GEMMA: A GENERIC, EXTENSIBLE AND MODULAR MULTI-SENSOR NAVIGATION ANALYSIS SYSTEM

J. A. Navarro[a,*], M. E. Parés[a,*], I. Colomina[b]

[a] CTTC, Av. Carl Friedrich Gauss 7, 08860 Castelldefels, Spain - (jose.navarro, eulalia.pares)@cttc.es
[b] GeoNumerics S.L., Av. Carl Friedrich Gauss 11, 08860 Castelldefels, Spain - ismael.colomina@geonumerics.com

ICWG III/I

KEY WORDS: Trajectory determination systems, Research toolset, Navigation, Sensor orientation

ABSTRACT:

This paper presents the concept of an architecture for a system that helps researchers in the field of Geomatics to speed up their daily research on kinematic geodesy, navigation and positioning fields. The presented ideas correspond to an extensible and modular software system aimed at the development of new navigation and positioning algorithms as well as at the evaluation of the performance of sensors. The concept, already implemented in the CTTC's system GEMMA is generic and extensible. This means that it is possible to incorporate new navigation algorithms or sensors at no maintenance cost. Only the effort related to the development tasks required to either create such algorithms or model sensors needs to be taken into account. As a consequence, change poses a much smaller problem for CTTC's research activities is this specific area. This system includes several standalone tools that may be combined in different ways to accomplish various goals; that is, it may be used to perform a variety of tasks, as, for instance, (1) define positioning and navigation scenarios, (2) simulate different kinds of sensors, (3) validate new navigation algorithms or (4) evaluate the quality of an estimated navigation solution.

1. INTRODUCTION

Nowadays, the research community is constantly developing new positioning and navigation algorithms. These algorithms aim at the determination of precise, accurate and robust trajectories. By trajectory we mean a path of an stochastic process; that is, it is one of the many time series realizations of an stochastic process. In the context of this paper, a trajectory is a time series of positions, velocities and attitudes of a moving object plus the calibration parameters of the instruments used to determine these as well as the estimated covariances of all the aforementioned values.

The navigation research community is currently dealing with two challenges: new technologies and their use in new applications. New sensors, able to provide data suitable for geoapplications, appear constantly in the market. As stated in (Groves et al., 2014a, Groves et al., 2014b), technology has to deal with new sensors—like plenoptic or photon-mixing cameras—, new performances—like the inertial sensors found in smartphones—and new environments—like indoor or urban canyons. Furthermore, there are still many issues to solve concerning the achievement of target precision, accuracy and reliability in the realm of positioning and navigation. The following are some examples: for mobile mapping purposes, GNSS accuracy is almost unachievable due to strong multipath in narrow urban environments (Xie and Petovello, 2015); completely autonomous or unattended driven systems (in aerial, marine or submarine environments) are still a wish for the surveying community due to the lack of reliability of navigation solutions (Velaga et al., 2012); kinematic airborne gravimetry is not able yet to reach the maximum precision because of the high level of noise present in INS/GNSS systems (Skaloud et al., 2015).

The examples above define an scenario where continuous and intense research in a steadily changing technological environment is taking place. This constant, uninterrupted change and evolution process constitutes a challenge (for instance, from the software engineering standpoint) for the research activities in this area.

Nowadays, researchers have at their hand a wide range of helpful tools (either proprietary, free or open source). However, these have been conceived to perform rather specific tasks and not as components of a larger (tool) ecosystem. It is possible, for instance, to find MATLAB® code to generate synthetic trajectories (Ahmadzadeh, 2015), signal (IFEN, 2015) and measurement (MathWorks, 2015) generators or reliable and widely known trajectory determination and analysis systems like the Applanix or the Waypoint ones. Unfortunately, and in spite of (and because of) this availability of tools, researchers will have to cope with problems as data format incompatibilities or, in the worst case, develop a full trajectory determination system when new algorithms need to be implemented.

This paper presents the concept of an architecture for a system whose target is to provide a reliable framework where research related to positioning, navigation and sensor modelling may take place. The goal of such system is to become the basic toolset for researchers in these areas, avoiding the need to start anew each time a new project begins or, thanks to some of the tools included in this architecture, not having to organize and execute costly data acquisition campaigns when (new or not) sensors need to be evaluated. Additionally, an implementation of the aforementioned concept and architecture, is presented here: GEMMA (*G*eneric *E*xtensible and *M*odular *M*ulti-sensor navigation *A*nalysis system).

GEMMA is a portable, extensible and modular software system aimed at the development of new navigation and positioning algorithms as well as at the evaluation of the performance of sensors. It has been conceived as a research toolset useful in laboratory environments (as opposed to workshop or factory ones) (Navarro, 1999).

It is, however, worthy to say, that although GEMMA *is not a commercial product*, the team in charge of its design and implementa-

*Corresponding authors.

tion decided to incorporate some characteristics present in products that are, indeed, commercial ones. For instance, GEMMA intends to be a *complete* toolset or suite, providing an *integrated environment* where research tasks and work flows are facilitated; a data abstraction and their corresponding interface has been devised as well, *to avoid impedances* between the tools integrating this suite, since the need of file converters is noticeably reduced; a batch interface exists, making possible the *automation of repetitive tasks*, while a—still very immature— graphic interface has been included to ease the interaction with end-users (researchers). In short, the essential traits (as scientific rigour) that are a must in a software toolset targeted at research have been complemented with other characteristics usually present in commercial systems.

