USING GNSS RAW MEASUREMENTS ON ANDROID DEVICES

WHITE PAPER

RAW MEASUREMENTS TASK FORCE
GSA GNSS

Towards better location performance in mass market applications

GALILEO EGNOS
NAVIGATION SOLUTIONS
POWERED BY EUROPE

European
Global Navigation
Satellite Systems
Agency

About the authors: The GSA GNSS Raw Measurements Task Force
Launched in June 2017 and coordinated by the European GNSS Agency (GSA), the GNSS Raw Measurements Task Force (Task Force) aims to share knowledge and expertise on Android raw measurements and its use, including its potential for high accuracy positioning techniques relevant to mass market applications. The Task Force includes GNSS experts, scientists and GNSS market players, all of whom are dedicated to promoting a wider use of these raw measurements.

This white paper has been jointly drafted by the following Task Force members:
- European GNSS Agency (GSA)
- European Space Agency (ESA)
- Nottingham Geospatial Institute, University of Nottingham
- Airbus Defence and Space GmbH

The following Task Force members contributed with test results, revisions and comments:
- European Commission Joint Research Centre (JRC)
- European Satellite Service Provider (ESSP)
- French Civil Aviation University (ENAC)
- GNSS Centre of Excellence (GCE)
- Rokubun
- University of Rijeka
- University of Zagreb
- The French Institute of Science and Technology for Transport, Development and Networks (IFSTTAR)
- The Swedish Research Institute (RISE)
- Waysure

The authors would like to convey their thanks to all Task Force members, the full list of which can be found at: https://www.gsa.europa.eu/gnss-raw-measurements-task-force.

Special thanks also goes to Google for the valuable comments they provided.

More information on the European Union is available on the Internet (http://europa.eu).
Luxembourg: Publications Office of the European Union, 2017
ISBN 978-92-9206-033-6
doi: 10.2878/449581

TABLE OF CONTENTS

EXECUTIVE SUMMARY

In May 2016, Google announced the availability of GNSS raw measurements from Android 7. For the first time, developers could access carrier and code measurements and decoded navigation messages from mass-market devices.

There are several advantages of using GNSS raw measurements on smartphones. Their use can lead to increased GNSS performance, as it opens the door to more advanced GNSS processing techniques that, until now, have been restricted to more professional GNSS receivers. These benefits have been demonstrated through Code Based Positioning, Code Aided Positioning, Differential Positioning and Precise Point Positioning. Although in normal conditions the position calculated from GNSS raw measurements may not be as optimal as a typical chipset output, in certain cases, when applying external corrections, using GNSS raw data may lead to improved accuracy of the solution. Several application areas stand to profit from this increased accuracy, such as augmented reality, location-based advertising, mobile health and asset management. The raw measurement allow also to optimise the multi-GNSS solutions, and to select the satellites based on their performances or differentiators. This is particularly relevant for Galileo, that is operational since December 2016 and still completing the full deployment, but offers already today excellent performances and signal advantages, such as the second frequency E5, and will offer soon differentiators such as the authentication on the Open Signal E1 and the Precise Point Positioning. For example, during the testing done to prepare this paper, it was possible to obtain Galileo-only positioning.

The availability of raw measurements is interesting also from a technological innovation point of view. It enables more detailed testing and post-processing, which in turn serves as a great educational tool. GNSS raw measurements may also support internal smartphone integrity by providing additional sources of information and allowing for the exploitation of receiver autonomous integrity monitoring algorithms.

Despite these advantages and opportunities, using GNSS raw measurements was however not as straightforward as it at first seemed. One year after the Google announcement only a few smartphone apps were making use of Android GNSS raw measurements. Two main reasons could explain this limited uptake. First, the GNSS experts may require assistance in understanding the specifics of raw measurements on Android. For example the standard formats, such as RINEX or NMEA, are not available on the Android platform. Secondly, the Java developers that are very familiar with the Android environment, don't usually understand the details of GNSS positioning.

In order to address this knowledge gap, the white paper makes a link between the GNSS raw measurements and their implementation on the Android platform[1]. It presents useful information on generating GNSS reference time and also step-by-step approach for deriving the pseudoranges from Android. It also highlights the results of several testing activities, which showcase working solutions based on GNSS raw measurements and in particular Galileo.

The White Paper is the combined work of several universities, public organisations and GNSS companies. The group, set up and coordinated by the European GNSS Agency (GSA), is known as the GNSS Raw Measurement Task Force.

1 The basic concepts of the raw measurements within this document are inspired by the Frank van Diggelen ION GNSS+ 2016's Android 7 GNSS raw measurements tutorial.

DOCUMENT STRUCTURE

The document is structured as follows:

- The first chapter provides a basic overview of GNSS, i.e. receiver architecture, GNSS time references, pseudoranges, navigation messages and position estimation. This introduction is focused on the necessary theoretical basics that are needed to reconstruct GNSS raw measurements using Android.
- Chapter 2 describes how to access the raw measurements and the first steps for using them, e.g. generation of pseudoranges and Doppler.
- Raw measurements bring a new spectrum of accuracy techniques to Android devices, including RTK and PPP. Chapter 3 depicts the most promising techniques and highlights the results collected from different sources. Moreover, the benefits and limitations of each technique are detailed.
- The last chapter presents the use cases that may benefit from the increased accuracy and integrity obtained with the use of GNSS raw measurements.

1 GNSS BASICS NEEDED FOR UNDERSTANDING RAW MEASUREMENTS

1.1 Introduction

This section introduces the basic concepts of GNSS that are essential for understanding raw measurements and their use. This is primarily a synthesis of existing academic literature [RD-03, RD-04, RD-05]. Experienced GNSS users are recommended to move directly to Chapter 2.

1.2 Global Navigation Satellite System

The GNSS space segment consists of various constellations of satellites orbiting Earth in MEO orbit, at an altitude of approximately 20 000 km and translating to the transmission delay of about 65 ms. There are currently four GNSS constellations in operation or in deployment phase: GPS (USA), GLONASS (Russia), BeiDou (China) and Galileo (Europe) [RD-02]. These are complemented by several regional GNSS and augmentation systems.

A user's position is estimated using the distance measurements (pseudoranges) between the user's receiver antenna and the position of at least four satellites. Both are determined by the receiver, which evaluates the satellite's signal and navigation message, respectively. This information is required by the PVT solution, which provides the user's position and time anywhere on the globe.

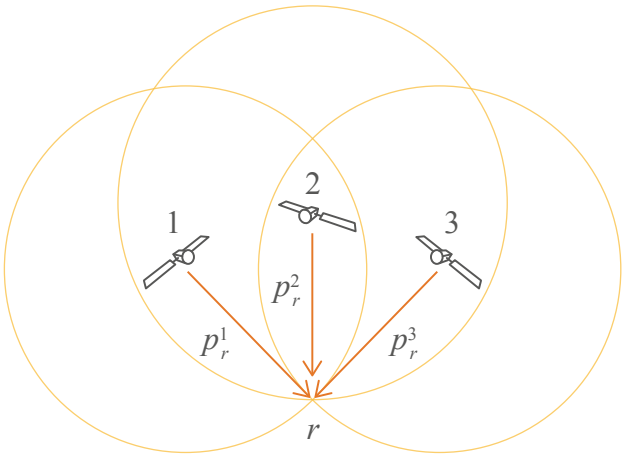

Figure 1: Intersecting spheres [RD-03]

To determine the geometric range, all GNSS signals are modulated with a specific, few milliseconds long, Pseudo-Random Noise (PRN) code that uniquely identifies the satellite. The receiver continuously compares and aligns a local copy of the PRN code with the received satellite signal. The measured delay of the received PRN code is equal to the transmission time if the transmitter and receiver clocks are perfectly synchronised. The PRN code is superimposed by the navigation data bits containing the position of the transmitting satellite. Knowing one satellite position, the receiver position lies somewhere on a sphere around the satellite with a radius equal to the range. If the range measurements of three satellites are available, the three spheres intersect at two points. Since one point is not located close to Earth surface, the second point is the true position of the receiver. In other words, the simultaneous measurement of the ranges to three satellites enables the determination of a fixed three-dimensional position, as illustrated in Figure 1. Due to the impact of the receiver clock, the three spheres do not intersect at a common point. This is estimated as the fourth unknown of the positioning problem, which is why at least four visible GNSS satellites are required [RD-03, RD-04, RD-05].

1.3 GNSS Receiver Architecture

A GNSS receiver process signals and provides the user with an estimated PVT solution. This PVT is based on the measured pseudoranges, information delivered through the navigation message, and optional assistance and augmentation provided by third parties. A generic block diagram of a GNSS receiver is shown in Figure 2.

The RF block (the left side of the diagram) includes the antenna and front-end, which are required for analogue signal processing. It can also include a low noise amplifier, filters and an intermediate-frequency down conversion. The final element in the block is the Analogue-to-Digital Converter (ADC).

In a smartphone, the base-band and PVT processing blocks (right side of the diagram) are software-based signal processing units that are designed to operate on a general-purpose hardware. The baseband processing is responsible for acquiring and tracking of the GNSS signals and decoding the navigation messages. Assisted data (external information) can be provided to reduce the time to fix.

The baseband processing block provides the raw data to the PVT block, which then computes the receiver's PVT. This process benefits from augmentation data (e.g. EGNOS) or accelerometers (sensors fusion), improving the accuracy and availability in harsh environments.

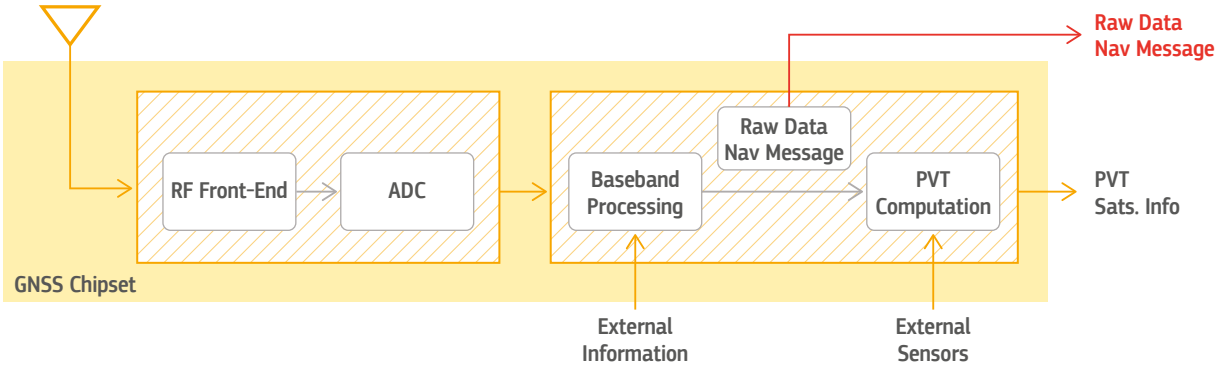

Figure 2: Generic block diagram of a GNSS receiver

Android 7 users can use ***android.location*** Application Programming Interface (API) to access the raw data required to calculate pseudoranges and decode navigation message (indicated by the red arrow on Figure 2). This data can be used to explore new algorithms and applications for mass-market devices.

The smartphone GNSS/navigation chip acts as a black box and only outputs the PVT and limited information from the tracked satellites. Some highly integrated chipsets use tight integration with cellular, WiFi and Bluetooth. Some even use motion/orientation sensors for cross aiding, which improves the accuracy and availability of the final position. In most cases, it is recommended to start your own algorithm using chipset PVT, periodically testing your solution against it.

1.4 Time

1.4.1 Introduction

While GNSS is best known for its positioning capabilities, it is also one of the most available and reliable sources of precise time.

Time must be defined in a coherent and uniform way. Hence, transformations between the reference times used by the different GNSS constellations are needed. Failing to account for this in a smartphone will lead to a reduction of the accuracy in the multi-constellation fix. Apart from a GNSS constellation's time reference, a universal reference time system is also defined, as further discussed in the following paragraph.

1.4.2 Reference times

The Temps Atomique International (TAI) second duration was defined in 1967 as: *The TAI second is the duration of 9 192 631 770 periods of the radiation corresponding to the transition between the two hyperfine levels of the ground state of the Caesium 133 atom.*

Universal Time (UT) is a solar time standard that reflects the average speed of the Earth's rotation, its most commonly used definition is known as UT1. Universal Time Coordinated (UTC) is the compromise between TAI and UT1. In fact, as an atomic time, UTC is as uniform as the TAI scale can be. However, in order to follow the Earth's rotation variations, it is always kept within 0.9 seconds with respect to UT1. This is accomplished by adding or subtracting a certain number of leap seconds to TAI.

The main concepts of GNSS positioning are based on timing - a time of arrival and time of transmission of signals. Each GNSS uses its own reference time, as outlined in Table 1.

Table 1: GNSS reference times

GNSS System	GNSS reference time
GPS	GPS Time (GPST)
Galileo	Galileo System Time (GST)
GLONASS	GLONASS Time (GLONASST)
Beidou	Beidou Time (BDT)

1.4.2.1 GPS Time

GPS Time (GPST) is continuous with no leap seconds. It starts at midnight (0 h UTC) between 5 and 6 January 1980. At that epoch, the difference between TAI and UTC was 19 s.

GPST is transmited using two parameters: Week Number (WN) and Time of Week (TOW). The first counts the weeks occurred from the start of the GPS Time to the current week. The weeks begin at midnight on Saturday. TOW counts the seconds occurred within the current week.

1.4.2.2 Galileo System Time

Galileo System Time is a continuous time scale starting 13 seconds before midnight between 21st August and 22nd August 1999, i.e. GST was equal to 13 seconds at 22nd August 1999 00:00:00 UTC. It was done so in order not to have any difference between GPST and GST.

1.4.2.3 GLONASS Time

GLONASS Time is generated by the GLONASS Central Synchroniser and the difference between the UTC from Russia (UTC(SU)). GLONASST should not exceed 1 ms + 3 h:

$$GLONASST = UTC(SU) + 3h + \tau,$$

where $|\tau| < 1ms$. GLONASS implements leap seconds like UTC. GLONASST is based on the second of the day and the day of the year.

1.4.2.4 BeiDou Time

BeiDou Time (BDT) is a continuous time scale starting at 0^{h} UTC on 1 January 2006 and is steered towards UTC.

1.4.2.5 Relationship of GNSS Reference Times

Multi-GNSS receivers can track and use different GNSS for the PVT solution. Usually, the receiver implements only one clock based on one reference time. Table 2 presents the relationships between the GNSS reference times, the TAI, and the relation between UTC and TAI. Figure 3 shows the time difference between the reference times and TAI.

Table 2: Relationships between Reference Times

Systems	Relationship
GPST - TAI	$TAI = GPST + 19s$
GST - TAI	$TAI = GST + 19s$
GLONASST - TAI	$TAI = GLONASST - 3h + leapsecond_{UTC\text{-}TAI}$
UTC - TAI	$UTC = TAI - leapsecond_{UTC\text{-}TAI}$
BDT-TAI	$TAI = BDT + 33s$

The time difference in seconds between UTC and TAI is defined as $leapsecond_{UTC\text{-}TAI}$. However, GNSS community describes the leap second term as the time difference between UTC and GPS. The difference, at the time of writing, can be expressed as

$$leapsecond = leapsecond_{UTC\text{-}TAI} - 19s$$

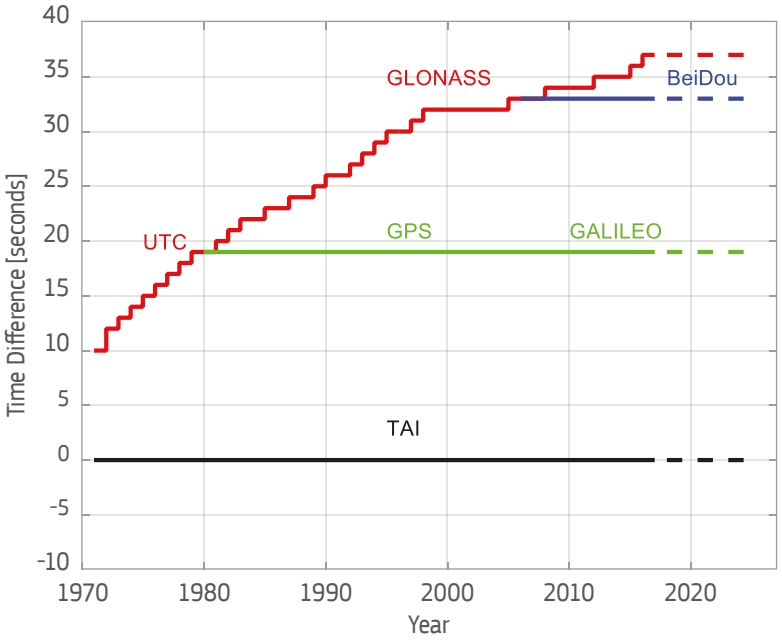

Figure 3: Time difference between reference times

GNSS receivers usually provide GNSS time in GPS System Time, which requires a conversion of GLONASS measurements – $GPST = GLONASST - 3h + leapsecond$, to generate the GLONASS pseudoranges based on the Android raw data.

