2021 IEEE 18th International Conference on Smart Communities: Improving Quality of Life Using ICT, IoT and AI (HONET) | 978-1-6654-2386-1/21/$31.00 ©2021 IEEE | DOI: 10.1109/HONET53078.2021.9615482

Integrating DGNSS/RTK Positioning with IoT and Smart City Applications

Ruowei Xiao
Faculty of Information Technology
and Communication Sciences
Tampere University, Finland
Email: ruowei.xiao@tuni.fi

Zhanwei Wu
School of Design
Shanghai Jiao Tong University
Shanghai, China
Email: zhanwei_wu@sjtu.edu.cn

Oğuz ’Oz’ Buruk
Faculty of Information Technology
and Communication Sciences
Tampere University, Finland
Email: oguz.buruk@tuni.fi

Juho Hamari
Faculty of Information Technology
and Communication Sciences
Tampere University, Finland
Email: juho.hamari@tuni.fi

***Abstract*—Real-time high-precision positioning will generate new added values and potentials for many IoT and smart city applications. The emerging Real-time Kinematic (RTK) technology has greatly enhanced the precision of Global Navigation Satellite System (GNSS) based positioning, however, it remains a siloed, highly specialized technology that lacks coherent development architecture compatible with current IoT technology stack. This study aims to use commercial-of-the-shelf (COTS) equipment and general Web technology stack for integrating RTK positioning into general-purpose, value-added IoT/smart city services. Our preliminary evaluation has showed positive results over multiple indices, including positioning precision, portability and overall cost-performance balance.**

I. Introduction

The Global Navigation Satellite System (GNSS), e.g., GPS, GALILEO, Beidou etc., has long been a scaffold technology for a myriad of human activities. In the Internet of Things (IoT), particularly the smart cities, location awareness is considered the foundation to realize position tracking/navigation of IoT nodes, autonomous motions and remote control of mobile devices [1], [2]. However, civilian GNSS signals only provide a positioning precision at a few meters level. Novel enhancing technologies have emerged since the 1990s, among which, Real-Time Kinematic (RTK) positioning is able to obtain centimeter-level precision. It relies on a GNSS receiver, also known as rover, to receive real-time correction data based on differential carrier wave measurement from one or more base stations [3]. As regional/national GNSS base station infrastructure and continuous operating reference station (CORS) networks gradually scaled up and covered many countries and areas, RTK has consequently pervaded some highly specialized domains, such as geodetic survey [4], building structure monitoring [5] and precision agriculture [6].

Real-time high-precision positioning is supposed to generate new added values and potentials also for many mass-market IoT and smart city applications, such as location intelligence, smart vehicle network, robotics and automation, just to name a few. However, there remain several unaddressed technical gaps. First, RTK has long been relying on heterogeneous, dedicated equipment, as well as monolithic, hardware/protocol/platform-specific technology stack. In recent decades, multi-constellation multi-frequency GNSS receivers with built-in micro-processing solutions have become miniaturized and cheaper, and it became affordable to use commercial-off-the-shelf (COTS) devices to replace costly apparatus. Yet, the interoperability of upper-layer architecture remains the biggest bottleneck for RTK positioning to be seamlessly integrated with IoT into general-purpose usages [7], [8]. Specifically for IoT and smart city application developers, who are likely to have limited expertise and know-how about GNSS/RTK and its underlying mechanism, it still appears to be a high hurdle for them to adopt this technology.

Moreover, although RTK positioning is supposed to achieve centimeter-level precision, it often entails using large-volume, stationary/semi-stationary antennas. Also, to avoid signal degradation caused by environmental obstruction etc., there is a strict requirement on the working environment that it must not have many obstacles like buildings or trees. Apparently, these restrictions made RTK not naturally ideal to be implemented in many smart city applications, specifically those launched in urban environment and expecting higher mobility. As a result, these technical silos have greatly hindered a full-fledged RTK application in IoT and smart city areas.

