GPS Solutions (2024) 28:169
https://doi.org/10.1007/s10291-024-01706-7

GPS TOOLBOX

HASPPP: an open-source Galileo HAS embeddable RTKLIB decoding package

Runzhi Zhang[1,2] · Rui Tu[3] · Xiaochun Lu[1,2]

Received: 9 March 2024 / Accepted: 17 July 2024

Abstract
Galileo supports global precise point positioning (PPP) service by delivering precise products via satellites, which is known as high accuracy service (HAS). To improve reception efficiency, the Galileo HAS employs a high-parity vertical reed-solomon encoding scheme, increasing the complexity of HAS corrections recovery. To promote research and application of the HAS, an open-source C/C++decoding package, HASPPP, has been developed for seamless embedding into the prevalent C/C++-based software, such as RTKLIB. HASPPP provides interfaces for effortless decoding support of raw HAS binary data from various manufacturers. The HASPPP manual offers code analysis along with simple to complex examples to aid users in swiftly mastering decoding. HASPPP has undergone rigorous validation of its decoding accuracy and reliability.

Keywords Galileo · Precise point positioning · High accuracy service · Decoder

Introduction

The European Galileo pioneeringly delivers global precise point positioning (PPP) (Zumberge et al. 1997) through the Internet and Galileo satellites on the E6-B frequency band, which is known as the high accuracy service (HAS) (EUSPA 2022). On January 24, 2023, the Galileo announced the provision of initial HAS service, offering global satellite products (EUSPA 2023). Naciri et al. (2023) comprehensively evaluated the quality and positioning performance of the HAS product, affirming its value in PPP applications. Research on the Galileo HAS is ongoing.

The GPS Toolbox is a topical collection dedicated to highlighting algorithms and source code utilized by GNSS engineers and scientists. If you have a program or software package you would like to share with our readers, please submit a paper to the GPS Toolbox.

✉ Rui Tu
turui-2004@126.com

1 National Time Service Center, Chinese Academy of Sciences, Xi'an 710600, China

2 University of Chinese Academy of Sciences, Beijing 100049, China

3 College of Geodesy and Geomatics, Shandong University of Science and Technology, Qingdao 266590, China

The HAS product, whether disseminated via the Internet or Galileo satellites, is provided in the form of state-space representation (SSR) of corrections. The Radio Technical Commission for Maritime Services (RTCM) organization standardized the network transmission protocol and specific format layout for SSR corrections (RTCM 2021). Some open-source C/C++global navigation satellite system (GNSS) software, such as RTKLIB (Takasu and Yasuda 2009) and Bundesamt für Kartographie und Geodäsie (BKG) ntrip client (BNC) (BKG 2007), along with others, provide RTCM decoding capabilities. However, due to the restricted baud rate of the E6-B signal, the HAS defined a format similar to compact SSR (CSSR) (Fernandez-Hernandez et al. 2022; Hirokawa et al. 2016). Additionally, to improve reception performance, the Galileo HAS adopts a high-parity vertical reed-solomon (HPVRS) encoding scheme (Fernandez-Hernandez et al. 2020). However, it increases the complexity of recovering HAS corrections. Therefore, cashing RTCM-formatted HAS corrections via the Internet has become the preferred data source for HAS research (Parra et al. 2023). However, this overlooks the value of satellite-based PPP enhancement, which mitigates excessive reliance on networks and their status.

Published online: 01 August 2024

Horst et al. (2022) have developed an open-source python HAS decoder called HASlib. In parallel, Gioia et al. (2022) developed a Galileo HAS parser (GHASP) also based on the python language, which was later refined and open-sourced to the GPS toolbox repository (Borio et al. 2023). The advent of these decoding tools has significantly advanced HAS research. However, the majority of popular open-source GNSS software is based on C/C++. Therefore, Prol et al. (2024) attempted to integrate HASlib and RTKLIB by converting satellite-based HAS corrections into RTCM format. However, this appears to be inconvenient and also adds complexity to mastering both python and C/C++. A more desirable mode would be to seamlessly embed HAS decoders into these GNSS software, similar to RTCM decoders. This contribution has been made and proven feasible in the C/C++embeddable decoding package HASPPP. The stability and accuracy of HASPPP have been confirmed through continuous global HAS time comparison (Zhang et al. 2024). The authors open-sourced HASPPP with the purpose of promoting research and application of the HAS, while also providing users with the most detailed analysis of decoding.

