Mosaic Aided Navigation: Tools, Methods and Results

Vadim Indelman, Pini Gurfil, Ehud Rivlin and Hector Rotstein

***Abstract*—The on-line construction of an image mosaic or panorama can be exploited to aid the navigation system of an airborne platform. The purpose of the paper is threefold. First, the paper presents some of the tools required for computing a mosaic and using the information collected as a side product within a navigation filter. These tools include a special variation of the Kalman filter and a new formulation of the tri-focal tensor from multi-frame vision. Second, the paper summarizes a general method for fusing the motion information obtained during the mosaicking process and also shows how "loop-closure" can be used to preserve navigation errors at their initial levels. Third, the paper discusses a number of illustrative examples to show how mosaic aiding can indeed result in a substantial improvement of the navigation solution.**

I. Introduction

Image mosaicking refers to the process by which a number of image frames taken by one or more cameras are combined to form a single image of a relatively large scene. Mosaicking is nowadays common in applications ranging from panoramic home photos to high-end aerial imagery, and assumes either implicitly or explicitly, that the motion of the imaging system in between frames is known to some precision. In the case of home panoramics, the motion between frames is assumed to be a pure rotation and the panorama is formed by looking for overlapping areas in between two frames. The rotation assumption guarantees no or negligible perspective distortion and hence the frames can be pasted seamless without transformations. In the case of aerial imagery from a flying platform, the problem is much more complex since the images depend also on the 3D structure of the scene and on the location and angular orientation of the platform, in addition to the earth roughly spherical shape. Hence aerial imagery usually assumes that the navigation solution of the platform is known up to the minimum accuracy to form a mosaic that is consistent in some sense. Motivated by the proliferation of mosaicking algorithms and the fact that a mosaic results from the platform or the sensor motion, one is led to consider the inverse problem relevant for navigation: is it possible to estimate or provide an update for the motion of a platform based on a mosaic? Several interesting practical problems would naturally benefit from such an approach:

1) Go Back Home. The navigation and/or guidance systems of many modern UAVs' rely on GPS for positioning. In case of a GPS outage, the algorithm switches back to survival variation of a vertical gyro to keep the platform on the air, while turning in circles at constant altitude or been flown back manually. Intuitively, if a mosaic was built while GPS is available, then this mosaic can be used as a map to fly the UAV automatically, and even return back to base in case of an outage.
2) Loitering. In many applications a flying platform must get to a location of interest and then fly around more or less in circles while performing surveillance or other similar mission. In this case, a mosaic may be generated when the platform arrives at the loitering location, and then used to keep the platform on a specified trajectory using the mosaic as a form of geo-referencing. Notice that the absolute location with respect to the earth may not be exact but a similar level of navigation error could in principle be maintained during the whole mission.
3) Information Sharing. When more than one air vehicle are involved in a cooperative mission, and all or some of the platforms are equiped with cameras, then a mosaic may in principle be built by combining the image frames from all the vehicles together with the respective navigation solution. Notice that the consistent combination of all the information sources is not trivial and may require substantial computational effort.

Mosaic assisted navigation requires several building blocks. It is assumed that a flying platform is provided with an inertial measurement unit (IMU) and a processor for computing a navigation solution. The platform has also a GPS receiver and antenna for computing position and velocity, and a camera or similar electro-optical device for acquiring images of the ground overflown by the platform. It is worth stressing that this is the setup of most autonomous and semi-autonomous platforms today. Given this hardware, considerable research efforts have been placed during the last decated into the development of algorithms for aiding the inertial solution from the IMU with GPS and visual data. A typical Vision Assisted Navigation (VAN) algorithm uses the information extracted from an image registration process, along with the information available from other sensors, for estimating the platform's states and possibly additional navigation parameters. For example, Ref. [1] proposed integrating the velocity-to-height vision-based estimation with additional on-board sensors; Ref. [2] applied the subspace-constraint approach [3] to partially estimate an airborne platform's states based on measurements from an image registration process injected into an implicit extended Kalman filter (EKF); and Ref. [4] utilized epipolar constraints formulated for each pair of matching

V. Indelman and P. Gurfil are with the Faculty of Aerospace Engineering, Technion - Israel Institute of Technology, Haifa 32000, Israel (e-mail: ivadim@tx.technion.ac.il, pgurfil@technion.ac.il).

E. Rivlin is with the Department of Computer Science, Technion - Israel Institute of Technology, Haifa 32000, Israel (e-mail: ehudr@cs.technion.ac.il).

H. Rotstein is with RAFAEL - Advanced Defense Systems Limited, Israel (e-mail: hector@rafael.co.il).

features to aid the inertial navigation of a ground vehicle. All the preceding methods rely only on information from inertial navigation sensors and an on-board camera, without assuming any *a-priori* available information or additional external sensors. This is also the approach adopted in the current work.

Alternatively, ref. [5] assumed that altimeter measurements are used for scaling the imaging sensors in order to improve state estimation during the landing phase of a space probe. Refs. [6] and [7] showed that absolute pose and motion estimation is possible when assuming that a DTM is available. Another fusion approach, more closely connected with this work, is that of map-based navigation, which assumes that a map of the operational area is given and that the vehicle navigates by fusing inertial measurements, images of the environment and a map [8], [9], [10]. For example, Refs. [9], [10] proposed vision-based navigation for an unmanned underwater vehicle relying on a previously-constructed mosaic image of the ocean floor.

This paper describes tools and methods for vision-aided navigation of an airborne platform utilizing the information contained in the *on-line* construction of a mosaic from images that are captured by an on-board camera. It is assumed that the airborne platform is equipped with a standard, possibly low-quality, inertial navigation system and a camera mounted on gimbals. No additional external information source is assumed to be available, including no *a-priori* additional information except, perhaps, for the initial alignment of the navigation system. In particular, the mosaic image that represents the map of the overflown region has to be constructed during the flight and its accuracy will be relative to the accuracy of the original navigation solution. If an additional source of information, e. g. GPS, is partially available, then the information of this sensor can be used to provide an absolute geo-reference for the mosaic.

In addition to the references mentioned above, Mosaic Aided Navigation can be compared with Simultaneous Localization and Mapping (SLAM) [11]-[16], in which the construction of the observed environment representation and the estimation of the navigation solution are performed simultaneously. The standard approach to SLAM consists of formulating a joint-estimation problem on the navigation errors and a description of the environmnet like the location of landmarks or occupation cells. When processing a measurement, e.g., the new frame acquired by a camera, the navigation solution like position and velocity are updated together with the description of the environment. SLAM has generated some remarkable results, and for instance it has been shown that under some restrictive assumptions, the estimated representation of the environment converges to the true one asymptotically. Nevertheless, SLAM has some drawbacks, and among them the fact that when the complexity of the scene grows, so does the representation and hence the computational load associated with solving the problem. The current work uses a mosaic as a representation for the environment, but this representation is not included with the motion parameters in the overall estimation scheme. This approach was chosen in order to mitigate one of the major deficiencies of the SLAM approach - the increasing computational load resulting from the augmentation of the state vector with the environment representation.

This work suggests a variation of the SLAM framework for coping with the aforementioned challenge: Separating the environment recovery from motion estimation. In order to do this, two types of mosaics are constructed: a *temporary* or *local* mosaic and a *main* or *global* mosaic. The parameter estimation is based on the first type of mosaic image, which is a small temporary mosaic constructed based on recent incoming camera-captured images. Once the temporary mosaic image reaches a certain size, its contents are removed and used for updating the main mosaic image in a background process. The advantage of this architecture is the low computational load required for parameter estimation, since it involves processing a constant-size state vector and only a portion of the temporary mosaic image. The main mosaic image, on the other hand, may be updated at a lower frequency using various algorithms [17]-[29] depending on the available computational resources.

Images registration and image-based motion estimation are important constituents in all VAN methods. The existence of overlapping regions between processed images is the common assumption to all vision-based motion estimation techniques. A large overlapping region between two images should allow a more accurate motion estimation relying on two-view geometry methods. If a mutual overlapping region for more than two images can be found, the performance may be further enhanced by applying multi-view-geometry-based methods.

The two-view-geometry-based methods include, for example, Ref. [30], in which the relative motion between two given views is extracted from an estimated essential matrix [31]. The motion parameters are then used for estimating the state vector, which is an augmented vector comprised of the vehicle's current pose and past poses for each captured image. When the observed scene is planar, the motion parameters can be calculated by estimating the homography matrix [32], [10], [33], [35]. Having in mind the requirements for real-time performance and a low computational load, in this work the estimated camera motion is related to a constant-size state vector comprised of the vehicle's current parameters only (in contrast to Ref. [30]).

The multi-view-geometry-based methods, in contrast to two-view-geometry, use connections among several images, assuming that a common overlapping region exists. This results in an increased observability and better estimation of the platform states. However, assuming that an overlapping region among several images exists may be invalid in many airborne applications. Violating this assumption usually degenerates the multi-view methods into the two-view methods discussed above.

Yet, when the platform trajectory contains loops, or alternatively, in the case of several platforms that share information among themselves, several images are likely to be available with a mutual overlapping area. Consequently, motion estimation based on multiple-view geometry, and in particular three-view geometry, may be applied. Indeed, several multi-view methods for navigation aiding have been already proposed [36], [38]. In Ref. [36], features that are observed within multiple images and the platform pose at the relevant time

instances are related using an augmented state vector. The state vector contains the current platform pose and the platform pose for each previously-captured image that has at least one feature that appears in the current image. Once a certain feature, observed in the previous images, is no longer present in the currently-captured image, all the stored information for this feature is used for estimating the platform parameters, and the pose entries that belong to these past images are casted aside from the state vector. However, should the same feature be re-observed at some later time instant, such as in a loop scenario, the method will not be able to use the data for the feature's first appearance. It was later proposed [37] to cope with loops using bundle adjustment [31]. However, this process involves processing *all* the images that are part of the loop sequence, and therefore real-time performance is hardly possible. Furthermore, the method contains an intermediate phase of structure reconstruction.

The proposed approach herein for fusing information based on at least three overlapping images relies on the same architecture of decoupling the parameter estimation from mosaicking mentioned above, utilizing a constant-size state vector. Constraints based on a geometry of a general scene observed by three views are formulated, and then fused with an inertial navigation system using an Implicit Extended Kalman Filter, allowing estimation of the position state, thereby reducing the position errors to the levels present while the first two images were obtained. Thus, loop scenarios are handled by processing only three images, thereby requiring a reduced computational load compared to the available state-of-the-art techniques for handling loop scenarios, which either use an augmented state (SLAM methods) or require processing all the image chain (bundle adjustment methods), rendering real-time performance impractical.

