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

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

[†]Department of EECS, UC Berkeley
Berkeley, CA 94720
omids@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^o in each axis of rotation.

1 Introduction

The problem of estimating 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, 12, 19]. 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 [4] 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 [2, 3, 5, 15, 16, 17].

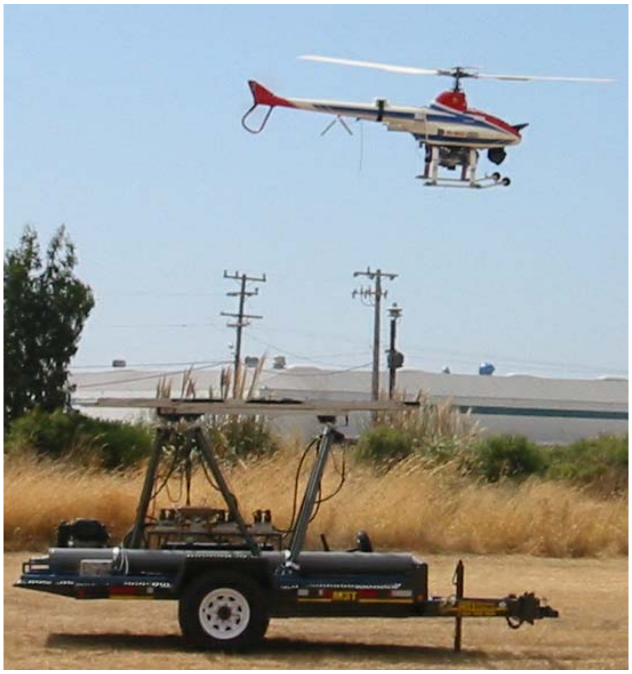

Figure 1: Berkeley UAV test-bed with on-board vision system: Yamaha R-50 helicopter with pan/tilt camera and computer box hovering above a landing platform.

In [12], 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 sensitive to noise. The nonlinear optimization algorithm requires adequate initialization 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 [12] 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 [12] with the following contributions:

- **Multiple view motion estimation:** We have developed and implemented an algorithm for estimating the motion of the UAV from multiple views of a landing target. The algorithm is based on the rank deficiency of the multiple view matrix and provides much more robust motion estimates than the previously used two-view algorithms.
- **Vision based control:** We have placed the vision sensor in the control loop of a hierarchical flight management system [14] and performed vision-based landing in real-time.

Paper Outline: Section 2 introduces the multiple view matrix for planar features and proposes a linear multiple view algorithm for motion estimation. Section 3 describes the real-time vision system for landing the UAV. Section 4 presents flight test results which show the performance of the vision-based motion estimation algorithms, as well as the stability and robustness of the vision-based controller. Section 5 concludes the paper.

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 [6, 7] 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), $P = [I, 0] \in \mathbb{R}^{3\times 4}$ 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 $\mathbf{P}$ in 3-D. This plane can be described by a vector $\pi = [a, b, c, d] \in \mathbb{R}^4$ such that the coordinates $\mathbf{X}$ of any point on this plane satisfy

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

Although we assume such an constraint on the 3-D coordinates $\mathbf{X}$ of p exists, we do not assume that we know π in advance.

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 $\Pi_i \in \mathbb{R}^{3\times 4}$ relates the i^{th} image of the point p to its world coordinates $\mathbf{X}$ by

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

In equations (2) and (3), except for the $\mathbf{x}_i$'s, everything else is unknown and subject to recovery. In general, solving λ_i's, Π_i's, π and $\mathbf{X}$ altogether from such equations is extremely difficult. A natural way to simplify the task is to decouple the recovery of the matrices Π_i's from the recovery of the λ_i's and $\mathbf{X}$. For that purpose, let us rewrite the system of equations (3) and the planar constraint (2) in a single matrix form

$$\begin{bmatrix} \mathbf{x}_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \mathbf{x}_m \\ 0 & \cdots & 0 \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \vdots \\ \lambda_m \end{bmatrix} = \begin{bmatrix} \Pi_1 \\ \vdots \\ \Pi_m \\ \pi \end{bmatrix} \mathbf{X} \tag{4}$$
$$\mathcal{I}\,\vec{\lambda} = \Pi\,\mathbf{X},$$

where we will call $\vec{\lambda} \in \mathbb{R}^{3m}$ the *depth scale vector*, and $\Pi \in \mathbb{R}^{(3m+1)\times 4}$ the *projection matrix* associated to the *multiple image matrix* $\mathcal{I} \in \mathbb{R}^{(3m+1)\times m}$.

