```
//     BALL AND BEAM BALANCE
//
// WIL SELBY & TOM TRAPP
// 2.151 - ADVANCED SYSTEM
// DYNAMICS & CONTROL
//

#include <AFMotor.h>
#include <math.h>
//#include <avr/io.h>
//#include <avr/interrupt.h>

AF_DCMotor motor(2);

// Define pins
#define pingPin 19    // Pin 19 is set to receive ultrasonic measurements
int tilt_in = 4;   //tilt sensor input
//#define encoder0PinA  2  //Channel one of encoder
//#define encoder0PinB 14 //Channel two of encoder

//volatile double encoder0Pos = 32768;  //set the number at half of its range to avoid ove
//volatile double theta_enc = 0;


double microsecondsToCentimeters(long microseconds)    //Rangefinder functions
{
 return microseconds / 29.0 / 2;
}

//arrays
double x[10];
double x_filter;
double x_dot[10];
double x_dot_filter;
double theta[10];
double theta_dot[10];
double theta_filter;
double theta_dot_filter;
double tilt_volts;
double tilt_num;
int ff = 1;
//double theta_dot[10];
int motor_speed[10];
//int chan_A[2] = {0};
//int chan_B[2] = {0};
//chan_A[0] = 0;
//chan_A[1] = 0;
//chan_A[2] = 0;
//chan_B[0] = 0;
//chan_B[1] = 0;
//chan_B[2] = 0;
```

```
//other variables
double ref = .180;          //reference x position for system to balance ball
long time[10];           //keep track of clock
long last_time = 0;       //
long dt = 0;              //time difference
double ts = 20;           //sample time in ms; keep so I can universally change sample fre
long ts_inc = ts;         //time inc to keep up with Arduino clock
int count = 0;
int sample_count = 100000;        //keep track of number of samples
double duration, inches, cm;
double K_x = -10.7*ff;            //1.0  //1.2
                                 //.6   -9.6665   -2.5777   -0.5865    0.0075
double K_v = -3.5*ff;             //.2  -38.6660   -6.4443   -0.8583    0.0046//0.6  //.2
double K_t = -5*ff;              //100   //70
double K_w = .1*ff;               //10   //10
double K_i = .01;
double error = 0;                //-2.4166   -0.8055   -0.2146    0.0189 - not enough umph
double error_sum = 0;            // -25.7773   -4.5110   -0.6294    0.0103
int i = 0;
int j = 0;
double Prop_x = 0;
double Deriv_x = 0;
double Prop_t = 0;
double Deriv_t = 0;
double Integ = 0;
double Command = 0;

void setup(){

  while( i <= 10){
   x[i] = 0;
   x_dot[i] = 0;
   theta[i] = 0;
   theta_dot[i] = 0;
   motor_speed[i] = 0;
   time[i] = 0;
   i++;
 }

i = 0;


// pinMode(encoder0PinA, INPUT);
// digitalWrite(encoder0PinA, HIGH);       // turn on pullup resistor
//pinMode(tilt_in, INPUT);
// digitalWrite(encoder0PinB, HIGH);       // turn on pullup resistor

// attachInterrupt(0, doEncoder, CHANGE);  // encoder pin on interrupt 0 - pin 2
 //interrupts();
 Serial.begin(9600);                    //initialize serial communication
 //Serial.println("hello Tom & Wil");
 motor.setSpeed(200);
```

```
 motor.run(RELEASE);
}


void loop(){


//setup step response



  time[0] = millis();      //start time

 digitalWrite(tilt_in, LOW);
 delay(1);
 tilt_num = analogRead(tilt_in);
 tilt_volts = .0049*tilt_num;
 theta[0] = asin((tilt_volts-2.4)/2);

 // The PING))) is triggered by a HIGH pulse of 2 or more microseconds.
 // Give a short LOW pulse beforehand to ensure a clean HIGH pulse:

 pinMode(pingPin, OUTPUT);
 digitalWrite(pingPin, LOW);
 delayMicroseconds(2);
 digitalWrite(pingPin, HIGH);
 delayMicroseconds(5);
 digitalWrite(pingPin, LOW);
 // The same pin is used to read the signal from the PING))): a HIGH
 // pulse whose duration is the time (in microseconds) from the sending
 // of the ping to the reception of its echo off of an object.
