Building a Robust Implementation of Bearing-Only Inertial SLAM for a UAV

Mitch Bryson and Salah Sukkarieh
ARC Centre of Excellence in Autonomous Systems
Australian Centre for Field Robotics
University of Sydney
NSW 2006, Australia
{m.bryson,salah}@cas.edu.au

Abstract

This paper presents the on-going design and implementation of a robust inertial sensor based Simultaneous Localisation And Mapping (SLAM) algorithm for an Unmanned Aerial Vehicle (UAV) using bearing-only observations. A single colour vision camera is used to observe the terrain from which image points corresponding to features in the environment are extracted. The SLAM algorithm estimates the complete 6-DoF motion of the UAV along with the three-dimensional position of the features in the environment. An Extended Kalman Filter (EKF) approach is used where a technique of delayed initialisation is performed to initialise the 3D position of features from bearing-only observations. Data association is achieved using a multi-hypothesis innovation gate based on the spatial uncertainty of each feature. Results are presented by running the algorithm off-line using inertial sensor and vision data collected during a flight test of a UAV.

1 Introduction

In applications such as search and rescue, surveillance/picture compilation and planetary exploration (Braun et al., 2004), where a high degree of maneuverability/vehicle speed and large area coverage ability is required, the use of autonomous airborne vehicles has gained a great amount of interest. In some of these applications, particularly during exploration and surveillance tasks, the vehicle is often required to operate over unknown terrain or where navigation grade terrain maps are unavailable. Additionally, in applications such as shown in (Braun et al., 2004), navigation infrastructure such as the Global Positioning System (GPS) is unavailable as a localisation aid for an airborne vehicle. Un-aided Inertial Navigation Systems (INS) could be used, however the high-cost, weight and power requirements of the necessary unit plus the limited operational time due to the eventual growth in system errors is prohibitive to many projects. Instead, a low-cost INS can be used where the INS errors are corrected using a terrain sensor that performs online generation of a map of the previously unknown environment; in many respects similar to a real-time surveying task. This is the basis for the paradigm known as Simultaneous Localisation And Mapping (SLAM). Since the seminal work by Smith et. al. (Smith et al., 1990), there have been several demonstrated implementations of SLAM using land (Dissanayake et al., 2001) and underwater (Williams et al., 2001) vehicles where two-dimensional, horizontal localisation and mapping is performed. SLAM has also been demonstrated on a UAV (Kim and Sukkarieh, 2004; Kim and Sukkarieh, 2003; Jung and Lacroix, 2003) where the three-dimensional position of features and the 6-DoF motion of the vehicle are generated.

Most implementations of SLAM require observations of range and bearing to features from the platform requiring the use of sensors such as a laser or radar. The use of these sensors, particularly in airborne applications can be disadvantageous for several reasons; the weight, power and cost requirements of these sensors restrict their deployment to larger sized vehicles. Additionally these sensors emit signals into the environment which can be detected thus reducing their usefulness in applications such as stealth and surveillance. Removing the requirement for sensing the range to features in the environment and only sensing the bearings to a feature (hence bearing-only observations), greatly expands the domain of applications in which SLAM can be deployed. Passive sensors such vision and Infra-Red (IR) can be used providing more suitable implementations for missions involving small-sized or micro-aerial vehicles or in low-cost airborne systems such as in civilian, non-military applications.

Performing SLAM with bearing-only observations poses two main additional challenges to the range and bearing case. Firstly, a single bearing-only observation provides insufficient information alone to localise a feature in 3D. Instead observations from two sufficiently different poses are required. Secondly, data association is complicated by bearing-only observations. Since the 3D position of the feature is not known from a single observation, the χ^2 test (Neira and Tardos, 2001), commonly used for validation gating in tracking tasks, cannot be performed in the standard way. The bearing-only SLAM problem has been tackled before, most commonly using ground-based vehicles and there is a wide variety of solutions that have been proposed to overcome parts of the problem faced. In this paper we provide a review of these previous approaches (Section 2).

The contribution of this paper is the analysis of the implementation of a bearing-only SLAM algorithm applied to a high-speed aerial vehicle. The end goal of the work is to implement a real-time algorithm which provides sufficient localisation accuracy to be used as feedback for the vehicle's on-board control system while at the same time building an accurate point feature map. The focus of this paper is therefore on developing a robust bearing-only SLAM algorithm and then moving towards solving the issues that face real-time implementation.

In our implementation an Extended Kalman Filter (EKF) is used in which the position, velocity and attitude of the vehicle along with the 3D positions of point features in the environment are estimated. A delayed initialisation technique is used to store information from bearing-only observations until there exists two observations with a sufficient baseline from which to initialise the 3D position of the feature. Data association is tackled by creating multi-hypothesis distributions of the possible feature locations in 3D (i.e. along the line of sight of an observation). Subsequent observations of the same feature can be associated by matching the most likely hypotheses and culling the hypotheses that don't match. The result is a localisation and mapping system which can be deployed into a wider variety of environments, without the need for range observations to features or any prior terrain information.

In this paper we attempt to highlight some of the specific issues faced in our aerial implementation. Inertial sensors used are low-cost and contain significant noise and some time-varying biases. The aerial vehicle for which the SLAM algorithm is implemented can perform rapid manoeuvres and high-speed flight. The dynamic motion poses several challenges to the implementation; the sensor field of view moves rapidly from looking at the ground and the sky where the feature extraction algorithms must often deal with large deviations in image brightness. Additionally features are often in view of the sensor for only a small number of frames at a time. High-speed manoeuvres result in highly non-linear process and observation models, corrupting the linearising assumptions in the EKF and requiring prediction rates to run at high speed. Additionally some modes of the vehicle's flight mean that the sensor is not pointing towards the ground for a large amount of time where features observations can be generated. This can result in extended periods of time where the localisation of the vehicle relies on the inertial navigation system alone. SLAM on an aerial vehicle means that the vehicle motion must be estimated in 6-DoF and the position of features in 3D. This adds to the size of the state vector from most ground vehicle applications where the vehicle motion and feature map is estimated in 2D.

An implementation of SLAM on a high-speed aerial vehicle involving many of the same challenges but using both range and bearing observations was demonstrated in (Kim and Sukkarieh, 2003). The implementation uses a vision camera to compute the bearing to features along with an estimate of the range based on the known size of artificial targets that have been placed in the environment. Range information in this case is only available through the use of an artificial feature. Having range information available helps with the stability of the estimates in the EKF. The range and bearing SLAM algorithm is also simplified significantly; no vehicle pose or observation data needs to be stored, data association can be performed using the χ^2 test for validation gating using the 3D map information in a standard fashion. In this paper we highlight and propose solutions to the challenges faced with implementing the bearing-only case of inertial SLAM.

Figure 1 illustrates the elements of the bearing-only inertial SLAM algorithm with a guide to the relevant sections of the paper. Section 2 provides an overview of existing methods for solving the problems faced in bearing-only observations in regards to feature initialisation and data association and provides the reasoning for the approach taken for the aerial vehicle case in this paper. Section 3 examines a basic algorithm for feature extraction from the vision data. Section 4 details the estimation cycle of the SLAM algorithm with prediction stage based on inertial sensing and update stage using feature observations. The bearing-only feature initialisation process using stored observation and vehicle pose data is covered in Section 5. Section 6 overviews the data association process. Section 7 looks at how the components in Sections 4, 5 and 6 are integrated together and the computational complexity of the algorithms as a whole. Section 8 describes the physical system and sensors used to drive the SLAM algorithm. Results of the algorithm are shown in Section 9. Conclusions and future work are covered in Section 10.

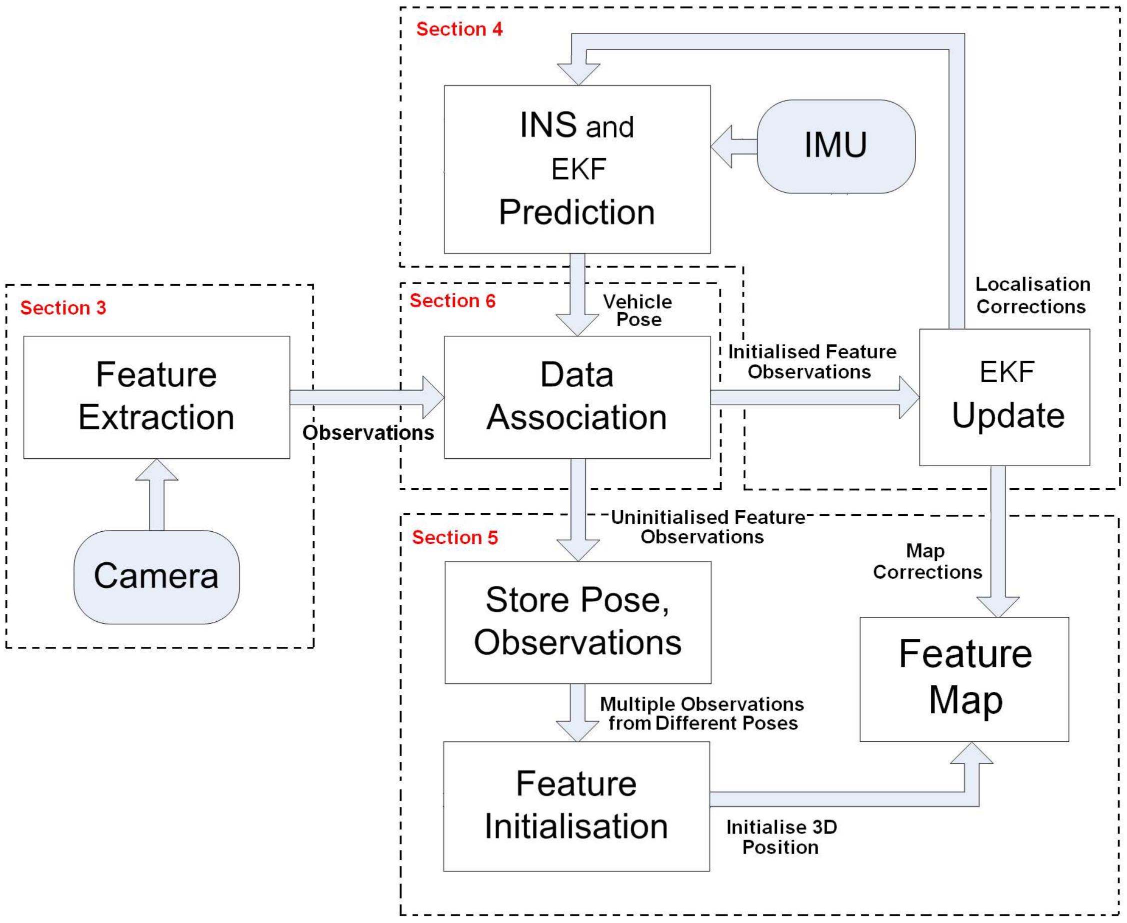

Figure 1: Overview of the bearing-only inertial SLAM algorithm and guide to relevant sections in the paper describing the feature extraction, Inertial Measuring Unit (IMU) and Inertial Navigation System (INS), Extended Kalman Filter (EKF), bearing-only feature initialisation and data association.

2 Literature Review

This section offers a background into existing solutions to aspects of the problems faced in implementing bearing-only SLAM and presents an overview to our approach for an aerial vehicle.

2.1 Overview of Previous Approaches

Once a well conditioned estimate of the feature position is available in SLAM, bearing-only tracking of the feature from subsequent observations can be tackled in the standard EKF framework. The issue however with implementing SLAM using a bearing-only sensor is that the initial 3D position of a feature cannot be determined from a single observation and thus the estimate of the feature location is ill-conditioned when represented as a Gaussian. Instead several measurements are required with sufficiently different baseline to determine the initial position accurately.

In the target tracking community, initialisation of 3D position from bearing observations has been tackled for example by representing the initial feature position as a non-Gaussian distribution using sums of Gaussians (Alspach and Sorenson, 1972) or with the use of particles (Gordon et al., 1993). These approaches are less popular for SLAM in which the initial feature position is correlated to the vehicle and the rest of the map, which results in high computational burden when correctly applying these representations to a high-dimensional state.

In (Davison, 2003) and (Fitzgibbons and Nebot, 2002) the authors tackle the initialisation problem by representing the position of the feature initially using particles. In both of these cases the authors maintain the particle distribution independently from the Gaussian probability distributions of estimates of the vehicle and the rest of the map. Observations made of the feature during the particle stage will however be correlated to one another through the vehicle errors and thus ignoring this coupling will result in a loss of information to the vehicle states and can lead to an inconsistent initialisation of the feature.

In (Kwok and Dissanayake, 2004) the authors use a multi-hypothesis filtering approach in which several hypotheses of the position of a landmark are created based along the line of sight of the first observation of a feature. Each of the hypotheses is then integrated into the filter and treated as a separate feature. Subsequent observations as the vehicle moves around the feature will eventually allow all but one of the hypotheses to be pruned out of the filter. Although computationally efficient, this approach losses information from observations made before initialisation. In (Sola et al., 2005) the authors similarly use a multi-hypothesis approach where the information from further observations of a feature before initialisation is transferred to each hypothesis using federated information sharing. In using this approach however there is no guarantee that estimates will be consistent due to the Kalman filter update of the incorrect hypotheses.

In (Montiel et al., 2006) the authors present an approach to the bearing-only SLAM problem that parameterises each features position into 6 states: the vehicle pose from which the feature was first observed, the angles to the line of sight of the observation and the inverse range to the feature. The authors claim that the use of the inverse range parameterisation (rather than range) results in better linearisation of the measurement equations and a better probabilistic representation of the feature position for when there are only a small number of observations made with very small baseline. There are two issues with this approach; firstly that inverse range is a good parameterisation for when the the baseline between observations is small but is not as good as the range representation when the baseline eventually becomes large. The second issue is that one must still define a mean value to the initial distribution of inverse range to the feature from a single observation. Setting this initial mean arbitrarily will result in filter inconsistency when the landmark's true initial inverse range differs.

Another approach to the bearing-only initialisation problem has been to store observations of a feature until sufficient observations from different angles are available to initialise the 3D position of the feature. Methods that store the data before initialisation are referred to as delayed approaches. In (Bailey, 2003) the author stores the vehicle pose and observation data in the state vector for a single observation and later uses constrained initialisation to compute the feature position when a second observation is available from a sufficiently different vehicle pose. Stored observations between the first and last observations can then be used in a batch update. By correctly maintaining correlations between the stored pose and the current vehicle pose, information from the first observation is transferred to the current vehicle pose estimate at initialisation in a consistent manner.

One question that arises in performing a delayed initialisation is in deciding when to initialise the feature, i.e. when are the stored observations sufficient. In (Bailey, 2003), the author uses the Kullback-Leibler distance to determine whether to initialise a feature by performing the initialisation and comparing the outcome with an approximation of a non-Gaussian initialisation. If the difference between the regular initialisation (using a Gaussian) and the non-Gaussian approximation (as measured by the Kullback-Leibler distance) is small then the feature is initialised into the map. This method is very computationally intensive due to the requirement for a numerical evaluation of the Kullback-Leibler distance.

2.2 Structure from Motion Approaches

There is a large body of work in the computer vision community on the structure from motion problem which is to reconstruct the motion or trajectory of a moving camera and also the structure of a stationary environment in which the camera moves. The structure from motion problem is closely related to the bearing-only SLAM problem where the methods used to solve each problem differ in the fact that bearing-only SLAM is mainly implemented as an online recursive estimation task and structure from motion is more commonly a batch reconstruction task (i.e. the map and trajectory are built when all of the data is collected and not online).

In (Triggs et al., 2000) the authors provide an excellent overview of techniques for bundle adjustment for solving the structure from motion problem with accurate and robust results. The idea of the bundle adjustment approach is to estimate the entire history of vehicle poses and the positions of features in the environment by running a batch update with all of the stored observations. The advantage of this approach is that estimates are well conditioned, the disadvantage being that the computational complexity of performing the batch update scales with the number of stored observations. This means that this approach becomes impractical for real-time localisation due to the growing computational cost.

(Deans, 2000) demonstrates a delayed approach to localisation and mapping with bearing-only observations by applying an adaptation of bundle adjustment. In their paper the authors propose a fusion between bundle adjustment and the application of recursive Kalman filters is required in order to provide a solution to SLAM which is both robust and consistent but also real-time applicable.

2.3 Overview of the Current Approach

In this paper we take a delayed approach to feature initialisation by storing observations and vehicle poses and recovering this information in a batch update step when sufficient base-line exists between two observations. In our approach the correlations between stored vehicle poses and the current vehicle pose along with the rest of the map is maintained in a consistent manner. It is our belief that initialising the feature in a consistent manner and recovering all of the information from observations made before initialisation is of most importance in SLAM, particularly in the aerial vehicle case where reliability of the navigation system is paramount.

