2019 International Conference on Indoor Positioning and Indoor Navigation (IPIN), 30 Sep. – 3 Oct. 2019, Pisa, Italy

Multi-Platform Architecture for Cooperative Pedestrian Navigation Applications

Pekka Peltola
CAR-CSIC, Center of Automation and Robotics, Spanish Research Council
Madrid, Spain
{pekka.peltola@csic.es}

Reza Montasari
University of Huddersfield
Department of Computer Science
Huddersfield, England
{r.montasari@hud.ac.uk}

Fernando Seco
CAR-CSIC, Center of Automation and Robotics, Spanish Research Council
Madrid, Spain
{fernando.seco@csic.es}

Antonio R. Jimenez
CAR-CSIC, Center of Automation and Robotics, Spanish Research Council
Madrid, Spain
{antonio.jimenez@csic.es}

Richard Hill
University of Huddersfield
Department of Computer Science
Huddersfield, England
{r.hill@hud.ac.uk}

***Abstract*—As the location-based services market is growing, the plethora of navigation applications is strongly increasing. The solutions are made for single purposes and platforms. And the communication and information exchange in between, for example, pedestrian navigation applications from different solution providers, remain non-existent. To fill this gap, we developed a software architecture. The architecture abstraction enables easy and modular replacement of map, object, method, user, communications and sensor implementations between applications for many navigation purposes and technology platforms. The prototype design was tested using inertial, GNSS and UWB technologies on a MATLAB environment using multiple users, methods and sensors.**

***Keywords*—cooperative, multi-user, navigation, architecture, foot-mounted pedestrian dead-reckoning, GNSS raw measurements, UWB positioning, application development, robotics**

I. Introduction

The location based services market is predicted to experience a steep growth in the coming years. Measures to standardise design practises on multiple levels of innovation and product cycle could enhance the efficiency and reduce the time needed for the development of pedestrian navigation applications. Moreover, the design effort between different participants would work smoother using improved tools and well-defined common agreements between developers.

The scalability of the solution development for different target platforms and for the rapidly changing application demands would benefit from easy modifiability and replicability. We approach this issue by applying the well-known design structures from robotics into the pedestrian navigation application design. The main difference between robotics and pedestrian navigation is that an autonomous robot is often replaced by a person that is accountable of the behaviour changes.

The challenge is to find the optimal phases in the navigation application development process, where the architecture choices has the most impact between saving coding time and the level of application specificity. This together with enabling scalability are our objectives for the architecture development work conducted in this paper.

The section II delves into the background of robotics and navigation application design. Section III introduces the novel architecture format for navigation applications. Section IV implements the design on a real world cooperative use case showing the architecture applicability on distributed multi-platform scenarios. Section V raises the issues on the security of location information sharing. Section VI concludes.

II. Background

A. Robotics Application Architectures

Already, in many warehouses, robotics plays an important role in transporting goods. Robots thus act in the place of people in deciding on the behaviour of the transfer, navigation and the organisation of the goods. Lowell [1], lists fire fighters, factory workers and miners as examples, where improved navigation and robotics would make the work more efficient. And in the future, workers could be replaced with autonomous robots performing the more challenging and tougher tasks.

Authors in [2, 3, 4, 5, 6], describe the different aspects of a general robotic system high level architecture. The modularity of the design enables the reuse of the design blocks for different applications. Weiss, [4], claims that system block design abstraction level is the most useful abstraction level for block replication and reuse. Not necessarily the raw code replication, since the code itself may contain undocumented tweaks. The general robotics system level specification consists of sensing, interpreting, planning and actuation parts.

Martinez, Herrero and Rossi et al. [7, 8], discuss the trade-off between a centralised and a distributed computation implementation. This is often a compromise between energy consumption and computation time, and a question, into which processor the sub-task is best to be delegated. The necessary data collection can either be crowdsourced or based on prebuilt models. For a flexible architecture to be able to deal with these differing data characteristics they need to be covered by the novel architecture. This is, the storing and handling and the decision between the dynamic or the static data sources [9, 10]. For navigation applications, this choice can be described partly as a decision between using the simultaneous location and mapping (SLAM) or not using it.

The authors in [5, 7, 10, 11, 12, 13] describe the common robotics architecture. Deliberate behaviour contains higher intelligence decisions, while reactive functions do not necessarily contain other than binary inference intelligence. High intelligence means reinforced, super- or unsupervised learning or having a model/ self-tuning implemented. Data sharing [7] between the acting agents is of high importance. Martinez and Herrero [7] display a blackboard structure for data sharing. Single or multi-thread data access and event or polling based implementation rules need to be defined when using separate processes accessing the same data.

Architectural minimisation [2] and implementation parallelism need to be taken into account to avoid wasting resources. Pre-processing [14] of sensor information can be implemented at the sensor side to safe computational power at the fusion node. The work of Fernandez et. al. [14] includes four parallel navigation systems. Using multiple information sources, the commitment or the goal evaluation [3] is done by the artificial intelligence that belongs to the deliberate planning blackboard in the robotics architecture.

