Range Estimation Using Kalman Filter Based Motion Stereo for UAV Obstacle Detection System

Fan Zhang
Carleton University, student number: 100296342

ABSTRACT

Motion stereo, using known camera motion to estimate range, is an important and popular research subject. In many applications such as navigation, the algorithm must be capable of real time processing, and incrementally integrating new information. Kalman filter is such a tool that fuses new measurements with existing estimates, and provides uncertainty estimates at the same time. In this project, two Kalman filter frameworks were designed for motion stereo problem. Framework 1, which based entirely on the output of an optical flow algorithm, is completely implemented and tested. The simulation result of framework 1 has proven that a Kalman filter based motion stereo algorithm is more reliable and stable than an optical flow algorithm alone. However, due to its simplicity, framework 1 is only applicable to aircraft motion with constant speed and slow rotation. To further exploit the potential of Kalman filter in motion stereo problem, and to make the framework applicable to more general aircraft maneuvers, framework 2 is also implemented. The second framework is also more sophisticated than model 1 as it fuses GPS and inertial navigation system (INS) data with the optical flow algorithm output. Although the implementation and simulation is not fully completed for framework 2, more work will be devoted in the future to complete this framework.

Keywords: Motion stereo; Kalman filter; Depth estimation; Obstacle detection

1. INTRODUCTION

Since 2004, Carleton University has been developing GeoSurv-II fix-wing UAV for Sander Geophysics Ltd to conduct low altitude geophysical survey. The UAV is expected to perform contour flight at low altitude to enable high resolution geophysical data acquisition. Although the flight path is planned with the aid of digital terrain map, the aircraft can still encounter unexpected obstacles such as tall trees or small hills. Obstacle detection and avoidance becomes an important issue for aircraft flying at low altitude.

In recent years, radar, sonar and laser system has been widely used on UAVs for obstacle detection and avoidance. However, these systems are not suitable for all UAV platforms due to their weight, size and cost. With recent advances of computer vision technology, cameras as passive sensors provide a low cost, light weight, yet feasible alternative.

Among all the passive vision systems proposed for range estimation, binocular stereo vision and monocular motion stereo have been the two most popular methods, with each method having its own advantages and disadvantages. Binocular stereo suffers the greatest error in range computation at the edge of the Field of View (FOV) of the camera where motion stereo-based method produces the most accurate result; conversely, in the vicinity of the Focus of Expansion (FOE) where the motion stereo-based method produces very large error, the error from a binocular stereo-based method is very small [1] . Therefore, it is not uncommon for an obstacle avoidance system to utilize a suit of multiple sensors, such as a single motion sensor with wide FOV combined with narrow FOV binocular sensors. Throughout the last few years, extensive work has been done at Carleton University towards the binocular stereo approach. However, not a lot of research and implementation has been done on motion stereo.

The goal of the project is to review previous related research, and implement an algorithm that generates range and range uncertainty estimation from monocular image sequence by using Kalman filter based optical flow algorithm. The use of a Kalman filter with optical flow algorithm is motivated by its ability to fuse new measurements into existing range estimates, and calculate uncertainty measures for the estimates. Such information provides valuable input to the data fusion process of the motion stereo and binocular stereo.

Project Scope

- A static scene with moving camera scenario will be considered.

- The Kalman filter framework is based on feature based optical flow algorithm.
- The framework should apply to arbitrary camera motion.
- The framework assumes feature points in the neighborhood move with similar velocity.
- The implementation is completed in Python utilizing the OpenCV library.

The rest of this paper is structured as follows. In the next section, an overview of the estimation framework is presented. Two frameworks were designed in this project, and they are described in detail in section 3 and 4. The simulation result of the first framework is presented in section 5. In section 6, I conclude the accomplishment of this project, and what can be done to further exploit the potential of the Kalman filter framework.

2. ESTIMATION FRAMEWORK

The algorithm uses image sequence with small frame to frame camera motion, because stable correspondence is easier to achieve with small camera motion. However, it also means the resolution in depth estimation is less accurate due to the small baseline between consecutive frame pairs. The problem can be resolved by incrementally integrating new measurement after new image is acquired. The incremental approach also offer real-time operation and demand less storage than processing many images in a batch.

The Kalman filter is a powerful tool for doing incremental and real-time estimation from measurements taken by noisy sensors. It is able to integrate information over time, be robust to measurement noise and system noise, and provide uncertainty estimation at the same time.

In this section, I first present the notation and equations of the Kalman filter, followed by a brief description of the two frameworks designed for motion stereo problem, and at last the camera model and depth extraction equations. The details of my algorithms are described in section 3 and 4.