To address the above gaps, this paper proposes a highly practicable, general-purpose architecture for designing and developing RTK-enhanced IoT and smart city services. Exemplified software, hardware implementations using only COTS equipment and general Web technology stack were developed based on the architecture. Our preliminary evaluation showed positive results over multiple indices, including positioning precision, portability and overall cost-performance balance.

II. Background

Prior to this millennium, Differential GNSS (DGNSS) systems and applications were predominantly constructed based

978-1-6654-2386-1/21/$31.00 ©2021 IEEE

on stationary/semi-stationary hardware, monolithic architecture and proprietary software. Specifically, the correction data was transmitted mostly via radio broadcasting, which was one-way communication and greatly restricted to limited bandwidths like HF, MF, LF etc. These siloed, tightly coupling systems relied highly, if not exclusively, on specialized equipment, standards, and/or transmission medium. It resulted in not only expensive apparatus, but also poor mobility, narrow application scenarios and high technical hurdles for developers from outside the field to reuse the technology in other domains.

In 2000, Japan's WIDE project[1] proposed to use the Internet to transfer RTK correction data, explicitly, the RTCM (Radio Technical Commission for Maritime Services) standard message [9]. TCP/UDP were adopted as the communication protocols, and a request-response paradigm was used to replace radio broadcasting, enabling bidirectional communications between servers and clients. Thus, users did not need to know the addresses of reference stations; instead, a centralized management agent would responded to the user with the optimal reference station among the network. The proposed architecture was able to reuse the general-purpose, worldwide Internet infrastructure, hence improving the availability, openness and on-demand service quality without imposing extra hardware requirements on users. Meanwhile, the major drawbacks lay in 1) Extra transmission latency was caused by introducing the Internet as an intermediate; 2) as geographic distribution was not specified in TCP/UDP standards, extra protocols were needed; 3) TCP/UDP provided only the most basic data package transport, while remaining issues like data encoding/decoding, inter-system and inter-application interoperability etc., must be handled all by atop applications.

Shortly after, Networked Transport of RTCM via Internet Protocol (NTRIP) was announced, which was based on the most adopted World Wide Web (WWW) protocol HTTP, and soon became the de-facto standard for Internet-transported DGNSS/RTK [10]. By reusing the Internet infrastructure and open Web technology stack, NTRIP is not only able to encapsulate differences among various heterogeneous systems and generalize data communication between DGNSS servers and clients, but also enable the establishment of a unified, open interface that is compatible with the existing IoT web architecture, which is also known as the "Web of Things" (WoT) [11]. Thus, it has laid the foundation for web servitization of RTK technology, paving the road towards integration with IoT via the process known as the "web service mashup" [12].

Currently, only sparse studies on DGNSS/RTK web services can be identified. Lim et al. proposed a server-based, thin-client RTK service framework, by migrating the computation of RTK solutions from client end to the Web server [13]. A web-based API was provided for users to submit NTRIP data to the server and acquire computed RTK results, hence the users did not need to develop RTK algorithms by themselves or use manufacture-specific software. However, this work was targeted at service providers rather than service users.

[1]https://www.wide.ad.jp/index_e.html

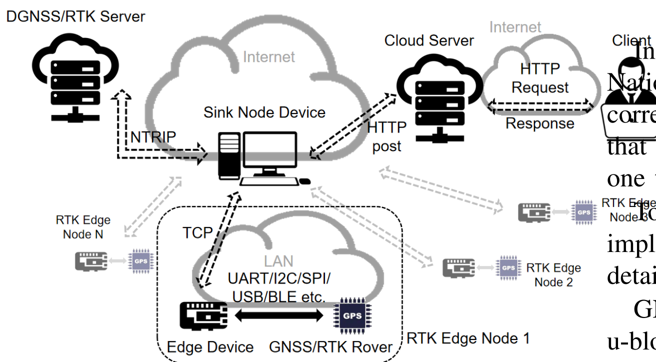
Similarly in [12], an RTK web service that complied with the standard Web service specifications initiated by the Open Geospatial Consortium (OGC) was proposed. It allowed end users to upload raw GNSS observation data from the rover, the reference station and corresponding navigation satellite parameters respectively, so that the server would then conduct the RTK computations and return the results via the same web interface. The authors further showcased integrating RTK service with other web services like Flickr, Google Maps into a geo-visualization application. However, the proposed web service leveraged historical data logs and only applied to post-process scenarios rather than the real-time ones.