The developed constraints and the trifocal tensor [31] are both constituted assuming a general three-view geometry. However, while the trifocal tensor utilizes only features that are observed from all the three images, the developed constraints may also be partially applied using features that are observed in each pair of images of the given three images. It should be noted that the trifocal tensor has been suggested for camera motion estimation [39], [40], and for localization of a robot and observed landmarks while performing a planar motion [41]. However, the trifocal tensor and in particular the constraints developed herein, have not been proposed so far for navigation aiding.

The camera field-of-view (FOV) is another important factor in the context of motion estimation. A special attention is given in this work to allow improved motion estimation when using narrow-FOV cameras, since many modern aerial platforms are equipped with narrow-FOV cameras of up to a few degrees to obtain large-zoom images with high resolution and feasible requirements on computational power. In the current work a method is proposed for increasing the overlapping regions between the images, by coupling the camera scanning and the online mosaic construction procedures.

II. Preliminaries

Throughout this paper, the following coordinate systems are used:

- E - Earth-fixed reference frame, also known as an Earth-centered, Earth-fixed (ECEF) coordinate system. Its origin is set at the center of the Earth, the Z_E axis coincides with the axis of rotation, X_E goes through the point latitude 0^o, longitude 0^o, and Y_E completes a Cartesian right-hand system.
- L - Local-level, local-north (LLLN) reference frame, also known as a north-east-down (NED) coordinate system. Its origin is set at the platform's center-of-mass. X_L points north, Y_L points east and Z_L is aligned with the plumb-bob vertical to complete a Cartesian right-hand system.
- B - Body-fixed reference frame. Its origin is set at the platform's center-of-mass. X_B points towards the nose tip, Y_B points toward the right wing and Z_B completes the setup to yield a Cartesian right-hand system.
- C - Camera-fixed reference frame. Its origin is set at the camera center-of-projection. X_C points toward the FOV center, Y_C points toward the right half of the FOV and Z_C completes the setup to yield a Cartesian right-hand system.

Notice that the camera is mounted on gimbals and performs pan and tilt movements with respect to the platform; the yaw and pitch angles between B and C are denoted by ψ_C and θ_C, respectively.

III. Mosaic-Aided Navigation Tools

Figure 1 shows the main components of the architecture under consideration. The specific system assumed in this work is an airborne platform equipped with a gimballed camera and an inertial navigation system (INS). Throughout this work, a narrow-FOV camera is assumed, since it is more realistic than wide-FOV camera for many cases of practical interest. As mentioned in the Introduction, a narrow-field makes the VAN and the image-based motion estimation problems more challenging. However, the proposed method is not restricted to cameras with narrow FOV, and is valid for other cameras as well. In addition, it is assumed that the observed ground area is sufficiently close to planar, or alternatively, that the flight altitude above ground level is high relative to ground changes in elevation[1].

The INS consists of an inertial measurement unit (IMU), a strapdown algorithm and a navigation Kalman Filter. The strapdown algorithm integrates the accelerations and angular rates (or rather, the velocity and angular increments) from the IMU to produce a navigation solution, which is comprised of platform position, velocity and attitude. Due to the unavoidable errors of the IMU sensors, the computed navigation parameters develop errors which increase unboundedly over time. It is well-known that for relatively low-grade inertial sensors, errors

[1]This assumption is made due to the construction process of the mosaic image, which is based on the homography transformation. However, the proposed approach for fusing image-based motion estimations and navigation data may be also applied without constructing a mosaic image, in which case non-planar scenes can be handled as well [45].

grow proportionally to time cubed, and hence an uncompensated inertial solution becomes useless in a relatively short period of time.

During the flight, an on-board camera captures images of ground regions according to a scanning procedure. The acquired images are directed to the image processing module that is accountable for mosaic image construction and for relative motion estimation. While all the images are used for updating the mosaic image, the motion estimation is performed at a lower frequency, utilizing only some of the images.

The mosaic construction is coupled with the camera scanning procedure, and is processed in two phases: 1) The camera-captured images are used for constructing a small temporary mosaic image. This temporary mosaic image is used for motion estimation at appropriate time instances. 2) After each motion estimation event, the temporary mosaic image is emptied and initialized to the most recent camera-captured image, while the removed images from the temporary mosaic image are used to update the main mosaic image in a background process.

The image-based motion estimation is reformulated into measurements, which are then injected into the navigation Kalman Filter in order to update the navigation system and thereby arrest the development of inertial navigation errors. In this way, the platform can navigate for long periods of time even with low-grade sensors.

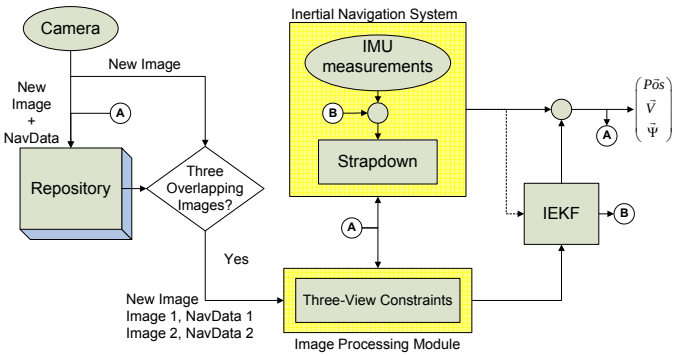


Fig. 1. Overview of the system concept.

A. Camera Scanning Procedure and Mosaic Construction Method

1) Scanning Procedure: During flight, the onboard camera captures images of the ground according to commands either from a human operator, an autonomous tracking algorithm of some features on the ground, or a scanning procedure. When captured, each new frame is processed and used to update the mosaic image of the flight area.

The scanning procedure is designed such that each image partially overlaps the preceding image as well as images from the previous scan stripe. The existence of overlapping regions is essential for performing image matching between captured images. In addition, and as opposed to most motion-from-structure methods, the additional overlapping region, provided by the camera scanning procedure, enables enhancement of motion estimation, as demonstrated in Ref. [44]. The proposed scan pattern also allows implementation of improved mosaic construction methods.

We assume that the scanning procedure modifies the pan angle of the camera, ψ_c, while keeping the camera tilt angle constant. Given camera angles at the current time instant, the calculation of camera angles for the next time instant is performed in two steps. First, the line-of-sight (LOS) vector for the next camera aiming point in the body-fixed reference frame, $\hat{\mathbf{r}}^B$, is determined according to

$$\hat{\mathbf{r}}^B = T_B^C(\psi_c)\frac{[f,\ d\cdot CCD_{\mathbf{Y}_C}/2,\ 0]^T}{\left\|[f,\ d\cdot CCD_{\mathbf{Y}_C}/2,\ 0]^T\right\|} \tag{1}$$

where $T_B^C(\psi_c)$ is the directional cosines matrix (DCM) transforming from the camera reference frame to the body frame, computed based on current camera angles; f is the camera focal length; d is the scan direction, so that $d = 1$ for increasing the camera pan angle and $d = -1$ for decreasing the camera pan angle; and $CCD_{\mathbf{Y}_C}$ is the size of the camera charged coupled device (CCD) in pixels along the $\mathbf{Y}_C$ axis.

The next step is to compute the new camera angles from $\hat{\mathbf{r}}^B$. The DCM transforming from B to C can be written as

$$T_C^B(\psi_c) = \begin{bmatrix} 0 & \sin\psi_c & \cos\psi_c \\ 0 & \cos\psi_c & -\sin\psi_c \\ -1 & 0 & 0 \end{bmatrix} \tag{2}$$

Since the aiming point vector in C is, by definition, $[1\ \ 0\ \ 0]^T$, one can write

$$\hat{\mathbf{r}}^B = T_B^C(\psi_c)\begin{bmatrix}1 & 0 & 0\end{bmatrix}^T = \begin{bmatrix}0 & \sin\psi_c & \cos\psi_c\end{bmatrix}^T \tag{3}$$

hence

$$\psi_c = \tan^{-1}\left[\frac{\hat{\mathbf{r}}^B(2)}{\hat{\mathbf{r}}^B(3)}\right] \tag{4}$$

The scanning direction, d, is switched once the camera pan angle, ψ_c, reaches a certain pre-specified level; this level is constrained by the correspnding gimbal limit but may be smaller than this mechanical limit.

For simplicity, it is assumed implicitly that the velocity-over-height ratio and the camera sampling frequency provide sufficient overlapping regions between each two adjacent images along the flight direction. Thus, the proposed scan methodology moves the camera only in a direction perpendicular to the flight trajectory. Notice that in a practical application this imposes a complex trade-off among flying altitude over ground, platform speed, FOV, scanning slant angle and resolution. However, the method presented here may be adjusted to work with a relaxed version of the assumption. Note also that no additional or *a-priori* information is required. An example of real mosaic images, constructed based on images acquired during the camera scanning procedure, as well as full details of the implemented scanning procedure, are given in Ref. [44].

2) Mosaic Construction Method: The construction of a mosaic from a series of single frames has several advantages from an application viewpoint. A mosaic is capable of showing the whole flight region in a single image, a feature that

constitutes an important aid to surveillance, communication and mission operation. Moreover, as shown in Ref. [44], the mosaic image and its construction process can be utilized for enhancing the accuracy of image-based relative motion estimation. We shall now present a partial description of the mosaic construction method using the camera scanning method described in Section III-A1.

The mosaic construction process and the camera scanning procedure are coupled. During the scan, images are captured using varying camera angles. While all the images contribute to the construction of a mosaic, only images taken while the camera was pointing downwards are used for motion estimation. These images are referred to as *downward-looking images*.

The mosaic construction process proceeds as follows. Two mosaic representations are constructed in the proposed approach: a temporary mosaic image that is used for motion estimation, and the main mosaic image which is the final mosaic image constructed based on the captured images.

The temporary mosaic image is initialized to a downward-looking image, once such an image is captured, and is updated with new non-downward-looking images. When a new downward-looking image is captured, it is matched to a relevant region in the temporary mosaic image, which is calculated utilizing information from the navigation system. Next, motion estimation can be performed, as discussed in Ref. [44].

The temporary mosaic image is expressed in the preceding downward-looking image system, defined as the coordinate system C of the previous downward-looking image. Therefore, the estimated motion describes the relative motion performed by the camera between two adjacent downward-looking images. This estimation will be used to correct developing inertial navigation errors.

