Rotary Motion Servo Plant: SRV02

Rotary Experiment #16: Solar Tracker

Solar Tracker using QUARC

Instructor Manual

How to contact Quanser:

+1 (905) 940-3575	Telephone
+1 (905) 940-3576	Facsimile
119 Spy Court Markham, ON Canada L3R 5H6	Mail
http://www.quanser.com	Web
mailto://info@quanser.com	General information

Table of Contents

1. Introduction

The objective of this experiment is to develop a feedback system that controls the position of the camera to track the moving light source. Using the proportional-integral-derivative (PID) family, a compensator is designed to meet a set of specifications.

The following topics are covered in this laboratory:

- Model the rotary servo using frequency response.
- Design a proportional-velocity (PV) control for the servo/camera to track a step reference signal.
- Identify the sensor gain of the camera to measure the angle between camera and the light source.
- Using the same PV controller designed previously, implement the solar tracker control (i.e. camera tracks the light position).
- Add integral control to the solar tracker controller.

Regarding Gray Boxes:

Gray boxes present in the **instructor manual** are not intended for the students as they provide solutions to the pre-lab assignments and contain typical experimental results from the laboratory procedure.

2. Prerequisites

In order to successfully carry out this laboratory, the user should be familiar with the following:

- Data acquisition card (e.g. Q8), the power amplifier (e.g. Voltpaq), and the main components of the SRV02 (e.g. actuator, sensors), as described in References [1], [4], and [5], respectively.
- Wiring and operating procedure of the SRV02 plant with the amplifier and data-acquisition device, as discussed in Reference [5].
- Transfer function fundamentals, e.g. obtaining a transfer function from a differential equation.
- Laboratory described in Reference [7] in order to be familiar using QUARC with the SRV02.

3. Overview of Files

Table 1 below lists and describes the various files supplied with the laboratory.

File Name	***Description***
Solar Tracker Laboratory – Student Manual.pdf	This laboratory guide contains pre-lab and in-lab exercises for the Quanser Solar Tracker using QUARC.
setup_st_srv02_model.m	This Matlab script is used in the *Frequency Response* experiment. Run this file only to setup the modeling laboratory.
setup_st_srv02_pos.m	Run this script to set the all the necessary parameters for the *SRV02 Position Control* lab.
setup_st_sensor_calib.m	This Matlab script is used in the *Sensor Calibration* experiment. Run this file only to setup the modeling laboratory.
setup_st_tracker.m	Run this script to set the all the necessary parameters for the *Solar Tracker Control* experiment.
d_model_param.m	Calculates the SRV02 model parameters *K* and *tau* based on the device specifications *Rm*, *kt*, *km*, *Kg*, *eta_g*, *Beq*, *Jeq*, and *eta_m*.
config_srv02.m	Returns the configuration-based SRV02 model specifications *Rm*, *kt*, *km*, *Kg*, *eta_g*, *Beq*, *Jeq*, and *eta_m*, the sensor calibration constants K_POT, K_ENC, and K_TACH, and the amplifier limits VMAX_AMP and IMAX_AMP.
calc_conversion_constants.m	Returns various conversions factors.
q_st_srv02_mdl	Use this Simulink diagram with QUARC to run the frequency response experiment on the SRV02 system.
q_st_srv02_pos.mdl	Simulink file that implements a closed-loop PV position controller on the SRV02 system using QUARC.
q_st_sensor_calib.mdl	Simulink model that implements a calibration algorithm to find the camera sensor gain using QUARC.
q_st_tracker.mdl	Use this Simulink diagram with QUARC to implement a PIV position controller on the servo for the camera to track the light source.
Solar Tracker Laboratory – Instructor Manual.pdf	Same as the student version except the gray boxes are no longer shaded to reveal the solution to the pre-lab and in-lab exercises.
solar_tracker.mws	Maple worksheet used to design the position controller for the experiment. Waterloo Maple 9, or a later release, is required to open, modify, and execute this file.
solar_tracker.html	HTML presentation of the Maple Worksheet. It allows users to view the content of the Maple file without having Maple 9

File Name	***Description***
	installed. No modifications to the equations can be performed when in this format.
d_pv_design.m	Matlab script file that calculates the control gains *kp* and *kv* based on the model parameters *K* and *tau* as well as the peak time an overshoot specifications *tp* and *PO*.
sample_data_*.mat	Sample measured data found in the *results* folder of each experiment. The results folder also includes script to plot this data.

Table 1: Files supplied with the SRV02 Position Control experiment.

4. Solar Tracker Laboratories

4.1. Frequency Response

The objective of this experiment is to identify the model parameters of a DC servo motor when mounted with a camera load.

4.1.1. Servo Model

The angular rate of the SRV02 load shaft with respect to the input motor voltage can be described by the following first-order transfer function

$$\frac{\Omega_l(s)}{V_m(s)} = \frac{K}{\tau s + 1} \qquad [1]$$

where the $\Omega_l(s)$ is the Laplace transform of the load shaft rate $\omega_l(t)$, $V_m(s)$ is the Laplace transform of motor input voltage $v_m(t)$, *K* is the steady-state gain, τ is the time constant, and *s* is the Laplace operator.

4.1.2. In-Lab: Finding the Model Parameters

In this laboratory, a sine wave of varying frequency is fed to the DC motor and the resulting speed is recorded. The bode plot of the system is constructed using the collected data and then, as discussed in Reference [8], the model parameters are found. The Simulink diagram used with QUARC to do this is shown in Figure 1.

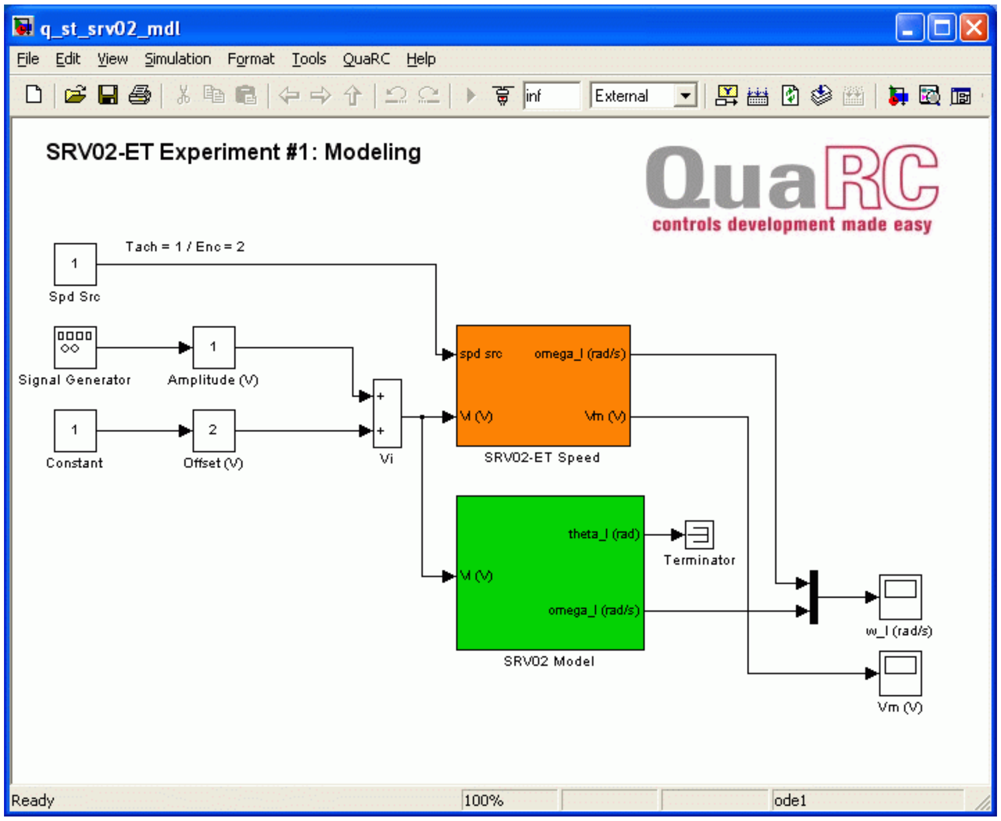

Figure 1: Simulink diagram used with QUARC to run the frequency response.

Note to Instructors:
The instructor can elect to make the student go through the modeling. In that case, see the SRV02 Modeling lab. Make sure everything is connected as per dictated in the SRV02 User Manual. The results from doing the frequency response laboratory are given below.

Alternatively, you can just supply the student with the K and τ parameters and move on to the position control exercise in Section 4.2,

Follow the steps below:

1. In the setup_st_srv02_model.m script, set MODELING_TYPE = MANUAL.
2. Run the setup_st_srv02_model.m script. This sets the model parameter to K = 1 and τ = 0.1.
3. In the Simulink diagram, double-click on the *Signal Generator* block and ensure the following parameters are set:
   Wave form: sine
   Amplitude: 0.0

Frequency: 1.0
Units: Hertz

4. Set the *Amplitude (V)* slider gain to 0.0 V.
5. Set the *Offset (V)* block to 2.0 V.
6. Open the load shaft speed scope, *w_l (rad/s)*, and the motor input voltage scope, *Vm (V)*.
7. Click on QUARC | Build to compile the Simulink diagram.
8. Select QUARC | Start to begin running the controller. The SRV02 unit should begin rotating in one constant direction. The scopes should be reading something similar to as shown in figures 2 and 3. Note that in the *w_l (rad/s)* scope, the yellow trace is the measured speed while the purple trace is the simulated speed (generated by the *SRV02 Model* block).

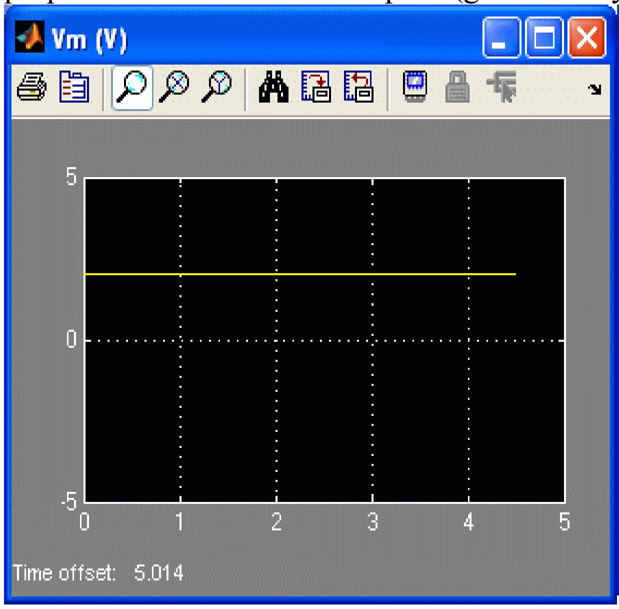

Figure 2: Constant input motor voltage.

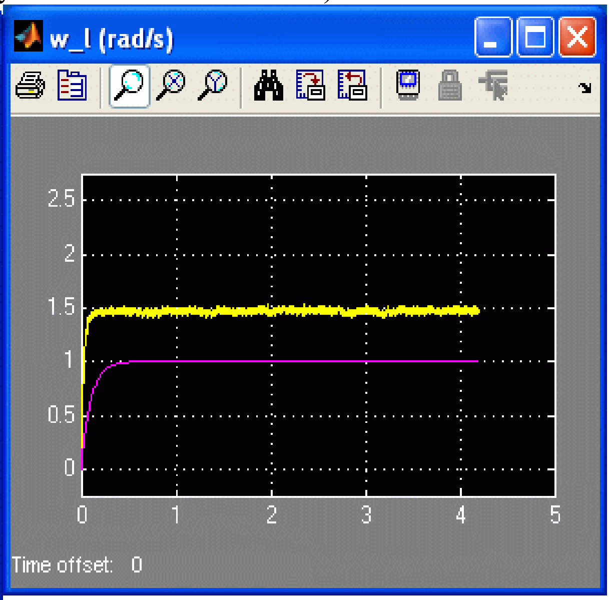

Figure 3: Load shaft speed response to a constant input.