Although the introduction of Web technology stack has improved interoperability, data and architecture openness, most RTK-enhanced applications were still anchored in highly specialized domains (such as geodetic survey and geofencing) and heavily relied on domain-specific standards, hardware and software. It hence failed to grasp broader, cross-domain scenarios that are close to people's daily life. Specifically in smart city applications, it is significant for RTK technology to be not only adaptable to complicated, always changing urban environment, but also able to incorporate large-scale, sometimes highly mobile IoT nodes that spread citywide. For example, Chiappini et al. proposed to combine RTK with cameras and laser scanners to automatically generate geo-referenced 3D point clouds of streets and buildings in city scale, and argued that this kind of tools can "support local authorities for the creation and operation of the Smart City." [14]; Roman et al. implemented an RTK-embedded smart car prototype for detecting accurate parking space in the city, however, the efficacy of this solution depended on large-scale crowdsensing for gathering enough data [15]; Kung et al. proposed an RTK-enhanced, multi-drone performance system, the practical usage of which entailed an architecture allowing different types of drones to connect or disconnect agilely [16].

Distinguished from previous RTK web servitization research, which were mostly technologically demanding, domain specific solutions, this study aims to establish a highly practicable, general-purpose architecture for designing and developing RTK-enhanced IoT/smart city services. Particularly, we have based our solution on a generalized, IoT-compatible technology stack that well corresponds to the highly versatile, dynamic application scenarios of smart city.

III. System Architecture Design

The design and development of DGNSS/RTK-enhanced IoT and smart city applications entail functionality and technical schemes very differently from the traditional specialized ones. The first and foremost concern is that instead of rural districts, these applications are more often launched in urban or suburban areas; rather than stationary observation, it is more likely for RTK rovers to be working frequently in a dynamic and mobile manner. Thus, it requires to well balance hardware loose-coupling, mobility between positioning precision. Moreover, to support flexible connections and disconnections of multiple heterogeneous nodes, a loose coupling, distributed computational architecture is a prerequisite. Finally, it should not be either too time/effort consuming or too technologically demanding for IoT developers to establish their desirable business logic. There is supposed to be certain mechanism, e.g., service mashup, rule configuration etc., to enable users with varied technical skills to assemble RTK positioning and other resources into value-added services. Therefore, well encapsulation of off-bottom details and ready inter-system/application interoperability are of specific significance. In consideration of these aspects, we hence propose the following DGNSS/RTK web servitization architecture, as shown in Fig.1:



Architecture.png

Fig. 1. GNSS/RTK Web Servitization Architecture.

- RTK Edge Node: We use compact edge devices such as Arduino, raspberry pi etc., for replacing physical connections with wireless bidirectional communication between the RTK Rover and the sink node device. If necessary, the edge device can also take over data preprocessing, e.g. storing, formatting, filtering of the raw GNSS measurements. While in our implementation, the edge device is only responsible for forwarding the raw GNSS data towards its TCP port, and sending back the correction data to the rover. An USB-C cable is used to physically connect the edge device and the rover, however, it is also feasible to replace with other compatible interfaces like UART, I²C etc. Together the edge device and the rover form an RTK edge unit as a whole.
- Sink Node Device: A sink node device is where data from multiple RTK edge units aggregate. Compared with the edge device, sink node devices possess richer computational and communicational resources, which usually can be desktop or laptop PCs. The tasks of a sink node include retrieving RTK correction data from an NTRIP server, conducting complicated data processing and eventually transmitting corrected RTK measurement to a cloud server. To achieve speedy data transmission, a sink node should be as "close" as possible to the downstream edge nodes in the sense of network topology. Therefore, we adopt a star network structure where all edge nodes are connected to the sink node via Local Area Network (LAN).
- Cloud Server: A cloud server will host real-time RTK data streams or static log files, and provide a standard web service interface for all clients, either end users or other web services, to retrieve the data. The communication between the server and the clients complies with a request/response paradigm using only standard HTTP methods, such as GET, POST etc. For the purpose of prototype briefness, we developed a loose-coupling, stateless interface by following the RESTful service style [17], but note that there also exist other alternative standards and ready-made tools for web servitization, e.g., OGC's Web Processing Service (WPS)[2], Keyhole Markup Language (KML)[3] for standard web geographic data format and PostgreSQL[4] for querying geographical data etc.

