```
public int w = 640;

public int h = 480;


import processing.serial.*;

Serial myPort;


import JMyron.*;

JMyron m;

PFont fontA; // font libary


import controlP5.*;

ControlP5 controlP5;

ControlWindow controlWindow;

ControlWindowCanvas cc;

public int RED = 255;

public int GREEN = 90;

public int BLUE = 0;

public int Threshold = 180;


// Joystick Controller

import procontroll.*;

// import net.java.games.input.*;

ControllIO controllIO;

ControllDevice joypad;

ControllStick stick;
```

```
ControllStick slider;


// Control Button Value
public boolean Laser,Flash,Swap,Safety,Scope,Auto,Start,Reset,Semi,Lock = false;
public boolean Manual = true;
public boolean Stop = true;
public boolean Wide = true;
// Hat Switch Variables
public float HX,HY = 0;
// Joy Stick Variables
public float JX,JY = 0;
public int StickSensitive = 0;
public int NaviKeySensitive = 0;
public int Balance = 50;
public int BR,BL=0;
public float OUTR,OUTL=0;
public float OUTMR,OUTML=0;
public int TJX = w/2;
public int TJY = h/2;


// Tracking Variable
public int pixel_x = w/2;
public int pixel_y = h/2;
public float area = 0;
public float range = 0;
```

```
public float range_m = 0;
public float calibrate_size = 25;
public float ObjectSize = 25;
public float RangeOffset = 0;

// Serial Out Control Variables
public int CYAW = TJX;
public int CPITCH = TJY;
public int digital_control = 0;
public int digital_control_old = 0;
public float CYAW_OLD = CYAW;
public float CPITCH_OLD= CPITCH;
public float OUTMR_OLD= OUTMR;
public float OUTML_OLD= OUTML;
public int fire_control = 0;
public int fire_control_old = 0;

// Pitch & Yaw Range Setting
public int pitch_max = 880;
public int pitch_min = 720;
public int pitch_center = 820;
public int yaw_min = 330;
public int yaw_max = 720;
public int yaw_center = 525;
```

```
// Taget Locking Variables
public float lock_range_setpoint = 0;
public int lock_center_setpoint = 320;
public boolean lock_center = false;
// Tacking PID CENTER
public float KP_C = 0.15;
public int error_C = 0;
public int position_C = 0;
public int ut_max_C = 50;
public int ut_min_C = -50;
public int ut_C = 0;
public float C_ramp_L =0;
public float C_ramp_R =0;
public int CPL = 0;
public int CPR = 0;
// Tacking PID RANGE
public float KP_R = 20;
public float error_R = 0;
public float position_R = 0;
public int ut_max_R = 50;
public int ut_min_R = -50;
public int ut_R = 0;
public float R_ramp_FW =0;
public float R_ramp_RW =0;
public int RFW = 0;
```

```
public int RRW = 0;



// Timer
public long last_timer = 0;
public long last_timer_start = 0;
public int send_start_sampling = 300; // Set sampling rate of Start/Stop Command
public int send_sampling = 30 ; // Set sampling rate of serial send
public int counting_send = 1;

// Draw Object in Second Windows
class MyCanvas extends ControlWindowCanvas {
  public void draw(PApplet theApplet) {

    // Color Seclective
     theApplet.fill(RED,GREEN,BLUE);
     theApplet.rect(10,540,100,68);
     theApplet.noFill();
    // Wheels L+R
    // L
    theApplet.fill(color(0,54,82,80));
    theApplet.rect(160,380,20,100);
    theApplet.fill(color(0,105,140));
    theApplet.rect(160,380+100-OUTL,20,OUTL);
    // R
```

```
theApplet.fill(color(0,54,82,80));
theApplet.rect(200,380,20,100);
theApplet.fill(color(0,105,140));
theApplet.rect(200,380+100-OUTR,20,OUTR);
theApplet.noStroke();
theApplet.noFill();

// Joypad Navigation
theApplet.translate(220, 240);
 theApplet.stroke(22,190,0);
 theApplet.rect(0, 0, 100, 100);
 theApplet.line(0, 50, 100, 50);
 theApplet.line(50, 0, 50, 100);
 //theApplet.rect(40, 40, 20, 20);
 if( JX != 0 || JY != 0){
 theApplet.fill(255,0,0);
 }
 float CJX = map(JX, -100, 100, -50, 50);
 float CJY = map(JY, -100, 100, -50, 50);
 theApplet.rect(45+CJX, 45+CJY, 10, 10);
 theApplet.noFill();
 theApplet.translate(-220, -240);

 // Hat Switch Navigation Key
 int Navifill = color(255,25,0);
```

```
theApplet.fill(40);
theApplet.stroke(22,190,0);
// Center
if ( HX == 0 && HY == 0 ){theApplet.fill(Navifill);}
theApplet.rect(70, 430, 20, 20);
theApplet.noFill();
// N
if ( HX == 0 && HY == -1 ){theApplet.fill(Navifill);}
theApplet.rect(70, 380, 20, 40);
theApplet.noFill();
// S
if ( HX == 0 && HY == 1 ){theApplet.fill(Navifill);}
theApplet.rect(70, 460, 20, 40);
theApplet.noFill();
// W
if ( HX == -1 && HY == 0 ){theApplet.fill(Navifill);}
theApplet.rect(20, 430, 40, 20);
theApplet.noFill();
// E
if ( HX == 1 && HY == 0 ){theApplet.fill(Navifill);}
theApplet.rect(100, 430, 40, 20);
theApplet.noFill();
// NE
if ( HX == 0.70710677 && HY == -0.70710677 ){theApplet.fill(Navifill);}
```

```
    theApplet.rect(100, 400, 20, 20);
    theApplet.noFill();
    // NW
