MODELING, IDENTIFICATION AND CONTROL, 2005, VOL. 26, NO. 3, 135–150
doi:10.4173/mic.2005.3.2

NavLab, a Generic Simulation and Post-processing Tool for Navigation†

KENNETH GADE*

Keywords: *Navigation Software, Simulation, Estimation, Analysis, Post-processing, Aided Inertial Navigation System, Kalman Filtering, Optimal Smoothing*

The ambition of getting one common tool for a great variety of navigation tasks was the background for the development of NavLab (Navigation Laboratory). The main emphasis during the development has been a solid theoretical foundation with a stringent mathematical representation to ensure that statistical optimality is maintained throughout the entire system. NavLab is implemented in Matlab, and consists of a simulator and an estimator.

- Simulations are carried out by specifying a trajectory for the vehicle, and the available types of sensors. The output is a set of simulated sensor measurements.
- The estimator is a flexible aided inertial navigation system, which makes optimal Kalman filtered and smoothed estimates of position, attitude and velocity based on the available set of measurements. The measurements can be either from the simulator or from real sensors of a vehicle.

This structure makes NavLab useful for a wide range of navigation applications, including research and development, analysis, real data post-processing and as a decision basis for sensor purchase and mission planning. NavLab has been used extensively for mass-production of accurate navigation results (having post-processed more than 5000 hours of real data in four continents). Vehicles navigated by NavLab include autonomous underwater vehicles (AUVs), remote operated vehicles (ROVs), ships and aircraft.

1. Introduction

For many navigation related activities it is very useful to have one common software tool. The tool should cover applications such as navigation system research and development, analysis and real data post-processing. With a long tradition of developing navigation systems, The Norwegian Defence Research Establishment (FFI) started development of such a tool in 1998. The result is NavLab (Navigation Laboratory), a powerful and versatile tool that serves a variety of navigation purposes. For the long-term success of this tool, a strong focus on a solid theoretical foundation and a flexible structure has been crucial.

1.1. *NavLab's theoretical foundation*

The most significant feature of NavLab is its solid theoretical foundation. NavLab is a result of an innovative research process to establish a completely general theoretical basis for navigation and for implementation of navigation systems. The development has led to the following contributions:

- A new stringent and unified system for notation and mathematical representation

*Norwegian Defence Research Establishment (FFI)
†This article appeared in the November 2004 issue of the European Journal of Navigation and is displayed with permission. Copyright 2004 GITC bv.

- A unified design and implementation of algorithms and aiding techniques for the Kalman filter, where statistical optimality is maintained throughout the entire system
- Elimination of numerical problems by
  - Deducing and implementing exact formulas (rather than approximations)
  - Using only nonsingular representations
  - Controlling accumulation of the computer's inherent round-off errors

Articles reporting the above work will be published, but currently the most relevant report available is Gade (1997).

1.2. *A flexible structure*

The main structure of NavLab is shown in Figure 1. NavLab's different components can be used alone or together, allowing a variety of applications. A list of usages is given in Section 4.

The simulator can simulate artificial measurements from a chosen scenario. The estimator will, based on the available set of measurements from either the simulator or from sensors of a real vehicle, make the best possible estimates of position, attitude,

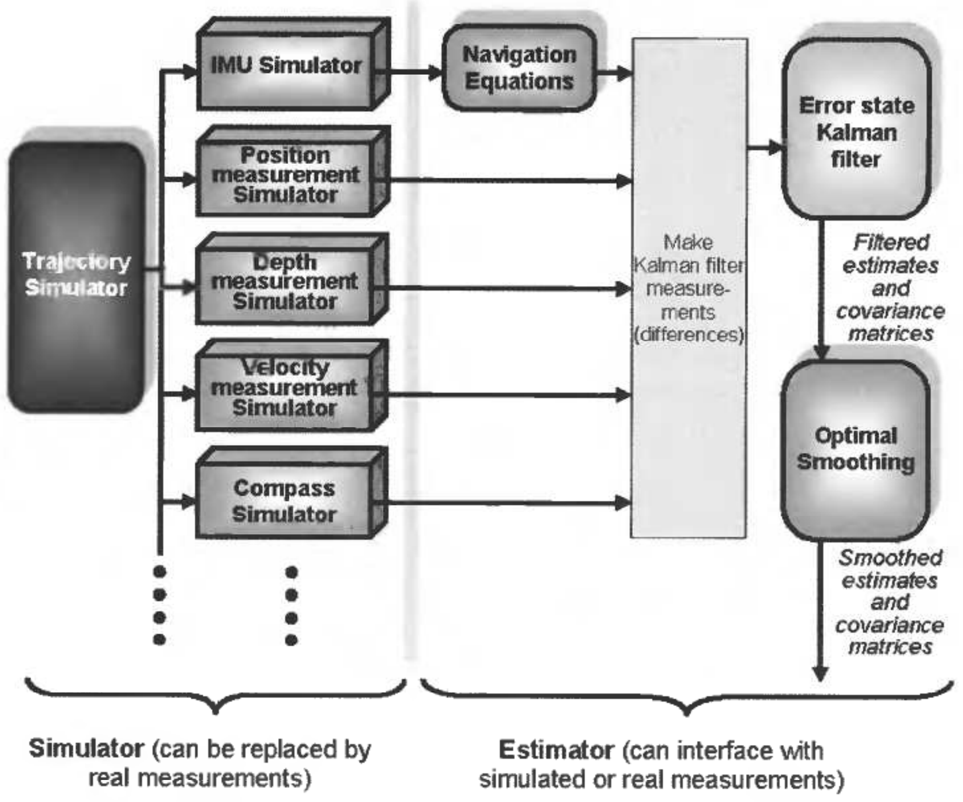
IMU Simulator
Navigation Equations
Position measurement Simulator
Error state Kalman filter
Trajectory Simulator
Depth measurement Simulator
Make Kalman filter measure-ments (differences)
Filtered estimates and covariance matrices
Velocity measurement Simulator
Optimal Smoothing
Compass Simulator
Smoothed estimates and covariance matrices
Simulator (can be replaced by real measurements)
Estimator (can interface with simulated or real measurements)

Figure 1. NavLab main structure. Note: The colors used in the figure correspond to the colors of the graphs generated by the different parts of NavLab (black is the true value, blue is the measurement etc).

