INTEGRATION OF GNSS, ELECTRO-OPTICAL SENSORS AND INERTIAL FOR NAVIGATION USING A PLUG-AND-PLAY ARCHITECTURE BASED ON THE ROBOTIC OPERATING SYSTEM (ROS)

Maarten Uijt de Haag [a], Daniel Serrano [b], Jordi Batlle [b], Evan Dill [a]

[a]Ohio University, Athens, Ohio, USA
[b]Aerospace Research and Technology Center (CTAE), Barcelona, Spain

KEY WORDS: Plug-and-play systems, sensor integration, Global Navigation Satellite Systems, inertial navigation, electro-optical sensors, robotic operating system.

ABSTRACT:

This paper discusses a hardware and software architecture that is capable of using all or a subset of its sensor inputs to determine a platform's position, velocity and attitude (PVA). The target application is PVA determination of unmanned aerial vehicles (UAV) and autonomous ground vehicles (AGV) in urban or indoor environments, where GNSS is only sparsely available or even unavailable, either intentional or unintentional. The current system input set may include data from GNSS receivers, data from inertial sensors, two-dimensional (2D) gray-level (intensity) imagery, 2D color imagery, and laser radar (Ladar) data. However, at any time during operation one or more sensors can be removed (unplugged) or added (plugged in) without PVA service interruption as long as the remaining sensors can achieve enough observability allowing the operator to change the sensor configuration during operation. This paper provides a detailed description of the proposed sensor acquisition; integration and PVA estimation approach from a hardware and software point of view using small size sensors, a small-size computer platform and the Robotic Operating System (ROS), an operating system-like framework for robotics software.

1. INTRODUCTION

Within the context of this paper the objective of a navigation system is to provide an accurate Position, Velocity, Attitude, and, if possible, Time (PVAT) estimate expressed in the coordinates of some geometric reference. The required navigation performance (RNP) depends on the platform's operational scenario that must be supported and is typically expressed in terms of accuracy, integrity, availability, and continuity. When a single navigation aid (navaids) is not capable of achieving the defined RNP, a combination (i.e. integration or fusion) of data from multiple sensors (or navigation aids) may be used to achieve the RNP.

To enable operation of Unmanned Aerial Vehicles (UAVs) and Autonomous Ground Vehicles (AGVs) at any time in any environment, a precision navigation, attitude, and time capability is required that is robust and not solely dependent on the Global Navigation Satellite System (GNSS). In urban and indoor environments a GNSS position capability may not be available due to shadowing, significant signal attenuation and multipath caused by buildings, or due to unintentional or intentional interference, denial or deception. To improve availability and guarantee continuity of service in these environments, GNSS can be integrated with an Inertial Measurement Unit (IMU) or improved by increasing its sensitivity by using external data sources (i.e. assisted GPS). This integration strategy is successful in many cases, but does not cover all possible scenarios. An alternative method is the topic of discussion in this paper. Note that integration of multiple sources of data may not only improve the accuracy of the position and attitude estimate, but also add integrity, continuity and availability to the solution.

Alternative navigation technologies may include (a) the integration of inertial sensors with imagery and Ladar [3], (b) beacon-based navigation (i.e. psuedolites) [4], (c) or navigation using signals of opportunity [4]. The focus in this paper is on a plug-and-play architecture for category (a). Two-dimensional (2D) laser scanners have been used extensively to enable navigation of robots in an indoor environment. For example, [5] describes a method to estimate the translation and rotation of a robot platform from a set of extracted lines and points using a 2D sensor. [6] discusses the feature extraction and localization aspects of mobile robots and addresses the statistical aspects of these methods. [7] introduces improved environment-dependent error models and establishes relationships between the position and heading uncertainty and the laser observations, thus enabling a statistical assessment of the quality of the estimates. In [8] 2D scanning Ladar measurements are tightly integrated with IMU measurements to estimate the relative position of a van in an urban environment. The idea of using planar surfaces for 2D localization is described in detail in [9]. Use of 3D features in Flash Ladar imagery was introduced in [10]. [11] describes the integration of Ladar with imagery and inertial data. In that concept the Ladar is used to remove the depth uncertainty present in the imagery data by simultaneously observing 2D features in a plane and 3D laser points in that same planar surface.

2. SENSOR INTEGRATION

As mentioned in the previous section, the goal of sensor integration is to improve one or more RNP parameters for operations where a single navaid solution would not suffice. The basic philosophy behind the integration approach in this paper is visualized in Figure 1. An Inertial Measurement Unit (IMU) is chosen as the core sensor since it is self-contained and

therefore does not depend on "external" infrastructure such as a feature-rich environment, or radio-frequency (RF) signals. Choosing to build the system around an IMU will provide it with position and attitude estimates in any environment and in any scenario. However, when an Inertial Navigation System (INS) is used in a standalone manner the position and attitude estimates will drift over time. Integration with secondary sensors is, therefore, required to mitigate the drift error by periodically "resetting" or estimating the inertial errors. The position and attitude change estimates from the 2D images and 3D Ladar data will be combined with the INS data to help estimate and periodically reset the inertial drift errors.

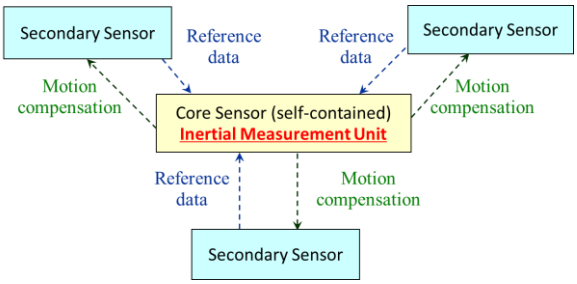

Figure 1. Integration philosophy with IMU as core sensor.