Lowell [1] mentions that to improve the information fusion, the incoming measurement information should be made either, of better quality or, to be more frequent. In addition, Weiss [4] approves that the design process has to be considered in parallel on each abstraction level. This way, the code stays consistent with the justifications. This is enabled by employing full encapsulation of objects for clarity, well defined interfaces, fault protection implementation and by implementing the top level to be generic and easy enough to be understood. This desire of encapsulation resembles the well-known MVC (Model, View and Control) in software architectures. The intra-, inter-, component and subsystem relationships need to be clearly defined. Further, [5, 10] state that the autonomy level of the system needs to be clearly understood for the current application, although for us here and for the pedestrian navigation applications in general, this is relatively low.

B. *Navigation Application Architectures*

Lowell [1], summarises the typical navigation system architecture. Dead-reckoning information is fused with absolute position or range measurements. The tracked position is represented with the reference coordinate system and map information containing obstacles and other users. Commonly a Kalman filter functions as the fusing information filter in the process. A particle filter is another often used alternative.

A context [15, 16] information helps in using the best filter alternative for the current sensor and environment configuration. Further, the history information of the track can be exploited as well, in determining, the following, most accurate state estimates [17], or adaptive algorithms like the novel zero updating method used for enhanced dead-reckoning tracking of the foot kinematics by Wagstaff [18].

In this paper, we highlight not only the history of the data (trajectory), but also the history of decisions (which code structure evaluated as true), in the design of the novel architecture for multi-platform navigation.

C. *Design Approach and its Expected Benefits*

In this work we apply and fuse the robotics system framework and the generic navigation system architecture. The work gathers the information from the robotics systems and the authors' knowledge in navigation application development together. The aim is to have an easily scalable and reusable architecture for navigation application design. This novel approach is explained in the next section and is followed by a minimal cooperative example to demonstrate the architecture and the thought process applicability and efficiency enhancement of the approach.

III. Multi-Platform Navigation Application Architecture

An entity runs on a platform and has a specific internal configuration. An entity is, called here, a part of an application or in some cases can be thought of as equivalent to an application. The Fig. 1. depicts the novel approach for a generic entity (application) architecture for navigation and robotics.

The external agents communicating with the entity are; The users who define the settings and goals for the entity; The sensors, either virtual, receiving data via communications interface, or real sensors with raw data not surrounded by communications protocols; The actuators that are being sent data in specific output formats, like displays; And the communications for outward data transfer infrastructure.

A settings file defines the configuration - that is, which implementation alternatives of IO, Code, Methods and State blocks the desired entity instantiation will contain. This settings file is first read, when an entity is created. The purpose of this entity architecture representation of Fig. 1. is to offer a generic solution to a wide variety of applications in navigation as well as in robotics. This entity can be then instantiated in multiple target configurations on differing platforms; Which is especially true in multi-user cooperative scenarios.

In the developed MATLAB code, the configuration for an entity is defined in the settings file. This file lays out what inputs, methods and states the entity will contain. An example excerpt Listing 1. is shown here (next page) defining the test case scenario settings for the user 1 (scenario described later).

The IO (input and output), in this first version as matrices, are defined last in IOlist. They are connected to the methods (number after the IO type) in the methods list. Methods, in turn, are connected to the states. (IO Format: "running identification number: IO type: connected methods in priority order: source file" and Method Format: "ID: type: connected states")

All settings file options are implemented in the Code. The order in which the numbers appear after the type of the input or output define the execution order. For example, when a measurement from the inertial measurement unit (IMU)

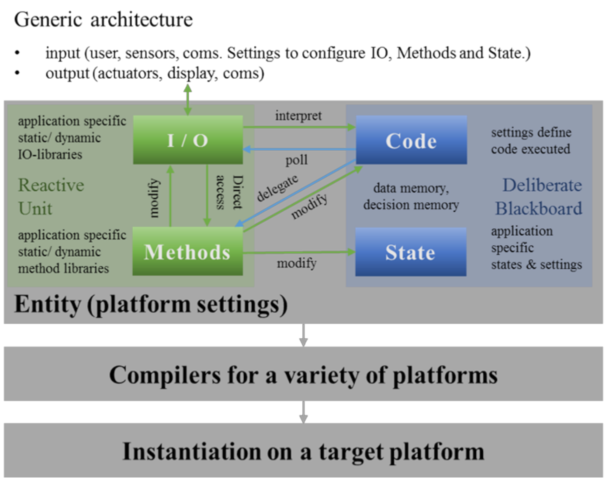

Fig.1. The generic architecture for a navigation entity (application). An entity holds the specific implementation options and objects that are selected by the user in the settings file.

```
STATElist
1:zupt:1
2:position:0:0:0
3:velocity:0:0:0
4:attitude:1:0:0:0
end
METHODlist
1:zupt-foot:1
2:kf:1:2:3:4
end
IOlist
1:dr-foot-imu-xsens:1:2:/input/imu-log1.mat
2:GNSS-raw-xiaomi-mi8:2:/input/gnss-mi8.mat
3:UWB-decawave-trek1000:2:/input/uwb-user1.mat
end
```

Listing 1. The structure of the entity settings file.

comes, the Zero UPdaTe (zupt-foot) method, in the METHODlist, has been set to have priority over the Kalman filter (kf). The methods are real-time capable implementations. The zupt-foot method is connected to the Kalman filter also via a hard-coded global variable "filter_initialisation_delay". This means that the zupt-foot filter is ahead in time compared to the Kalman filter. This global variable is thought to be transferred into the settings file as an option for the user to be able to define the real-time delay.