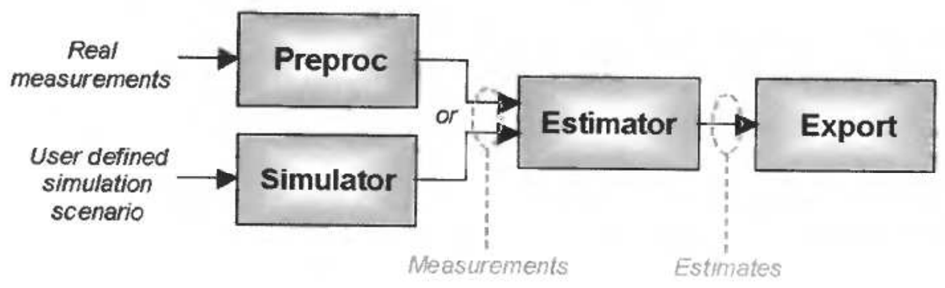

Figure 2. NavLab program modules.

velocity and sensor errors. The simulator and estimator are described in more detail in Sections 2 and 3.

In addition to the simulator and estimator, NavLab includes:

- A pre-processing tool (Preproc), which is used to handle real measurements (by removing outliers, compensating for lever arms and misaligned sensors, converting measurements to the correct format etc).
- An export tool, which creates files for exporting to other programs (containing the estimated position, attitude and velocity).

Figure 2 shows the NavLab program modules. Different modules are used in different cases. Typical examples are:

- *Simulations*: Simulator $\rightarrow$ Estimator
- *Post-processing of real data*: Preproc $\rightarrow$ Estimator $\rightarrow$ Export

The modules interface each other via files of a specified format (see Gade (2003)), or via memory to save time.

2. Simulator

The trajectory simulator can simulate any vehicle trajectory specified by the user. In addition, the user specifies a set of available sensors and their characteristics. Based on the specified trajectory and sensor characteristics, the sensor simulators calculate a set of artificial sensor measurements.

2.1. *Trajectory simulator*

The coordinate systems *I* (Inertial), *E* (Earth), *L* (Local) and *B* (Body) are simulated (see Gade (2003) for definitions). All relevant positions, orientations, linear and angular velocities, accelerations and forces describing the trajectory are calculated.

Features:

- Any trajectory in the vicinity of the Earth can be simulated (with unlimited complexity).
- All vehicle attitudes can be simulated without singularities.
- All possible vehicle positions relative to the Earth can be simulated without singularities.

- Includes all Coriolis and centripetal effects due to the rotating Earth and own movement over the Earth curvature.
- Includes WGS-84 gravity model and elliptic Earth model.

Trajectories are specified in the trajectory simulator by first giving the initial position, attitude and velocity, and then specifying changes in attitude and velocity as a function of time. When developing a trajectory simulator, the actual mathematical quantities that are used to describe these changes must be selected carefully, to ensure that it is simple for the user to express a trajectory that follows the Earth ellipsoid in both position and attitude. Selecting the mathematical quantities[1] $\boldsymbol{\omega}_{LB}^{B}$ and $\dot{\boldsymbol{v}}_{EB}^{B}$ actually makes this just as

