An Experimental Study of the Autonomous Helicopter Landing Problem

Srikanth Saripalli[1], Gaurav S. Sukhatme[1], and James F. Montgomery[2]

[1] Department of Computer Science
University of Southern California
Los Angeles, California, USA
srik,gaurav@pollux.usc.edu
[2] Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California, USA

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 on board 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.

1 Introduction

Image-based information is useful for a variety of autonomous vehicle applications such as obstacle avoidance, map generation, target tracking and motion estimation. Cameras are inexpensive and operate at a high temporal rate. This, coupled with advances in processing power, makes image-based techniques invaluable in autonomous systems operation.

In this paper **we propose and experimentally investigate a vision-based technique for autonomously landing a robotic helicopter.** We model the solution to the landing problem discretely using a finite state machine, responsible for detecting the landing site, navigating toward it, and landing on it. Data from a single on-board camera are combined with attitude and position measurements from an on-board inertial navigation unit. These are the inputs to the on-board control system: a set of controllers running in parallel which are responsible for controlling the individual degrees of freedom of the helicopter. The resulting hybrid control system is simple, yet effective as shown experimentally by trials in nominal and perturbed conditions.

We experimentally test our algorithm by initializing the helicopter in hover at an arbitrary location. The helicopter is required to autonomously locate a helipad (imprinted with a known visual landmark), align with it and land on it. Results from experiments show that our method is able to land the helicopter on the helipad repeatably and accurately. On an average the helicopter landed to within 35 cm

B. Siciliano and P. Dario (Eds.): Experimental Robotics VIII, STAR 5, pp. 466–475, 2003

position accuracy and to within 7^o in orientation as measured from the center of the helipad and its principal axis respectively. In this paper we focus on experimental evidence showing the robustness of the algorithm. In particular we show that 1. the helicopter is able to visually re-acquire the helipad after losing it momentarily, and 2. the helicopter is capable of tracking a moving helipad and landing on it, once the helipad has stopped. In the tracking experiments the helipad was moved a significant distance (7 m on an average). Importantly the same algorithm is used across these conditions - no specific modifications were made to handle the various cases.

2 Assumptions and Related Work

Our approach 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 and should be distinguishable from the surrounding environment (i.e. it should have a different intensity). Second, our helicopter [1] 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 [2].

Several techniques have been applied for vision-based control of helicopters. The visual odometer [3] can visually lock-on to ground objects and sense relative helicopter position in real time, but does not integrate vision-based sensing with autonomous landing. The problem of autonomous landing is particularly difficult because the inherent instability of the helicopter near the ground [4]. In work closely related to ours [7], a vision-based approach to landing is presented which uses a landing pad of known shape to simplify the vision problem. In previous work [6], we have demonstrated vision-based landing site detection and selection without the use of structured landmarks. In [5] a biologically inspired approach to landing based on optical flow is being studied.

We use invariant moment descriptors for landing pad detection. These descriptors are invariant to rotation, translation and scaling [8], thus we do not impose any restriction on the shape of the landing pad except that it be planar. We implement our controller as a behavior-based system in which higher level behaviors can be layered on top of low level behaviors, without changing them. The vision algorithm for landing pad detection is layered on top of the helicopter controller. This modular approach allows us to implement complex algorithms without changing the underlying structure. We assume the following:

- 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 landing pad are different from that of the neighboring regions.

The first assumption determines the shape of the landing pad. The second assumption determines the accuracy of landing. We assume that the landing pad has

different intensity than the surrounding regions and base our algorithm on it. In practice 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.

3 Approach

3.1 Finite State Model

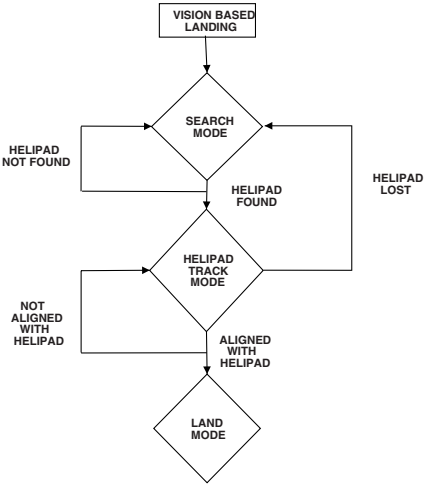

Fig. 1. The state transition diagram for the landing task

The overall landing strategy is best described as a simple finite state machine with three states[1] : *search, track,* and *land*. Initially the helicopter is in the *search* mode. The vision algorithm (described below) 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 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 (Figure 1).