The aim of our approach is to benefit from well conditioned SLAM estimates by storing observations and delaying our update until we can generate a well conditioned estimate of the 3D position of the feature. Once the 3D position of a feature is initialised, updating our localisation and mapping estimates from future observations can be performed in a incremental fashion using the EKF. This removes the growing computational burden of storing observations once a feature is initialised.

Our approach to data association is based on matching observations made by the feature extraction algorithm to known features in the map by the spatial knowledge of the feature's position in the environment using innovation gating. The aim of the data association method is to be applicable to situations in which features in the environment have little or no visual properties to distinguish from one another (an example is given by the features shown in Section 3). It should be noted that visual distinction could be used to suppliment our approach in the case where the features used could be discriminated from one another based on visual properties (such as colour or texture).

3 Extracting Features from Image Data

The SLAM algorithms described in the following sections of the paper rely on observations of point features in the environment. In order to facilitate the estimation algorithm development, 1x1m white plastic squares (see Figure 2) have been placed into the environment to act as artificial features. These artificial features have a higher visual intensity compared to the background terrain and are thus relatively easy to distinguish in the vision data. In most environments there exist natural features that when viewed from the air can have their positions classified by a point (such as the centroid of bushes and rocks). The visual appearance of these features can be encoded into the feature extraction process and model-based matching techniques (Nixon and Aguado, 2001) can be used to extract them from the image data. Alternatively, when localisation is the main priority, features may be points in the sensor data that are distinct and easily recognisable and techniques like SIFT features (Lowe, 2004) can be used. The algorithms to extract natural features from vision data are however beyond the scope of this paper.

Figure 3 shows a sample image from the on-board camera taken while in flight. The white plastic square feature extraction process finds the normalised intensity of each pixel in the image, applies an intensity threshold and finds which pixels lie above the threshold (white

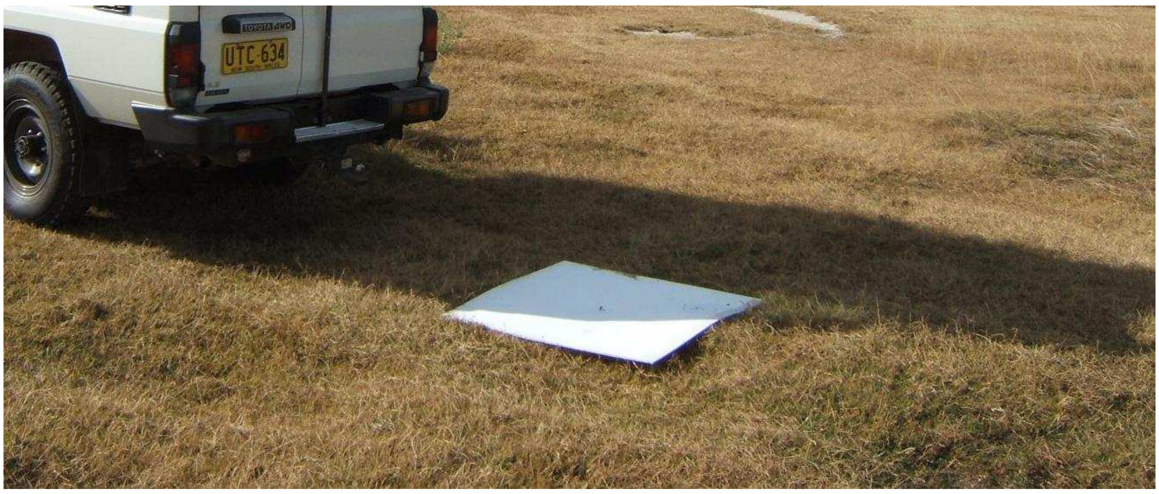
Figure 2: SLAM features: 1x1m white plastic squares have been placed in the environment to act as features for the SLAM algorithm.

areas in image on right-hand side of Figure 3). Areas of interconnected pixels are then placed into groups where each group is a potential feature. The suitability of each pixel group is assessed by considering the pixel count, size and shape of each group. Pixel groups are firstly checked to see if they are roughly square or circular in shape (determined by maximum pixel distance from the group's centroid divided by the number of pixels) and have a pixel count below a threshold (computed for a 1x1 meter feature, looking straight down at 100m altitude). The intensity of the pixels surrounding the group is then averaged and the average is checked to see that it is below a given threshold so as to ensure that the pixel group stands out from the background intensity. Each pixel group that passes all of the manually tuned thresholds is declared a feature, and an observation is generated as the centroid of the group in the image (yellow circles in right-hand side of Figure 3).

4 The Inertial SLAM Algorithm

The inertial SLAM algorithm, as shown in (Kim and Sukkarieh, 2003), is formulated using an EKF in which map feature locations and the vehicle's position, velocity and attitude are estimated using relative observations between the vehicle and each feature. Figure 4 illustrates the relevant relationships between the frames of reference in the SLAM algorithm.

4.1 Process Model

The estimated state vector $\hat{\mathbf{x}}(k)$ contains the three-dimensional vehicle position $(\mathbf{p}^n)$, velocity $(\mathbf{v}^n)$ and Euler angles $(\mathbf{\Psi}^n = [\phi, \theta, \psi])$ and the three-dimensional locations of each feature $(\mathbf{m}_i^n)$ in the environment where $i = 1, ..., N_{feats}$, N_{feats} is the number of features and the superscript n indicates the vector is referenced in a local-level navigation frame. The vehicle operates over a small area (relative to the curvature of the Earth) and thus it is assumed that the local-level frame does not rotate as the vehicle moves. The starting point for the

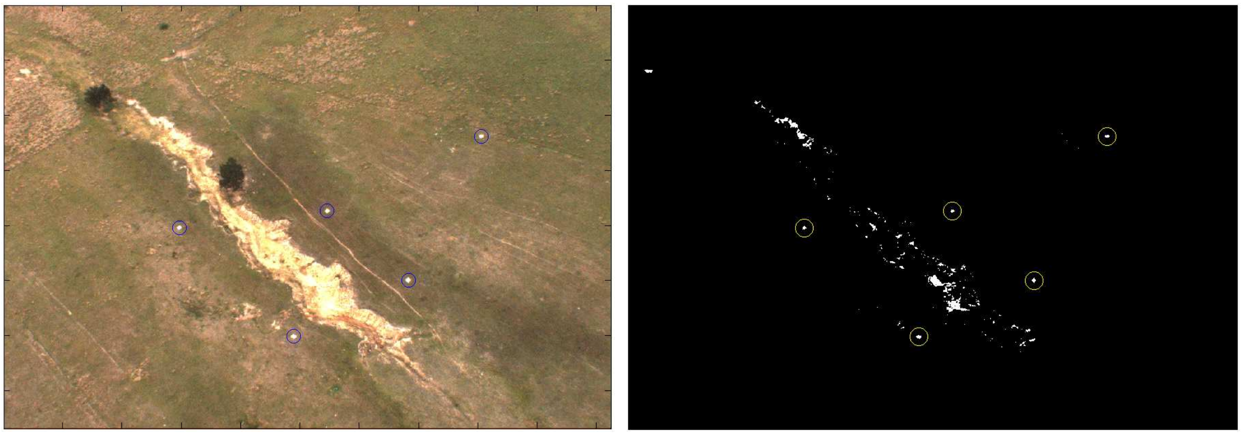
Figure 3: Feature extraction: Left, a sample vision frame from the vehicle's on-board camera with extracted features (white targets) highlighted by dark circles. Right, pixels that are above the intensity threshold of the colour image are shown in white. Features are extracted (light circles) as appropriately sized groups of connected pixels that are likely to correspond to white targets.

local-level frame is at the initial horizontal position of the vehicle and is vertically referenced to mean sea level. The state estimate $\hat{\mathbf{x}}(k)$ is predicted forward in time from $\hat{\mathbf{x}}(k-1)$ via the process model:

$$\hat{\mathbf{x}}(k) = \mathbf{F}(\hat{\mathbf{x}}(k-1), \mathbf{u}(k), k) + \mathbf{w}(k) \tag{1}$$

where $\mathbf{F}(.,.,k)$ is the non-linear state transition function at time k, $\mathbf{u}(k)$ is the system input at time k, and $\mathbf{w}(k)$ is uncorrelated, zero-mean vehicle process noise errors of covariance $\mathbf{Q}$. The process model is the standard 6-DoF inertial navigation equations which predict the position, velocity and attitude of the vehicle. An inertial-frame mechanization (Titterton and Weston, 1997) is implemented:

$$\begin{bmatrix} \mathbf{p}^n(k) \\ \mathbf{v}^n(k) \\ \mathbf{\Psi}^n(k) \end{bmatrix} = \begin{bmatrix} \mathbf{p}^n(k-1) + \mathbf{v}^n(k)\Delta t \\ \mathbf{v}^n(k-1) + [\mathbf{C}_b^n(k-1)\mathbf{f}^b(k) + \mathbf{g}^n]\Delta t \\ \mathbf{\Psi}^n(k-1) + \mathbf{E}_b^n(k-1)\omega^b(k)\Delta t \end{bmatrix} \tag{2}$$

where $\mathbf{f}^b$ and ω^b are the body-frame referenced vehicle accelerations and rotation rates which are provided by the Inertial Measuring Unit (IMU) on the vehicle and $\mathbf{g}^n$ is the acceleration due to gravity. The inertial-frame mechanisation simplifies the INS equations by ignoring small effects such as the Earth's rotation which have negligible effects when operating over a small area relative to the Earth's surface and for a small amount of time. Other mechanisations of the equations can be used when the vehicle operates over a very large area or for long periods of time. Euler angles are used to represent the attitude of the platform rather than quaternions in order to reduce the number of parameters in the estimator. In our case, the vehicle's pitch angle is constrained so as to avoid the Euler angle singularity at 90^o pitch, however our algorithm could be easily changed to implement quaternions in the event that the platform motion is not constrained in such a way. The vehicle body-frame (b) lies at the center of and has its axes aligned with the IMU, where the x-axis points out the aircraft's nose, y-axis out the right wing and z-axis down. The direction cosine matrix $\mathbf{C}_b^n$ and rotation rate transformation matrix $\mathbf{E}_b^n$ between the body and navigation frames are

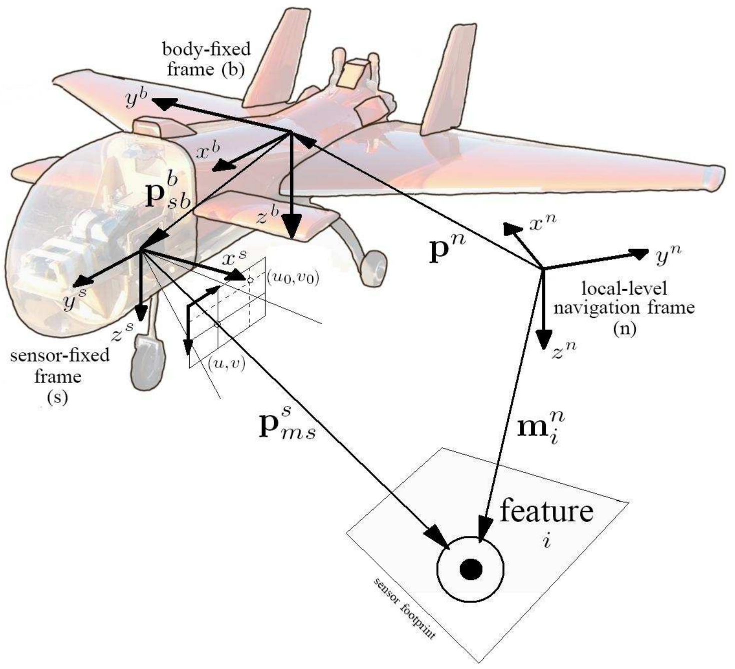

Figure 4: Vectors and frames of reference in the inertial SLAM algorithm: The local-level navigation frame (n), body-fixed frame (b) and sensor-fixed frame (s) and the relationship between the vehicle position (p^n), feature position (m^n) and sensor observation are shown.

given as:

$$\mathbf{C}^n_b = \begin{bmatrix} c_\psi c_\theta & c_\psi s_\theta s_\phi - s_\psi c_\phi & c_\psi s_\theta c_\phi + s_\psi s_\phi \\ s_\psi c_\theta & s_\psi s_\theta s_\phi + c_\psi c_\phi & s_\psi s_\theta c_\phi - c_\psi s_\phi \\ -s_\theta & c_\theta s_\phi & c_\theta c_\phi \end{bmatrix} \tag{3}$$

$$\mathbf{E}^n_b = \begin{bmatrix} 1 & s_\phi t_\theta & c_\phi t_\theta \\ 0 & c_\phi & -s_\phi \\ 0 & s_\phi sec_\theta & c_\phi sec_\theta \end{bmatrix} \tag{4}$$

where $s_{(.)}$, $c_{(.)}$ and $t_{(.)}$ represent sin(.), cos(.) and tan(.) respectively. Feature locations are assumed to be stationary and thus the process model for the position of the i^{th} feature is given as:

$$\mathbf{m}^n_i(k) = \mathbf{m}^n_i(k-1) \tag{5}$$

4.2 Observation Model

An on-board vision camera makes relative bearing observations $\mathbf{z}_i(k)$ to features within the sensor frame. The observations are related to the estimated states via:

$$\mathbf{z}_i(k) = \mathbf{H}_i(\mathbf{p}^n(k), \mathbf{\Psi}^n(k), \mathbf{m}^n_i(k), k) + \mathbf{v}(k) \tag{6}$$

where $\mathbf{H}_i(.,.,.,k)$ is a function of the feature location, vehicle position and Euler angles and $\mathbf{v}(k)$ is uncorrelated, zero-mean observation noise errors of covariance $\mathbf{R}$. There are two forms that can be used to represent a bearing-only observation. The observation $\mathbf{z}_i(k)$ can be represented by azimuth (φ_i) and elevation angles (ϑ_i):

$$\mathbf{z}_{i,ang}(k) = \begin{bmatrix} \varphi_i \\ \vartheta_i \end{bmatrix} = \begin{bmatrix} \tan^{-1}\left(\frac{y^s}{x^s}\right) \\ \tan^{-1}\left(\frac{z^s}{\sqrt{(x^s)^2+(y^s)^2}}\right) \end{bmatrix} \tag{7}$$

where x_s, y_s and z_s are the cartesian co-ordinates of $\mathbf{p}^s_{ms}$, the relative position of the feature w.r.t the sensor, measured in the sensor frame and $\mathbf{R}_{ang}$ is the angular noise covariance. For vision camera, the observation is better represented as pixels in the image of the camera, using a pinhole camera model:

$$\mathbf{z}_{i,pix}(k) = \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} f_u(\frac{y^s}{x^s}) + u_0 \\ f_v(\frac{z^s}{x^s}) + v_0 \end{bmatrix} \tag{8}$$

where u_0, v_0, f_u and f_v are calibration parameters for the camera and the pixel noise covariance is $\mathbf{R}_{pix}$. The relationship between the pixel co-ordinates and the azimuth and elevation angles is given by Equation 9:

$$\begin{bmatrix} \varphi \\ \vartheta \end{bmatrix} = \begin{bmatrix} \tan^{-1}\frac{(u-u_0)}{f_u} \\ \tan^{-1}(\frac{(v-v_0)}{f_v}\cos\varphi) \end{bmatrix} \tag{9}$$

The pixel noise is assumed to be additive and Gaussian and the angular noise covariance can be approximated as Gaussian:

$$\mathbf{R}_{ang} = \mathbf{R}_{pix} \begin{bmatrix} \frac{1}{f_u^2} & 0 \\ 0 & \frac{1}{f_v^2} \end{bmatrix} \tag{10}$$

Each of the different representations have their advantages in the algorithm. Observation in pixel form are used in the EKF update stage as explained in section 4.3 and observations in azimuth and elevation form are used in the data association process in section 6.

The position of the feature in the sensor frame in cartesian form $\mathbf{p}^s_{ms}$ is related to the vehicle and feature positions in the local-level navigation frame and the attitude of the vehicle using Equation 11:

$$\mathbf{p}^s_{ms} = \mathbf{C}^s_b\mathbf{C}^b_n[\mathbf{m}^n_i - \mathbf{p}^n - \mathbf{C}^n_b\mathbf{p}^b_{sb}] \tag{11}$$

where C^s_b is the transformation matrix from the body frame to sensor frame and $\mathbf{p}^b_{sb}$ is the sensor offset from the body frame (offset of the vision sensor w.r.t the IMU), measured in the body frame.

