Multiple View Motion Estimation and Control for Landing an Unmanned Aerial Vehicle

Omid Shakernia[†], René Vidal[†], Courtney S. Sharp[†], Yi Ma[‡], Shankar Sastry[†]

[†]Department of EECS, UC Berkeley
Berkeley, CA 94720
{omids,rvidal,cssharp,sastry}@eecs.berkeley.edu

[‡]Department of ECE, UIUC
Urbana, IL 61801
yima@uiuc.edu

Abstract

We present a multiple view algorithm for vision based landing of an unmanned aerial vehicle. Our algorithm is based on our recent results in multiple view geometry which exploit the rank deficiency of the so called *multiple view matrix*. We show how the use of multiple views significantly improves motion and structure estimation. We compare our algorithm to our previous linear and non-linear two-view algorithms using an actual flight test. Our results show that the vision-based state estimates are accurate to within 7cm in each axis of translation and 4 degrees in each axis of rotation.

1 Introduction

The problem of using computer vision to estimate the motion of an unmanned aerial vehicle (UAV) relative to a landing target has recently been an active topic of research [8, 9, 10, 11, 15]. The problem can be considered as a special case of the structure from motion problem in computer vision, in which all the feature points lie on a plane. This makes the problem a degenerate case since the generic eight-point algorithm for two views [3] fails to work. Therefore, a special algorithm which explicitly uses the knowledge that all feature points are coplanar has to be used. This specialized version of the eight-point algorithm is based on the *homography constraint*, and has received much attention in the computer vision literature [1, 4, 14].

In [11], we presented two customized algorithms (linear and nonlinear) for solving the problem. The linear algorithm is a modified version of the homography-based algorithm in which the feature points are known. The nonlinear algorithm is based on a Newton-Raphson iteration which minimizes the *re-projection error* of the feature points. The linear optimization algorithm is globally robust but is biased in the presence of noise.

Figure 1: Berkeley UAV test-bed with on-board vision system: Yamaha R-50 helicopter with pan/tilt camera and computer box hovering above a landing platform.

The nonlinear optimization algorithm requires adequate initialization because of the many local minima, but is less sensitive to noise. Thus, to solve the motion estimation problem, we used the solution of the linear algorithm to initialize the nonlinear algorithm. Additionally, in [11] we introduced the design and implementation of a real-time vision system that estimates the motion of a rotor-craft UAV relative to a known landing target. The vision system on board the UAV (see Figure 1) uses *customized* vision algorithms and *off-the-shelf* hardware to perform in real-time: image processing, segmentation, feature point extraction,

camera control, as well as both linear and nonlinear optimization algorithms for motion estimation.

In this paper, we extend the results of [11] with the following contributions:

- **Multiple view motion estimation:** We present an algorithm for estimating the motion of the UAV from multiple views of a landing target. The algorithm provides more robust motion estimates than the previously used two-view algorithms because it incorporates all algebraically independent constraints among multiple views.
- **Vision based control:** We have placed the vision sensor in the control loop of a hierarchical flight management system [13] and performed fully autonomous vision-based landing.

Our work is differentiated from other recent works in structure and motion estimation from multiple image sequences in the following significant respect: The multiple views in our work come from a *single* camera undergoing an *unknown* motion, as compared to works such as [2, 7] where multiple cameras placed at known relative configurations are used to synchronously capture images of the scene.

2 Motion Estimation

In this section, we extend the existing 2-view algorithms to the case of multiple views. A new fundamental tool needed is the so-called rank condition proposed in [5, 6] and we here show how to apply such a condition to the planar case.

