In IEEE Transactions on Robotics and Automation, Vol 19, No 3, pp 371-381, Jun 2003

Visually-Guided Landing of an Unmanned Aerial Vehicle

Srikanth Saripalli, *Student Member, IEEE,* James F. Montgomery, and Gaurav S. Sukhatme, *Member, IEEE*

***Abstract*— We present the design and implementation of a real-time, vision-based landing algorithm for an autonomous helicopter. The landing algorithm is integrated with algorithms for visual acquisition of the target (a helipad), and navigation to the target, from an arbitrary initial position and orientation. We use vision for precise target detection and recognition, and a combination of vision and GPS for navigation. The helicopter updates its landing target parameters based on vision and uses an onboard behavior-based controller to follow a path to the landing site. We present significant results from flight trials in the field which demonstrate that our detection, recognition and control algorithms are accurate, robust and repeatable.**

***Index Terms*— Unmanned Aerial Vehicle, Autonomous landing, Autonomous helicopter, Vision-based navigation.**

I. Introduction

UNMANNED AERIAL VEHICLES (UAVs) are sustained in flight by aerodynamic lift and guided without an onboard crew. They may be expendable or recoverable and can fly autonomously or semi-autonomously. Historically the greatest use of UAVs has been in the areas of surveillance and reconnaissance [1]. They are indispensable for various applications where human intervention is impossible, risky or expensive e.g. hazardous material recovery, traffic monitoring, disaster relief support, military operations etc. A helicopter is a highly maneuverable and versatile UAV platform for several reasons: it can take off and land vertically, hover in place, perform longitudinal and lateral flight as well as drop and retrieve objects from otherwise inaccessible places.

An autonomous helicopter must operate without human intervention, yet must meet the rigorous requirements associated with any airborne platform. A basic requirement is robust autonomous flight, for which *autonomous landing* is a crucial capability. Vision provides a natural sensing modality for object detection and landing. In many contexts (e.g. urban areas, airports) the structured nature of landing makes it suitable for vision-based state estimation and control. Inherently, vision alone cannot provide the necessary feedback for autonomous landing. This is because vision can only sense the motion created by applied forces, not the forces themselves. It is impossible for a controller to completely eliminate undesired movements due to disturbances (e.g. due to wind) after they are sensed. Precise helicopter maneuvers such as takeoff, trajectory following, and landing thus require inertial sensing and control.

We combine vision with low-level postural control to achieve precise autonomous landing of an unmanned model helicopter. The vision-based system described here acts as an overall controller sending navigational commands to a low-level controller which is responsible for robust autonomous flight and landing. The result is an overall algorithm for vision-based autonomous landing of a helicopter in an unstructured (other than the helipad, which is a structured cue) 3D environment. The vision problem being solved here is a special case of the ego-motion problem where all the feature points lie on a planar surface (in this case the landing pad) [2].

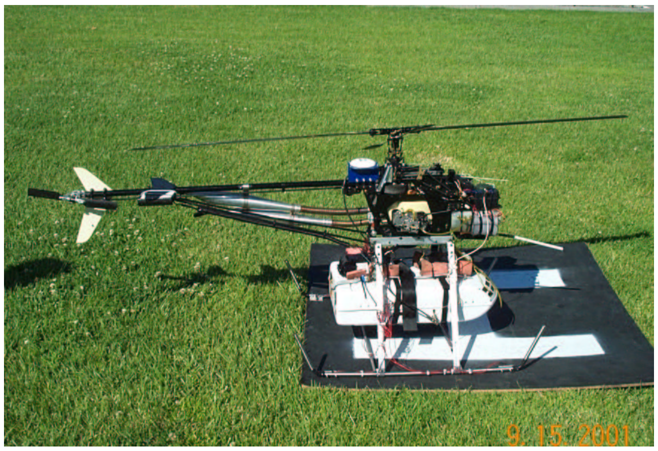

Fig. 1. The USC AVATAR (Autonomous Vehicle Aerial Tracking And Reconnaissance) after landing on a helipad

In our experiments the helicopter is initialized in hover at an arbitrary location. It is required to autonomously locate and recognize a helipad [Figure 1], align with it and land on it. After describing the structure of the helipad detection and control algorithms, we present results based on flight data from field tests which show that our method is able to land the helicopter on the helipad repeatably and accurately. On an average, the helicopter landed to within 31 cm position accuracy and to within 6^o in orientation as measured from the center of helipad and its principal axis respectively. We also present results which show the robustness of the algorithm, which allows the helicopter to find the helipad after losing it momentarily. Results are presented which show that the algorithm is capable of tracking a moving target and land on it, once it has stopped. In these experiments the helipad was moved a significant distance (7 m on an average).

Srikanth Saripalli and Gaurav S. Sukhatme are with the Robotic Embedded Systems Laboratory, Center for Robotics and Embedded Systems, University of Southern California, Los Angeles, USA. E-mail: srik,gaurav@robotics.usc.edu. James F. Montgomery is with the Jet Propulsion Laboratory, California Institute of Technology, Pasadena, USA. E-mail: monty@helios.jpl.nasa.gov

II. Related Work

Autonomous aerial vehicles have been an active area of research for several years. Autonomous model helicopters have been used as testbeds to investigate problems ranging from control, navigation, path planning to object tracking and following.

An early autonomous navigation system for a model-scale helicopter (the Hummingbird) was reported in [3]. The unique feature of this system was the sole use of GPS as the navigation sensor replacing the Inertial Measurement Unit, which is conventionally favored as the primary navigation sensor. The hummingbird GPS system consisted of a common oscillator and four separate carrier-phase receivers with four antennae mounted at strategic points of the helicopter body providing the position, velocity, attitude and angular information for vehicle control. The autonomous helicopter reported in [4] used an INS/GPS combination. The inertial measurements were sampled and processed by the onboard computer running numerical integration, the Kalman filtering algorithm [5], and simple PID control as the low-level vehicle control. The control gains were determined by tuning-on-the-fly. In [6] a system based on integration of the onboard INS and GPS was used to produce accurate position and velocity estimates. The autonomous helicopter reported in [7], [8], had a combination of vision and GPS for navigation capability. The onboard DSP-based vision processor provided navigation information such as position, velocity and attitude at an acceptable delay (on the order of 10ms), which was combined with GPS and IMU data for accurate attitude and position measurements.

Vision-based robot control has been an active topic of research [9]–[11]. In [4], a vision augmented navigation system is discussed for autonomous helicopter control which uses vision in-the-loop to control a helicopter. A notable vision-based technique used in autonomous helicopter control, is the visual odometer [12], which provides accurate navigational information (position and velocity) which is combined with inertial measurements. An early approach to autonomous landing [8] decoupled the landing problem from vision-based tracking. In [13] a vision-based solution is given for safe-landing site detection in unstructured terrain where the key problem is for the onboard vision system to detect a suitable place to land, without the aid of a structured landmark such as a helipad.