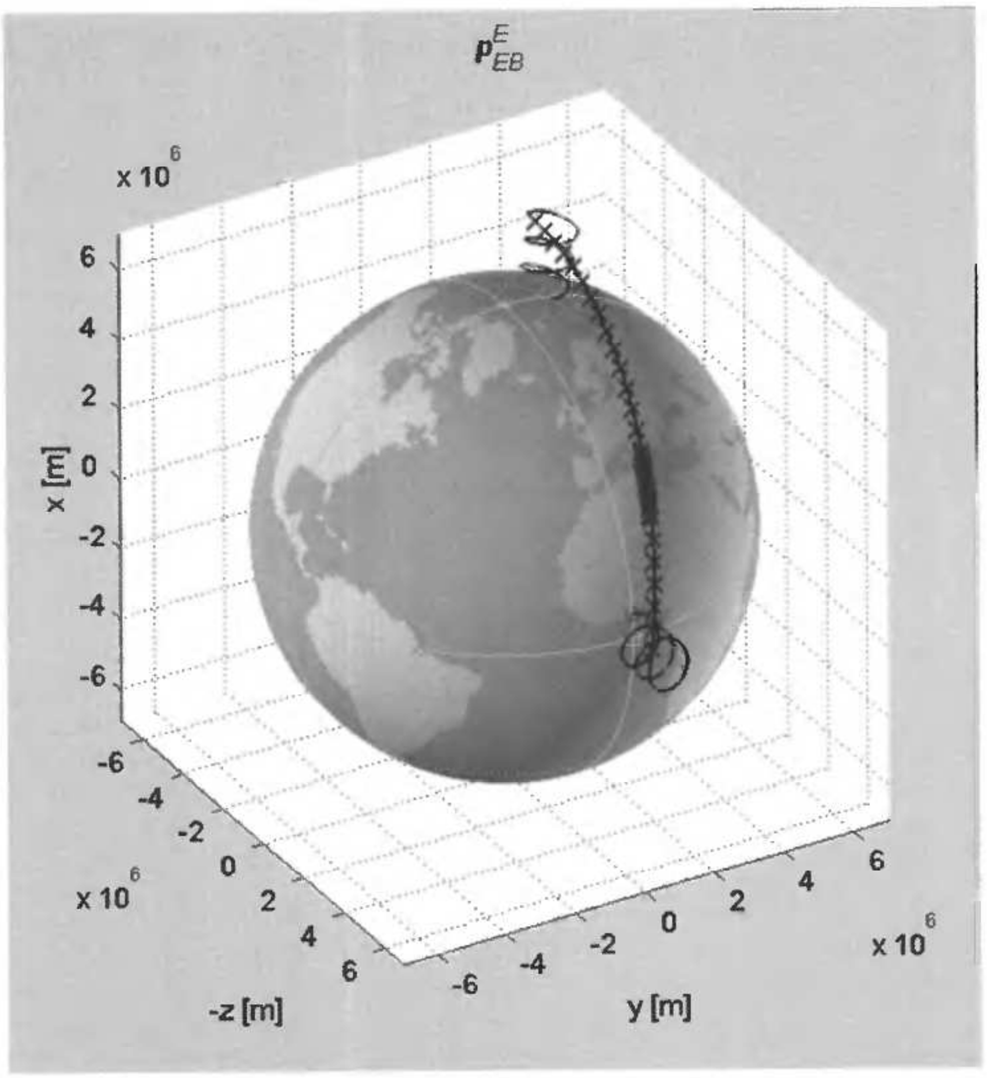

Figure 3. Earth plot from NavLab. Black circle: Starting point. Black line: True trajectory (from the trajectory simulator). Blue crosses: Simulated position measurements.

[1] $\boldsymbol{\omega}_{LB}^{B}$ is the angular velocity of the body, *B*, relative to *L*, where *L* is a local system with zero angular velocity relative to Earth about its vertical axis (see Gade (1997) or Gade (2003) for more details). $\dot{\boldsymbol{v}}_{EB}^{B}$ is the velocity of *B* relative to Earth, differentiated in the *B* system.

simple for the user as it would have been if the surface of the Earth were planar. Thus if no changes in these quantities are specified, the vehicle will travel around the Earth at constant depth/height if the initial velocity was horizontal.

Figure 3 shows an example trajectory from the simulator. This trajectory is simply specified by two periods of constant change in attitude (angular velocity about z, $\boldsymbol{\omega}_{LB}^{B} = [0 \quad 0 \quad 1]^T$ deg/s) and two periods of constant change in velocity (deceleration/acceleration in z, $\dot{\boldsymbol{v}}_{EB}^{B} = [0 \quad 0 \quad 20]^T$ m/s^2).

Using a plugin for NavLab, it is also possible to specify the trajectory by giving a dynamical model of the vehicle and then marking 3D waypoints in a map, see Svartveit & Berglund (2003).

2.2. *Sensor simulators*

The most significant error types, such as white-noise, colored noise and scale factor error are included in the sensor simulators, and any other types can also be added. The magnitude, time-constants and other parameters that describe the different errors are user selectable, and can be given as fixed values or as functions of time.

The sensor simulators can produce measurements at any user-specified time. This can be specified as a constant rate during the entire simulation, different rates in different intervals, or each single time of measurement can be specified in a time-series. Figure 3 shows position measurements with one period of high rate, and also periods of low and zero rate.

3. **Estimator**

The main purpose of the estimator is to estimate a vehicle's position, attitude and velocity. This is done by combining all available knowledge such as sensor measurements and mathematical models of the sensor errors. The optimal (given certain assumptions) method of combining this knowledge is by means of a Kalman filter[2] (see Minkler & Minkler (1993) for details). Thus, if the model used in the Kalman filter is correct, all information is used optimally, and no better estimates can be made. An example illustrating this is the concept of gyrocompassing, i.e. finding north by inspecting the direction of the Earth's angular velocity, measured by the gyros. Gyrocompasses are manufactured containing gyros, accelerometers and dedicated algorithms for this purpose. When the same sensors are available for the estimator, it will gyrocompass optimally as a natural part of its estimation procedure.

The main structure of the estimator is given in Figure 4. Measurements from the IMU (Inertial Measurement Unit) are integrated by the navigation equations (see Section 3.1) to calculate position, attitude and velocity. Each time-step where a measurement from any of the aiding sensors is available, it will be compared to the corresponding quantity from the navigation equations, and the difference is sent as a measurement to the Kalman filter.

Note that each of the sensors shown in Figure 4 are general and can represent different types, e.g. NavLab has used different types of position measurements, including range measurements to a known position (see Jalving *et al.* (2003a) or Jalving *et al.* (2003b) for examples of different sensor types that have been integrated).

[2] If future measurements are available, a better estimator exists, see Section 3.2.