Due to the coupling between the scanning procedure and the mosaic construction process, an enlarged overlapping area between the new downward-looking image and the temporary mosaic image is achieved. This, and the quality of the constructed temporary mosaic image, are the two factors that allow better motion estimation in certain scenarios, as demonstrated in Ref. [44].

After motion estimation is performed, the temporary mosaic image is reset and initialized to the new downward-looking image. The images that were removed from the temporary mosaic image are then used for updating the main mosaic image. Since the main mosaic image is not used for motion estimation, it may be updated in a background process. This may be performed by applying various algorithms [17]-[29], [34], depending on the available computational resources.

An example of the mosaic image construction process, based on real images acquired using the scanning procedure described above from Google Earth, is given in Ref. [44].

It should be noted that loop scenarios may be also handled in this background process yielding an improved main mosaic image. In case of a loop, motion estimation and navigation aiding are performed based on the temporary mosaic image, following the method suggested herein. However, a different approach is required for utilizing the full potential of the available information in such an event (e. g. three overlapping images), as discussed in Sections III-D and IV.

B. Multi-Point Kalman Filter

The problem considered in this note is the following. Given is the system:

$$x(t+1) = \Phi(t,t+1)x(t) + w(t), \tag{5}$$

where $\Phi \in I\!R^{n\times n}$, and $w(t)$ is a Gaussian white noise process with $Ew(t) = 0$ and $Ew(t)w(t)^T = Q(t)$. What makes the system "interesting" is the fact that the measurements at time t can be modeled by:

$$\begin{aligned} z(t) = H_2(t)x(t-2) + G_2(t)\nu(t-2) + \\ +H_1(t)x(t-1) + G_1(t)\nu(t-1) + \\ +H_0(t)x(t) + G_0(t)\nu(t), \end{aligned} \tag{6}$$

where $H_i(t)$ and $G_i(t)$ are time dependent measurement and measurement-noise matrices, and $\nu(t)$ is a Gaussian white noise process with expected value $E\nu(t) = 0$ and covariance $E\nu(t)\nu(t)^T = R(t)$. For ease of notation, it will be assumed here that $Q(t) = Q$ and $R(t) = R$. The problem is to design an optimal estimator $x(t/t)$ for the state $x(t)$ using all the measurements available up to time t. The error between the states and its estimate is denoted by:

$$\widetilde{x}\,(t/\tau) = x(t) - x(t,\tau), \quad \tau \le t. \tag{7}$$

The estimator is to be unscented, so that $E\widetilde{x}\,(t/\tau) = 0$. The covariance of this estimation error is $P(t/\tau)$.

1) Back to the KF: In order to find a solution to the estimation problem, we will try to re-write it using the Kalman Filtering formalism. To do that, define:

$$x_a(t) = \begin{bmatrix} x(t-2) \\ x(t-1) \\ x(t) \end{bmatrix} \tag{8}$$

and

$$\Phi_a(t,t+1) = \begin{bmatrix} 0 & I & 0 \\ 0 & 0 & I \\ 0 & 0 & \Phi(t,t+1) \end{bmatrix} \tag{9}$$

$$w_a(t) = \begin{bmatrix} 0 \\ 0 \\ w(t) \end{bmatrix}. \tag{10}$$

Then, from (5),

$$x_a(t+1) = \Phi_a(t,t+1)x_a(t) + w_a(t). \tag{11}$$

Likewise, taking

$$H_a(t) = \begin{bmatrix} H_2(t) & H_1(t) & H_0(t) \end{bmatrix}, \tag{12}$$

$$G_a(t) = \begin{bmatrix} G_2(t) & G_1(t) & G_0(t) \end{bmatrix}, \tag{13}$$

$$\nu_a = \begin{bmatrix} \nu(t-2) \\ \nu(t-1) \\ \nu(t) \end{bmatrix} \tag{14}$$

one can write:

$$z(t) = H_a(t)x_a(t) + G_a(t)\nu_a(t). \tag{15}$$

Notice that:

$$Q_a = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & Q \end{bmatrix},$$
$$R_a = \begin{bmatrix} R & 0 & 0 \\ 0 & R & 0 \\ 0 & 0 & R \end{bmatrix}.$$

2) Propagation Stage: Define the propagation stage:

$$x_a(t+1/t) = \Phi_a(t,t+1)x_a(t/t) \tag{16}$$

Then:

$$\widetilde{x}\,(t+1/t) = \Phi_a(t,t+1)\widetilde{x}\,(t/t) + w_a(t) \tag{17}$$

and

$$P(t+1/t) = \Phi_a(t,t+1)P(t/t)\Phi_a(t,t+1)^T + Q_a. \tag{18}$$

3) A-Posteriori Estimate: The correction stage is substantially more complicated than the propagation stage discussed above. Using the KF formalism, take:

$$\begin{aligned} x_a(t/t) &= x(t/t-1) + K_a(t)\left[z(t) - \right. \\ &\quad \left. - H_a(t)x(t/t-1)\right] = \\ &= x(t/t-1) + K_a(t)\left[H_a(t)\widetilde{x}\,(t/t-1) + \right. \\ &\quad \left. + G_a(t)\nu_a(t)\right], \end{aligned}$$

so that

$$\begin{aligned} \widetilde{x}\,(t/t) &= \left[I - K_a(t)H_a(t)\right]\widetilde{x}\,(t/t-1) \\ &\quad - K_a(t)G_a(t)\nu_a(t). \end{aligned} \tag{19}$$

The objective is now to compute $K_a(t)$ in an optimal manner, namely, so that it minimizes the trace of the covariance $P_a(t/t)$. From (19):

$$\begin{aligned} P_a(t/t) &= \left[I - K_a(t)H_a(t)\right]P_a(t/t-1)\cdot \\ &\quad \cdot\left[I - K_a(t)H_a(t)\right]^T + K_a(t)G_a(t)R_aG_a(t)^TK_a(t)^T - \\ &\quad - \left[I - K_a(t)H_a(t)\right]E\left[\widetilde{x}\,(t/t-1)\,\nu_a(t)^T\right]\cdot \\ &\quad \cdot G_a(t)^TK_a(t)^T - K_a(t)G_a(t)\cdot \\ &\quad \cdot E\left[\nu_a(t)\widetilde{x}\,(t/t-1)^T\right]\left[I - K_a(t)H_a(t)\right]^T \end{aligned}$$

Let

$$P_{x\nu}(t) = E\widetilde{x}\,(t/t-1)\,\nu_a(t)^T \tag{20}$$

Then, the optimal filter gain can be computed to be:

$$\begin{aligned} K_a(t) &= \left[P_a(t/t-1)H_a(t)^T + P_{x\nu}(t)R_aG_a(t)^T\right]\cdot \\ &\quad \cdot\left[H_a(t)P_a(t/t-1)H_a(t)^T + G_a(t)R_aG_a(t)^T + \right. \\ &\quad \left. + H_a(t)P_{x\nu}(t)G_a(t)^T + G_a(t)P_{x\nu}(t)^TH_a(t)^T\right]^{-1} \end{aligned}$$

with *a-posteriori* covariance:

$$\begin{aligned} P_a(t/t) &= P_a(t/t-1) - \\ &- K_a(t)\left[H_a(t)P_a(t/t-1) + G_a(t)R_aP_{x\nu}(t)^T\right] \end{aligned} \tag{21}$$

4) Computation of the Cross-Covariance: This section contains a derivation for the cross-covariance matrix $P_{x\nu}(t)$ required for implementing the formula above. From (20) and (17),

$$\begin{aligned} P_{x\nu}(t) &= E\left[\Phi_a(t-1,t)\widetilde{x}\,(t-1/t-1) + w_a(t-1)\right]\nu_a(t)^T \\ &= \Phi_a(t-1,t)E\widetilde{x}\,(t-1/t-1)\,\nu_a(t)^T. \end{aligned}$$

Using (19):

$$\begin{aligned} P_{x\nu}(t) &= \Phi_a(t-1,t)E\left\{\left[I - \right.\right. \\ &\quad \left. -K_a(t-1)H_a(t-1)\right]\widetilde{x}\,(t-1/t-2) - \\ &\quad \left.K_a(t-1)G_a(t-1)\nu_a(t-1)\right\}\nu_a(t)^T \\ &= \Phi_a(t-1,t)\left[I - \right. \\ &\quad \left. -K_a(t-1)H_a(t-1)\right]E\widetilde{x}\,(t-1/t-2)\,\nu_a(t)^T - \\ &\quad -\Phi_a(t-1,t)K_a(t-1)G_a(t-1)\begin{bmatrix} 0 & 0 & 0 \\ R & 0 & 0 \\ 0 & R & 0 \end{bmatrix} \end{aligned}$$

Again, by using (17) and (19),

$$\begin{aligned} &E\widetilde{x}\,(t-1/t-2)\,\nu_a(t)^T = \\ &\quad \Phi_a(t-2,t-1)E\widetilde{x}\,(t-2/t-2)\,\nu_a(t)^T\cdot \\ &\quad \cdot\Phi_a(t-2,t-1)\left[I - \right. \\ &\quad \left. -K_a(t-2)H_a(t-2)\right]E\widetilde{x}\,(t-2/t-3)\,\nu_a(t)^T - \\ &\quad -\Phi_a(t-2,t-1)K_a(t-2)G_a(t-2)\nu_a(t-2)\nu_a(t)^T \end{aligned}$$

The first term in this expression is zero since no correlation exists between $\nu_a(t)$ and estimates prior to $t-2$. Consequently,

$$\begin{aligned} &E\widetilde{x}\,(t-1/t-2)\,\nu_a(t)^T = \\ &\quad -\Phi_a(t-2,t-1)K_a(t-2)G_a(t-2)\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ R & 0 & 0 \end{bmatrix} \end{aligned} \tag{22}$$

Therefore:

$$\begin{aligned} P_{x\nu}(t) &= -\Phi_a(t-1,t)\left[I - K_a(t-1)H_a(t-1)\right]\cdot \\ &\quad \cdot\Phi_a(t-2,t-1)K_a(t-2)G_a(t-2)\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ R & 0 & 0 \end{bmatrix} - \\ &\quad -\Phi_a(t-1,t)K_a(t-1)G_a(t-1)\begin{bmatrix} 0 & 0 & 0 \\ R & 0 & 0 \\ 0 & R & 0 \end{bmatrix} \end{aligned}$$

C. Special Case: Filter applied each 3 measurements

The equations above are substantially simplified if the filter is applied using each measurement only once. That is to say, if the estimates are updated only each 3 measurement sample times. In that case, and using the notation above, $K_a(t-1) = K_a(t-2) = 0$. As a consequence, $P_{x\nu}(t) = 0$, and denoting:

$$\begin{aligned} P_0 &= P_a(t-2/t-3) \\ P_1 &= \Phi(t-2,t-1)P_0\Phi(t-2,t-1)^T + Q \\ P_2 &= \Phi(t-1,t)P_1\Phi(t-1,t)^T + Q \end{aligned}$$

one can compute

$$P_a(t/t-3) = \begin{bmatrix} P_0 & P_{10}^T & P_{20}^T \\ P_{10} & P_1 & P_{21}^T \\ P_{20} & P_{21} & P_2 \end{bmatrix}. \tag{23}$$

with $P_{10} \doteq \Phi(t-2,t-1)P_0$, $P_{20} \doteq \Phi(t-2,t)P_0$ and $P_{21} \doteq \Phi(t-1,t)P_1$.

