Mini-Project 2 Code

```
% Clean up to set up for new run
clc
clear
```

Setup control system + Servo setup

```
% Create arduino and arduino position servo ref-objects
    [robotArduino, tiltServo, panServo, blinkLED, rawRangeIn] = SETUPARDUINO('COM9');
% Turn on board LED on and off to signal program has started
    Blink(robotArduino,blinkLED,5);
    disp('Warning! Data Collection Active!')
```

```
Warning! Data Collection Active!
```

```
    disp('Warning! Pan and Tilt Servos Active!')
```

```
Warning! Pan and Tilt Servos Active!
```

```

% Configure test loop to collect n data points
    nDegrees = input(['Enter the number of stops during robot movement, followed by enter: ']);
    nTests = input(["Enter the number of loops to complete for testing, followed by enter: "]);

    clc;        % clear command window

    r = rateControl(0.25);      % create a 0.25 hz loop rate
    reset(r);
```

Run Robot Control Loop (code that runs over and over)

```
controlFlag = 1;                        % create a loop control

% scale desired angle to 0-1 for writeposition Arduino command
% 0 = 0 deg on servo, 1 = max angle (180 deg for servos)
pan = linspace(0, 0.75, nDegrees);
tilt = linspace(0, 1, nDegrees);

while(controlFlag  < nTests + 1)    % loop till ntests data captured
    % run the act function to collect data
    sensorData = ACTRCServoArduino(panServo, tiltServo, tilt, pan, robotArduino, rawRangeIn);
    Blink(robotArduino, blinkLED, 1); % Check the loop is running
    waitfor(r);                             % wait for loop cycle to complete
    controlFlag = controlFlag + 1;          % increment loop
end
```

Robot Functions (store local functions here)

Clean shut down

finally, with most embedded robot controllers, its good practice to put all actuators into a safe position and then release all control objects and shut down all communication paths. This keeps systems from jamming when you want to run again.

```
% Stop program and clean up the connection to Arduino
% when no longer needed
writePosition(panServo, 0.5);           % always end servo at 0.5
writePosition(tiltServo, 0.5);           % always end servo at 0.5
clc
disp('Arduino program has ended')
```

```
Arduino program has ended
```

```
clear robotArduino
beep
```

```
function [robotArduino, panServo, tiltServo, blinkLED, rawRangeIn] = SETUPARDUINO(COMPORT)
% SETUPARDUINO creates and configures an arduino to be a simple robot
% controller. It requires which COM port your Arduino is attached to as an
% input and returns an Arduino object called robotArduino

% create a glocal arduino object so that it can be used in functions

robotArduino = arduino(COMPORT, 'Uno', 'Libraries', 'Servo');

% configure pin 13 as digital-out LED
blinkLED = 'D13';
configurePin(robotArduino, blinkLED, 'DigitalOutput');

% create a servo object driving PWM pin 9
% Min pulse duration: 1120 microseconds
% center: 1520 microseconds
% MaxPulseDuration: 1920 microseconds
panServo = servo(robotArduino, 'D9', 'MinPulseDuration', 10*10^-6,...
    'MaxPulseDuration', 1925*10^-6);
tiltServo = servo(robotArduino, 'D10', 'MinPulseDuration', 10*10^-6,...
    'MaxPulseDuration', 1925*10^-6);

% configure A0 pin as an analog input
    rawRangeIn = 'A0';
    configurePin(robotArduino,rawRangeIn,'AnalogInput');

% RC mode expects to start up with the joystick centered
% in MATLAB function servo position is 0-1 so 0.5 (1520) is centered

writePosition(panServo, 0); % always start servo-command at 0
writePosition(tiltServo, 0); % always start servo-command at 0

pause(5.0); % wait for arduino to send stable PWM
pause(2.0);

end
```

```
function [] = Blink(a, LED, n)
% Blink toggles Arduino a LED on and off to indicate program running
% input n is number of blinks
% no output is returned

    for bIndex = 1:n
        writeDigitalPin(a, LED, 1);
        pause(0.2);
        writeDigitalPin(a, LED, 0);
        pause(0.2);
    end
end
```

Act (store Act related local functions here)

```
function sensorData = ACTRCServoArduino(panServo, tiltServo, tilt, pan, robotArduino, rawRangeI
% Code to move the pan and tilt servos as well as take in the data
% from the sharp and the positions of the pan and tilt servos
% Note that in setup we switched around the pan/tilt names

sensorData = [0;0;0;]; % Set up the blank matrix for data collection
savedPanAngles = []; % Set up pan angle collection
savedTiltAngles = []; % Set up tilt angle collection

for i = pan
    % write the pan angle based on the pan linspace
    panAngle = i;
    writePosition(tiltServo, panAngle); % Move pan servo
    pause(1.0); % Wait some time
    for j = tilt
        % write the tilt angle based on the tilt linspace
        tiltAngle = j;
        writePosition(panServo, tiltAngle); % Move tilt servo
        pause(1.0); % Wait some time
        % Save the data within the sensorData
        sensorData =[sensorData(1,:), readVoltage(robotArduino, rawRangeIn)
                     sensorData(2,:), tiltAngle
                     sensorData(3,:), panAngle];
    end
end

end
```