In work closely related to ours, the UC Berkeley team has recently shown a real time computer vision system for tracking a landing target [14], [15] and have successfully coupled it with a helicopter controller to achieve landing [16].

While several techniques have been applied for vision-based control of helicopters, few have shown landing of an autonomous helicopter on a helipad. The problem of autonomous landing is particularly difficult because the inherent instability of the helicopter near the ground [17]. Also since the dynamics of a helicopter are non-linear, only an approximate model of the helicopter can be constructed [3].

Our approach, presented here differs from prior approaches in two ways. First, we impose no constraints on the design of the landing pad except that it should lie on a two dimensional plane. Hence we use moment descriptors to determine the location and orientation of the landing target. Second, our helicopter controller is model-free and behavior-based which provides a clean decoupling between the higher level tasks (e.g. target recognition, tracking and navigation) and the low level attitude controller. We validate our strategy in experimental flight trials where our algorithm is not only able to detect and land on a given target but is also able to track a target which moves intermittently, and land on it.

III. The Testbed and Experimental Task

Our experimental testbed, the AVATAR (Autonomous Vehicle Aerial Tracking And Reconnaissance) [18] is a gas-powered radio-controlled model helicopter fitted with a PC-104 stack augmented with sensors (Figure 1). A Novatel RT-20 DGPS system provides positional accuracy of 20 cm CEP (Circular Error Probable, i.e. the radius of a circle, centered at the true location of a receiver antenna, that contains 50% of the individual position measurements made using a particular navigational system). A Boeing CMIGTS-II INS unit with a three axis accelerometer and a three-axis gyroscope provides state information to the onboard computer. The helicopter is equipped with a downward-pointing color CCD camera and an ultrasonic sonar. The ground station is a laptop that is used to send high-level control commands and differential GPS corrections to the helicopter. Communication with the ground station is carried via 2.4 GHz wireless Ethernet and 1.8Ghz wireless video. Autonomous flight is achieved using a *behavior-based* control architecture [19], discussed further in Section V.

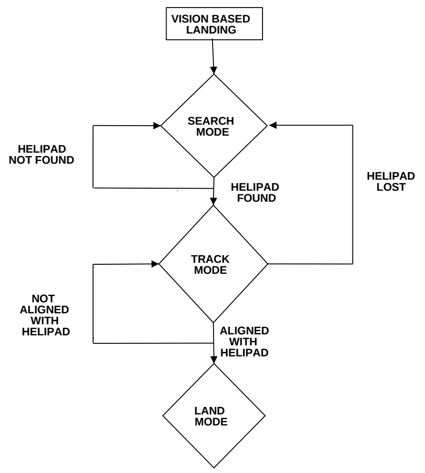

Fig. 2. The state transition diagram for landing.

The overall landing strategy is best described as a simple finite state machine (Figure 2) with three states[1] : *search*, *track*, and *land*. Initially the helicopter is in the *search* mode. The vision algorithm (Section IV) scans for the landing target. Once the landing target is detected, the system transitions to the *track* mode. In this mode, the state estimation algorithm

[1]We will call these states, modes, to avoid confusion with the conventional use of state in control theory to denote variables like the position and velocity

sends navigational commands to the helicopter controller. When the helicopter is aligned with the landing target the vision-based controller commands the helicopter to land and the system transitions to the *land* mode. If the target is lost when the helicopter is in *track* mode, the system transitions back to the *search* mode. Similarly, if alignment with the target is lost during the *land* mode, the system transitions back to the *track* mode. Next, we describe the vision and state estimation algorithms.

IV. Vision Algorithm

The vision algorithm is described below in three parts; preprocessing, geometric invariant extraction, object recognition and state estimation.

A. Assumptions

We make the following assumptions:

- The camera is perpendicular to the ground plane and is pointing downward.
- The vertical axis of the camera coincides with the principal axis of the helicopter.
- The intensity values of the helipad are different from that of the neighboring regions.

The first assumption affects the perceived shape of the landing pad. The second assumption affects the accuracy of landing. In reality the first two assumptions are often in conflict. In the case when the helicopter is a stable hover (nose pitched down), if the camera's vertical axis coincides with the principal axis of the helicopter, then necessarily the camera is not perpendicular to the ground. However the misalignment is small and in our assumptions, we ignore it.

B. Preprocessing

The goal of this stage is to locate and extract the landing target. Figure 3 (a) shows an aerial view of the helipad used in our experiments.

1) Thresholding and Filtering: Thresholding converts the color image to a binary image. The image obtained from the camera is noisy and the frame grabber is of low quality, hence we work with binary images to reduce the computational cost and increase the effectiveness of the algorithm. The image is first converted to grayscale by eliminating the hue and saturation information while retaining the luminance. This is accomplished by the following equation [20]

$$Y = 0.299 * R + 0.587 * G + 0.114 * B \tag{1}$$

where R,G,B represent the red, green and blue values in the image respectively.

The thresholding algorithm must produce a binary image which preserves the landing target but effectively removes most of the other data from the image. One of the important aspects of threshold selection is the capability to reliably identify the mode peaks in the image histogram [20]. This capability is of utmost importance for automatic thresholding where the image characteristics can change over a broad range of intensity distributions. Choosing a good threshold in an outdoor environment is difficult because the intensity values vary significantly depending on sunlight, orientation of the camera, heading of the helicopter etc. However, after various runs with offline images obtained from the camera, it was decided to threshold the image at a fixed threshold. We defer the implementation of a variable threshold strategy to future work.

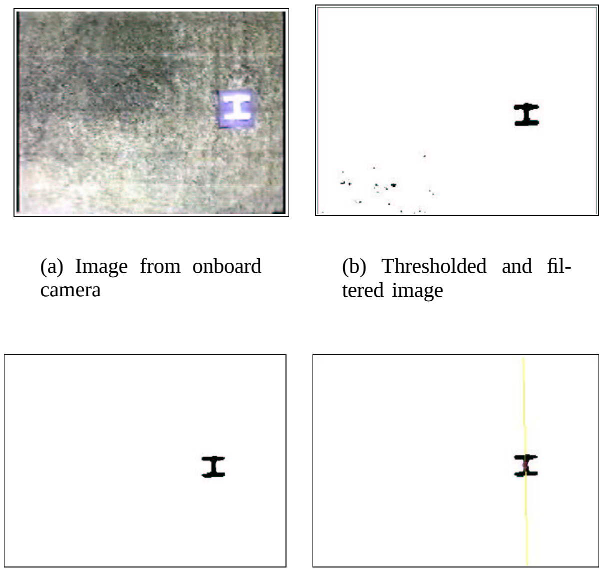
(a) Image from onboard camera
(b) Thresholded and filtered image
(c) Segmented image
(d) Final image

Fig. 3. Image processing results. Image (a) is captured in flight from the downward-pointing camera on the helicopter

A 7×7 *Median-filter* is applied to the thresholded image for removing noise and to preserve the edge details effectively. Median-filters have low-pass characteristics and they remove additive white noise [20]. They preserve the edge sharpness [21] in an image and are particularly suitable for the recognition of geometric objects such as the helipad. Figure 3(b) shows the image after thresholding and filtering.