2.1 Multiple Views of Planar Features

An image $\mathbf{x}(t) = [x(t), y(t), z(t)]^T \in \mathbb{R}^3$ (in homogeneous coordinates) of a point p, with homogeneous coordinates $\mathbf{X} = [X, Y, Z, 1]^T \in \mathbb{R}^4$ relative to a fixed world coordinate frame, taken by a moving camera satisfies the following relationship

$$\lambda(t)\mathbf{x}(t) = A(t)Pg(t)\mathbf{X}, \tag{1}$$

where $\lambda(t) \in \mathbb{R}_+$ is the (unknown) depth of the point p relative to the camera frame, $A(t) \in SL(3)$ is the camera calibration matrix (at time t), $P = [I, 0] \in \mathbb{R}^{3\times4}$ is the constant projection matrix and $g(t) \in SE(3)$ is the coordinate transformation from the world frame to the camera frame at time t.

We further assume that the feature point p lies on a plane $\mathbf{P}$. This plane can be described by an *unknown* vector $\pi = [\pi^1, \pi^2] \in \mathbb{R}^4$, where $\pi^1 \in \mathbb{R}^3$ is the unit normal to $\mathbf{P}$, and $\pi^2 \in \mathbb{R}$ is the distance from $\mathbf{P}$ to the camera optical center. The coordinates $\mathbf{X}$ of any point on this plane satisfy

$$\pi\mathbf{X} = 0. \tag{2}$$

In a realistic situation, we obtain "sampled" images of $\mathbf{x}(t)$ at some time instances, say $t_1, t_2, \ldots, t_m \in \mathbb{R}$. For simplicity we denote $\lambda_i = \lambda(t_i)$, $\mathbf{x}_i = \mathbf{x}(t_i)$, $\Pi_i = A(t_i)Pg(t_i)$. The matrix $\Pi_i \in \mathbb{R}^{3\times4}$ relates the i^{th} image of the point p to its world coordinates $\mathbf{X}$ by

$$\lambda_i\mathbf{x}_i = \Pi_i\mathbf{X} \quad i = 1, \ldots, m. \tag{3}$$

Without loss of generality, we may assume that the first camera frame is chosen to be the reference frame. This gives the projection matrices

$$\Pi_1 = [I, 0],\ \Pi_2 = [R_2, T_2], \ldots\ \Pi_m = [R_m, T_m] \in \mathbb{R}^{3\times4},$$

where the columns of $R_i \in \mathbb{R}^{3\times3}$ are the first three columns of Π_i and $T_i \in \mathbb{R}^3$ is the fourth column of $\Pi_i, i = 2, \ldots, m$. Thus, from (3) we have $\lambda_i\mathbf{x}_i = \lambda_1 R_i\mathbf{x}_1 + T_i$, which implies[1]:

$$0 = [\widehat{\mathbf{x}}_i R_i\mathbf{x}_1\ \widehat{\mathbf{x}}_i T_i] \begin{bmatrix} \lambda_1 \\ 1 \end{bmatrix}, \quad i = 2, \ldots, m. \tag{4}$$

Now, from (4) and (2) we have the condition $M[\lambda_1, 1]^T = 0$, where the matrix

$$M \doteq \begin{bmatrix} \widehat{\mathbf{x}}_2 R_2\mathbf{x}_1 & \widehat{\mathbf{x}}_2 T_2 \\ \vdots & \vdots \\ \widehat{\mathbf{x}}_m R_m\mathbf{x}_1 & \widehat{\mathbf{x}}_m T_m \\ \pi^1\mathbf{x}_1 & \pi^2 \end{bmatrix} \in \mathbb{R}^{(3m-2)\times2}, \tag{5}$$

called the *multiple view matrix*, satisfies $\text{rank}(M) \leq 1$.

