HASlib: An Open-Source Decoder for the Galileo High Accuracy Service

Oliver Horst, Martti Kirkko-Jaakkola, Tuomo Malkamäki, Sanna Kaasalainen, *Finnish Geospatial Research Institute*
Ignacio Fernández-Hernández, *European Commission*
Adrián Chamorro Moreno, Simón Cancela Díaz, *GMV*

BIOGRAPHY

Oliver Horst received his M.Sc. degree from Aalto University, Finland, and Technical University of Berlin, Germany, in 2021. In 2021, he worked as a Research Assistant at FGI. Since 2022, he works full time at Kinexon, Munich, Germany. His research interests include positioning technology and multi-agent systems.

Martti Kirkko-Jaakkola received his M.Sc. and D.Sc. (Tech.) degrees from Tampere University of Technology, Finland, in 2008 and 2013, respectively. From 2013 till 2020 he worked full time at FGI as a Senior Research Scientist and Research Manager. As of 2021, his main occupation is with Nordic Inertial Oy, Akaa, Finland, with a secondary affiliation with the FGI. His research interests include low-cost GNSS and inertial navigation systems.

Tuomo Malkamäki received his M.Sc. and Lic.Sc. (Tech.) degrees from Aalto University, Finland, in 2010 and 2013, respectively. Since 2015, he has worked at FGI as Research Scientist and Research Group Manager. His research interests include navigation and positioning technology and sensors suitable for GNSS degraded environments.

Sanna Kaasalainen is a professor and director of the Department of navigation and positioning at the FGI of the National Land Survey. Her research interests are resilient PNT, situational awareness, and optical sensors. She also has research experience in remote sensing, environmental studies, laser scanning applications, and astronomy.

Ignacio Fernández-Hernández has led Galileo high accuracy and authentication services over the last years at the European Commission, DG DEFIS. He is an ICAI engineer and has a PhD degree in Electronic Systems from Aalborg University. He is also the latest recipient of ION's Thurlow Award.

Adrián Chamorro Moreno holds a MSc in Space Systems by the Technical University of Madrid. He has 5 years of experience in the field of GNSS high accuracy positioning in GMV. He is currently the technical leader of the PAULA project from European Commission and has collaborated in some other projects of high accuracy positioning for automotive such as the development of the safe and accurate Positioning Engine for BMW's Autonomous Driving module.

Simón Cancela Díaz holds an MSc in Advanced Mathematics by the Universidad Complutense de Madrid. He joined GMV in 2015 and has been involved in projects related with the design and development of GNSS authentication, anti-spoofing algorithms and high accuracy positioning. He is now focused on activities related to Galileo High Accuracy and Authentication services.

ABSTRACT

The upcoming Galileo High Accuracy Service (HAS) will provide decimeter-level Precise Point Positioning (PPP) corrections free of charge. The HAS corrections are High-Parity Vertical Reed–Solomon encoded on the Galileo E6 navigation data in a format reminiscent of but not identical to many State Space Representation (SSR) formats. In this paper we present the HASlib which is an open-source library intended to facilitate the use of the HAS in R&D purposes. HASlib decodes the E6 navigation data frames and outputs the corrections in RTCM 3 or IGS SSR formats, as configured by the user. This makes it possible to use HAS with existing GNSS receivers and PPP engines that do not natively support the HAS formats. In the first release, BINEX and Septentrio SBF are supported as input formats. HASlib is written in Python language and its design is modular, which implies that support for other input or output formats is straightforward to implement. This paper describes the architecture and main features of the library. Furthermore, proof-of-concept results are presented to show that the library can be run in real time on a miniature desktop computer and that the output is compatible with the open-source PPP-Wizard package for PPP computation.

I. INTRODUCTION

In order to reach a sub-meter positioning accuracy, a GNSS receiver needs to apply some kind of correction data to compensate for deterministic biases such as satellite orbit and clock errors. This is typically done by connecting to a third party such as a Differential GNSS (DGNSS) service or a Real-Time Kinematic (RTK) network which requires a separate data link, e.g., cellular connectivity, and often a service subscription. Satellite-based augmentation systems (SBASs) use geostationary satellites to broadcast the corrections on a signal similar to the actual GNSS signals, thus eliminating the need for a separate data link, but the performance guarantees of SBASs exceed 1 meter; for instance, the 2-sigma horizontal accuracy of the European Geostationary

Navigation Overlay System Service (EGNOS) is 3 meters for the worst user location (GSA, 2017).