Section 2. describes the architecture (and components) of GEMMA from the conceptual and implementation standpoints (showing how the GEMMA system materializes the concept). Section 3. details how this system may be used in different scenarios (use-cases or work flows).

2. CONCEPT AND ARCHITECTURE

The architecture (and components) of a framework willing to provide with a useful set of tools and procedures to facilitate the research tasks related to a specific set of disciplines may obviously vary depending on the actual experience of the people involved first in its inception and, later on, in its design and implementation stages. This paper presents, therefore, the view of its authors, view that has been heavily influenced by their participation in a series of projects where the framework discussed here played a key role. Nonetheless, after several years of being put to the test, the implementation of the architecture discussed below has been able to cope with all the challenges faced up to the moment.

The components included in a positioning and navigation research framework should be able to offer, at least, the following set of features:

- Generation (densification) of realistic trajectories for any kind of platform: from spacecrafts, to aerial, terrestrial or marine vehicles, or even alive organisms;
- simulation of signals (measurements) for several types of sensors, as, for instance, IMUs or GNSS receivers,
- sequencing of sensor measurements,
- trajectory estimation and
- (trajectory) quality assessment.

Each of the features (and related tools) in the list above is important by itself. For instance, the ability to generate and densify trajectories avoids having to obtain such trajectories by other methods; when signal (measurement) generators are available, there is no need to organize data collection campaigns. In short, these tools help to save time and reduce costs, thus facilitating the work of researchers. However, it is the combination of different subsets of these features (tools) what reveals the versatility of the concept, and how it responds to different research use cases (see section 3.).

GEMMA implements these features, providing a tool for each of them. In particular, it includes four signal generators, for IMUs, magnetometers, odometers and GNSS receivers.

Figure 1 depicts all the features just discussed (implemented as a set of tools in the GEMMA system). Additionally, *all* the possible data flows between these are shown. Depending on the actual scenario being tackled (see section 3. for some examples), only a subset of these tools and data flows will be used. For instance, when not working in a real-time environment, the trajectory estimation tool will not be connected to an acquisition system but will use logged (pre-recorded) data (measurements), information coming from one or more of the available signal generators, or both.

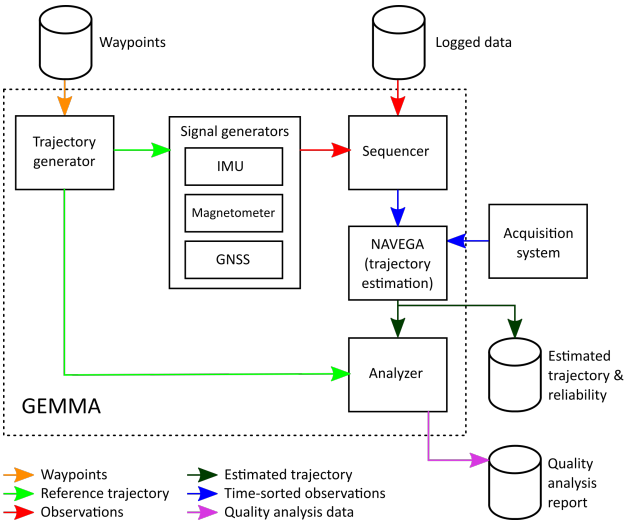

Figure 1: GEMMA components and work flow

The following sections describe briefly the different components of the architecture and show how GEMMA implemented these.

2.1 Trajectory Generation

The ultimate goal of the trajectory generator tool is to create *reference trajectories* to compare with when these do not exist or it is too expensive to obtain them—for instance, flying a real plane.

Strictly speaking, this tool is not about fully trajectory *generation* but about trajectory *densification*. Given a set of way points (time-tagged positions, and eventually attitudes) an infinite number of trajectories can be derived. The tool should be able to produce a dense *realistic* trajectory including not only the previous way points but a much more extensive list of them as well as related velocities and orientations. Figure 2 depicts a densified trajectory using a few way points.

Figure 2: Example of a densified trajectory

In order these trajectories to be as realistic as possible they should be continuous and infinitely derivable. Moreover, since not all the vehicles (platforms) follow the same dynamic equations, the tool should be designed in such a way that the densification of

the way points and the related orientation definition adhere to the specific dynamics of the selected platform. This translates into an extensible software component where the densification process is independent of the selected vehicle dynamics.

For versatility (and flexibility) reasons, the generator must be able to accept way points generated by different sources, as for instance, KML files or trajectories output by (realistic) flight simulators—file format converters may be needed to do so. The parameters controlling the behaviour of the tool (as the output frequency) must be also configurable by the user.