It is easy to see that the case $\text{rank}(M) = 0$ corresponds to a degenerate configuration in which the camera centers lie on the plane $\mathbf{P}$. So the only interesting case is when $\text{rank}(M) = 1$. In this case the columns of M are parallel, which implies that the images satisfy the well-known bilinear (epipolar) constraints [3]:

$$\mathbf{x}_i^T \widehat{T}_i R_i\mathbf{x}_1 = 0. \tag{6}$$

Now, given vectors $a_1, \ldots, a_n, b_1, \ldots, b_n \in \mathbb{R}^3$, the matrix $\begin{bmatrix} a_1 & b_1 \\ \vdots & \vdots \\ a_n & b_n \end{bmatrix} \in \mathbb{R}^{3n\times2}$ is rank deficient if and only if $a_i b_j^T - b_i a_j^T = 0$ for all $i, j = 1, \ldots, n$. Applying this

[1] For a vector $u \in \mathbb{R}^3$, $\widehat{u} \in \mathbb{R}^{3\times3}$ denotes the associated skew-symmetric matrix such that $\widehat{u}w = u \times w$ for all $w \in \mathbb{R}^3$.

to the multiple view matrix M, we obtain the well-known trilinear constraints [12]:

$$\widehat{\mathbf{x}}_i(T_i\mathbf{x}_1^T R_j^T - R_i\mathbf{x}_1 T_j^T)\widehat{\mathbf{x}}_j = 0 \tag{7}$$

for all $i, j = 2, \ldots, m$. In addition to those, we also obtain extra constraints from minors of M that include the last row, which are due to the planar condition:

$$\widehat{\mathbf{x}}_i T_i \pi^1 \mathbf{x}_1 - \widehat{\mathbf{x}}_i R_i \mathbf{x}_1 \pi^2 = 0, \quad i = 2, \ldots, m. \tag{8}$$

If the plane $\mathbf{P}$ does not cross the camera center o_1, *i.e.* $\pi^2 \neq 0$, the constraints in (8) give the well-known homography constraints for planar feature points:

$$\widehat{\mathbf{x}}_i \left(R_i - \frac{1}{\pi^2} T_i \pi^1 \right) \mathbf{x}_1 = 0, \quad i = 2, \ldots, m \tag{9}$$

between the 1^{st} and the i^{th} views. The matrix $H = \left(R_i - \frac{1}{\pi^2} T_i \pi^1\right)$ in the equation is the well-known homography matrix between the two views of a plane [4].

2.2 Multiple View Motion Estimation

The rank condition on M for coplanar points allows us to utilize simultaneously all multilinear constraints and homography among multiple images for recovering 3-D motion and structure. Existing algorithms for planar features usually exploit only the homography which is only part of all the constraints among multiple images and can only be used for pairwise views.

As described in [5], the problem of motion and structure reconstruction of a set of planar features from multiple images can be solved with a slight modification of the generic multiple view algorithm [6]. For simplicity, we assume that the camera is perfectly calibrated, hence $\Pi_i = (R_i, T_i) \in SE(3)$ corresponds to the actual Euclidean motion of the camera.

Suppose that m images $\mathbf{x}_1^j, \ldots, \mathbf{x}_m^j$ of n points p^j, $j = 1, \ldots, n$ lying on a plane are given and we want to use them to estimate the unknown projection matrices Π_i and parameters π of the plane $\mathbf{P}$. Setting $\alpha^j = 1/\lambda_1^j$, the rank condition on M can be written as:

$$\alpha^j \begin{bmatrix} \widehat{\mathbf{x}}_2^j T_2 \\ \vdots \\ \widehat{\mathbf{x}}_m^j T_m \\ \pi^2 \end{bmatrix} + \begin{bmatrix} \widehat{\mathbf{x}}_2^j R_2 \mathbf{x}_1^j \\ \vdots \\ \widehat{\mathbf{x}}_m^j R_m \mathbf{x}_1^j \\ \pi^1 \mathbf{x}_1^j \end{bmatrix} = 0. \tag{10}$$