9. Measure the speed of the load shaft. The measurement can be done directly from the scope. Alternatively, users can use Matlab commands to find the maximum load speed using the saved *w_l* variable. When the controller is stopped, the *w_l (rad/s)* scope saves the last 5 seconds of response data to the Matlab workspace in the *w_l* parameter. It has the following structure: *w_l(:,1)* is the time vector, *w_l(:,2)* is the measured speed, and *w_l(:,3)* is the simulated speed. In either method, enter the speed measurement in Table 2 under the f = 0 Hz row.

f (Hz)	*Amplitude (V)*	*Maximum Load Speed (rad/s)*	*Gain: |G(ω)| (rad/s/V)*	*Gain: |G(ω)| (rad/s/V, dB)*
0.0	2.0	3.33	1.67	4.43
1.0	2.0	3.16	1.58	3.97
2.0	2.0	3.07	1.54	3.72
3.0	2.0	2.87	1.44	3.14
4.0	2.0	2.69	1.35	2.57
5.0	2.0	2.46	1.23	1.8
6.0	2.0	2.33	1.17	1.33
7.0	2.0	2.21	1.11	0.87
8.0	2.0	2.00	1	0

0	1	2

Table 2: Collected frequency response data.

10. As instructed in *Rotary Experiment #1: Modeling* (Reference [8]), calculate the gain when the input signal is a constant signal and express the values in both the linear and decibel (dB) units. Show your calculations below and enter the resulting numerical value in the f = 0 Hz row of Table 2. Remark that this is the steady-state gain of the system. Enter its non-decibel result in Table 3 below.

Solution:

The frequency response magnitude at 0 Hz, i.e. ω = 0, is

$$\left|G_{wl,\,v}(0)\right| = \left|\frac{\Omega_l(0)}{V_m(0)}\right| . \qquad [s1]$$

As shown in Table 2, at f = 0.0 Hz the maximum load speed measured is $\Omega_l(2\pi) = 3.33$ rad/s and the voltage is $V_m(0) = 2.0$ V. The gain is therefore

$$\left|G_{wl,\,v}(0)\right| = 1.66\left[\frac{rad}{s\ V}\right] . \qquad [s2]$$

Using the expression

$$\left|G_{wl,\,v}(0)\right|_{dB} = 20\ \mathrm{Log10}(1.66) \qquad [s3]$$

the gain in dB is

$$\left|G_{wl,\,v}(0)\right|_{dB} = 4.40\ [dB] . \qquad [s4]$$

This is the steady-state gain of the system

0	1	2

11. Set the *Offset (V)* block to 0 V.
12. Set the *Amplitude (V)* slider gain to 2.0 V.
13. The SRV02 unit should begin rotating smoothly back and forth and the scopes should be reading a response similar to as shown in figures 4 and 5.

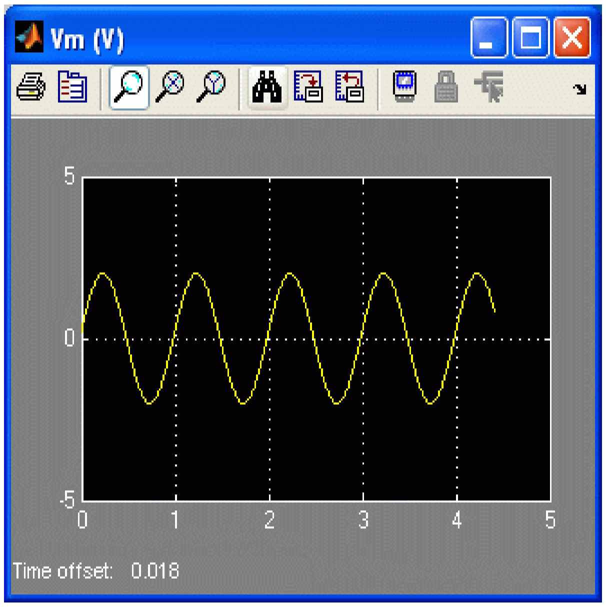

Figure 4: Input motor voltage scope.

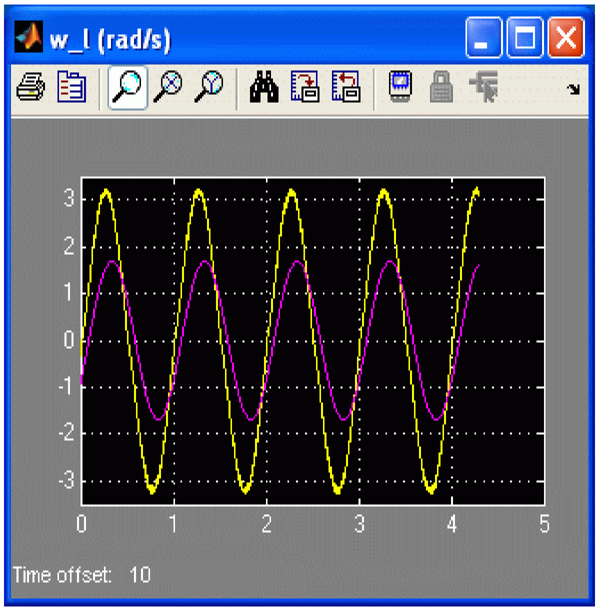

Figure 5: Load shaft speed sine wave response.

14. Measure the maximum positive speed of the load shaft. As before, this measurement can be done directly from the scope or, preferably, users can use Matlab commands to find the maximum load speed using the saved *w_l* variable. Enter the maximum load speed when f = 1.0 Hz in Table 2 above.
15. Calculate the gain of the system (in both linear and dB) and enter the results in Table 2.
16. By adjusting the frequency parameter in the *Signal Generator* block, measure the maximum load speed and calculate the gain for all frequencies listed in Table 2.
17. Generate a magnitude bode plot and attach it to the report. Make sure the amplitude and frequency scales are in decibels. When plotting the bode, ignore the f = 0 Hz entry as the logarithm of 0 is not defined.

Solution:
The sample_freq_rsp.m script contains the collected data in Table 2 and generates the corresponding magnitude bode plot shown in Figure 6.

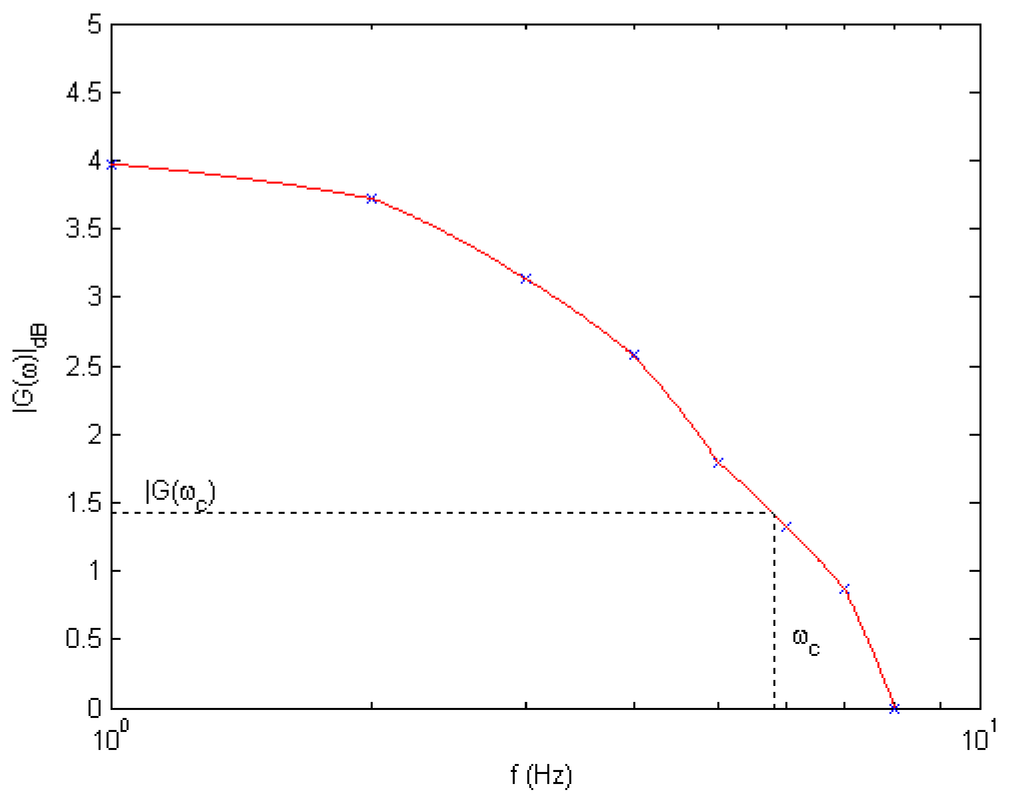

Figure 6: Sample Bode plot after performing frequency response laboratory.

0	1	2

18. Calculate the time constant derived using the frequency response method, $\tau_{e,f}$, using the obtained bode plot by finding the cutoff frequency. Label the bode plot with the -3 dB gain and the cutoff frequency and enter the resulting time constant in Table 3. *Hint*: Use the *ginput* command to obtain values from a Matlab figure.

Solution:
As illustrated in Figure 6, the -3 dB gain is

$$\left| G_{wl,v}(\omega_c) \right|_{dB} = 1.36\ [dB] \qquad [s5]$$

and the corresponding cutoff frequency is

$$f_c = 5.79\ [Hz] \qquad [s6]$$

or

$$\omega_c = 36.4 \left[\frac{rad}{s} \right] \qquad [s7]$$

Given that $\tau_e=1/\omega_c$, the time constant is

$$\tau_{e,f} = 0.0275\ [s] \qquad [s8]$$

0 | 1 | 2

19. Click on the *Stop* button on the Simulink diagram tool bar (or select QUARC | Stop from the menu) to stop running the code.
20. Shut off the power of the amplifier if no more experiments will be performed on the SRV02 in this session.

4.2. Camera Position Control

As in *Rotary Experiment #2: SRV02 Position Control* (Reference [9]), in this section you will design a proportional-velocity controller to control the position of the servo and hence, the camera.

4.2.1. PV Controller

In Section 4.1, the SRV02 voltage-to-speed transfer function was derived. Placing that model in series with an integrator gives the open-loop voltage-to-load gear position transfer function

$$P(s) = \frac{K}{(\tau s + 1) s} . \qquad [2]$$

Using a PV compensator with a second-order plant, such as in [2], results in a closed-loop transfer function with the form

$$\frac{Y(s)}{R(s)} = \frac{\omega_n^2}{s^2 + 2 \zeta \omega_n s + \omega_n^2} , \qquad [3]$$

where ω_n is the natural undamped frequency and ζ is the damping ratio. This is called the *standard second-order* transfer function and the properties of its response depend on the values of ω_n and ζ parameters.

4.2.2. Peak Time and Overshoot Equations

In a second-order system, the amount of overshoot depends solely on the damping ratio parameter and it can be calculated using the equation

$$PO = 100\,\mathbf{e}^{\left(-\frac{\pi\zeta}{\sqrt{1-\zeta^2}}\right)} \quad [4]$$

The peak time depends on both the damping ratio and natural frequency of the system and it can be derived that the relationship between them is

$$t_p = \frac{\pi}{\omega_n\sqrt{1-\zeta^2}} \quad [5]$$

Generally speaking then, the damping ratio affects the shape of the response while the natural frequency affects the speed of the response.