GEMMA includes a trajectory generation tool complying with the functional requirements discussed above. Specifically, it is able to achieve the infinitely derivable position and orientation by convolving with a C^∞ function and later on interpolating the resulting trajectory at the rate selected by the user; the realistic orientation for different platforms is obtained from the dynamics formulas of those vehicles (Rajamani, 2011, Roskam, 1995).

2.2 Signal Generators

A (software) signal[1] generator simulates the observations that would have been delivered by a sensor if it would have actually been used in a real campaign.

Signal generators take as input a reference trajectory—however obtained—and use the *models* defining the behaviour of the sensor these mimic to deliver the simulated observations. Note that these models must include the *errors* affecting the sensors, so output data mirrors this *defective* behaviour.

Errors may correspond to various stochastic processes (as random walks, white noise or Gauss-Markov ones). Additionally, errors may be originated by different sources, as the instrument itself, the environment (temperature, magnetic fields, humidity among many others) or the platform the instrument is mounted on (vehicles, human or animal carriers). These kinds and sources of errors must be taken into account by a signal generator so reality is properly mirrored.

Signal generators may be either software or hardware components and the presented architecture makes no distinctions between these. Software components, apparently, are less prone to obsolescence since it is possible to include new features just by maintaining the code.

There are different situations where the use of such generators prove to be useful. These are some, among others:

- To check how the quality of an estimated trajectory is improved by incorporating a sensor that is not available—but correctly modelled.
- To select the appropriate quality of a sensor according to the requisites that the estimated navigation solution must meet. For instance, a tactical grade GNSS receiver may not be necessary in all situations.

For instance, Figure 3 shows the behaviour of two different IMUs travelling along the same reference trajectory. Purple data corresponds to a low-cost IMU unit while the green bar represents how a navigation-grade IMU behaves. The different responses of these two sensors are easily identifiable.

[1] The words *signal*, *measurement* and *observation* are incorrectly used as synonyms here for the sake of simplicity. For instance, in some situations, additional processing may be needed to obtain a measurement out of a signal, but such distinction is not made in this paper.

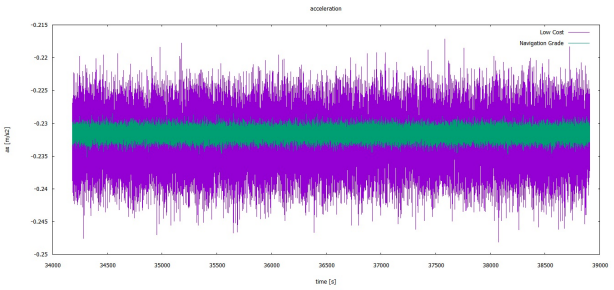

Figure 3: Example of output of an IMU signal generation tool

At the moment of writing this paper, GEMMA includes four types of signal generators fully compliant with the functional description above, namely odometers, magnetometers, GNSS receivers and IMUs (Parés et al., 2015). Others may be included in the future when needed. A hardware GNSS simulator, the IFEN NavX-NCS PROFESSIONAL GNSS simulator (IFEN, 2015), is also part of the system.

It is important to highlight that there exist nowadays many signal generators in the market for a variety of sensors—see (Ebinuma, 2015) and (Nievinski and Larson, 2014) for two examples of available GNSS simulators or (Yang et al., 2007) and (Yan et al., 2015) for IMU ones. In spite of this availability, the team in charge of the development of GEMMA decided to implement its own signal generators.

The extra cost of developing tools that already exist in the market is justified by several reasons that were deemed strategic. First, an in-house development allows for a higher degree of freedom when implementing a sensor simulator; additionally, it is possible to incorporate any kind of error models that other tools might have not considered; moreover, the data formats used by in-house developments may be GEMMA's own, reducing the data conversion steps needed to make data suitable for other tools in the toolset; last, but not least, it is possible to design these generators in such a way that these may be integrated in batch production environments (to avoid the unnecessary repetition of routine tasks) and, at the same time, include a user-friendly graphic interface facilitating the interaction between the researcher and the tool when required.

2.3 Sequencer

The goal of the sequencer is to sort a series of heterogeneous observations in ascending time order. In this context, *heterogeneous* stands for *originating in different kinds of sensors*. The need to sort input data is motivated by the fact that the estimation of the successive values of the states integrating a trajectory is based on the use of observations that are *close* in time.

A sequencer takes as input a series of files or TCP/IP (*T*ransmission *C*ontrol *P*rotocol / *I*nternet *P*rotocol) socket connections containing or transmitting time-tagged observations measured by different sensors and outputs a single stream of data including these same observations conveniently sorted. This output stream may be either written to a file (for local, batch processing) or sent through a TCP/IP socket connection (remote, batch processing).

When processing data received through TCP/IP socket connections, the signal generator must be able to discard data whose time tag is prior to the current time being processed. A time tolerance parameter to decide if such data is accepted in spite of being too old is a very convenient feature to have; normally, reading data through network connections imply real-time processing

and, in this situation, some delays or shifts related to time stamps are to be expected. Therefore, such a tolerance parameter may be used to alleviate these discordances. Obviously, the trajectory estimation tool (see section 2.4) must be able to cope with such non-perfectly time-sorted stream of data for this feature to be of any use at all.