2.1 Kalman Filter

Kalman filter is a Bayesian estimation technique that based on three very important but reasonable assumptions. (1) The system to be modeled linear, (2) The measurement noise is Gaussian in nature, and (3) The measurement noise is white. [2] The equations of the filter is summarized in table 1.

Table 1 Discrete Kalman Filter Equations [3]

System Model	$x_k = \mathrm{F}_{k-1}x_{k-1} + w_{k-1},\ w_k \sim N(0, Q_k)$
Measurement Model	$z_k = H_k x_k + v_k,\ v_k \sim N(0, R_k)$
Prior Model	$E[x(0)] = \hat{x}_0,\ E[(x(0) - \hat{x}_0)(x(0) - \hat{x}_0)^T] = P_0$
Other Assumptions	$E\left[w_k {v_j}^T\right] = 0, \quad for\ all\ j, k$
State Estimation Extrapolation	$\hat{x}_k^- = \mathrm{F}_{k-1}\hat{x}_{k-1}^+ + w_{k-1}$
Error Covariance Extrapolation	$P_k^- = \mathrm{F}_{k-1}P_{k-1}^+\mathrm{F}_{k-1}^T + Q_{k-1}$
State Estimation Update	$\hat{x}_k^+ = \hat{x}_k^- + K_k[z_k - H_k\hat{x}_k^-]$
Error Covariance Update	$P_k^+ = [I - K_k H_k]P_k^-$
Kalman Gain Matrix	$K_k = P_k^- H_k^T[H_k P_k^- H_k^T + R_k]^{-1}$

The Kalman filter is based on three probability models. The *system model* relates the previous state vector x_k to the current one through the transition matrix F_k and an addition of Gaussian white noise with covariance of Q_k. The *measurement model* relates the measurement vector z_k to the current state vector through a measurement matrix H_k and an addition of Gaussian white noise with covariance of R_k. The *prior model* describes the knowledge of the system vector and its covariance matrix before the first measurement takes place. In addition, the system noise and the noise from measurements are assumed to be uncorrelated.

Upon the filter initialization through the prior model, the filter enters a two phases operation. In extrapolation phase, the filter predicts the current state through the knowledge from the past. In update phase, the filter corrects its prediction according to the new measurement. The gain matrix K_k used in the update phase decides how much the filter corrects its prediction, and is determined from the predicted state covariance matrix and the measurement covariance matrix.

2.2 Application of Kalman Filter to Motion Stereo

In this project, two Kalman filter frameworks were designed to target the motion stereo problem. The first framework fuses only the results from processing the image sequence while the second framework integrates the camera motion into the extrapolation and update phase of the framework.

The first step towards designing a Kalman filter is to define the state vector. In the first framework, the state vector contains the pixel location as well as their disparity between frames. In the second framework, the state vector is defined to be the world location of the feature in camera coordinate, which will be defined in section 2.3.

The extrapolation phase and update phase are also different in the two frameworks. The first framework is a linear system. It extrapolates the current state vector based on the knowledge of the past feature disparity in the image sequence. As a result, this framework fuses only the optical flow algorithm output. The second framework is not a linear system in its measurement model, and must be linearized. However, this framework predicts the next state vector based on the known camera motion, and corrects its prediction using the optical flow output. Therefore, the second framework fuses information from two distinct sensor systems.

2.3 Measuring Disparity

Feature extraction and feature correspondence problem are each a popular research topic on their own. As my project goal is to establish the Kalman filter framework, I chose two pre-implemented algorithms from OpenCV for feature extraction and feature tracking task. Feature extraction is achieved via the *GoodFeaturesToTrack()* function which based on Shi and Tomasi corner detector. Feature tracking is accomplished through the *CalcOpticalFlowPyrLK()* function, which is a feature based pyramid implementation of the Lucas-Kanade optical flow algorithm. [2]

2.4 Camera Model and Depth from Motion Equations

2.4.1 Camera Model

A camera model is essential to associate the optical flow to the actual feature depth. In this project, the optical distortion is not considered, and a pin-hole camera model will be used, which is illustrated in figure 1. The model defines a Cartesian coordinate that follows the right hand rule with its origin located at the camera focal point, and Z axis pointed along the camera optical axis. It specifies (c_x, c_y) as the origin of the image plane, and s_x and s_y as scaling factor that is a combination of the camera focal length and image sensor pixel size.

