2007-06-04

Vision-Based Control and Flight Optimization of a Rotorcraft UAV

David Christian Hubbard
Brigham Young University - Provo

VISION-BASED CONTROL AND FLIGHT OPTIMIZATION OF A ROTORCRAFT UAV

by

David Hubbard

A thesis submitted to the faculty of

Brigham Young University

in partial fulfillment of the requirements for the degree of

Master of Science

Department of Computer Science

Brigham Young University

August 2007

BRIGHAM YOUNG UNIVERSITY

GRADUATE COMMITTEE APPROVAL

of a thesis submitted by

David Hubbard

This thesis has been read by each member of the following graduate committee and by majority vote has been found to be satisfactory.

Date	Bryan S. Morse, Chair
Date	Timothy W. McLain
Date	Michael A. Goodrich

BRIGHAM YOUNG UNIVERSITY

As chair of the candidate's graduate committee, I have read the thesis of David Hubbard in its final form and have found that (1) its format, citations, and bibliographical style are consistent and acceptable and fulfill university and department style requirements; (2) its illustrative materials including figures, tables, and charts are in place; and (3) the final manuscript is satisfactory to the graduate committee and is ready for submission to the university library.

Date

Bryan S. Morse
Chair, Graduate Committee

Accepted for the Department

Parris Egbert
Graduate Coordinator

Accepted for the College

Thomas W. Sederberg
Associate Dean, College of Physical
and Mathematical Sciences

ABSTRACT

VISION-BASED CONTROL AND FLIGHT OPTIMIZATION OF A ROTORCRAFT UAV

David Hubbard

Department of Computer Science

Master of Science

A Rotorcraft UAV provides an ideal experimental platform for vision-based navigation. This thesis describes the flight tests of the US Army PALACE project, which implements Moravec's pseudo-normalized correlation tracking algorithm. The tracker uses the movement of the landing site in the camera, a laser range, and the aircraft attitude from an IMU to estimate the relative motion of the UAV. The position estimate functions as a GPS equivalent to enable the rotorcraft to maneuver without the aid of GPS. Flight tests were performed with obstacles and over concrete, asphalt, and grass in daylight conditions with a safe landing area determined by a separate method. The tracking algorithm and position estimation performance are compared to GPS. Accurate time synchronization of the inputs to the position estimation algorithm directly affect the closed-loop stability of the system, proportional with altitude. By identifying the frequency response of each input and adding filters to delay some of the inputs, the closed-loop system maintains stable flight above 18 m above ground, where the system was unstable without the additional filters.

ACKNOWLEDGMENTS

I would like to gratefully acknowledge a few of the people that have made a large project like this possible. Successful research can involve coordinating a lot of independent people's efforts, and there is room here only to name a few, but I would very much like to thank the BYU Multi-Agent Intelligent Coordinated Control Lab for educating me so that the NASA/Army Autonomous Rotorcraft Project, the U.S. Army Aeroflightdynamics Directorate, and the JPL Mobility and Robotic Systems Section would invest their considerable resources in my behalf. And Dr. Bryan Morse and Dr. Timothy McLain have given countless hours to guide and review my work.

Lastly, I would be remiss not to acknowledge my friends and family, without which I might still be only partway through the research process. My mom and dad are my greatest supporters in this endeavor, giving me just enough encouragement yet pushing me to greater achievement. And since the beginning, Jesus Christ has saved me from all sin and sorrow. To Him, and to all who might benefit from the key points of this research, I dedicate this thesis.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Unmanned Aerial Vehicles (UAVs) offer significant advantages over manned vehicles, mainly by eliminating the pilot and cockpit in the aircraft. This also becomes their biggest weakness, however. From single vehicles remotely piloted to multi-UAV missions completed entirely autonomously, the greatest engineering challenges continue to stem from a lack of on-board capability on par with a piloted aircraft. Tele-operated UAVs with a pilot on the ground must rely completely on a communication link to the pilot, sometimes with a reduced situational awareness. Alternatively, adding a certain degree of autonomy to the aircraft can increase the capability of the UAV while decreasing its dependence on the operator. Ultimately, full mission autonomy completely eliminates the need for a human in the loop, obviating the need for man-made landing sites and intervention in the event of in-flight failures.

Fully autonomous flight may be defined various ways. Typically, the ability to fly according to a flight plan represents sufficient autonomy that the operator considers the UAV fully autonomous. At the end of the flight plan, the UAV might loiter or land if so instructed. A new research area under consideration by the US Army is UAV takeoff and landing from an unprepared site. This eliminates the need for the runway or helipad, which enables a UAV to perform challenging new tasks previously not possible even for a ground vehicle, such as:

- Forward Arming and Refueling Point (FARP) and precision resupply operations
- Rescue from hostile or disaster areas

- UAV missions in highly urban areas where closely-spaced buildings and fences prevent fixed-wing UAV or ground-based robot operations
- Site selection or preparation in remote, rugged, or treacherous terrain (e.g. scientific work on ice shelfs or on other planets)

Naturally, a vertical takeoff or landing (VTOL) vehicle makes the landing process much easier, since the UAV requires a smooth, level landing site only slightly larger than itself. Additional advantages gained by having autonomous takeoff and landing capabilities include:

- Automatic mitigation of in-flight failures (e.g. a fuel leak or loss of communications forces the UAV down)
- Automatic deployment and relocation of manned or unmanned aircraft
- Low-altitude surveys and real-time surveillance at sites without *a priori* information about the site or nearby obstacles
- Ground loiter anywhere, greatly increasing endurance capability
- End-to-end mission autonomy for rotorcraft UAVs (allowing operation without a trained pilot)

1.1 PALACE

The US Army Precision Autonomous Landing Adaptive Control Experiment (PALACE) focuses on this specific aspect of UAV research. The PALACE project has been described in detail in [1, 2]. To summarize, the PALACE project goals are for a rotorcraft UAV to take off, navigate a preset course and land autonomously. This thesis will only discuss the landing portion.

Since several conditions can render GPS unusable, the PALACE project uses machine vision for terrain sensing and self-localization. The PALACE project also includes an advanced model-following flight control system (FCS) with accurate flight-identified dynamic models, sensor integration with smooth switching between GPS

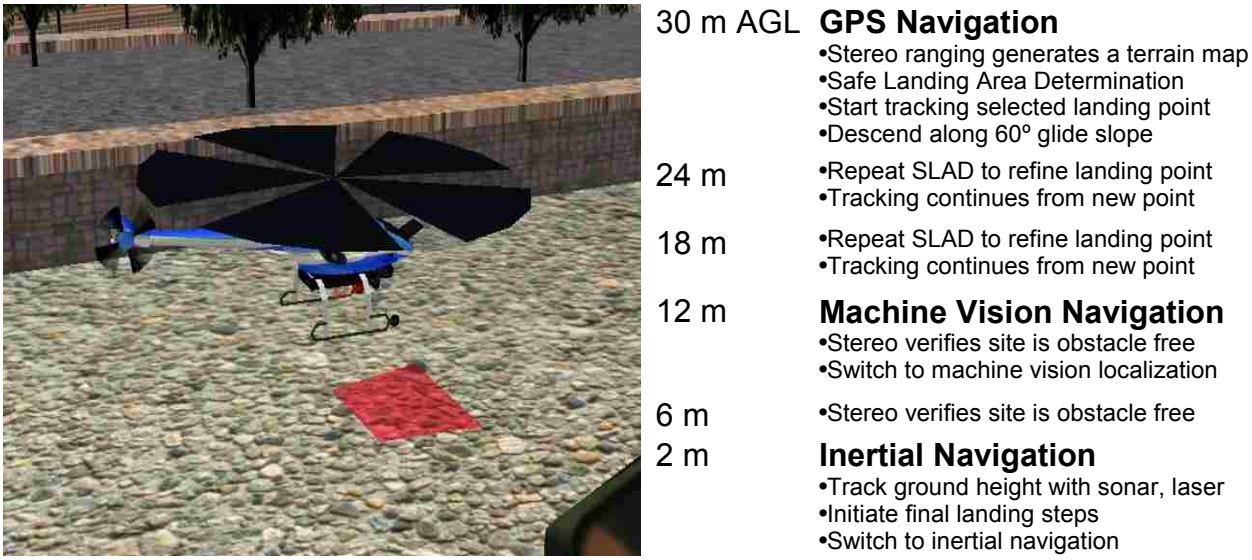


Figure 1.1: A typical PALACE landing starts at 30 m AGL and proceeds downward in 6 m increments. Navigation by machine vision operates from 12 m to 2 m AGL.

and vision-based waypoint control, and an on-board mission manager for decision making and coordination of individual system components. A typical PALACE landing follows the steps in Figure 1.1.

The choice of 30 m above ground as a starting point is somewhat arbitrary, but based on the several navigation techniques used during the landing. Currently, the Global Positioning System represents the *de facto* standard for positioning and navigation. During normal flight operations, an aircraft must maintain a safe height above the ground, which generally eliminates all interference so that a GPS signal is particularly robust. A UAV that encounters GPS reception loss, perhaps from a hostile GPS jammer, can simply climb out until a GPS signal returns. These types of strategies make GPS navigation the method of choice for waypoint flying.

At some point though, the UAV will be instructed to land. Assuming it has a good GPS signal, it may begin by hovering over the designated landing area. Then, a safe landing area determination (SLAD) algorithm such as in [1] calculates the optimal landing site within the UAV's field of view. The 3D terrain map generated by stereo ranging provides a good idea of what obstacles lie in the landing zone.

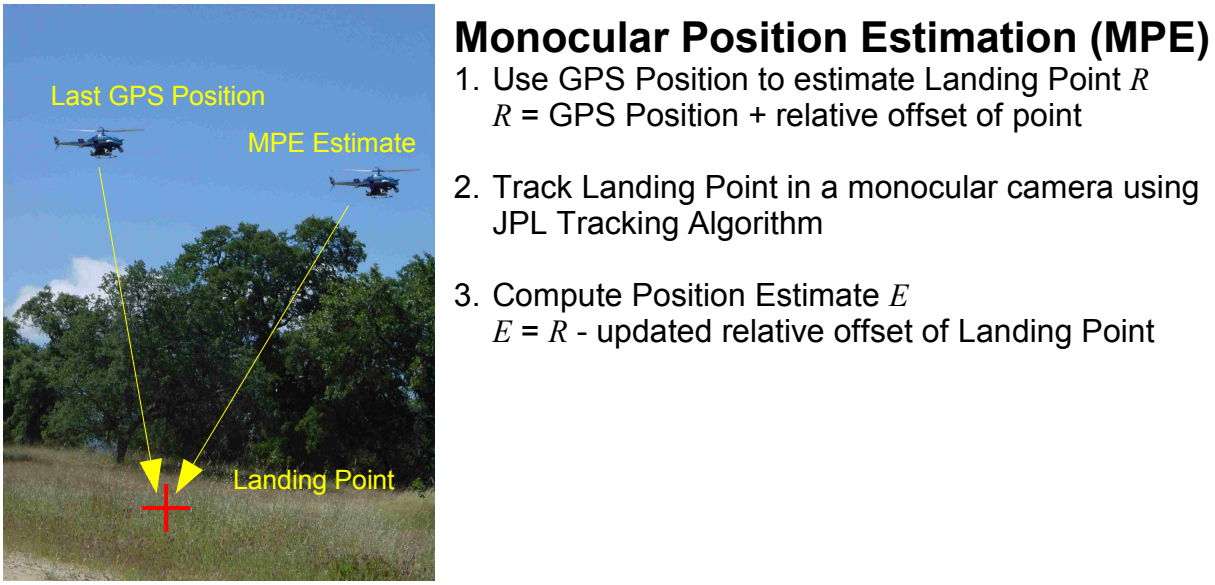


Figure 1.2: Monocular Position Estimation (MPE) initializes from GPS coordinates and then observes the apparent motion in the camera of a stationary ground reference point. From the camera motion and a laser range finder, the aircraft motion can be estimated.

Mostly due to the characteristics of the stereo algorithm, further refinements on the optimal landing site need to be made at 6 m increments during the descent. The UAV follows a 60° glide slope, chosen for minor reasons such as the ease of backing out of a landing, although preliminary testing of the PALACE landing at other descent angles (e.g. 90°) shows reasonable performance.

During the descent process, one of the PALACE project goals is to switch to a vision-based self-localization mode. Machine vision using a passive sensor is ideal for a GPS-denied landing, although it has known limitations such as daylight-only operation, and certain constraints on what patterns it can successfully track. A monocular position estimation (MPE) algorithm provides the self localization replacement which functions as a drop-in GPS replacement (see Figure 1.2 and Section 2.2.3 on page 26).

The typical landing procedure switches out GPS for MPE at 12 m AGL. Data about the stability of the system led to this choice, although stable flight under MPE control has been demonstrated at 30 m AGL.

Figure 1.3: The Sierra Nevada Corporation UCARS-V2 system is used by the Northrop-Grumman Firescout.

1.2 Other Approaches

Ever increasing use of UAVs for a broad range of applications in military and civilian arenas continues to push the envelope of UAV capabilities. The following UAV landing technologies achieve autonomous landings comparable to the PALACE project, each using a slightly different approach. The PALACE project is the only project to date to land autonomously at an unprepared site.

1.2.1 Sierra Nevada Corporation UCARS / TALS system

Sierra Nevada Corporation developed the UAV Common Automatic Recovery System (UCARS), a millimeter-wave radar landing system used to guide a UAV to a designated landing site. UCARS was first demonstrated on a Hunter UAV in 2002 [3], and UCARS and the follow-on Tactical Automatic Landing System (TALS) are currently commercially available. Both have been used for numerous landings in military settings and have a high success rate. The UCARS system was extensively used on the Pioneer UAV, followed by the UCARS-V2 system now used on the Northrop Grumman Fire Scout VTUAV (Figure 1.3) and the TALS system on the AAI Shadow 200 UAV. In terms of reliability and widespread use, the UCARS

Onboard Camera View:

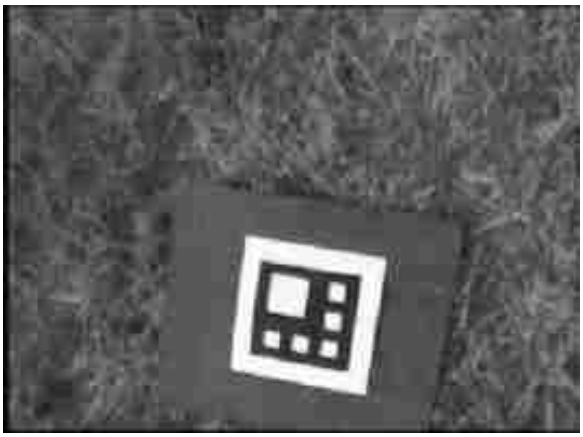


Figure 1.4: The Berkeley Aerobot is an R-50 UAV that uses a designated target to estimate its position.

system has a clear lead, only limited by the radar station which must be deployed at the landing site before the landing.

1.2.2 Berkeley Aerobot

The University of California Berkeley operates six helicopters, with other UAVs under development [4]. Using multi-view motion estimation, they have demonstrated landing a Yamaha R-50 UAV at a designated landing target (Figure 1.4). The landing target is a special case of the motion estimation problem because all the feature points approximate a plane—a degenerate case where the generic eight-point algorithm fails. However, an algorithm based on the *homography constraint,* which explicitly relies on all the feature points lying in a plane, can estimate the UAV's position.

The multi-view algorithm resolves the scale ambiguity by using a landing target of a known size. Then, using a linear algorithm to initialize a non-linear algorithm (to reduce sensitivity to noise), the rotation and translation of the camera in each image is recovered, accurate to 7 cm in translation and 4° in rotation. [5]

Figure 1.5: The University of Southern California AVATAR lands on a helipad marked with an H.

1.2.3 University of Southern California

The University of Southern California, under a grant from the JPL (Jet Propulsion Laboratory) Mobile Autonomous Robotics Software program, developed a similar approach for landing a rotorcraft UAV using vision-based control in 2003. The USC Autonomous Vehicle Aerial Tracking And Reconnaissance (AVATAR) operates initially in *Search Mode* under GPS control until the vision algorithm identifies the helipad, marked with an H. Once the system locates the helipad, it transitions to vision-based control. The system uses sonar ranging to estimate its distance to the ground. This allows it to operate without a GPS signal during the landing (Figure 1.5). The vision algorithm can robustly track the H symbol and land the UAV within 40 cm of the center of the helipad, within 7° in orientation (average performance over 14 flights) [6].

Additionally, landings were performed when the target was in motion. The system still performed at similar accuracy levels (28 cm and 5° error during three flight trials). Using a Kalman filter approach, the system estimated the target position and planned a trajectory to descend toward the target [7].

Figure 1.6: The Georgia Tech active contours algorithm has the ability to track a target even when it moves through an occlusion.

Figure 1.7: The Georgia Tech rotorcraft UAV tracks a target to hold a position.

1.2.4 Georgia Tech

The Georgia Institute of Technology also has an UAV research program, with both fixed-wing and rotorcraft UAVs. With a Yamaha R-MAX, Georgia Tech uses combined vision and INS (Inertial Navigation System) data to control the rotorcraft during hover [8]. They measure the position and orientation of the target, and close a control loop around these measurements to maintain a hover at a position relative to the target (Figure 1.7).

They use a novel approach in their vision system, which tracks contours in the image (see Figure 1.6) using optical flow as an initial estimate of the contour movement [9]. Further, they use a multi-scale grid for the gradient descent to resolve the optical flow vector field. An extended Kalman filter estimates the contour motion through occlusions. Each of these algorithms has been optimized to run in real time.

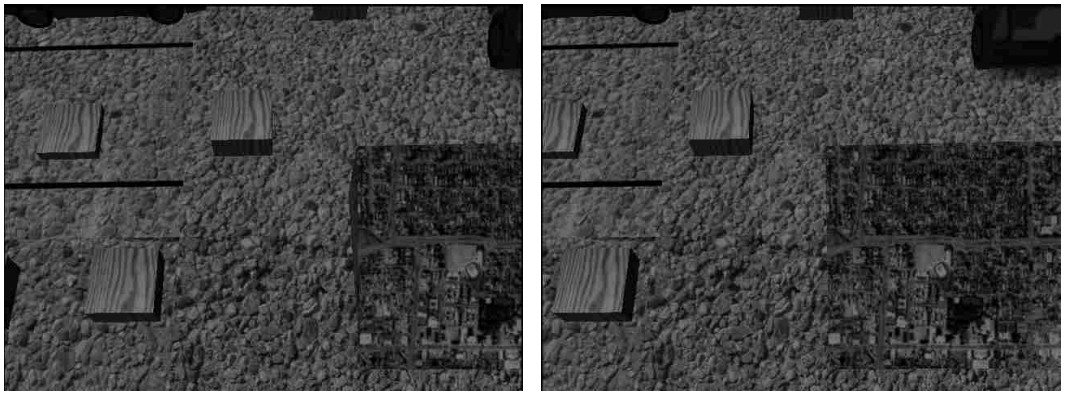


Figure 1.8: The left and right images are the input to the JPL Safe Landing Area Determination (SLAD) algorithm.

1.2.5 SLAD

The safe landing area determination (SLAD) algorithm developed at JPL operates on 3D terrain data, computing the best landing site for the PALACE project. A pair of stereo cameras or a scanning laser can provide the terrain data, transformed to the world coordinate frame. The SLAD algorithm searches the terrain for a smooth, level site large enough for the UAV to land on.

Figure 1.8 shows an obstacle field in the PALACE integrated simulation and the operation of the SLAD algorithm. From a stereo pair of cameras, a stereo correspondence algorithm generates a range map, shown in Figure 1.9. After transforming the range map to world coordinates, and proceeding with the SLAD algorithm, the result is a region in the image with enough space around it to allow the UAV to land. The SLAD algorithm is used by the PALACE project to initiate an autonomous landing, and has received extensive treatment in previous work [2]. This thesis studies the performance of the UAV control system during the landing phase, after the safe landing site has been selected.

1.3 Thesis Statement

This thesis addresses the problem of implementing a vision-based navigation system in hardware. After showing the necessary implementation details to meet

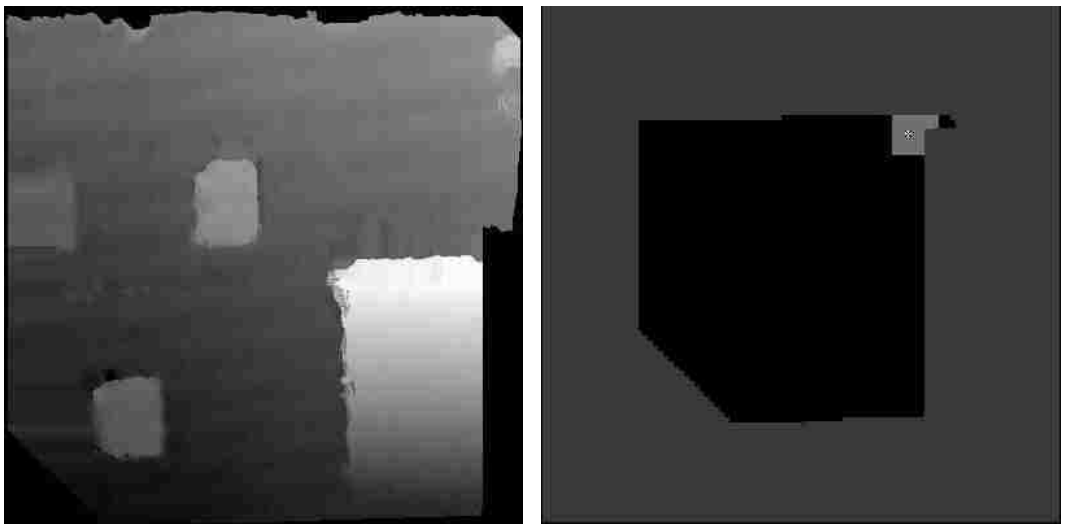


Figure 1.9: The SLAD safe area (right) is found in the range map (left) computed from the stereo images.

real-time constraints and integrate sensors with various characteristics, such as the frequency response and phase delay of each, the thesis presents the performance analysis of the tracking and position estimation algorithms.