In addition, we utilize NTRIP service provided by the National Land Survey of Finland (NLS) for DGNSS/RTK correction. There are also many other free or paid services that provide different service qualities for users to select the one that best suits their needs.

To verify the feasibility of proposed architecture, a typical implementation based on COTS devices was developed, with detailed specifications depicted as below:

GNSS/RTK rover: A SparkFun GPS-RTK2 Board with u-blox ZED-F9P on board, which was a multi-frequency (L2OF, L2C, E1B/C, B2I, E5b, L1C/A, L1OF, B1I), multi-constellation (BeiDou, Galileo, GLONASS, GPS/QZSS) GNSS/RTK receiver module. As a result of compromising measurement precision with device portability and mobility, a Taoglas MagmaX AA.171 patch antenna was used instead of large-volume, stationary survey antenna.

Edge device: A Raspberry Pi Zero W with Raspbian OS 10 (buster) installed, powered by a normal 5000mA USB power bank. Since it was physically coupled with the rover device, we 3D printed a case to accommodate all electronic components. The Raspberry Pi ran a socat application to continuously forward the received RTK data towards the TCP port 2101 at a time interval of 10ms. The overall device size was 55*40*90 mm, and the weight was 165g (without power bank). By physically decoupling from other components, the edge device became compact and light-weighted, hence able to support fast prototyping and test of various mobile IoT applications.

Sink node device: A MSI GE63 8RF-004CN laptop computer (Intel i7 6 cores, Windows 10 Home OS, 8GB*2 DDR4 memory, 1 TB HDD+265GB SSD hard disk) was used as the sink node device. This PC ran a GNSS evaluation software, u-center[5], for continuously listening to the edge device's TCP port, streaming the raw GNSS data in NMEA format to the NTRIP server and then receiving RTCM messages back. We adopted the u-center because it was fully compatible with the f9p module and with multiple ready-made functionalities integrated, such as data encoding/decoding, remote communication, data logging etc. Surely technology-savvy users are

[2] http://docs.opengeospatial.org/is/14-065/14-065.html
[3] http://docs.opengeospatial.org/is/12-007r2/12-007r2.html
[4] https://www.postgresql.org
[5] https://www.u-blox.com/en/product/u-center

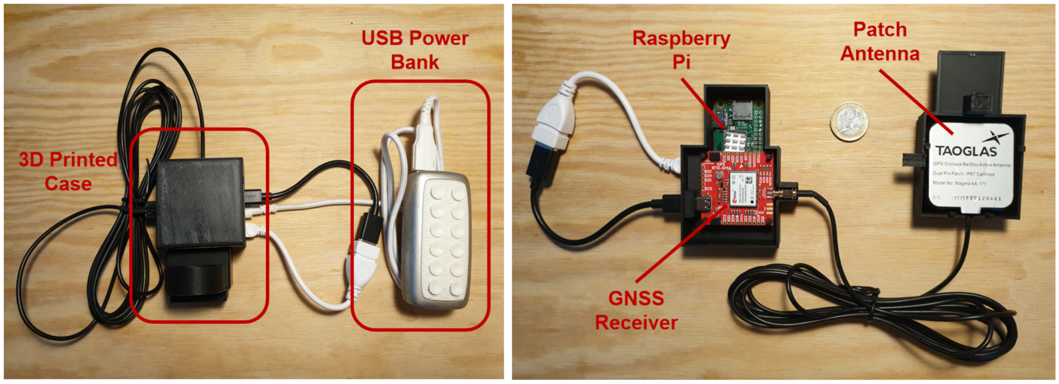


Fig. 2. Hardware prototype of the RTK Edge Unit.