The set of equations in (10) is equivalent to finding vectors $\pi \in \mathbb{R}^4$, $\vec{R}_i = [r_{11}, r_{12}, r_{13}, r_{21}, r_{22}, r_{23}, r_{31}, r_{32}, r_{33}]^T \in \mathbb{R}^9$ and $\vec{T}_i = T_i \in \mathbb{R}^3$, $i = 2, \ldots, m$, such that:

$$Q\pi^T = \begin{bmatrix} \mathbf{x}_1^{1T} & \alpha^1 \\ \vdots & \vdots \\ \mathbf{x}_1^{nT} & \alpha^n \end{bmatrix} \pi^T = 0, \tag{11}$$

$$P_i \begin{bmatrix} \vec{T}_i \\ \vec{R}_i \end{bmatrix} = \begin{bmatrix} \alpha^1 \widehat{\mathbf{x}}_i^1 & \widehat{\mathbf{x}}_i^1 * \mathbf{x}_1^{1T} \\ \vdots & \vdots \\ \alpha^n \widehat{\mathbf{x}}_i^n & \widehat{\mathbf{x}}_i^n * \mathbf{x}_1^{nT} \end{bmatrix} \begin{bmatrix} \vec{T}_i \\ \vec{R}_i \end{bmatrix} = 0 \tag{12}$$

where $A * B$ is the *Kronecker product* of A and B.

Given the first two images of (at least) four points in general configuration, $\pi \in \mathbb{R}^4$, $T_2 \in \mathbb{R}^3$ and $R_2 \in SO(3)$ can be estimated using the standard *four point planar algorithm* for two views [1]. In general, there are two physically possible solutions for (π, R_2, T_2) from the four point algorithm, with π^2 and T_2 recovered up to the same scale.[2] Given these two solutions for (π, R_2, T_2), we can solve for α from the equations in the first and last rows of (10). These equations are $\alpha^j \widehat{\mathbf{x}}_2^j T_2 = -\widehat{\mathbf{x}}_2^j R_2 \mathbf{x}_1^j$ and $\alpha^j \pi^2 = -\pi^1 \mathbf{x}_1^j$, whose least squares solution up to scale (the inverse of the common scale of π^2 and T_2) for each j is given by:

$$\alpha^j = -\frac{(\widehat{\mathbf{x}}_2^j T_2)^T \widehat{\mathbf{x}}_2^j R_2 \mathbf{x}_1^j + \pi^2 \pi^1 \mathbf{x}_1^j}{||\widehat{\mathbf{x}}_2^j T_2||^2 + (\pi^2)^2}. \tag{13}$$

Since there are two possible values for (π, R_2, T_2) from the four point planar algorithm, there are two possible values for α. Given these two values for α, equations (11) and (12) become linear, thus one can solve for the rest of (R_i, T_i) and re-estimate π. Therefore, in principle there are two possible solutions for (π, R_i, T_i) provided that $\text{rank}(P_i) = 11$ and $\text{rank}(Q) = 3$. One can show that this is indeed the case if at least 6 feature points are in a general position in 3-D. However, since here all points lie on the same plane, the maximum rank of P_i becomes 8 instead, while the rank of Q is always 3 for points in a general configuration on the plane. It is straightforward to verify that the solution $[\vec{T}_i^T, \vec{R}_i^T]^T \in \mathbb{R}^{12}$ is in the four dimensional kernel of P_i which is spanned by the columns of the following matrix:

$$K_i \doteq \begin{bmatrix} \pi^2 & 0 & 0 & 0 \\ 0 & \pi^2 & 0 & 0 \\ 0 & 0 & \pi^2 & 0 \\ \pi^{1T} & 0_{3\times1} & 0_{3\times1} & r_1 - \frac{T_{i1}}{\pi^2}\pi^{1T} \\ 0_{3\times1} & \pi^{1T} & 0_{3\times1} & r_2 - \frac{T_{i2}}{\pi^2}\pi^{1T} \\ 0_{3\times1} & 0_{3\times1} & \pi^{1T} & r_3 - \frac{T_{i3}}{\pi^2}\pi^{1T} \end{bmatrix} \in \mathbb{R}^{12\times4}, \tag{14}$$

where $[r_1^T, r_2^T, r_3^T]^T \doteq \vec{R}_i$ and $[T_{i1}, T_{i2}, T_{i3}]^T \doteq \vec{T}_i$. The last column yields exactly the homography matrix $H_i \doteq (R_i - \frac{1}{\pi^2} T_i \pi^1) \in \mathbb{R}^{3\times3}$ between the i^{th} and the 1^{st} views. Therefore, given the two values for α, one can find the homography H_i from the vector in the null-space of P_i whose first three components are zero. Given the two homographies H_i's, one can obtain two solutions (π_i, R_i, T_i) from the four point algorithm, where π_i is the plane with respect to frame

[2] This scale can easily be fixed by choosing $\|T_2\| = 1$.