1.5 Navigation Message and Tracking Status

The receiver needs to synchronise with the satellite transmitted signal. Currently, GPS transmits four different signals in L1 [RD-09], with the Coarse/Acquisition (C/A) code signal being the most important and most often used signal for mass-market devices. The PRN code is based on a 1-ms length code at a chipping rate of 1.023 chips per ms. A local replica of the C/A code is generated with the proper delay and Doppler. Figure 4 shows a perfect synchronisation between the periodical C/A code, transmitted by the satellite (repeated every 1 ms), and its local replica generated by the receiver (in blue).

At this stage, the receiver is not able to provide the full transmitted time, and only fractional pseudoranges are available. For instance, at t_0, the receiver can only provide the relative delay to the beginning of the C/A code period. In the same way, at t_1, the receiver provides the delay to the beginning of the current C/A code. The number of entire C/A codes between t_0 and t_1 is unknown.

If the receiver is only synchronised with the C/A code, the valid range of the transmitted time is from 0 to 1 ms, making the computation of the travel-time ambiguous.

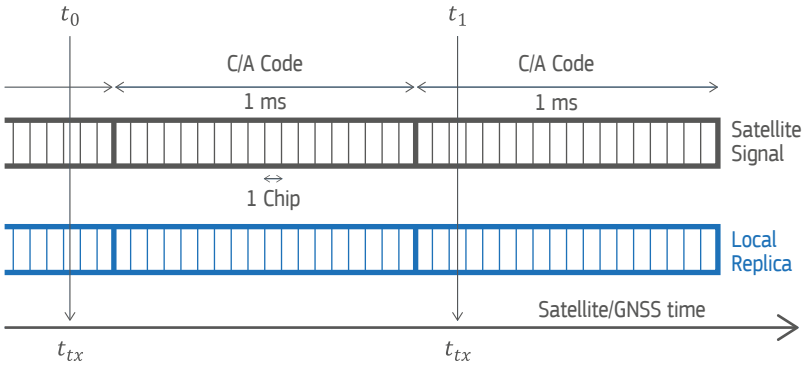

Figure 4: Synchronisation with the C/A code

To solve this time ambiguity, the receiver can exploit the navigation message structure. Figure 5 shows the different synchronisation stages required by the receiver to obtain the satellite time:

- **Code Lock:** the receiver is locked to the C/A code. The valid range is 0–1 ms.
- **Bit Sync:** the receiver is synchronised with the bits. The valid range is 0–20 ms.
- **Subframe Sync:** the receiver is synchronised with the subframes. The valid range is 0–6 s.
- **TOW Decoded:** all the subframes contain the TOW. Therefore, once the TOW is decoded, the valid range is 0–1 week.

Existing literature describes more advanced algorithms capable of computing the PVT solution using only fractional pseudorange. This navigation solution is known as a coarse-time navigation problem [RD-10]. Furthermore, A-GNSS receivers can also obtain the TOW through external systems and do not need to decode the broadcasted message.

In the case of GPS, five consecutive subframes construct a frame, with a navigation message formed by 25 frames. More information related to the GPS message structure can be found in [RD-09].

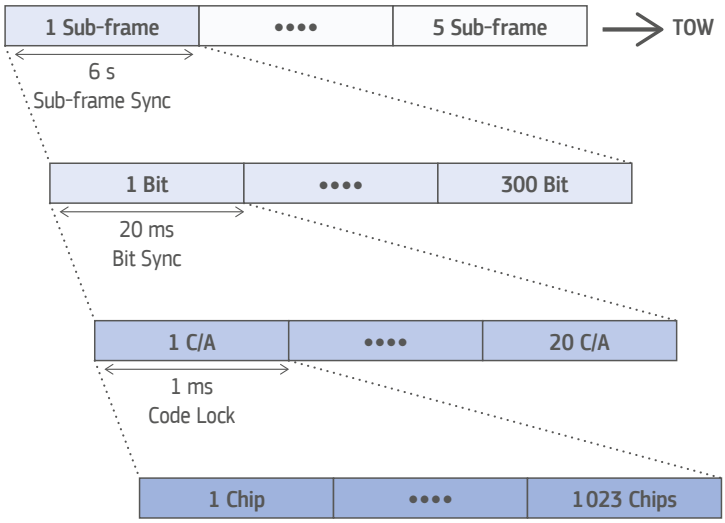

Figure 5: GPS L1 C/A code and navigation message structure

Galileo uses the E1 Open Service (OS) modulation - the multiplexing of two components:

- **E1B:** the E1 OS Data component. It is generated from the I/NAV navigation data stream and the primary code (E1BC).
- **E1C:** the E1 OS Pilot component. It is composed of the primary code (E1BC) and the secondary code (E1C_2nd).

Figure 6 shows the structure of both the E1B and E1C components. The first is based on a primary code E1BC with a length of 4 ms, with the symbols of the same length as the E1BC code. The pages (I/NAV service) are composed of 500 symbols, including the TOW. A sub-frame is composed of 15 pages. The full I/NAV message is composed of 34 subframes.

The pilot component is composed of a primary code with a length of 4 ms and with a secondary code E1C_2nd placed on top of it. It is 25 chips long, with each chip lasting 4 ms. The full secondary code has a length of 100 ms. Since there is no navigation message in the pilot component, the receiver can track the signal coherently for longer periods, allowing it to receive a much weaker signal. Many receivers track the pilot component when the ephemerides and clocks are obtained through external sources or decoded from the navigation message.

Once the receiver is synchronised with the satellite signal and it has obtained GNSS time, the unambiguous transmitted time can be computed. GNSS time can also be obtained from external sources, but these methods is outside the scope of this document. Tracking the secondary Galileo code results in a time ambiguity of 100 ms. As the time that the signal travels from the satellite to the receiver is 70 ms, the full pseudorange can be easily generated, unlike the GPS one.

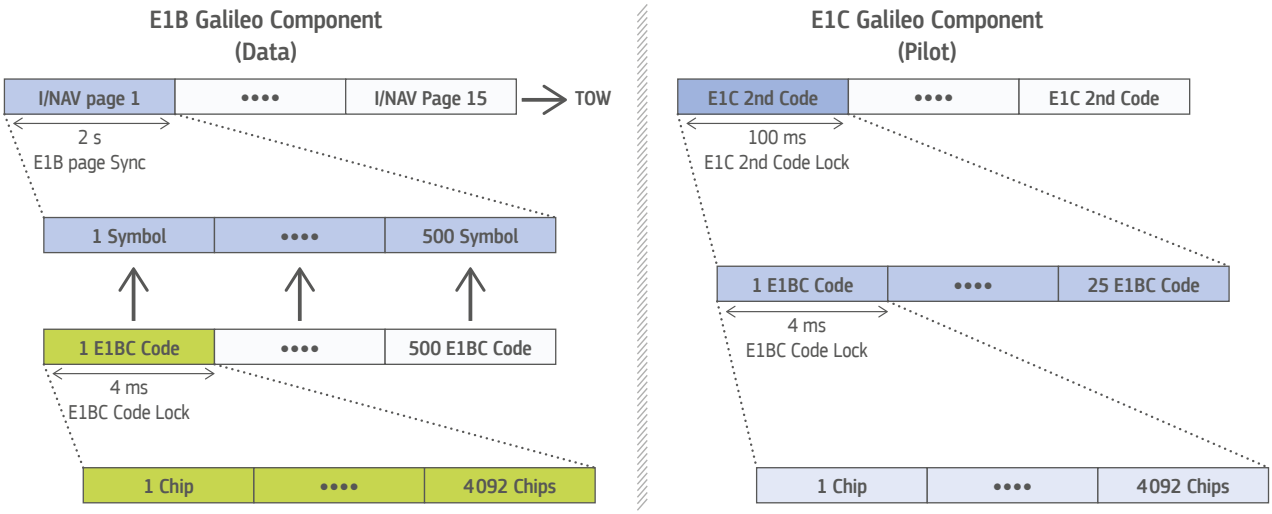

Figure 6: E1B Galileo component (left side) and E1C Galileo component (right side)

The structure of Beidou and GLONASS navigation messages and the tracking status are similar to GPS. Table 3 summarises time ambiguity and the tracking status for the four main constellations.

Table 3: Time ambiguity based on the tracking status

GPS		GALILEO		GLONASS		BeiDou	
Sync Status	**Time**	**Sync Status**	**Time**	**Sync Status**	**Time**	**Sync Status**	**Time**
C/A code	1 ms	E1BC code	4 ms	C/A code	1 ms	C/A code	1 ms
Bit	20 ms	E1C 2nd code	100 ms	Bit	20 ms	Bit	20 ms
Subframe sync	6 s	E1B page	2 s	String	2 s	Subframe sync	6 s
TOW	1 week	TOW	1 week	Time of Day	1 day	TOW	1 week

1.6 Pseudorange Generation

Although the concept of the pseudorange is simple, its generation is anything but straightforward, as this measurement of distance is obtained through time measurements. GNSS receivers process the received signals to obtain the transmitted (t_{Tx}) and received time (t_{Rx}). The difference of both is the signal's time of travel from the satellite to the receiver (assuming no extra delays due to ionosphere, troposphere and other elements). The pseudorange can be computed as

$$\rho = (t_{Rx} - t_{Tx}) \cdot c,$$

where c is the speed of light in vacuum. t_{Tx} is also needed to compute the satellite position.

Pseudorange generation and PVT computation can be implemented through two different methods: common reception time or common transmission time [RD-11]. In the following paragraphs, pseudorange generation is based on the common reception time method.

Common reception time is known as the measurement time (t_{meas}^{Rx}). Four satellites transmit the signals at the same time, with the same TOW being transmitted by all the satellites at the same time epoch (left side of Figure 7). Due to different propagation paths, the four signals arrive at the receiver with different delays. The receiver then computes the time offset between the TOW and the current time at the epoch. This is known as the measurement time (t_{meas}^{Rx}). As a result, the transmitted satellite times (provided in GNSS system time) at measurement time can be expressed as

$$t_{Tx,Sat1} = t_{TOW}^{GNSS} + \Delta_1,$$
$$t_{Tx,Sat2} = t_{TOW}^{GNSS} + \Delta_2,$$
$$t_{Tx,Sat3} = t_{TOW}^{GNSS} + \Delta_3,$$
$$t_{Tx,Sat4} = t_{TOW}^{GNSS} + \Delta_4,$$

where t_{TOW}^{GNSS} is the transmitted TOW and Δ_i is the delay between the TOW and the measurement time.

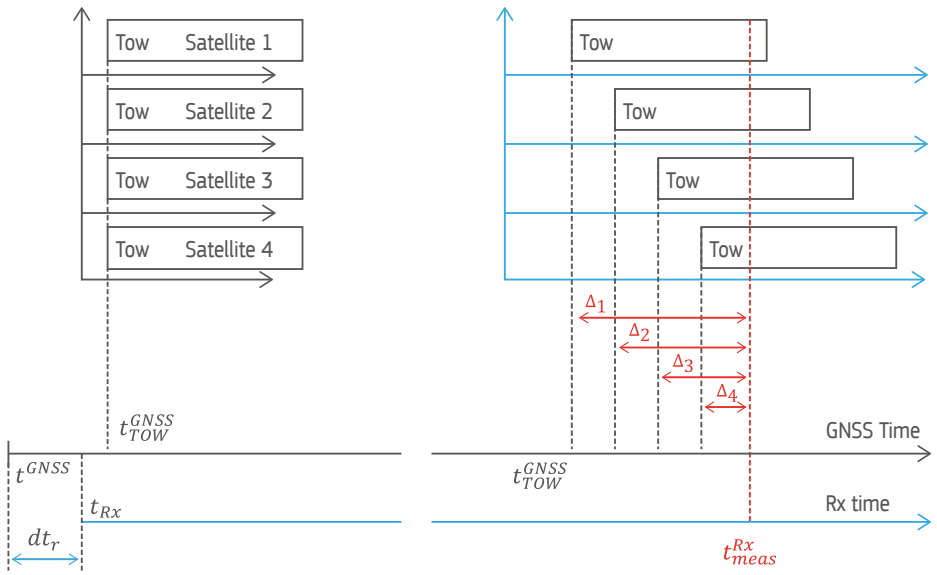

Figure 7: Pseudorange computation based on common reception time

Before it computes the first PVT and decodes the TOW from at least one satellite, the receiver has no information about the GNSS time. Thus, it must make assumptions in order to generate the Rx time and to compute the first set of pseudoranges. The first satellite signal to arrive is used as a reference. The received time (t_{meas}^{Rx}) is the transmitted time plus a reference propagation time (t_{ref}^{path}). A standard value between 65 and 85 ms is usually assumed. Therefore, the first measured time can be computed as:

$$t_{meas}^{Rx}[1] = t_{Tx,Sat1}[1] + t_{ref}^{path}.$$

All other pseudoranges are generated relative to the first one. As constant error to all the satellites is present (since t_{ref}^{path} has been fixed, not estimated), a bias between the GNSS and the received time is introduced (dt_r). The measurement time, in GNSS reference time, can be expressed as:

$$t_{meas}^{Rx,GNSS}[k] = t_{meas}^{Rx}[k] - dt_r[1],$$

where k is the k-th measurement time. Once the first set of pseudoranges has been computed and the first PVT obtained, $dt_r[1]$ is determined and used for all the following sets of pseudoranges. However, the receiver computes and updates the clock bias in all PVT sets. Therefore, the following sets of pseudoranges can be computed as:

$$\rho_1[k] = (t_{meas}^{Rx}[k] - t_{Tx,Sat1}[k] + dt_r[1])c,$$
$$\rho_2[k] = (t_{meas}^{Rx}[k] - t_{Tx,Sat2}[k] + dt_r[1])c,$$
$$\rho_3[k] = (t_{meas}^{Rx}[k] - t_{Tx,Sat3}[k] + dt_r[1])c,$$
$$\rho_4[k] = (t_{meas}^{Rx}[k] - t_{Tx,Sat4}[k] + dt_r[1])c,$$

where c is the speed of light in vacuum. Some receivers can provide the $t_{meas}^{Rx}[k]$ as the sum of the measurement time and an offset.

1.7 Position Estimation

1.7.1 Single-GNSS constellation

Neglecting further impacts on the pseudorange, the range measurement R^j between the receiver antenna and the j-th satellite consists of the real range ρ^j and the receiver clock error multiplied by the speed of light in vacuum:

$$R^j = \rho^j + c \cdot dt_r.$$

The aim is to estimate the receiver antenna coordinates (x, y, z) and the clock offset dt_r from a set of pseudorange measurements R^j. Knowing the position (x^j, y^j, z^j) of the j-th satellite obtained from the transmitted navigation message or other external sources, the pseudorange can be rewritten as:

$$R^j = \sqrt{(x^j - x)^2 + (y^j - y)^2 + (z^j - z)^2} + c \cdot dt_r, j = 1,2, \dots, n(n \geq 4).$$

The simultaneous range measurements to at least four satellites in view is sufficient to solve the problem. As the equation defines a non-linear system, we solve it with an iterative linearization, starting from an approximate position (x_0, y_0, z_0) of the receiver:

$$R^j - \rho_0^j = \frac{x_0 - x^j}{\rho_0^j} dx + \frac{y_0 - y^j}{\rho_0^j} dy + \frac{z_0 - z^j}{\rho_0^j} dz + c \cdot dt_r, j = 1,2, \dots, n(n \geq 4),$$

where $dx = x - x_0, dy = y - y_0, dz = z - z_0$ and ρ_0^j is the computed range to satellite j based on its coordinates (x^j, y^j, z^j) and the approximate solution (x_0, y_0, z_0) of the receiver's position. The equation system can be expressed in matrix notation as:

$$\begin{bmatrix} R^1 - \rho_0^1 \\ \vdots \\ R^n - \rho_0^n \end{bmatrix} = \underbrace{\begin{pmatrix} \frac{x_0 - x^1}{\rho_0^1} & \frac{y_0 - y^1}{\rho_0^1} & \frac{z_0 - z^1}{\rho_0^1} & 1 \\ \vdots & \vdots & \vdots & \vdots \\ \frac{x_0 - x^n}{\rho_0^n} & \frac{y_0 - y^n}{\rho_0^n} & \frac{z_0 - z^n}{\rho_0^n} & 1 \end{pmatrix}}_{H} \begin{bmatrix} dx \\ dy \\ dz \\ c \cdot dt_r \end{bmatrix}.$$

To solve for the four unknowns, at least four satellites in view are necessary. As the pseudorange contains various sources of error (Figure 8), it is beneficial to use all available pseudorange measurements to estimate the position.