GEMMA includes a sequencer tool implementing almost all the features described above. The only limitation is that it is only able to accept input data stored in files (network input is not yet accepted), so the discussion related to time tolerances do not apply here. The ability of the sequencer to send its output through a network connection opens the way to *distributed research*, which has been put into practice in several projects, as ATENEA (Fernández et al., 2011): observation data is obtained in one place, sequenced there, and sent via sockets to a second place, where it is processed.

2.4 Trajectory Estimation

The goal of this tool is to estimate trajectories (navigation solution) out of a series of time-sorted, heterogeneous sensor observations.

New sensors—the source of observations—appear constantly in the market and old ones are improved or modified. To cope with constant change and innovation, a trajectory estimator should be designed to be a generic and extensible tool, so the heavy toll that should be paid in terms of software maintenance due to such evolution may be avoided. To do so, the following principles should be the cornerstones on which a tool like this should rely:

- *Separation of estimation and modelling*. The tool must separate the estimation—"number crunching"—engine from the data being used and its relations. This allows a quick extension of the software when new sensors appear. When the tool starts, it must load the components related to the intervening sensors as well as some mission-related states. These components, whatever they are, must not affect how the computational kernel (the estimation engine) behaves. In this way, the inclusion of new sensors imply the development of only relatively small fragments of code that are materialized as the aforementioned loadable components.
- *Rigorous data modelling*. Data must be modelled to represent the essential traits defining the four entities identified as key players in the positioning and navigation realm: (input) measurements, (input) auxiliary instrument constant values, (output) states and (equation) models relating all those elements. The abstraction must therefore be generic and able to manage, at the same time, the variety of available sensors. Relying in a rigorous data model implies that the implementation or maintenance of sensor models may benefit of code reuse and object-orientation techniques—and its encapsulation, inheritance and polymorphism mechanisms. This leads to a substantial simplification of the system and much shorter times to implement new sensors, measurement and state models in the software.
- *Generic and adaptable interface*. The input / output interface of the trajectory estimator must mirror the data abstraction process just stated, so the inclusion of new sensors implies no changes in the way data are handled. To enable either batch or real-time processing, at least two data interfaces should be defined: file and network—this last one based on TCP/IP sockets.
- *Computational strategy object*. Sensors (equipment) are not the only factor affecting the quality of a navigation solution. Environmental conditions must also be taken into account (Groves et al., 2014a, Groves et al., 2014b); the kind of vehicle used (airplane, helicopter, terrestrial vehicle, to mention some) imposes a very different set of constraints on what kind of trajectories are possible. For instance, sudden changes in the direction of movement are possible when equipment is carried by people, while airplanes are not able to do so. A trajectory determination system should be aware of these limitations that depend on the context, either because it has been informed by the user or being able to determine these by itself when no information is available. This is the task of the computational strategy object. This component must be able to determine—depending on the context it is working in—whether there is enough information to compute a solution and what to do when the answer is negative—as for instance, deliver a partial solution or just warning about its inability to proceed. The computational strategy object must be independent from the estimation ("number crunching") engine; this approach allows for the combination of different sensor models and environments (contexts) at no (source code) maintenance cost.

The channels used for input (observations) and output (trajectories) data must be selectable freely; that is, any combination of file or TCP/IP sockets data channels for input or output must be possible. In this way, it is possible to use the trajectory estimation tool in a wide variety of scenarios. For instance, it may work as a real time (navigation) *data logger* using sockets on input and files on output; if both input and output channels are TCP/IP sockets, then the tool works as a real-time navigation system. Using files in both channels transforms the trajectory estimator into a batch tool.

NAVEGA (Parés and Colomina, 2015) is the trajectory estimation component included in GEMMA. It has been designed according to the principles described above. In short, the most important features of NAVEGA are:

- Robust trajectory estimation,
- using a geodetic approach, that is able to
- perform multi-sensor navigation,
- detecting and isolating faults,
- running either in real-time or batch environments, and that
- is available for the most common operating systems.

Currently, the number crunching engine of NAVEGA includes a family of Kalman filter algorithms and provides with Gaussian states, this is, the output are expected values and covariance matrices. Furthermore, it also provides the user with expected residuals and standard deviation values, suitable for a posteriori trajectory analysis (see section 2.5).

The extensibility of NAVEGA has been proven processing data provided by several types of sensor like IMUs (Molina et al., 2012b), GNSS (Silva et al., 2011), redundant IMUs (Molina et al., 2012b), LIDAR (Montaño et al., 2015), camera images (Angelats et al., 2014) and odometers (Angelats et al., 2011). NAVEGA is validated using indirect methods—because not all the sensor models implemented are available in commercial systems. When NAVEGA is used as a position provider (remote sensing platforms, as photogrammetric systems for instance), its output is compared with topographic measurements, which, up to now, have served to show the success of this software component; that is, the actual and predicted state errors match, or, in short, the expected quality is achieved. NAVEGA has also been successfully used for processing data collected in a wide range of environments: airplanes, helicopters (Molina et al., 2012b) and terrestrial vehicles, either outdoors and indoors (Angelats et al., 2011).