This thesis contributes to UAV research in several ways. By building on previous theses of the PALACE project, a final hardware demonstration is achieved. About 30 successful autonomous landings are presented. This represents one of the first rotorcraft UAV's to land at an unprepared site. In addition, the thesis gives requirements for successfully replacing a GPS sensor, the standard postioning sensor for robots of all kinds. Meeting these requirements, though challenging, provides a degree of robustness necessary in order for robotics to become an integrated part of society.

1.4 Outline

Briefly, the rest of this thesis is in the following outline. Chapter 2 details the facilities used to develop the vision-based navigation system, with a description of hardware and software contributed by outside sources. Chapter 3 discusses the implementation details used to meet the goals of real-time closed-loop operation.

Prioritizing the available CPU time and computing expected MPE error aided in designing the control loops. Chapter 4 demonstrates the results of the position estimation algorithm before closed-loop operation. The accuracy of the vision system compares well with a GPS reference. Chapter 5 provides details from the closed-loop operation of the system, engaging the vision-based navigation as the position reference for the flight control system. After closing the control loops, issues of timing and separability become apparent. Chapter 6 reviews the conclusions reached and the potential for future work.

Chapter 2

Facilities

An extensive array of prior work has contributed to this thesis. The design of the rotorcraft UAV and its flight control system by the Army Autonomous Rotorcraft Project, the software for high-speed communication among processes, and the ground station all provide the necessary development environment enabling closed-loop vision research. An introduction to the facilities provides the necessary context for the results of the vision algorithms. Additionally, a discussion of such key issues as timing accuracy requires a complete picture of the system to be fully understood.

2.1 RMAX Flight System

2.1.1 The Autonomous Rotorcraft Project

The Army ARP (Autonomous Rotorcraft Project) at NASA Ames Research Center has developed a complete UAV package on a Yamaha R-MAX, a popular rotorcraft UAV platform. The Yamaha R-MAX has a payload capacity of around 60 lb and a maximum flight time of around 30 minutes. Yamaha R-MAX helicopters are used in Japan for crop spraying operations with several INS/GPS solutions available for experiments.

The R-MAX in its stock configuration comes with a radio transmitter, a built-in control system that provides some yaw-rate damping and system diagnostics. The built-in controller interfaces well with additional on-board control, providing critical measurements such as pilot stick positions and engine speed. The ARP project outfits the R-MAX with additional sensors for fuel level, engine temperature, and electrical

Figure 2.1: The ARP project operates a Yamaha R-MAX UAV platform with a high-precision digital IMU, differential GPS, ground height measurements from radar, sonar, and laser, weight-on-wheels switches, and digital cameras for navigation by machine vision algorithms.

diagnostics. A Crossbow IMU (Inertial Measurement Unit) and Ashtech Differential GPS provide excellent attitude and position information. A downward-pointing sonar and radar measure height above ground (See Figure 2.2). Weight-on-wheels switches indicate when an autonomous landing is complete. In addition, three Point Grey Flea cameras provide various video inputs, mounted on an articulated axis which can point the cameras ahead or downward.

The ARP engineers have identified the full-state system response of the aircraft and developed a simulation model for pre-flight validation of experiments. A complete hardware-in-the-loop simulation is possible using the onboard CPUs to run new software in a beta testing environment, including long test runs and possible radio interference.

Although a safety pilot and safety officer always monitor the aircraft, the ARP system performs all flight maneuvers, from takeoff to landing. A spline-based path planning module directs the aircraft in waypoint maneuvers with numerous parameters to control horizontal ground speed, vertical speed, angular rate, and the physical limits of the actuators.

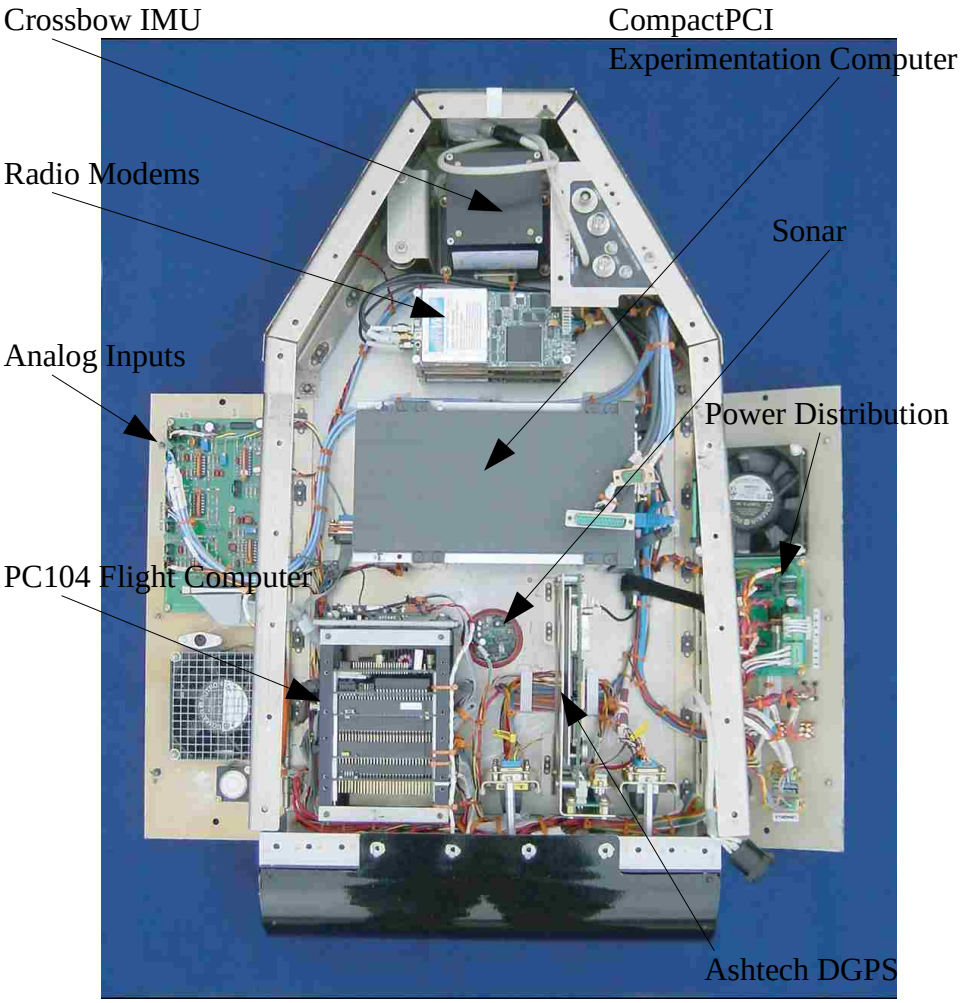


Figure 2.2: The ARP project sensor payload offers an extensive sensor suite for experimentation.

2.1.2 Flight Control System

The R-MAX Flight Control System has five principal axes of motion: collective pitch control (which alters the angle of attack of the main rotor, controlling lift), lateral and longitudinal pitch control (also known as *cyclic* pitch control, which controls the aircraft roll and pitch rates), pedal (the tail rotor angle of attack to control yaw rate), and throttle. The throttle and engine dynamics need some form of control system to maintain sufficient rotor speed under varying loads. The ARP UAV uses a model-following controller. The ARP flight control system reads the aircraft attitude from the IMU and controls the actuators to maintain a desired attitude. Figure 2.3 shows a simplified diagram of the feedback loops, with the position control loop providing attitude commands to the attitude loop. Note that the position control loop operates in a world reference frame, while the actuators affect attitude in the body reference frame; thus the x and y command from the position loop must be rotated by the aircraft heading.

As the inner attitude control loop moves the actuators, the aircraft tilts in the desired direction; then the physics of lift and gravity, thrust and drag, move the rotorcraft toward the desired position. The position control loop achieves both hover and low-speed flight in any direction this way, using a *white rabbit*, a target position which the aircraft follows around its desired path [10]. Even perturbations, such as wind gusts or input from the safety pilot, are resisted by the system.

In Figure 2.3, the position of the aircraft is shown coming directly out of the aircraft (G). This is somewhat unrealistic. The GPS receiver or the vision system must measure the aircraft position. A more accurate diagram showing the source of the position measurement under vision control is Figure 2.6 (page 26). The position estimation system, whether GPS or an alternate, forms the basis for successful navigation during a landing.

2.1.3 Experimentation Computer

The ARP flight control system implements all of the feedback loops shown in Figure 2.3 digitally. For example, the attitude Kalman filter and attitude control loops

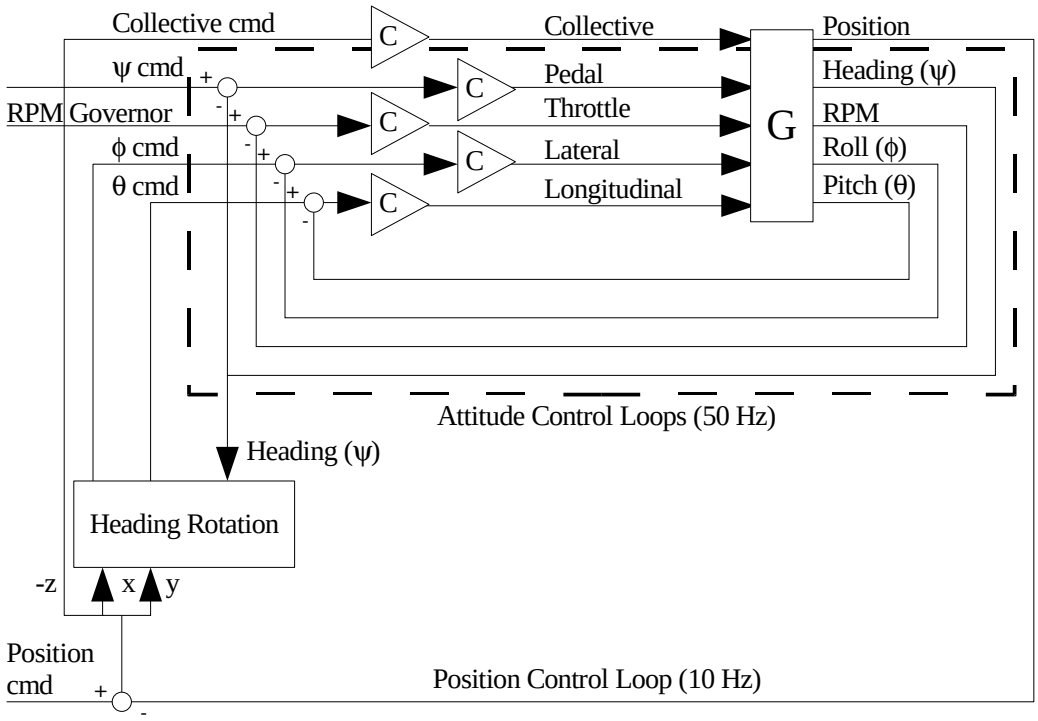


Figure 2.3: The ARP control loops use feedback control to maintain the UAV position and attitude. There are two loops operating at 50 Hz and 10 Hz. It is important to maintain the separation of these loops, as discussed in Chapter 5.

run at a sample rate of 50 Hz, while the position filter and control loops update at 10 Hz. Image processing in vision algorithms requires a great deal of CPU time, and is only required to operate at 10 Hz. Thus, the ARP system includes an experimentation computer in addition to the flight computer to separate processing-intensive image processing from time-sensitive digital control loops that directly affect the stability of the UAV.

A two-CPU setup ensures that the control laws on the flight CPU maintain critical operation in the feedback loop, even when experimental software on the other CPU may cause unexpected behavior. Any rotorcraft UAV has inherent instability which must be actively controlled to maintain nominal flight conditions. (A software crash could precipitate a more serious crash.) The systems have separate busses, so that the bandwidth-intensive video data does not compete with critical commands

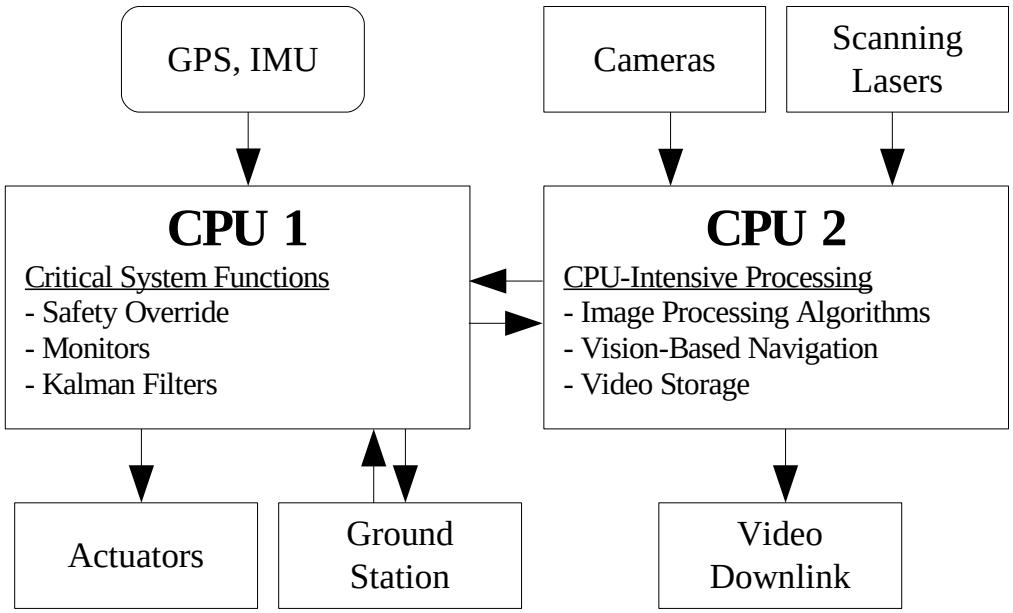


Figure 2.4: The ARP dual-CPU design isolates the functions of the critical control system software from experimental software.

from the ground station or control data. Communication between the computers is limited to a fast Ethernet connection (see Figure 2.4). The decision to attach the scanning laser to the experimentation computer was made because the primary need for the laser return is in the vision-based navigation algorithm. However, the laser data is shared by both computers using the DOMS messaging software.

2.1.4 DOMS

DOMS, the Distributed Open Messaging System, was developed by the ARP project as the communication layer between processes on the aircraft and the ground station. DOMS distributes important system status messages from the several processes on the aircraft CPUs to many monitoring processes on the ground. This abstraction, using a publish/subscribe metaphor, speeds up development time on such a complex system. However, DOMS has some limitations which become significant in the design of the MPE architecture.

2.2 JPL Algorithms

Two JPL algorithms sit at the core of the vision-based navigation system. The Förstner interest operator processes an image to find the strongest feature to track, and the pseudo-normalized correlation function correlates an image from frame to frame. These algorithms are quite useful, and JPL has the expertise in computer vision to extensively test them, and identify their strengths and weaknesses.

2.2.1 Förstner Operator

To successfully identify an object on the ground in frame after frame, the vision system automatically identifies the *strongest feature* using the Förstner operator. Sensor noise from the camera's pixels or the finite resolution of the camera could make near shades of gray and small translations impossible to distinguish; but seeking out the strongest intensity gradients in the image maximizes the signal to noise ratio, a familiar problem in sensor design. Roberts first used a *feature detector*, which ranks pixels in the image on the steepness of the intensity gradient present [11]. Several variations of this concept like the Harris operator and the Kanade-Lucas-Tomasi (KLT) operator find the maximum gradient of the intensity of the image, thus identifying the strongest signals in the image. There are also multiple methods for approximating the intensity gradient of a digital image I. The gradient is denoted by I_x and I_y. (I is not continuous in x, y, or pixel intensity.) The Förstner algorithm uses 3-pixel gradient functions.

$$I_x(x,y) = I(x-1,y) - I(x+1,y)$$
$$I_y(x,y) = I(x,y-1) - I(x,y+1)$$

The particularly important part of the Förstner interest operator [12] is the matrix of second moments of the gradient at every point in the image, called a local structure matrix, $C(x,y)$.

$$C(x,y) = \begin{bmatrix} M_{xx}^2 & M_{xy}^2 \\ M_{xy}^2 & M_{yy}^2 \end{bmatrix} \tag{2.1}$$

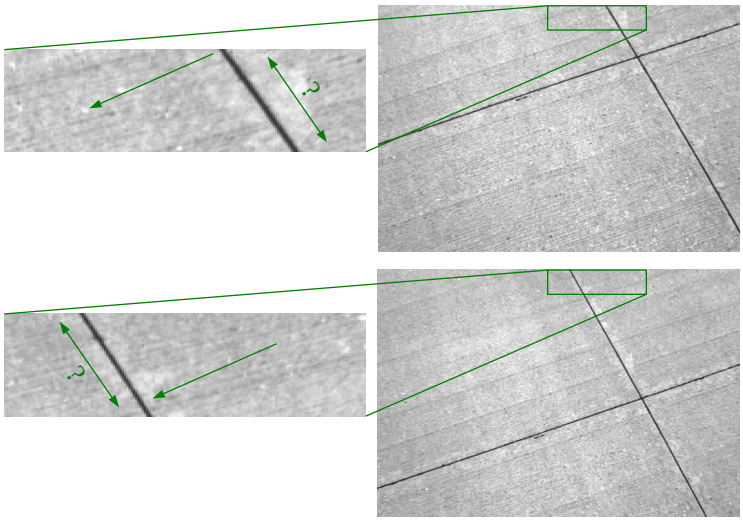


Figure 2.5: The aperture problem stems from the limited aperture, or field of view, visible in each camera image. In the images on the left, the strongest intensity gradient (a seam in the concrete) provides only one principal component of motion (roughly a horizontal motion). It is ambiguous whether the seam has travelled vertically or not. In the images on the right, with a larger aperture, both principal components of motion can be seen.

The second moments are computed by defining a feature size, D_F, which roughly represents the amount of averaging performed to compute a good gradient (See Algorithm 1). Reducing the feature size makes the Förstner algorithm more sensitive to noise. For example, a 3-pixel gradient approximation uses $D_F = 1$, because D_F is measured from the center pixel outward in both directions. $D_F > 7$ is effective. As D_F gets larger, values are more immune to noise by sampling more gradients to compute the local structure matrix. However, the time required to compute the local structure matrix increases as the square of D_F. The Förstner algorithm compares the x gradient to the y gradient by both the second moments of each and the second moment of both taken together. The largest eigenvalue of the local structure matrix C measures the strength of gradients in both x and y, and the Förstner interest operator selects the point in the image with the largest eigenvalue. This point is considered the most *interesting* because it has large gradients (see Algorithm 1).

Input:

W = Width of image (pixels)
H = Height of image (pixels)
D_F = Feature size (pixels from center)
I_0 = Image

1. For every point (x, y) in I_0, compute the gradients I_x and I_y:
   $I_x = I_0(x-1, y) - I_0(x+1, y)$
   $I_y = I_0(x, y-1) - I_0(x, y+1)$.

2. Compute the moments for a feature size D_F and the local structure matrix C:
   $M_{xx}(x, y) = \sum_{u=-D_F/2}^{D_F/2} \sum_{v=-D_F/2}^{D_F/2} [I_x(x+u, y+v)]^2$
   $M_{yy}(x, y) = \sum_{u=-D_F/2}^{D_F/2} \sum_{v=-D_F/2}^{D_F/2} [I_y(x+u, y+v)]^2$
   $M_{xy}(x, y) = \sum_{u=-D_F/2}^{D_F/2} \sum_{v=-D_F/2}^{D_F/2} [I_x(x+u, y+v) I_y(x+u, y+v)]$
   $C(x, y) = \begin{bmatrix} M_{xx} & M_{xy} \\ M_{xy} & M_{yy} \end{bmatrix}$.

3. If C is singular, assign it a value 0. Otherwise, C has two eigenvalues:
   $\lambda_1 = \frac{1}{2}\left(M_{xx} + M_{yy} + \sqrt{4M_{xy}^2 + (M_{xx} - M_{yy})^2}\right)$,
   $\lambda_2 = \frac{1}{2}\left(M_{xx} + M_{yy} - \sqrt{4M_{xy}^2 + (M_{xx} - M_{yy})^2}\right)$
   Since $\lambda_1 > \lambda_2$, we need only use λ_1 from every $C(x, y)$ in the image to find $(x_0, y_0) = \arg\max_{(x,y)} \lambda_1$.

Output:

(x_0, y_0) = Strongest feature in image

Algorithm 1: The Förstner Interest Operator

During a landing, many large man-made objects are partially visible, which means a sharp intensity edge could extend straight through the frame. This makes it unwise to select the absolute strongest feature in the image for the tracking template. The aperture problem ([13] page 193, [14]) causes motion parallel with the edge to be undetectable, e.g. Figure 2.5 shows two views of concrete. The seam only provides movement information orthogonally; movement along it cannot be measured because it is the strongest feature in the image. There are smaller, less pronounced gradients in the image that provide the information needed to reconstruct the camera motion, but naïvely selecting the strongest feature introduces an unresolvable ambiguity. The Förstner operator finds the strongest feature, the most interesting point in the image. Intuitively, a strong corner has the strongest gradients in both x and y. This means the Förstner operator will sometimes choose the maximum eigenvalue for a point on a very strong gradient in just one axis, and thus encounter ambiguity from the aperture problem. A process for addressing this condition is described in Section 4.1 (Page 45).

2.2.2 Correlation Algorithm

The pseudo-normalized correlation algorithm starts from the point selected by the Förstner interest operator, and then tracks a small image piece from one video frame to the next. At 10 Hz, an image from the camera is compared to the last image, in which the navigation system knew the location of the point being tracked. In the previous image, a small *template window* is captured. The PALACE landing uses a template window of 21×21 pixels around the tracked point, balancing a low computational cost with sufficient texture to follow the movements of the R-MAX at 30 m AGL or less.

The algorithm searches for the template window in the new image and a score is assigned at each location. The most likely new position of the tracked point is the highest-scoring point in the new image. The tracking algorithm chooses the best *coherence* score, using the pseudo-normalized correlation defined by Moravec [15], and thus computes the motion of the tracked point in the camera image. Naturally, the scoring function determines what kinds of noise or camera effects have little effect

on the global maximum in the image—and which effects completely bury the right match within the local maxima of false positives elsewhere in the image. Since the scoring function is related to the 2-D convolution, it matches images best when they have only been shifted in x or y from the original image. If the new image is a 3-D affine transformation of the previous image, with rotation, scaling, or perspective foreshortening, the score of the correct match will decrease. To prevent the score from decreasing so far that a local maximum from a false positive captures the best score, the tracking process should cycle as quickly as possible, reducing the magnitude of all transformations.