    if ( HX == -0.70710677 && HY == -0.70710677 ){theApplet.fill(Navifill);}
    theApplet.rect(40, 400, 20, 20);
    theApplet.noFill();
    // SE
    if ( HX == 0.70710677 && HY == 0.70710677 ){theApplet.fill(Navifill);}
    theApplet.rect(100, 460, 20, 20);
    theApplet.noFill();
    // SW
    if ( HX == -0.70710677 && HY == 0.70710677 ){theApplet.fill(Navifill);}
    theApplet.rect(40, 460, 20, 20);
    theApplet.noFill();
//     HX = 0;
//     HY = 0;
  }
}

void setup(){
 // Serial Setup
 String portName = Serial.list()[1];
 myPort = new Serial(this, portName, 9600);

 // font setup
```

```
fontA = loadFont("CourierNew36.vlw");
textFont(fontA, 10);

// GUI Setup
controlP5 = new ControlP5(this);
controlP5.setAutoDraw(false);
controlWindow = controlP5.addControlWindow("controlP5window",100,100,350,630);
controlWindow.setBackground(color(40));
controlWindow.setUpdateMode(ControlWindow.NORMAL); // Update Control Windows Every time

// Hat switch speed control
Controller SPEED = controlP5.addSlider("SPEED",0,100,240,380,20,100);
SPEED.setWindow(controlWindow);
// Balance R
Controller BCR = controlP5.addSlider("BR",0,100,290,380,10,100);
BCR.setWindow(controlWindow);
controlP5.controller("BR").setValue(100);
// Balance L
Controller BCL = controlP5.addSlider("BL",0,100,320,380,10,100);
BCL.setWindow(controlWindow);
controlP5.controller("BL").setValue(100);

// Joystick Sensitive
Controller JST = controlP5.addSlider("StickSensitive",0,100,10,330,100,10);
JST.setWindow(controlWindow);
```

```
controlP5.controller("StickSensitive").setValue(9);

// Hat Switch Naviagation Sensitive
Controller NST = controlP5.addSlider("NaviKeySensitive",0,100,10,300,100,10);
NST.setWindow(controlWindow);
controlP5.controller("NaviKeySensitive").setValue(9);

Controller OS = controlP5.addSlider("ObjectSize",0,100,10,270,100,10);
OS.setWindow(controlWindow);
controlP5.controller("ObjectSize").setValue(25);

Controller RF = controlP5.addSlider("RangeOffset",-3,3,10,240,100,10);
RF.setWindow(controlWindow);
controlP5.controller("RangeOffset").setValue(0);

// RGB Adjust Control Bar
Controller Radjustbar = controlP5.addSlider("RED",0,255,170,540,100,10);
Radjustbar.setWindow(controlWindow);
Controller Gadjustbar = controlP5.addSlider("GREEN",0,255,170,560,100,10);
Gadjustbar.setWindow(controlWindow);
Controller Badjustbar = controlP5.addSlider("BLUE",0,255,170,580,100,10);
Badjustbar.setWindow(controlWindow);
Controller Thresholdbar = controlP5.addSlider("Threshold",0,255,170,600,100,10);
Thresholdbar.setWindow(controlWindow);
```

```
// RGB Selection button
Controller RED_button = controlP5.addButton("RE",0,120,540,20,17);
RED_button.setWindow(controlWindow);
RED_button.setColorBackground(color(255,0,0));
RED_button.setColorActive(color(255,0,0));
RED_button.setId(1);

Controller YELLOW_button = controlP5.addButton("YE",0,120,557,20,17);
YELLOW_button.setWindow(controlWindow);
YELLOW_button.setColorBackground(color(255,255,0));
YELLOW_button.setColorActive(color(255,255,0));
YELLOW_button.setId(2);

Controller GREEN_button = controlP5.addButton("GR",0,120,574,20,17);
GREEN_button.setWindow(controlWindow);
GREEN_button.setColorBackground(color(0,255,0));
GREEN_button.setColorActive(color(0,255,0));
GREEN_button.setId(3);

Controller LIGHTBLUE_button = controlP5.addButton("LB",0,120,591,20,17);
LIGHTBLUE_button.setWindow(controlWindow);
LIGHTBLUE_button.setColorBackground(color(0,255,255));
LIGHTBLUE_button.setColorActive(color(0,255,255));
LIGHTBLUE_button.setId(4);
```

```
Controller BLUE_button = controlP5.addButton("BU",10,140,591,20,17);
BLUE_button.setWindow(controlWindow);
BLUE_button.setColorBackground(color(0,0,255));
BLUE_button.setColorActive(color(0,0,255));
BLUE_button.setId(5);

Controller PINK_button = controlP5.addButton("PI",10,140,574,20,17);
PINK_button.setWindow(controlWindow);
PINK_button.setColorBackground(color(255,0,255));
PINK_button.setColorActive(color(255,0,255));
PINK_button.setId(6);

Controller WHITE_button = controlP5.addButton("WH",10,140,557,20,17);
WHITE_button.setWindow(controlWindow);
WHITE_button.setColorBackground(color(255,255,255));
WHITE_button.setColorActive(color(255,255,255));
WHITE_button.setId(7);

Controller BLACK_button = controlP5.addButton("BK",10,140,540,20,17);
BLACK_button.setWindow(controlWindow);
BLACK_button.setColorBackground(color(0,0,0));
BLACK_button.setColorActive(color(0,0,0));
BLACK_button.setId(8);

// Begin Control Buttons
```

```
int C_ColorBackground = color(0,100,0);
int C_setColorActive = color(0,150,0);

Controller Laser_button = controlP5.addButton("Laser Sight",0,5,5,80,40);
Laser_button.setWindow(controlWindow);