The remaining sections are organized as follows: the advantages of HASPPP are first introduced; then, the supported platforms, data, and outputs are introduced, followed by a brief description of the architecture and embedding steps; finally, the results of RTKLIB embedded with HASPPP for time comparison applications are presented.

Advantages of the C/C++embedding package HASPPP

The python language is a prominent scientific computing tool, and its influence, as well as its advantages in computation within HPVRS, cannot be overlooked. However, numerous open-source GNSS software are predominantly built on C/C++, with many functionalities being well-implemented. The appeal of the C/C++HAS decoding package remains irresistible. The primary advantages of HASPPP are as follows:

- Embedding HASPPP directly into C/C++GNSS software is a more ideal and convenient mode;
- HASPPP decoding package eliminates the inconvenience of installing various libraries;
- HASPPP helps save time by eliminating the need to become proficient in additional languages;
- Convenient portability; merely requiring a cross-compiler enables the embedded HASPPP C/C++GNSS project to be ported to various environments;
- HASPPP provides the most detailed explanation available of the decoding process and programming.

Platforms, supported data, and output

HASPPP is platform-agnostic, operating seamlessly across both Windows and Linux environments. HASPPP inherits macro definitions and data structures from RTKLIB. Consequently, it can be effortlessly embedded into RTKLIB programs. Moreover, the embedment is not limited solely to RTKLIB programs. The HASPPP decoding function returns the decoded HAS satellite products using the ***ssr_t*** structure, which is defined in RTKLIB to store SSR corrections. It simply requires an interface to convert the HAS product stored in ***ssr_t*** to other C/C++GNSS software.

Presently, HASPPP supports raw binary HAS data from three GNSS receiver types: Septentrio, NovAtel, and China SINO. HASPPP has also established a framework that facilitates the decoding of HAS binary data from various manufacturers. Users are required to develop a function to process the manufacturers' custom data frame header, extracting raw Galileo C/NAV data for input into the HASPPP decoding function.

RTKLIB programs provide functionality to encode ***ssr_t*** data into RTCM. The only missing component is an embeddable HAS decoding package. Hence, HASPPP does not offer binary RTCM output. Instead, it provides a detailed HPVRS decoding output file (**pid.txt**), package log files (**clock.txt** and **trace.txt**), and a plaintext output file ontaining recovered HAS corrections (**SSRA.bnc**) consistent with BNC corrections output. The detailed specifications can be referenced in the HASPPP manual.

Code architecture and embedding steps

The decoding architecture and function invocation interface are depicted in Fig. 1. Italicized text denotes function names, while orange-filled boxes represent external calling interfaces. Unfilled boxes denote private functions exclusive to HASPPP. The invocation consists of four steps:

- **Step 1**- Set up the working directory.

HASPPP employs a table lookup method for HPVRS decoding and automatically matches output files. All of these functionalities require setting up the working directory.

- **Step 2**- Load raw HAS binary data.