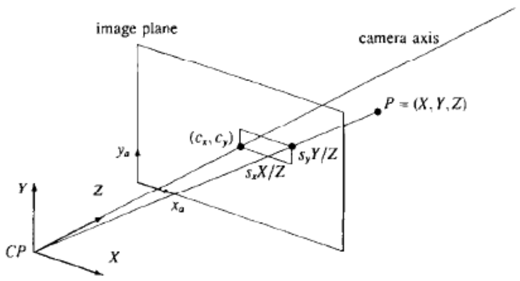

Figure 1 Camera Model [4]

2.4.2 Depth from Motion Equations

If camera motion is small, the disparity produced by a feature in the world location $[X, Y, Z]$ from two consecutive frames can be related to the camera translational velocity $[T_x, T_y, T_z]$, and rotational velocity $[R_x, R_y, R_z]$ through (1).

$$\frac{dX}{dt} = -T_x - R_y Z + R_z Y$$
$$\frac{dY}{dt} = -T_y - R_z X + R_x Z \qquad (1)\ [4]$$
$$\frac{dZ}{dt} = -T_z - R_x Y + R_y X$$

Projecting the point at [X, Y, Z] onto the image plane through $x = s_x \frac{X}{Z}$, and $y = s_y \frac{Y}{Z}$, and substitute them into (1), leads to the famous equations of optical flow:

$$\begin{bmatrix} \Delta u \\ \Delta v \end{bmatrix} = \frac{1}{Z}\begin{bmatrix} -s_x & 0 & u \\ 0 & -s_y & v \end{bmatrix}\begin{bmatrix} T_x \\ T_y \\ T_z \end{bmatrix} + \begin{bmatrix} \frac{uv}{s_y} & -(s_x + \frac{u^2}{s_x}) & v \\ (s_y + \frac{v^2}{s_y}) & -\frac{uv}{s_x} & -u \end{bmatrix}\begin{bmatrix} R_x \\ R_y \\ R_z \end{bmatrix} \qquad (2)$$

where $[u, v]$ is the feature coordinate in image plane with (c_x, c_y) being the origin , and $[\Delta u, \Delta v]$ is the feature disparity in the image plane.

3. FRAMEWORK 1

This section describes the Kalman filter framework that estimates feature disparity entirely based on the past optical flow. The block diagram of the algorithm is shown in figure2.

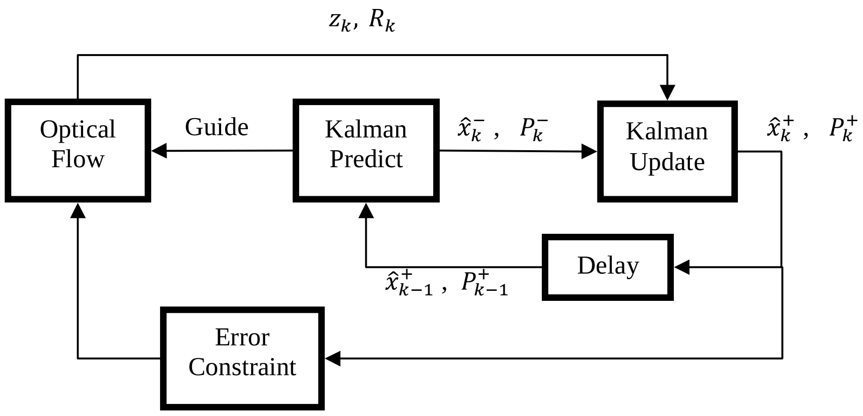

Figure 2 Framework 1 block diagram

The state vector of the framework $x_k = [u, v, \Delta u, \Delta v]^T$ contains the image plane coordinate of the feature point, and its disparity resulted from the previous and current frame. The measurement vector $z_k = [u, v, \Delta u, \Delta v]^T$ contains the exact same components. Therefore, the transition matrix F_k and measurement matrix H_k are defined as follow:

$$F_k = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (3)$$

$$H_k = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (4)$$

The uncertainty in the position of the feature points can be model as a bivariate Gaussian random variable with σ_u^2 and σ_v^2 as its variance in x and y direction in the image plane. The disparity is calculated from the feature position in two consecutive frames, and therefore is related to the feature position by:

$$\begin{bmatrix}\Delta u_k \\ u_k\end{bmatrix} = \begin{bmatrix}1 & -1 \\ 1 & 0\end{bmatrix}\begin{bmatrix}u_k \\ u_{k-1}\end{bmatrix} = A\begin{bmatrix}u_k \\ u_{k-1}\end{bmatrix} \tag{5}$$