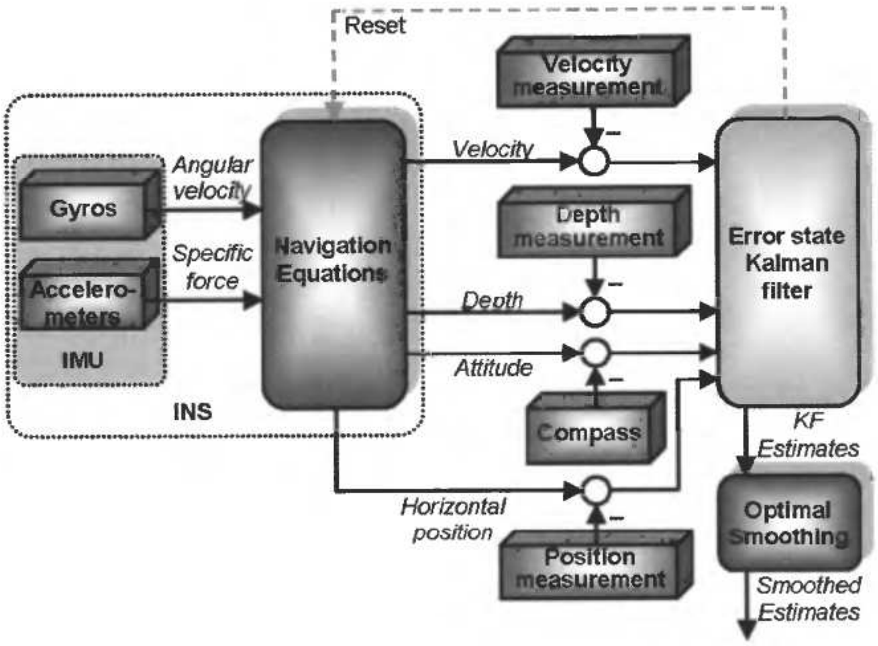

Figure 4. Estimator main structure (simplified). The sensors shown can be either simulated or real. (INS: Inertial Navigation System).

The navigation equations and optimal smoothing are described in Sections 3.1 and 3.2.

Features:

- The estimator accepts arbitrary time-series of measurements from all sensors.
- Along with each single sensor measurement, new sensor parameters can be specified, describing that particular measurement, hence describing a varying quality.
- Zero velocity update (ZUPT) and depth/height measurements are included in the same Kalman filter in an optimal manner.
- The horizontal position measurements are nonsingular (i.e. with maximum accuracy also near/at the poles).
- Iterated Extended Kalman filter is used to improve the performance in cases of significant nonlinearities.

3.1. *Navigation Equations*

The navigation equations calculate position, attitude and velocity based on the IMU measurements, as shown in Figure 4.

Features:

- Nonsingular for all positions and attitudes
- Foucault wander azimuth
- Direction cosine matrix attitude update

- Numeric drift control
- WGS-84 gravity model and elliptic Earth model
- Trapezoid updates to prevent systematic errors from the forward or backward Euler methods

3.2. *Optimal Smoothing*

The Kalman filter is the optimal estimator at time t, when measurements before and including t are used, thus it is well suited for real-time estimation. However, if measurements after t are also available (which is the case for post-processing, see Section 4.1), it is possible to make a better estimator at time t, by using these additional measurements. The best possible algorithm, utilizing all measurements both before and after t, is called *optimal smoothing* (see Minkler & Minkler (1993) for details).

- This algorithm is effectively doubling the set of relevant measurements for each estimate, since the *next* x seconds of measurements are normally just as important as the *previous* x seconds.
- A symmetrical interval of past and future measurements prevents a systematical delay in the estimates, which is unavoidable in real-time estimators.
- Another limitation of an optimal real-time estimator (Kalman filter) is its inability to deliver estimates that are in accordance with the process model. At each time-step such estimators make a prediction (that is in accordance with the process model), but when a new measurement arrives, it is weighed against the prediction to give a new updated estimate. Unexpected[3] measurements thus lead to jumps in the estimates that are not in accordance with the process model (e.g. an unexpected velocity measurement leads to a jump in the velocity estimate that corresponds to an acceleration that is too large according to the process model). Since no measurements are unexpected for the smoothing algorithm, this problem is eliminated, and the smoothed estimate is always in accordance with the process model (hence the name "smoothing").

Figure 5 shows an example of position estimation uncertainty (1σ) in the Kalman filter and in the optimal smoothing. Position measurements are unavailable in an interval of 2 hours, and in this period the Kalman filter estimation uncertainty grows, before dropping instantly when position measurements become available at the end. The smoothing algorithm on the other hand, utilizes the position measurements at the end during the whole interval, and thus has a maximum uncertainty in the middle of the interval. At the last time-step, no future measurements are available and the two algorithms give equal estimates.

3.2.1. *Performance in cases with large modeling errors (robustness)* Another property of the smoothing algorithm, that is often even more important than the improved accuracy, is its robustness. As mentioned above, smoothed estimates are always in accordance with the process model, and this quality is crucial in cases with wrong models or faulty measurements. If a measurement has an error that is significantly larger than what was modeled in the Kalman filter, a large jump in the estimates from the real-time filter is inevitable. A real-data example of such a jump is shown in Figure 6,

[3] All measurements that are not exactly equal to the predicted value are unexpected, which in practice means every measurement.

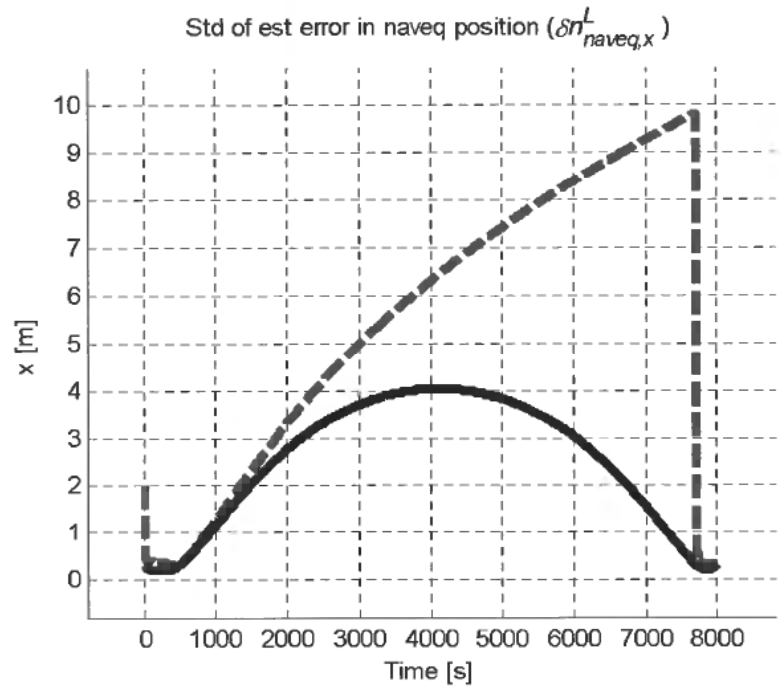

Figure 5. Estimation uncertainty in north-position by Kalman filter (green/dashed) and optimal smoothing (red/solid). (A straight-line trajectory to the east, at latitude 45° is simulated. Sensors: 1 nmi/h class IMU, 600 kHz DVL. Position measurements are available the first 500 seconds and the last 300 seconds.)