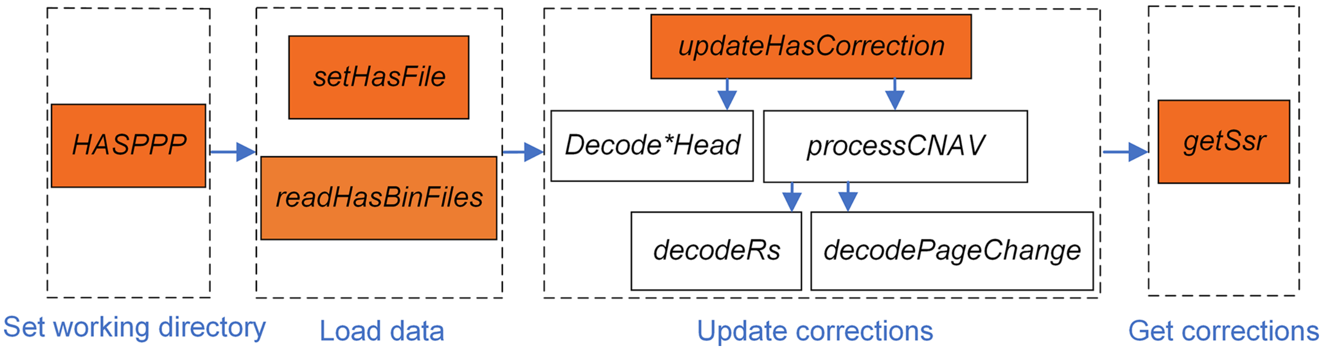

Fig. 1 Architecture of HASPPP

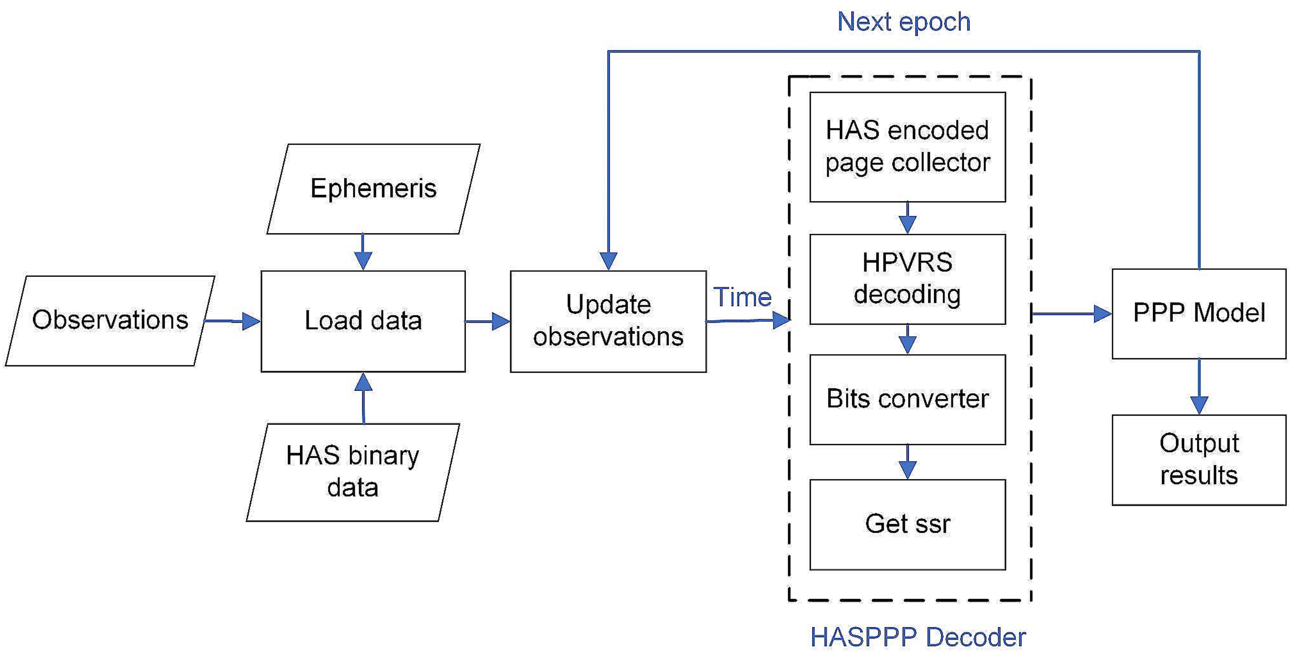

Fig. 2 Embedding HASPPP within RTKLIB

In post-processing scenarios, raw HAS binary data is cached in the program's dynamic buffer, facilitating the emulation of real-time PPP.

- **Step 3**- Decode and update corrections.

After invoking *updateHasCorrection*, the HAS decoding persists until it is updated to the specified moment provided as input. Meanwhile, the core of HASPPP decoding package lies within the *processCNAV* function, providing an interface for real-time decoding.

- **Step 4**- Output of the status of recovered HAS corrections.

Calling *getSsr* will return the ***ssr_t*** structure representing the current corrections.

