SANDIA REPORT

SAND94-8223 • UC-402
Unlimited Release
Printed March 1994

The Computation of Cloud Base Height from Paired Whole-Sky Imaging Cameras

M. C. Allmen, W. P. Kegelmeyer, Jr.

Prepared by
Sandia National Laboratories
Albuquerque, New Mexico 87185 and Livermore, California 94551
for the United States Department of Energy
under Contract DE-AC04-94AL85000

UC-402

SAND94-8223
Unlimited Release
Printed March 1994

THE COMPUTATION OF CLOUD BASE HEIGHT FROM PAIRED WHOLE-SKY IMAGING CAMERAS

Mark C. Allmen and W. Philip Kegelmeyer, Jr.
Scientific Computing Department
Sandia National Laboratories/California

Abstract

A major goal for global change studies is to improve the accuracy of general circulation models (GCMs) capable of predicting the timing and magnitude of greenhouse gas-induced global warming. Research has shown that cloud radiative feedback is the single most important effect determining the magnitude of possible climate responses to human activity. Of particular value to reducing the uncertainties associated with cloud-radiation interactions is the measurement of cloud base height (CBH), both because it is a dominant factor in determining the infrared radiative properties of clouds with respect to the earth's surface and lower atmosphere and because CBHs are essential to measuring cloud cover fraction.

We have developed a novel approach to the extraction of cloud base height from pairs of whole sky imaging (WSI) cameras. The core problem is to spatially register cloud fields from widely separated WSI cameras; this complete, triangulation provides the CBH measurements. The wide camera separation (necessary to cover the desired observation area) and the self-similarity of clouds defeats all standard matching algorithms when applied to static views of the sky. To address this, our approach is based on optical flow methods that exploit the fact that modern WSIs provide *sequences* of images.

We will describe the algorithm and present its performance as evaluated both on real data validated by ceilometer measurements and on a variety of simulated cases.

ACKNOWLEDGMENT

This research was supported by the United States Department of Energy through the Atmospheric Radiation Measurement Program, Sandia National Laboratories, Livermore, contract # DE-AC04-76DO00789. The Khoros[1] system, developed by Rasure [11], was used for code development and digital image visualization. Code for the generation of synthetic cloud scenes was provided by TASC–The Analytic Sciences Corporation, and the code which simulates radiance maps and projects 3-D cloud volumes into simulated WSI imagery was written by Chen-Hui Sun of Sandia National Laboratories.

[1] A publicly available integrated software development environment for information processing and visualization. Send e-mail to `khoros@chama.eece.unm.edu` for further information

Contents

Chapter 1

Introduction

Cloud base height (CBH) is a dominant factor in determining the infrared radiative properties of clouds. However, cloud base heights are not well known. To address this fact, this paper presents a novel approach for the extraction of cloud base height from pairs of whole sky imaging (WSI) cameras. Much like the human visual system can perceive depth by using two eyes, cloud height can be computed using two cameras viewing the same cloud field.

Given two images from widely separated WSI cameras, the core problem of computing cloud height is to find corresponding points between the two images, i.e., find points which are projections from the same point on a cloud. This complete, triangulation provides the CBH measurements. Given the orientation and distance between the cameras, for any given point in one image, the corresponding point must lie along a well-defined line in the other image. Correlation of intensity values is a standard technique for finding the corresponding point along this line. However, the wide camera separation (necessary to cover the observation area required by the field measurements site) and the self-similarity of clouds defeat this approach when applied to static views of the sky.

To compute CBH we exploit the fact that modern WSIs provide *sequences* of images. With these sequences, the optical flow field, i.e., a field of vectors which represent the image motion of points, can be recovered and used to aid in finding corresponding points. In the WSI case, the correlation approach is augmented to include this motion information along with the intensity information at each pixel.

The following chapters will motivate this problem and the use of WSI cameras, describe the central principle of the height extraction algorithm, and present quantitative measures of its performance as evaluated on real and simulated data.

1.1 Background

1.1.1 General Circulation Models and Clouds

A major goal for the Department of Energy's (DOE's) global change efforts is to improve the accuracy of general circulation models (GCMs) capable of predicting the timing and magnitude of greenhouse gas-induced global warming. Research has shown cloud radiative feedback is the single most important feedback effect determining the magnitude of possible climate responses to human activity. Yet, as pointed out by Cess [5], clouds are not well

parameterized in GCMs and are, in fact, presently the greatest factor limiting the accuracy of atmospheric GCMs. Thus clouds exert the largest influence while at the same time present the largest uncertainties in predicting global climate change. As a result, cloud studies are critical to understanding global climate change and improving the predictive accuracy of GCMs. In recognition of this problem, (e.g., Rossow [13]) a number of important national and international programs have recently been initiated to characterize cloud-radiation interactions, including DOE's Atmospheric Radiation Measurement Program ARM, and the International Satellite Cloud Climatology Project.

In the ARM program a key to this cloud-radiation characterization is the effective treatment of cloud formation and cloud properties in GCMs as supported by a field measurements program — "an important feature of the ARM Program Plan is to establish a surface-based cloud imaging system at the research sites that will provide appropriate information for parameterizing solar flux over an entire grid cell" . The first such Cloud And Radiation Testbed (CART) site makes measurements, including cloud measurements, over a 30-km diameter region.

1.1.2 Macroscopic Cloud Properties

A well-recognized approach to reducing the uncertainties associated with cloud-radiation interactions involves measuring the macroscopic properties of clouds (shape, size, extent, cloud cover fraction, radiance, altitude, etc.) on the mesoscale (20-200 km). Of these measurements, cloud base height is particularly important because it is a dominant factor in determining the infrared radiation from clouds to the lower atmosphere and the earth's surface. Furthermore, as shown by Rossow [13], base heights are essential to measuring the cloud cover fraction at low, medium and high altitude, and these in turn are needed to establish a cloud-radiation climatology.

1.1.3 Whole-Sky Imagers and Digital Imagery

Digital whole-sky imagers have a number of advantages for making cloud studies. They are passive and therefore relatively inexpensive and reliable, can be used in unattended operation, and obtain images of the whole sky dome rapidly. The particular passive camera that has generated our existing data is the Whole Sky Imager developed by the Marine Physical Laboratory (MPL) at the Scripps Institution of Oceanography, as described in Shields [14]. These imagers are rugged and have demonstrated many years of high-reliability field service. Full-resolution (1/3°) *digital* images can be acquired at one per minute. This is rapid enough to capture most of the cloud dynamics of interest and fully utilize the image motion of the clouds. As the images produced by the MPL WSI are already digital, there is no need for a digitization step before the data can be electronically stored and manipulated by a computer. The former insures permanent and undegraded data availability, and the latter is the only practical way to handle the volumes of data to be gathered.

Most prior cloud dynamics studies have been carried out using time-lapse photography to investigate the inception, growth, persistence, transport and breakup of clouds. These are important elements of the cloud life cycle, and so are presently of central importance in the ARM Program. However, the painstaking labor and individual attention required by

the use of photographs will make similar studies impossible when attempting to process the volume of data which will be collected by the CART sites. Analysis of the data will thus require digital image processing.

1.2 Method

1.2.1 An Overview

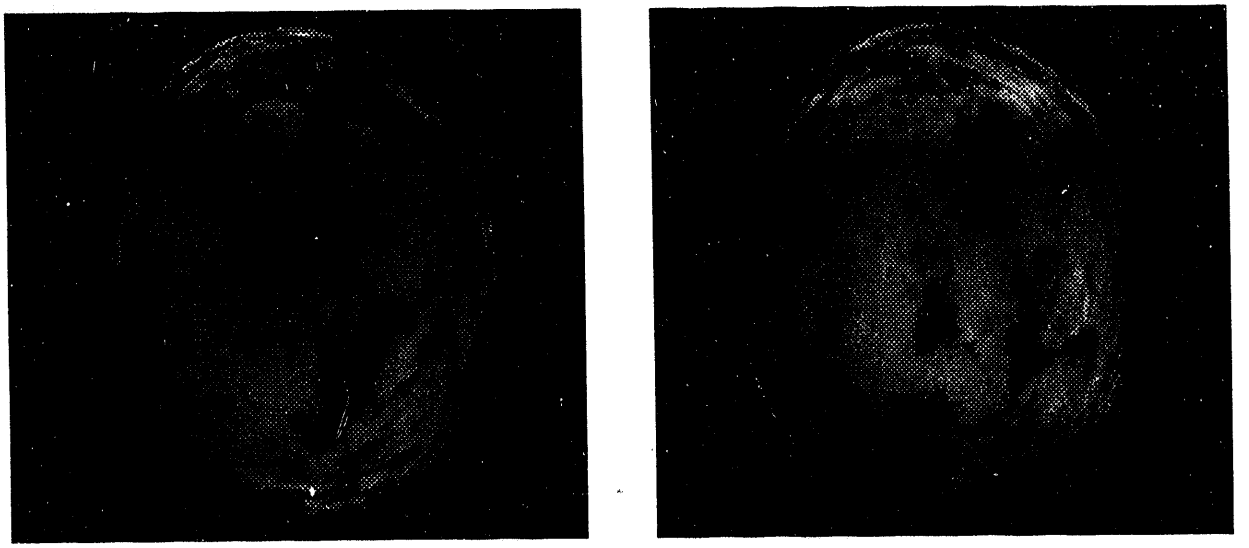
Unfortunately, the state of the art in cloud imagery processing is not yet capable of extracting measures as central and important as cloud bottom heights. The main problem is to spatially match up cloud fields from widely separated WSI cameras; once registered against each other, computation of cloud bottom heights proceeds in a straightforward fashion from triangulation and knowledge of the camera locations. Our solution to this problem utilizes *temporal* flow fields from each camera separately. In the following subsections we will review the prior history of this problem, illustrate its difficulty, and suggest why flow fields provide the necessary additional constraints.

1.2.2 Prior Work

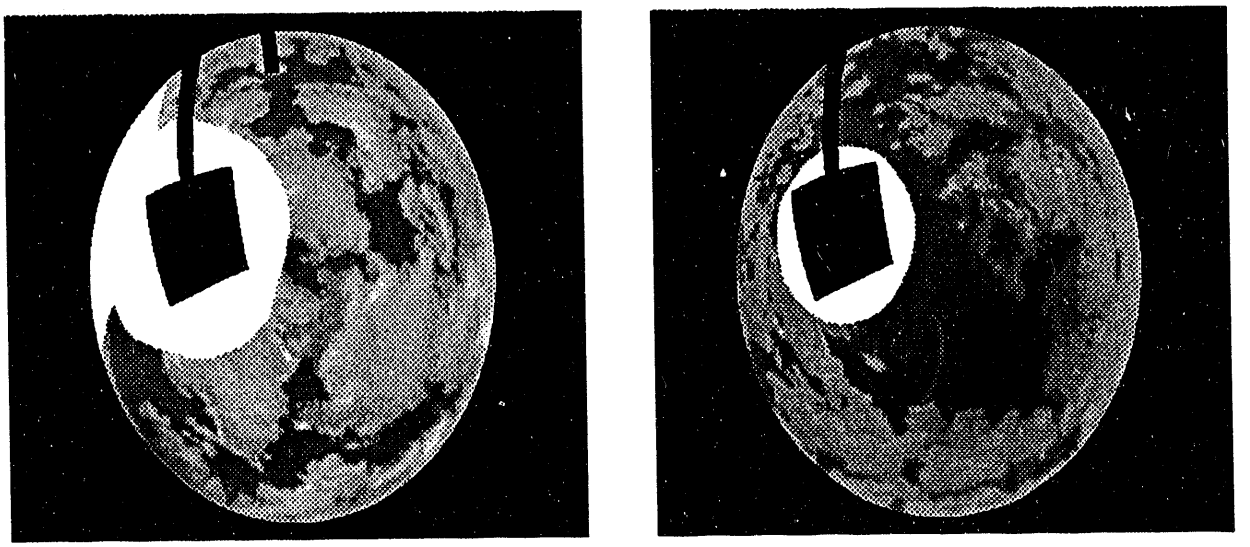
Extracting cloud bottom heights via triangulation of registered points has been in the cloud stereoscopy literature for twenty years, and is well-understood. The registration itself, however, has not been well addressed. In the earlier literature, e.g., Bradbury [3] and Lyons [10], the problem was side-stepped though human intervention, registering the images by hand before triangulation. More recently, e.g., Rock [12], the automatic registration problem has been successfully handled, but only for nearly parallel views of the sky (i.e., cameras spaced closed together). In that case the stereoscopic nature of the views permitted simple limited-displacement correlation to suffice as a registration algorithm.