4.2.1 Camera Calibration and Offset

The camera calibration parameters u_0, v_0, f_u and f_v, the offset of the camera from the IMU (i.e. the relative position of the sensor frame w.r.t the body frame) and the relative direction cosine matrix between the body and sensor frames are computed before the flight in an offline calibration procedure. The camera calibration procedure is first performed using the MATLAB camera calibration toolbox (Bouguet, 2006) in which a sequence of camera frames are taken from different poses showing images of a calibration checkerboard of known dimensions. From these images, the corners of each square in the checker board are extracted manually (by clicking on them in the images) and a batch linear-least squares estimator is applied to estimate each pose of the camera relative to the board and the camera calibration parameters u_0, v_0, f_u and f_v.

Once the camera calibration parameters have been computed, the camera is connected to the aircraft body along with the IMU and a second procedure is performed to compute the relative transformation between camera and IMU. The relative position between the two ($\mathbf{p}^b_{sb}$) is measured by hand along the axes of the IMU. The direction cosine matrix between the body frame and sensor frame (C^s_b) is firstly computed by measuring it's Euler angle components by hand and then refined by manually tuning the Euler angles of the direction cosine matrix using an image taken by the camera of the calibration board place in a known position and orientation w.r.t the aircraft body.

4.3 Estimation Process

The estimation process is recursive and is broken into two steps:

4.3.1 Prediction

The vehicle position, velocity and attitude are predicted forward in time using (1) and (2) with data supplied by the inertial sensors. The state covariance $\mathbf{P}$ is propagated forward via:

$$\mathbf{P}(k|k-1) = \nabla\mathbf{F}_x(k)\mathbf{P}(k-1|k-1)\nabla\mathbf{F}_x^T(k) + \nabla\mathbf{F}_w(k)\mathbf{Q}\nabla\mathbf{F}_w^T(k) \tag{12}$$

where $\nabla\mathbf{F}_x$ and $\nabla\mathbf{F}_w$ are the jacobians of the state transition function w.r.t the state vector $\hat{\mathbf{x}}(k)$ and the noise input $\mathbf{w}(k)$ respectively.

4.3.2 Update

Assuming that we have already initialised the three-dimensional position of a feature, the state estimate is updated from further observations via:

$$\hat{\mathbf{x}}(k|k) = \hat{\mathbf{x}}(k|k-1) + \mathbf{W}(k)\nu(k) \tag{13}$$

where the gain matrix $\mathbf{W}(k)$ and innovation $\nu(k)$ are calculated as:

$$\begin{aligned} \nu(k) &= \mathbf{z}_i(k) - \mathbf{H}_i(\hat{\mathbf{x}}(k|k-1)) && (14)\\ \mathbf{W}(k) &= \mathbf{P}(k|k-1)\nabla\mathbf{H}_x^T(k)\mathbf{S}^{-1}(k) && (15)\\ \mathbf{S}(k) &= \nabla\mathbf{H}_x(k)\mathbf{P}(k|k-1)\nabla\mathbf{H}_x^T(k) + \mathbf{R} && (16) \end{aligned}$$

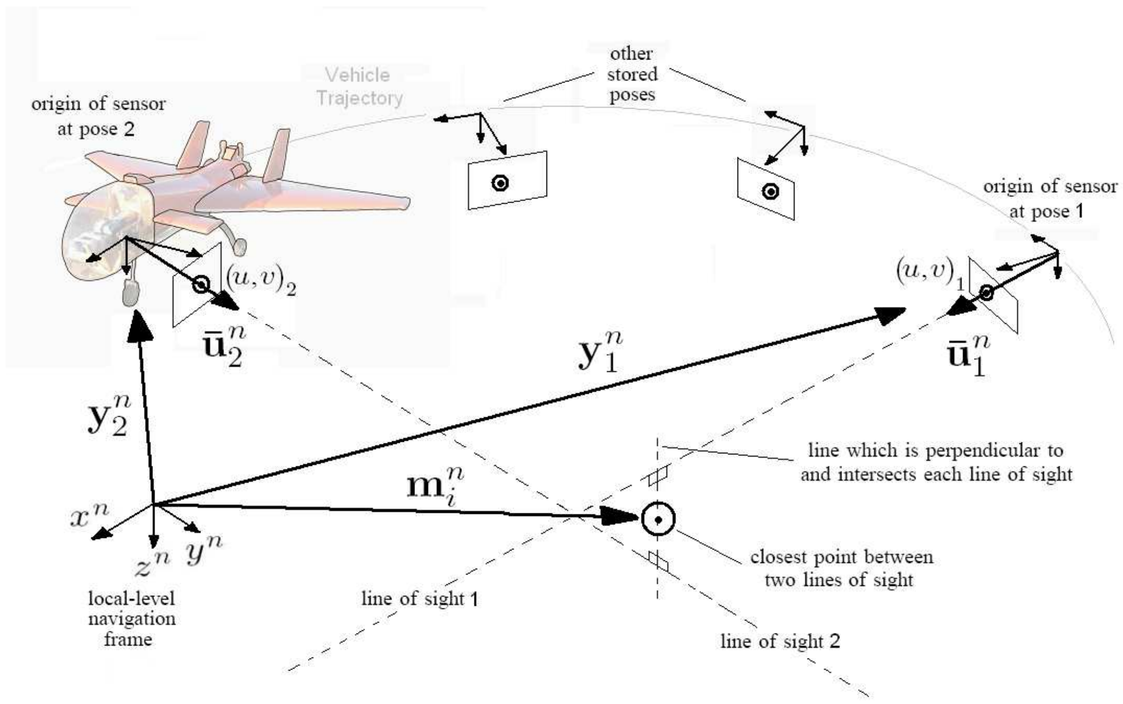

Figure 5: Initialising the 3D position of a feature: The two observations of the feature with the largest angular baseline between them is used to initialise the feature. The feature position is the closest point between the the two lines of sight of each observation and stored vehicle pose.

where $\nabla\mathbf{H}_x(k)$ is the jacobian of the observation function w.r.t the predicted state vector $\hat{\mathbf{x}}(k|k-1)$. The state covariance $\mathbf{P}(k|k)$ after the observation is updated via:

$$\mathbf{P}(k|k) = \mathbf{P}(k|k-1) - \mathbf{W}(k)\mathbf{S}(k)\mathbf{W}^T(k) \tag{17}$$

5 Initialisation of Feature Positions from Bearing-only Observations

As mentioned in Section 4, a single bearing-only observation is insufficient to initialise the 3D position of a feature into the SLAM filter with Gaussian uncertainty. In the following subsection, we outline a method for delayed initialisation of a feature into the filter by using stored observations and vehicle pose information. Figure 5 illustrates the relevant frames and vectors for the 3D initialisation of a feature.

5.1 Storing Feature Observations and Vehicle Pose Information

When an observation of an un-initialised feature is made, the current bearing-only observation is stored and the SLAM state vector and covariance matrix are augmented to include the current vehicle pose (3 position states and 3 Euler angle states):

$$\hat{\mathbf{x}}_v = \begin{bmatrix} \mathbf{p}^n(k) \\ \mathbf{v}^n(k) \\ \mathbf{\Psi}^n(k) \end{bmatrix}, \hat{\mathbf{x}}_p = \begin{bmatrix} \mathbf{p}^n(k) \\ \mathbf{\Psi}^n(k) \end{bmatrix} \tag{18}$$

$$\hat{\mathbf{x}}_{aug} = \begin{bmatrix} \hat{\mathbf{x}}_v(k) \\ \mathbf{m}^n(k) \\ \hat{\mathbf{x}}_p(k) \end{bmatrix} \tag{19}$$

$$\mathbf{P}_{aug}(k) = \begin{bmatrix} \mathbf{P}_{vv} & \mathbf{P}_{vm} & \mathbf{P}_{vp} \\ \mathbf{P}_{mv} & \mathbf{P}_{mm} & \mathbf{P}_{mp} \\ \mathbf{P}_{pv} & \mathbf{P}_{pm} & \mathbf{P}_{pp} \end{bmatrix} \tag{20}$$

$\hat{\mathbf{x}}_v$ is concatenation of the vehicle position, velocity and attitude states where $\hat{\mathbf{x}}_p$ is the concatenation of the vehicle position and attitude (i.e. the vehicle pose states) at the time of the observation. $\hat{\mathbf{x}}_{aug}$ is the augmented state vector which comprises of the vehicle states ($\hat{\mathbf{x}}_v$), the 3D positions of all of the map features ($\mathbf{m}^n$) and the added vehicle pose states ($\hat{\mathbf{x}}_p$).

The covariance term $\mathbf{P}_{pp}$ (covariance of the pose states) is derived by taking the position and attitude covariance matrix subblocks from within $\mathbf{P}_{vv}$ (since these states have the same value and the same covariance as the current vehicle position and attitude). Similarly the covariance subblock $\mathbf{P}_{pm}$ is taken from the existing cross correlations between the current vehicle states and map states (i.e. subblocks of $\mathbf{P}_{vm}$ corresponding just to position and attitude). $\mathbf{P}_{pv}$ is the cross correlation between the all of the current vehicle states (i.e. position, attitude and velocity) and the current pose states. This is thus taken from subblocks of $\mathbf{P}_{vv}$ itself since there are states in common (position and attitude) and thus parts of the added vehicle pose states ($\hat{\mathbf{x}}_p$) are completely correlated to the current vehicle state ($\hat{\mathbf{x}}_v$). As the process model of the vehicle comes into play and the filter moves forward in time, the correlations between the stored pose and the new time vehicle states (i.e. $\hat{\mathbf{x}}_v(k+1)$) will become less correlated (due to the process noise on $\hat{\mathbf{x}}_v(k+1)$).

The observation (co-ordinates of the feature in the image plane) is stored separately from the EKF state vector.

5.2 Initialising a 3D Feature Position Estimate

When it is decided to initialise the 3D position of a feature into the map, the two stored observations of the feature which are separated by the largest angle are used to create an initial estimate of the feature position. Each bearing-only observation can be represented by a 3D point in space $\mathbf{y}^n$ from where the observation was made (at the origin of the sensor) along with a unit vector $\bar{\mathbf{u}}^n$ pointing along the line of sight of the observation, thus:

$$\mathbf{y}^n = \mathbf{p}^n + C_b^n \mathbf{p}_{sb}^b \tag{21}$$

$$\bar{\mathbf{u}}^n = C_b^n C_s^b \bar{\mathbf{p}}_{ms}^s \tag{22}$$

where $\bar{\mathbf{x}}$ indicates the unit vector of a vector $\mathbf{x}$. $\mathbf{p}^n$ and C_b^n are determined from the stored pose data associated to each observation and $\bar{\mathbf{p}}_{ms}^s$ is determined from the observation data itself using Equation 9 to convert the pixel observation to azimuth and elevation angles and Equation 23 to convert to a unit vector:

$$\bar{\mathbf{p}}^{s}_{ms} = \begin{bmatrix} \cos(\varphi_i)\cos(\vartheta_i) \\ \sin(\varphi_i)\cos(\vartheta_i) \\ \sin(\vartheta_i) \end{bmatrix} \tag{23}$$

The lines of sight generated by each observation should intersect at one point in 3D space corresponding to the feature location. Since the observations and stored vehicle pose information is noisy, the lines of sight will generally not intersect perfectly. Instead, the initial feature position is computed as the closest point between the two lines for each observation:

$$\begin{aligned} \mathbf{m}^n_i &= \mathbf{G}(\mathbf{p}^n_1, \mathbf{p}^n_2, \mathbf{\Psi}^n_1, \mathbf{\Psi}^n_2, \mathbf{z}_1, \mathbf{z}_2) \\ &= \frac{1}{2}(\mathbf{y}^n_1 + \mathbf{y}^n_2 + p_1.\bar{\mathbf{u}}^n_1 + p_2.\bar{\mathbf{u}}^n_2) \end{aligned} \tag{24}$$

$$p_1 = \frac{((\mathbf{y}^n_2 - \mathbf{y}^n_1) \times \bar{\mathbf{u}}^n_2) \cdot (\bar{\mathbf{u}}^n_1 \times \bar{\mathbf{u}}^n_2)}{|\bar{\mathbf{u}}^n_1 \times \bar{\mathbf{u}}^n_2|^2} \tag{25}$$

$$p_2 = \frac{((\mathbf{y}^n_1 - \mathbf{y}^n_2) \times \bar{\mathbf{u}}^n_1) \cdot (\bar{\mathbf{u}}^n_2 \times \bar{\mathbf{u}}^n_1)}{|\bar{\mathbf{u}}^n_2 \times \bar{\mathbf{u}}^n_1|^2} \tag{26}$$

In the event that there is a large discrepancy between the two lines (i.e. the minimum distance between the closest two points, one on each line, is larger than a threshold), the observations may be incorrect possibly due to a miss-association in the data association or if the feature is moving for some reason, and thus the observations are discarded and the feature is not initialised yet. Provided there is no large discrepancy between the lines, the state vector and covariance matrix in the SLAM filter are then augmented to include the estimate of the new feature:

$$\hat{\mathbf{x}}_{aug}(k) = \begin{bmatrix} \hat{\mathbf{x}}(k) \\ \mathbf{m}^n_i(k) \end{bmatrix} \tag{27}$$

$$\mathbf{P}_{aug}(k) = \begin{bmatrix} \mathbf{I} & 0 \\ \nabla\mathbf{G}_p & \nabla\mathbf{G}_z \end{bmatrix} \begin{bmatrix} \mathbf{P}(k) & 0 \\ 0 & \mathbf{R}_{2x2} \end{bmatrix} \begin{bmatrix} \mathbf{I} & 0 \\ \nabla\mathbf{G}_p & \nabla\mathbf{G}_z \end{bmatrix}^T \tag{28}$$

where $\nabla\mathbf{G}_p$ and $\nabla\mathbf{G}_z$ are the jacobians of the initialization function $\mathbf{G}(.)$ w.r.t the pose states $(\mathbf{p}^n_1, \mathbf{p}^n_2, \mathbf{\Psi}^n_1, \mathbf{\Psi}^n_2)$ and the observations $(\mathbf{z}_1, \mathbf{z}_2)$ respectively and $\mathbf{R}_{2x2}$ is:

$$\mathbf{R}_{2x2} = \begin{bmatrix} \mathbf{R} & 0 \\ 0 & \mathbf{R} \end{bmatrix} \tag{29}$$

5.3 Recovering the Information from Remaining Stored Observations

Once two observations have been used to initialise the 3D position of the feature into the filter, the remaining stored observations $(\mathbf{z}_1, \mathbf{z}_2, ..., \mathbf{z}_j)$ are run through a batch EKF update. The update corrects not only the current feature being initialised but also the other features in the map and the current vehicle state estimates. The updated state vector is calculated:

$$\hat{\mathbf{x}}(k|k) = \hat{\mathbf{x}}(k|k-1) + \mathbf{W}(k)\nu(k) \tag{30}$$

Where the innovation $\nu(k)$ is composed by all of the stored observations for the feature:

$$\nu(k) = \begin{bmatrix} \mathbf{z}_{1,pix} - \mathbf{H}(\mathbf{p}_1^n, \mathbf{\Psi}_1^n, \mathbf{m}_i^n) \\ \mathbf{z}_{2,pix} - \mathbf{H}(\mathbf{p}_2^n, \mathbf{\Psi}_2^n, \mathbf{m}_i^n) \\ \vdots \\ \mathbf{z}_{j,pix} - \mathbf{H}(\mathbf{p}_j^n, \mathbf{\Psi}_j^n, \mathbf{m}_i^n) \end{bmatrix} \tag{31}$$

The gain matrix $\mathbf{W}(k)$ and innovation covariance $\mathbf{S}(k)$ are calculated as:

$$\mathbf{W}(k) = \mathbf{P}(k|k-1)\nabla\mathbf{H}_x^T(k)\mathbf{S}^{-1}(k) \tag{32}$$

$$\mathbf{S}(k) = \nabla\mathbf{H}_x(k)\mathbf{P}(k|k-1)\nabla\mathbf{H}_x^T(k) + \mathbf{R} \tag{33}$$

where $\nabla\mathbf{H}_x(k)$ is the composition of jacobians of the observation function for each stored observation w.r.t the predicted state vector $\hat{\mathbf{x}}(k|k-1)$. The state covariance $\mathbf{P}(k|k)$ after the observation is updated via:

$$\mathbf{P}(k|k) = \mathbf{P}(k|k-1) - \mathbf{W}(k)\mathbf{S}(k)\mathbf{W}^T(k) \tag{34}$$

Once the update has been completed, pose states that no longer have any associated stored observations are removed from the state vector and their corresponding rows and columns removed from the covariance matrix.