Although the framework described in this paper mainly focuses on Electro-Optical (EO) sensors (i.e. 2D imagery, Ladar data, 3D imagery, etc.), other sensor methods such as beacons or signals-of-opportunity may be included in future versions [3].

When using EO sensors as secondary sensors two main position estimation approaches exist: feature-based estimation and correlation-based (or map-based) estimation. In the feature-based approach one or more EO sensors is used to observe features in the environment. The changes in location and orientation of these features are then used to estimate the user position, velocity and attitude.

Table 1. Example features for the EO sensors.

	Measurements	Features
2D Vision	Intensity	Edges [12][13], corners [14][15], lines [16], planar surfaces, SIFT features [17], SURF features [18], etc.
2D Ladar	2D Point cloud	Lines [19], corners, point features
3D Imager	3D Point cloud, intensity	Planar surfaces [20], curved features, line features, point features.

3D imaging camera data has no depth uncertainty, but does typically not observe many strong features (i.e. planar surfaces). In contrary, 2D vision camera data typically contains more observable features than the 3D point cloud data, but these features do have an unknown depth. When observing an indoor or urban environment with 3D (point cloud data) and 2D (intensity data) sensors, various features can be utilized for navigation purposes.

The correlation-based approach uses one or more EO-sensor to observe the environment and form a database or map of that environment. This observed map (or parts of the map) is then correlated/compared to either an a priori map or a previously derived map to estimate the user position, velocity and attitude.

3. ARCHITECTURE

The current proposed architecture for the plug-and-play sensor integrator consists of five major components: the connection monitor, the calibration unit, the feature extraction unit, the PVA estimator and mapping unit, and the integrity monitor. The architecture is shown in Figure 2. The connection monitor detects sensor connect and disconnect events and keeps track of the current status of each of the devices in a Sensor Status Table (SST). In case of a connect event, the monitor determines if the sensor in question requires online calibration and, if so, dispatches the sensor device to the calibration unit for calibration. Examples of sensors requiring online calibration are the various EO sensors (i.e. 2D imaging sensors). In the event that no calibration is required or the calibration has been completed, a feature extraction unit is assigned to those sensors that require one, and the sensor is connected to the PVA estimator and mapping unit. In case of a disconnect event the device will be removed from the SST. The feature extraction unit extracts the necessary features from the sensors such as planar surfaces, lines and points from 2D imagery and Ladar.

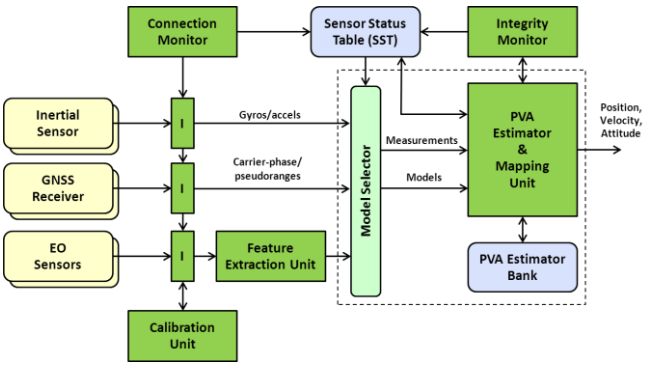

Figure 2. Reconfigurable navigation solution architecture.

The PVA estimator (and mapping) unit uses the Sensor Status Table to determine the structure of the PVA estimator that should be used for the current sensor configuration, and performs an observability analysis to determine if a PVA estimate can still be obtained with that configuration. This unit furthermore, performs a smooth transition from one filter to the next using a filter structure similar to an Interacting Multiple Modeling (IMM) filter. Currently, various estimator strategies are included in filter structure to support the various sensor configurations including complementary extended Kalman filters and Self-Localization and Mapping (SLAM) algorithms. Finally, the integrity monitor detects if any off-nominal condition does exist in the available sensor data and passes that information to the connection monitor, which can then remove it from the SST temporarily or permanently.

4. INCORPORATION IN ROS

The Robotic Operating System (ROS) is an open-source software framework intended for software development for robot applications. ROS provides "*hardware abstraction, device drivers, libraries, visualizers, message-passing, package management, and more* [21]." Not only does ROS give the user access to the standard libraries and tools, it also gives the user access to a large amount of third party open-source software developed by the ROS community. The availability of a wide variety of tools, allows for quick prototyping of various implementations. This software includes sensor interfaces with a wide variety of sensors including many EO sensors such as

cameras and laser range scanners. Table 2 shows that many of the components available in ROS can be exploited within proposed framework.

Table 2. ROS compatibility with framework components.

	ROS Standard & Third-party	Additional
Connection Monitor	Various sensor device drivers	Custom code and interface (see I)
Calibration Unit	OpenCV, image pipeline	Custom code and interface
Feature Extraction Unit	OpenCV, Point Cloud Library (PCL)	Custom code and interface
PVA Estimator	Navigation stacks (2D and 3D)	Custom code
Integrity Monitor	N/A	Custom code

In ROS the basic element of computation is referred to as a node. Nodes can communicate using a publisher/subscriber service by publishing messages to topics. In [21] topics are defined as a "*named bus over which nodes exchange messages.*" Messages are simple data structures that can be defined by the users in a manner similar to C structures using standard data types such as floating points, integers, characters, arrays of these data types and structures. Other methods of node communication exist such as client/server operation, remote procedure call services and the parameter server.

In Figure 2 all blocks will have an associated node. In case of the sensors, either the connection manager or the node itself will ensure that the output measurements adhere to a predefined interface format enabling sensors from different vendors to be interfaced with the other nodes in the framework. Message structures for the IMU, GNSS sensor, Ladar and 2D imagery have already been defined and are extensions of the basic ROS messages: Imu.msg, LaserScan.msg, PointCloud.msg and Image.msg. For the GNSS sensor a complete new message was introduced capturing data structures for all the raw data output by the GNSS receiver including carrier-phase measurements, pseudoranges, carrier-to-noise ratios, ephemerides, etc. An important addition to the message structure is a more detailed timestamp structure that includes local CPU-based timestamps, sensor timestamps, or other user-defined timestamps. Sensor specification will be made available to the other nodes via the parameter server. Important parameters include the sensor update rates and quality related parameters such as the gyro and accelerometer scale factors, bias stabilities, and random walk behaviour.

