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

Vision Based Terrain Recovery for Landing Unmanned Aerial Vehicles

Marci Meingast, Christopher Geyer, and *Shankar Sastry*
{marci,cgeyer,sastry}@eecs.berkeley.edu
University of California, Berkeley

***Abstract*— We propose a system for landing Unmanned Aerial Vehicles (UAV), specifically an autonomous rotorcraft, in uncontrolled, arbitrary, terrains. We present plans for and progress on a vision-based system for the recovery of the geometry and material properties of local terrain from a mounted stereo rig for the purposes of finding an optimal landing site. A system is developed which integrates motion estimation from tracked features, and an algorithm for approximate estimation of a dense elevation map in a world coordinate system.**

I. Introduction

Navigation of *unmanned aerial vehicles* (UAV's) is a topic of great interest to the control community. The specific task we focus on in this paper is the landing of an unmanned helicopter in an arbitrary uncontrolled environment where a landing site cannot be carefully pre-selected. Due to power constraints and the desire to avoid detection we do not use radar or other active range sensing devices. Instead we propose a method for autonomous landing based on principles of computer vision. The feasibility of using computer vision has been demonstrated by reliable and robust on-board computer vision algorithms in the control loop, whether tightly coupled in the case of visual servoing [1] and helicopter landing [2].

Vision-based control for autonomous model helicopter landing have been demonstrated by [2], [3] and [4]. In the first two cases, the vision-based landing was facilitated by the presence of a predefined landing target or pattern. This allowed the vision algorithm to tractably locate the landing target using its predetermined characteristics and continuously estimate the ego-motion with respect to the target. In the former case, the method enabled fast vision algorithms in the control loop and made landing on a non-stationary target possible. In certain settings, however, it is not realistic to assume planned or preselected landing sites. In [4], a system is proposed for applying stereo imagery to locate potential hazards. This system differs in that we will integrate sensor data over multiple time-frames to mitigate errors in depth. Another approach that does not assume a controlled environment is taken by Srinivasan [5], in which landing is performed by controlling so that image velocity constant. In that paper the concern was not so much the selection of a safe landing site, as the problem of safe visual servoing of the vehicle to the ground. Other work connected with computer vision and autonomous aircraft is the work by Amidi et al. [6] where the authors use a single point feature to aid inertial measurements in position and motion estimation. In [7] and [8], the authors use a 1D scanning laser range-finder to derive 3D models.

In this paper, we propose a novel integration of concepts from computer vision to (a) estimate a digital elevation map with passive sensors; (b) select a safe landing site from elevation and appearance, avoiding trees and tall grasses; and (c) drive the UAV to the selected landing site. The overall landing system consists of a low-level vision sub-system which recovers a ground elevation map, a map-building component which classifies ground objects using appearance and the recovered topography, and a high-level navigation system for specifying way-points to the desired landing target. The organization of the paper is as follows. In section 2, we discuss the overall landing system and introduce each of the components. Section 3 goes into the detail of the vision sub-system and the algorithms used in this area. Section 4 discusses the map building sub-system and navigation architecture. Concluding remarks and future work are discussed in section 5.

II. System Setup

Our system is composed of three main parts:(1) a vision sub-system (2) a map building sub-system and (3) a navigation sub-system. The inputs are readings from GPS and INS as well as images from a stereo camera pair. The output will be a set of way points that map the path from the current UAV position to the desired landing position which will be fed to the UAV's low level controller. The setup can be seen in 1.

In designing the vision sub-system our primary goals are: (1) recovery a dense ground elevation map; (2) accurate height estimation; and (3) real-time operation capable of supplying height estimates at 1hz or faster. One choice crucial to the design of the system is the number of cameras to use. Dense reconstruction suggests the use of a stereo rig. A stereo rig, however, has several disadvantages: they need to be calibrated; two frames potentially require extra computation per frame; there is an increase in system complexity; necessity of a rigid stereo platform; and, the fact that error in depth is *proportional to the square of the depth*. On the other hand, in the event of poor past height estimates, a stereo system can be used to estimate depth in a single time step, and stereo may also be used to aid in outlier rejection for structure-from-motion algorithms. We

0-7803-8682-5/04/$20.00 ©2004 IEEE

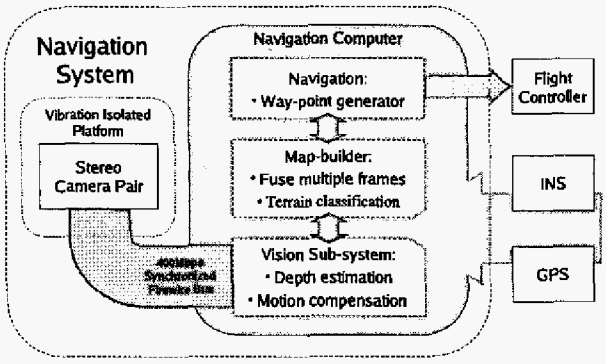


Fig. 1. A depiction of the overall architecture of the vision-based navigation system.