1 estimated from frames 1 and i. Since $\pi_i^1 = \pi_2^1$, there are only two solutions for the plane π and all relative motions $(R_i, T_i), i = 2, \ldots, m$, rather than the 2^{m-1} possible combinations. Furthermore, if $m \geq 3$, one can show that only one of these two solutions satisfies $\pi_i^1 = \pi_2^1, i = 3, \ldots, m$. Therefore, we conclude that there are two solutions for $m = 2$ and a unique solution for $m \geq 3$.

Finally, recall that π_i^2 and T_i are recovered up to the same scale. However, in the multiple view case all translation vectors and π^2 should be recovered up to one scale only. It is straightforward to see that the relative scale between T_i and T_2 is π_2^2/π_i^2. Therefore, we have the following linear algorithm for multiple view motion and structure estimation from planar feature points:

Algorithm 1 (Multiple View Planar Algorithm) *Given m images $\mathbf{x}_1^j, \ldots, \mathbf{x}_m^j$ of points p^j, $j = 1, \ldots, n$, which lie in a plane in 3-D space, we can estimate the motions $(R_i, T_i) \in SE(3)$, $i = 2, \ldots, m$, the plane π and the inverse depth α as follows:*

1. *Initialization*
   (a) *Set k=0 and find the two solutions for (π_2, R_2, T_2) using the four point algorithm applied to the first two views.*
   (b) *Compute the two solutions for α_k^j from (13). Normalize so that $\alpha_k^1 = 1$.*
2. *Find the two solutions for the homography matrix $H_i, i = 2, \ldots, m$, from the vector in the kernel of P_i whose first three components are zero.*
3. *Find the two solutions for $(\pi_i, R_i, \tilde{T}_i)$ from H_i, $i = 2, \ldots, m$, using the four point algorithm.*
4. *If $m \geq 3$, find the unique solution that satisfies $\|\pi_i^1 - \pi_2^1\| \leq \epsilon, i = 3, \ldots, m$, for some small $\epsilon > 0$.*
5. *Let $\pi = [\pi^1, 1]$, where π^1 is recovered from SVD of all $\pi_i^1 \in \mathbb{R}^3$ for $i = 1, \ldots, m$. Let $T_i = \tilde{T}_i \pi_2^2/\pi_i^2, i = 3, \ldots, m$.*
6. *Solve for α from (10) using linear least squares:*

$$\alpha_{k+1}^j = -\frac{\sum_{i=2}^m (\widehat{\mathbf{x}}_i^j T_i)^T \widehat{\mathbf{x}}_i^j R_i \mathbf{x}_1^j + \pi^2 \pi^1 \mathbf{x}_1^j}{\sum_{i=2}^m ||\widehat{\mathbf{x}}_i^j T_i||^2 + (\pi^2)^2}. \quad (15)$$

   Normalize so that $\alpha_{k+1}^1 = 1$.
7. *If $||\alpha_k - \alpha_{k+1}|| < \epsilon$, for a pre-specified $\epsilon > 0$, then $k = k + 1$ and goto 2. Else stop.*

The camera motion is then $(R_i, T_i) \in SE(3)$ for $i = 2, \ldots, m$, the plane is π, and the structure of the points (with respect to the first camera frame) is given by the converged depth scalar $\lambda_1^j = 1/\alpha^j$, $i = 1, \ldots, n$.