5.4 Deciding when to Initialise a Feature

A remaining issue with the delayed initialisation method is the question of when the initialisation should be made. Initialising a feature too early (i.e. by not using enough observations or observations with insufficient separating angle) can result in inconsistency as the true probability distribution of the feature location is not well represented by a Gaussian. The disadvantage with over-delaying the initialisation is that the uncertainty in the vehicle states continues to grow before the initialisation. As the uncertainty in the current vehicle state grows, linearisation errors can effect the consistency of the filter. We wish to initialise and recover the information as quickly as possible to limit the effect the inconsistency can have. Additionally it may be desired to quickly recover the stored information which contributes towards the accuracy of the vehicle localisation estimates which may be used as feedback for the control of the vehicle. Another upper limit on deciding how long to delay the initialisation is driven by reducing the increased computational burden imposed by adding stored observations to the state vector.

It was found that initialising a feature too early had a greater potential to cause inconsistency in the filter due to incorrect triangulation of the features position. For this reason, rather than performing the initialisation as soon as possible by testing the conditioning of the estimate such as shown in (Bailey, 2003), it was found that a heuristic such as setting the minimum angle between observations necessary to initialise a feature to a conservative threshold (in our case 40 degrees) provides a practical way of deciding when to initialise.

6 Data Association

Data association is the process of matching observations of features from the camera (which are generally provided as 2D points in the image which are not necessarily distinct from any other point) with the estimated 3D position of the feature within the map. The aim of this section is to develop a method of data association that does not rely on any visual information about a feature from the image data.

6.1 Data Association Matching Test

The validity of potential associations between observations and features is assessed using the Mahalanobis distance (γ) (Neira and Tardos, 2001) in the sensor space (azimuth and elevation):

$$\gamma = \nu^T \mathbf{S}^{-1} \nu \tag{35}$$

where ν and $\mathbf{S}$ are the innovation and innovation covariance for the observation and a given feature in the map. In the data association, we compare the Mahalanobis distance in the space of azimuth and elevation angles rather than with pixels. The reason for this is that when a feature's 3D position is projected into the sensor space using pixel co-ordinates, the projection becomes more ill-conditioned the closer the features lies at an angle of 90^o to the pointing direction of the camera. Thus the projected uncertainty of features near the edge of the camera's field of view can result in the feature being incorrectly associated if pixels are used.

When checking the association of a given observation with the initialised features in the map, γ is calculated for each initialised feature using Equations 14 and 16. Matchings that fall within a defined threshold of γ corresponding to a 95% level of confidence are considered acceptable.

6.2 Associating Observations of Un-Initialised Features

Issues arise when attempting to find a data association test that can be performed for un-initialised features. Since the exact 3D position of the feature is not-known, we cannot consistently calculate the innovation or innovation covariance of the feature. Instead, from only one or a small number of observations with a small baseline, the observation could lie anywhere in 3D space along the line of sight of the observation.

In order to associate observations of features that have not yet been initialised into the 3D map we will define a multi-hypothesis of Gaussian distributions of the possible 3D locations of the feature along the line of sight vector for our first observation of the feature.

6.2.1 Starting a New Feature

When an observation is made in the image that cannot be associated to any other previously seen feature, initialised or un-initialised, it is assumed that this observation has come

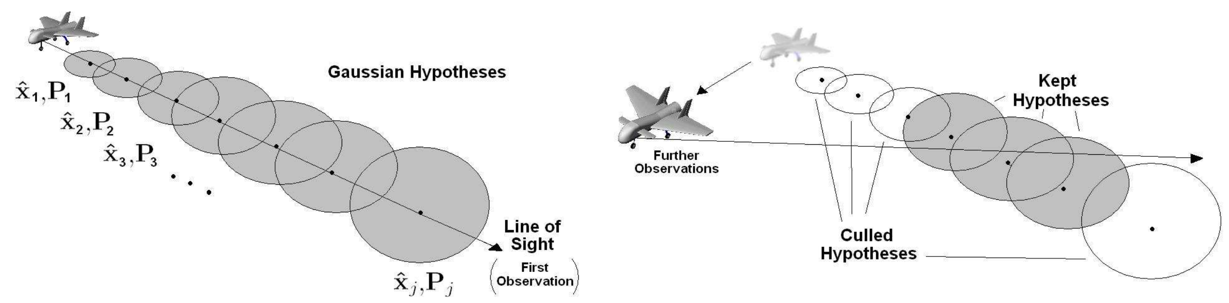

Figure 6: Data association of observations to un-initialised features: When a feature is seen for the first time, a set of hypotheses for the 3D position of the feature is generated at equal range increments along the line of sight (left). Future observations are checked for matches to any of the hypotheses. When a match is made to one of the hypotheses, the remaining hypotheses that don't match are culled (right).

from a new feature that we have not seen before. We start by storing the observation and augmenting the EKF state vector with the current vehicle pose (see Section 5.1).

From our single observation we create a set of equally weighted hypotheses for where the feature could lie in 3D space along the line of sight. The mean ($\hat{\mathbf{x}}_j$) and covariance ($\mathbf{P}_j$) for each hypothesis is calculated for several different values of range (r_j) in equal increments from an expected minimum and maximum sensor range as shown in the left of Figure 6 using Equations 36 and 37:

$$\begin{aligned}\hat{\mathbf{x}}_j &= \mathbf{G}(\mathbf{p}^n(k), \mathbf{\Psi}^n(k), \mathbf{z}_i(k), r_j) \\ &= \mathbf{p}^n + C_b^n \mathbf{p}_{sb}^b + r_j.(C_b^n C_s^b \mathbf{p}_{ms}^s) \qquad (36) \\ \mathbf{P}_j &= \nabla\mathbf{G}_v \mathbf{P}_{vv} \nabla\mathbf{G}_v^T + \nabla\mathbf{G}_z \mathbf{R}_{ang} \nabla\mathbf{G}_z^T \qquad (37)\end{aligned}$$

where $\mathbf{p}_{ms}^s$ is calculated from the observation data using Equation 23 and $\nabla\mathbf{G}_v, \nabla\mathbf{G}_z$ are the jacobians of the function $\mathbf{G}(.)$ w.r.t vehicle states and the observation and range data respectively. The number of hypotheses used and the maximum and minimum range and thus the spacing between the hypotheses depends on the application. For the UAV, 20 hypotheses equally spaced in increments of 20 meters from a minimum range of 50 meters to a maximum range of 450 meters is used. The noise in range for each hypothesis is set 3σ = 20 meters so as to provide a large level of overlap between each hypothesis. These values were chosen based on the vehicle altitude and the maximum range of the feature extraction process. In the case where features that are further away can be extracted, different schemes from hypothesis placing can be used (i.e. rather than linearly spaced hypotheses, we could space the hypotheses based on a geometric series (Sola et al., 2005) and extend the range uncertainty in each hypothesis).

A record of the multi-hypothesis distribution is maintained separately from the state vector and is used only to assist in associating future observations of the feature.

6.2.2 Associating Future Observations and Maintaining Feature Hypotheses

Since each hypothesis is Gaussian with a mean defined in 3D space, the innovation and innovation covariance can be calculated for each hypotheses for each un-initialised feature using:

$$\begin{aligned} \nu(k) &= \mathbf{z},\mathbf{ang}_i(k) - \mathbf{H}(\mathbf{p}^n(k), \mathbf{\Psi}^n(k), \hat{\mathbf{x}}_j) & (38) \\ \mathbf{S}(k) &= \nabla\mathbf{H}_x(k)\mathbf{P}_{hyp}(k)\nabla\mathbf{H}_x^T(k) + \mathbf{R}_{ang} & (39) \end{aligned}$$

$$\mathbf{P}_{hyp}(k) = \begin{bmatrix} \mathbf{P}_{pp} & 0 \\ 0 & \mathbf{P}_j \end{bmatrix} \qquad (40)$$

where $\mathbf{P}_{pp}$ is the covariance sub blocks of the current vehicle position and attitude states. We make the approximation that the current vehicle state and the hypothesis are uncorrelated in order to simplify the data association process. For a given observation and for each hypothesis in a given un-initialised feature, γ is calculated using Equations 35, 38 and 39. If the value of gamma is below the threshold corresponding to a 95% confidence for at least one of the hypotheses, then the observation is matched to this uninitialised feature.

In order to simplify the association of un-initialised features, when an association is made between an observation and one of the hypotheses for a given feature, all other hypotheses of this feature for which the observation does not match are culled from the set of hypotheses from which to associate future observations. As the vehicle moves around an un-initialised feature, the number of hypotheses gradually drops until only one hypothesis matches, the one that is closest to the true 3D feature location. The right side of Figure 6 illustrates this process.

6.3 Data Association Procedure

The data association process as a whole is illustrated in Figure 7. Each time observations from the feature extraction process are received, we begin by using Equation 35 to evaluate the potential matching between each observation and each of the 3D initialised features. Observations that match 3D initialised features are associated and sent on to the SLAM filter to be updated. In the event of multiple features matching a single observation, the matching with the lowest value of γ will be accepted.

The remaining observations are tested for matches with each of the hypotheses for each un-initialised feature. Observations that match with at least one hypothesis of an un-initialised feature are associated to this feature. The observation itself is stored and the vehicle pose at the current time is then added to the state vector (see Section 5.1). In the event of multiple un-initialised features matching a single observation, all matchings to this observation are rejected.

For each remaining observation not matched to an initialised or un-initialsed feature, a new set of hypotheses is created (see Section 6.2.1).

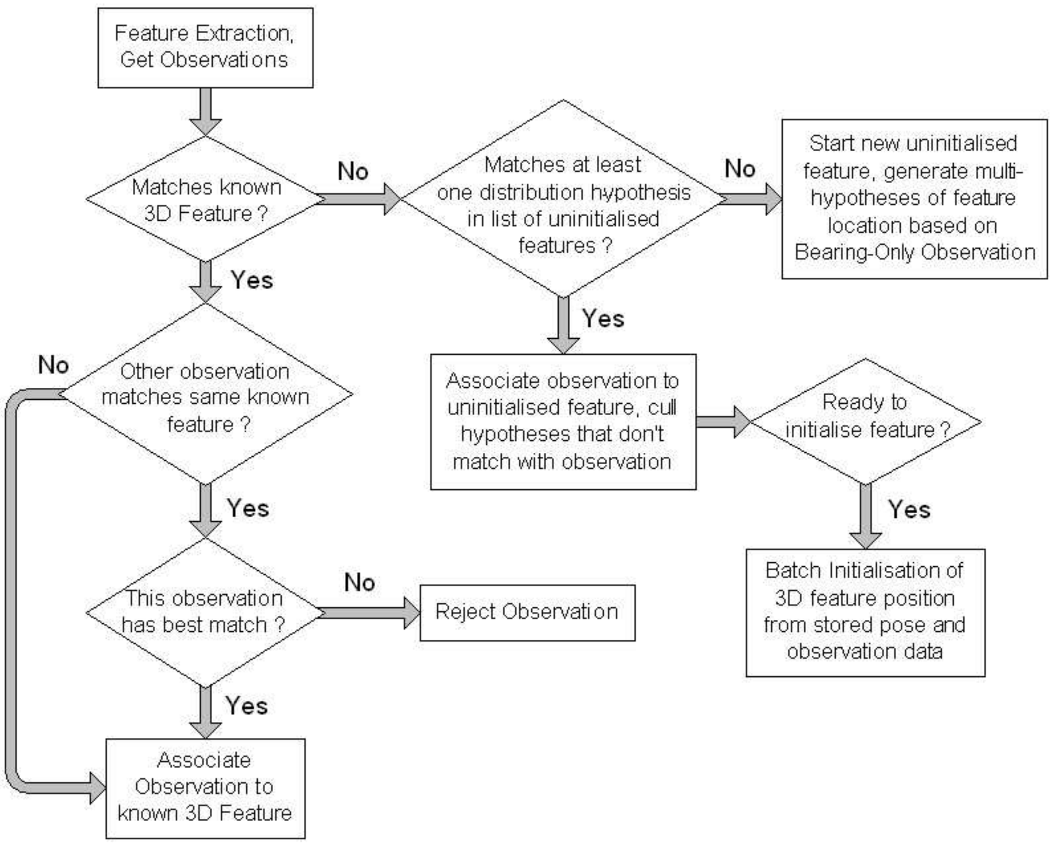
Feature Extraction, Get Observations
Matches known 3D Feature ?
No
Matches at least one distribution hypothesis in list of uninitialised features ?
No
Start new uninitialised feature, generate multi-hypotheses of feature location based on Bearing-Only Observation
Yes
Yes
No
Other observation matches same known feature ?
Associate observation to uninitialised feature, cull hypotheses that don't match with observation
Ready to initialise feature ?
Yes
Yes
This observation has best match ?
No
Reject Observation
Batch Initialisation of 3D feature position from stored pose and observation data
Yes
Associate Observation to known 3D Feature

Figure 7: Data association procedure for initialised and un-initialised Features: For each feature extracted from the vision data, we first test for matches to initialised features. If no matches exist then matches are tested for each hypothesis for each un-initialised feature. If no matches are present then a new, un-initialised feature is created with a multi-hypothesis distribution along the line of sight of the observation.

7 Bearing-Only SLAM Algorithm Integration

This section analyses the integration of the algorithmic components of bearing-only SLAM described in the above sections, concentrating on the algorithmic flow, timing and computational complexity and storage required by the equations.

7.1 Algorithm Flow

Figure 8 illustrates the algorithmic progression of the bearing-only SLAM algorithm as a whole. Acceleration and rotation rates of the vehicle are read from the IMU at 400 Hz and used to drive the inertial navigation equations in a 400 Hz INS loop. In the INS we have opted to use a computationally efficient first order integration step (which is sufficient to capture the dynamics of the platform when running at 400 Hz). Higher-order integration techniques could be used if a faster, more highly dynamic platform was used or if the inertial output was at a lower rate, provided that the computational power was available. Due to the

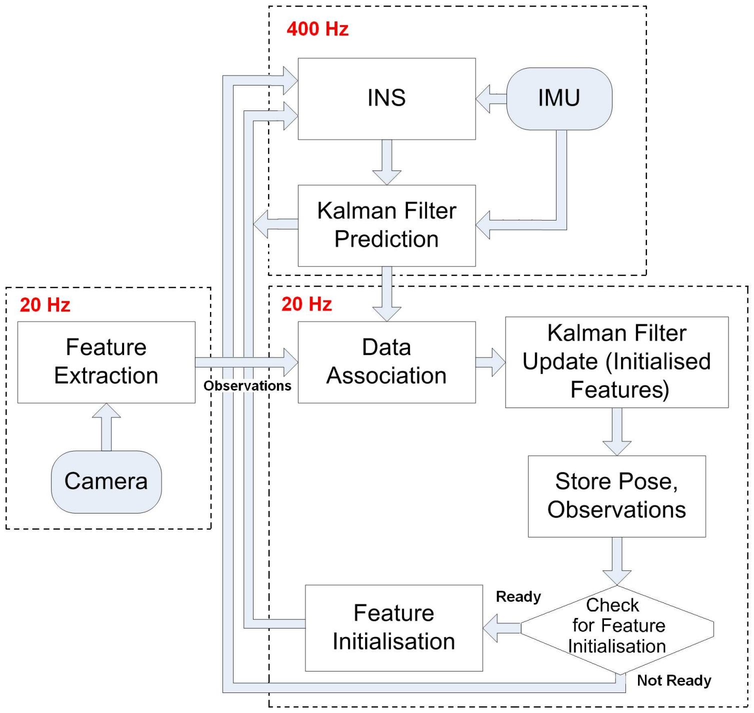

Figure 8: Bearing-only SLAM algorithm timing and progression: The Inertial Navigation System (INS) and EKF prediction step is run at 400 Hz. Features are extracted from the camera data in a separate process at 20 Hz. Data association, EKF update, storing pose data and initialising new features into the map is all performed at 20 Hz when observations are available from the camera.

linearisation of system matricies in Equation 12, the prediction step in the EKF needs to be run at a rate sufficiently high to capture the non-linearities in the motion of the vehicle. The results presented show the prediction rate running at 400 Hz alongside the INS. Frames from the vision camera are captured and feature observations extracted at 20 Hz. The feature extraction algorithm is external to the SLAM algorithm and can be run in a separate process, where observations are communicated to the SLAM algorithm when available.

When a set of observations, corresponding to a single frame from the camera are received, we begin by running the data association process on the observations. For observations of initialised features the EKF update stage is run. Any remaining observations (relating to un-initialised features) are stored and the current vehicle pose is augmented into the state vector. We then initialise any features into the map which are ready. All of the above tasks are being performed at 20 Hz while observations of features are being made.