After computing its location in the new frame, the template window is replaced with the small set of pixels from the new frame. It is the motion of the template window from frame to frame that provides the estimate of the motion of the aircraft, based on the assumption that the object in the template window has not moved. However, the correlation algorithm has a very short memory. Every frame, the template window gets replaced with the new best match. The algorithm can drift from its original point, partially due to the discretization of the image into pixels. If the algorithm cannot track the point (e.g. the point moved outside the camera's field of view), some other position in the image with the highest score is selected, which introduces a discontinuity in the tracking. Chapter 6 discusses future work that could reduce this form of error.

The correlation algorithm (Algorithm 2) uses a *search window*, instead of searching the entire image. For a previous image I_{N-1} and a new image I_N, the correlation algorithm finds a new (x_N, y_N) from the previous (x_{N-1}, y_{N-1}). The correlation algorithm searches a rectangular region in the new image, I_N, centered around the previous point (x_{N-1}, y_{N-1}). The best match becomes the new (x_N, y_N).

This is where the search window dimensions, S_W and S_H, are used. A larger search window means the tracked object in the template window can move farther and still be found. Typically, this movement stems more from the angular rates of the aircraft than from translation of the aircraft. If the camera images are processed at 10 Hz, a search window of 61×61 pixels has been sufficient to find the template window

in the new image. This corresponds to 610 pixels per second, or a maximum pitch rate of $610 * 0.421 = 256.59\frac{rad}{s}$ (one pixel sweeps approximately $tan^{-1}\left(\frac{480}{1073}\right) = 0.421rad$ of pitch, for $f = 1073$ and a 640×480 image using the pinhole camera model). The search window $a(1...S_W, 1...S_H)$ is the region $I_N(x_{N-1} - \frac{S_W}{2}...x_{N-1} + \frac{S_W}{2}, y_{N-1} - \frac{S_H}{2}...y_{N-1} + \frac{S_H}{2})$.

The template window from the previous image I_{N-1} becomes $b(1...T_W, 1...T_H)$, which the correlation algorithm uses to search I_N. The algorithm need only store the pixels $I_{N-1}(x_{N-1} - \frac{T_W}{2}...x_{N-1} + \frac{T_W}{2}, y_{N-1} - \frac{T_H}{2}...y_{N-1} + \frac{T_H}{2})$, defined as $b(1...T_W, 1...T_H)$ in preparation for the new image I_N.

The algorithm by Moravec differs from standard Normalized Cross-Correlation (NCC [16]) in several ways. Moravec's paper [15] points out that the arithmetic mean can replace the geometric mean, and uses the dominant terms only to find a pseudo-normalized correlation. The definition of Normalized Cross-Correlation in [16] is

$$\lambda(u,v) = \frac{\Sigma_{x,y}[a(x+u, y+v) - \mu_a][b(x,y) - \mu_b]}{\sqrt{\Sigma_{x,y}[a(x+u, y+v) - \mu_a]^2 \Sigma_{x,y}[b(u,v) - \mu_b]^2}}. \tag{2.2}$$

Where μ_a is the mean of a (over the template) and μ_b is the mean of b. Note that the terms in the denominator are variances, so if σ_b is the standard deviation of b and $\sigma_{a(u,v)}$ is the standard deviation of the region of a under the template, then

$$\lambda(u,v) = \frac{\Sigma_{x,y}[a(x+u, y+v) - \mu_a][b(u,v) - \mu_b]}{\sqrt{\sigma^2_{a(u,v)}\sigma^2_b}}.$$

Moravec then uses the approximation

$$\text{coherence}(u,v) = \frac{\frac{\Sigma_{x,y}\left[a(x+u, y+v)b(x,y)\right]}{T_W T_H} - \mu_a\mu_b}{\frac{\sigma_a^2 + \sigma_b^2}{2}}. \tag{2.3}$$

Note that the arithmetic mean replaces the geometric mean in the denominator, and the terms $a(x+u, y+v)\mu_b$ and $b(x,y)\mu_a$ are dropped from the numerator. Moravec describes his approach in more detail in his paper [15]. He uses some optimizations, but follows the basic algorithm detailed in Algorithm 2.

Input:

$$\begin{aligned}(S_W, S_H) &= \text{Search window size (pixels)}\\ (T_W, T_H) &= \text{Template window size (pixels)}\\ a &= \text{Search window}\\ b &= \text{Template window}\end{aligned}$$

1. Repeat over the search window $a(u, v)$

$$\begin{aligned}\text{from} \quad a(1,1) &= I_N(x_{N-1} - \tfrac{S_W}{2}, y_{N-1} - \tfrac{S_H}{2})\\ \text{to} \quad a(S_W, S_H) &= I_N(x_{N-1} + \tfrac{S_W}{2}, y_{N-1} + \tfrac{S_H}{2})\end{aligned}$$

2. Compute means and variances:

$$\begin{aligned}\mu_a &= \frac{\Sigma_{x,y} a(u+x, y+v)}{T_W T_H}\\ \sigma_a^2 &= \frac{\Sigma_{x,y} a(u+x, v+y)^2}{T_W T_H} - \mu_a^2\\ \mu_b &= \frac{\Sigma_{x,y} b(x, y)}{T_W T_H}\\ \sigma_b^2 &= \frac{\Sigma_{x,y} b(x, y)^2}{T_W T_H} - \mu_b^2\end{aligned}$$

3. Compute the pseudo-normalized correlation score at (u, v):

$$\text{coherence}(u, v) = \frac{\frac{\Sigma_{x,y}\left[a(x+u, y+v) b(x, y)\right]}{T_W T_H} - \mu_a \mu_b}{\frac{\sigma_a^2 + \sigma_b^2}{2}}$$

4. $(x_N, y_N) = \arg\max_{(u,v)} \text{coherence}(u, v)$

Output:

$$(x_N, y_N) = \text{position of best correlation match in search window (pixels)}$$

Algorithm 2: Pseudo-Normalized Correlation

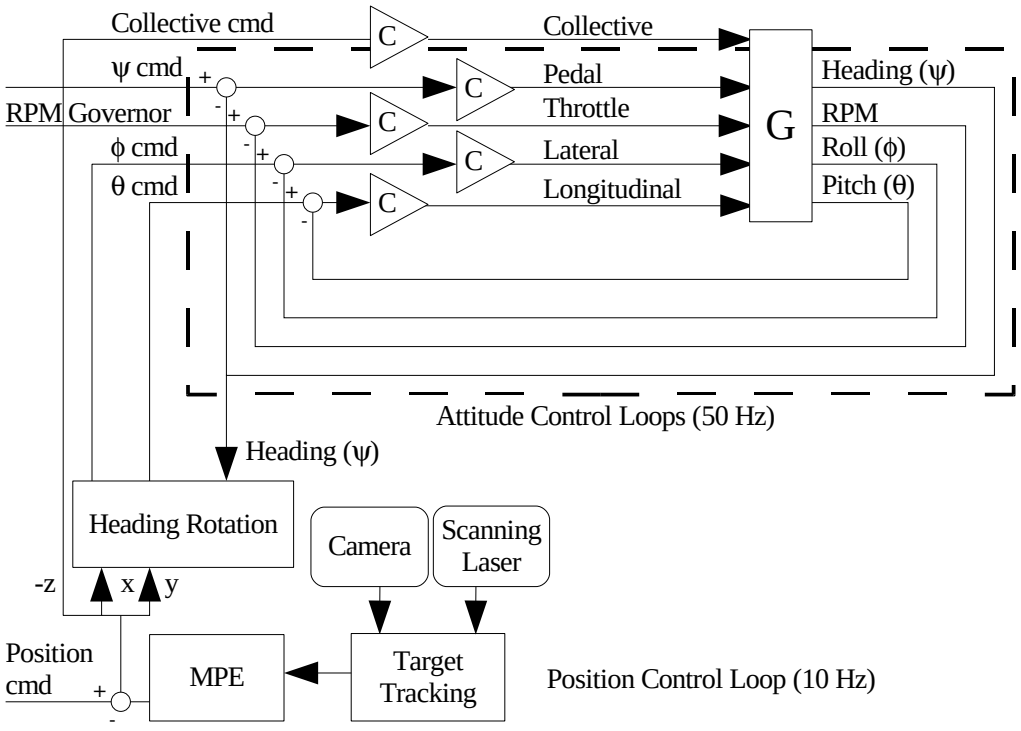


Figure 2.6: The ARP control loops under MPE control function almost entirely identically to those in Figure 2.3. However, the GPS sensor is replaced with the tracking and MPE algorithms.

These two algorithms developed at JPL accomplish core image processing tasks, essential to navigating solely under vision control. By first selecting the highest-scoring point using the Förstner interest operator, and then following that target from frame to frame with the pseudo-normalized correlation algorithm, the vision-based navigation system can calculate the movement of the target in the camera. Assuming the target remains stationary on the ground, this perceived movement is resolved as movement of the camera, and is used to estimate the aircraft's position.

2.2.3 Monocular Position Estimation

The product of the JPL tracking algorithms is a pixel location of a target in image coordinates. The Monocular Position Estimation (MPE) algorithm transforms this 2-D position into the 3-D position of the camera in world coordinates, which

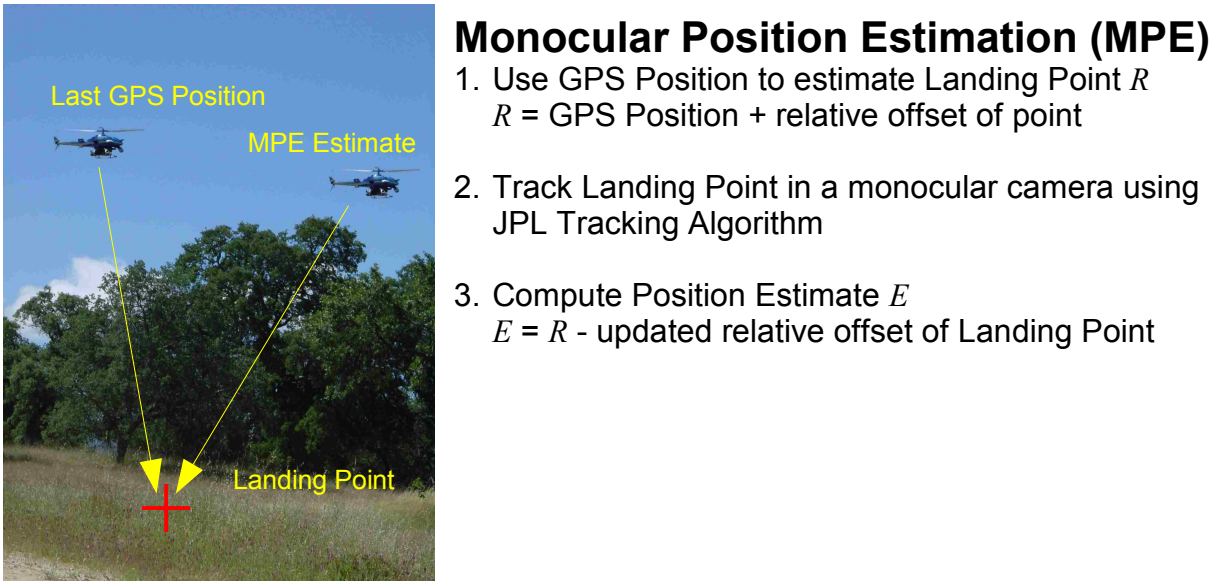


Figure 2.7: Monocular Position Estimation (MPE) initializes from GPS coordinates and then observes the apparent motion in the camera of a stationary ground reference point. From the camera motion and a laser range finder, the aircraft motion can be estimated (see Figure 1.2).

can replace the position reference provided by GPS for the control of the aircraft (Figure 2.6). As long as the target remains in the camera's field of view, the aircraft can move about freely. This setup works well for an autonomous landing. The landing point is selected in the image as the tracking target. Then, as the aircraft descends toward the landing point, the target will remain in the camera's view.

Figure 2.7 shows that any movement of the landing point is resolved as movement of the aircraft. Initially, the Förstner algorithm is used to find a point to track near the landing point. This point becomes the target tracked by the vision system. A range to target is taken from the scanning laser, and using the aircraft attitude data from the IMU, the coordinates of the target are saved as the *ground reference*. This is step 1 in Figure 2.7. For step 2, the correlation algorithm finds a new target location in pixel coordinates, and an updated position estimate is calculated with new IMU and range data. In this way, by tracking the apparent motion of the target in the camera image, MPE senses the aircraft's position just as a GPS sensor would.

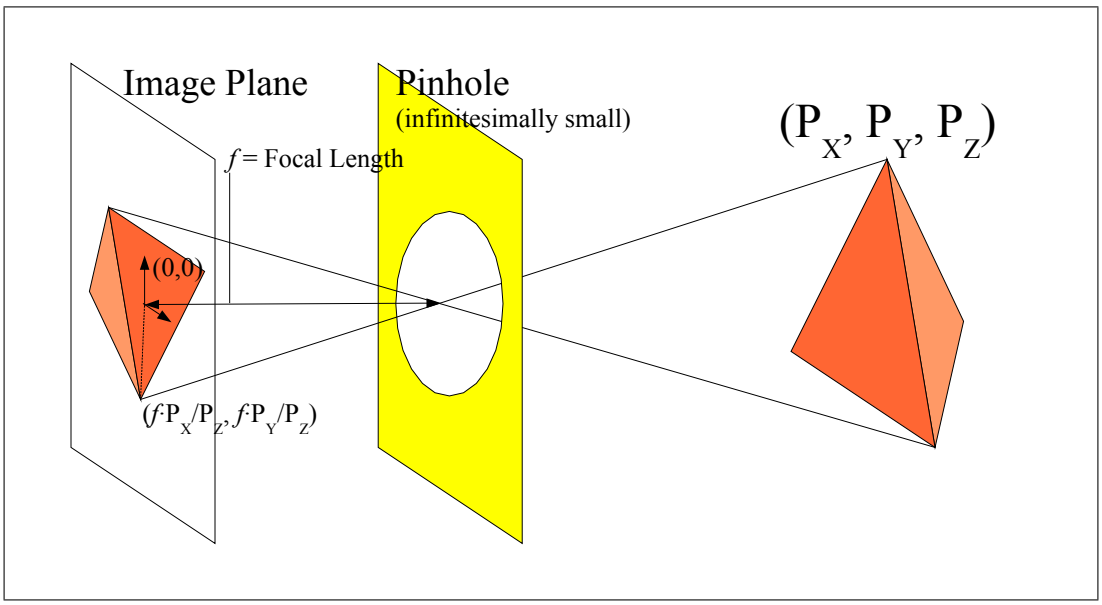


Figure 2.8: The pinhole camera model demonstrates the projection of an object in 3-space onto the 2-D camera plane.

The derivation of Algorithms 3 and 4 for generating the ground reference point and computing an updated position estimate start with the pinhole camera model. A real camera with lenses has slightly different optical properties, but a pinhole model is a fairly good approximation. The pinhole camera model (Figure 2.8) equates the position of an object in 3-space (in camera coordinates) with the point on the image plane in 2-D image coordinates where the object is projected by the optics of the pinhole camera. All the light projected onto the image plane passes through an infinitesimally small point and is rotated 180°. Since the image plane is at the focal length f behind the pinhole, the point (P_X, P_Y, P_Z) is projected onto the image plane at (x_I, y_I)

$$\begin{pmatrix} x_I \\ y_I \end{pmatrix} = \begin{pmatrix} \frac{f \cdot P_X}{P_Z} \\ \frac{f \cdot P_Y}{P_Z} \end{pmatrix}. \tag{2.4}$$

Equation 2.4 stems from the similar triangles in Figure 2.8. The triangle from the pinhole to the image plane origin to $(x_I, 0, 0)$ is similar to the triangle from the pinhole $(0, 0, f)$ to $(0, 0, P_Z)$ to $(P_X, 0, P_Z)$. The image plane origin is at a distance f

Input:

(E_{G0}, N_{G0}, H_{G0}) = Current camera position in the GPS reference frame (UTM coordinates)

R_0 = Distance reported by range sensor (meters)

M_0 = 3×3 transformation matrix giving camera orientation

W = Width of image (pixels)

H = Height of image (pixels)

(x_0, y_0) = position of ground reference in image I_0 (pixels)

$$\begin{pmatrix} E_R \\ N_R \\ H_R \end{pmatrix} = \begin{pmatrix} E_{G0} \\ N_{G0} \\ H_{G0} \end{pmatrix} + M_0 \cdot \begin{pmatrix} \frac{R_0}{f}(x_0 - \frac{W}{2}) \\ \frac{R_0}{f}(y_0 - \frac{H}{2}) \\ R_0 \end{pmatrix}.$$

Output:

(E_R, N_R, H_R) = Ground Reference (UTM coordinates)

Algorithm 3: Compute The Ground Reference, as developed in [17]

from the pinhole. The side P_X is similar to x_I, and P_Z is similar to f. Thus $x_I = \frac{f \cdot P_X}{P_Z}$ and similarly, $y_I = \frac{f \cdot P_Y}{P_Z}$.

The algorithms that follow invert the operation of the pinhole camera. They compute the point in 3-space given the point in the image plane. Inverting the pinhole camera model is possible by introducing a range, R_N, which directly measures P_Z. Then the equations for P_X and P_Y follow from Equation 2.4, with the effect of sampling the image indicated by the floor operator ($\lfloor \rfloor$).

$$\begin{pmatrix} P_X \\ P_Y \\ P_Z \end{pmatrix} = \begin{pmatrix} \frac{R_N \cdot \lfloor x_I \rfloor}{f} \\ \frac{R_N \cdot \lfloor y_I \rfloor}{f} \\ R_N \end{pmatrix}. \tag{2.5}$$

The ground reference point is computed using Equation 2.5. By inverting the pinhole camera model, the coordinates of the ground reference point are transformed from image coordinates (2-D) to 3-D coordinates in the camera reference frame. Then they are transformed to world coordinates using the M_0 transformation matrix, which encodes the attitude and position of the camera as reported by the IMU (see Algorithm 3).

Input:

(E_R, N_R, H_R) = Ground Reference in GPS reference frame (UTM coordinates)

R_N = Distance reported by range sensor (meters)

M_N = 3×3 transformation matrix giving camera orientation

W = Width of image (pixels)

H = Height of image (pixels)

(x_N, y_N) = Updated position of Ground Reference in the newest image I_N (pixels)

$$\begin{pmatrix} E_E \\ N_E \\ H_E \end{pmatrix} = \begin{pmatrix} E_R \\ N_R \\ H_R \end{pmatrix} - M_N \cdot \begin{pmatrix} \frac{R_N}{f}(x_N - \frac{W}{2}) \\ \frac{R_N}{f}(y_N - \frac{H}{2}) \\ R_N \end{pmatrix}.$$

Output:

(E_E, N_E, H_E) = The position of the camera (UTM coordinates)

Algorithm 4: Compute The Updated Position Estimate, as developed in [17]

The updated position estimate is computed in the same manner. Algorithm 4 computes a new position estimate using the ground reference point (see $E = R - X$ in Figure 2.7 and Equation 2.5). Possible sources of noise and error come from the correlation algorithm, which resolves only to the precision of one pixel in a discretized image, and from the laser range finder. These sources of noise and their effect on the accuracy of MPE are discussed in Chapter 3.

2.3 Integrated Full-Mission Simulation

The Real-time Interactive Prototype Technology Integration / Development Environment (RIPTIDE) developed at NASA Ames Research Center provides the simulation environment for the PALACE project. By simulating the vision algorithms first, engineering time is saved by identifying and correcting most of the performance issues before flight. Therefore, the first step toward a successful autonomous landing is a full mission simulation. Previous work has demonstrated the key issues learned from the RIPTIDE simulation [2].

The full-mission simulation uses RIPTIDE to generate OpenGL camera scenes for the vision algorithm. The identified model of the aircraft and the control laws

provides a realistic simulation of the aircraft dynamics. Real-time constraints thus become apparent, and the simulation models effects of excessive time delay accurately.

The simulation provides an initial working knowledge of how the SLAD algorithm should select a target landing site. Some testing of tracking performance is also possible, with varying wind and turbulence values [18, 2].

2.4 Summary

These simulation tools, math models, image processing algorithms, and hardware provide the necessary facilities to implement and test a vision-based navigation system suitable for an autonomous landing. The JPL algorithms process camera images captured by the ARP R-MAX and feed 2-D position data to the MPE algorithms, which operate in the position loop just as a GPS receiver would. This thesis contributes an analysis of different noise characteristics between MPE and GPS, and the incremental error build-up inherent in the JPL tracking algorithm, in Chapter 4. This thesis also discusses the frequency response of each of the sensors used by MPE, leading to accurate time synchronization in Chapter 5. The limitations of the algorithms, such as the possibility of a discontinuity in the tracking are future work, discussed in Chapter 6.

Chapter 3

Implementation

Continuing the presentation of the previous chapter, the software architecture computes digital control responses within strict timing constraints and provides safety measures that mitigate possible failures in operation. The properly designed architecture will quickly and accurately transfer data between the algorithms discussed in Chapter 2, fulfilling all requirements imposed by the software without posing any obstacles. Architecture design decisions, safety considerations, and limits of the digital sensors are discussed in some detail in this chapter.

3.1 Architecture

The ARP R-MAX offers sufficient computing resources for vision-based navigation. However, several significant challenges require design decisions before vision-based navigation can control the R-MAX. The ARP control law implementation smoothly transitions between various software position inputs (e.g. GPS and MPE). The camera images captured by the experimentation CPU have various destinations, all of which perform best with a real-time zero-copy implementation for low latency. In order to actually land under vision-based control, capture of each of the sensory inputs must occur as close as possible to the same instant. These inputs combine to form the position estimate as long as the data points approximate a single point in time. Using careful timestamping in the several processes on the ARP R-MAX, and by scheduling the process priorities, the experimentation CPU can generate a reliable and accurate position estimate.