If the coder wants or needs to add a new IO, Method or State (or modify an existing), this can be done by adjusting the statelist.m, methodlist.m or the iolist.m files. Additionally, init_io.m needs to be modified to create the initialization tasks for each new state, method or IO.

The process_measurement.m file contains the execution control (Code) for the different configurations found in the settings file. This is also the file that defines the execution order for the methods and partly the fusion strategy of the configuration. For example, the coder can first define the context information call order (zero update method on foot-mounted PDR system or similar) and after that the prediction and update methods of the Kalman filter. This file can be modified likewise, according to the application requirements.

Fig. 2. depicts the entity main loop processing that is based on the arrival of the next measurement. The code is implemented for real-time, although the input used in the examples is here pre-recorded.

A. *Reactive Unit*

In the Fig. 1. the reactive unit contains the static libraries or dynamic instantiations of input and output devices and of all the possible methods available for the entity. The entity (application) may contain only reactive elements if the demand for the system intelligence is not high. For example, to actuate one motor by pressing one button.

- The IO block interprets the incoming data. Then depending on the access rights of the input device it can either use directly the internal Methods of the entity or send the information to the evaluation engine of the entity that is the Code (process_measurement.m). The IO block actuates or displays according to the information coming from either the Code or the Methods block.
- The IO block contains the implemented communication interface agreements between external entities. Standards for these improve the application development efficiency. This is crucial for the cooperative navigation applications. In our prototype test cases we used a simple time tagged file format.
- The Methods block is a collection of algorithms grouped by the application designers and developers. In case of navigation applications, these can be filters for context inference or for deriving the position and velocity estimations. Virtualisation and overloading are useful tools in implementing methods for differing configurations.

B. *Deliberate Blackboard*

The deliberate blackboard consists of State, and of the Code structure that handles the priority of Methods execution.

- The state is a data storage of past decisions, the history of the states and the current state. Decisions are e.g. collections of branching histories in crucial parts of the code (learning). Alike the IO and Methods block, State block does not have intelligence implemented inside it.
- We call an intelligent system here to be a system capable of autonomous decision-making according to the goals defined by the settings file. Thus the Code (process_measurement.m) is the block in where the intelligence is implemented. This is to use the Methods in a deliberate order to fulfil the defined goals. In this block, the learning Methods execution is likewise controlled according to the priority order value assigned in the settings file. The Code block together with the methods block define the level of intelligent behaviour that the entity is capable of reaching.

1. Read **settings** file
2. Initialise input and output
3. Process measurement

- Initialises the structure of the navigation entity. At first, how many states will it have, then which methods and finally which inputs and outputs - according to the **statelist.m**, **methodlist.m** and **iolist.m** files.
- **init_io.m** further executes setup tasks for the io, methods and states according to the given type definitions and input files.

- **process_measurement.m** file first goes through the inputs and delivers the next measurement that arrived.
- Then all associated methods are executed in the order defined by the settings file and according to the order in **process_measurement.m** file that change the states accordingly.

Fig.2. The main loop for the entity. The settings file is first interpreted to initialise the necessary objects. These are then initialised, in this prototype to read the recorded files. Finally, based on arrival, or the time tag of the measurement, the measurement is interpreted according to the implemented methods in an order defined in the settings and process_measurement.m file. The methods interact with the states of the entity. The methods have also their internal states.

C. *Introduction to the Example Use Cases*

To test our architecture, we implement a cooperative navigation application using it. The example consists of two cases with two users that use a different navigation entity configuration on their mobile devices. These cases show how even a minimal cooperation can improve the navigation capabilities for both users and how the development time is reduced.

The Fig. 3 shows the available options that are implemented in the general architecture. Fig. 4. shows the entity settings for the users in both cases 1 and 2.

In **case 1**, the user 1 has GNSS raw ranges recorded using the new mobile phone, Xiaomi Mi8, and the position estimates are derived using a modified code from Frank van Diggelen [19] and from the GoGPS project [20]. The indoor positions are derived using the least squares and the UWB raw ranges from 5 decawave anchors as in [21, 22]. A foot mounted IMU dead reckoning sensor (XSens) is the basis when fusing in a 15-error state Kalman Filter. A communications module (interfacing with external entities implemented in the IO block; in the example use cases these are shared files in Matlab) sends the absolute position of the user 1 and the UWB range between the two users to user 2. The user 2 has a foot mounted IMU (XSens) and an UWB between the users ranging sensors. The UWB ranging sensor estimates the distance in between the two users. The two IO sources for user 2 are also fused using a 15-error state Kalman filter.

In **case 2**, the user 1 has GNSS raw ranges, an UWB between users ranging sensor estimating the distance between the two users, and a foot mounted IMU. The user 2 provides his absolute position to the User 1 together with the range between the two users. The user 2 has UWB positions derived using least squares from the raw ranges of 5 anchors. This is fused in a Kalman filter with the IMU.

