Path Markup Language for Indoor Navigation

Yang Cai, Florian Alber, and Sean Hackett

Carnegie Mellon University, 4720 Forbes Ave., Pittsburgh, USA
ycai@cmu.edu,falber@andrew.cmu.edu,shackett@andrew.cmu.edu

Abstract. Indoor navigation is critical in many tasks such as firefighting, emergency medical response, and SWAT response, where GPS signals are not available. Prevailing approaches such as beacons, radio signal triangulation, SLAM, and IMU methods are either expensive or impractical in extreme conditions, e.g. poor visibility and sensory drifting. In this study, we develop a path markup language for pre-planning routes and interacting with the user on a mobile device for real-time indoor navigation. The interactive map is annotated with walkable paths and landmarks that can be used for inertial motion sensor-based navigation. The wall-following and landmark-checking algorithms help to cancel drifting errors along the way. Our preliminary experiments show that the approach is affordable and efficient to generate annotated building floor path maps and it is feasible to use the map for indoor navigation in real-time on a mobile device with motion sensors. The method can be applied to intelligent helmets and mobile phones, including potential applications of first responders, tour guide for buildings, and assistance for visually impaired users.

Keywords: indoor navigation, SLAM, IMU, EMS, firefighting, markup language, map.

1 Introduction

Indoor navigation is a growing routine in modern urban lives. People need to navigate through large infrastructures such as airports, hospitals, museums, schools, shopping malls, subways, and factories. It is critical for extreme cases such as firefighting, emergency medical responses, and SWAT team responses. It is logical to consider indoor navigation like a GPS navigator, including display of the floor map, updating current position on the map, and updating the landmarks nearby. Unfortunately, GPS signals are normally weak or available in buildings. It is rather difficult to sense the location of the user without infrastructure-dependent sensory systems. Furthermore, most available floor plans contain either too much irrelevant information such as toilet seats and furniture in the room, or too little navigational information, e.g. no connection to the user's location nor user's orientation. First responders often have a brief look at the paper drawings of a building and try to memorize it before entering. To make the matter worse, first responders often encounter the buildings with heavy smoke and poor visibility, where prevailing vision-based navigation methods would fail because we virtually have to navigate in a dark environment.

Who navigates well in the dark? The answer is blind people. In the 1970's, there was a blackout in New York City. Many people had to walk to their home without any

ICCS Camera Ready Version 2020
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-50436-6_25

light. Some blind people volunteered to guide sighted people home. In this study, we want to explore a markup language to annotate building's floor plans so that the user can navigate through the building with inertial motion sensors without infrastructure-dependent beacons.

2 Related Studies

Similar to a GPS navigation system, we need a digitally annotated map that works with positioning systems. There are many geographically tagging markup languages and geocodes. For example, Keyhole Markup Language (KML) enables geographically tagging buildings, streets, and events with GPS coordinates [1]. KML has been adopted by Google Maps and Google Earth and it is a part of the Open Geospatial Consortium (OGC) [2]. OpenStreetMap [3] is an open source for millions of footprints of buildings, contributed by users. In addition to the GPS coordinates, there is the Military Grid Reference System (MGRS) [4] standardized by NATO militaries for locating points on Earth. MGRS is a multi-resolution geocode system that consists of a grid zone designator, followed by the 100,000-meter square identifier and then the numerical location with a pair of easting and northing values. The longer digits, the more accuracy. For example, 4QFJ 123 678 has a precision level of 100 m and 4QFJ 12345 67890 can reach a precision level of 1 m. Geocode can be also embedded into images, for example, GeoTiff image format turns pixels into GPS coordinates with its metadata [5]; and HDF (Hierarchical Data Format) standardizes NASA Earth Observation System (EOS) data products, including multiple spectrum satellite sensory data and multiple object data files [6].