Figure 3.1: The ARP left and color camera are mounted on the left end of the payload. The right stereo camera is mounted on the opposite side.

3.1.1 Camera Images

The ARP R-MAX offers three Point Grey Flea^{TM} cameras on an IEEE 1394 Firewire-400 bus. The cameras automatically synchronize to each other on the bus, so images from the cameras are taken at about the same time. Initial testing of the acquired images showed about 100μs difference between the images. An external trigger input is available on the cameras, and if the aircraft were moving at high speed, an external trigger would practically eliminate that difference. For hovering or low speeds, the automatic self-synchronization feature suffices.

The stereo images used as input to the safe landing area determination (SLAD) algorithm require synchronized cameras. The first two cameras capture simultaneous left and right images in a stereo pair (see Figure 3.1). Without simultaneous capture of the images, the calibration of the cameras for the stereo results degrades significantly when the aircraft is in motion. Although the design decisions made during the

implementation stage take this requirement into consideration, this thesis does not discuss SLAD performance. SLAD only autonomously provides the location of the landing site and target for the tracking algorithm (see Section 1.2.5 on page 9).

The third camera captures a color image for live display at the ground station. The ARP R-MAX uses two radio links, a radio modem and an IEEE 802.11g wireless link to enforce separation of control commands and live video. The radio modem has greater range and reliability, and is the link for DOMS messages. The flight CPU with the FCS processes receives commands over this link. The experimentation CPU transmits a motion-JPEG stream over the 802.11g link, but if some of the video is dropped when the aircraft is operating at a distance from the ground station, the video software can automatically recover.

In addition, a flash device captures images and state from the running vision system. The vibration on board the R-MAX prevents the use of a normal hard disk, a typical limitation of aviation storage devices, but a Flash device with a DMA-capable IDE interface stores all the information necessary to reconstruct the operation of the algorithm after a flight. Images in this thesis typically are drawn from this data stream.

Figure 3.2 shows the four processes that use the video stream from the cameras. Each process could potentially monopolize the experimentation CPU. A design decision was made to prioritize the CPU time, and Figure 3.2 shows the time divisions among the processes. The capture process has the simplest job, maintaining the video buffer from which each of the other processes may read, reading from other sensors such as the scanning laser, and monitoring the synchronization of the cameras. (Occasionally, the Firewire bus causes the cameras to come out of sync. This is flagged with a warning.) The capture process is named Sensors, and the cameras feed frames into it at 30 Hz. The Sensors process is given hard real-time priority over all other processes on the experimentation CPU, including priority over kernel processes. This prevents delays that can stem from another process (such as the disk storage process) which may flood the kernel with requests. Storing all the data generated by the vision system requires a non-trivial amount of bus bandwidth and a high-speed

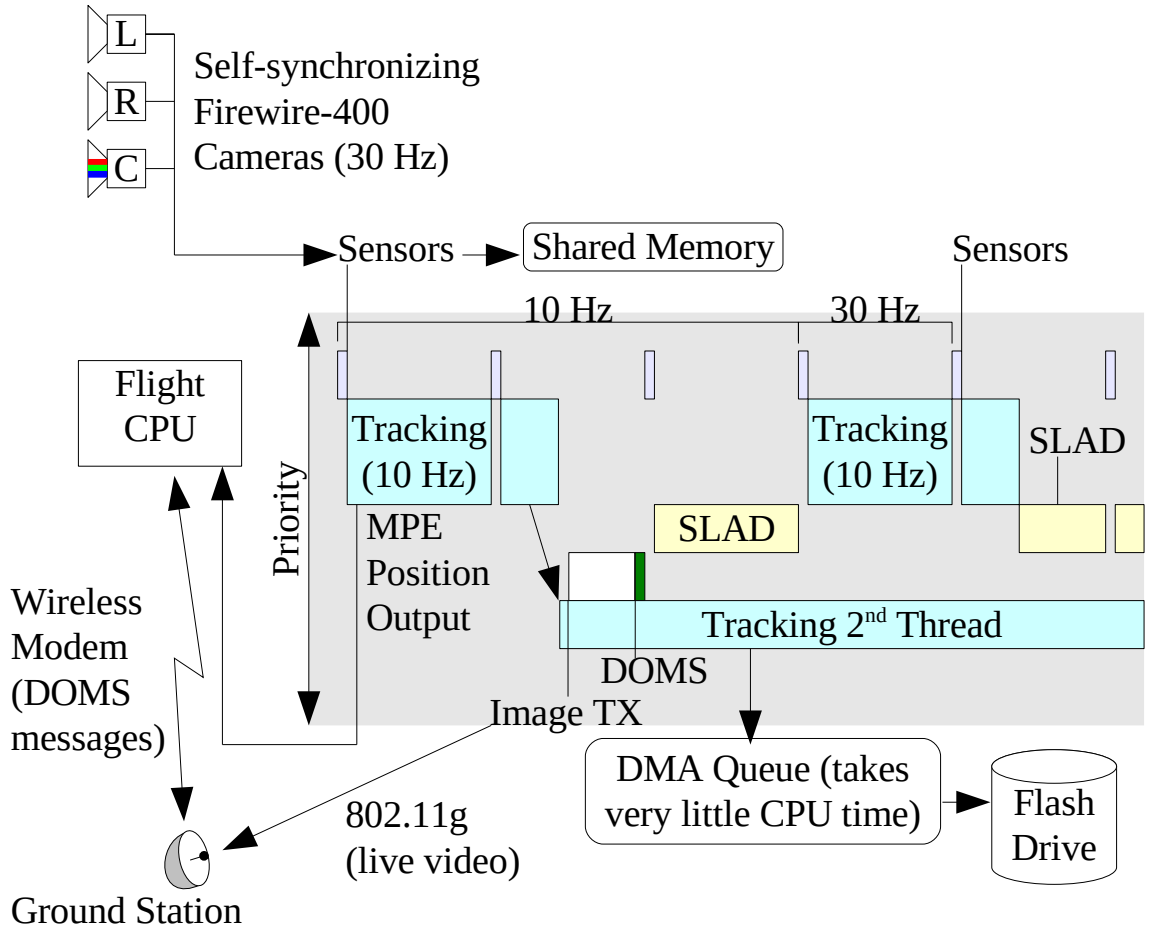


Figure 3.2: The timing on the experimentation CPU must be carefully prioritized to allow the position estimate to be computed at a steady 10 Hz.

high-capacity storage device, but does not have to finish in real time, so Sensors does not log data.

The vision navigation process receives second highest priority. This process, called Tracking, must operate at 10 Hz to continuously provide updates to the Flight Control System on the other CPU. Longer-running tasks, such as the stereo ranging calculations for SLAD, fill in the unused CPU time after the tracking process has completed. At the end of a tracking cycle, a second thread in Tracking lines up the generated data in a queue and logs the system state to the disk. Since the kernel disk driver uses the DMA (Direct Memory Access) controller and very little CPU, this proceeds simultaneously with other processes.

The SLAD algorithm receives third priority, but is not always running. Only when the operator requests a stereo analysis of the landing site, which happens periodically during a landing, the other lower-priority processes are suspended while SLAD computes a safe landing site. When SLAD completes, all the low-priority processes can resume in the unused CPU time, including Image TX, which compresses raw video frames and transmits them to the ground station.

By scheduling each of the processes and using shared memory to access video frames, the experimentation CPU operates at almost 100% utilization, yet important timing deadlines like the output of the Tracking process to the flight CPU are always met. The flight CPU sees a consistent 10 Hz position update, just as it would see from a GPS receiver, and the experimentation CPU keeps a comprehensive log of all the input and output used by vision system.

3.1.2 Mission Manager

The flight CPU only switches inputs to the vision-based navigation when the vision system has operated within safe limits for two seconds (Figure 3.3). The communication between the flight CPU and the experimentation CPU coordinates when the FCS finally performs a smooth transition to the estimate provided by MPE. In addition, the FCS sends GPS coordinates and IMU data (such as aircraft attitudes) to the experimentation CPU.

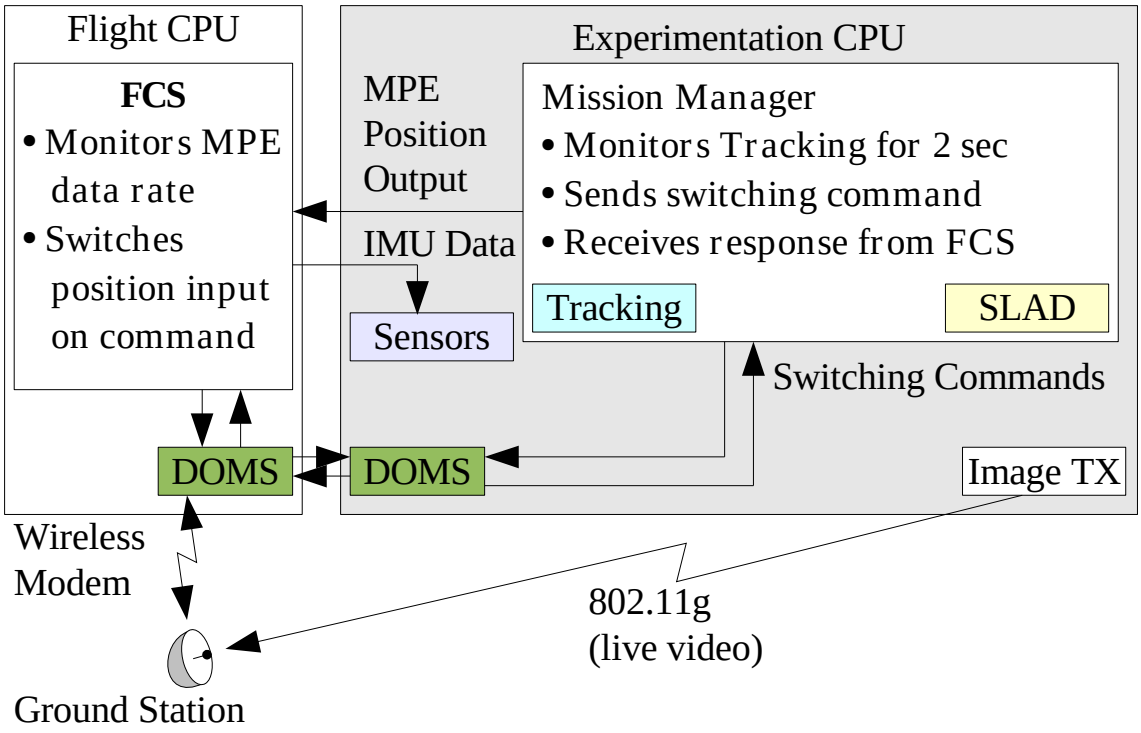


Figure 3.3: The Mission Manager sends switching commands to the flight CPU.

A Mission Manager process on the experimentation CPU monitors the tracking performance and sends the command to the flight CPU to switch modes. If an error occurs during vision-based navigation, the Mission Manager can request a switch back to GPS. If the flight CPU receives erratic position data (not at 10 Hz, or otherwise outside safe limits, as discussed in Section 3.2), it will notify the Mission Manager that it switched back to GPS.

The Mission Manager uses the GPS coordinates to initialize MPE with real GPS data. MPE does not require an actual GPS position, but the comparison demonstrates the accuracy of vision-based navigation. The Mission Manager stores GPS data in the log with the MPE data for post-flight validation, but disregards GPS after switching to MPE.

Although the switching messages in Figure 3.3 use DOMS, the Tracking process communicates with the flight CPU using UDP. This decision stems from a characteristic 20-60 ms delay incurred by DOMS. Figure 3.2 shows that the Tracking process outputs a position from MPE after around 40 ms, so an additional 60 ms represents a significant delay. Even more important, the variability of the DOMS message timing could cause unpredictable system behavior. Eliminating this delay in the position control loop by bypassing DOMS and using a UDP socket instead allows the system to behave predictably. The position loop has a lower, and more reliable time delay (see Figure 2.6 on page 26).

3.1.3 Camera Accuracy

The JPL tracking algorithm in Section 2.2 finds the landing site location down to a single pixel in the camera image. From Equation 2.5 (page 29), the equation for inverting the pinhole camera model includes the floor operator

$$\begin{pmatrix} P_X \\ P_Y \\ P_Z \end{pmatrix} = \begin{pmatrix} \frac{R_N \cdot \lfloor x_I \rfloor}{f} \\ \frac{R_N \cdot \lfloor y_I \rfloor}{f} \\ R_N \end{pmatrix}.$$

Because the JPL tracking algorithm is only accurate to a single pixel, the return from the laser range finder, R_N, affects the accuracy of the position estimate. A larger R_N means that the pixel (x_I, y_I) covers a larger area, and so the position estimate is less accurate. The derivation of the noise caused by the floor operator assumes that P_X is independently, identically distributed with respect to x_I, R_N and f, and from Equation 2.5 that P_X falls in the range $[\lfloor x_I \rfloor, \lfloor x_I \rfloor + 1)$. The derivations of the standard deviation of P_X and P_Y are identical, so only the noise model for P_X is derived here. First, the expected values, $E(P_X)$ and $E(P_X^2)$ are computed. Note that this is a basic derivation of a uniform distribution through a linear transformation.

$$
\begin{aligned}
\text{let } x_0 &= \lfloor x_I \rfloor
\end{aligned}
$$

$$
\begin{aligned}
E(P_X) &= \int_{x_0}^{x_0+1} \frac{R_N x}{f} dx \\
E(P_X) &= \frac{R_N}{f} \int_{x_0}^{x_0+1} x dx \\
E(P_X) &= \frac{R_N}{f} \left(\frac{(x_0+1)^2}{2} - \frac{x_0^2}{2} \right) \\
E(P_X) &= \frac{R_N}{f} \left(\frac{x_0^2}{2} + x_0 + \frac{1}{2} - \frac{x_0^2}{2} \right) \\
E(P_X) &= \frac{R_N}{f} \left(x_0 + \frac{1}{2} \right)
\end{aligned}
$$

$$
\begin{aligned}
E(P_X^2) &= \int_{x_0}^{x_0+1} \left(\frac{R_N x}{f} \right)^2 dx \\
E(P_X^2) &= \frac{R_N^2}{f^2} \int_{x_0}^{x_0+1} x^2 dx \\
E(P_X^2) &= \frac{R_N^2}{f^2} \left(\frac{(x_0+1)^3}{3} - \frac{x_0^3}{3} \right) \\
E(P_X^2) &= \frac{R_N^2}{f^2} \left(\frac{x_0^3}{3} + x_0^2 + x_0 + \frac{1}{3} - \frac{x_0^3}{3} \right) \\
E(P_X^2) &= \frac{R_N^2}{f^2} \left(x_0^2 + x_0 + \frac{1}{3} \right)
\end{aligned}
$$

The error from the floor operator is the difference between the truncated value (x_0) and the actual value (P_X).

$$
d = P_X - \frac{R_N x_0}{f} \tag{3.1}
$$

The mean of d ($E(d)$) and standard deviation of d are derived below.

$$
\begin{aligned}
E(d) &= \int_{x_0}^{x_0+1} P_X - \frac{R_N x_0}{f} dx \\
E(d) &= \int_{x_0}^{x_0+1} P_X dx - \int_{x_0}^{x_0+1} \frac{R_N x_0}{f} dx \\
E(d) &= E(P_X) - \frac{R_N x_0}{f} \int_{x_0}^{x_0+1} dx \\
E(d) &= \frac{R_N}{f} \left(x_0 + \frac{1}{2} \right) - \frac{R_N x_0}{f} \left(x_0 + 1 - x_0 \right) \\
E(d) &= \frac{R_N}{f} \left(x_0 + \frac{1}{2} \right) - \frac{R_N x_0}{f} \\
E(d) &= \frac{R_N}{f} \frac{1}{2}
\end{aligned}
$$

$$
E(d) = \frac{R_N}{2f} \tag{3.2}
$$

$$
\begin{aligned}
E(d^2) &= \int_{x_0}^{x_0+1} \left(P_X - \tfrac{R_N x_0}{f}\right)^2 dx \\
E(d^2) &= \int_{x_0}^{x_0+1} P_X^2 - \tfrac{2R_N P_X x_0}{f} + \tfrac{R_N^2 x_0^2}{f^2} dx \\
E(d^2) &= \int_{x_0}^{x_0+1} Px^2 dx - \tfrac{2R_N x_0}{f} \int_{x_0}^{x_0+1} Px dx + \tfrac{R_N^2 x_0^2}{f^2} \int_{x_0}^{x_0+1} dx \\
E(d^2) &= E(Px^2) - \tfrac{2R_N x_0}{f} E(Px) + \tfrac{R_N^2 x_0^2}{f^2} \int_{x_0}^{x_0+1} dx \\
E(d^2) &= \left(\tfrac{R_N^2}{f^2}\left(x_0^2 + x_0 + \tfrac{1}{3}\right)\right) - \tfrac{2R_N x_0}{f}\left(\tfrac{R_N}{f}\left(x_0 + \tfrac{1}{2}\right)\right) + \tfrac{R_N^2 x_0^2}{f^2} \\
E(d^2) &= \tfrac{R_N^2}{f^2}\left(\left(x_0^2 + x_0 + \tfrac{1}{3}\right) - 2x_0\left(x_0 + 1/2\right) + x_0^2\right) \\
E(d^2) &= \tfrac{R_N^2}{f^2}\left(x_0^2 + x_0 + \tfrac{1}{3} - 2x_0^2 - x_0 + x_0^2\right) \\
E(d^2) &= \tfrac{R_N^2}{f^2}\left(\tfrac{1}{3}\right) \\
E(d^2) &= \tfrac{R_N^2}{3f^2}
\end{aligned}
$$

$$
\begin{aligned}
\sigma &= \sqrt{E(d^2) - E(d)^2} \\
\sigma &= \sqrt{\tfrac{R_N^2}{3f^2} - \left(\tfrac{R_N}{2f}\right)^2} \\
\sigma &= \sqrt{\tfrac{R_N^2}{f^2}\left(\tfrac{1}{3} - \tfrac{1}{4}\right)} \\
\sigma &= \tfrac{R_N}{f}\sqrt{\tfrac{1}{12}}
\end{aligned}
$$

$$\sigma = \frac{R_N}{f\sqrt{12}} \tag{3.3}$$

Or, in general terms, the mean and standard deviation of the error increase linearly with R_N (f being a constant).

3.1.4 Laser Range Finder

The ARP R-MAX has a Sick PLS-300 mounted on the nose. The Sick PLS-300 measures a sweep of laser range returns up to 320 m with 13 bit precision. The laser is mounted vertically so that the sweep progresses from above the horizon downward to the ground and past vertical. The laser is also tilted 27.2° above the aircraft level line. By mounting the laser vertically, the vision software can select the laser return which corresponds to the cameras' tilt angle. The PALACE project points the cameras 60° down from horizontal, and the landing site is selected. Then the aircraft descends

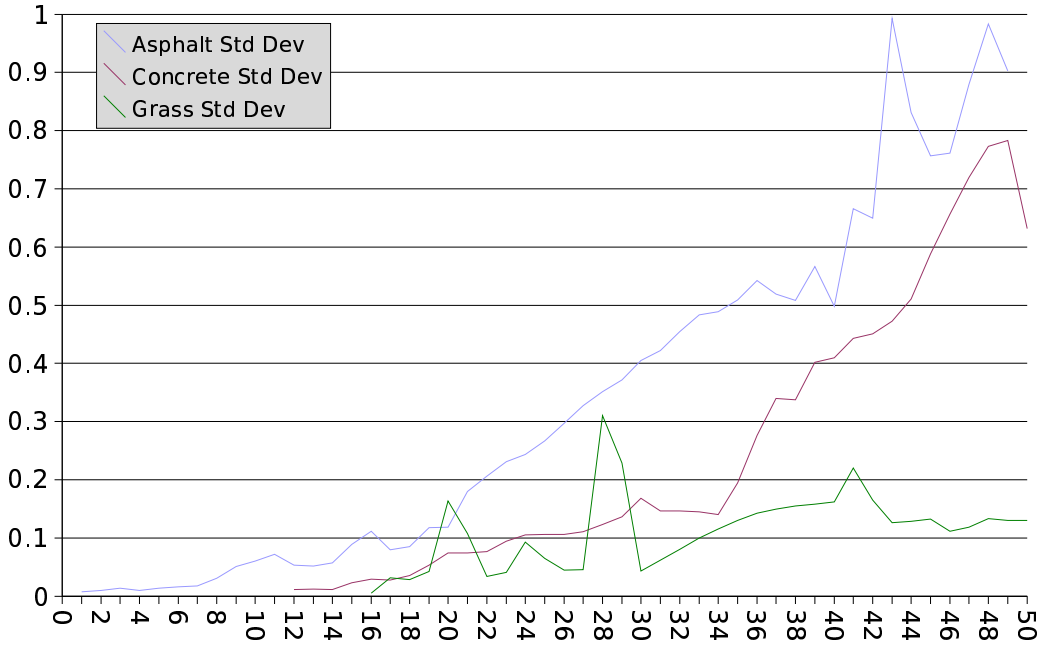


Figure 3.4: Laser measurement error increases proportionally with the distance of the surface being measured. This graph plots the standard deviation of the error for the three types of material the PALACE project used in experiments (concrete, asphalt, and grass).

along the 60° glide slope, which keeps the landing point in view for the entire descent (see Section 1.1). The laser return resolves the third unknown in the pinhole camera model (Equation 2.5 on page 29).

The PALACE project configures the PLS-300 to return no data outside an 80 m range, which is plenty for a landing from a height of 30 m AGL. Thus, the 13 bits of precision measure ranges to the centimeter. Sick laser rangefinders are used in many industrial and research robotics applications because of the rugged design and excellent error characteristics. Using flight data from hover flights at altitudes up to 30 m above ground, Figure 3.4 demonstrates the standard deviation of a range value reported by the Sick PLS-300. Data comes from flat surfaces of asphalt, concrete, and grass. Laser measurements to a range of 50 m are collected as the Sick PLS-300 scans from slightly behind the aircraft out to the horizon. By fitting a line with least square error to find the ground plane, each range reported by the Sick PLS-300 becomes an

error measurement. Atmospheric effects such as dust combine with erratic reflections from rough surfaces, like asphalt and grass, to drive the margin of error up. However, the laser range finder still provides very good data to 30 m above ground and higher. The PALACE project primarily needs data over asphalt, where the ARP R-MAX will land, and at a range of 30 m (the highest point in the PALACE mission) the returns will be within 40% of a standard deviation. The value of R_N is used directly in Equation 2.5 for P_X, so the noise on P_X will be the same as for R_N.