The connection monitor is an important node since it determines what sensors are available in the integration configuration. Even though the connection manager may use many of the available features of ROS such as the various available device drivers and their message structures, custom code will be developed to enable automatic node start-up when a sensor is connected to, for example, a USB port.

5. PVA ESTIMATOR

Figure 3 shows a detailed block diagram for the proposed PVA estimator and mapping unit. The top part, blocks (a) and (b), show the inertial mechanization. The shown mechanization implements the low-cost inertial mechanization described in detail in references [22] and [23]. The mechanization is split into two parts: the computation of the attitude and the computation of position and velocity in the navigation frame.

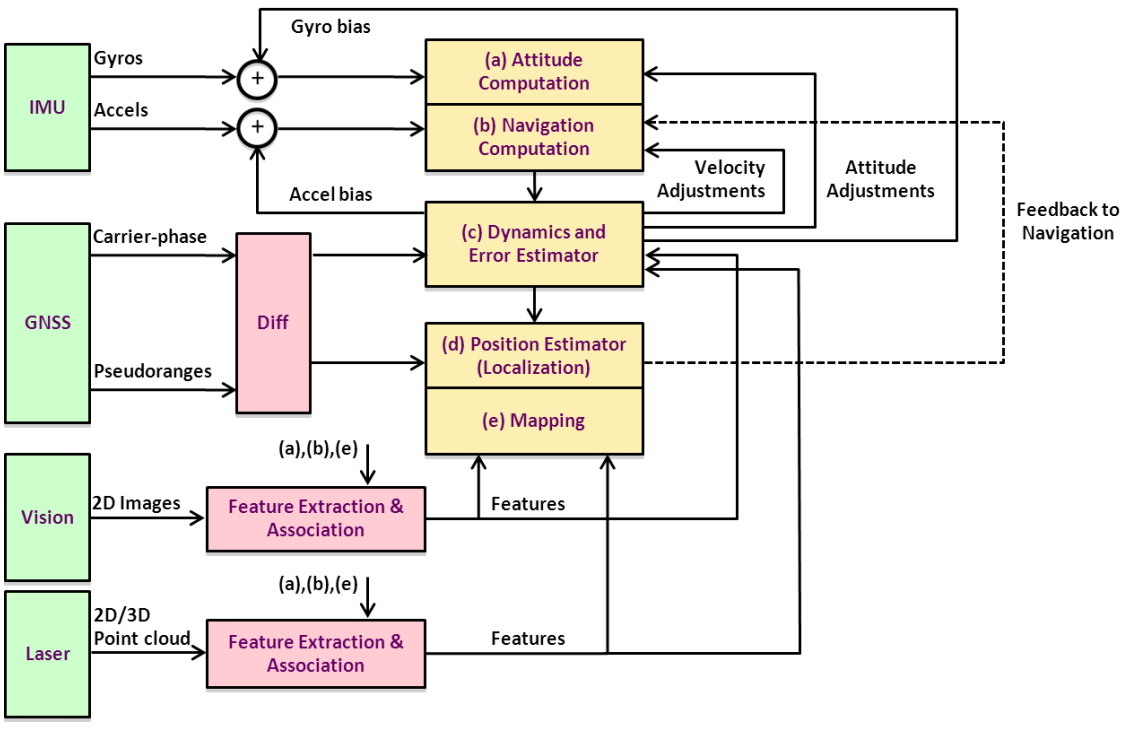

Figure 3. PVA estimator block diagram.

5.1 GNSS receiver as secondary sensor

In the presence of a GNSS receiver as a secondary sensor, a tight GNSS/Inertial integration strategy is chosen allowing estimation of inertial errors even in the presence of less than measurements from four satellites. The latter operational scenario must be expected in challenging environments like urban canyons. Various integration methods exist for GNSS and inertial sensors:

- GPS pseudoranges with inertial position
- GPS pseudoranges/carrier phases with inertial position
- GPS pseudoranges with inertial Δv, Δθ measurements
- GPS pseudoranges/carrier phases with inertial Δv, Δθ

The fourth approach has been implemented within our framework to estimate the inertial errors. The block diagram in Figure 2 shows a separated approach in which the dynamics and position components are separated in a manner similar to the one described in [22]. In that implementation the carrier-phase measurements are combined with the inertial measurements to obtain an estimate of the state vector:

$$\mathbf{x} = \begin{bmatrix} \delta\mathbf{v}^n & \delta\boldsymbol{\psi}_{nb} & \delta\boldsymbol{\omega}_{ib}^b & \delta\mathbf{f}_b \end{bmatrix}^T \tag{1}$$

where $\delta\mathbf{v}^n$ is the velocity error in the NED frame,
$\delta\boldsymbol{\psi}_{nb}$ is the is the miss-orientation attitude vector,
$\delta\boldsymbol{\omega}_{ib}^b$ is the gyro bias vector,
$\delta\mathbf{f}_b$ is the specific force bias error.

with corresponding continuous-time state transition matrix:

$$\mathbf{F} = \begin{bmatrix} \mathbf{0}_{3\times3} & -\left[\mathbf{f}^n\times\right] & \mathbf{0}_{3\times3} & -\mathbf{C}_b^n \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & -\mathbf{C}_b^n & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \tau_g^{-1}\mathbf{I}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \tau_a^{-1}\mathbf{I}_{3\times3} \end{bmatrix} \tag{2}$$