Outdoor map markup languages and geocode provide starting points and context for indoor navigation. A few indoor geocode and markup languages are extensions of outdoor ones. OGC's CityGML [7] is an open data model and XML-based format for the storage and exchange of virtual 3D city models. The aim of the development of CityML is to reach an international standardization for the basic entities, attributes, and relations of a 3D city model. Some schematic descriptions are relevant to emergency responses such as `RoofSurfaceType`, `WallSurfaceType`, `FloorSurfaceType`, and `GroundSurfaceType` [9]. The three dimensional descriptions about the buildings, bridges, streets, and grounds provide important information about the elevation or depth of city objects and benefit to indoor navigation. In 2014, OGC further released IndoorGML in 2014 [8], which is a data model and exchange format for the representation of the indoor navigation aspects. IndoorGML provides a topographic and semantic model of indoor space that connects to related standards like CityGML, Open Floor Plan [9], and OpenStreetMap. IndoorGML also describes multiple layers of indoor components including topographic space, WiFi sensor space, and RFID sensor space. All of the geocode, standards and exchange formats enable data sharing and emergency services [10]. In addition, OGC also released Augmented Reality Markup Language 2.0 (ARML) [11] to allow users to describe virtual objects in an augmented reality (AR) scene with their appearances and their location related to the real world. ARML 2.0 also defines a

ICCS Camera Ready Version 2020
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-50436-6_25

script language to dynamically modify the AR scene based on user behavior and user input, e.g. turning head or raising a hand, etc. In summary, the OGC has moved from 2D geocode to 3D geocode, and moved from cities to building, from outdoor to indoor, and from schematic world to virtual reality, and augmented reality worlds. Perhaps, the most valuable outcome of the OGC geocode and standards lies in its potential of crowdsourcing for massive geocoded data about buildings, interiors, and floor plans. However, despite a broad spectrum of indoor facility markup languages, the existing methods are too abstract and complicated; and there are many gaps to fill in order to be useful in indoor navigation. For example, how to convert a floor plan drawing in PDF format into a geocoded floor plan? How to use the floor plan interactively in real-time indoor navigation?

Localization is the most critical component in indoor navigation. We need to know where the user is and which direction the user is heading in the building. Traditional Dead Reckoning, or Inertial Motion Unit (IMU) sensor-based approaches can work in totally dark environments, but they have notorious accumulative drifting problems over a period of time [12]. Recent Renaissance of IMU sensor-based methods are enhanced for better accuracy by placing IMU sensors on shoes [13] or fusing with other sensors [14]. The prevailing approach is the beacon-based localization, including ultrasound [15], LoRa [16], and WiFi [17]. Installing and calibrating beacons in a building are expensive and there are wall-attenuation problems [18]. There are growing technologies of infrastructure-free localization by mobile beacons [19] or collaborative positioning [20]. Rapidly growing mobile robotics technologies bring new dimensions to indoor navigation. Simultaneous Localization and Mapping (SLAM) algorithm [21] has been popular for 3D modeling from motion, tracking and mapping at the same time. Single RGB camera-based visual SLAM can generate the motion trajectory in a relative 3D space. Stereo and RGB-Depth camera-based SLAM yield absolute 3D coordinates of the trajectory. Visual SLAM is computationally expensive and it often fails in poor lighting, smoky, or feature-less environments such as a painted white wall. Some RGB-D sensor-based SLAM incorporate IMU sensors for more accurate localization results. LiDAR-based SLAM can work in the dark by tracking the 3D point clouds but its cost is very expensive [22]. Thermal IR cameras can also be used for SLAM but its images are rather low-resolution and it's expensive as well [23].

In summary, there is no silver bullet in indoor localization. The technologies for large-scale localization in normal environments or extreme conditions such as fire and smoke are not mature yet. There are gaps between the available technologies and applications. For example, there is little connection between the geographic markup languages and indoor navigation technologies. Many sensors need pre-calibration. Some of them such as magnetic field sensors need to be calibrated each time of usage. Self-calibration methods have implemented, for example, DJI drones use a motor to rotate the magnetic sensor before taking off [24]. A few novel concepts might pave the way for affordable and practical indoor localization, for example, the mobile device for helping visually impaired users to navigate indoors [25]. The assistive technology is affordable and interactive with a wall-following function. In nature, there are also other modalities for navigation based on smell intensity, lighting, sound,

ICCS Camera Ready Version 2020
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-50436-6_25

magnetic field, and simply tactile sensing [26]. Biomimicry teaches us to look into novel sensors and fusion algorithms, for example, the one-dimensional LiDAR and IMU sensor for first-person view imaging [14].