Regarding robustness issues, NAVEGA's implemented analysis tools are able to deal with odometers and camera outliers. When working with GNSS modelling, the system is able to detect outliers when these affect just one satellite. At this moment, more than one outlier cannot be properly detected.

Integrability is also a fact. It has been used in the framework of several projects as a location provider of different services, like mobile mapping systems (Fernández et al., 2011) or a GNSS receiver with tight coupling capabilities (Silva et al., 2006).

The use of NAVEGA in the aforementioned examples served to verify that this component is not only available for a wide range of platforms but also that its performance suits the needs of these. For instance, NAVEGA is able to process INS/GNSS solutions at 5000 Hz in real time when running on an Odroid-XU3 (HardKernel, 2015) with a negligible latency.

Finally, but no less important, the dual interface of the system (file and GUI) makes NAVEGA suitable for both Kalman filter experts and inexperienced users as well.

2.5 Analyser

The last tool in the system is the trajectory analyser. As its name indicates, the aim of this tool is to perform several quality analysis over an estimated trajectory. The main features it should include are:

- Given a reference trajectory, which is considered as the absolute truth, the system should be able to "translate" one trajectory to the other one and to compare them. By "translation" we mean to apply the user's requested offset and boresight matrix. The analyser should be able to compute statistics (like, mean, standard deviation and RMSE) fully describing the difference between these, which is the cornerstone to assess the quality of the estimated trajectory.
- Given a reference trajectory, the system should be able not only to compare estimated and reference trajectory as previously stated but also to check its coherence with the predicted error, and to determine if those last values are under- or overestimated.

Given the residuals or the innovations of an estimated trajectory the system should be able to determine the stochastic process beneath them, and to check if it is coherent with the expected one (typically white noise).

GEMMA's trajectory analyser implements all the aforementioned features. Additionally, it is able to produce a series of plots depicting some of the statistics discussed above to help to better understand them.

Figure 4 is an example of one of these plots: there, the actual error of an estimated trajectory (comparison between estimated and reference trajectories—in red) and the standard deviation of the estimator predicted error (in green) are shown. It may be observed how the estimated and actual errors do not match; thus, the trajectory determination system is being optimistic.

3. USE CASES

A toolset such the one presented in this paper may be used in a variety of research use cases, just combining, in different ways, the components that integrate it. The most usual use cases are:

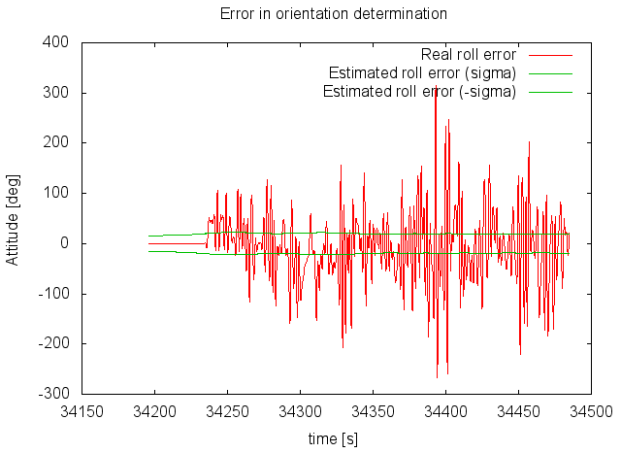

Figure 4: Quality plot from the analyser tool

- Technology selection or validation.
- Algorithm verification and validation.
- Navigation and positioning in real-life environments.

The following sections describe these use cases and the role that play the different components of the toolset in each of these situations. All the presented use cases have been successfully addressed with GEMMA.

3.1 Use Case: Technology Selection or Validation

This use case takes into account two possible situations:

- It is necessary to select a sensor among a family of these whose quality is good enough—but not necessarily better—to estimate output trajectories that must match some precision and accuracy requirements. For instance, not all applications need a navigation-grade IMU, but it may be costly enough to determine what is the actual quality level that will guarantee the required output.
- Assessment of the quality of a single sensor, that is, determining whether it is appropriate for the requisites affecting the estimated trajectory.

The advantage of using GEMMA in this case is that it is possible to *simulate* the signals produced by these sensors and then estimate and evaluate, for each candidate, the respective output trajectories. Needless to say, proper modelling of the behaviour of the intervening sensors is required.

The components of GEMMA involved in this use case are shown in Figure 5. There, those GEMMA components that do not intervene in the use case are shown in pale grey.

The following would be the typical work flow to follow:

- First of all, a reference trajectory is needed to compare it with the output estimated ones. It may be either synthetically generated by means of the trajectory generator component or taken from any field campaign.
- The signal generator(s) corresponding to the sensor(s) to evaluate would then be used to simulate the signals (measurements, observations) that would have been produced if the sensor would have actually travelled along the reference trajectory.