3.2 Safety Measures

Before switching to MPE, the Mission Manager allows the tracking algorithm to run for 2 seconds and monitors the coherence function. If the coherence function is high enough, the Mission Manager sends a request to the FCS to smoothly transition to MPE. The FCS monitors the position signal from MPE, which it receives for 2 seconds before the switching command. If at any point the position estimate deviates from 10 Hz, or indicates a change in position above a safe margin, the FCS assumes a tracking discontinuity and switches back to GPS. The FCS also monitors its internal performance with additional built-in safety measures if anything goes awry.

After the tracking algorithm begins operating, it monitors the tracked point, and if the point moves too close to the edge of the camera image, it switches to GPS. (It is impossible to track a point once it has gone off screen.) A safe soft abort maneuver allows the Mission Manager to switch to GPS and move away from the ground if a problem is detected while landing. The soft abort is a climb 5 meters straight up. This way, the aircraft does not hover near the ground for very long.

As the aircraft approaches the ground, the laser range finder reaches a minimum signal strength below which it cannot compute the distance to the ground. This typically happens at 0.7 m AGL. At higher altitudes, the asphalt may fail to generate a laser return, but the MPE algorithm searches over a 3° range for a return. If nothing is found, the software must soft abort. However, once the aircraft is below 1 m AGL, if the laser return is not received, the Mission Manager instructs the FCS to perform an autonomous landing (which represents success). From about 1 m AGL,

the FCS uses a sonar and inertial navigation computed from the IMU accelerometers to set the aircraft down. Weight-on-wheels switches sense when the skids touch the ground, and then the FCS stops the aircraft from flying.

These safety measures sufficiently address potential problems while the R-MAX flies without GPS control. To develop the software and test these safety measures, the position estimation algorithm was first operated in open loop mode (Chapter 4). When sufficient data had been collected to justify feeding the position estimate back to the FCS and actually disabling the GPS, a series of closed-loop tests were performed which ultimately verified the time delay and real-time performance of the system design (Chapter 5).

Chapter 4

Position Estimation Performance

The JPL initialization and tracking algorithms introduced in Section 2.2 have certain requirements for correct operation. The output of these algorithms feeds the position estimation algorithm (Section 2.2.3 on page 26). This chapter presents hardware tests to demonstrate acceptable performance of the initialization and tracking algorithms and the conditions required to achieve successful tracking.

4.1 Initialization Performance

The first of the JPL algorithms uses the Förstner operator to select the point to track. In this stage, the algorithm evaluates the landing surface texture for areas that will give a good motion estimate. Some surfaces suffer from overexposure, appearing washed out or completely white. Other surfaces, such as seams in concrete or lines of paint also have too little surface texture relative to the strong linear gradient. The Förstner operator primarily aims to avert the aperture problem (see Section 2.2.1 on page 19).

However, the second JPL algorithm can perform poorly even when initialized by the Förstner operator. If in the first 2 seconds of operation the correlation algorithm (Section 2.2.2 on page 22) reports a failure, the system repeats the initialization step to try again.

The field of view of the tracking camera offers plenty of room for the aircraft to maneuver when high above the ground. When the aircraft is about to land, the tracking camera allows for very little movement. Figure 4.1 shows a graphical representation of the field of view of the camera in simulation. As the aircraft nears the

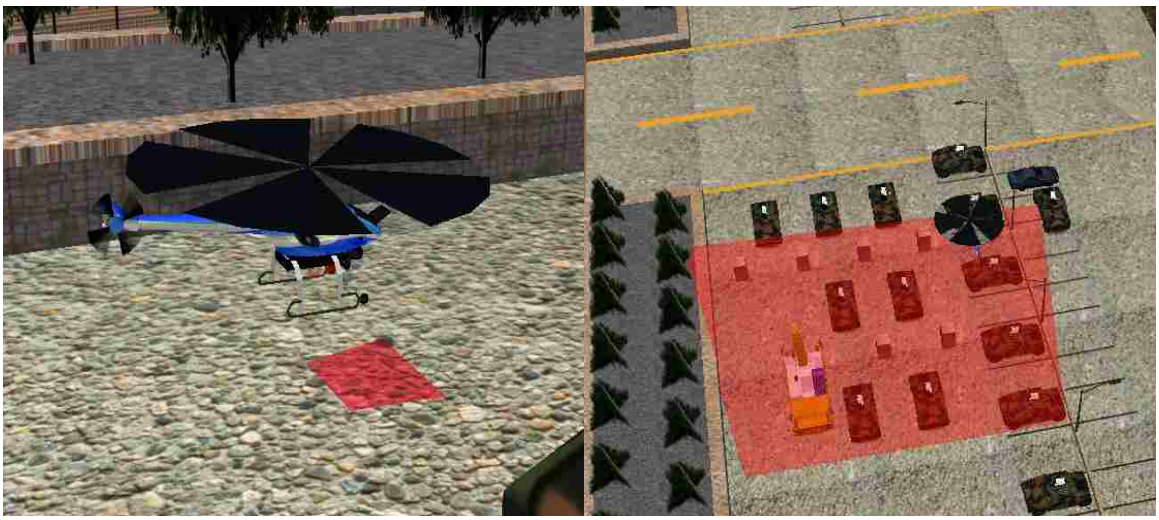


Figure 4.1: The decreasing field of view as the aircraft nears the ground will restrict its movement greatly. In the figure on the right, the aircraft hovers high above an obstacle field (in simulation). The aircraft can track any landing site in the red-hilighted field of view. In the figure on the left, the aircraft has selected the landing site north of the crane. Now that the aircraft is near the ground, very little motion in the field of view is possible before the target is out of sight.

ground, its movement is greatly restricted. The field of view expands outward from the camera, forming a truncated pyramid. If the camera is near the ground, the JPL tracking algorithm can only handle small movements before the tracked point has completely left the camera's field of view.

Since the landing is the most important part of the PALACE project, a modification made to the mission manager enables it to repeat the initialization process during flight. If the tracked point moves too near the edge of the camera image, the mission manager will choose a new point to track near the center of the image so that the aircraft can move more freely while landing. The decision to use this technique came after a consideration of the noise characteristics inherent in recalculating the ground reference point mid-flight. Each time the initialization is repeated, the drift error is incorporated into the ground reference point; therefore, by only reinitializing once or twice near the end of the landing, this error is minimized.

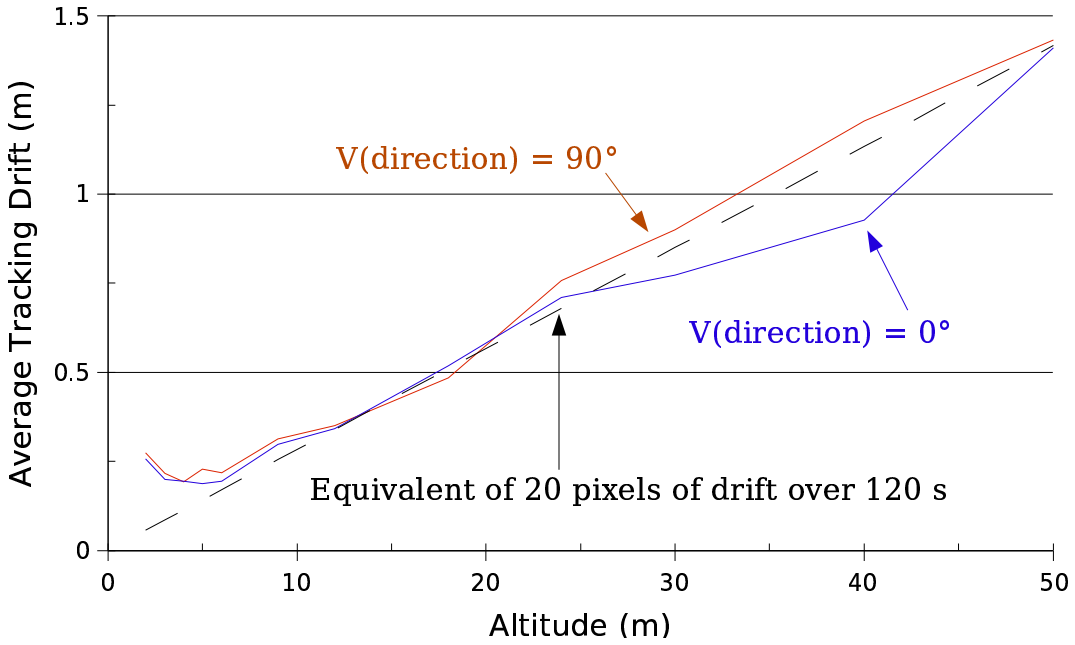


Figure 4.2: Simulation indicates an average of 20 pixels drift during a 120 s hover flight. By repeating the 2 min hover multiple times at many altitudes, an average drift rate can be estimated.

4.2 Tracker Drift

Drift refers to the cumulative rounding error from each iteration of the correlation algorithm (Section 2.2.2 on page 22) which causes the position estimate to gradually diverge from the original landing site. Drift originates in the camera image, so it follows from Equation 2.5 that the noise and cumulative error are scaled by R_N, the distance to the landing site measured by the laser range finder.

Figures 4.2 and 4.3 demonstrate simulation work that characterizes the drift rates of the corelation algorithm over a 2 minute period. Simulation data shows a drift rate of 20 pixels average (Figure 4.2) and 34 pixels maximum (Figure 4.3) over 2 minutes (1200 frames at 10 Hz). 30 runs are averaged in each data point. For example, the maximum drift at 50 m AGL is the average of the maximum drift over 30 runs simulated with the UAV facing into the wind ($V = 0°$), for an average of 2.28 m drift, and 30 runs simulated sideways ($V = 90°$), for an average of 2.48 m drift. Simulations used the Dryden wind model with gusts of varying intensity.

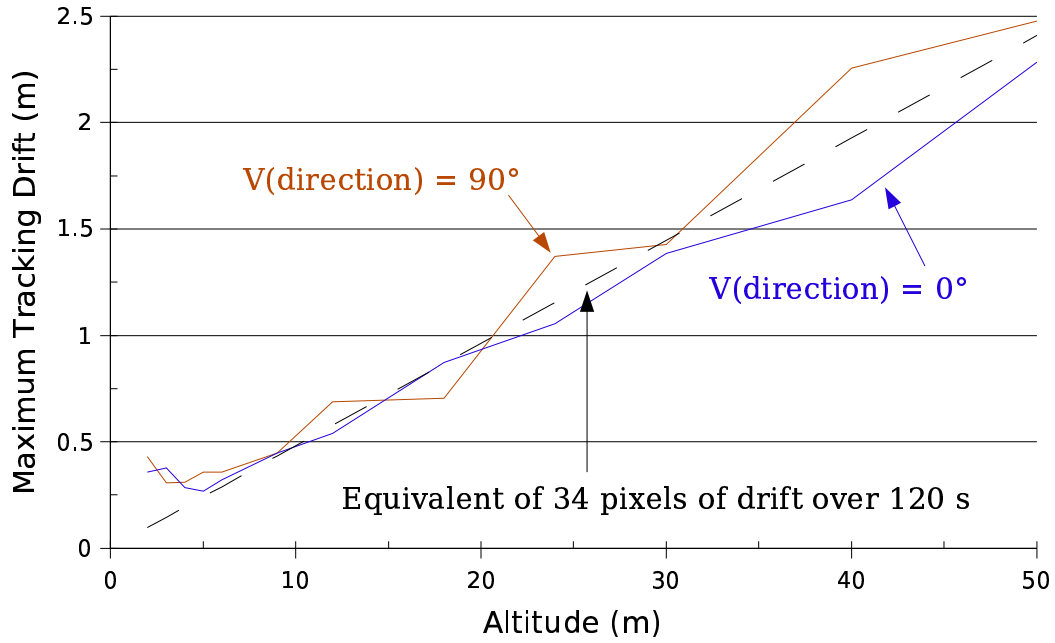


Figure 4.3: Simulation indicates a maximum of 34 pixels drift during a 120 s hover flight. By repeating the 2 min hover multiple times at many altitudes, an upper bound on drift can be estimated. The drift process during a single flight is random, but the largest deviation from the actual position of the UAV for each flight is used to compute the maximum drift for various altitude and wind conditions.

These simulations all measure drift using the same texture; the simulation design prevents the tracker from encountering tracking failures (such as a discontinuity in tracking). Similar data gathered in flight over asphalt, concrete, and grass, confirms that the drift rate in simulation is a reasonable figure. The flight data lacks the exhaustive repetitions used in simulation to reduce the margin of error; however, the simulation did not model all the sources of noise present in flight, as discussed in Chapter 3. The flight data demonstrates more noise than simulation, but the maximum observed drift rate is lower.

4.2.1 Tracker Drift In Flight

Flight data shows drift rates of at most 22.8 pixels (Figure 4.4). Table 4.1 on page 51 also shows the same data in detail, summarizing the drift rates over three surfaces: concrete, asphalt, and grass. As mentioned previously, noise from the

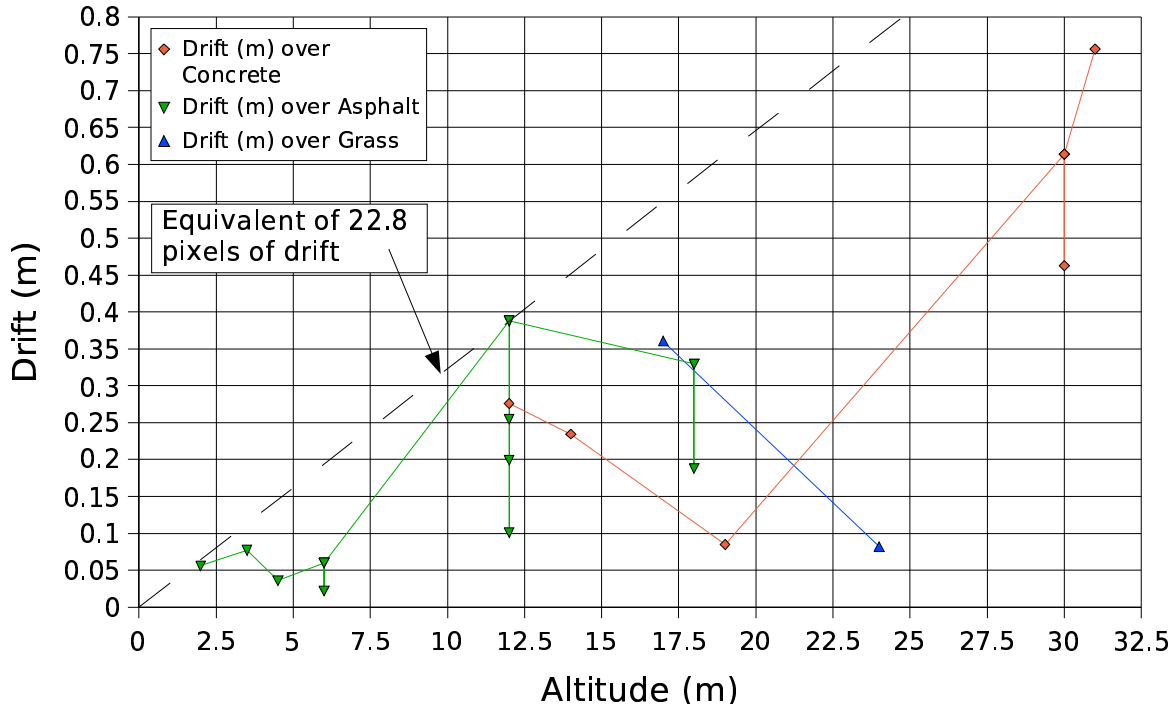


Figure 4.4: Flight data drifts less in 120 s than simulation data. (22.8 pixels in 120 s is less than 34 pixels in 120 s.) Data points over concrete, asphalt, and grass are plotted, along with an envelope for each type of surface. Multiple runs over asphalt provide a range of drift measurements at 6, 12, and 18 m AGL. Multiple runs over concrete at 30 m AGL provide a similar range.

laser range finder and the correlation algorithm increase linearly with altitude (see Figures 4.5, 4.6, 4.7).

The drift process is not a monotonically increasing function. Drift in one direction can be cancelled by drift in the opposite direction. Drift is also not a linear process. Statistically, however, there is a greater deviation from the original point as time passes; thus, to measure the amount of drift during a flight, at each point in time (10 Hz) the vector from GPS to the position estimate is computed. A line is fit to this set of vectors using a linear least-square fit. The 10 points farthest from the line are recursively discarded while fitting the line. The 2-norm distance from the initial and final points on this line gives the drift distance.

When there is not 120 s of data for a flight, the drift in meters can be extrapolated linearly to 120 s to provide a basis for comparison. None of these shorter runs affected the maximum drift rate, which occured at 12 m AGL over asphalt and had the equivalent of 22.8 pixels of drift (in Table 4.1, the drift is not extrapolated, but the equivalent drift in pixels is; in Figure 4.4 the drift is extrapolated for ease of comparison). If anything, such a linear extrapolation overestimates the amount of drift for a flight.

4.3 Lighting

The Point Grey Flea cameras used by the PALACE project automatically adjust to lighting conditions. For a vision project the automatic gain might introduce variations and unpredictable behavior. Some tests were performed with the Flea cameras set in a fixed-gain mode, but it was determined that the cameras operate best when they can automatically adjust to lighting conditions.

The PALACE flights all occured out of doors in daytime lighting conditions, sometimes in bright sunlight and sometimes in overcast conditions. Experiments in indoor lighting showed that although vision algorithms could operate indoors, the characteristics of the cameras would introduce unexpected behavior. The first effect of low light would be motion blurring (see Figure 4.8). The CCD in the cameras automatically adjusts the integration time from microseconds up to about a millisecond. This is significant because the vibration of the UAV platform can change the orientation of the camera in that much time, which causes blurring in the image. The cameras are mounted on a vibration-isolated wing which reduces the angular motion of the camera, but the entire UAV still translates with the vibration, and this can be seen at low altitudes, even in daylight conditions. The images in Figure 4.8 were taken at 0.5 m AGL. This is below the normal operating range of the vision algorithms, and so does not effect the success of the PALACE project. However, indoor lighting conditions would require a larger integration time by approximately a factor of 10; then motion blur would be significant, especially since indoor spaces tend to be smaller.

Date	AGL	Surface	Time (s)	Drift (m)	σ	Equiv. Drift
Jul 12, 2005 10:53am	30 m	Concrete	120 s	0.4626	0.0635	10.882 pixels
Aug 18,2005 09:49am	30 m	Concrete	120 s	0.6137	0.0810	14.438 pixels
Oct 19, 2005 11:50am	31 m	Concrete	76.2 s	0.4802	0.0761	17.217 pixels
Aug 18, 2005 09:52am	19 m	Concrete	120 s	0.0848	0.0581	3.152 pixels
Sep 09, 2005 09:24am	14 m	Concrete	120 s	0.2346	0.0434	11.829 pixels
Aug 18, 2005 09:32am	12 m	Concrete	34.3 s	0.0789	0.0227	16.229 pixels
Sep 20, 2005 09:30am	18 m	Asphalt	120 s	0.3295	0.0507	12.917 pixels
Nov 02, 2005 09:26am	18 m	Asphalt	120 s	0.1880	0.0541	7.373 pixels
Sep 20, 2005 09:21am	12 m	Asphalt	120 s	0.1011	0.0325	5.945 pixels
Oct 19, 2005 10:50am	12 m	Asphalt	120 s	0.2548	0.0289	14.986 pixels
Oct 19, 2005 10:57am	**12 m**	**Asphalt**	**120 s**	**0.3879**	**0.0481**	**22.812 pixels**
Nov 08, 2005 11:24am	12 m	Asphalt	120 s	0.1991	0.0354	11.710 pixels
Sep 20, 2005 09:26am	6 m	Asphalt	120 s	0.0600	0.0218	7.057 pixels
Oct 19, 2005 11:05am	6 m	Asphalt	120 s	0.0217	0.0176	2.560 pixels
Oct 19, 2005 11:09am	4.5 m	Asphalt	120 s	0.0360	0.0152	5.650 pixels
Oct 19, 2005 11:12am	3.5 m	Asphalt	120 s	0.0769	0.0164	15.497 pixels
Oct 19, 2005 11:17am	2 m	Asphalt	120 s	0.0560	0.0224	19.770 pixels
Nov 10, 2005 10:45am	24 m	Grass	107.6 s	0.0735	0.0793	2.411 pixels
Nov 10, 2005 10:53am	17 m	Grass	72.1 s	0.2167	0.1711	14.971 pixels

Table 4.1: Flight data shows a 22.8 pixel maximum drift rate.