Assuming u_{k-1} and u_k are two independent random variables, the covariance matrix of random variable $[u_k, \Delta u_k]$ can be calculated from equation (6). The same calculation applies to $\Lambda_{\Delta v,v}$

$$\Lambda_{\Delta u_k, u_k} = A\Lambda_{\mathrm{u_{k-1},u_k}}\mathrm{A^T} = \begin{bmatrix}2\sigma_u^2 & \sigma_u^2 \\ \sigma_u^2 & \sigma_u^2\end{bmatrix} \qquad (6)\ [5]$$

Therefore the covariance matrix of the state vector is defined as:

$$P_k = \begin{bmatrix}1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 2 & 0 \\ 0 & 1 & 0 & 2\end{bmatrix} \cdot \begin{bmatrix}\sigma_u^2 \\ \sigma_v^2 \\ \sigma_u^2 \\ \sigma_v^2\end{bmatrix} \tag{7}$$

3.1 Filter initialization

The filter is initialized by the optical flow calculated from the first two frames of the image sequence. For a block of feature points, we obtain the mean disparity $[\overline{\Delta u_{1-0}}, \overline{\Delta v_{1-0}}\ ]$ in x and y direction for the block. The initial state vector for every feature points in the block is then defined as $\hat{x}_0^+ = [u_1 \quad v_1 \quad \overline{\Delta u_{1-0}} \quad \overline{\Delta v_{1-0}}]^T$. The initial state vector covariance matrix P_0^+ is calculated for every feature point in the block, and is defined as:

$$P_0 = \begin{bmatrix}0.5 & 0 & 0.5 & 0 \\ 0 & 0.5 & 0 & 0.5 \\ 0.5 & 0 & 1 & 0 \\ 0 & 0.5 & 0 & 1\end{bmatrix} \cdot \begin{bmatrix}\sigma_{\Delta u}^2 \\ \sigma_{\Delta v}^2 \\ \sigma_{\Delta u}^2 \\ \sigma_{\Delta v}^2\end{bmatrix} \tag{8}$$

where

$$\sigma_{\Delta u}^2 = \left[\Delta u - \overline{(\Delta u_{1-0})}\right]^2, \sigma_{\Delta v}^2 = \left[\Delta v - \overline{(\Delta v_{1-0})}\right]^2 \tag{9}$$

3.2 Extrapolation

The prediction phase is carried out through the matrix multiplication $\hat{x}_k^- = \mathrm{F}_{k-1}\hat{x}_{k-1}^+$. Element wise, it can be seen that the multiplication simply states:

$$\begin{aligned} u_k &= u_{k-1} + \Delta u_{k-1} \\ v_k &= v_{k-1} + \Delta v_{k-1} \\ \Delta u_k &= \Delta u_{k-1} \\ \Delta v_k &= \Delta v_{k-1} \end{aligned} \tag{10}$$

The noise $w_k$input to the extrapolation equation describes the uncertainty on the prediction due the information on which the prediction is based is from the past. I interpret that this uncertainty is related to the kind of motion the aircraft is undergoing. When flying at a constant speed on a straight line, the noise can be model with a constant variance. However, when the aircraft is accelerating or decelerating in translation or rotation, a constant variance no longer describes the situation. In fact, the framework must integrate inertial sensor data to model these flying conditions properly. Fortunately, given the image sequence used for testing the algorithm is taken during a geophysical survey flight, in which the aircraft flies a straight line with constant speed, and as little rotation as possible, I chose to implement a simplified version of the framework, and model the noise $w_k$with a constant variance. The covariance matrix of w_k is therefore given by

$$Q_{k-1} = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 2 & 0 \\ 0 & 1 & 0 & 2 \end{bmatrix} \cdot \begin{bmatrix} \sigma_w^2 \\ \sigma_w^2 \\ \sigma_w^2 \\ \sigma_w^2 \end{bmatrix} \tag{11}$$

where σ_w^2 is approximately the variance of T_y projected onto the image plane.

3.3 Update

In update phase, the filter corrects its prediction according to the new measurement, and the confidence of the prediction. The covariance matrix of the measurement is calculated for every pixel in the block using equation:

$$R_k = \begin{bmatrix} 0.5 & 0 & 0.5 & 0 \\ 0 & 0.5 & 0 & 0.5 \\ 0.5 & 0 & 1 & 0 \\ 0 & 0.5 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \sigma_{\Delta u_k}^2 \\ \sigma_{\Delta v_k}^2 \\ \sigma_{\Delta u_k}^2 \\ \sigma_{\Delta v_k}^2 \end{bmatrix} \tag{12}$$

The gain Kalman matrix K_k, and the updated state vector are then calculated using the equations in table 1. The K_k is a correction factor that describe how much the predicted state vector $\hat{x}_k^-$ should be corrected by the measurements. When the measurement covariance matrix is small, more emphasis is put on the measurements, and vice versa when measurement error is great.