3 Path Markup Language

Although OGC's IndoorGML includes the IndoorNavigation module, it only provides standards and a high-level framework, rather than functional indoor navigation solutions. In this study, we propose the Path Markup Language (PML) as a data model and schema specifically for indoor navigation pre-incident planning and real-time navigation guidance. Currently, PML contains the following geocode elements: footprint, floor plan, path, landmark, and waypoint. These objects can be expressed in XML schema.

Footprint is the boundary of a building in a polygon. It can be extracted directly from Google Earth manually, or with machine vision. The coordinate points can also be downloaded from Google Maps or OpenStreetMap but it is not guaranteed because it depends on user online contributions. The coordinates are normally in GPS format and the sequence is counter-clockwise. The XME schema of Footprint is following:

```
<pml:Footprint>
        <pml:Polygon>
                <pml:coordinates>0,0 100,0 100,100 0,100 0,0
                </coordinates>
        </pml:Polygon>
</pml:Footprint>
```

The floor plan is a hierarchical structure of anchor points, footprint, rooms, paths, landmarks, and waypoints. It takes at least 3 anchor points to scale and align a floor plan to a georeferenced map such as Google Maps. Normally floor plan drawings are CAD drawings without any georeference. In PML, we overlay the floor plan to Google Maps by scaling, rotating and translating to extract the GPS coordinates directly.

Path is a critical element in indoor navigation. We assume a building is not an empty stadium. Instead, it contains walls, hallways, and barriers. We assume that humans and robots can only walk on the paths without breaking walls or barriers. This assumption helps to reduce the IMU-based localization drifting through walls. Instead, the estimated trajectory will be along the Paths. In the PML, a Path is omnidirectional and it is a sequence of line segments with widths.

```
<pml:Path>
      <pml:Name> "Hallway" </pml:Name>
      <pml:Width> 1 </pml:Width>
            <pml:Line>
                 <pml:coordinates>0,0 10,0 10,10 0,10
                 </coordinates>
            </pml:Line>
</pml:Path>
```

ICCS Camera Ready Version 2020
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-50436-6_25

Landmark is also a critical element in PML. Here we assume the IMU-based indoor navigation system has the sensory drifting problem and there are landmarks along the paths. When the user approaches the landmark nearby, the navigation system will send a confirmation request. Once the landmark is acknowledged, the localization track starts over again and the drift is canceled before it is accumulated further. A Landmark can be labeled with a symbol, for example, “E” as elevator and “S” as stairs. It also can be displayed with an icon.

```
<pml:Landmark>
      <pml:Name> “E” </pml:Name>
      <pml:Icon>elevatorIcon.pmg </pml:Icon>
                <pml:coordinates>5,5
                </coordinates>
</pml:Landmark>
```

Finally, Waypoint is the location of the user including heading and coordinates. It will be updated in real-time to display the current position and orientation of the user. Waypoints can be stored and displayed as a digital pheromone along the Paths. The pheromone trace can be turned off (0), without decay (1), or with decay (2).

```
<pml:Waypoint>
      <pml:Name> “Me” </pml:Name>
      <pml:Icon>RedArrow.pmg </pml:Icon>
      <pml:coordinates>7,9</coordinates>
      <pml:heading>245</pml:heading>
      <pml:trace>2</pml:trace>
</pml:Waypoint>
```

4 Mobile System Architecture

To implement PML, we aim to combine geographic markup language with real-time indoor navigation algorithms into a simple and affordable working system. The system contains two modules like displayed in Fig. 1: Map Generation and Navigation Guide. For the Map Generation module, a map can be generated for any building with a floor plan and that can be GPS tagged using downloaded or online Google Maps. The floor plan of the building is required in order to map the building's indoor features. This floor plan is imported as an image and overlaid with the footprint of the building from Google Maps which provides the GPS coordinates for navigation within the floor plan. The overlaid floor plan can be scaled and rotated to match the building's footprint on Google Maps. The map is then annotated with important features including the rooms, hallways, flights of stairs, elevators, doorways, etc. The annotated map is then exported as a csv file for example, which can then be used by the navigation guide application.