where a position outlier (wild-point) with an error of about 41 meters is present.[4] Since the Kalman filter expects a total position measurement uncertainty of 2.4 m (1σ), the error of this measurement is above 17 sigma, and hence extremely unlikely according to the model. In the example, the outlier is followed by a period of position measurement dropout (which is typical), and thus the filtering error remains until the sound[5] measurements bring the estimate back on track. The smoothing algorithm however, also seeing the measurements from all sensors *after* the outlier, is barely affected, even though it uses the same sensor model as the Kalman filter.

The optimal smoothing algorithm is also robust against systematic sensor errors. In a HUGIN 3000 navigation accuracy verification sea trial in October 2000 (described in Section 5.2.2), there was a constant error in the DVL (Doppler Velocity Log) measurements that was above 8.3 sigma (due to an incorrect DVL configuration in this particular trial). This huge[6] unmodeled velocity error led to a position error in the order of 10–15 m for the real-time estimates, while the smoothing, using the same model, proved a performance of 1.2 and 1.7 meters (1σ north and east), see Figure 8 in Section 5.2.2.

[4] Outliers of this magnitude will by default be automatically removed in NavLab by a wild-point detection algorithm, but is left here for demonstration.

[5] I.e. in accordance with the Kalman filter model.

[6] According to the model, the probability of an error of this magnitude in one measurement is only about 10^{-16}, and in this trial all measurements had this error!

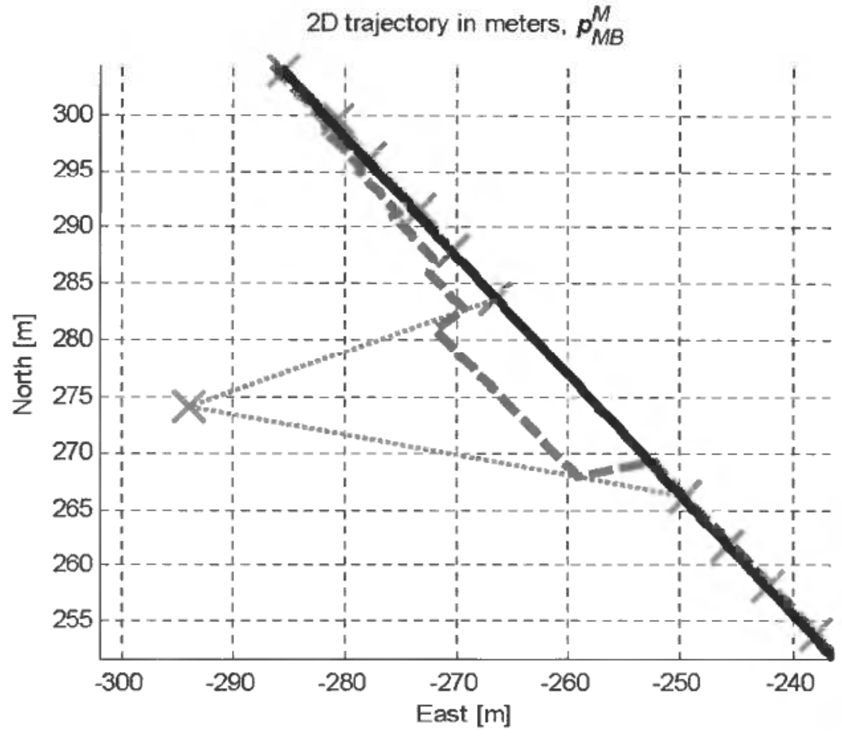

Figure 6. Trajectory from the HUGIN 1000 AUV. The track shows the vehicle going northwest. Blue/dotted: position measurement from DGPS + USBL (differential GPS + Ultra Short Base Line acoustic positioning). Green/dashed: Kalman filtered estimate. Red/solid: Smoothed estimate.

4. NavLab usage

NavLab has been extensively used by numerous different users since 1999, including several international research groups, universities and commercial survey companies. The flexible structure of NavLab makes it useful for a wide range of applications. Some users are only working with simulated data, whereas others use the estimator alone to post-process real data. Finally, there are many cases where both simulations and real data processing are of interest. A summary of current NavLab usage is given below.

Navigation system research and development (using simulations and real data)

- Development, testing and comparison of new navigation concepts and algorithms, including new aiding sensors and aiding techniques.
- Development of real-time navigation systems, where the algorithms are implemented and tested in NavLab, and then ported to the real-time system. A typical development process is:

  – Implement algorithms in NavLab
  – Test in simulations (NavLab)
  – Test with real data (NavLab)

- Port algorithms to the real-time navigation system (C++ or similar program language)
- Test real-time system

The real-time navigation system in the HUGIN vehicles was developed using NavLab (see Jalving *et al.* (2003b) for a description of the real-time navigation system and Hagen *et al.* (2003) or The HUGIN AUV Programme homepage for an overview of the HUGIN AUV Programme).