4.2.3. Camera Position Control Specifications

The time-domain specifications for controlling the position of the SRV02 load shaft are:

$$t_p = 0.20\,[s] \quad [6]$$

and

$$PO = 5.0\,[\text{"\%"}] \quad [7]$$

Thus when tracking the load shaft reference, the transient response should have a peak time less than or equal to 0.20 seconds, an overshoot less than or equal to 5 %, and the steady-state response should have no error.

4.2.4. PV Control Design

4.2.4.1. Closed-loop Transfer Function

The proportional-velocity (PV) compensator used to control the position of the SRV02 has the structure

$$V_m(t) = k_p(\theta_d(t) - \theta_l(t)) - k_v\left(\frac{d}{dt}\theta_l(t)\right) \quad [8]$$

where k_p is the proportional control gain, k_v is the velocity control gain, $\theta_d(t)$ is the setpoint or reference load angle, and $\theta_l(t)$ is the measured load shaft angle, and $V_m(t)$ is the SRV02 input voltage. The block diagram of the PV control is illustrated in Figure 7.

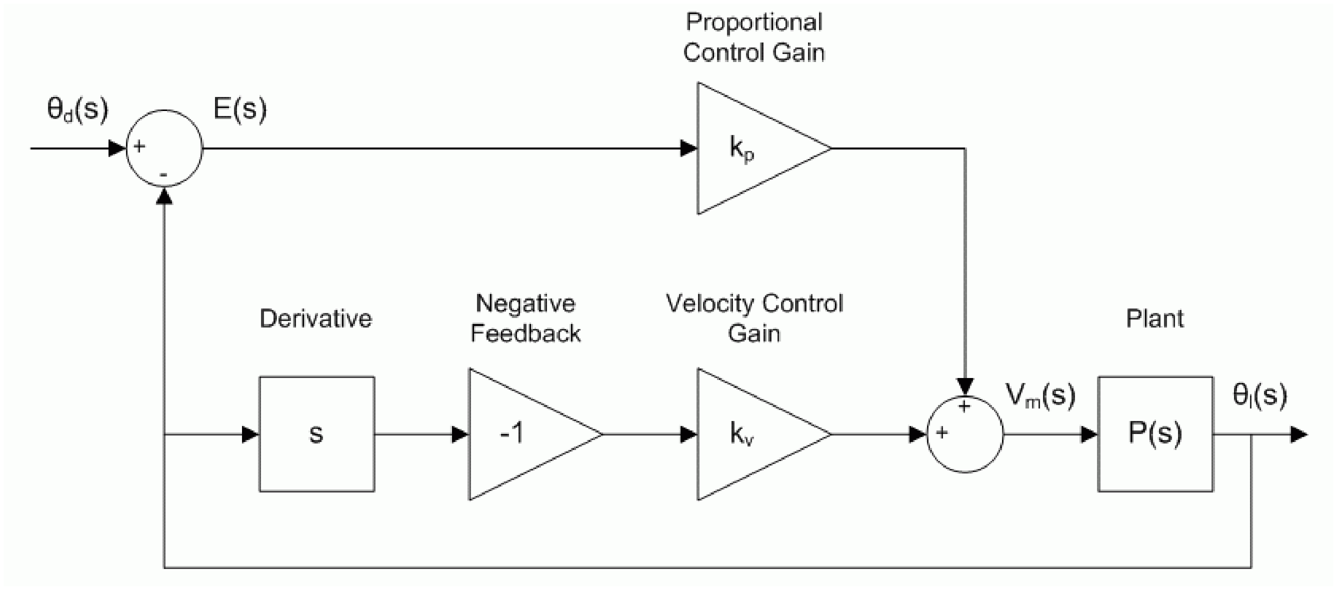

Figure 7: Block diagram of SRV02 PV position control.

1. Find the closed-loop SRV02 position control transfer function, $\Theta_l(s)/\Theta_d(s)$, using the time-domain PV control in Equation [8], the block diagram in Figure 7, and the process model in [2]. Assume the zero initial conditions, thus $\theta_l(0^-) = 0$.

Solution:
The controller transfer function is

$$V_m(s) = k_p(\Theta_d(s) - \Theta_l(s)) - k_v s \Theta_l(s) \qquad [s9]$$

and it can be obtained by taking the Laplace transform of [8] or using the block diagram in Figure 7. Substituting this into the process transfer function in [2] and solving for $\Theta_l(s)/\Theta_d(s)$ gives the SRV02 closed-loop transfer function

$$\frac{\Theta_l(s)}{\Theta_d(s)} = \frac{K k_p}{s^2 \tau + (1 + K k_v) s + K k_p} . \qquad [s10]$$

0 | 1 | 2

4.2.4.2. Finding Gains to Satisfy Specifications

1. The resulting SRV02 closed-loop transfer function has the same structure as the *standard second-order system* given in [3]. The denominator of [3] is called the *standard characteristic equation,*

$$s^2 + 2 \zeta \omega_n s + \omega_n^2 . \qquad [9]$$

Find the control gains k_p and k_v that map the characteristic equation of the SRV02 closed-loop system to the standard characteristic equation given above. With these two equations, the control gains can be designed based on a specified natural frequency, ω_n, and damping ratio, ζ.

Solution:

The characteristic equation of the SRV02 closed-loop transfer function in [s10] is

$$s^2 \tau + s + K k_p + K k_v s \qquad [s11]$$

and can be re-structured into the form

$$s^2 + \frac{(1 + K k_v) s}{\tau} + \frac{K k_p}{\tau} . \qquad [s12]$$

Equating this with the standard characteristic equation in [9] gives the expressions

$$\frac{K k_p}{\tau} = \omega_n^2 \qquad [s13]$$

and

$$\frac{1 + K k_v}{\tau} = 2 \zeta \omega_n . \qquad [s14]$$

Solve for k_p and k_v to obtain the control gains equations that meet the ω_n and ζ specifications. Thus the proportional gain is

$$k_p = \frac{\omega_n^2 \tau}{K} \qquad [s15]$$

and the velocity gain is

$$k_v = \frac{-1 + 2 \zeta \omega_n \tau}{K} . \qquad [s15]$$

0	1	2

2. Using the equations described in Section 4.2.2, express the natural frequency and the damping ratio in terms of percentage overshoot and peak time specifications.

Solution:
To find the damping ratio given a percentage overshoot specification, solve for ζ in Equation [4] and get the expression

$$\zeta = -\ln\left(\frac{PO}{100}\right)\sqrt{\frac{1}{\ln\left(\frac{PO}{100}\right)^2 + \pi^2}} \qquad [s16]$$

Solving for ω_n in Equation [5] expresses the natural frequency in terms of the peak time, that is

$$\omega_n = \frac{\pi}{t_p\sqrt{1-\zeta^2}} \qquad [s17]$$

0 | 1 | 2

3. Calculate the minimum damping ratio and natural frequency required to meet the specifications given in Section 4.2.3.

Solution:
Substitute the percentage overshoot specifications given in [7] into Equation [s16] to get the required damping ratio

$$\zeta = 0.690 \qquad [s18]$$

Using this result and the desired peak time, given in [6], with Equation [s17] gives the minimum natural frequency needed

$$\omega_n = 21.7\left[\frac{rad}{s}\right] \qquad [s19]$$

0 | 1 | 2

4. Based on the nominal SRV02 model parameters, K and τ, found in Experiment #1: SRV02 Modeling (Reference [8]), calculate the control gains needed to satisfy the time-domain response requirements.

Solution:
Using the model parameters

$$K = 1.67 \left[\frac{rad}{s\ V} \right] \qquad [s20]$$

and

$$\tau = 0.0275\ [s] \qquad [s21]$$

as well as the desired natural frequency found in [s19] with Equation [s15], generates the proportional control gain

$$k_p = 7.77 \left[\frac{V}{rad} \right]. \qquad [s22]$$

Similarly, the velocity control gain is obtained by substituting the model parameters given above with the minimum damping ratio specification, in [s18], into Equation [s15]

$$k_v = -0.106 \left[\frac{V\,s}{rad} \right]. \qquad [s23]$$

Thus the position response of the load gear on an SRV02 with a disc load when using the PV controller with the gains above will satisfy the specifications listed in Section 4.2.3.

0	1	2

4.2.5. In-Lab: SRV02 Position Control

The *q_srv02_pos* Simulink diagram shown in Figure 8 is used to perform the position control exercises in this laboratory. The *SRV02-ET* subsystem contains QUARC blocks that interface with the DC motor and sensors of the SRV02 system, as discussed in Reference [7]. The *PV Control* subsystem implements the PV control detailed in Section 4.2.4.2, except a high-pass filter is used to obtain the velocity signal (as opposed to taking the direct derivative).

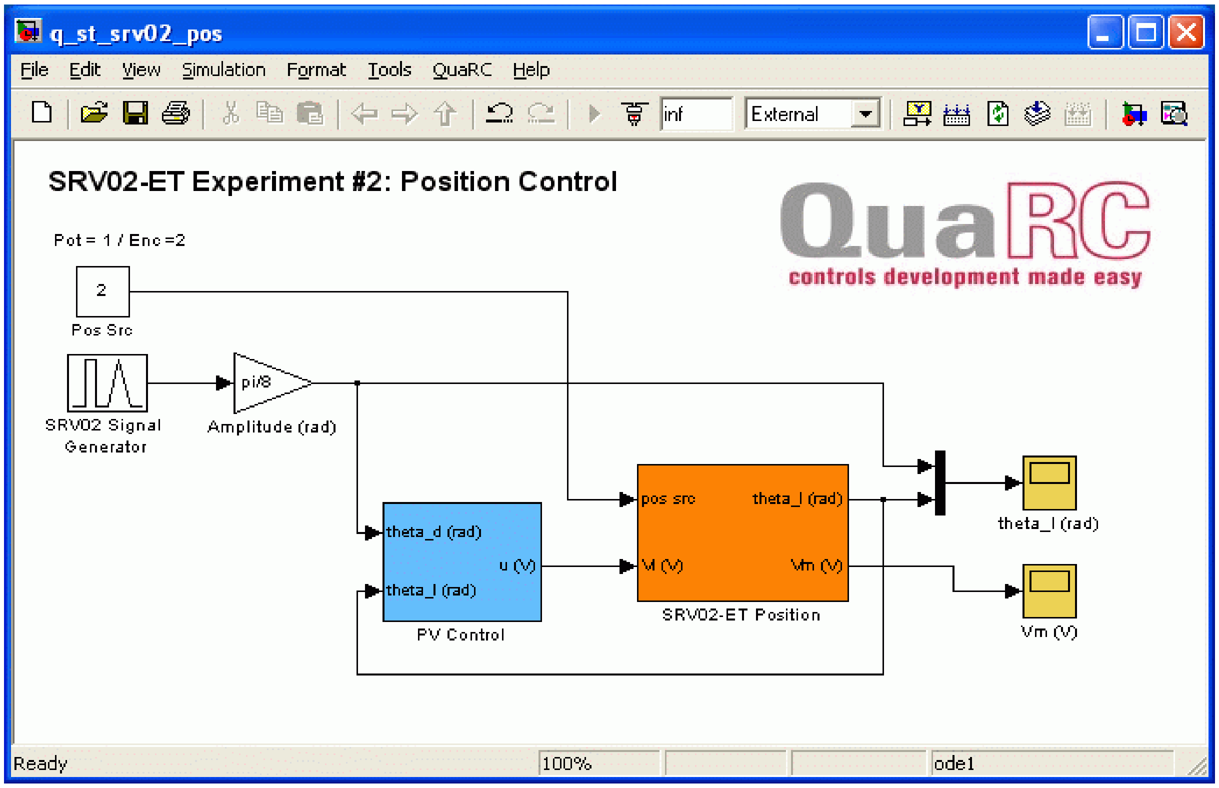

Figure 8: Simulink model used with QUARC to run the PV and PIV position controllers on the SRV02.