[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

3.2 Vision-Based Helipad Detection

The vision algorithm consists of preprocessing, geometric invariant extraction, object recognition and state estimation.

Preprocessing: The preprocessing stage consists of thresholding, filtering, using a median filter and segmentation. The thresholding algorithm must produce a binary image which preserves the landing target but effectively removes most of the other data from the image. A robust implementation is to threshold the image at a fixed percentage (80%) between the minimum and the maximum gray levels. A 7×7 *Median-filter* is applied to the subsequent image for removing noise and to preserve the edge details effectively. Median-filters have low-pass characteristics and they remove additive white noise [8]. They preserve the edge sharpness in an image and are particularly suitable for the recognition of geometric objects such as the helipad.

The image obtained after thresholding and filtering may consist of objects other than the helipad. In the segmentation step the various regions of interest are identified and labeled. 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 (≤ 80 pixels) are discarded. Similarly objects whose area is ≥ 700 pixels are discarded. The remaining objects are our ROI (regions of interest) and are candidates for the landing target.

Invariant Moments: Geometric shapes possess features such as *perimeter, area, moments* that 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 and are ideally suited for the present task. Based of 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 [9] is based on the moments of inertia of an object.

The geometric invariant extraction stage calculates the Hu's moments of inertia [9] for the segmented regions. The algorithm is initially calibrated offline using a set of images collected in prior flights. 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 1a,1b,1c).

$$\phi_1 = \eta_{20} + \eta_{02} \tag{1a}$$

$$\phi_2 = (\eta_{20} - \eta_{02})^2 + 4\eta_{11}^2 \tag{1b}$$

$$\phi_3 = (\eta_{30} - 3\eta_{12})^2 + (3\eta_{21} - \eta_{03})^2 \tag{1c}$$

where $\phi_1, ...\phi_3$ are the three lower order invariant moments, η_{pq} is the normalized central moment, defined as

$$\eta_{pq} = \frac{\mu_{pq}}{{\mu_{00}}^{\gamma}} \tag{2}$$

where $\gamma = \frac{p+q}{2} + 1$ for $p + q = 2, 3, \ldots$ and μ_{pq} represents the central moment of an object about the center of gravity and is given by

$$\mu_{pq} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} (x - \bar{x})^p (y - \bar{y})^q f(x, y) dx dy \tag{3}$$

$p, q = 0, 1, 2..., f(x, y)$ represents a two-dimensional object as a continuous function, $(\bar{x}, \bar{y})$ represents the center of gravity of the object and μ_{pq} is the $(p + q)^{th}$ order central moment. The orientation of the helipad with respect to the helicopter is derived by minimizing the function

$$S(\theta) = \sum \sum_{(i,j) \in \Re} [(i - \bar{x}) \cos\theta - (j - \bar{y}) \sin\theta]^2 \tag{4}$$

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

$$\theta = \frac{1}{2} \arctan\left(\frac{2\mu_{11}}{\mu_{20} - \mu_{02}}\right) \tag{5}$$

Object Recognition: The calibration values stored were the mean values of the moments of inertia. During actual flight the moments of inertia of the segmented regions for each frame are calculated and compared to the calibration values. If they lie within a tolerance of $\pm 10\%$ of the stored values then the object (in this case the helipad) is said to be recognized and the algorithm proceeds to the next step of state estimation.

State Estimation: The coordinates of the center of gravity of the helipad are given by

$$\bar{x} = \frac{m_{10}}{m_{00}} \qquad \bar{y} = \frac{m_{01}}{m_{00}} \tag{6}$$

where m_{pq} is the $(p + q)^{th}$ order moment of an image $f(x, y)$ given by

$$m_{pq} = \sum_i \sum_j i^p j^q f(i, j) \tag{7}$$

where the indices *i, j* correspond to the coordinate axes *x, y* respectively and $f(i, j)$ is the intensity of the image at point (x, y).

The state estimation algorithm calculates the coordinates and orientation of the landing target relative to the helicopter using Equations 5 and 6. These state estimates are sent to the low-level helicopter controller.