Eq. (23) can also be re-written as:

$$\begin{aligned} P_a(t/t-3) = & \left[\begin{array}{c} I \\ \Phi(t-2,t-1) \\ \Phi(t-2,t) \end{array}\right] P_0 \cdot \\ & \cdot \left[\begin{array}{ccc} I & \Phi(t-2,t-1)^T & \Phi(t-2,t)^T \end{array}\right] + \\ & + \left[\begin{array}{c} 0 \\ I \\ \Phi(t-1,t) \end{array}\right] Q \left[\begin{array}{ccc} 0 & I & \Phi(t-1,t)^T \end{array}\right] + \\ & + \left[\begin{array}{ccc} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & Q \end{array}\right]. \end{aligned}$$

Define:

$$\begin{aligned} H_{b1}(t) &= H_2(t) + H_1(t)\Phi(t-2,t-1) + \\ &\quad + H_0(t)\Phi(t-2,t) \\ H_{b2}(t) &= H_1(t) + H_0(t)\Phi(t-1,t). \end{aligned}$$

Then, the Kalman gain simplifies to:

$$\begin{aligned} K_a(t) = & \left(\left[\begin{array}{c} I \\ \Phi(t-2,t-1) \\ \Phi(t-2,t) \end{array}\right] P_0 H_{b1}^T + \right. \\ & \left. + \left[\begin{array}{c} 0 \\ I \\ \Phi(t-1,t) \end{array}\right] Q H_{b2}^T + \left[\begin{array}{c} 0 \\ 0 \\ I \end{array}\right] Q H_0^T \right) P_{zz}^{-1}(t) \end{aligned} \quad (24)$$

where

$$P_{zz} \doteq H_{b1} P_0 H_{b1}^T + H_{b2} Q H_{b2}^T + H_0 Q H_0^T + G_a(t) R_a G_a(t)^T.$$

Notice that in this special case, the state needs to be updated at time t only, so that the filter gain can be reduced to:

$$\begin{aligned} K_{ar}(t) &= \left[\begin{array}{ccc} 0 & 0 & I \end{array}\right] K_a(t) \\ &= \left[\Phi(t-2,t)P_0 H_{b1}^T + \Phi(t-1,t) Q H_{b2}^T + \right. \\ &\quad \left. + Q H_0^T \right] P_{zz}^{-1}. \end{aligned} \quad (25)$$

The state should be updated using

$$x(t/t) = x(t/t-3) + K_{ar} z(t) \quad (26)$$

with corresponding a-posteriori covariance matrix

$$\begin{aligned} P(t/t) &= \left[\begin{array}{ccc} 0 & 0 & I \end{array}\right] P_a \left[\begin{array}{c} 0 \\ 0 \\ I \end{array}\right] \quad (27) \\ &= P_2 - \left[\Phi(t-2,t)P_0 H_{b1}^T + \Phi(t-1,t) Q H_{b2}^T + \right. \\ &\quad \left. + Q H_0^T \right] P_{zz}^{-1} \left[\Phi(t-2,t)P_0 H_{b1}^T + \right. \\ &\quad \left. + \Phi(t-1,t) Q H_{b2}^T + Q H_0^T \right]^T \end{aligned}$$

D. Epipolar and Trifocal Constraints

In this section a development of constraints based on a general three-view geometry is presented. Figure 2 presents the considered scenario, in which a single ground landmark p is observed from three images captured at time instances t_1, t_2 and t_3, where $t_1 < t_2 < t_3$. Denote by $\mathbf{T}_{ij}$ the camera translation motion from the ith to jth view, with $i, j \in \{1,2,3\}$ and $i \neq j$. Let also $\mathbf{q}_i$ and λ_i be the line of sight (LOS) vector and range, respectively, to the ground landmark p at time t_i.

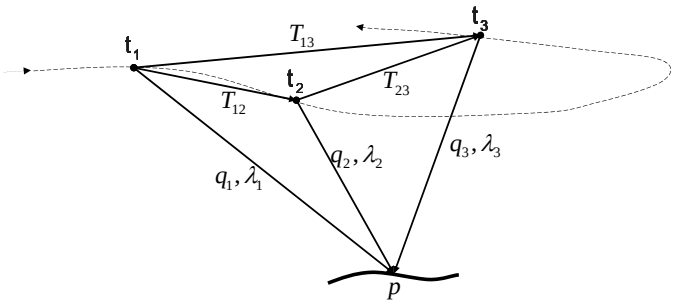


Fig. 2. Three view geometry

The position of a ground landmark p relative to the camera position at t_1, expressed in the LLLN system of t_2 may be written as:

$$\lambda_1 C_{L_2}^{C_1} \mathbf{q}_1^{C_1} = C_{L_2}^{C_1} \mathbf{T}_{12}^{C_1} + \lambda_2 C_{L_2}^{C_2} \mathbf{q}_2^{C_2} \quad (28a)$$

$$\lambda_1 C_{L_2}^{C_1} \mathbf{q}_1^{C_1} = C_{L_2}^{C_1} \mathbf{T}_{12}^{C_1} + C_{L_2}^{C_2} \mathbf{T}_{23}^{C_2} + \lambda_3 C_{L_2}^{C_3} \mathbf{q}_3^{C_3} \quad (28b)$$

where $\mathbf{q}_i^{C_i}$ is the LOS to the ground feature at t_i, expressed in a camera system at t_i; λ_i is the range between the platform position at time instant t_i and the landmark p; $C_{L_2}^{C_i}$ is a directional cosine matrix (DCM) transforming from camera system at t_i to the LLLN system at t_2; $\mathbf{T}_{ij}^{C_i}$ is the platform translation from time t_i to t_j, expressed in the camera system at t_i. Here $i, j \in \{1,2,3\}, i \neq j$.

Subtraction of Eq. (28a) from Eq. (28b) and some basic algebraic manipulations give

$$\mathbf{0} = \lambda_1 C_{L_2}^{C_1} \mathbf{q}_1^{C_1} - \lambda_2 C_{L_2}^{C_2} \mathbf{q}_2^{C_2} - C_{L_2}^{C_1} \mathbf{T}_{12}^{C_1} \quad (29a)$$

$$\mathbf{0} = \lambda_2 C_{L_2}^{C_2} \mathbf{q}_2^{C_2} - \lambda_3 C_{L_2}^{C_3} \mathbf{q}_3^{C_3} - C_{L_2}^{C_1} \mathbf{T}_{23}^{C_1} \quad (29b)$$

Since the range parameters $\lambda_1, \lambda_2, \lambda_3$ are neither required nor known, we wish to form constraints on $\mathbf{T}_{23}$ without using these parameters, or in other words, avoid structure reconstruction. For this purpose, Eq. (29) is rewritten into the matrix form

$$\begin{bmatrix} \mathbf{q}_1 & -\mathbf{q}_2 & \mathbf{0}_{3\times 1} & -\mathbf{T}_{12} \\ \mathbf{0}_{3\times 1} & \mathbf{q}_2 & -\mathbf{q}_3 & -\mathbf{T}_{23} \end{bmatrix}_{6\times 4} \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \lambda_3 \\ 1 \end{bmatrix}_{4\times 1} = \mathbf{0}_{6\times 1} \quad (30)$$

For the sake of brevity, the superscript L_2 was omitted, e. g. $\mathbf{q}_1 \equiv \mathbf{q}_1^{L_2} = C_{L_2}^{C_1} \mathbf{q}_1^{C_1}$.

Let

$$A = \begin{bmatrix} \mathbf{q}_1 & -\mathbf{q}_2 & \mathbf{0}_{3\times 1} & -\mathbf{T}_{12} \\ \mathbf{0}_{3\times 1} & \mathbf{q}_2 & -\mathbf{q}_3 & -\mathbf{T}_{23} \end{bmatrix} \in \mathbb{R}^{6\times 4} \quad (31)$$

In a similar manner to Refs. [31] and [47], since all the components in $\begin{bmatrix} \lambda_1 & \lambda_2 & \lambda_3 & 1 \end{bmatrix}^T$ are nonzero, it follows that $\mathrm{rank}(A) < 4$. In particular, the determinant of any 4×4 submatrix of A should be equal to zero. A careful examination of all such possible submatrices of A will give a complete set of constraints derived from a general three-view geometry.

All the 4×4 submatrices of A comprised of the first three rows with any of the other rows of A yield the epipolar constraint for the first two views:

$$\mathbf{q}_1^T(\mathbf{T}_{12} \times \mathbf{q}_2) = 0 \tag{32}$$

This constraint, when applied to a single pair of matching features, forces the three vectors $\mathbf{q}_1, \mathbf{T}_{12}, \mathbf{q}_2$ to be coplanar. In case several matching points are considered, it is possible to determine $\mathbf{T}_{12}$ only up to scale.

In the same manner, the last three rows, with any of the first three rows of A provide the epipolar constraint for the second and the third views:

$$\mathbf{q}_2^T(\mathbf{T}_{23} \times \mathbf{q}_3) = 0 \tag{33}$$

The more interesting result, however, stems from analyzing the determinants of all the other possible submatrices of A. After extensive algebra (cf. Ref. [46]) one gets:

$$(\mathbf{q}_2 \times \mathbf{q}_1)^T(\mathbf{q}_3 \times \mathbf{T}_{23}) \quad = \quad (\mathbf{q}_1 \times \mathbf{T}_{12})^T(\mathbf{q}_3 \times \mathbf{q}_2) \tag{34}$$

The constraints (32,33) force the translation vectors to be co-planar with the LOS vectors. Given multiple matching features, one can determine from Eqs. (32) and (33) the translation vectors $\mathbf{T}_{12}$ and $\mathbf{T}_{23}$, respectively, up to scale. In general, these two scale unknowns are different. The two scales are connected through Eq. (34), which relates between the magnitudes of $\mathbf{T}_{23}$ and $\mathbf{T}_{12}$. Consequently, if the magnitude of $\mathbf{T}_{12}$ is known,it is possible to calculate both the direction and the magnitude of $\mathbf{T}_{23}$, given multiple matching features.