For more than four satellites (n>4), an over-determined system is obtained, which can be solved using the non-linear least-squares (or related Kalman filter) estimation procedure. After solving the equation, the updated estimate of the receiver coordinates is:

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} x_0 \\ y_0 \\ z_0 \end{bmatrix} + \begin{bmatrix} dx \\ dy \\ dz \end{bmatrix}.$$

In a consecutive step, this pseudorange equations can be linearized again using the updated receiver position (x_0, y_0, z_0). These steps are repeated until the difference between two consecutive iterations is below a given threshold. Further details regarding positioning are given in [RD-12, RD-13, RD-14].

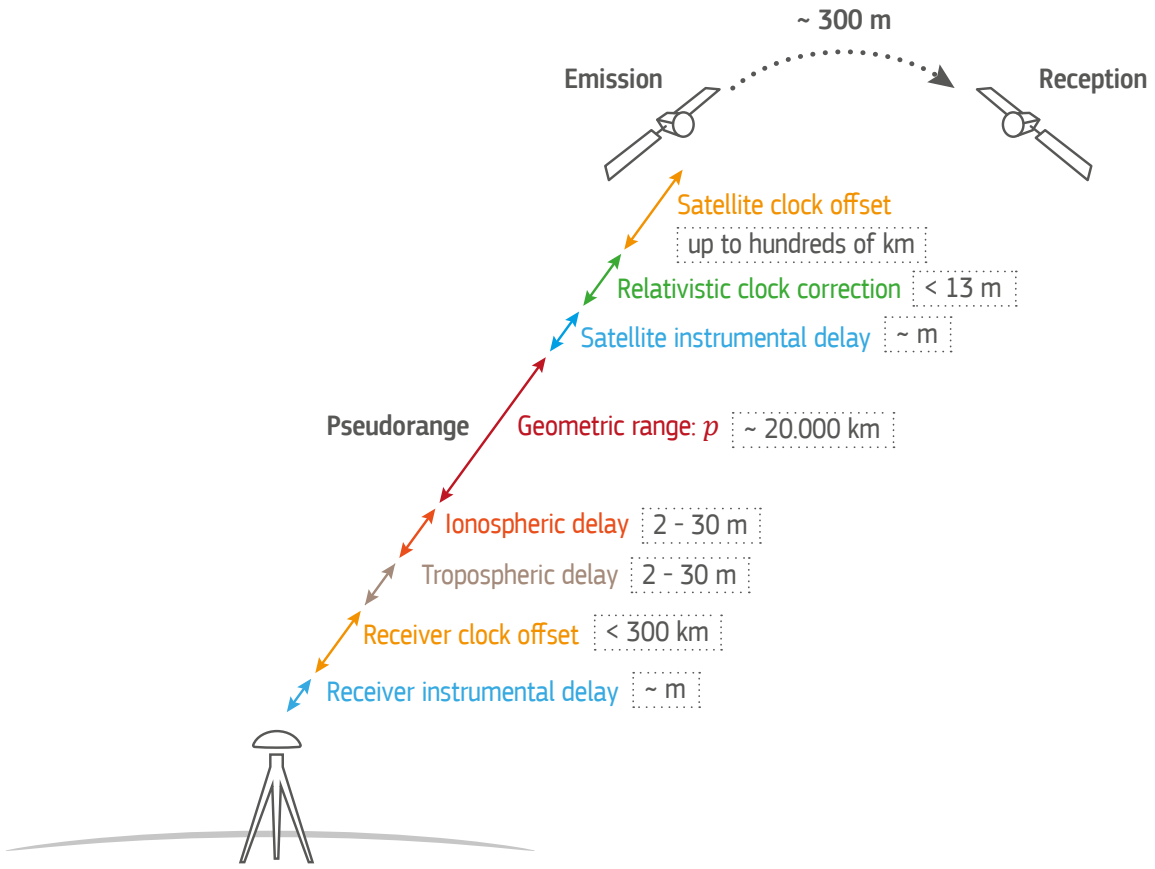

Figure 8: Sources of GNSS pseudorange measurement errors [RD-05]

1.7.2 Multi-GNSS constellation

Multiple GNSS constellations can be used to solve the position solution. In harsh environments, a combination of multiple GNSS constellations can increase positioning accuracy. However, combining signals from different constellations in the PVT requires one to account for the time bias between the systems (Inter-System Bias, ISB) [RD-16, RD-17]. Otherwise, the range measurements will contain an additional error, thus degrading the position solution. One strategy is the use of a-priori knowledge of the ISB. For instance, thanks to Galileo's inter-system operability with GPS, the GPS to Galileo Time Offset (GGTO) is transmitted in the navigation message. It can be also provided to the user through assisted data. If the ISB is eliminated, the position solution can be computed for a single satellite navigation system [RD-16].

The ISB can be also be estimated by introducing it as an additional unknown parameter in the position equation. If all four systems (GPS, GLONASS, Galileo and BeiDou) are used for positioning, the three inter-system biases between GLONASS, Galileo and BeiDou compared to GPS must be determined, resulting in the extended navigation equation:

$$\begin{bmatrix} R^1 - \rho_0^1 \\ \vdots \\ R^n - \rho_0^n \end{bmatrix} = H \cdot \begin{bmatrix} dx \\ dy \\ dz \\ c \cdot dt_r \\ c \cdot d\tau_{GPS/GLONASS} \\ c \cdot d\tau_{GPS/Galileo} \\ c \cdot d\tau_{GPS/BeiDou} \end{bmatrix}.$$

In this case, the range measurements of at least seven satellites are required. Note that this also impacts the actual Dilution of Precision (DOP) of the PVT solution, following the extra-State Theorem: *The addition of an extra state to a least-squares problem makes all DOPs greater than or equal to their original values increasing the positioning error.*

2. HOW TO ACCESS GNSS RAW MEASUREMENTS USING ANDROID APIS

2.1 Introduction

The most common way to obtain a position on the Android platform is via a fused location provider that combines several sources (GNSS, Wi-Fi or even mobile networks) in order to improve the accuracy, time to first fix, availability or power consumption. This is available via ***android.gsm.location*** API. The new Android 7 (Nougat) version, along with subsequent versions (Android O), introduces the new Android Location API (***android.location***) and brings significant innovations to the location service.

This chapter describes the ***android.location*** API and the methodology needed to obtain the typical GNSS parameters from the raw measurements itself. It focuses only on GNSS location capabilities. Other location methods (Wi-Fi, mobile networks, etc.) are not discussed.

2.2 Location API Before Android 7 – ***android.gsm.location***

The interaction between Android applications and the different mobile sensors, such as GNSS, is conducted using the Android framework API. Each new version of the Android platform is associated with a new API, and the configuration, interaction and user access to the GNSS chipset all depend on the API level. Table 4 shows the relationship between Android versions and API levels.

Table 4: Android versions and API levels

Version	Codename	API
2.3.3 – 2.3.7	Gingerbread	10
4.0.3 – 4.0.4	Ice Cream Sandwich	15
4.1.x 4.2.x 4.3	Jelly Bean	16 17 18
4.4	Kitkat	19
5.0 5.1	Lollipop	21 22
6.0	Marshmallow	23
7.0 7.1	Nougat	24 25
8.0	Oreo	26

Figure 9 summarises the structure of the Location API pre-Android 7 (API level 24), including the GNSS chipset drivers and the data transmission between the chipset and the OS.

User applications access the GNSS data using the framework Location API. Until Android API 23, this was limited to the data listed at the bottom of Figure 9, namely: GPS satellite information (C/No, azimuth, elevation if a particular satellite has been used in the PVT), NMEA sentences and PVT solution with the proper time stamp. Users can send basic configuration commands to the chipset, including restart/starting the GNSS chipset or cleaning the assisted data. However, all the configuration settings that include GNSS constellation priorities and different PVT algorithms are driven by the chipset itself.

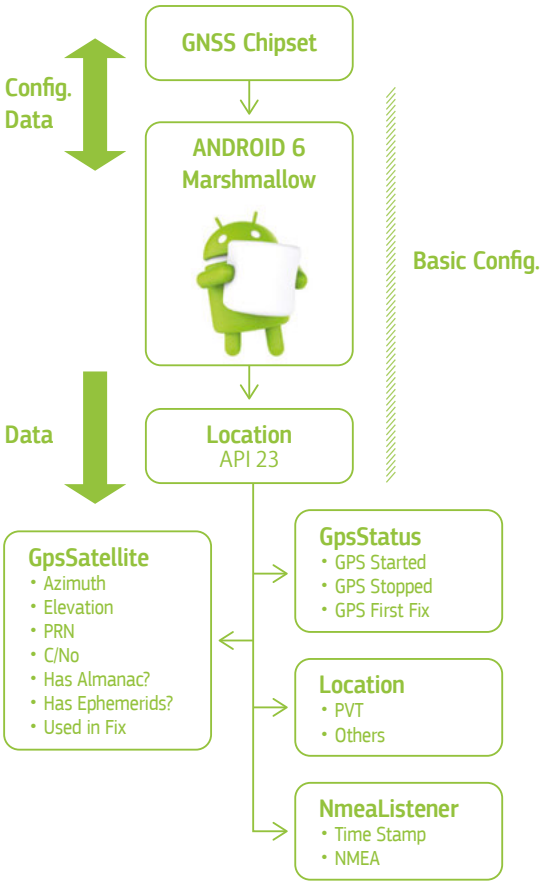

Figure 9: Location API in Android API Level 23

Starting with API 23 (Android 6), developers gained access to the following Android classes:

- GPS Satellite, containing such basic satellite information as azimuth, elevation, PRN and C/No. It also flags if the satellite is used in the PVT solution and the availability of almanac and ephemerides.
- GPS Status provides information about the status and solution of the GNSS chipset.
- Location, indicating if a positional and time solution is provided.
- NMEA Listener, providing basic NMEA sentences.

2.3 Location API in Android 7

Any new API level is compatible with previous versions, meaning that all the capabilities from a previous API level are still valid in the new version. Figure 10 shows the location API level 24 as implemented in Android 7.

From API 24 (Android 7), developers have access to the following GNSS raw and computed information via Android classes:

- GNSS Clock, that contains:
  - Receiver time (used to compute the pseudorange);
  - Clock bias.
- GNSS Navigation Message that contains:
  - Navigation Message bits (all the constellations);
  - Navigation message status.
- GNSS Measurement that contains:
  - Received Satellite Time (used to compute the pseudorange);
  - Code;
  - Carrier phase.

While this data comes directly from the GNSS chipset, we do not have direct access to the chipset itself.

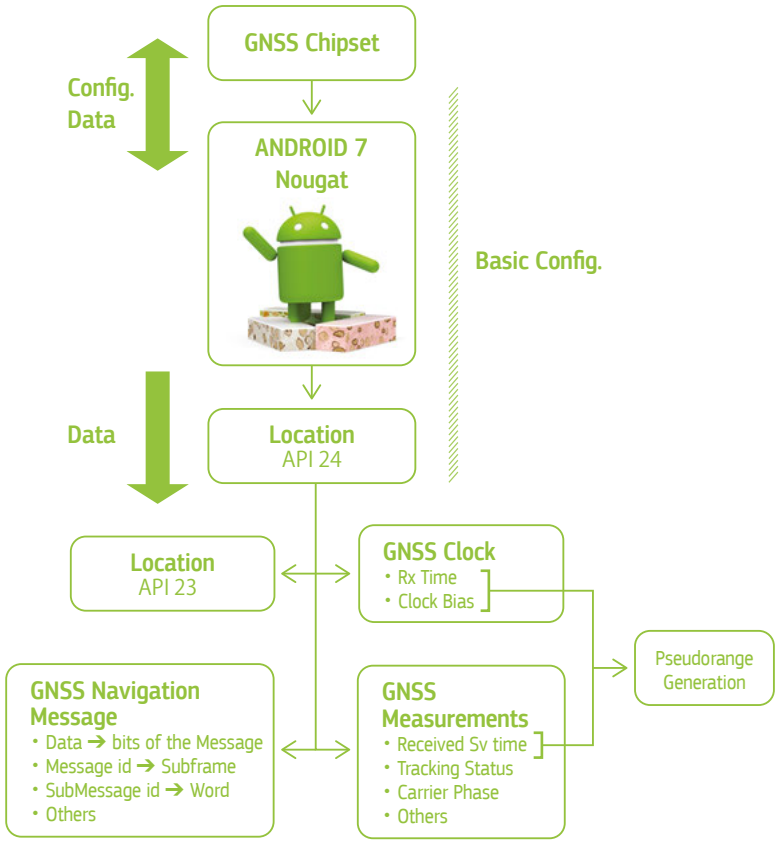

Figure 10: Location API in Android API Level 24/25/26

2.4 Using GNSS Raw Measurements

The main goal of this chapter is to explain how to obtain typical GNSS parameters from the raw measurements. **Android API does not provide a straightforward pseudorange.** The list of fields of the new Android 7 Location API (Table 5) does not include GNSS time or pseudorange. These need to be calculated.

Table 5: Android 7 Location API - Clock and Measurements fields

Android 7 Location - Clock and Measurements		
ANDROID CLASS	**FIELD**	**DESCRIPTION**
GNSSClock	*TimeNanos*	GNSS receiver's internal hardware clock value in nanoseconds
GNSSClock	*BiasNanos*	Clock's sub-nanosecond bias
GNSSClock	*FullBiasNanos*	Difference between TimeNanos inside the GPS receiver and the true GPS time since 0000Z, 6 January 1980
GNSSClock	*DriftNanosPerSecond*	Clock's drift
GNSSClock	*HardwareClockDiscontinuityCount*	Count of hardware clock discontinuities
GNSSClock	*LeapSecond*	Leap second associated with the clock's time
GNSSMeasurement	*ConstellationType*	Constellation type
GNSSMeasurement	*Svid*	Satellite ID
GNSSMeasurement	*State*	Current state of the GNSS engine

GNSSMeasurement	*ReceivedSvTimeNanos*	Received GNSS satellite time at the measurement time
GNSSMeasurement	*AccumulatedDeltaRangeMeters*	Accumulated delta range since the last channel reset
GNSSMeasurement	*Cn0DbHz*	Carrier-to-noise density
GNSSMeasurement	*TimeOffsetNanos*	Time offset at which the measurement was taken in nanoseconds
GNSSMeasurement	*CarrierCycles*	Number of full carrier cycles between the satellite and the receiver
GNSSMeasurement	*CarrierFrequencyHz*	Carrier frequency at which codes and messages are modulated
GNSSMeasurement	*PseudorangeRatemetersperSecond*	Gets the Pseudorange rate at the timestamp

Android navigation message information is compatible with GPS C/A, BeiDou D1 & D2, GLONASS L1 C/A and Galileo F/NAV and I/NAV, as listed in Table 6. However, exact information provided depends both on the GNSS chipset and the smartphone manufacturer.