In order to eliminate the unknowns $\vec{\lambda}$ and $\mathbf{X}$, we notice that the $m+4$ columns of the matrix $N \doteq [\Pi,\ \mathcal{I}] \in \mathbb{R}^{(3m+1)\times(m+4)}$ are *linearly dependent*, since it is clear from (4) that $v \doteq [\mathbf{X}^T, -\vec{\lambda}^T]^T \in \mathbb{R}^{m+4}$ is in the null space of N. We conclude that $\text{rank}(N) \leq m+3$.

Without loss of generality, we may assume that the first camera frame is chosen to be the reference frame. That gives the projection matrices $\Pi_i, i = 1, \ldots, m$ the general form $\Pi_i = [R_i, T_i] \in \mathbb{R}^{3\times 4}$, where the columns of $R_i \in \mathbb{R}^{3\times 3}$ are the first three columns of Π_i and $T_i \in \mathbb{R}^3$ is the fourth column of $\Pi_i, i = 2, \ldots, m$. Also, given $\pi = [a, b, c, d]$, define $\pi^1 = [a, b, c] \in \mathbb{R}^3$ and $\pi^2 = d \in \mathbb{R}$. Under this assumption matrix N

becomes

$$N \doteq \begin{bmatrix} I & 0 & \mathbf{x}_1 & 0 & \cdots & 0 \\ R_2 & T_2 & 0 & \mathbf{x}_2 & \ddots & \vdots \\ \vdots & \vdots & \vdots & \ddots & \ddots & 0 \\ R_m & T_m & 0 & \cdots & 0 & \mathbf{x}_m \\ \pi^1 & \pi^2 & 0 & \cdots & \cdots & 0 \end{bmatrix}. \tag{5}$$

After a column manipulation (which eliminates $\mathbf{x}_1$ from the first row), it is easy to see that N has the same rank as the matrix

$$\begin{bmatrix} I & 0 & 0 & 0 & \cdots & 0 \\ R_2 & T_2 & R_2\mathbf{x}_1 & \mathbf{x}_2 & \ddots & \vdots \\ \vdots & \vdots & \vdots & 0 & \ddots & 0 \\ R_m & T_m & R_m\mathbf{x}_1 & \vdots & \ddots & \mathbf{x}_m \\ \pi^1 & \pi^2 & \pi^1\mathbf{x}_1 & 0 & \cdots & 0 \end{bmatrix} = \left[\begin{array}{c|c} I & 0 \\ \hline R_2 & \\ \vdots & N' \\ R_m & \\ \pi^1 & \end{array}\right],$$

where N' satisfies that $\text{rank}(N') = \text{rank}(N) - 3 \leq m$. Left multiplying N' by the following matrix[1]

$$D = \begin{bmatrix} \mathbf{x}_2^T & 0 & 0 & 0 \\ \widehat{\mathbf{x}}_2 & 0 & 0 & 0 \\ 0 & \ddots & \vdots & \vdots \\ \vdots & \ddots & \mathbf{x}_m^T & 0 \\ 0 & 0 & \widehat{\mathbf{x}}_m & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \in \mathbb{R}^{(4m-3)\times(3m-2)}$$

yields

$$DN' = \begin{bmatrix} \mathbf{x}_2^T T_2 & \mathbf{x}_2^T R_2\mathbf{x}_1 & \mathbf{x}_2^T\mathbf{x}_2 & 0 & 0 \\ \widehat{\mathbf{x}}_2 T_2 & \widehat{\mathbf{x}}_2 R_2\mathbf{x}_1 & 0 & 0 & 0 \\ \vdots & \vdots & 0 & \ddots & 0 \\ \vdots & \vdots & 0 & \ddots & 0 \\ \mathbf{x}_m^T T_m & \mathbf{x}_m^T R_m\mathbf{x}_1 & 0 & 0 & \mathbf{x}_m^T\mathbf{x}_m \\ \widehat{\mathbf{x}}_m T_m & \widehat{\mathbf{x}}_m R_m\mathbf{x}_1 & 0 & 0 & 0 \\ \pi^2 & \pi^1\mathbf{x}_1 & 0 & 0 & 0 \end{bmatrix}.$$