Fig. 2. A typical autonomous landing maneuver

3.3 Control Strategy

The helicopter is controlled using a hierarchical behavior-based control architecture [2]. Briefly, a behavior-based controller [10] 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 is shown in Figure 3. 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, initial-descent and final-descent*. The *hover* 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 is required to 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

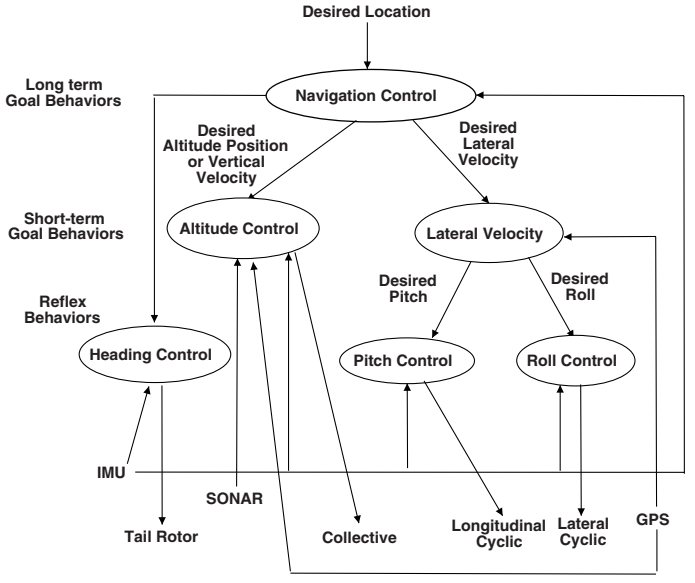

Fig. 3. Behavior-Based Controller

to the helicopter. This is shown in Equation 8 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 surveyed point.

$$\tau = K_p(g(\theta_{dlat}, \theta_{dlon}) - g(\theta_{lat}, \theta_{lon})) \tag{8}$$

Once the helipad has been located and the helicopter is aligned with it the *initial-descent* sub-behavior takes over from the *hover* sub-behavior. This phase of descent is implemented as a velocity-based PI controller, since the measured height from the GPS is contaminated with significant noise, and cannot be reliably used for height regulation. The helicopter starts to descend till reliable values are obtained from a downward-looking sonar sensor. The *final-descent* sub-behavior takes over at this point until touchdown. The *initial-descent* sub-behavior is shown in Equation 9 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.

$$\tau = K_p(v_d - v) + K_i \int (v_d - v)dt \tag{9}$$

The *final-descent* sub-behavior is shown in Equation 10, 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.

$$\tau = K_p(x_d - x) + K_i \int (x_d - x)dt \tag{10}$$

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

4 Experimental Results

A total of fourteen landings were performed to validate the algorithm. We have previously [11] presented a detailed analysis of the results obtained in the first nine landings, which dealt with the nominal case, i.e. the helipad was stationary and always visible to the helicopter. In this paper we consider the case of tracking a moving helipad, and then landing on it. Out of the five test flights, two trials 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.

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. Figure 4(a) shows the helicopter in search mode trying to find the helipad. When the helipad is in view again the helicopter is able to track the helipad 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.

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. Figures 4(b)-(d) depict the results. The initial location and the final location of the helipad as well as the trajectory of the helicopter is shown in Figure 4(b). The helicopter maintained a (nearly) constant orientation and height while it was tracking the landing pad as shown in Figures 4(c),(d). The average position error after landing during these two flights was 31 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 6°. The helipad was not in motion from the time the helicopter started final descent.

5 Conclusion

We have presented the design and implementation of a real-time vision-based algorithm for autonomously landing a robotic helicopter. We adopt a fast, robust and