Table 6: Android 7 Location API - Navigation Message parameters

Android 7 Location - Navigation Message	
PARAMETER	**DESCRIPTION**
Data	Gets the data of the reported GPS message. • For **GPS L1 C/A, Beidou D1 and Beidou D2**, each subframe contains 10 30-bit words. Each word (30 bits) should fit into the last 30 bits in a 4-byte word (skip B31 and B32), with MSB first, for a total of 40 bytes, covering a time period of 6, 6 and 0.6 s, respectively. • For **Glonass L1 C/A**, each string contains 85 data bits, including the checksum. These bits should fit into 11 bytes, with MSB first (skip B86-B88), covering a time period of 2 s. • For **Galileo F/NAV**, each word consists of 238 bits (sync and tail symbols excluded). Each word should fit into 30 bytes, with MSB first (skip B239, B240), covering a time period of 10 s. • For **Galileo I/NAV**, each page contains two page parts, even and odd, with a total of 2 x 114 = 228 bits (sync and tail excluded) that should fit into 29 bytes, with MSB first (skip B229–B232).
Message Id	• For **GPS L1 C/A** subframes 4 and 5, this value corresponds to the 'frame id' of the navigation message in the range of 1–25 (subframes 1, 2 and 3 do not contain a 'frame id' and this value can be set to -1). • For **Glonass L1 C/A**, this refers to the frame ID in the range of 1–5. • For **BeiDou D1**, this refers to the frame number in the range of 1–24. • For **Beidou D2**, this refers to the frame number in the range of 1–120. • For **Galileo F/NAV's** nominal frame structure, this refers to the subframe number in the range of 1–12. • For **Galileo I/NAV's** nominal frame structure, this refers to the subframe number in the range of 1–24.
Status	Gets the Status of the navigation message.
Submessage Id	• For GPS L1 C/A, BeiDou D1 and BeiDou D2, this corresponds to the subframe number of the navigation message in the range of 1–5. • For Glonass L1 C/A, this refers to the string number in the range from 1–15. • For Galileo F/NAV, this refers to the page type in the range 1 6. • For Galileo I/NAV, this refers to the word type in the range 1–10. Especially for Galileo, information embedded within the data bits may be even more useful for interpretation than the nominal page and word types provided in this field.
Svid	Satellite ID
Type	Gets the type of the navigation message.

The following paragraphs explain how to obtain:

- GNSS time
- Pseudoranges
- Carrier Phase Measurements
- Doppler
- Satellite ID
- Constellation ID

Moreover, it is discussed how the duty cycle technique, used in smartphones for reducing power consumption, is affecting positioning performance.

2.4.1 GPS time generation

Android 7 does not provide the GNSS time directly, but the internal hardware clock and the bias to the true GPS time (in nanoseconds) is provided if the receiver has estimated the GNSS reference time. When the receiver has estimated the GPS time, it can be computed as:

$$GpsTime = TimeNanos - (FullBiasNanos + BiasNanos)[ns],$$

where $FullBiasNanos$ is the bias between the receiver clock and the GPS time in nanoseconds and $BiasNanos$ is in sub-nanoseconds. If the receiver has estimated the time using a non-GPS constellation, the estimated GPS time can be computed as:

$$GpsTime = TimeNanos - (FullBiasNanos + BiasNanos) - InterSystemsBias,$$

where $InterSystemsBias$ is the offset between GPS time and the GNSS time used in the time estimation. For example, if the time is estimated using the Galileo System Time, $InterSystemsBias$ equals GGTO.

2.4.2 Pseudorange generation

As already stated in 2.4, the Android system does not directly provide pseudoranges. Instead, it provides all the parameters needed to compute them. As discussed in section 1.2, the construction of the pseudorange is based on the time difference, i.e. the time difference between the received time (measurement time) and the transmitted time:

$$\rho = \frac{(t_{Rx} - t_{Tx})}{1E9} * c,$$

where t_{Tx} is the received GNSS satellite time at the measurement time, i.e. the GNSS reference time when the signal was transmitted, t_{Rx} is the measurement time and c is the speed of light in vacuum. t_{Tx} is provided by the Android 7 system as:

$$t_{Tx} = ReceivedSvTimeNanos[ns],$$

where $ReceivedSvTimeNanos$ is the received GNSS satellite time at the measurement time in nanoseconds. The valid range of t_{Tx} depends on the tracking status (Chapter 1) and, if the tracking status does not equal TOW decoded (GPS), t_{Tx} becomes ambiguous, i.e. the pseudorange becomes ambiguous. Computation of PVT through ambiguous pseudoranges, i.e. coarse-time navigation problem, is not covered in this document (for details see [RD-10]). To obtain full pseudoranges (unambiguous), the following tracking status is required:

- GPS: TOW Decoded
- BeiDou: TOW Decoded
- GLONASS: Time of Day
- Galileo:
  - TOW Decoded
  - E1C 2nd Code: Despite the valid range of 0–100 ms, these measurements must be used for the pseudorange computation.

    Practical tests show that after a short period of time, Galileo measurements go from TOW decoded to E1C 2nd Code status and can remain in this status for minutes. Some Galileo ready chips track the data component in order to decode the navigation message. Once the ephemerides and clocks have been properly decoded, they start tracking the E1C component (pilot component) and the tracking status is flagged as the E1C 2nd Code status, with the ambiguity of the pseudorange as 100 ms, allowing for the reconstruction of the full pseudoranges. ***Authors strongly encourage developers to use the Galileo measurements only with E1C 2nd Code status.***

The measurement time, in full GNSS time, can be reconstructed as:

$$t_{Rx_{GNSS}} = TimeNanos + TimeOffsetNanos - (FullBiasNanos(1) + BiasNanos(1))[ns],$$

where $TimeOffsetNanos$ is the time offset at the measurement time in nanoseconds. Only the first value (see section 1.6) of $FullBiasNanos$ and $BiasNanos$ is used to compute all the received times. This operation applies until there is a discontinuity in the internal received time. This usually only happens when the GNSS module is restarted.

$t_{Rx_{GNSS}}$ is only provided in the GNSS reference system, which is used to compute the receiver time. However, t_{Tx} is provided for each GNSS system, e.g. GPST for GPS measurements or GLONASST for GLONASS measurements. Therefore, t_{Tx} must be converted to the same reference time system as $t_{Rx_{GNSS}}$ and vice versa. Usually, GNSS receivers implement GPS as a default GNSS reference time.

As $t_{Rx_{GNSS}}$ is in full GNSS time, with the range of t_{Tx} depending on the tracking status, when the pseudorange is computed, both components must be in the same range.

The next paragraphs demonstrate, using two different approaches, how to compute the pseudoranges for GPS, Galileo, GLONASS and BeiDou based on the $t_{Rx_{GNSS}}$ and t_{Tx} measurements, which require two transformations:

- both measurements to the same GNSS time system; and
- both measurements to the same valid range.

2.4.2.1 Approach 1:

Following the Matlab code provided by Google [RD-18], the measurement time can be computed for GPS and Galileo with TOW decoded status as:

$$t_{Rx} = t_{Rx_{GNSS}} - weekNumberNanos[ns]$$

for BeiDou with TOW decoded status as:

$$t_{Rx} = t_{Rx_{GNSS}} - weekNumberNanos[ns] - 14s$$

for Galileo with E1C 2nd code status as:

$$t_{Rx} = t_{Rx_{GNSS}} - milliSecondsNumberNanos[ns]$$

for GLONASS (Time of Day) as:

$$t_{Rx} = t_{Rx_{GNSS}} - DayNumberNanos + 3h - leapsecond[ns]$$

All the parameters in the equations must be in nanoseconds. $weekNumberNanos$ is the number of nanoseconds that have occurred from the beginning of GPS time to the current WN. $DayNumberNanos$ is the number of nanoseconds that have occurred from the beginning of GPS time to the current day. $milliSecondsNumberNanos$ is the number of milliseconds that have occurred from the beginning of the GPS time. They can be computed as:

$$weekNumberNanos = floor\left(\frac{-FullBiasNanos}{NumberNanoSecondsWeek}\right) * NumberNanoSecondsWeek[ns],$$

$$DayNumberNanos = floor\left(\frac{-FullBiasNanos}{NumberNanoSecondsDay}\right) * NumberNanoSecondsDay[ns]$$

and

$$milliSecondsNumberNanos = floor\left(\frac{-FullBiasNanos}{NumberNanoSeconds100Milli}\right) * NumberNanoSeconds100Milli[ns],$$

where $NumberNanoSeconds$ is the number of nanoseconds within one week, i.e. $NumberNanoSeconds = 604800e9$. $NumberNanoSecondsDay$ is the number of nanoseconds within one day and $NumberNanoSeconds100Milli$ is the number of nanoseconds within 100 ms.

2.4.2.2 Approach 2:

Using the mod operator, the measurement time for GPS and Galileo with TOW decoded can be obtained as:

$$t_{Rx} = mod(t_{Rx_{GNSS}}, NumberNanoSecondsWeek)[ns],$$

for BeiDou with TOW decoded as:

$$t_{Rx} = mod(t_{Rx_{GNSS}}, NumberNanoSecondsWeek) + 14s[ns],$$

for Galileo with E1C 2nd status and as:

$$t_{Rx} = mod(t_{Rx_{GNSS}}, NumberNanoSeconds100Milli)[ns]$$

and

$$t_{Rx} = mod(t_{Rx_{GNSS}}, NumberNanoSecondsDay) + 3h - leapsecond[ns]$$

for GLONASS with Time of Day status.

Having t_{Rx} and t_{Tx}, the construction of pseudorange is as straightforward as:

$$\rho = \frac{(t_{Rx} - t_{Tx})}{1E9} * c. [s].$$

2.4.2.3 Examples

2.4.2.3.1 Example 1:

The following Matlab code snippet shows how to compute the pseudorange for Galileo, GPS and BeiDou signals when the TOW is encoded:

```
% Select GPS + GAL TOW decoded (state bit 3 enabled)
pos = find( (gnss.Const == 1 | gnss.Const == 6) & bitand(gnss.State,2^3);
% Generate the measured time in full GNSS time
tRx_GNSS = gnss.timeNano(pos) - (gnss.FullBiasNano(1) + gnss.BiasNano(1));
% Change the valid range from full GNSS to TOW
tRx = mod(tRx_GNSS(pos),WEEKSEC*1e9);
% Generate the satellite time
tTx = gnss.ReceivedSvTime(pos) + gnss.TimeOffsetNano(pos);
% Generate the pseudorange
prMilliSeconds = (tRx - tTx );
pr = prMilliSeconds *Constant.C*1e-9;
```

2.4.2.3.2 Example 2:

The following Matlab code snippet shows how to compute the pseudorange for GLONASS signals when the Time of Day is decoded:

```
% Select GLONASS and Time of Day decoded (state bit 7 enabled)
pos = find( (gnss.Const == 3) & bitand(gnss.State,2^7);
% Generate the measured time in full GNSS time
tRx_GNSS = gnss.timeNano(pos) - (gnss.FullBiasNano(1) + gnss.BiasNano(1));
% Change the valid range from full GNSS to Time of Day and to compensate the
% GPS to GLONASS time difference: 3 hours + the LeapSecons
tRx = mod(tRx_GNSS, DAYSEC*1e9)+ (3*60*60-Constant.LeapSecond)*1e9;
% Generate the satellite time
tTx = gnss.ReceivedSvTime(pos) + gnss.TimeOffsetNano(pos);
% Generate the pseudorange
prMilliSeconds = (tRx - tTx );
pr = prMilliSeconds *Constant.C*1e-9;
```

2.4.3 Carrier phase measurements

Carrier phase measurements are provided by Android 7 as *AccumulatedDeltaRangeMeters* in metres. They are ambiguous, without the time information, meaning the receiver can only count the number of cycles occurring between epochs. If a cycle slip occurs, the receiver loses this count. The validity of the carrier measurement is provided via *AccumulatedDeltaRangeState* field. The possible flags are listed in Table 7. **Only valid measurements should be used for calculation.**

Table 7: AccumulatedDeltaRangeState

AccumulatedDeltaRangeState		
STATUS	VALUE	DESCRIPTION
ADR_STATE_CYCLE_SLIP	4	A cycle slip has been detected.
ADR_STATE_RESET	2	A reset has been detected.
ADR_STATE_VALID	1	The state is valid.
ADR_STATE_UNKNOWN	0	The state is invalid or unknown.

2.4.3.1 Duty cycle

Smartphone vendors prioritise power consumption. Since a continuously operating GNSS chipset will drain the battery, several techniques can be implemented to maintain a low power consumption. The most common one is the duty cycle, depicted in Figure 11.

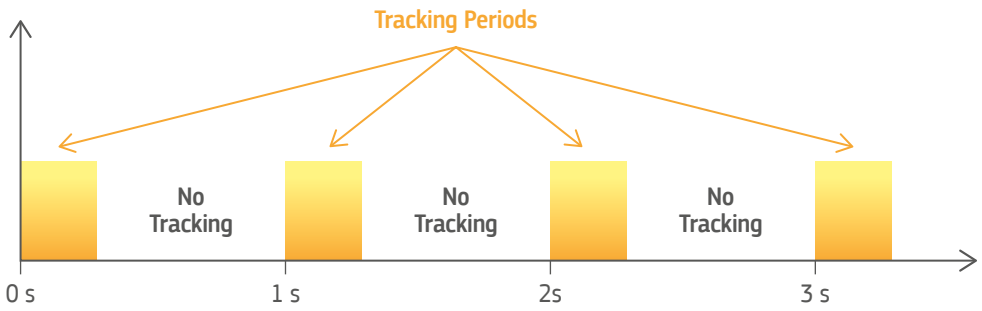

Figure 11: Duty cycle versus time

There are two different implementations of the duty-cycle, depending on the GNSS clock status:

1. **TCXO Hardware clock is not continuous (turning on and off) during the non-tracking periods.** The phone has two oscillators: higher precision compensated for temperature variations (TCXO) that is being used for GNSS tracking and an alternative, low-power consumption, crystal oscillator (XO). During the short tracking period, when the chipset tracks GNSS data, TCXO is used. Outside of this period, when a chip shuts down for several hundreds of milliseconds, XO is providing time.

Therefore, the *HardwareClockDiscontinuityCount* increases and the duty cycle can be detected using the raw measurements.

2. **TCXO Hardware clock is continuous during the non-tracking periods:** when the GNSS chipset is turned off, the TCXO clock keeps running, even though no signal is being tracked. Users cannot detect when the receivers are utilising duty cycle, as there is no discontinuity in the *GNSS clock and the HardwareClockDiscontinuityCount* keeps the same value.

Even though this technique is transparent for the users, with a new location being computed every second, it does have an impact on the carrier phase measurements. Without tracking continuity, several cycle slips may occur between two consecutive measurements, severely limiting the use of such advance phase techniques as Real Time Kinematic (RTK) or Precise Point Positioning (PPP).

2.4.4 Doppler

The Doppler shift that results from satellite movement can be derived from the *PseudorangeRateMetersPerSecond*, provided that the pseudorange rate, at the timestamp, is in m/s. This value in Hz does not include corrections for receiver and

satellite clock frequency errors (uncorrected value). A positive 'uncorrected' value indicates that the satellite is moving away from the receiver. The sign of the 'uncorrected' 'pseudorange rate' and its relation to the sign of 'doppler shift' is given by the equation:

$$pseudorangerate = -k * dopplershift,$$

where k is a constant depending on the signal's centre frequency (e.g. f_c for L1 = 1575.42e6 Hz) and the speed of light (c) as follows $k = c/f_c$, i.e. the wavelength.

2.4.5 Satellite ID

The satellite ID is provided directly by Android 7 using *Svid* field. The expected values for each constellation are:

- GPS: 1–32
- SBAS: 120–151, 183–192
- GLONASS: either orbital slot number (OSN) or frequency channel number (FCN) + 100
  - 1–24 as the OSN (preferred, if known)
  - 93–106 as the FCN (−7 to +6) + 100. That is, FCN of −7 is encoded as 93, 0 as 100 and +6 as 106. Since two GLONASS satellites transmit navigation signals on the same carrier frequency, assisted data is needed to identify the correct OSN.
  - QZSS: 193–200
- Galileo: 1–36
- Beidou: 1–37

2.4.6 Constellation

The constellation ID is provided directly by Android 7 using the *ConstellationType* field. Values for each constellation are listed in Table 8.

Table 8: ConstellationType

ConstellationType	
STATUS	VALUE
CONSTELLATION_BEIDOU	5
CONSTELLATION_GALILEO	6
CONSTELLATION_GLONASS	3
CONSTELLATION_GPS	1
CONSTELLATION_QZSS	4
CONSTELLATION_SBAS	2
CONSTELLATION_UNKNOWN	9

2.5 Raw Data Architecture

User applications access GNSS data using the framework API Location, which before the release of the Android 7.0, was limited to the Google Play Services (***android.gms.location***). This API focuses on encapsulated and simplified positional information, using a battery optimised fused location provider and combining multiple sensors, including GNSS, WiFi and Bluetooth. It can be seen as a black box as everything is done inside chip: the acquisition and tracking blocks decode navigation message and generates the pseudoranges, phase measurements and time. These are corrected using the navigation message (clock errors, ionosphere and troposphere, etc.). Finally, the position, velocity and time (PVT) is calculated and output by the chipset.

