43rd IEEE Conference on Decision and Control
December 14-17, 2004
Atlantis, Paradise Island, Bahamas

Real-Time and 3D Vision for Autonomous Small and Micro Air Vehicles

Takeo Kanade, Omead Amidi, Qifa Ke
Robotics Institute, Carnegie Mellon University
Pittsburgh PA, USA

Abstract— *Autonomous control of small and micro air vehicles (SMAV) requires precise estimation of both vehicle state and its surrounding environment. Small cameras, which are available today at very low cost, are attractive sensors for SMAV. 3D vision by video and laser scanning has distinct advantages in that they provide positional information relative to objects and environments, in which the vehicle operates, that is critical to obstacle avoidance and mapping of the environment. This paper presents work on real-time 3D vision algorithms for recovering motion and structure from a video sequence, 3D terrain mapping from a laser range finder onboard a small autonomous helicopter, and sensor fusion of visual and GPS/INS sensors.*

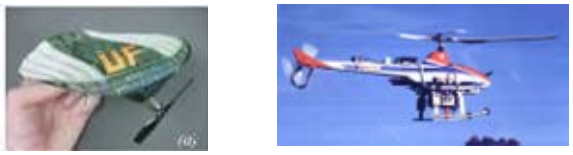
I. Introduction

One of the critical capabilities for making a small- and micro-air vehicle (SMAV) autonomous and useful is the precise estimation of the SMAV states (pose and position) and 3D mapping of its surrounding environment. Among many sensors for these purposes, 3D vision by video and laser scanning has distinct advantages. Unlike navigational sensors (such as GPS and gyro) that provide information of only the vehicle's own motion with respect to the inertial frame, vision can provide information relative to the environment – how close the vehicle is to an obstacle or whether there are moving objects in the environment. Unlike GPS, which does not work in the shadow of satellite visibility, vision works in a cluttered urban environment or even indoors. At the same time, camera images are view dependent, tend to be noisy, and require a substantial amount of processing in order to extract useful information from them.

This paper presents a set of robust real-time vision algorithms suitable for the purpose of structure from motion vision, with video from a small low-cost AUV on-board camera. Also presented is the environmental mapping system that integrates a large number of 3D slice data obtained by a small helicopter which autonomously flies over and scans a terrain or an urban area with an on-board laser scanner.

II. AUV Vision Architecture

On-board real-time active vision, when combined with the other inertial sensors and GPS, and glued by adaptive model-based robust control, makes the UAV self-sufficient and capable of agile maneuvering in a cluttered complex 3D environment. At the moment we work with two types of air vehicles: a micro fixed-wing University of Florida air vehicle, shown in Figure 1 (a), that has a single video camera, and a small Yamaha R50-based autonomous helicopter [8], shown in Figure 1(b), that has a camera, GPS, gyros, and a laser scanner.



(a) (b)

Figure 1. (a) University of Florida fixed wing micro air vehicle, (b) YamahaR50-based autonomous helicopter

Figure 2 shows the overall architecture of a real-time 3D vision system that we have been developing for these vehicles.

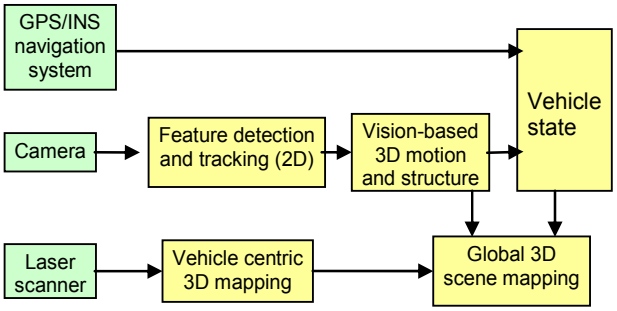


Figure 2 : Overview of vision system. Vision-based motion estimation and navigation system provides the vehicle state (pose and position). Laser scanner collects the range data, which are integrated into a global 3D map using the precise estimate of vehicle state. The laser 3D map is also fused with the image texture as well as the 3D estimation from vision based estimation.

The sensory input to the system includes 1) video stream(s) captured from the on-board single or multiple cameras, 2) positional and inertial motion sensor data from GPS and gyros, and 3) three-dimensional slices of the environment

below, obtained by an on-board laser scanner.

We design the real-time vision module to be capable to work for itself, if necessary, considering the cases where the vehicle is too small to carry other sensors (true for the moment with University of Florida vehicle), or in situations when other sensor is not available temporarily (such as GPS in a shadow).

The real-time 3D vision module consists of three sub modules: feature selection, feature tracking, and structure from motion. With these, both 3D structure of the scene (position and shape of obstacles) and vehicle's own motion (position and pose) are recovered from the input video. It must work robustly to cope with low-quality video, and in real time with minimum latency to be usable for control. The laser scanner's output is used at this moment only for mapping purposes, not for navigation. As the vehicle flies, the sensor scans the terrain below, and obtains a sequence of three-dimensional slices or profiles. The data are converted into a common coordinate frame to create a 3D map of the terrain.