also free to employ other open-source tools, e.g., rtklib[6], or self-developed programs instead of proprietary software. While u-center was configured to store the corrected RTK data in a local log file, we also wrote a Python script running on the same PC for streaming the logged data to the cloud server via HTTP at the same time.

Cloud server: We established a web server leveraging C-Pouta service provided by CSC (IT Center for Science Ltd.), Finland. The server was an instantiated virtual machine initially assigned with 3.9 GB RAM, 3 virtual CPUs and 80GB disk. It was also associated with an IP address (195.148.**.66). Another Python script was running on the cloud server to get and post data via HTTP through 8080 port. Thus, real-time RTK positioning data can be retrieved anytime and anywhere by accessing the corresponding URL using web browsers or any other HTTP-compatible applications.

As shown above, the whole system was constructed by only COTS hardware and free/open software, with a total cost around several hundred euros. By physically decoupling the RTK rover from data processing/computing devices, the system achieves better potability and mobility that will facilitate more versatile IoT and smart city application scenarios. Meanwhile, the distributed computing structure consisting of edge, sink and cloud devices can flexibly coordinate multi-node, dynamic tasks.

IV. Usage Demo and Evaluation

In this section, we provided a brief usage demo for showcasing how to rapidly integrate the proposed RTK service with other web service component, i.e. the Google Maps in this case. We adopted HSML, an IoT Web mashup toolset for composing different RESTful web services using a HTML-like domain specific language [18], [19]. To specifically note that, we created this real-time 2D trajectory geovisualization application only for demonstration and test purposes, and additional IoT nodes were not present. However, any other sensor/actuator/IoT device can also be integrated in a similar way, as long as they possess standard RESTful interface.

[6]http://www.rtklib.com

Moreover, to evaluate the system performance under a restricted condition as close to real-world application environments as possible, we tested our prototype in an urban district, which was a bridge within the city center area with dense surrounding buildings (see Figure 3 left). According to the available quality of satellite signals, which is likely to degrade due to disruptive multipath and non-line-of-sight (NLOS) interferences such as occlusion from trees and buildings, RTK is able to achieve the highest positioning precision below an inch only with a fixed solution and secondary precision around a few inches with a float solution. It is noteworthy that rather than the average or best system performance, we intentionally investigated the "lower-limit" precision of RTK positioning under float solution mode in this experiment. While a higher positioning precision can be expected on general occasions, GNSS signal deterioration caused by environmental obstruction can also be mitigated by either integrating extra sensors like Inertial Measurement Unit [20] or simply replacing with larger helix antennas (e.g. Taoglas AQHA.11), which may trade off the system mobility/portability to some degree.

In the experiment, data collection was done by moving the prototype along one of the guard rails on the bridge, from one end to the other. This process was repeated four times (round way for each guard rail). Meanwhile, the collected data was transmitted to the cloud server and became web accessible in real-time manner. We then retrieved the positioning data stream and further composed it with the Google Maps Web API to visualize the 2D motion trajectory. To achieve this, the following text was input to the online HSML editor[7]:

```
<loc id="RTKNode1" type="realtime" viz="point"
   lat="http://195.148.**.66/node1/lat" lnt=
   "http://195.148.**.66/node1/lnt"></loc>
```

Among the line, "http://195.148.**.66/node1" referred to the RESTful URI of the servitized RTK edge node; "type" attribute specified the resource as a real-time data source. Optionally, users can also upload RTK data logs in XML/CSV format as static data source. Attribute "viz" specified the data visualization style, which was dot plot in this case.

