Android Wi-Fi RTT Locator (WRL) User Guide

Date: 28th June, 2023, Application V1.9, User Guide Rev 1.9

Introduction

The Android Wi-Fi RTT Locator (WRL) application is designed to demonstrate precise indoor location with an accuracy of 1-2 meters. It uses the IEEE 802.11mc protocol for measuring the time-of-flight of Wi-Fi packets between a phone and an Access Point (AP). As radio waves travel at the speed of light (*c*), the measured time is converted to a distance by multiplying by *c*. After estimating the distance to three or more Access Points at known positions, the phone can calculate its own position using a process called multilateration. WRL is built around the Android API for the 802.11mc protocol also known as Wi-Fi Round Trip Time (RTT), and is described in the developer pages. A more complete explanation of the Android Wi-Fi RTT API, how you can use it to calculate an accurate indoor position, and examples of its use were shown at Google IO'18 and captured in this video. You can see an overview of the WRL application launch, configuration, and positioning in this short video clip.

Application Features

WRL enables a user to automatically position a Wi-Fi RTT capable Android smartphone within an area supported by IEEE 802.11mc capable Access Points, and display its position on a floor-plan. It does this by requesting a configuration file (.csv) and a floor plan (.png) file. The floor plan is overlaid onto Google maps shown on the application's primary display; see Figure 1a. The floor plan's geographic position is described in the configuration file, along with the positions of the Access Points. You can provide predefined ground truth points to measure the accuracy of the estimated position, along with a set of parameter values that are used to control the app's behavior. The configuration file is described in detail in Configuration File Format. In addition, while the phone is positioning itself on the floor plan, this data is simultaneously logged to a local file that the user can save for future analysis, after positioning has been stopped. Both the floor plan view and the data log file are private to the user, and they can decide if, when, and where it is stored, to respect user privacy.

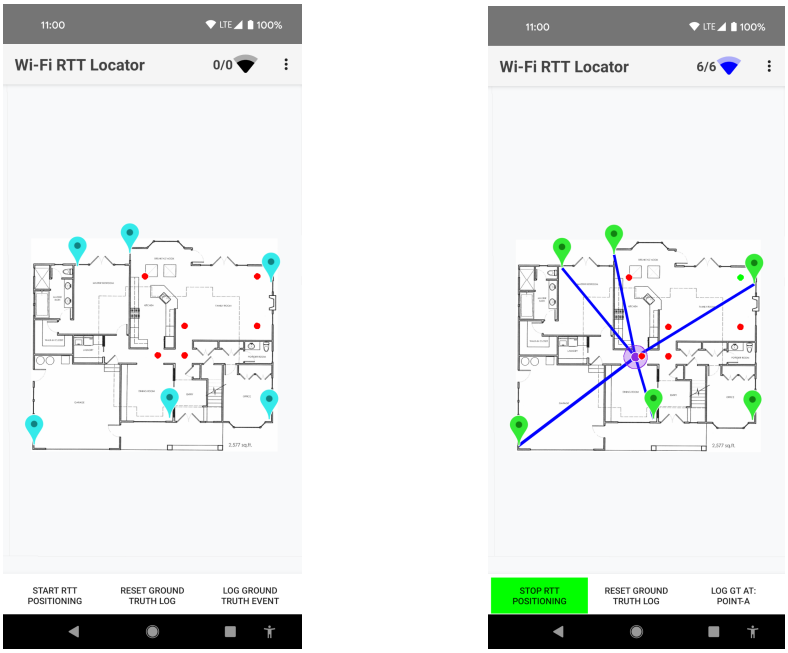

Figure 1: a) The WRL application once configured b) WRL view while positioning

When positioning is active, the display will show the RTT position as a purple dot with a translucent circle around it indicating the estimated standard error; see Figure 1b. Access points are shown with a place marker. These markers are light blue to start with, but if discovered through a Wi-Fi scan, will become dark blue. When the phone selects an AP to range to, the color will turn green, and a blue ranging-line will be drawn from your position to the AP. If the ranging protocol has a fault during this process, the AP color will turn red, until the next successful ranging event, or dark blue again if it is no longer selected for Wi-Fi ranging.

Ground Truth (GT) points are used to evaluate the performance of the WRL application during positioning, and compare the user's estimated path with the actual path taken. GT markers are

shown as small red circles at the time the configuration file is loaded. When **Start RTT Positioning** is tapped, the first GT point (circle) turns green. As you walk to that point and tap the **Log GT** button, it turns grey and the next one on the GT path turns green. The data log records these positions and timestamps for later processing. After the last GT point has been recorded, you can **Stop RTT positioning** and will then be prompted to save the data log file to a location of your choice.

New in v1.2

In v1.2, we added the ability to move between floors. The app uses the phone's barometer (when available), the number of RTT ranging errors, and the results of a recent Wi-Fi Scan, to determine if the user has changed floor. If the user has switched floors, the app will start scanning the APs on the new floor to determine if the user is on a known floor – i.e. a floor specified in the CSV file (details listed below). If the floor is found, the new floorplan will be displayed automatically, along with a toast message to alert the user. At that point, RTT ranging resumes and will correctly show the user's position on the new floorplan. Some additional considerations are:

- For the map switching to work properly, the user must provide multiple overlays during the overlay file selection process, and the overlay file names must match the strings used to describe the floor in the config file (see the CSV configuration details below).

- The app will only show the APs and GT points associated with the overlay file currently being shown. To see APs and/or GT points located on other overlays, level change support must be enabled and the user needs to go to the level/location specified by that overlay file. The app will detect the change and switch overlay files. At that time, it will display only the APs and GT points associated with the new overlay.

- When the app decides to change levels, it will start ranging to the nearest APs regardless of which floor they are on. This will be observed on the UI as ranging-lines that terminate without showing an AP icon – this indicates the AP is on another floor.

New in V1.8

In v.1.8, we added optimizations to make the navigation experience smoother and more accurate. These options can be enabled in the configuration file.

- **Wi-Fi scan optimization**: This can be turned on if the AP channel frequencies are known in advance. The frequencies are provided as a list of integers contained in a parameter, or specified in each AP record. Once configured, the app will only perform

one Wi-Fi scan each time positioning is started, eliminating Wi-Fi Scans while navigating.

- **Automatic calibration error removal**: The app can now estimate the calibration error automatically, without the need for manual correction. It uses a bounded running median of estimated errors to reduce the effect of outliers. The size of the bounding window can be set in a parameter.