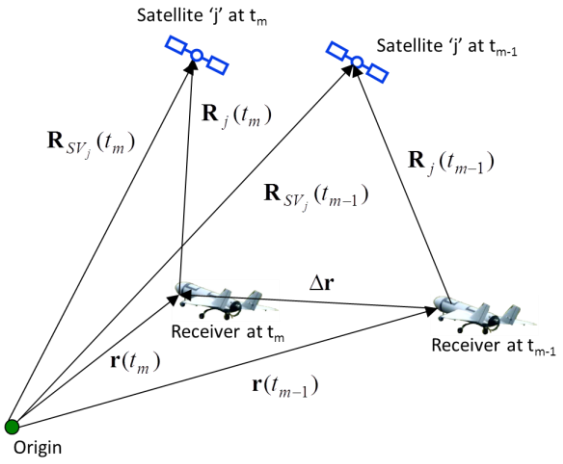

Figure 4. GNSS Sequential difference geometry.

An alternative implementation combines the position and dynamics filter by extending the error state with the position error, $\delta\mathbf{r}^n$:

$$\mathbf{x} = \begin{bmatrix} \delta\mathbf{r}^n & \delta\mathbf{v}^n & \delta\boldsymbol{\psi}_{nb} & \delta\boldsymbol{\omega}_{ib}^b & \delta\mathbf{f}_b \end{bmatrix}^T \tag{3}$$

Also, a clock drift term could be added. As shown in Figure 2, the error states are fed back to the attitude and navigation computation blocks as soon as they are available.

The dynamics filter in Figure 2 is set up as a complementary Kalman filter with a measurement vector consisting of:

$$z_{j,GNSS} = \mathbf{h}_j \Delta\mathbf{r}_{INS}^n - \Delta\phi_j + a_j + b_j + \delta t_{rcvr} \tag{4}$$

where $\Delta\phi_j$ is the sequential carrier-phase difference for satellite 'j', $\Delta\mathbf{r}_{INS}^n$ is the change in user position as computed by the inertial, $\mathbf{h}_j$ consists of the transpose of the line-of-sight vector to satellite 'j', $\mathbf{h}_j \equiv \mathbf{e}_j^T = \mathbf{R}_j^T / \left|\mathbf{R}_j\right|$, a_j, b_j are two compensation terms for geometry and Doppler change correspondingly [23][25], δt_{rcvr} is the sequential clock drift error.

Note that equation (2) can be extended to a difference-of-difference measurement by taking the difference of the sequential difference of satellite 'j' and the sequential difference of a key satellite 'k'. This new difference term will remove the receiver clock drift error from the state vector.

The measurement matrix, **H**, which relates the error state in (1) to equation (4) can be derived by expanding and evaluating the sensitivity of the inertial term in (4):

$$\mathbf{h}_j(t_m)\Delta\tilde{\mathbf{r}}_{INS} = \mathbf{h}_j(t_m)\int_{t_{m-1}}^{t_m} \tilde{\mathbf{v}}_e^n dt + \mathbf{h}_j(t_m)\left[\tilde{\mathbf{C}}_b^n(t_m) - \tilde{\mathbf{C}}_b^n(t_{m-1})\right]\mathbf{d} \tag{5}$$

In equation (4) the tilde '~' indicates parameters affected by the inertial errors. Evaluation of (5) results in the following measurement matrix:

$$\mathbf{H} = \begin{bmatrix} \mathbf{h}_1(t_m)\left[\mathbf{I}_{3x3} \mid \mathbf{0}_{3x12}\right]\left[\int_{t_{m-}}^{t_m} \boldsymbol{\Phi}(\tau, t_{m-1})d\tau\right]\boldsymbol{\Phi}^{-1}(t_m, t_{m-1}) \\ \vdots \\ \mathbf{h}_N(t_m)\left[\mathbf{I}_{3x3} \mid \mathbf{0}_{3x12}\right]\left[\int_{t_{m-}}^{t_m} \boldsymbol{\Phi}(\tau, t_{m-1})d\tau\right]\boldsymbol{\Phi}^{-1}(t_m, t_{m-1}) \end{bmatrix} + \begin{bmatrix} \mathbf{0}_{3x3} & \left[\delta\boldsymbol{\xi}\times\mathbf{h}_1^T(t_m)\right] & \left[(t_m - t_{m-1})\boldsymbol{\xi}(t_{m-1})\times\mathbf{h}_1^T(t_m)\right] & \mathbf{0}_{3x3} \\ \vdots & \vdots & \vdots & \vdots \\ \mathbf{0}_{3x3} & \left[\delta\boldsymbol{\xi}\times\mathbf{h}_N^T(t_m)\right] & \left[(t_m - t_{m-1})\boldsymbol{\xi}(t_{m-1})\times\mathbf{h}_N^T(t_m)\right] & \mathbf{0}_{3x3} \end{bmatrix} \tag{6}$$

where **d** is the lever arm between the IMU and the GNSS receiver. $\boldsymbol{\xi}(t_m) = \mathbf{C}_b^n(t_m)\mathbf{d}$, $\delta\boldsymbol{\xi} = \boldsymbol{\xi}(t_m) - \boldsymbol{\xi}(t_{m-1})$ and $\boldsymbol{\Phi}(t_2, t_1)$ is the discrete-time state transition matrix from time epoch t_1 to time epoch t_2 derived from **F** in (2)

5.2 Ladar or 3D imager receiver as secondary sensor

In case a 3D imaging (Ladar) sensor is available, features, such as planar surfaces, can be extracted from the point cloud data directly using, for example, the ROS PCL. In that case a change in position and attitude can be directly observed from the change in closest distance, ρ_i, from the imager origin to the planar surface and the normal vector, **n**, of that surface

expressed in the camera frame at time epochs t_m and t_{m-1} following:

$$\begin{bmatrix} \mathbf{n}_1^T \\ \vdots \\ \mathbf{n}_N^T \end{bmatrix} \Delta\mathbf{r} = \begin{bmatrix} \rho_1(t_m) - \rho_1(t_{m-1}) \\ \vdots \\ \rho_N(t_m) - \rho_N(t_{m-1}) \end{bmatrix} \Rightarrow \mathbf{H}\Delta\mathbf{r} = \Delta\boldsymbol{\rho}_L \quad (6)$$