The upcoming Galileo High Accuracy Service (HAS) will enable stand-alone GNSS receivers to achieve a 20 cm horizontal position accuracy at 2-sigma confidence (GSA, 2020). This is implemented by broadcasting real-time correction data on the Galileo E6 signal. This signal is not encrypted, implying that any E6-capable receiver can benefit from the HAS free of charge. Obviously, the broadcast correction data must be valid globally, therefore, the HAS is not differential but provides corrections for the individual error components. In other words, the HAS is intended for Precise Point Positioning (PPP). Receivers that do not support the E6 band may download the HAS corrections from the Internet the same way as they would connect to an external correction service provider (GSA, 2020).

As opposed to DGNSS and RTK where Radio Technical Commission for Maritime Services (RTCM) standards are widely supported, PPP does not have a globally adopted standard for the provision of correction data. The standardization of the RTCM State Space Representation (SSR) format RTCM (2013) has been pending for several years, and alternatives have been proposed in the meantime, many of them very similar to RTCM SSR. Galileo HAS corrections are provided in a format of its own, reminiscent of the Compact SSR format; furthermore, they are encoded into so-called High-Parity Vertical Reed–Solomon (HPVRS) codes for optimizing the HAS message reception from multiple satellites (Fernández-Hernández et al., 2020). As a result, legacy GNSS receivers and PPP processing software are unlikely to be able to benefit from the HAS without a software upgrade.

To facilitate the use of Galileo HAS with legacy receivers and software, we are releasing an open-source application that decodes the HAS HPVRS-encoded messages navigation data frames and converts the corrections to either RTCM-SSR or IGS-SSR (user configurable). Using this library, HAS corrections can be input to any GNSS receiver or PPP processing software that supports RTCM-SSR or IGS-SSR correction data streams even if they do not natively support HAS. The first release of the application supports Septentrio SBF and Binary Exchange Format (BINEX) inputs, and the output can be sent to a TCP socket, file, or the standard output. The application has been written in the Python language using a modular class structure so that it can be easily customized and extended.

To validate the implemented interfaces, we show a proof-of-concept result of processing Galileo HAS corrections with the open-source PPP-Wizard package (PPP-Wizard, 2021). A HAS test transmission campaign took place in late 2021, and a commercial off-the-shelf Septentrio PolaRx 5 receiver was used to collect both raw measurements and HAS navigation data frames. The receiver was connected to a static rooftop antenna in Kirkkonummi, Finland.

This paper is based on the work conducted in (Horst, 2021) which is a Master's thesis describing the development of HASlib. At the publication of this paper, the library is released to the general public. The remainder of this paper is organized as follows. First, Section II describes the context of PPP in terms of services, data formats, and existing software. Then, the design and features of the library are presented in Section III and proof-of-concept tests are conducted in Section IV. Finally, Section V concludes the paper.

II. BACKGROUND

Sub-meter GNSS positioning requires the availability of a service to provide correction data. Such data can be expressed in a variety of formats. Furthermore, a processing engine supporting the provided correction data is needed. These positioning services, data formats, and open-source PPP libraries are presented in the following sections.

1. Precise GNSS Positioning Services

Centimeter-level GNSS positioning accuracies have been common in professional applications for decades, and the increasing availability of dual-frequency chipsets in the mass market can be expected to keep the demand for precise GNSS positioning services growing.

Any GNSS receiver is subject to several ranging errors originating from the satellite (orbit and clock errors as well as signal biases), atmosphere (ionosphere and troposphere), and the reception environment (multipath). Since most of these errors are spatially correlated, their individual contributions can be modeled and isolated using a network of reference receivers at known locations, which is the backbone of a precise GNSS positioning service. The resolved error information is transmitted to the users in the form of correction data. Traditionally, the correction data were provided in the *observation space*, i.e., as raw reference station measurements and/or lump-sum correction values for measurements, for differential GNSS or Real-Time Kinematic users. More recently, the *state space representation* (SSR), providing information on the behavior of the individual error sources instead of their total sum, has gained a lot of interest, one of the drivers being scalability: An observation space correction service needs to tailor the data to the user's location before transmission; in contrast, SSR corrections can be broadcast to a wider area, putting the computational burden on the user side.

SSR corrections can be employed for PPP, and such data can be obtained from various sources. For instance, the International GNSS Service (IGS) provides correction data free of charge, and there exist various commercial service providers such as Trimble, TerraStar, and PointPerfect (formerly known as SAPCORDA). The Galileo HAS will eliminate the need for a third party or external data link by broadcasting SSR corrections on the Galileo E6 signals. Notwithstanding, the correction data will also be accessible online for receivers that do not support the E6 band but do have Internet connectivity GSA (2020).