Laser_button.setColorBackground(C_ColorBackground);
Laser_button.setColorActive(C_setColorActive);
Laser_button.setId(100);

Controller Flash_button = controlP5.addButton("Flash Light",0,95,5,80,40);
Flash_button.setWindow(controlWindow);
Flash_button.setColorBackground(C_ColorBackground);
Flash_button.setColorActive(C_setColorActive);
Flash_button.setId(101);

Controller Swap_button = controlP5.addButton("Swap Mode",0,185,5,80,40);
Swap_button.setWindow(controlWindow);
Swap_button.setColorBackground(C_ColorBackground);
Swap_button.setColorActive(C_setColorActive);
Swap_button.setId(102);

Controller Safety_button = controlP5.addButton("SAFETY",0,275,5,70,40);
Safety_button.setWindow(controlWindow);
Safety_button.setColorBackground(C_ColorBackground);
Safety_button.setColorActive(C_ColorBackground);
```

```
Safety_button.setId(103);

Controller Scope_button = controlP5.addButton("Scope Vision",0,5,55,120,40);
Scope_button.setWindow(controlWindow);
Scope_button.setColorBackground(C_ColorBackground);
Scope_button.setColorActive(C_setColorActive);
Scope_button.setId(104);

Controller Wide_button = controlP5.addButton("Wide Vision",0,135,55,120,40);
Wide_button.setWindow(controlWindow);
Wide_button.setColorBackground(C_ColorBackground);
Wide_button.setColorActive(C_setColorActive);
Wide_button.setId(105);

Controller Auto_button = controlP5.addButton("AUTO",0,265,55,80,40);
Auto_button.setWindow(controlWindow);
Auto_button.setColorBackground(C_ColorBackground);
Auto_button.setColorActive(C_setColorActive);
Auto_button.setId(106);

Controller Start_button = controlP5.addButton("START",0,5,105,80,40);
Start_button.setWindow(controlWindow);
Start_button.setColorBackground(C_ColorBackground);
Start_button.setColorActive(C_setColorActive);
Start_button.setId(107);
```

```
Controller Stop_button = controlP5.addButton("STOP",0,95,105,80,40);
Stop_button.setWindow(controlWindow);
Stop_button.setColorBackground(C_ColorBackground);
Stop_button.setColorActive(C_setColorActive);
Stop_button.setId(108);

Controller Reset_button = controlP5.addButton("RESET",0,185,105,70,40);
Reset_button.setWindow(controlWindow);
Reset_button.setColorBackground(C_ColorBackground);
Reset_button.setColorActive(C_setColorActive);
Reset_button.setId(109);

Controller Semi_button = controlP5.addButton("Semi-Auto",0,265,105,80,40);
Semi_button.setWindow(controlWindow);
Semi_button.setColorBackground(C_ColorBackground);
Semi_button.setColorActive(C_setColorActive);
Semi_button.setId(110);

Controller Manual_button = controlP5.addButton("MANUAL",0,265,155,80,40);
Manual_button.setWindow(controlWindow);
Manual_button.setColorBackground(C_ColorBackground);
Manual_button.setColorActive(C_setColorActive);
Manual_button.setId(111);
```

```
Controller LOCK_button = controlP5.addButton("LOCK",0,185,155,70,40);
LOCK_button.setWindow(controlWindow);
LOCK_button.setColorBackground(C_ColorBackground);
LOCK_button.setColorActive(C_setColorActive);
LOCK_button.setId(112);



// Object Control in Second Windows
cc = new MyCanvas();
cc.pre();
controlWindow.addCanvas(cc);

// JoyStick Interface
controllIO = ControllIO.getInstance(this);
joypad = controllIO.getDevice("4-axis 8-button joystick");
joypad.plug(this, "NaviKey", ControllIO.WHILE_PRESS, 0);
joypad.plug(this, "NaviKey_Release", ControllIO.ON_RELEASE, 0);
joypad.plug(this, "button_1_press", ControllIO.ON_PRESS, 1);
joypad.plug(this, "button_1_release", ControllIO.ON_RELEASE, 1);
joypad.plug(this, "button_2_press", ControllIO.ON_PRESS, 2);
joypad.plug(this, "button_2_release", ControllIO.ON_RELEASE, 2);
joypad.plug(this, "button_3_press", ControllIO.ON_PRESS, 3);
joypad.plug(this, "button_3_release", ControllIO.ON_RELEASE, 3);
joypad.plug(this, "button_4_press", ControllIO.ON_PRESS, 4);
joypad.plug(this, "button_4_release", ControllIO.ON_RELEASE, 4);
```

```
  joypad.plug(this, "button_5_press", ControllIO.ON_PRESS, 5);
  joypad.plug(this, "button_5_release", ControllIO.ON_RELEASE, 5);
  joypad.plug(this, "button_6_press", ControllIO.ON_PRESS, 6);
  joypad.plug(this, "button_6_release", ControllIO.ON_RELEASE, 6);
  joypad.plug(this, "button_7_press", ControllIO.ON_PRESS, 7);
  joypad.plug(this, "button_7_release", ControllIO.ON_RELEASE, 7);
  joypad.plug(this, "button_8_press", ControllIO.ON_PRESS, 8);
  joypad.plug(this, "button_8_release", ControllIO.ON_RELEASE, 8);;

stick = joypad.getStick(0);
slider = joypad.getStick(1);

stick.setTolerance(0.1f);
stick.setMultiplier(50);

slider.setTolerance(0.1f);
slider.setMultiplier(50);

  // JMyron Setup
  size(w,h);
  m = new JMyron();
  m.start(w,h);
  m.findGlobs(1);
  m.trackColor(RED,GREEN,BLUE,Threshold);
}
```

```
// End void setup


// Key Setting