The feature extraction processing is decoupled from the SLAM algorithm but can have a significant effect when there are large delays in the image processing. When more sophisticated feature extraction algorithms are used, these delays can become substantial. One way to tackle this delay could be to add a vehicle pose to the state vector at the time of vision capture using the same method as in Section 5.1. When the feature extraction has finished and the observation has been processed into the data association, we can use the observation to update the stored pose rather than the current vehicle state. This method would use many of the concepts developed in this paper for delayed initialisation, but the actual implementation is beyond the current scope and is the interest of future work.

7.2 Algorithm Computational Complexity and Storage

In this section we analyse the computational requirements of the system and examine some of the methods used to reduce the computational complexity in order to make the system more suitable for real-time implementation.

There are three main storage requirements of the algorithm:

1. **EKF State Vector and Covariance Matrix** - The EKF state vector $\mathbf{x}$ is comprised of 9 vehicle states, $3N_{feats}$ initialised feature states and $6N_{poses}$ stored pose states where N_{feats} is the number of initialised features in the map and N_{poses} is the number of stored vehicle poses. The total size of the state vector is thus $N_{state} = 9 + 3N_{feats} + 6N_{poses}$. The covariance matrix $\mathbf{P}$ of the EKF has N_{state}^2 elements however is a symmetric matrix meaning only the upper or lower triangle need be stored.
2. **Stored Observations** - Stored observations are stored separately to the EKF state vector and take up $2N_{storeobs}$ elements where $N_{storeobs}$ is the number of stored observations.
3. **Multi-Hypotheses Distributions** - The multi-hypotheses distributions stored separately from the EKF and used to assist in data association initially take up $12N_{hyp}$ elements per un-initialised feature where N_{hyp} is the number of hypotheses generated when a feature is seen for the very first time. This storage could be reduced to $9N_{hyp}$ by only storing the upper triangle of the covariance. The number of hypotheses N_{hyp} begins at 20 hypotheses and quickly drops as more observations of the feature are made and hypotheses are culled.

The following points indicate the main areas of computation required by the algorithm:

1. **EKF Prediction** - The computational load induced by the prediction stage is due to the matrix multiplications in Equation 12. Standard prediction algorithm optimisation techniques (see (Guivant and Nebot, 2001) for details) can be used such that the complexity of a prediction step is $\sim \mathbf{O}(N_{state})$, that is in the order of or proportional to the number of states in the state vector.
2. **EKF Update** - Standard EKF update algorithm optimisation techniques (see (Guivant and Nebot, 2001) for details) can be used such that the complexity of an update step is $\sim \mathbf{O}(N_{state}^2)$ for each observations.
3. **Data Association** - Data association involves computing the innovation and innovation covariance for each initialised feature in the map and each hypotheses for each un-initialised feature. This must be performed every time a set of observations is received. The upper limit on the computational complexity is therefore

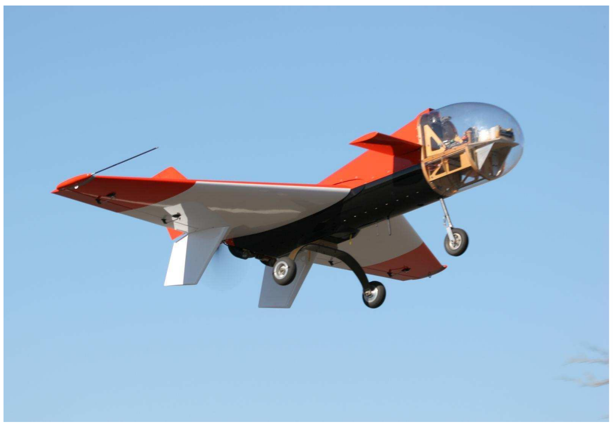
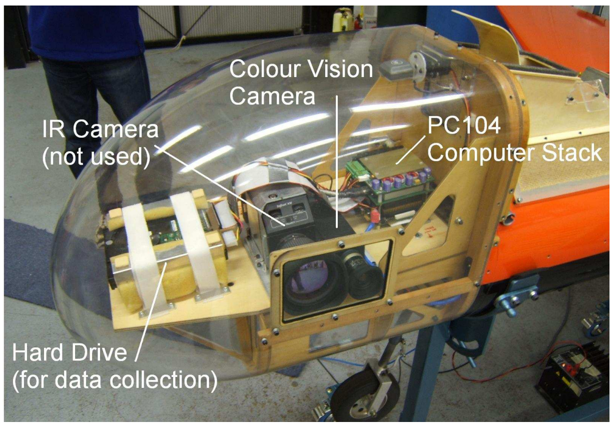

Figure 9: Left, the Brumby MkIII UAV, weighing 40kg with a wing span of 2.8 meters, capable of carrying a payload of 13.5 kg and flying at 100kts. Right, sideways mounted colour camera and PC104 computer stack used for vision processing.

$\sim \mathbf{O}(N_{feats} + N_{hyp}N_{uninit})$ where N_{uninit} is the number of uninitialised features. The computation required is reduced as the number of hypotheses for different features is reduced from subsequent observations.

4. **Feature Initialisation** - Feature initialisation basically involves a EKF update stage but using several stored observations at one time. The computational complexity is thus $\sim \mathbf{O}(N_{storeobs}N_{state}^2)$ where $N_{storeobs}$ corresponds to the number of stored observations of the feature being initialised. This step in the algorithm usually creates the largest computational burden. This burden can be mitigated to some degree by trading off the information in the update by discarding some of the stored observations.

Obviously the computation required by the algorithm increases as the number of features in the map and the number of stored vehicle poses increases. Reduction of the computational burden of the algorithm can be achieved through several means. Firstly, map management techniques can be applied to limit the number of feature in the map (Dissanayake et al., 2002). Secondly similar techniques could be used to reduce the number of stored poses used to initialise the 3D position of a feature. Simple schemes could be used (i.e. only store every second observation and pose of an un-initialised feature) or alternatively only store observation which contribute the most information towards the feature initialisation (see (Stevens et al., 1995) for a technique that selects the observations that will maximise the mutual information gain in the estimated states).

A real-time implementation of the whole algorithm has not yet been developed however this is the topic of future work and is discussed in the conclusions in Section 10.

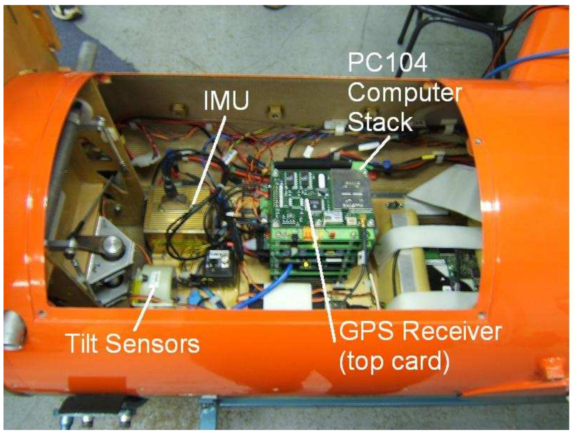

Figure 10: Left, inside the fuselage of the Brumby MkIII UAV, low-cost IMU, GPS receiver and PC-104 computer stack used for autonomous navigation and control of the vehicle. Right, farmland environment that the vehicle operates over.

8 Experimental Setup

This section provides an overview of the flight vehicle, inertial and vision sensors used and the environment the vehicle operates within.

8.1 Brumby MkIII Flight Vehicle

The Brumby MkIII is a small sized, delta-wing UAV weighing approximately 40kg with a wing span of 2.8 meters (see left side of Figure 9). The vehicle can carry a payload of 13.5kg and is capable at flying at speeds of 100kts. The vehicle is mainly used as a research testbed to demonstrate real-time algorithms for Decentralised Data Fusion (DDF) (Nettleton, 2003) and co-operative control strategies (Cole et al., 2005) involving multiple vehicles.

The flight vehicle has an autonomous flight control system (Kim et al., 2003) that follows a fixed path of orbits around features of interest on the ground. The on-board navigation system uses differentially corrected GPS to aid an on-board IMU which provides about 1-2 meter positioning accuracy and about 1-2 degrees orientation accuracy when GPS is available. Errors occur in the current navigation system when the number of GPS satellites visible to the GPS antenna on the aircraft drops below four and a GPS fix from the on-board receiver cannot be achieved. When GPS outages are infrequent, the results of the on-board INS-GPS navigation system can be used as a reasonable approximation of the true position, velocity and attitude of the vehicle and are thus used in the results to compare the accuracy of the SLAM solution.

8.2 Sensors and Hardware

The vehicle carries a colour vision camera mounted sideways w.r.t the vehicle body which is used to observe the terrain below when the vehicle banks to turn (see right side of Figure

Vision Camera		**IMU**	
Sampling Rate	20 Hz	Sampling Rate	400 Hz
FOV	$30^o \times 22^o$	Accelerometer Noise	$0.05 m/s^2$ (1σ)
Resolution	1024 x 768 pixels	Gyro Noise	$0.05 deg/s$ (1σ)
Angular Resolution	0.0285 deg	Accelerometer Bias Stability	$\pm 0.05 m/s^2$
Pointing Direction	Sideways (left)	Gyro Bias Stability	$\pm 0.05 deg/s$

Table 1: Sensor Specifications: Digital colour vision camera and IMU.

9). A visual fix on an object on the ground can be achieved when the vehicle maintains a steady turn or orbit around the object. Data from the colour vision camera is used to extract artificial features that have been placed in the environment. On-board the vehicle is a low-cost IMU which contains three accelerometers and three gyros mounted in a tri-axial configuration (see left side of Figure 10).

The flight vehicle has one on-board PC-104 computer which is currently used for logging the vision camera frames and a second PC-104 used to run navigation and control algorithms using data from the IMU and on-board GPS receiver. Frames from the vision camera are logged at 20Hz and data from the IMU is logged at 400Hz.

8.3 Environment

The operating environment is a 2x2km area of land in rural New South Wales, Australia. The terrain is mostly farmland, covered by grass and shrubs (see right side of Figure 10). Features visible in the vision sensor data include natural features such as trees, rocks and a river and man-made features such as roads, fences, dams and a four-wheel drive. Several 1x1m white plastic squares are placed in the environment to act as artificial features for the SLAM algorithm. The position of each of these features is surveyed using differentially corrected GPS to an accuracy of 50cm. The surveyed positions are then used to examine the SLAM mapping accuracy (not actually used in the SLAM algorithm itself). The vehicle flies at an altitude of about 100-200m above the terrain.

8.4 SLAM Algorithm Initialisation

In our experimental results we assume that the initial attitude and velocity of the vehicle is known or given to a certain accuracy. This assumption is consistent with two different scenarios. In the first scenario the aircraft operates a navigation system such as GPS-aided INS and the SLAM algorithm begins to run at a point in time where all GPS signals are lost. In this case the initial velocity and attitude of the vehicle are provided by the last navigation output. The second scenario involves the vehicle stationary on the ground ready for takeoff, but with no previous navigation system to provide the initial vehicle velocity and attitude. In this scenario the roll and pitch (tilt) angles of the platform can be deduced from the IMU accelerometer readings and the vehicle velocity is assumed to be zero by the navigation system. The heading angle of the vehicle is more difficult to initialise and will often require the use of magnetometers. The following experimental results are all derived under the assumption of the first scenario, due to the difficulty of finding and tracking features during the aircraft's takeoff. In this case the initial covariance of the vehicle position state is derived from the covariance of the last GPS-aided INS reading with a 0.5m/s velocity standard deviation and a 1deg attitude standard deviation.

Before the flight, an alignment procedure is used to compute the values of the IMU biases while on the ground. This procedure uses a set of fluidic tilt-sensors and the assumption that the vehicle is stationary during alignment in order to calibrate the biases in the accelerometers and gyros. These values are then used to correct the IMU for the duration of the flight while the SLAM algorithm is running. This alignment procedure has proved to be consistent over many flight trials.

9 Results

This section analyses the results of the SLAM algorithm when applied to the system described in Section 8 using logged flight data where the algorithm is run in an off-line mode. There are three sets of results presented. The first section presents results of the algorithm for a small single orbit trajectory (25 second flight time) and focuses on the feature initialisation process. The second section of the results examines the SLAM algorithms running on a small sized vehicle trajectory (90 second flight time). The small sized trajectory is such that the features are close to one another and feature observations are continuously being made as the vehicle spends most of the trajectory banking to turn such that the camera is pointed at the terrain below and to the side of the vehicle. The third section of the results examines the SLAM algorithm results where the trajectory of the vehicle covers a larger area (200 second flight time). In this case the vehicle spends large segments of time flying straight and level where the sideways mounted camera on the vehicle points to the horizon and feature observation are not made. As can be seen in the results this causes large periods of time where the errors in the vehicle localisation states grows (due to the unconstrained INS) and can result in degraded SLAM algorithm performance.

9.1 Feature Initialisation

In this section we highlight the results of the feature initialisation method in the SLAM algorithm by presenting results of the SLAM algorithm running over a single orbit in which several features are initialised into the map.

Figure 11 shows six captured vision frames with observations, the projected positions of 3D initialised features and multiple hypotheses of un-initialised features overlaid on the image. As features are seen for the first time, an array of hypotheses represented by the green ellipses is projected into the image. Further observations of the feature begin to cull unmatched hypotheses before finally the angular threshold for different observations of a feature is reached and the 3D position of the feature is initialised into the SLAM map.

Figure 12 shows both the SLAM estimated and GPS-aided INS trajectory of the vehicle plus the estimated locations of the initialised features and their 10σ confidence bounds. The

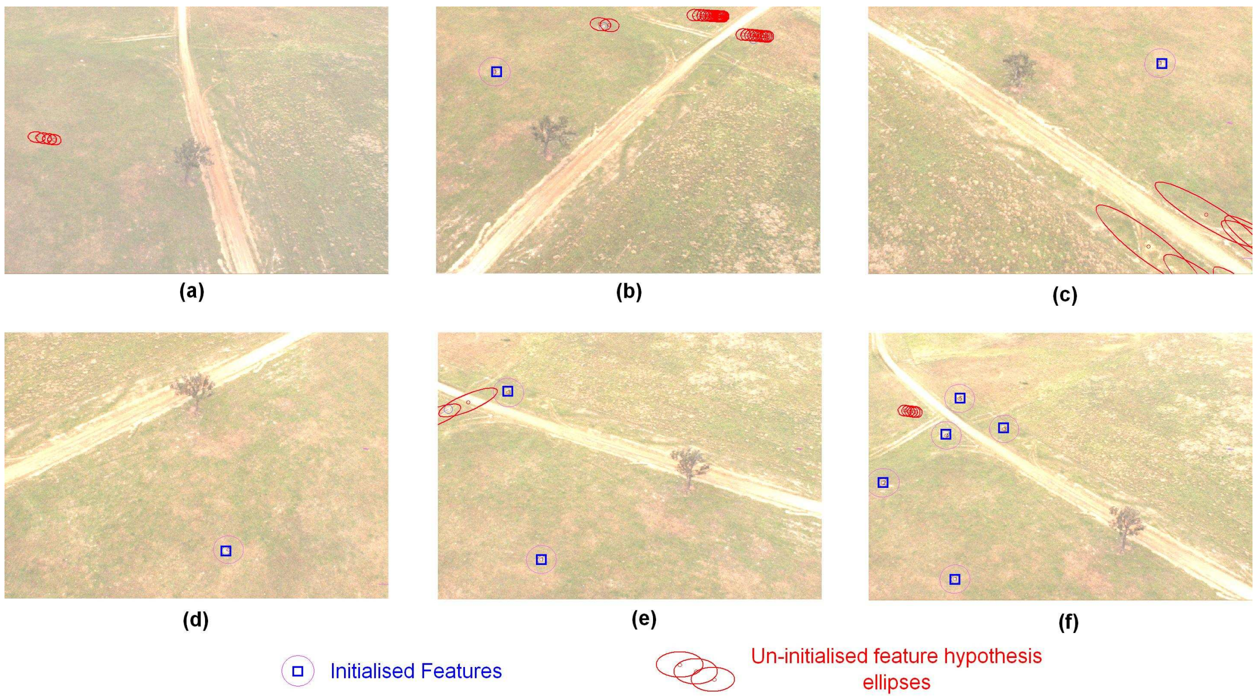

Figure 11: Six captured vision frames with the projected positions of 3D initialised features (squares) and multiple hypotheses of un-initialised features (ellipses) overlaid on the image.

vehicle makes sufficient observations from varying vehicle poses to initialise the 3D positions of five features into the map from the single orbit trajectory. From the figure there appears to be a bias in the estimated position of the map features when compared to the surveyed positions. This highlights a general issue of the SLAM algorithm; only a relative map of the environment can be constructed due to the lack of any global information from which to place the generated map within (such as knowing the global position of one of the features or the global position of the vehicle using GPS). Thus any errors in the initial position of the vehicle found using GPS carry through to form a bias in the generated map. The relative position of each feature w.r.t each other feature however can be consistently estimated without any biases.