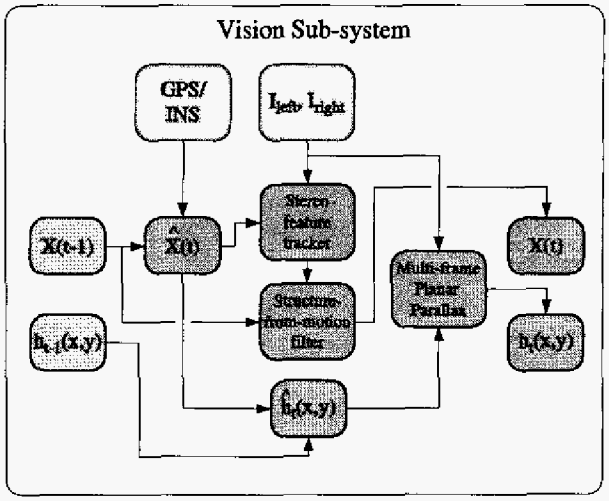


Fig. 2. The architecture of the vision sub-system.

decided that given the nature and danger of UAV flight that system complexity is a reasonable price to pay for greater robustness, and so we choose to use a stereo system. However, to compensate for the inaccuracy of fixed-baseline stereo, we choose an approach which measures height over several frames from the observed parallax.

The vision sub-system combines two main components: (a) ego-motion estimation from sparse features using a structure-from-motion filter as presented in [9]; and (b) dense elevation estimation using a multi-frame planar parallax estimation developed by Irani et al. in [10] and [11]. This second algorithm assumes multiple images view a single virtual plane in space; all images can be warped so that this virtual plane occupies the same part of the image in all images. For any space point not on the virtual plane, the point's height above or below the plane can be determined from the point's parallax or relative motion in all registered frames. This procedure requires all images to be warped to the same virtual plane. We combine *inertial navigation system* (INS), *global positioning system* (GPS) and ego-motion estimates using structure-from-motion to estimate the helicopter's position with respect to a world coordinate system; then, we let the virtual plane be the ground plane.

The elevation map is to be fed to a map building sub-system which further analyzes the terrain images to determine where a safe site for landing may be. This sub-system has two main tasks: (1) to fuse multiple frames of terrain together and (2) to classify the terrain as to quality of landing. Judging quality of landing for the terrain ought to take into account issues such as slope of the land and terrain type (i.e. water, tall grass, etc.) which are issues that could inhibit safe UAV landing.

Once a landing site is determined, then the navigation sub-system will choose a set of way-points to a safe landing site. The navigation sub-system is still under design and in the rest of this paper, we will focus on the vision sub-system and its details. We will go over the general outline for the map building sub-system and navigation architecture, but the emphasis is on the computer vision methods used by the vision sub-system for recovering a dense elevation map.

III. Vision Sub-system

The vision sub-system has three main algorithmic tasks in order to provide an estimation of the elevation map of the terrain: (1) Ego-motion estimation (2) Motion estimation filtering and (3) planar parallax computation. As shown in figure 2, previous ego-motion results along with current reading from INS, GPS and the camera images are the input for the motion estimation filtering. Here, predictions of the motion of the image points at the current time step are computed. These predictions are fed to the ego-motion estimator and used as bounds to determine where feature points will lie. Once the feature points are tracked, these are given to the multi-frame planar parallax algorithm, along with the current images, in order to compute an elevation map.