The Navigation Guide module utilises the generated map as a bounded region to navigate within. Tracking begins at the entrance to a building where Android Localisation (GPS, mobile network, etc.) is still accurate and can be used as a true starting point. This starting position can then be confirmed or be set manually if

ICCS Camera Ready Version 2020
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-50436-6_25

localistion is not accurate e.g. starting inside the building. The use of the `Android Step Detection` and `Android IMU` are then used to track the movement and orientation of the user through the buildings walkable paths. The wall-following algorithm reduces drifting by bounding the tracked path within the walkable paths and hugging corners. The landmark-checking is a manual approach to correct drifting when the user approaches a landmark. After reaching the landmark the user can confirm this and the user position will be updated to this landmark.

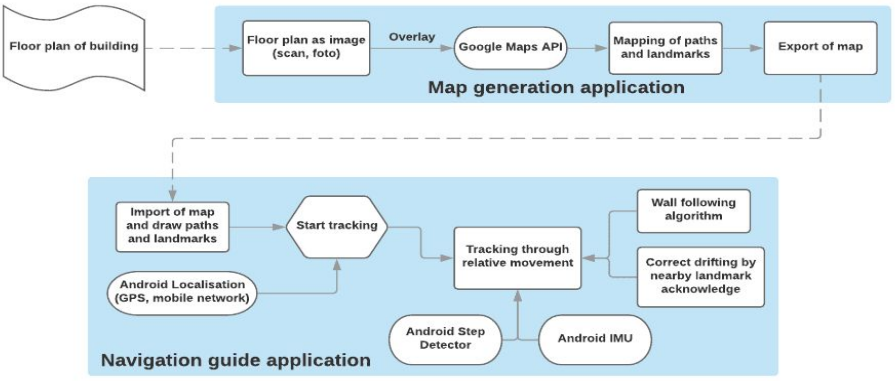

Fig. 1. System architecture

The pseudo code for map generation is as follows:

```
load Google Maps API;
user imports floor plan image as overlay;
transform image to fit GPS footprint on Google Maps;
lock overlay image based on true GPS coordinates;
start mapping based on overlayed floor plan:
        draw paths;
        set landmarks;
export map;
```

PML also includes real-time human-computer interaction interfaces. The pseudo Code for Indoor Navigation:

```
import map;
draw simple map of paths and landmarks;
set starting point based on Android Localisation;
start tracking:
        get step event from Android Step Detector;
        get imu data to calculate direction of movement;
        Wall-Following Algorithm;
        if user confirms landmark:
                update position and correct drift;
```

ICCS Camera Ready Version 2020
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-50436-6_25

5 Map Annotation

For the navigation app to have a floor plan with a walkable area and identifiable features, a geocode-annotated map must be supplied. After an image of the building's floor plan is overlayed with the Google Maps building footprint, all annotations that are added will be tagged with GPS coordinates. The path of walkable areas can be added as polygons and a number of landmarks including stairs, doors, elevators, corners can be tagged on the map with a corresponding icon shown in Fig. 2. The left image of Fig. 3 shows the annotated floor plan with paths and landmarks. The right images on Fig. 3 shows the display on an Android phone during the live indoor navigation.

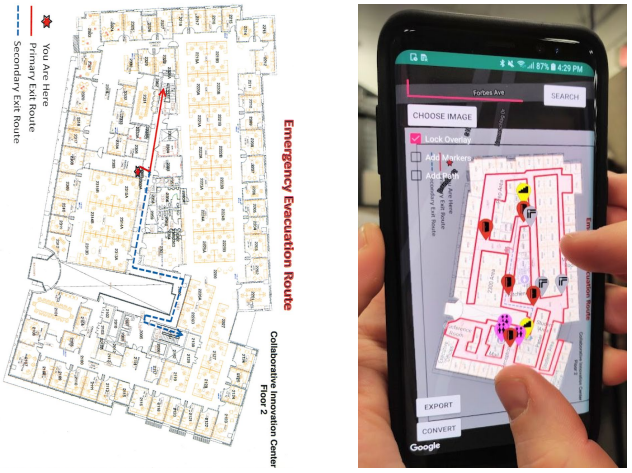

Fig. 2. The floor plan (left) and overlaid floor plan on top of Google Maps building (right)