- **Uses Location Configuration Information (LCI) when available**: When 802.11mc capable Access Points are configured to provide their own Latitude and Longitude, both entries in the configuration file can be set to 0.0 degs, and the phone will automatically use the locations provided by the Access Points delivered using the 802.11mc protocol.

Getting Started

To set up an accurate indoor location system, you need a scaled floor plan, along with precise information about the positions of the Wi-Fi Access Point deployment. It is important to perform the following steps as accurately as possible, as any errors will be reflected in the final accuracy achieved by the WRL application.

Before you install and use the WRL application, you will need the following:

- **An Android Smartphone that supports Wi-Fi RTT (802.11mc)**: Any of the Pixel 4 to Pixel 6 phone models [1] have this capability, and should be running an Android Q (10) or a later OS.

- **A set of access points with IEEE 802.11mc capability**: Google Wi-Fi 2016 or 2020 [2] are recommended. Note: Once a Google Wi-Fi AP is configured using the corresponding Google app, it will serve as a simple 802.11mc transponder, even with the internet disconnected. Only mains power is then needed for deployment, which may be useful for some locations with no Ethernet. Three APs are the minimum required to support an indoor positioning system. A more-interesting deployment, in a larger building, would perhaps use 6 or more APs; Google Wi-Fi can be purchased cost-effectively in 3-packs. If a network with more than 6 APs is planned, it's recommended that you do not set it up as a mesh. APs should be placed about 50 feet (15m) apart in roughly a grid pattern extending to the outer walls of the positioning region.

- **A detailed floor plan (PNG format) of the building hosting your positioning system**: This is used as an overlay on Google Maps and provides the context for the RTT estimated position. An example house floor plan is provided **here**. You will need to determine the latitude and longitude of the SW and NE corner of the bounding box containing your floor plan overlay, with the top-side facing North when positioned on Google Maps. This process is best carried out using a laptop computer running Google Earth Pro. Use these overlay instructions to position and scale your floor plan

appropriately. You can determine the position of the SW corner by using the push-pin tool positioned over the point. Google Earth displays the latitude and longitude of a point (using six decimal places), which you can record in a spreadsheet for documentation. Repeat for the NE Corner.

- **The position in latitude and longitude (lat/lng) of each AP in the system**: Use the Google Earth overlay you created for the floor plan earlier in this document to identify and determine the lat/lng of each AP using the push-pin marker tool. Use all six decimal places provided by Google Earth to preserve the position accuracy. Record this in your documentation spreadsheet, along with a unique name for each AP and its wireless Media Access Control/Basic Service Set Identifier (MAC/BSSID) address. Next, physically install your APs in the positions you identified on the floor plan, as high up as possible, and at a similar height to each other. Make a note of this height in meters.

- **Define a set of Ground Truth (GT) points**: Repeat the earlier process, recording each lat/lng position with an identifying label. Labels should be sequential, such as P1, P2, P3 … or A, B, C… The positions of the GT points should be chosen so they are easily identifiable in the building, and make it possible to walk in a straight line between each pair. They should not be too far apart, so you can walk between them at approximately constant speed. This is recommended so that for each estimated RTT position along the path, there will be a corresponding interpolated GT position.

Before installing the application, it's recommended that you turn off Wi-Fi scan throttling in the Android settings menu. This can be found under **Developer Options > Networking > Wi-Fi scan throttling**).

Install the Application

Now you can install the application from the Google Play Store, and copy the example configuration file to your computer. Make a copy of this file, and rename it something like “my-house.csv”, and then customize its content for your system.

The configuration file is a text file using a comma separated variable (csv) format. Blank lines are allowed, and comments can be added to help organize the information by using a ‘#’ character at the beginning of a line. The record types used in the file are described in the Configuration File Format; also see the overview in Table 1.

Record Type	Meaning	Record Type	Meaning
BD	Building	GT	Ground Truth

AP	Access Point	PARAM	Parameter

Table 1: The principal record types used in the configuration file (.csv)

You can use the example file as a template and replace the BD, AP, and GT records to match your own system. The PARAM values provided in the example can be left as they are for now, as they are general enough for an initial deployment.

When you start the WRL application, give it the requested permissions. You will then be prompted to load a configuration file. The easiest way to proceed is to store the configuration file (.csv) in a folder on your Google Drive and navigate to it. You can also place your floor plan (.png) file in the same folder, as you will be prompted to load it next.

If there were any formatting errors in the configuration file, you will get a warning pop-up message. Correct these issues before proceeding; compare your file with the provided template and the information that follows to confirm the correct format.

Next, tap the **Start RTT positioning** button. On tapping the button, a Wi-Fi scan is initiated. The Wi-Fi icon at the top right will turn from black to green during scanning, and back to black afterwards. When complete, if there are three or more 802.11mc APs discovered, the floor plan will show your position with a purple dot. It will also draw blue-lines from the estimated position to the APs that are being used for ranging, to help you understand how it's working. The Wi-Fi scans will continue at a rate determined in the configuration file, defaulting to every 15 seconds. Once all APs defined in the configuration file are discovered, the Wi-Fi Scan icon turns blue (the fraction discovered is also shown) and scanning stops. If you are setting up a large positioning system (greater than 20 APs) walk around the area until all APs have been discovered so that positioning can proceed without any scan interruptions. The Wi-Fi Scan-icon will turn red when the scan cannot find any of the APs listed in the Config file.

When you **Stop RTT Positioning**, the application prompts you to save the positioning data log file for later analysis. If you decline, this data will be deleted the next time you start positioning.

Options and Settings Menu

WRL has an options menu in the top right corner, indicated by three vertical dots. This allows you to modify the configuration, either clearing the most-recently-loaded configuration file or loading a new one. After requesting to load a new configuration, you will be prompted to select a new configuration file (.csv) and floor plan (.png), similar to the process at startup. This allows you to run the app in a variety of buildings, and select the corresponding files in each case.

Other menu options allow you to see the application build-date, version, licensing information, and parameter values successfully loaded from CSV files. There is also a settings option that

provides a sub-menu to control features of the application that can be set easily from the user interface, rather than in the configuration file at startup:

- **Set Scanning Period (secs)**: This is the Wi-Fi scan period. A scan takes about two seconds (depending on the phone type), and RTT positioning cannot occur during this time. You can adjust this time (overriding the setting in the configuration file) to suit your experiments. Note: Once all APs have been discovered, Wi-Fi scans are disabled for the application. A potential strategy is to set a short Wi-Fi scan period, such as every 10-15 seconds, so that all the APs are discovered as soon as possible.