3.4 Interaction with the Optical Flow Algorithm

The Kalman state vector influences the optical flow algorithm from two aspects. Firstly, the output of the Kalman predictor is supplied to the optical flow algorithm as a guide. The optical flow algorithm first looks for the best matching points in the neighborhood of the guided location before it widen its search to the entire image. Secondly, the error constraint module acts as a input selector for the optical flow algorithm. When using optical flow algorithm alone for feature tracking, we usually feed the output of the optical flow function from previous frame as its input for the current frame. In this framework, the raw output was first examined by the error constraint block before it is fed to the optical flow algorithm. The selection criteria is described by equation (13)

$$\begin{aligned} &if\ \sigma_{\Delta u_k}^2 < 3P_{k-1}^+(3,3)\ and\ \sigma_{\Delta v_k}^2 < 3P_{k-1}^+(4,4) \\ &\qquad optical\ flow\ input(k) = optical\ flow\ output(k-1) \\ &else \\ &\qquad optical\ flow\ input(k) = updated\ Kalman\ state\ vector(k-1) \end{aligned} \tag{13}$$

4. FRAMEWORK 2

This section describes the detail of the second framework. The algorithm block diagram is shown in figure3.

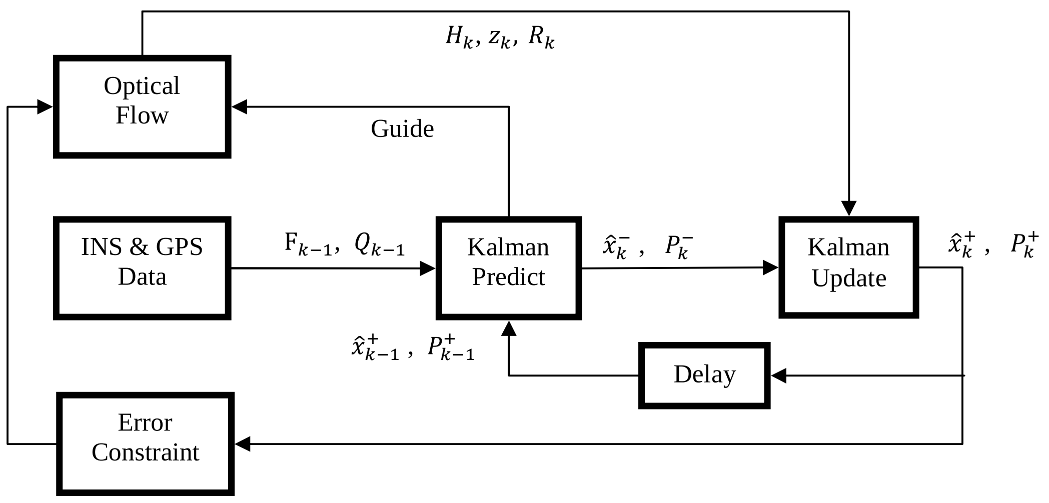

Figure 3 Framework 2 block diagram

The state vector of framework 2 contains the world coordinate of the feature point $x_k = [X, Y, Z, 1]^T$. The transition matrix is defined by the inverse coordinate transformation matrix that relates the camera coordinate from one frame to the next (figure4). The detail equations are given in (14).

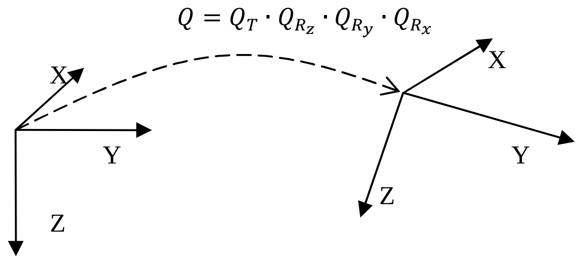

Figure 4 Camera coordinate system transformation

$$F = Q_{inv} = Q_{R_{x}\_inv} \cdot Q_{R_{y}\_inv} \cdot Q_{R_{z}\_inv} \cdot Q_{T\_inv}$$

$$Q_{R_{x}\_inv} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & cos(R_x) & sin(R_y) & 0 \\ 0 & -sin(R_y) & cos(R_y) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad Q_{R_{y}\_inv} = \begin{bmatrix} cos(R_y) & 0 & -sin(R_y) & 0 \\ 0 & 1 & 0 & 0 \\ sin(R_y) & 0 & cos(R_y) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$Q_{R_{z}\_inv} = \begin{bmatrix} cos(R_z) & sin(R_z) & 0 & 0 \\ -sin(R_z) & cos(R_z) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad Q_{T_{inv}} = \begin{bmatrix} 1 & 0 & 0 & -T_x \\ 0 & 1 & 0 & -T_y \\ 0 & 0 & 1 & -T_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