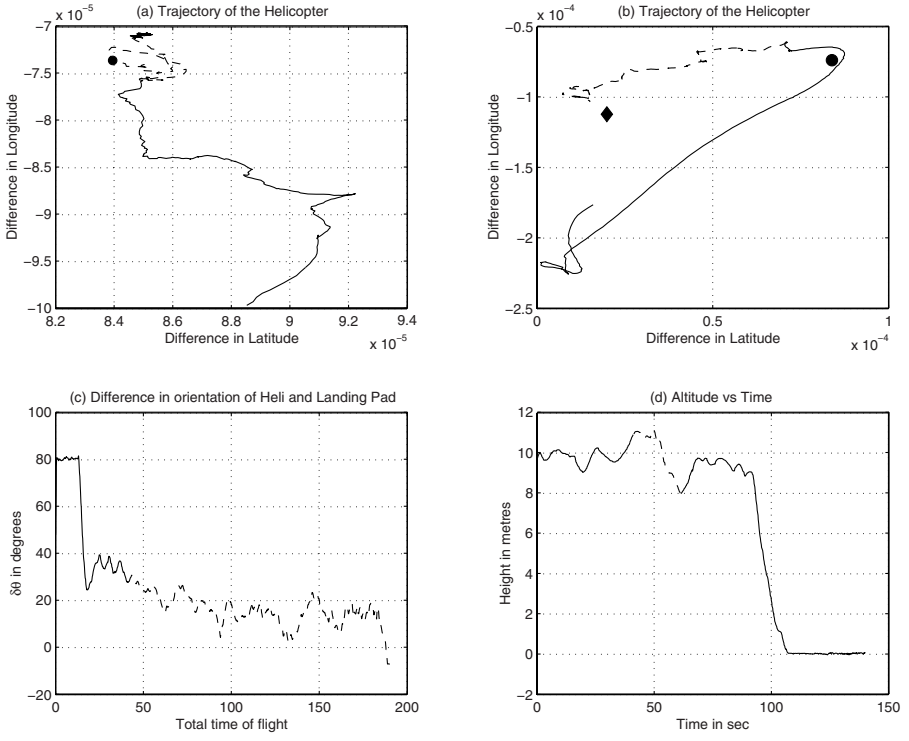

Fig. 4. Vision algorithm in conjunction with the landing controller (In (a) the circle represents the position of the helipad. The graph represents the trajectory of the helicopter landing on a stationary helipad. In (b) the circle represents the initial position of the helicopter and the diamond represents the final position. The $x-y$ axis of the graphs show the distance traversed by the helicopter from its initial location to the final location as a function of latitude and longitude. (b) represents the trajectory of the helicopter tracking a moving helipad and landing on it. In (c) the orientation of the helicopter w.r.t the helipad is shown. In (d) the altitude of the helicopter from the ground while landing is depicted.)

computationally inexpensive visual processing strategy. It relies on the assumptions that 1. the landing target has a well-defined geometric shape and 2. all the feature points of the landing target are coplanar. Since we chose a landing target composed of polygons and the helicopter controller keeps the camera roughly perpendicular to the ground, these two assumptions are justified. A finite state controller is implemented which uses the vision-based strategy for landing site detection. This controller uses a combination of vision and inertial measurements to successfully navigate towards the helipad and land on it. Data from flight trials show that our algorithm and landing strategy work accurately and repeatably even when the landing target is in motion or

is temporarily hidden. In the future we plan to focus our attention on the problem of safe and precise landing of the helicopter in unstructured and dynamic environments.

References

1. University of Southern California Autonomous Flying Vehicle Homepage. http://www-robotics.usc.edu/˜avatar.
2. J. F. Montgomery. (1999) Learning Helicopter Control through Teaching by Showing. Ph.D. thesis, University of Southern California.
3. O. Amidi. (1996) An Autonomous Vision-Guided Helicopter. Ph.D. thesis, Robotics Institute, Carnegie Mellon University.
4. B. Sinopoli, M. Micheli, G. Donato, and T. J. Koo. (2001) Vision-based Navigation for an Unmanned Aerial Vehicle. IEEE International Conference on Robotics and Automation, Seoul, Korea pp. 1757–1765.
5. M. V. Srinivasan, S .W. Zhang, and J. S. Chahl (2001) Landing strategies in honeybees, and possible applications to autonomous airborne vehicles. Biological Bulletin **200**, pp 216–221.
6. P. J. Garcia-Padro, G. S. Sukhatme, J. F. Montgomery (2001) Towards Vision-Based Safe Landing for an Autonomous Helicopter. Robotics and Autonomous Systems. **38(1)**, pp 19–29.
7. O. Shakernia, R. Vidal, C. S. Sharp, Y. Ma, and S. S. Sastry. (2002) Multiple View Motion Estimation and Control for Landing an Unmanned Aerial Vehicle. IEEE International Conference on Robotics and Automation, Washington, D.C. pp. 2793–2798.
8. R. Gonzalez and R.Woods. (1992) Digital Image Processing. Addison-Wesley.
9. M. K. Hu. (1962) Visual Pattern Recognition by Moment Invariants. IRE Transactions on Information Theory. **IT-8**, pp. 179–187.
10. M. J. Mataric (1997) 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. **9**, no. 2-3, pp. 67–83.
11. S. Saripalli, J. F. Montgomery, and G. S. Sukhatme. (2002) Vision-based Autonomous Landing of an Unmanned Aerial Vehicle. IEEE International Conference on Robotics and Automation, Washington D.C. pp. 2799–2804