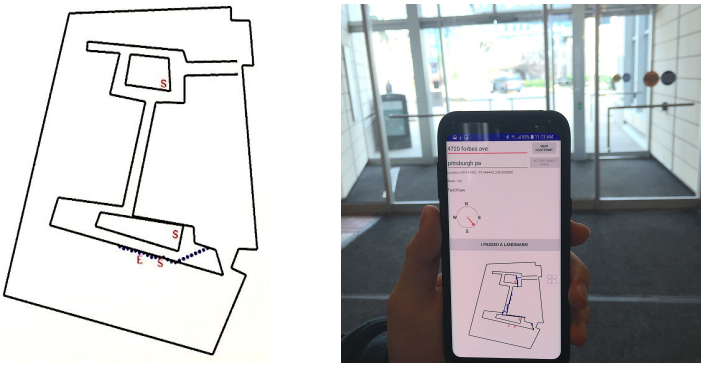

Fig. 3. Generated navigation map with footprint, paths and landmarks (left) and the display on an Android phone (right)

ICCS Camera Ready Version 2020
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-50436-6_25

6 Inertial Sensory Fusion for Steps and Orientation

The Navigation Guide application currently uses the built-in sensors of a standard Android phone. For this approach data from the IMU and from the magnetic field sensor is used to detect the relative movement inside a building. The first challenge is to detect the movement of the user, which can be defined for a person walking over the steps taken. This is a simple approach to track the person and enables it already to test our navigation concept. For later use it would be necessary to detect the size of the steps or calibrate the application for every user and its own step size. Additionally, for a real usage scenario it's necessary to update this movement detection to a more complex one, with which different moving styles can be tracked. Especially for the firefighter scenario, where a variety of walking, crawling, shuffle walking and other movement styles are frequently used. To detect the steps for a walking scenario on an Android phone, it's possible to detect the steps over a simple state machine based on the peaks in the acceleration data or to use the already built-in Step Detector in the Android SDK as described in [27]. During the first trial runs it was noticeable that the already built-in feature can detect steps very accurately. The Step Detector analyzes the acceleration data of the phone and based on that it detects a step movement, which triggers an event. This event can be used to account for the step and track the movement of the person.

With the detection of the movement of the user it is now important to detect where the user is heading. For that the approach is to use the magnetic field sensor and detect the direction of the movement, with the limitation that external magnetic fields can disturb the detection. In this application the data of the magnetic field sensor gets read out and it gets filtered by a lowpass filter in the form of an exponentially weighted moving average like:

$$C[i] = \alpha \cdot B[i] + (1 - \alpha) \cdot C[i - 1] \tag{1}$$

where, B[i] and C[i] are input and output on the discrete time-domain data with $i \in N_0$ and α as the corresponding weight variable.

Those filtered values are a relative measurement from the phone and now it's necessary to define the orientation of the phone to calculate the right directions. This is already possible with built-in functions of the Android SDK. First a so-called rotation matrix can be calculated, which transforms the magnetic field measurement based on the gravity measurement from the device coordinate system in a global coordinate system like described in [27]. The rotation matrix can then be used to calculate the orientation of the phone as Azimuth, Pitch and Roll. For our movement the Azimuth is especially important, because it describes the rotation around the gravity axis as the angle between the facing direction of the user and the direction to the magnetic north pole. For that reason the Azimuth can directly be used to define the orientation of the movement of the user.

ICCS Camera Ready Version 2020
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-50436-6_25

This angle could be inaccurate if only one measurement of the direction of the movement gets evaluated. For that reason the phone constantly measures the orientation and whenever a step is taken, we can average the measuring values during the step and use the average angle of the direction to place the next step. The new location of the step at $x[i]$ and $y[i]$ coordinates can be calculated over the current position and the average Azimuth φ_{avg} to:

$$x[i] = x[i-1] + \Delta x \cdot \sin(\varphi_{avg}), \qquad y[i] = y[i-1] + \Delta y \cdot \cos(\varphi_{avg}) \tag{2}$$

with the scaled step size Δx and Δy. Those scaled step sizes result from the scaling of the step size to the geological coordinates of the steps, which converts the feet per step to latitude and longitude per step. With that it's possible to define the next step and the direction of the movement.

