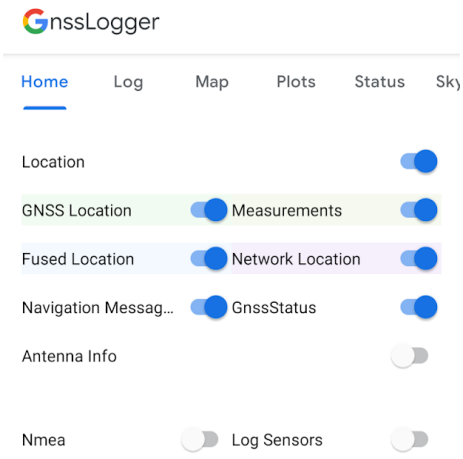
Homework 2 (Part A and B)

Submission Date: Nov 29th, 2024

Part A: Android-based GNSS Measurements (MAX SCORE: 100)

GNSS (Global Navigation Satellite System) today includes a variety of satellite constellations, including the GPS (American, global), GLONASS (Russian, global), BeiDou (Chinese, global), Galileo (European, global), NavIC (Indian, regional) and QZSS (Japanese, regional). Depending on your smartphone's hardware capabilities, you can tap into signals from such constellations and update your location. Google provides an array of open-source tools for performing GNSS-related measurements on commodity Android smartphones and analyzing such datasets.



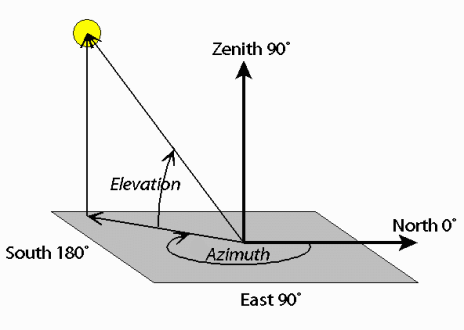
Download and install this app (GnssLogger App – Apps on Google Play) to get started. Ensure the location permissions are provided to this app, and you have turned on your phone's location service (e.g., GPS). Turn on the "location" switch in the GNSSLogger app, as shown in the figure. Go to the next tab, "Log", and check whether you receive the GNSS specific diagnostic messages that include your location coordinates (latitude and longitude fix). You can save the log data to a file and export this to the SD card or share it to your drive/email for offline use. Try playing with this app and make sure you are receiving the logs correctly. Cross-check the location coordinates reported by the app in Google maps.

Concentrate on the lines starting with keywords with "Fix" and "Status" (see GNSS Status API). "Fix" related lines will provide you with the location coordinates of your phone. Only consider lines with "Fix, GPS" (raw GPS) and not "Fix, FLP" (fused location, uses IMU for correction).

- ❏ Sample fix line (we are interested only in the red highlighted part):
  Fix,Provider,**LatitudeDegrees,LongitudeDegrees**,AltitudeMeters,SpeedMps,AccuracyMeters,BearingDegrees,UnixTimeMillis,SpeedAccuracyMps,BearingAccuracyDegrees,elapsedRealtimeNanos
  Example: Fix,GPS,**12.997769,80.240794**,-100.513824,0.000000,5.360000,0.000000,1616658843000,0.49396357,0.0,1742433115036168

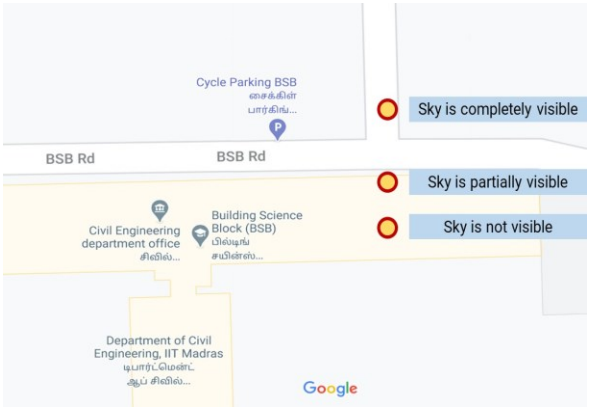
- ❏ Sample status line (we are interested only in the red highlighted part):
  Status,UnixTimeMillis,SignalCount,SignalIndex,**ConstellationType,Svid,**CarrierFrequencyHz,**Cn0DbHz,AzimuthDegrees,ElevationDegrees,UsedInFix**,HasAlmanacData,HasEphemerisData,BasebandCn0DbHz
  Example: Status,1616658843000,40,38,**6,1**,1575420032,**15.00,355.00,37.00**,**1**,1,0,

- ❑ **ConstellationType:** Only use GPS (for now, please ignore data from GLONASS, BeiDou etc., even if your phone supports it). Flag for GPS = **1**, (see this)
- ❑ **Svid:** Satellite ID
- ❑ **Cn0DbHz:** Signal Strength or SNR of the signal received from the particular satellite.
- ❑ **AzimuthDegrees (0 - 360)** & **Elevation Degrees (0 - 90)** of the satellite (Wiki)



- ❑ **UsedInFix:** Indicates whether this particular satellite was used for the latest fix for multilateration (remember, you need at least 4 of them?).