4.2.5.1. Setup for Position Control Implementation

Before beginning the in-lab exercises on the SRV02 device, the q_st_srv02_pos Simulink diagram and the setup_st_srv02_pos.m script must be configured.

Follow these steps to get the system ready for this lab:

1. Load the Matlab software.
2. Browse through the *Current Directory* window in Matlab and find the *\Lab Files\02 - SRV02 Position Control* folder that contains *q_st_srv02_pos.mdl*.
3. Double-click on the *q_st_srv02_pos.mdl* file to open the Simulink diagram shown in Figure 8.
4. **Configure DAQ**: Double-click on the HIL Initialize block in the *SRV02-ET* subsystem (which is located inside the *SRV02-ET Position* subsystem) and ensure it is configured for the DAQ device that is installed in your system. See Reference [7] for more information on configuring the HIL Initialize block.
5. **Configure Sensor**: The position of the load shaft can be measured using various sensors. Set the *Pos Src* Source block in q_srv02_pos, as shown in Figure 8, as follows:
   - 1 to use the potentiometer
   - 2 to use to the encoder

   Note that when using the potentiometer, there will be a discontinuity.
6. **Configure setup script**: Set the parameters in the setup_st_srv02_pos_cntrl.m script as follows,

```
%% SRV02 Configuration
% External Gear Configuration: set to 'HIGH' or 'LOW'
EXT_GEAR_CONFIG = 'HIGH';
% Encoder Type: set to 'E' or 'EHR'
ENCODER_TYPE = 'E';
% Is SRV02 equipped with Tachometer? (i.e. option T): set to 'YES' or
'NO'
TACH_OPTION = 'YES';
% Type of Load: set to 'NONE', 'DISC', or 'BAR'
LOAD_TYPE = 'SOLAR_TRACKER';
% Cable Gain used: set to 1, 3, or 5
K_AMP = 1;
% Power Amplifier Type: set to 'Q3', or VoltPAQ
AMP_TYPE = 'VoltPAQ';
% Digital-to-Analog Maximum Voltage (V)
VMAX_DAC = 10;
%
%% Lab Configuration
CONTROL_TYPE = 'MANUAL';
```

Text 1: Setup script parameter settings

7. In the *Calculate Control Gains* section, enter the control you found in Section 4.2.4.2.

```
%% Calculate Control Gains
if strcmp ( CONTROL_TYPE , 'MANUAL' )
    % PV control gains
    kp = 0;
    kv = 0;
```

8. Run the setup_st_srv02_pos.m script.

Instructor:
In the script set:
CONTROL_TYPE = "AUTO_PV"
LOAD_TYPE = 'SOLAR_TRACKER'

These setting will automatically compute the PV gains. However, for best results, it is recommended you use the K and τ parameters that were found in the modeling exercise and enter them as show below:

```
elseif strcmp ( CONTROL_TYPE , 'AUTO_PV' )
    % Load model parameters based on SRV02 configuration.
    % [K,tau] = d_model_param(Rm, kt, km, Kg, eta_g, Beq, Jeq, eta_m,
AMP_TYPE);
    K = 1.67;
    tau = 0.0275;
    % Calculate PV control gains given specifications.
    [ kp, kv ] = d_pv_design( K, tau, PO, tp, AMP_TYPE  );
end
```

IMPORTANT: Make sure you remove the d_model_param.m and d_pv_design.m scripts from the student's folder. Otherwise, they can calculate the PV gains automatically.

0	1	2

4.2.5.2. Implementing Step Response using PV

In this lab, the angular position of the SRV02 load shaft, i.e. the disc load, will be controlled using the developed PV control. Measurements will then be taken to ensure that the specifications are satisfied.

Follow the steps below:

1. Run the setup_srv02_exp02_pos.m script.
2. Enter the proportional and velocity control gains found in Section 4.2.4.2.
3. Set *Signal Type* in the SRV02 Signal Generator to *square* to generate a step reference.
4. Set the *Amplitude (rad)* gain block to $\pi/8$ to generate a step with an amplitude of 45 degrees.
5. Open the load shaft position scope, *theta_l (rad)*, and the motor input voltage scope, *Vm (V)*.
6. Click on QUARC | Build to compile the Simulink diagram.
7. Select QUARC | Start to begin running the controller. The scopes should be displaying responses similar to figures Figure 9 and Figure 10. Note that in the *theta_l (rad)* scope, the yellow trace is the setpoint position while the purple trace is the measured position.

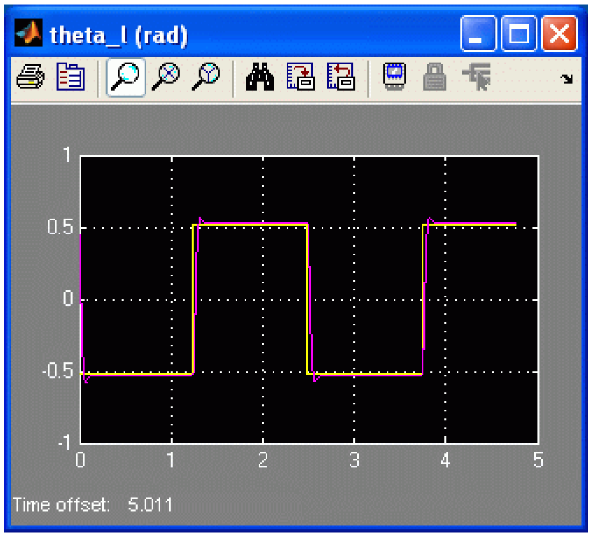

Figure 9: Measured PV step response.

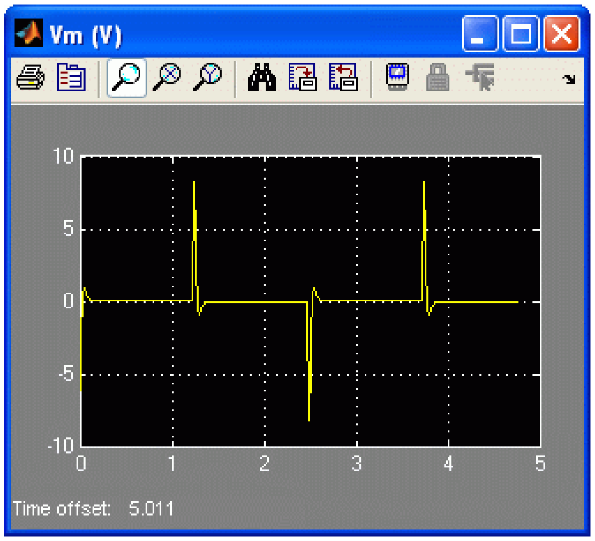

Figure 10: PV control input voltage.

8. When a suitable response is obtained, click on the *Stop* button in the Simulink diagram tool bar (or select QUARC | Stop from the menu) to stop running the code.
9. Generate a Matlab figure showing the PV position response and its input voltage. Attach it to your report. When the controller is stopped each scope automatically saves their response to a variable in the Matlab workspace. The *theta_l (rad)* scope saves its response to the *data_pos variable* and the *Vm (V)* scope saves its data to the *data_vm* variable.

Solution:
The measured SRV02 closed-loop position step response when using the PV control is shown in Figure 11. To generate this response, execute the *sample_meas_tp_os.m* script with the saved MAT files *data_step_rsp_theta.mat* and *data_step_rsp_Vm.mat*. Alternatively, to generate a Matlab figure from a new experimental run do the following:

1. Run the *q_st_srv02_pos* Simulink model until a response fills the scopes.
2. Stop QUARC.
3. Execute the *sample_meas_tp_os.m* script.

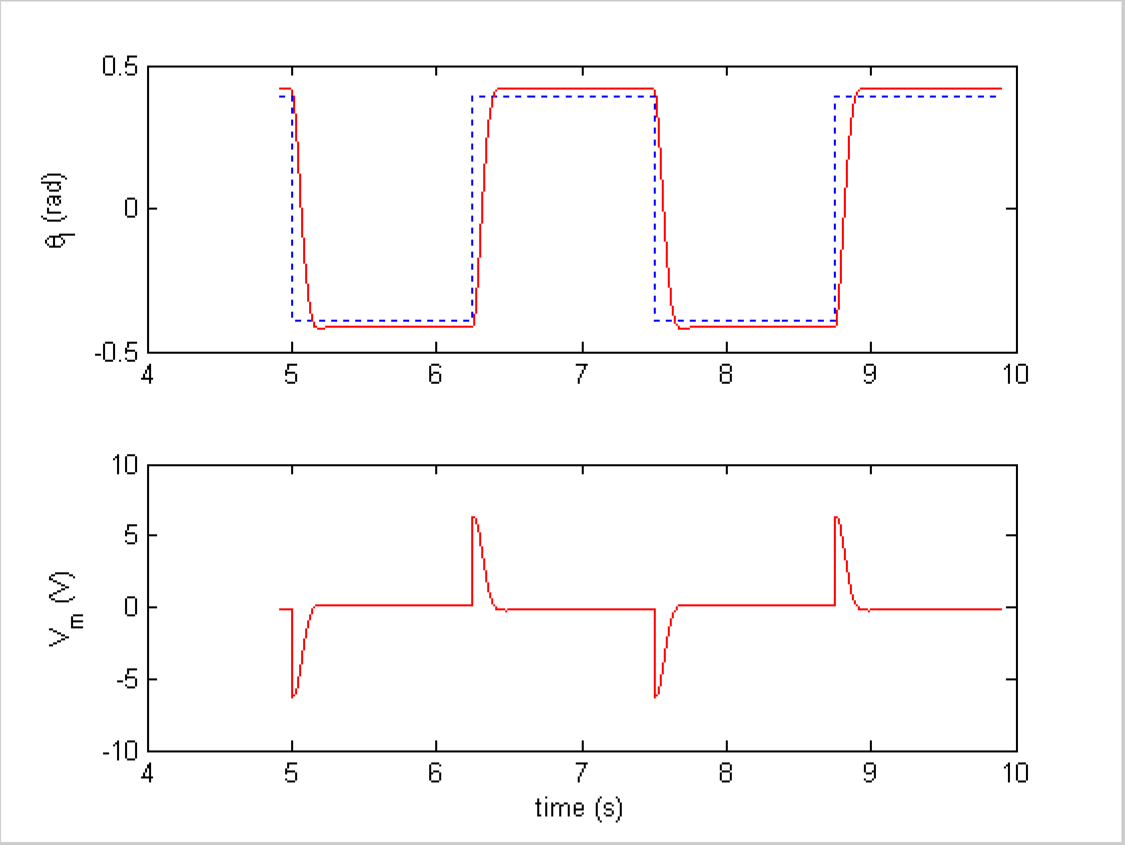

Figure 11: Measured SRV02 step response using PV.

0	1	2

10. Measure the percentage overshoot and the peak time of the SRV02 load gear. Does the response satisfy the specifications given in Section 4.2.3?

Solution:
The peak time and percentage overshoot of the response shown in Figure 11 is

$$t_p = 0.154 \ [s] \qquad [s24]$$

and

$$PO = 1.56 \ [\text{"\%"}] \qquad [s25]$$

The actual measured SRV02 response satisfies the specifications given in Section 4.2.3. To find the peak time and percentage overshoot of a response saved in *data_pos* automatically, run the *sample_meas_tp_os.m* script after running *q_st_srv02_pos* or using the responses saved in the MAT files *data_step_rsp_theta.mat* and *data_step_rsp_Vm.mat*.