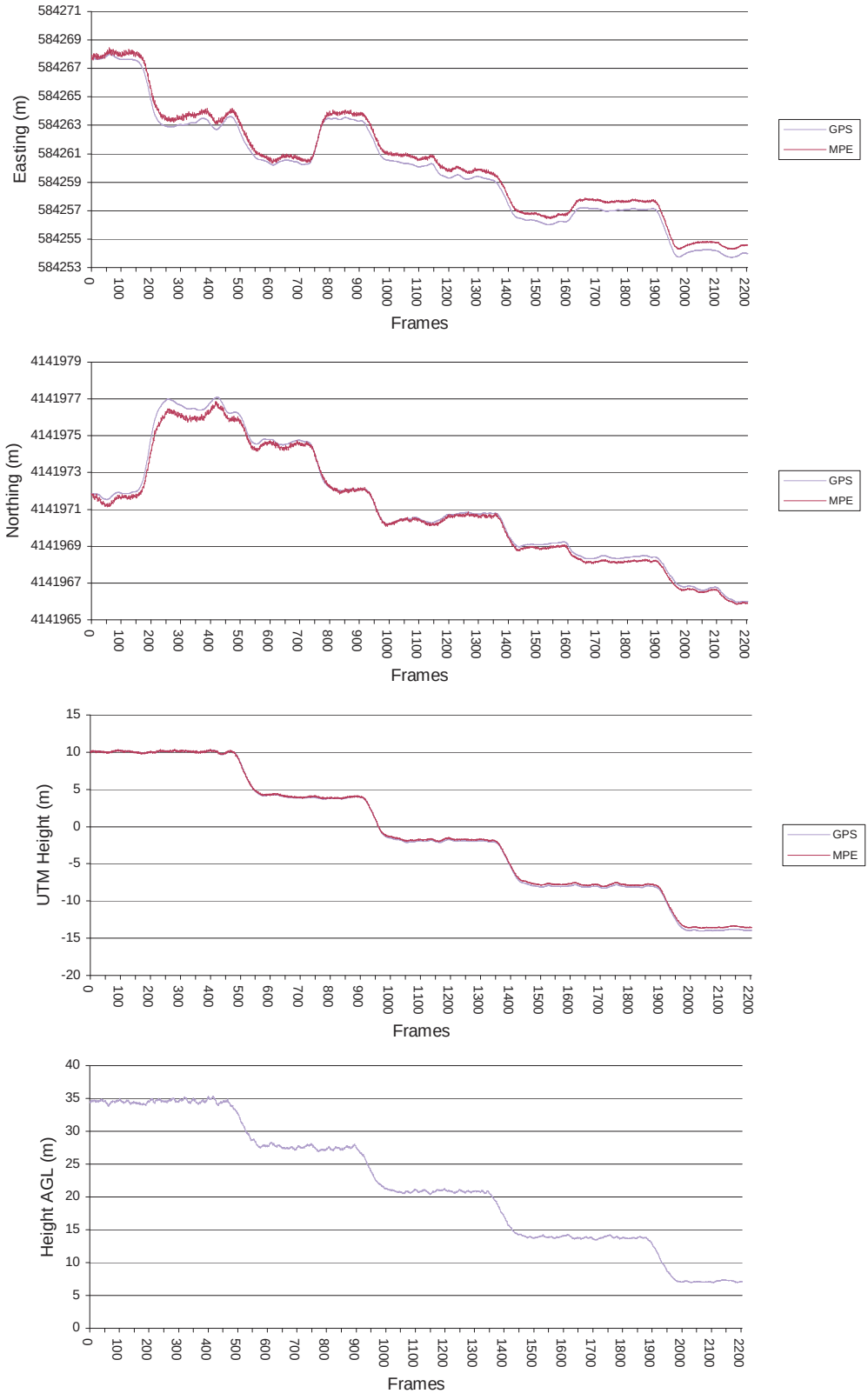


Figure 4.5: Position estimation over concrete shows a gradually increasing drift, and a zero-mean noise proportional with altitude.

Figure 4.6: Position estimation over asphalt shows a gradually increasing drift, and a zero-mean noise proportional with altitude.

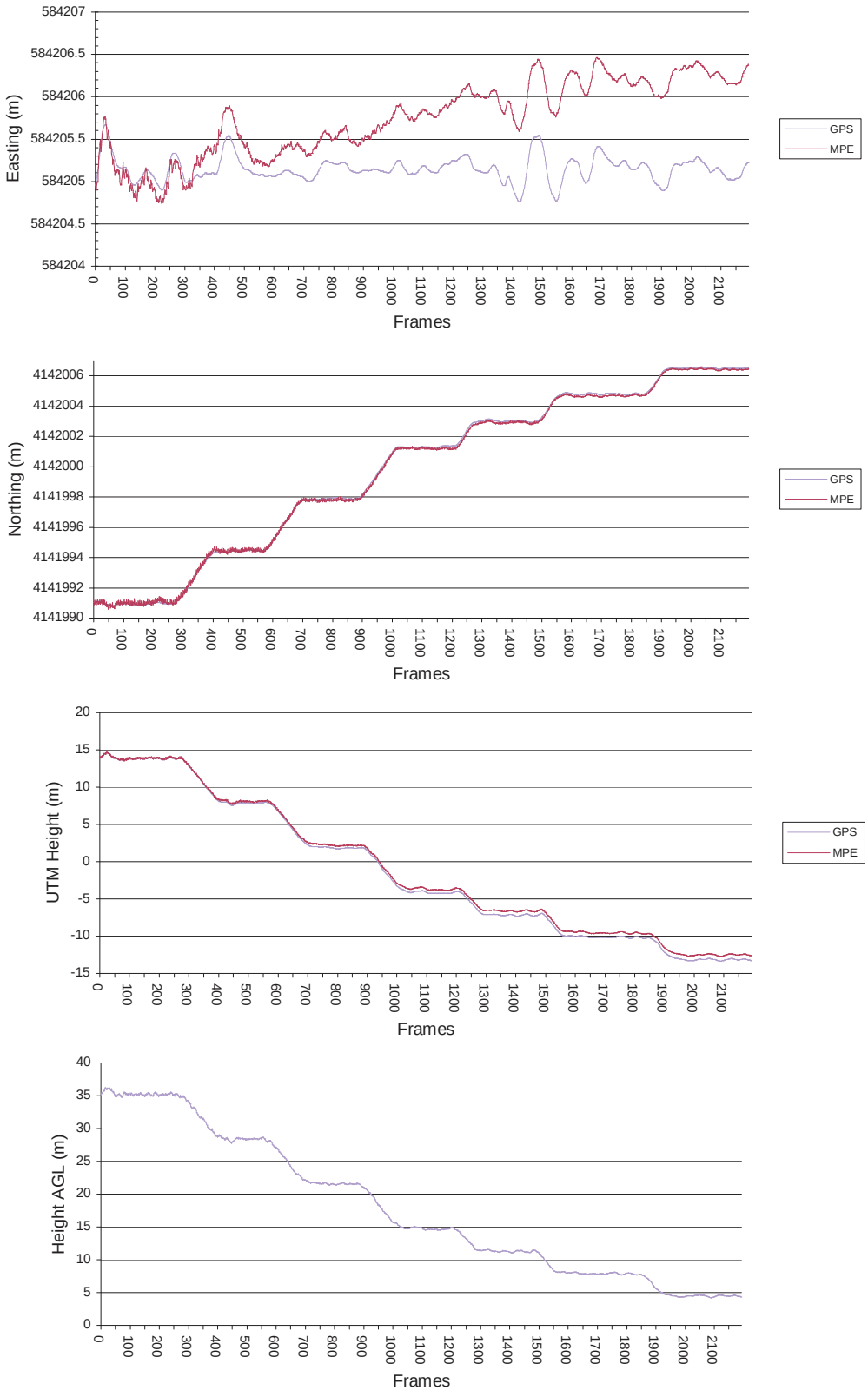
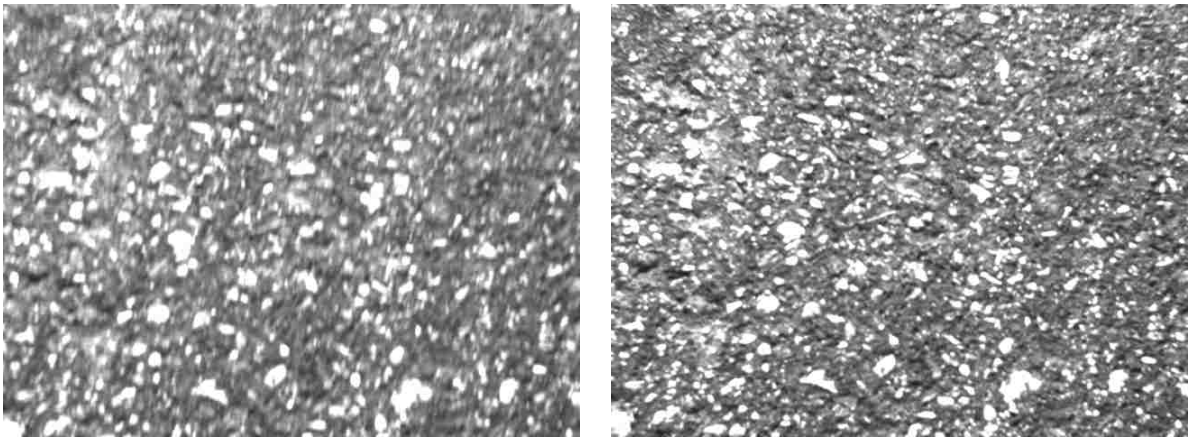


Figure 4.7: Position estimation over grass shows a gradually increasing drift, and a zero-mean noise proportional with altitude.

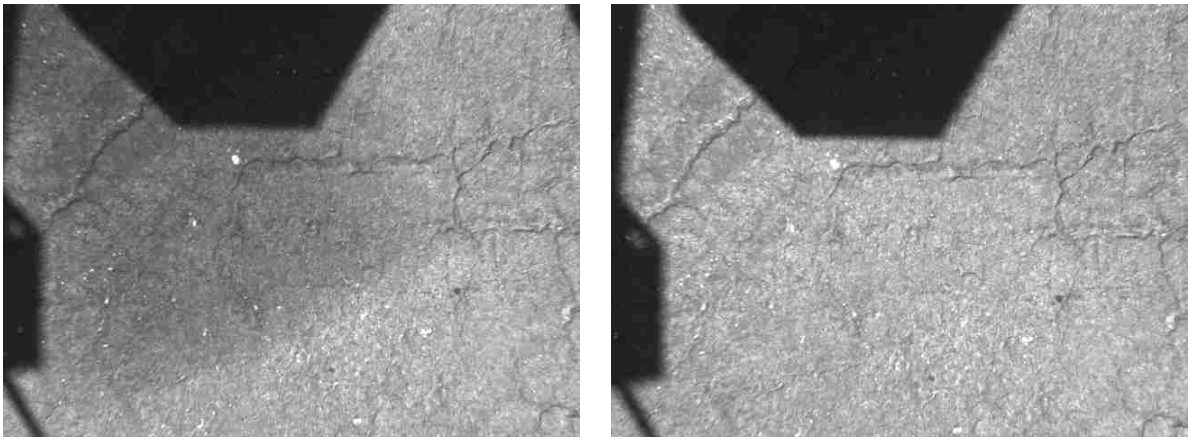


Frame 4465 (blurred) Frame 4482

Figure 4.8: Motion blur occurs in daylight conditions below 0.5 m AGL. In artificial lighting, motion blur occurs at larger distances. Motion blur can quickly degrade tracking performance.

Motion blur adversely effects the tracking algorithm because it blurs away the small details which the tracking algorithm uses to follow the movement in the image. The algorithm compensates for the mean brightness in the image, so general changes in brightness are not a factor in tracking performance. But the tracking algorithm uses the image texture to estimate the image motion.

The automatic gain control algorithm in the Flea cameras produces another effect. The algorithm increases the integration time and gain until a few of the pixels captured by the camera are at maximum intensity (255). This way, the dynamic range of the image is not lost. But the response of the CCD near full saturation is somewhat nonlinear, and the pixels around the brightest spots in the image do not have sufficient texture for the tracking algorithm. This loss of information in areas that might otherwise have been effective areas to track degrades the overall performance of the tracking algorithm. Initial experiments that program the gain and integration time of the cameras to fixed values show that these brightest points were selected by the Förstner Interest Operator (Section 2.2.1) as good points to track.



Frame 311 (rotor, body shadow visible) Frame 312 (body shadow visible)

Figure 4.9: Shadows affect tracking performance. Near the ground, the shadow cast by the UAV interferes with the tracking algorithm, even when navigating to minimize the amount of shadow visible by the cameras.

A third effect related to lighting occurs when the UAV nears the ground. By calculating the position of the sun and wind conditions, the UAV's shadow can be deliberately placed out of the way while landing. This assumes that the UAV has some freedom to approach the landing site from any direction. However, once the UAV nears the ground, its shadow might cross the image (see Figure 4.9). Because the shadow is dark enough, the tracker becomes confused by the intensity gradient and drifts from the original tracking point. This effect occurs very close to the ground, so the UAV has already transitioned to inertial navigation (integrating the accelerometers to estimate position) and will not be affected. However, the effect of the shadow is magnified by the limited dynamic range of the CCD sensor.

4.4 Conclusions

This chapter provides insight into the operation of the vision-based navigation system and how closely it compares to a GPS sensor. Simulation and flight data provide evidence that the system can operate over the duration of an autonomous landing.

Several limitations of the system could become future research topics. Controlling the automatic gain and integration time of the CCD would show modest improvements by reducing the drift rate and incidence of tracking discontinuities. Working around the limitations of CCD cameras could expand the flight conditions where the UAV would successfully land to include low light conditions and landing paths where the UAV's shadow might interfere. Integrating night vision cameras would allow operation in extreme low-light conditions, though the sensors would pick up more interference in the image and might require larger a template window. An automatic transition could be developed to enable the night-vision mode when the ambient light is low. New CMOS cameras also have a much better signal to noise ratio in low-light conditions and have an exponential response to light intensity, reducing the need for automatic gain controls and the effect of shadows. A measurement of the sharpness of a shadow or the intensity of a backlight could quantify the effect of shadows on tracking performance, leading to a more formalized system of evaluating UAV tracking performance.

Chapter 5

Closed-Loop Results

This chapter builds on the tracking results of Chapter 4 by showing the importance of synchronizing the separate pieces of the vision-based navigation system. In simulation and in flight, the time synchronization and accurate identification of inherent delays proved the key to achieving the PALACE goal of autonomous navigation to the landing point. Slight variations in time synchronization are not immediately obvious in open loop hover tests, such as preformed in Chapter 4, and only show up in closed-loop operation. Because the PALACE approach fuses the data from multiple sensors, accurate identification of the frequency response and phase delay of each sensor, so that the sensor inputs have an appropriate transformation to compensate for their response characteristics, significantly improves lateral and longitudinal stability of the control system. The MPE algorithm then performs close to GPS levels.

5.1 Modeling Time Delays

The full-mission simulation (Section 2.3 on page 30) demonstrated the importance of good synchronization between processes. Initial work on the vision-based navigation had errors in synchronization with the RIPTIDE environment which could cause the tracking algorithm to use old attitude information. The effect was instability when the simulation operated in closed-loop mode at 18 m AGL or higher.

Formally, the inner and outer loops of the control system should be independent of each other. The inner loop samples data at 50 Hz, while the outer loop samples at 10 Hz (Figure 2.6 on page 26). However, the monocular position estimation algorithm uses the pinhole camera model to estimate position. The camera takes

measurements relative to the aircraft, which makes it necessary to rotate the results using the attitude *at the time the image was taken* to obtain a position estimate in the world reference frame. The initial simulation results demonstrated instability by using attitude measurements with an induced delay, which was not consistent with the rotation in the simulated camera image. The position control is robust to a phase delay on the order of hundreds of milliseconds, due to its longer response time. However, the result of the position loop calculations is fed into the attitude loop. The attitude loop is more sensitive to a delayed, fed-back signal. The attitude loop becomes unstable when the delay is magnified by about 18 m (see Equation 2.5 on page 29) and the R-UAV oscillates out of control. Attitude should not enter the attitude loop through the slower position loop, or through a delay caused by poorly synchronized attitude measurements used in the MPE algorithm. The attitude loop is designed to operate independently of position.

In the same vein, the SiCK PLS-300 laser measurement may also have a delay. The Flea cameras are the least delay-prone, operating at 30 Hz (so the CCD sensor will integrate for 33 ms at most, but this value is typically around 1 ms in daylight conditions). The fast camera response becomes advantageous because the tracking algorithm can complete its operations in about 10 ms. If any of the other two sensors (laser or IMU) is delayed by more than 10 ms, the tracking algorithm is no longer the principal source of delay in computing the position estimate. This implies that the MPE algorithm is not CPU-bound, and that more CPU-intensive tracking algorithms could be used without any degradation in closed-loop performance.

5.2 Stability Margins

By simulating the aircraft in the RIPTIDE simulator, it becomes possible to visualize the effects of altitude and poor synchronization on the stability of the attitude control loop. The lateral and longitudinal cyclic control axes are the most sensitive to MPE error, so performance metrics for these axes (Figure 5.1) are used to demonstrate the overall system stability.

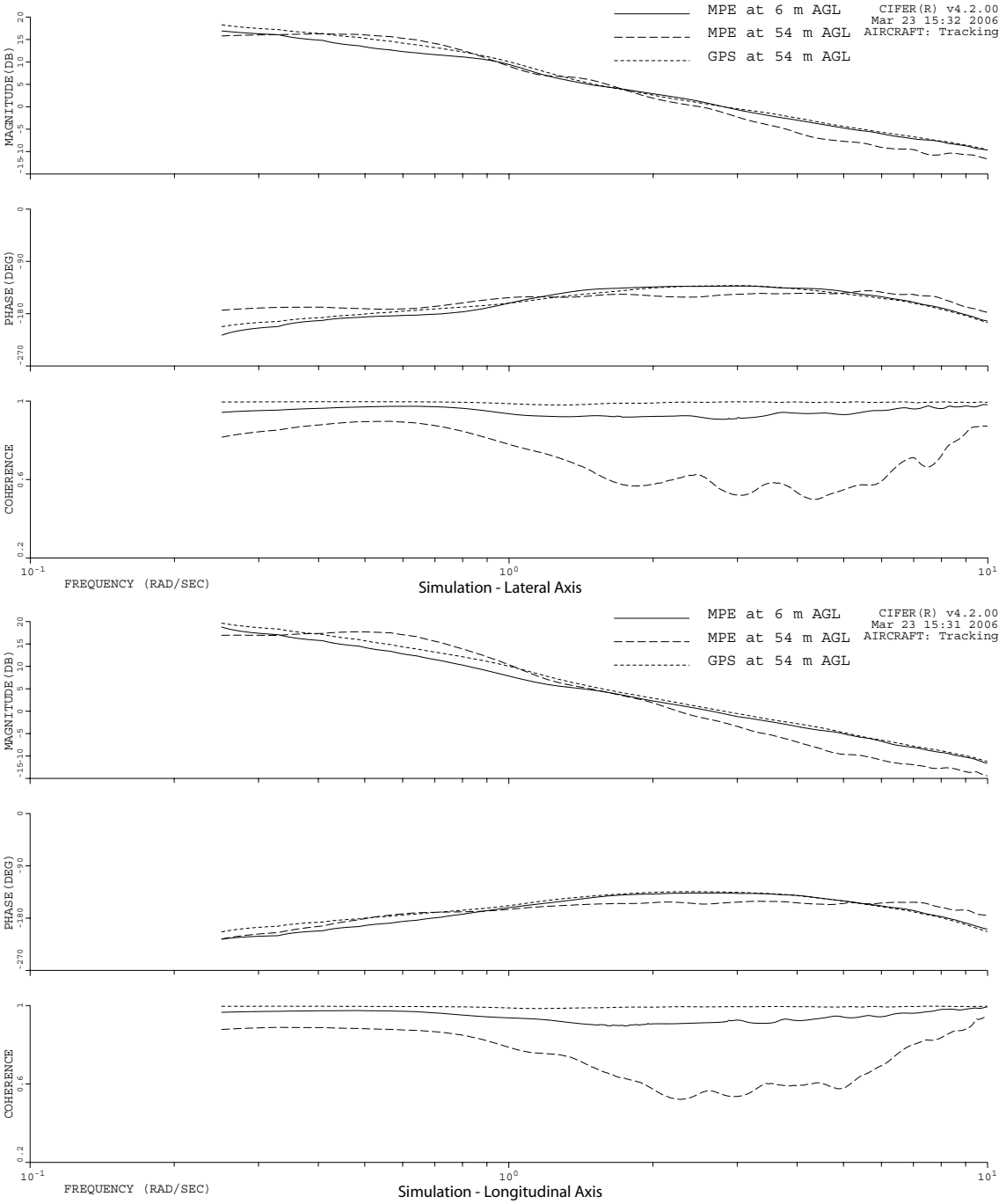


Figure 5.1: Sweeps of the lateral and longitudinal cyclic actuators performed in simulation give insight into the effect of increasing altitude on the stability of the system. Even well-synchronized attitude information is not perfect. GPS control is not affected by altitude in simulation. The simulation at 54 m AGL produces a lateral phase margin of 48.09° at 2.81 rad/s and gain margin of 8.26 dB at 8.67 rad/s. The longitudinal phase margin is 44.97° at 2.81 rad/s, and the longitudinal gain margin is 8.91 dB at 8.07 rad/s.

Figure 5.1 gives plots for lateral and longitudinal control sweeps performed in the RIPTIDE simulator in closed-loop MPE mode, and compares it side-by-side with perfect information as might be received in GPS mode. The control loops are broken at the actuator, the final output of the control law. A sweep provides useful information in several ways. The sweep function starts with a low-frequency sine wave and gradually increases until the sine wave is well above the observed natural frequencies of the aircraft (see Section 5.2.1). This function is added to the control loop at the point where the loop is broken, in this case at the actuator. The sweep generates the transfer function of the system which can be analyzed by a frequency response tool such as CIFER [19] (Comprehensive Identification of Frequency Response). The system is considered marginally stable when the gain and phase margin are positive, and acceptable levels of stability are around a phase margin of 45° for lateral and longitudinal response of the aircraft.

In Table 5.1, step-by-step simulation of synchronized and delayed attitude data in closed-loop MPE mode demonstrate the diminishing stability margins caused by poor synchronization. The MPE data with a 100 ms delay becomes unstable above 18 m AGL.

Thus motivated, experiments with the R-MAX UAV demonstrated the necessity of identifying the frequency response for each sensor. There are several methods for performing a frequency sweep. First, a sweep signal was induced on the aircraft on the ground and sensor data was collected from the camera, the laser, and the IMU. (These are the inputs to the MPE algorithm.) Figure 5.2 correlates the camera to the laser. Both report the pitch of the aircraft without any delay. Figure 5.3 (camera vs. IMU) and Figure 5.4 (laser vs. IMU) show the IMU reporting data 44 ms late.