The registration problem facing ARM, however, is considerably harder in that the camera spacing is on the order of 5 km, as this is required to achieve adequate coverage at the required resolution with a small number of cameras. With this baseline spacing, the three-dimensional nature of clouds generates occlusion and perspective effects that will cause them to image differently at the various cameras. Because of this, correlation-based registration using intensity only is insufficient. Further, the visual self-similarity of clouds, defeats token matching (the detection and matching of a small number of visually distinctive regions), which is the only common alternative approach.

As an example, the top of Figure 1.1 contains a synthetic example pair of simultaneous frames (see Chapter 5 for how the data was generated). Careful examination will show that corresponding points appear shifted to the top in the right image. This can be difficult to determine by eye, primarily due to the difference in perspective experienced by the widely separated cameras. However, this synthetic cloud scene was very flat and did not contain lighting variation due to the sun. Therefore, finding corresponding points in WSI images of real-world cloud fields is even more difficult, creating the need to also use flow fields to find corresponding points.



(a)



(b)

Figure 1.1: (a) One frame of a synthetic altocumulous cloud scene as viewed from the bottom (left image) and top (right image) cameras. Corresponding points appear shifted to the top in the right image. (b) One frame of a real cloud scene as viewed from two WSI cameras. This example shows how difficult the correspondence problem can be. Corresponding appear shifted down and to the left in the right image.

1.2.3 Flow Field Correlation

The use of image sequences to identify the correspondence suggests how one can automate the registration process. WSI images are acquired at a rate of one per minute, which provides a comparatively dynamic view of the sky. This provides a means to overcome the registration obstacles mentioned before, which apply only to the attempt to register two *static* views of the sky.

The temporal sampling rate of the WSI camera is high enough that optical flow fields can be computed using hierarchical correlation methods such as those developed by Burt [4]. With the optical flow fields computed for the images from both WSI cameras, each image pixel becomes associated with a vector indicating the image motion of that point. Combining this with the intensity value at the pixel, a multi-dimensional quantity is associated with each image pixel. It is the core of our approach that these flow and intensity fields from the separated WSIs are *jointly* registered against each other. In this way, the additional constraints provided by the flow field are exploited to make the matching unique.

Chapter 2

Extracting Cloud Base Heights

In this section the geometry between WSI sites will be developed. Using the resulting equations the height of a registered cloud point (i.e., the azimuth and zenith of the point in two WSI cameras is known) can be computed. Further, the *epi-polar line* can be determined. Given a point in one WSI image, the epi-polar line in an image from another WSI site defines the line along which the corresponding point must be located. Hence, this line limits the search while finding correspondences between WSI images from different sites. The uncertainty and sensitivity of these equations will also be examined.

2.1 Mapping Azimuth and Zenith Between WSI Sites

Given a point in the sky, the equations describing the azimuth and zenith relative to two WSI cameras are developed. Figure 2.1 shows the geometry between two WSI sites separated by a distance d. Without loss of generality, let site S_2 be directly east of site S_1. Assume that the two WSIs are aligned in parallel and that S_1 and S_2 are at the same altitude.[1] Given the zenith, θ_1, and azimuth, ϕ_1, of a point relative to S_1 and h, the height of the point, we will determine the zenith, θ_2, and azimuth, ϕ_2, of the point relative to S_2.

First we find ϕ_2. The following are obvious:

$$d = d_1' + d_2' \tag{2.1}$$

$$d_1 \cos(\phi_1) = p \tag{2.2}$$

$$\tan(\phi_2 - 270) = \frac{p}{d_2'} \tag{2.3}$$

$$d_1 \sin(\phi_1) = d_1' \tag{2.4}$$

$$\tan(\theta_1) = \frac{d_1}{h} \tag{2.5}$$

Eq. (2.1) is by definition. Substituting Eqs. (2.1) and (2.2) into Eq. (2.3),

$$-\cot(\phi_2) = \frac{d_1 \cos(\phi_1)}{d - d_1'} \tag{2.6}$$

[1]The effects when these assumptions are violated are addressed in Section 2.4

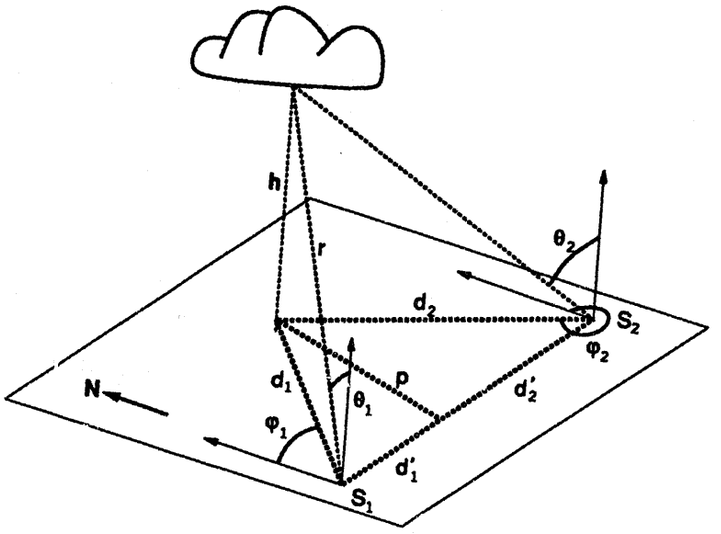


Figure 2.1: The geometry of two WSI sites S_1 and S_2. ϕ_i and θ_i represent the azimuth and zenith, respectively. The height of the point is h. The distance to a position underneath the point is given by d_i. The range from S_1 to the point is given by r.

Substituting Eq. (2.4) into Eq. (2.6) gives

$$\begin{aligned} -\cot(\phi_2) &= \frac{d_1 \cos(\phi_1)}{d - d_1 \sin(\phi_1)} \\ -\tan(\phi_2) &= \frac{d - d_1 \sin(\phi_1)}{d_1 \cos(\phi_1)} \\ \phi_2 &= \arctan\left(\tan(\phi_1) - \frac{d}{d_1 \cos(\phi_1)}\right) \end{aligned} \tag{2.7}$$

Finally, substituting Eq. (2.5) into Eq. (2.7) we find

$$\phi_2 = \arctan\left(\tan(\phi_1) - \frac{d}{h \tan(\theta_1) \cos(\phi_1)}\right) \tag{2.8}$$

Next we find θ_2. The following are obvious:

$$h = d_1 \cot(\theta_1) \tag{2.9}$$

$$d_2 \cos(\phi_2 - 270) = d_2'$$

$$d_2 = \frac{-d_2'}{\sin(\phi_2)} \tag{2.10}$$

$$h = d_2 \cot(\theta_2) \tag{2.11}$$

Substituting Eq. (2.4) into Eq. (2.9) gives

$$\begin{aligned} h &= \frac{d_1'}{\sin(\phi_1)} \cot(\theta_1) \\ d_1' &= h \tan(\theta_1) \sin(\phi_1) \end{aligned} \tag{2.12}$$

Substituting Eq. (2.10) into Eq. (2.11) gives

$$\begin{aligned} h &= \frac{-d_2'}{\sin(\phi_2)} \cot(\theta_2) \\ &= \frac{d_1' - d}{\sin(\phi_2)} \cot(\theta_2) \\ d_1' &= h \tan(\theta_2) \sin(\phi_2) + d \end{aligned} \tag{2.13}$$

Setting Eqs. (2.12) and (2.13) equal and solving for θ_2 we have

$$h \tan(\theta_1) \sin(\phi_1) = h \tan(\theta_2) \sin(\phi_2) + d \tag{2.14}$$

$$\theta_2 = \arctan\left(\frac{h \tan(\theta_1) \sin(\phi_1) - d}{h \sin(\phi_2)}\right) \tag{2.15}$$

By rearranging Eq. (2.15) we have the equation for the height of a point given its azimuth and zenith at sites 1 and 2.

$$h = \frac{d}{\tan(\theta_1) \sin(\phi_1) - \tan(\theta_2) \sin(\phi_2)} \tag{2.16}$$

In the next section Eqs. (2.8) and (2.15) will be used to define the eqi-polar line.

2.2 Computing the Epi-Polar Line

The equations of the previous section can be used to help find corresponding points between images from two WSI sites. Given a point in an WSI image, to compute it's height we

must find the projection of that point in another WSI image. Given the point in an image the azimuth and zenith of that point is known. However, the range, r, of that point from the WSI is not known. As the range is varied from zero to infinity, the location of the corresponding point in another WSI sweeps out a line called the epi-polar line. So, given a point in an image, its corresponding point in another image must lie along the well-defined epi-polar line. Using the equations above, the epi-polar line at S_2 for a given θ_1 and ϕ_1 are developed in this section.

All quantities in Eq. (2.8) are constant as r varies except h. However, $h\tan(\theta_1) = d_1 = r\sin(\theta)$. Substituting this into Eq. (2.8) gives

$$\phi_2 = \arctan\left(\tan(\phi_1) - \frac{d}{r\sin(\theta_1)\cos(\phi_1)}\right) \tag{2.17}$$

This equation gives the change of ϕ_2 as a function of r.

We now need to find θ_2 as a function of r. Substituting $h\tan(\theta_1) = r\sin(\theta_1)$ and $h = r\cos(\theta_1)$ into Eq. (2.15) gives:

$$\theta_2 = \arctan\left(\frac{r\sin(\theta_1)\sin(\phi_1) - d}{r\cos(\theta_1)\sin(\phi_2)}\right) \tag{2.18}$$

Eqs. (2.17) and (2.18) describe how the azimuth and zenith change as a function of r. The following equations from [9] define the point within a WSI image which corresponds to a particular azimuth and zenith. These camera calibration equations were determined experimentally and will be discussed in Section 2.4.6.

$$d = \theta(3.032 - 0.00259\theta) \tag{2.19}$$
$$x = 255 + 0.8d\sin(\phi) \tag{2.20}$$
$$y = 240 + d\cos(\phi) \tag{2.21}$$

d is the distance from the center of the image to a point. So using Eqs. (2.17)–(2.21) a line is defined in an image. This is the epi-polar line and is used to limit the search when looking for corresponding points.

To determine the height of a point in a WSI image we compute the azimuth and zenith of that point relative the WSI site. With the azimuth, zenith and relative location of another WSI site given, we compute the epi-polar line in an image from the other WSI site. We then search along the epi-polar line for the matching point. Once the corresponding point is found its azimuth and zenith is computed relative the the second site. Eq. (2.16) is then used to compute the height.

More specifically, at this point the algorithm is as follows:

- For every point, p_1, in image 1
  - Compute the epi-polar line for p_1 in image 2. [Section 2.2, Eqs. (2.17)–(2.21)].
  - For each point, p_2, along the epi-polar line
    - Match p_1 and p_2 [Section 4.5].

- Find the best matching point along the epi-polar line. Call this point p_2'.
- Compute the cloud height using p_1 and p_2' as corresponding points. [Eq. (2.16)].

The next section will describe a confidence metric for the computed base heights. How to perform the match of points p_1 and p_2 will be discussed in Chapter 4. The following sections will describe a confidence metric for the computed base heights and discuss the sensitivity and uncertainty of the system.

2.3 A Confidence Metric

Given that the WSI cameras are widely separated, it is possible that many cloud points will be visible in one camera but not the other. When this happens, the algorithm for finding corresponding points between images will report a matching point (as whichever point best matches), but the resulting computed height will most likely be incorrect. In cases where the corresponding point does not exist, a height should not be computed and the algorithm should report this. In order to detect cases where a cloud point is only visible in a single image, we describe a confidence metric which indicates the confidence of the algorithm that it has found two corresponding points.

To compute the confidence, we effectively find corresponding points twice. First, corresponding points for the points in image 1 are found. Second, corresponding points for the points in image 2 are found. Let p_1 be a point in WSI image 1. The epi-polar line for point p_1 is computed in WSI image 2. The best matching point along this epi-polar line is found. Call this point p_2'. This is shown in Figure 2.3(a). Next, the epi-polar line for point p_2' is computed in WSI image 1. The best matching point along this epi-polar line is found. Call this point p_1'. In the ideal case, p_1 and p_1' are the same point. The distance between points p_1 and p_1' is the confidence value. So the smaller the value, the higher the confidence of the algorithm. This is shown in Figure 2.3(b).