- **Set Ranging Period (secs)**: This is the time between calculating Wi-Fi RTT positions. For most smartphones, the fastest rate of positioning is about 3Hz, so the shortest period would be 0.33 seconds. Setting it to a smaller value will likely have no effect, but you can set much longer ranging periods for some types of experiments.

- **Android Fused Location Provider (FLP) Location (checkbox)**: If you mark this checkbox, the app shows the FLP position, with the traditional blue dot and translucent blue error-circle, on the same floor plan as the RTT position purple dot, and the error circle. This allows you to compare raw RTT position accuracy against FLP RSSI positioning accuracy, to see the improvement.

- **Camera Tracking (checkbox)**: If you mark this checkbox, the map/floor-plan automatically pans as you move, to keep your position in view as you walk through the building.

- **Image Overlay (checkbox)**: When this checkbox is marked (the default), your overlay floor-plan will show up on the app's Google Maps view. When unset, the floor-plan is hidden. This can be useful to temporarily see features hidden by the overlay that are on the underlying map.

Performing a Positioning Experiment with Ground Truth

Once RTT positioning is underway, you can see the estimated position change in real-time as you move around the building. If you want to characterize the accuracy of the estimated path, you can use the Ground Truth (GT) logging feature.

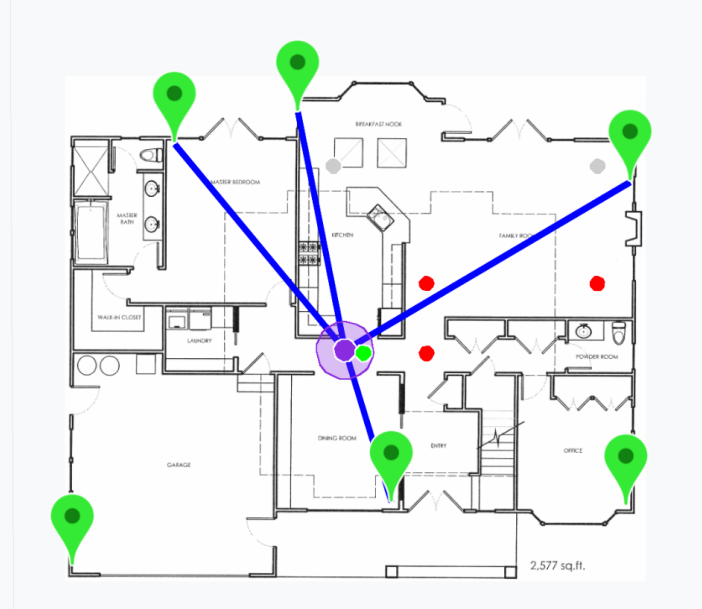

Figure 2: Two Ground Truth points already selected (grey), and the third highlighted green.

To begin an experiment, you must have predefined a set of GT points in the configuration file. While positioning, you may then walk between the Ground Truth (GT) points in order, and at constant speed, tapping the **LOG GT <label>** button at each point, as shown in Figure 2.

Note: Identify the point in your actual building, rather than where the WRL app indicates you are on the floor plan before you tap the button. In some cases the position may be clear, such as the intersection of two hallways, but others GT points may be less clear. Purchase some small, sticky labels that you can attach to the floor to ensure you walk over the same exact GT point for each experiment you carry out.

When the last GT point has been logged, tap the **Stop RTT Positioning** button to stop the experiment. You are asked to save the log file. It's recommended that you save this in another Google Drive folder to make it easy to access and analyze at a later time. This also makes it easy to read the data into Google Sheets, or other online tools, to calculate the RTT position error as a cumulative distribution function (CDF) over the entire path. A full description of the WRL log file and the meaning of fields in its various records, is provided in Positioning Data Log File Format.

Configuration File Format

Below is an example configuration file, which can be downloaded, copied and customized to characterize your own positioning system. In particular, you should customize the building layout, the positions of the APs, and GT points. The various parameters are described in the format description that follows the example, and can be adjusted to suit your needs once you have some experience with the default operation.

<Example-House-Config.csv file>

```
##############################################################################
#
#  EXAMPLE HOUSE CONFIGURATION FILE (Single Level)
#
##############################################################################

#
# PARAMETERS commonly updated by a user
#
PARAM,WIFI_SCAN_INTERVAL_SECS,15.0
PARAM,RTT_MIN_INTERVAL_SECS,0.33
PARAM,FLP_PERIOD_SECS,5.0
PARAM,MAP_INITIAL_ZOOM,20.98
PARAM,IS_RANGE_IN_METERS,true
PARAM,MOBILE_CARRY_HEIGHT_METERS,1.0
PARAM,POSITION_ELPF_DISCOUNT_FACTOR,1.0
PARAM,LOG_LEVEL,0
#
# PARAMETERS (Advanced)
#
PARAM,FILTER_MIN_RANGES_NEEDED,3
PARAM,FILTER_MAX_RANGES_USED,5
PARAM,MAX_APS_IN_RTT_SCAN,6
PARAM,RTT_CALIBRATION_OFFSET_METERS,0.5
PARAM,RTT_CALIBRATION_SLOPE,0.92
PARAM,HISTORY_RANGE_TIME_TO_LIVE_SECS,5.0
PARAM,HISTORY_LINEAR_DISCOUNT_FACTOR,0.2
PARAM,RTT_MIN_DISTANCE_LIMIT_METERS,0.0
PARAM,RTT_MAX_DISTANCE_LIMIT_METERS,30.0
PARAM,RANGE_AVERAGING_WINDOW_SECS,60.0
PARAM,MAX_ALLOWED_POSITION_ERROR_METERS,5.0

#
# BUILDING Floor Plan and Location
#
BD,Floor-Plan-1,37.358195,-122.071502,37.358354,-122.071271,1

#
# AP Positions
#
AP,60:B7:6E:9E:7E:26,37.358321,-122.071277,RTT-20H-1,2.5,1,Floor-Plan-1,living-room
AP,60:B7:6E:9E:82:3B,37.358219,-122.071279,RTT-20H-2,2.5,1,Floor-Plan-1,office
AP,60:B7:6E:9E:7E:1B,37.358220,-122.071372,RTT-20H-3,2.5,1,Floor-Plan-1,dining-room
AP,60:B7:6E:D6:26:96,37.358200,-122.071499,RTT-20H-4,2.5,1,Floor-Plan-1,garage
AP,60:B7:6E:D6:2D:3F,37.358333,-122.071458,RTT-20H-5,2.5,1,Floor-Plan-1,master-bedroom
AP,60:B7:6E:D6:2E:7E,37.358343,-122.071409,RTT-20H-6,2.5,1,Floor-Plan-1,kitchen

#
# GROUND TRUTH (GT) Positions
#
```

```
GT,0,0,37.358326,-122.071290,1.0,0.1,POINT-A
GT,0,0,37.358326,-122.071395,1.0,0.1,POINT-B
GT,0,0,37.358267,-122.071383,1.0,0.1,POINT-C
GT,0,0,37.358267,-122.071358,1.0,0.1,POINT-D
GT,0,0,37.358289,-122.071358,1.0,0.1,POINT-E
GT,0,0,37.358289,-122.071290,1.0,0.1,POINT-F
```