More details and a derivation of the attitude estimator can be found in [27]. Often only a 2D Ladar (laser scanner) is used since it is more accurate, lighter and cost-effective. In that case 2D features such as lines can be extracted and mapped on 3D space [8]. However, estimation of position and attitude for more complex 3D motion will not be possible. An alternative has been proposed and evaluated successfully in [28]. The method in that paper exploits existing or intentional motion of the platform to increase the field-of-view of the Ladar resulting in point cloud measurements in 3D rather than 2D. Note that the introduced motion must be taken into account carefully.

When integrating the 3D features with the IMU measurements another complementary filter can be set up with the following measurement:

$$z_{j,Lp} = \Delta\rho_{j,L} - \Delta\rho_{j,INS} \quad (7a)$$

$$\mathbf{z}_{j,La} = \mathbf{n}_{j,L} - \mathbf{n}_{j,INS} \quad (7b)$$

where $\Delta\rho_{j,INS}$ is the change in closest distance derived from the inertial mechanization and $\mathbf{n}_{j,INS}$ is the normal computed from the previous Ladar output and the inertial position and attitude estimates:

$$\Delta\tilde{\rho}_{j,INS} = \tilde{\mathbf{n}}_j^T \tilde{\mathbf{C}}_n^b \Delta\mathbf{r}^n \quad (8a)$$

$$\tilde{\mathbf{n}}_{j,INS}(t_m) = \mathbf{C}_n^b(t_m)\mathbf{C}_b^n(t_{m-1})\tilde{\mathbf{n}}_{j,L}(t_{m-1}) \quad (8b)$$

The derivation of the measurement equation that relates equation (7a) and (7b) to the augmented state vector (3) given the erroneous parameters as defined in (8a) and (8b) can be found in [24]. One row of each of these **H** matrices is provided here:

$$\mathbf{h}_{j,Lp} = \begin{bmatrix} \mathbf{n}_j^T(t_{m-1})\mathbf{C}_n^b(t_{m-1}) & \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} \end{bmatrix}^T \quad (9a)$$

$$\mathbf{h}_{j,La} = \begin{bmatrix} \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} & \left(\mathbf{C}_n^b(t_m)\mathbf{C}_b^n(t_{m-1})\mathbf{n}_j(t_{m-1})\times\right)^T & \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} \end{bmatrix}^T \quad (9b)$$

Other features that have been considered are 3D points and 3D lines. For a detailed discussion and derivation on the measurement equations for those features the reader is referred to [24].

5.3 2D imager as secondary sensor

Unlike the GNSS- and Ladar-based integration approaches, no direct measurement can be derived for the 2D image features. Instead, a constraint-based (or implicit) complementary Kalman filter will be used. In this filter the input to the filter is a constraint equation rather than the difference between a measurement and a quantity synthesized using outputs from the inertial mechanization. While various features can be considered for 2D imagery, only point features are shortly considered here. For a discussion on other 2D imagery features the reader is referred to [24].

An example of such a constraint (the so-called epipolar constraint) can be derived given the geometry shown in Figure 5. The 3D point **p** is observed at two time epochs in the respective body frames of the camera resulting in two unit-length pointing vector $\mathbf{e}_I^b(t_{m-1})$ and $\mathbf{e}_I^b(t_m)$. Note that the actual vectors ($\mathbf{p}_I^b(t_{m-1})$ and $\mathbf{p}_I^b(t_m)$) are unknown due to the unknown length (the "depth" problem).

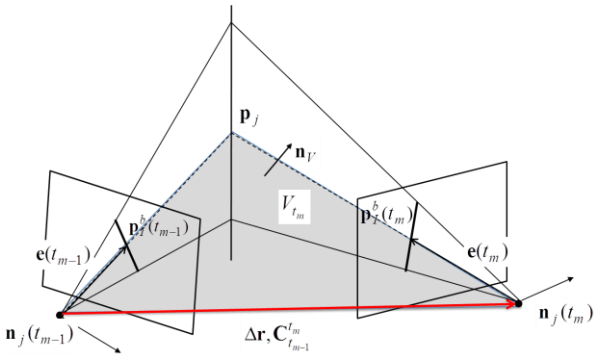

Figure 5. Point feature constraints.

Now the normal vector (expressed in the "navigation" frame) of the plane spanned by $\mathbf{e}_I^n(t_{m-1})$ and $\Delta\mathbf{r}^n$ is given by $\mathbf{n}_V = \mathbf{e}_I^n(t_{m-1}) \times \Delta\mathbf{r}^n$. As can be seen in Figure 5, this vector is perpendicular to $\mathbf{e}_I^n(t_m)$, resulting in the following constraint equation:

$$\begin{aligned} &\mathbf{e}_I^n(t_m) \cdot \left(\mathbf{e}_I^n(t_{m-1}) \times \Delta\mathbf{r}^n\right) = 0 \Rightarrow \\ &\left(\tilde{\mathbf{C}}_b^n(t_m)\mathbf{e}_I^b(t_m)\right) \cdot \left(\tilde{\mathbf{C}}_b^n(t_{m-1})\mathbf{e}_I^b(t_{m-1}) \times \Delta\tilde{\mathbf{r}}^n\right) = 0 \end{aligned} \quad (10)$$

Another constraint that can be observed from Figure 5 is given by:

$$\begin{aligned} &\mathbf{p}_I^n(t_m) = \Delta\mathbf{r}^n - \mathbf{p}_I^n(t_{m-1}) \Rightarrow \\ &\tilde{\mathbf{C}}_b^n(t_m)\mathbf{p}_I^b(t_m) - \Delta\tilde{\mathbf{r}}^n + \tilde{\mathbf{C}}_b^n(t_{m-1})\mathbf{p}_I^b(t_{m-1}) = 0 \end{aligned} \quad (11)$$

Note that the tilde is used in both equation (10) and equation (11) to indicate the use of an erroneous estimate for the INS mechanization. The use of erroneous inertial values (in addition to the noise that may be present on the derived features themselves), allows for the derivation of an appropriate H matrix. Because of the complexity of this matrix it is not given here, but can be found in [24].