2) Segmentation and Connected Component Labeling: The image obtained after thresholding and filtering may consist of objects other than the helipad. The objective of segmentation is to partition the image into regions. If $\Re$ represents the entire image region, then segmentation is the process which partitions $\Re$ into n sub-regions $\Re_1, \Re_2, \ldots, \Re_n$ such that

- $\Re_1 \cup \Re_2 \cup \ldots \Re_n = \Re$
- $\Re_i$ is a connected region for $i = 1, 2, \ldots, n$
- $\Re_i \cap \Re_j = \phi$ for all $i = 1, 2, \ldots, n$
- $f(\Re_i)$ = TRUE for all $i = 1, 2, \ldots, n$
- $f(\Re_i \cup \Re_j)$ = FALSE for all $i = 1, 2, \ldots, n$

where $f(\Re_i)$ is some classifier which classifies the region. We use the image intensity to define $f(\Re_i)$ as

$$f(\Re_i) = I(i,j) \;\; \forall I(i,j) \in \Re_i \;\; a \le I(i,j) \le b$$
$$0 \le a \le b \le 255 \tag{2}$$

where $I(i,j)$ represents the intensity value of a pixel at (i,j) and $[a,b]$ represents the range of intensity values belonging to a particular region.

The image is scanned row-wise until the first pixel at a boundary is hit. All the pixels which belong to the 8-neighborhood of the current pixel are marked as belonging to the current object. This operation is continued recursively until all pixels belonging to the object are counted. A product of this process is the area of the particular object in pixels. Objects whose area is less than a particular threshold area ($\leq$ 80 pixels) are discarded. Similarly objects whose area is $\geq$ 700 pixels are discarded. The remaining objects are the regions of interest and are candidates for the landing target (Figure 3 (c)).

C. Interlude: Invariant Moments

Geometric shapes possess features such as *perimeter, area, moments* that often carry sufficient information for the task of object recognition. Such features can be used as object descriptors, resulting in significant data compression, because they can represent the geometric shape by a relatively small feature vector. They are thus ideally suited for the present task. Based on the geometric features of an object one can calculate a set of descriptors which are invariant to rotation, translation and scaling. These shape descriptors are widely used in optical character recognition and pose estimation. One such class of descriptors [22] is based on the moments of inertia of an object. For a $2-\mathrm{D}$ continuous function $f(x,y)$ the moment is given by

$$m_{pq} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} x^p y^q f(x,y)dxdy \quad p,q = 0,1.. \tag{3}$$

where *(p+q)* represent the order of the moments. A uniqueness theorem [23] states that if $f(x,y)$ is piece-wise continuous and has non-zero values only in the finite part of the *xy* plane, moments of all orders exist and the moment sequence m_{pq} is uniquely determined by $f(x,y)$. Conversely, m_{pq} uniquely determines $f(x,y)$. Hence the moments of an object in an image can be used for object detection.

The $(p+q)^{th}$ order moment of an image $I(x,y)$, where $I(x,y)$ is the discrete (intensity) function is given by

$$m_{pq} = \sum_i \sum_j i^p j^q I(i,j) \tag{4}$$

where the indices *i, j* correspond to the coordinate axes *x, y* respectively.

The center of gravity of the object is specified by

$$\overline{x} = \frac{m_{10}}{m_{00}} \qquad \overline{y} = \frac{m_{01}}{m_{00}} \tag{5}$$

The central moments of an object are the moments defined about the center of gravity and are given by

$$\mu_{pq} = \sum_i \sum_j (i-\overline{x})^p (j-\overline{y})^q I(i,j) \tag{6}$$

where $p,q = 0,1,2...$ and μ_{pq} is the $(p+q)^{th}$ order central moment. The the indices *i, j* correspond to the coordinate axes *x, y* respectively. The normalized central moments, denoted by η_{pq}, are defined as

$$\eta_{pq} = \frac{\mu_{pq}}{\mu_{00}{}^{\gamma}} \tag{7}$$

where

$$\gamma = \frac{p+q}{2} + 1 \qquad for \quad p+q = 2,3,... \tag{8}$$

¿From the definition it can be shown that the central moments up-to third order are given by the following relations [24]

$$\mu_{00} = m_{00} = \mu \tag{9}$$

where μ is the area of the object in the discrete case.

$$\mu_{10} = \mu_{01} = 0 \tag{10}$$

$$\mu_{20} = m_{20} - \mu\overline{x}^2 \tag{11}$$

$$\mu_{11} = m_{11} - \mu\overline{x}\,\overline{y} \tag{12}$$

$$\mu_{02} = m_{02} - \mu\overline{y}^2 \tag{13}$$

$$\mu_{30} = m_{30} - 3m_{20}\overline{x} + 2\mu\overline{x}^3 \tag{14}$$

$$\mu_{21} = m_{21} - m_{20}\overline{y} - 2m_{11}\overline{x} + 2\mu\overline{x}^2\overline{y} \tag{15}$$

$$\mu_{12} = m_{12} - m_{02}\overline{x} - 2m_{11}\overline{y} + 2\mu\overline{x}\,\overline{y}^2 \tag{16}$$

$$\mu_{03} = m_{03} - 3m_{02}\overline{y} + 2\mu\overline{y}^3 \tag{17}$$

Normalized central moments can be employed to produce a set of invariant moments. The five lower-order invariants $\phi_1,....\phi_5$ are given in terms of the second and third order central moments [22] by

$$\phi_1 = \eta_{20} + \eta_{02} \tag{18}$$

$$\phi_2 = (\eta_{20} - \eta_{02})^2 + 4\eta_{11}^2 \tag{19}$$

$$\phi_3 = (\eta_{30} - 3\eta_{12})^2 + (3\eta_{21} - \eta_{03})^2 \tag{20}$$

$$\phi_4 = (\mu_{30} + \mu_{12})^2 + (\mu_{21} + \mu_{03})^2 \tag{21}$$

$$\begin{aligned}\phi_5 = (\mu_{30} - 3\mu_{12})(\mu_3 0 + \mu_{12})[(\mu_{30} + \mu_{12})^2 \\ -3(\mu_{21} + \mu_{03})^2] + (3\mu_{21} - \mu_{03})(\mu_{21} + \mu_{03}) \\ +[3(\mu_{30} + \mu_{12})^2 - (\mu_{21} + \mu_{03})^2]\end{aligned} \tag{22}$$

$\phi_1,...\phi_5$ are scale, rotation and translation invariant. Object eccentricity is given by

$$\varepsilon = [\frac{\mu_{02}\cos^2\theta + \mu_{20}\sin^2\theta - \mu_{11}\sin 2\theta}{\mu_{02}\sin^2\theta + \mu_{20}\cos^2\theta - \mu_{11}\cos 2\theta}]^2 \tag{23}$$

The eccentricity ε is invariant to rotation, scaling and translation. Since the eccentricity of an object can easily be computed from the normalized moments we use it for validating the helipad detection.

