```
#include <Servo.h>
//variable that receives the command from the python code
int  incomingMessage=0;
//Sets the left side home position
int  leftHomePosition=115;
//Keeps track of the horizontal angle
int horizontalAngleTrack=leftHomePosition;

int downHomePosition=90;
//Keeps track of the vertical angle
int verticalAngleTrack=downHomePosition;
//Sets the increment of the horizontal angle every time the
//move command is called
int  horizontalIncrement=1;
//Sets the increment of the vertical angle every time the move
//command is called
int verticalIncrement=1;

//Declares a servo called horizontal_servo
Servo horizontalServo;
//Declares a servo called vertical_servo
Servo  verticalServo;

void setup()                 {
   //Sets up the sensor in the analog pin A5
  pinMode(A5, INPUT);
  //Initializes the serial port
  Serial.begin(9600);
  //Declares the pin that the servo controlling horizontal
  //angle is connected to
 horizontalServo.attach(7);
  //Declares the pin that the servo controlling vertical
  //angle is connected to
 verticalServo.attach(8);
}

// the loop routine runs over and over again forever:
void loop() {

  //Sends data only when it is received
  if (Serial.available()>0){
    //Waits for a message from the serial port (in this case,
    // from python code)
   incomingMessage=Serial.read();

    //Moves to the right
```

```
import numpy as np
from mpl_toolkits.mplot3d import Axes3D
import matplotlib.pyplot as plt
import serial
import time
from lidar_classdef import *

ser=serial.Serial('/dev/ttyACM0', 9600) #Defines the serial port to use

data=[] #List that stores the data from the arduino

horzBegin=115 #Sets the home horizontal angle
horzEnd=60 #Sets the end horizontal angle

vertBegin=64 #sets the home vertical angle
vertEnd=103 #Sets the end vertical angle

print "Waiting for arduino to be ready....." #Indicates that the programme is
waiting for the arduino to initialize

time.sleep(1) #Waits for the arduino to initialize

print "Programme beginning now" #Indicates that the python programme is running

#Moves the servo to the starting position

ser.write('5') #Sets the servo over to its home position on the left
time.sleep(0.5) #Waits half a second to ensure the arduino is ready for next command
ser.write('6') #Sets the servo to it's starting vertical angle

horzAngle=range(horzBegin, horzEnd, -1) #creates the horizontal range of angles that
the servo should sweep through
vertAngle=range(vertBegin,vertEnd,1) #Creates the vertical range of angles that the
servo should sweep through

for eachVerticalAngle in vertAngle:


        for eachHorizontalAngle in horzAngle:

                ser.write('7') #Tells the arduino to send back a distance reading

                distance_response=ser.readline() #Receives the distance reading from
the arduino
                cleanReading=distance_response[0:-2] #Removes the last 2 characters
("\r\n") from the arduino output
                distance=float(cleanReading) #Type-casts it as an float so that it
can be plotted

                ser.write('8') #Tells the arduino to send back the current vertical
and horizontal angles

                angle_response=ser.readline() #Receives the vertical and horizontal
angles from the arduino as a string

                cleanReading=angle_response[0:-2] #Removes the last 2 characters
("\r\n") from the arduino output
                seperator_index=cleanReading.index(',') #Figures out at what index
the comma appears (which is how we delineate vertical and horizontal angle)
                vertical_angle=cleanReading[0:seperator_index] #Retreives the
vertical angle from the string that is received from the arduino
                vertical_angle=int(vertical_angle) #Converts the angle from string
to integer
```

```
                horizontal_angle=cleanReading[(seperator_index+1): ] #Retreives the
horizontal angle from the string received from the arduino
                horizontal_angle=int(horizontal_angle) #Converts the angle from a
string to a integer

                processed_full_data_holder=lidar(horizontal_angle, vertical_angle,
distance) #Passes the distance reading and angles into a lidar object (defined
seperately)

                data.append(processed_full_data_holder) #Appends the new lidar
object to the data list

                ser.write('1') #Moves the servo one degree to the right after taking
a reading

        ser.write('3') #At the end of the horizontal sweep, increase the vertical
angle by 1 degree
        time.sleep(0.5) #Waits to allow the arduino to execute the command and be
ready for another

        ser.write('5') #Resets the arduino back to the left to perform a full
horizontal sweep
        time.sleep(0.5) #Waits for the arduino to execute the command and be ready
for another

#def randrange(n, vmin, vmax):
#    return (vmax-vmin)*np.random.rand(n) + vmin

fig = plt.figure() #Creates a new pyplot figure
ax = fig.add_subplot(111, projection='3d') #adds a 3d plot axis to it

xs = [] #Creates a new list for x axis values
ys = [] #Creates a new list for y axis values
zs = [] #Creates a new list for z axis values

for eachPoint in data:
        xs.append(eachPoint.x_pos) #Adds each x axis value to the x axis data list
        ys.append(eachPoint.y_pos) #Adds each y axis value to the y axis data list
        zs.append(eachPoint.z_pos) #Adds each z axis value to the z axis data list