5.4 Overall filter implementation

Sections 5.1, 5.2 and 5.3 show just a subset of complementary filters that can be used within our framework. It is important to note that the PVA estimator can still be configured in various ways. In one approach the **H** matrices derived from all available features are used in one big complementary Kalman filter. An alternative approach would be to use an IMM that has separate filters for each of the secondary sensors and combines the separate filters using a Markov model. A more detailed discussion of these various filters and filter configuration as

well as a completely difference approach using a Marginalized SLAM approach will be the focus of a future paper.

6. SIMULATION RESULTS

In order to test the effectiveness of parts of the PVA estimator, a simulation was set up. First, a 3D hallway environment was simulated containing observable static features including, points, corners, lines and planar surfaces. Next, a platform trajectory was defined through the constructed hallway. This artificial environment is illustrated from multiple angles in Figure 6.

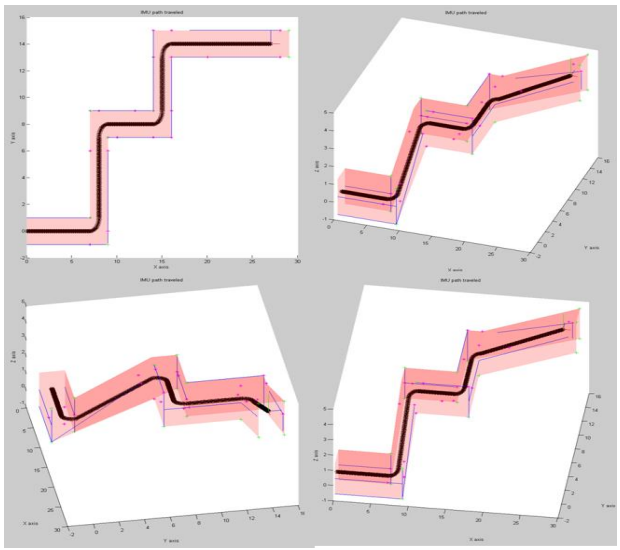

Figure 6. Simulation platform trajectories.

After the definition of the platform trajectory, raw IMU measurements were synthesized at an update rate of 100Hz following three types of IMU error models.

	Perfect IMU	Tactical-grade IMU	Commercial -grade IMU
Gyro bias	0 deg/hr	60 deg/hr	6000 deg/hr
Gyro noise	N(0,0) deg/hr	N(0,$(0.6)^2$) deg/hr	N(0,$(6)^2$) deg/hr
Accelerometer bias	0 mg	1 mg	10mg
Accelerometer noise	N(0,0) mg	N(0,$(0.1)^2$) mg	N(0,1) mg

The time elapsed from the beginning of the trajectory to the end was ten minutes. The 2D and 3D features were created assuming a sensor with a 500-by-500 pixel resolution, a 6mm focal length, and a Field-of-View (FoV) of 60 degrees in both directions. The error on these 2D and 3D features were modelled as Additive White Gaussian Noise (AWGN) models with varying mean and standard deviations.

For this simulation, GNSS measurements were considered completely unavailable. In other words, just the methods addressed in Section 5.2 and 5.3 were evaluated. Evaluation of the operation of the method proposed in Section 5.1 was performed earlier [22].

In the absence of GNSS or any EO sensor, the results are as expected for a free-running and drifting inertial as can be seen in Figure 7 for the tactical-grade IMU.

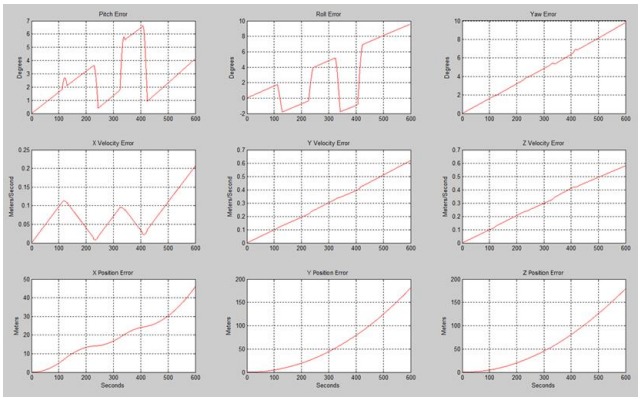

Figure 7. Results for an unaided tactical-grade INS.

The integration results for the tactical grade IMU integrated with and all available planar surfaces within the FoV are shown in Figure 8. Typical noise values were assigned to the centroid (1 cm, 1-σ) and its normal vector (1 mm, 1-σ).

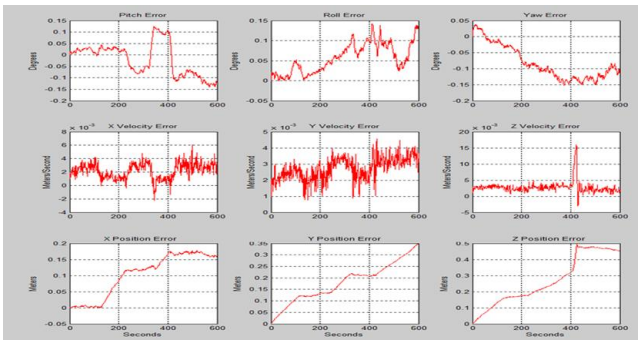

Figure 8. Tactical grade integrated with all available planar surfaces.