2.3 Multi-view Motion For Landing

Here we describe how we apply Algorithm 1 to the landing problem. As described briefly in the following section (and in detail in [11]), we have designed a landing target to simplify the image processing and feature extraction tasks. We set the first frame in Algorithm 1 as the reference frame with the stored image of the landing target with the UAV at a desired landing configuration. For each captured image, we set $\mathbf{x}_m^j$, $j = 1, \ldots, n$, to be the feature points extracted from the current view, and $\mathbf{x}_{m-i}^j$, $i = 1, \ldots, m-2$, to be the feature points extracted from the previous views. This setup has two very nice properties: (1) $(R_m, T_m) \in SE(3)$ is the relative Euclidean motion from the desired landing configuration to the current camera frame, (2) knowledge of the landing target geometry allows to *uniquely* recover $T_m \in \mathbb{R}^3$, instead of up to an arbitrary scale.

3 Vision System Test-bed

This section briefly describes our real-time vision system [11]. Further, we describe our UAV test-bed [13] and computer vision based supervisory controller used in a hierarchical flight management system for autonomous landing.

3.1 Vision System Software

Our vision system software consists of two main stages of execution: feature extraction and motion estimation. The function of the feature extraction is to *extract* and *label*, for each frame m, the images $\mathbf{x}_m^j$, $j = 1, \ldots, n$, of points on a landing target. We use corner features on a specially designed target to simplify the feature extraction stage [11]. Figure 2 shows our designed landing target and feature extraction in action.

Given the feature points $\mathbf{x}_m^j$, we have developed and implemented different methodologies for estimating the UAV's motion relative to the landing target: The two-view linear and two-view nonlinear algorithms described in [11], and the multiple view linear algorithm described in Section 2.

3.2 Vision System Hardware

Our real-time vision system consists of the following off-the-shelf hardware components [11]:

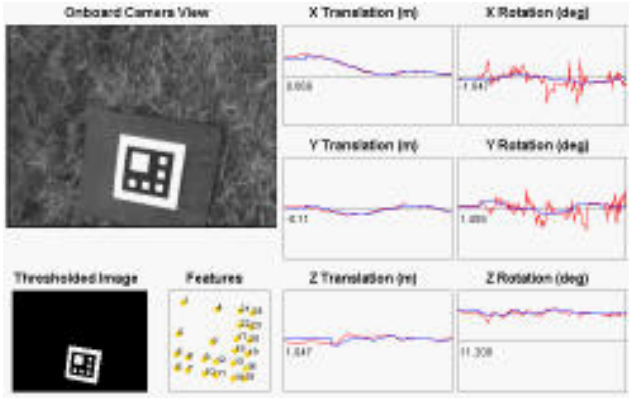

Figure 2: Vision monitoring station

- *Vision computer*: Pentium 233MHz-based LittleBoard PC running Linux; responsible for vision algorithms and camera control.
- *Camera*: Sony EVI-D30 Pan/Tilt/Zoom camera that can actively pan and tilt to keep the landing target centered in the field of view.
- *Framegrabber*: Imagenation PXC200 for capturing 320×240 resolution images at 30Hz.
- *Wireless Ethernet*: IEEE 802.11b for monitoring vision system from ground.

3.3 UAV Test-bed

Our custom-designed UAV test-bed is based on a Yamaha R-50 industrial helicopter, on which we have mounted the following [13]:

- *Navigation Computer*: Pentium 233MHz-based LittleBoard PC running QNX real-time OS, responsible for sensor management and hard real-time flight control.
- *Inertial Measurement Unit*: NovAtel MillenRT2 GPS and Boeing DQI-NP INS/GPS integration system, with 2cm accuracy motion estimates.

The flight control system is capable of autonomous hover, pirouette, and low-speed flight with fixed heading. The interface to the flight control is a novel framework called Vehicle Control Language (VCL) [13], which specifies a sequence of flight-modes and desired coordinates. VCL provides an abstraction between a high-level supervisory controller and the hard real-time vehicle stabilization and control layer.

