Appendix

A.1 Controller Code (m-file)

```
%% ======================================================%%
%%                  Agile Eye Robot                     %%
%%===========    Tracker Controller    =================%%

Ts=0.006;           % sampling Time
A=[0 1;0 -4.27]     % system matrix of motor 1
B=[0 14.95]';       % Input matrix
Ad=[1 0.006;0 0.02562]; % Discrete system matrix
Bd=[0 0.0897]'; % Discrete input matrix
C=[1 0]; % output matrix
Ae=[Ad zeros(2,1);-C 0]; % Extended system matrix
Be=[Bd;0]; % Extended input matrix
q=eye(3); % state weight matrix
r=1.5; % input weight matrix
q(3,3)=150
ke=lqr(Ae,Be,q,r); % Evaluate the gain by using lqr command, where Ke
is the extended gain
k1=ke(1)
k2=ke(2)
ki=-ke(3)
k=ke(1:2)
```

A.2 Microcontroller C Code

```
#define MAX_BUF        (64)  // What is the longest message Arduino can store?
char buffer[MAX_BUF];  // where we store the message until we get a ';'
int count;  // how much is in the buffer
#include <Encoder.h>
#include <TimerOne.h>
Encoder x_dir(2,3);
Encoder y_dir(18,19);
Encoder z_dir(20,21);
const int mcw1=22;
const int mccw1=24;
const int pwm1=7;
const int mcw2=26;
const int mccw2=28;
const int pwm2=8;
const int mcw3=32;
const int mccw3=30;
const int pwm3=9;

float x1=0.0;
float x1_dot=0.0;
float y1=0.0;
float y1_dot=0.0;
float z1=0.0;
float z1_dot=0.0;

float lastTime=0.0;
float sx=0.0;//////////
float sy=0.0;////////////
float sz=0.0;////////////
```

```
float outx=0.0,etax=0.0,eta_nx=0.0;
float outy=0.0,etay=0.0,eta_ny=0.0;
float outz=0.0,etaz=0.0,eta_nz=0.0;

float x1_p=0.0,y1_p=0.0;
float z1_p=0.0;

float forcex=0.0,voltagex=0.0,Kfvx=1;
float forcey=0.0,voltagey=0.0,Kfvy=1;
float forcez=0.0,voltagez=0.0,Kfvz=1;

double K1x=4.3;
double K2x=0.048;
double Kix=8.89;

double K1y=4.3;
double K2y=0.048;
double Kiy=4.95;

double K1z=5;
double K2z=0.07;
double Kiz=6.782;

double X,Y,Z;
double x_in=0.0,y_in=0.0,z_in=0.0;
double Kpm_x=360.0/32000.0;/////////////
double Kpm_y=360.0/62000.0;/////////////////
double Kpm_z=360.0/62500.0;/////////////////
float T=0.03;
```

```
void setup()
{
  pinMode(mcw1,OUTPUT);
 pinMode(mccw1,OUTPUT);
  pinMode(pwm1,OUTPUT);
   pinMode(mcw2,OUTPUT);
 pinMode(mccw2,OUTPUT);
  pinMode(pwm2,OUTPUT);
   pinMode(mcw3,OUTPUT);
 pinMode(mccw3,OUTPUT);
  pinMode(pwm3,OUTPUT);
    Timer1.initialize(30000);
  Timer1.attachInterrupt( timerIsr );
    Serial.begin(9600);
  Serial.println("Basic Encoder Test:");
}

void loop()
{

  // listen for serial commands
  while(Serial.available() > 0)
  {  // if something is available
    char c=Serial.read();  // get it
    if(count<MAX_BUF) buffer[count++]=c;  // store it
    if(buffer[count-1]==';') ;  // entire message received
  }//while

    if(count>0 && buffer[count-1]==';')
  {// we got a message and it ends with a semicolon
```

```
    buffer[count]=0;  // end the buffer so string functions work right
