controller design as they don’t monitor the output response therefore the response cannot be corrected if an error occurs.

|  |  |
| --- | --- |
|  | 0.0223 |
| **ωn** | 5 |
| **Kp** | 0.023 |
| **Initial Negative** | -0.9 |

**Table 2**



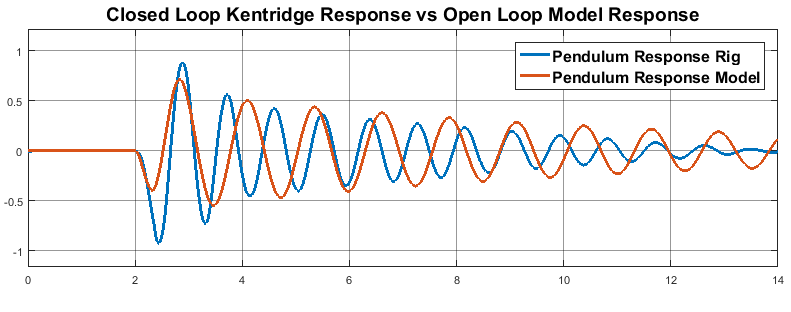
**Figure 5 Transfer Function Attempt #2**

# Closed Loop

Closed loop data was gathered from the rig based on the pendulum response and compared with the transfer functions. Closed loop systems are faster and more precise to open loop due to the feedback from the response which alters the input based on error and corrects the external or internal problem to continue creating a preferred output response. Model two transfer function could not be created in closed loop due to the model being created from both cart and pendulum feedback.

Instead this had to be reproduced with the model one closed loop transfer function and compared with the real pendulum. What should be expected as an end result is a similar transfer function to Model 3’s validated transfer function.

**Figure 6 Real Data of Closed Loop System**

****Table 2 shows the validated closed loop of model one used with G(s)[2] and the result of their multiplication to create an overall validated model G(s)[3]. Fig. 7 shows G(s)[3] against the real closed loop data.

**Figure 7 CL Rig vs OL Model**

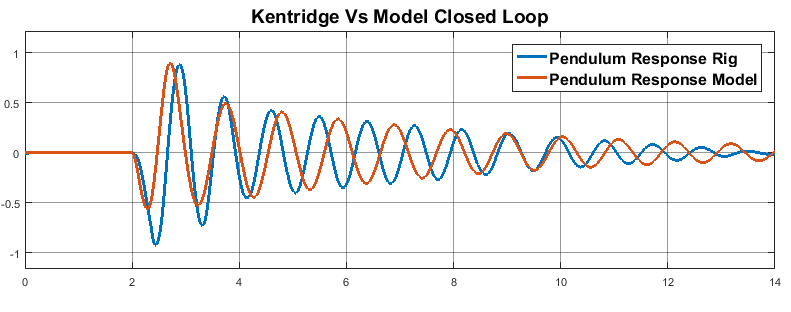
|  |  |
| --- | --- |
| **Model One** |  |
| **G(s)** |  |

Changes had to be made to Model two in order to create an accurate transfer function. Model one cannot be changed due to it already being accurately tested against closed loop data of the arm. The gain Kp in model two can be changed to control the amplitude of the response and ζ which will control the level of dampening of the pendulums oscillation.

Table 3 shows the altered values of Kp, ζ, Wn and initial negative which improved the transfer functions oscillation and dampening to be accurate enough to the closed loop data. The comparison is shown in fig.8 using G(s)[4] as the transfer function, the first two oscillations are more than accurate enough for the transfer function to be used as the model in simulation.

**Table 3 Transfer Functions**

|  |  |
| --- | --- |
| **ζ** | 0.03 |
| **Kp** | 0.022 |
| **Wn** | 6 |
| **Initial Neg** | -1.2 |



**Table 4 Altered Values**

**Figure 8 Closed Loop Trans Func Attempt #2**

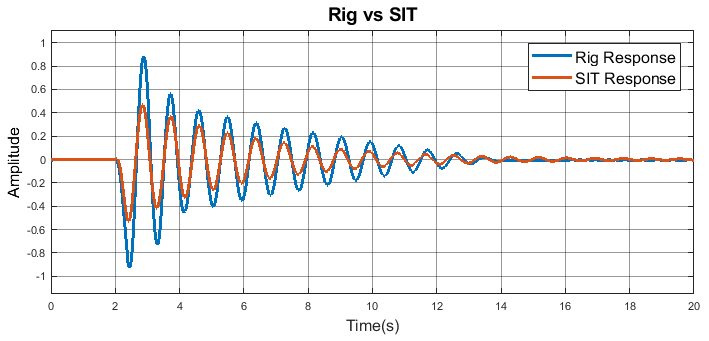
# System Identification Toolbox

The system identification toolbox was used with closed loop data as an alternative method for creating a validated closed loop transfer function for the model. With this tool the pulse was imported as input data and the pendulum as output as time domain data. When inputting the data the sampling and starting time must also be input and replicate the rig. In this case the sampling time is 0.001s and starting time 0.