Figure 13 shows the number of states stored in the EKF state vector (comprising of the nine vehicle states, three states for each map feature and six states for each stored pose) as a function of time. The state vector dimension initially increases as vehicle pose data is stored for each observation made of a feature. At about 954 seconds, sufficient observations with sufficient baseline are made of the first feature, which is initialsed into the 3D map. Once this occurs, stored pose data that is no longer required is discarded from the state vector resulting in the drop in the number of states. The state vector continues to rise as observations are stored for more features, until which time as sufficient observations are stored and the 3D position of these features is initialised into the map. Occasionally observations are made of features on the horizon that are too far away for the vehicle to gain a sufficient baseline to initialise the 3D position of the feature. If no further observations of these features are made

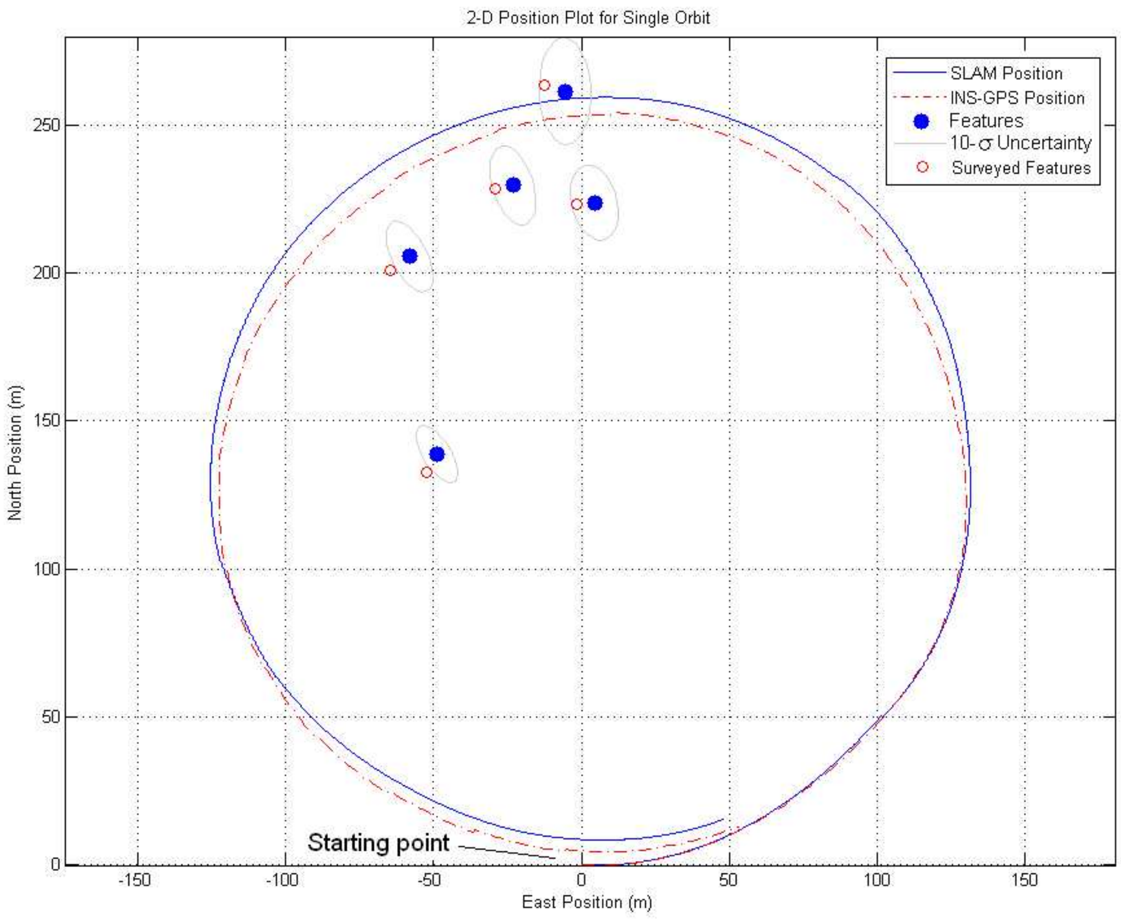

Figure 12: SLAM results over a small, single orbit trajectory: The dark line indicates the 2D trajectory of the vehicle estimated by the SLAM navigation system, the dotted line indicates the trajectory estimated by the GPS-aided INS for reference. The dark points and light ellipses represent the estimated locations of features plus associated 10σ as estimated by the SLAM filter. The circles represent the surveyed position of the artificial features used for comparison with the SLAM map.

within a certain time limit, then these features are considered to be either too far away or to be noise in the feature extraction process and thus their corresponding stored pose data is dropped from the filter, in order to reduce the computational load of an excessive number of filter states. This can be seen by the drop in filter states at the 961 and 962 second marks.

Figure 14 shows the standard deviation of the absolute position error of the vehicle taken from the covariance matrix of the SLAM filter. The small drops in the standard deviation of error at the 954, 957 and 960 second marks correspond to the times of the initialisation of the first three features. This is due to the feature initialisation process recovering the information towards the vehicle localisation states that is contained in the stored vehicle poses and observations of a feature before initialisation. This information is recoverable at the time of intialisation as the correlations between stored pose states and the current position, velocity and attitude of the vehicle are maintained in the filter covariance matrix.

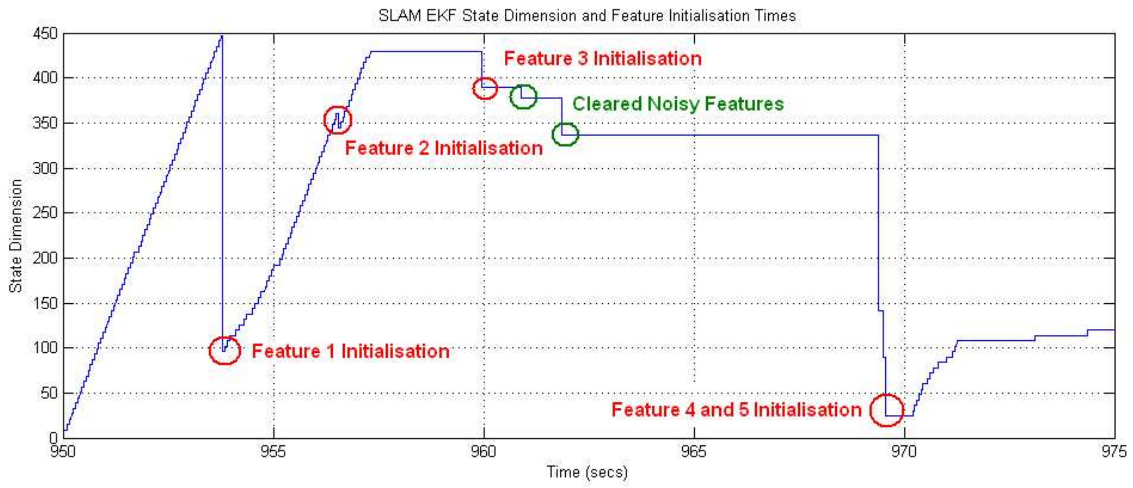

Figure 13: State dimension of the SLAM filter for a single orbit: The dimension of the state vector changes as vehicle pose information is added for stored observations and removed when the observations are used to initialise the 3D position of the feature.

The drops in the covariance at the 967 and 969 seconds marks are due to features leaving the field of view of the camera where no EKF updates are made for a small section of time (due to disturbances in the vehicle bank angle). The jumps correspond to when the feature is reacquired.

9.2 Localisation and Mapping Results: Small Trajectory

In this section we analyse the results for a small sized trajectory run using the logged sensor data.

The trajectory of the vehicle and the position of point features in the map is shown in Figure 15 with results from both the GPS-aided INS solution and the un-aided INS shown for comparison. It should be noted that the trajectory shown is the estimated position of the vehicle at the time in which the vehicle was at that position and not the estimated position from any stored pose at that point. For this reason, jumps in the trajectory can be seen for example when the vehicle closes the loop. The trajectory consists of three orbits all closely spaced such that the time the vehicle spends flying straight and level, where the camera points off at the horizon and does not receive feature observations is minimised. During the orbit segments of the trajectory (any non-straight line segments) the vehicle banks to turn at a roll angle of about 50 degrees during which time the camera points towards the ground below and to the side of the vehicle. During this time feature extractions are made from the camera data and these observations are used to constrain the drift in the errors in the inertial navigation solution. It can be seen that the position trajectory computed using the un-aided INS quickly drifts from the INS-GPS solution, even over the reasonably short time span of the trajectory.

Figure 16 shows the size of the SLAM EKF state vector as a function of time. The number of

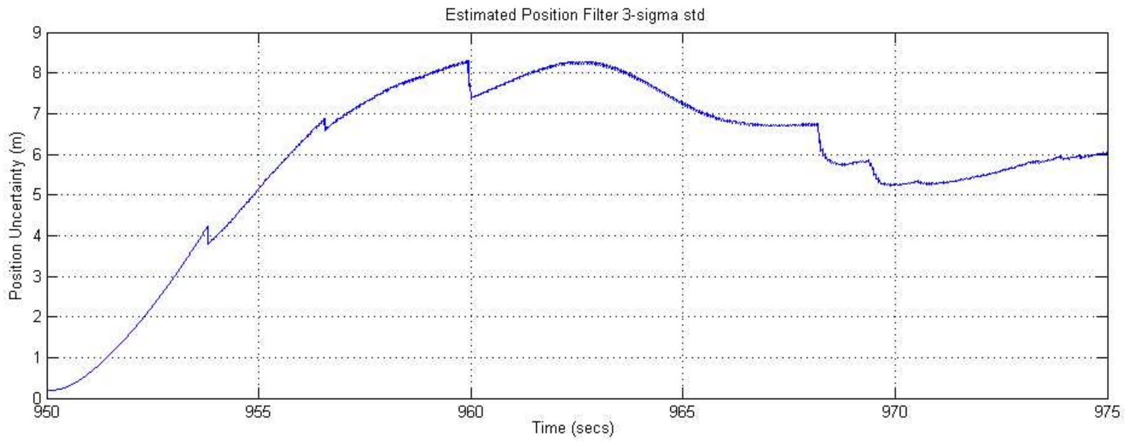

Figure 14: Absolute position filter covariance for a single orbit: The sharp drops in the covariance correspond to feature initialisations, at which time information about the position of the vehicle is recovered from the stored observations.

states fluctuates due to the storing of pose data and the removing of pose data with feature initialisations. As the vehicle explores it expands the number of features in the map and the size of the state dimension on average slowly grows.

Figure 17 illustrates the difference between the SLAM and INS-GPS computed positions. Under the assumption that the INS-GPS solution gives a good approximation to the true position of the vehicle, we can think of this difference as just containing the errors in the SLAM solution. Figure 17 also shows the expected 3σ uncertainty bounds on the position solution as reported by the SLAM EKF. It can be seen that the error lies in between the uncertainty bounds for the large majority of the trajectory.

Figure 18 shows the errors in the SLAM filter velocity measurements and filter reported 3σ uncertainty bounds, where the errors are the difference between the SLAM and INS-GPS velocities. It can be seen that the velocity errors lie mostly within the 3σ bounds.

Figure 19 shows the errors in the SLAM computed attitude as being the difference between the SLAM and INS-GPS computed Euler angles. Since the errors in the Euler angles computed by the INS-GPS are of a magnitude comparable to that of the SLAM Euler angle errors, the uncertainty bounds shown are the sum of the 3σ uncertainties in the SLAM filter and the 3σ uncertainties in the INS-GPS filter. Thus we have no real comparison to the true platform Euler angles. We can however see that the errors in the roll, pitch and yaw angles of the vehicle are consistent with the error bounds reported by the SLAM and INS-GPS filters.

9.3 Localisation and Mapping Results: Larger Trajectory

In this section we analyse the results for a larger trajectory run using the logged sensor data.

The trajectory of the vehicle and the position of point features in the map is shown in

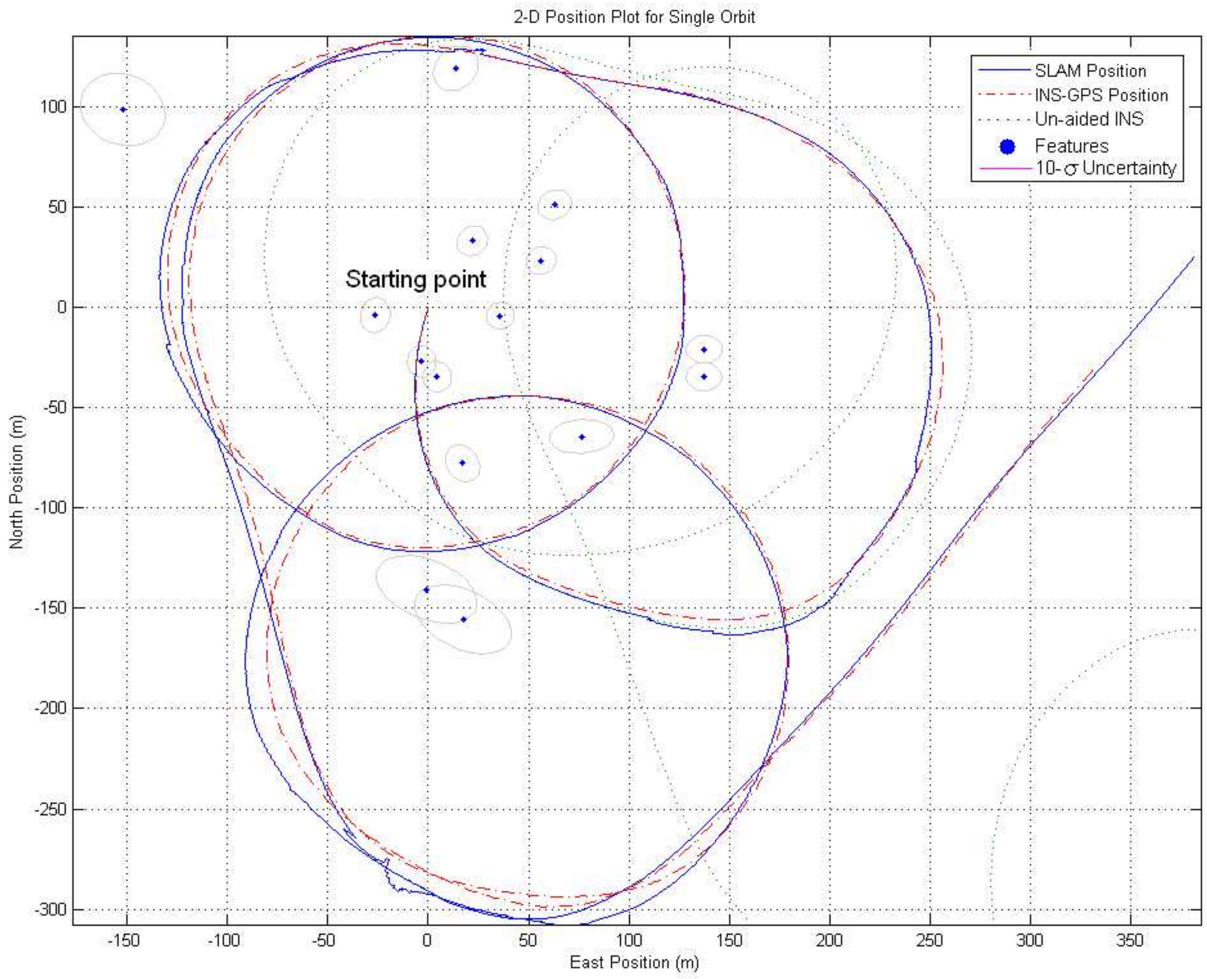

Figure 15: Small trajectory results: Vehicle trajectory from SLAM results (solid) and INS-GPS system (dotted) and un-aided INS (fine dotted), SLAM mapped features (solid points) with associated 10σ uncertainty ellipses (light ellipses).

Figure 20 with results from both the GPS-aided INS solution and the un-aided INS shown for comparison. As noted in the previous subsection, the trajectory shown is the estimated position of the vehicle at the time in which the vehicle was at that position and not the estimated position from any stored pose at that point. The vehicle performs several orbits around different locations, observing and building up a map of features below as the vehicle banks and the camera is pointing down towards the ground. The vehicle integrates 30 features into the map during the flight. In this trajectory, the vehicle spends a larger amount of time in straight and level flight where observations of features are not being made and the errors in the localisation estimates grow. It can be seen that the position trajectory computed using the un-aided INS quickly drifts from the INS-GPS solution.