If there is no correct corresponding point to p_1 in the right image, then the point labeled as p_2' is clearly not the correct corresponding point. When the corresponding point for p_2' is found in the left image, it is more likely that its correct corresponding point will be found rather than p_1 and the confidence of the computed height of p_1 will be low.

The algorithm, now with the confidence computation, is as follows:

- For every point, p_1, in image 1
  - Compute the epi-polar line for p_1 in image 2. [Section 2.2, Eqs. (2.17)–(2.21)].
  - For each point, p_2, along the epi-polar line
    - Match p_1 and p_2 [Section 4.5].
  - Find the best matching point along the epi-polar line. Call this point p_2'.
  - Compute the epi-polar line for p_2' in image 1. [Section 2.2 Eqs. (2.17)–(2.21)].
  - For each point, p_3, along the epi-polar line
    - Match p_2' and p_3.

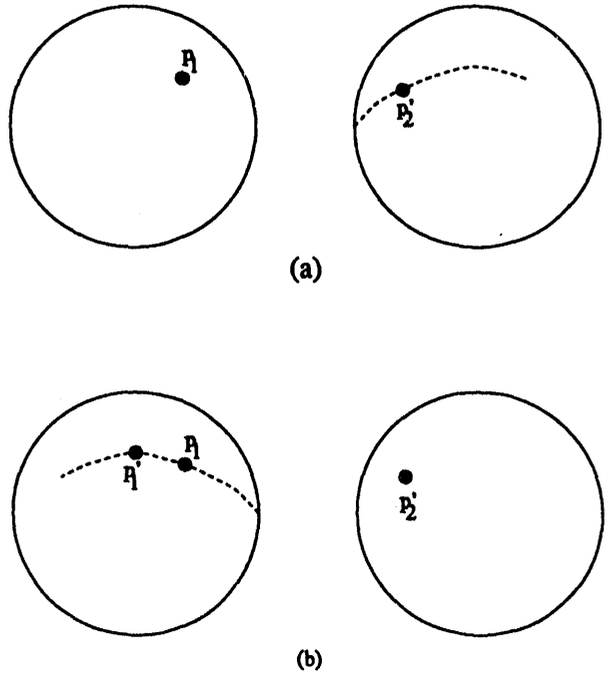


Figure 2.2: (a) Let p_1 be a point in the left WSI image. The epi-polar line for point p_1 is computed in the right WSI image and the best matching point, p_2', along this epi-polar line is found. (b) The epi-polar line for point p_2' is computed in the left WSI image. The best matching point, p_1', along this epi-polar line is found. In the ideal case, p_1 and p_1' are the same point. The distance between points p_1 and p_1' is the confidence value.

- Find the point of maximum correlation value along the epi-polar line. Call this point p_1'.
- Compute the cloud height using p_1 and p_2' as corresponding points. [Eq. (2.16)].
- Compute the distance between p_1 and p_1'. This is the confidence value of point p_1.

2.4 Sensitivity and Uncertainty

In this section the sensitivity and uncertainty of the system will be analized. It will be shown how much a point in the sky could move vertically without a change in the computed base height. Further, there are five sources of uncertainty in the system: 1) the relative altitude of the WSIs; 2) the positioning of the WSIs; 3) the alignment of the WSIs; 4) the plumbness of the WSIs, and 5) the projective properties of the WSIs. These five sources of error will be examined in this section. In all cases, the error can result in the computed epi-polar line missing the correct matching point. Below we will show the difference between the azimuth and zenith with no error in camera positioning or alignment and the azimuth and zenith

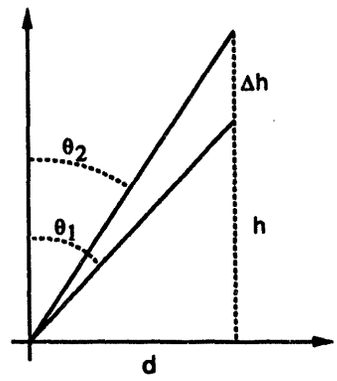


Figure 2.3: The amount that the height of a point can change, Δh, without being detected by a WSI, i.e., $\theta_1 - \theta_2 < 0.3°$.

when there is error. In this way we get an estimate of the amount of displacement that the matching point is off the epi-polar line. For a small displacement there is a reasonable chance that the correct matching point along the epi-polar line will be found. But for large displacements, there is little guarantee of finding the correct point.

Note that if we did not restrict our attention to the matching along the epi-polar line, then the correct matching point might still be found regardless of these errors. However, there will still be error in the computed cloud base height. For completeness the error in computed cloud height given the correct matching points is included. The error in this case is the difference between the computed height with the WSI perfectly oriented and positioned and the computed height with the WSI mispositioned or misoriented.

2.4.1 Sensitivity

In this subsection the sensitivity of the camera configuration will be shown. That is, we will develop equations which show how much a point in the sky could move vertically without being detected by either camera.

The angular resolution of the WSI camera is $\frac{1}{3}°$ per pixel. So given a point in the sky, one simply needs to move that point vertically until the zenith angle has changed by $\frac{1}{3}°$ in either of the two cameras. This is the sensitivity of the configuration for that particular point and height.

From Figure 2.3 it is clear that

$$\tan(\theta_1) = \frac{d}{h} \tag{2.22}$$

and that

$$\tan(\theta_2) = \frac{d}{h + \Delta h} \tag{2.23}$$

d is the distance from the camera to the position underneath the point and h is the height of the point.

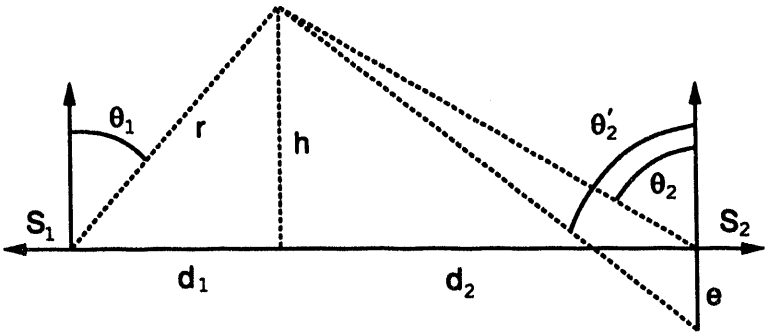


Figure 2.4: The change of θ_2 as the elevation between WSI sites differ. Note this figure is effectively Figure 2.1 viewed edge-on.

Subtracting Eqs. (2.22) and (2.23), setting the result equal to $\frac{1}{3}$ and solving for Δh gives the sensitivity.

$$\begin{aligned} \frac{1}{3} &= \arctan\left(\frac{d}{h}\right) - \arctan\left(\frac{d}{h+\Delta h}\right) \\ \tan(\frac{1}{3}) &= \frac{\frac{d}{h} - \frac{d}{h+\Delta h}}{1 + \left(\frac{d}{h}\right)\left(\frac{d}{h+\Delta h}\right)} \\ \Delta h &= \frac{\tan(\frac{1}{3})\ (h^2 + d^2)}{d - h\ \tan(\frac{1}{3})} \end{aligned}$$

Δh is computed for each camera and the smallest value is the sensitivity of the system.

2.4.2 Uncertainty of Elevation

The case where two WSI sites are at different elevations can be dealt with, but in this section we will examine the amount of error induced if the two sites are not at the exact specified elevation. We will assume without loss of generality that the two sites are suppose to be at the same elevation. Clearly, the azimuth, ϕ_2, is unchanged by a difference in elevation. However, the zenith will change. Figure 2.4 shows the vertical geometry between S_1 and S_2. Also shown is the geometry when S_2 differs in elevation from S_1 by a distance e. θ_2 is the zenith angle if the WSIs were at the same elevation and θ_2' is the zenith angle if the WSI at Site 2 is not at the same elevation as Site 1.

The following two equations are obvious.

$$\tan(\theta_2) = \frac{d_2}{h}$$

$$\tan(\theta_2') = \frac{d_2}{h + e}$$

Solving both equations for d_2, setting equal and rearranging gives:

$$\tan(\theta_2') = \frac{h}{h + e}\tan(\theta_2) \tag{2.24}$$

It is known that a one-third degree change in zenith angle results in a change of one pixel in the image. Therefore, the number of pixels, P, that the matching point will be away from where it should be is given by

$$\begin{aligned} P &= \frac{1}{3}(\theta_2 - \theta_2') \\ \tan(3P) &= \tan(\theta_2 - \theta_2') \\ &= \frac{\tan(\theta_2) - \tan(\theta_2')}{1 + \tan(\theta_2)\tan(\theta_2')} \end{aligned} \tag{2.25}$$

Substituting Eq. (2.24) into Eq. (2.25) gives:

$$P = 0.33 \tan^{-1}\left(\frac{\tan(\theta_2)\left(1 - \frac{h}{h+e}\right)}{1 + \frac{h}{h+e}\tan^2(\theta_2)}\right)$$

In the event that the correct matching point is found, the error in computed height can be found by computing the difference between the computed height when the WSIs are at the same elevation and the computed height when one WSI is elevated with respect to the other.

The height of a point with both WSIs at the same elevation is given by Eq. (2.16). The computed height of a point with one WSI offset is

$$h_{elevation-diff} = \frac{d}{\tan(\theta_1)\sin(\phi_1) - \tan(\theta_2')\sin(\phi_2)} \tag{2.26}$$

Substituting Eq. (2.24) into Eq. (2.26) gives

$$h_{elevation-diff} = \frac{d}{\tan(\theta_1)\sin(\phi_1) - \frac{h}{h+e}\sin(\phi_2)\tan(\theta_2)} \tag{2.27}$$

Subtracting Eq. (2.27) from Eq. (2.16) gives the error due to a difference in elevation.

2.4.3 Uncertainty of Position

There are two types of uncertainty in positioning of one WSI relative to another: the distance between the cameras and the direction of one camera relative the other. Without loss of generality, assume that camera 1 is situated at azimuth 90° relative to camera 2 and is distance d away. Figure 2.5 shows the configuration at camera 2, which is located at the origin. Given a potential misplacement of camera 2 by e_x and e_y, the new azimuth, ϕ_2' is given by

$$\sin(\phi_2') = \frac{x + e_x}{\sqrt{(x + e_x)^2 + (y + e_y)^2}} \tag{2.28}$$

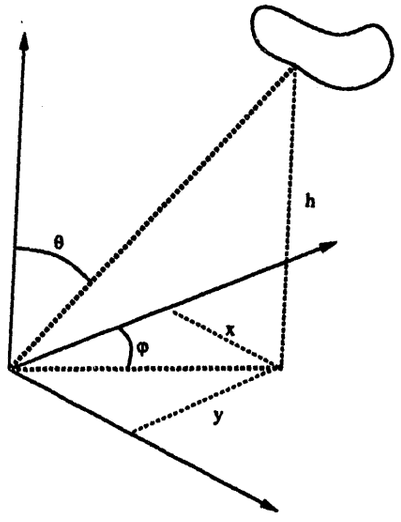


Figure 2.5: Configuration at site of WSI 2. The cloud is at (x, y) relative to the camera location. If the WSI is misplaced then the position of the cloud relative to the WSI changes. The position is then $(x + e_x, y + e_y)$.

The new zenith, θ_2', is given by

$$\tan(\theta_2') = \frac{\sqrt{(x+e_x)^2 + (y+e_y)^2}}{h} \tag{2.29}$$

ϕ_2 and θ_2 are given by

$$\sin(\phi_2) = \frac{x}{\sqrt{x^2+y^2}} \tag{2.30}$$

$$\tan(\theta_2) = \frac{\sqrt{x^2+y^2}}{h} \tag{2.31}$$

The process to find the number of pixels that a point moves in the image as a result of the camera being mispositioned is straightforward. Given a zenith angle, θ, the distance from the center of the image, R, that a point at that zenith appears is given by

$$R = \theta(3.032 - 0.00259\ \theta) \tag{2.32}$$

The i and j position of that point in the image is then given by

$$j = 255 + 0.8\ R\ \sin(\phi) \tag{2.33}$$

$$i = 240 + R\ \cos(\phi) \tag{2.34}$$

where ϕ is the azimuth angle. Using Eqs. (2.32), (2.33) and (2.34), i and j is computed for the point specified by θ_2 and ϕ_2 and for the point specified by θ_2' and ϕ_2'. The distance in the image between these two points is then computed.

To compute the potential error from the uncertainty in camera placement in the event that the correct matching point is found, we compute the difference between the true height and the computed height with camera 2 mispositioned by its uncertainty.

