# Vision-based Landing of an Unmanned Air Vehicle

O. Shakernia, R. Vidal, S. Sastry
Department of EECS, UC Berkeley

# Applications of Vision-based Control

Predator

Global Hawk

UCAV X-45


SR/71

Fire Scout

# Goal: Autonomous landing on a ship deck

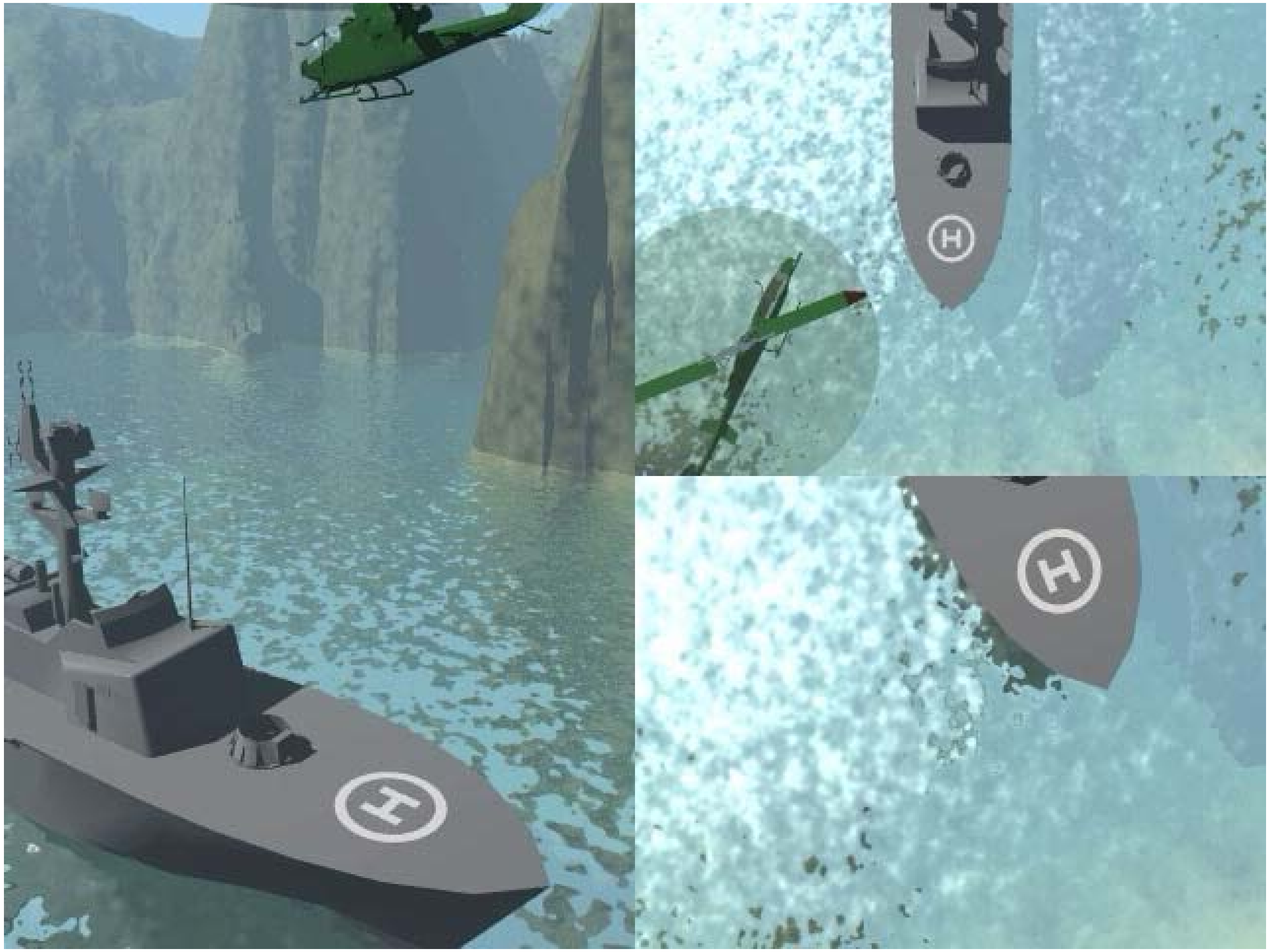
## Challenges

- Hostile environments
  - Ground effect
  - Pitching deck
  - High winds, etc

## Why vision?

- Passive sensor
- Observes relative motion

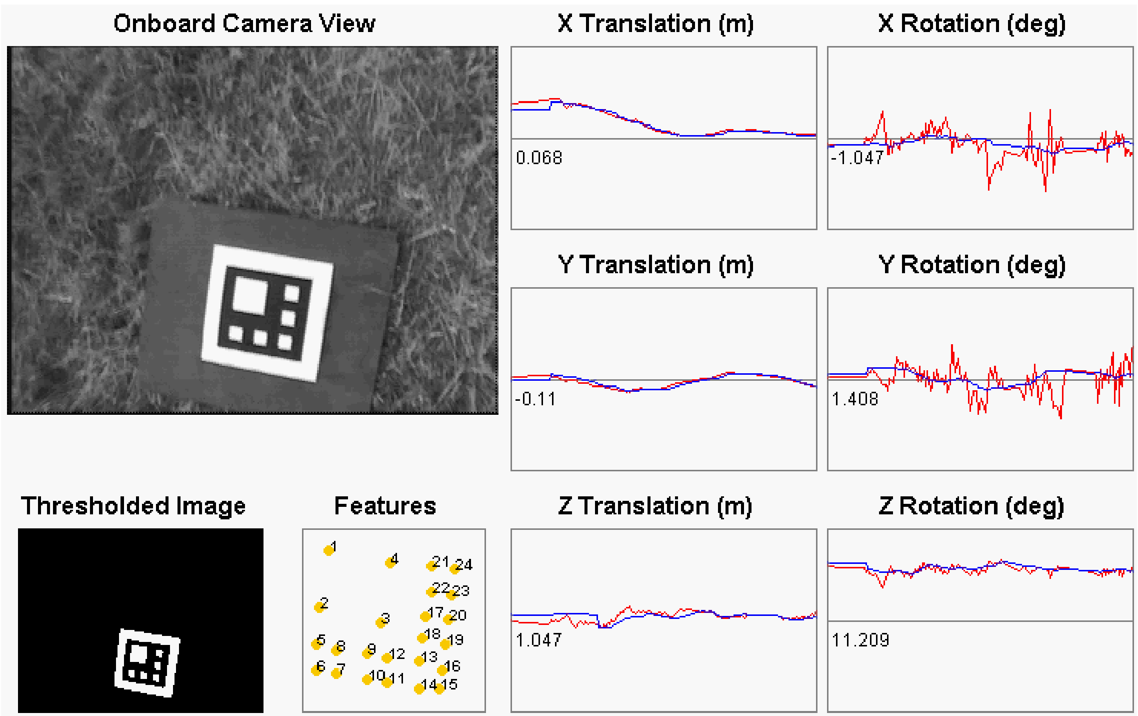
# Simulation: Vision in the loop

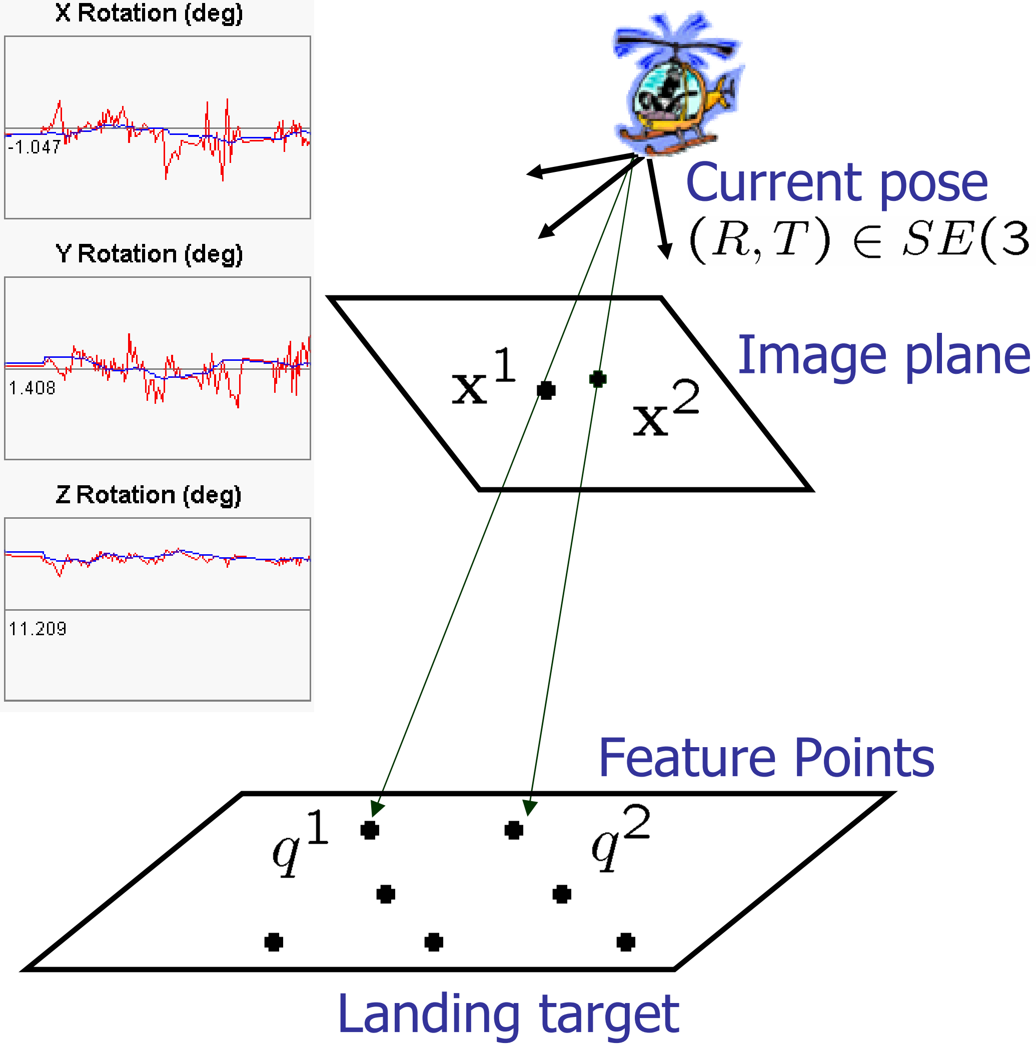


# Vision-Based Landing of a UAV

- Motion estimation algorithms
  - Linear, nonlinear, multiple-view
  - Error: 5cm translation, 4° rotation