This data is further supported by in-flight sweeps. The sweep is added to the actuator command in software, and the data is collected for later analysis in CIFER. Figure 5.5 compares the performance of the aircraft (lateral control) without any correction for the delay of the IMU data and with a 43 ms delay added to camera and laser data to agree with IMU data. Figure 5.6 compares the longitudinal control without any correction and with a 43 ms correction. As seen in the figures, and as

AGL	Delay	Lateral Phase Margin	Lateral Gain Margin	Longitudinal Phase Margin	Longitudinal Gain Margin
6 m	0 ms	47.37° at 2.81$\frac{rad}{s}$	8.60 dB at 8.83$\frac{rad}{s}$	43.20° at 2.65$\frac{rad}{s}$	9.42 dB at 8.26$\frac{rad}{s}$
12 m	0 ms	45.25° at 2.60$\frac{rad}{s}$	9.00 dB at 8.94$\frac{rad}{s}$	42.45° at 2.53$\frac{rad}{s}$	10.15 dB at 8.39$\frac{rad}{s}$
18 m	0 ms	44.05° at 2.65$\frac{rad}{s}$	9.45 dB at 9.10$\frac{rad}{s}$	41.23° at 2.67$\frac{rad}{s}$	10.60 dB at 8.49$\frac{rad}{s}$
24 m	0 ms	44.02° at 2.59$\frac{rad}{s}$	9.73 dB at 9.18$\frac{rad}{s}$	38.15° at 2.53$\frac{rad}{s}$	11.17 dB at 8.61$\frac{rad}{s}$
30 m	0 ms	42.56° at 2.54$\frac{rad}{s}$	10.69 dB at 9.59$\frac{rad}{s}$	36.91° at 2.31$\frac{rad}{s}$	12.23 dB at 9.48$\frac{rad}{s}$
36 m	0 ms	35.04° at 2.47$\frac{rad}{s}$	10.73 dB at 9.62$\frac{rad}{s}$	31.92° at 2.32$\frac{rad}{s}$	13.04 dB at 9.31$\frac{rad}{s}$
42 m	0 ms	32.77° at 2.53$\frac{rad}{s}$	10.97 dB at 9.65$\frac{rad}{s}$	28.80° at 2.39$\frac{rad}{s}$	13.33 dB at 9.63$\frac{rad}{s}$
48 m	0 ms	30.86° at 2.46$\frac{rad}{s}$	11.90 dB at 10.22$\frac{rad}{s}$	28.00° at 2.28$\frac{rad}{s}$	13.69 dB at 10.17$\frac{rad}{s}$
54 m	0 ms	28.76° at 2.19$\frac{rad}{s}$	11.07 dB at 9.68$\frac{rad}{s}$	29.51° at 2.35$\frac{rad}{s}$	13.48 dB at 9.47$\frac{rad}{s}$
6 m	100 ms	39.54° at 2.31$\frac{rad}{s}$	9.91 dB at 9.92$\frac{rad}{s}$	36.14° at 2.37$\frac{rad}{s}$	11.59 dB at 9.57$\frac{rad}{s}$
9 m	100 ms	32.67° at 2.29$\frac{rad}{s}$	10.36 dB at 10.44$\frac{rad}{s}$	31.45° at 2.29$\frac{rad}{s}$	12.22 dB at 10.21$\frac{rad}{s}$
12 m	100 ms	27.38° at 2.27$\frac{rad}{s}$	10.43 dB at 11.05$\frac{rad}{s}$	22.21° at 2.15$\frac{rad}{s}$	12.29 dB at 10.88$\frac{rad}{s}$
15 m	100 ms	24.06° at 2.22$\frac{rad}{s}$	10.18 dB at 11.38$\frac{rad}{s}$	19.46° at 2.16$\frac{rad}{s}$	11.79 dB at 11.32$\frac{rad}{s}$
18 m	100 ms	14.72° at 2.17$\frac{rad}{s}$	9.71 dB at 11.65$\frac{rad}{s}$	11.66° at 2.20$\frac{rad}{s}$	11.27 dB at 11.76$\frac{rad}{s}$

Table 5.1: Sweeps of the lateral and longitudinal cyclic actuators performed in simulation under MPE control give insight into the effect of increasing altitude on the stability of the system. If the attitude information is correctly synchronized, the effect is minimal. When the attitude information is poorly synchronized, the effect of altitude becomes very significant.

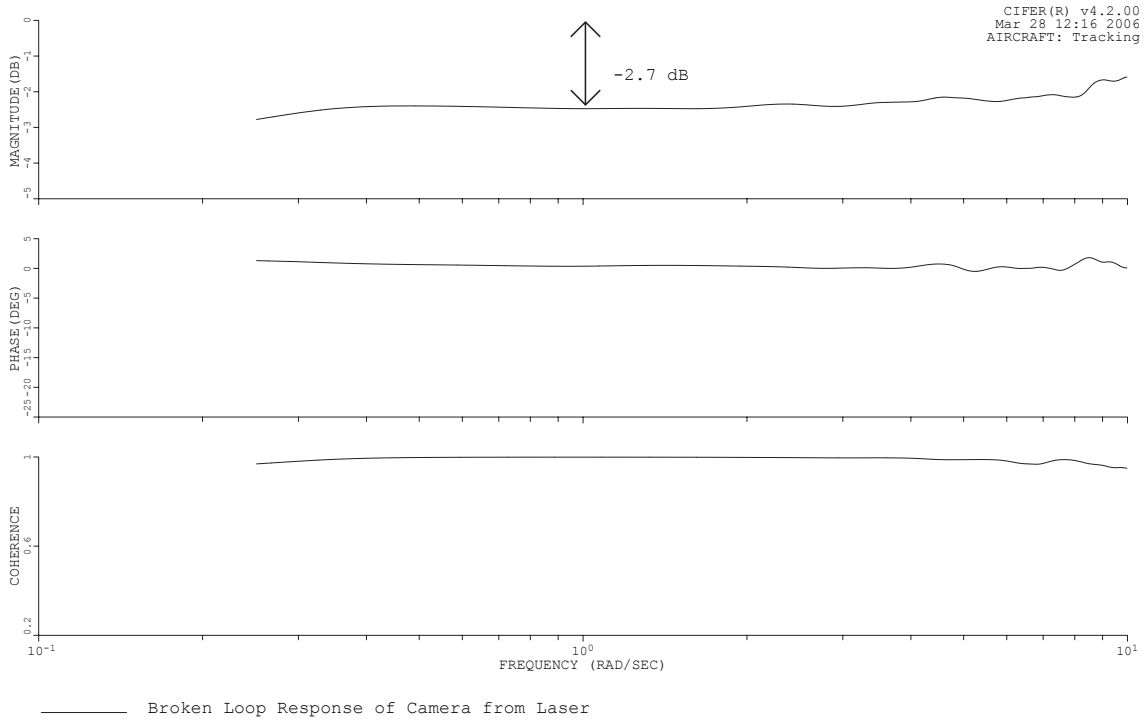


Figure 5.2: A sweep on the pitch axis comparing the camera and the laser sensor shows a flat gain and phase curve, indicating no delay between the two sensors. The camera has 2.7 dB more gain due to its field of view.

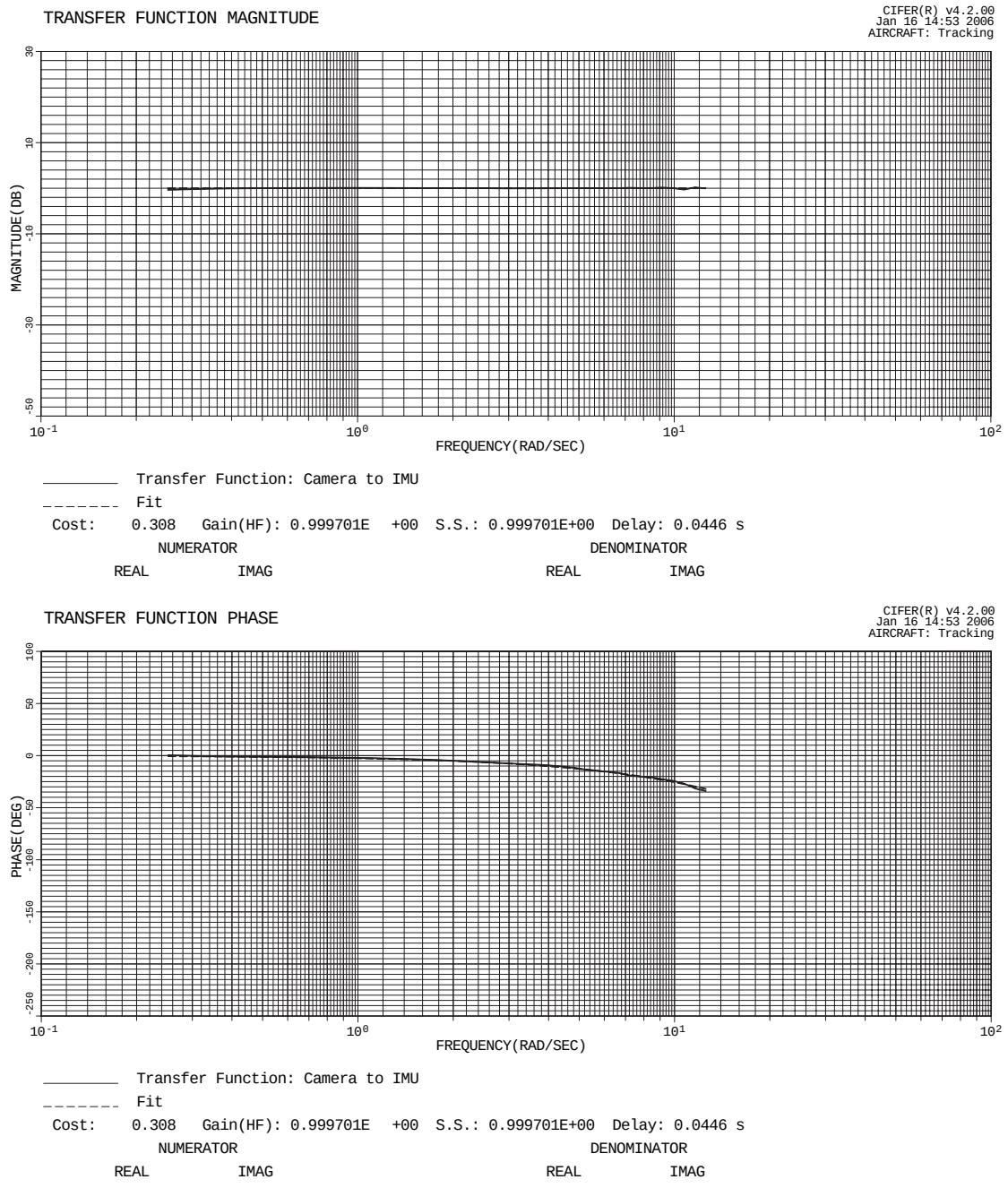


Figure 5.3: A sweep on the pitch axis comparing the camera and the IMU shows a gain and phase curve consistent with a 44 ms delay.

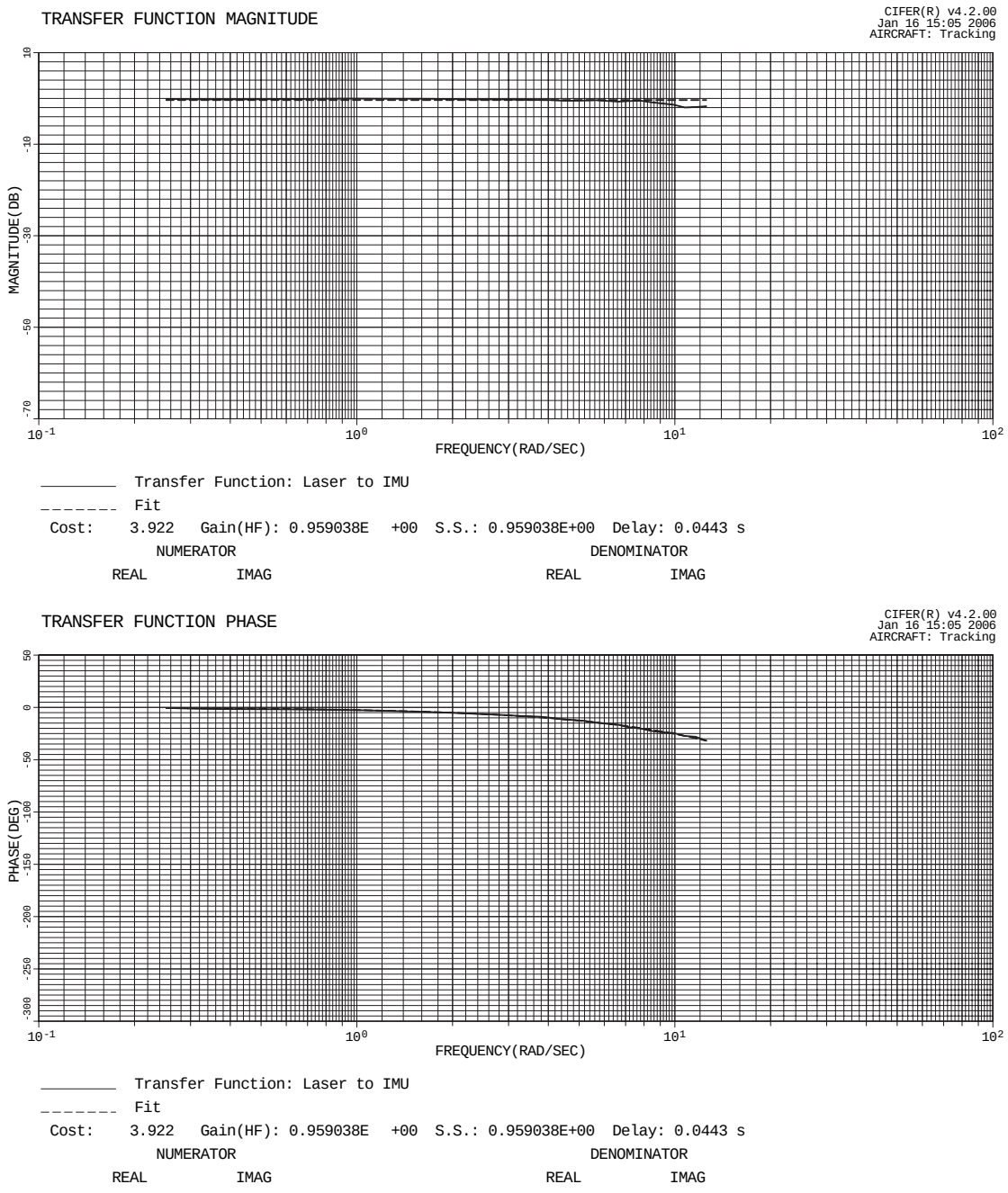


Figure 5.4: A sweep on the pitch axis comparing the laser and the IMU shows a gain and phase curve consistent with a 44 ms delay.

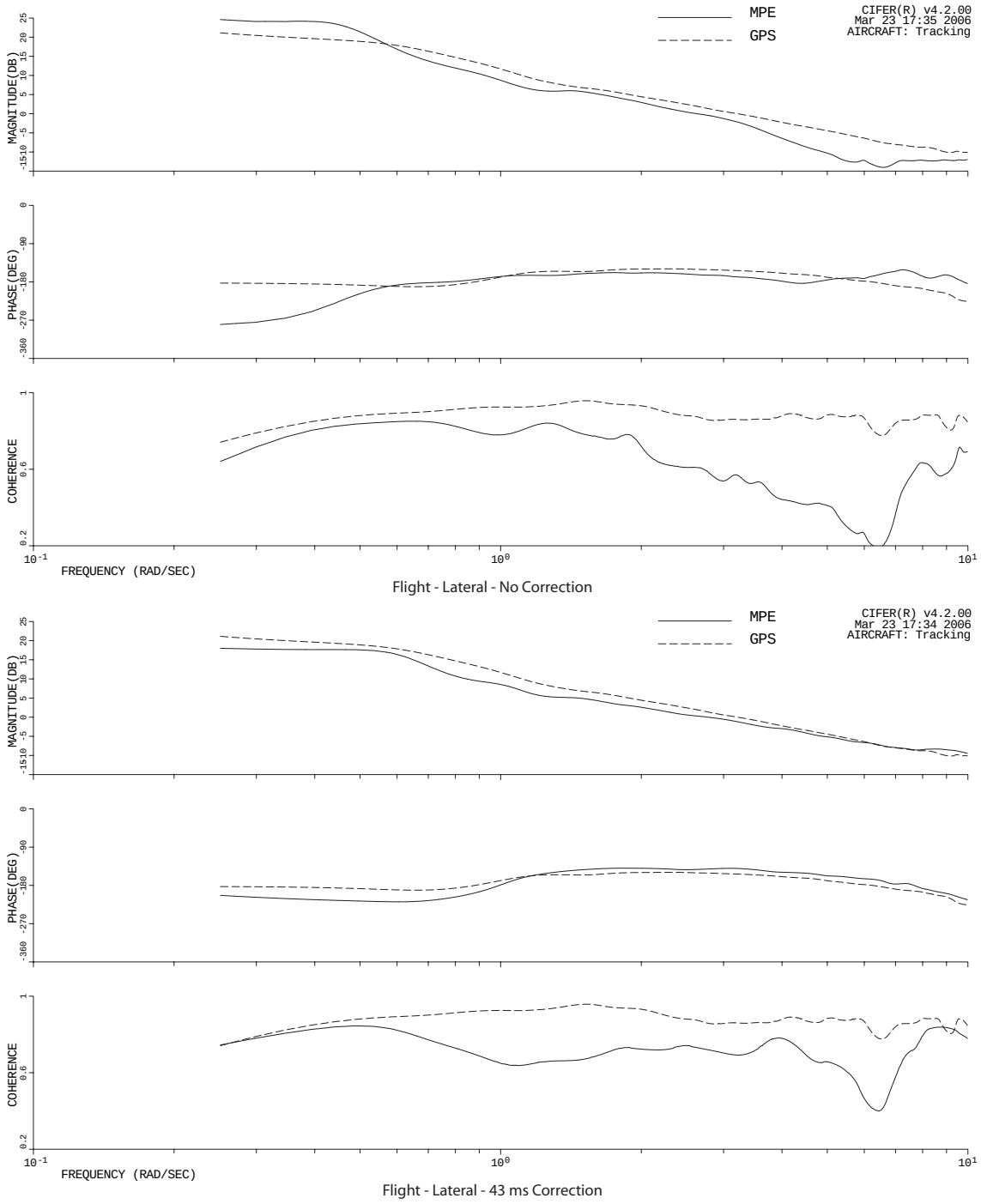


Figure 5.5: A sweep of the lateral cyclic control without any attitude correction (top) demonstrates a borderline stability margin (17.01° at $2.63\frac{rad}{s}$) when the MPE algorithm is in closed-loop mode. The bottom figure shows the same type of sweep with the 43 ms attitude correction added to correctly synchronize the inputs to the MPE algorithm. Its stability margin (38.50° at $2.78\frac{rad}{s}$) is close to GPS levels.

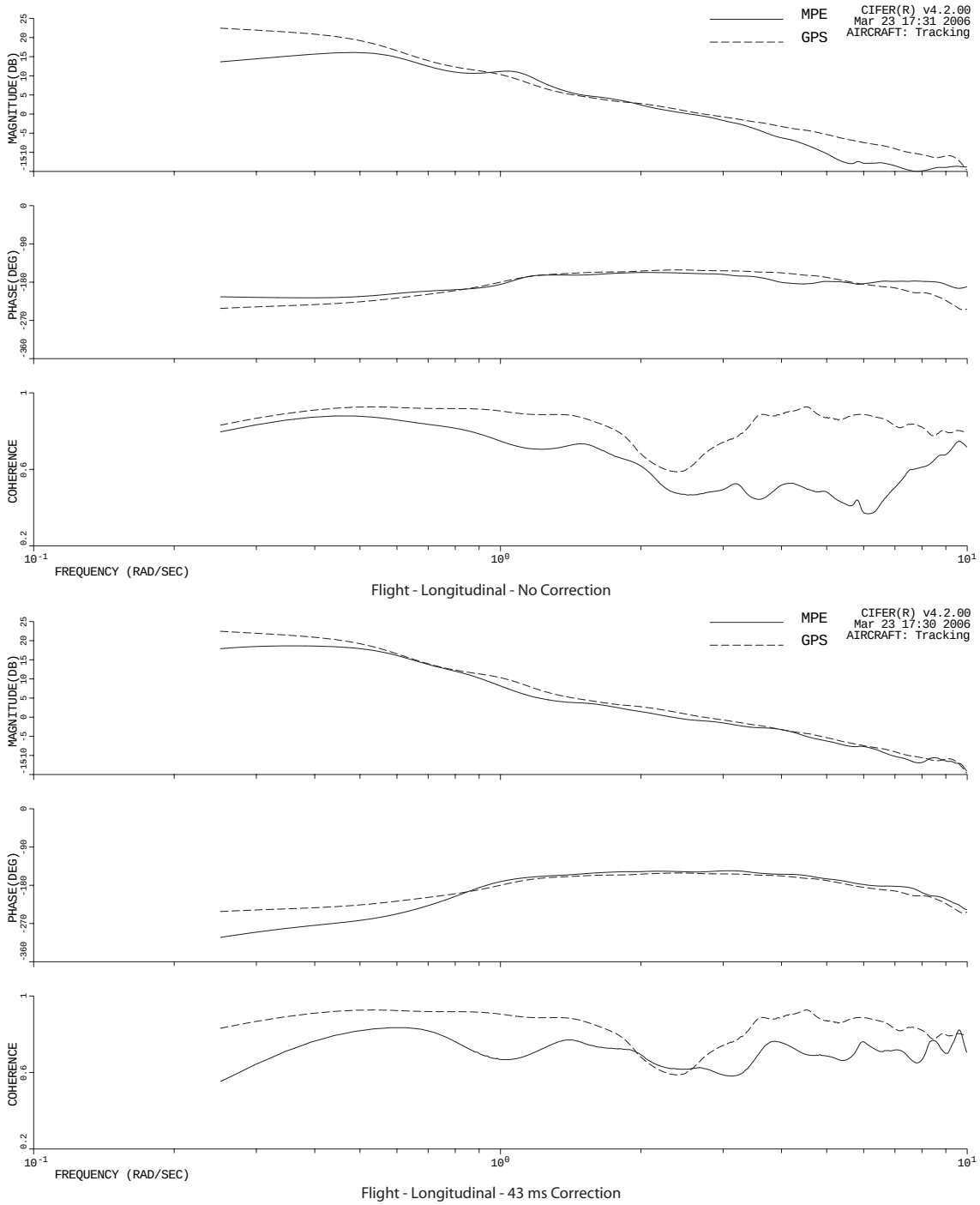


Figure 5.6: A sweep of the longitudinal cyclic control without any attitude correction (top) demonstrates a borderline stability margin (20.70° at $2.54\frac{rad}{s}$) when the MPE algorithm is in closed-loop mode. The bottom figure shows the same type of sweep with the 43 ms attitude correction added to correctly synchronize the inputs to the MPE algorithm. Its stability margin (32.68° at $2.33\frac{rad}{s}$) is close to GPS levels.