Figure 21 shows the size of the SLAM EKF state vector as a function of time. Equivalently to the small sized trajectory the number of states fluctuates but with a small steady growth as the number of features in the map grows.

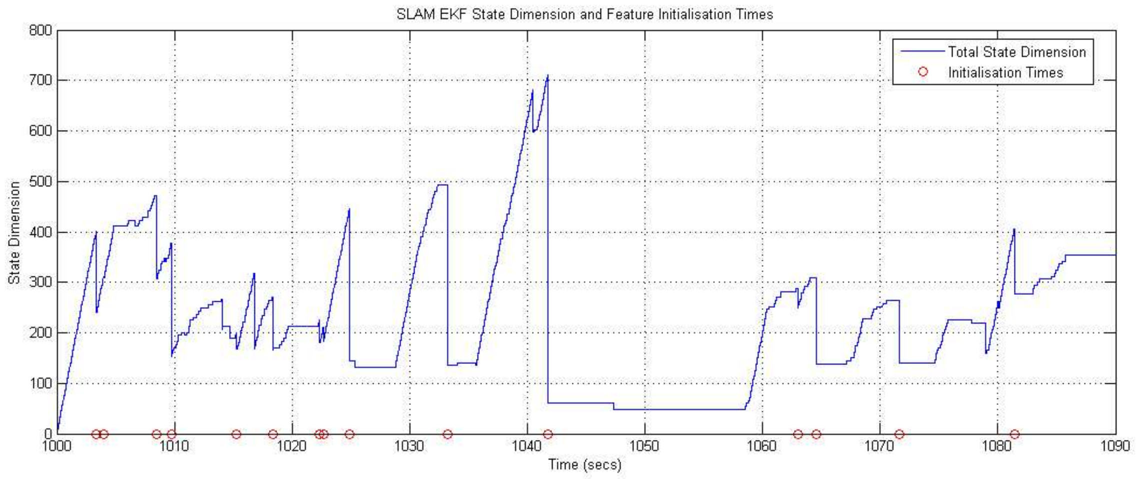

Figure 16: State dimension of the SLAM filter for small trajectory and the times for feature initialisations (circles).

Figure 22 illustrates the difference between the SLAM and INS-GPS computed positions and the expected 3σ uncertainty bounds on the position solution as reported by the SLAM EKF. The actual errors (difference to INS-GPS solution) are often slightly outside of the bounds computed by the filter and thus the solution is becoming inconsistent. The cause of this inconsistency is due to the buildup in errors in the localisation states due to some of the large gaps between feature observations (i.e. during straight and level flight). When large, these errors can violate the linearising assumptions in the EKF which leads to inconsistency.

Figure 23 shows the errors in the SLAM filter and filter reported 3σ uncertainty bounds, where the errors are the difference between the SLAM and INS-GPS velocities. It can be seen that the velocity errors lie mostly within the 3σ bounds.

Figure 24 shows the errors in the SLAM computed attitude as being the difference between the SLAM and INS-GPS computed Euler angles. We can see that the errors in the roll and pitch angles of the vehicle are consistent with the error bounds reported by the SLAM and INS-GPS filters. In the yaw angle case we can see that the error grows outside of the filter error bounds. As in the case of the position estimates this is attributed to the violation of linear assumptions affecting the consistency of the EKF during large periods of time in straight and level flight. Solving the consistency issue in the algorithm is a major part of the current and future work as discussed in the next section.

10 Conclusions and Future Work

This paper has demonstrated an implementation of inertial SLAM using bearing-only observations on an aerial vehicle using data logged during a flight test. Bearing-only initialisation of features in the map has been tackled using a delayed, batch EKF update using stored observation and vehicle pose data. A method for data association that does not rely on visual properties of features from the image data has been shown. It has been shown that

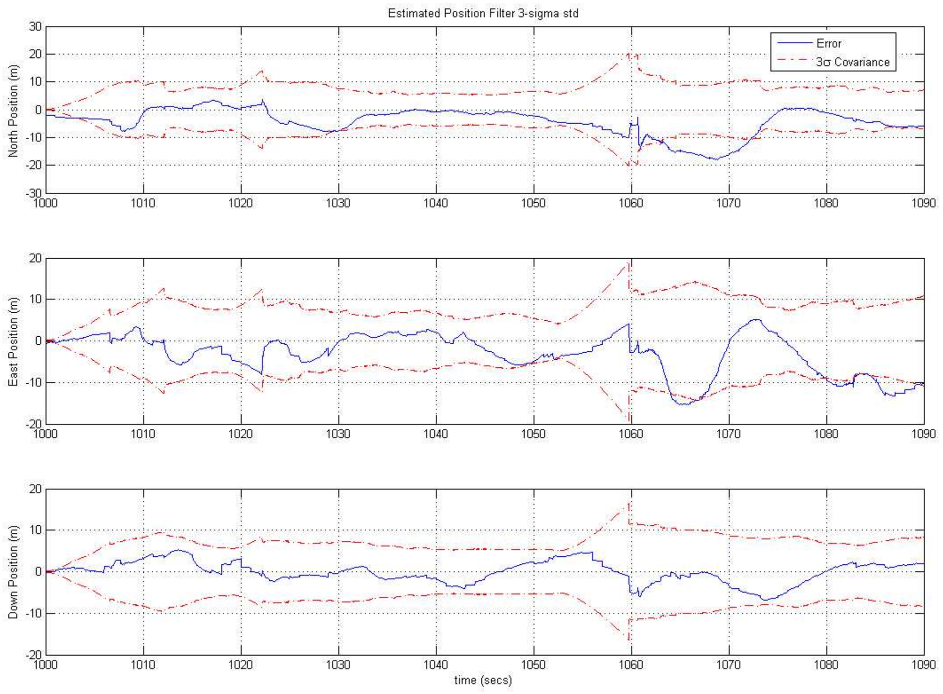

Figure 17: Small trajectory results: SLAM position error (difference between SLAM and INS-GPS system positions) (solid) and sum of 3σ uncertainty bounds for both the SLAM filter and INS-GPS filter (dotted).

the SLAM algorithm can constrain the errors in the inertial navigation system while building a 3D point feature map, without the need for GPS, known feature locations or range observations. An analysis of the computational complexity of the algorithm is presented, highlighting the steps towards achieving a real-time implementation. Results of the algorithm are shown in non-real-time using logged sensor data from a flight test of a UAV.

The performance of the SLAM algorithm has been analysed over two different trajectories. In the first case where the trajectory is small and there are no large segments of time where feature observations are not available, the SLAM algorithm is able to consistently constrain the errors in the vehicle localisation states while building a point feature map. In the second case where the trajectory of the vehicle is larger and there exists large segments of time where feature observations are not being received, the algorithm begins to become inconsistent. This is explained by buildup in localisation uncertainty and errors which violate the linearising assumptions of the EKF. The inconsistency is mainly apparent in the position and heading of the vehicle, whereas the vehicle's velocity, roll and pitch angles seem to be consistently estimated. The velocity estimates remain more consistent than the position estimate due to the fact that position must be double-integrated from the accelerometer measurements and thus drifts faster (where as velocity is a single-integration). Roll and pitch angles seem to remain more consistent than the heading (yaw) angle due to the part that the vertical gravity reference plays in the observability of aided-INS. Similar properties are sometime seen in GPS-aided INS systems.

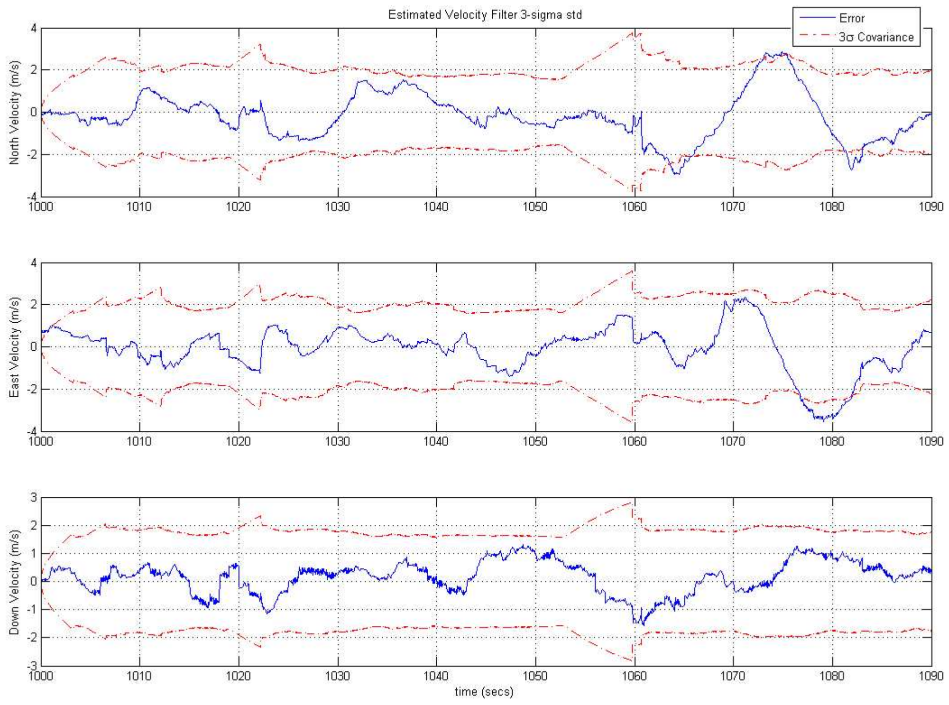

Figure 18: Small trajectory results: SLAM velocity error (difference between SLAM and INS-GPS system velocity) (solid) and sum of 3σ uncertainty bounds for both the SLAM filter and INS-GPS filter (dotted).

There are three main focal points for future work. The first point involves tackling the consistency issue in the SLAM algorithm. Filter inconsistency becomes an important issue when the vehicle is building a map over a larger environment and can lead to filter instability if left unchecked. One option that is being examined to overcome the consistency issue is the use of sub-maps (Estrada et al., 2005), which improve the consistency of the estimates by breaking the map up into small relative sub-maps. Different estimators such as using the UKF (Martinez-Cantin and Castellanos, 2005) which seems to deal well with inconsistencies brought about by non-linearities, are also being considered. Future work will also look at online IMU error calibration to combat biases and misalignments in the IMU, which should aid in the consistency by reducing the drift in localisation estimates between feature observations.

As has been demonstrated in the results, estimate inconsistency can be mitigated when operating over an appropriate vehicle trajectory that couples well with the placement of the feature sensor on the vehicle such that feature observations are frequent. The second main focal point of future work will look at more suitable pointing direction for the onboard camera and coupling in the control of the vehicle with the SLAM algorithm such that the vehicle makes online control decisions which will maximise the accuracy and consistency of the algorithm. It could be suggested that perhaps the camera could be pointed in a downwards direction in order to overcome the lack of features during straight and level flight. However, in this case features may not be observed during turns and the maneuver profiles required to attain a sufficient baseline to initialise a feature's 3D position may be impractical or difficult

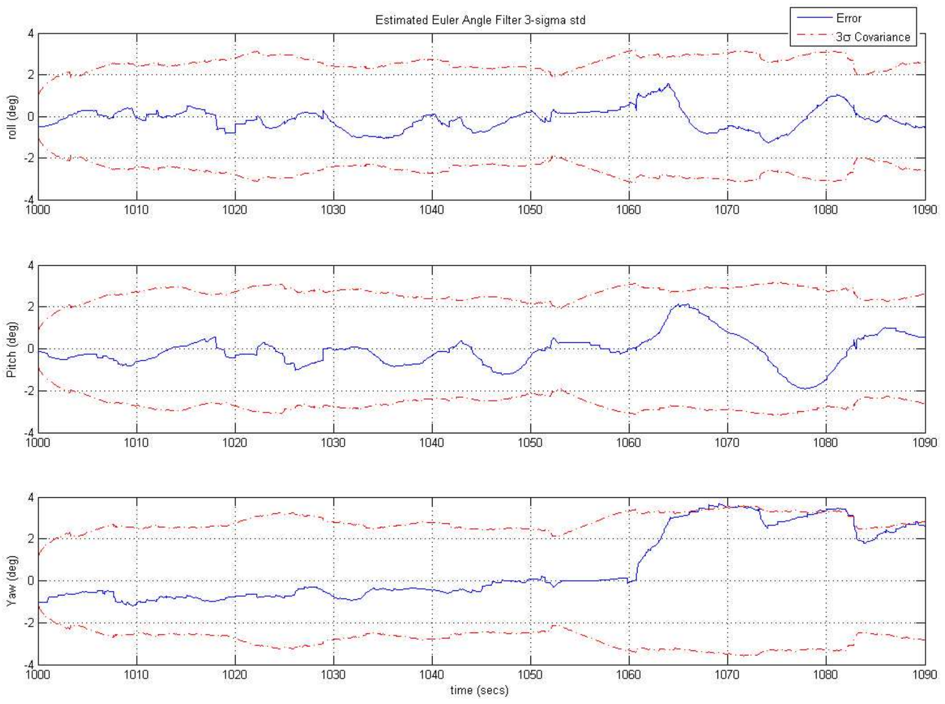

Figure 19: Small trajectory results: SLAM euler angle errors (difference between SLAM and INS-GPS system euler angles) (solid) and sum of 3σ uncertainty bounds for both the SLAM filter and INS-GPS filter (dotted).

for a fixed wing air vehicle. An observability analysis of the inertial SLAM algorithm in (Bryson and Sukkarieh, 2006) has also shown that vehicle maneuvers that excite a lateral acceleration (i.e. turns) are necessary to ensure observability and thus constrain the errors in the algorithm.

It is our belief that the question of camera pointing direction should be coupled to the maneuver profiles of the vehicle which in turn should be actively controlled in order to maximise the accuracy and consistency of the algorithm. These choices would be dependent on a information-theoretic framework for understanding the relationships between sensor pointing, vehicle maneuvers and algorithm accuracy and consistency. Such a framework is thus the topic of future work.

The third main focal point of the future work involves implementing the demonstrated algorithms into a software framework capable of running in real-time. The analysis performed in this paper and the methods discussed to manage the computational complexity will form the basis for a real-time implementation that can handle environments of the same size as demonstrated in the results. Further work will examine methods for mitigating the computational complexity allowing for real-time operation in much larger environments.

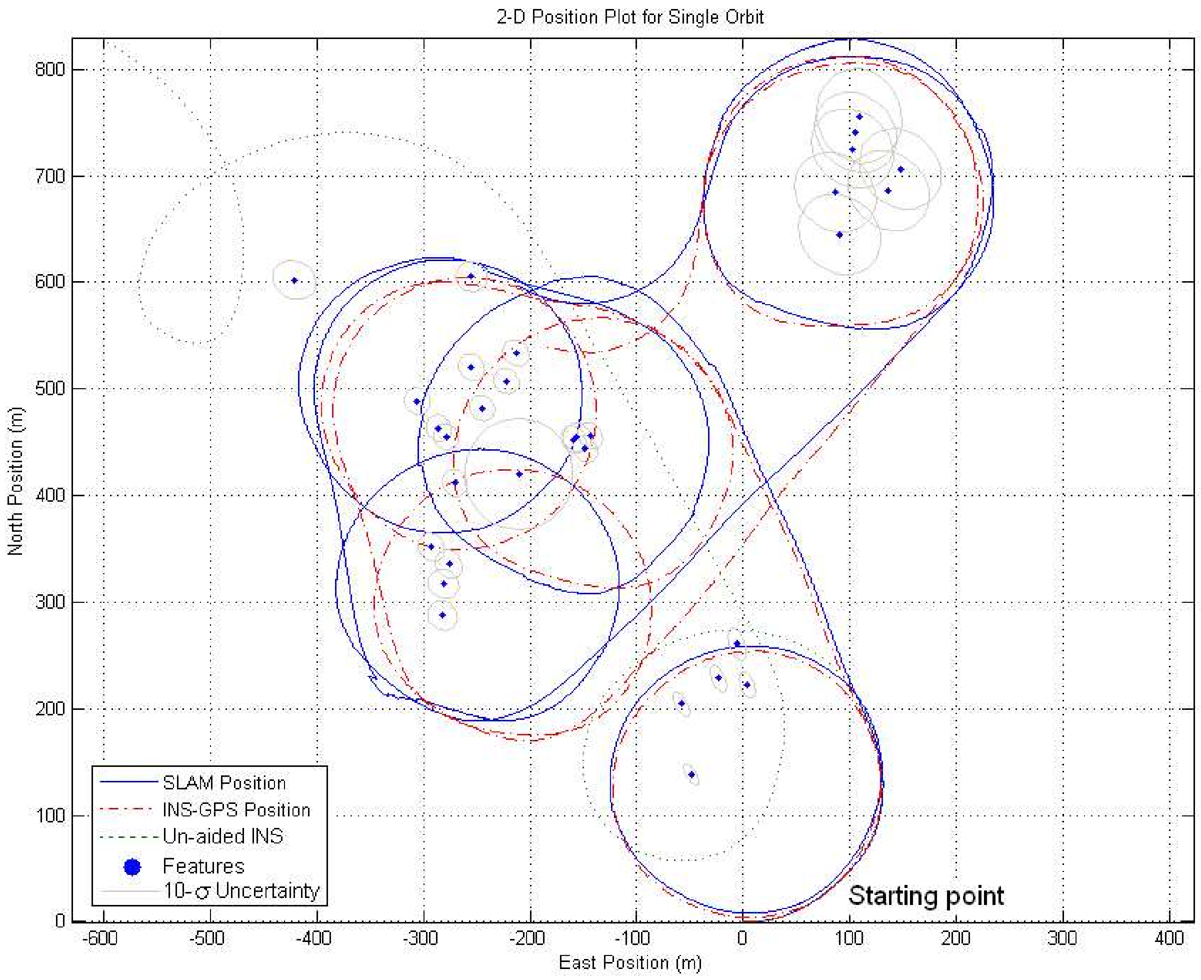

Figure 20: Vehicle trajectory from SLAM results (solid) and INS-GPS system (dotted) and un-aided INS (fine dotted), SLAM mapped features (dark points) with associated 10σ uncertainty ellipses (light ellipses).