A. Ego-motion estimation

In order to apply the results of the multi-frame planar parallax algorithm, it is necessary to warp the images to a constant reference plane in the scene. Only then can the relative parallax at each point be measured. In this case we measure parallax with respect to the local ground plane at sea-level, where for sufficiently small distances the curvature of the earth can be ignored. The spatial relationship between this plane and the helicopter is determined using GPS and INS measurements. However, the global position estimates can be inaccurate and the attitude tracking can drift; therefore, to increase accuracy in ego-motion tracking we incorporate estimates from a so-called structure-from-motion filter [9]. Such a filter uses trajectories of a sparse set of image features, as well as the constraints of two view geometry, to estimate the ego-motion of a camera. There is no strictly optimal finite-dimensional recursive filter for the structure-from-motion problem—that is, the simultaneous recovery of motion and structure from a set of images—an approximation can be obtained, though, by linearizing the

state and measurement equations and applying an *extended Kalman filter* (EKF).

The state of the system is encoded in a vector which incorporates the spatial coordinates of the tracked features as well as the motion of the camera. Specifically the state is described by the following vector

$$\mathbf{X}(t) \;=\; (\mathbf{x}_1, \ldots, \mathbf{x}_n, \mathbf{R}(t), \mathbf{T}(t), \omega(t), \mathbf{v}(t))^T\,,$$

which is an element of $\mathbb{R}^{3n+12}$. We assume the scene to be rigid, and so each $\mathbf{x}_i \in \mathbb{R}^3$, which represents a point in the world, is constant for all time. The remaining $\mathbf{R}(t)$, $\mathbf{T}(t)$, $\omega(t)$ and $\mathbf{v}(t)$ respectively encode the rotation, the translation, the angular velocity, and the linear velocity with respect to the world coordinate system. This evolution of the system is dependent on the dynamics of the helicopter, we assume that these dynamics can be modeled by a discrete-time non-linear system:

$$\mathbf{X}(t+1) \;=\; f(\mathbf{X}(t), \mathbf{u}(t))\,, \qquad (1)$$

for some f and where $u(t)$ is the input to the system.

At time t the camera perspectively projects any point $\mathbf{x}$ in the scene to the left and right cameras' image planes using camera projection matrices, so that for example:

$$\mathbf{P}_L(t) \;=\; \begin{pmatrix}\mathbf{R}_L\,\mathbf{R}(t) & \mathbf{R}_L\,\mathbf{T}(t) + \mathbf{T}_L\end{pmatrix}$$

is the left camera projection matrix, where $R_{L,R}$ and $T_{L,R}$ determine the coordinate systems of the left and right cameras with respect to the helicopter's coordinate system. The measurement equation for the left camera then becomes

$$\mathbf{y}^L(t) \;=\; p\left(\mathbf{P}_L(t)\begin{pmatrix}\mathbf{x}\\ 1\end{pmatrix}\right) + \mathbf{n}_L\,. \qquad (2)$$

where $\mathbf{n}_L, \mathbf{n}_R \sim N(\mathbf{0}, \mathrm{diag}(\sigma, \sigma, 0))$, and p is the canonical perspective projection

$$\pi(x, y, z) \;=\; (x/z, y/z, 1)^T$$

so that $\mathbf{y}^{L,R}(t) \in \mathbb{P}^2$, the projective plane represented in homogeneous coordinates. Equation (2) implicitly assumes that we are able to measure the projection of some constant point $\mathbf{x}_i$ in space. In practice we identify trackable patches in the first left image, find a match in the first right image obeying the epipolar geometry of the stereo pair and with similar appearance (measured by SSD of intensities), and then track the features into the next frame, again by minimizing SSD of intensities. Linear and angular velocity estimates can be used we predict where the feature will appear in the next pair of frames, so as to identify the corresponding point in the next frame.

Without going into further details of the tracking algorithm, which is a modification of [12], [13], the implicit measurement equation is therefore

$$\mathbf{Y}_1(t) \;=\; \left(\mathbf{y}_1^L(t), \ldots, \mathbf{y}_n^L(t), \mathbf{y}_1^R(t), \ldots, \mathbf{y}_n^R(t)\right)^T + \mathbf{N}_1\,,$$

where $\mathbf{N}_1 \sim N(\mathbf{0}, \Sigma_{\text{track}})$, with Σ_{track} being an approximation of the block diagonal covariance in feature tracking, and image points $\mathbf{y}_i^L$ are from the left camera, $\mathbf{y}_i^R$ from the right. The homogeneous coordinates again have zero covariance. This vector corresponds to the projections of points $\mathbf{x}_1$ through $\mathbf{x}_n$ obtained using equation (2). In addition we incorporate estimates of position, attitude and velocities from GPS and INS:

$$\mathbf{Y}_2(t) \;=\; \left(\hat{\mathbf{R}}_{\mathrm{G}}(t), \hat{\mathbf{T}}_{\mathrm{G}}(t), \hat{\omega}_{\mathrm{G}}(t), \hat{\mathbf{v}}_{\mathrm{G}}(t)\right)^T + \mathbf{N}_2\,.$$

where $\mathbf{N}_2 \sim N(\mathbf{0}, \Sigma_G)$, $\hat{\mathbf{R}}_{\mathrm{G}}$ and $\hat{\mathbf{T}}_{\mathrm{G}}$ are estimates of attitude and global position, and $\hat{\omega}_{\mathrm{G}}$ and $\hat{\mathbf{v}}_{\mathrm{G}}$ are estimates of angular and linear velocity respectively. This equation is to be interpreted as a linear approximation, in particular, the covariances lie within the tangent spaces of $\mathrm{SO}(3)$ and $\mathfrak{so}(3)$ respectively. We assume that Σ_G is provided by the inertial navigation unit. The entire measurement equation is $\mathbf{Y}(t) = (\mathbf{Y}_1(t), \mathbf{Y}_2(t)) = g(\mathbf{X}(t))$ for some non-linear g. The combined observation covariance is $\mathrm{diag}(\Sigma_{\text{track}}, \Sigma_G)$. Details on the implementation of the extended Kalman filter can be found in [14]. One important detail is what to do in case new features come into view. The position of any newly introduced feature is unknown up to a ray in space, therefore directly inserting it into the filter will disturb the estimates of the other states. They suggest a parallel sub-filter for point estimation using already built estimates of motion.

B. Multi-frame Planar Parallax

Having determined the motion of the vehicle, the next step is to estimate a digital elevation map (DEM). We do so by applying a method called multi-frame planar-parallax, developed by Irani et al. in [10] and [11]. This method exploits the decomposition of apparent motion of an image point into two components: a planar component, or equivalently a planar homography, and a parallax component—that is apparent motion due to non-planarity of a surface whose magnitude is dictated by the height above the plane.

To begin with we construct a coordinate system whose origin $\mathbf{x}_0 = (0, 0, 0)$ corresponds to a designated point in space lying at sea-level, and such that the z-axis is perpendicular to the earth's spheroid; thus the plane represented by $\pi_{sea} = (0, 0, 1, 0)$ is tangent to this spheroid at sea-level. An integrated GPS and INS sensor yields estimates $\hat{\mathbf{R}}_G$ of the rotation and $\hat{T}_G$ of the translation of the helicopter coordinates relative to a coordinate system determined by the direction to the north pole and the vertical directions, as well as some local position on a local sea level plane.

For each camera at any given time there is a 4×3 matrix representing the transformation from homogeneous coordinates in the image plane to homogeneous space coordinates of points on π_{sea}. For the left camera this transformation is:

$$\mathbf{G}_L(t) \;=\; \begin{pmatrix}\mathbf{P}_L(t)\\ \pi_{sea}\end{pmatrix}^{-1}\begin{pmatrix}I\\ 0\end{pmatrix}.$$

For then $\mathbf{G}_L(t)$ satisfies $\pi_{sea}^T \mathbf{G}_L(t)\mathbf{y} = 0$ and $\mathbf{y} = p(\mathbf{P}_L(t)\mathbf{G}_L(t)\mathbf{y})$, i.e. it sends $\mathbf{y}$ to a point on π_{sea} whose

image is $\mathbf{y}$.

The transformation $\mathbf{P}_L(t_1)\mathbf{H}_L(t_0)$ induces a homography, that is, a 3×3 transformation of projective planes, such that if the point $\mathbf{x}$ lies on π_{sea}, then its image at $t = t_1$ can be written in terms of its image at $t = t_0$:

$$\mathbf{y}^L(t_1) \;=\; \underbrace{p(\mathbf{P}_L(t_1)\mathbf{H}_L(t_0)\mathbf{y}^L(t_0))}_{w_{L,t_0\to L,t_1}(\mathbf{y}^L(t_0))}.$$