Fig. 5 shows the true tracks for the two users. The user 1 starts outdoors and enters indoors while user 2 walks indoors only.

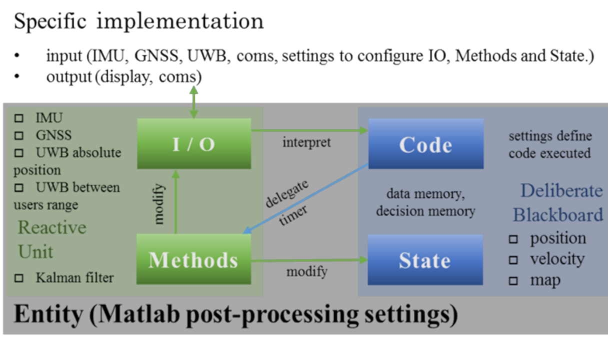

Fig.3. The implemented code options for the two entities (users) and for the cooperative application, coded for the IPIN 2019 test cases.

In both use cases the users have the states; position, velocity and map and the fusion engine is the 15 error state Kalman filter. IO for users in the use cases are:

1. **Use case 1**:
   User 1 settings: IMU, GNSS, UWB absolute position/ Kalman Filter/ Position, Velocity and Map
   User 2 settings: IMU, UWB between users range
2. **Use case 2**:
   User 1 settings: IMU, GNSS, UWB between users range
   User 2 settings: IMU, UWB absolute position

Fig.4. The settings (options can be seen in Fig.3) for the two users at the two use cases.

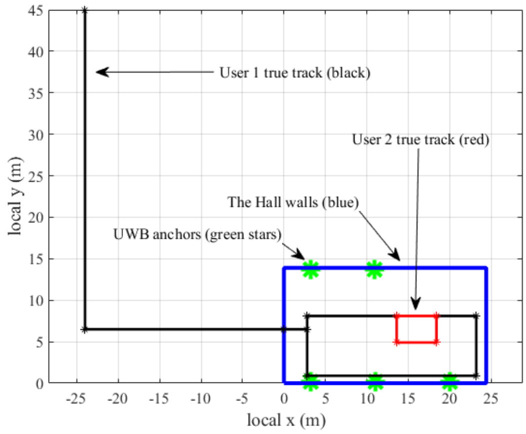

Fig.5. The true tracks for both cases 1 and 2. User one track is black, starts outdoors and walks three rounds around the black track indoors. User 2 walks around the red square asynchronously with the user 1. UWB anchor placement is shown by the green stars.

IV. Examining The Cooperative Test Cases That Implement The Novel Entity Architecture

In the test case 1, the user 1 starts outdoors at a known absolute position and with a known heading. The foot mounted IMU follows the user with a small drift. The bad quality GNSS measurements are fused in the Kalman filter with a low weight. When the user 1 enters the building, the quality of the GNSS measurements gets worse and are discarded. The accurate UWB position measurements replace the GNSS indoors and the user can be followed accurately.

Fig. 6 shows (next page) the resulting user 1 track in blue. Red stars are GNSS measurements and green stars are the UWB absolute position measurements.

The user 2 starts at a known position and with a known heading. At first, in the top of Fig. 6. the user 2 only uses a foot mounted IMU. The resulting track (in black) drifts. The greatest error is approximately 10m that is caused by the IMU drift during the experiment.

Bottom of the Fig. 6 depicts the situation where user 2 has additional information. This is the ranging information between the two users. The user 1 sends his position estimation together with the ranging result between the users.

In these use cases, the user with an access to absolute UWB position measurements is processed first. His location information together with the in-between users ranging information is then sent for the other user.

A. *Case 1 and the Fusion Strategy*

In case 1, the user 1 sends his position estimate and the ranging measurement between the two users. User 2 fuses these in his Kalman filter.

Our approach was to do a position and attitude update according to the sign of the residual and the length of the residual. The residual is taken as the difference between the distance that is between the current position estimates of the two users and the measured distance. The position update is the position that the measured distance indicates on the line formed by the current position estimates of the two users. This is shown in Fig. 7.