Several remarks are in order. First, Eqs. (32-34) also contain rotation parameters, since all the quantities are assumed to be expressed in the LLLN system at t_2. Second, structure reconstruction is not required. As shown in the sequel, this allows to maintain a constant-size state vector comprised of the vehicle's parameters only, resulting in a reduced computational load.

IV. Mosaic Aided Navigation Methods

In this section we present a technique for fusing the three-view geometry constraints, with a standard navigation system, assuming three images with a mutual overlapping area had been identified. The data fusion is performed using an indirect IEKF that estimates the navigation parameter errors instead of the parameters themselves. These estimated errors are then used for correcting the navigation solution computed by the navigation system.

When real imagery and navigation data are considered, the existence of navigation errors and image noise renders the constraints of Eqs. (32-34) inaccurate. Thus, the following residual measurement is defined:

$$\mathbf{z} \doteq \begin{bmatrix} U \\ F \\ 0 \end{bmatrix}_{N\times 3} \mathbf{T}_{23} - \begin{bmatrix} W \\ 0 \\ G \end{bmatrix}_{N\times 3} \mathbf{T}_{12} \doteq \mathcal{A}\mathbf{T}_{23} - \mathcal{B}\mathbf{T}_{12} \tag{35}$$

where

$$\begin{aligned} U &= \begin{bmatrix} \mathbf{u}_1 & \dots & \mathbf{u}_{N_{123}} \end{bmatrix}^T & W &= \begin{bmatrix} \mathbf{w}_1 & \dots & \mathbf{w}_{N_{123}} \end{bmatrix}^T \\ F &= \begin{bmatrix} \mathbf{f}_1 & \dots & \mathbf{f}_{N_{23}} \end{bmatrix}^T & G &= \begin{bmatrix} \mathbf{g}_1 & \dots & \mathbf{g}_{N_{12}} \end{bmatrix}^T \end{aligned}$$

Notice that N_{ij} is the number of corresponding feature points between frame i, and frame j, and likewise for N_{ijk} between frames i, j and k. Also:

$$\mathbf{f}^T \doteq (\mathbf{q}_2 \times \mathbf{q}_3)^T \tag{36}$$

$$\mathbf{g}^T \doteq (\mathbf{q}_1 \times \mathbf{q}_2)^T \tag{37}$$

$$\mathbf{u}^T \doteq (\mathbf{q}_1 \times \mathbf{q}_2)^T[\mathbf{q}_3]_\times = \mathbf{g}^T[\mathbf{q}_3]_\times \tag{38}$$

$$\mathbf{w}^T \doteq (\mathbf{q}_2 \times \mathbf{q}_3)^T[\mathbf{q}_1]_\times = \mathbf{f}^T[\mathbf{q}_1]_\times \tag{39}$$

Full development of these equations may be found in Ref. [46].

Since $\mathbf{T}_{12} = \mathbf{Pos}(t_2) - \mathbf{Pos}(t_1)$, $\mathbf{T}_{23} = \mathbf{Pos}(t_3) - \mathbf{Pos}(t_2)$, and the matrices F, G, U, W are functions of the LOS vectors, the residual measurement $\mathbf{z}$ is actually a nonlinear function of the following parameters[2]:

$$\begin{aligned} \mathbf{z} \quad = \quad & \mathbf{h}\left(\mathbf{Pos}(t_3), \mathbf{\Psi}(t_3), \mathbf{Pos}(t_2), \mathbf{\Psi}(t_2),\right. \\ & \left. \mathbf{Pos}(t_1), \mathbf{\Psi}(t_1), \left\{\mathbf{q}_{1_i}^{C_1}, \mathbf{q}_{2_i}^{C_2}, \mathbf{q}_{3_i}^{C_3}\right\}\right) \end{aligned} \tag{40}$$

Here (t_3, t_2, t_1) denote the time instances in which the three mutually overlapping images were captured, with t_3 being the current time.

We now define the state vector as

$$\mathbf{X} = \begin{bmatrix} \Delta\mathbf{P}^T & \Delta\mathbf{V}^T & \Delta\mathbf{\Psi}^T & \mathbf{d}^T & \mathbf{b}^T \end{bmatrix}^T \tag{41}$$

where $\Delta\mathbf{P} \in \mathbb{R}^3, \Delta\mathbf{V} \in \mathbb{R}^3, \Delta\mathbf{\Psi} \in SO(3)$ are the position, velocity and attitude errors, respectively, and $(\mathbf{d}, \mathbf{b})$ is the parameterization of errors in the inertial sensor measurements: $\mathbf{d} \in \mathbb{R}^3$ is the gyro drift, and $\mathbf{b} \in \mathbb{R}^3$ is the accelerometer bias. The first 9 components of $\mathbf{X}$ are given in LLLN coordinates, while the last 6 are written in a body-fixed reference frame. The corresponding transition matrix $\Phi_d(t_2, t_1)$ satisfying $\mathbf{X}(t_2) = \Phi_d(t_2, t_1)\mathbf{X}(t_1)$ is given in Ref. [45].

Noting that $\mathbf{X}$ contains navigation information only for the current time instant, while the residual measurement $\mathbf{z}$ contains navigation and imagery information from the three time instances t_1, t_2 and t_3, we define the following augmented state vector:

$$\boldsymbol{\mathcal{X}}(t_3, t_2, t_1) \doteq \begin{bmatrix} \mathbf{X}(t_3)^T & \mathbf{X}(t_2)^T & \mathbf{X}(t_1)^T \end{bmatrix}^T \tag{42}$$

Linearizing $\mathbf{h}$ about $\mathbf{Pos}(t_3), \mathbf{\Psi}(t_3), \mathbf{Pos}(t_2), \mathbf{\Psi}(t_2)$, $\mathbf{Pos}(t_1), \mathbf{\Psi}(t_1)$ and $\left\{\mathbf{q}_{1_i}^{C_1}, \mathbf{q}_{2_i}^{C_2}, \mathbf{q}_{3_i}^{C_3}\right\}$, and keeping the first order terms yields

$$\mathbf{z} \approx \mathcal{H}\mathcal{X} + \mathcal{D}\mathbf{v} \tag{43}$$

where $\mathcal{H}$ is the gradient of $\mathbf{h}$ with respect to $\boldsymbol{\mathcal{X}}$,

$$\mathcal{H}_{N\times 45} = \begin{bmatrix} H_3 & H_2 & H_1 \end{bmatrix} \tag{44}$$

with H_i being the gradient of $\mathbf{h}$ with respect to $\mathbf{X}(t_i)$, $i \in \{1, 2, 3\}$. The matrix $\mathcal{D}$ is the gradient of $\mathbf{h}$ with respect to the LOS vectors and $\mathbf{v}$ is the image noise associated with the LOS vectors with covariance matrix R. The development of the matrices $\mathcal{H}, \mathcal{D}, R$ is given in Ref. [46].

[2] In Eq. (41), $\left\{\mathbf{q}_{1_i}^{C_1}, \mathbf{q}_{2_i}^{C_2}, \mathbf{q}_{3_i}^{C_3}\right\}$ refers to the fact that LOS vectors from all the three images are used for calculating the residual measurement $\mathbf{z}$. Note that each of the matrices F, G, U, W is a function of a different set of matching points.

Since it is unknown *a-priori* which three images will have a mutual overlapping area, and in order to maintain a constant-size state vector, each captured image should be stored and associated with the relevant navigation information[3]. The navigation data that should be attached to each image are the platform position, attitude, gimbal angles and the filter's uncertainty covariance matrix.

The resulting estimation problem can be solved using results discussed in section III-B.

A. Computational Requirements

A single filter update step, given three mutually overlapping images, involves computation of the matrices $\mathcal{A}, \mathcal{B}$ and the Jacobian matrices $\mathcal{H}, D$. These calculations are linear in N. Noting that the state vector is constant in size, the most computationally expensive operation in the filter update step is the inversion of an $N \times N$ matrix that is required for the calculation of the gain matrix.

Although the augmented state vector $\boldsymbol{\mathcal{X}}$ is a 45-element vector, the actual calculations are performed using the basic state vector $\mathbf{X} \in \mathbb{R}^{15\times 1}$. The computational load is not significantly increased even when the augmented state vector $\boldsymbol{\mathcal{X}}$ is used (given that the correlation terms involving the current time are known), since the propagation step is still carried out using the basic state vector $\mathbf{X}$. The calculation of the gain matrix $\mathcal{K}$ involves inversion of an $N \times N$ matrix, while *a-posteriori* estimation of only the first 15 elements of $\boldsymbol{\mathcal{X}}$ and of the 15×15 upper left sub-matrix of the augmented covariance matrix $\mathcal{P}$ should be performed.

The computational load of the proposed method does not change significantly over time (depending on the variation of N), regardless of the scenarios in which the algorithm is applied to (including loops). Moreover, if the computational capability is limited, it is possible to utilize only part of the available matching pairs and triplets, or eliminate the epipolar constraints, thus reducing the computational load even further.

In contrast to the above, the computational requirements of other existing methods are much higher. The SLAM approach entails constantly increasing computational requirements, due to the augmentation of the state vector. Furthermore, the high computational load is induced in *each* filter propagation step. Methods that perform state augmentation until a certain size of the state vector is reached (e. g. Ref. [36]), handle loops trajectory by applying bundle adjustment over *all* the images that were captured during the loop chain, as opposed to processing only three images as done in our approach.

V. Results

This section contains results of the developed mosaic-aided navigation method that are based on two-view and three-view geometry. The two-view method is demonstrated using images from Google Earth[4] and a simulation navigation. The three-view method is demonstrated in an experiment involving real imagery and navigation data. Additional results are given in Refs. [44], [45] and [46].

[3]Practically, since saving all the captured images might be difficult in certain applications, one may settle for storing the images at some high frequency, or following different logic, at the expense of having less images in the repository to seek for potential overlapping areas.

[4]http://earth.google.com/index.html.