Object orientation is defined as the angle between the major axis of the object and the x-axis. It can be derived by minimizing the function

$$S(\theta) = \sum \sum_{(i,j)\in\Re} [(i - \bar{x})\cos\theta - (j - \bar{y})\sin\theta]^2 \qquad (24)$$

where (i, j) belong to $\Re$ (the space representing the image). Minimizing $S(\theta)$ gives the object orientation θ as

$$\theta = \frac{1}{2}\arctan(\frac{2\mu_{11}}{\mu_{20} - \mu_{02}}) \qquad (25)$$

D. Object Recognition and State Estimation

Initial trials with test data showed that the first, second and third moments of inertia were sufficient to distinguish between the landing target and other objects present in the image (Equations (18),(19),(20)). The algorithm was calibrated offline using a set of images collected in prior flights. The calibration values stored were the mean values of the moments of inertia. During actual flight the moments of inertia of each frame are calculated and compared to the calibration values. If they lie within a tolerance of $\pm 10\%$ of the stored values and the computed eccentricity is close to the correct value, then the object (in this case the helipad) is said to be recognized and the algorithm proceeds to the next step of state estimation.

The state estimation algorithm calculates the coordinates and orientation of the landing target relative to the helicopter. The heading is calculated using Equation 25, while the coordinates of the landing target are calculated using Equation 5.

The coordinates of the helipad so obtained are in the image frame. These coordinates are transformed into state estimates relative to the helicopter, based on the height of the helicopter above the ground. The height of the helicopter is obtained from the onboard differential GPS. The x-coordinate of the helipad in the helicopter frame of reference is given by

$$x_{heli} = \frac{height \times \tan\frac{\phi_h}{2} \times x_{image}}{\text{resolution of the camera along the x axis}} \qquad (26)$$

where ϕ_h is the field of view of the camera in the x direction, x_{image} is the x-coordinate of the helipad in the image plane. Similarly the y-coordinate of the helipad is given by

$$y_{heli} = \frac{height \times \tan\frac{\phi_v}{2} \times y_{image}}{\text{resolution of the camera along the y axis}} \qquad (27)$$

where ϕ_v is the field of view of the camera in the y direction, y_{image} is the y-coordinate of the helipad in the image plane. The x and y coordinates so obtained are converted into velocity commands by dividing them with an appropriate scaling factor. This scaling factor is inversely proportional to the height and is determined empirically. These state estimates are sent to the helicopter controller, described next.

V. Control Architecture

The AVATAR is controlled using a hierarchical behavior-based control architecture. Briefly, a behavior-based controller [25] partitions the control problem into a set of loosely coupled behaviors. Each behavior is responsible for a particular task. The behaviors act in parallel to achieve the overall goal. Low-level behaviors are responsible for robot functions requiring quick response while higher-level behaviors meet less time-critical needs. The behavior-based control architecture used for the AVATAR is shown in Figure 4. The low-level behaviors have been extensively described in previous work [19], we give a brief summary below and focus on the behaviors specific to the landing problem.

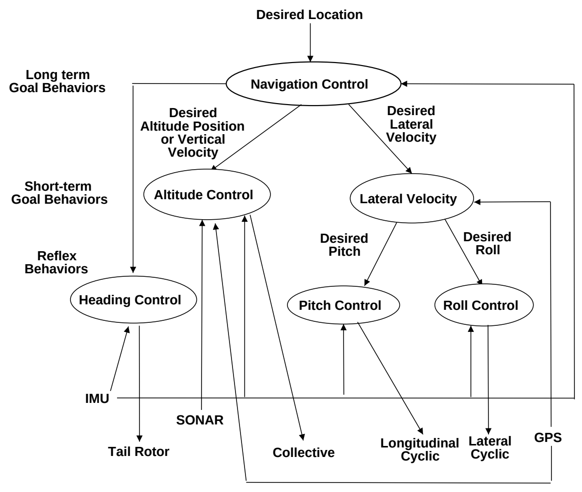

Fig. 4. AVATAR behavior-based controller

At the lowest level the robot has a set of reflex behaviors that maintain stability by holding the craft in hover. The *heading control* behavior attempts to hold the desired heading by using data from the IMU (Inertial Measurement Unit) to actuate the tail rotor. The *altitude control* behavior uses the sonar, GPS and IMU to control the collective and the throttle. The *pitch and roll control* behaviors maintain the desired roll and pitch angles received from the *lateral velocity* behavior. The *lateral velocity* behavior generates desired pitch and roll values that are given to the *pitch and roll control* behaviors to achieve a desired lateral velocity. At the top level the *navigation control* behavior inputs a desired heading to the *heading control*, a desired altitude or vertical velocity to the *altitude control* and a desired lateral velocity to the *lateral control* behavior. A key advantage of such a control algorithm is the ability to build complex behaviors on top of the existing low level behaviors, without changing them.

The low-level and short-term goal behaviors *roll, pitch, heading, altitude and lateral control* behaviors are implemented with proportional controllers (The *altitude control* behavior is implemented as a proportional-plus-integral(PI) controller). For example the *roll control* behavior reads in the current roll angle from the IMU and outputs a lateral cyclic command to the helicopter. This is shown in Equation 28 where τ is the servo command, θ is the roll angle and θ_d is the desired roll angle.

$$\tau = K_p(\theta_d - \theta) \tag{28}$$

The long-term goal behavior *navigation control* is responsible for overall task planning and execution. If the heading error is small, the *navigation control* behavior gives desired lateral velocities to the *lateral velocity* behavior. If the heading error is large, the *heading control* behavior is commanded to align the helicopter with the goal while maintaining zero lateral velocity.