The new API (***android.location***) provides direct access to both the raw GNSS observations and the PVT solution. Figure 12 shows an unassisted receiver architecture and the main differences between new and old API. For clarity, only newly accessible parameters (white boxes) are depicted. Other parameters, such as C/No or satellite position, are omitted. Pseudoranges are not provided by the new API directly, but the parameters needed to generate them are.

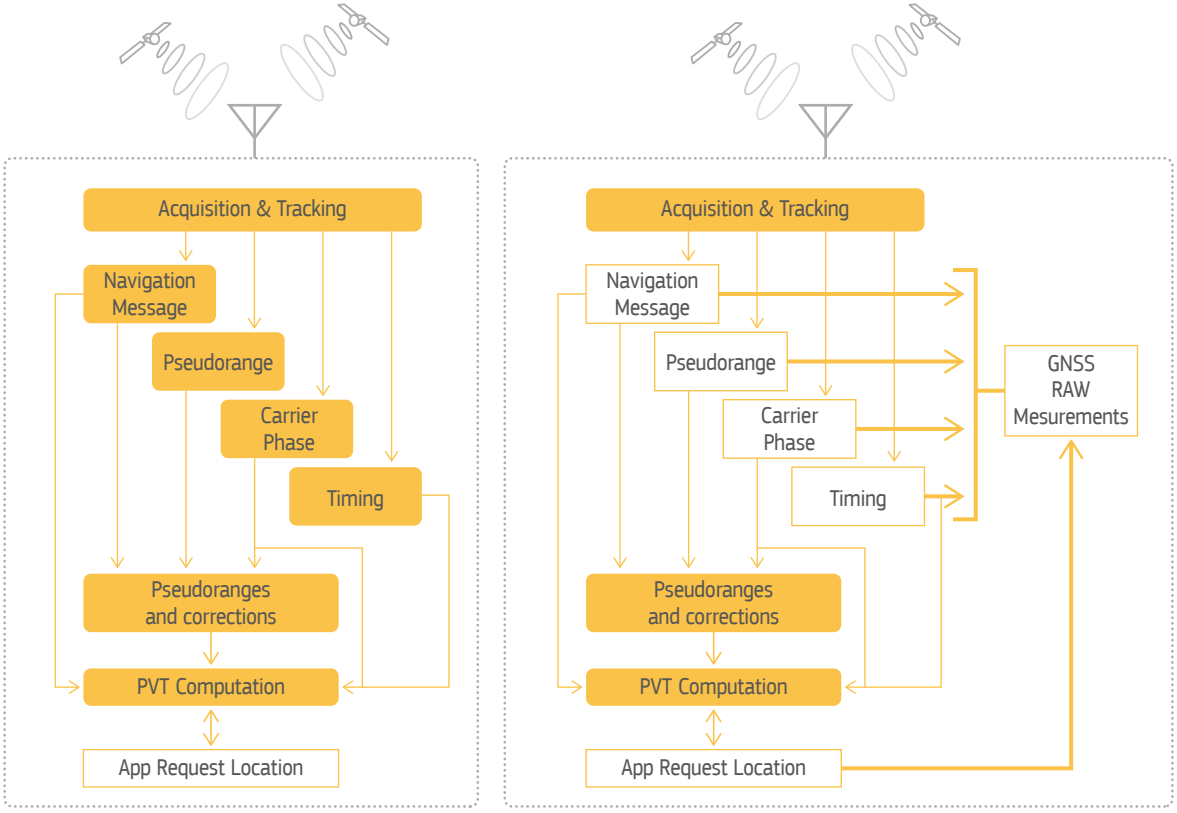

Figure 12: ***android.gms.location*** (before Android 7) vs ***android.location*** (Android 7)

Android's Location Based Apps (LBA) request location through the ***android.location.LocationManager*** class by specifying location provider (Network, Wi-Fi, or GPS/GNSS) or using the fused provider, depending on the accuracy and power criteria.

If we utilise raw measurements, this architecture can be extended to the Custom Location Provider, which can be based, for instance, on the smoothed pseudoranges or differential GNSS data. The custom location must be calculated at the system operation level. This new architecture is depicted in Figure 13, which shows existing infrastructure in blue and new, example capabilities enable by the raw measurements in orange.

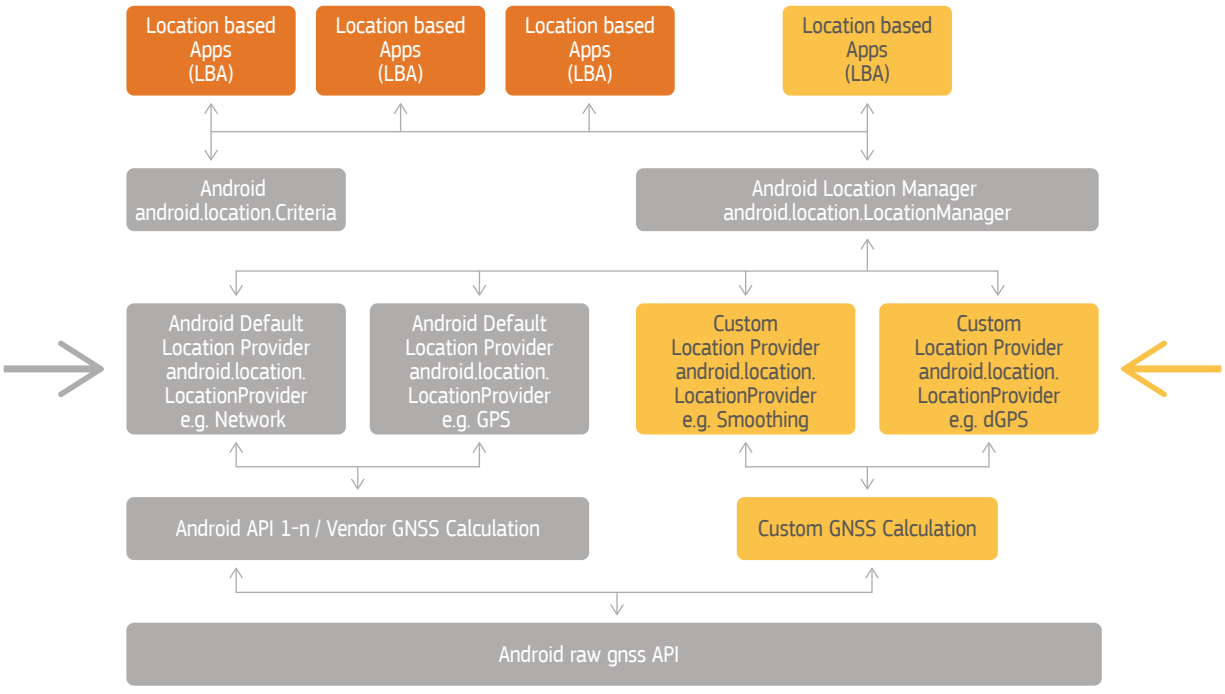

Figure 13: Android GNSS raw data architecture

2.6 Existing Applications and Devices

A small number of Android applications are already utilising GNSS raw measurements. Each of the examples below can be used both as a demonstration of what is possible and as a starting point for further applications.

- **PPP Wizlite** (https://play.google.com/store/apps/details?id=jocs.fr.gnss_ppp) is an Android port of CNES PPP-Wizard project that allows one to calculate PPP position on a Nexus 5X device. It requires internet connection to download corrections.
- **RTCM Converter** (https://play.google.com/store/apps/details?id=jocs.fr.gnss_tortcm) converts Android Raw measurements to the widely used RTCM format, thus making your Android phone a base station.
- **Geo++ Rinex Logger** (https://play.google.com/store/apps/details?id=de.geopp.rinexlogger) records Android Raw measurements in the RINEX file, allowing for further processing in geodetic software.
- **Airbus GNSS Data Collector** shows the satellite information and PVT in real time: C/No, satellites used for the PVT, accuracy, etc. More so, it computes and stores, in real time, the RINEX observation file of GPS, Galileo and GLONASS. A csv file with all the raw measurements is also stored.

A list of devices capable of providing raw measurements is maintained at [RD-25]. Information also includes constellations availability and the availability of the phase measurements. All devices need to run Android 7 or later.

3. OPPORTUNITIES AND PRACTICAL USE OF GNSS RAW MEASUREMENTS

This chapter summarises the positioning techniques available to the Android devices that provide the raw measurements. It starts by describing hardware limitations and current performance. It then describes the most promising positioning techniques, as summarised in Figure 14 and Table 9.

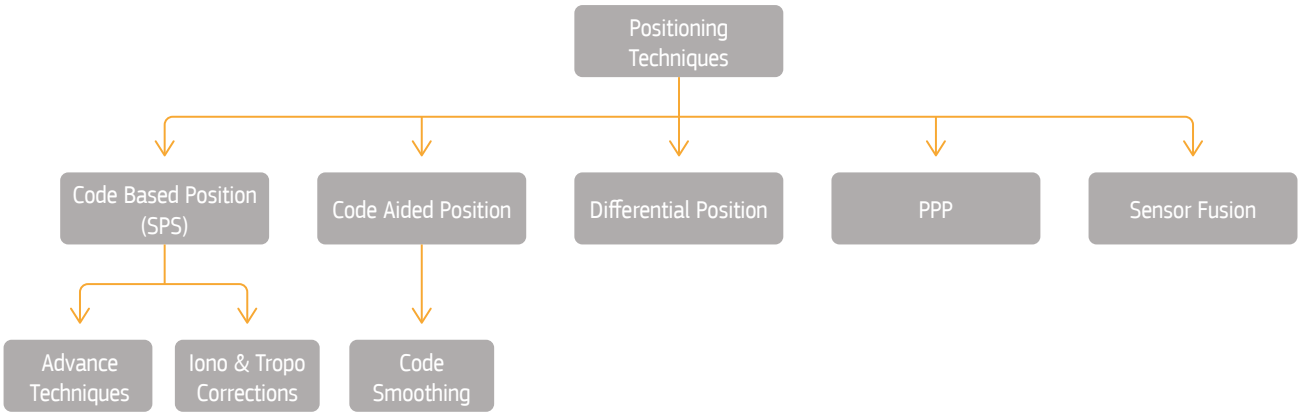

Figure 14: Positioning techniques

Table 9: Positioning techniques comparison

Positioning Technique	Accuracy	Availability	Carrier Phase	External Information	Comments
Code based Single Point Positioning	Medium	High	No	Optional	Local atmospheric models can be implemented
Code Aided	Medium/High	Medium	Optional	Optional	No external data is needed
Differential	High	Low	Optional	Needed	Several reference stations (within 20 km)
PPP	High	Low	Needed	Needed	Long convergence time is needed
Sensor Fusion	High	High	Optional	Needed	Internal device sensors can be used

The architecture of GNSS mobile chipsets prioritises the user experience by minimalizing TTFF (time to first fix) to seconds and by improving position availability and continuity. Recently enabled raw measurements, available via ***android.location*** API, allow us to use the advanced positioning techniques summarised in Figure 14 and Table 9. Using this additional information to reduce errors (outlined in Figure 8) can improve positioning accuracy. Raw measurements also support integration with the other sensors found inside the mobile phone and, with additional observables (such as Doppler and SNR), can provide better accuracy estimation, navigation robustness and an additional layer of security.

3.1 Mobile A-GNSS Chipsets Overview

Mobile phone designs focus on battery optimisation and user experience, which impact both the selection of hardware components and software algorithms. A GNSS/navigation chipset uses A-GNSS architecture, which is different from the chipsets used in other GNSS devices. Phones deploy low-cost components, the most critical for navigation being the built-in antenna and oscillator.

The oscillator is located outside of the single die chip and compensates for temperature variations (TCXO), thus improving frequency stability. It is switched on only when the GNSS chip is used in order to reduce power consumption. When GNSS is off, an alternative, low-power consumption crystal oscillator (XO) maintains the time with an accuracy degradation of 6 seconds per week [RD-10]. A device that has been turned off for a few days will have a coarse a-priori time estimation when started up.

The main aim of A-GNSS is to reduce TTFF, which is achieved by using built-in access to the communication device (such as a 4G modem or WiFi connection). The mobile device will obtain:

1. The approximate estimation of the smartphone position, e.g. through Cell-ID positioning, and the initial time, with accuracy depending on the network infrastructure.
2. The assistance data (the Assisted GNSS protocol), including satellite ephemeris, clock correction and ionospheric models (same as the one broadcasted by the GNSS satellites) [RD-10; RD-19]. An extension/alternative is a predictive GNSS ephemerides [RD-20] or long term orbit products, which allow the receiver to compute the future positions of the satellites for days, even weeks, into the future without an active network connection.
3. Improving reference frequency by calibrating the local oscillator in the phone. It is calibrated by the signal received from the cell tower, whose frequency is typically known to within 50 ppb, and the mobile phone carrier frequency known to within 100 ppb [RD-10].

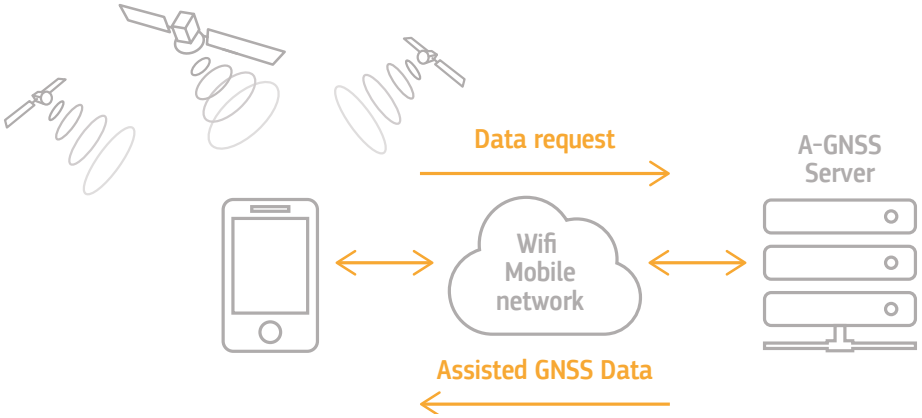

Figure 15: Assisted GNSS

This information increases the receiver's sensitivity and provides a first position fix within a few seconds, even in degraded conditions (urban or light indoor environments), as all information is available at the start. The knowledge of ephemeris data also increases the availability and the accuracy of the position fix in harsh environments. The downside is that the interference between neighbouring radio-transmitters, which are often cluttered within a small space inside the smartphone and collocated in the same chip, increases the complexity of the GNSS chip architecture and can worsen its performance.

The typical antenna used in mobile phones is an inexpensive, PIFA (Planar Inverted-F Antenna), usually a microstrip antenna built using a PCB piece. It is suitable for any constellation in the L1 frequency band, such as GPS/Galileo, GLONASS/Beidou. However, its linear polarization (instead of circular) and the directivity of the radiation pattern leads to several dBs of signal loss. Particularly, the directivity can heavily attenuate some satellites while properly receiving the signal from others. The antenna location is dictated more by the smartphone's design than by RF constraints. This can result in sub-optimal locations where the interaction with user's hand further weakens the reception of GNSS signals. The relative loss, with respect to a standard patch antenna, is estimated to be around 11dB with highly irregular gain pattern [RD-21; RD-22].

The phone also deploys the duty cycle technique to reduce power consumption (discussed in section 2.4.3.1), which impacts oscillators and leads to carrier phase tracking discontinuity, such as when several cycle slips occur between two consecutive measurements, making the use of such advance phase techniques as Real Time Kinematic (RTK) or Precise Point Positioning (PPP) very difficult.

Hardware design choices impact the quality of the GNSS raw data, especially carrier phase measurements, thus directly affecting the performance of the positioning. Due to increased levels of multipath and a higher probability of fading, this effect will be even worse in the urban environment, where most phones are used.