Therefore, embedding HASPPP into RTKLIB is notably facile, as depicted in Fig. 2. Following the completion of Steps 1 and 2, whenever epoch observations are refreshed, Step 3 is invoked to update corrections. Subsequently, in Step 4, the status of corrections can be retrieved directly. These, alongside the observations and ephemeris, are utilized within PPP model.

Results

HASPPP provides three examples: official set of collected HAS encoded pages, cashed HAS binary data, and RTKLIB programs embedded with HASPPP. Further intricate details can be elucidated by referencing its manual. The article focuses only on showcasing the PPP solution of RTKLIB embedded with HASPPP.

Table 1 Details of GNSS receivers and time links

Station	Receiver	Antenna	Clock source	Distance from BRUX (km)
BRUX	SEPT POLARX5TR	JAVRINGANT_DM	EXTERNAL IMASER 3000	---
KIRU	SEPT POLARX5TR	SEPCHOKE_B3E6	EXTERNAL CESIUM	2097
TWTF	SEPT POLARX4TR	ASH701945C_M	EXTERNAL STEERED H-MASER	8713

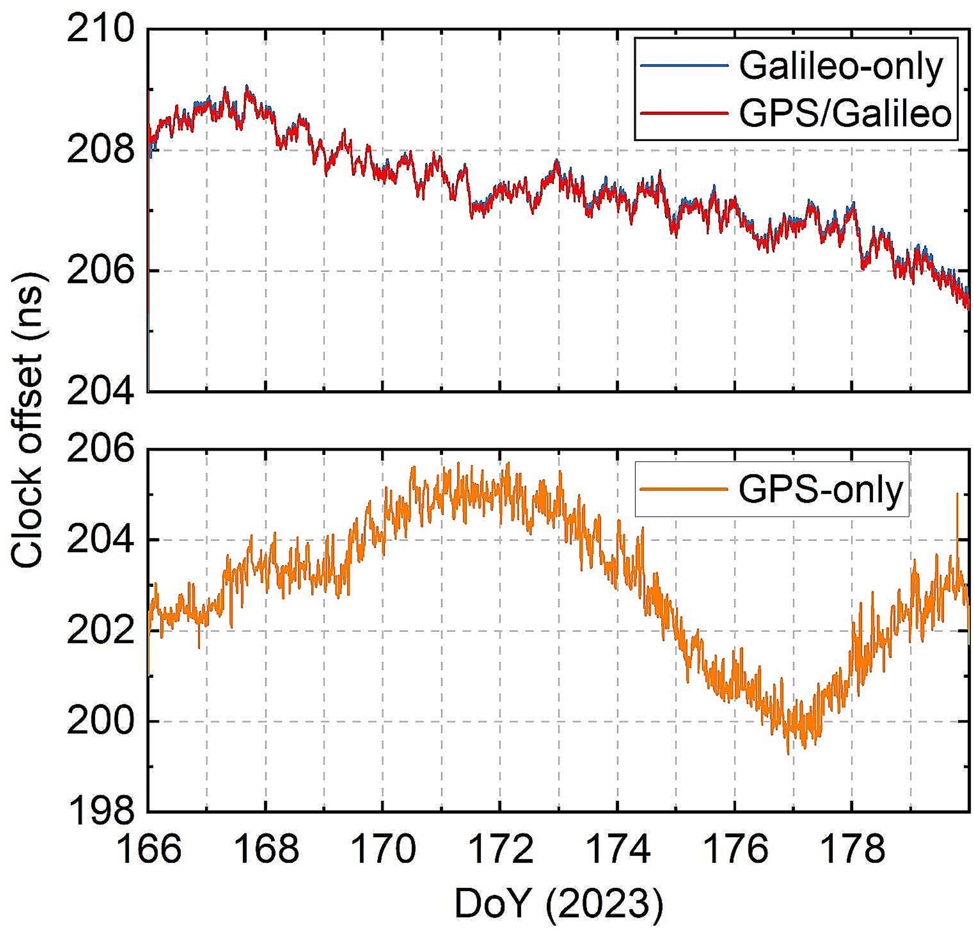

Fig. 3 BRUX receiver clock offsets calculated using the HAS product

In carrier phase time comparison via the PPP, it requires programs to operate stably over prolonged periods to estimate the receiver clock offset. Therefore, continuous and stable HAS time transfer challenges the reliability of HASPPP in decoding.