7 Wall-Following Algorithm

Our major assumption is that the user only walks, crawls, or runs along the predefined paths. The user won't walk through a wall, for example. This assumption helps the indoor navigation algorithm to reduce the impact of the IMU sensory integral drifting errors. The wall-following algorithm estimates the user's position by the measurement data from the accelerometers and magnetic sensors. Due to the drifting error, the estimated position may drift away from the annotated path, for example, pass through the wall in the hallway. Therefore, we need to detect the collision between the estimated user location and the boundary of the path, e.g. a wall.

A collision occurs whenever the next step would be outside of a walkable path and the line of the step intersects with the path outline. For that reason it's possible to check for collisions after every new calculated step, if it would be outside an allowed path. The implementation of this check iterates over the path outline polygons and checks if there is an intersection with the connection line between the last step to the new one. After iterating over the path outline polygons, the result directly shows if a collision for this new step would occur. When the collision is detected, the algorithm corrects the trajectory and updates the user's position *along the border of the path*. Fig. 4 illustrates the wall following method.

Additionally, it could be possible that at a path crossing the tracking takes a wrong turn and follows the wrong path like displayed in the right illustration of Fig. 4. If the direction of those two paths diverge, then the tracking would sooner or later head into a wall. If we now account for the steps it would make in that direction and solve those conflicts with the above described collision detection, so it could be possible that those counted steps would reach over to the other path. If that is the case, we can cut this corner and move the position to the other path and go from there. With that we lose accuracy, but we can undo a potentially fatal error of the tracking approach.

ICCS Camera Ready Version 2020
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-50436-6_25

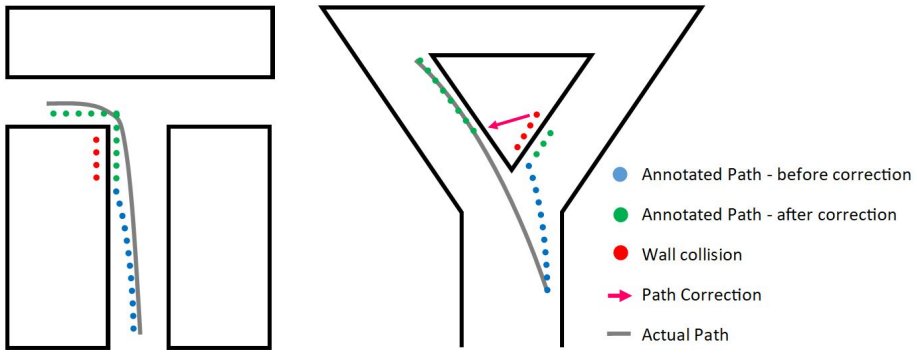

Fig. 4. Wall following (left) and corner cutting (right)

8 Indoor Navigation Experiments

A preliminary lab experiment has been conducted at an office building on the first floor including hallways, elevators and stairs. The length of the building is about 200 meters. The footprint and floor plan are available from Google Maps. We also obtained the scanned floor plan with details of rooms. After aligning the scanned floor plan with Google Maps, we obtained the geocode coordinates of the floor plan. We then annotated landmarks on the floor plan with elevators, stairs, and doors. Fig. 6 through 7 shows the results of four tests in the building. Our tests show that the wall-following algorithm indeed corrected the IMU drifting errors and put the trajectories back to the path. We found that corners on the floor plan might be helpful to be additional landmarks. The tests also show weaknesses of the algorithm to be improved, for example, the starting point of indoor navigation. We need to start tracking the location waypoint before entering the building when the GPS signal is available. We also found the collision detection algorithm may get stuck at a certain point when the walking angle is perpendicular to the wall. Besides, if the landmarks are too far apart, or the drifting error is too large, then the navigation might fail. Nevertheless, our initial experiments prove that this simple and affordable approach is feasible in a realistic building environment and have a reasonable accuracy within 1 - 1.5 m, which is acceptable to many humanitarian rescue and recovery tasks.