<end-of-file>

Example of Multi Floor setup in v1.2 and optimizations in v1.8

<Example-House-2-Floor-Config.csv file>

```
################################################################################
#
#  EXAMPLE HOUSE CONFIGURATION FILE (Dual level)
#
################################################################################

#
# PARAMETERS commonly updated by a user
#
PARAM,WIFI_SCAN_INTERVAL_SECS,15.0
PARAM,RTT_MIN_INTERVAL_SECS,0.33
PARAM,FLP_PERIOD_SECS,5.0
PARAM,MAP_INITIAL_ZOOM,20.98
PARAM,IS_RANGE_IN_METERS,true
PARAM,MOBILE_CARRY_HEIGHT_METERS,1.0
PARAM,POSITION_ELPF_DISCOUNT_FACTOR,1.0
PARAM,LOG_LEVEL,0
#
# PARAMETERS (Advanced)
#
PARAM,FILTER_MIN_RANGES_NEEDED,3
PARAM,FILTER_MAX_RANGES_USED,5
PARAM,MAX_APS_IN_RTT_SCAN,6
# Note calibration offset changed to 0 as automated in the v1.8 param
PARAM,RTT_CALIBRATION_OFFSET_METERS,0
PARAM,RTT_CALIBRATION_SLOPE,0.92
PARAM,HISTORY_RANGE_TIME_TO_LIVE_SECS,5.0
PARAM,HISTORY_LINEAR_DISCOUNT_FACTOR,0.2
PARAM,RTT_MIN_DISTANCE_LIMIT_METERS,0.0
PARAM,RTT_MAX_DISTANCE_LIMIT_METERS,30.0
PARAM,RANGE_AVERAGING_WINDOW_SECS,60.0
PARAM,MAX_ALLOWED_POSITION_ERROR_METERS,5.0
# Params available from V1.8
PARAM,OPTIMIZE_WIFI_SCAN,true
PARAM,ACCESS_POINT_FREQUENCY_LIST,5180:5745
PARAM,AUTO_REMOVE_CALIBRATION_OFFSET,true
PARAM,NG_MEDIAN_WINDOW,180

#
# PARAMETERS (Multiple Floor Support)
```

```
#

# Parameter that must be enabled.
PARAM,ENABLE_LEVEL_CHANGE,true

# Recommended parameter for smoother level-change.
# Turning this on may consume more power.
PARAM,ENABLE_BAROMETER_SENSOR,true

# The other level change parameters can be left at their default values.
# For more parameter information read the “Parameters” overview section below.

#
# BUILDING Floor Plan and Location
#
BD,Floor-Plan-1,37.358195,-122.071502,37.358354,-122.071271,1

#
# Floor #1 AP Positions
#
AP,60:B7:6E:9E:7E:26,37.358321,-122.071277,RTT-20H-1,2.5,1,Floor-Plan-1,living-room
AP,60:B7:6E:9E:82:3B,37.358219,-122.071279,RTT-20H-2,2.5,1,Floor-Plan-1,office
AP,60:B7:6E:9E:7E:1B,37.358220,-122.071372,RTT-20H-3,2.5,1,Floor-Plan-1,dining-room
AP,60:B7:6E:D6:26:96,37.358200,-122.071499,RTT-20H-4,2.5,1,Floor-Plan-1,garage
AP,60:B7:6E:D6:2D:3F,37.358333,-122.071458,RTT-20H-5,2.5,1,Floor-Plan-1,master-bedroom
AP,60:B7:6E:D6:2E:7E,37.358343,-122.071409,RTT-20H-6,2.5,1,Floor-Plan-1,kitchen

# Add the 2nd floor with the same dimensions as the first, although it can be different.
# The building layout string (Floor-Plan-2) must be used as the building layout string for all
# APs on this floor. In addition, an overlay with the same name with a .png  extension (e.g.
# Floor-Plan-2.png) must be provided after loading this CSV file
#
BD,Floor-Plan-2,37.358195,-122.071502,37.358354,-122.071271,1

#
# Floor #2 AP Positions
#
AP,60:B7:6E:9E:7E:26,37.358331,-122.071277,RTT-20H-7,2.0,2,Floor-Plan-2,Upstairs Bedroom 1
AP,60:B7:6E:9E:82:3B,37.358229,-122.071279,RTT-20H-8,2.0,2,Floor-Plan-2,Upstairs Bedroom 2

#
# GROUND TRUTH (GT) Positions
# Uses new v1.2 feature where the 2nd field is used to indicate the overlay map used
# for the GT point. In this case, 3 points are on Floor-Plan-1 followed by 2 points on
# Floor-Plan-2 and then a last point on Floor-Plan-1.
#
GT,0,Floor-Plan-1,37.358326,-122.071290,1.0,0.1,POINT-A
GT,0,Floor-Plan-1,37.358326,-122.071395,1.0,0.1,POINT-B
GT,0,Floor-Plan-1,37.358267,-122.071383,1.0,0.1,POINT-C
GT,0,Floor-Plan-2,37.358267,-122.071358,1.0,0.1,POINT-D
GT,0,Floor-Plan-2,37.358289,-122.071358,1.0,0.1,POINT-E
GT,0,Floor-Plan-1,37.358289,-122.071290,1.0,0.1,POINT-F
```