The height of a point with both WSIs perfectly positioned is given by Eq. (2.16). The computed height of a point with one WSI mispositioned is

$$h_{misposition} = \frac{d}{\tan(\theta_1)\sin(\phi_1) - \sin(\phi_2')\tan(\theta_2')} \tag{2.35}$$

where ϕ_2' and θ_2' are the azimuth and zenith of the point, respectively, at site 2 with the WSI mispositioned.

Solving Eqs. (2.28) and (2.30) for x, setting the results equal and solving for $\sin(\phi_2')$ gives

$$\sin(\phi_2') = \frac{\sin(\phi_2)\sqrt{x^2+y^2} + e_x}{\sqrt{(x+e_x)^2 + (y+e_y)^2}} \tag{2.36}$$

Solving Eqs. (2.29) and (2.31) for h, setting the results equal and solving for $\tan(\theta_2')$ gives

$$\tan(\theta_2') = \frac{\tan(\theta_2)\sqrt{(x+e_x)^2 + (y+e_y)^2}}{\sqrt{x^2+y^2}} \tag{2.37}$$

Substituting Eqs.(2.36) and (2.37) into Eq. (2.35) and rearranging gives

$$h_{misposition} = \frac{d}{\tan(\theta_1)\sin(\phi_1) - \sin(\phi_2)\tan(\theta_2) + \frac{e_x \tan(\theta_2)}{\sqrt{x^2+y^2}}} \tag{2.38}$$

Using Eq. (2.31), Eq. (2.38) becomes

$$h_{misposition} = \frac{d}{\tan(\theta_1)\sin(\phi_1) - \sin(\phi_2)\tan(\theta_2) + \frac{e_x}{h}} \tag{2.39}$$

Subtracting Eq. (2.39) from Eq. (2.16) gives the error due to misposition of the WSIs.

2.4.4 Uncertainty of Alignment

If two cameras are not perfectly aligned, i.e., they are not pointing in the same compass direction, error will result in the computed height. Rather than quantifying the error that two cameras are misaligned with magnetic north, we will quantify the amount that one camera is misaligned with respect to the other. The effect of the misalignment is an incorrect azimuth for the corresponding point in a camera. In this case, the new azimuth is simply $\phi_2 + \phi_\epsilon$ where ϕ_ϵ is the amount of misalignment.

The process to find the number of pixels that a point moves in the image as a result of the camera being mispositioned is straightforward. Since the zenith angle does not change, a misaligned camera will not change the distance from the center of the image that the matching point appears. So only Eqs. (2.32), (2.33) and (2.34) are needed to compute the x and y position of a point specified by θ_2 and $\phi_2 + \phi_\epsilon$. This difference between this location and the location specified by θ_2 and ϕ_2 is then computed.

To compute the error induced by misalignment in the case where the correct matching point is found, the difference between the true height and the computed height with the camera misaligned is computed. The height of a point with both WSIs perfectly aligned is given by Eq. (2.16). The computed height of a point with the WSIs misaligned is

$$h_{misalignment} = \frac{d}{\tan(\theta_1)\sin(\phi_1) - \sin(\phi_2 + \phi_\epsilon)\tan(\theta_2)} \tag{2.40}$$

where ϕ_ϵ is the amount of misalignment. Subtracting Eq. (2.40) from Eq. (2.16) gives the error due to misalignment of the WSIs.

2.4.5 Uncertainty of Plumbness

If the WSI is not perfectly plumb then the azimuth and zenith of points will be incorrect. Unlike earlier cases where only the relative difference between WSI was important, the relative difference in elevation for example, in this case the absolute difference from plumb of both WSIs must be considered. This results because even if all WSIs were oriented in parallel, if the WSI were not plumb the incorrect height would be computed.

If a WSI is not plumb, the lens will be rotated relative to zenith angle zero. The center of rotation is most likely about the base of the WSI, resulting in the lens being mispositioned in addition to being rotated. However, mispositioning will be on the order of centimeters whereas the distance of clouds is on the order of kilometers. Therefore, any mispositioning will be ignored and it will be assumed that the center of rotation is at the lens.

To compute the potential error from the uncertainty in plumbness, the difference between the true height and the computed height with the WSI out of plumb is computed. To compute how the azimuth and zenith change from a plumb WSI to an out-of-plumb WSI, we will assume without loss of generality that the lens remains fixed and the sky rotates around the lens. If the center of the WSI lens is considered the center of a coordinate system, it is straightforward to compute the change of position, and therefore change of azimuth and zenith, of points in the sky due to rotation.

Let ϕ_1' and θ_1' be the azimuth and zenith of the point, respectively, at site 1 with the WSI out of plumb. Similarly, ϕ_2' and θ_2' are the azimuth and zenith of the point, respectively, at site 2 with the WSI out of plumb.

Figure 2.5 shows the geometry of a point on a cloud with the WSI at the origin. Without loss of generality, assume the distance from the origin to the cloud is 1. With the assumption, and considering Figure 2.5, the following are clear from inspection:

$$\begin{aligned} x &= \sin(\phi)\sin(\theta) \\ y &= \cos(\phi)\sin(\theta) \\ h &= \cos(\theta) \end{aligned}$$

Since we are only concerned with the amount of error to be expected from cameras out plumb, we assume that the WSI is tilted around the x-axis. A point at $[x,\ y,\ h]$ rotated by an angle α around the x-axis gives

$$[\sin(\phi)\sin(\theta),\ \cos(\phi)\sin(\theta),\ \cos(\theta)] \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\alpha) & -\sin(\alpha) \\ 0 & \sin(\alpha) & \cos(\alpha) \end{bmatrix} =$$

$$
\begin{aligned}
&[\sin(\phi)\sin(\theta), \\
&\cos(\phi)\sin(\theta)\cos(\alpha) \;+\; \cos(\theta)\sin(\alpha), \\
&-\cos(\phi)\sin(\theta)\sin(\alpha) \;+\; \cos(\theta)\cos(\alpha)\;]
\end{aligned}
\tag{2.41}
$$

Since it is assumed that the point is distance 1 from the WSI, the following are true:

$$\theta = \arccos(h) \tag{2.42}$$

$$\phi = \arctan\left(\frac{x}{y}\right) \tag{2.43}$$

By substituting the x, y and h components of Eq (2.41) into Eqs. (2.42) and (2.43) we have the new azimuth, ϕ', and the new zenith, θ', due to the rotation of the point around the WSI. Or equivalently, we have the new azimuth and zenith as a result of the WSI being out of plumb.

$$\theta' = \arccos\left(-\cos(\phi)\sin(\theta)\sin(\alpha) \;+\; \cos(\theta)\cos(\alpha)\right) \tag{2.44}$$

$$\phi' = \arctan\left(\frac{\sin(\phi)\sin(\theta)}{\cos(\phi)\sin(\theta)\cos(\alpha) \;+\; \cos(\theta)\sin(\alpha)}\right) \tag{2.45}$$

The process to find the number of pixels that a point moves in the image as a result of the camera being out of plumb is straightforward. Using Eqs. (2.32), (2.33) and (2.34), the x and y position is computed for the point specified by θ_2 and ϕ_2 and for the point specified by θ_2' and ϕ_2'. The distance in the image between these two points is then computed.

In the event that the correct matching point is found, the height of a point with both WSIs perfectly positioned is given by Eq. (2.16). The computed height of a point with two WSIs out of plumb, the height is given by

$$h_{out-of-plumb} = \frac{d}{\tan(\theta_1')\sin(\phi_1') \;-\; \sin(\phi_2')\tan(\theta_2')} \tag{2.46}$$

Computing the difference gives the error in the computed height due to cameras out of plumb.

2.4.6 Uncertainty in Camera Calibration

Up to now, the use of the camera calibration equations (the equations that map an azimuth and zenith to a point in an image) have been used as if they define the projection process perfectly. In fact, they do not. These equations, first shown as Eqs. (2.19), (2.20), and (2.21) were derived by Koehler [8]. Koehler positioned lines at specific zenith angles and measured the number of pixels from the center of the image to where each line appeared in the image. This gave a table of distances from the center of the image as a function of zenith angle. A second-order equation was then fit to the data. This is Eq. (2.19). Koehler reported a half pixel difference between the equation and data for zenith angles greater than or equal to 70 degrees. For zenith angles less than 70, there was no detectable error. But note that this is just the error between the collected data and the best fit equation to describe the data; experimental error is not included.

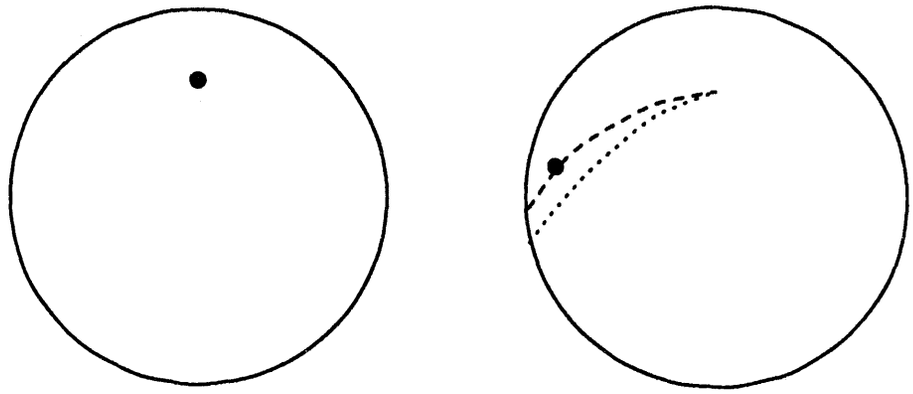


Figure 2.6: Effects of an incorrect calibration equation. A point is shown in the left image and two epi-polar lines are shown in the right image. The correct matching point is shown along the "correct", long-dashed epi-polar line. However, the camera calibration equations may result in the other epi-polar line since the equations are in error in some areas of the image.

Figure 2.6 shows an example of how an incorrect calibration equation results in the correct matching point being off the epi-polar line. A point is shown in the left image and two epi-polar lines are shown in the right image. The correct matching point is shown along the "correct", long-dashed epi-polar line. However, the camera calibration equations may result in the other epi-polar line since the equations are in error in some areas of the image. The undesirable effects of errors in the calibration equations have been exaggerated to show the point. Koehler's experiments show that errors in the calibration equations result in less than a pixel error in areas of interest, i.e., away from the extreme edge of the image.

To compute the error induced by errors in the calibration equation in the case where the correct matching point is found, the difference between the true height and the computed height with incorrect zenith angles is computed. The calibration equations are only used to compute the zenith angle given a point in the image and vice verse. So if the calibration equations are wrong, only the zenith angles are in error. The computed height in this case is

$$h_{miscalibration} = \frac{d}{\tan(\theta_1 + \theta_{\epsilon 1})\sin(\phi_1) - \sin(\phi_2)\tan(\theta_2 + \theta_{\epsilon 2})} \tag{2.47}$$

where $\theta_{\epsilon 1}$ and $\theta_{\epsilon 2}$ are the errors in the zenith angles θ_1 and θ_2, respectively.

Subtracting Eq. (2.47) from Eq. (2.16) gives the error due to errors in the calibration equation. From Koehler [8] we know that $\theta_{\epsilon 1}$ and $\theta_{\epsilon 2}$ are positive and at most 0.33 degrees since the angular resolution along the zenith is one third degree per pixel.

2.4.7 Typical Uncertainties

Typical values for the five previously discussed sources of uncertainty and cloud base height error were chosen and the total potential displacement of pixels in an image was computed. The following values were used:

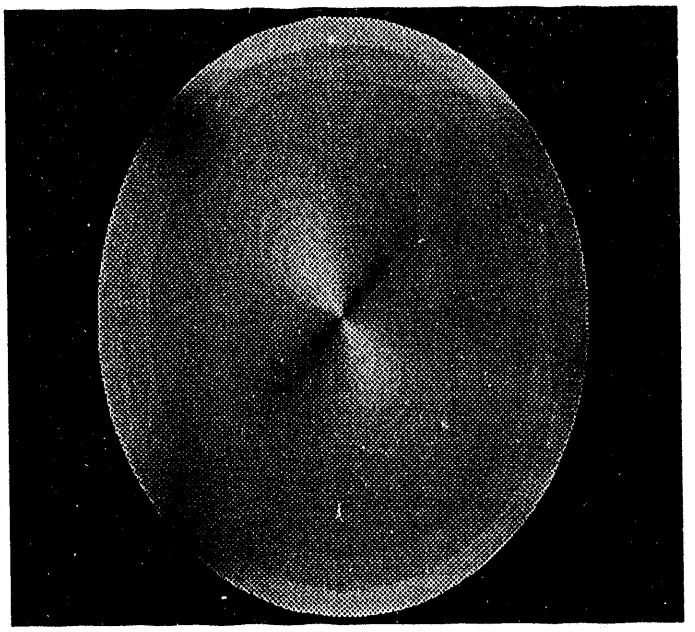


Figure 2.7: This image indicates the most that clouds can be displaced in an image for a typical cloud height. The brightest area indicates a displacement 8 pixels while the darkest area shows a displacement of 3 pixels.