Acknowledgments

This work is supported in part by the ARC Centre of Excellence programme, funded by the Australian Research Council (ARC) and the New South Wales State Government.

References

Alspach, D.L., Sorenson, H.W., (1972). Nonlinear Bayesian Estimation using Gaussian Sum Approximations. In *IEEE Trans. on Automatic Control*, vol. 17, no. 4, pp. 439-448.

Bailey, T., (2003). Constrained Initialisation for Bearing-Only SLAM. In *Proceedings of IEEE International Conference on Robotics and Automation*, Taipei.

Bouguet, J., *Camera Calibration Toolbox for MATLAB*, Retreived 4th October, 2006 from http://www.vision.caltech.edu/bouguetj/calib_doc/.

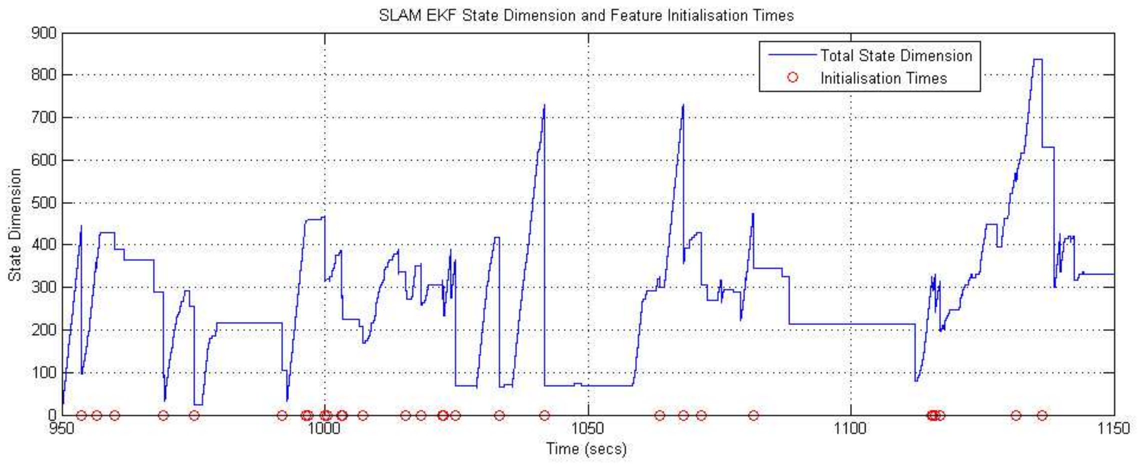

Figure 21: State dimension of the SLAM filter for larger trajectory and the times for feature initialisations (circles).

Braun, R.D., Wright, H.S., Croom, M.A., Levine, J.S., Spencer, D.A., (2004). The Mars Airplane: a Credible Science Platform. In *Proceedings of IEEE Aerospace Conference*, Big Sky, MT.

Bryson, M., Sukkarieh, S., (2006). Active Airborne Localisation and Exploration in Unknown Environments using Inertial SLAM. In *Proceedings of IEEE Aerospace Conference*, Big Sky, MT.

Cole, D.T., Sukkarieh, S., Goktogan, A.H., Stone, H., Hardwick-Jones, R., (2005). The Development of a Real-Time Modular Architecture for the Control of UAV Teams. In *Proceedings of the 5th International Conference on Field and Service Robotics*, Port Douglas.

Davison, A.J., (2003). Real-Time Simultaneous Localisation and Mapping with a Single Camera. In *Proceedings of IEEE Conference on Computer Vision*, New York.

Deans, M., Hebert, M., (2000). Experimental Comparison of Techniques for Localization and Mapping using a Bearings Only Sensor. In *Proceedings of the 7th International Symposium on Experimental Robotics*, Honolulu.

Dissanayake, M.W.M.G., Newman, P., Clark, S., Durrant-Whyte, H.F., Csorba, M., (2001). A Solution to the Simultaneous Localization and Map Building (SLAM) Problem. In *IEEE Trans. on Robotics and Automation*, vol. 17, no. 3, pp. 229-241.

Dissanayake, G., Williams, S.B., Durrant-Whyte, H.F., Bailey, T., (2002). Map Management for Efficient Simultaneous Localization and Mapping. In *Autonomous Robots* vol. 12, pp. 267-286.

Estrada, C., Neira, J., Tardos, J.D., (2005). Hierarchical SLAM: Real-Time Accurate Mapping of Large Environments. In *IEEE Trans. on Robotics and Automation*, vol. 21, no. 4, pp. 588-596.

Fitzgibbons, T., Nebot, E., (2002). Bearing-Only SLAM using Colour-based Feature Tracking. In *Proceedings of Australasian Conference on Robotics and Automation*, Auckland.

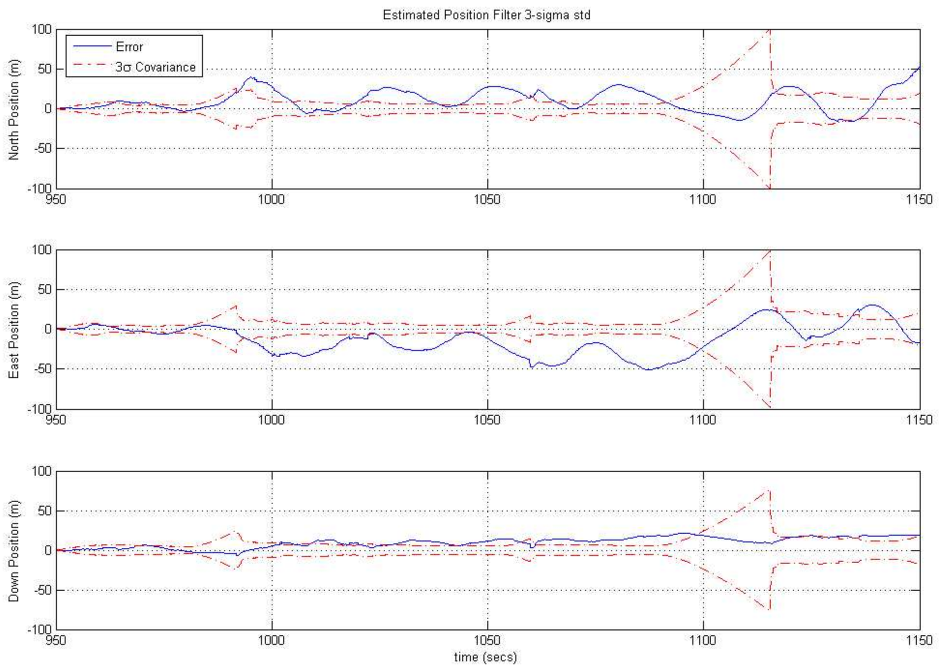

Figure 22: SLAM position error (difference between SLAM and INS-GPS system positions) (solid) and sum of 3σ uncertainty bounds for both the SLAM filter and INS-GPS filter (dotted).

Gebre-Egziabher, D., Hayward, R.C., Powell, J.D., (1998). A Low-cost GPS/inertial Attitude Heading Reference System (AHRS) for General Aviation Applications. In *Proceedings of IEEE Position Location and Navigation Symposium.*

Gordon, N.J., Salmond, D.J., Smith, A.F.M., (1993). A Novel Approach to Nonlinear/Non-Gaussian Bayesian State Estimation. In *IEE Proceedings-F, Radar and Signal Processing*, vol. 140, no. 2, pp.107-113.

Guivant, J., Nebot, E., (2001). Optimization of the Simultaneous Localization and Map-Building Algorithm for Real-Time Implementation. In *IEEE Trans. on Robotics and Automation*, vol. 17, no. 3, pp. 242-257.

Jung, I., Lacroix, S., (2003). High Resolution Terrain Mapping using Low Altitude Aerial Stereo Imagery. In *Proceedings of IEEE International Conference on Computer Vision*, Nice.

Kim, J.H., Wishart, S., Sukkarieh, S., (2003). Real-Time Navigation, Guidance and Control of a UAV using Low-cost Sensors. In *Proceedings of the 4th International Conference on Field and Service Robotics*, Yamanashi.

Kim, J.H., Sukkarieh, S., (2003). Airborne Simultaneous Localization and Map Building. In *Proceedings of IEEE International Conference on Robotics and Automation*, Taipei.

Kim, J.H., Sukkarieh, S., (2004). Autonomous Airborne Navigation in Unknown Terrain Environments. In em IEEE Trans. on Aerospace and Electronic Systems, vol. 40, no. 3, pp. 1031-1045.

Kwok, N.M., Dissanayake, G., (2004). An Efficient Multiple Hypothesis Filter for Bearing

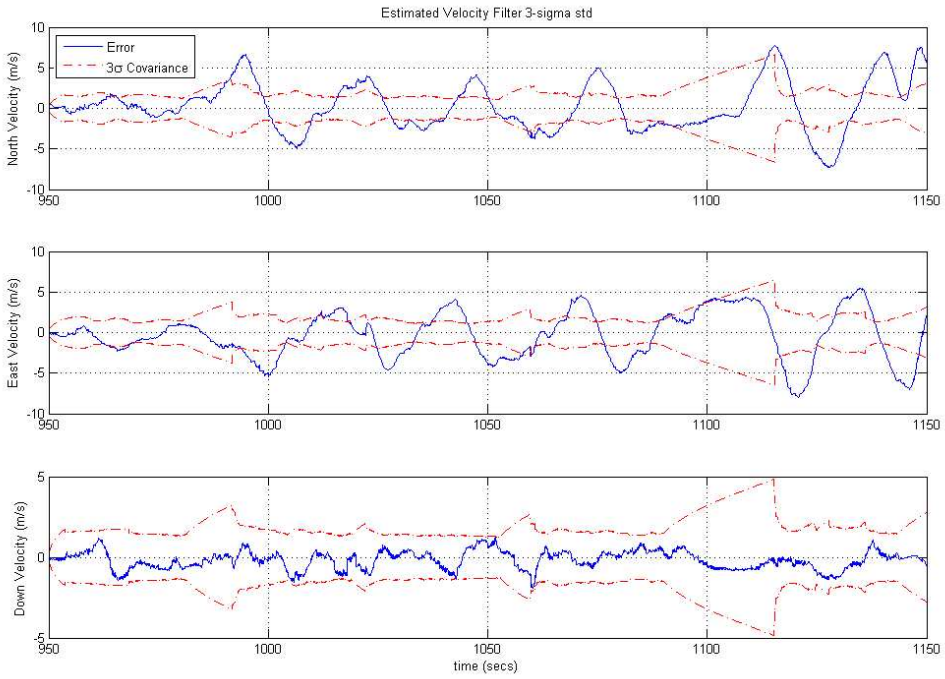

Figure 23: SLAM velocity error (difference between SLAM and INS-GPS system velocity) (solid) and sum of 3σ uncertainty bounds for both the SLAM filter and INS-GPS filter (dotted).

Only SLAM. In *Proceddings of IEEE/RSJ International Conference on Intelligent Robots and Systems*, Sendai.

Lowe, D.G., (2004). Distinctive Image Features from Scale-Invariant Keypoints. In *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91-110.

Martinez-Cantin, R., Castellanos, J.A., (2005). Unscented SLAM for Large-scale Outdoor Environments. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*, Edmonton.

Montiel, J.M.M., Civera, J., Davison, A.J., (2006). Unified Inverse Depth Parameterization for Monocular SLAM. In *Proceedings of 'Robotics: Science and Systems' Conference*, Philadelphia.

Neira, J., Tardos, J.D., (2001). Data association in stochastic mapping using the joint compatibility test. In *IEEE Trans. on Robotics and Automation*, vol. 17, no. 6, pp. 890-897.

Nettleton, E. (2003). *Decentralised Architectures for Tracking and Navigation with Multiple Flight Vehicles*. PhD Thesis, University of Sydney.

Nixon, M.S., Aguado A.S., (2001). *Feature Extraction and Image Processing*. Butterworth Heinmann/Newnes, Oxford.

Smith, R., Self, M., Cheeseman, P., (1990). Estimating Uncertain Spatial Relationships in Robotics. In *Autonomous Robot Vehicles*, Springer Verlag, New York.

Sola, J., Monin, A., Devy, M., Lemaire, T., (2005). Undelayed Initialisation in Bearing-Only SLAM. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*, Edmonton.

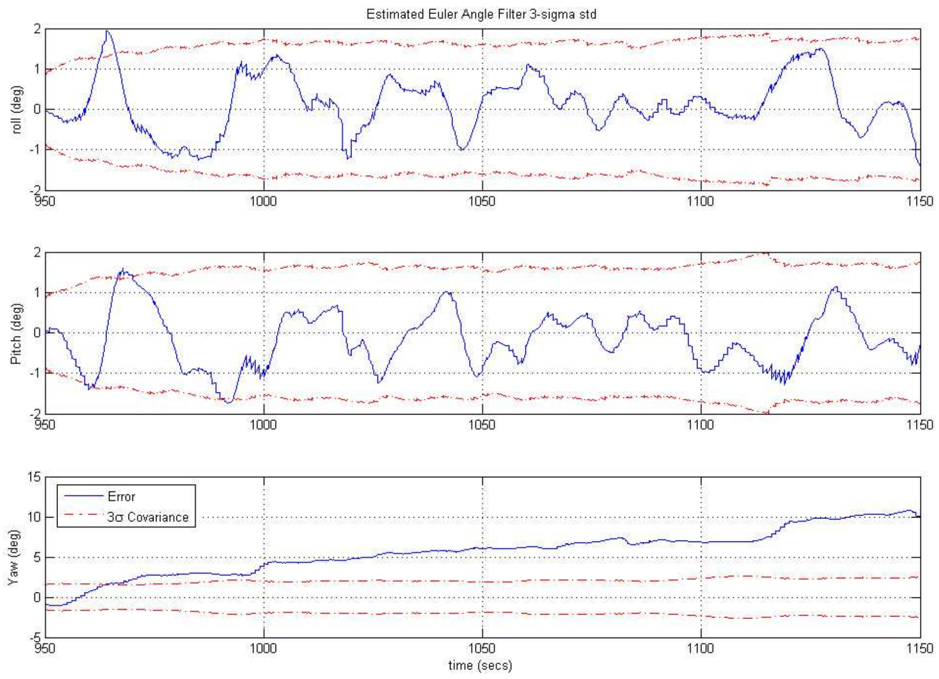

Figure 24: SLAM euler angle errors (difference between SLAM and INS-GPS system euler angles) (solid) and sum of 3σ uncertainty bounds for both the SLAM filter and INS-GPS filter (dotted).

Stevens, A., Stevens, M., Durrant-Whyte, H.F., (1995). OxNav: Reliable autonomous navigation. In *Proceedings of IEEE International Conference on Robotics and Automation*, Nagoya, Japan.

Titterton, D., Weston, J., (1997). *Strapdown Inertial Navigation Technology.* Peter Peregrinus Ltd., London.

Triggs, B., McLauchlan, P., Hartley, R., Fitzgibbon, A., (2000). Bundle Adjustment - A Modern Synthesis. In *Vision Algorithms: Theory and Practice*, Springer Verlag.

Williams, S.B., Dissanayake, G., Durrant-Whyte, H., (2001). Towards Terrain-Aided Navigation for Underwater Robotics. In *Advanced Robotics*, vol. 15, no. 5, pp. 533-550.