	Lateral Phase Margin	Lateral Gain Margin	Longitudinal Phase Margin	Longitudinal Gain Margin
Simulation using GPS	45.84° at 2.92$\frac{rad}{s}$	8.37 dB at 8.80$\frac{rad}{s}$	44.39° at 2.79$\frac{rad}{s}$	8.86 dB at 8.05$\frac{rad}{s}$
GPS Flight	26.67° at 3.21$\frac{rad}{s}$	6.91 dB at 6.24$\frac{rad}{s}$	27.35° at 2.73$\frac{rad}{s}$	6.83 dB at 5.65$\frac{rad}{s}$
MPE Flight w/o correction	17.01° at 2.63$\frac{rad}{s}$	6.88 dB at 4.09$\frac{rad}{s}$	20.70° at 2.54$\frac{rad}{s}$	6.14 dB at 3.98$\frac{rad}{s}$
MPE Flight w/43 ms correction	38.50° at 2.78$\frac{rad}{s}$	8.53 dB at 7.66$\frac{rad}{s}$	32.68° at 2.33$\frac{rad}{s}$	7.96 dB at 6.16$\frac{rad}{s}$

Table 5.2: Sweeps of lateral and longitudinal actuators in simulation and flight, comparing GPS to MPE with and without a 43 ms correction to attitude information, show that the MPE stability margins are as good as GPS when all inputs to the MPE algorithm are correctly synchronized.

summarized in Table 5.2, the results are uniformly positive. By correctly identifying the response of each of the sensors and synchronizing the input to the MPE algorithm, the system is stable and MPE performs as well as GPS.

5.2.1 Power Spectrum

Another way of visualizing the change affected by adding 43 ms delay to the camera and laser data is to look at the power spectrum of the flight control loops. The aircraft has a natural frequency at which the peak energy is seen on the frequency plot. Essentially, this is the frequency at which the UAV has its dominant mode, or resonant frequency. Figures 5.7 and 5.8 show the power spectrum of each cyclic actuator before and after the 43 ms correction. The feedback caused by incorrectly cancelling the attitude in the MPE algorithm produces a peak in the frequency plot which represents the instability and oscillations of the system.

5.3 Closed-Loop Performance

The two-CPU configuration (Section 2.1 on page 13) is designed to precisely synchronize the clocks of each CPU using NTP (Network Time Protocol). This allows software on the experimentation CPU to synchronize attitude information

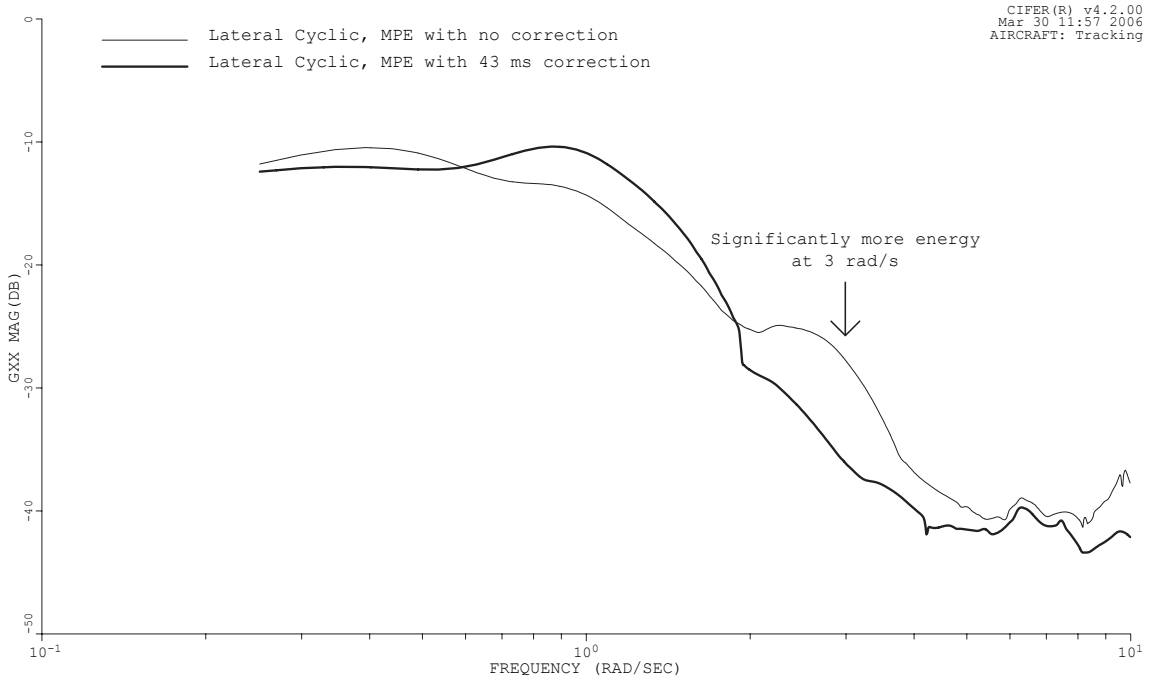


Figure 5.7: The lateral cyclic actuator plotted in the frequency domain demonstrates the improved stability gained by synchronizing the inputs to the MPE algorithm. At crossover (around $3\frac{rad}{s}$), where the control system has a resonant frequency, the system generates significantly less energy (i.e. oscillation) when the 43 ms correction is added.

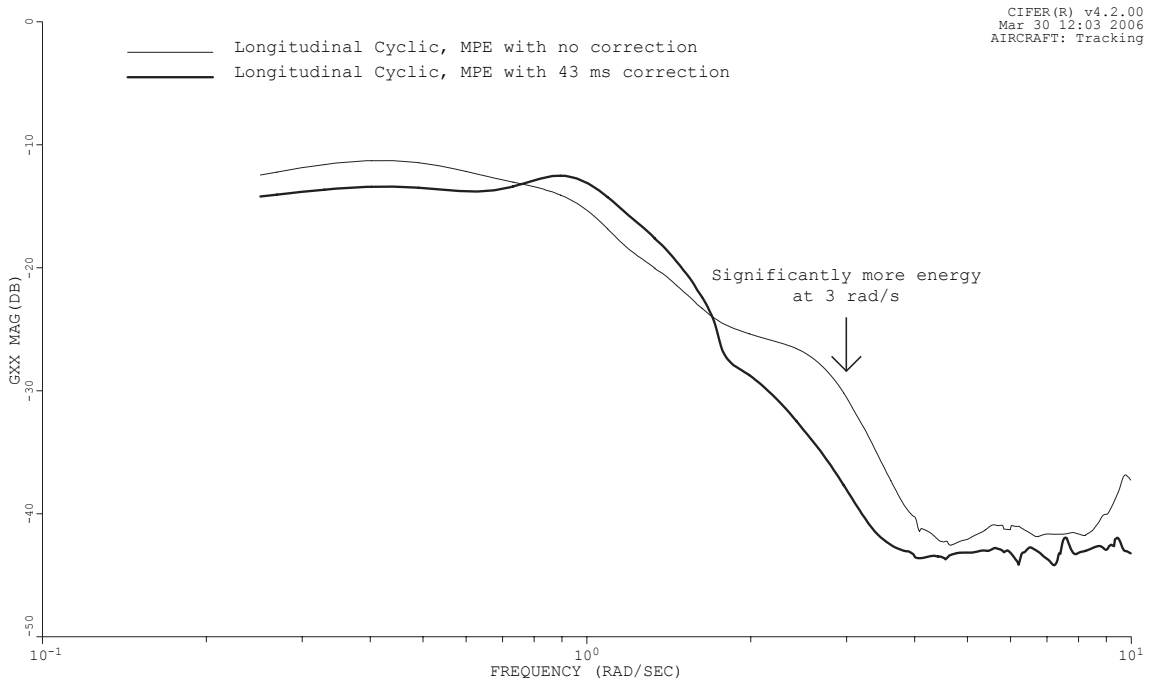


Figure 5.8: The longitudinal cyclic actuator plotted in the frequency domain also demonstrates the improved stability gained by synchronizing the inputs to the MPE algorithm.

from the flight CPU using its internal clock. Experiments with artificial clock errors demonstrate that this can cause instability at moderate altitude (18 m AGL) just as a lack of synchronization can.

When the system operates in closed-loop mode, and the aircraft tracks the position output of the MPE algorithm, one accuracy metric is to compare the MPE coordinates with GPS coordinates. The GPS receiver continues to operate during MPE closed-loop mode, though GPS data is not used to navigate. Just as in open-loop mode, the difference between the GPS position and the MPE estimate provides a metric on how much the tracking algorithm has drifted. Even if the aircraft is under MPE control, the measurement is as valid as when the aircraft is controlled by GPS. In GPS open-loop mode, the difference measures the amount of drift in the MPE estimate directly. In MPE closed-loop mode, the difference measures the drift indirectly by measuring the GPS position that is offset from the perceived MPE estimate, while the aircraft holds the position reported by MPE.

5.4 Landing Accuracy

One PALACE project goal is to achieve a high success rate for landings. Table 5.3 lists 16 landings performed. Two of the landings (both on 27 Jan 2006) did not run to completion. That is an 87.5% success rate. January 27 was the demonstration day for the PALACE project, and numerous observers were on site. At 1:15 the normal aircraft motion was enough to move the tracked point outside the camera's field of view while hovering at 2 m AGL (see Section 4.1 on page 45). At 1:48, the safety pilot stopped the landing at 1 m AGL, after MPE had successfully completed its portion of the landing and the aircraft was performing the final landing stages. The safety pilot chose to terminate the landing because the UAV was near a cardboard obstacle, directly behind the UAV from the safety pilot's line of sight. This made the situation particularly risky. Overall, the number of successful landings can be attributed to the robustness of the vision-guided navigation system.

The drift measurement presented in Table 5.3 represents the cumulative drift for the duration of MPE operation. Since the mission manager reinitializes the MPE algorithm during flight (for instance, when new stereo range data improves on the landing point selection) the MPE algorithm is initialized again from the GPS. Instead of using the final position reported by MPE compared to GPS, which would assume there was zero drift before reinitialization, it is necessary to compute a drift vector before the mission manager reinitializes MPE. Then at the end of the flight, each drift vector computed at reinitialization is added to the final difference between MPE and GPS. The total drift does not exceed the maximum drift from simulation or open-loop flight (Section 4.2), but the limited number of flights does not represent an exhaustive analysis of drift characteristics.

5.5 Conclusions

This chapter demonstrates the performance of the vision-baed navigation system when the control loops are closed and the UAV flight depends on the system to function correctly. Frequency response analysis shows that the MPE algorithm is capable of functioning with as good stability margins as GPS. In fact, there is no

Date	Time	Surface	Delay	Duration	MPE drift
23 Nov 2005	9:31 AM	Asphalt	None	438.3 s	0.65 m
23 Nov 2005	9:57 AM	Asphalt	None	365.6 s	0.93 m
23 Nov 2005	10:26 AM	Asphalt	None	477.6 s	0.64 m
15 Dec 2005	9:55 AM	Asphalt	None	285.9 s	0.63 m
15 Dec 2005	10:47 AM	Asphalt	None	337.4 s	1.12 m
15 Dec 2005	11:43 AM	Asphalt	None	382.0 s	0.62 m
12 Jan 2006	11:43 AM	Asphalt	None	357.1 s	0.52 m
20 Jan 2006	11:10 AM	Gravel	None	357.4 s	0.40 m
20 Jan 2006	10:52 AM	Gravel	43 ms	306.8 s	0.68 m
24 Jan 2006	9:46 AM	Gravel	43 ms	486.2 s	0.71 m
24 Jan 2006	10:41 AM	Gravel	43 ms	310.6 s	0.88 m
25 Jan 2006	9:37 AM	Gravel	43 ms	327.4 s	0.54 m
25 Jan 2006	10:21 AM	Gravel	43 ms	301.8 s	0.71 m
25 Jan 2006	10:40 AM	Gravel	43 ms	300.4 s	0.28 m
27 Jan 2006	1:15 PM	Gravel	43 ms	251.4 s	0.79 m
27 Jan 2006	1:48 PM	Gravel	43 ms	321.6 s	0.65 m

Table 5.3: Data from 16 landings shows the accuracy of the MPE algorithm.

practical limitation on the bandwidth of the MPE algorithm. The limiting factors are the bandwidth of the input sensors (camera, laser, and IMU), and, to a lesser degree, the processing time required by the tracking algorithm (10 ms).

Although the MPE algorithm has been tested in a closed-loop hover at 30 m AGL, the landing procedure was designed with the 18 m AGL ceiling in mind to protect against the potential instability caused by poorly synchronized sensors. Further research in the use of the MPE algorithm may be able to raise its operational ceiling to as much as 50 m AGL.

Chapter 6

Conclusions

The PALACE project successfully demonstrated an autonomous landing at an unprepared site on January 27, 2006. The vision-based navigation system provides an effective GPS replacement with stability margins close to those of the GPS sensor. The ability to land at an unprepared site without the aid of GPS represents a novel solution to the self-localization problem. This expands the types of autonomous missions a rotorcraft UAV can accomplish to include difficult and dangerous operations in regions with unknown terrain.

6.1 Contributions

This vision-based navigation research offers several significant contributions to the field of robotics, including the field testing, the accuracy data, and the work to achieve GPS-level stability margins. By demonstrating autonomous landing at an unprepared site with an 87.5% success rate, the MPE algorithm achieves a track record that should qualify it for consideration in future robotics projects. Extensive field testing over varied terrain provides performance data useful for implementing vision-based navigation systems.

The tracking drift measured in simulation and in flight forms a baseline metric for tracking system performance. Researchers can compare and contrast different tracking methods and hardware solutions in real-world flight conditions based on the data presented. The tracking algorithm operates with sufficient accuracy to allow an R-UAV to navigate to a landing site without drifting outside the chosen landing area.

The effective bandwidth of the MPE algorithm is only limited by the bandwidth of the sensors it needs (camera, laser range finder, and IMU). The improvement in closed-loop performance by accurately identifying the frequency response characteristics of each sensor serves as a reminder of the importance of time synchronization and frequency response analysis for control systems.

These contributions become especially important when applying vision-based navigation techniques on a UAV. By seeking to operate at similar stability margins and accuracy levels as GPS, the vision system becomes an effective GPS replacement for added system reliability, especially in environments where GPS may not be available.

6.2 Equipment Limitations

Several limitations in the equipment used in the PALACE project had a significant effect on the design of the experiment. In Section 3.2, the SiCK PLS-300 laser range finder does not measure distances below 0.7 m. In addition, the limited field of view in the camera at low altitude increases the probability of tracking failure below about 2 m AGL (Section 4.1).

Several other camera limitations related to lighting (Section 4.3) also affect the experiments. Camera noise and motion blur increase in low light conditions. Even restricted to daylight conditions, the flight tests have motion blur below 0.5 m AGL.

6.3 Software Limitations

Additional limitations exist in the algorithms used. Although in the ideal case the MPE algorithm has unlimited bandwidth and precision, the error incident to rounding the position estimate to a camera pixel leads to increasing noise at higher altitudes. Closed-loop MPE control has only been tested up to 30 m AGL in flight. In simulation, MPE control is stable up to about 50 m AGL (Section 5.5). The tracking algorithm does not perform well when the UAV's shadow moves across the landing site (Section 4.3). Large man-made objects that are visible through the whole camera

image can also produce a high-ranking interest points where the tracking algorithm fails due to the aperture problem (Section 2.2.1).

6.4 Future Work

As discussed in Section 4.4, the automatic gain and integration time of the CCD would help reduce the drift and tracking discontinuities. Situations involving low lighting or landing paths where the UAV's shadow interferes with tracking would benefit from a camera with an exponential response curve (such as some CMOS cameras). Work with night-vision cameras would also expand the types of missions possible.

Additional future work testing and comparing other vision algorithms to the Förstner interest operator and the pseudo-normalized correlation would benefit from these performance measurements. For example, there may be key textures that prove difficult for the JPL algorithms to track accurately for which alternative algorithms do well. Identifying such textures algorithmically should not prove difficult and could be integrated as a safety measure in future robotics applications so that the autonomous vehicle does not enter an area with dangerous textures or switches to other feature trackers.

Bibliography

[1] C. Theodore, W. Dai, T. McLain, and M. Takahashi, "Full mission simulation of a rotorcraft unmanned aerial vehicle for landing in a non-cooperative environment," in *Annual Forum of the American Helicopter Society*, June 2005. [Online]. Available: http://aeronautics.arc.nasa.gov/assets/pdf/AHS_2005_Theodore.pdf

[2] D. D. Rowley, "Real-time evaluation of vision-based navigation for autonomous landing of a rotorcraft unmanned aerial vehicle in a non-cooperative environment," Master's thesis, Brigham Young University, April 2005. [Online]. Available: http://contentdm.lib.byu.edu/ETD/image/etd697.pdf

[3] A. L. Peterson, "Launched to return," *Unmanned Vehicles Magazine*, vol. 1, no. 1, January 2003. [Online]. Available: http://www.sncorp.com/media/APUV.pdf

[4] *BEAR: Berkeley Aerobot Research*, The University of California Berkeley, March 2006, website accessed March 2006. [Online]. Available: http://robotics.eecs.berkeley.edu/bear/

[5] O. Shakernia, R. Vidal, C. S. Sharp, Y. Ma, and S. Sastry, "Multiple view motion estimation and control for landing an unmanned aerial vehicle," in *IEEE ICRA: Proceedings of IEEE International Conference on Robotics and Automation*, vol. 3, IEEE. IEEE Conference Proceedings, May 2002, pp. 2793–2798. [Online]. Available: http://robotics.eecs.berkeley.edu/bear/publications.html

[6] S. Saripalli, J. F. Montgomery, and G. S. Sukhatme, "Visually guided landing of an unmanned aerial vehicle," *IEEE TRA: Transactions on Robotics and*

Automation, vol. 19, no. 3, pp. 371–380, June 2003. [Online]. Available: http://robotics.usc.edu/~srik/papers/landing2002.pdf

[7] S. Saripalli and G. S. Sukhatme, "Landing on a moving target using an autonomous helicopter," in *Proceedings of the International Conference on Field and Service Robotics*. IEEE, July 2003. [Online]. Available: http://robotics.usc.edu/~srik/publication.php

[8] E. N. Johnson, A. J. Calise, A. R. Tannenbaum, S. Soatto, N. Hovakimyam, and A. J. Yezzi, "Active-vision control systems for complex adversarial 3-D environments," in *IEEE ACC: American Control Conference*. IEEE American Control Conference, June 2005. [Online]. Available: http://www.ae.gatech.edu/people/ejohnson/acctutorial2005.pdf

[9] J. Ha, C. Alvino, G. Prior, M. Niethammer, E. N. Johnson, and A. R. Tannenbaum, "Active contours and optical flow for automatic tracking of flying vehicles," in *IEEE ACC: Proceedings of the American Control Conference*, vol. 4. IEEE Conference Proceedings, June 2004, pp. 3441–3446. [Online]. Available: http://controls.ae.gatech.edu/papers/ha_acc_04.pdf

[10] M. Whalley, M. Freed, M. Takahashi, D. Christian, A. Patterson-Hine, G. Schulein, and R. Harris, "The NASA/Army autonomous rotorcraft project," in *Annual Forum of the American Helicopter Society*, May 2003. [Online]. Available: http://human-factors.arc.nasa.gov/apex/docs/papers/arpahs03/ARP_AHS03_v10.pdf

[11] L. G. Roberts, "Machine perception of three-dimensional solids," *Optical and Electro-optical Information Processing*, 1965, ed. J. T. Tippett (Cambridge: The MIT Press).

[12] W. Forstner, "Reliability analysis of parameter estimation in linear models with application to mensuration problems in computer vision," *Computer Vision, Graphics, and Image Processing*, vol. 40, no. 3, pp. 273–310, 1987.

[13] E. Trucco and A. Verri, *Introductory Techniques for 3-D Computer Vision*. Upper Saddle River, New Jersey 07458: Prentice Hall, 1998.

[14] D. Todorovic, "A gem from the past: Pleikart Stumpf's (1911) anticipation of the aperture problem, Reichardt detectors, and perceived motion loss at equiluminance," *Perception*, vol. 25, no. 10, pp. 1235–1242, 1996. [Online]. Available: http://www.perceptionweb.com/perc1096/conts.html

[15] H. Moravec and D. Gennery, "Cart project progress report," Stanford University, Tech. Rep., September 1976. [Online]. Available: http://www.frc.ri.cmu.edu/~hpm/project.archive/robot.papers/1976/nasa1.txt

[16] J. P. Lewis, "Fast normalized cross-correlation," in *Vision Interface*, Canada, 1995.

[17] 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. [Online]. Available: http://contentdm.lib.byu.edu/cdm4/item_viewer.php?CISOROOT=/ETD&CISOPTR=88

[18] J. Hintze, M. Tischler, D. Christian, T. McLain, C. Theodore, and J. Montgomery, "Simulated autonomous landing of a rotorcraft unmanned aerial vehicle in a non-cooperative environment," in *Annual Forum of the American Helicopter Society*, June 2004.

[19] *Comprehensive Identification of Frequency Response*, U.S. Army Aeroflightdynamics Directorate, NASA Ames Research Center, Moffett Field, CA 94035, website accessed 31 May 2007. [Online]. Available: http://www.nasa.gov/centers/ames/research/technology-onepagers/paramid.html