Since D is of full rank $3m-2$, we conclude that

$$\text{rank}(N') = \text{rank}(DN').$$

Hence the original matrix N is rank deficient if and only if the following sub-matrix of DN'

$$M \doteq \begin{bmatrix} \widehat{\mathbf{x}}_2 R_2\mathbf{x}_1 & \widehat{\mathbf{x}}_2 T_2 \\ \widehat{\mathbf{x}}_3 R_3\mathbf{x}_1 & \widehat{\mathbf{x}}_3 T_3 \\ \vdots & \vdots \\ \widehat{\mathbf{x}}_m R_m\mathbf{x}_1 & \widehat{\mathbf{x}}_m T_m \\ \pi^1\mathbf{x}_1 & \pi^2 \end{bmatrix} \in \mathbb{R}^{(3m-2)\times 2} \tag{6}$$

is rank deficient. Hence we have the following [6]:

Theorem 1 (Planar Rank Deficiency Condition) *For the two matrices $N \in \mathbb{R}^{(3m+1)\times(m+4)}$ and $M \in \mathbb{R}^{(3m-2)\times 2}$, defined in equations (5) and (6) respectively, we always have rank(N) $= (m+2) +$ rank(M). That is rank(N) $= m+3$ if and only if rank(M) $= 1$, and rank(N) $= m+2$ if and only if rank(M) $= 0$.*

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, it is necessary that the images satisfy the bilinear (epipolar) constraints

$$\mathbf{x}_i^T \widehat{T}_i R_i \mathbf{x}_1 = 0 \tag{7}$$

and trilinear constraints

$$\widehat{\mathbf{x}}_i (T_i \mathbf{x}_1^T R_j^T - R_i \mathbf{x}_1 T_j^T) \widehat{\mathbf{x}}_j = 0 \tag{8}$$

for all $i, j = 2, \ldots, m$. In addition to those, we also obtain extra constraints due to the planar condition

$$\widehat{\mathbf{x}}_i T_i \pi^1 \mathbf{x}_1 - \widehat{\mathbf{x}}_i R_i \mathbf{x}_1 \pi^2 = 0, \quad \forall i = 2, \ldots, m. \tag{9}$$

If the plane $\mathbf{P}$ does not cross the camera center o_1, *i.e.* $\pi^2 \neq 0$, the constraints in (9) give the well-known homography constraints for planar feature points:

$$\widehat{\mathbf{x}}_i \left(R_i - \frac{1}{\pi^2} T_i \pi^1 \right) \mathbf{x}_1 = 0, \quad \forall i = 2, \ldots, m \tag{10}$$

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.

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 [6], 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 [7]. For simplicity, we assume that the camera is perfectly calibrated, hence $\Pi = (R, T) \in SE(3)$ corresponds to the actual Euclidean motion of the camera.

Suppose that m images $\mathbf{x}_1^i, \ldots, \mathbf{x}_m^i$ of n points p^i, $i = 1, \ldots, n$ lying on a plane are given and we want to use them to estimate the unknown projection matrix Π

[1] For a three dimensional vector $u \in \mathbb{R}^3$, we use $\widehat{u} \in \mathbb{R}^{3\times 3}$ to denote the skew symmetric matrix associated to u such that for any vector $v \in \mathbb{R}^3$, we have $\widehat{u}v = u \times v$.

and the parameters of the plane π. The rank condition on the M matrix can be written as:

$$\alpha^i \begin{bmatrix} \widehat{\mathbf{x}}_2^i T_2 \\ \widehat{\mathbf{x}}_3^i T_3 \\ \vdots \\ \widehat{\mathbf{x}}_m^i T_m \\ \pi^2 \end{bmatrix} + \begin{bmatrix} \widehat{\mathbf{x}}_2^i R_2 \mathbf{x}_1^i \\ \widehat{\mathbf{x}}_3^i R_3 \mathbf{x}_1^i \\ \vdots \\ \widehat{\mathbf{x}}_m^i R_m \mathbf{x}_1^i \\ \pi^1 \mathbf{x}_1^i \end{bmatrix} = 0 \qquad (11)$$

for proper $\alpha^i \in \mathbb{R}, i = 1, \ldots n$.