(14) [6]

The measurement vector is $z_k = [u \quad v]^T$, where u and v is the coordinate of the feature point in image plane. The function that relates the measurements to the state vector is

$$z_k = \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} s_x X/Z \\ s_y Y/Z \end{bmatrix} = h(x_k) \qquad (15)\ [3]$$

Clearly the $h(x_k)$ is not linear, and must be linearized.

$$H = \frac{\partial h(x_k)}{\partial x_k} = \begin{bmatrix} \frac{s_x}{Z} & 0 & -\frac{s_x X}{Z^2} & 0 \\ 0 & \frac{s_y}{Z} & -\frac{s_y Y}{Z^2} & 0 \end{bmatrix} \qquad (16)\ [3]$$

The error modeling of this framework requires more research. As the implementation stands now, the content of the error matrix Q_k has only accounted for the GPS error, which can be calculated using a similar method as equation (5) and (6). The error in rotational velocity is temporarily ignored. As for the measurement covariance matrix, the implementation is the same as framework 1.

5. SIMULATION OF FRAMEWORK 1

5.1 Data Preparation

Sensor data and video footage used in this project is provided by Sander Geophysics Ltd., an Ottawa based company specialized in airborne geophysical survey.

The essential information for range calculation, besides the disparity estimation, is the camera motion and camera intrinsic parameters. The camera intrinsic parameters are obtained through camera calibration using the same model of camera as the one which captured the test video. Camera translational velocity is obtained through converting the GPS data into camera coordinate. Camera rotational velocity is directly obtained from the roll, pitch, and magnetic heading recording of the navigation system of the aircraft. However, due to the fact that GPS and INS sensor data are recorded at 10Hz which is different from video frames at 30Hz, and the video frames are only time stamped to the second, some data preparation must be carried out before they can be used in this project. The following steps were done to prepare the data for this project:

- Produce a look up table with a time stamp in second and its associated starting frame number.
- The frame number is aligned to the sensor data through a linear fit on the time stamps.
- Sensor data are interpolated to every frame using the *spline* function in Matlab.

To convert GPS data to camera translational velocity, the following steps were done:

- Convert GPS data to UTM format
- Transform the UTM format data into camera coordinate as shown in figure5 and using equations (17)
- Obtain T_z and $[R_x, R_y, R_z]$ by calculating the element wise difference of the corresponding sensor data.

$$\begin{aligned} T_y &= r\cos(\theta + \varphi - \frac{\pi}{2}) \\ T_x &= r\sin(\theta + \varphi - \frac{\pi}{2}) \\ \theta &= magnetic\ heading - 13.7° \end{aligned} \qquad (17)$$