<end-of-file>

Explanation of Configuration File Records

BUILDING

BD,<layout-file-name>,<SW-lat>,<SW-lng>,<NE-lat>,<NE-lng>,<Floor>

Example
BD, Floor-Plan-1, 37.430887, -121.92155, 37.431065, -121.920991,1

Field #	Field Name	Type	Function
0	BD	string	Record key **BD** = Building
1	Building Layout	string	Building name, encoded with floor number (eg. “Floor-Plan-1”. This should match the layout file name, but shouldn’t include the .png suffix. **New in v1.2** Multi-level floor plans are now available. You can specify multiple floors for the same building as separate BD lines. This “*Building Layout*” string should be used in the AP descriptions below to identify which APs are on which floor. Note: for each building layout string, a corresponding overlay file (with the exact same name and a .png extension) must be provided during the overlay file selection dialog.
2	SW Latitude	float	Latitude of SW corner in **degrees**
3	SW Longitude	float	Longitude of SW corner in **degrees**
4	NE Latitude	float	Latitude of NE corner in **degrees**
5	NE Longitude	float	Longitude of NE corner in **degrees**
6	Floor	string	Floor number, such as 1, or label, such as 4a Note: this field is just used for your own records. It is not used by the new level-change routines. Those routines use the building layout string as described above.

ACCESS POINT

AP,<bssid>,<lat>,<lng>,<name>,<height_m>,<floor_label>,<building_layout>,<location_label>,<OPTIONAL: channel frequency>

Example
AP, 3C:28:6D:78:AC:2E, 37.4163339, -122.0825435,RTT6-01,3.0,1,Floor-Plan-1,office,5180

Field #	Field Name	Units	Function
0	AP	string	Record key **AP** = Access Point
1	BSSID	string	Basic Service Set IDentifier: an AP's 48-bit MAC address. Format: xx:xx:xx:xx:xx:xx (x=hex digit)
2	Latitude	float	Latitude in **degrees.** If LCI location information is enabled in the Access Point, this can be left as 0.0 degs and will be filled in automatically. **New in V1.8**
3	Longitude	float	Longitude in **degrees.** If LCI location information is enabled in the Access Point, this can be left as 0.0 degs and will be filled in automatically. **New in V1.8**
4	Name	string	Unique name for AP
5	Height	float	Distance above floor in **meters**
6	Floor	string	Floor number, such as 1, or label, such as 4a Note: this field is just used for your own records. It is not used by the new level-change routines. Those routines use the building layout string as described below.
7	Building Layout	string	Building name, encoded with floor number (such as "Floor-Plan-1". This should match the corresponding entry in the **BD** record definition, and the layout file name, but shouldn't include the .png suffix. **New in v1.2** In v1.2, this building layout overlay string is used to determine if the user is still on the same floor, or has moved floors. Every AP on the same floor needs to have the same Building Layout string. It does not matter what the string is, but every AP must share the same string. APs with different strings are considered to be on different floors. The building layout string must match what was defined in the corresponding BD entry above.

			The v1.2 app will only range to APs located on the same floor, unless the app determines that the user is switching floors. In this case, the app will range to all nearby APs to determine which floor the user is currently on.
8	Location	string	Human-readable label for the location, such as room or area name for reference
9 (Optional)	Frequency	integer	**New in v1.8** The center frequency of the primary 20 MHz frequency (in MHz) of the channel the access point is using.

GROUND TRUTH

GT,<0>,<overlay label>,<latitude>,<longitude>,<altitude>,<error>,<location_label>

Example
GT,0,Floor-Plan-1,37.4218452,-122.0859171,100.0,1.0,Waypoint-C

Field #	Field Name	Units	Function
0	GT	string	Record key **GT** = Ground Truth
1	0	Reserved	-
2	Overlay Label	string	**New in v1.2** Overlay used to display this GT point. This can be used to put GT points on different floors. Set to 0 to disable this feature.
3	Latitude	float	Latitude in **degrees**
4	Longitude	float	Longitude in **degrees**
5	Altitude	float	Altitude in **meters** (WGS84 standard) [Not currently used]
6	Error	float	Estimated horizontal std. error in **meters**
7	Location	string	Human-readable label for the location

PARAMETER

PARAM,<parameter_key>,<parameter_value>

Example
PARAM, WIFI_SCAN_INTERVAL_SECS, 30.0

Field #	Field Name	Type	Function
0	PARAM	string	Record key **PARAM** = Parameter
1	parameter_key	string	Configuration parameter key to be set at startup. These are recorded in the data log file.
2	parameter_value	variable	Value the parameter key is set to.

SUMMARY OF PARAMETER KEY NAMES USED (Parameters use default values if not set)

PARAMETER NAME	Default	Type	Function
WIFI_SCAN_INTERVAL_SECS	15.0	float	Time between Wi-Fi Scans **seconds**
RTT_MIN_INTERVAL_SECS	0.35	float	Min Time between RTT Scans **seconds**
RTT_CALIBRATION_OFFSET_METERS	0.5	float	Correction bias **meters** subtracted from range.
RTT_CALIBRATION_SLOPE	0.92	float	Range slope correction (**no units**) multiplied by the range after bias (above) is subtracted.
HISTORY_RANGE_TIME_TO_LIVE_SECS	5.0	float	Range history time-to-live in **seconds** for each AP being ranged. Used when ranging errors occur.
HISTORY_LINEAR_DISCOUNT_FACTOR	0.2	float	Linear weighting for history (**no units**)
FILTER_MIN_RANGES_NEEDED	3	integer	Min ranges used to find a position
FILTER_MAX_RANGES_USED	4	integer	Max ranges used to find a position
MAX_APS_IN_RTT_SCAN	6	integer	Max number of APs in an range request up to the limit specified by Android (10 in Android S/T)
MOBILE_CARRY_HEIGHT_METERS	1.0	float	Height carried above floor in **meters**
RTT_MIN_DISTANCE_LIMIT_METERS	0	float	Min range estimate in **meters**
RTT_MAX_DISTANCE_LIMIT_METERS	30	float	Max range estimate (or reject) in **meters**

LOG_LEVEL	0	hex	Logcat Level (see table below)
IS_RANGE_IN_METERS	false	boolean	**Centimeters**: false, **Meters**: True This determines the units for the RTT position output as described in the RTT format table.
FLP_PERIOD_SECS	5.0	float	Time between FLP estimates. Range: 2.0 - 10.0 **seconds**
MAP_INITIAL_ZOOM	2.98	float	Map Zoom factor to show overlay map
POSITION_ELPF_DISCOUNT_FACTOR	1.0	float	Discount, or Smooth Factor, for Exponential Low Pass Filter Value: 0.0 (max filter) to 1.0 (no filter) (**no units**)
MAX_ALLOWED_POSITION_ERROR_METERS	5.0	float	If the position error is greater than this value, no position is returned for the current set of ranges. Max error value is in **meters**.