III. VISION-BASED 3D MOTION ESTIMATION FOR UAV

Real-time 3D vision can estimate from the video of an on-board camera the vehicle state (position and pose), as well as 3D of surrounding environment. This task has been studied extensively in the field of computer vision as the structure from motion (SFM) problem. For its use for small and micro air vehicle control, however, there are two critical differences that make the SFM solution far more difficult than typical off-line SFM applications, such as scene modeling from video and motion recovery for image-based rendering. Firstly, the input videos are of lower quality. On-board cameras tend to have lower resolution and to be noisy. Secondly, unlike the computer graphics applications, videos are not taken by design, and therefore they include large motion or motion blur due to fast motion, or degenerate motions that may make some solution methods singular. Thirdly, the process must give the best solution using the images up to that point in time; unlike off-line applications, "future" images cannot be used.

The key to successful use of SFM for small and micro air vehicle control is to make the SFM processes robust to these difficulties. The SFM process includes feature detection, feature tracking, and reconstruction.

Feature Detection and Tracking

Feature tracker finds where the features, defined in the previous frames, have moved to in the current frame. In order to define "good" features to track, a tracking method has to be defined first. Given a feature point (x,y) in the current image I, we want to estimate its position in the next image I'. According to [4], we use a small window centered at (x,y) in the first image I. Assuming that the corresponding region has the same appearance, the tracking algorithm finds the displacement $\boldsymbol{d} = (d_x, d_y)$ by minimizing the following $\boldsymbol{L}_2$ norm:

$$\begin{aligned}(d_x, d_y) &= \arg\min J(d_x, d_y) \\ &= \arg\min \sum_{(x,y)\in W} \left[I(x,y,t) - I(x+d_x, y+d_y, t+\Delta t)\right]^2 \end{aligned} \quad (1)$$

Assuming small displacement $\boldsymbol{d}$, the linear closed-form solution [4] (so-called Lucas-Kanade feature tracking algorithm) is:

$$\boldsymbol{d} = \boldsymbol{H}^{-1}\boldsymbol{b} \quad (2)$$

where

$$\boldsymbol{H} = \begin{pmatrix} \sum_{(x,y)\in W} I_x I_x & \sum_{(x,y)\in W} I_x I_y \\ \sum_{(x,y)\in W} I_y I_x & \sum_{(x,y)\in W} I_y I_y \end{pmatrix} \quad (3)$$

$$\boldsymbol{b} = \begin{pmatrix} -\sum_{(x,y)\in W} I_x I_t \\ -\sum_{(x,y)\in W} I_y I_t \end{pmatrix}$$

and I_x, I_y, and I_t are spatial-x, spatial-y and time derivatives of image. While simple, this technique is known to be efficient and works well in most situations.

Feature Point Selection

Equation 2, which needs to be solved for tracking the feature motion $\boldsymbol{d}$, suggests the property that a "good" feature must possess. For equation 2 to be stable in the presence of noises, we must select a feature point (an image template W) whose corresponding 2x2 matrix $\boldsymbol{H}$ is stably invertible. In other words, the two singular values λ_1 and λ_2 of $\boldsymbol{H}$ should be large and sufficiently close to each other [5, 6]. So, we define the "goodness" of a window to be

$$\lambda = min(\lambda_1, \lambda_2) \quad (4)$$

Our feature selection process is quite simple: for each pixel of a current frame, the Hessian matrix $\boldsymbol{H}$ is computed using the 7x7 window. Features are selected at the local maximums of λ, such that they are separated by at least 7 pixels from each other.

Dealing with illumination change

The L_2 norm that Lucas-Kanade tracker uses assumes that corresponding pixels in different images have the same appearance. In real life, such an assumption may not hold due to imaging noises, lighting change and view change.

In our implementation, we first smooth the images with Gaussian filtering to reduce noise level. We also model lighting change using a scale-and-offset model for each corresponding point:

$$aI(\xi, \zeta, t) + b \quad (5)$$

The cost function thus becomes:

$$J(d) = \sum_{(x,y)\in W} \left[aI(\xi,\zeta,t) + b - I(x,y,t+\Delta t)\right]^2 \quad (6)$$

Here a and b are assumed constant inside each window, but is different for different feature points. We can still simultaneously solve the four unknowns (d_x, d_y, a, b) using least squared estimation similar to equation 2. We notice that the introduction of a and b makes the linear system in Eq.(2) more unreliable. For such reason we scale the affine motion parameters m and lighting-related parameters a and b respectively to make the system in Eq.(2) reliable.

Dealing with Large Motion

In equation 2, it is assumed that the pixel movement ***d*** is small. We use the following techniques to deal with large motion in real life:

- Image pyramid. We construct a three-level Gaussian image pyramid. Each level of the pyramid effectively doubles the range of pixel movement that our system can handle.
- 2D affine motion model. When the motion is large, the translational model (d_x,d_y) may not be sufficient to model the pixel movements inside the small window. We use 2D affine motion model where the 2D motion is described by six parameters to account for rotation, scaling, shearing, and translation.
- Progressive model refinement. We use a simple translational motion model at the highest (coarsest) level, and affine motion model at the lowest (finest) level. This helps stabilize the estimate.
- Motion prediction. We use simple Kalman filtering to predict the position of each feature point at current image.