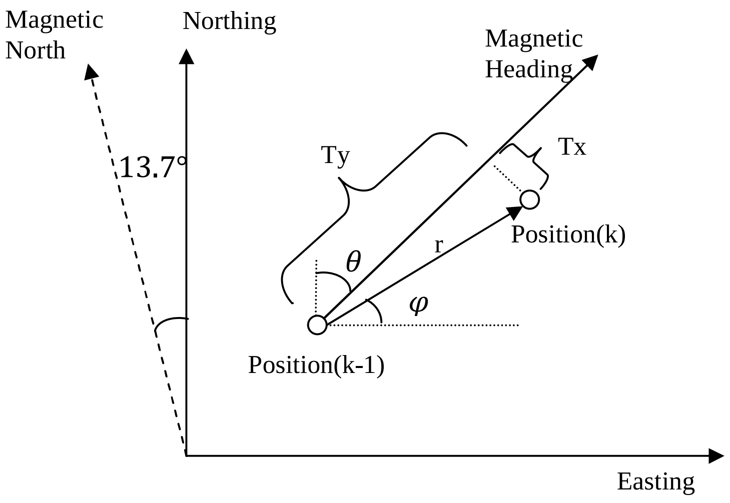

Figure 5 GPS to camera translational velocity conversion

5.2 Algorithm Performance

The algorithm is tested against a piece of real aerial video. The camera is mounted at the nose of a small fixed wing aircraft with its optical axis pointing downwards to the ground, and Y axis of the coordinate is approximately aligned with aircraft heading. Figure 6 shows a series of screen captures of a block of feature points being tracked by the Kalman filter. The aqua points mark $\hat{x}_{k-1}^{-}(1:2)$, blue points mark $\hat{x}_{k}^{-}(1:2)$, white points mark $\hat{x}_{k}^{+}(1:2)$, and read points mark $z_k(1:2)$. In figure6(a), the filter is correcting the predicted state vector heavily with the new measurements because of the small covariance of the measurements. Moving into figure6(b), the optical flow algorithm produced a few outliers, and the predicted state vectors were kept for those points. In figure6(c), the optical flow is even more unreliable due to the illumination changes from frame 14 to 15. These unreliable results are completely discarded by the filter, and the updated state vectors are mostly determined from the predicted vectors. In both (b) and (c), the error constraint applied, and the updated state vectors were fed to the optical flow algorithm as input points. Finally in figure6(d), the optical flow algorithm regain its ground, and produce highly reliable results, and these result are taken into the updated vector with great emphasis again by the filter.

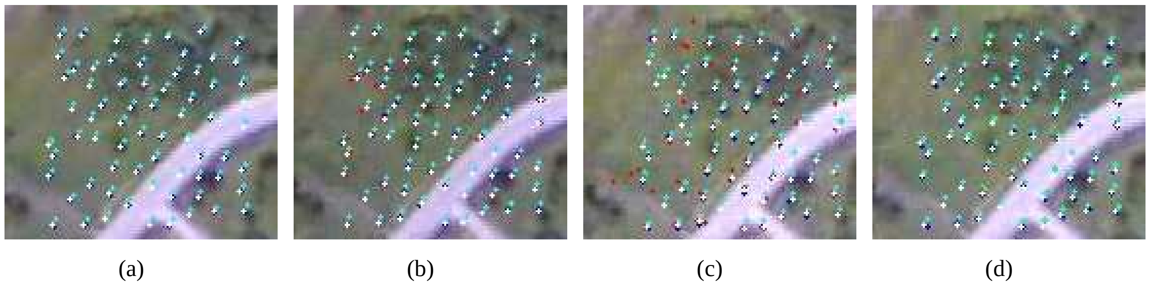

Figure 6 Framework 1 performance. (a) frame 13, (b) frame 14, (c) frame 15, (d) frame 16. The aqua points mark $\hat{x}_{k-1}^{-}[1:2]$, blue points mark $\hat{x}_{k}^{-}[1:2]$, white points mark $\hat{x}_{k}^{+}[1:2]$, and read points mark the $z_k[1:2]$

Figure7 shows a comparison of *maximum* disparity variance that is being estimated by Kalman filter vs. the block variance of the raw optical flow output. It can be seen that the illumination change between frame 14 and 15, which induced big error in optical flow algorithm, has little impact on the Kalman estimation.

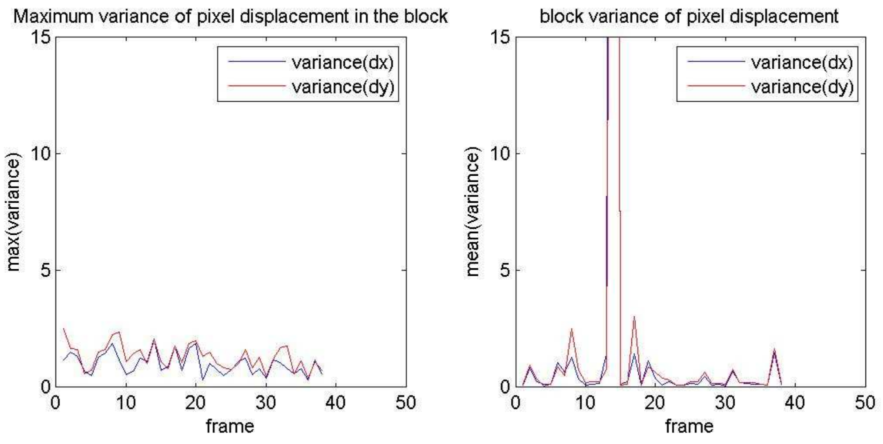

Figure 7 Variance comparison. Left: maximum disparity variance in the block. Right: block variance of raw optical flow output.