New in v1.2

PARAMETER NAME	Default	Type	Function
ENABLE_LEVEL_CHANGE	false	boolean	Top level switch to enable the level change capability. **true** = on, **false** = off Note: when level change is enabled, the Floor identification string in the **AP** and **BD** records are compared to determine the set of APs used for ranging. If the values don't match, the APs will not be part of the ranging set, even if they are on the same floor. The only exception is when a level change is detected. In that case, the app will range to the nearest APs (regardless of floor) to determine where it is.
ENABLE_BAROMETER_SENSOR	false	boolean	**true**: use the barometer for level change detection **false**: turn off the barometer. **Note**: when **true**, new barometer value entries are added to the log file. See Barometer entry in the log file details below.
LEVEL_CHANGE_MIN_RTT_RANGE_FAILURES_THRESHOLD	No default	integer	The minimum absolute number of RTT ranging failures that must occur before a floor change is detected. This parameter, if set, overrides the percentage value below.
LEVEL_CHANGE_MIN_RTT_RANGE_FAILURES_PERCENTAGE	0.8	float	The minimum (fractional) percentage of RTT ranging failures that must occur before a floor change is detected. This parameter

			is only used when a more specific threshold value (see parameter above) is not set. E.g., 0.1 = 10%, 0.01 = 1%, 0.5 = 50%
LEVEL_CHANGE_MIN_RTT_STDEV_ERROR_THRESHOLD	3.0	float	When the barometer is not used, the app will also use the RTT standard deviation error as an indicator of floor change. This parameter controls the minimum error threshold at which a floor change is detected. This parameter is not used when the barometer is active as the barometer is much more precise than using the standard error. This parameter is used when the barometer is not active, as it can trigger faster than using just the RTT ranging error, and Wi-Fi scan results. However, it is disabled when the barometer is active as it can cause false positives in weak Wi-Fi signal areas.
LEVEL_CHANGE_MIN_WIFI_SCAN_OTHER_FLOOR_AP_COUNT_THRESHOLD	No default	integer	The minimum absolute number of APs that are on a different *“Floor”* (as defined in the AP definition above) from the user. When this threshold is reached, the floor change routines are activated. This parameter, if set, overrides the percentage value below.
LEVEL_CHANGE_MIN_WIFI_SCAN_OTHER_FLOOR_AP_COUNT_PERCENTAGE	0.5	float	The minimum (fractional) percentage of APs that are on a different *“Floor”* (as defined in the AP definition above) from the user. When this percentage is reached, the floor change routines are activated. This parameter is only used when a more specific threshold value (see parameter above) is not set. E.g., 0.1 = 10%, 0.01 = 1%, 0.5 = 50%
LEVEL_CHANGE_DETECTION_MAX_ROUNDS_TO_REPEAT	10	integer	When a floor change is detected, the app will enter a mode where it scans all nearby APs (regardless of their floor) to determine the floor the user is currently on. This parameter determines how long this mode will be active.
LEVEL_CHANGE_RSSI_ARRAY_MAX_LENGTH	10	integer	The app keeps track of the AP with the strongest RSSI to determine the current floor - it assumes it is on the same floor as the APs with the strongest RSSI over the last **N** readings. This parameter sets the value of **N**.
LEVEL_CHANGE_RTT_ARRAY_MAX_LENGTH	10	integer	The app keeps track of the AP with the smallest RTT range to determine the

			current floor – the app assumes it is on the same floor as the APs with the smallest RTT ranges over the last **N** readings. This parameter sets the value of **N**. The RTT range takes precedence over the RSSI value (above), and in the event of a tie.
LEVEL_CHANGE_RELATIVE_BAROMETER_PRESSURE_MIN_THRESHOLD	0.175	float	To determine a floor change using the barometer, the app uses the change in relative pressure (hPa) seen over a small time window, and compares it to the relative pressure seen over a longer time window. If the difference is large enough, the user has changed elevations indicating a floor change. This parameter determines the minimum pressure-difference between the two time windows before a floor change is detected.
LEVEL_CHANGE_BAROMETER_SHORT_WINDOW_ARRAY_MAX_LENGTH	5	integer	The length of the short window (in seconds) used to calculate a relative pressure change.
LEVEL_CHANGE_BAROMETER_LONG_WINDOW_ARRAY_MAX_LENGTH	30	integer	The length of the long window (in seconds) used to calculate a relative pressure change.

New in v1.8

PARAMETER NAME	Default	Type	Function
OPTIMIZE_WIFI_SCAN	false	boolean	**true**: enable Wi-Fi scan optimization. **false**: disable Wi=Fi scan optimization. This can be turned on if the frequencies of APs are known in advance. The frequencies can be provided as a list of integers in ACCESS_POINT_FREQUENCY_LIST, or specified in each AP record. Once configured, the app will only perform one Wi-Fi scan each time positioning is started, eliminating pauses due to Wi-Fi scans during navigation.
ACCESS_POINT_FREQUENCY_LIST	5180:5745	colon separated integers	The list of channel frequencies used for Wi-Fi scan optimization. Each integer is the center frequency of the primary 20 MHz frequency (in MHz) of the channel

			the access point is using.
AUTO_REMOVE_CALIBRATION_OFFSET	false	boolean	**true**: enable automatic calibration error removal. **false**: disable automatic calibration error removal. Estimate calibration errors together with positions, and remove the errors automatically.
NG_MEDIAN_WINDOW	1	integer	The size of the bounding window for calculating the running median of estimated calibration errors. Note: the default window size is 1, which is equivalent to using the latest estimated calibration error (i.e. not using the running median). A suitable value for us with the running media window is 180.

LOG LEVEL DESCRIPTION (defines what appears in logcat)

Log Levels can be added together to combine log functions (controlled by bit position)
E.g. PARAM,LOG_LEVEL,7 turns on Basic stats + Status with key + AP scanning logging

Log Level (hex)	Function	Log Level (hex)	Function
0 (default)	Minimal logging (for example, exceptions)	08	FLP position data
01	Basic status	10	RTT position data
02	Status with key/values	20	RTT range data
04	Access point scanning	40+	Reserved

Positioning Data Log File Format

Explanation of Log File Records

PHONE_VERSION

PHONE_VERSION, <build_id>

Example

PHONE_VERSION,google/redfin/redfin:12/SP2A.220505.002/8353555:userdebug/dev-keys

Field #	Field Name	Type	Function
0	PHONE_VERSION	string	Record key **PHONE_VERSION**
1	Build Id	string	Device model name and build ID of the version of Android running on the phone.