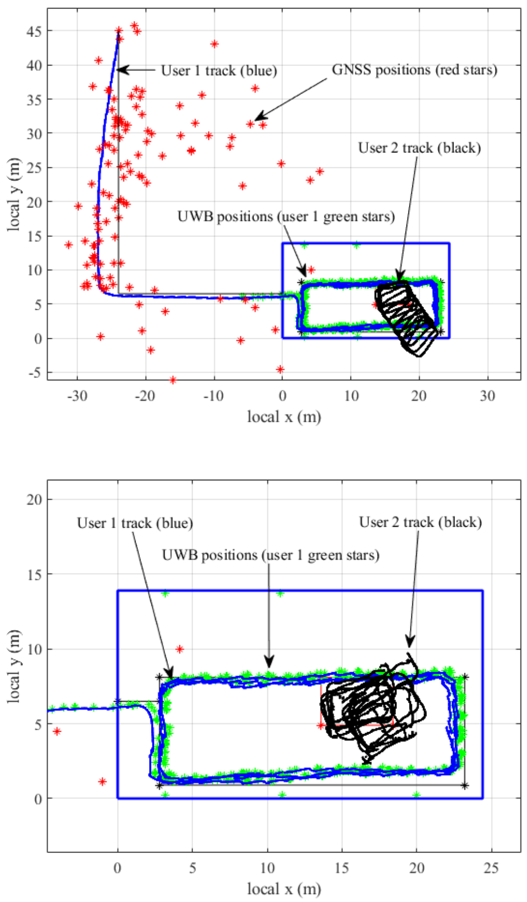

Fig.6. User 1 starts outdoors and walks indoors. User 1 shares his position and in-between users ranging data with User 2 in the bottom figure and the drift is limited, while the User 2 track drifts in the top figure without the in-between ranging information.

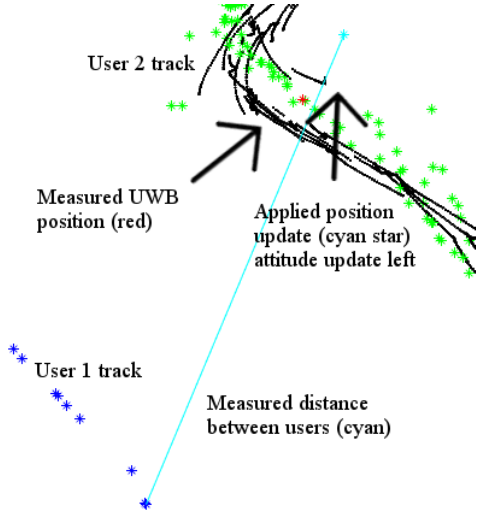

Fig.7. The position update applied for the user 2 Kalman filter is shown with a cyan star. The cyan line is the measured distance between the two users. In this case the measured distance is too long between the estimated user positions. As a result, a small constant attitude update to the left is applied.

Here, in Fig. 7, the user 1 positions are shown in blue stars. The UWB positions (not fused in User 2 Kalman filter in case 1) in green. The cyan line indicates the range measurement between the users. The cyan star shows the position update with a constant weight that is applied for the Kalman filter of the user 2. Also, a constant attitude update is applied depending on whether the range measurement is too short or too long. In this case a small constant turn towards left is applied.

B. *Advantage of Cooperation*

From Fig. 6 we see that the update method limits the drift of the user 2. The greatest error in this case is approximately 4m. The use case was completed with only the minimal set, of two users. It is expected that with more users the applied update will be better. This can be thought of as having anchors that move for the least squares position estimation if there are 3 other users or more in the vicinity.

C. *Disadvantage of Cooperation*

The Fig. 7 shows the trouble with this fusion approach. A levering effect can be seen depicted. This means that if the range measurement is slightly biased for a short period, the track is easily levered to a wrong direction. This is especially true for the used UWB positioning system that tends to have larger ranges when the line of sight condition between the transceivers is blocked by the user body. This is the major reason for the ranges that are too long at the situation of the Fig. 7. The levering, although compensates itself often after a while when the measured ranges incline the user towards the opposite way.

The levering is very susceptible to the trajectory forms that the users decide to take. In this particular case, the user 1 walked a large square and the user 2 the small square.

D. *Case 2 with reverse roles*

The case 1 describes a situation where the user 1 knows the absolute position all the time, both indoors and outdoors and can share it with the other user. The user two has only relative foot mounted information and loses knowledge of his position as time goes by. The ranging information from the user 1 helps the user 2 to maintain a more accurate estimate of his position.

The use case 2 is a similar scenario but where the user 1 knows his position outdoors but not indoors, while user 2 knows his position indoors and shares it and the between-the-users ranges with the user 1. Fig. 8. shows the results for case 2 two where user 1 first, on top, does not use the mutual between-users ranging information indoors. At the bottom of Fig.8, he does use it. The greatest error is approximately 8m for the unaided and 4m for the case where the ranging information between the users is applied.

The use case 2 shows that the ranging between the two users helps to reduce the drift (especially in the x direction in this case). The user 1 made three rounds around the large square. We see that the two first rounds were very aligned. But as the users were not synchronised in any way, in the third round, both of the users were very near to each other and levering occurred between the two in the last turn of user 1. This is shown in more detail in Fig. 9. Again, the effect of too long measured ranges is seen when looking at the cyan line and the trajectory. User 2's body obstructed the line of sight condition between the users.

It is not feasible to examine the cumulative error between the presented cases, since each test walk will have its unique levering effects and foot-mounted IMU drift characteristics.