0	1	2

11. Stop QUARC.
12. Enter the PV gains used and the resulting peak time and percentage overshoot measured in Table 3.
13. Shut off the amplifier if no more experiments will be performed on the SRV02 in this session.

4.3. Sensor Calibration

In this section, the goal is to find a sensor gain that can be used to measure the angle between the camera and the light source using the differential light intensity reading.

4.3.1. Camera Sensor Gain

The typical measured camera response when the servo swivels +/- 90 deg with respect to the light source is shown in Figure 12.

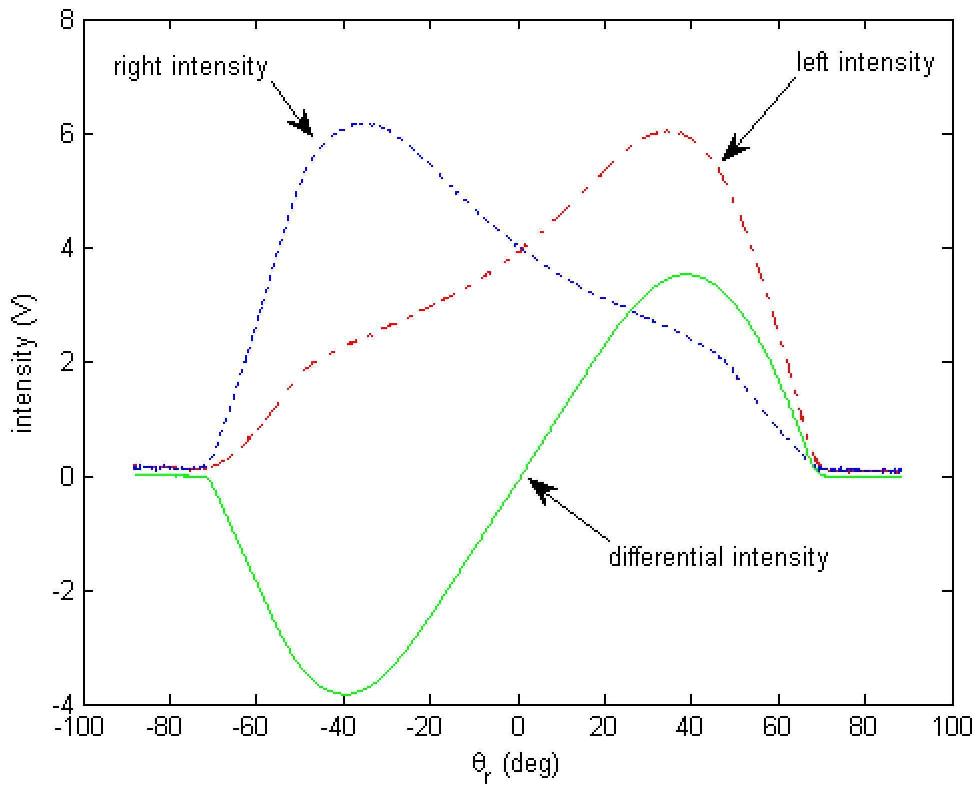

Figure 12: Typical camera response when servo rotating +/- 90 deg with respect to light source.

The differential intensity, A_d, is the difference between the right intensity, A_r, and left intensity, A_l:

$$A_d = A_r - A_l \quad [10]$$

The relative angle between the camera and the light source, shown in Figure 13, can be measured by the camera using the relationship

$$\theta_r = K_s A_d \quad [11]$$

where K_s is the sensor gain and A_d is the differential intensity described in Equation [10]. The sensor gain is the inverse slope of the linear range of the A_d curve shown in Figure 12.

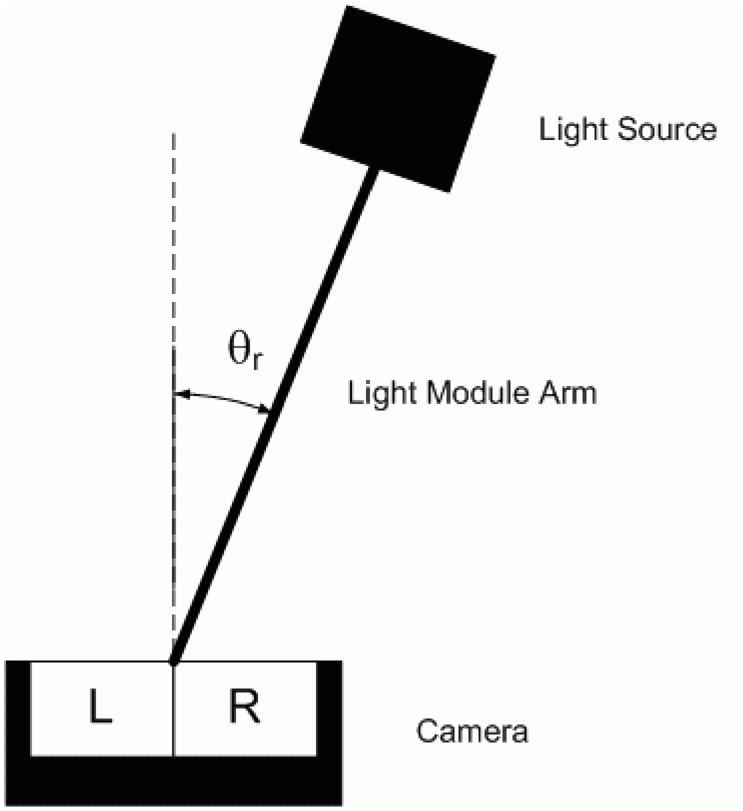

Figure 13: Angle between camera and light as light rotates.

In order to find K_s we can fix the light at 0 degrees and swivel the servo by a known angle (using the servo position controller), as illustrated in Figure 14. Given the measured servo angle θ_l and the differential intensity A_d, the sensor gain can be found using

$$K_s = \frac{\theta_l}{A_d} \quad [12]$$

.

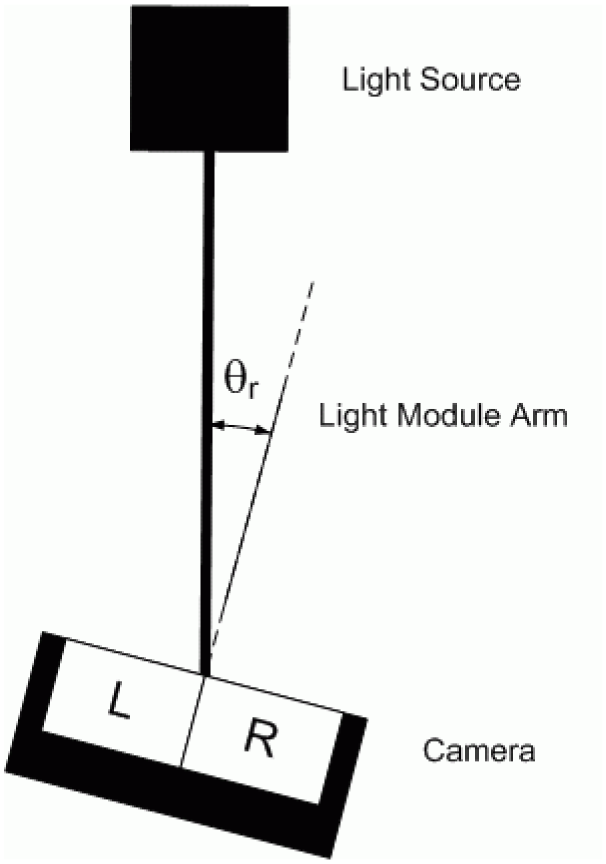

Figure 14: Relative angle between light and servo when servo rotates

4.3.2. In-Lab: Finding Sensor Gain

The sensor gain of the Solar Tracker camera will be found with QUARC using the Simulink diagram shown in Figure 15.

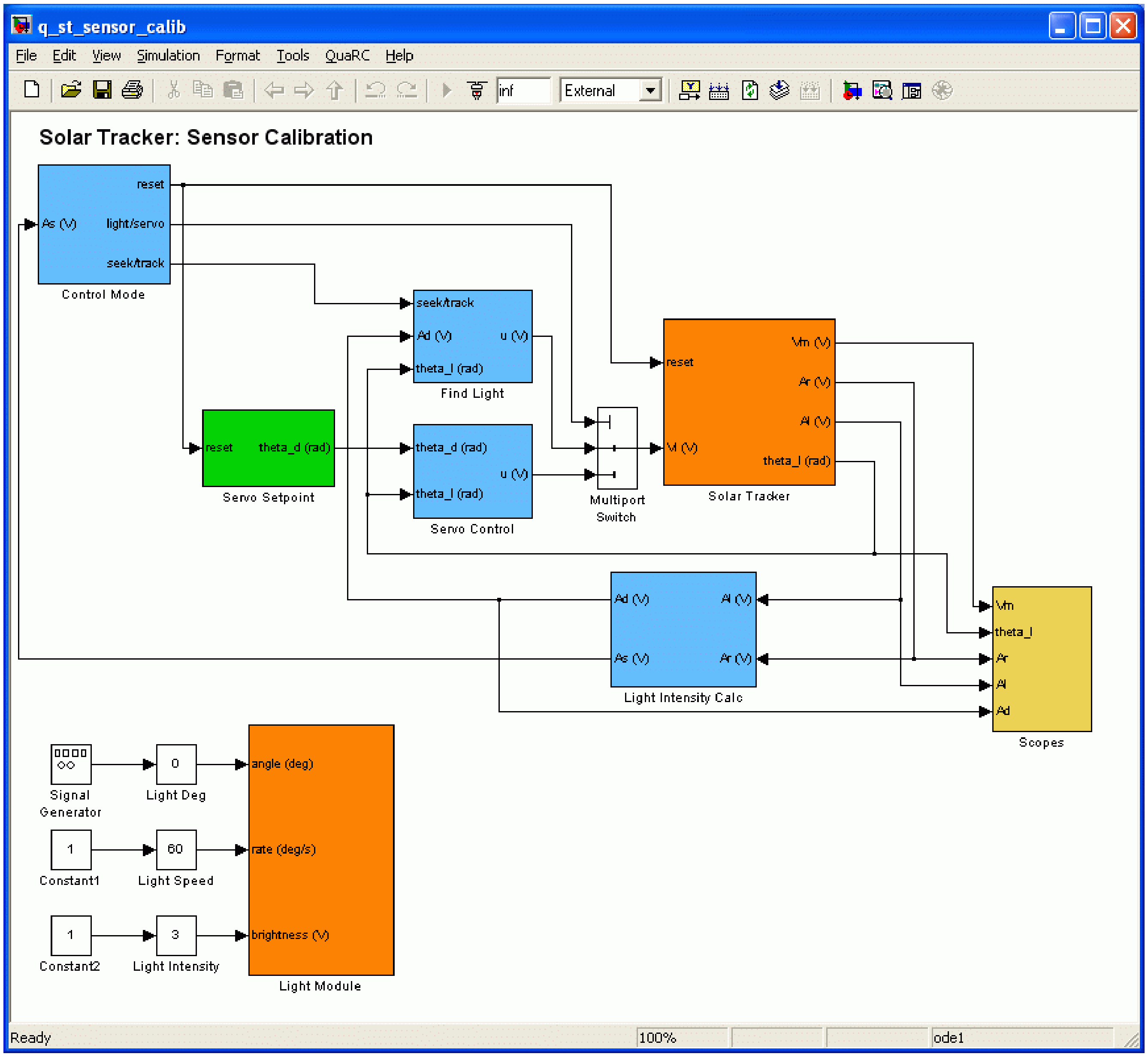

Figure 15: Simulink diagram used with QUARC to perform the Solar Tracker sensor calibration.