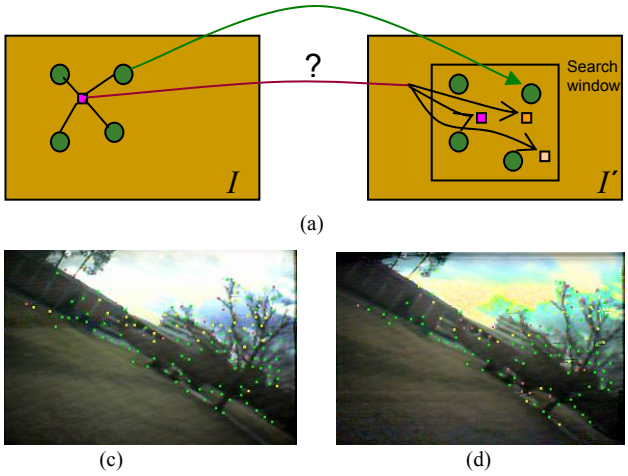
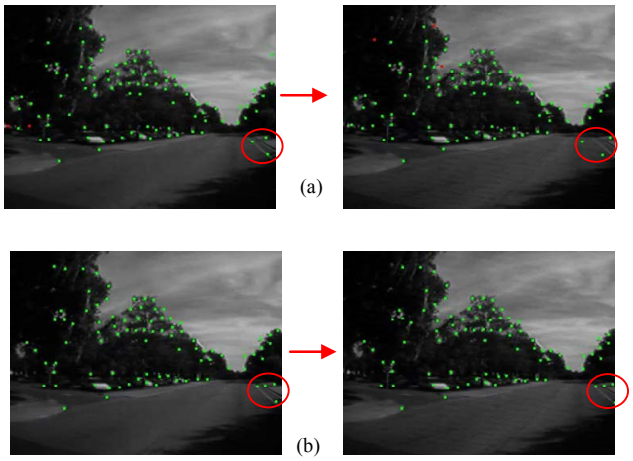


Figure 3.1: Combined tracker. (a): The green points are reliable trackers by template registration, and are used as landmarks. The pink square is tracked by establishing the graph relationship with the landmarks. (c) and(d): two snapshots of the tracking using combined tracker. The red points are lost trackers in LK algorithm, but salvaged by the combined tracker.

Combined Tracker

The Lucas-Kanade tracker uses template registration. Another type of tracker uses point matching. In point matching, feature points are detected in both images. For each point in the first image, its best corresponding point in second image is found by exhausted search within some predefined search window. The point matching method is more robust to noise but it often gives ambiguous correspondences (one-to-many correspondences). We use a new scheme that combines template registration and point matching. In our scheme, feature points that are reliably tracked by Lucas-Kanade algorithm are marked as landmarks. Other feather points that can not be reliably tracked by Lucas-Kanade are referenced by the landmarks in its neighborhood. Their correspondences in the second image are searched by simple graph matching approach. See Figure 3.1 for an illustration of the combined tracker.



Feature 3: Tracking results (input image size 360x240). The green dots are selected feature points. (a): Tracking using the original Lucas-Kanade algorithm; (b): Tracking using our extension to Lucas-Kanade. The red circle shows some example feature points that are tracked well in our system, but are not handled well using the original Lucas-Kanade tracker.

Feature 3.3: Tracking results after five frames (input image size 360x240): (a) selected feature points in first frame; (b): tracked points in the 5th frame using the original Lucas-kanade algorithm; (c): tracked points in the 5th frame using our extension to Lucas-Kanade. Our extension tracks much more feature points successfully.

Tracking results

The feature tracker that has implemented all of the previous extension was test applied to a video sequence taken from a real flying micro-AUV and from the same camera attached to a car navigating on the ground. It performs well despite the large image distortion, frequent lighting change, and large image motion.

Figure 3.2 shows two snapshots of tracking results, comparing classical Lucas-Kanade tracker with the extended one; the latter has eliminated most of erroneous tracking. Figure 3.3 shows tracked points across five frames. As can be seen our extension to the original Lucas-Kanade algorithm tracks more features successfully.

The feature tracker monitors the quality of tracking results of each feature by means of the value of J and the property of $\boldsymbol{H}$. It discards a feature once it is found to be no longer easy to track well. New feature points are generated from the current frame to keep the total number of active features above a certain number.

Two-Frame Motion Estimation

From the 2D correspondences of the tracked features, the SFM algorithm estimates the relative motion of the vehicle. For UAV real-time control application, the SFM solution must provide the camera (i.e., vehicle) motion from the most recent image frames.

Let us denote a 3D point in the scene corresponding to the i-th feature as $\boldsymbol{M}_i = (X_i, Y_i, Z_i)^T$. Here we use the first camera's coordinate frame as the world coordinate frame. The point $\boldsymbol{M}_i$ is projected to the first image as an image point $\boldsymbol{m}_i = (x_i, y_i, 1)^T$. Also, let us assume that the camera moves by rotation $\boldsymbol{R}$ and translation $\boldsymbol{r}$. In the second camera's coordinates, the same point appears as $\boldsymbol{M}_i' = (X_i', Y_i', Z_i')^T$ and it is projected to the second image as $\boldsymbol{m}'_i = (x'_i, y'_i, 1)^T$.