void keyPressed() {
  if(key == 's'){
  m.settings();
  }
}


// Button Even Control
void controlEvent(ControlEvent theEvent) {
  if(theEvent.controller().id() == 1)
    { RED = 255;
      GREEN = 0;
      BLUE =0;
      controlP5.controller("RED").setValue(255);
      controlP5.controller("GREEN").setValue(0);
      controlP5.controller("BLUE").setValue(0);
    }
  if(theEvent.controller().id() == 2)
    { RED = 255;
      GREEN = 255;
      BLUE =0;
      controlP5.controller("RED").setValue(255);
      controlP5.controller("GREEN").setValue(255);
```

```
    controlP5.controller("BLUE").setValue(0);
  }
if(theEvent.controller().id() == 3)
  { RED = 0;
    GREEN = 255;
    BLUE =0;
    controlP5.controller("RED").setValue(0);
    controlP5.controller("GREEN").setValue(255);
    controlP5.controller("BLUE").setValue(0);
  }
if(theEvent.controller().id() == 4)
  { RED = 0;
    GREEN = 255;
    BLUE =255;
    controlP5.controller("RED").setValue(0);
    controlP5.controller("GREEN").setValue(255);
    controlP5.controller("BLUE").setValue(255);
  }
if(theEvent.controller().id() == 5)
  { RED = 0;
    GREEN = 0;
    BLUE =255;
    controlP5.controller("RED").setValue(0);
    controlP5.controller("GREEN").setValue(0);
    controlP5.controller("BLUE").setValue(255);
```

```
  }
if(theEvent.controller().id() == 6)
  { RED = 255;
    GREEN = 0;
    BLUE =255;
    controlP5.controller("RED").setValue(255);
    controlP5.controller("GREEN").setValue(0);
    controlP5.controller("BLUE").setValue(255);
  }
if(theEvent.controller().id() == 7)
  { RED = 255;
    GREEN = 255;
    BLUE =255;
    controlP5.controller("RED").setValue(255);
    controlP5.controller("GREEN").setValue(255);
    controlP5.controller("BLUE").setValue(255);
  }
if(theEvent.controller().id() == 8)
  { RED = 0;
    GREEN = 0;
    BLUE =0;
    controlP5.controller("RED").setValue(0);
    controlP5.controller("GREEN").setValue(0);
    controlP5.controller("BLUE").setValue(0);
  }
```

```
 // Button Logic
if(theEvent.controller().id() == 100)
 {
  if(Laser == false){Laser = true; digital_control = 131;}
  else{Laser = false; digital_control = 231;}
 }
if(theEvent.controller().id() == 101)
 {
  if(Flash == false){Flash = true; digital_control = 130;}
  else{Flash = false; digital_control = 230;}
 }
if(theEvent.controller().id() == 102)
 {
  if(Swap == false){Swap = true;}
  else{Swap = false;}
 }
if(theEvent.controller().id() == 103)
 {
 }
if(theEvent.controller().id() == 104)
 {
  if(Scope == false){Scope = true; Wide = false; digital_control = 133;}
  Manual = true; Auto = false; Semi = false;
 }
```

```
if(theEvent.controller().id() == 105)
 {
   if(Wide == false){Wide = true; Scope = false; digital_control = 233;}
 }
if(theEvent.controller().id() == 107)
 {
   if(Start == false){Start = true; Stop = false;}
 }
if(theEvent.controller().id() == 108)
 {
   if(Stop == false){Stop = true; Start = false;}
 }
if(theEvent.controller().id() == 106)
 {
   if(Scope != true)
   {
     if(Auto == false){Auto = true; Semi = false; Manual = false;}
   }
 }
if(theEvent.controller().id() == 110)
 {
   if(Scope != true)
   {
   if(Semi == false){Semi = true; Auto = false; Manual = false;}
   }
```

```
  }
 if(theEvent.controller().id() == 111)
  {
    if(Manual == false){Manual = true; Auto = false; Semi = false;}
  }
 if(theEvent.controller().id() == 112)
  {
    if(Lock == false && Auto == true){
      Lock = true;
      lock_range_setpoint = range_m;
      }
    else{Lock = false; lock_range_setpoint = 0;}
  }
 if(theEvent.controller().id() == 109){
  Lock = false;
  lock_range_setpoint =0;
 }
}
// End Button Even


void draw(){
 m.trackColor(RED,GREEN,BLUE,Threshold);
 m.update();
 int[] img = m.image();
```

```
//first draw the camera view onto the screen
loadPixels();
for(int i=0;i<width*height;i++){
   pixels[i] = img[i];
}
updatePixels();


noFill();
int[][] a;



//draw center points of globs
a = m.globCenters();
stroke(255,255,0);
for(int i=0;i<a.length;i++){
  int[] p = a[i];
  point(p[0],p[1]);
}



//draw bounding boxes of globs
a = m.globBoxes();
float [] d = new float [999];
stroke(255,0,0);
for(int i=0;i<a.length;i++){
```

```
 int[] b = a[i];
 d[i]  = (b[2]*b[3]);
 rect(b[0], b[1], b[2], b[3]);
}

// sort array and draw biggest area
d = sort(d);
d = reverse(d);
for(int i=0;i<a.length;i++){
  int[] b = a[i];
  if ( b[2]*b[3] == d[0]){
    int x = b[0];
    int y = b[1];
    int w = b[2];
    int h = b[3];
    stroke(255,255,0);
    line(x+w/2,y,x+w/2,y+h);
    line(x,y+h/2,x+w,y+h/2);
    pixel_x = x+w/2;
    pixel_y = y+h/2;

    // range finder
    int xs = pixel_x;
    int ys = pixel_y;
    float xr = 1;
```

```
    float yr = 1;