**Figure 9 SIT transfer function calculated**

The transfer function created was set up as a 3rd order system. This was decided as the calculated transfer function G(s)[4] was also 3rd order. 2nd order would also suffice.

Fig. 9 shows what was calculated for the SIT transfer function against the pendulum response in closed loop. Fig. 10 then shows the exact same data against each other but in Simulink. The SIT transfer function has great oscillation which is in sync with the real response however the amplitude doesn’t match the system therefore G(s)[4] will be used as the validated closed loop transfer function.



**Figure 10 SIT transfer function simulation**

# Controller Design

Two closed loop controllers and an open loop controller are discussed for the controller design. Proportional will be observed as a single loop controller and Posicast in open loop and with feedback. The aim of these designs is too have the model and rig work at a specified rate. The specifications can be seen in table 4. For the Kentridge pendulum a period equals 1.03 seconds

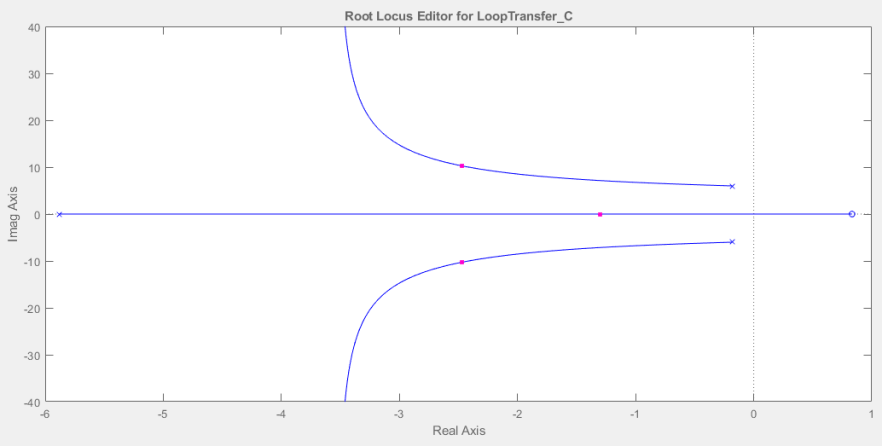
|  |  |  |
| --- | --- | --- |
|  | **Arm** | **Pendulum** |
| **Reference Point** | 3 | 0 |
| **Overshoot** | >5% | 50% reduction in oscillation |
| **Settling Time** | ~1 second | 6-12 periods |

**Table 5 Controller Design Specifications**

# Single Loop Controller Design

To design a single loop controller the validated closed loop transfer function created earlier was used with the root locus tool. The root locus tool allows a visual view of the poles and zeros for a controller. The values for the poles and zeros can be altered to better the response of the system.

# Proportional Controller



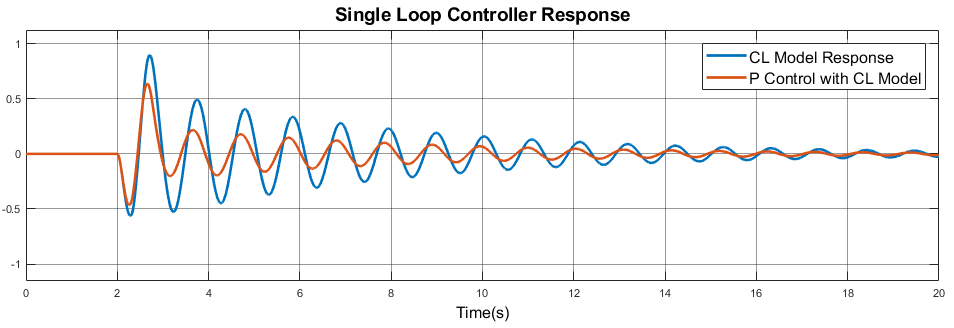
**Figure 11 Rltool editor**

The proportional controller was designed using the root locus tool found on MATLAB. This tool allows the controller values to be chosen based on the poles and zeroes from the S plane. In order to get a fast response the pole of the root locus need to be distant from the centre point and for a reduction in overshoot must be close to the real axis. For the system to be stable the poles must also be on the left hand side of the centre point. Table 5 shows the pole values seen in fig. 11. This created a Kp value of -3.3 for the proportional controller.

**Table 6 Pole values**

|  |  |
| --- | --- |
| **Poles** | -1.3 |
| -2.47 + 10.3i |
| -2.47 – 10.3i |

Fig. 12 shows the P controller with the model system. In comparison with the model system it reduces the oscillation and allows it to settle much faster. Compared with the specification the pendulum settles after 9 oscillations however the first period does not reduce by 50%.