Tasks



Collect such measurement logs at three distinct locations as indicated in the picture on the left. (i) The smartphone is under the open sky a few meters from the entrance/door of your building, (ii) The smartphone is at the entrance (or some window) of your building, the sky is partially visible, and (iii) The smartphone is completely inside the building, very limited portion of the sky is visible maybe through the door/window. At each of these locations, collect GNSSLogger data for about 5 - 10 minutes. (perform the experiments on a sunny day with a clear sky)

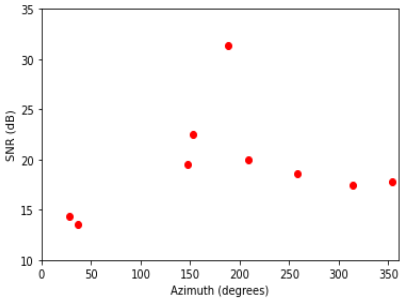
1. Attach photos of the three locations with the phone, where data is collected, particularly the exposure to the sky. **[10 points]**

2. Calculate the mean location coordinates for all three locations. Assume those three locations as the “groundtruth”. Compute the error for each location fix sample, which is the distance between the location sample and the groundtruth location. You must use haversine distance (Wiki). Plot the CDF (cumulative distribution function) of these errors for the three locations. Comment on the variance of the three distributions. **[15 points]**

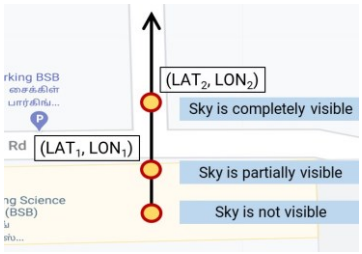
3. Check the “status” entries prior to receiving a “fix”. **[45 points]**
    a. How many satellites were used for the fix (check the UsedInFix entry for the satellites) for the three locations? (state the median # of satellites)

b. How does the distance error for a fix correlate to the number of satellites used for that fix? Show scatter plots for the three locations. Summarize your observations.
c. How does the distance error for a fix correlate to the average SNR (Cn0DbHz entry) of the satellites that were used in the fix? Show scatter plots for the three locations. Summarize your observations.

4. Only consider the “status” instances where a particular satellite was used for a location fix. **[20 points]**
   a. Show a table for the “Svid”, “Average Azimuth”, “Average Elevation”, “Average SNR” (each row for a specific satellite).
   b. Plot “Average Azimuth” (0 - 360 degrees) with “Average SNR” (see Qs 5)
   c. Plot “Average Elevation” (0, 90 degrees) with “Average SNR”



5. See plot 4(b). Two example plots are also shown on the right for your reference. Either you can plot in the cartesian space or in the angular (R, theta) space. Magnitude of R can signify your SNR. To make it more interesting, you can superimpose the angular plot on a satellite image, say from Google maps (of course, you need to rotate the image to align the azimuths). This should show the effect of buildings, trees, obstacles on the SNR. Does your plot (particularly, for location 2, where the phone is kept on the window sill or door) peak for a certain azimuth? This azimuth should roughly match with the direction of the clear sky from that location. **[10 points]**

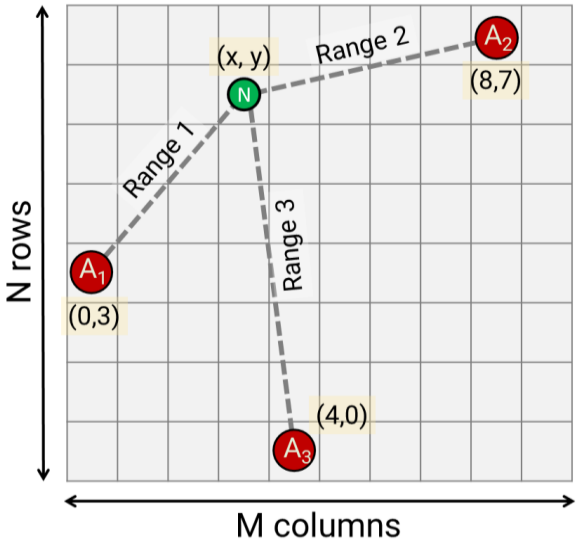


How to calculate the azimuth corresponding to the direction of the window or door? Take the two points (Lat1, Lon1) and (Lat2, Lon2) and use this website, https://www.fcc.gov/media/radio/distance-and-azimuths or any other online azimuth calculator. Basically, the azimuth gives you the angle the black line makes with 0 degrees north.