APP_VERSION

APP_VERSION, <build_date-version>

Example

APP_VERSION,220410-V1.2

Field #	Field Name	Type	Function
0	APP_VERSION	string	Record key **APP_VERSION** (application version)
1	Build date-version	string	App build <YYMMDD>-V<version-id>

ACCESS POINT

Note: This information is similar to data in the **ap_location.csv** file, which is duplicated in some fields here to allow the data log to be self contained, and to include the AP reference points.

AP,<status>,<time>,<bssid>,<name>,<lat>,<lng>,<altitude>,<height>,<error>,<floor><building_layout><location_label>

Example

AP,0,0,3c:28:6d:98:b9:5d,RTT-P6-33,37.4160061,-122.08297,-1.0,2.8,0.1,6,PLYMOUTH-1625-6,Office

Field #	Field Name	Type	Function
0	AP	string	Record key **AP** = Access Point

1	Status	int	0=operational, -1 decommissioned
2	Time	int	0 (time the AP information was read)
3	BSSID	string	Basic Service Set ID: an AP's 48-bit MAC address. Format: xx:xx:xx:xx:xx:xx (x=hex digit)
4	Name	string	Unique name of AP
5	Latitude	float	Latitude in **degrees** (from config or LCI)
6	Longitude	float	Longitude in **degrees** (from config or LCI)
7	Altitude	float	Altitude in **meters** (WGS84 standard); obtained from Android's Fused Location Provider
8	Height	float	Distance above floor in **meters**
9	Error	float	Estimated horizontal std. error in **meters**
10	Floor	string	The label used to describe a floor
11	Building layout	string	The building layout file name (+ floor suffix)
12	Location	string	Human-readable name for the location

PARAMETERS

These are the values of the parameters read in from the input CSV. Any parameter not listed will have the default values listed in the parameter table provided earlier.

PARAM,<status>,<time>,<key>,<value>

Example

PARAM,0,0,MAX_APS_IN_RTT_SCAN,8

Field #	Field Name	Units	Function
0	PARAM	string	Record key **PARAM** = Parameter
1	Status	int	0 (meaning operational parameter)
2	Time	int	0 (time the parameter was applied)
1	Parameter key	string	Configuration parameter key set at startup
2	Parameter value	string	Value the parameter key is set to

INIT TIME

INIT_TIME,<timestamp_in_ms>

Example

INIT_TIME,1651097864144

Field #	Field Name	Units	Function
0	INIT_TIME	string	Record key **INIT_TIME** = Initialisation time for the RTT, FLP, LE records. All time values for those records will be relative to this base value

ROUND TRIP TIME (RTT) RANGE ESTIMATE

RTT,<status>,<time>,<bssid>,<rssi>,<distance>,<distance_std_dev>,<-1>,<-1>,<burst_size>,<burst_successes>,<-1>

Example

RTT,0,14725,3c:28:6d:7d:c3:f0,-59,250,16,-1,-1,8,7,-1

Field #	Field Name	Units	Function
0	RTT	string	Record key **RTT** = Round Trip Time
1	Status	int	Return code (0 = success)
2	Time	int	Time the measurement was made in millisecs since “Start RTT Positioning”. This value is relative to the base value provided by **INIT_TIME**
3	BSSID	string	802.11mc Access Point MAC address
4	RSSI	int	Received Signal Strength in **dBm**
5	Distance	int / float	Estimated distance **cm** or **meters** (See PARAM “IS_RANGE_IN_METERS”)
6	Distance std. dev.	int / float	Standard deviation **cm** or **meters** (See PARAM “IS_RANGE_IN_METERS”)
7	-1	reserved	Reserved

8	-1	reserved	Reserved
9	Burst Size	int	Number of frames transmitted in a burst
10	Burst Successes	int	Number of frames acked successfully
11	-1	reserved	Reserved

LOCATION ESTIMATE (w/ RTT)

LE,<status>,<timestamp_ms>,<lat>,<lng>,<altitude>,<standard deviation>

Example

LE,0,14725,37.4160953,-122.0825995,1.0,0.83

Field #	Field Name	Type	Function
0	LE	string	Record key **LE** = Location Estimate (from RTT)
1	Status	int	Return code (0: Success, non-zero: Fault)
2	Time	int	Time the measurement was made in **milliseconds.** This value is relative to the base value provided by **INIT_TIME**
3	Latitude	float	Latitude in **degrees**
4	Longitude	float	Longitude in **degrees**
5	Altitude	float	Altitude in **meters** (WGS84 standard); obtained from Android's Fused Location Provider
6	Error	float	Estimated horizontal sd error in **meters**

FUSED LOCATION PROVIDER (FLP) ESTIMATE

FLP,<status>,<time>,<lat>,<lng>,<altitude>,<error>

Example

FLP,0,14725,37.4160749,-122.0824975,-5.20,5.70

Field #	Field Name	Units	Function
0	FLP	string	Record key **FLP** = Fused Location Position
1	Status	int	Return code (0: Success)
2	Time	int	Time the measurement was made in **milliseconds.** This value is relative to the base value provided by **INIT_TIME**
3	Latitude	float	Latitude in **degrees**
4	Longitude	float	Longitude in **degrees**
5	Altitude	float	Altitude in **meters** (WGS84 standard)
6	Error	float	Estimated horizontal std. dev. error in **meters**

GROUND TRUTH

GT,<status>,<time>,<latitude>,<longitude>,<altitude>,<error>,<location_name>

Example

GT,0,20421,37.4160929,-122.0826701,1.0,2.0,POINT-B

Field #	Field Name	Units	Function
0	GT	string	Record key **GT** = Ground Truth
1	Status	int	Return code
2	Time	int	Time the measurement was made in **milliseconds**
3	Latitude	float	Latitude in **degrees**
4	Longitude	float	Longitude in **degrees**
5	Altitude	float	Altitude in **meters** (WGS84 standard)
6	Error	float	Estimated horizontal sd error in **meters**
7	Location	string	Human-readable name for the location

New in v1.2

SENSORS INIT TIME

SENSORS_INIT_TIME,<timestamp_in_ms>

Example

SENSORS_INIT_TIME,1651097860620

Field #	Field Name	Units	Function
0	SENSORS_INIT_TIME	string	Record key **SENSORS_INIT_TIME** = Initialisation time for all sensor records. E.g. BARO records. All time values for those records will be relative to this base value