3.2 Baseline Performance – Code Positioning

To visualise the hardware limitations described in the previous section, let's consider a dynamic scenario where the Android device is placed on the dashboard of a moving car. The results of kinematic trials are shown in Figure 16, where the thick yellow line is the truth provided by the tactical grade IMU, the blue line represents the Android solution obtained from a Nexus 9 tablet without data connection and using GPS and GLONASS constellations. The green line is a code-only solution obtained using the low-cost single-frequency receiver with multipath-resistant geodetic antenna. The main contributor to difference between blue and green lines is mainly the linearly polarised antenna, which is poor at removing multipath in the urban environment (left side image). In the clear rural environment (right side), Android can maintain comparable performance with a low-cost GNSS receiver if no multipath is present and it is being visibly affected by the stray signals from the building on the left.

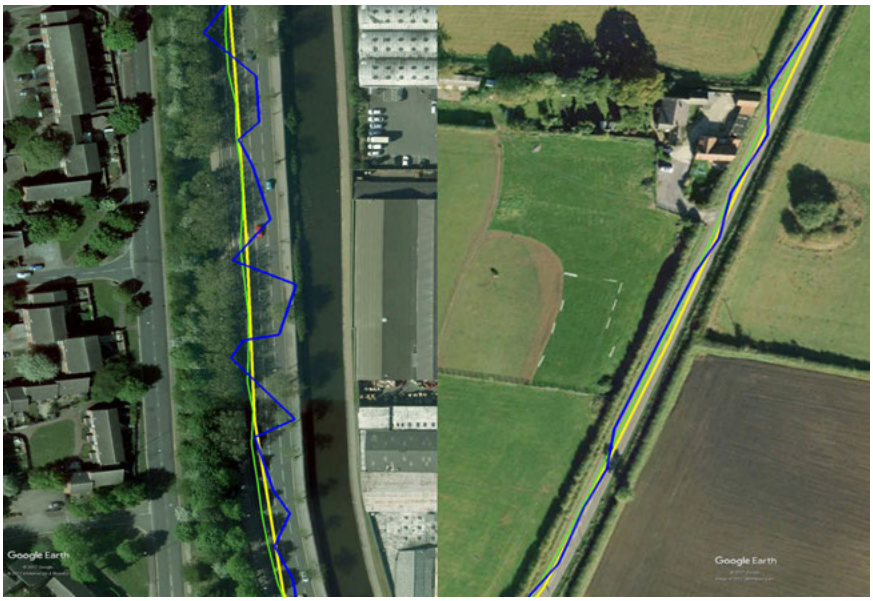

Figure 16: Kinematic test results

[These results have been computed and provided by the University of Nottingham]

Demonstrated Android performance is also being affected by the device's position on the dashboard behind the vehicle windshield. Figure 17 show the difference in terms of satellites used for the PVT solution, between antenna located on the roof and inside a vehicle. Data was collected in separate six-hour scenarios with a mix of motorway and urban environments. The roof solution outperforms the dashboard one, especially during the motorway stretches of the route that happen at the beginning and the end of the data collection, where the difference is an almost doubling in the number of satellites. This is mostly due to vehicle structure obstruction and additional attenuation (and multipath) of the received signal. Interestingly, in the limited sky scenario, such as bridges on the motorway and urban canyons, a smartphone can occasionally see more satellites. This could be the effect of already described assistance data, leading to the faster TTFF and a possible higher sensitivity at a receiver level.

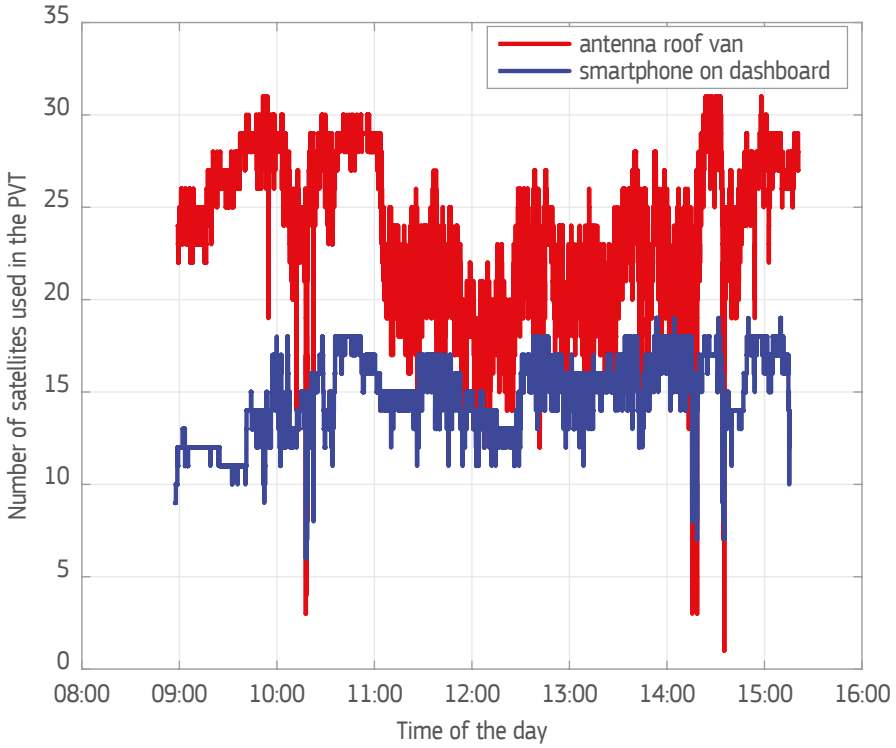

Figure 17: Number of satellites used in PVT (GPS+GAL+GLO) for a mass-market receiver connected to an antenna on the roof of the van and a smartphone installed on the van dashboard

[These results has been computed and provided by ESA]

3.3 Improving Position

3.3.1 Multiple constellations

Let's now focus on the starting motorway section to visualise the benefit of using multiple constellations. Figure 18 compares 3D instantaneous position error of the PVT solution calculated first using GPS only (black dots) with the PVT from using three constellations (GPS, GLONASS, Galileo). For simplicity, we will visualise an accuracy increase (error decrease) with a green dot and the accuracy decrease by a red one.

While the minimum satellite requirement is four, to obtain a reliable positioning, especially in the presence of noise and obstruction, we need 8-10 satellites. In case of limited sky visibility, common in urban areas, this can only be achieved using a multi-GNSS solution. This is visualised by the green circles that cover the top part of the graph - the area of poor GPS performance.

In open sky conditions, when 8-10 satellites can be obtained from a single constellation, just increasing their number will not lead to better accuracy. This is mostly due to the build-up of noise and multipath effect, as demonstrated by the cluster of red circles at the bottom of the graph. In this case, pre-selection of the best satellites for PVT would be recommended over increasing the overall number above 10. It should also be noted that the ISB between the systems needs to be properly estimated.

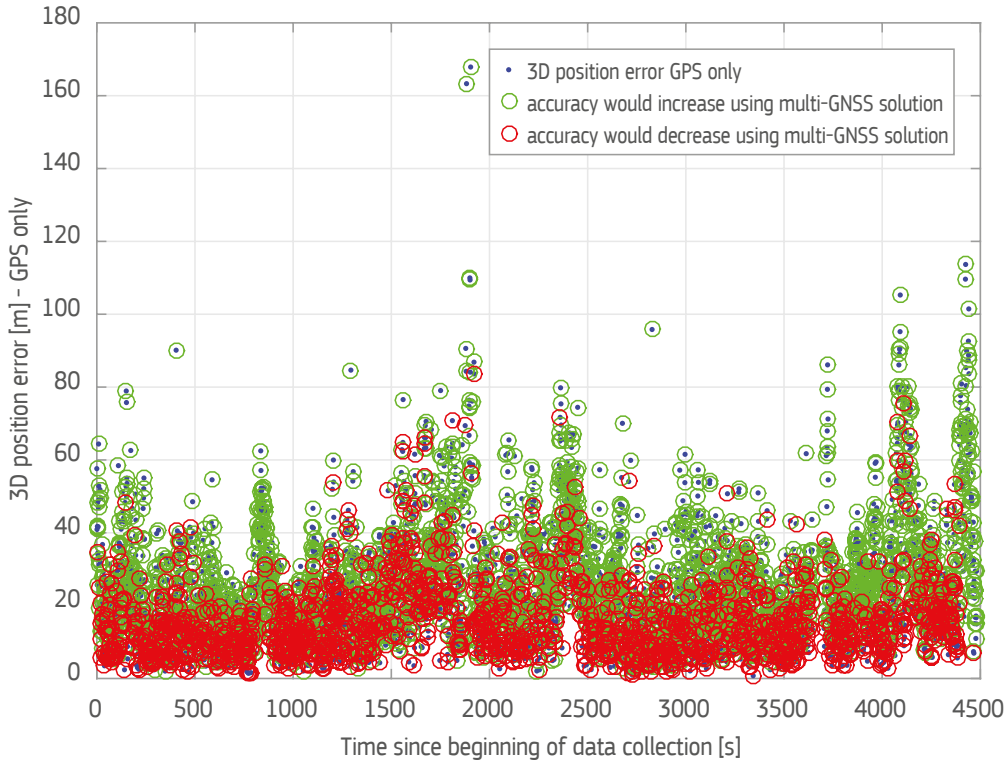

Figure 18: 3D position error with GPS only and GPS + GAL + GLO

[These results has been computed and provided by ESA]

Table 10 quantifies the accuracy and availability of the position for all the constellation combinations, demonstrating the benefit of using multi-GNSS. All data has been processed in the open source RTKLib package [RD-36], using kinematic Android data.

Table 10[2]: Impact of multi-constellation in PVT accuracy and availability

PVT configuration	Horizontal confidence level [meters]		
	68%	95%	Availability
GPS	13.36	25.51	97.79%
GPS + GAL	12.48	23.78	98.04%
GPS + GLO + GAL	11.24	21.57	98.30%
GPS + GLO + GAL + BEI	11.17	21.44	98.30%

[These results have been computed and provided by ESA]

3.3.2 Using information inside chipsets – Doppler smoothing of the code observables

To improve static positioning accuracy, we can use the Hatch Filter to smooth code observation [RD-23].Traditionally, we use carrier phase, but duty cycle introduces repeatable cycle slips that require filter reset. Instead, we can use Doppler measurements that are more resistant to a duty cycle. Figure 19 shows a comparison of a static PVT obtained from raw measurement data using code only and Doppler smoothing, with different integration times. By using a smoothing window of 20 seconds, the 3D error (at 68% confidence level) is reduced by almost a factor of three, from 19.7 metres to 7.6 metres.

2 Since both Galileo and Beidou are not yet in Full Operational Capability, better performance is expected when they are fully deployed.

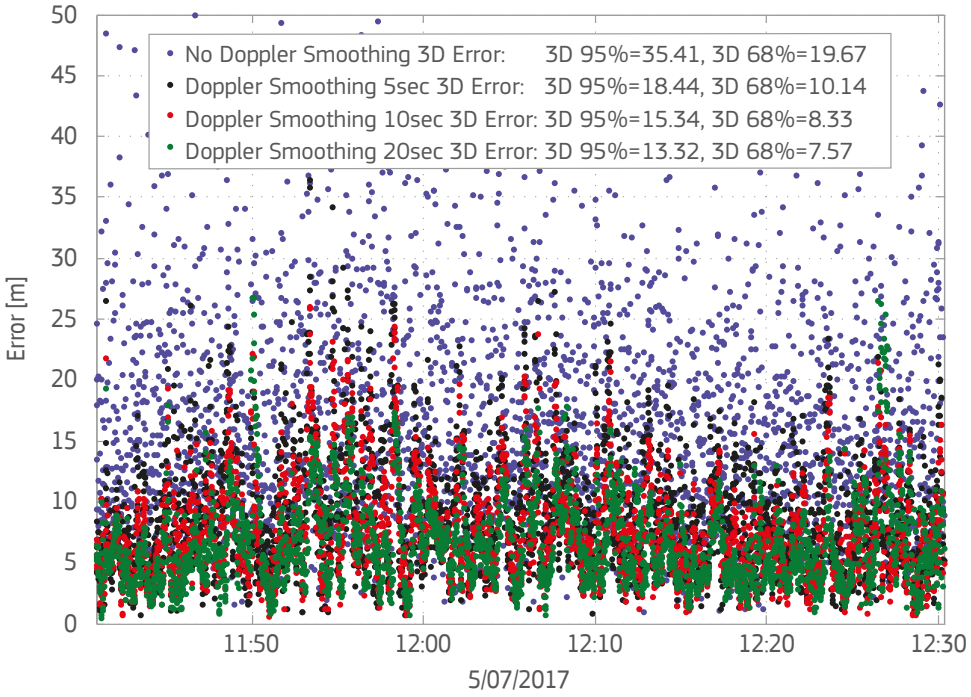

Figure 19: GPS + GLO + GAL PVT performance comparison – Doppler Smoothing

[These results have been computed and provided by ESA]

To assess this method in the kinematic scenario, let`s once again consider the motorway scenario with the 20 seconds smoothing window, which was selected based on the good results achieved with the static scenario. Smoothing increased accuracy, albeit smaller than in the static scenario, leading to a much smoother path, even in the highly dynamic motorway scenario (Figure 20). The reference trajectory (green) has been calculated with a high-end GNSS receiver.

Figure 20: PVT kinematic solutions – Raw code (left) ad doppler smoothed (right) measurements

Table 11[2]: PVT kinematic performance comparison – Doppler Smoothing

Multi-GNSS configurations		Horizontal confidence level [meters]		
		68%	95%	Availability
Doppler Smoothing 20sec	PVT GPS	9.38	18.87	97.79%
	PVT GPS + GAL	9.21	18.63	98.04%
	PVT GPS + GLO + GAL	7.72	16.65	98.30%
	PVT GPS + GLO + GAL + BEI	7.69	16.53	98.30%
No Doppler Smoothing	PVT GPS	13.36	25.51	97.79%
	PVT GPS + GAL	12.48	23.78	98.04%
	PVT GPS + GLO + GAL	11.24	21.57	98.30%
	PVT GPS + GLO + GAL + BEI	11.17	21.44	98.30%

[These results have been computed and provided by ESA]

3.4 Taking It Beyond The Phone - Differential Observations

The minimum requirement for successful positioning is four GNSS satellites from a single constellation (as we need to solve for position and internal clock offset). Primary GNSS error sources are satellite-based (orbits and clock), atmosphere (ionospheric and tropospheric delays), noise and local effects (including multipath), for more see Figure 8. Differential observation improves positioning accuracy by providing external corrections to those errors, but usually does not correct local effects such as multipath or interference. There are several techniques for doing so, with the most commonly used being DGNSS (or DGPS, Differential GNSS/GPS) for code solution and the Real Time Kinematics (RTK), Network RTK and Precise Point Positioning (PPP) for the carrier phase solutions. The most accurate results can be obtained when using carrier phase in RTK and PPP. Both require a good quality carrier phase and, ideally, a static convergence time to solve the ambiguities and increase the accuracy. Convergence time is considerably longer with PPP, but its data requirement is more flexible.

Corrections are transmitted via radio or the internet, but do not provide integrity information. Instead, this is provided by the large area SBAS services such as EGNOS, which also provide a Klobuchar-like ionospheric correction model tailored for Europe and Africa. This model can be improved by a more localised model like, for example, the bespoke ionospheric correction model for Northern Adriatic for quiet space weather conditions in the summer time [RD-24].

Table 12 and Figure 21 show the difference between processing the static raw observables using a PPP algorithm (blue) and a device based code-based positioning solution (red). Reference location is indicated by a black cross.

2 Since both Galileo and Beidou are not yet in Full Operational Capability, better performance is expected when they are fully deployed.