    float vtc = 0;
    if(pixel_x > 320){
      xs = int(map(pixel_x, 321, 640,320,1)); //invert x coordinate value
    }
    if(pixel_y > 240){
      ys = int(map(pixel_y, 241, 480,240,1)); //invert y coordinate value
    }
//    xr = 0.949483 + (pow(0.990907,xs)) - 2.88876/(xs - 22.2815); // reprise x coordinate
//    yr = 0.784774 + 63.4961/(40.5983 + ys); // reprise y coordinate
    float object_ratio = calibrate_size/(ObjectSize); // calculate taget size to ratio
    area = (w*h); // calculate area
    vtc = sqrt((xs*xs)+(ys*ys)); // calculate vector size
    area = area*((229.624 + 0.593473*vtc)/(53.7319 + vtc)); // covert area back by vector size
//    float rpx = (xr*area)-area;
//    float rpy = (yr*area)-area;
    //area = area+rpx+rpy;
    //area = area + (abs(rpx-rpy));
    //area = ((xr*area)+(yr*area))/2;
    range = (16744.5*(pow(area,-0.514742)))/object_ratio; // convert size to distant
    range_m = (range/100)+RangeOffset; // convert cm to meter
//    print("X  ");
//    print(x+w/2);
//    print("  Y  ");
//    print(y+h/2);
```

```
   // println(range);
    textFont(fontA, 20);
    text(range_m,pixel_x+w/2,pixel_y+h/2);
    text("meters",pixel_x+w/2,pixel_y+h/2+25);
    textFont(fontA, 10);
    }
  }
// Joystick Button Interface Control
if(Auto != true){Lock = false; lock_range_setpoint =0;}

if(Laser == true)
{
  controlP5.controller("Laser Sight").setColorBackground(color(0,200,0));
  controlP5.controller("Laser Sight").setColorActive(color(0,200,0));
}
else{
  controlP5.controller("Laser Sight").setColorBackground(color(0,100,0));
  controlP5.controller("Laser Sight").setColorActive(color(0,150,0));
}
if(Flash == true)
{
  controlP5.controller("Flash Light").setColorBackground(color(0,200,0));
  controlP5.controller("Flash Light").setColorActive(color(0,200,0));
}
else{
```

```
  controlP5.controller("Flash Light").setColorBackground(color(0,100,0));
  controlP5.controller("Flash Light").setColorActive(color(0,150,0));
}
if(Swap == true)
{
  controlP5.controller("Swap Mode").setColorBackground(color(0,200,0));
  controlP5.controller("Swap Mode").setColorActive(color(0,200,0));
}
else{
  controlP5.controller("Swap Mode").setColorBackground(color(0,100,0));
  controlP5.controller("Swap Mode").setColorActive(color(0,150,0));
}
if(Scope == true)
{
  controlP5.controller("Scope Vision").setColorBackground(color(0,200,0));
  controlP5.controller("Scope Vision").setColorActive(color(0,200,0));
}
else{
  controlP5.controller("Scope Vision").setColorBackground(color(0,100,0));
  controlP5.controller("Scope Vision").setColorActive(color(0,150,0));
}
if(Wide == true)
{
  controlP5.controller("Wide Vision").setColorBackground(color(0,200,0));
  controlP5.controller("Wide Vision").setColorActive(color(0,200,0));
```

```
}
else{
  controlP5.controller("Wide Vision").setColorBackground(color(0,100,0));
  controlP5.controller("Wide Vision").setColorActive(color(0,150,0));
}
if(Safety == true)
{
  controlP5.controller("SAFETY").setColorBackground(color(255,10,0));
}
else{
  controlP5.controller("SAFETY").setColorBackground(color(0,100,0));
}
if(Start == true)
{
  controlP5.controller("START").setColorBackground(color(255,100,0));
  controlP5.controller("START").setColorActive(color(255,55,0));
}
else{
  controlP5.controller("START").setColorBackground(color(0,100,0));
  controlP5.controller("START").setColorActive(color(0,150,0));
}
if(Manual == true)
{
  controlP5.controller("MANUAL").setColorBackground(color(0,200,0));
  controlP5.controller("MANUAL").setColorActive(color(0,200,0));
```

```
}
else{
  controlP5.controller("MANUAL").setColorBackground(color(0,100,0));
  controlP5.controller("MANUAL").setColorActive(color(0,150,0));
}
if(Auto == true)
{
  controlP5.controller("AUTO").setColorBackground(color(0,45,90));
  controlP5.controller("AUTO").setColorActive(color(0,30,60));
}
else{
  controlP5.controller("AUTO").setColorBackground(color(0,100,0));
  controlP5.controller("AUTO").setColorActive(color(0,150,0));
}
if(Semi == true)
{
  controlP5.controller("Semi-Auto").setColorBackground(color(225,225,0));
  controlP5.controller("Semi-Auto").setColorActive(color(225,225,0));
}
else{
  controlP5.controller("Semi-Auto").setColorBackground(color(0,100,0));
  controlP5.controller("Semi-Auto").setColorActive(color(0,150,0));
}
if(Lock == true)
{
```

```
  controlP5.controller("LOCK").setColorBackground(color(255,90,0));
  controlP5.controller("LOCK").setColorActive(color(255,90,0));
}
else{
  controlP5.controller("LOCK").setColorBackground(color(0,100,0));
  controlP5.controller("LOCK").setColorActive(color(0,150,0));
}



// Joystick Control Calculation

    // Stick X