ICCS Camera Ready Version 2020
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-50436-6_25

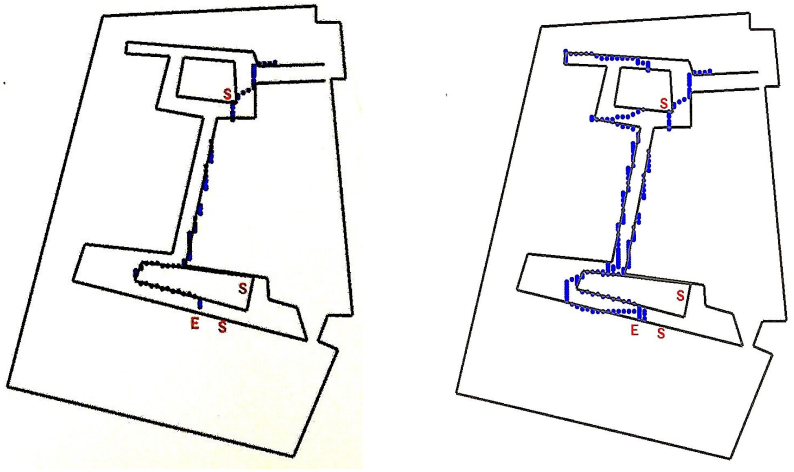

Fig.6. Indoor navigation experiment at the office building (part 1)

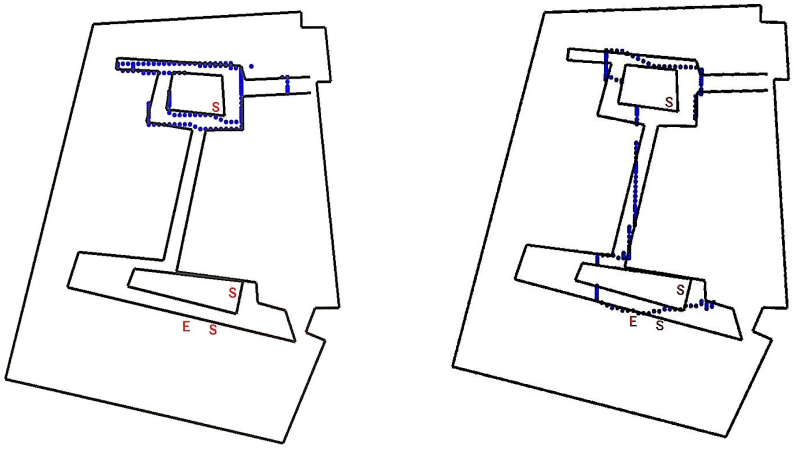

Fig.7. Indoor navigation experiment at the office building (part 2)

9 Conclusions

In this paper, we proposed a novel Path Markup Language for indoor navigation applications. We have shown that the mapping and navigation can be integrated into a modular system and can be used to solve a real world problem tackling indoor navigation without the use of expensive beacons. The Path Markup Language can be used to intuitively create a floor plan featuring landmarks for navigation purposes. The navigation application can then track a users position using pedometers and wall

ICCS Camera Ready Version 2020
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-50436-6_25

following algorithms to reduce errors. Landmark polling is then used to reset this error, allowing human supervision to overcome the inaccuracies of the system. The technology can be used for indoor navigation in large building facilities such as malls, airports, subways, museums, schools, office buildings, and factories for tour guidance, and emergency services.

10 Future Work

Several features are planned to be added to the mapping and navigation applications to further enhance their functionality. Adding multiple floors to a building in the mapping app and the ability to transition between these floors in the navigation app. This would be achieved through a push notification when in the location of stairs/elevator landmark, e.g. “Up 1 floor”, “Down 1 floor”. With these features a 3D view of the building's floors and the user's current position could be added.

Like satellite navigation systems in vehicles do not display a full road map, only a small section that the car is currently in, and that rotates based on the orientation of the car, this feature would improve the view and intuition of the navigation app.

Automatic sensor calibration is necessary for the future work, including walking stride calibration, magnetic sensor calibration, as well as altimeter calibration. The more sensors we throw in the system, the more calibration we need. Automatic calibration can be implemented with sensory fusion, e.g. calibrating stride with laser distance measurement and calibrating altimeter with satellite signals while the system is outside of the building.

In addition, we are developing the indoor navigation on a helmet for first responders to view the navigational information from a projected heads-up display (HUD). This would free up their hands for emergency services and enhance the augmented reality experience in harsh environments, for example, see-through smoke and walls.