[7]http://www2.kmd.keio.ac.jp/~ruowei.xiao/hsml/

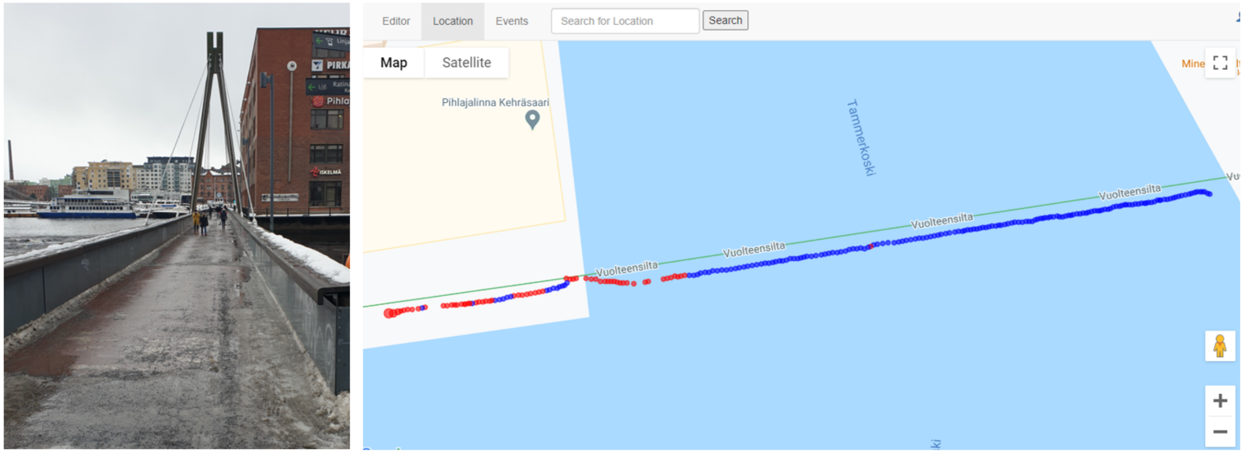


Fig. 3. 2D Visualized Trajectory via HSML (Right) and Actual Test Location (Left).

The visualized trajectory was shown in the right of Fig. 3. Within the visualized trajectory, the blue dots referred to the data collected with RTK float solution, while the red ones were collected when the device dropped out of float mode. In the post-test analysis, we deployed statistical methods to evaluate general system performance regarding the following indices:

- Float mode stability: As can be observed from Fig. 3, part of the trajectory, specifically near the buildings at the end of the bridge, indicated that the RTK rover fell out of the float mode from time to time. In average, stable float mode accounted for 78.6% of the overall data set.
- Positioning precision: The positioning data was collected following strictly the contour of the bridge. Ideally, if the error caused by manually moving the device can be ignored (usually around 3 to 5 cm), the collected data was supposed to be linearly distributed. Therefore, the measurement precision can be estimated approximately by calculating the standard deviation. The mean square error of all data σ_{all} was 0.877, while the mean square error of only RTK Float data σ_{float} was 0.865, which implied the average distance error was around 0.8 meters. Our performance test showed that submeter precision can be achieved by using the proposed prototype in urban environment, and a higher positioning precision of 0.06 meters can be achieved, if the rover yields a fixed solution.
- Time delay: Similar to any other web-based services, the introduction of Internet infrastructure and application-layer protocols imposes extra communication and computation costs. As a result, there is always an offset between interoperability and system latency. The time delay of the proposed architecture is mostly generated by the edge-sink and sink-cloud data transfer, as well as web servitization on the cloud server. While the former highly depends on actual tele-communication networks and may vary greatly according to specific conditions, we intend to provide a referential estimation on the magnitude of the latency here. In the demo, the rover's sampling frequency was set to 10 Hz, and every 100ms the data was transmitted once from the sink node to the cloud server. While we considered that the current settings would be adequate for most general-purpose IoT and smart city applications, both settings can be reconfigured to meet different requirements. The mean time cost from each GNSS measurement recorded by the rover to the time when it arrives at the cloud server was calculated, with both system clocks calibrated according to the UTC time. The average time delay was 0.35s at maximum, which indicated a system latency at 10^{-1} second level. The time expense for a client to access the Web service is considered out of the system latency.