    float pojoyx = stick.getX();
    pojoyx = constrain(pojoyx, -31, 31);
    pojoyx = map(pojoyx, -31, 31, -100, 100);
    JX = pojoyx;
    //println(stick.getX());
    //println(stick.getY());

    // Stick Y
    float pojoyy = stick.getY();
    pojoyy = constrain(pojoyy, -26, 26);
    pojoyy = map(pojoyy, -26, 26, -100, 100);
    JY = pojoyy;
```

```
// Joy Stick Manual Aim
float stm,stn=0; // stick
float ntm,ntn=0; // Navi
stm = StickSensitive;
stn = StickSensitive*(-1);
ntm = NaviKeySensitive;
ntn = NaviKeySensitive*(-1);

// Swap Mode
if(Swap == false)
 {
   TJX += map(JX, -100, 100, stn, stm);
   TJY += map(JY, -100, 100, stn, stm);
 }
if(Swap == true)
 {
   TJX += map(HX, -1, 1, ntn, ntm);
   TJY += map(HY, -1, 1, ntn, ntm);
 }

//println(TJX+":"+TJY);

// SPEED
```

```
    float speed = slider.getX();
    speed = constrain(speed, -40, 30);
    speed = map(speed, 30, -40, 0, 100);
    controlP5.controller("SPEED").setValue(speed);
    float SPEEDR,SPEEDL = 0;

    // Balance R + L
    int SBR = 0;
    int SBL = 0;
    int ispeed = int(speed); // convert speed to int

     SBR = (ispeed*BR)/100;
     SBL = (ispeed*BL)/100;

// Wheel mixer
    if(Swap == true)
     {
       // calculate joystick wheels control
       float joyangle = atan2(JX,JY);
       joyangle = (joyangle*180)/PI;
       joyangle = map(joyangle, -180, 180, 0, 360);
       float vectorjxy = sqrt(JX*JX+JY*JY);
       vectorjxy = constrain(vectorjxy, 0, 100);

       //println(joyangle);
```

```
// NE
  if( joyangle >= 270 && joyangle <= 359 )
    {
      OUTL = ((JY*-1)*SBL)/100;
      OUTR = ((vectorjxy - JX)*SBR)/100;
      OUTML = (OUTL+400); //FW
      OUTMR = (OUTR+400); //FW
    }
// NW
  if( joyangle >= 0 && joyangle <= 90 )
    {
      OUTL = ((vectorjxy - (JX*-1))*SBL)/100;
      OUTR = ((JY*-1)*SBR)/100;
      OUTML = (OUTL+400); //FW
      OUTMR = (OUTR+400); //FW
    }
 // SW
  if ( joyangle >= 91 && joyangle <= 179 )
    {
      OUTL = ((vectorjxy - (JX*-1))*SBL)/100;
      OUTR = (JY*SBR)/100;
      OUTML = (OUTL+200); //RW
      OUTMR = (OUTR+200); //RW
    }
 // SE
```

```
          if ( joyangle >= 181 && joyangle <= 269 )
            {
              OUTL = (JY*SBL)/100;
              OUTR = ((vectorjxy - JX)*SBR)/100;
              OUTML = (OUTL+200); //RW
              OUTMR = (OUTR+200); //RW
            }
          // North
          if ( joyangle == 360 )
            {
              OUTL = ((JY*-1)*SBL/100);
              OUTR = ((JY*-1)*SBR/100);
              OUTML = (OUTL+400); //FW
              OUTMR = (OUTR+400); //FW
            }
          // South
          if ( joyangle == 180 && JY > 0 )
            {
              OUTL = ((JY*SBL)/100);
              OUTR = ((JY*SBR)/100);
              OUTML = (OUTL+200); //RW
              OUTMR = (OUTR+200); //RW
            }
//          // West
//          if ( joyangle == 90 )
```

```
//            {
//              OUTL = (((JX*-1)*SBL)/100);
//              OUTR = (((JX*-1)*SBR)/100);
//            }
//          // East
//           if ( joyangle == 270 )
//            {
//              OUTL = ((JX*SBL)/100);
//              OUTR = ((JX*SBR)/100);
//            }
          // No move
           if( joyangle == 180 && JX == 0 && JY == 0)
             {
               OUTL=0;
               OUTR=0;
               OUTML = OUTL+100; //STOP
               OUTMR = OUTR+100; //STOP
             }
         }
        if(Swap == false)
         {
         // Center
         if ( HX == 0 && HY == 0 )
         {
           OUTL =0;
```

```
  OUTR =0;
  OUTML = OUTL+100; //STOP
  OUTMR = OUTR+100; //STOP
}
// N
if ( HX == 0 && HY == -1 )
{
  OUTL = SBL;
  OUTR = SBR;
  OUTML = (OUTL+400); //FW
  OUTMR = (OUTR+400); //FW
}
// S
if ( HX == 0 && HY == 1 )
{
  OUTL = SBL;
  OUTR = SBR;
  OUTML = (OUTL+200); //RW
  OUTMR = (OUTR+200); //RW
}
// W
if ( HX == -1 && HY == 0 )
{
  OUTL = SBL;
  OUTR = SBR;
```

```
  OUTML = (OUTL+200); //RW
  OUTMR = (OUTR+400); //FW
}
// E
if ( HX == 1 && HY == 0 )
{
  OUTL = SBL;
  OUTR = SBR;
  OUTML = (OUTL+400); //FW
  OUTMR = (OUTR+200); //RW
}
// NE
if ( HX == 0.70710677 && HY == -0.70710677 )
{
  OUTL = SBL;
  OUTR = SBR*0.707;
  OUTML = (OUTL+400); //FW
  OUTMR = (OUTR+200); //RW
}
// NW
if ( HX == -0.70710677 && HY == -0.70710677 )
{
  OUTL = SBL*0.707;
  OUTR = SBR;
  OUTML = (OUTL+200); //RW
```

```
          OUTMR = (OUTR+400); //FW
        }
        // SE
        if ( HX == 0.70710677 && HY == 0.70710677 )
        {
          OUTL = SBL;
          OUTR = SBR*0.707;
          OUTML = (OUTL+400); //FW
          OUTMR = (OUTR+200); //RW
        }
        // SW
        if ( HX == -0.70710677 && HY == 0.70710677 )