The accuracy and area of validity depend on the coverage and density of the reference receiver network. Consequently, there exist differences in the service level and coverage area between PPP services: for instance, IGS data are globally applicable while regional service providers may offer, e.g., higher resolution atmospheric models. Galileo HAS will have two different service levels (GSA, 2020): although the HAS is globally available, atmospheric corrections will be provided for the European coverage area, which improves the convergence time.

2. Formats for State-Space Representation Data

There exists a number of published open and proprietary formats created for the purpose of efficient dissemination of SSR information. The constraints of the individual application areas they are developed for play a large role in the design of them. However, data efficiency can generally be identified as a common design goal which plays an especially large role if satellites are used as a transmitting medium, due to a smaller bandwidth availability.

Vana et al. (2019) and Hirokawa and Fernández-Hernández (2020) identified the most relevant SSR formats to be RTCM 3.0, Compact SSR (CSSR), and SPARTN (formerly SAPA). Furthermore, the open IGS SSR standard by the International GNSS Service (IGS) and the preliminary format of the Galileo HAS will be presented alongside. In the context of cellular services, 3GPP LPP is aimed at the mobile network based dissemination of correction data via LTE or 5G; this format is limited to a single application area and is excluded from the scope of this paper.

The most relevant features supported by the different SSR formats are summarized in Table 1. Some of the features have been in the draft stage for RTCM 3, which are pointed out. It can be noted that, generally, the differences between the formats in terms of information content are minuscule. The most critical features of the HAS correction data can be converted to any of the other formats; the only exception is SPARTN which does not support the Galileo constellation.

Table 1: Summary of features supported by various SSR formats. Parentheses indicate features that are planned or optional.

Feature		HAS	RTCM 3	IGS	SPARTN	CSSR
Constellations	GPS	✓	✓	✓	✓	✓
	GLONASS		✓	✓	✓	✓
	Galileo	✓	✓	✓		✓
	BeiDou		✓	✓		✓
Orbit corrections	Constant	✓	✓	✓	✓	✓
	1st/2nd order		✓	✓		
Clock corrections	Constant	✓	✓	✓	✓	✓
	1st/2nd order		✓	✓		
	Multiplier	✓				
	High rate	(✓)	✓	✓		
Code bias		✓	✓	✓	✓	✓
Phase bias		✓	draft	✓	✓	✓
	Yaw		draft	✓	(✓)	
	Integer		draft	✓		
	Wide lane		draft	✓		
URA		(✓)	✓	✓	✓	✓
Ionosphere		(✓)	draft	✓	✓	✓
Troposphere					✓	✓
Do-not-use flags		✓			✓	

3. Open-Source PPP Software Packages

There exist various options for PPP processing, ranging from receiver built-in features to commercial and open-source software. Since the purpose of HASlib is to facilitate access to Galileo HAS, we wish to evaluate its functionality with a PPP package that is also available as open source. Possible alternatives are identified in this section.

Probably the most popular open-source suite for precise GNSS processing is RTKLIB (Takasu, 2013). Its first release took place in 2006 and it is still being developed, although the documentation is severely running behind the most recent updates. Although the name suggests otherwise, RTKLIB does support PPP processing too. However, the stable version only supports the finalized RTCM SSR messages, implying that there is no support for PPP with the Galileo constellation or phase bias messages.

There exists an Open-Source fork of RTKLIB (RTKLIBexplorer, 2021), based on the development version 2.4.3. In this version, Galileo and BeiDou constellations are supported and the draft RTCM 3 phase bias messages are decoded. However, processing of the phase bias information is not yet implemented, thus PPP with ambiguity resolution is not possible at this point.

Developed by Laurichesse and Privat (2015), PPP-Wizard is a package that extends RTKLIB 2.4.2 with PPP features up to ambiguity resolution. The new features include Galileo and BeiDou support, but unfortunately the list of supported receivers is limited; as a result, the Galileo F/NAV message cannot be decoded. Unfortunately, the development of the open-source PPP-Wizard has been discontinued, thus only this outdated version is available.

As a project independent of RTKLIB, Chen and Chang (2021) have implemented PPPlib. It supports all GNSS constellations and comes with modern data visualization tools. As a downside, PPPlib only supports IGS post-processing file formats, implying that it cannot be used together with Galileo HAS without extensive modifications.