Cloud Height	10000 meters
Uncertainty in x position	100 meters
Uncertainty in y position	100 meters
Uncertainty in elevation	10 meters
Uncertainty in alignment	1°
Uncertainty in plumbness	1°
Uncertainty in calibration Equation	0 pixels if zenith $< 65°$
	0.5 pixels if zenith $65° - 75°$
	1 pixel if zenith $> 75°$

While in practice one source of error could counteract another source of error, in this example all errors were cumulative. Figure 2.7 shows displacement error, with bright indicating more error. The brightest section near the center has pixel displacement of 8 while the dark area also near the center image has pixel displacement of 3. Only the displacement due to a difference in elevation between the cameras is not significant. Roughly, all other sources contribute equally to the displacement shown in Figure 2.7.

When searching along the epi-polar line, the size of the correlation window is 19×19 (see Chapter 6), so it is likely that the correct match can be found at a displacement of 3, but a displacement of 8 will make finding the correct match difficult.

The sensitivity, the amount that a point could vertically move without being detected, is shown in Figure 2.8. The brightest area has error up to 5% (500 meters for this scenario) while the darkest area has error around 1.5% (150 meters for this scenario).

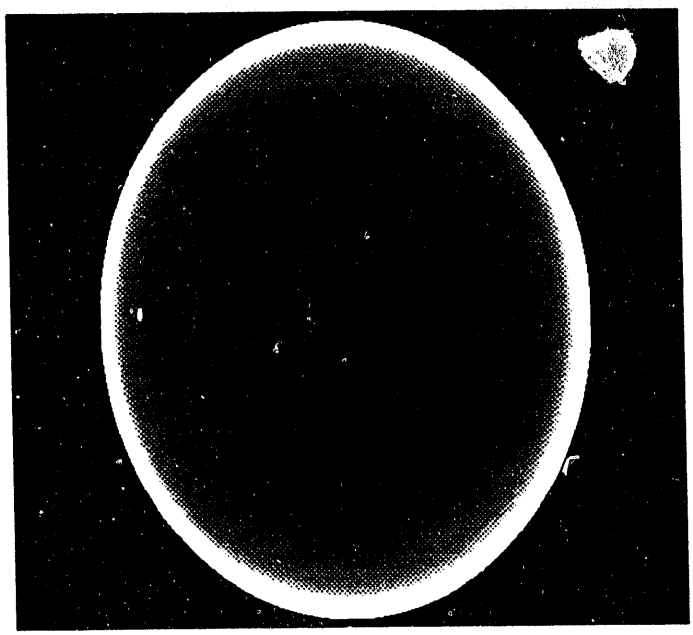


Figure 2.8: This image indicates the worse the error can be for a typical cloud. The brightest area has error up to 5% while the darkest area has error around 1.5%.

Chapter 3

Monocular Optical Flow for Clouds

To compute cloud base heights (CBH) we exploit the fact that modern WSIs provide *sequences* of images. With these sequences, the optical flow field, i.e., a field of vectors where each vector represents the image motion of that point, can be recovered and used to aid in finding corresponding points between image pairs. In this section we will present and justify a hierarchical correlation method for computing optical flow fields for cloud image sequences.

In order to find or develop an appropriate optical flow algorithm it is useful to first list the image properties of the objects which will have their optical flow computed. The two most noteworthy properties of clouds, from the point of view of optical flow algorithms, are as follows. First, they move at a range of speeds, from almost zero pixels per frame to tens of pixels per frame. Second, they are very dynamic, i.e., they change shape.

Optical flow algorithms can be classified into two types, feature-based and gradient-based. Gradient based methods use the variation of the gray-level pattern from one frame to the next to determine the flow. Feature-based algorithms first detect features in images, cloud boundaries for example, then track those features from one frame to the next.

Gradient-based methods are desirable because optical flow can be computed most everywhere and their behavior is well understood. Feature-based methods are desirable because fewer assumptions are required. Disadvantages of gradient-based optical flow methods include: interframe image motion must be small, object motion must be rigid, and the flow must be constant. Disadvantages of feature-based optical flow methods include: performance is not well understood and a dense flow field does not usually result.

In order for the optical flow fields to be useful for finding corresponding points they must be dense, that is, to exist at all pixels. For this reason, standard feature-based methods will not work. However, often the motion of clouds from one frame to the next is greater than can be handled by many gradient-based methods. For these reasons we adopt a method that has elements of both feature-based and gradient-based methods.

A standard correlation algorithm for computing optical flow takes a small neighborhood around a point in one image and compares it to a small neighborhood around points in the next image of the sequence. This approach has both feature-based aspects and gradient-based aspects. It is feature-based in the sense that it "tracks" a feature, the grey-level pattern around a point, from one frame to the next. It is gradient-based in the sense that it examines the gray-level pattern between frames. But unlike a strictly feature-based approached, a flow vector is obtained at every pixel. And unlike a gradient-based approached, large motions

can be dealt with by comparing the small neighborhood around a point to a large number of candidate points in the next image.

We thus use a correlation-based algorithm; however, a straightforward correlation approach as described above is computationally prohibitive if large cloud motion is to be detected. So a hierarchical correlation algorithm is used. With this algorithm large motions can be detected and the computational increase is manageable.

The hierarchical correlation algorithm used is exactly as described by Burt [4]. Motions are first detected at coarse scales and these are refined at finer scales. More specifically, consider two images in an sequence. The two images are "blurred" and subsampled so that the new images are half the size in each dimension as the original. The resulting images are blurred and subsampled again. This process continues until the resulting image is only a few pixels in each dimension. This is called an image pyramid since progressively smaller versions of an image can be viewed as being stacked one upon the other, forming a pyramid.

Given two pyramids from two images in a sequence, every point in the top, smallest level of the first pyramid is correlated against the points in the top, smallest level of the second pyramid. The detected displacements give the optical flow for each pixel in the top level of the pyramid. These vectors are then used by the next level down in pyramid as "suggestions" as to where the actual motion is. Using this information, the finer level only correlates points in the area where the level above indicated the motion occurred. This process continues down to the bottom level. The optical flow field for the bottom level is then the optical flow field for the image.

Vectors in optical flow fields are typically two-dimensional, $\mathbf{V} = (\Delta x,\ \Delta y)$, with Δx and Δy indicating the direction of motion and the length, $\|\mathbf{V}\|$, indicating the speed. An optical flow vector, $\mathbf{V}$, can be converted to a three-dimensional vector, $\mathbf{V_3}$, by adding a temporal component which varies from 1 to 0 as $\|\mathbf{V}\|$ varies from 0 to infinity. This third component represents the speed and is a function of the time between frames and Δx and Δy. We can assume that the time between frames is 1, so the third component is set to 1. The three-dimensional vector is then unit vector. So

$$\mathbf{V_3} = \frac{(\Delta x,\ \Delta y,\ 1)}{\sqrt{(\Delta x)^2 + (\Delta y)^2 + 1}}$$

There are two reasons for converting two-dimension flow vectors to three-dimensional flow vectors. First, in the next section we will describe a method for evaluating optical flow fields which requires three-dimensional flow fields. Second, when using flow fields to find matching points between corresponding images, the correlation needs to be normalized so that areas with large motion are not unjustly favored. But by normalizing, the speed, or length, of two-dimensional vectors becomes meaningless. By using the three-dimensional flow vectors, the speed information is retained in the third component even with normalized correlation.

Below we will describe a method for determining how well an optical flow algorithm performs and use it to show that hierarchical correlation is in fact the algorithm of choice when dealing with cloud sequences. Another correlation algorithm and a gradient-based algorithm will also be used in the comparison. Hierarchical correlation with different parameters will also be compared to determine the optimal settings.

3.1 Performance of Optical Flow

In a domain such as atmospheric imaging there is little accurate groundtruth available, particularly for true cloud motion and its projection into image motion. In such contexts, where there is no available knowledge as to the true motion, there is also no established method for objectively measuring the performance of optical flow algorithms. Thus, competing approaches have been evaluated subjectively and visually, typically by superimposing a flow representation on an animation of the subsequent frames and attempting to observe whether the computed flows line up with the apparent motion. This is clearly unsatisfactory for rigorous analysis.

Our method of measuring the performance of optical flow algorithms has two parts: the generation of a new image sequence from a starting actual image and a sequence of flow fields, and the objective comparison of the generated images with the actual image sequence.

In the first part of our evaluation method, the goal is to simulate the actual image sequence. We start with a seed true image frame and alter it for multiple frames as indicated by the optical flow field calculated on the seed image by a candidate algorithm. The idea is that a flow field sequence which can do this well must be capturing the important essence of the evolution of the imagery. This simulated sequence serves as the basis for both subjective and objective evaluation of the accuracy of the flow. By viewing this simulated sequence it can be visually compared to the actual sequence, providing a subjective indication of the accuracy of a *sequence* of optical flow fields.

In the second part of our evaluation method, the simulated sequence is objectively compared to frames in the actual sequence. There are two specific objective metrics for comparative performance, POOFC and POOFP. POOFC is based on the straightforward correlation of the simulated pixels with their correct values, and is useful in cases where the conservation of intensity assumption is likely to be violated. A related but simpler metric, POOFP, is the percentage of relevant simulated pixels which equal the corresponding pixel in the actual image. These metrics are used to determine the relative performance of a set of representative optical flow algorithms. For both metrics a value of one is best and zero is worst. This work is described in greater detail in [1].

3.2 Optical Flow Results

The optical flow algorithms which we consider here include the gradient-based Uras *et. al.* [16] algorithm and correlation-based approaches. The correlation-based approaches include Anandan's hierarchical algorithm [2], Burt's hierarchical correlation [4] and non-hierarchical ("flat") correlation. There are two types of Burt hierarchical correlation algorithms, 3×3 pyramid and 5×5 pyramid (shown as "3p" and "5p", respectively), indicating the size of the neighborhood used to construct the higher level in the pyramid. The flat correlation algorithm is indicated by "5f" and is essentially the special case case of a pyramid with one level ("5f" denotes that the maximum pixel motion detectable by this algorithm is within a 5×5 neighborhood). In all correlation algorithms, the size of the neighborhood used to compute the correlation was 9×9.

For both the POOFC and POOFP metrics, higher values indicate better performance.

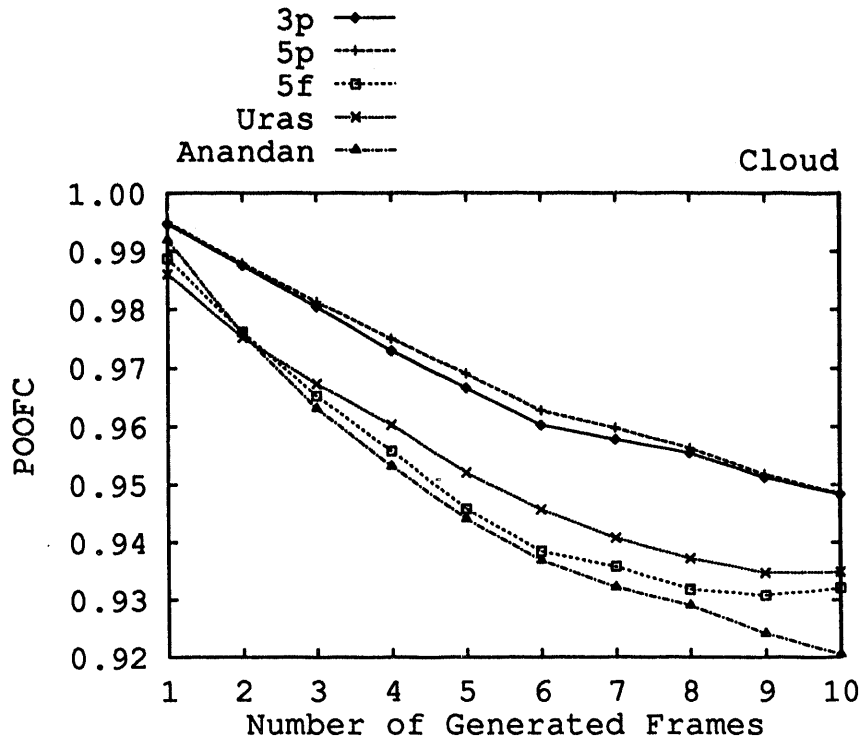
In all cases shown below, the differences between all pairs of POOFC and POOFP values are significant by the tenth frame, and in all but one case are significant by the first frame.