Use the Jupyter Notebook Provided

Part B: Localization of IoT Devices (MAX SCORE: 60)

Consider a 2D-grid of MxN cells where we want to locate an IoT device (marked as N) using multilateration. The lower-left corner cell can be treated as the origin (0,0) while the location of the top-right cell is (M, N). Assume that all anchors (A_1, A_2, and A_3) are kept at known locations.



Task 1: Dataset Generation. Simulate grid of size 100x100. Consider 3 anchors. **[10 points]**

a. Randomly generate 100 unique anchor location sets and for each randomly generated anchor location set, generate 50 random unique node locations (*programming hint: use a `Set` ADT while generating the data*). All such location points should be within the simulated grid's bounding box. Write this data to a file, *<true_locations.csv>*

$[(x_1,y_1),(x_2,y_2),(x_3,y_3)]^1$, (NX_1, NY_1), .. (NX_{50}, NY_{50})
$[(x_1,y_1),(x_2,y_2),(x_3,y_3)]^2$, (NX_1, NY_1), .. (NX_{50}, NY_{50})

... ...

$[(x_1,y_1),(x_2,y_2),(x_3,y_3)]^{100}$, (NX_1, NY_1), .. (NX_{50}, NY_{50})

b. Generate the range values for each node location to the respective anchors and write them to a file *<pure_ranges.csv>*

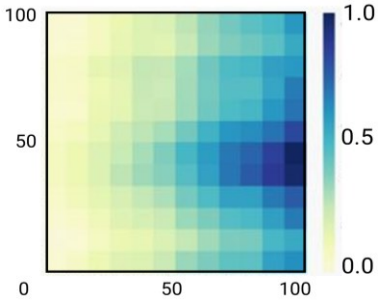
$[(x_1,y_1),(x_2,y_2),(x_3,y_3)]^1$, $(R_1,R_2,R_3)^1$, .. $(R_1,R_2,R_3)^{50}$

... ...

$[(x_1,y_1),(x_2,y_2),(x_3,y_3)]^{100}$, $(R_1,R_2,R_3)^1$, .. $(R_1,R_2,R_3)^{50}$

c. Now generate noisy range values for each node location by adding Gaussian noise to each pure range value as in *<pure_ranges.csv>*. $R_{noisy} = R_{pure} + N(\mu,\sigma)$. Generate three datasets, for μ = 0.5, 1 and 2. Assume σ = 0.1 for all cases. Write these data into three files *<noisy_ranges_05.csv>, <noisy_ranges_1.csv>* and *<noisy_ranges_2.csv>*

Task 2: Range Equations. Generate two random numbers, A in [1, 100] and B in [1, 50]. Choose the A^{th} line and the B^{th} node location from the files in (a), (b), and (c). **[20 + 10 points]**



a. Form the cost function (use *root mean square error*) using the three range equations for that particular node and anchor locations. Evaluate the "cost value" for all the 100x100 cells for *<pure_ranges.csv>, <noisy_ranges_05.csv>, <noisy_ranges_1.csv>* and *<noisy_ranges_2.csv>*. While evaluating your cost function put (X = i and Y = j) where i and j are integers in [0,99].

Visualize the normalized cost values (scale: 0 to 1) in the form of a heatmap. Plot the 4 heatmaps. Don't interpolate your heatmaps.

b. Summarize your observations. Did you realize the fact that we are trying to use a brute-force approach to scan all possible cells for the solution? Is the cell with the global minima near (or, how far from) the actual node location? (Cost value = 0 for non-noisy case) Are there local minima present in the noisy cases?

Task 3: Trilateration. Now use an optimizer to solve for the node location for all 100x50 node locations (pure + 3 noisy versions). If you are using python, use `lmfit` (https://lmfit.github.io/lmfit-py/). Round off the output of the solver to the nearest integer value, say if the solver outputs (50.69, 45.23) - round it off to (51, 45) for the estimated location cell. Also make sure the solver knows the limits of the solution (i.e., the bounding box). Write the solvers output into 4 files, *<pure_locs.csv>, <noisy_locs_05.csv>, <noisy_locs_1.csv>* and *<noisy_locs_2.csv>.* Compare the solved locations with the entries in *<true_locations.csv>* and compute the localization errors (euclidean distance between true and estimated location). **[20 + 10 points]**

a. Plot 4 CDFs (each for the 5000 error values) on the same graph, to compare the 4 cases.

For the 4 cases state the: (i) median error, (ii) 75th percentile error, and (iii) 95th percentile error.

Deliverables | Submit by: 29th November, 11:59PM (Friday)

1. A PDF report (for part B) summarizing your activities, figures, plots etc.
2. Source code and log files (for part A + B), Jupyter notebook for part A, including all references, web repositories consulted.
3. A README file documenting the contents and methodology to run the code. Also mention the names of all students with whom you have discussed your homework with.
4. Upload the content as a zip file name: **CS6650WEB_HW2_<rollnumber>.zip**