A 14-day time comparison from day of year (DoY) 166 to 179 in 2023 was showed. Observations were collected from three International GNSS Service (IGS) stations, denoted as BRUX, KIRU, and TWTF, all equipped with external high-precision atomic clocks. With the BRUX station at the timekeeping laboratory Royal Observatory of Belgium (ORB) serving as the central node, two time comparison links were established, namely the short baseline BRUX-KIRU and long baseline BRUX-TWTF. Detailed specifications can be found in Table 1. The ionosphere-free combination model was applied in the PPP solution, combining L1C/A-L2P observations for GPS and E1-E5b observations for Galileo. Performance of a simulated real-time HAS time comparison was analyzed in static mode. Further details are provided in the HASPPP manual and in the authors' publication (Zhang et al. 2024).

Figure 3 depicts the receiver clock offset of the BRUX station. The reference time of the HAS product is the Galileo System Time (GST). The receiver clock offset in the PPP solution using the HAS product is the local time offset relative to the GST, incorporating the receiver-side uncalibrated hardware delays. In dual-system scenarios, the clock offset calculated from the Galileo equations serves as the receiver clock offset. Therefore, the BRUX receiver clock

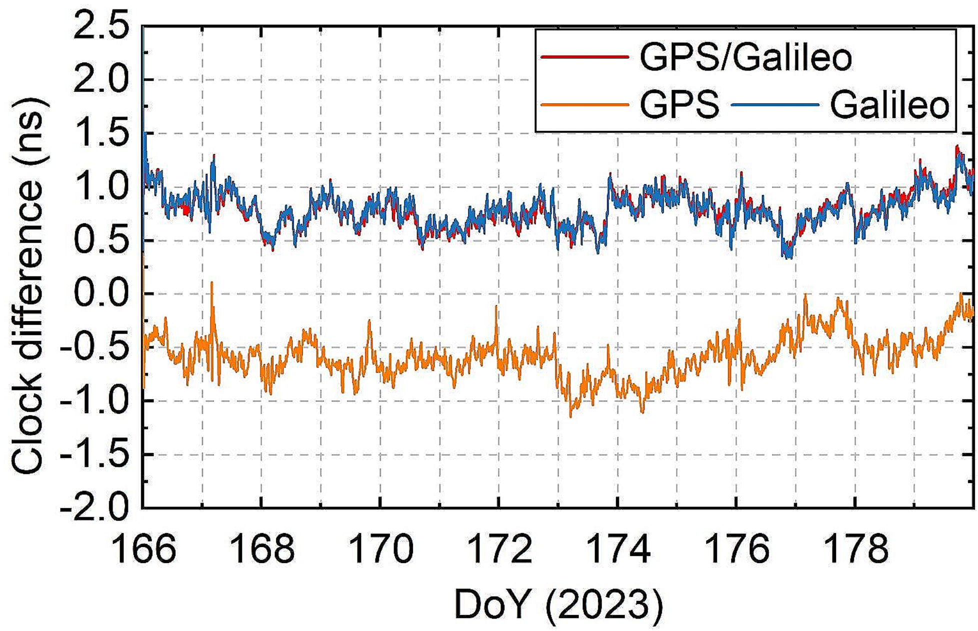

Fig. 4 BURX-KIRU time comparison error series

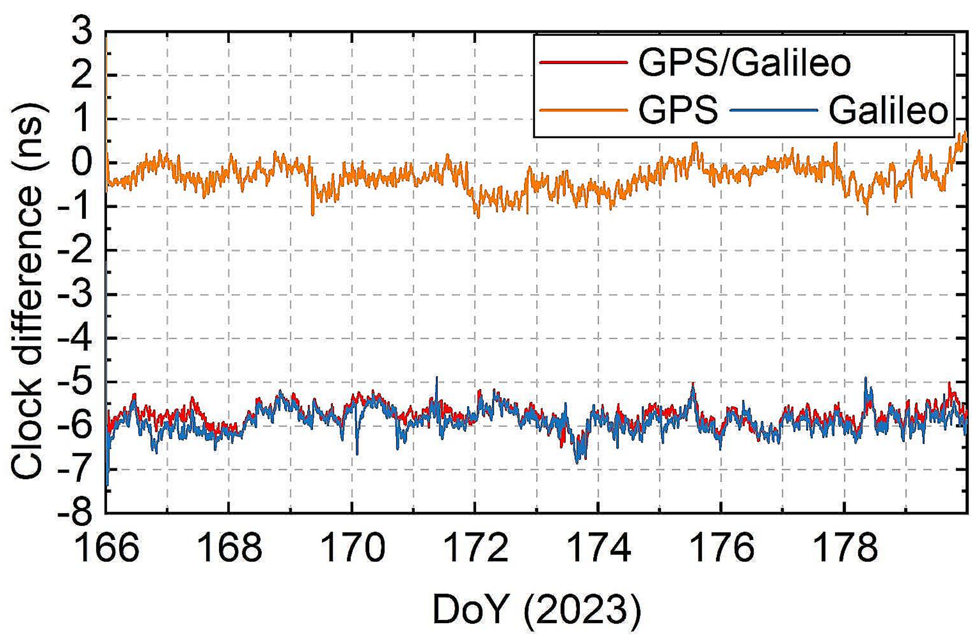

Fig. 5 BURX-TWTF time comparison error series

offset trends for Galileo and GPS/Galileo exhibit congruence in Fig. 3.

Figures 4 and 5 respectively show the time comparison errors with respect to the IGS final clock product. A systematic bias is present in GPS and Galileo HAS time comparison, attributed to receiver-side uncalibrated hardware delays (Zhang et al. 2024). The standard deviation (STD) and mean values of the time comparison errors are listed in Table 2. The STD value for the short baseline is approximately 0.19 ns, while for the long baseline it is around 0.3 ns. The HAS decoding has demonstrated consistent and reliable operation. The HASPPP decoding package has been proven to be stable and reliable.