The *altitude control* behavior is further split into three sub-behaviors, *hover control, velocity control and sonar control*. The *hover control* sub-behavior is activated when the helicopter is either flying to a goal or is hovering over the target. This sub-behavior is used during the object recognition and object tracking state when the helicopter should move laterally at a constant altitude. The hover controller is implemented as a proportional controller. It reads the desired GPS location and the current location and calculates the collective command to the helicopter. This is shown in Equation 29 where τ is the collective command sent to the helicopter servos, $g(\theta_{lat}, \theta_{lon})$ is a function of the current latitude and longitude $g(\theta_{dlat}, \theta_{dlon})$ is a function of the desired latitude and the longitude, K_p is the proportional gain. The function g converts a given latitude and longitude to the corresponding distance in meters from a pre-surveyed point at the experimental site.

$$\tau = K_p(g(\theta_{dlat}, \theta_{dlon}) - g(\theta_{lat}, \theta_{lon})) \tag{29}$$

Once the helipad has been located and the helicopter is aligned with the helipad the *velocity control* sub-behavior takes over from the *hover control* sub-behavior. It is implemented as a PI controller. The integral term is added to reduce the steady state error. The helicopter starts to descend till reliable values are obtained from the sonar. The *sonar control* sub-behavior takes over at this point until touchdown. This is also implemented as a PI controller. The *velocity control* sub-behavior is shown in Equation 30 where τ is the collective command sent to the helicopter servos, v is the current velocity v_d is the desired velocity, K_p is the proportional gain and K_i is the integral gain.

$$\tau = K_p(v_d - v) + K_i \int (v_d - v)dt \tag{30}$$

The *sonar control* sub-behavior is shown in Equation 31, where τ is the collective command to the helicopter servos, x is the current position, x_d is the desired position, K_p is the proportional gain and K_i is the integral gain.

$$\tau = K_p(x_d - x) + K_i \int (x_d - x)dt \tag{31}$$

The values K_p, K_i represent the gains associated with the PI controller. Currently these values are obtained empirically during flight tests. We plan to obtain these values analytically and tune them in the future.

VI. Experimental Results and Discussion

A total of fourteen landings were performed on two different days in different weather conditions to validate the algorithm. Out of the fourteen landings, nine were vision-based autonomous landings where the helipad was stationary and the helicopter had to autonomously detect it, align with it and land on it. Two landings consisted of a transitional stage where the helipad was intermittently hidden to test the robustness of the algorithm. In the remaining three landings the helicopter was required to track a moving helipad and land on it once it stopped. The following subsections describe the experimental results in detail.

A. Autonomous Landing

The helicopter is initially commanded to autonomously fly toward the helipad based on GPS [*search mode*]. Once the helipad is in view, the controller switches to vision - based control [*track mode*] [Figure 2]. If for any reason the helicopter loses sight of the helipad, the controller switches back to *search mode*. Once in *track mode* the low-level control behaviors on the helicopter receive commands from the vision controller. The vision system sends orientation offset, velocity forward and velocity right commands with respect to the image coordinate frame to the helicopter controller. The commands are then converted into velocity-north and velocity-east commands based on the current GPS and heading. The navigational control behavior takes these lateral velocity and heading commands and sends the appropriate commands to the low-level behaviors for the control of the helicopter.

TABLE I
Data from Flight Tests

Trial	Total flight time	Landing time	$\delta\theta$
1	306 s	108 s	5°
2	156 s	63 s	15°
3	316 s	112 s	0°
4	308 s	106 s	3°
5	178 s	62 s	10°
6	186 s	87 s	10°
7	194 s	66 s	2°
8	156 s	77 s	10°
9	203 s	93 s	4°

When the helicopter is oriented with the helipad it starts descending [*land mode*]. At this juncture the helicopter is controlled by the *velocity control* sub-behavior. If it descends to a height of 2 meters or less the *sonar control* is activated. From this point onwards the helicopter's altitude is regulated by sonar, till it lands.

A total of nine test flights were conducted for testing autonomous landing only. The data obtained are shown in Table I. The final average orientation error ($\delta\theta$) is approximately 7°. The computational cost for image processing is shown in Table II. The time taken for computing the moments of inertia is only 10% of the total time. We conclude that if the landing target has a well defined shape, the vision algorithm is computationally inexpensive. It may be noted that our frame grabber was offboard, thus the captured images were contaminated with wireless transmission artifacts, artificially

TABLE II
COMPUTATIONAL COST FOR IMAGE PROCESSING AT 10 FRAMES PER SECOND

Image Processing	*CPU time*
Image Acquisition	$\approx 20\%$
Thresholding and Filtering	$\approx 12\%$
Segmentation	$\approx 40\%$
Component Labeling	$\approx 7\%$
Hu's Moments of Inertia	$\approx 10\%$
GUI and displaying images	$\leq 11\%$

increasing the difficulty of the problem. Further, because of the limited bandwidth of the wireless video transmitter we were able to process only 10 frames per second.

TABLE III
ERRORS IN THE OBJECT RECOGNITION ALGORITHM

Total No of Frames	22060
Helipad present	16094
Helipad detected	15632

Table III shows the accuracy of the algorithm used. The data were obtained from approximately 22000 frames during the nine flight trials. Each flight was of a duration of approximately 3 minutes. Out of a total number of 22060 frames processed, the landing pad was present in 16094 frames while it was detected in 15632 frames. The algorithm failed to detect the helipad in 462 out of 16094 frames - an error rate of 2.87%. The moments of inertia are invariant to rotation, scaling and translation but vary when the plane in which the image lies is continually changing. The helicopter pitches and rolls in flight, which changes the relative orientation of the image plane; this distorts the image which results in errors. In the future we plan to integrate measurements from the IMU with the vision controller to nullify the effects caused by the roll and pitch motion.

TABLE IV
AVERAGE RESULTS FROM FLIGHT TESTS

Mean time to land	81 s
Mean autonomous flight time	198 s
Mean error in orientation	6^o
Standard Deviation in orientation	5^o
Mean error in position	42 cm

Table IV shows the results averaged over the nine flight trials. We were able to control the heading of the helicopter remarkably well. During the landing phase, the downward velocity is always restricted to a maximum of 0.2 m/s (see Figure 5 (d)). This was implemented for a smooth descent trajectory, as well as for safety purposes. The trajectory of the craft during descent for a representative trial is shown in Figure 5 (c). Although initially there are some variations in height, the helicopter descends smoothly during the later part. For the helicopter to finally land it has to overcome ground effect and turbulence. So when the helicopter is 1m from the ground we ramp down the collective servo to force a decrease in the pitch of the main blades, thereby reducing the upward thrust, hence the downward velocity ramps up to a maximum. This can be seen in Figure 5 (d), when the downward velocity reaches 0.6 meters/second. The difference between the orientation of the helipad and the helicopter for a representative trial is shown in Figure 5 (a). The controller is able to maintain the orientation of the craft aligned with the helipad.