Figures 3.1 and 3.2 show the POOFC and POOFP results for two cloud sequences. The hierarchical correlation algorithms consistently perform well. Based on this, the fact that the Anandan algorithm takes a prohibitively long time to compute, and that 3p is faster to compute than 5p, we were able to determine that 3p is the optical flow algorithm of choice for our cloud sequences. However, the size of the correlation neighborhood still needed to be decided. Figure 3.3 shows the POOFC results for the 3p algorithm as the correlation size varies. From these we were able to determine that, to a point, smaller neighborhood sizes perform better due to the dynamic nature of clouds.

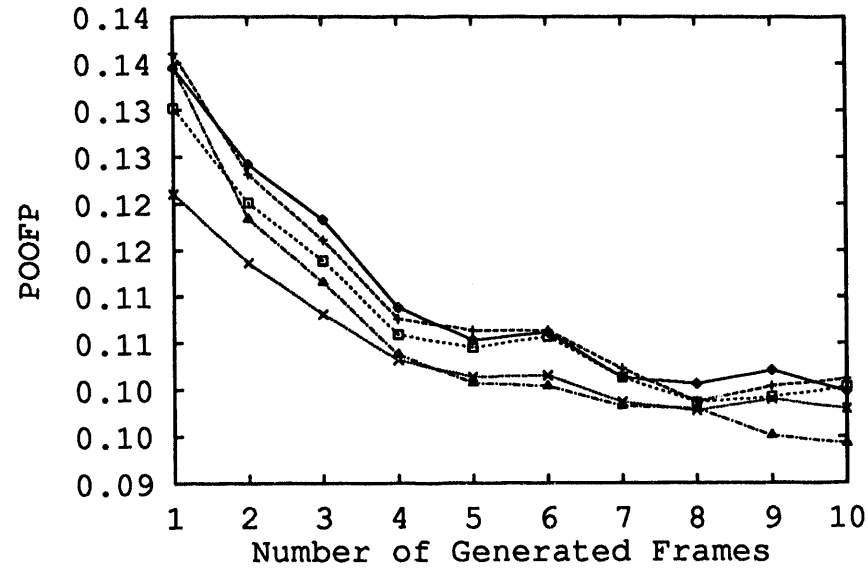
3.3 Using Optical Flow for Cloud Base Height Computation

Now that we have a means of computing optical flow fields, the dynamics of clouds can also be used to find corresponding points between images. The algorithm at this point is as follows:

- Compute the optical flow from WSI 1
- Compute the optical flow from WSI 2
- For every point, p_1, in image 1
  - Compute the epi-polar line for p_1 in image 2. [Section 2.2, Eqs. (2.17)–(2.21)].
  - For each point, p_2, along the epi-polar line
    - Match p_1 and p_2 using both pixel values and the flow field around the points [Section 4.5].
  - Find the best matching point along the epi-polar line. Call this point p_2'.
  - Compute the epi-polar line for p_2' in image 1. [Section 2.2 Eqs. (2.17)–(2.21)].
  - For each point, p_3, along the epi-polar line
    - Match p_2' and p_3 using both pixel values and the flow field around the points.
  - Find the point of maximum correlation value along the epi-polar line. Call this point p_1'.
  - Compute the cloud height using p_1 and p_2' as corresponding points. [Eq. (2.16)].
  - Compute the distance between p_1 and p_1'. This is the confidence of point p_1.

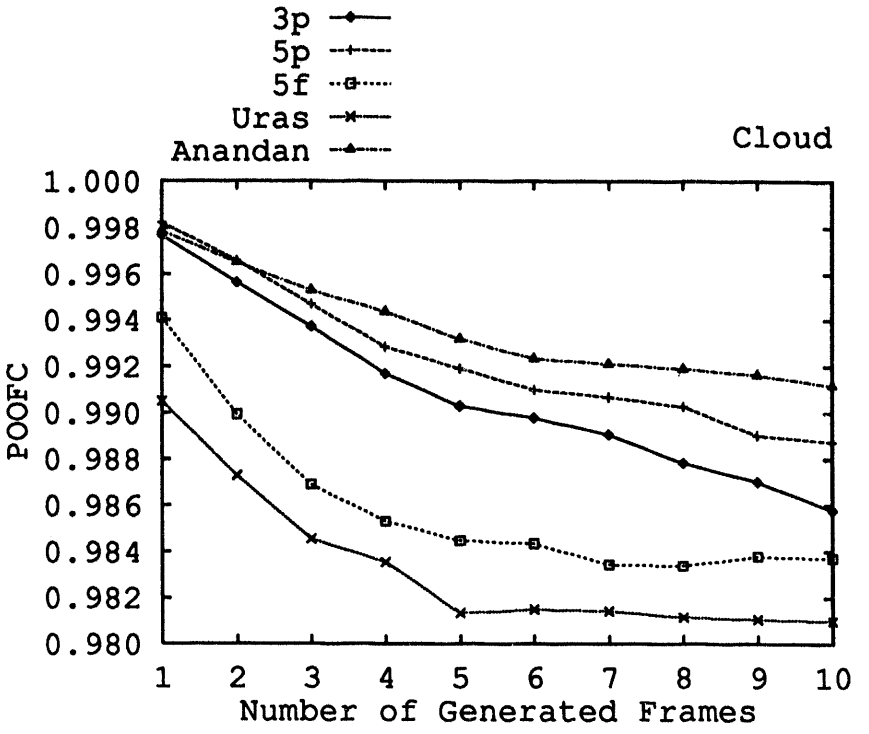


(a)

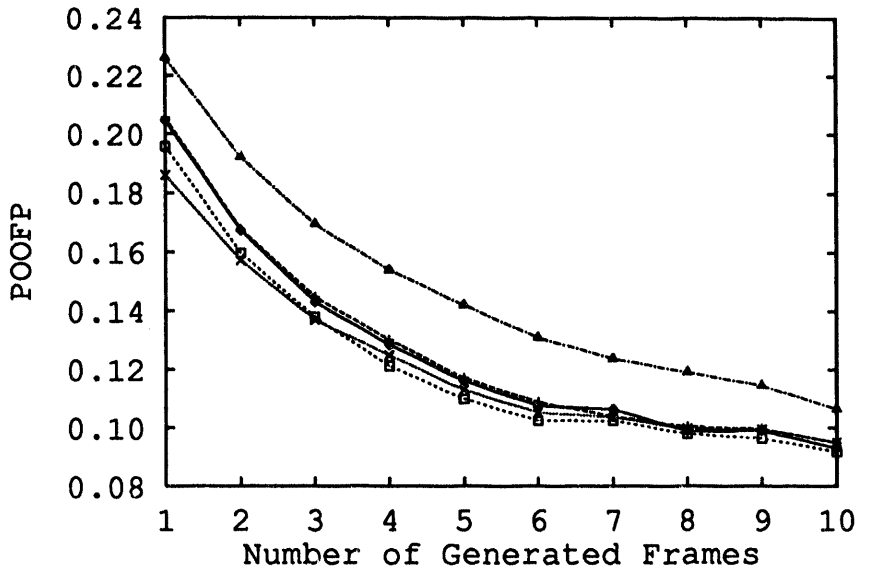


(b)

Figure 3.1: POOFC (a) and POOFP (b) metrics for the five optical flow algorithms given the same cloud sequence. Higher values indicate better performance.

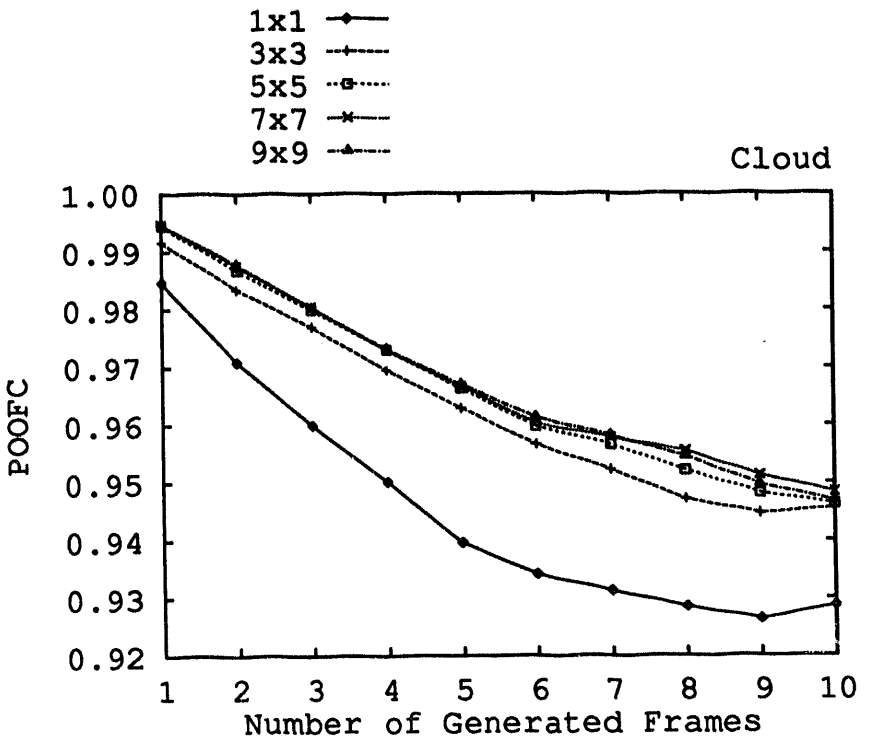


(a)

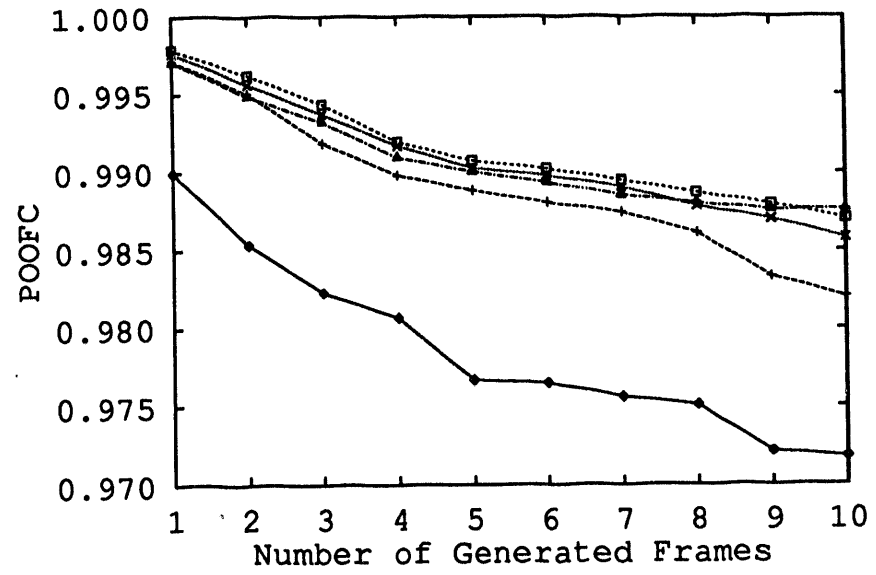


(b)

Figure 3.2: POOFC (a) and POOFP (b) metrics for the five optical flow algorithms given the same cloud sequence. Higher values indicate better performance.



(a)



(b)

Figure 3.3: Different size correlation neighborhoods were used in the 3p correlation algorithm and the resulting flow fields for the cloud sequences were tested. (a) The POOFC metrics for cloud sequence 1. (b) The POOFC metrics for cloud sequence 2.

Chapter 4

Determining The Common Field of Reference

Because of the projective effects of the WSI camera, images from separate WSIs must be transformed so that shape and velocity of the same cloud appears the same in both WSIs. Consider the case where a cloud moves horizontally over a WSI with constant velocity. In the fisheye WSI view, radial distance from the center of the image is proportional to zenith angle. Therefore, as the cloud enters the field of view of the WSI at high zenith angle, a unit displacement of the cloud in the scene produces a smaller displacement in the image than if the cloud were at small zenith angle. So when the cloud enters the field of view it moves slowly, accelerates as it passes overhead, then decelerates as it approaches the horizon. Similarly, the shapes of clouds are compressed as the zenith angle increases.