Out of the above alternatives, this paper chooses PPP-Wizard as the package to interface with as a proof of concept: although the discontinued development brings a significant risk, PPP-Wizard is regarded as the engine that needs the least modifications to be usable with HASlib.

III. HAS DECODING LIBRARY DESIGN

The design of the library was driven by three main principles. First, the implementation must not be too resource-hungry to allow real-time operation on various platforms. Second, it should be portable, not depend on commercial software or libraries, as well as easy to use and integrate to existing projects and workflows. Third, the library should be straightforward to configure or modify.

In line with the above requirements, HASlib has been written in Python language. The features and structure of the library are presented in this section.

1. Reed–Solomon Decoding

The HAS corrections are received from several Galileo satellites simultaneously at a rate of 448 bits per second per satellite (GSA, 2020). In order to optimize the time needed to download the data, the HAS exploits the availability of several satellites by encoding the messages in a Reed–Solomon scheme. This encoding distributes the information content of a single 32-page message over a total of 255 pages. Then, in order to receive a single message consisting of k pages, the receiver needs to receive any k different pages to recover the information. In other words, the number of possible pages increases while the amount of information to be downloaded remains constant. If the messages are transmitted in the optimal way, this scheme enables a very efficient dissemination of data Senni et al. (2022).

While one of the common advantages of the Reed–Solomon coding is the error correction capability, incorporating the message cyclic redundancy check (CRC) also enables to consider the transmission channel as an *erasure channel*. In this model, only pages with a valid CRC are decoded, without attempting to recover the erroneous pages. This can speed up the decoding.

Reed–Solomon encoding is a linear operation in a Galois field. Consequently, both encoding and decoding operations can be expressed as a matrix multiplication inside this field. Let $\boldsymbol{c}$ represent the 32-element message vector. Then, it can be encoded with the generator matrix G to the code vector $\boldsymbol{\Gamma}$ of length 255 as (EU, 2022):

$$\boldsymbol{\Gamma} = \begin{bmatrix} c_0 \\ c_1 \\ \vdots \\ c_{31} \\ \gamma_0 \\ \gamma_1 \\ \vdots \\ \gamma_{222} \end{bmatrix} = \begin{bmatrix} \boldsymbol{c} \\ \boldsymbol{\gamma} \end{bmatrix} = \begin{bmatrix} \mathrm{I} \\ \mathrm{P} \end{bmatrix} \boldsymbol{c} = \mathrm{G}\boldsymbol{c} \tag{1}$$

The generator matrix G is composed of blocks I and P. The top block I is an identity matrix of size 32×32, implying that the first 32 elements of the 255-page sequence are the original HAS message pages in clear text. The bottom block P is a dense matrix with size 223×32, and it corresponds to the remaining 223 parity pages.

This formulation makes the decoder straightforward to implement, without need for specialized algorithms. First, the received encoded pages are collected according to the page ID which can be found in the clear-text page header. Then, once the necessary number k of different pages have been received, the message bytes are vertically grouped into words, and the decoding matrix $\bar{\mathrm{G}}$, obtained as the $k \times k$ matrix consisting of the corresponding rows of G in the same order and its first k columns, is formed. The obtained matrix $\bar{\mathrm{G}}$ is invertible, and it is applicable to each vertical word; consequently, one matrix inversion is sufficient to decode the entire message. Reed–Solomon encoding and decoding, including the definition of the generator matrix G, is explained in more detail in the interface control document (EU, 2022).

2. Selection of Supported Output Formats

Evidently, selecting the supported output format(s) involves a tradeoff analysis just as the choice of OS PPP package does. The most important parameters to consider are the support for the information content of the HAS and the adoption in the navigation community. As a third parameter, the bandwidth requirements are considered.

Out of the formats listed in Table 1, RTCM 3 is the *de facto* standard in differential GNSS and RTK correction data distribution. Despite the slow progress with SSR, their role is significant: alternative formats formats like IGS or CSSR can be regarded as mostly compatible with RTCM. For instance, the IGS SSR mostly finalizes the message types that remain pending within the RTCM; however, the user adoption seems to be limited to a considerable degree to the IGS itself. Similarly, the SPARTN format which is mainly used by the u-blox PointPerfect service. As a fourth alternative, the CSSR format has influenced the definition of the messages for Galileo HAS, but it is mainly utilized by Japanese QZSS for PPP services in Japan.