- Real-time vision system
  - Customized software
  - Off-the-shelf hardware
- Vision in Control Loop
- Landing on stationary deck
- Tracking of pitching deck

# Vision-based Motion Estimation




**Pinhole Camera**

$$\lambda^j \mathbf{x}^j = [R\ T] q^j$$

# Pose Estimation: Linear Optimization

- Pinhole Camera: $\lambda_i \mathbf{x}_i = [R \;\; T] q_i$
- Epipolar Constraint: $0 = \widehat{\mathbf{x}_i}[R \;\; T] q_i$
- Planar constraint: $0 = e_3^T q_i \qquad \forall i$

$$0 = \widehat{\mathbf{x}_i}[r_1 \;\; r_2 \;\; T] \begin{bmatrix} q_{i1} \\ q_{i2} \\ 1 \end{bmatrix} \quad \Rightarrow \quad G \begin{bmatrix} r_1 \\ r_2 \\ T \end{bmatrix} = 0$$

- More than 4 feature points $\Rightarrow \text{rank}(G) = 8$
- Solve linearly for $[r_1^T \;\; r_2^T \;\; T^T]^T \in \mathbb{R}^9$
- Project $[r_1 \;\; r_2 \;\; 0] \in \mathbb{R}^{3\times 3}$ onto $SO(3)$ to recover $R$

# Pose Estimation: Nonlinear Refinement

- Objective: minimize error

$$G_i = \widehat{\mathbf{x}_i}[R\ T]q_i$$

$$G = [G_1^T\ \ \dots\ \ G_n^T]^T$$

- Parameterize rotation by Euler angles

$$R = R_{\mathbf{z}}(\psi)R_{\mathbf{y}}(\theta)R_{\mathbf{x}}(\phi)$$

$$\beta = [\psi\ \theta\ \phi\ T_x\ T_y\ T_z]^T \in \mathbb{R}^6$$

- Minimize by Newton-Raphson iteration

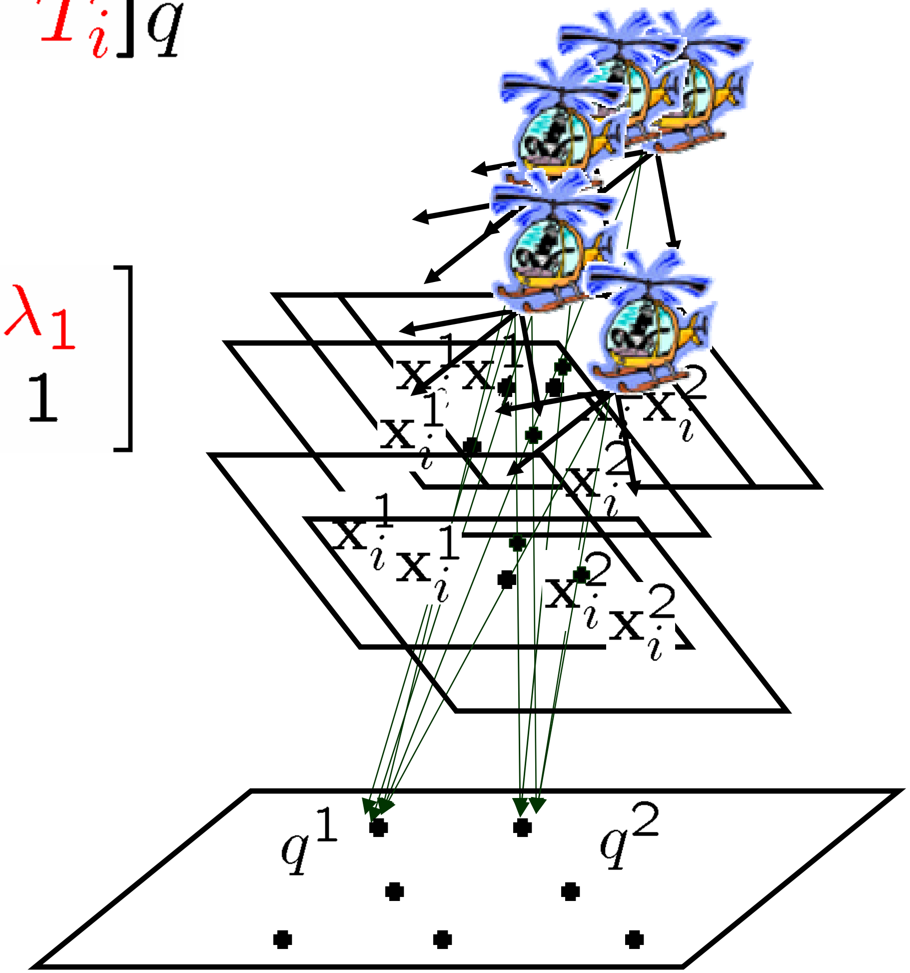
$$\beta_{n+1} = \beta_n - k_n(D_\beta G|_{\beta_n})^\dagger G(q, \mathbf{x}, \beta_n)$$