In order for movement and shape in the image to reflect the movement and shape in the scene regardless of zenith angle, one transforms the WSI image into pseudo-cartesian coordinates [9]. In this section we show that the pseudo-cartesian transformation (PCT) results in an image sequence where a horizontal motion in the scene results in an image motion (or flow vector) that is independent of where that point projects into the image, i.e., it is independent of zenith angle. Similarly, it is shown that horizontal shape in the scene results in an image shape that is independent of zenith angle. The next section describes the PCT. Then it is shown that after PCT-ing images from two WSIs, the corresponding features in both images will have identical shape. Then it is shown how to PCT flow vectors so that corresponding flow vectors from two WSIs will be identical.

4.1 The Pseudo Cartesian Transformation (PCT)

In order to counteract the projective effects of the WSI, the PCT rescales distance from the center of the image to a dependence that varies linearly with the tangent of the zenith angle. Let R_{wsi} and R_{pct} be the distance from the center of the image to a point in WSI and PCT coordinates, respectively. Let θ be the zenith angle. The coordinate transformations are defined as follows [9]. WSI:

$$R_{wsi} = \theta(3.032 + 0.00259\theta) \tag{4.1}$$

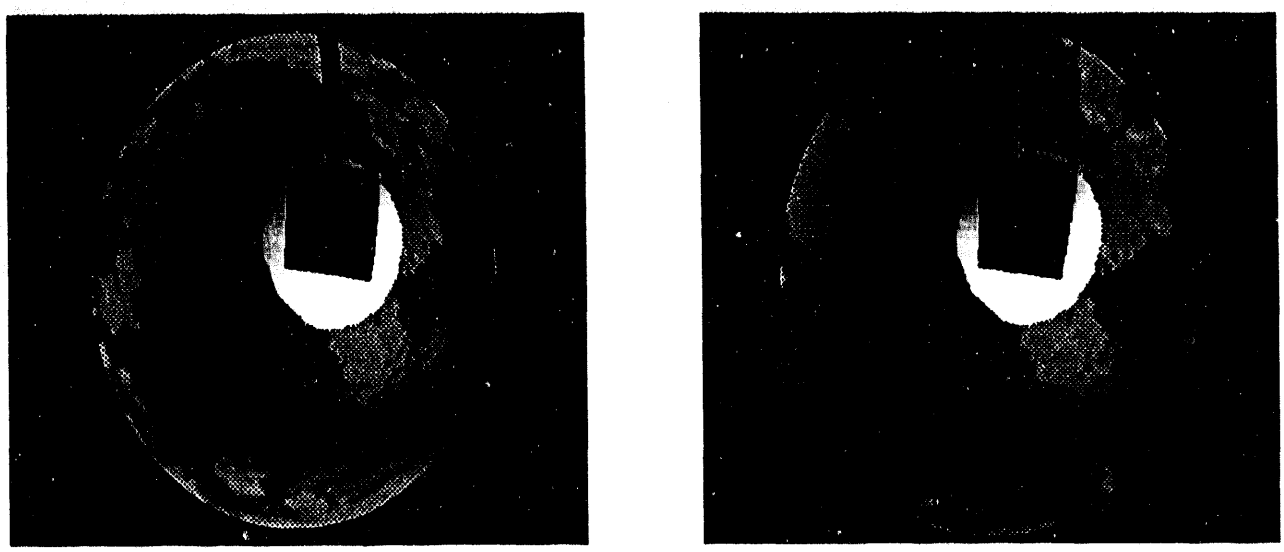


Figure 4.1: (a) A WSI image before PCT. (b) The same image after PCT.

Pseudo-Cartesian:

$$\theta = \arctan\left(\frac{R_{pct}\ \tan(65°)}{235}\right) \tag{4.2}$$

R_{wsi} and R_{pct} are the distances from the center of the image to a point in WSI and PCT coordinates, respectively. Note that Eq. (4.1) is the WSI calibration equation described in Section 2.4.6. The choice of "tan(65°)" and "235" in Eq. 4.2 is arbitrary and will be discussed in Section 4.4.

The procedure for constructing the PCT from the fisheye image begins with identifying a pixel, p_{pct}, at coordinates (x_{pct}, y_{pct}), in the PCT image. The zenith of the point is found using Eq. (4.2). The computed zenith is then substituted into Eq. (4.1) to find the distance, R_{wsi}, that the point is from the center of the image. That pixel is then copied to (x_{pct}, y_{pct}) in the PCT image. The azimuth is not changed by the PCT.

Figure 4.1(a) shows an image before PCT. Figure 4.1(b) shows the same image after PCT. The "stretching" of the image away from the center is clearly visible. This stretching counteracts the compression of the fisheye camera.

When finding corresponding points between pairs of images, we need to perform the PCT on small neighborhoods so that the neighborhoods are in the same frame of reference. Let p_{wsi} be a point, around which a neighborhood is to be PCT-ed. The zenith of p_{wsi} is found using Eq. (4.1). The computed zenith is then substituted into Eq. (4.2) to find the distance, R_{pct}, giving p_{pct}. See Figure 4.2. Each point in the neighborhood around p_{pct} has its distance from the center of the image computed, R_{pct}, and substituted into Eq. (4.2) to find the zenith of that point. Eq. (4.1) is used then to find the point in the WSI that maps to it. So the mapping is first from WSI to pseudo cartesian then from pseudo cartesian to WSI. This is shown by the arrows in Figure 4.2. One cannot simply PCT each point around p_{wsi} to obtain

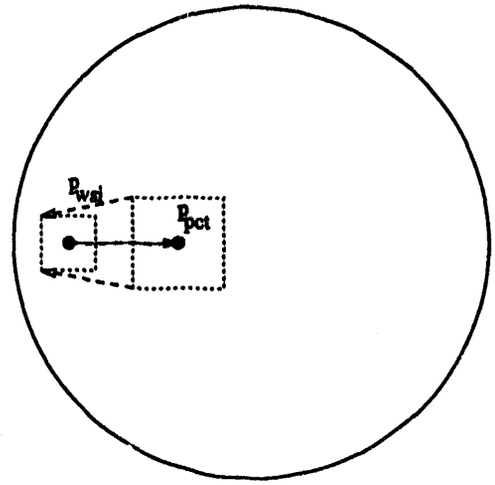


Figure 4.2: If two neighborhoods are to be compared they must be PCT-ed first. The neighborhood around p_{wsi} is one such neighborhood. To PCT it, the center point, p_{wsi} is PCT-ed giving p_{pct}. For each point around p_{pct}, the point that maps to it from around p_{wsi} is then found. The solid arrow shows the mapping from WSI to PCT. The dashed arrow show the mapping from PCT to WSI.

the neighborhood around p_{pct} because not every point in the neighborhood around p_{pct} will necessarily have a value mapped to it.

4.2 Comparing Cloud Shape After PCT

Given a horizontal shape in the sky we need to show how to make the resulting shape in two WSI images identical. Without the pseudo-cartesian transformation, the resulting shapes are clearly not equal. Figure 4.3 shows the geometry of the zenith angle. Positions p and p' are the location of two points. The zenith and azimuth angles are indicated by θ and ϕ, respectively, with the subscript indicating the site. The superscript indicates p or p'.

If it can be shown that the displacement between any two points is the same in PCT-ed images from two WSI sites, then the shape of clouds will appear the same in both images. Therefore, it needs to be shown that

$$x_1' - x_1 = x_2' - x_2 \tag{4.3}$$
$$y_1' - y_1 = y_2' - y_2 \tag{4.4}$$

After PCT, the x and y position are given by [9]:

$$x = 255 + 0.8R\sin(\phi) \tag{4.5}$$
$$y = 240 + R\cos(\phi) \tag{4.6}$$

where

$$R = 235\frac{\tan(\theta)}{\tan(65)} \tag{4.7}$$

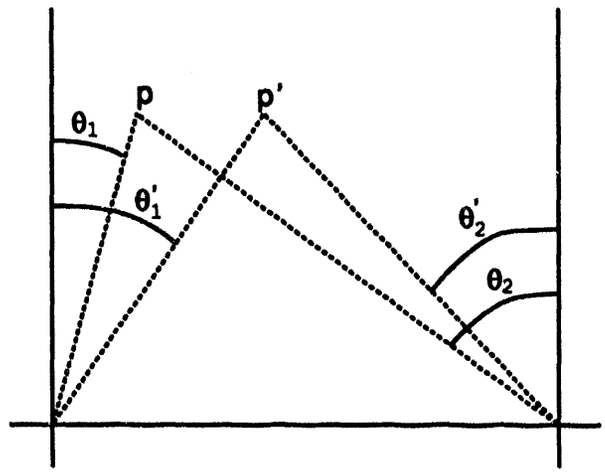


Figure 4.3: The change of zenith as a change of horizontal motion in the scene.

Substituting Eqs. (4.5) and (4.7) into Eq. (4.3) gives:

$$\begin{aligned} x_1' - x_1 &= (255 + 0.8\,R\sin(\phi_1')) - (255 + 0.8\,R\sin(\phi_1)) \\ &= \frac{(0.8)\,(235)}{\tan(65)}\left(\tan(\theta_1')\sin(\phi_1') - \tan(\theta_1)\sin(\phi_1)\right) \end{aligned} \tag{4.8}$$

Similarly,

$$x_2' - x_2 = \frac{(0.8)\,(235)}{\tan(65)}\left(\tan(\theta_2')\sin(\phi_2') - \tan(\theta_2)\sin(\phi_2)\right) \tag{4.9}$$

The equations describing the relationship between sites 1 and 2 are given by:

$$\phi_2 = \arctan\left(\tan(\phi_1) - \frac{d}{h\tan(\theta_1)\cos(\phi_1)}\right) \tag{4.10}$$

$$\theta_2 = \arctan\left(\frac{h\tan(\theta_1)\sin(\phi_1) - d}{h\sin(\phi_2)}\right) \tag{4.11}$$

Substituting Eq. (4.11) into Eq. (4.9) gives:

$$\begin{aligned} x_2' - x_2 &= \frac{(0.8)\,(235)}{\tan(65)}\left(\frac{h'\tan(\theta_1')\sin(\phi_1') - d}{h'\sin(\phi_2')}\sin(\phi_2') - \frac{h\tan(\theta_1)\sin(\phi_1) - d}{h\sin(\phi_2)}\sin(\phi_2)\right) \\ &= \frac{(0.8)\,(235)}{\tan(65)}\left(\frac{h'\tan(\theta_1')\sin(\phi_1') - d}{h'} - \frac{h\tan(\theta_1)\sin(\phi_1) - d}{h}\right) \end{aligned} \tag{4.12}$$

Assuming that $h' = h$, i.e., assuming that the shape in the scene is horizontal,

$$\begin{aligned} x_2' - x_2 &= \frac{(0.8)\,(235)}{\tan(65)}\left(\frac{h\tan(\theta_1')\sin(\phi_1') - h\tan(\theta_1)\sin(\phi_1)}{h}\right) \\ &= \frac{(0.8)\,(235)}{\tan(65)}\left(\tan(\theta_1')\sin(\phi_1') - \tan(\theta_1)\sin(\phi_1)\right) \end{aligned}$$

which equals Eq. (4.8).