It can be observed that drift cannot be avoided, just mitigated in the absence of a priori information on the environment (i.e known location and orientation of planar surfaces).

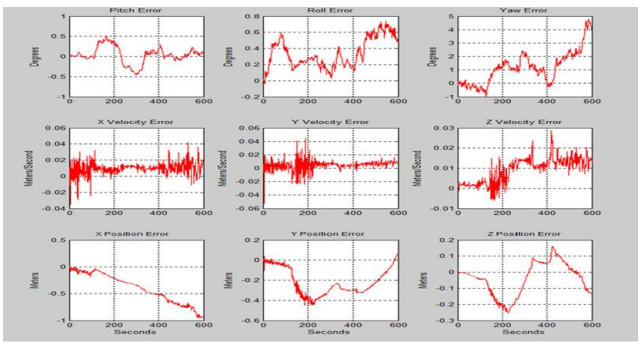

Figure 9. Tactical grade integrated with one planar surface and all available 2D point features.

Next, only one planar surface was observed, but also all available 2D point features. Noise on the point features was assumed to be 1 pixel. The results are shown in Figure 9. Again, remaining drift can be observed, but it is significantly reduced from the unaided tactical grade results. Under the limited FoV and motion, the algorithm would not be able to resolve the depth uncertainty of the point features, leading to divergence of the solution. However, even a single planar surface allows the algorithm to operate under unknown depth successfully. Again, *a priori* knowledge of the location of the 2D points in a 3D world would allow a result to an absolute position.

7. DESIGN AND IMPLEMENTATION DETAILS

The basic hardware architecture is shown in Figure 10. At the moment the supported plug-and-play sensors include a Novatel OEMSTAR single channel GPS/GLONASS receiver, an Xsense IMU, one or more PointGrey FireFly MV monochrome cameras, and a Hokuyo UTM-30LX laser scanner (Ladar). These sensors can be connected and disconnected via a Universal Serial Bus (USB) to an embedded microprocessor. At the moment this microprocessor is a pico-ITX Intel Atom Z530 running at 1.6GHz.

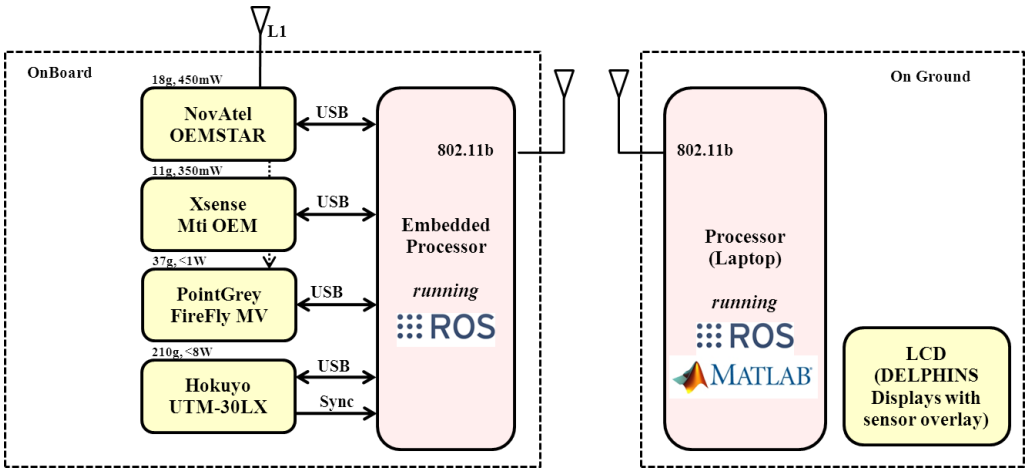

Figure 10. Hardware architecture.

Since it is the intention to fly this plug-and-play navigation payload on a four-rotor UAV such as the Pelican by Ascending Technologies (see Figure 11), the software will more than likely be ported to the processor on-board the Pelican platform. Fortunately, this platform will support both Linux and ROS.

Figure 11. Pelican four-rotor UAV platform.

To enable accurate time synchronization, the SyncOut of the Xsense, the PPS output (or MARK output) of the OEMSTAR, the synchronization output of the Hokuyo and the FireFly MV external trigger are all connected to a microcontroller which communicates with the embedded processor though a USB-RS232 connection.

A ground-system component is planned as well that allows a user to remotely observe the state of the UAV via a Primary Flight Display (PFD) and a Navigation Display (ND) as well as the observed imagery overlaid or as an inset of the PFD. For this development we are collaborating with Delft University of Technology who has developed these display concepts for commercial aircraft or remote piloting of UAVs in the national airspace.

With respect to the ROS software development, at the current time the implementation of the GNSS-node, the IMU-node and large parts of the 2D Vision, and Ladar nodes has been completed as well as large parts of the PVA estimator. Data has been collected on a ground platform using the GNSS, IMU and 2D vision nodes.

Figure 12. Ground-based data collection setup using the CTAE/ASCAMM Husky unmanned ground vehicle.

Figure 13. Preliminary data collected and mapping results using the Husky in the vicinity of the ASCAMM building.

8. SUMMARY AND CONCLUSIONS

This paper has introduced a plug-and-play architecture based on ROS that is capable of detecting the connected sensor configuration and configuring an appropriate PVA estimator. For a variety of sensors the state vector and measurement equations were derived and the combined complementary filter was evaluated in a simulation environment. The simulations have shown that the drift errors can be significantly reduced through integration with features derived from 2D imagery and Ladar even in the absence of GNSS.

Various ROS components and message structures were identified that will be used within the proposed framework and new ROS nodes were developed. Finally, the hardware platform with appropriate hardware sensor interfaces and timing circuitry was designed.