4.3.2.1. Setup Experiment Files

Similarly as in in Section 4.2.5.1, follow these steps to get the system ready for this lab:

1. In Matlab, open the *q_st_sensor_calib.mdl* file in the *\Lab Files\03 - Sensor Calibration* folder.
2. **Configure DAQ**: Double-click on the HIL Initialize block in the *Solar Tracker* subsystem and ensure it is configured for the DAQ device that is installed in your system. See Reference [7] for more information on configuring the HIL Initialize block.
3. **Configure setup script**: Set the parameters in the *setup_st_sensor_calib.m* script as shown in Text 1.

4. In the *Calculate Control Gains* section, enter the control you found in Section 4.2.4.2.

```
%% Calculate Control Gains
if strcmp ( CONTROL_TYPE , 'MANUAL' )
    % PV control gains
    kp = 0;
    kv = 0;
    % Sensor gain (deg/V)
    Ks = 0;
```

5. Run the setup_st_sensor_calib.m script.

Instructor:
In the script set:
CONTROL_TYPE = "AUTO_PV"
LOAD_TYPE = 'SOLAR_TRACKER'

These setting will automatically compute the PV gains. However, for best results, it is recommended you use the K and τ parameters that were found in the modeling exercise and enter them as show below:

```
elseif strcmp ( CONTROL_TYPE , 'AUTO_PV' )
    % Load model parameters based on SRV02 configuration.
    % [K,tau] = d_model_param(Rm, kt, km, Kg, eta_g, Beq, Jeq, eta_m,
AMP_TYPE);
    K = 1.67;
    tau = 0.0275;
    % Calculate PV control gains given specifications.
    [ kp, kv ] = d_pv_design( K, tau, PO, tp, AMP_TYPE  );
    % Sensor gain (deg/V)
    Ks = (40+40) / (3.5+4);
end
```

IMPORTANT: Make sure you change the value for Ks entered above to Ks = 0. Also, remove the d_model_param.m and d_pv_design.m scripts from the student's folder if you don't want the student to calculate the PV gains automatically.

0	1	2

4.3.2.2. Running the QUARC Controller

In this lab, the angular position of the SRV02 load shaft, i.e. the disc load, will be controlled using the Follow the steps below:

1. Power on the Solar Tracker system.
2. Wait for the light module to be centered.
3. Power on the amplifier (which drives the SRV02).
4. Run the setup_st_sensor_calib.m script.
5. Open the load shaft position scope, *theta_l (rad)*, the measured light intensity scopes, *Intensity (V)*, and the *Intensity vs. Servo Response* X-Y Figure. These are all located in the *Scopes*

subsystem.

6. Click on QUARC | Build to compile Simulink diagram.
7. Select QUARC | Start to begin running the controller. The camera will move around to find the light source, align itself to the light, and then begin to swivel +/- 90 deg. When it's in this *calibration* mode, the scopes should have a responses similarly as shown in Figure 16 and Figure 17. In the *Intensity (V)* scope, the right intensity is yellow, the left intensity is purple, and the differential intensity is cyan.

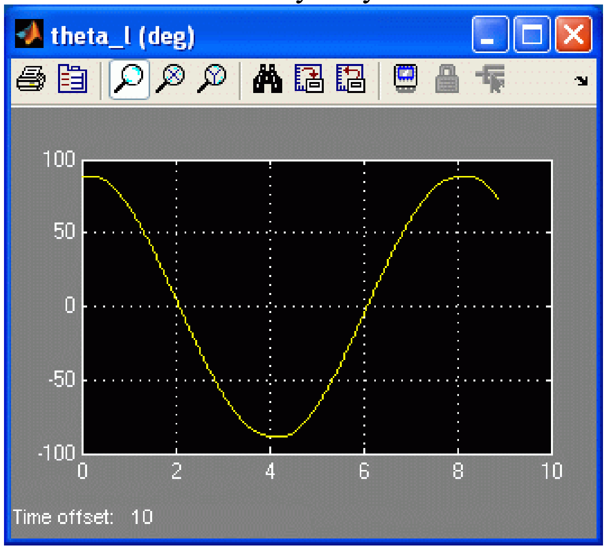

Figure 16: Servo angle response in calibration mode

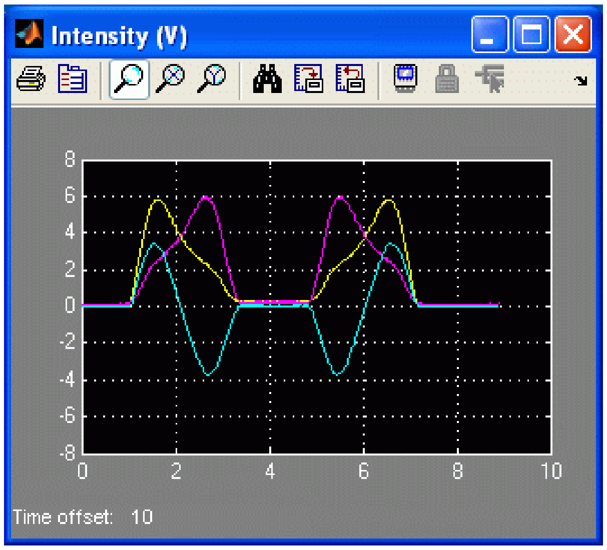

Figure 17: Light intensity response in calibration mode

8. The X-Y Figure should have a response similarly as shown in Figure 18.

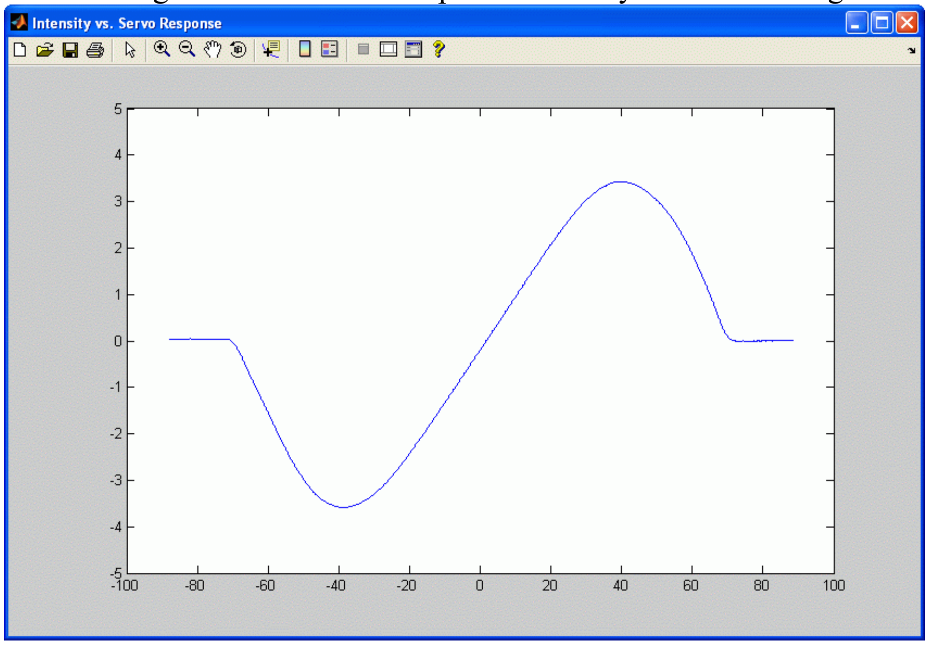

Figure 18: Differential intensity versus camera/servo angle response

9. When you have a suitable response in the X-Y Figure, stop the QUARC controller.

10. Using the X-Y Figure, find the sensor gain K_s. If desired, you can use the Data Cursor tool to obtain a more accurate reading. You can also use the saved data from the *theta_l (deg)* and *Intensity (V)* scopes, i.e. *data_theta_l* and *data_intensity*.

Solution:
As detailed in plot_response.m, using the *ginput* command it was found that the min and max differential intensity is -4.0 V and 3.6 V, and the servo ranges from -42.6 to 38.5 degrees. The sensor gain for the response in Figure 18 is therefore

$$K_s = 10.7 \left[\frac{deg}{V} \right] . \qquad [s26]$$

0	1	2

11. Enter sensor gain value you just found in Matlab as variable *Ks*.
12. Start the QUARC controller.
13. Open the *theta_r (deg)* scope.
14. This scope displays the angle between the camera and the light measured by the SRV02 encoder (yellow) and by the the camera (purple). Do the responses match (at least, in a certain region)? If not, then you need to recalculate K_s. Otherwise, attach a plot of the response to your report.

Solution:

The response should be similarly as shown in Figure 19. The blue dash-dot line is the servo angle, θ_l, and the solid red line is the camera angle, θ_r.

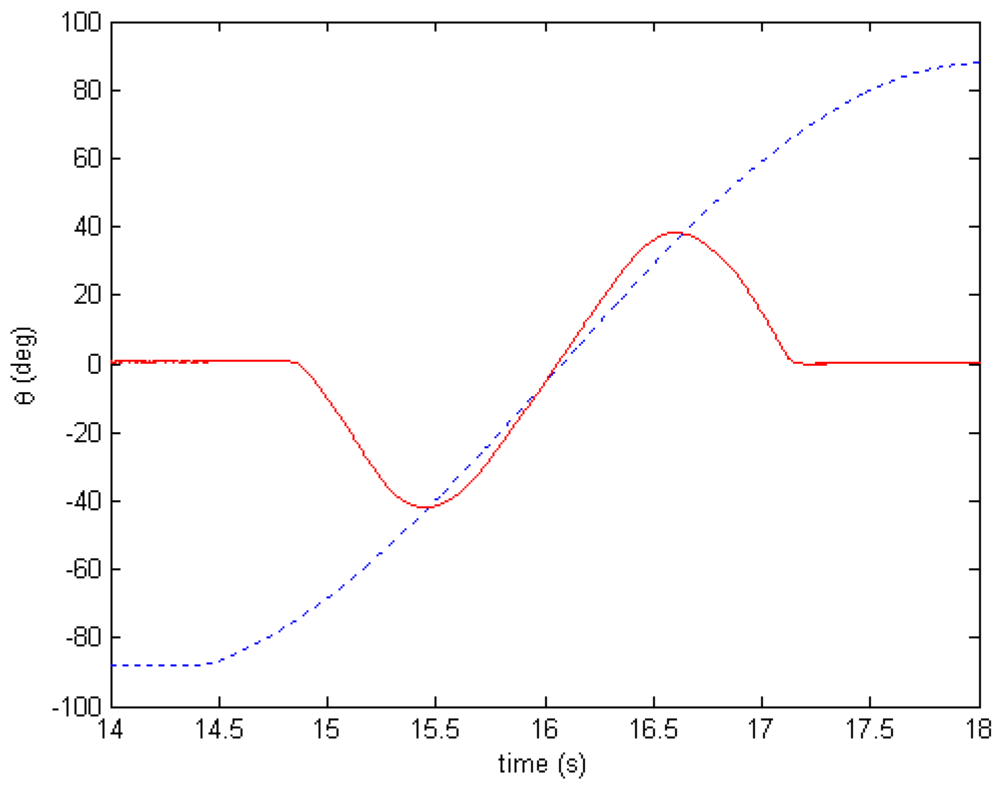

Figure 19: Angle between servo and light measured by SRV02 encoder and camera.

0	1	2

15. When using the camera, can you measure the relative angle between the camera and the light source for the entire 180 degree range? Explain why or why not.

Solution:
The linear range of the camera is approximately -40 to 40 degrees. Beyond this range, the camera response is very different and it cannot be used to measure the relative angle θ_r.