        {
          OUTL = SBL*0.707;
          OUTR = SBR;
          OUTML = (OUTL+200); //RW
          OUTMR = (OUTR+400); //FW
        }
      }
//// End Wheels Mixer //
      // End Navi Control

    //println(slider.getX());
    //println(SBR+":"+SBL);
```

```
TJX = constrain(TJX, 0, w);
TJY = constrain(TJY, 0, h);

// Scope Sight
if(Scope == false){
rectMode(CENTER);
stroke(255,0,0);
fill(255,0,0,99);
rect(TJX, TJY, 20,20);
noStroke();
rectMode(CORNER);
}else{
  rectMode(CENTER);
  stroke(0,200,0);
  fill(0,200,0,99);
  rect(TJX, TJY, 20,20);
  noStroke();
  noFill();
  rectMode(CORNER);

  stroke(0,200,0);
  line(0,240,640,240);
  line(320,0,320,480);
  noStroke();
}
```

```
//println(TJX+":"+TJY);
//println(JX+":"+JY);
//println(HX+":"+HY);

if(Manual == true) // Manual Mode
{
  float JYAW = map(TJX, 0, w, yaw_min, yaw_max);
  CYAW = int(JYAW);
  float JPITCH = map(TJY, 0, h, pitch_min, pitch_max);
  CPITCH = int(JPITCH);
}
if(Semi == true) // Semi Auto Mode
{
  float AYAW = (0.3639*pixel_x) +409.4824; // YAW Y = MX+C
  CYAW = int(AYAW);
  CYAW = constrain(CYAW, yaw_min, yaw_max);

  float ADPITCH15 = (-0.3021*pixel_y)+(886.715);
  float ADPITCH25 = (-0.3284*pixel_y)+907.3915;
  float ADP_ERROR = ADPITCH25 - ADPITCH15;
  float ADPITCH = (ADP_ERROR * (range_m-1.5))+ ADPITCH15 ;
  CPITCH = int(ADPITCH);
  CPITCH = constrain(CPITCH, pitch_min, pitch_max);
}
```

```
if(Auto == true) // Auto Mode
{
  OUTML = 100;
  OUTMR = 100;

  float AYAW = (0.3639*pixel_x) +409.4824; // YAW Y = MX+C
  CYAW = int(AYAW);
  CYAW = constrain(CYAW, yaw_min, yaw_max);

  float ADPITCH15 = (-0.3021*pixel_y)+(886.715);
  float ADPITCH25 = (-0.3284*pixel_y)+907.3915;
  float ADP_ERROR = ADPITCH25 - ADPITCH15;
  float ADPITCH = (ADP_ERROR * (range_m-1.5))+ ADPITCH15 ;
  CPITCH = int(ADPITCH);
  CPITCH = constrain(CPITCH, pitch_min, pitch_max);

  if(Lock == true){
  int limit_C_max = 10;
  int limit_C_min = -10;
  int limit_R_max = 7;
  int limit_R_min = -7;
  PID_C();
  // Turn Right
  if(ut_C > limit_C_max){
```

```
        C_ramp_L = 0;
        R_ramp_FW = 0;
        R_ramp_RW = 0;

        C_ramp_R = C_ramp_R+0.5;
        if(C_ramp_R > ut_C){ C_ramp_R = ut_C;}
//        print("Right  ");
//        println(C_ramp_R);
        CPR = int(C_ramp_R);

        OUTML = (CPR+200);
        OUTMR = (CPR+400);
      }
      // Turn Left
      else if(ut_C < limit_C_min){
        C_ramp_R = 0;
        R_ramp_FW = 0;
        R_ramp_RW = 0;

        C_ramp_L = C_ramp_L+(-0.5);
        if(C_ramp_L < ut_C){ C_ramp_L = ut_C;}
//        print("Left  ");
//        println(C_ramp_L);
        CPL = int(C_ramp_L);
```

```
  OUTML = ((CPL*-1)+400);
  OUTMR = ((CPL*-1)+200);
 }
 // Center Point
 else if(ut_C <= limit_C_max && ut_C >= limit_C_min){
  PID_R();
 // Reward
  if(ut_R > limit_R_max){ // RW
  C_ramp_L = 0;
  C_ramp_R = 0;
  R_ramp_FW = 0;

  R_ramp_RW = R_ramp_RW+7.5;
  if(R_ramp_RW > ut_R){R_ramp_RW = ut_R;}
  RRW = int(R_ramp_RW);
//   print("RW  ");
//   println(RRW);

   OUTML = (RRW+200);
   OUTMR = (RRW+200);
  }
 // Forward
  else if(ut_R < limit_R_min){ // FW
  C_ramp_L = 0;
  C_ramp_R = 0;
```

```
        R_ramp_RW = 0;


       R_ramp_FW = R_ramp_FW+(-7.5);
       if(R_ramp_FW < ut_R){R_ramp_FW = ut_R;}
       RFW = int(R_ramp_FW);
//       print("FW  ");
//       println(R_ramp_FW);


         OUTML = ((RFW*-1)+400);
         OUTMR = ((RFW*-1)+400);
        }
        else{
            OUTML = 100;
            OUTMR = 100;
        }
      }
      else{
      C_ramp_L = 0;
      C_ramp_R = 0;
      R_ramp_FW = 0;
      R_ramp_RW = 0;
      OUTML = 100;
      OUTMR = 100;
      }
      // debug
```

```
//       print("error  ");
//       print(error_C);
//       print("  ut  ");
//       println(ut_C);
// debug
//       print("error  ");
//       print(error_R);
//       print("  ut  ");
//       println(ut_R);
      }
     }

     // Command Serial Send Timer Control
     long timer_s = millis();
     if(timer_s - last_timer_start >= send_start_sampling){
       last_timer_start = timer_s;
       if(Start == true){
         sendstart();
       }else{
         sendstop();
       }
     }

     long timer_m = millis();
     if (timer_m - last_timer >= send_sampling){
```

```
last_timer = timer_m;
if(Start == true){ // Start Send When Start Button On
 //println("start");
 //sendstart();

 if(OUTML > 200 || OUTML != OUTML_OLD && counting_send == 1){
 sendleft();
 OUTML_OLD= OUTML;
 //print(OUTML);
 //println("  Left");
 }else{
  if(counting_send == 1){
  counting_send = counting_send  +1;
  }
 }

 if(OUTMR > 200 || OUTMR != OUTMR_OLD && counting_send  == 2){
 sendright();
 OUTMR_OLD= OUTMR;
 //print(OUTMR);