Based on the above results, we further compared our proposed system with several existing RTK solutions, the details of which were reported in [21]. To note that, the benchmarking systems were all tested under good weather and environment conditions without obstacles. As shown in Table I, even when degraded to RTK float mode, the positioning precision of our prototype still exceeded its equivalent or mid-class GNSS solutions (Thales MobileMapper Pro, Stonex S5); while slightly inferior to the high-end equipment (Stonex S7-G), the proposed system still possessed advantages in price, device size and weight, which indicates a good overall balance between system performance, cost and portability.

V. Conclusion

This research intends to exploit the DGNSS/RTK positioning for enhancing IoT and smart city applications from a practical engineering perspective. We have proved the feasibility to increase the positioning precision of current IoT services to at least decimeter level using only COTS devices and standard Web technology stack. Compared with existing RTK solutions, the proposed software and hardware architecture further extends the overall mobility, loose coupling, as well as the interoperability that allows quick mashup with various

TABLE I
COMPARISON BETWEEN PROPOSED PROTOTYPE AND EXISTING RTK SOLUTIONS

	Price (USD)	Size (mm)	Weight (g)	Precision (m)
Thales MobileMapper Pro	500	165*74*31	227	1.43 (GNSS)
Stonex S5	1400	119*86*32	290	1.37 (GNSS) - 0.36 (RTK Fixed)
Stonex S7-G	2300	234*99*56	895	0.47 (GNSS) - 0.07 (RTK Fixed)
Proposed Prototype	300	90*55*40	165	0.8 (RTK Float) - 0.06 (RTK Fixed)

IoT components. It contributes to a both financially and technologically affordable solution, thus facilitating the ideations and innovations in various IoT and smart city domains.

As an on-going study, we are currently developing more sophisticated applications and use cases. Comprehensive evaluation can further be expected in an extended version of the paper. We believe that by merging RTK positioning with advanced IoT technologies such as Unmanned Aerial Vehicle (UAV), wearable technology, robotics and automation, it will foster service innovation and novel business values in a wider range of application domains, e.g. location intelligence, smart vehicle network, somatosensory tourism etc., thus further enhancing the overall urban smartness, service quality and degree of automation in the future smart city.

ACKNOWLEDGMENT

This publication has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Sklodowska-Curie grant agreement No 833731, WEARTUAL, and Business Finland decisions 5654/31/2018, GArMEnt.