Analysis of a given navigation system (using simulations and real data)

- Analysis of navigation system behavior under different maneuvers/trajectories and sensor configurations.
- Robustness analysis. The performance of the estimator is studied for the cases of:
  - Wrong sensor models used in the Kalman filter
  - Sensor dropouts
  - Sensor errors

Teaching navigation theory (using simulations)

By specifying appropriate simulations, everything from basic principles to complex mechanisms can be demonstrated and visualized.

Decision basis for navigation sensor selection/purchase (using simulations)

Simulations of the relevant scenarios are carried out to investigate how varying quality of the different sensors will affect the obtainable navigation performance. Parameters for different sensors available in the market are usually entered for comparison. The goal is to achieve a well-balanced and economical sensor suite.

Decision basis for mission planning (using simulations)

Even if the set of sensors is given, the navigation accuracy can vary significantly with the mission type. Important mission parameters include:

- Activation/deactivation of sensors or change of measurement rate (reasons to deactivate might be to stay covert, avoid interference with other systems or just to save power)
- Going to areas where certain measurements are available or are more accurate (e.g. go close to bottom to get DVL bottom track, go close to a transponder or go to surface for GPS measurements)
- Running maneuvers to increase the observability in the estimator
- Running in patterns that cancel out error growth

When setting up complex mission plans, simulations are helpful to ensure effective missions that meet the navigation accuracy requirements for all parts of the mission (transit phase, mapping phase etc).

Post-processing of real navigation data (using real data)

Post-processing of real data improves the navigation accuracy, robustness and integrity compared to a real-time navigation solution. See 4.1 for more details.

Tuning of real-time and post-processing navigation systems (using real data)

Proper Kalman filter tuning is essential for optimal estimation accuracy. Tuning is often based on the sensor specifications, but the actual sensor performance can differ from these numbers, and in such cases the tuning should be based on empirical data. Finding the correct tuning based on a recorded data set is best done by means of the error estimates from the smoothing algorithm.

Sensor evaluation (using real data)

After purchasing a new sensor, an evaluation of the sensor is usually desired. Large sensor errors might be detected by inspecting the measurements from this sensor alone, but for a more thorough sensor evaluation, the measurements should be compared with other sensors (with uncorrelated errors) or a known reference. Running a relevant mission or lab test and analyzing the result in NavLab will usually reveal errors above the specification and often also the characteristics of such errors.

Improving sensor calibration (using real data)

Even if a sensor is approved in an evaluation, it can exhibit systematic errors, typically due to imperfect calibration or misaligned mounting. Such (deterministic) errors should be removed before sending the measurements to the estimator, otherwise the performance will be reduced (in particular for the real-time Kalman filter). To find these systematic errors, the smoothing algorithm should be used, as it is significantly better than the real-time filter at estimating such errors. When systematic errors are known, they can be compensated for in future missions.

4.1. *Using NavLab for real data post-processing*

For vehicles storing their navigation sensor measurements during missions, it is possible to make post-processed estimates of position, attitude, velocity and sensor errors. There are many situations where these estimates are of great interest after the mission is finished, for instance if the vehicle has recorded payload data that require accurate geo-referencing (e.g. bathymetric data for terrain maps or image data for object detection). NavLab is well suited and extensively used to produce optimal post-processed navigation results. These results are valuable also when the vehicle has calculated and stored real-time navigation estimates. When the time constraints allow, post-processed estimates are preferred to the real-time estimation results, since important properties such as estimation accuracy, robustness and integrity are improved:

- *Increased accuracy* is mainly due to the use of the optimal smoothing (see Section 3.2). In addition, real-time issues like delayed measurements and incomplete data sets from remote sensors[7] are eliminated. Finally, the absence of a real-time computing requirement makes it possible to use iterations to improve estimation performance.
- *Improved robustness* is partly due to the smoothing algorithm, which in general is more robust against degraded sensor performance than the real-time Kalman filter

[7] For instance a surface vehicle measuring the AUV position by means of DGPS + USBL. A full set of measurements is not transmitted to the AUV in real time, but is available for use in NavLab after the mission.

(see Figure 6 and Figure 8). In addition, the possibility of rerunning the estimation increases the ability to recover a faulty data set. To do so, one can modify either the degraded sensor measurements or the filter tuning (or both) to get the best possible navigation for the faulty data set.

- The *Integrity* of the estimator, i.e. the ability to detect degraded sensor performance and degraded total navigation performance, is critical for the users of the navigation data. The optimal smoothing algorithm has a very high capability of detecting reduced sensor quality. In addition it can often tell which sensor is having problems. When deviations are detected, the data can usually be rerun as described above, and the final estimates will be reliable (i.e. more accurate and associated with a trustworthy accuracy estimate). In practice, the ability to recover the navigation data in the case of degraded sensor performance means that the need for a new mission is avoided.

Also, the smoothing might allow purchasing less expensive sensors or using them less frequently, and still obtaining the required accuracy. For instance, a submerged vehicle might need to surface to get position measurements. In Figure 5, we see that with a position accuracy requirement of 5 meters, the real-time filter would require position measurements after a period of 2500 seconds, while with smoothing a position accuracy better than 5 meters is obtained even with a 2 hours dropout interval.

Post-processing of real data has become one of the most important NavLab applications, and through mass-production of accurate navigation results more than 5000 hours of recorded payload data has been positioned. Any vehicle with recorded sensor data can be navigated, and currently AUVs, ROVs, ships and aircraft have been navigated with NavLab.

4.2. *Practical usage*

NavLab is written in the mathematical programming language Matlab (see Mathworks homepage), but it can also be compiled to a Windows application (exe file). Post-processing of a recorded data set with 3–5 Hz Kalman filter update rate and 100 Hz IMU data, is approximately 15 times faster than real-time, when using a 3 GHz Pentium 4 processor.