TABLE I
Initial Navigation Errors and IMU Errors

Parameter	Description	Value	Units
$\Delta\mathbf{P}$	Initial position error	$(100, 100, 100)^T$	m
$\Delta\mathbf{V}$	Initial velocity error	$(0.3, 0.3, 0.3)^T$	m/s
$\Delta\boldsymbol{\Psi}$	Initial attitude error	$(0.1, 0.1, 0.1)^T$	deg
$\mathbf{d}$	IMU drift	$(1, 1, 1)^T$	deg/hr
$\mathbf{b}$	IMU bias	$(1, 1, 1)^T$	mg

A. Vision-Aided Navigation - Two-view Method

This section contains results of the developed mosaic-aided navigation method based on the camera scanning and mosaic construction procedures described in Section III-A. The results are based on images obtained from Google Earth and a navigation simulation. A complete description of the fusion between the products of these two procedures and the navigation system is given in Ref. [44] and Ref. [45].

The results are compared to vision-aided navigation based on a statndard two-view motion estimation technique. The examined scenario consists of a narrow-FOV camera ($5^o \times 3^o$) and a low-texture scene. The platform performs a straight and level north-heading flight. The observed scene along this trajectory is of a planar nature with about 50 m elevation above sea level.

The assumed values of IMU errors and initial navigation errors are given in Table I. The IMU errors are then used for contaminating the pure IMU readings, while the initial input to the strapdown module is set according to initial platform parameters and initial navigation errors (cf. Section V).

The experiment consisted of 50 sec of inertial flight, followed by a 50 sec of vision-aided phase, during which the mosaic- and two-view-based motion estimations were injected into the navigation system. The last phase is another inertial navigation flight segment for 50 sec. Figures 3-6 provide the experimental results, comparing the navigation performance for the two examined methods (mosaic and two-view). In addition, the development of inertial navigation errors is given for reference.

The enhanced performance of mosaic-aided navigation can be clearly seen. During the vision-aided phase, the position and velocity errors (Figures 3 and 4) perpendicular to the flight heading are significantly reduced. The mosaic-based aiding yields better results than two-view-based aiding, due to more accurate vision-based motion estimation. As for the roll angle error (Figure 5), although this error is smaller with the two-view method, it is expected to reach higher values if more measurements were accepted by the filter.

When examining the behavior of navigation errors in an inertial segment (after the vision-aided phase), one can notice the slow development of inertial errors when using mosaic aiding. The reason for this is the improved bias estimation compared to the estimation using the two-view method, as shown in Figure 6: b_z is almost exactly estimated and thus

it does not contribute to the growth of inertial position and velocity errors in the down axis. The drift state was not estimated at all, because all the relative rotation measurements were rejected by the filter due to their low quality.

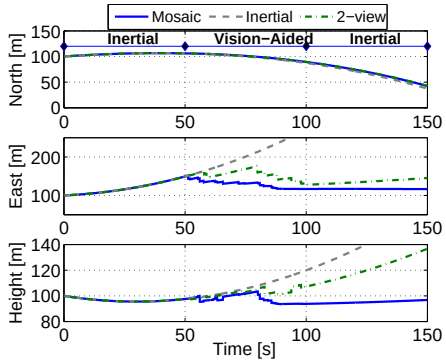


Fig. 3. Mosaic aiding applying two-view techniques vs. conventional two-view aiding: Position errors. Inertial error development in the north direction due to lack of observability. Reduced errors in the east and down directions, with a significant improvement in favor of the mosaic aiding.

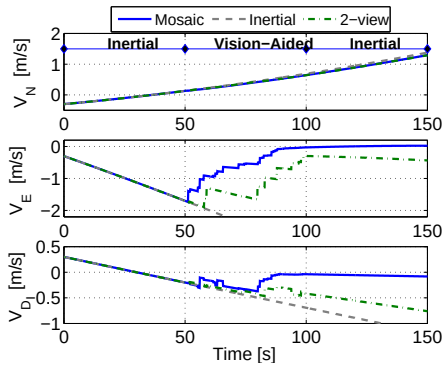


Fig. 4. Mosaic aiding applying two-view techniques vs. conventional two-view aiding: Velocity errors. Inertial error development in the north direction due to lack of observability. Reduced errors in the east and down directions, with a significant improvement in favor of the mosaic aiding.

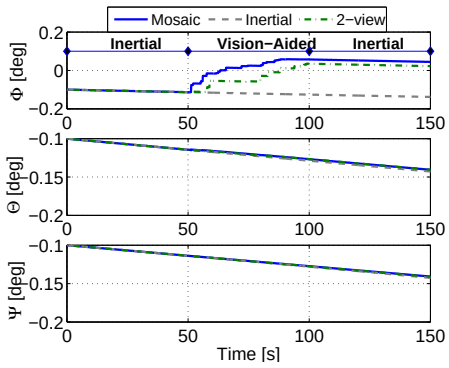


Fig. 5. Mosaic aiding applying two-view techniques vs. conventional two-view aiding: Euler angle errors. Roll angle error estimation for both motion estimation methods. Pitch and yaw angles errors are not reduced due to lack of observability.

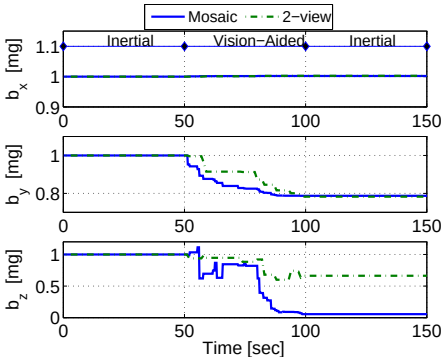


Fig. 6. Mosaic aiding applying two-view techniques vs. conventional two-view aiding: Bias estimation errors. Considerably improved b_z estimation in favor of mosaic-aided navigation.

B. Mosaic-Aided Navigation - Three View method

In this section results are shown for navigation aiding based on the developed three-view-geometry method. First, the method is applied on real images and simulated navigation. Next, the method is demonstrated in an experiment involving real imagery and navigation data. A statistical study of the method based on synthetic imagery and simulation navigation can be found in Ref. [46].

1) Results based on Real Images and Simulated Navigation: The set of three partially overlapping images used in this section is shown in Figure 7. The image matching process

(a) First image (b) Second image

(c) Third image

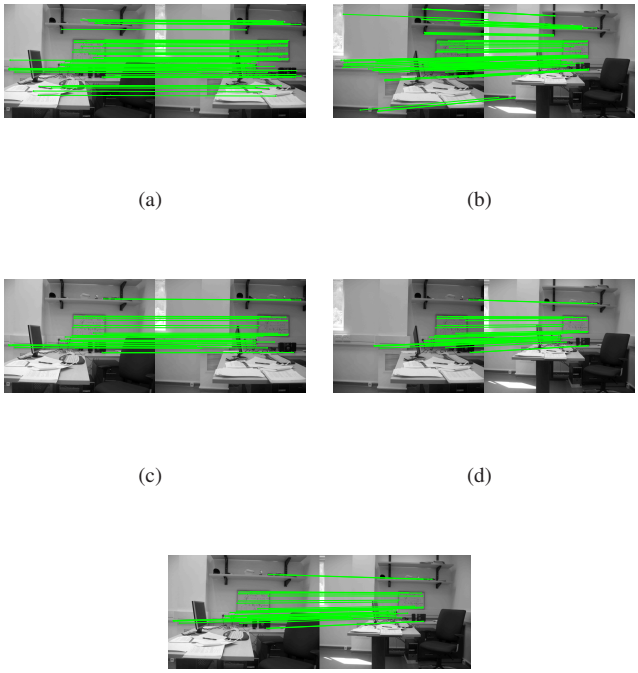
Fig. 7. Original indoor images.

is given in Figure 8.

The camera orientation was identical while the three images were captured. The relative translation motion of the camera (manually measured) between these images is

$$\mathbf{T}_{12} = \begin{pmatrix} 59.7 & 0 & 0 \end{pmatrix}^T cm$$
$$\mathbf{T}_{23} = \begin{pmatrix} -36.4 & -67.9 & 0 \end{pmatrix}^T cm$$

Since three-view geometry constrains the camera motion up to a common projective transformation, we can create any



(a) (b) (c) (d) (e)

Fig. 8. Image matching. (a) Matching pairs between image 1 and 2: $\{\mathbf{x}_1^i, \mathbf{x}_2^i\}_{i=1}^{N_{12}}$; (b) Matching pairs between image 2 and 3: $\{\mathbf{x}_2^i, \mathbf{x}_3^i\}_{i=1}^{N_{23}}$; (c) Matching triplets between image 1 and 2: $(\mathbf{x}_1, \mathbf{x}_2) \in \{\mathbf{x}_1^i, \mathbf{x}_2^i, \mathbf{x}_3^i\}_{i=1}^{N_{123}}$; (d) Matching triplets between image 2 and 3: $(\mathbf{x}_2, \mathbf{x}_3) \in \{\mathbf{x}_1^i, \mathbf{x}_2^i, \mathbf{x}_3^i\}_{i=1}^{N_{123}}$; (e) Matching triplets between image 1 and 3: $(\mathbf{x}_1, \mathbf{x}_3) \in \{\mathbf{x}_1^i, \mathbf{x}_2^i, \mathbf{x}_3^i\}_{i=1}^{N_{123}}$.

trajectory and set the transformation accordingly. Next, we present such a trajectory, and show the results when fusing the algorithm with an inertial navigation of the platform.

The assumed trajectory is given in Figure 9. As can be seen, the platform performs a shifted loop while maintaining a constant altitude, and observes the same scene three times (the camera orientation is denoted by arrows in the figure) at the following time instants: $t_1 = 22.74$ sec, $t_2 = 50.00$ sec, $t_3 = 405.48$ sec. Note that the crossover occurs at t_3.

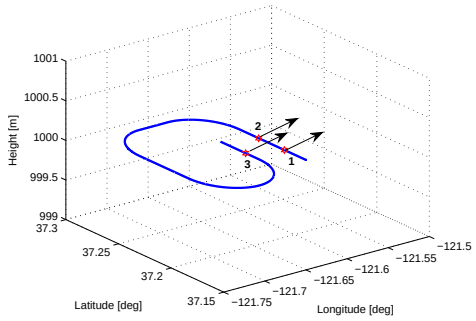


Fig. 9. Platform trajectory. The three views and camera orientation are denoted as well.

The true translation in the appropriate LLLN coordinate frame between these three time instants is[5]

$$\mathbf{T}_{12}^{L_1} = \begin{pmatrix} 2725.6 & 0 & 0 \end{pmatrix}^T m$$
$$\mathbf{T}_{23}^{L_2} = \begin{pmatrix} -1661.9 & -3100 & 0 \end{pmatrix}^T m$$

Figures 10-13 shows the development of navigation errors and the square root of filter covariance. As can be seen, the position error in all axes (Figure 10) is reduced to the levels of position error that were present while the first image was captured ($t_1 = 22.74$ sec), e. g. the error in east component drops from around 1900 m to only 90 m, while the initial error (at $t = 0$ sec) was 100 m.