- Initialize with linear algorithm

# Multiple-View Motion Estimation

Pinhole Camera $\lambda_i \mathbf{x}_i = [R_i \;\; T_i] q$

- $R_1 \doteq I_{3\times3}, \quad T_1 \doteq 0$
- $\lambda_i \mathbf{x}_i = \lambda_1 R_i \mathbf{x}_1 + T_i$
- $0 = [\widehat{\mathbf{x}}_i R_i \mathbf{x}_1 \;\; \widehat{\mathbf{x}}_i T_i] \begin{bmatrix} \lambda_1 \\ 1 \end{bmatrix}$
- $0 = \lambda_1 \pi^1 \mathbf{x}_1 + \pi^2$




Multiple View Matrix

$$M \doteq \begin{bmatrix} \widehat{\mathbf{x}}_2 R_2 \mathbf{x}_1 & \widehat{\mathbf{x}}_2 T_2 \\ \vdots & \vdots \\ \widehat{\mathbf{x}}_m R_m \mathbf{x}_1 & \widehat{\mathbf{x}}_m T_m \\ \pi^1 \mathbf{x}_1 & \pi^2 \end{bmatrix}$$

Rank deficiency constraint

$$M\,[\lambda_1, 1]^T = 0 \quad \Rightarrow \quad \boxed{\mathrm{rank}(M) \leq 1}$$

# Multiple-View Motion Estimation

- *n* points in *m* views

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

- Equivalent to finding $\pi \in \mathbb{R}^4,\ \vec{R}_i \in \mathbb{R}^9$ and $\vec{T}_i \in \mathbb{R}^3$, s.t.

$$\begin{bmatrix} {\mathbf{x}_1^1}^T & \alpha^1 \\ \vdots & \vdots \\ {\mathbf{x}_1^n}^T & \alpha^n \end{bmatrix} \pi^T = 0, \qquad \begin{bmatrix} \alpha^1 \widehat{\mathbf{x}}_i^1 & \widehat{\mathbf{x}}_i^1 * {\mathbf{x}_1^1}^T \\ \vdots & \vdots \\ \alpha^n \widehat{\mathbf{x}}_i^n & \widehat{\mathbf{x}}_i^n * {\mathbf{x}_1^n}^T \end{bmatrix} \begin{bmatrix} \vec{T}_i \\ \vec{R}_i \end{bmatrix} = 0$$

- Initialize $R_2, T_2, \pi$ with two-view linear solution

- Least squared solution: $\alpha^j = -\dfrac{(\widehat{\mathbf{x}}_2^j T_2)^T \widehat{\mathbf{x}}_2^j R_2 \mathbf{x}_1^j + \pi^2 \pi^1 \mathbf{x}_1^j}{||\widehat{\mathbf{x}}_2^j T_2||^2 + (\pi^2)^2}.$

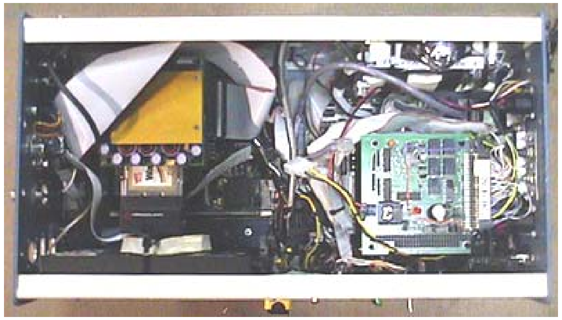
- Use $\alpha^j$ to linearly solve for $\pi, \vec{R}_i, \vec{T}_i$
- Iterate until $\alpha^j$ converge

# Real-time Vision System

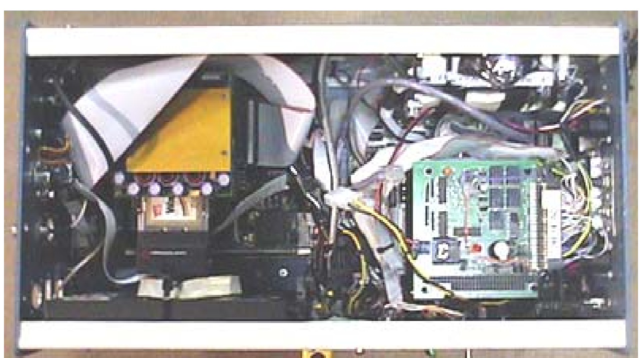
- Ampro embedded Little Board PC
  - Pentium 233MHz running LINUX
  - 440 MB flashdisk HD robust to vibration
  - Runs motion estimation algorithm
  - Controls Pan/Tilt/Zoom camera
- Motion estimation algorithms
  - Written and optimized in C++ using LAPACK
  - Estimate relative position and orientation at 30 Hz