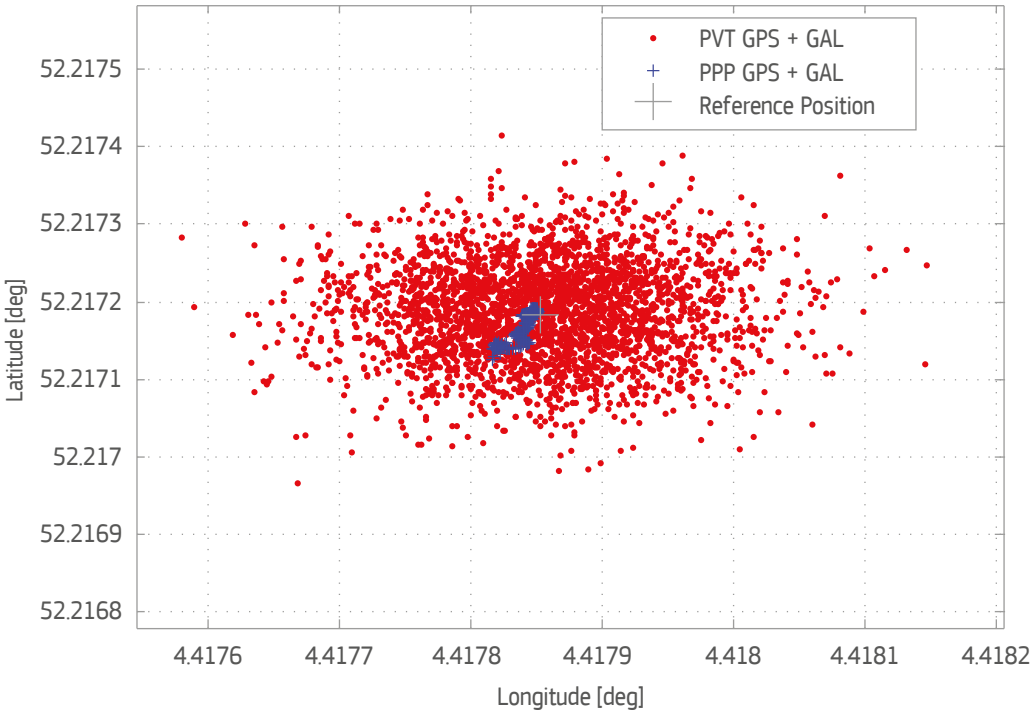

Figure 21: PPP vs PVT solutions

[These results have been computed and provided by ESA]

Table 12[2]: Impact of PPP in user accuracy [m]

PPP vs PVT configurations	Static Scenario				
	H68%	H95%	V68%	V95%	Availability
PVT GPS + GAL	9.18	15.43	17.43	34.67	99.70%
PPP GPS + GAL	0.90	2.35	0.62	1.43	99.70%

[These results have been computed and provided by ESA]

An important improvement will be the introduction of dual frequency chipsets, which on top of L1/E1 will provide the GPS L5 and Galileo E5 signals that are much more resistant to multipath. Initial trials show improved performance and the cycle slip detection and correction [[RD-32]].

3.4.1 Duty cycle

The duty cycle, deployed by the phone to reduce power consumption, affects the oscillator and introduces the carrier phase tracking discontinuity. Figure 22 shows the static data, collected without the active cycle (function was turned off). More than 90% of the phase measurement is valid, allowing for the use of an RTK or PPP algorithm. Three satellites (5,21,27) show an unusually high number of cycle slips, which could be due to a combination of low elevation and noise. These have been removed from the calculations.

2 Since both Galileo and Beidou are not yet in Full Operational Capability, better performance is expected when they are fully deployed.

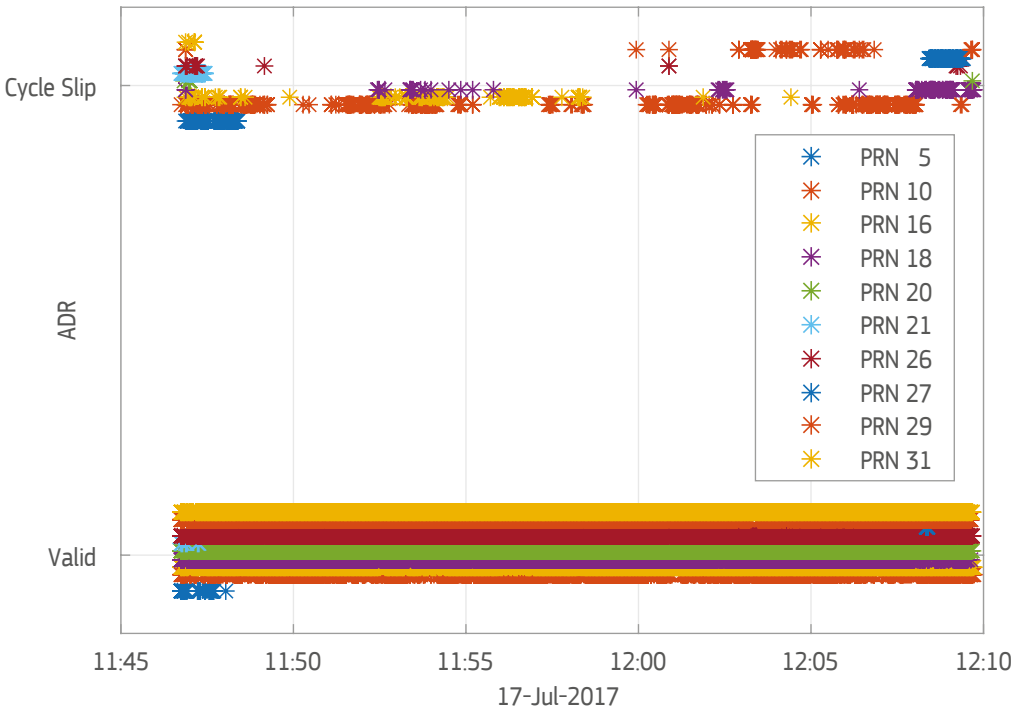

Figure 22: Accumulated delta range validity flag: duty cycle disabled

[These results have been computed and provided by Airbus Defence and Space GmbH]

Table 13: Phase validity (no duty cycle) per satellite

Satellite	Phase Valid [%]	Cycle Slip [%]
Sat 5	26.73	73.26
Sat 10	78.75	21.24
Sat 16	92.96	7.03
Sat 18	94.92	5.07
Sat 20	99.63	0.36
Sat 21	13.95	86.04
Sat 26	98.98	1.01
Sat 27	7.69	92.30
Sat 29	96.51	3.48
Sat 31	99.63	0.36
GPS Aggregated	**92.86**	**7.13**

[These results have been computed and provided by Airbus Defence and Space GmbH]

The results from the smartphone with active duty cycle show only 60% of usable data (Table 14), making it of very limited use for RTK or PPP. This comparison shows that disabling duty cycle would improve the performance and, as such, is recommended for high precision applications, even at the cost of increased power consumption.

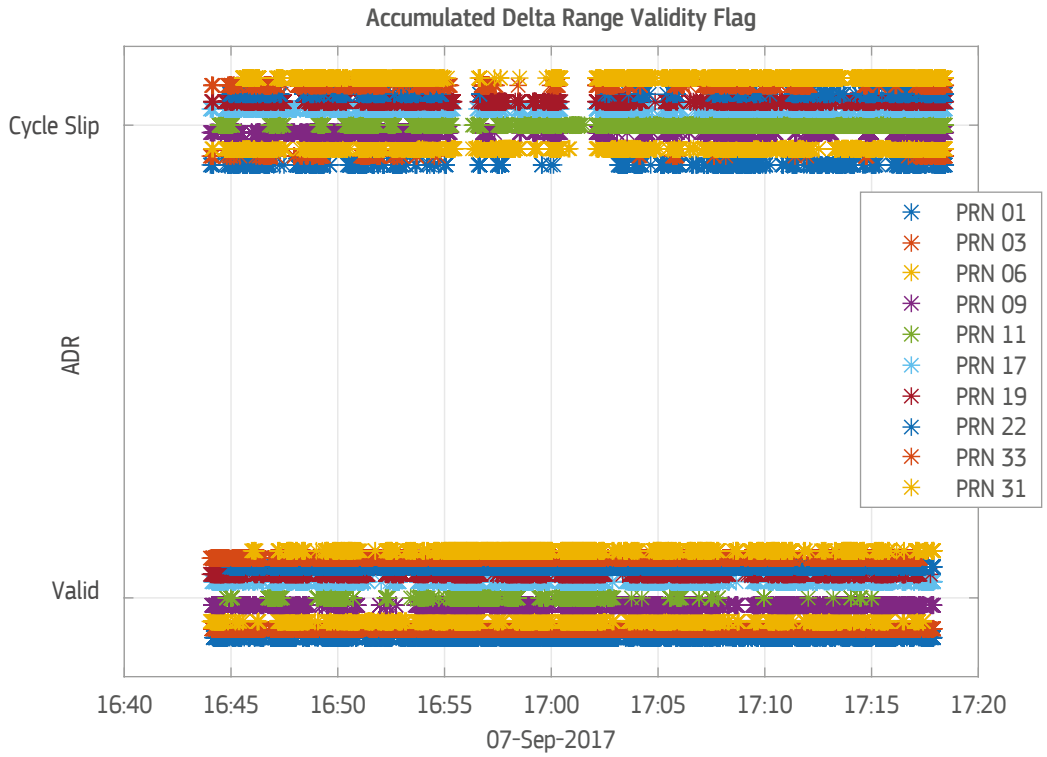

Figure 23: Accumulated delta range validity flag: duty cycle enabled

[These results have been computed and provided by Airbus Defence and Space GmbH]

Table 14: Phase validity per satellite

Satellite	Phase Valid [%]	Cycle Slip [%]
Sat 1	64.05	35.94
Sat 3	85.29	14.70
Sat 6	49.72	50.27
Sat 9	58.81	41.18
Sat 11	32.45	67.54
Sat 17	44.26	55.73
Sat 19	51.76	48.23
Sat 22	76.47	23.52
Sat 33	85.34	14.76
Sat 31	46.23	53.76
GPS Aggregated	**59.44**	**40.55**

3.4.2 Sensor fusion

Google's ***android.gsm.position***'s fused location provider uses GNSS, network location, accelerometer, gyroscope, barometer and magnetometer to provide a position. Apart from accessing GNSS via ***android.position***, you also get access to the following positioning sensors, via ***android.sensors*** [RD-35]:

- GNSS chipsets
- WiFi
- Bluetooth Low Energy (BLE)
- Network location (NLP)
- Inertial Measurement Unit (IMU) – low cost MEMS accelerometer, gyroscope and magnetometer
- Pressure and light sensor
- Step counter (mechanical)

As an example, let's demonstrate the fusion of the IMU – accelerometer, gyroscope and magnetometer. This combination benefits from the short-term accuracy and high update rate of the IMU, while the inertial drift is eliminated by the GNSS positioning. There are classically two architectures (of sensor fusion between GNSS and IMU):

1. The loosely-coupled approach that uses the GNSS position and velocity. This is deployed by the fused positioning inside Android API.
2. The tightly-coupled approach, which is enabled by Android 7 raw measurements and time synchronisation. This can extend position availability and continuity as the combined position requires less than 4 satellites and benefits from additional information.

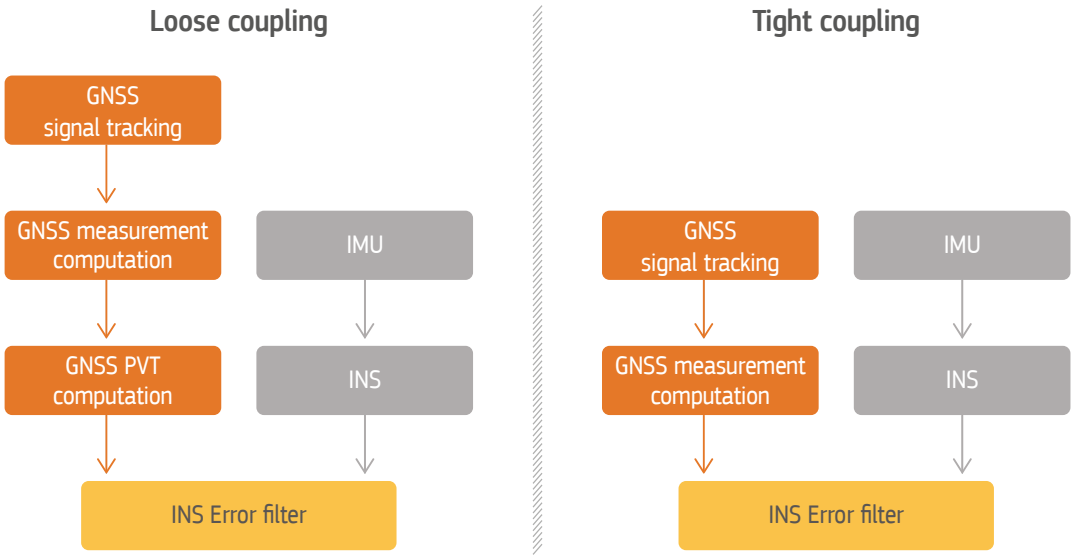

Figure 24: Sensor fusion architecture

3.5 Educational and Scientific Applications

Raw measurements are also very exciting from a scientific perspective. Android devices can be deployed as a grid of sensors, given the reasonably low hardware cost and no need for dedicated firmware development. One of the scientific uses here is the provision of detailed atmosphere monitoring using only one frequency. Other applications and uses includes base station deployment, interference detection and as a part of a smart city's sensors.

Secondly, as the observations are provided in a much coarser form, android devices an excellent teaching tool, allowing one to understand, in detail, how GNSS measurements are obtained and calculated. Observations can also be used to compare solutions from single constellations (e.g. Galileo-only positioning in Figure 25), eliminate specific satellites or visualise worst-scenario performance. Apart from education, the same approach can also be used to test hardware and software solutions, observe performance level and compare this performance with the benchmark solution provided by the chipset itself.

An example is the GNSS Analysis tool provided by Google [RD-18]. It calculates weighted least squares SPS PVT solution using logged raw measurements (static and kinematic), which can then be compared against the known reference. The software provides visualising signal (SNR, skyplot), clocks (satellite, receiver and duty cycle) and measurements (position accuracy, pseudoranges and carrier residuals), providing the user with a dashboard overview of the system.

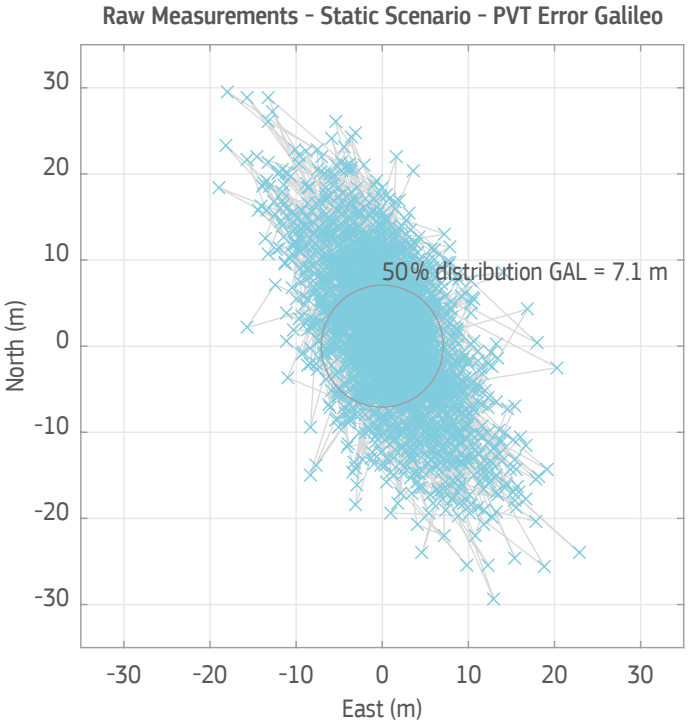

Figure 25[2]: Static Galileo-only positioning error

[These results has been computed and provided by Airbus Defence and Space GmbH]

3.6 High Integrity Solutions

There is a growing interest in low-cost capability for detecting the presence of RF interferences within GNSS signals. This is, principally, required to investigate and diagnose poor GNSS performance and resolve denial of service incidents. A grid of mobile phones could provide a "crowd-sourced" picture of disruptions within the GNSS frequency. Access to raw measurements, in the form of individual pseudoranges and C/No, as well as AGC values, will offer the opportunity to generate new ways to detect RF interferences using the device itself. Furthermore, through the combination of measurement data from multiple Android devices within a region, there will also be the potential to locate the source of the interference. Currently, this may be a niche application, yet access to such capabilities will create opportunities to develop novel services to assist GNSS users, service providers, infrastructure operators and national frequency authorities in protecting the GNSS spectrum.

RAIM is an algorithmic assessment of GNSS signal integrity, providing confidence level assessment of navigation performance that can warn a user if performance is below the set threshold and another solution is required. These are intended for the safety-critical GNSS applications, such as in aviation, marine navigation or, most recently, in the rail industry. With small changes, it can be extended to the Android platform, providing users with better positional confidence, additional verification and warning.