The average position error after landing during these nine flights was 42 cm from the center of the helipad. This value is calculated as the distance from the center of the helipad to the center of the helicopter after landing. This error is small when compared to the size of the helipad (120 x 120 cm) and the helicopter(150 x 180cm). Presently the camera is statically mounted below the undercarriage of the helicopter pointing down. Depending on the height of the craft even a small inclination in the craft causes a large change in the horizontal distance on the ground, making it difficult to track the landing target precisely. Mounting the camera on a gimbal would solve the problem. Also precise control of the helicopter near the ground is difficult because of the air-cushion developed by the downward thrust from the main rotor of the helicopter.

B. Autonomous Landing with Helipad Momentarily Hidden

Two trials were performed where the helipad was momentarily hidden when the helicopter was in *track* mode. The helicopter successfully went into *search* mode, and when the helipad was visible again, it was able to successfully track the helipad, orient appropriately and land on it. During the time the helipad is lost the helicopter maintains a near constant altitude. The average position error after landing during these two flights was 41 cm from the center of the helipad. This value is calculated as the distance from the center of the helipad to the center of the helicopter after landing. The average error in orientation was 8°. This is the difference between the orientation of the helicopter after it has landed, and the principal axis of the helipad. These values are similar to the values obtained in the landing experiments described previously, where the helipad was continuously visible.

C. Autonomous Landing with Helipad in Motion

Three flight trials were performed with the helipad in motion. As soon as the helipad was in the field of view of the camera it was manually moved, so that the helicopter could track it. Figures 6 (a)-(d) depict the results. The initial location and the final location of the helipad as well as the trajectory of the helicopter are shown in Figure 6 (c) (In the figure the circle represents the initial position of the helipad and the diamond represents the final position). The helicopter maintained a constant orientation and height while it was tracking the landing pad as shown in Figures 6 (a) and (b). Table V shows the error in the orientation of the helicopter with respect to the helipad, distance of the center of the camera from the center of the helipad and the distance moved by the helipad after the helicopter has landed. The helipad was not in motion from the time the helicopter started landing. Surprisingly, these values are much better than the values obtained in the above landing experiments. This can be

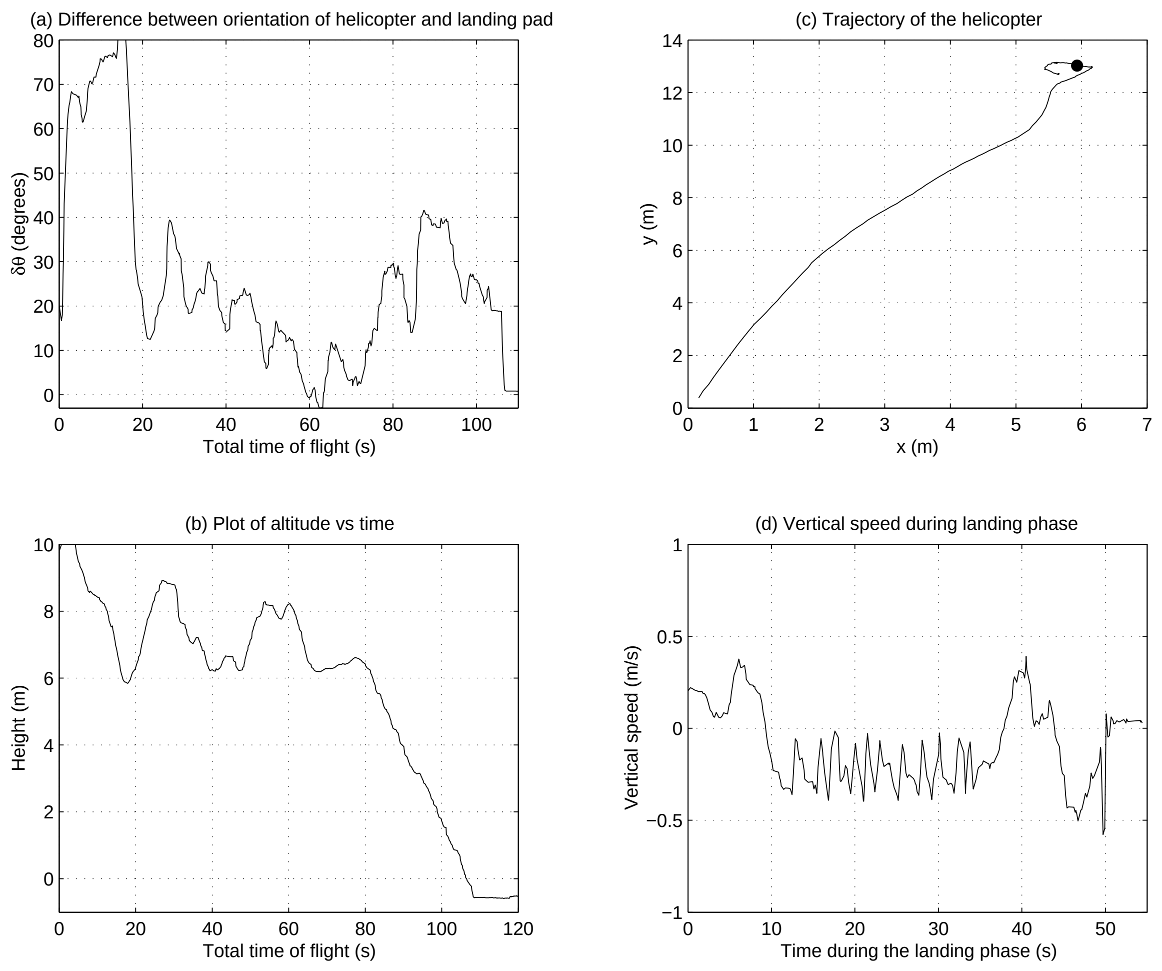

Fig. 5. Performance of the vision algorithm in conjunction with the landing controller for a stationary helipad.

attributed to the fact that the initial set of experiments (with a stationary helipad) were performed on a very windy day while there were near-perfect weather conditions during the trials with the helipad in motion.