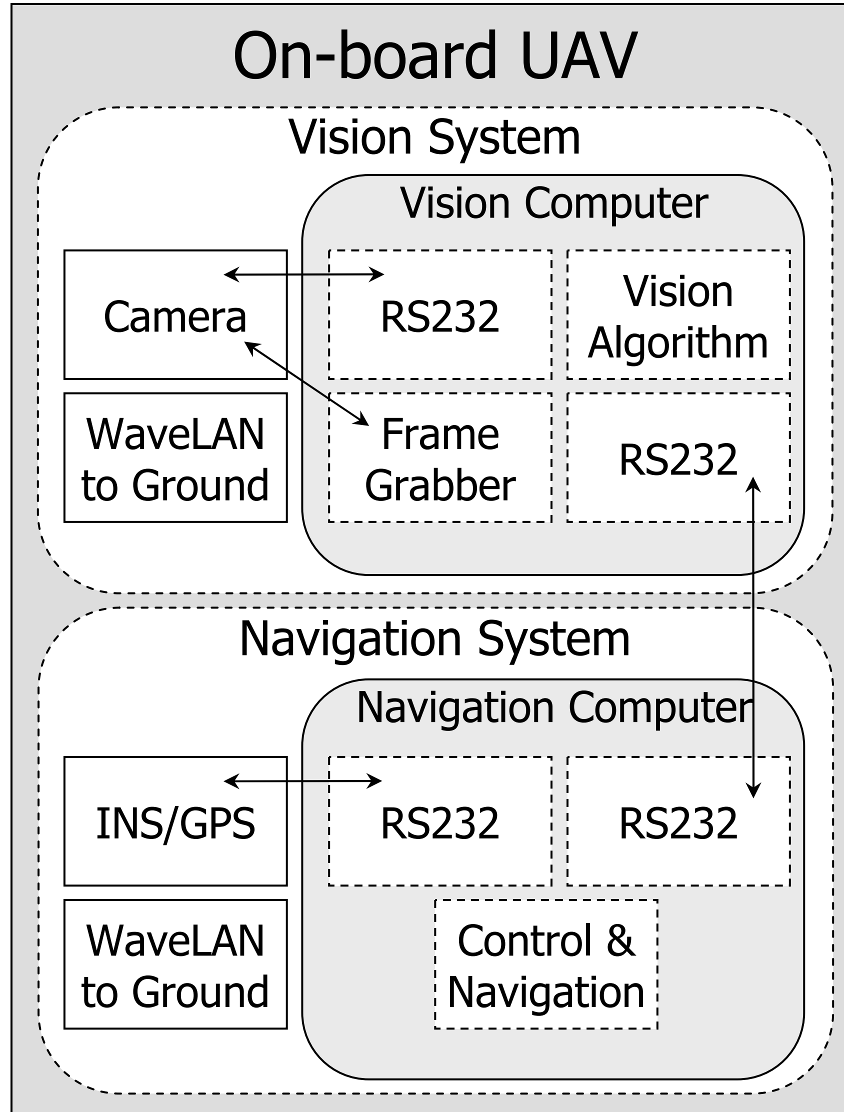

UAV


Pan/Tilt Camera


Onboard Computer

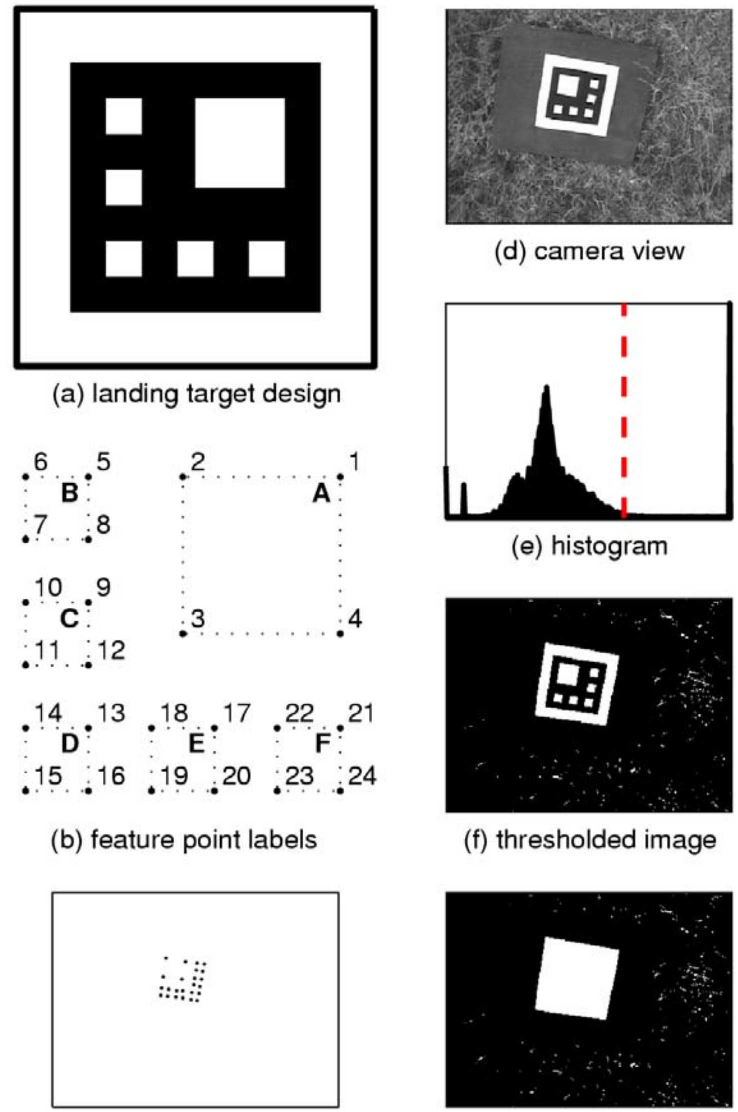
# Hardware Configuration





# Feature Extraction

- Acquire Image
- Threshold Histogram
- Segmentation
- Target Detection
- Corner Detection
- Correspondence