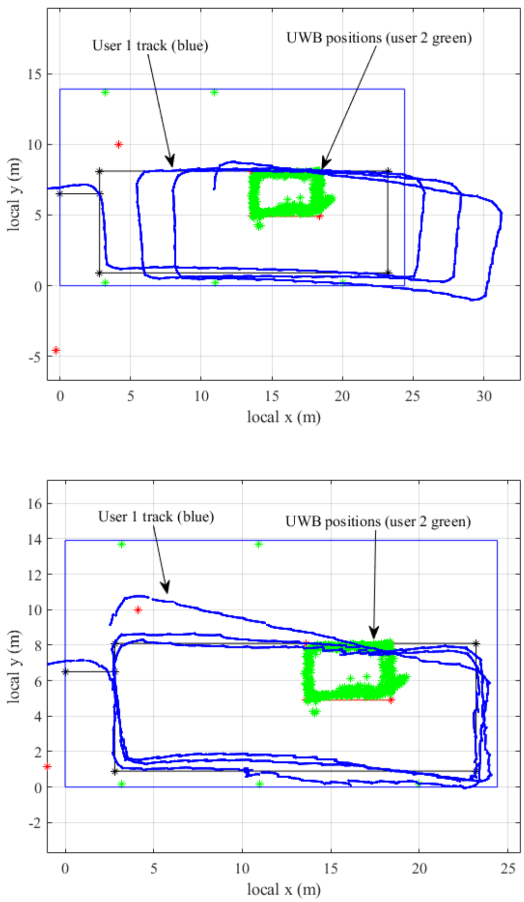

Fig. 8. On top is shown the trajectory of user 1 without the ranging data between the two users fused into the Kalman filter. Below the resulting track of user 1, when fused with the information of the ranges between the users.

E. *The Advantage of using the Novel Architecture on the Test Cases*

We coded the states, methods, IO and the code block implementations to cover all the necessary objects for our test application scenario. Now by only changing the configuration in the settings file the same code and architecture was used for both entities, the user 1 and the user 2 configurations. For a real-time application this can be thought as using the same base code but compiling it with a different setup file for the chosen platform on each user.

The advantages of this approach are the reusability of the code, the automatisation that the architecture enables for the software production of navigation applications and the scalability for multi-user, multi-state, multi-map, multi-sensor, multi-context-inference and especially for cooperative navigation applications.

The architecture enables easily modifiable code for distributed computing. This is enabled by using the settings files to define computing capable nodes with different computing settings and defining the IO communications between them. The architecture enables asynchronous communications with corresponding modules and is specially thought for real-time applications although the prototype version used pre-recorded files.

New methods can be added, and new sensor configurations can be fused with modifiable fusion algorithms that follow the general pattern for navigation

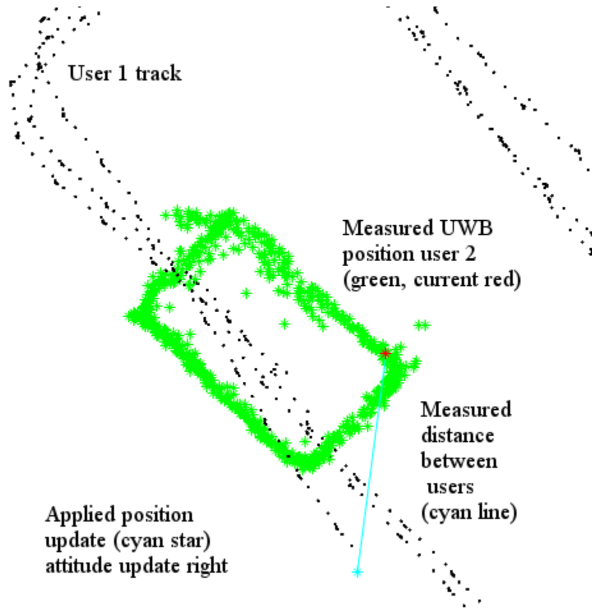

Fig.9. On the third round of the user 1, the user 2 was close behind. The levering effect resulted in the track to wander towards right. This result is the “hook” at the end of the user 1 trajectory in Fig. 8.

applications; to setup, to initialise and to respond to measurements according to the time of arrival of the measurement.

The design is generic and applicable to navigation applications. The data (states, decision and their history) information can be static or dynamic and is held in only one place and unnecessary copying of the information that consumes computation time, is avoided. The application specific code (the rules and execution priority) are separated from the well-known context inference and fusion methods. This enables the easy modification, when the purpose of the application changes.

V. On Security Issues

The inclusion of location-awareness abilities in mobile devices has had significant personal and business impacts. Localisation facilitates the establishment of geographical position and allows someone to check where a person is (without the person's consent). Despite its many benefits, it poses significant privacy challenges for the users of mobile devices.

Location-aware mobile devices can gather, store and reveal a large amount of sensitive data, giving rise to potential security and privacy concerns [23]. The main issue related to the privacy of localisation of mobile device users pertains to anonymisation. In particular, this concerns whether location data can identify a specific user, and if so, how sensitive that information is.

The information might reveal the types of activities that the user has engaged with or the locations of those activities, potentially divulging visits to sensitive destinations such as medical clinics, courts or political rallies. For instance, recordings of users’ activities and context-related information can reveal personal information such as health status and sexual orientation.