ax.scatter(xs,ys,zs, c ='r', marker = 'o') #Creates a scatter plot of all data

ax.set_xlabel('X Distance in centimeters') #X axis label
ax.set_ylabel('Y Distance in centimeters') #Y axis label
ax.set_zlabel('Z Distance in centimeters') #Z axis label

plt.show() #show the pyplot figure
```

```
import math #Imports math to do trig calculations

class lidar: #Definition of a new class called lidar
        def __init__(self, input_horz_ang=0, input_vert_ang=0, radius=0):
                horz_ang=math.radians(input_horz_ang) #converts the horizontal angle
from degrees to radians
                vert_ang=math.radians(input_vert_ang) #converts the vertical angle
from degrees to radians

                self.z_pos = -radius*math.cos(vert_ang) #Calculates the length of
the position vector projected on the z-axis
                self.y_pos = radius*math.sin(vert_ang)*math.sin(horz_ang)
#Calculates the length of the position vector projected onto the y axis
                self.x_pos = radius*math.sin(vert_ang)*math.cos(horz_ang)
#Calculates the length of the position vector projected onto the x axis

        def display_data(self): #Instance method to display the data encapsulated in
the lidar object

                print "X Position = ", self.x_pos
                print "Y Position = ", self.y_pos
                print "Z Position = ", self.z_pos
```

```
#include <Servo.h>
//variable that receives the command from the python code
int  incomingMessage=0;
//Sets the left side home position
int  leftHomePosition=115;
//Keeps track of the horizontal angle
int horizontalAngleTrack=leftHomePosition;

int downHomePosition=90;
//Keeps track of the vertical angle
int verticalAngleTrack=downHomePosition;
//Sets the increment of the horizontal angle every time the
//move command is called
int  horizontalIncrement=1;
//Sets the increment of the vertical angle every time the move
//command is called
int verticalIncrement=1;

//Declares a servo called horizontal_servo
Servo horizontalServo;
//Declares a servo called vertical_servo
Servo  verticalServo;

void setup()                 {
   //Sets up the sensor in the analog pin A5
  pinMode(A5, INPUT);
  //Initializes the serial port
  Serial.begin(9600);
  //Declares the pin that the servo controlling horizontal
  //angle is connected to
 horizontalServo.attach(7);
  //Declares the pin that the servo controlling vertical
  //angle is connected to
 verticalServo.attach(8);
}

// the loop routine runs over and over again forever:
void loop() {

  //Sends data only when it is received
  if (Serial.available()>0){
    //Waits for a message from the serial port (in this case,
    // from python code)
   incomingMessage=Serial.read();

    //Moves to the right
```

```
if (incomingMessage=='1'){
  //Decrements the horizontal angle by the specified
  //increment to
  //move it to the right
 horizontalAngleTrack=horizontalAngleTrack-horizontalIncrement;
  //writes the new angle to the servo to move it there
  horizontalServo.write(horizontalAngleTrack);
}

//Moves to the left
if (incomingMessage=='2'){
  //Increases the horizontal angle by the specified
  //increment to move
  //it to the left
horizontalAngleTrack=horizontalAngleTrack+horizontalIncrement;
 //writes the new angle to the servo to move it there
  horizontalServo.write(horizontalAngleTrack);
}

//Moves the servo up
if (incomingMessage=='3'){
  //Increases the vertical angle by the specified
  //increment to move it up
 verticalAngleTrack=verticalAngleTrack+verticalIncrement;
 //Writes that angle to the servo to move it there
  verticalServo.write(verticalAngleTrack);
}

//Moves the servo down
if (incomingMessage=='4'){
  //Decreases the vertical angle by the specified increment
  //to move it down
  verticalAngleTrack=verticalAngleTrack-verticalIncrement;
  //Writes that angle to the servo to move it there
  verticalServo.write(verticalAngleTrack);
}

//Moves sensor all the way left
if (incomingMessage=='5'){
  //Sets the horizontal angle to the left_home_position
 horizontalAngleTrack=leftHomePosition;
 //writes that angle to the servo to move it there
  horizontalServo.write(horizontalAngleTrack);
}

//Moves sensor to the lowest possible vertical angle setting
```

```
    if (incomingMessage=='6'){
      //Sets the vertical angle to the down_home_position
      verticalAngleTrack=downHomePosition;
      //Writes that angle to the servo to move it there
      verticalServo.write(verticalAngleTrack);
    }

    //Takes a reading from the sensor
    if (incomingMessage=='7'){
      //Takes a reading from the sensor connected to port A5
      int reading =analogRead(A5);
      //Uses the model we derived to calculate distance from
      //the sensor reading obtained
      float distance=25732.834527*pow(reading,-1.1314581);
      //Prints that distance to the serial port for the python
      //code to receive
      Serial.println(distance);
    }

    //Sends back the current horizontal and vertical angles
    if (incomingMessage=='8'){
      //Packs the current vertical and horizontal angle into a
      //concatenated string to print to serial port
      String anglePacked=String(verticalAngleTrack)+','+String(horizontalAngleT
      Serial.println(anglePacked);//Prints that concatenated
      //string to the serial port for python to receive
    }
}
}
```