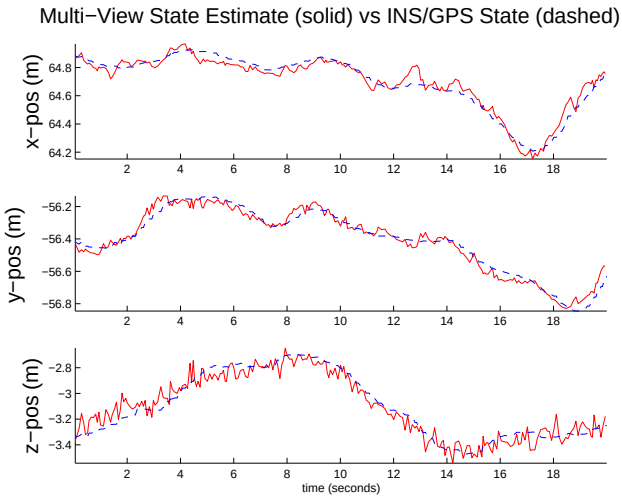

Figure 3: Comparing multiple view planar motion algorithm estimates with inertial navigation system measurements in a real flight test.

3.4 Vision in Control Loop

We designed a simple vision-based supervisory controller which commands the UAV to hover above the visually estimated location of the landing target. The supervisory controller runs on the vision computer, and sends control commands to the navigation computer over a serial link at an update rate of 10Hz. When the landing target is in camera view, the supervisory controller sends the estimated (x, y) position of the landing target as the desired set-point for the UAV to hover.

4 Experimental Results

For the flight test, the UAV hovered autonomously above a stationary landing pad with the vision system running in real-time. The vision-based state estimates were used by the supervisory controller to command the UAV to hover above the landing target, making it a truly closed-loop vision controlled flight experiment. State estimates from the INS/GPS navigation system were synchronously gathered for later comparison.

Figure 3 shows the results from a flight test, comparing the output of the multiple view motion estimation algorithm with the INS/GPS measurements of the UAV navigation system (which are accurate to 2cm). Recall that at least 3 frames are necessary in the multiple view algorithm to uniquely estimate the motion. We observed that using more than 4 frames did not improve the accuracy of the motion estimates, and only increased computation time. Thus, we used a moving

of 3 frames plus the reference frame in the multiple view algorithm for the experiments.

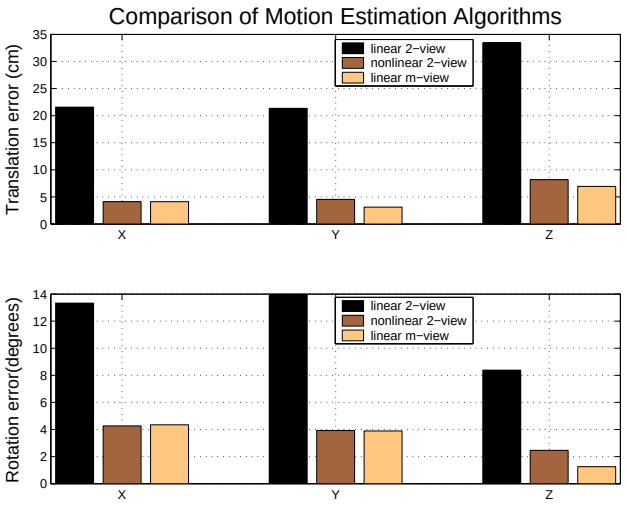

Figure 4: Comparison of mean squared errors of vision algorithm estimates during a real flight test. The 'linear 2-view' and 'nonlinear 2-view' algorithms are described in [11]. The 'linear m-view' algorithm is described in Section 2.