The sensitivity of location information can also subject it to misuse by companies and governments. Location data could expose an individual's religion, sexuality, political affiliations and health, all of which could be considered "sensitive personal data" under the Data Protection Act 1998 (DPA). Therefore, if not properly addressed, users' activities and habits can be profiled against their will, giving rise to, for instance, unsolicited advertising or harassments. For instance, the location of a multi-user mobile device stored at one particular time can facilitate strong inferences about the user of the device. Furthermore, if location information is recorded together with a series of authenticated transactions, then the information can be linked to a particular user, as opposed to the device alone. [24, 25, 26]

Thus, in order to take advantage of the localisation services, it will be of paramount importance, to take into account the privacy implications of the localisation of people and develop defences essential to safeguard such privacy. [27]

When thinking of our two-user experiment, we identify three relationships in the information sharing. First, both users communicate with the infrastructure. This involves having a permission from the infrastructure provider to access to the information of the infrastructure that is the anchors, their location information and other payload possibly available at them.

The second relationship is between the mobile device manufacturer and the user. The manufacturer of the device might be able to track the user for his purposes and interests. [28]

The third relationship is between the users themselves. Users can have an option whether if they want to share their positioning information for cooperative localization situations. The users with differing mobile devices by different manufacturers might have non-compatible hardware and software. For this purpose and for new mobile devices, the developed architecture would help in easing the task of interfacing designs between separate mobile device solution providers.

VI. Conclusions

A. On the Architecture

The novel architecture offers a flexible way for developing multi-user navigation applications. The same code structure can be used for a smart sensor on the user's foot and for the mobile device that he holds in his hands by only changing the settings file. This is a simple example of a distributed application that consists of two entities implemented using the same code but different setting files.

In the same way, the architecture increases the scalability for multiple mobile devices. Only the settings file defining the differing mobile configuration needs to be defined. The architecture doesn't take a stance on what individual technologies are used for the communication exchange in between the entities instantiated at different platforms. It is the application designer's freedom to choose in which way he focuses his application/entities and which hardware he uses for his purposes on different users.

The architecture leaves the technology choices open which are necessary to be known when estimating the scalability and computational load for different use scenarios. This work serves as a boost for the discussion of different players and how they could cooperate to make the development of novel location based services more flexible.

Personally, from the point of view of a researcher in a research centre, we see that the architecture helps in prototyping and sharing, for example the new context inference methods or novel sensor fusion methods. These are for example inferring, whether the user is indoors or outdoors using the sensor data or when fusing sensor data in a modified Kalman filter.

A Kalman filter can be exchanged by a particle filter if that is preferred, by changing the settings file or by a hybrid filter that switches between particle or Kalman filter depending on the environmental conditions [15]. Moreover, the map used to display the user location can be switched dynamically to another on the fly. The previous work mostly focuses on hard-coded alternatives while this work introduces a more dynamic and flexible approach to the navigation application testing research and development.

B. On the Test Cases

Our test cases dealt with only two users in an indoor-outdoor scenario. The two cases demonstrate the usability of the architecture. Although, only the sensor (IO) settings differ between the two user entities, a wider variety of example cases is easy to elaborate for either distributed computing scenarios or scenarios with many more users.

Entities using different map implementations is an example. If we consider an application for a fire fighter squad. For example, the entity implementation used in a control centre could use a wider range of display and analysis methods compared to the map implementation on an individual fire fighter client display. The fire fighter communications module would transfer his positioning information to the control centre. At the control centre, the states of the entity there would include this position information of the individual fire fighter and show it in a more suitable manner for the control centre display, differently to that at the individual fire fighter's display.

Acknowledgment

The authors acknowledge the projects - MICROCEBUS (TIN2018, RTI2018-095168-B-C55), TARSIUS (TIN2015-71564-C4-2-R) and REPNIN Plus (TEC2017-90808-REDT).

References

[1] J. R. Lowell, "Military applications of localization, tracking, and targeting," in IEEE Wireless Communications, vol. 18, no. 2, pp. 60-65, April 2011.

[2] Toshihiko NOMI, Hirosaka IKEDA, Proposal for technology architecture analysis, Synthesiology, 2015, Volume 8, Issue 1, Pages 1-14, Released March 17, 2015, Online.

[3] Corchado Rodriguez, Juan & Rodriguez, Sara & Llinas, James & Molina, Jose. (2009). International symposium on distributed computing and artificial intelligence 2008 (DCAI08). Selected papers based on the presentations at the symposium, Salamanca, Spain, October 22-24, 2008.

[4] Anne Weiss, Kathryn & C. Ong, Elwin & G. Leveson, Nancy. (2003). Reusable software architectures for aerospace systems. Aircraft Engineering and Aerospace Technology - AIRCRAFT ENG AEROSP TECHNOL. 75. 461-469.

[5] Tzafestas, Spyros. (2018). Mobile Robot Control and Navigation: A Global Overview. Journal of Intelligent & Robotic Systems. 91.

[6] Ferrer, B. R., Mohammed, W. M., Martinez Lastra, J. L., Villalonga, A., Beruvides, G., Castano, F., & Haber, R. E. (2018). Towards the Adoption of Cyber-Physical Systems of Systems Paradigm in Smart

Manufacturing Environments. 2018 IEEE 16th International Conference on Industrial Informatics (INDIN).

[7] Humberto Martinez-Barbera, David Herrero-Perez, Multilayer distributed intelligent control of an autonomous car, Transportation Research Part C: Emerging Technologies, Volume 39, 2014, Pages 94-112.