REFERENCES

[1] D. Del Gaudio and P. Hirmer, "Fulfilling the IoT Vision: Are We There Yet?," in IoTBDS, 2020, pp. 367–374.
[2] Z. M. Win, F. Meyer, Z. Liu, W. Dai, S. Bartoletti, and A. Conti, "Efficient multisensor localization for the Internet of Things: Exploring a new class of scalable localization algorithms," IEEE Signal Processing Magazine, vol. 35(5), 2018, 153–167.
[3] L. Wanninger, "Introduction to network RTK," in International Association to Geodesy meeting, Working Group, vol. 4, No. 1, 2004.
[4] H. Xu, "Application of GPS-RTK technology in the land change survey," Procedia Engineering, vol. 29, 2012, pp. 3454–3459.
[5] L. Poluzzi, L. Tavasci, F. Corsini, M. Barbarella, and S. Gandolfi, "Low-cost GNSS sensors for monitoring applications," Applied Geomatics, 2019, pp. 1–10.
[6] H. Lan, M. Elsheikh, W. Abdelfatah, A. Wahdan, and N. El-Sheimy, "Integrated RTK/INS navigation for precision agriculture," in Proceedings of the 32nd International Technical Meeting of the Satellite Division of The Institute of Navigation, 2019, pp. 4076–4086.
[7] A. Jokinen, A. Pirsiavash, M. Mendonça, R. Yang, H. Yu, M. Tsai, and Y. F. Tseng, "Challenges and Opportunities in Mass Market RTK," in Proceedings of the 33rd International Technical Meeting of the Satellite Division of The Institute of Navigation, 2020, pp. 2896–2910.
[8] W. Roberts, J. Critchley-Marrows, M. Ivanovici, M. Siutkowska, V. Barreau, L. Arzel, and A. Piech, "High Accuracy Navigation for the Mass Market-the FLAMINGO Initiative," in 2020 European Navigation Conference, 2020, pp. 1–10.
[9] H. Hada, H. Sunahara, K. Uehara, J. Murai, I. Petrovski, H. Torimoto, and Kawaguchi, S, "DGPS and RTK positioning using the Internet," GPS solutions, vol. 4(1), 2000, pp. 34–44.
[10] T. L. Dammalage, P. Srinuandee, L. Samarakoon, J. Susaki, and T. Srisahakit, "Potential accuracy and practical benefits of NTRIP protocol over conventional RTK and DGPS observation methods," 2006, Map Asia vol.29.
[11] Guinard, D., and Trifa, V. (2009, April). Towards the web of things: Web mashups for embedded devices. In Workshop on Mashups, Enterprise Mashups and Lightweight Composition on the Web (MEM 2009), in proceedings of WWW (International World Wide Web Conferences), Madrid, Spain (Vol. 15, p. 8).
[12] D. Yoshida, E. Realini, M. Reguzzoni, and V. Raghavan, "Integrating low-cost RTK positioning services with a web-based track log management system," Applied Geomatics, vol.5(2), 2013, pp. 99–108.
[13] S. Lim, and C. Rizos, "A new framework for server-based and thin-client GNSS operations for high accuracy applications in surveying and navigation," in Proceedings of the 20th International Technical Meeting of the Satellite Division of The Institute of Navigation, 2007, pp. 2169–2177.
[14] S. Chiappini, A. Fini, E. S. Malinverni, E. Frontoni, G. Racioppi, and R. Pierdicca, "Cost Effective Spherical Photogrammetry: a Novel Framework for the Smart Management of Complex Urban Environments," The International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences, vol. 43, 2020, pp. 441–448.
[15] C. Roman, R. Liao, P. Ball, S. Ou and M. de Heaver, "Detecting on-street parking spaces in smart cities: Performance evaluation of fixed and mobile sensing systems," IEEE transactions on intelligent transportation systems, vol. 19(7), 2018, pp. 2234–2245.
[16] C. M. Kung, W. S. Yang, T. Y. Wei and S. T. Chao, "The fast flight trajectory verification algorithm for Drone Dance System," in 2020 IEEE International Conference on Industry 4.0, Artificial Intelligence, and Communications Technology, 2020, pp. 97–101.
[17] R. T. Fielding, "Architectural styles and the design of network-based software architectures,", 2000, University of California, Irvine.
[18] R. Xiao, Z. Wu and K. Sugiura, "A semantic HTML based approach for geosensor media," GeoInformatica, vol.22(1), 2018, pp. 105–126.
[19] R. Xiao, Z. Wu and D. Wang, "A Finite-State-Machine model driven service composition architecture for internet of things rapid prototyping," Future Generation Computer Systems, vol. 99, 2019, pp. 473–488.
[20] Niu, Z., Nie, P., Tao, L., Sun, J. and Zhu, B. "RTK with the assistance of an IMU-based pedestrian navigation algorithm for smartphones," Sensors, vol. 19(14), 2019, pp. 3228.
[21] Catania, P., Comparetti, A., Febo, P., Morello, G., Orlando, S., Roma, E. and Vallone, M, "Positioning accuracy comparison of GNSS receivers used for mapping and guidance of agricultural machines," Agronomy, vol. 10(7), 2020, pp. 924.