(a) landing target design

(b) feature point labels

(c) detected corners

(d) camera view

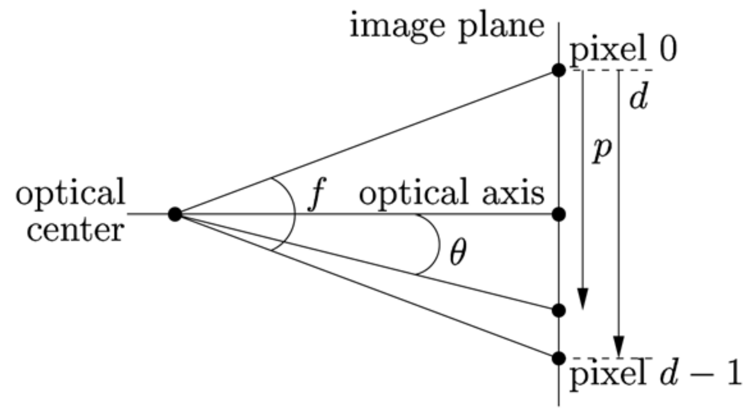
(e) histogram

(f) thresholded image

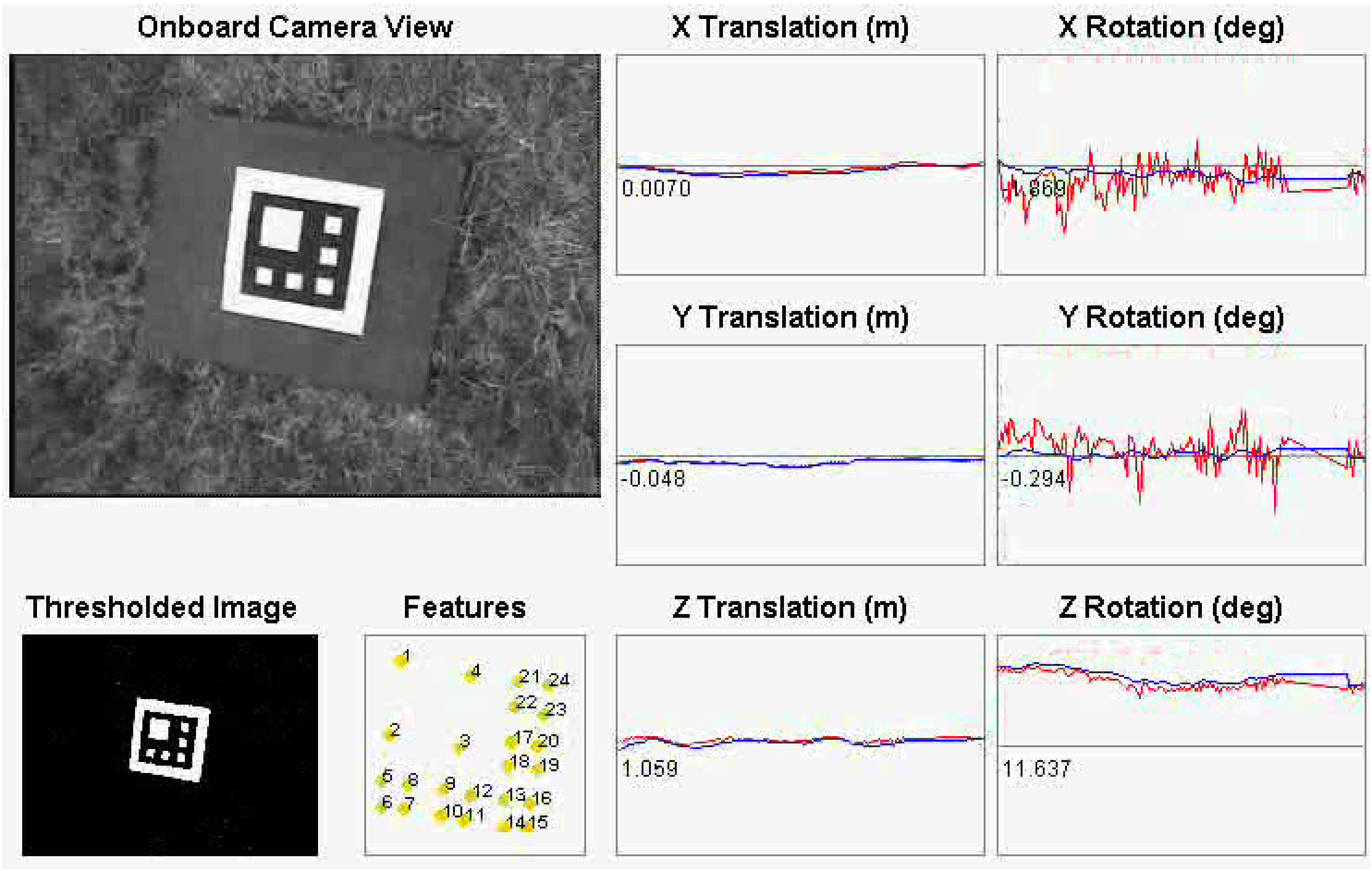
(g) foreground regions