5.3 Range calculation

Using equation (2), feature distance in Z axis can be calculated from the disparity. The result is compared to the radar altimeter result, and is shown in figure 8. The estimated Z has a good correlation with the radar data, but with fairly big error. The error can be caused by (1) the region for which we are estimating the distance is not the same region that radar altimeter is measuring, and (2) the camera Y axis does not exactly align with the aircraft magnetic heading, therefore introduce error in the calculation of T_x and T_y

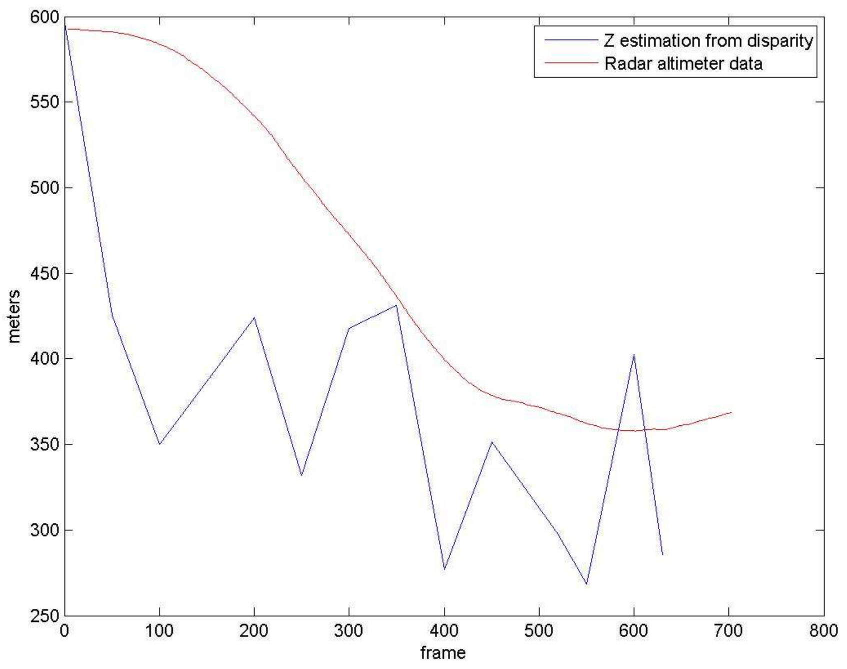

Figure 8 Distance in Z direction calculated from estimated disparity

6. FUTURE WORK

This project has enabled me to understand Kalman filter, apply it to motion stereo problem, and experiment with its performance. The test result of framework 1 has proven the advantages of using Kalman filter in motion stereo problem. To further exploit the potential of Kalman filter on motion stereo problem, I intend to complete the implementation and simulation of framework 2. The 2nd framework described in this report has advantage over the 1st framework as it fuses information from two distinctive sources. In addition, the 2nd framework is better model for describing a wider range of aircraft maneuver.

7. CONCLUSION

This paper presented two Kalman filter frameworks applied to motion stereo problem. The first framework is a linear system. Its extrapolation phase is based entirely on the knowledge of previous optical flow. Therefore, this framework is fusing the current information with the past. On the other hand, the second framework integrates the GPS and INS data into the extrapolation stage of the framework. Because of this integration, the system is nonlinear, and must be linearized for Kalman filter to be applied. The advantage of the second framework is that it fuses information from two distinctive sources.

The first framework is implemented and tested. From the result of the simulation, it is concluded that optical flow algorithm built on Kalman filter framework is more reliable and stable than a single optical flow algorithm by itself. The Kalman filter based optical flow algorithm is also more tolerated to temporary illumination change which occurs very frequently in the image sequence.

A large portion of the second framework is also implemented, except the error propagation of the sensor data to the state vector. More research is required to model this error propagation correctly. Due to the advantage of the second framework, and the great potential of applying Kalman filter on motion stereo problem, more work will be devoted into completing and testing of framework 2.

8. REFERENCES

[1] Bhanu, B., Das, S., Symosek, P., Snyder, S., and Roberts, B., “Synergism of Binocular and Motion Stereo for Passive Ranging”, IEEE Transactions on Aerospace and Electronic Systems 30(3), 709 – 721 (1994) .

[2] Kaehler, G., Bradski, A., [Learning OpenCV], O'Reilly, 2008.

[3] Hoff, W., Sklair, C., "Terrain shape estimation from optical flow using Kalman filtering." Proceedings of the SPIE - The international society for Optical Engineering 1260, 86-97 (1990)

[4] Matthies, L., Kanade, T., and Szeliski, R., “Kalman filter-based algorithms for estimating depth from image sequences,” International Journal of Computer Vision 3, 209-236 (1989).

[5] Galko, P., [Stochastic Processes Course Notes], University of Ottawa, Ottawa, 107 (2009).

[6] Payeur, P., www.site.uottawa.ca/~ppayeur/ELG5163. (2009)