Table 2 STD and mean values of the HAS time comparison errors (unit: ns)

Time links	GPS		Galileo		GPS/Galileo	
	STD	Mean	STD	Mean	STD	Mean
BRUX-KIRU	0.19	-0.58	0.16	0.79	0.15	0.79
BRUX-TWTF	0.30	-0.32	0.26	-5.84	0.25	-5.76

Summary

The Galileo HAS adopts an efficient HPVRS scheme for disseminating satellite products, which also increases the complexity of recovering HAS corrections. An open-source C/C++HAS decoding package HASPPP has been released to advance research and applications of the HAS.

HASPPP provides a mode for seamless integration into RTKLIB. Additionally, it also offers the ***ssr_t*** interface for embedding into other C/C++GNSS software. HASPPP retains the original C/NAV decoding functions, enabling seamless portability to any GNSS receiver supporting HAS binary output. Its manual also offers detailed log files and comprehensive program code analysis.

The stability, reliability, and accuracy of HASPPP decoding have been demonstrated in continuous and stable PPP time transfer, as well as in three provided examples ranging from simple to complex.

Acknowledgements The authors acknowledge the support of IGS for providing observations and satellite products.

Author contributions R.Z. authored the architecture and core code; R.T., and X.L. enhanced the functionality; R.Z., R.T., and X.L. wrote the manuscript.

Funding This study was funded by the National Natural Science Foundation of China (No. 41974032; 42274019).

Data availability The dataset and results can be found at https://github.com/ZhangRunzhi20/High-Accuracy-Service-public-dataset. The more datasets are available from authors upon reasonable request.

Declarations

Ethics approval and consent to participate Not applicable.

Consent for publication All authors have reviewed and consented to the publication of the article.

Competing interests The authors declare no competing interests.