The user interface can vary from "Scientific", where all parameters and steps are fully controllable, to "One-click" (see Svartveit (2004)) where all processes are automated. In Scientific mode, a general multi-menu based plot function is used after a simulation or estimation. This function plots a range of figures containing numerical summaries and many different 2D and 3D plots with a total of more than 500 graphs, for results analysis. The plot function is also programmable to show only a predefined subset of plots for users wanting just a simplified summary of the results. The very simplest output is used in the One-click mode, where a green/red light at the end of the estimation indicates if the data was OK or not.

5. Verification of estimator performance

Verification of the estimator performance has been a crucial part of the NavLab development. Both the Kalman filter and the optimal smoothing calculate an expected uncertainty for their estimates, which is the theoretically optimal accuracy obtainable for the given scenario. When using a correct model in the estimator, the actual estimation error should be as small as the theoretical uncertainty limit. A correct model can be used when the measurements are from the simulator, but since the real world has infinite complexity, it is impossible to use a completely correct model in the estimator when using real data. In cases where the model used by the estimator differs from the model generating the measurements, the actual estimation error will be larger than the theoretical limit. The most challenging part of the estimator development is to keep its error as close as possible to the theoretical limit in cases of modeling errors (and nonlinearities). To minimize the loss of accuracy, a very careful design and implementation of all parts of the estimator is vital. In this section it is demonstrated that it is possible to achieve a performance close to the optimal under a range of different non-ideal conditions.

5.1. *Verifying performance using the simulator*

The simulator, having a more complex nonlinear system model than the estimator, is an effective tool for verifying the estimator performance. Any scenario can be tested and different modeling errors can be used. After running the estimator, the plot function will calculate and plot the true estimation error and compare it with the theoretical estimation uncertainty (also Monte Carlo simulations can be run to determine the statistics of the error). Thorough and extensive testing of the estimator since 1999 by different research groups, testing a variety of scenarios, has proven the estimator to be very robust and to give close to optimal performance in all scenarios.

5.2. *Verifying performance using real data*

The ultimate test of the estimator is to use real data from a representative mission, where the trajectory and all sensor errors are (by definition) totally realistic. The challenge with real runs is that it is more difficult to investigate the estimation errors, since the true trajectory is unknown. However, some possibilities do exist, and these are discussed in the following.

5.2.1. *Redundant sensors* A significant sensor measurement can be made unavailable for the navigation system, and later be used as a reference. For instance, a surface ship might follow a submerged AUV, continually measuring its position using DGPS + USBL, but not sending the measurements to the AUV. The AUV, typically using an IMU, a depth sensor, a DVL and in some cases a compass, will have a drift in position that after a while will be significantly larger than the uncertainty in the DGPS + USBL position measurements. Hence the estimation error is observable and is compared with the theoretical uncertainty. All such tests have documented a very high estimator performance, that was in accordance with the theoretical uncertainty, see Jalving *et al.* (2004) and Jalving *et al.* (2003b).

5.2.2. *Verifying the positioning by means of mapped objects* For a seabed mapping vehicle, an accurate positioning of the final map is essential, and estimates of the vehicle's 6 degrees of freedom (position and attitude) are used to position the bathymetric data. Estimation errors in vehicle position will be directly translated to errors in the map position, while the effect of attitude errors will depend on the geometry between the vehicle and a given patch of the seafloor. A crucial test of the entire navigation system is to verify the position accuracy in the final maps. In such tests, all available aiding sensors are used so that the maximum accuracy is evaluated.

Customers buying HUGIN and NavLab for detailed seabed mapping have had a strong focus on position accuracy of the maps and have thus run navigation performance trials as part of the customer acceptance tests. These trials determine if the real-life performance of the estimator match the accuracy that was predicted in NavLab simulations before the vehicle was built. The standard method is to map the same object at the seafloor several times, comparing the position estimate of each individual object observation. Errors that are uncorrelated between each passing will be visible, as the object will be positioned differently in each observation. Correlated errors are typically following the AUV or a ship giving DGPS + USBL measurements (e.g. timing problems, systematic velocity error and misaligned acoustic positioning transducer). Hence, to also reveal these errors, different headings are used for the AUV and ship for each passing ("wagon wheel pattern", as shown in Figure 7). Figure 7 shows maps from HUGIN 3000 in an accuracy test carried out by the HUGIN customer C&C Technologies at 1300 m water depth in the Gulf of Mexico in October 2000. 11 different headings were used (5 of the lines were mapped in opposite directions) when mapping the object (a wellhead), to maximize the visibility of any correlated errors following the AUV or ship. The positions of the wellhead observations when using NavLab smoothing are shown in Figure 8, obtaining an accuracy of 1.2 m and 1.7 m (1σ) north and east (even with a large unmodeled DVL error present, see Section 3.2.1). The theoretical estimation uncertainty in the smoothed position was about 1.7 m (1σ, north and east) during the passings. 60 m from the wellhead, but within the swath width, another object (natural feature) was also visible in the data. Since the object is 60 m off the center of the maps, a somewhat higher uncertainty is expected due to the AUV heading uncertainty (and also due to the increased mapping sonar uncertainty), and indeed this object had a distribution of 1.3 m north and 1.9 m east.