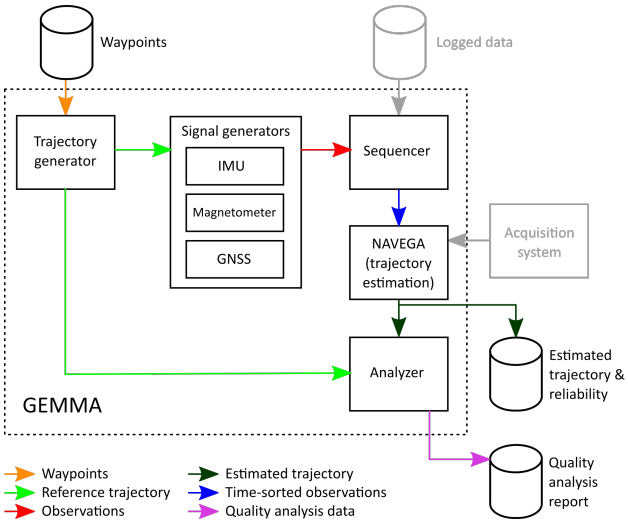

Figure 5: The technology selection use case.

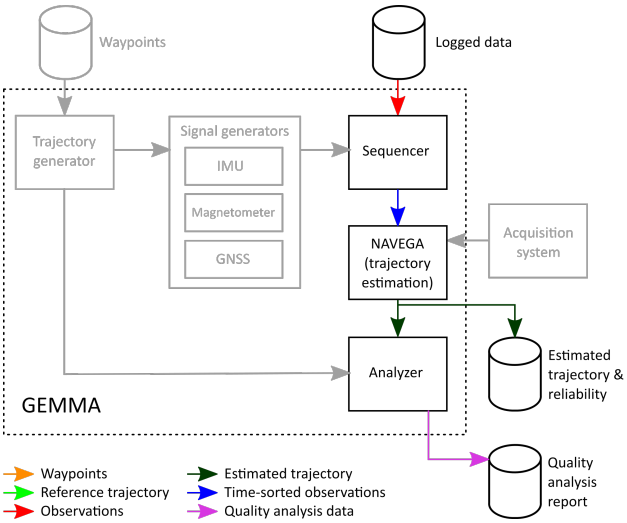

Figure 6: Algorithm verification and validation use case.

- Then the sequencer would time-sort all the observations produced by the signal generators.
- The trajectory estimator component (NAVEGA) would be used to estimate the output trajectory, and, finally,
- the quality of the aforementioned trajectory would be assessed by the analyser component.

When comparing several sensor candidates to select the most appropriate one, the process above should be repeated for every sensor being evaluated. If the goal is just to determine whether a sensor is good enough to fulfil the stated goal (trajectory quality) then a single run should suffice.

A real project where this use case was put to the test was GINSEC (Navarro et al., 2015).

3.2 Use Case: Algorithm Verification and Validation

It has been stated several times in this paper that GEMMA is generic and extensible. This means that new sensors may be included into the system—or, those that have been modified, easily adapted—at almost no cost.

Including a new sensor means writing the mathematical equations modelling its behaviour and generating a loadable library including the new model. There is no need to change already existing software to make the new model available to GEMMA; that is, when adding new sensors there is no need to maintain, change or adapt the existing code base. The new model will be loaded dynamically upon request—that is, when data related to such model is processed.

As it happens to any kind of algorithm, it is necessary to verify and validate new models in order to guarantee their correctness (that is, the new code contains no bugs) and performance (the new sensor is correctly modelled). This is the algorithm verification and validation use case.

Figure 6 shows the components in the GEMMA system playing a role in this situation (note that, in this figure, those components not intervening in the process are shown in pale grey).

The sequence of steps to take when verifying and validating a new algorithm is defined below. A reliable reference trajectory, however, must be generated beforehand. To create such trajectory, a data collection campaign involving not only a set of *trusted sensors*—that is, already correctly modelled—but also the new one must take place.

The reference trajectory must be estimated using only trusted sensors, *so it is a reliable one to compare with* (note that Figure 7 describes this procedure). Data coming from the new sensor *is just logged*, and will be later used to verify and validate the new model being developed.

Once the reference trajectory and the new sensor data are available, the procedure to follow is:

1. Using the sequencer tool, data coming from *all* the sensors involved in the data collection campaign must be time-sorted.
2. The mathematical equations modelling the new sensor must be written. This, essentially, means filling the gaps in a template provided in GEMMA's developer kit; such template adheres to the API (*A*pplication *P*rogramming *I*nterface) defined by the system. In this way, developers need to have no prior knowledge about the internals of GEMMA; their knowledge may be just limited to sensor modelling
3. A dynamically loadable library is then created and included in the GEMMA system.
4. NAVEGA is used to estimate a new trajectory.
5. The trajectory analyser is then used to compare the reference trajectory with the newly estimated one, thus assessing its quality.

Steps 2 to 5 may have to be repeated several times until the quality of the mathematical model is satisfactory.

Several projects that took place in the past were clear examples of this use case: GAL (Skaloud et al., 2015), ENCORE (Colomina et al., 2012) and ATENEA (Fernández et al., 2011).