From (3) we have $\lambda_j^i \mathbf{x}_j^i = \lambda_1^i R_j \mathbf{x}_1^i + T_j$. Multiplying by $\widehat{\mathbf{x}}_j^i$ we obtain $\widehat{\mathbf{x}}_j^i (R_j \mathbf{x}_1^i + T_j / \lambda_1^i) = 0$. Therefore $\alpha^i = 1/\lambda_1^i$ can be interpreted as the inverse of the depth of point p^i with respect to the first frame. The set of equations in (11) is equivalent to finding vectors $\pi \in \mathbb{R}^4$, $\vec{R_j} = [r_{11}, r_{12}, r_{13}, r_{21}, r_{22}, r_{23}, r_{31}, r_{32}, r_{33}]^T \in \mathbb{R}^9$ and $\vec{T_j} = T_j \in \mathbb{R}^3$, $j = 2, \ldots, m$, such that:

$$Q\pi^T = \begin{bmatrix} \mathbf{x}_1^{1\,T} & \alpha^1 \\ \mathbf{x}_1^{2\,T} & \alpha^2 \\ \vdots & \vdots \\ \mathbf{x}_1^{n\,T} & \alpha^n \end{bmatrix} \pi^T = 0, \qquad (12)$$

$$P_j \begin{bmatrix} \vec{T_j} \\ \vec{R_j} \end{bmatrix} = \begin{bmatrix} \alpha^1 \widehat{\mathbf{x}}_j^1 & \widehat{\mathbf{x}}_j^1 * \mathbf{x}_1^{1\,T} \\ \alpha^2 \widehat{\mathbf{x}}_j^2 & \widehat{\mathbf{x}}_j^2 * \mathbf{x}_1^{2\,T} \\ \vdots & \vdots \\ \alpha^n \widehat{\mathbf{x}}_j^n & \widehat{\mathbf{x}}_j^n * \mathbf{x}_1^{n\,T} \end{bmatrix} \begin{bmatrix} \vec{T_j} \\ \vec{R_j} \end{bmatrix} = 0 \quad (13)$$

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 (11). These equations are $\alpha^i \widehat{\mathbf{x}}_2^i T_2 = -\widehat{\mathbf{x}}_2^i R_2 \mathbf{x}_1^i$ and $\alpha^i \pi^2 = -\pi^1 \mathbf{x}_1^i$, whose least squares solution up to scale (the inverse of the common scale of π^2 and T_2) for each i is given by:

$$\alpha^i = -\frac{(\widehat{\mathbf{x}}_2^i T_2)^T \widehat{\mathbf{x}}_2^i R_2 \mathbf{x}_1^i + \pi^2 \pi^1 \mathbf{x}_1^i}{||\widehat{\mathbf{x}}_2^i T_2||^2 + (\pi^2)^2}. \qquad (14)$$

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 (12) and (13) become linear, thus one can solve for the rest of (R_j, T_j) and re-estimate π. Therefore, in principle there are two possible solutions for (π, R_j, T_j) provided that rank$(P_j) = 11$ and 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_j 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 $[\vec{T_j}^T, \vec{R_j}^T]^T \in \mathbb{R}^{12}$ is in the four dimensional kernel of P_j which is spanned by the columns of the following matrix:

$$K_j \doteq \begin{bmatrix} \pi^2 & 0 & 0 & 0 \\ 0 & \pi^2 & 0 & 0 \\ 0 & 0 & \pi^2 & 0 \\ \pi^{1\,T} & 0_{3\times1} & 0_{3\times1} & r_1 - \frac{T_{j1}}{\pi^2}\pi^{1\,T} \\ 0_{3\times1} & \pi^{1\,T} & 0_{3\times1} & r_2 - \frac{T_{j2}}{\pi^2}\pi^{1\,T} \\ 0_{3\times1} & 0_{3\times1} & \pi^{1\,T} & r_3 - \frac{T_{j3}}{\pi^2}\pi^{1\,T} \end{bmatrix} \in \mathbb{R}^{12\times4}, \qquad (15)$$

where $[r_1^T, r_2^T, r_3^T]^T \doteq \vec{R_j}$ and $[T_{j1}, T_{j2}, T_{j3}]^T \doteq \vec{T_j}$. The last column yields exactly the homography matrix $H_j \doteq (R_j - \frac{1}{\pi^2} T_j \pi^1) \in \mathbb{R}^{3\times3}$ between the j^{th} and the 1^{st} views. Therefore, given the two values for α, one can find the homography H_j from the vector in the null-space of P_j whose first three components are zero. Given the two homographies H_j's, one can obtain two solutions (π_j, R_j, T_j) from the four point algorithm. Since $\pi_j^1 = \pi_2^1$, there are only two solutions for the plane π and all relative motions $(R_j, T_j), j = 2, \ldots, m$, rather than the 2^{m-1} possible combinations. Furthermore, if $m \geq 3$, one can show that only one of these two solutions satisfies $\pi_j^1 = \pi_2^1, j = 3, \ldots, m$. Therefore, we conclude that there are two solutions for $m = 2$ and a unique solution for $m \geq 3$.

Finally, recall that π_j^2 and T_j 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_j and T_2 is π_2^2/π_j^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 $\mathbf{x}_1^i, \ldots, \mathbf{x}_m^i$ of points p^i, $i = 1, \ldots, n$, which lie in a plane in 3-D space, we can estimate the motions $(R_j, T_j) \in SE(3)$, $j = 2, \ldots, m$, the plane π and the inverse depth α as follows:*

1. *Initialization*

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

(a) *Set k=0 and find the two solutions for* (π, R_2, T_2) *using the four point algorithm applied to the first two views.*

(b) *Compute the two solutions for* α_k^i *from (14). Normalize so that* $\alpha_k^1 = 1$*.*

2. *Find the two solutions for the homography matrix* $H_j, j = 2, \ldots, m$*, from the vector in the kernel of* P_j *whose first three components are zero.*

3. *Find the two solutions for* $(\pi_j, R_j, \tilde{T}_j)$ *from* H_j*,* $j = 2, \ldots, m$*, using the four point algorithm.*

4. *If* $m \geq 3$*, find the unique solution that satisfies* $\pi_j^2 = \pi_2^2, j = 3, \ldots, m$*.*

5. *Let* $\pi = [\pi_1, 1]$*,* $T_j = \tilde{T}_j \pi_2^2 / \pi_j^2, j = 3, \ldots, m$*.*

6. *Solve for* α *from (11) using linear least squares:*

$$\alpha_{k+1}^i = -\frac{\sum_{j=2}^{m} (\widehat{\mathbf{x}}_j^i T_j)^T \widehat{\mathbf{x}}_j^i R_j \mathbf{x}_1^i + \pi^2 \pi^1 \mathbf{x}_1^i}{\sum_{j=2}^{m} ||\widehat{\mathbf{x}}_j^i T_j||^2 + (\pi^2)^2}. \quad (16)$$

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

2.3 Multi-view Motion For Landing

In this section we describe how we apply Algorithm 1 to the landing problem. As described briefly in the following section (and in detail in [12]), we have designed a landing target to simplify the image processing and feature extraction tasks. We set the first frame in Algorithm 1 to be the coordinate frame that is one focal length above the landing target along the optical axis. Further, since we know the geometry of the landing target, we use the first frame's camera position, the camera calibration matrix, and the perspective projection model to set the feature points for the first frame. Next for each captured image, we set $\mathbf{x}_m^i$, $i = 1, \ldots, n$, to be the feature points extracted from the current view, and $\mathbf{x}_{m-j}^i$, $j = 1, \ldots, m-2$, to be the feature points extracted from the previous j frames. This setup has two very nice properties: (1) $(R_m, T_m) \in SE(3)$ is the Euclidean motion from the landing target to the current camera frame, (2) knowledge of the landing target geometry allows to *uniquely* recovery $T_m \in \mathbb{R}^3$, instead of up to an arbitrary scale.

