```
//to submit

#include <LiquidCrystal.h>
#include <avr/io.h>
#include <Stepper.h>
LiquidCrystal lcd(8, 9, 4, 5, 6, 7);

// define some values used by the panel and buttons
int lcd_key = 0;
#define btnUP 0
#define btnDOWN 1
#define btnRIGHT 2
#define btnLEFT 3
#define btnSELECT 4
#define btnNONE 5

int currentMode = 10;
#define modeStart 10
#define modeDebug 20
#define modeIR 30
#define modeCM 40
#define modePM 50
#define subCM 60

volatile float distance;
unsigned long previousMillis = millis();
bool blink = true;
int selection = 1;
int cmselection = 1;
int pmselection = 1;
volatile int seconds = 0, minutes = 0;
```

```
//for CM
int steps[8] = {0b0001,
        0b0101,
        0b0100,
        0b0110,
        0b0010,
        0b1010,
        0b1000,
        0b1001};

int motor_delay = 100;
int current_motor_speed_selection = 2;
int rotation = 1;

//for PM
int set_angle = 0;
int remaining_angle = 0;
int step_number;
  const int stepsPerRevolution = 2048;
  Stepper myStepper(stepsPerRevolution, 0, 1, 2, 3);

void setup()
{
  lcd.begin(16, 2);
  DDRD |= (1 << PD0) | (1 << PD1) | (1 << PD3) | (1 << PD2);
}

void loop()
{
  lcd.setCursor(0, 1);
```

```
lcd_key = read_LCD_buttons();


switch (currentMode)
{
  case modeStart:
    startMode();
    break;
  case modeDebug:
    debugMode();
    break;
  case modeIR:
    irMode();
    break;
  case modeCM:
    cmMode();
    break;
  case modePM:
    pmMode();
    break;
  case subCM:
    subcmMode();
    break;
}
switch (current_motor_speed_selection)
{
  // modePark
  case 1:
    myStepper.setSpeed(1);
    break;
  //modeSlow
  case 2:
```

```
    myStepper.setSpeed(3);
     break;
    //modeFast
    case 3:
    myStepper.setSpeed(6);
      break;
  }
  remaining_angle = set_angle;
}

void startMode()
{
  lcd.setCursor(0, 0);
  minutes = millis() / 60000;
  seconds = (millis() % 60000) / 1000;

  lcd.print(minutes);
  lcd.print(":");
  lcd.print(seconds);

  lcd.setCursor(0, 1);
  lcd.print("14124200");
  if (read_LCD_buttons() == btnSELECT)
  {
    currentMode = modeDebug;
    delay(100);
  }
}

void debugMode()
{
```

```
lcd.setCursor(0, 0);
lcd.print("Debug Mode   ");
lcd.setCursor(0, 1);
if (millis() - previousMillis > 500)
{
  blink = !blink;
  previousMillis = millis();
}

switch (selection)
{
  case 1:
    if (blink) {
      lcd.print("IR CM PM        ");
    }
    else {
      lcd.print("__ CM PM        ");
    }
    break;

  case 2:
    if (blink)
    {
      lcd.print("IR CM PM        ");
    }
    else {
      lcd.print("IR __ PM        ");
    }
    break;

  case 3:
```

```
    if (blink)
    {
      lcd.print("IR CM PM         ");
    }
    else
    {
      lcd.print("IR CM __         ");
    }
    break;
}

if (read_LCD_buttons() == btnRIGHT && selection < 3)
{
  selection++;
  delay(100);
}

if (read_LCD_buttons() == btnLEFT && selection > 1)
{
  selection--;
  delay(100);
}

if (read_LCD_buttons() == btnSELECT && selection == 1)
{
  currentMode = modeIR;
  delay(100);
}

if (read_LCD_buttons() == btnSELECT && selection == 2)
{
```

```
   currentMode = modeCM;
   delay(100);
  }

  if (read_LCD_buttons() == btnSELECT && selection == 3)
  {
   lcd.clear();
   currentMode = modePM;
   delay(100);
  }
}

void irMode()
{
  lcd.setCursor(0, 0);
  lcd.print("IR Mode    ");
  lcd.setCursor(0, 1);

  int value1 = analogRead(A1);
  delay(20);
  int value2 = analogRead(A1);
  delay(20);
  int value3 = analogRead(A1);
  int x = (value1 + value2 + value3) / 3;