# Camera Control

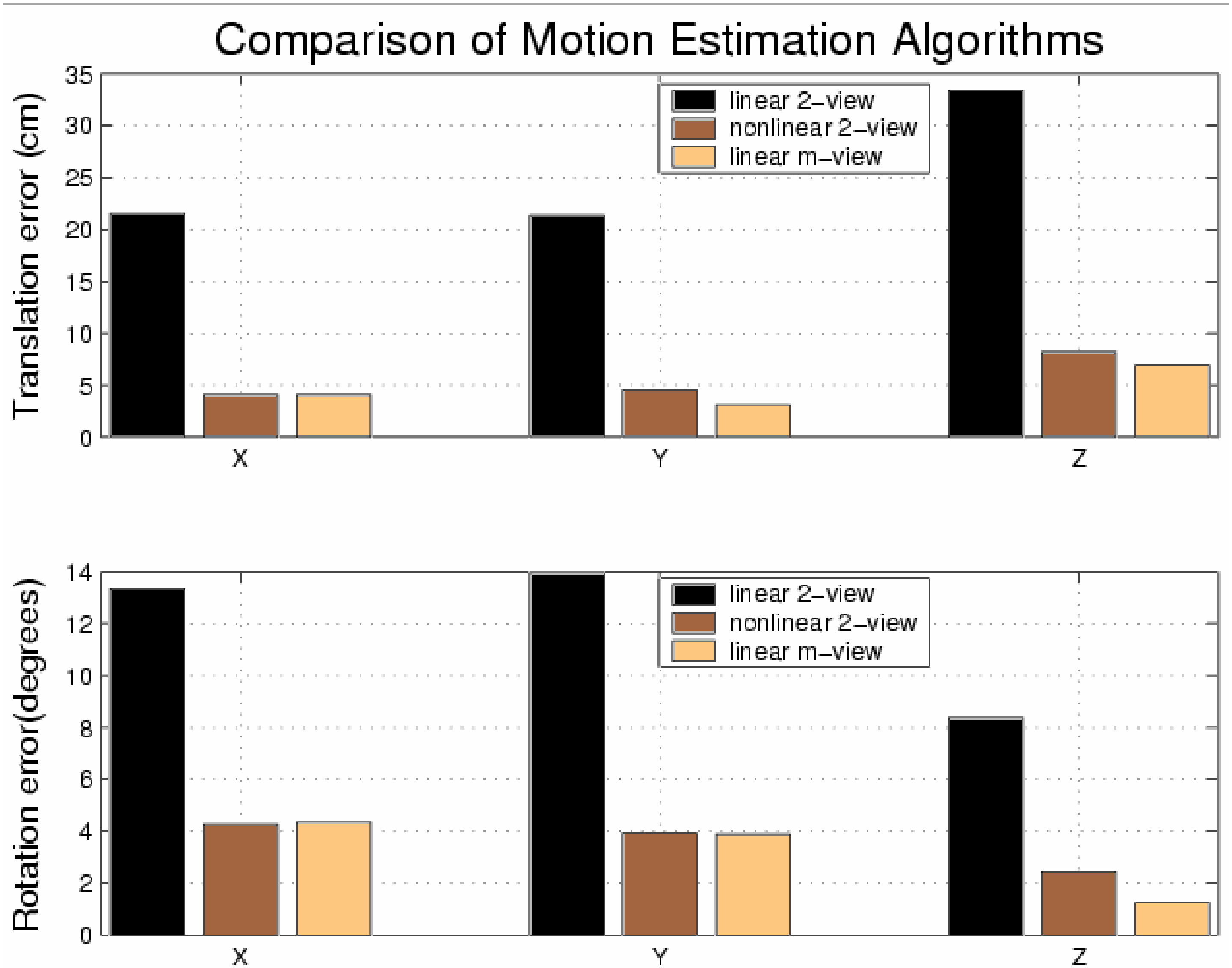
- Pan/Tilt to keep features in image center
  - Prevent features from leaving field of view
  - Increased Field of View
  - Increased range of motion of UAV