For space points not lying on π_{sea}, $\mathbf{y}^L(t_1) \neq w(\mathbf{y}^L(t_0))$, where w is short-hand for the warp $w_{L,t_0\to L,t_1}$. In fact, in [15] it is shown that

$$\mathbf{y}^L(t_1) - w(\mathbf{y}^L(t_0)) \;=\; \frac{\frac{h}{z}}{\frac{1}{T_3}+\frac{h}{z}}\left(\mathbf{y}^L(t_0) - p(\mathbf{T})\right) \quad (3)$$

where z is the height of the left camera's viewpoint at $t = t_1$, h is the perpendicular height of h above π_{sea}, and here $\mathbf{T}$ is the image of the viewpoint at $t = t_0$ in the camera at $t = t_1$, i.e. the epipole, equal to:

$$\underset{L,t_1}{\mathbf{T}_{L,t_0}} = \mathbf{R}_L(\mathbf{T}(t_1) - \mathbf{R}(t_1)\mathbf{R}(t_0)^T(\mathbf{T}(t_0) + \mathbf{R}_L^T\mathbf{T}_L)) + \mathbf{T}_L.$$

In equation (3), the only unknown is h; all others, including the height z, the epipole $\mathbf{T}$, and the images $\mathbf{y}^L(t_0)$ and $\mathbf{y}^L(t_1)$, will already have been computed or determined. A similar equation of course exists between any pair of frames, left or right; only the computation of $\mathbf{T}$ changes.

The problem then is to determine $h(x, y)$, as a function in the image plane, such that

$$\mathcal{I}_1(\mathbf{y}) \approx \mathcal{I}_i\Bigg(\underbrace{\mathbf{y} + \frac{h(\mathbf{y})}{\frac{z}{T_3^i} + h(\mathbf{y})}\left(\mathbf{y} - p(\mathbf{T}^i)\right)}_{\mathbf{u}_i(h(\mathbf{y}),\mathbf{y})}\Bigg),$$

The criterion for choosing h will again be the SSD of intensities, so that

$$h = \underset{h:\mathbb{R}^2\to\mathbb{R}}{\arg\min} \sum_i \int_{\mathbf{y}} \int_{\substack{\mathbf{x}\in \\ \text{win}(\mathbf{y})}} \left[\mathcal{I}_1(\mathbf{x}) - \mathcal{I}_i\left(\mathbf{x} + \mathbf{u}_i(h(\mathbf{x}), \mathbf{x})\right)\right]^2$$

The summation is applied over multiple frames from left and right cameras, and the difference is taken with a constant left or right image. Finding the minimum is not trivial, but not essentially different from optical flow equations, see the classic [16]. The differences are that (a) the flow is constrained by the epipolar geometry; and (b) the number of unknowns is constant, that is, independent of the number of frames. The mechanism for finding h is gradient descent; to estimate the gradient we must differentiate the image, which we do by fitting a linear model to the image at each pixel, i.e.,

$$\mathcal{I}(x + dx, y + dy) \;=\; \tfrac{\partial \mathcal{I}}{\partial x}(x, y) \cdot dx + \tfrac{\partial \mathcal{I}}{\partial y}(x, y) \cdot dy\,. \quad (4)$$

The partial derivative in x is approximated by convolving the image with the kernel $(-2, -5, 0, 5, 2)/18$ in the x-direction and $(1, 4, 6, 4, 1)/16$ in the y-direction, resulting in an approximation of the partial derivative of a Gaussian kernel.

Initial estimates for h can be taken to be zero or interpolation of feature points. Then the question is, for what initial values of h do we expect to converge to the global minimum? The two factors influencing convergence are the radii for which approximations given by (4) are valid and the magnitude of the flow. The quality of the approximation is determined by the image, but beyond one to two pixels is usually unreliable. To obtain convergence for flows beyond one to two pixels we can subsample the image thereby effectively halving the magnitude of all flows. This multi-scale approach is suggested in [11], and is a typical way to dealing with large motions.

IV. Quality of Landing

Once the elevation map of the land has been determined from the vision system, it will be fed to a map building subsystem. Here it is decided whether there is an available patch of ground the helicopter could land on. Multiple factors play a role in this determination.

A. Qualities for Landing

In deciding what constitutes a good landing site for the helicopter, we must look at different characteristics of the terrain. First, there must be an area large enough for the helicopter to land. Based on the distance of the helicopter from the ground, we can determine what size area of ground a pixel in the image represents. Then, we can build a window that represents the size of land the helicopter will need. Using this window, we can search over the elevation map from the vision system, to determine if there is a sufficiently smooth and planar area of the correct size.