For simplicity, let us assume that the camera is calibrated, (i.e., the camera intrinsic parameters are known). Then these entities are related to each other by

$$\begin{aligned} \boldsymbol{m}_i &= \frac{1}{Z_i}\boldsymbol{M}_i \\ \boldsymbol{m}'_i &= \frac{1}{Z_i}\boldsymbol{M}'_i \\ \boldsymbol{M}'_i &= \boldsymbol{R}\boldsymbol{M}_i + \boldsymbol{r} \end{aligned} \tag{7}$$

The two-frame SFM estimates the relative camera pose (R,t) between two camera positions, and the 3D point positions in the coordinate frame of the first camera.

The basic solution of the problem is as follows. From equation 7 we have:

$$\boldsymbol{m}'_i(\boldsymbol{r} \times \boldsymbol{R})\boldsymbol{m}_i = 0 \tag{8}$$

where $\times$ is the 3-vector cross product. The 3×3 matrix $\boldsymbol{E} = \boldsymbol{r} \times \boldsymbol{R}$ is the so-called Essential matrix in computer vision. Given 8 or more pairs of feature point correspondences, we can compute $\boldsymbol{f}$ (i.e., $\boldsymbol{E}$) using the linear Eight Points algorithm [3]. Equation 8 can be rewritten as:

$$\boldsymbol{A}_{8\times 9}\boldsymbol{f} = 0 \quad s.t. \quad \|\boldsymbol{f}\| = 1 \tag{9}$$

Here $\boldsymbol{f}$ contains the 9 parameters of matrix $\boldsymbol{E}$. The solution $\boldsymbol{f}$ is the null space of $\boldsymbol{A}$, i.e., the eigenvector of $\boldsymbol{A}^T\boldsymbol{A}$ corresponding to its smallest eigenvalue. In principle, if more than 8 correspondences are available, then we have a least-squared solution, and once we have $\boldsymbol{E}$, we can factorize it into rotation $\boldsymbol{R}$ and translation $\boldsymbol{r}$.

Detecting Unreliable Estimation

The basic algorithm presented above works well as long as feature tracking results are good and the camera motion contains a large translational component $|\boldsymbol{r}|$.

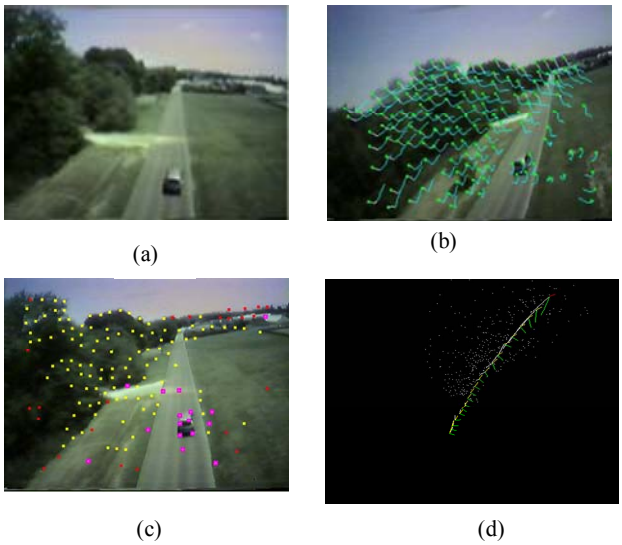
Outliers in feature tracking come both from gross errors and from feature points on moving objects. Since least-squared technique is sensitive to outliers, we use the RANSAC algorithm to detect the outliers and estimate the Essential matrix.

Dealing with a case where the camera translation is too small is far more difficult. In the extreme case, when the camera undergoes only rotation, the eight-point algorithm outputs a random translational vector $\boldsymbol{r}$. Using the amount of 2D motion is not enough to detect such degeneracy, since there are indeed large 2D motions when the camera undergoes even only rotation.

For the no-translation case, the 8-point algorithm outputs random estimation to $\boldsymbol{r}$, but still outputs correct rotation estimation (see appendix for a simple proof). Therefore the degeneracy can be handled by the following method:

1) Estimate the camera rotation using 8-point algorithm.
2) Transfer the feature points $\boldsymbol{M}$ in the first camera to the second camera by $\boldsymbol{M}_t = \boldsymbol{R}\,\boldsymbol{M}$.
3) Compute $\Delta\boldsymbol{r} = \boldsymbol{M}' - \boldsymbol{M}_t$, where $\boldsymbol{M}'$ in the second frame is the correspondence of $\boldsymbol{M}$ in the first frame. The displacement $\Delta\boldsymbol{r}$ characterizes the amount of parallax about the point $\boldsymbol{M}$. If the camera translation is zero, then $\Delta\boldsymbol{r}$ will be zero, too.
4) If all of the points have small $|\Delta\boldsymbol{r}|$, then we declare degenerate camera motion. In such a case, we only update the camera orientation, and wait until there are enough points with large $|\Delta\boldsymbol{r}|$ for the estimation of camera translation.