**Figure 12 P Controller response**

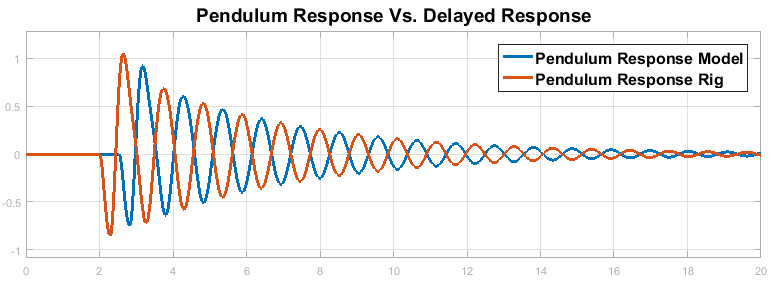
# Posicast Feedback Controller Design

Posicast is a controller which is used for systems were dampening is present. The controller involves a feedforward signal with a gain and delay. In this situation posicast will create a second pulse input which when given the correct delay and gain value will cancel the pendulum response from first pulse.

In this section a reasonable gain and delay to decay the oscillation of the pendulum were calculated. The delay should be half the period and will cause the delay response to be out of phase by 180 degrees when compared to the pendulum response. The gain for the posicast will reduce the amplitude of the oscillation. The reason for this gain is because the pendulum is dampening therefore when the delay signal occurs it will be in phase with an oscillation with similar amplitude

# Simulation

To calculate posicast values for the validated model G(s)[4] the feedback was removed to calculate posicast values since it is an open loop technique. Figure 13 shows the response of this model without feedback which was the basic of calculating both gain and delay.

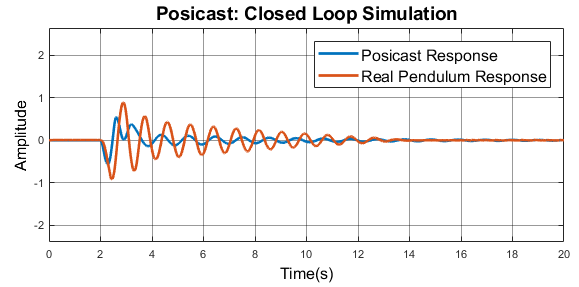
Figure 14 displays the comparison when table values are implemented for a posicast controller. The transfer function response and posicast response can be seen out of phase by 180 degrees. This will cancel the oscillation of the pendulum response based on the amplitude of the delayed response.

**Figure 13 CL\_ Pendulum response without feedback**

Figure 15 shows the response when posicast is applied and feedback has been added back to the system. Compared with specifications the max oscillation is reduces by 50% once the delay pulse is introduced after one period. The pendulum settles after five periods just leaving it out of the aim of 6 – 12 periods.

**Figure 14 Pendulum vs Delay response**

|  |  |
| --- | --- |
| **Gain** | 0.88 |
| **Delay** | 0.53 |



**Figure 15 Posicast controller response**

# Rig

With the rig, posicast gathered data from the open loop test of the rig where a gain and delay was calculated seen in table 7.

|  |  |  |
| --- | --- | --- |
| **Controller Type** | **Posicast Feedback Controller** | |
| **Sim or Real** | Real | Simulation |
| **Controller Gain** | 0.9 | 0.88 |
| **Controller Delay** | 0.62 | 0.53 |
| **Pendulum Peak to Peak Deviation (rads)** | 1.4563 | 0.9149 |
| **Pendulum Settling Time(s)** | 4.8 | 6 |
| **Control Signal Maximum Amplitude(V)** | 3.6 | n/a |

These values were then used with the rig in a closed loop system. The response for this in fig. 16 shows when the delayed pulse gets introduced it reduces the trought of the 2nd period but the peak remains at the amplitude of the system. The 3rd period has a small reduction in amplitude and then rapidly drops once the 4th period occurs. The response settles after ~3 seconds but the amplitude of the oscillations present doesn’t dampen by a lot in the first 2 oscillations once the second pulse occurs. With posicast control the oscillation gets reduced by 99% and the settling time is 3 periods therefore this only meets half the specifications

**Figure 16 Posicast controller with Kentridge Rig**

**Table 7 Posicast Controller Results**

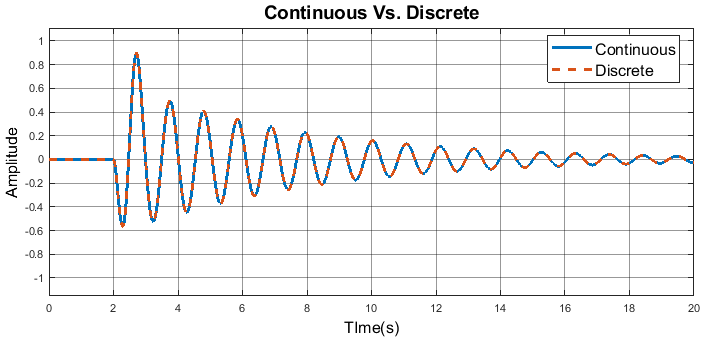
# Continuous to Discrete

The continuous transfer function G(s)[4] was converted to a discrete time form by using the c2d function on MATLAB. A sampling time for discrete was chosen based on 🡪

The half way point was chosen as a sampling time and calculated as:

.

Fig.16 shows the continuous transfer function against the discrete transfer function. The discrete response is identical to continuous and shows no error. When used with posicast and single loop controller the result are also identical showing no altercations. G(z)[1] is the transfer function for the discrete model.



**Figure 17 Continuous vs. Discrete**