Barometer

BARO,<status>,<app_time>,<sensor_time>,<pressure>

Example

BARO,0,8600,175506677033410,1013.5179

Field #	Field Name	Units	Function
0	BARO	string	Record key **BARO** = Barometer
1	Status	int	Return code
2	App Time	int	Time the measurement was read in the app in **milliseconds.** This value is relative to the base value provided by **SENSORS_INIT_TIME**
3	Sensor Time	int	Time measurement was made as recorded by the sensor in **nanoseconds**
4	Pressure	float	Barometric pressure in **hPa**

Data Log Analysis

The RTT logs can be analyzed with the POLAR webtool. It allows you to see the RTT estimated path, and the ground truth path, superimposed on the floorplan. Further, the tool will provide an estimate of the position error plotted as a Cumulative Distribution Function (CDF) over the recorded path. Three kinds of CDF are generated: 1) Euclidean error, 2) Cross Track error (X-TRK) and 3) Along Track error (A-TRK).

The tool can be accessed using the URL: **https://polar.webapps.google.com**

When accessing the site you will be presented with the Home tab overview (Fig. 3a), a Map tab (3b) allowing you to upload a log file and floorplan, a Process tab (3c) to see the ground truth path (green), estimated path (blue) and AP positions (yellow). Finally, the Metrics tab (3d) shows various analytic tables, and a CDF of the track errors, when you scroll to the bottom of the page.

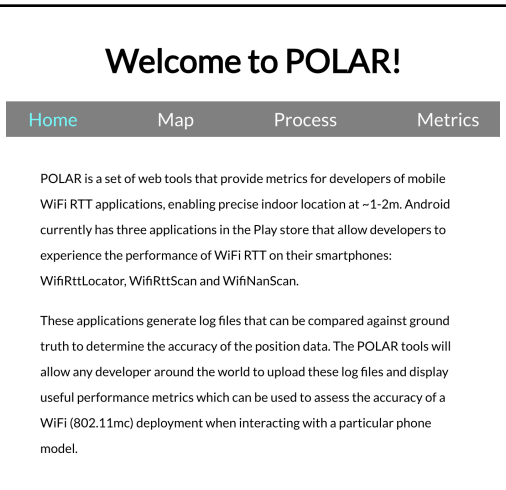

Fig 3a: Home Page

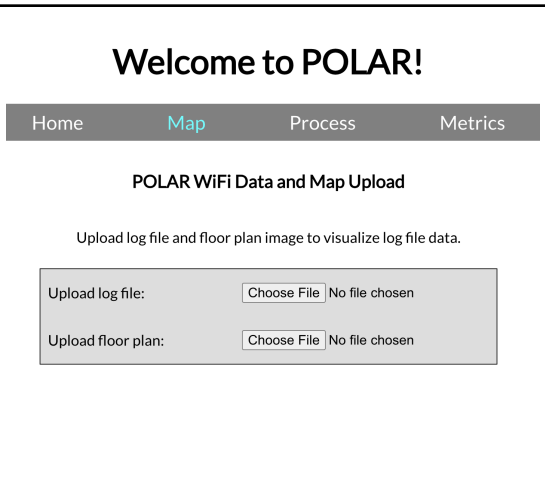

Fig 3b: Polar Map Page

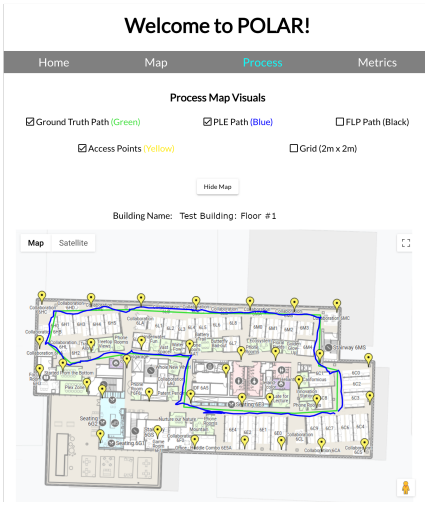

Fig 3c: Process

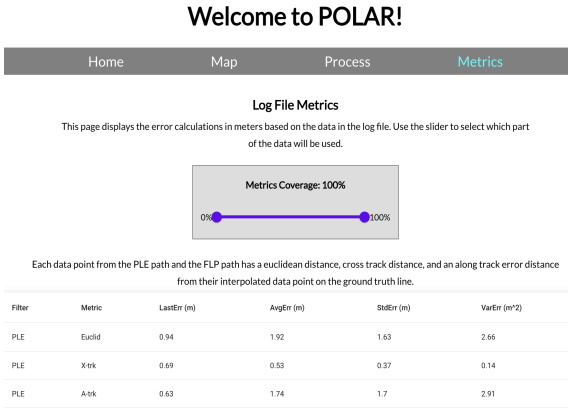

Fig 3d: Metrics

Conclusion

The Wi-Fi RTT Locator (WRL) application has been designed to help developers understand the accuracy and properties of Wi-Fi RTT positioning by itself, that is, without help from an Inertial Management Unit (IMU) or additional radio technology. Fusion of Wi-Fi RTT with other technologies can improve its accuracy further, but that is not the goal of this test application. We do, however, provide a post-positioning filter based on the Exponential Low Pass Filter (ELPF), configured off by default, but can be enabled using the POSITION_ELPF_DISCOUNT_FACTOR parameter, with a value less than 1.0 and greater than 0.0. More effective filters such as a Particle Filter or Kalman Filter are recommended for full system solutions. Use of alternate filters for positioning can also be evaluated by post-processing the data log file.

We hope you find this application useful for understanding and experimenting with Wi-Fi RTT positioning based on the IEEE 802.11mc standard, and that it inspires you to build applications customized to your precise indoor-positioning needs.

References

1. **Pixel Phones**: Models: Pixel 4, 4XL, 4a (5G) , 5, 5a (5G), 6, 6 Pro, 7, 7 Pro https://store.google.com/us/category/phones?hl=en-US

2. **Google Wi-Fi 2020:** with IEEE 802.11mc (RTT): https://store.google.com/us/product/google_wifi_2nd_gen?hl=en-US

3. **Wi-Fi RTT Video**: How to get one-meter location-accuracy from Android devices GoogleIO'18 video presentation on Precise Indoor Location using W-Fi RTT.