    Serial.print(F("\r\n"));  // echo a return character for humans
    processCommand();  // do something with the command
    ready();
  }//if

 if(outx >=0.0){
 digitalWrite(mcw1,1);
digitalWrite(mccw1,0);
if(outx<5){outx=0.0;}
 analogWrite(pwm1,outx);
  }
  if(outx < 0.0) {
digitalWrite(mcw1,0);
digitalWrite(mccw1,1);
if(-5<outx){outx=0.0;}
 analogWrite(pwm1,0-outx);
  }
  ////////////////////////////
  ////////////////////////////

   if(outy>=0.0){
 digitalWrite(mcw2,1);
digitalWrite(mccw2,0);
if(outz<5){outz=0.0;}
 analogWrite(pwm2,outy);
  }
  if(outy < 0.0) {
 digitalWrite(mcw2,0);
digitalWrite(mccw2,1);
```

```
if(-5<outy){outy=0.0;}
 analogWrite(pwm2,0-outy);
  }
  //////////////////////////
  //////////////////////
    if(outz>=0.0){
 digitalWrite(mcw3,1);
digitalWrite(mccw3,0);
if(outz<5){outz=0.0;}
 analogWrite(pwm3,outz);
  }
  if(outz < 0.0) {
 digitalWrite(mcw3,0);
digitalWrite(mccw3,1);
if(-5<outz){outz=0.0;}
 analogWrite(pwm3,0-outz);
  }
}
  ////////////////////////////////
  ////////////////////
   void timerIsr()
   {
  unsigned long now;
    now = millis();
    double dT,errorx=0.0,dErrx=0.0;
    dT =(float)(now - lastTime); //Our Sampling time
    x1=((float)(x_dir.read()))*Kpm_x;
    y1=((float)(y_dir.read()))*Kpm_y;
    z1=((float)(z_dir.read()))*Kpm_z;
  Serial.print(x1);
```

```
Serial.print("    ");
Serial.print(y1);
Serial.print("    ");
Serial.println(z1);
  X=x1;
  Y=y1;
  Z=z1;

  x1_dot=(float)((x1-x1_p)/(T));
  y1_dot=(float)((y1-y1_p)/(T));
  z1_dot=(float)((z1-z1_p)/(T));

  eta_nx=(sx-X)*T+etax;
  eta_ny=(sy-Y)*T+etay;
  eta_nz=(sz-Z)*T+etaz;

  forcex=Kix*etax-K1x*x1-K2x*x1_dot;
  voltagex=forcex*(Kfvx);
  outx=voltagex;//*255.0/5.0;
  if(outx>250){outx = 250;}
  else if(outx<-250){outx = -250;}
  x1_p=x1;
  etax=eta_nx;

forcey=Kiy*etay-K1y*y1-K2y*y1_dot;
  voltagey=forcey*(Kfvy);
  outy=voltagey;//*255.0/5.0;

if(outy>250){outy= 250;}
else if(outy<-250){outy = -250;}
```

```
    y1_p=y1;
   etay=eta_ny;

     forcez=Kiz*etaz-K1z*z1-K2z*z1_dot;
   voltagez=forcez*(Kfvz);
   outz=voltagez;//*255.0/5.0;

 if(outz>250){outz= 250;}
 else if(outz<-250){outz = -250;}
    z1_p=z1;
   etaz=eta_nz;
 }
 void processCommand(){
 sx = parsenumber('a',-1);
 sy = parsenumber('b',-1);
 sz = parsenumber('c',-1);
 Serial.println("Ready .............");
 Serial.print ("Theta_1  =   ");
 Serial.println(sx);
 Serial.print ("Theta_2  =   ");
 Serial.println(sy);
 Serial.print ("Theta_3  =   ");
 Serial.println(sz);
 Serial.println(">>");
 Serial.println("******************");
}//processCommand();

float parsenumber(char code,float val) {
 char *ptr=buffer;
 while(ptr && *ptr && ptr<buffer+count) {
```

```
    if(*ptr==code) {
      return atof(ptr+1);
    }//if
    ptr=strchr(ptr,' ')+1;
  }//while
  return val;
}//parsenumber
void ready() {
  count=0;  // clear input buffer
  Serial.print(F(">"));  // signal ready to receive input
}//ready();
```