 //println("  Right");
 }else{
  if(counting_send == 2){
  counting_send = counting_send  +1;
  }
```

```
}

if(CPITCH != CPITCH_OLD && counting_send == 3){
sendpitch();
CPITCH_OLD= CPITCH;
}else{
  if(counting_send == 3){
  counting_send = counting_send  +1;
  }
}

if(CYAW != CYAW_OLD && counting_send == 4){
sendyaw();
CYAW_OLD = CYAW;
}else{
  if(counting_send == 4){
  counting_send = counting_send  +1;
  }
}

if(digital_control != digital_control_old && counting_send == 5){
senddigital();
digital_control_old = digital_control;
}else{
  if(counting_send == 5){
```

```
    counting_send = counting_send  +1;
    }
   }
   counting_send = counting_send+1;
   if(counting_send > 5){
    counting_send = 1;
   }

   if(fire_control != fire_control_old){
    sendfirecommand();
    fire_control_old = fire_control;
   }
  }
  //else{
   //sendstop();
   //println("stop");
  //}
}
// Display Framerate
fill(0,0,0, 99);
noStroke();
rect(0, 0, 70, 12);
fill(255,255,255,255);
text("fps", 5, 10);
text(frameRate, 20, 10);
```

```
    noFill();
    //println(Manual);
    //println(OUTML+":"+OUTMR);

}
// End Void draw //

public void stop(){
  m.stop();
  super.stop();
}
// Begin JoyStick Even Controll
void NaviKey(final float Nx,final float Ny){
  HX = Nx;
  HY = Ny;
}
void NaviKey_Release(final float Nx,final float Ny){
  HX = 0;
  HY = 0;
}
void button_1_press(){
  if(Safety == true){
  fire_control = 999;
  }
}
```

```
void button_1_release(){
  fire_control = 555;
}
void button_2_press(){
  Safety = true;
}
void button_2_release(){
  Safety = false;
  fire_control = 555;
}
void button_3_press(){
}
void button_3_release(){
  if(Scope == true)
  {Scope = false; Wide = true; digital_control = 233;}
  else{Scope = true; Wide = false; digital_control = 133;}
  if(Scope == true){Manual = true; Auto = false; Semi = false;}
}
void button_4_press(){
}
void button_4_release(){
  if(Swap == false)
  {Swap = true;
  }else
  {Swap = false;}
```

```
}
void button_5_press(){
}
void button_5_release(){
  if(Laser == false)
  {Laser = true; digital_control = 131;
  }else
  {Laser = false; digital_control = 231;}
}
void button_6_press(){
}
void button_6_release(){
  if(Flash == false)
  {Flash = true; digital_control = 130;
  }else
  {Flash = false; digital_control = 230;}
}
void button_7_press(){
}
void button_7_release(){
 int cnt = 0;
 if(Scope != true)
 {
   if(Auto == true){cnt = 1;}
   if(Semi == true){cnt = 2;}
```

```
   if(Manual == true){cnt =3;}
   if(cnt == 1){Auto = false; Semi = true; Manual = false;}
   if(cnt == 2){Semi = false; Manual = true; Auto = false;}
   if(cnt == 3){Manual = false; Auto = true; Semi = false;}
 }else{Manual = true; Auto = false; Semi = false;}
}
void button_8_press(){
}
void button_8_release(){
  if(Start == true)
  {
    Start = false;
    Stop = true;
  }
  else{
  Start = true;
  Stop = false;
  }
}
// Send Left Wheel Command with ASCII encode
void sendleft(){
  int A = int(OUTML);
  int B=(A/100)+'0';
  int M=(A%100);
  int C=(M/10)+'0';
```

```
  int D=M%10+'0';
  myPort.write('L');
  myPort.write(B);
  myPort.write(C);
  myPort.write(D);
}
// Send Right Wheel Command with ASCII encode
void sendright(){
  int A = int(OUTMR);
  int B=(A/100)+'0';
  int M=(A%100);
  int C=(M/10)+'0';
  int D=M%10+'0';
  myPort.write('R');
  myPort.write(B);
  myPort.write(C);
  myPort.write(D);
}
void sendpitch(){
  int A = int(CPITCH);
  int B=(A/100)+'0';
  int M=(A%100);
  int C=(M/10)+'0';
  int D=M%10+'0';
  myPort.write('P');
```

```
  myPort.write(B);
  myPort.write(C);
  myPort.write(D);
}
void sendyaw(){
  int A = int(CYAW);
  int B=(A/100)+'0';
  int M=(A%100);
  int C=(M/10)+'0';
  int D=M%10+'0';
  myPort.write('Y');
  myPort.write(B);
  myPort.write(C);
  myPort.write(D);
}
void senddigital(){
  int A = int(digital_control);
  int B=(A/100)+'0';
  int M=(A%100);
  int C=(M/10)+'0';
  int D=M%10+'0';
  myPort.write('C');
  myPort.write(B);
  myPort.write(C);
  myPort.write(D);
```

```
}
void sendstart(){
  myPort.write('S');
}
void sendstop(){
  myPort.write('T');
}
void sendfirecommand(){
  int A = int(fire_control);
  int B=(A/100)+'0';
  int M=(A%100);
  int C=(M/10)+'0';
  int D=M%10+'0';
  myPort.write('C');
  myPort.write(B);
  myPort.write(C);
  myPort.write(D);
}
void PID_C(){
  position_C = pixel_x;
  error_C = lock_center_setpoint - position_C;
  ut_C = int(KP_C*error_C);
  if(ut_C > ut_max_C){ ut_C = ut_max_C;}
  if(ut_C < ut_min_C){ ut_C = ut_min_C;}
}
```

```
void PID_R(){
  position_R = range_m;
  error_R = lock_range_setpoint - position_R;
  ut_R = int(KP_R*error_R);
  if(ut_R > ut_max_R){ ut_R = ut_max_R;}
  if(ut_R < ut_min_R){ ut_R = ut_min_R;}
}
```