The second degenerate case is when the rank of $\boldsymbol{A}^T\boldsymbol{A}$ (a 9x9 matrix) is less than 8. We can prove that if camera translation is zero, then the rank of $\boldsymbol{A}^T\boldsymbol{A}$ is no more than 7. Since the essential matrix is the one-dimensional null space of $\boldsymbol{A}^T\boldsymbol{A}$, it requires the rank of $\boldsymbol{A}^T\boldsymbol{A}$ be 8 for a reliable estimation of $\boldsymbol{f}$, i.e., the essential matrix. Therefore, if the last two smallest eigen values of $\boldsymbol{A}^T\boldsymbol{A}$ are similar, that signals unreliable estimation of translation, but still outputs the camera rotation.



(a) (b) (c) (d)

Figure 4.1: A snapshot of our system. (a): image source view; (b) tracker view shows 2D image motion trajectory; (c) detected outliers, larger pink squares are moving objects, and small red dots are missing tracks; (d) the final vehicle trajectory and 3D of feature points recovered. This video is a real video taken from an actual micro air vhecle.

Long Sequence SFM by Merging Multiple Two-Frame SFM

The above two-frame SFM is applied whenever the 2D features have sufficient motion in the image sequence. Since we can only recover the translational direction, the depth of the 3D point is defined up to an unknown scale. To unify the 3D scales from multiple two-frame reconstructions, we need to merge the 3D structures multiple SFM reconstructions by finding the scale among them. Such scale estimation is critical in a long-sequence SFM. For this purpose, previous work relies on either initial two-frame SFM reconstruction [1] or "reliable" feature track [2]. It is desirable to avoid such critical dependence on particular information that may or may not correct.

Suppose that we have N points in the scene whose current 3D positions are $\boldsymbol{M}_i$, $i=1, 2, ..., N$ in the reference world coordinate frame, and that we denote their corresponding positions in the coordinate frame of the current image (obtained from two-frame SFM estimation) by $\boldsymbol{M}'_i$. For each point $\boldsymbol{M}_i$ we can estimate the scale according to

$$M_i' = s_i(R_1 M_i + t_1);\ \ i = 1, ..., N$$

Then the scale s between the two reconstructions is:

$$s = \frac{\sum_i w_i s_i}{\sum_i w_i}$$

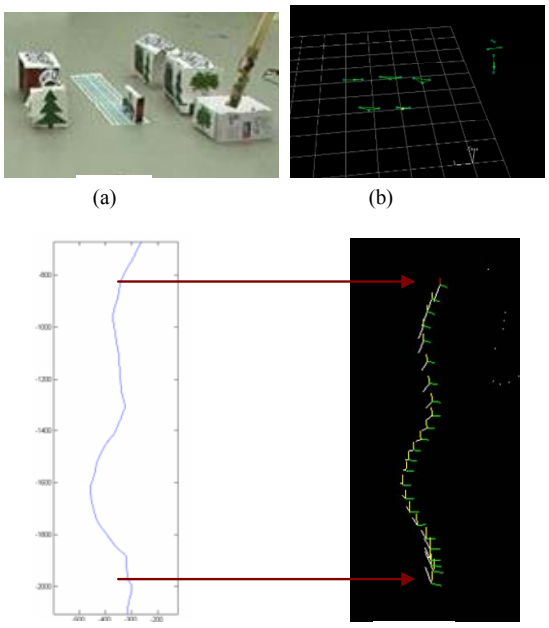
Here w_i is the weight for point $\boldsymbol{M}_i$ based on reconstruction reliability, which depends on camera configuration and depth of $\boldsymbol{M}_i$:

$$w_i = \rho(z_i, \theta_i, r_i)$$

Here ρ is a robust function, is the angle between the two rays cast from the camera centers to the 3D point in the scene, and r_i is the intensity residual from 2D tracking. In such a weighting scheme, a reliably tracked point closer with more parallax in image plane will receive larger weight. Due to noise and/or outliers in the 3D estimation, the above estimation could be un-reliable, too, if done naively. We use LMedS [10] to initialize the estimation, and then use a weighting algorithm to derive the final maximum likelihood estimation; points with large reconstruction variance or near the direction of the camera translation receive less weight.

Results of Vision based 3D motion estimation

Figure 4.1 shows an example output of our system consisting of feature detector, tracker and SFM. The system runs in real-time with a standard 2-GHz single-CPU PC. Images from the camera are fed to the system and are displayed in view (a). The tracking results are displayed in (b), where yellow dots are newly generated feature points in the current frame. View (c) shows the outliers as well: pink squares indicate moving objects (the car and the moving light), and red points that are considered missed (disappeared) during the tracking. The recovered camera motion is displayed in (d).



(a) (b) (c) (d)

Figure 4.2: Verification by ground truth from motion capture system: (a) the scene model; (b) one view of the motion capture system; (c) camera trajectory from motion capture system; (d) camera trajectory from our vision system.