3.3 Use Case: Navigation and Positioning in Real-life Environments

This is the use case where less components of GEMMA are involved: NAVEGA is used as a standalone tool (see Figure 7). In this situation, NAVEGA is used as a real-time trajectory estimator.

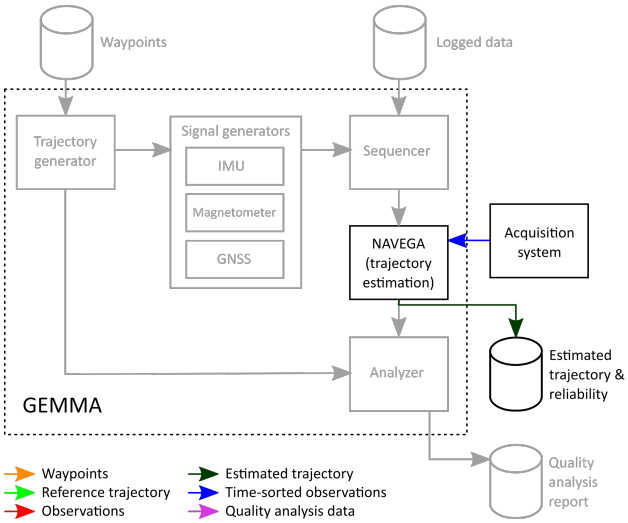

Figure 7: Real-time environments use case.

Loaded with the appropriate set of models (those characterizing the sensors integrating an external acquisition system) NAVEGA might be used as a real-time server providing successive position and attitude values to some other subsystem(s). For instance, an autopilot system might decide to delegate the task of positioning the aircraft to NAVEGA, feeding this component with data coming from the different sensors on board, instead of performing this process itself. The GINSEC project (Navarro et al., 2015) is an example of this approach, where an implementation of a positioning server inspired by NAVEGA was installed on a low-end board. Of course, other subsystems in need of georeferencing in real-time may benefit from such positioning server concept.

A variation of the example above would compute position and attitude in real-time but log these data to permanent storage instead. This situation is typical in post-processing environments, when higher accuracy and precision are required. Processing the data collected with NAVEGA by means of techniques as PPP (*P*oint *P*recise *P*ositioning) would improve the quality of the estimated trajectory to the level required by professional applications (as network bundle adjustment, for instance).

The project CLOSE-SEARCH (Molina et al., 2012a) is a typical example of this use case.

4. CONCLUSIONS

GEMMA is not only a concept but also a real, seasoned, research-oriented system, that has made possible a significant number of research projects in the past, covering a wide spectrum of situations within the realm of kinematic geodesy, positioning, navigation, and related disciplines and applications. This has been possible thanks to the principles supporting its architecture, mainly genericity and extensibilty, as well as a powerful set of data and interface abstractions. These principles were crucial to cope with change and innovation from the very beginning, and still are.

GEMMA is composed of a variety of tools whose combination opens the path to its exploitation (as a research tool) in the most usual scenarios where navigation is involved, as the evaluation / modelling of new sensors or real-time navigation. At least one of these tools, the IMU signal simulator, as well as the definition of the generic and extensible data interface, are candidates to be put into the public domain (for more details, contact any of the corresponding authors).

ACKNOWLEDGEMENTS

The research reported in this paper started around eight years ago at the former Institute of Geomatics and has been funded by means of several European projects. We would like to highlight IADIRA, ATENEA, ENCORE, CLOSE-SEARCH, GAL and GINSEC.

REFERENCES

Ahmadzadeh, R., 2015. Trajectory generation (3rd & 5th orders). `http://www.mathworks.com/matlabcentral/fileexchange/40278-trajectory-generation--3rd---5th-orders-`. Accessed: 24 November 2015.

Angelats, E., Molina, P., Parés, M. E. and Colomina, I., 2014. A parallax based robust image matching for improving multisensor navigation in GNSS-denied environments. In: Proceedings of the 27th International Technical Meeting of The Satellite Division of the Institute of Navigation (ION GNSS+ 2014), Tampa, USA, pp. 2132–2138.

Angelats, E., Parés, M. E. and Colomina, I., 2011. Methods, algorithms and tools for precise terrestrial navigation.

Colomina, I., Miranda, C. and Parés, M. E., 2012. Galileo's surveying potential. GPS World.

Ebinuma, T., 2015. GPS-SDR-SIM. `https://github.com/osqzss/gps-sdr-sim`. Accessed: 21 March 2015.

Fernández, A., Diez, J., de Castro, D., Dovis, F., Silva, P., Friess, P., Wis, M., Colomina, I., Lindenberger, J. and Fernández, I., 2011. ATENEA: Advanced techniques for deeply integrated GNSS/INS/LiDAR navigation. In: Proceedings of the 24th International Technical Meeting of The Satellite Division of the Institute of Navigation (ION GNSS), Portland, OR, USA, pp. 2395–2405.

Groves, P. D., Wang, L., Martin, H. and Voutsis, K., 2014a. Toward a unified PNT - Part 1, complexity and context: Key challenges of multisensor positioning. GPS World 25(10), pp. 18–49.