0	1	2

16. Stop QUARC.
17. If this is your last lab session with this device, turn off the amplifier.

4.4. Solar Tracker

In this section, the controller that makes the camera track the light module will be implemented.

4.4.1. PIV Solar Tracker Control

The block diagram in Figure 20 shows all the components involved in the solar tracker control. The commands to the light module controls the position (V_{pos}), rate (V_{spd}), and brightness (V_{light}) of the light. This changes the angle between the camera and the light, i.e. it changes the setpoint θ_d. Using the camera measurements with the identified sensor gain, K_s, we can measure the angle between the camera and the light, θ_r, and use a control to move the servo towards the light. The *PV Controller*, $C_{pv}(s)$, is the PV servo control developed in Section 4.2. Integral control can also be added to improve the response.

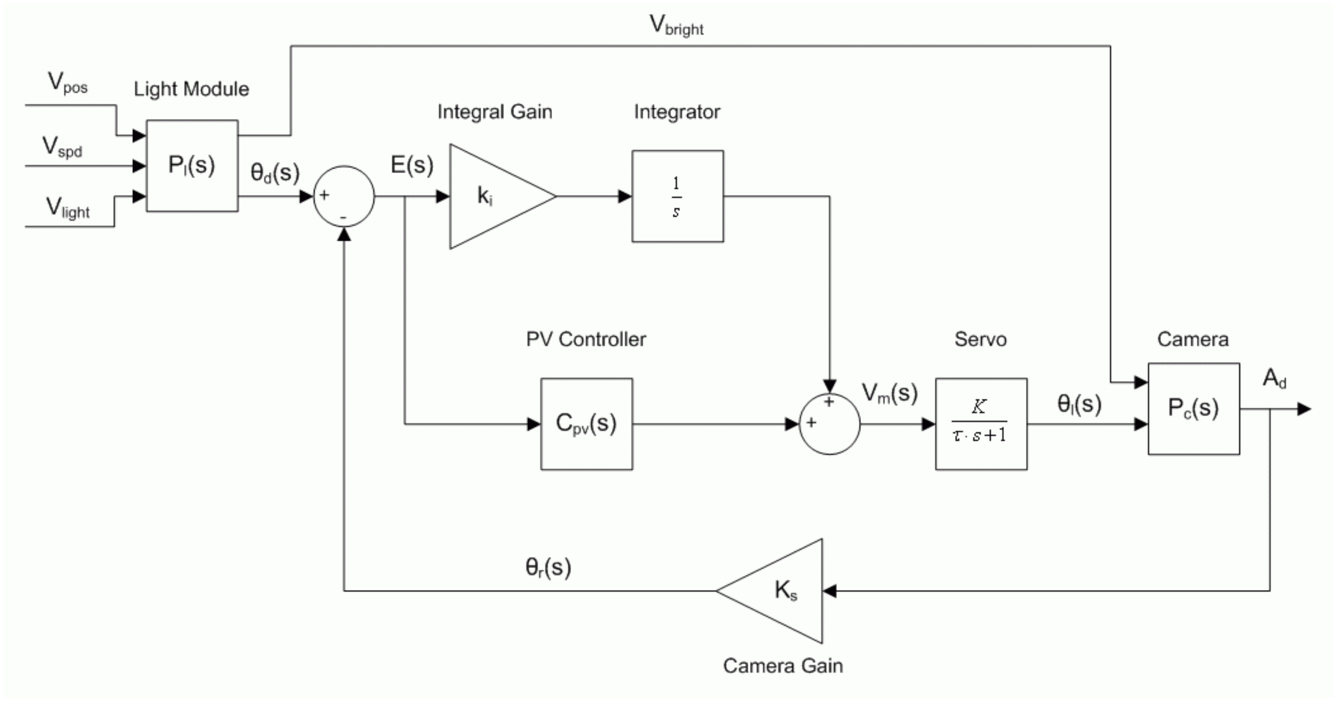

Figure 20: Solar Tracker control

4.4.2. In-Lab: Running the Solar Tracker Control

The solar tracker control is implemented using the Simulink diagram shown in Figure 21 with QUARC.

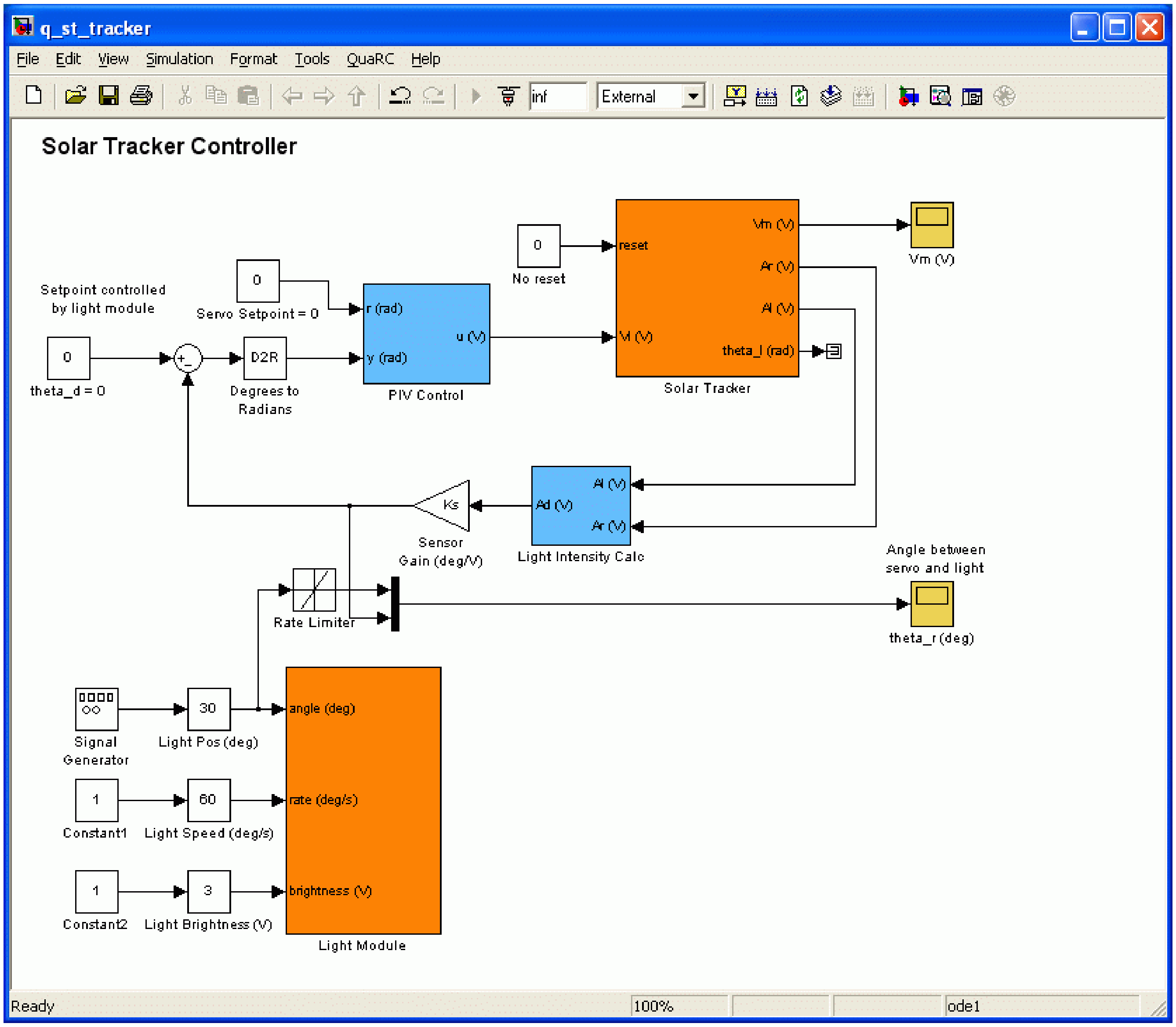

Figure 21: Simulink diagram used to implement Solar Tracker controller in QUARC.

4.4.2.1. Setup Experiment Files

Similarly as in in Section 4.2.5.1, follow these steps to get the system ready for this lab:

1. In Matlab, open the *q_st_tracker.mdl* file in the *\Lab Files\04 – Solar Tracking* folder.
2. **Configure DAQ**: Double-click on the HIL Initialize block in the *Solar Tracker* subsystem and ensure it is configured for the DAQ device that is installed in your system. See Reference [7] for more information on configuring the HIL Initialize block.
3. **Configure setup script**: Set the parameters in the *setup_st_tracker.m* script as shown in Text 1.
4. In the *Calculate Control Gains* section, enter the control you found in Section 4.2.4.2.

```
%% Calculate Control Gains
if strcmp ( CONTROL_TYPE , 'MANUAL' )
    % PV control gains
    kp = 0;
    kv = 0;
    % Sensor gain (deg/V)
    Ks = 0;
```

5. Run the setup_st_tracker.m script.

Instructor:
In the script set:
CONTROL_TYPE = “AUTO_PV”
LOAD_TYPE = 'SOLAR_TRACKER'

These setting will automatically compute the PV gains. However, for best results, it is recommended you use the K and τ parameters that were found in the modeling exercise and enter them as show below:

```
elseif strcmp ( CONTROL_TYPE , 'AUTO_PV' )
    % Load model parameters based on SRV02 configuration.
    % [K,tau] = d_model_param(Rm, kt, km, Kg, eta_g, Beq, Jeq, eta_m, 
AMP_TYPE);
    K = 1.67;
    tau = 0.0275;
    % Calculate PV control gains given specifications.
    [ kp, kv ] = d_pv_design( K, tau, PO, tp, AMP_TYPE  );
    % Sensor gain (deg/V)
    Ks = (40+40) / (3.5+4);
end
```

IMPORTANT: If you want the student's to find the sensor gain Ks by themselves, make sure you set Ks = 0 in the script shown above. Also, remove the d_model_param.m and d_pv_design.m scripts from the student's folder if you don't want the student to calculate the PV gains automatically.

0	1	2

4.4.2.2. PV Solar Tracker Control

In this lab, the angular position of the SRV02 load shaft, i.e. the disc load, will be controlled using the Follow the steps below:

1. Power on the Solar Tracker system.
2. Wait for the light module to be centered.
3. Power on the amplifier (which drives the SRV02).
4. Run the setup_st_tracker.m script.
5. Open the *theta_r (deg)* scope, which shoes the angle between the camera and the light as measured by the photo diodes.

6. Set the *Light Pos (deg)* slider gain to 0.
7. Click on QUARC | Build to compile Simulink diagram.
8. Select QUARC | Start to begin running the controller.
9. Set the *Light Pos (deg)* to 30 deg, the *Light Speed (deg/s)* to 60, and the *Light Brightness (V)* to 3. The light module should not be rotating +/- 30 deg and the camera should be tracking it. The *theta_r (deg)* scope response should be similarly as shown in Figure 22.

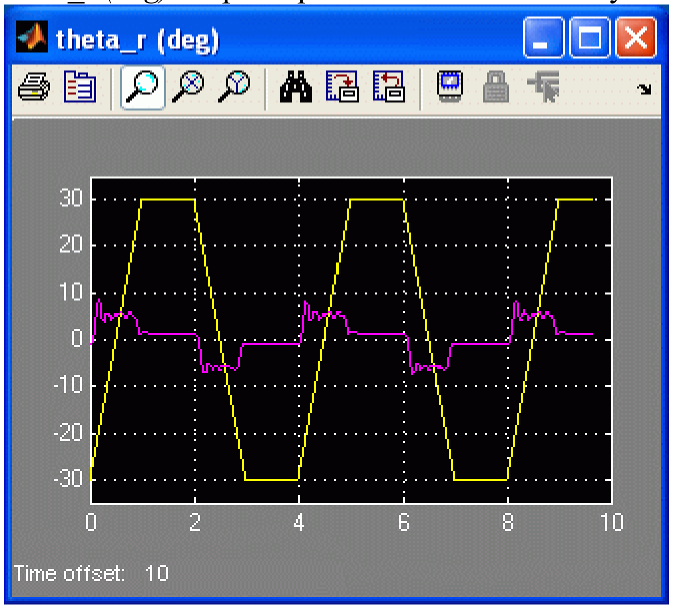

Figure 22: Typical response when tracking light going +/- 30 deg.