3 Vision System Test-bed

This section briefly describes our real-time vision system [12]. Further, we describe our UAV test-bed [14] and the use of a computer vision based supervisory controller 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 $\mathbf{x}_m^i$, $i = 1, \ldots, n$, of points on a landing target. We use corner features on a specially designed black and white landing target to simplify the feature extraction stage [12]. Figure 2 shows our designed landing target and feature extraction in action.

Given the feature points $\mathbf{x}_m^i$, 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 [12], and the multi-view linear algorithm described in Section 2.

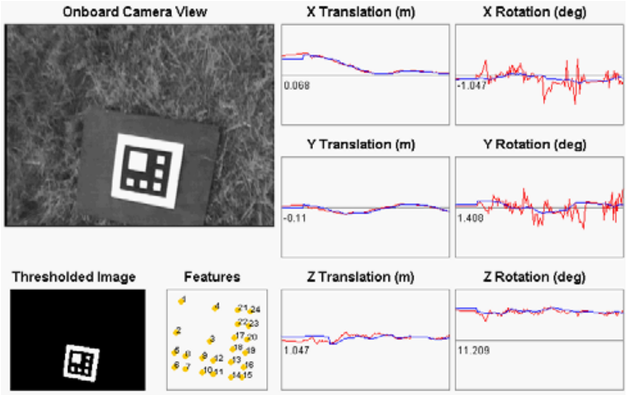

Figure 2: Vision monitoring station

3.2 Vision System Hardware

Our real-time vision system consists of the following off-the-shelf hardware components [12]:

- *Vision computer*: PC104 Pentium 233MHz-based PC running Linux. Responsible for grabbing images, vision algorithms and camera control.
- *Camera*: Sony EVI-D30 Pan/Tilt/Zoom camera. Actively pan/tilt to keep the landing target centered int he field of view.

- *Framegrabber*: Imagenation PXC200. Captures 320 × 240 resolution images at 30Hz.
- *Wireless Ethernet*: ORiNOCO Gold card 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, 14]:

- *Navigation Computer*: PC104 Pentium 233MHz-based PC running QNX real-time OS. Responsible for sensor management and hard real-time flight control.
- *Inertial Measurement Unit*: NovAtel MillenRT2 GPS system (2cm accuracy) and Boeing DQI-NP INS/GPS integration system.

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, 14], 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.

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 through the Vehicle Control Language over a serial link. 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. In flight experiments, this supervisory control worked very well to keep the UAV above the landing target even in very high winds.

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.

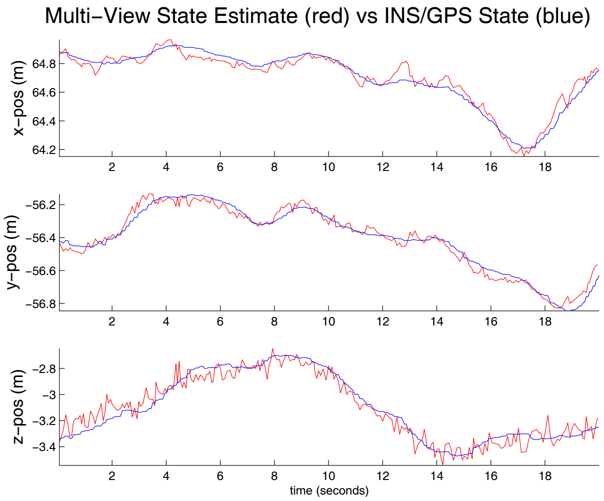

Figure 3: Comparing multi-view planar motion estimation algorithm with inertial navigation system estimates in a real flight test.

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 $m = 4$ in the multiple view algorithm for the experiments.