If a smooth planar area is found, then the slope of that area needs to be analyzed. The the helicopter may not be able to land in an area that has too high of a slope for dynamical reasons. Thus, we want not only a smooth planar area, but one that has a reasonable slant to it, given the constraints of our helicopter.

The classification of terrain that is seen in the image must also be assessed. Even if all the previously mentioned qualities are satisfied, if the terrain we are looking at is water, we do not want to touch down in this area. Treetops, tall grasses, water and other sorts of terrain pose problems for landing.

B. Quality of Landing Function

Using desired qualities of landing, we build a function to analyze each pixel in the image with regards to these aspects. The function is a weighted sum of the factors and is defined as follows:

$$Q_l(x, y) \;=\; -\alpha\, F(h) - \beta\, G(h, \mathcal{I}) \quad (5)$$

where h is the elevation from sea level found in the planar parallax method, and F, G are functionals of h and in the case of G, an estimate of appearance of the surface. This

function can be easily extended to include other factors, for instance fuel economy for landing, if desired—e.g., finding a non-optimal landing site is better than falling from the sky.

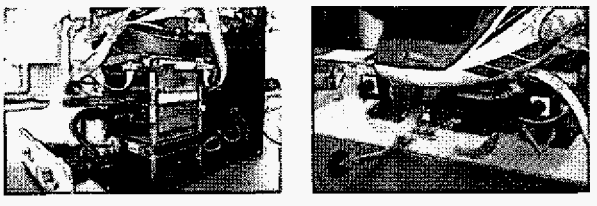
When a desirable landing site has been found, using the current velocity and position of the helicopter in world coordinates and the desired landing site world coordinates, a navigation system can map a landing path. Setting up way points that will guide the helicopter to the landing site with the desired orientation and speed, these can be fed to the helicopter controller. When the helicopter reaches a way point, the vision system will analyze the terrain from this location for an elevation map and the map builder will determine if there is still a safe area for landing available.

V. Testbed

Implementation of the algorithms described here is being executed on our helicopter testbed. The following sections describe the vision system hardware, vision software design, and flight computer.

A. Hardware

The vision hardware consists of off-the-shelf components: (i) two Firewire cameras from Videre Design with Bayer color pattern mounted on a vibration isolated platform; (ii) a 1.6Ghz Pentium M with 1GB of memory, a Firewire interface, and a IEEE 802.11b PCMCIA wireless card, all in a PC/104+ stack from Advanced Digital Logic. See figures 3.a-b. In designing the stereo camera platform we were primarily interested in ensuring that camera vibration, when it occurs, is coupled; therefore, the two cameras are attached to the platform via a stiff graphite fiber rod; the platform itself is attached to the helicopter via dampers at its four corners. In chosing the CPU, we wanted to maximize speed and cache size; the 1.6Ghz Pentium M has a 1MB L2 cache, and though it is based on the Pentium 3, it includes MMX, SSE and SSE2 instructions which we hope to take advantage of in the future, and it exceeds a 2.4Ghz Pentium 4 in performance [17].



(a) (b)

Fig. 3. (a) Photo of PC/104+ stack out of box; the left lower panel holds the Firewire connectors; CPU resides on top. (b) Stereo camera mount; cameras rotated upwards for picture only.

B. Software

The architecture for the vision landing system is a set of asynchronous threads that

- capture images from the Firewire cameras with the *libdc1394* API for DCAM compatible digital cameras [18];
- make captured frames available in a repository and garbage collect old frames;
- selects and tracks features taken from repository;
- provides repository for features;
- estimates motion from features;
- warps frames to reference frame;
- estimation of digital elevation map from parallax;
- map construction and landing site selection.

The first four of these threads have been implemented in C++ on the vision computer which runs MandrakeLinux 9.1, where threading support is provided by the POSIX Pthreads library. The monocular tracking system currently tracks at least 50 features at 50hz.

C. Yamaha R-50 Testbed

The actual testbed helicopter is a customized industrial Yamaha R-50 model helicopter equipped with instrumentation described in detail in [19]. In brief, this system consists of: (1) a Pentium based (233Mhz LittleBoard PC) navigation computer running QNX and responsible for real-time flight control; and (2) an inertial measurement unit consisting of a NovAtel MillenRT2 GPS and Boeing DQI-NP INS/GPS integration system.