9. REFERENCES

[1] Soloviev, A., et al., "Assessment of GPS Signal Quality in Urban Environments Using Deeply Integrated GS/IMU," Proceedings of the National Technical Meeting 2007, San Diego, CA, January 2007.

[2] M. Uijt de Haag, D. Venable, M. Smearcheck, "Use of 3D laser radar for navigation of unmanned aerial and ground vehicles in urban and indoor environments," Proceedings of the SPIE - Volume 6550, SPIE Defense and Security Symposium, Orlando, FL, April 9- 13, 2007.

[3] M. M. Miller, M. Uijt de Haag, A. Soloviev, M. Veth, "Navigating in Difficult Environments: Alternatives to GPS – 1," Proceedings of the NATO RTO Lecture Series on "Low Cost Navigation Sensors and Integration Technology," SET-116, November 2008.

[4] M. M. Miller, J. Raquet, M. Uijt de Haag, "Navigating in Difficult Environments: Alternatives to GPS – 2," Proceedings of the NATO RTO Lecture Series on "Low Cost Navigation Sensors and Integration Technology," SET-116, November 2008.

[5] Borges, et al. "Optimal Robot Pose Estimation using Geometrical Maps," *IEEE Transactions on Robotics and Automation*, Vol. 18, No. 1, February 2002.

[6] Pfister, S. T., "Algorithms for Mobile Robot Localization and Mapping Incorporating Detailed Noise Modeling and Multi-scale Feature Extraction," Ph.D. Dissertation, California Institute of Technology, 2006.

[7] Bates, D., "Navigation Using Optical Tracking of Objects at Unknown Locations," M.S.E.E. Thesis, Ohio University, November 2006.

[8] Soloviev, A., D. Bates, and F. van Graas, "Tight Coupling of Laser Scanner and Inertial Measurements for a Fully Autonomous Relative Navigation Solution," *NAVIGATION, Journal of the Institute of Navigation*, Vol. 54, No. 3, Fall 2007, pp. 189-205.

[9] Horn, J. P., "Bahnführung eines mobilen Roboters mittels absoluter Lagebestimmung durch Fusion von Entfernungsbild- und Koppelnavigations-daten," Ph.D. Dissertation, Technical University of Munich, 1997.

[10] Campbell, J. L. et al., "Flash-LADAR Inertial Navigator Aiding," IEEE/ION Position Location and Navigation Symposium, April 2006.

[11] Soloviev, N. Gans, M. Uijt de Haag, "Integration of Video Camera with 2D Laser Scanner for 3D Navigation," Proceedings of the 2009 International Technical Meeting of the Institute of Navigation, January 26 - 28, 2009, Anaheim, CA, pp. 767 – 776.

[12] Canny, John, "A Computational Approach to Edge Detection," *IEEE Transactions on Pattern Analysis ad Machine Intelligence*, Viol. PAMI-8, No. 6, November 1986.

[13] Marr, D., E. Hildreth, "Theory of Edge Detection," Proceedings of the Royal Society of London Series B., Biological Sciences, Vol. 207, No. 1167, pp. 187-217.

[14] Harris, C. and M.J. Stephens. A combined corner and edge detector. In Alvey Vision Conference, pages 147–152, 1988.

[15] Rosten, E., R. Porter and T. Drummond, "Faster and Better: A Machine Learning Approach to Corner Detection," *IEEE Transactions on Pattern Analysis and Machine Analysis*, Vol. 32, No. 1, January 2010.

[16] Hough, P. V. C., "Machine Analysis of Bubble Chamber Pictures," Proceedings of the International Conference on High Energy Accelerators and Instrumentation, 1959.

[17] Lowe, D. "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, pp. 91–110, 2004.

[18] Bay, H., T. Tuytelaars, L. van Gool, "Surf: Speeded Up Robust Features," Proceedings of the ECCV, 2006.

[19] Nguyen, V. et al., "A Comparison of Line Extraction Algorithms using 2D Laser Rangefinder for Indoor Mobile Robotics, Proceedings of the Conference on Intelligent Robot and Systems, IROS 2005, Edmonton, Canada.

[20] Venable, D., "Implementation of a 3D Imaging Sensor Aided Inertial Measurement Unit Navigation System," M.S.E.E. Thesis, Ohio University, 2008.

[21] www.ros.org, Robotic operating Systems (ROS) website, accessed January 2 2012.

[22] Farrell, J. L., *GNSS Aided Navigation & Tracking – Inertially Augmented or Autonomous*, American Literary Press, 2007.

[23] Uijt de Haag, M., 2009. Inertial Navigation Course Notes, Ohio University, Athens, Ohio.

[24] Dill, E., "Integration of 3D and 2D Imaging Data for Assured Navigation in Unknown Environments," M.S.E.E. Thesis, March 2011.

[25] Van Graas, F. and A. Soloviev, "Precise Velocity Estimation Using a Stand-Alone GPS Receiver," NAVIGATION, Vol. 51, No. 4.

[26] Angrisano, A. (GNSS/INS Integration Methods. PhD Thesis, Dipartimento di Scienze Applicate, Universita' Degli Studi Di Napoli "Parethenope." 2000.

[27] Uijt de Haag, M., D. Venable, and M. Smearcheck "Integration of an Inertial Measurement Unit and 3D Imaging Sensor for Urban and Indoor Navigation of Unmanned Vehicles," in Proceedings of the Institute of Navigation National Technical Meeting 2007, Jan. 2007, pp. 829-840.

[28] Soloviev, A. and M. Uijt de Haag, "Three-Dimensional Navigation of Autonomous Vehicles Using Scanning Laser Radars: Concept and Initial Verification," IEEE Transactions on Aerospace and Electronic Systems, Vol. 46, Issue 1, 2010.