  distance = 40969 * pow(x, -1.22);

  if (distance <= 150 && distance >= 20)
  {
   lcd.print("dist.: ");
   lcd.print(distance);
   lcd.print("mm ");
```

```
  }

  if (read_LCD_buttons() == btnSELECT)
  {
    currentMode = modeDebug;
    delay(100);
  }
}

void cmMode()
{
  lcd.setCursor(0, 0);
  lcd.print("CM Mode        ");
  lcd.setCursor(0, 1);

  if (millis() - previousMillis > 500)
  {
    blink = !blink;
    previousMillis = millis();
  }

  switch (cmselection)
  {
    case 1:
      if (blink) {
        lcd.print("Start Exit        ");
      }
      else {
        lcd.print("_____ Exit        ");
      }
      break;
```

```
  case 2:
    if (blink)
    {
      lcd.print("Start Exit        ");
    }
    else {
      lcd.print("Start ____        ");
    }
    break;
}

if (read_LCD_buttons() == btnRIGHT && cmselection < 2)
{
  cmselection++;
  delay(100);
}

if (read_LCD_buttons() == btnLEFT && cmselection > 1)
{
  cmselection--;
  delay(100);
}

if (read_LCD_buttons() == btnSELECT && cmselection == 1)
{
  currentMode = subCM;
  delay(100);
}

if (read_LCD_buttons() == btnSELECT && cmselection == 2)
```

```
  {
    currentMode = modeDebug;
    delay(100);
  }
}

void subcmMode()
{
  lcd.setCursor(0, 1);
  if (rotation == 1)
  {
    lcd.print("CW  ");
  }
  else {
    lcd.print("CCW ");
  }
  lcd.print("Speed: ");
  lcd.print(current_motor_speed_selection);

if(rotation==1)
{
     myStepper.step(6);
}
else
{
     myStepper.step(-6);
}

  if (read_LCD_buttons() == btnUP)
  {
    if (current_motor_speed_selection < 3)
```

```
  {
    current_motor_speed_selection++;
  }
  else {
    delay(100);
  }
}

if (read_LCD_buttons() == btnDOWN)
{
  if (current_motor_speed_selection > 1)
  {
    current_motor_speed_selection--;
  }
  else {
    delay(100);
  }
}

if (read_LCD_buttons() == btnLEFT)
{
  rotation = 1;
  delay(100);
}

if (read_LCD_buttons() == btnRIGHT)
{
  rotation = 2;
  delay(100);
}
//display data on screen:
```

```
  if (read_LCD_buttons() == btnSELECT)
  {
    currentMode = modeCM;
    delay(100);
  }
}

void pmMode()
{
  lcd.setCursor(0, 0);
  lcd.print("PM Mode");

  lcd.setCursor(0, 1);
  lcd.print("Set:");
  lcd.setCursor(4, 1);
  lcd.print(set_angle);
  lcd.print("  ");
  lcd.setCursor(9, 1);
  lcd.print("Rem:");
  lcd.setCursor(13, 1);
  lcd.print(remaining_angle);
  lcd.print("   ");

  if (read_LCD_buttons() == btnRIGHT)
  {
    myStepper.setSpeed(5);
    remaining_angle = set_angle;
    for (int i = set_angle; i > 0; i--) {
      myStepper.step(6);
      remaining_angle--;
      delay(100);
```

```
    lcd.setCursor(4, 1);
    lcd.print(set_angle);
    lcd.setCursor(13, 1);
    lcd.print(remaining_angle);
    lcd.print("  ");
   }
  }

  if (read_LCD_buttons() == btnUP && set_angle < 360)
  {
   set_angle += 15;
   delay(100);
  }
  if (read_LCD_buttons() == btnDOWN && set_angle > 0)
  {
   set_angle -= 15;
   delay(100);
  }
  if (read_LCD_buttons() == btnLEFT)
  {
   set_angle = 0;
   delay(100);
  }
  if (read_LCD_buttons() == btnSELECT)
  {
   currentMode = modeDebug;
   delay(100);
  }
}

int read_LCD_buttons()
```

```
{
  int adc = analogRead(A0);
  delay(50);
  if (adc > 1000)
  {
    return btnNONE;
  }
  else if (adc < 50)
  {
    return btnRIGHT;
  }
  else if (adc < 300)
  {
    return btnUP;
  }
  else if (adc < 500)
  {
    return btnDOWN;
  }
  else if (adc < 700)
  {
    return btnLEFT;
  }
  else if (adc < 900)
  {
    return btnSELECT;
  }
  else   {
    return btnNONE;
  }
}
```