VI. Conclusion

In this paper, we looked at the issue of how to land a helicopter in unknown terrain. We described the overall system and looked in detail at the vision sub-system. The vision sub-system uses a combination of feature tracking, motion estimation, and multi-frame planar-parallax in order to estimate a digital elevation map of the terrain. The elevation map, in combination with the map-building sub-system and navigation sub-system, will allow the determination of a safe landing area of terrain and map way-points to a desired landing spot.

We are currently looking at how to classify terrain for the map building sub-system. Using the color aspect from the cameras, texture of the image, and a library of terrain images, we want to determine what type of terrain is seen by the cameras in order to implement the quality of landing function. We are also looking into how to choose an optimal path and way points in the navigation sub-system. An open problem is how to implement multi-frame planar parallax recursively, appropriately weighting past estimates of a digital elevation map. The implementation will be completed the Yamaha RS-50 testbed; remaining issues are appropriately deciding rates at which individual threads should run, and balancing trade-offs in image capture, tracking, waypoint specification and map estimation.

Acknowledgements

The authors are grateful for the generous support through DARPA grants DAAD-19-02-1-0383, Boeing sub-contract

Z40705R of DARPA funded SEC program managed by AFRL, and NSF grant IIS-0122599. The authors would like to thank the anonymous reviewers for their comments, and the contributions of members of Boeing Phantom Works, AFRL, and the SEC Program.

REFERENCES

[1] G. Hager, S. Hutchinson, and P. Corke, "A tutorial introduction to visual servo control," *IEEE Transactions on Robotics and Automation*, vol. 12, no. 5, pp. 651–670, 1996.

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

[3] S. Saripalli, J. F. Montgomery, and G. S. Sukhatme, "Visually-guided landing of an unmanned aerial vehicle," *IEEE Transactions on Robotics and Automation*, vol. 19, pp. 371–381, June 2003.

[4] J. Hintze, *Autonomous Landing of a Rotary Unmanned Aerial Vehicle in a Non-cooperative Environment Using Machine Vision*. Master's thesis, Brigham Young University, April 2004.

[5] K.Weber, S.Venkatesh, and M. Srinivasan, "Insect inspired behaviours for the autonomous control of mobile robots," *ICPR*, August 1996.

[6] O. Amidi, T. Kanade, and K. Fujita, "A visual odometer for autonomous helicopter fight," in *Proceedings of the Fifth International Conference on Intelligent Autonomous Systems (IAS-5)*, June 1998.

[7] N. Vandapel, O. Amidi, and J. R. Miller, "Toward laser pulse waveform analysis for scene interpretation," in *IEEE International Conference on Robotics and Automation*, May 2004.

[8] J. R. Miller and O. Amidi, "3-d site mapping with the cmu autonomous helicopter," in *Proceedings of the 5th International Conference on Intelligent Autonomous Systems (IAS-5)*, June 1998.

[9] S. Soatto, R. Frezza, and P. Perona, "Motion estimation via dynamic vision," *IEEE Transactions on Automatic Control*, vol. 41, pp. 393–414, March 1996.

[10] M.Irani, P. Anandan, and D. Weinshall, "From reference frames to references planes: Multi-view parallax geometry and applications," *ECCV*, June 1998.

[11] M. Irani, P. Anandan, and M. Cohen, "Direct recovery of planar-parallax from multiple frames," *ICCV*, September 1999.

[12] C. Tomasi and T. Kanade, *Detection and Tracking of Point Features*. Carnegie Mellon University Technical Report CMU-CS-91-132, April 1991.

[13] J. Shi and C. Tomasi, "Good features to track," in *IEEE Conf. Computer Vision and Pattern Recognition*, (Seattle, WA, June 21-23), pp. 593–600, 1994.

[14] Y. Ma, S. Soatto, J. Kosecka, and S. Sastry, *An Invitation to 3-D Vision*. Springer Verlag, 2003.

[15] H. Sawhney, "3D geometry from planar parallax," in *IEEE Conf. Computer Vision and Pattern Recognition*, (Seattle, WA, June 21-23), 1994.

[16] B. Horn, *Robot Vision*. MIT Press, 1986.

[17] Panopsys, LTD., "Pentium M Review," *http://www.cpuid.com/-PentiumM/index.php*, 2004.

[18] 1394 Trade Association, *IIDC 1394-based Digital Camera Specification*. 2000.

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