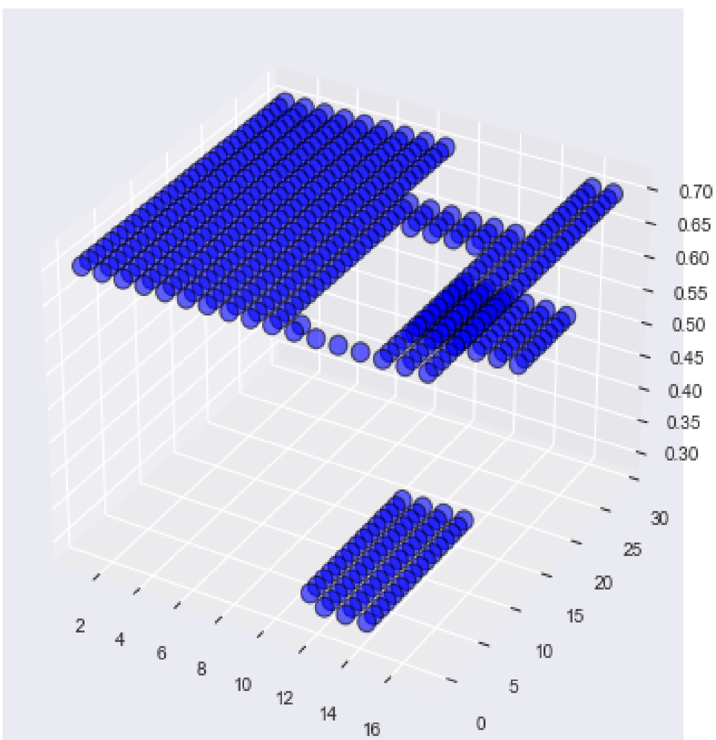
Introduction

SONAR, or Sound Navigation and Ranging, is using sound to figure out distance from a certain point, by using echolocation. SONAR is really helpful in learning about the ocean floor, because we can't really go down there and map it. SONAR allows you to drive a boat around, and create a 3D map of the ocean floor. SONAR helps a lot, but the fatal flaw for it is anything that absorbs sound.

Methods

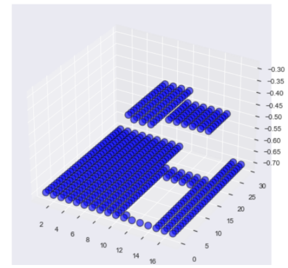
I was given the challenge of figuring out what was under my desk, using only SONAR. This is a hard problem, but Akea (my partner) and I had a strategy. We taped rows every 6 cm on the surface. Then, row by row, we sent the SONAR module under the table. We measured the distance to the floor (0.7 meters), and then we checked the whole row for abnormalities.



This way you don't have to check every point, because you will know when a different number then 0.7 shows up. The main flaw, however, is that you miss very subtle height differences. If you had a thing less than a cm, it would be very hard to spot with this method. The SONAR module that we used was the Vernier SONAR Go!Motion, and the software that we used to measure was Vernier Graphical Analysis. On the left side, you can see two bumps, with straight lines, meaning that there are two boxes there. Throughout the rest of the graph there is nothing else, implying that it is entirely floor.

Analysis

For this project, I decided that I wanted to code a 3D graph reader by myself. I used python, with matplotlib and pandas. This code takes a x,y,z matrix, and converts them into points. Then, it puts them into a scatter plot. This took a long time to figure out, because I did not know how to use Pandas. The first thing that my code does is it takes my csv, and puts it into a data frame, which is essentially a 2D list, with x and y coordinates for each point. Then, it takes a column, and adds all of the numbers in that column to the y list, and puts the x and z coordinates in the same spot relative to it. Once it runs through the whole matrix, it sends the coordinates to a matplotlib scatter plot, which loops through the lists and adds points at the corresponding points.



```
import pandas as pd

df=pd.read_csv('data.csv')

x=[]
y=[]
z=[]
loop=0
cols=['2','3','4','5','6','7','8','9','10','11','12','13','14','15','16','17','18']
#print(df['0'].values[0])
for n in cols:
    loop+=1
    for i in range(len(df[n])):
        x.append(loop)
        y.append(i)
        z.append(df[n].values[i])

import matplotlib.pyplot as plt
plt.style.use('seaborn')

ax = plt.axes(projection ='3d')
ax.scatter(x, y, z, s=100, alpha=0.6, c='blue', edgecolor='black', linewidth=1)
plt.tight_layout()
plt.show()
```

Something that kind of surprised me, but makes sense, is that the objects that were closer to the sonar device are further away, because when you measure the distance, it is coming from a bird's eye view. That means that the lower numbers are further away, so they appear lower on the graph. The solution to this is multiplying the numbers by -1, which gives us a graph like this.

Conclusion

The way to read the first graph is to take the points that are closest, and say that those are the floor, then take the furthest ones, and say that those are objects above the floor. Basically flip the graph upside down. What we found from our measurements were two boxes, one longer and taller than the other one. When we looked under, that is what we saw. I think that you could improve my method by measuring specific points, but given the time constraint, I think that this was our best option. SONAR is used all over the world, to help measure distance. One of the most common examples of SONAR is measuring the ocean floor. They put a bunch of SONAR devices on a boat, and then they can measure the floor of the ocean, and make a 3d map, using programming similar to mine.