To verify our system, we use a motion capture system to capture the motion of the camera. The motions output from the motion capture system are very accurate and can be used as ground truth to verify the camera motions recovered by

our system. Figure 4.2 shows the verification result. Figure 4.2(a) shows the scene model. We use the same camera as the one used in MAV vision system. The camera is attached to a stick and moved by hand to simulate the MAV motion. Figure 4.2(b) shows a snapshot of the captured motion and 3D recovered by motion captured system. Figure 4.2(c) shows the motion trajectory output by motion capture system. Figure 4.2(d) shows the camera trajectory recovered by our 3D vision system. It can be seen from (c) and (d) that our vision system recovers good camera motion trajectory.

IV. Scene Mapping by Laser Scanner from a Small UAV Platform

A laser ranger finder is an effective sensor to map the three-dimensional environment from a small UAV platform, when combined with precise estimates of vehicle's position and pose. The map information is in turn used for three purposes: automatic target recognition to extract the location of potential targets; feature recognition algorithms to classify the scene and identify features such as buildings, roads, and bridges; visual odometry to maintain high quality navigational updates in the event of GPS disruption.

Autonomous Helicopters

We have been developing vision-based autonomous helicopters [8]. The current fleet consists of three mid-sized (~3m long) unmanned helicopters. On-Board systems include: a state estimator (integrating GPS/IMU/vision), flight controller (capable of accurate (~0.2m) hovering and tested for autonomous forward flight (up to 40 Knots and~30 degree bank angle), laser scanner (900 nm, 120 m range, 12 KHz frequency) with optics for calibrated color sensing, actuated pan-tilt camera system (Sony DXC-9000), and multi-CPU computing for general purpose vision algorithms.

The demonstrated capabilities so far include: 1) unmanned take off and landing, 2) 3D terrain mapping that was deployed in NASA's Haughton-Mars expedition (1998) [7], surveying the US Airways flight crash site (Sept, 2001), and mapping the MOUT site at Fort Polk for the DARPA Perceptor Program, 3) vision-based locating of a 10-cm diameter pack on the ground and retrieving it from the air by visually servoing a magnet at the end of strings to reach it (winner of 1997 Unmanned Aerial Robotics Competition by perfectly completing the task), 4) forward scouting the obstacles and holes by 3D laser vision for autonomous ground vehicles, and 5) pointing a laser to a target at 100m away at precision <50cm.

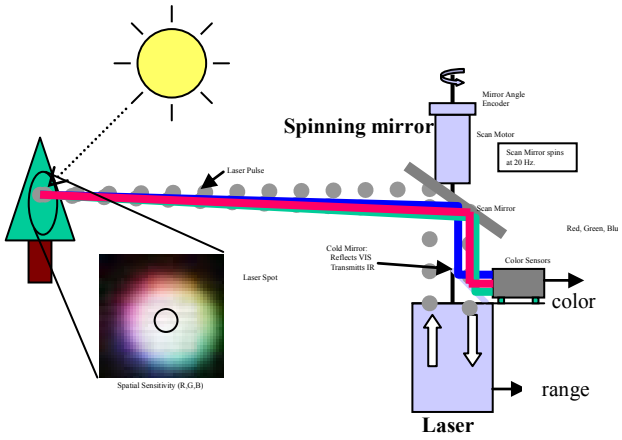


Figure 5: Color sensing combined with range sensing. Color optics is co-axially aligned with the laser path so that color and range information is automatically registered.

On-board Scanner

A helicopter with an onboard scanner flies over and scans the terrain. The range finder on board is a pulse-based (1ns) time-of-flight sensor (LADAR) with one-axis scanning mirror, collecting up to 6000 range data per second with raw range precision of 2cm.

In addition to the range measurement, our sensor can also measure the color of the surface [10]. As shown in Figure 5, a high-sensitivity color sensor is optically aligned with the laser path so that the color information (the sun light reflected by the surface) is measured simultaneously.

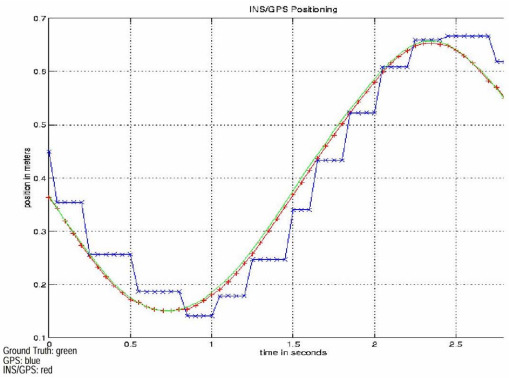


Figure 6 State estimation by GPS/INS integration. The blue line is the GPS position measurement. The red curve is the integrated state estimate while the green curve is the ground truth.