TABLE V
DATA OBTAINED WITH MOVING HELIPAD

Trial	δd	$\delta\theta$	Distance moved
1	25 cm	10°	838 cm
2	12.5 cm	4°	713 cm
3	50 cm	2°	487 cm

D. *Discussion of Results*

An initial set of seven landings were performed on day one to validate the vision and the control algorithm. During these flights the helicopter had to find a helipad, orient and land on it.The average position error after landing during these seven flights was 40 cm from the center of the helipad and the average orientation error was 7°. To test the algorithm two additional landings were performed after a week in completely different conditions. Since we use an intensity-based descriptor, we wanted to test the robustness of the algorithm to external light conditions.The average position error after landing during these two flights was 42 cm from the center of the helipad. The average orientation error was 7°. The results obtained were similar to the ones obtained earlier. This provides partial validation for using moments of inertia as a shape descriptor for the helipad in diverse environmental conditions. Two flights were performed on day two to test whether the control algorithm could fall back to *search mode* i.e, to search for the helipad once it was not visible, and also to test how coherently the vision and control algorithms performed in tandem. The average position error after landing during these two flights was 41 cm from the center of the helipad. The average error in orientation was 8°. The results obtained were satisfactory and were comparable to the previous results.

Finally, on day two we tested (in three flight trials) how well the algorithm could track a moving helipad and land on it. We were able to consistently track and land on the helipad with an average error of 5° in orientation and an average error of 28 cm in distance. This result was much better than the values obtained in previous trials where the average error was approximately 7° in orientation and 40cm in distance. This could be attributed to the fact that on day two there was little

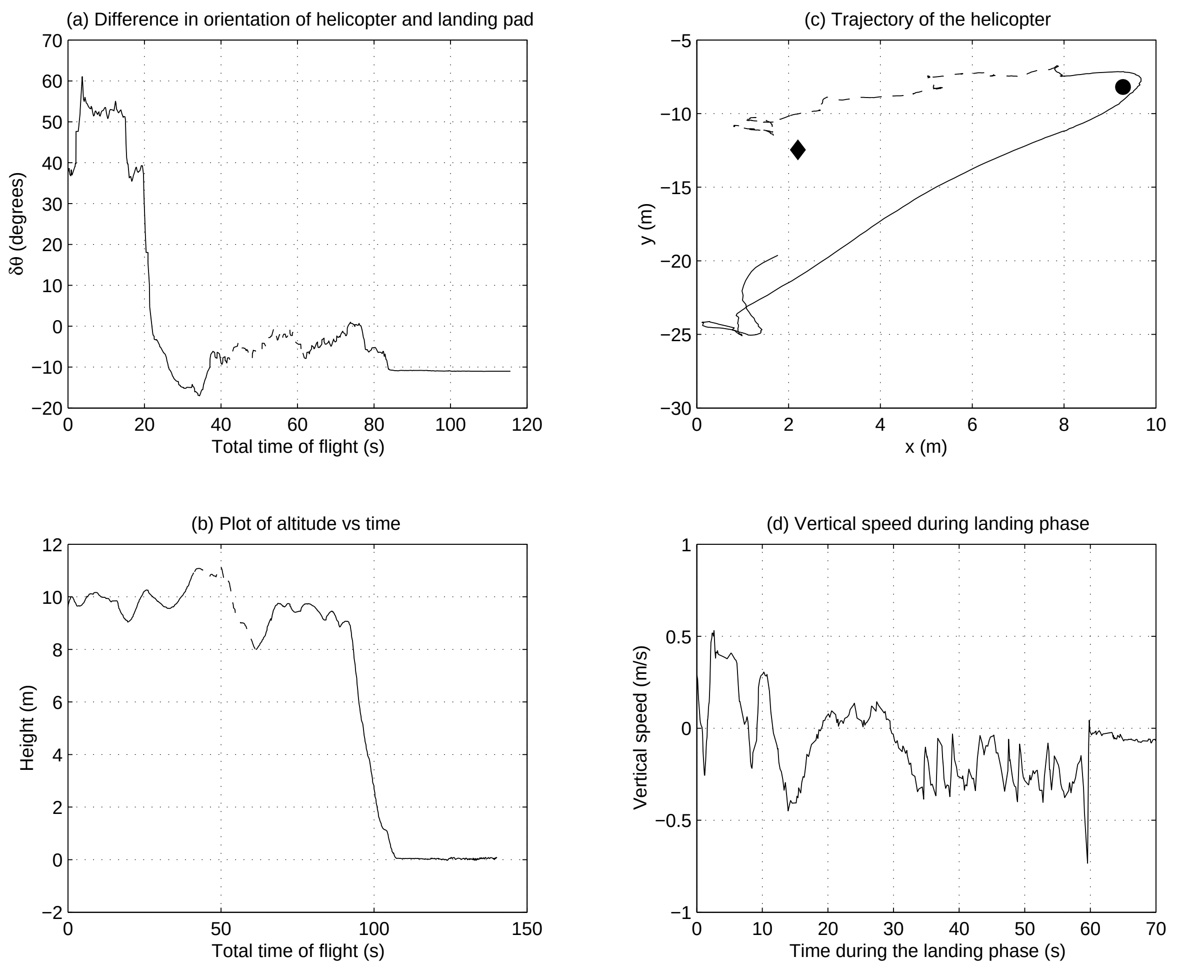

Fig. 6. Vision algorithm in conjunction with the landing controller when the helipad is moved.

or almost no wind and near-perfect flight conditions. [2]

VII. Conclusion and Future Work

We have presented the design and implementation of a real-time vision-based system for detecting a landing target (stationary or in intermittent motion) and a controller to autonomously land a helicopter on the target. The vision algorithm is fast, robust and computationally inexpensive. It relies on the assumptions that the landing target has a well-defined geometric shape, and all the feature points of the landing target are coplanar. Since we chose a landing target composed of polygons and the helicopter keeps the camera roughly perpendicular to the ground, these two assumptions were justified.

Data from fourteen flight trials show that our algorithm and landing strategy works accurately and repeatably even when the landing target is in motion or is temporarily hidden. The helicopter achieved autonomous landing to within 40 cm positional accuracy and 7° orientation accuracy measured relative to the helipad. In the future we plan to integrate measurements from the IMU with the algorithm described here to nullify the effects caused by the roll and pitch motion thereby improving the detection of the landing target. At present the gains used in the behavior-based controller for the helicopter are manually determined. In the future, we plan to determine these gains based on system identification of the dynamics of the helicopter. Also, we plan to mount the camera on a gimbal to nullify the effects caused by changes in the inclination of the helicopter, so that we are able to track the helipad more accurately.

In the future we plan to focus our attention on the problem of safe and precise landing of the helicopter in unstructured harsh 3D environments. The applications of such a system are enormous; from space exploration to target tracking and acquisition.

[2] In Figures 5 and 6 the total time of flight in each of the subfigures is different since the plots are from different experimental trials

Acknowledgments

The authors gratefully acknowledge the following individuals for their support: Gerald Bohne, Kale Harbick, Stefan Hrabar, Boyoon Jung, Sanjeev Koppal, David Naffin, Sajid Siddiqi and Doug Wilson. This work is supported in part by NASA under JPL/Caltech contract 1231521, by DARPA grant DABT63-99-1-0015 as part of the Mobile Autonomous Robotic Software (MARS) program, and by ONR grant N00014-00-1-0638 under the DURIP program.