Figure 4 shows a comparison of the the root mean squared error of the estimated state for three different vision-based state estimation algorithms: The "linear 2-view" and "nonlinear 2-view" algorithms which were presented in [11], and the "linear m-view" algorithm described in Section 2. The multiple view algorithm slightly outperforms the nonlinear algorithm. Further, the multiple view algorithm is globally robust, while the nonlinear algorithm has many local minima and is sensitive to initialization. The performance and robustness of the multiple view algorithm make it the clear winner among the algorithms.

5 Conclusion

In this paper, we presented a novel multiple view motion estimation algorithm for autonomous landing of an Unmanned Aerial Vehicle. The algorithm is based on very recent results in multiple view geometry which exploit the rank deficiency condition of the *multiple view matrix*. We compared this algorithm with previous linear and non-linear two-view algorithms using an actual flight test of our real-time vision system. Flight test results show that the use of multiple images significantly improves the robustness and the accuracy of vision-based motion estimates, which are are accurate to within 7cm in each axis of translation and 4^o in each axis of rotation.

Acknowledgment

We thank David Shim, Hoam Chung and Ron Tal for their support in the performing the flight experiments. This research was supported by ONR grant N00014-00-1-0621.

References

[1] O. Faugeras. *Three-Dimensional Computer Vision*. The MIT Press, 1993.

[2] A. Hoover and B. Olsen. A real-time occupancy map from multiple video streams. In *Proceedings of IEEE ICRA*, pages 2261–2266, 1999.

[3] H.C. Longuet-Higgins. A computer algorithm for reconstructing a scene from two projections. In *Nature*, volume 293, pages 133–135, London, UK, 1981.

[4] H.C. Longuet-Higgins. The reconstruction of a plane surface from two perspective projections. In *Proc. of Royal Society of London*, volume 227 of *B*, pages 399–410, 1986.

[5] Y. Ma, J. Kosecka, and K. Huang. Rank deficiency condition of the multiple view matrix for mixed point and line features. In *Proc. of fifth Asian Conference on Computer Vision*, January 2002.

[6] Y. Ma, R. Vidal, K. Huang, and S. Sastry. New rank deficiency condition for multiple view geometry of point features. Technical Report UILU-ENG 01-2208 (DC-200), UIUC, May 2001.

[7] P.W. Rander, P.J. Narayanan, and T. Kanade. Recovery of dynamic scene structure from multiple image sequences. In *Int'l Conf. on Multisensor Fusion and Integration for Intelligent Systems*, pages 305–312, 1996.

[8] F.R. Schell and E.D. Dickmanns. Autonomous landing of airplanes by dynamic machine vision. *Machine Vision and Applications*, 7:127–134, 1994.

[9] O. Shakernia, Y. Ma, T.J. Koo, J. Hespanha, and S. Sastry. Vision guided landing of an unmanned air vehicle. In *Proceedings of IEEE CDC*, pages 4143–4148, 1999.

[10] O. Shakernia, Y. Ma, T.J. Koo, and S. Sastry. Landing an unmanned air vehicle: Vision based motion estimation and nonlinear control. *Asian Journal of Control*, 1(3):128–145, September 1999.

[11] C.S. Sharp, O. Shakernia, and S. Sastry. A vision system for landing an unmanned aerial vehicle. In *Proceedings of IEEE ICRA*, pages 1720–1727, May 2001.

[12] A. Shashua. Trilinearity in visual recognition by alignment. In *Proceedings of ECCV*, pages 479–484, 1994.

[13] D.H. Shim, H.J. Kim, and S. Sastry. Hierarchical control system synthesis for rotorcraft-based unmanned aerial vehicles. In *Proceedings of AIAA Conference on Guidance, Navigation and Control*, Denver, 2000.

[14] J. Weng, T.S. Huang, and N. Ahuja. *Motion and Structure from Image Sequences*. Springer-Verlag, 1993.

[15] Z.F. Yang and W.H. Tsai. Using parallel line information for vision-based landmark location estimation and an application to automatic helicopter landing. *Robotics and Computer-Integrated Manufacturing*, 14(4):297–306, 1998.