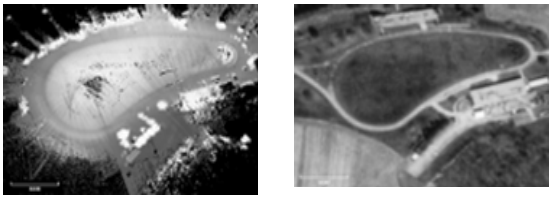
Helicopter State Estimation

Robust and precise state estimation is among the most important capabilities for UAV's. "Robust" refers to the high level of dependability that is required for successful low-altitude flying. The state estimator must be tolerant of sensor failures and be intelligent in managing available

resources to maintain relative state at all times.

Our existing complete state estimation capability uses an on-board inertial measurement unit (Litton LN-200) and GPS receiver (NovAtel RT2 dual-band carrier-phase), magnetic compass (KVH), and laser altimeter (Yamaha). The on-board computing runs our custom-built inertial navigation system software. The system implements a latitude-longitude mechanization and fuses inertial and GPS data using a 13th order Kalman Filter. The filter updates position, velocity, pose, and sensor parameters such as accelerometer biases modeled as stochastic processes.

The performance of the system is evaluated by ground truth sensors and has been verified when used with a GPS receiver as a direct position input (not satellite raw pseudo-range). Figure 6 shows an example. The blue (jagged) graph is GPS data, the green is ground-truth position measured by strapped down transducers and the red is the resolved position estimate of the helicopter. Note the high accuracy of the predicted position after each GPS sample.



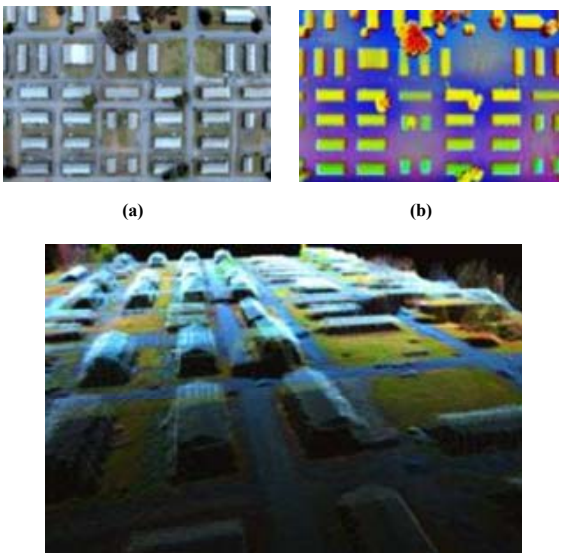
(a) (b)

Figure 7: (a) Terrain mapping result by an autonomous helicopter displayed as a 3D elevation map; (b) comparing this with USGS orthophoto of the same site demonstrate the precision achived.

Map Generation

The raw LADAR sensor, combined with the forward motion of the helicopter, collects a set of terrain profiles, each of which is sensed along the flight path. A swath of terrain up to 200 m wide (and any length) is covered in a single pass. Larger areas are scanned by systematic flying patterns that completely cover the area. The collected data are transformed into a single rectified coordinate system (geodesic or certain task-oriented coordinates), using the precise knowledge of pose and position estimate – the output of the state estimator. Figure 7 shows one such result of mapping [9]. A test site is approximately 300m x 300m, and contains an asphalt road surrounding an open field with two large buildings and trees surrounding the area. The flight at this site was approximately 5 minutes in duration, and produced over 2.5 million 3D data points. A digital elevation map (figure 7(a)) with a 0.5m-square grid spacing was generated from the scan data. The intensity of each pixel indicates the average elevation measured for that location. Figure 7(b) shows a side-by-side comparison of this image with the USGS orthophoto. Ground truth measurements have verified the spatial accuracy of < 10 cm in all three axes.

Figure 8 shows another more recent example: (a) color image captured by the range sensor's color sensor, (b) range map (color coded – red to blue for high to low), and (c) perspective view.



(a) (b)

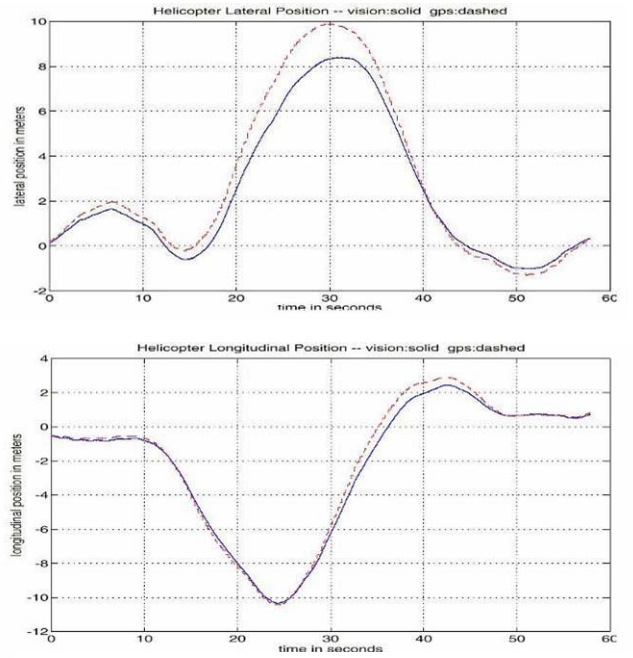
(c)