The data format candidates also differ in terms of bandwidth requirements, as analyzed by Vana et al. (2019). In brief, SPARTN and CSSR are more bandwidth-efficient than RTCM or IGS SSR, partly because of the use of PRN masks instead of PRN numbers: on average, for a message containing corrections for 20 satellites, SPARTN and CSSR can use up to 61 % and 80 % less bits, respectively. The difference is larger for short clock correction messages than longer orbit corrections messages. However, RTCM and IGS SSR formats do convey more correction information too: their orbit and clock corrections are not constant values but polynomials, which can reduce the correction update rate as the information remains valid for a longer time.

In general, none of the formats stands out in terms of the feature support and community adoption. Although less bandwidth-efficient, RTCM 3 and IGS SSR are chosen as the output formats for the first release of HASlib, even if CSSR may seem like a closer match with HAS in terms of the supported features. Extending the HASlib to support SPARTN is highly favorable once SPARTAN is extended with support for the Galileo constellation because it is the only one to support the do-not-use information; moreover, the existing user base of u-blox users underlines its importance.

3. Architecture Overview

The main design aim for the presented Galileo HAS decoder was to promote the wide adoption of this open-source library for research and evaluation purposes. In order to achieve this, the library was designed for ease of adoption by emphasizing modularity, code readability and easy extension through clear interface API. The modular structure enables straightforward re-use of parts of, or the whole library in future applications and custom solutions. In addition to the input and output modes chosen to support two of the most meaningful file formats, several different interfacing methods typically used in such applications, namely, streams, files and TCP connection, were developed. In order to accommodate future needs, adding new input and output formats is also designed to be as straightforward as possible.

The top-level architecture of the decoder library is illustrated in Fig. 1, followed by a description of the most important classes and their functionalities.

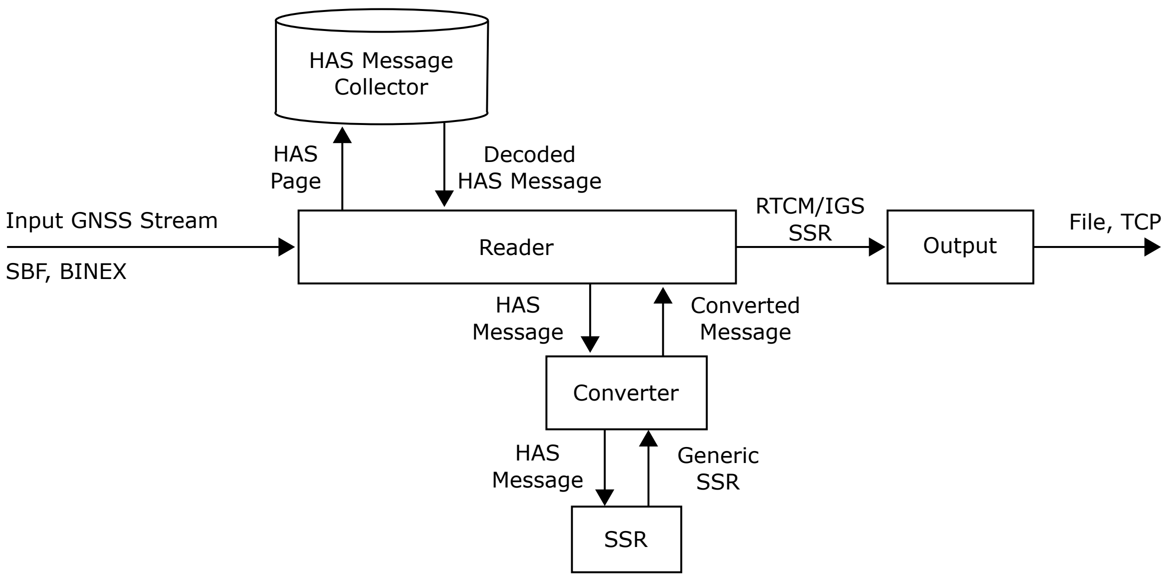

Figure 1: High-level architecture and data flow of HASlib

a) Reader

The decoder library comprises of several reader classes that handle the input streams and parsing of this data for the processing routines. Recorded data can be read from SBF and BINEX (UNAVCO, 2020) files. Both, forward and backwards readable BINEX records are supported. Furthermore, real-time or pre-processed SBF and BINEX format data can be read either from serial port or TCP server. Output of the HAS messages in raw data format, such as, binary or ASCII, is not yet implemented, but would provide valuable future work.