# Comparing Vision with INS/GPS

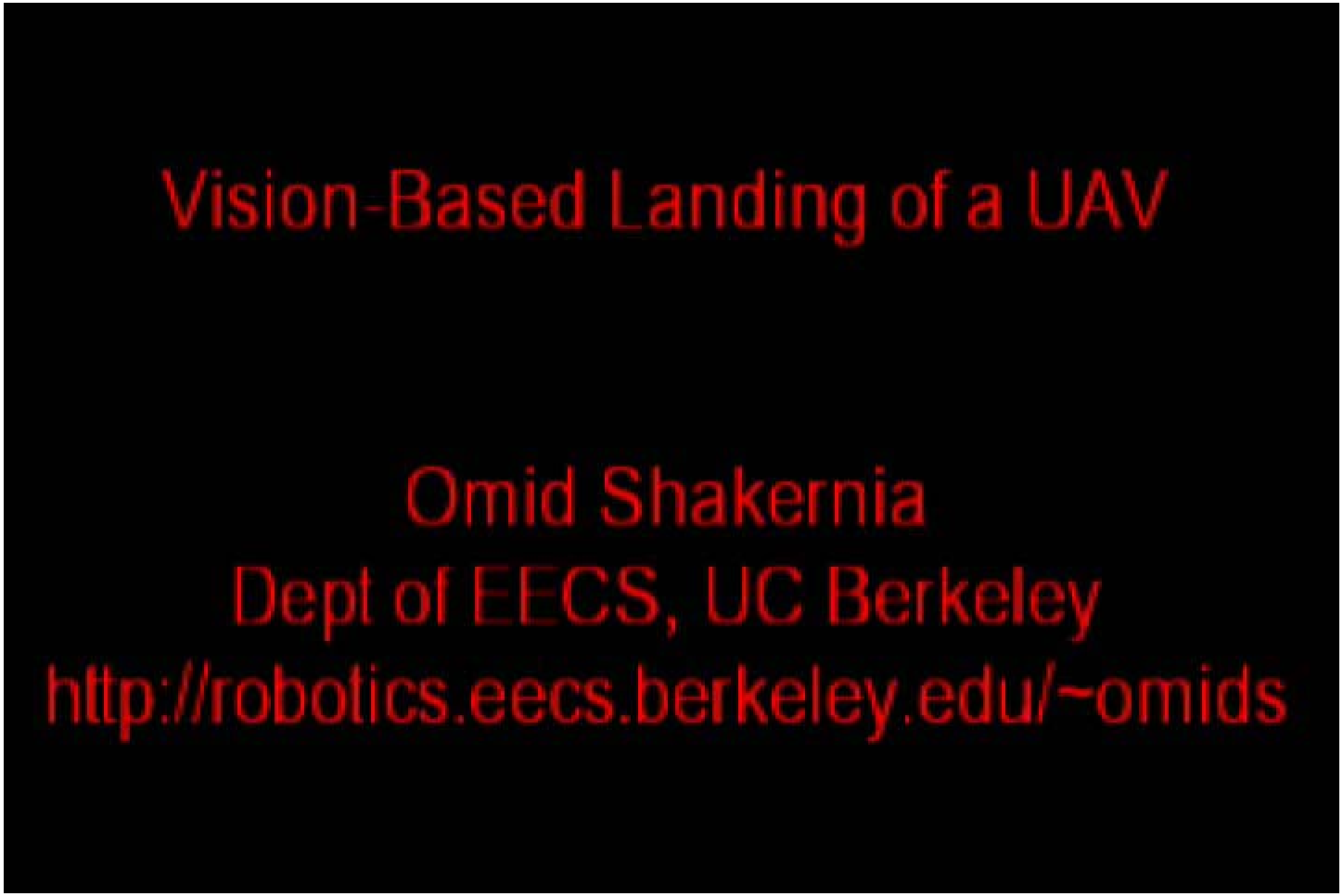


# Motion Estimation in Real Flight Tests



# Landing on Stationary Target

# Tracking Pitching Target



# Conclusions

- Contributions
  - Vision-based motion estimation (5cm accuracy)
  - Real-time vision system in control loop
  - Demonstrated proof of concept prototype: first vision-based UAV landing
- Extensions
  - Dynamic vision: Filtering motion estimates
  - Symmetry-based motion estimation
  - Fixed-wing UAVs: Vision-based landing on runways
  - Modeling and prediction of ship deck motion
  - Landing gear that grabs ship deck
  - Unstructured environments: Recognizing good landing spots (grassy field, roof top etc)