Figure 8: Another example of terrain mapping

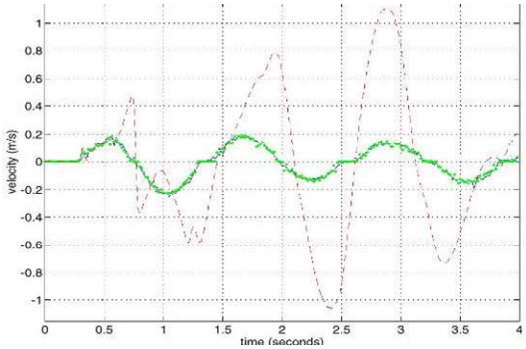
V. SENSOR FUSION FOR STATE ESTIMATION

The real-time motion (and scene) recovery method in section III was purely vision-based to be used for micro-UAV, and the state estimator of the small autonomous helicopter in section IV was mainly GPS/INS-based.

The two sources of information, vision and GPS/INS, for air vehicle state estimation are complementary. For example, we can intuitively see the following. Distinguishing between small rotation and small translation is extremely difficult from images alone, because with a limited field of view a yaw rotational motion can appear virtually the same as a lateral translation. This ambiguity can be resolved by fusing gyro sensors to counteract image ambiguity. We can see this in the equations of the two-frame SFM. If the rotation is available from other sensors, we can use it to transfer the feature points in the first image to the second image, and then compute the 2D parallax $\Delta \boldsymbol{m} = \textit{Projection-of}(\Delta \boldsymbol{M})$. The camera translational direction can be estimated as the focus of expansion (FOE) via the intersection of all parallax vector $\Delta \boldsymbol{m}$. Such intersection can be estimated reliably in a maximum likelihood framework because of a large amount of redundancy (estimating two parameters with many parallax vectors). Likewise, certain knowledge of translation will help stabilize the computation of rotation.



(a) Comparison of vision-based state estimation and GPS/INS based state estimation: Red dashed – vision, Blue-solid –GPS/INS



(b) Fusion of vision and inertial sensor: red dashed – inertial sensor integration only, green – inertial plus vision

Figure 9: Comparison and integration of multi-sensor fusion for state estimation

On the other hand, estimating a large motion by integrating inertial sensor output over a period is prone to drift (especially at slow speed), but vision can provide a good estimate as it ensures a long baseline. Further, while GPS provides stable geodesic position information, it can suffer from intermittent loss of measurement due to a shadow or jamming problem. The vision-based position estimation can sustain the state estimation.

We have performed preliminary experiments on comparing and fusing the GPS/INS-based state estimator and the vision-based estimator (visual odometer) in an autonomous helicopter – the particular program for visual odometer in this experiment is an older program than the one presented above.

Figure 9 (a) shows comparison of the output of the visual odometer's position (solid) with GPS-based position (dashed) while flying diagonally forward and backward. Figure 9 (b) shows the comparison of velocity estimation between simple integration of inertial sensor (dashed) and fusion of gyro and vision (green) while it is hovering. The results shown demonstrate promise for sensor fusion approach.

VI. CONCLUSION

Small and micro autonomous UAV's (SMUAV's) have great potential for various applications. Real-time 3D vision is a critical and integral part of such SMUAV's.

References

[1] P. A. Beardsley, A. Zisserman, and D. W. Murray. Sequential updating of projective and affine structure from motion. *International Journalon Computer Vision*, 23:235–259, June 1997.

[2] A. Chiuso, P. Favaro, H. Jin, and S. Soatto. Structure from motion causally integrated over time. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 24(4):523–535, April 2002.

[3] R. I. Hartley and A. Zisserman. Multiple View Geometry in Computer *Vision*, Cambridge University Press, 2000.

[4] B.D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *IJCAI81*, pages 674–679, 1981.

[5] Jianbo Shi and Carlo Tomasi. Good features to track. In *CVPR'94*.

[6] Carlo Tomasi and Takeo Kanade. Detection and tracking of point features.Technical Report CMU-CS-91-132, Carnegie Mellon University,1991.

[7] P.Lee, "The Haughton-Mars Project: Study of the Haughton Crater – Devon Island, Canadian High Arctic Viewed as a Mars Analog", NASA Technical Report.

[8] O. Amidi, T. Kanade, and R. Miller, "Vision-based Autonomous Helicopter Research at Carnegie Mellon Robotics Institute" 、 Proceedings of Heli Japan '98. Gifu, Japan April 1998

[9] R. Miller and O. Amidi, "3D Site Mapping with the CMU Autonomous Helicopter", Proc. 5th International Conferenece on Intelligent Autonomous Systems (IAS-5), June 1998

[10] R. Miller, "Modeling System for Small Autonomous Helicopters", PhD Thesis Dissertation, The Robotics Institute, Carnegie Mellon University, Pittsburgh, Pennsylvania 15213, Spring 2002.

[11] Z.Y. Zhang, "Determining the epipolar geometry and its uncertainty:A review," International Journal of Computer Vision, vol. 27, pp. 161--195, March 1998