b) HAS

HAS message collection class is used to temporarily store the received HAS message pages. This class is also responsible for the sorting and handling of HAS messages, including storing and parsing of received masks and, furthermore, decoding the messages when enough pages have been acquired.

c) SSR

SSR class is a quasi-universal structure allowing the extension of the library to support other SSR formats. This class essentially enables storing all the information depicted in the HAS SIS ICD 1.0 (EU, 2022), such as, clock and orbit corrections, satellite signal biases and other relevant information, e.g. masks.

d) Converter

Acting in between the Reader class and SSR class, the converter class is used to convert HAS messages to other SSR formats.

e) Output

Output classes are used to format the converted messages for different output types, with each output type having it's own class: simple file output is achieved via one class, and another class enables real-time output via TCP server. Additionally, a class designed specifically for the interfacing to the PPPWizard library is included. This PPPWizard class can output data either via file or standard output stream format.

f) HAS Converter

HAS converter class (which is not shown in Fig. 1 is an all-encompassing interface and abstraction class for the entire library. This library consolidates access to all the other classes and provides a clean and simple user interface to most of the required functions and subroutines of the other classes via selected parameter driven functions.

4. License and Distribution

HASlib is publicly available on the GitHub platform (National Land Survey of Finland, 2022). It is released under the European Union Public License (EUPL). The copyright is owned by the European Union.

IV. TESTING

To validate the functionality of the HASlib, we conduct two tests. First, we evaluate the processing time of a single HAS message to show that the library is real-time capable. Then, the output of HASlib is passed to the PPP-Wizard package to demonstrate that the interfaces work. For these tests, we use Galileo HAS data that were logged from the satellites during the HAS Public Observation Phase in September 2021.

1. Decoding time

The most resource-consuming tasks within HASlib are message decoding and conversion. In order to ensure that these tasks can be run in real time, the processing time distribution was measured with HASlib executed on an Intel NUC (NUC10i7FNH2) miniature computer running non-real-time Linux operating system.

It can be expected that the processing time depends on the block length as the decoding process involves some linear algebra (see Eq. 1). During the HAS Public Observation Phase, two main block types were broadcast: short clock corrections at a higher rate and longer blocks including orbit corrections, satellite masks, etc. at a lower update rate.

The results are plotted in Fig. 2; the processing time distribution is clearly bimodal because of the two message types being broadcast. Nevertheless, 99 % of the received messages were processed in 31 ms or less, which implies that HASlib is capable of real-time operation. In particular, a majority of the more frequent clock correction messages were processed in less than 20 ms.

It should be noted that the library is primarily targeted for R&D purposes and hence, the tests were run on the aforementioned, relatively powerful top-of-the-line system based on x86 architecture. However, we expect reasonable, if not similar, performance on more resource constrained platforms.

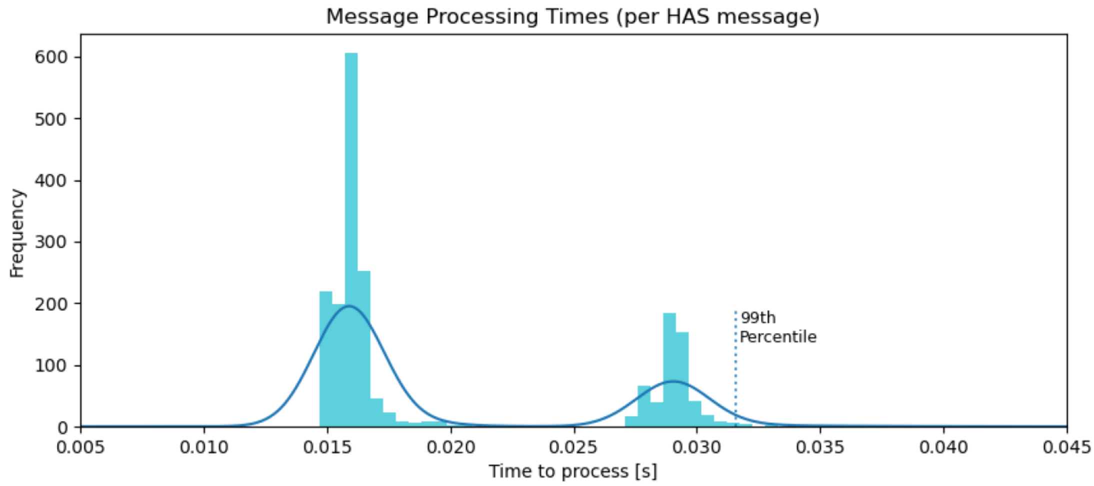

Figure 2: Distribution of HAS message processing times as evaluated on a miniature desktop computer. The left cluster corresponds to two-page MT1 clock messages while the other one stems from longer messages including more correction information. (Horst, 2021)