As to the other parameters in the state vector, the velocity errors were also significantly reduced (Figure 11), while the attitude error was somewhat deteriorated. The drift state (Figure 13) is not estimated, while the bias state is estimated in z axis, and partially estimated in x axis.

It should be noted, that a different scenario (e. g. different camera orientation) might change the components in bias and drift states that may be estimated. However, the estimation of position error , as described above, will be obtained regardless to the examined scenario, as long as three partially overlapping images are available.

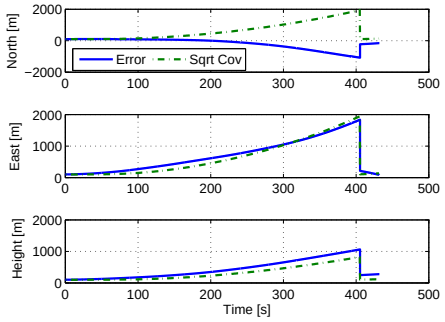


Fig. 10. Three-view method: a single loop update based on real images - Position errors

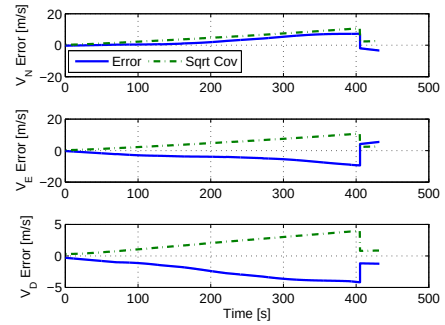


Fig. 11. Three-view method: a single loop update based on real images - Velocity errors

[5]Thus, the scaling constant of three-view geometry is around 45.6 in this scenario.

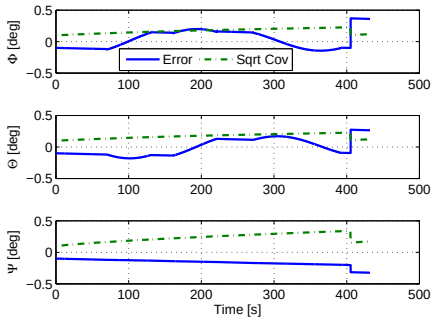


Fig. 12. Three-view method: a single loop update based on real images - Euler angle errors

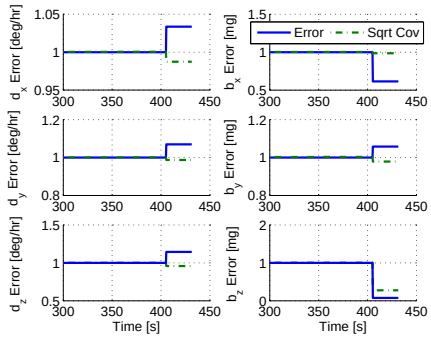


Fig. 13. Three-view method: a single loop update based on real images - Drift and bias estimation errors (zoomed)

2) Experimental Results: The experimental setup contains an MTi-G Xsens[6] IMU/INS and a 207MW Axis network camera[7] that were mounted on top of a ground vehicle. The vehicle was manually commanded using a joystick, while the camera captured images perpendicular to the motion heading. During the experiment, the inertial sensor measurements and camera images were recorded for post-processing at 100 and 15 Hz frequency, respectively. In addition, these two data sources were synchronized by associating to each image a time stamp from the navigation timeline. Figure 14 shows the described hardware.

(a) (b)

Fig. 14. Hardware used in the experiment. (a) Axis 207MW camera. (b) XSens MTi-G IMU/INS.

The true vehicle trajectory was manually measured during the experiment and associated with a timeline by post-processing the inertial sensors readings. The obtained reference trajectory is presented in Figure 15 and Figure 16. The diamond marks denote the manual measurements of the vehicle position, while the solid line represents a linear interpolation between each two marks. The vehicle began its motion at $t \approx 76$ seconds. As can be seen from Figure 15, the vehicle performed the same closed trajectory twice (see also Figure 16).

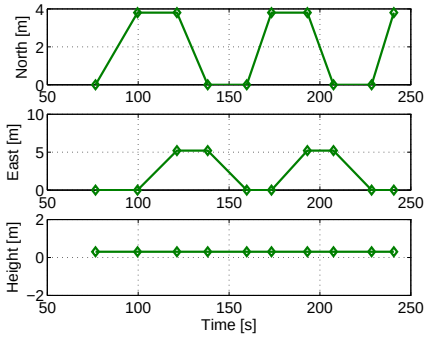


Fig. 15. Trajectory performed in the experiment.

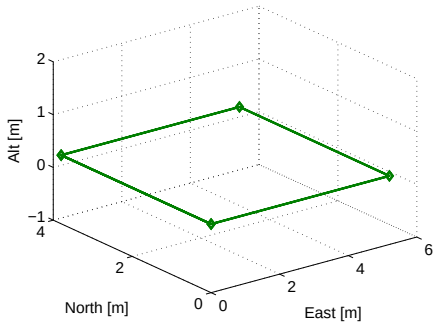


Fig. 16. Trajectory performed in the experiment - 3D view

The recorded inertial sensor measurements were processed by the strapdown block yielding an inertial navigation solution. Sets of three mutually overlapping images were manually identified and chosen. The proposed algorithm was applied for each such set and used for updating the navigation system. Two different update modes are demonstrated in this experiment: a) "Sequential update" in which all the three images were acquired closely to each other, and b) "Loop update" in which the first two images were captured while the platform passed the region for the first time, whereas the third image was obtained at the second passing. The algorithm activation is the same in both cases.

The image matching process for the first set of three mutually overlapping images is partially shown in Figure 17. Two camera captured images (of the three) are given in Figures 17(a)-17(b). The set of matching pairs $\left\{\mathbf{x}_1^i, \mathbf{x}_2^i\right\}_{i=1}^{N_{12}}$ is provided in Figure 17(c). As can be seen, the images have a considerable mutual overlapping area, and thus it was possible to obtain a large number of matching pairs (and triplets); only

[6]http://www.xsens.com/en/general/mti-g.

[7]http://www.axis.com/products/cam_207mw/index.htm.

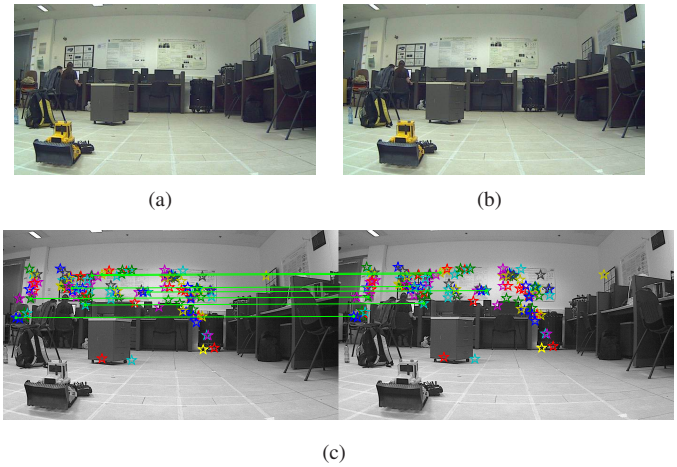


Fig. 17. Image matching process. (a),(b) Two camera-captured overlapping images. (c) Matching pairs: $(\mathbf{x}_1, \mathbf{x}_2) \in \{\mathbf{x}_1^i, \mathbf{x}_2^i, \mathbf{x}_3^i\}_{i=1}^{N_{123}}$. For clarity, only the first few matches are explicitly shown; the rest of the matches are denoted by marks in each image.

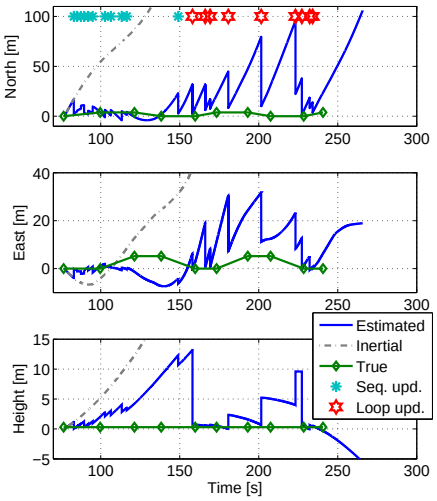


Fig. 18. Estimated position in an experiment. Reduced position errors in the sequential mode. The position error is reset back to its prior levels in the loop updates.

a few of them are explicitly shown in Figure 17(c), while the rest of the matches are denoted by marks.

The results are shown in Figure 18, that presents the estimated position compared to the true position. In addition, an inertial scenario (based only on inertial navigation data) is given. The update mode of the method in the experiment was as follows (also indicated in the figure). Until $t \approx 150$ seconds sequential updates were performed; Loops updates were carried out after the platform has completed a loop, starting from $t \approx 158$ seconds.

During the sequential updates phase, the time instances (t_1, t_2, t_3) were chosen such that $t_2 - t_1 \approx 1$ seconds and $t_3 - t_2 \approx 5$ seconds. As can be seen, while sequential updates are active, the position is estimated up to several meters, whereas the inertial solution rapidly diverges.

After the vehicle had completed its first loop, the algorithm was applied in a "loop update" mode. As can be seen, each such update, significantly reduces the inertially accumulated position error, allowing to maintain a small error of several meters after over 150 seconds of operation. For comparison, the inertial error approaches 1100 meter (in the north axis) over this period of time, indicating the low quality of the inertial sensors. Note that the position error is reduced in *all* axes, including along the motion direction, which is not possible in two-view methods for navigation aiding.

VI. Conclusions

The paper has shown that the on-line construction of an image mosaic can be effectively exploited to aid the navigation system of an airborne platform. The paper contains three main sections. On the first, some of the tools required for computing a mosaic and using the information collected as a side product within a navigation filter were discussed, including mosaic construction, multi-point Kalman filter and a new formulation of the tri-focal tensor from multi-frame computer vision. As discussed, the multi-point filter is required since the measurement process includes the navigation solution at several time instants. Also, the new formulation of the three-frame vision constraint is adequate for fusing in an implicit Kalman filtering setup. The next section deals with a collection of methods for fusing multi-frame constraints and the side-products of the mosaic construction procedure with a navigation solution. The following section presents some illustrative examples to show how the mosaic aided approach can slow down the error build-up in an inertial navigation algorithm and also reduce these errors to their initial level when a loop-closure is detected. Overall, mosaic-aided navigation appears to provide an adequate solution for a number of relevant mission scenarios, including the return to base, long-time loitering and possibly navigation information sharing between several cooperative platforms.