Camera Not Tracking?: If the camera is not tracking the light, make sure the light is in the camera's range, i.e. the camera is not pointing the opposite direction of the light source. You can stop the controller, rotate the camera to point to the light, and start the QUARC controller again.

Rate Limiter Note: The rate limiter is used to mimic the speed of the light module. This way, a more realistic response is shown in the *theta_r (deg)* scope. However, if the speed of the light module is changed, then make sure the Rate Limiter block parameters are changed as well, e.g. if *Light Speed (deg)* is set to 45 deg, then set the rising and falling slew rate to 45 in the Rate Limiter block.

10. When you have a suitable response in the scope, stop the QUARC controller.
11. Attach a plot of the response. You can use the saved data called *data_theta_r* from the *theta_r (deg)* scope.

Solution:
A typical response of the light module position (blue dotted line) and the relative angle measured by the camera (solid red line) is shown in Figure 23. Use the plot_theta_r.m script with the saved *sample_data_theta_r_pv.mat* file to generate this plot.

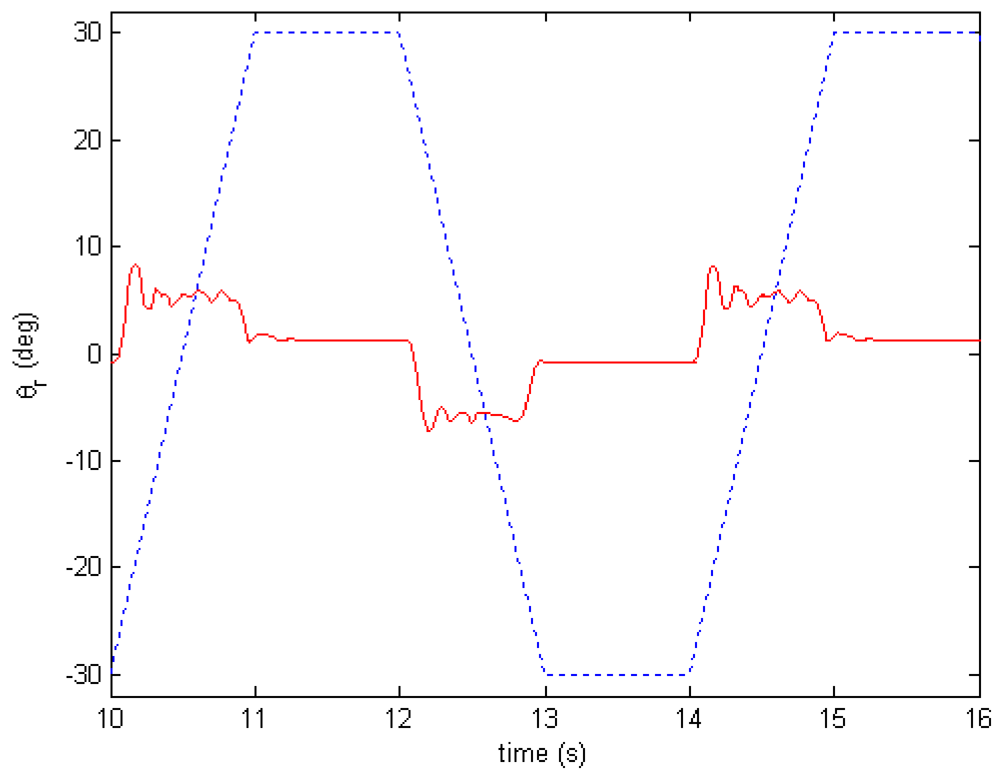

Figure 23: PV response when tracking light going +/- 30 deg.

0	1	2

12. Measure the peak time and steady-state error. To do this, measure the time of the first peak when the light starts moving. For the steady-state error, since the θ_r is essentially the error between the light and the camera, you only have to measure the steady-state angle of θ_r. Make sure the angle is measured when the light source is moving.

Solution:
This data was measured at the 10 sec mark shown in Figure 23. The first peak starts at 10.18 sec, so the peak time is

$$t_p = 0.18\ [s] \qquad [s27]$$

The steady-state error is

$$\theta_{r,\,ss} = 4.9\ [deg] \qquad [s27]$$

This steady-state error is the angle measured at around 10.9 sec, which is a bit before the light stops moving.

0	1	2

4.4.2.3. PIV Solar Tracker Control

We will now add integral control and examine its effect on the response.

1. Start the QUARC controller. The camera should be tracking the light as it moves +/- 30 deg at 60 deg/s.
2. Go into the *PIV Control* subsystem and change the *Integral Gain (V/rad/s)* slider gain to some value between 0 and 50.
3. How does adding integral control effect the tracking response (i.e. how does it change the θ_r response)?

Solution:
The tracking is improved, i.e. the steady-state error is minimized as it eventually converges to 0 deg.

0	1	2

4. When you have found a suitable integral gain, attach a plot of the Proportional-Integral-Velocity (PIV) response.

Solution:
A typical response of the light module position (blue dotted line) and the relative angle measured by the camera (solid red line) is shown in Figure 24 when using an integral gain of 25. Use the plot_theta_r.m script with the saved *sample_data_theta_r_piv.mat* file to generate this plot.

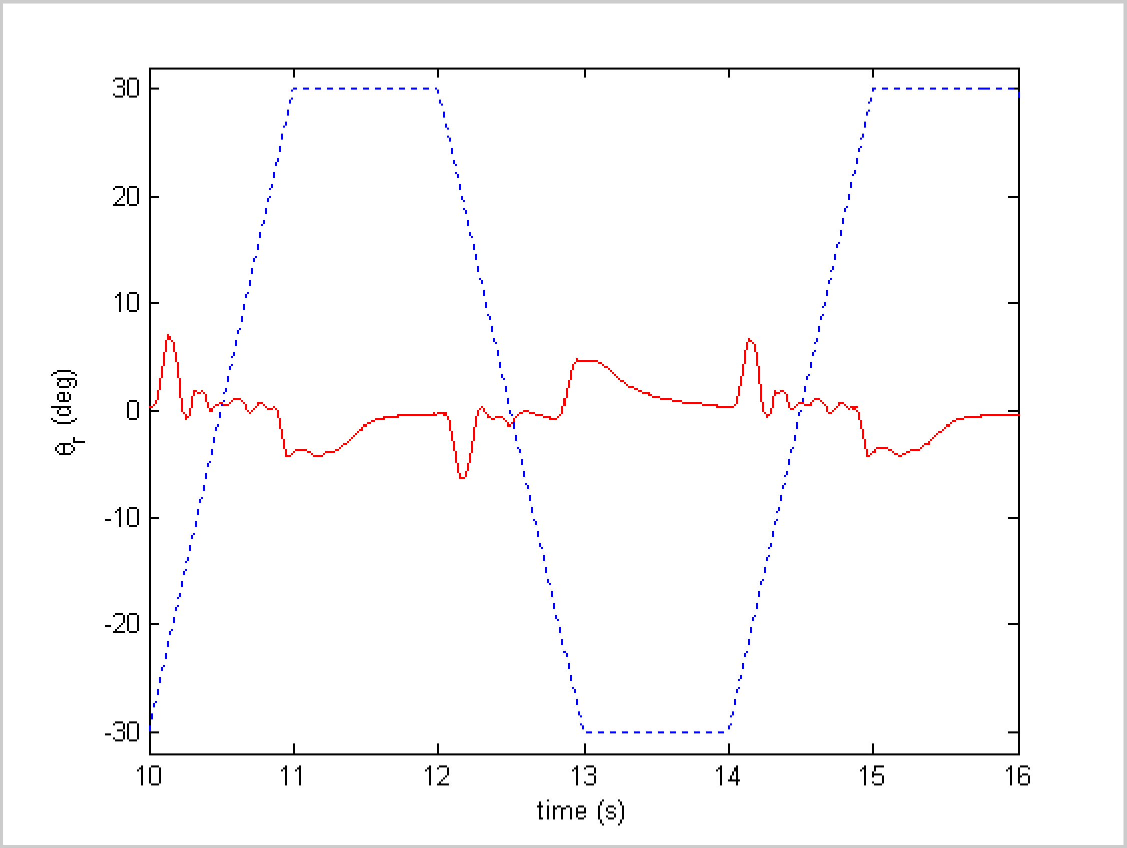

Figure 24: PIV response when tracking light going +/- 30 deg.

0	1	2

5. What is the resulting peak time and steady-state error? How does the peak time and steady-state error compare between using the PV and PIV controls?

Solution:
This data was measured at the 10 sec mark shown in Figure 24. The first peak occurs at 10.14 sec, so the peak time is

$$t_p = 0.14\ [s] \quad \text{[s27]}$$

The steady-state error is approximately

$$\theta_{r,\,ss} = 0.5\ [deg] \quad \text{[s27]}$$

The PIV control improved both the peak time and the steady-state error. Using the PIV control, when the light module is in motion the camera "catches up" to the light and the angle between them goes near 0 deg. In the PV control, the camera was always lagging by a constant angle.

0	1	2

6. Stop QUARC.
7. If this is your last lab session with this device, turn off the amplifier.

5. Results Summary

Fill out Table 3, below, with the pre-lab and in-lab results obtained above. For instance, enter the frequency response model parameters that were found in Section 4.1.

Section	*Description*	*Symbol*	*Value*	*Unit*
4.1	***In-Lab: Frequency Response Modeling***			
	Open-Loop Steady-State Gain	K	1.67	rad/(V.s)
	Open-Loop Time Constant	τ	0.0272	s
4.2.4.2	***Pre-Lab: PV Gain Design***			
	Proportional gain	k_p	7.76	V/rad
	Velocity gain	k_v	-0.106	V.s/rad
4.2.5.2	***In-Lab: PV Step Response***			
	Peak time	t_p	0.154	s
	Percentage overshoot	PO	1.56	%
4.3.2.2	***In-Lab: Sensor Gain***			

	Sensor gain	K_s	10.7	deg/V
4.4.2.2	***In-Lab: Solar Tracking PV Control***			
	Peak time	$t_{p,pv}$	0.18	s
	Steady-state error	$\theta_{r,ss,pv}$	4.9	deg
4.4.2.3	***In-Lab: Solar Tracking PIV Control***			
	Integral gain	k_i	25	V/(rad.s)
	Peak time	$t_{p,piv}$	0.14	s
	Steady-state error	$\theta_{r,ss,piv}$	0.5	deg

Table 3: Solar Tracker results summary

6. References

[1] Quanser. *DAQ User Manual*.
[2] Quanser. *QUARC User Manual* (type `doc quarc` in Matlab to access).
[3] Quanser. *QUARC Installation Manual*.
[4] Quanser. *Amplifier User Manual*.
[5] Quanser. *SRV02 User Manual*.
[6] Quanser. *Solar Tracker User Manual*.
[7] Quanser. *SRV02 QUARC Integration – Instructor Manual*.
[8] Quanser. *Rotary Experiment #1: Modeling*.
[9] Quanser. *Rotary Experiment #2: Position Control*.