2. Interfacing with PPP-Wizard

Figure 3 illustrates the effect of using HASlib with PPP-Wizard. In the figure, a total of 7:50 hours of data have been processed in sections of 20 minutes each (September 22, 2021, 06:45–14:34 UTC); the different processing sections have been indicated in different colors. In order to reduce the effect of initial converge transients, the first 5 minutes have been omitted from each section following the convergence time estimate given in GSA (2020). The reference position was determined by the AUSPOS Online GPS Processing Service v. 2.3 (Geoscience Australia, 2021) using a separate set of data.

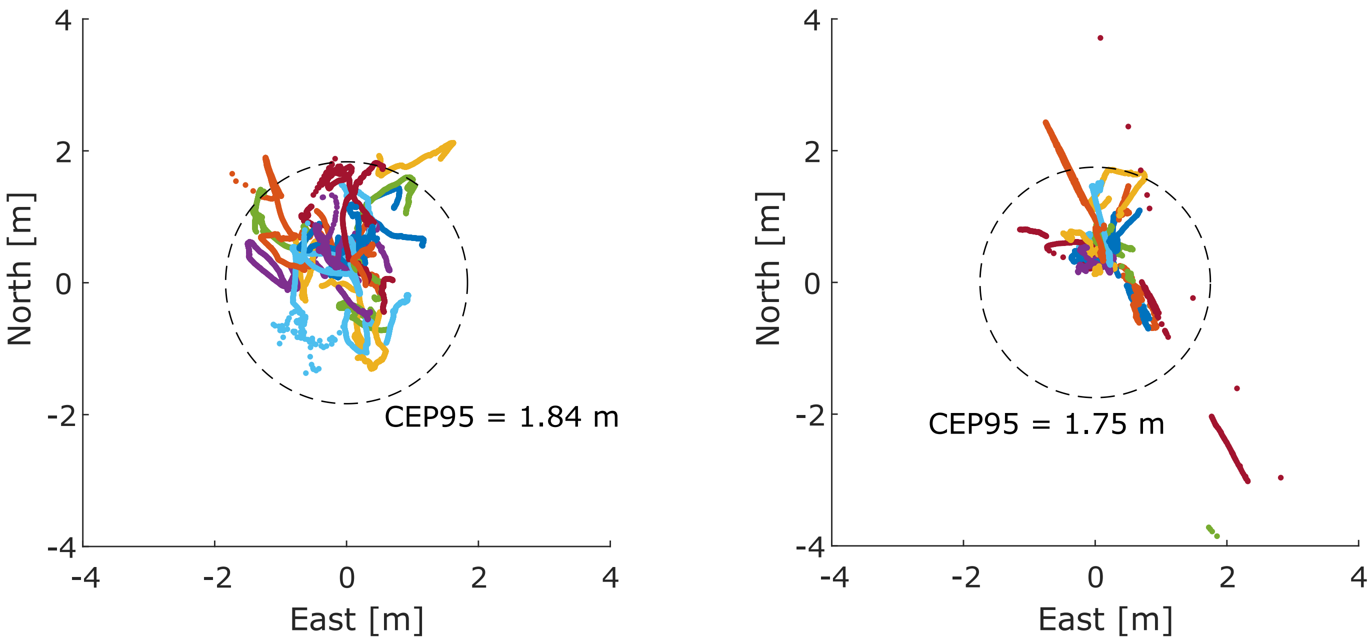

Figure 3: Effect of HASlib on the PPP-Wizard solution. Left: dual frequency solution without corrections; right: PPP with HAS

By visual inspection of Fig. 3, it can be seen that enabling HAS changes the behavior of the solution. Unfortunately, there are some outliers in the HAS solutions, which inflates the 95 % Circular Error Probable (CEP95). Even if the most obvious outliers would be discarded, these first results would not meet the 20 cm 2-sigma accuracy of the HAS. Several possible reasons can be identified. First, it should be noted that no valid phase biases messages were transmitted during the Galileo HAS Public Observation phase. Therefore, the interfacing test was conducted with self-constructed data, and the full HAS could not be tested with the PPP-Wizard yet. Second, there are a variety of configuration parameters for PPP-Wizard which could possibly be tuned to improve the performance. Third, the solution point clouds tend slightly to the North with respect to the origin, indicating a possible bias in the reference position.