References

[1] Merhav, S. and Bresler, Y., "On-Line Vehicle Motion Estimation from Visual Terrain Information Part 1: Recursive Image Registration," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 22, No. 5, 1986, pp. 583–587.

[2] Gurfil, P. and Rotstein, H., "Partial Aircraft State Estimation from Visual Motion Using the Subspace Constraints Approach," *Journal of Guidance, Control and Dynamics*, Vol. 24, No. 5, July 2001, pp. 1016–1028.

[3] Soatto, S. and Perona, P., "Recursive 3-D Visual Motion Estimation Using Subspace Constraints," *International Journal of Computer Vision*, Vol. 22, No. 3, 1997, pp. 235–259.

[4] Diel, D., DeBitetto, P. and Teller, S., "Epipolar Constraints for Vision-Aided Inertial Navigation," *Proceedings of the IEEE Workshop on Motion and Video Computing*, Vol. 2, January 2005, pp. 221–228.

[5] Roumeliotis, S., Johnson, A. and Montgomery, J., "Augmenting Inertial Navigation with Image-Based Motion Estimation," *IEEE International Conference on Robotics and Automation*, Vol. 4, 2002, pp. 4326–4333.

[6] Merhav, S. and Bresler, Y., "On-Line Vehicle Motion Estimation from Visual Terrain Information Part 2: Ground Velocity and Position Estimation," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 22, No. 5, 1986, pp. 588–604.

[7] Lerner, R., Rivlin, E., and Rotstein, H., "Pose and Motion Recovery from Feature Correspondences and a Digital Terrain Map," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 28, No. 9, September 2006, pp. 1404–1417.

[8] Sim, D., Park, R., Kim, R., Lee, S., and Kim, I., "Integrated Position Estimation Using Aerial Image Sequences," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 24, No. 1, January 2002, pp. 1–18.

[9] Gracias, N., Zwaan, S., Bernardino, A. and Santos-Victor, J., "Results on Underwater Mosaic-based Navigation," *IEEE OCEANS*, Vol. 3, 2002, pp. 1588–1594.

[10] Gracias, N. and Santos-Victor, J., "Underwater Video Mosaics as Visual Navigation Maps," *Computer Vision and Image Understanding*, Vol. 79, 2000, pp. 66–91.

[11] Garcia, R., "A proposal to estimate the motion of an underwater vehicle through visual mosaicking," *Phd thesis. University of Girona, Spain*, 2002.

[12] Garcia, R., Puig, J., Ridao, P., and Cufi, X., "Augmented State Kalman Filtering for AUV Navigation," *IEEE Proceedings on International Conference Robotics and Automation*, Vol. 4, 2002, pp. 4010– 4015.

[13] Davison, A.J., Reid, I.D., Molton, N.D. and Stasse, O., "MonoSLAM: Real-Time Single Camera SLAM," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 29, No. 6, 2007.

[14] Bryson, M. and Sukkarieh, S., "Active Airborne Localization and Exploration in Unknown Environments using Inertial SLAM," *IEEE Aerospace Conference*, 2006.

[15] Bryson, M. and Sukkarieh, S., "Bearing-Only SLAM for an Airborne Vehicle," *Australasian Conference on Robotics and Automation*, 2005.

[16] Kim, J. and Sukkarieh, S., "6DoF SLAM aided GNSS/INS Navigation in GNSS Denied and Unknown Environments," *Journal of Global Positioning Systems*, Vol. 4, No. 1-2, pp. 120–128, 2005.

[17] Peleg, S. and Herman, J., "Panoramic mosaics by manifold projection," *In CVPR*, 1997, pp. 338–343.

[18] Szeliski, R., "Image alignment and stitching: A tutorial," *Tech. Rep. MSR-TR-2004-92, Microsoft Research*, 2005.

[19] Fleischer, S., Wang, H., and Rock, S., "Video Mosaicking Along Arbitrary Vehicle Paths," *Proceedings of the Symposium on Vehicle Technology*, 1996, pp. 293–299.

[20] Gracias, N., Costeira, J., and J.Santos-Victor, "Linear Global mosaic For Underwater Surveying," *IAV2004*, 2004.

[21] Kanazawa, Y. and Kanatani, K., "Image Mosaicing by Stratified Matching," *Image and Vision Computing*, Vol. 22, 2004, pp. 93–103.

[22] Peleg, S., Rousso, B., Rav-Acha, A., and Zomet, A., "Mosaicing on Adaptive Manifolds," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 22, No. 10, 2000, pp. 1144–1154.

[23] Shum, H. and Szeliski, R., "Systems and Experiment Paper: Construction of Panroamic Image Mosaics with Global and Local Alignment," *International Journal of Computer Vision*, Vol. 36, No. 2, 2000, pp. 101–130.

[24] Zelnik-Manor, L. and Irani, M., "Multiview Constraints on Homographies," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 24, No. 2, 2002, pp. 214–223.

[25] Richmond, K. and Rock, S., "An Operational Real-Time Large-Scale Visual Mosaicking and Navigation System," *OCEANS*, September 2006, pp. 1–6.

[26] Ferrer, J., Elibol, A., Delaunoy, O., Gracias, N. and Garcia, R., "Large-Area Photo-Mosaics Using Global Alignment and Navigation Data," *OCEANS*, September 2007, pp. 1–9.

[27] Irani, M., Anandan, P., and Hsu, S., "Mosaic Based Representations of Video Sequences and Their Applications," *Proc. of IEEE ICCV*, 1995, pp. 605–611.

[28] Zhang, P., Milios, E. E., and Gu, J., "Graph-based Automatic Consistent Image Mosaicking," *IEEE International Conference on Robotics and Biomimetics, Shenyang*, China, Paper no. 332, 2004.

[29] Zhu, Z., Hanson, A., and Riseman, E., "Generalized Parallel-Perspective Stereo Mosaics from Airborne Video," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 26, No. 2, 2004, pp. 226–237.

[30] Eustice, R. M., Pizarro, O. and Singth, H., "Visually Augmented Navigation for Autonomous Underwater Vehicles," *IEEE Journal of Oceanic Engineering*, Vol. 33, No. 2, 2008, pp. 103–122.

[31] Hartley, R. and Zisserman, A., "Multiple View Geometry," *Cambridge University Press*, 2000.

[32] Tsai, R., Huang, T. and Zhu, W., "Estimating Three-Dimensional Motion Parameters of a Rigid Planar Patch, II: Singular Value Decomposition," *IEEE Transactions on Acoustics, Speech and Signal Processing*, Vol. 30, No. 4, August 1982, pp. 525–534.

[33] Caballero, F., Merino, L., Ferruz, J. and Ollero, A., "Improving Vision-based Planar Motion Estimation for Unmanned Aerial Vehicles through Online Mosaicing," *IEEE International Conference on Robotics and Automation*, Orlando, Florida, May 2006, pp. 2860–2865.

[34] Caballero, F., Merino, L., Ferruz, J. and Ollero, A., "Homography Based Kalman Filter for Mosaic Building. Applications to UAV position estimation," *IEEE International Conference on Robotics and Automation*, Roma, Italy, April 2007, pp. 2004–2009.

[35] Caballero, F., Merino, L., Ferruz, J. and Ollero, A., "Vision-Based Odometry and SLAM for Medium and High Altitude UAVs," *Journal of Intelligent and Robotic Systems*, Vol. 54, No. 1-3, March 2009, pp. 137–161.

[36] Mourikis, A. and Roumeliotis, I., "A Multi-State Constraint Kalman Filter for Vision-aided Inertial Navigation," *International Conference on Robotics and Automation*, April 2007, pp. 3565–3572.

[37] Mourikis, A. and Roumeliotis, I., "A Dual-Layer Estimator Architecture for Long-term Localization," *IEEE Computer Vision and Pattern Recognition Workshops*, June 2008, pp. 1–8.

[38] Shakernia, O., Vidal, R., Sharp, C., Ma, Y. and Sastry, S., "Multiple View Motion Estimation and Control for Landing an Unmanned Aerial Vehicle", *IEEE International Conference on Robotics and Automation*, May 2002, pp. 2793–2798.

[39] Yu, Y. K., Wong, K. H., Chang, M. M. Y. and Or, S. H., " Recursive Camera-Motion Estimation With the Trifocal Tensor," *IEEE Transactions on Systems, Man, And Cybernetics - Part B: Cybernetics*, Vol. 36, No. 5, October 2006, pp. 1081–1090.

[40] Yu, Y. K., Wong, K. H., Or, S. H. and Chang, M. M. Y., " Robust 3-D Motion Tracking From Stereo Images: A Model-Less Method," *IEEE Transactions on Instrumentation and Measurement*, Vol. 57, No. 3, March 2008, pp. 622–630.

[41] Guerrero, J. J., Murillo, A. C. and Sagües, C.," Localization and Matching Using the Planar Trifocal Tensor with Bearing-Only Data," *IEEE Transactions on Robotics*, Vol. 24, No. 2, April 2008, pp. 494–501.

[42] Lowe, D., "Distinctive Image Features from Scale-Invariant Keypoints", *International Journal of Computer Vision*, Vol. 60, No. 2, November 2004, pp. 91–110.

[43] Fischler, M. and Bolles, R., "Random Sample Consensus: a Paradigm for Model Fitting with Application to Image Analysis and Automated Cartography," *Commun. Assoc. Comp. Mach.*, Vol. 24, 1981, pp. 381–395.

[44] Indelman, V., Gurfil, P., Rivlin, E. and Rotstein,H., "Real-Time Mosaic-Aided Aircraft Navigation: I. Motion Estimation," *AIAA GN&C Conference*, USA, 2009.

[45] Indelman, V., Gurfil, P., Rivlin, E. and Rotstein,H., "Real-Time Mosaic-Aided Aircraft Navigation: II. Sensor Fusion," *AIAA GN&C Conference*, USA, 2009.

[46] Indelman, V., Gurfil, P., Rivlin, E. and Rotstein, H., "Real-Time Vision-Aided Localization and Navigation Based on Three-View Geometry," preprint, 2010.

[47] Ma, Y., Huang, K., Vidal, R., Kosecka, J. and Sastry, S., " Rank Conditions on the Multiple-View Matrix," *International Journal of Computer Vision*, May 2004, pp. 115–137.