 pinMode(pingPin, INPUT);
 duration = pulseIn(pingPin, HIGH);

 // convert the time into a distance
 cm = microsecondsToCentimeters(duration);
 //Serial.println(cm);
 //xfilter

 x[0] = cm/100;

 if (x[0]>.36) x[0] = .36;
 if (x[0]<.04) x[0] = .04;
 if (x[0] == .36) x[0] = x[1];
 //  Serial.print(cm);
 //  Serial.print("cm");
 //  Serial.println();
 //theta[0] = theta_enc;

//x_dot filter

x_filter = (3*x[0] + 2*x[1] + x[2])/6;
```

```
//x_filter = x[0];

x_dot_filter = (3*(x[0]-x[1]) + 2*(x[1]-x[2]) + x[2]-x[3])/(6*ts/1000);
//x_dot_filter = (x[0]-x[1])/ts/1000;

theta_filter = (3*theta[0] + 2*theta[1] + theta[2])/6;

//theta_filter = theta[0];

theta_dot_filter = (3*(theta[0]-theta[1]) + 2*(theta[1]-theta[2]) + theta[2]-theta[3])/(6*

//theta_dot_filter = (theta[0]-theta[1])/ts/1000;

//x_dot[0] = (x[0] - x[1])/(ts/1000);
//theta_dot[0] =(theta[0] + theta[1] + theta[2])/ (3*ts);

//calculate error sum
//error = (x[0] - ref)*ts/1000;
//error_sum = (error_sum+error);

//Control calc

Prop_x =  K_x*(x_filter-ref);
Deriv_x = K_v*(x_dot_filter);
Prop_t =  K_t*(theta_filter);
Deriv_t = K_w*(theta_dot_filter);
//Integ = -K_i*(error_sum);

Command = Prop_x + Deriv_x + Prop_t + Deriv_t; // + Integ;
motor_speed[0] = Command*255/12;

//if (motor_speed[0] > 50) motor_speed[0] = 50;
//if (motor_speed[0] < -50) motor_speed[0] = -50;

//if (theta_enc > 50) motor_speed[0] = 0;
//if (theta_enc < -50) motor_speed[0] = 0;

if (motor_speed[0] > 0){
motor_speed[0] = motor_speed[0];
  if (motor_speed[0] > 255) motor_speed[0] = 255;
  motor.run(FORWARD);
  motor.setSpeed(abs(motor_speed[0]));
}
if (motor_speed[0] < 0) {
  motor.run(BACKWARD);
if (motor_speed[0] < -255) motor_speed[0] = -255;
  motor.setSpeed(abs(motor_speed[0]));
}
if(j == 5){
Serial.print(time[0]);
Serial.print(",  ");
Serial.println(x_filter);
```

```
j = 0;
}

//Serial.print("ts ");
//Serial.print(ts/1000);
//Serial.print(";  ");


//Serial.print("x0  ");
//Serial.print(",  ");
////Serial.print("error  ");
//Serial.print(x_filter-ref);
//Serial.print(",  ");
////Serial.print("Prop  ");
//Serial.print(Prop_x);
//Serial.print(",  ");
////Serial.print("xdot  ");
//Serial.print(x_dot_filter);
//Serial.print(",  ");
////Serial.print("Dx  ");
//Serial.print(Deriv_x);
//Serial.print(",  ");
////Serial.print("theta ");
//Serial.print(theta[0]);
//Serial.print(",  ");
//Serial.print(Prop_t);
//Serial.print(";  ");
////Serial.print("thetadot ");
//Serial.print(theta_dot_filter);
//Serial.print(",  ");
////Serial.print("Dt ");
//Serial.print(Deriv_t);
//Serial.print(";  ");
////Serial.print("Speed ");
////Serial.print(motor_speed[0]);
////Serial.print(",  ");
////Serial.print("motor_speed[0]  ");
//Serial.print(Command);
//Serial.print(";  ");
//Serial.println(motor_speed[0]);


//age variables
x[3] = x[2];
x[2] = x[1];
x[1] = x[0];
theta[3] = theta[2];
theta[2] = theta[1];
theta[1] = theta[0];

//control sample time
time[1] = millis();      //current time
```

```
if(time[1] < time[0] + ts){
delay(time[0] + ts - time[1]);
time[1] = time[0];
j = j+1;
}

}

//   theta_enc = (encoder0Pos-32768)*90/187.68;      //Beam angle in degrees
//
//
//  chan_A[2] = chan_A[1];
//  chan_A[1] = chan_A[0];
//  chan_B[2] = chan_B[1];
//  chan_B[1] = chan_B[0];
//}

//void doEncoder(){
// //time = millis();
// if (digitalRead(encoder0PinA) == HIGH) {   // found a low-to-high on channel A
//   if (digitalRead(encoder0PinB) == LOW) {  // check channel B to see which way
//     // encoder is turning
//     encoder0Pos = encoder0Pos + 1;         // CCW
//   }
//   else {
//     encoder0Pos = encoder0Pos - 1;         // CW
//   }
// }
// else                                        // found a high-to-low on channel A
// {
//   if (digitalRead(encoder0PinB) == LOW) {   // check channel B to see which way
//     // encoder is turning
//     encoder0Pos = encoder0Pos - 1;          // CW
//   }
//   else {
//     encoder0Pos = encoder0Pos + 1;          // CCW
//   }
```