References

BKG (2007) BKG Ntrip Client (BNC). https://igs.bkg.bund.de/ntrip/bnc

Borio D, Susi M, Gioia C (2023) GHASP: a Galileo HAS parser. GPS Solutions 27(4):197. https://doi.org/10.1007/s10291-023-01529-y

EUSPA (2023) Galileo High Accuracy Service goes live! https://www.euspa.europa.eu/newsroom/news/galileo-high-accuracy-service-now-operational

EUSPA (2022) Galileo High Accuracy Service Signal-In-Space Interface Control Document (HAS SIS ICD). https://www.gsc-europa.eu/sites/default/files/sites/all/files/Galileo_HAS_SIS_ICD_v1.0.pdf

Fernandez-Hernandez I, Senni T, Borio D, Vecchione G (2020) High-parity vertical Reed-Solomon codes for long GNSS high-accuracy messages. Navigation 67(2):365–378. https://doi.org/10.1002/navi.357

Fernandez-Hernandez I et al (2022) Galileo high accuracy service: initial definition and performance. GPS Solutions 26(3):65. https://doi.org/10.1007/s10291-022-01247-x

Gioia C, Borio D, Susi M, Fernandez-Hernandez I (2022) The Galileo High Accuracy Service (HAS): Decoding and Processing Live Corrections for Code-Based Positioning. ION ITM 2022, Institute of Navigation, Long Beach, California, USA, January 25–27, pp. 1065–1074. https://doi.org/10.33012/2022.18204

Hirokawa R, Sato Y, Fujita S, Miya M (2016) Compact SSR messages with Integrity Information for Satellite Based PPP-RTK Service. ION GNSS 2016, Institute of Navigation, Portland, Oregon, USA, September. 3372–3376. https://doi.org/10.33012/2016.14794

Horst O, Kirkko-Jaakkola M, Malkamäki T, Kaasalainen S, Fernández-Hernández I, Chamorro-Moreno A, Cancela-Díaz S (2022) HASlib: an Open-Source Decoder for the Galileo High Accuracy Service. 2625–2633 ION GNSS 2022, Institute of Navigation, Denver, Colorado, USA, September 19–23. https://doi.org/10.33012/2022.18508

Naciri N, Yi D, Bisnath S, De Blas FJ, Capua R (2023) Assessment of Galileo High Accuracy Service (HAS) test signals and preliminary positioning performance. GPS Solutions 27(2):73. https://doi.org/10.1007/s10291-023-01410-y

Parra C, Schütz A, Pany T, Hugentobler U, Baumann S (2023) The Galileo High Accuracy Service: Assessment of the Quality of Corrections and Preliminary PPP Performance. ION GNSS 2023, Institute of Navigation, Denver, Colorado, UAS, September 11–15, pp. 2581–2591. https://doi.org/10.33012/2023.19221

Prol F, Kirkko-Jaakkola M, Horst O, Malkamäki T, Bhuiyan M, Kaasalainen S, Fernández-Hernández I (2024) Enabling the Galileo high accuracy service with open-source software: integration of HASlib and RTKLIB. GPS Solutions 28(2):71. https://doi.org/10.1007/s10291-024-01617-7

RTCM (2021) RTCM 10410.1 Standard for Networked Transport of RTCM via Internet Protocol (Ntrip) Version 2.0. https://rtcm.myshopify.com/collections/differential-global-navigation-satellite-dgnss-standards/products/rtcm-10410-1-standard-for-networked-transport-of-rtcm-via-internet-protocol-ntrip-version-2-0-with-amendment-1-june-28-2011

Takasu T, Yasuda A (2009) Development of the low-cost RTK-GPS receiver with an open source program package RTKLIB. International symposium on GPS/GNSS, International Symposium on GPS/GNSS. Korea, November 4–6

Zhang R, Tu R, Lu X, He Z, Guang W, Xiao G (2024) Initial and comprehensive analysis of PPP time transfer based on Galileo high accuracy service. GPS Solutions 28(2):94. https://doi.org/10.1007/s10291-024-01633-7

Zumberge JF, Heflin MB, Jefferson DC, Watkins MM, Webb FH (1997) Precise point positioning for the efficient and robust analysis of GPS data from large networks. J Geophys Research: Solid Earth 102(B3):5005–5017. https://doi.org/10.1029/96jb03860

Runzhi Zhang is pursuing a Ph.D. degree at the NTSC. His primary research focuses on real-time GNSS time transfer.

Rui Tu is a professor at Shandong University of Science and Technology. He received a Ph.D. in 2014 from GFZ. His primary research interests include GNSS PPP, RTK positioning, and applications in geosciences.

Xiaochun Lu is a professor and Ph.D. supervisor at the UCAS. Her main research focuses on GNSS time transfer.