Acknowledgement

This work was performed under the financial assistance award 70NANB17H173 from the U.S. Department of Commerce, National Institute of Standards and Technology, PSCR Division and PSIA Program. This project is also funded in part by Carnegie Mellon University’s Mobility21 National University Transportation Center, which is sponsored by the US Department of Transportation. The authors are grateful to the NIST PSCR Program Manager Jeb Benson for his comments and suggestions about the technical development of the hyper-reality helmet system.

References

1. Keyhole Markup Language (KML): https://developers.google.com/kml
2. Open Geospatial Consortium (OGC): https://www.opengeospatial.org/standards/kml/
3. OpenStreetMap: https://www.openstreetmap.org/#map=5/38.007/-95.844
4. Military Grid Reference System: https://en.wikipedia.org/wiki/Military_Grid_Reference_System

ICCS Camera Ready Version 2020
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-50436-6_25

5. GeoTiff, WikiPedia: https://en.wikipedia.org/wiki/GeoTIFF
6. HDF: https://nsidc.org/data/hdfeos/intro.html
7. CityGML: https://www.opengeospatial.org/standards/citygml
8. IndoorGML: http://www.indoorgml.net/
9. Schema of OGC CityML: http://schemas.opengis.net/citygml/building/2.0/building.xsd
10. OGC Hosts Indoor Location and Floor Plan Standards Forum: https://www.opengeospatial.org/pressroom/pressreleases/1122
11. ARML: https://www.opengeospatial.org/standards/arml
12. Dead Reckoning: https://en.wikipedia.org/wiki/Dead_reckoning
13. Xiao, Z. Wen, H., Markham, A. and Trigoni, N.: Robust Indoor Positioning with Lifelong Learning: https://www.cs.ox.ac.uk/files/9047/Xiao%20et%20al.%202015.pdf
14. Cai, Y., Hackett, S. and Alber, F.: Heads-Up LiDAR Imaging, to appear on IS&T, Electronic Imaging Conference, Jan. 20, 2020
15. Lin Q., An Z. and Yang L.: Robooting ultrasonic positioning systems for ultrasound-incapable smart devices: https://arxiv.org/pdf/1812.02349.pdf
16. Indoor positioning via LoRaWAN, indoornavigation.com: https://www.indoornavigation.com/wiki-en/indoor-positioning-via-lorawan
17. WiFi positioning system, WikiPedia: https://en.wikipedia.org/wiki/Wi-Fi_positioning_system
18. Zafari F., Gkelias A. and Leung K. K.: A survey of indoor localization systems and technologies. arXiv: https://arxiv.org/pdf/1709.01015.pdf
19. Wang Q., Lou H., Men A. and Zhao F.: An infrastructure-free indoor localization algorithm on smartphone, Sensors 18(10):3317: https://www.researchgate.net/publication/328067193_An_Infrastructure-Free_Indoor_Localization_Algorithm_for_Smartphones
20. Noh Y., Yamaguchi H., Lee U.: Infrastructure-free collaborative indoor positioning schema for time-critical team operations. IEEE Trans. on SMC. Vol. 48, No.3 (2018): https://ieeexplore.ieee.org/abstract/document/7747408
21. SLAM, WikiPedia: https://en.wikipedia.org/wiki/Simultaneous_localization_and_mapping
22. Mathworks. Implement SLAM with Lidar Scans (2020) https://www.mathworks.com/help/nav/ug/implement-simultaneous-localization-and-mapping-with-lidar-scans.html
23. Shin YS and Kim A.: Sparse depth enhanced direct thermal-infrared SLAM beyond the visible spectrum. arXiv:1902.10892: https://arxiv.org/abs/1902.10892
24. DJI Mavic Pro manual: https://dl.djicdn.com/downloads/mavic/20171219/Mavic%20Pro%20User%20Manual%20V2.0.pdf
25. Sato D., Oh U., Naito K., Takagi H., Kitani K., Asakawa C.: NavCog3: an evaluation of a smartphone-based blind indoor navigation assistant with semantic features in a large-scale environment. ASSET'17, Oct. 29-Nov. 1, 2017, Baltimore, MD, USA: https://www.ri.cmu.edu/wp-content/uploads/2018/01/p270-sato.pdf
26. Barrie D. Supernavigators: exploring the wonders of how animals find their way. The Experiment, LLC (2019)
27. Google, Android Developer API references: https://developer.android.com/reference

ICCS Camera Ready Version 2020
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-50436-6_25