[8] Rossi, C, Heyi, MH, Scullino, F. A service oriented cloud ‐ based architecture for mobile geolocated emergency services. Concurrency Computat: Pract Exper. 2017.

[9] C. Giovannangeli and P. Gaussier, "Autonomous vision-based navigation: Goal-oriented action planning by transient states prediction, cognitive map building, and sensory-motor learning," 2008 IEEE/RSJ International Conference on Intelligent Robots and Systems, Nice, 2008, pp. 676-683.

[10] Mendonza, Marcio & Augusto Angelico, Bruno & Arruda, Lucia & Jr, Flavio. (2013). A Subsumption Architecture to Develop Dynamic Cognitive Network-Based Models With Autonomous Navigation Application. Journal of Control, Automation and Electrical Systems. 24.

[11] Beruvides, G., Castaño, F., Haber, R.E., Quiza, R. and Villalonga, A., 2017. Coping with Complexity When Predicting Surface Roughness in Milling Processes: Hybrid Incremental Model with Optimal Parametrization. Complexity, 2017.

[12] Iarovyi, S., Lastra, J. L. M., Haber, R., & del Toro, R. (2015). From artificial cognitive systems and open architectures to cognitive manufacturing systems. 2015 IEEE 13th International Conference on Industrial Informatics (INDIN).

[13] Haber, R.E., Juanes, C., Del Toro, R. and Beruvides, G., 2015. Artificial cognitive control with self-x capabilities: A case study of a micro-manufacturing process. Computers in Industry, 74, pp. 135-150.

[14] Leandro C. Fernandes, Jefferson R. Souza, Gustavo Pessin, Patrick Y. Shinzato, Daniel Sales, Caio Mendes, Marcos Prado, Rafael Klaser, Andre Chaves Magalhaes, Alberto Hata, Daniel Pigatto, Kalinka Castelo Branco, Valdir Grassi, Fernando S. Osorio, Denis F. Wolf, CaRINA Intelligent Robotic Car: Architectural design and applications, Journal of Systems Architecture, Volume 60, Issue 4, 2014, Pages 372-392.

[15] Peltola, Pekka. "Towards seamless pedestrian navigation." PhD diss., University of Nottingham, 2018.

[16] P. Peltola, J. Xiao, C. Hill, T. Moore, F. Seco and A. R. Jiménez, "A tripartite filter design for seamless pedestrian navigation using recursive 2-means clustering and Tukey update," 2018 IEEE/ION Position, Location and Navigation Symposium (PLANS), Monterey, CA, 2018, pp. 739-748.

[17] P. Peltola, F. Seco, A. R. Jiménez, T. Moore and A. Catherall, "Corrective Track Form Matching for Real-Time Pedestrian Navigation," 2018 International Conference on Indoor Positioning and Indoor Navigation (IPIN), Nantes, 2018, pp. 1-7.

[18] B. Wagstaff and J. Kelly, "LSTM-Based Zero-Velocity Detection for Robust Inertial Navigation," 2018 International Conference on Indoor Positioning and Indoor Navigation (IPIN), Nantes, 2018, pp. 1-8.

[19] https://insidegnss.com/gnss-analysis-tools-from-google/, referred 24th May 2019.

[20] https://github.com/goGPS-Project, referred 24th May 2019.

[21] A. R. Jiménez and F. Seco, Comparing Ubisense, Bespoon and Decawave UWB location systems: indoor performance analysis, IEEE Transactions on Instrumentation and Measurement , vol. 66, no. 8, pp. 2106-2117, August 2017.

[22] A. R. Jiménez and F. Seco, Finding objects using UWB or BLE localization technology: a museum-like use case, 2017 Indoor Positioning and Indoor Navigation Conference (IPIN), pp. --, Sapporo, Japan, September 18-21 (2017).

[23] Montasari, Reza, Richard Hill, Victoria Carpenter and Farshad Montaseri. "Digital Forensic Investigation of Social Media, Acquisition and Analysis of Digital Evidence," International Journal of Strategic Engineering (IJoSE) 2 (2019): 1.

[24] Montasari, R., Peltola, P. and Evans, D., 2015, September. Integrated computer forensics investigation process model (ICFIPM) for computer crime investigations. In International Conference on Global Security, Safety, and Sustainability (pp. 83-95). Springer, Cham.

[25] Montasari, R., 2016. A comprehensive digital forensic investigation process model. International Journal of Electronic Security and Digital Forensics, 8(4), pp.285-302.

[26] Montasari, R., 2016. Review and assessment of the existing digital forensic investigation process models. International Journal of Computer Applications, 147(7), pp.41-49.

[27] Montasari, R. and Hill, R., 2019, January. Next-Generation Digital Forensics: Challenges and Future Paradigms. In 2019 IEEE 12th International Conference on Global Security, Safety and Sustainability (ICGS3) (pp. 205-212). IEEE.

[28] Montasari, Reza, Amin Hosseinian-Far, Richard Hill, Farshad Montaseri, Mak Sharma and Shahid Shabbir. "Are Timing-Based Side-Channel Attacks Feasible in Shared, Modern Computing Hardware?," International Journal of Organizational and Collective Intelligence (IJOCI) 8 (2018).