REFERENCES

[1] Office of the Under Secretary of Defense, "Unmanned aerial vehicles annual report," Defense Airborne Reconnaissance Office, Pentagon,Washington DC, Tech. Rep., July 1998.

[2] F. Dellart, S. Seitz, C. Thorpe, and S. Thrun, "Structure from motion without correspondence," Robotics Institute, Carnegie Mellon University, Tech. Rep. CMU-RI-TR-99-44, December 1999.

[3] A. R. Conway, "Autonomous control of an unstable helicopter using carrier phase gps only," Ph.D. dissertation, Stanford University, March 1995.

[4] M. Bosse, "A vision augmented navigation system for an autonomous helicopter," Master's thesis, Boston University, 1997.

[5] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Trans ASME*, vol. 82, no. D, pp. 35–45, March 1960.

[6] M. Jun, S. I. Roumeliotis, and G. S. Sukhatme, "State estimation of an autonomous flying helicopter," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, October 1999, pp. 1346–1353.

[7] O. Amidi, "An autonomous vision-guided helicopter," Ph.D. dissertation, Robotics Institute, Carnegie Mellon University, 1996.

[8] R. Miller, B. Mettler, and O. Amidi, "Carnegie mellon university's 1997 international aerial robotics competition entry," in *International Aerial Robotics Competition*, 1997.

[9] Y. Ma, J. Kosecka, and S. S. Sastry, "Vision guided navigation for a nonholonomic mobile robot," *IEEE Transactions on Robotics and Automation*, vol. 15, no. 3, pp. 521–537, June 1999.

[10] A. Lazanas and J. C. Latombe, "Landmark-based robot navigation," in *proceedings of the Tenth National Conference on Artificial INtelligence*, 1992, pp. 816–822.

[11] S. Hutchinson, G. D. Hager, and P. I. Corke, "A tutorial on visual servo control," in *IEEE Transaction on Robotics and Automation*, vol. 12(5), October 1996, pp. 651–670.

[12] O. Amidi, T. Kanade, and K. Fujita, "A visual odometer for autonomous helicopter flight," *Robotics and Autonomous Systems*, vol. 28, no. 185-193, 1999.

[13] P. J. Garcia-Padro, G. S.Sukhatme, and J. F. Montgomery, "Towards vision-based safe landing for an autonomous helicopter," *Robotics and Autonomous Systems*, vol. 38, no. 1, pp. 19–29, 2001.

[14] C. S. Sharp, O. Shakernia, and S. Sastry, "A vision system for landing an unmanned aerial vehicle," in *proceedings of IEEE International Conference on Robotics and Automation*, 2001, pp. 1720–1728.

[15] B. Sinopoli, M. Micheli, G. Donato, T. J. Koo, and S. S. Sastry, "Vision based navigation for an unmanned aerial vehicle," in *proceedings of IEEE International Conference on Robotics and Automation*, 2001, pp. 1757–1765.

[16] O. Shakernia, R. Vidal, C. S. Sharp, Y. Ma, and S. S. Sastry, "Multiple view motion estimation and control for landing an unmanned aerial vehicle," in *proceedings of IEEE International Conference on Robotics and Automation*, May 2002, pp. 2793–2798.

[17] O. Shakernia, Y. Ma, T. J. Koo, and S. S. Sastry, "Landing an unmanned air vehicle:vision based motion estimation and non-linear control," in *Asian Journal of Control*, vol. 1, September 1999, pp. 128–145.

[18] Univeristy of Southern California Autonomous Flying Vehicle Homepage, "http://robotics.usc.edu/˜avatar."

[19] J. F. Montgomery, "Learning helicopter control through 'teaching by showing'," Ph.D. dissertation, University of Southern california, May 1999.

[20] R. Gonzalez and R. Woods, *Digital Image Processing*. Addison-Wesley, 1992.

[21] I. Pitas, *Digital Image Processing Algorithms*. Prentice-Hall, 1993.

[22] M. K. Hu, "Visual pattern recognition by moment invariants," in *IRE Transactions on Information Theory*, vol. IT-8, 1962, pp. 179–187.

[23] A. Papoulis, *Probability, Random Variables, and Stochastic Processes*. McGraw-Hill, New York, 1965.

[24] M. D. Levine, *Vision in man and machine*. McGraw-Hill, 1985.

[25] M. J. Mataric, "Behavior-based control: Examples from navigation, learning and group behavior," *Journal of Experimental and Theoretical Artificial Intelligence, special issue on Software Architecture for Physical Agents*, vol. 9, no. 2-3, pp. 67–83, 1997.

Srikanth Saripalli Srikanth Saripalli is a Ph.D. student in the Computer Science department at University of Southern California. His research interests include control of unmanned aerial vehicles, vision based control, visual servoing and control of complex hybrid systems. Srikanth received his B.Tech in mechanical engineering from Birla Institute of Technology and Sciences, India in 2000 and his MS in computer science from University of Southern California, Los Angeles in 2003.

James F. Montgomery James F. Montgomery is a Senior Member of Technical Staff at the Jet Propulsion Laboratory in Pasadena, CA. His general research interest is in the field of autonomous robotics with special interest in autonomous aerial robots, including helicopters and blimps. He is currently leading a task that uses an autonomous helicopter as a platform for developing machine vision based techniques for autonomous safe landing in hazardous terrain. He is also a member of a team that is developing autonomy technologies for an aerobot meant for future exploration of Titan, a moon of Saturn. James received his BS in computer science from the University of Michigan, Ann Arbor in 1986 and his MS and PhD in computer science from the University of Southern California, Los Angeles in 1992 and 1999, respectively. He can be reached at monty@robotics.jpl.nasa.gov

Gaurav S. Sukhatme Dr. Gaurav Sukhatme is an Assistant Professor in the Computer Science Department at the University of Southern California (USC) and the co-director of the Robotics Research Laboratory within the Center for Robotics and Embedded Systems. His research interests include distributed mobile robotics, embedded systems, and sensor networks. Dr. Sukhatme has served as PI or CoPI on several NSF, DARPA and NASA grants and contracts. His directs the Robotic Embedded Systems Lab, which performs research in two related areas: 1) The control and coordination of large numbers of distributed embedded systems (devices, sensors and mobile robots), and 2) The control of systems with complex dynamics (hopping robots, robotic helicopters and haptic interfaces). Dr. Sukhatme is a 2002 NSF CAREER awardee, a member of AAAI, IEEE and the ACM and has served on several conference program committees. He has published over 60 technical papers, a book chapter and workshop papers.