Groves, P. D., Wang, L., Martin, H. and Voutsis, K., 2014b. Toward a unified PNT - Part 2, ambiguity and environmental data: Two further key challenges of multisensor positioning. GPS World 25(11), pp. 18–35.

HardKernel, 2015. Odroid XU3 — Hardkernel. `http://www.hardkernel.com/main/products/prdt_info.php?g_code=G140448267127`. Accessed: 22 March 2015.

IFEN, 2015. NavX-NCS Professional GNSS Simulator. `http://www.ifen.com/products/navx-ncs-professional-gnss-simulator.html`. Accessed: 6 October 2015.

MathWorks, 2015. Three-Axis Inertial Measurement Unit. `http://es.mathworks.com/help/aeroblks/threeaxisinertialmeasurementunit.html`. Accessed: 25 November 2015.

Molina, P., Colomina, I., Vitoria, T., Silva, P. F., Skaloud, J., Kornus, W., Prades, R. and Aguilera, C., 2012a. Searching lost people with UAVs: the system and results of the CLOSE-SEARCH project. In: International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, Vol. XXXIX-B1, Melbourne, Australia, pp. 299–306.

Molina, P., Parés, M. E., Colomina, I., Victoria, T., Silva, P., Skaloud, J., Kornus, W., Prades, R. and Aguilera, C., 2012b. Drones to the rescue! Inside GNSS July/August, pp. 36–47.

Montaño, J., Wis, M., Pulido, J. A., Latorre, A., Molina, P., Fernández, E., Angelats, E. and Colomina, I., 2015. Validation of inertial and imaging navigation techniques for space applications with UAVs. In: Proceedings of Data Systems in Aerospace, Barcelona, Spain.

Navarro, J. A., 1999. Object-oriented technologies and beyond for software generation and integration in Geomatics. PhD thesis, Universitat de les Illes Balears.

Navarro, J. A., Parés, M. E., Colomina, I., Bianchi, G., Pluchino, S., Baddour, R., Consoli, A., Ayadi, J., Gameiro, A., Sekkas, O., Tsetsos, V., Gatsos, T. and Navoni, R., 2015. A redundant GNSS-INS low-cost UAV navigation solution for professional applications. In: International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, Vol. XL-3/W3, La Grande Motte, France, pp. 299–306.

Nievinski, F. and Larson, K., 2014. An open source GPS multipath simulator in Matlab/Octave. GPS solutions 18(3), pp. 473–481.

Parés, M. E. and Colomina, I., 2015. On software architecture concepts for an unified, generic and extensible trajectory determination system. In: Proceedings of the ION GNSS+, Tampa, USA, pp. 2518–2526.

Parés, M. E., Navarro, J. A. and Colomina, I., 2015. On the generation of realistic simulated inertial measurements. In: Proceedings of the ISS Gyro Symposium, Karlsruhe, Germany.

Rajamani, R., 2011. Vehicle dynamics and control. Springer Science & Business Media.

Roskam, J., 1995. Airplane flight dynamics and automatic flight controls. DAR corporation.

Silva, P., Silva, J., Caramagno, A., Wis, M., Parés, M. E., Colomina, I., Fernández, J., Diez, J. and Gabaglio, V., 2006. IADIRA: Inertial aided deeply integrated receiver architecture. In: Proceedings of the 19th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS 2006, Fort Worth, USA, pp. 2686–2694.

Silva, P., Silva, J., Peres, T., Colomina, I., Miranda, C., Parés, M. E., Andreotti, M., Hill, C., Galera, J., Camargo, P., Diez, J., Palomo, J. M., Barbin, S., Moreira, J., Streiff, G., Granemann, E. and Aguilera, C., 2011. ENCORE: Enhanced Galileo code receiver for surveying applications. In: Proceedings of the 24th International Technical Meeting of The Satellite Division of the Institute of Navigation (ION GNSS 2011), Portland, USA, pp. 3679–3689.

Skaloud, J., Colomina, I., Parés, M. E., Blázquez, M., Silva, J. and Chersich, M., 2015. Progress in airborne gravimetry by combining strapdown inertial and new satellite observations via dynamic networks. In: Proceedings of the IUGG 2015 conference, Prague, Czech Republic.

Velaga, N. R., Quddus, M., Bristow, A. L., Zheng, Y. et al., 2012. Map-aided integrity monitoring of a land vehicle navigation system. IEEE Transactions on Intelligent Transportation Systems 13(2), pp. 848–858.

Xie, P. and Petovello, M. G., 2015. Measuring GNSS multipath distributions in urban canyon environments. IEEE Transactions on Instrumentation and Measurement 64(2), pp. 366–377.

Yan, G., Wang, J. and Zhou, X., 2015. High-precision simulator for strapdown inertial navigation systems based on real dinamics from GNSS and IMU integration. In: Proceedings of the China satellite navigation conference (CSNC), Vol. 3, Xi'an, China, pp. 789–799.

Yang, Y., El-Sheimi, N., Goodall, C. and Xiaoji, N., 2007. IMU signal software simulator. In: Proceedings of the 2007 National Technical Meeting of The Institute of Navigation, San Diego, USA, pp. 532–538.