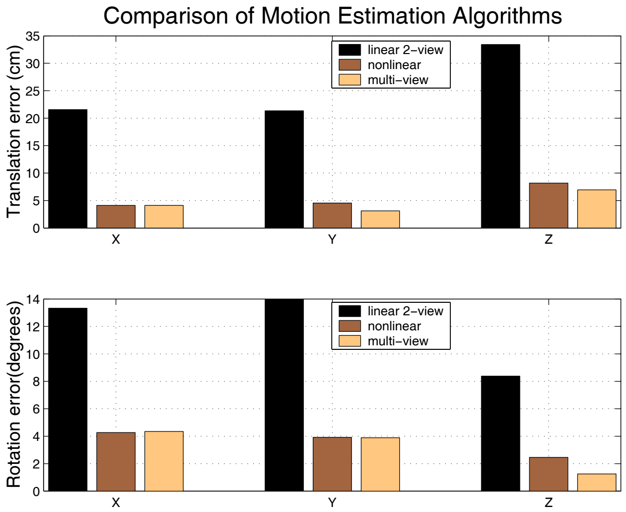

Figure 4: Comparison of mean squared errors of motion estimation algorithms during a real flight test. The ‘linear 2-view’ and ‘nonlinear’ algorithms are described in [12]. The ‘multi-view’ algorithm is described in Section 2.

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" algorithms which were presented in [12], and the "multi-view" algorithm described in Section 2. The multi-view algorithm slightly outperforms the nonlinear algorithm. Further, the multi-view algorithm is globally robust, while the nonlinear algorithm has many local minima and is sensitive to initialization. The performance and robustness of the multi-view algorithm make it the clear winner among the algorithms.

5 Conclusion and Future Work

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. In future work, we will use this motion-estimation algorithm and design vision-based controller to land a UAV onto the moving landing deck shown in Figure 1.

Acknowledgment

The authors would like to thank Hoam Chung and Ron Tal for their support in the performing the flight test 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] O.D. Faugeras and F. Lustman. Motion and structure from motion in a piecewise planar environment. *International Journal of Pattern Recognition and Artificial Intelligence*, 2(3):485–508, 1988.

[3] K. Kanatani. Detecting the motion of a planar surface by line & surface integrals. In *Computer Vision, Graphics, and Image Processing*, volume 29, pages 13–22, 1985.

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

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

[6] Y. Ma, K. Huang, R. Vidal, J. Košecká, and S. Sastry. Rank conditions of multiple view matrix in multiple view geometry. Technical Report UILU-ENG 01-2214 (DC-220), UIUC, June 2001.

[7] 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.

[8] A. Petruszka and A. Stentz. Stereo vision automatic landing of VTOL UAVS. In *Proceedings of Assoc. Unmanned Vehicle Syst. Int.*, pages 245–63, 1996.

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

[10] O. Shakernia, Y. Ma, T.J. Koo, J. Hespanha, and S. Sastry. Vision guided landing of an unmanned air vehicle. In *Proceedings of CDC*, Phoenix, Arizona, December 1999.

[11] 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.

[12] C.S. Sharp, O. Shakernia, and S. Sastry. A vision system for landing an unmanned aerial vehicle. In *IEEE International Conference on Robotics and Automation*, pages 1720–1727, Seoul, Korea, May 2001.

[13] D.H. Shim. *Hierarchical Flight Control System Synthesis for Rotorcraft-Based Unmanned Aerial Vehicles*. PhD thesis, UC Berkeley, 2000.

[14] 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.

[15] M. Subbarao and A.M. Waxman. On the uniqueness of image flow solutions for planar surfaces in motion. *Computer Vision, Graphics, and Image Processing*, 36:208–228, 1986.

[16] A. Waxman and S. Ullman. Surface structure and three-dimensional motion from image flow kinematics. *Int. J. Robotics Research*, 4(3):72–94, 1985.

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

[18] S. Werner, S. Furst, D. Dickmanns, and E.D. Dickmanns. A vision-based multi-sensor machine perception system for autonomous aircraft landing approach. In *Proceedings of the SPIE - The International Society for Optical Engineering*, volume 2736, pages 54–63, Orlando, FL, USA, 1996.

[19] 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.