Once the HAS starts transmitting phase bias corrections, it will be interesting to test PPP-Wizard in the ambiguity resolution mode; it is expected to improve the accuracy and stability.

V. SUMMARY

This paper presented HASlib, an open-source library intended to facilitate the adoption of Galileo HAS for research and development purposes. The library has been written in Python and it decodes the HPVRS-encoded HAS correction messages and outputs them in either the RTCM 3 or the IGS SSR format; in the first release, the supported input formats are SBF and BINEX. The architecture of the library has been designed to be easily extensible to other input and output formats, implementing which is considered future work.

The experimental results validate that HASlib is capable of real-time operation on a miniature desktop computer. As a proof of concept, it was shown that the interfaces are compatible with the open-source PPP-Wizard package, and enabling the HAS corrections improved the positioning precision over un-assisted dual-frequency processing; more extensive testing and verification of the library is foreseen after the HAS initial service declaration. In total, HASlib makes it possible to use Galileo HAS with receivers and software that do not support HAS out of the box.

ACKNOWLEDGMENTS

This work was conducted under the project Precise and Authentic User Location Analysis (PAULA), funded by the European Commission DG-DEFIS under contract DEFIS/2020/OP/0002.

REFERENCES

Chen, C. and Chang, G. (2021). PPPLib: An open-source software for precise point positioning using GPS, BeiDou, Galileo, GLONASS, and QZSS with multi-frequency observations. *GPS Solutions*, 25(18).

EU (2022). *Galileo High Accuracy Service Signal-in-Space Interface Control Document (HAS SIS ICD)*. European Union. Issue. 1.0.

Fernández-Hernández, I., Senni, T., Borio, D., and Vecchione, G. (2020). High-parity vertical Reed-Solomon codes for long GNSS high-accuracy messages. *Navigation*, 67(2):365–378.

Geoscience Australia (2021). Auspos online gps processing service. Online, available: `https://gnss.ga.gov.au/auspos`. accessed 30 Jun 2022.

GSA (2017). *EGNOS Open Service (OS) Service Definition Document*. European GNSS Agency. Rev. 2.3.

GSA (2020). *Galileo High Accuracy Service (HAS) Info Note*. European GNSS Agency. doi:10.2878/581340.

Hirokawa, R. and Fernández-Hernández, I. (2020). Open format specifications for PPP/PPP-RTK services: Overview and interoperability assessment. In *Proc. 33rd International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS+ 2020)*, pages 1268–1290.

Horst, O. (2021). Implementation of an open-source software suite for the Galileo High Accuracy Service. M.Sc. (Tech.) thesis, Aalto University, School of Electrical Engineering, Espoo, Finland. URN:NBN:fi:aalto-202202061786.

Laurichesse, D. and Privat, A. (2015). An open-source PPP client implementation for the CNES PPP-WIZARD demonstrator. In *Proc. 28th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS+ 2015)*, pages 2780–2789, Tampa, Florida.

National Land Survey of Finland (2022). HASlib. Online, available: `https://github.com/nlsfi/HASlib`.

PPP-Wizard (2021). The PPP-Wizard project. Online, available: `http://www.ppp-wizard.net/`. accessed March 3, 2022.

RTCM (2013). Differential GNSS (Global Navigation Satellite Systems) services – version 3. RTCM Standard 10403.2, Radio Technical Commission for Maritime Services, Arlington, VA. with Amendments 1 and 2.

RTKLIBexplorer (2021). rtklibexplorer/rtklib. Online, available: `https://github.com/rtklibexplorer/RTKLIB`. accessed June 27, 2022.

Senni, T., Fernandez-Hernandez, I., and Cancela, S. (2022). Evaluation of the Hungarian algorithm for optimal transmission of the Galileo HAS message from multiple satellites. In *Proc. International Conference on Localization and GNSS*, Tampere, Finland.

Takasu, T. (2013). RTKLIB: An open source program package for GNSS positioning. Online, available: http://www.rtklib.com/. accessed June 27, 2022.

UNAVCO (2020). BINEX: Binary Exchange format. Online, available: https://www.unavco.org/data/gps-gnss/data-formats/binex/binex.html. accessed June 28, 2022.

Vana, S., Aggrey, J., Bisnath, S., Leandro, R., Urquhart, L., and Gonzalez, P. (2019). Analysis of GNSS correction data standards for the automotive market. *Navigation*, 66(3):577–592.