This would benefit mobile devices utilised in safety critical applications for aviation, marine or rail, or when positional estimation is life critical (such as ambulance unit dispatch). Such verification can also provide another layer of security to online transactions, give more clarity in communication and reduce human-introduced errors. Combined with other information, such as sky view, this can be extended to a system predicting areas where your GNSS performance might be compromised [RD-34]. RAIM can also be used in the Volunteered Geographical Information (VGI), where user-generated geographical data requires both confidence in position and Quality Control of collected information [RD-33]. We can also use the same principle to provide the visual warning to the user, informing when a reported Android position may not meet specific task precision requirements, allowing user to make more educated use of positional information.

2 Since both Galileo and Beidou are not yet in Full Operational Capability, better performance is expected when they are fully deployed.

4. THE NEED AND USE CASES FOR HIGHER ACCURACY IN THE MASS MARKET

Currently, over 5 billion GNSS devices are in use around the world, 80% of which are smartphones. More than 50% of applications in the Google Play and Apple stores use location information. Smartphones are increasingly used in applications that are at the border of safety-critical or high precision ones, from navigating a glade or leisure boat to mapping trees for city park management. The higher location accuracy that can be obtained in the mass market will further increase the use of smartphones and wearables in semi-professional applications and enable a new range of consumer applications that are still not possible today.

As demonstrated in Chapter 3, the typical performance of today's mass market mobile devices is in the range of metres to even tens of metres in difficult conditions, such as urban canyons. However, the use of multi-constellation, dual-frequency chipsets and the provision of external information promise to increase this accuracy to sub-metre levels in the near future. Modern mobile phones, deploying system on chip, are more capable than the mainframe computers of 20 years ago. This results in lower costs for deploying and testing new applications with quicker update cycles and without the burden of having to develop hardware. Android raw measurements will provide additional layers of integrity and robust position, enabling the development of robust, reliable and interference-resilient position-based services.

4.1 Main Application Areas to Benefit from Improved Location Accuracy

Many mass market applications can benefit from increased accuracy. A list of examples can be found in Figure 26.

Figure 26: Applications utilising higher location accuracy

4.1.1 Mobile applications

In location-based advertising, engagement rates can increase up to 30% when location data is used to customise creative advertisement [RD-26]. If targeting mobile users located within the vicinity of a particular business, an inaccurate position will deem the campaign ineffective. A 2014 study by the Mobile Marketing Association, which tracked location data accuracy to improve advertisement performance, found that only ⅓ of location-based advertisement requests were accurate to within 50-100 metres of the stated location.

A similar situation applies to all geofencing applications, such as a taxi company constraining their cars to a specific zone, a city defining paid access to the centre, and parents tracking children. In geofencing applications, absolute location accuracy is not critical. What does matter is the ability to locate the user within or outside of the virtual fence.

Here, deliberate interference, such as jamming or spoofing, will be an issue – especially if the geofencing applications are lined with liability or payment (such as location-based parking). As discussed in Chapter 3, the use of raw measurements can also support the integrity check and detect spoofing.

Location accuracy is also important in augmented reality (AR) applications. Augmented reality enhances one's perception of reality with computer-generated images that are superimposed over a user's view of the real-world. Location in three dimensions is the key for the synchronisation of the physical and digital worlds, with accuracy dependent on final application. Some AR applications require a very high degree of synchronisation, meaning higher location accuracy is needed. The global augmented reality market is expected to reach €108 bln by 2022 [RD-01].

There are three categories of augmented reality tools: augmented reality 3D viewers, augmented reality browsers and immersive gaming experiences. An example of the first category can be found in the construction industry. The industry is moving towards using augmented reality to visualise geo-referenced models of construction sites, underground structures, cables and pipes using mobile devices. Here, high location accuracy is a key requirement. For example, if an AR system allows the user to "see" a pipe underground, and the location of that pipe is not accurate enough, contractors might hit the pipe when they start digging. Construction companies need to be able to rely on the tools they use, whether it is their construction management software or new augmented reality device.

4.1.2 Safety-related applications

An example from the safety-related application area where consumer devices are used is mobile health (mhealth). Mobile health covers medical and public health practices supported by mobile devices, including communication and data collection. For the purpose of this white paper, we also include all well-being and fitness applications using GNSS-based wearables. An enhanced location accuracy would benefit, for example, sport and fitness tracking applications, the navigation of visually impaired and physically disabled people, and geo-fencing for Alzheimer's patients. The range of applications in this category is already very wide. With an ageing population, the number of application is also increasing rapidly, especially with the funding available for innovation around social inclusion.

The enhanced location accuracy improves estimation of effort, distance and movement speed in sports like rowing, golf, athletics, soccer training, skiing, etc., allowing one to identify individual performance and implement improvements. Higher-end GNSS receivers are already widely used by professional athletes in various sports. Improved Android accuracy could see the rise of low-cost devices used as a cost-effective alternative for analysing performance [RD-27].

GNSS technology, through auditory and haptic interfaces, allows blind and visually impaired people to better understand their surrounding environment and supports their mobility. Accuracy is needed to allow the user to identify a specific destination or interest point, for example a shop or medical facility they are in front of at certain moment.

GNSS-enabled e112 emergency call is a location-enhanced version of the 112 universal European emergency number, where the location information is automatically provided to the emergency centre with the call itself. According to EENA [RD-30], 70-80% of emergency calls in Europe originate from a mobile phone, but caller location information provided to the emergency services is often inaccurate and delayed. Currently, Cell-ID is the most widely used solution. Use of GNSS provides precise positioning and allows for better access and resources estimation and allocation, thus decreasing response time. Medical research indicates that a one minute reduction in response time improves the odds of survival by 24% [RD-30].

4.1.3 Semi-professional applications

The Internet of Things (IoT) allows physical devices, vehicles, buildings and other objects to be inter-connected and remotely controlled, creating its own infrastructure (mesh network). IoT is expected to rapidly grow across network infrastructures, creating a mesh of inter-connected sensors and technologies, with GNSS providing localisation and timing. It will be a major LBS enabler and, with data from mobile phones, an important source of big data information. This allows us to create very personal computer-generated services and improve positioning accuracy by creating data-based models. Smart city asset management, like smart lighting or garbage management, are just a couple of examples of applications that might benefit from improved accuracy.

The concept of crowd sourcing follows the same principle, collecting information from multiple sources. This approach will benefit from the extra security and integrity offered by raw measurements. While this provides the general public with rapid information collection and map-making, privacy is an issue [RD-29]. Localisation information is highly sensitive and can be unduly exploited, creating risks for individuals and companies alike. The European Union introduced the General Data Protection Regulation (GDPR) - a legal framework harmonising data privacy laws across Europe, including personal data usage in electronic communication and data retention [RD-31]. Rapid developmental of IoT and big data will require further discussion, not only to identify institutional and technical controls, but to create a general understanding about relevant opportunities and challenges [RD-30].

The applications that may benefit from increased accuracy include all workforce and fleet management apps, especially those that protect lone workers in case of emergency notification. Crowdsourced mapping apps, depending on their purpose, may also benefit from sub-metre accuracy as well. Furthermore, a higher level of accuracy available at the smartphone level may also enable the device to be used for less demanding mapping tasks where metre accuracy is sufficient.

ENJOYED READING THE WHITE PAPER? GET IN TOUCH!

Any comment and feedback is more than welcome and should be addressed to market@gsa.europa.eu.
Do you want to join the GSA GNSS Raw Measurements Task Force? Use the same contact above to reach us.

LIST OF FIGURES

LIST OF TABLES

ACRONYMS

API	Application Programming Interface
AR	Augmented reality
C/A code	Coarse Acquisition Code
C/No	Carrier-to-noise-density ratio
F/NAV	Freely accessible Navigation Message
FCN	Frequency Channel Number
GGTO	GPS to Galileo time offset
GLONASST	GLONASS Time
GNSS	Global Navigation Satellite System
GPST	GPS Time
GSA	European GNSS Agency
GST	Galileo System Time
IMU	Inertial Measurement Unit
I/NAV	Integrity Navigation Message
ISB	Inter-System Bias
IoT	Internet of Things
LBS	Location Based Services
LNA	Low Noise Amplifier
NMEA	National Marine Electronics Association
OS	Open Service
OSNMA	Open Service Navigation Message Authentication
PIFA	Planar Inverted-F Antenna
PPP	Precise Point Positioning
PVT	Position Velocity and Time
RAIM	Receiver Autonomous Integrity Monitoring
RINEX	Receiver Independent Exchange Format
RTK	Real Time Kinematic
SoD	Second of Day
SPS	Standard Positioning Service
TAI	Temps Atomique International
TCXO	Temperature-Compensated Crystal Oscillator
TOW	Time of Week
TTFF	Time To First Fix
UT	Universal Time
UTC	Universal Time Coordinated
WN	Week Number
XO	Crystal Oscillator

REFERENCE DOCUMENTS

[RD-01]	GNSS Market Report, Issue 5, GSA. https://www.gsa.europa.eu/system/files/reports/gnss_mr_2017.pdf
[RD-02]	Current and Planned Global and Regional Navigation Satellite Systems and Satellite-based Augmentations Systems. UN OOSA. Vienna, Austria.
[RD-03]	Peter J.G. Teunissen and Oliver Montenbruck, Springer Handbook of Global Navigation Satellite Systems, Springer (2017)
[RD-04]	http://www.navipedia.net/index.php/Code_Based_Positioning_(SPS)
[RD-05]	Sanz Subirana, J et al. (2013). "GNSS Data Processing – Vol. I: Fundamentals and Algorithms". European Space Agency (ESA). Nordwijk, The Netherlands. Available at: http://bit.ly/1QV4KAL, accessed on 5 February, 2018
[RD-06]	Petrowski, Ivan G. (2014). "GPS, GLONASS, Galileo and Beidou for Mobile Devices". Cambridge University Press. Cambridge, UK
[RD-07]	Linty, Nicola & Lo Presti, Letizia & Dovis, Fabio & Crosta, Paolo. (2014). Performance analysis of duty-cycle power saving techniques in GNSS mass-market receivers. Record - IEEE PLANS, Position Location and Navigation Symposium. 1096-1104. 10.1109/PLANS.2014.6851479.
[RD-08]	http://www.navipedia.net/index.php/Time_References
[RD-09]	Interface Specification IS-GPS-200, september 2013 available at: http://www.gps.gov/technical/icwg/IS-GPS-200H.pdf
[RD-10]	Van Diggelen, Frank Stephen Tromp. 2009. A-GPS: assisted GPS, GNSS, and SBAS. Boston: Artech House.
[RD-11]	Code Tracking and Pseudoranges, Inside GNSS, January/February 2012 available at: http://www.insidegnss.com/auto/IGM_janfeb12-Solutions.pdf
[RD-12]	Gustafsson, F. (2010). "Statistical sensor fusion". Studentlitteratur. Linkoeping University. Linkoeping, Sweden
[RD-13]	Rahemi, N et al. (2014). "Accurate Solution of Navigation Equations in GPS Receivers for Very High Velocities Using Pseudorange Measurements". Adv in Aerospace Eng, 2014, Article ID 435891, 8 pages
[RD-14]	Brown, R G i Hwang, P W C. (2009). "Introduction to Random Signals and Applied Kalman Filtering (3rd ed)". John Wiley & Sons. New York, NY
[RD-16]	Petrowski, Ivan G. (2014). "GPS, GLONASS, Galileo and Beidou for Mobile Devices". Cambridge University Press. Cambridge, UK
[RD-17]	GNSS Time Offset, Inside GNSS, September/October 2007 available at: http://www.insidegnss.com/auto/SepOct07-GNSS_time_offset.pdf
[RD-18]	GPS Measurement Tools, Google available at: https://github.com/google/gps-measurement-tools
[RD-19]	Nicolas Couronneau (2013) Performance analysis of assisted-GNSS receivers, PhD, University of Cambridge
[RD-20]	http://rxnetworks.com/location-io/predicted--gnss-ephemeris/
[RD-21]	Pesyna, K. M.; Heath, R. W. & Humphreys, T. E. Centimeter Positioning with a Smartphone-Quality GNSS Antenna Proceedings of the 2014 ION GNSS Conference, 2014

[RD-22]	Humphreys, T. E.; Murrian, M.; van Diggelen, F.; Podshivalov, S. & Kenneth M. Pesyna, J. On the Feasibility of cm-Accurate Positioning via a Smartphone's Antenna and GNSS Chip Proceedings of the 2016 IEEE/ION PLANS Conference, 2016
[RD-23]	http://www.navipedia.net/index.php/Carrier-smoothing_of_code_pseudoranges
[RD-24]	Brcic, D. (2015). A model of non-specific daily pattern of the satellite positioning signal ionospheric delay (PhDthesis). Faculty of Maritime Studies, University of Rijeka. Rijeka, Croatia
[RD-25]	https://developer.android.com/guide/topics/sensors/gnss.html
[RD-26]	Location Score Index: Q2 2016, Mobile Advertising's Guide to Location Accuracy, Thinknear.
[RD-27]	https://www.peakendurancesport.com/endurance-products-and-technology/electronics- and-software/sports-equipment-gps-best-route-performance-analysis/
[RD-28]	http://gpsworld.com/nvs-technologies-selected-by-alberding-for-sub-meter-gnss-receiver/
[RD-29]	Gilad Rosner, Privacy and the Internet of Things (2017), O'Reilly
[RD-30]	http://emj.bmj.com/content/early/2010/08/25/emj.2009.086363
[RD-31]	http://www.eugdpr.org/
[RD-32]	Riley, S.; Lentz, W. & Clare, A. (2017) *On the Path to Precision – Observations with Android GNSS Observable 2017 ION GNSS Conference*
[RD-33]	Leibovici et al (2017) Earth Observation for Citizen Science validation, or, Citizen Science for Earth Observation validation? Role of the Quality Assurance of Volunteered Observations
[RD-34]	Roberts et al (2017) Predictive Intelligence for a Rail Traffic Management System, ION GNSS+ 2017
[RD-35]	https://developer.android.com/guide/topics/sensors/sensors_overview.html
[RD-36]	http://www.rtklib.com/

TS-07-17-174-EN-N

The European GNSS Agency (GSA)

The GSA's mission is to support European Union objectives and achieve the highest return on European GNSS investment, in terms of benefits to users and economic growth and competitiveness by:

- Designing and enabling services that fully respond to user needs, while continuously improving the European GNSS services and infrastructures;
- Managing the provision of quality services that ensure user satisfaction in the most cost-efficient manner;
- Engaging market stakeholders to develop innovative and effective applications, added-value services and user technology that promote the achievement of full European GNSS adoption;
- Ensuring that European GNSS services and operations are thoroughly secure, safe and accessible.

Galileo

Galileo is the European Union's Global Satellite Navigation System (GNSS) that provides accurate positioning and timing information. Galileo is a programme under civilian control and its services can be used for a broad range of applications. It is autonomous but also interoperable with existing satellite navigation systems. Galileo Full Operational Capability (FOC) is planned for 2020.

On 15 December 2016, the Declaration of Initial Services marked the beginning of Galileo's operational phase. This means that anyone with a Galileo-enabled device is now, in combination with other GNSS systems, able to use signals provided by Galileo's global satellite constellation for positioning, navigation and timing.

The fully deployed Galileo system will provide four different services:

- Open Service (OS) that targets the mass market. The OS will offer either single (E1) or dual frequency (E1 and E5) and will be the first to broadcast authentication data through its Navigation Message Authentication (OS NMA).
- Commercial Service (CS) will complement the OS and provide a higher positioning accuracy and improved signal robustness due to authentication. It will be broadcast on E6 frequency.
- Public Regulated Service (PRS) will be dedicated to government-authorised users. It will be encrypted and secured against jamming and spoofing.
- Search and Rescue (SAR) service will allow near real time alert localisation and message detection, higher beacon localisation accuracy, high availability and global satellite coverage. It will have a return link, which is unique to Galileo, and will reduce the rate of false alerts.

useGALILEO.eu

Mass-market devices containing a Galileo-enabled chipset, such as smartphones or vehicle navigation devices, can use Galileo signals for positioning, navigation and timing. The www.useGALILEO.eu tool helps you fnd Galileo-enabled chipsets, smartphones, wearables and tracking devices.

www.gsa.europa.eu

@EU_GNSS

EuropeanGnssAgency

European-GNSS-Agency

European GNSS Agency

EU_GNSS

European
Global Navigation
Satellite Systems
Agency

GALILEO EGNOS
NAVIGATION SOLUTIONS
POWERED BY EUROPE

ISBN 978-92-9206-033-6
doi:10.2878/449581