The test shown is the only test where a large unmodeled sensor error was present. After this test many similar navigation accuracy evaluations have been carried out by

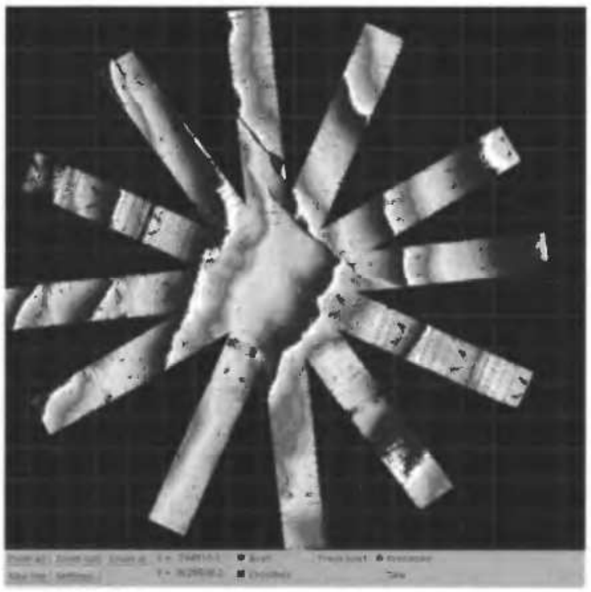

Figure 7. A wellhead is mapped repeatedly with different headings to evaluate the positioning accuracy of the final map.

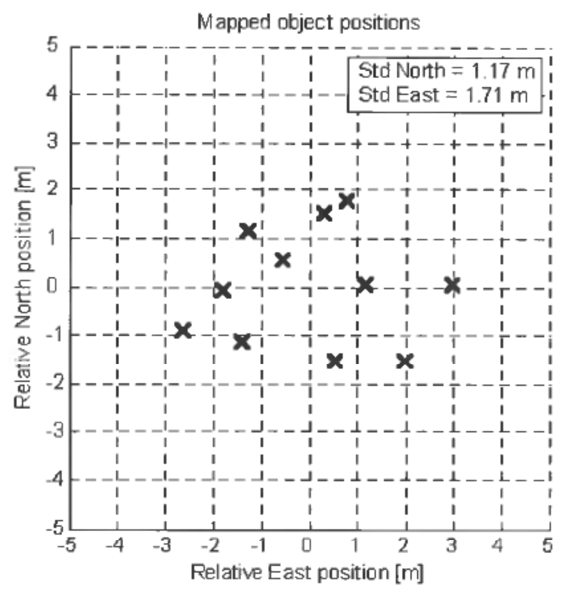

Figure 8. The mapped positions of the wellhead (at 1300 m depth) using NavLab smoothing.

different HUGIN customers, with other vehicles and navigation sensors. The accuracy has been tested down to a depth of 2200 m (obtaining 2.3 m and 3.3 m accuracy in north and east), and in general the tests have proven exceptionally good estimator accuracy, even slightly better than the anticipated theoretical uncertainty limit. The reason for this is a combination of the navigation sensors performing somewhat better than their specifications and the estimator producing close to optimal estimates.

6. Conclusions

NavLab is a powerful and versatile tool with usage ranging from research and development by scientists and academics, to mass production of high-accuracy maps by commercial companies (having post-processed more than 5000 hours of data from around the world).

Even when a real-time navigation system is available, it is often beneficial to post-process the data with NavLab:

- The navigation results, i.e. estimates of position, attitude and velocity, will be more accurate and smooth (no jumps in the data).
- The navigation results will be more reliable (any critical sensor errors are detected).
- Even in cases of sensor degradation or failure, accurate navigation can often be obtained (no need for a new mission). This is due to the increased robustness of smoothing and the possibility to rerun the data.
- Lower quality navigation sensors might be used, while still obtaining satisfactory navigational accuracy.

The most significant feature of NavLab is its theoretical foundation, where statistical optimality is maintained throughout the entire system. This has been repeatedly demonstrated through extensive performance verifications, both with simulations and real missions. These tests have proven very high estimator performance, close to the theoretical optimum.

Acknowledgements

The author wishes to thank the rest of the Navigation Group at FFI, Bjorn Jalving in particular, for significant contributions to the development of NavLab. Also thanks to Bjorn Jalving, Kongsberg Maritime and HUGIN customers for organizing, carrying out and reporting a range of different tests of the navigation system accuracy.

References

GADE, K. (1997). Integrering av treghetsnavigasjon i en autonom undervannsfarkost (in Norwegian), FFI/RAPPORT-97/03179.

GADE, K. (2003). NavLab—Overview and User Guide November 2003, FFI/RAPPORT-2003/02128.

HAGEN, P. E., STORKERSEN, N. & VESTGARD, K. (2003). The HUGIN AUVs—multi-role capability for challenging underwater survey operations. EEZ International, Summer 2003.

JALVING, B., BOVIO, E. & GADE, K. (2003a). Integrated inertial navigation systems for AUVs for REA applications. Proceedings from MREP 2003, NATO SACLANT Undersea Research Centre, May 2003, Italy.

JALVING, B., GADE, K., HAGEN, O. K. & VESTGARD, K. (2003b). A Toolbox of Aiding Techniques for the HUGIN AUV Integrated Inertial Navigation System. Proceedings from Oceans 2003, September 22–26, San Diego, USA.

JALVING, B., GADE, K., SVARTVEIT, K., WILLUMSEN, A. & SORHAGEN, R. (2004). DVL Velocity Aiding in the HUGIN 1000 Integrated Inertial Navigation System. Proceedings from ADCPs in Action 2004, June 3–4, Nice, France.

MATHWORKS HOMEPAGE: www.mathworks.com.

MINKLER, G. & MINKLER, J. (1993). Theory and Application of Kalman Filtering, Magellan Book Company.

SVARTVEIT, K. & BERGLUND, E. (2003). NavLab Plugin: Waypoint editor, FFI (to be published).

SVARTVEIT, K. (2004). NavLab One-Click, FFI (to be published).

THE HUGIN AUV PROGRAMME HOMEPAGE: www.ffi.no/hugin.