A similar procedure is performed for the y component. Substituting Eqs. (4.6) and (4.7) into Eq. (4.4) gives:

$$\begin{aligned} y_1' - y_1 &= (255 + R\cos(\phi_1')) - (255 + R\cos(\phi_1)) \\ &= \frac{235}{\tan(65)} \left(\tan(\theta_1')\cos(\phi_1') - \tan(\theta_1)\cos(\phi_1)\right) \end{aligned} \tag{4.13}$$

Similarly,

$$y_2' - y_2 = \frac{235}{\tan(65)} \left(\tan(\theta_2')\cos(\phi_2') - \tan(\theta_2)\cos(\phi_2)\right) \tag{4.14}$$

Substituting Eq. (4.11) into Eq. (4.14) gives:

$$\begin{aligned} y_2' - y_2 &= \frac{235}{\tan(65)} \left(\frac{h' \tan(\theta_1')\sin(\phi_1') - d}{h'\sin(\phi_2')} \cos(\phi_2') - \frac{h\tan(\theta_1)\sin(\phi_1) - d}{h\sin(\phi_2)} \cos(\phi_2) \right) \\ &= \frac{235}{\tan(65)} \left(\frac{h' \tan(\theta_1')\sin(\phi_1') - d}{h'\tan(\phi_2')} - \frac{h\tan(\theta_1)\sin(\phi_1) - d}{h\tan(\phi_2)} \right) \end{aligned} \tag{4.15}$$

Substituting Eq. (4.10) into Eq. (4.15) gives:

$$\begin{aligned} y_1' - y_1 &= \frac{235}{\tan(65)} \left(\frac{h' \tan(\theta_1')\sin(\phi_1') - d}{h'(\tan(\phi_1') - \frac{d}{h'\tan(\theta_1')\cos(\phi_1')})} - \frac{h\tan(\theta_1)\sin(\phi_1) - d}{h(\tan(\phi_1) - \frac{d}{h\tan(\theta_1)\cos(\phi_1)})} \right) \\ &= \frac{235}{\tan(65)} \left(\frac{h' \tan(\theta_1')\sin(\phi_1') - d}{\frac{h'\tan(\theta_1')\sin(\phi_1') - d}{\tan(\theta_1')\cos(\phi_1')}} - \frac{h\tan(\theta_1)\sin(\phi_1) - d}{\frac{h\tan(\theta_1)\sin(\phi_1) - d}{\tan(\theta_1)\cos(\phi_1)}} \right) \\ &= \frac{235}{\tan(65)} \left(\tan(\theta_1')\cos(\phi_1') - \tan(\theta_1)\cos(\phi_1)\right) \end{aligned}$$

which equals Eq. (4.13).

By showing that the displacement between any two points is the same in PCT-ed images from two WSI sites, we have shown that the shape of horizontal features in the sky will appear the same in both images, and thus that PCT will permit cloud shapes to be properly matched.

4.3 PCT-ing Flow Vectors

In this section it is shown that if WSI images are PCT-ed and flow fields computed, then corresponding vectors will be identical. However, as explained in Chapter 3 optical flow fields are computed on unPCT-ed images. Therefore, the resulting flow fields must be PCT-ed so that corresponding flow vectors are equivalent. By PCT-ing the flow fields, corresponding vectors will be equal.

There are two ways that flow vectors can be altered: they can repositioned and their length and/or orientation can be changed. In the next subsection it is shown that PCT-ing flow vectors results in corresponding vectors being equal. The following subsection then discusses how to reposition flow vectors.

4.3.1 Changing Length and Orientation of Vectors

Given a horizontal movement of a point in the sky we need to show how to make the resulting flow vector in two WSI images equal. Without the pseudo-cartesian transformation, the resulting flow vectors are clearly not equal. But with PCT the flow will be equal. The procedure for showing this is identical to the method used to show that horizontal shapes in two WSI images are identical after PCT-ing. In Section 4.2 it was shown that the relative spacing between two points at the same altitude will be identical in two WSI images if the images are PCT. If those two points are thought of not as two separate points, but rather the same point at two different times, then the same derivation can be applied.

Let p and p' be the location of a point at time t and t', respectively. The zenith and azimuth angles are indicated by θ and ϕ, respectively, with the subscript indicating the site. The superscript indicates p or p'. It needs to be shown that

$$x_1' - x_1 = x_2' - x_2 \tag{4.16}$$
$$y_1' - y_1 = y_2' - y_2 \tag{4.17}$$

After PCT, the x and y position are given by [9]:

$$x = 255 + 0.8R\sin(\phi) \tag{4.18}$$
$$y = 240 + R\cos(\phi) \tag{4.19}$$

where

$$R = 235\frac{\tan(\theta)}{\tan(65)} \tag{4.20}$$

Substituting Eqs. (4.18) and (4.20) into Eq. (4.16) gives:

$$\begin{aligned} x_1' - x_1 &= (255 + 0.8\,R\sin(\phi_1')) - (255 + 0.8\,R\sin(\phi_1)) \\ &= \frac{(0.8)\,(235)}{\tan(65)}\left(\tan(\theta_1')\sin(\phi_1') - \tan(\theta_1)\sin(\phi_1)\right) \end{aligned} \tag{4.21}$$

Similarly,

$$x_2' - x_2 = \frac{(0.8)\,(235)}{\tan(65)}\left(\tan(\theta_2')\sin(\phi_2') - \tan(\theta_2)\sin(\phi_2)\right) \tag{4.22}$$

The equations describing the relationship between sites 1 and 2 are given by:

$$\phi_2 = \arctan\left(\tan(\phi_1) - \frac{d}{h\tan(\theta_1)\cos(\phi_1)}\right) \tag{4.23}$$
$$\theta_2 = \arctan\left(\frac{h\tan(\theta_1)\sin(\phi_1) - d}{h\sin(\phi_2)}\right) \tag{4.24}$$

Substituting Eq. (4.24) into Eq. (4.22) gives:

$$\begin{aligned} x_2' - x_2 &= \frac{(0.8)\,(235)}{\tan(65)}\left(\frac{h'\tan(\theta_1')\sin(\phi_1') - d}{h'\sin(\phi_2')}\sin(\phi_2') - \frac{h\tan(\theta_1)\sin(\phi_1) - d}{h\sin(\phi_2)}\sin(\phi_2)\right) \\ &= \frac{(0.8)\,(235)}{\tan(65)}\left(\frac{h'\tan(\theta_1')\sin(\phi_1') - d}{h'} - \frac{h\tan(\theta_1)\sin(\phi_1) - d}{h}\right) \end{aligned} \tag{4.25}$$

Assuming that $h' = h$, i.e., assuming the point undergoes horizontal motion,

$$\begin{aligned} x_2' - x_2 &= \frac{(0.8)\ (235)}{\tan(65)} \left(\frac{h \tan(\theta_1') \sin(\phi_1') \ - \ h \tan(\theta_1) \sin(\phi_1)}{h} \right) \\ &= \frac{(0.8)\ (235)}{\tan(65)} \left(\tan(\theta_1') \sin(\phi_1') \ - \ \tan(\theta_1) \sin(\phi_1) \right) \end{aligned}$$

which equals Eq. (4.21).

A similar procedure (shown in Section 4.2) is performed for the y component.

4.3.2 Repositioning Flow Vectors

Once a flow vector is altered, it must be repositioned. If repositioning is not performed, then while corresponding vectors will be equal, the relative placement between corresponding vectors will not be correct. After a vector is PCT-ed the base of the vector, i.e., the pixel that the vector is flowing from, is PCT-ed. The new position is the new location of the flow vector. The resulting vector and its position is now identical to the vector that would have resulted from computing the flow field on a sequence of PCT-ed images.

4.4 Setting the Parameters of the PCT

The choice of "tan(65°)" and "235" as standard parameters for the PCT is arbitrary. After explaining why the parameters should change continuously, depending upon the location of points being compared, we will show how to choose the parameters.

When an image is PCT-ed the outer portion of the image is expanded and the inner portion of the image is compressed. The standard parameters of the PCT were chosen so as to provide a balance between the expansion and the compression. Two images and their flow fields could be PCT-ed once with any suitable parameters and the resulting images and flow fields could be correlated to find corresponding points. However, in general, when correlating two neighborhoods from the two images and flow fields, the PCT parameters will not be optimal. It is possible that both neighborhoods will be compressed, throwing away useful pixels, or that both neighborhoods will be expanded, copying pixels resulting in an effectively smaller correlation neighborhood.

One way to achieve a balance is to choose the parameters so that the amount of compression and expansion of the two neighborhoods is minimized. When a neighborhood is PCT-ed, it can expand or compress in the radial dimension (along the zenith) and/or in the circular dimension (along the azimuth). Given two neighborhoods, the expansion and compression along these two dimensions will be formulated. Then the parameters of the PCT can be chosen such that the amount of expansion and compression is minimized.

Figure 4.4 shows two points, p_1 and p_2, from different WSI images, neighborhoods around which are to be PCT-ed then correlated. Clearly, since the distance from the center of the image of the two points differ, the outer neighborhood will have fewer pixels than the inner neighborhood when doing the correlation. Figure 4.4 also shows in gray the neighborhoods of the two points after PCT. Now the neighborhoods are the same size but the inner neighborhood has skipped over some pixels and the outer neighborhood has multiply copied pixels.

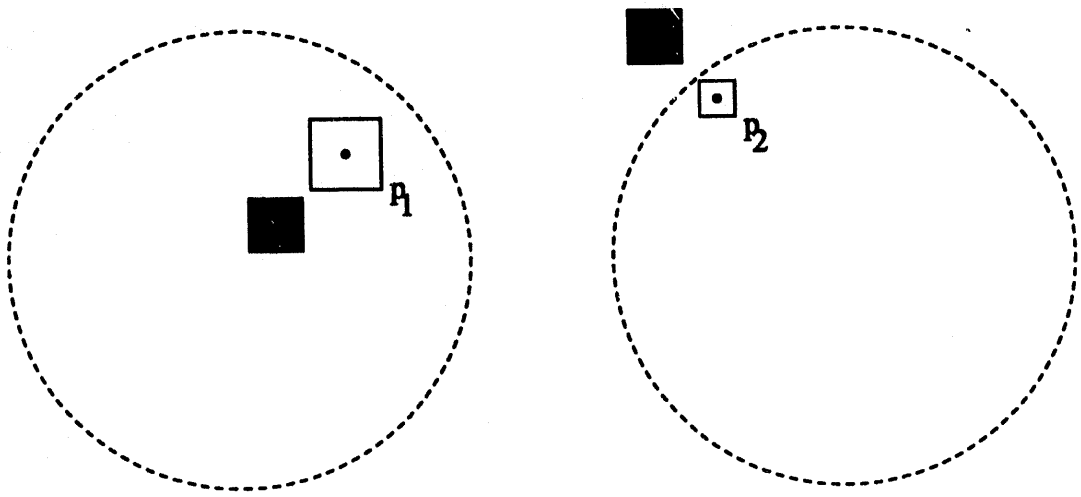


Figure 4.4: Two points, p_1 and p_2, from different WSI images, are to be PCT-ed and then correlated. The neighborhoods after PCT are shown in gray. The two resulting gray neighborhoods are the same size so can be correlated, but in the left image the WSI neighborhood expanded whereas in the right image the WSI neighborhood contracted.

The skipping of pixels resulted because the PCT can move pixels toward the center along the zenith direction. Since this compresses pixels together, some pixels must be skipped. Pixels are copied more than once because the pixels can be moved away from the center of the image by the PCT, therefore, some pixels must be copied multiple times to fill in gaps. This is inevitable, but now we will quantify these effects and show how to minimized them.

The camera calibration equation (discussed in Section 2.4.6) relating the distance from the center of the image to a point at zenith θ before PCT is:

$$R_f(\theta) \;=\; \theta\,(173.72 \;-\; 8.5\,\theta)$$

where θ is in radians. After PCT we have:

$$R_p(\theta) \;=\; \frac{235}{\tan(65°)}\tan(\theta)$$

where θ is in radians. Combining $\frac{235}{\tan(65°)}$ into to one term, C, to be determined below, we have:

$$R_p(\theta) \;=\; C\,\tan(\theta)$$

In order to find the amount of compression or expansion along the radial direction we consider a small neighborhood around a point which is to be PCT-ed. The radial extent of the neighborhood corresponds to some small change in zenith angle, $\Delta\theta$. In order for there to be no compression or expansion, the radial extent of the neighborhood must be the same before and after PCT for the some $\Delta\theta$. That is, we want to minimize the function $g_i(C)$ (The subscript on g indicates the point being considered.)

$$g_i(C) \;=\; R_p(\theta_i + \Delta\theta_i) \;-\; R_p(\theta_i) \;-\; R_f(\theta_i + \Delta\theta_i) \;-\; R_f(\theta_i)$$

Equivalently, the square of the function can be minimized. This gives

$$\begin{aligned} g_i(C) &= [(R_p(\theta_i + \Delta\theta_i) - R_p(\theta_i)) - (R_f(\theta_i + \Delta\theta_i) - R_f(\theta_i))]^2 \\ &= [((\theta_i + \Delta\theta_i)(173.72 - 8.5(\theta_i + \Delta\theta_i)) - \theta_i(173.72 - 8.5\theta_i)) - \\ &\quad (C\tan(\theta_i + \Delta\theta_i) - C\tan(\theta_i))]^2 \\ &= \left[(173.72\Delta\theta_i - 17.0\theta_i\Delta\theta_i - 8.5\Delta\theta_i^2) - C(\tan(\theta_i + \Delta\theta_i) - \tan(\theta_i))\right]^2 \end{aligned} \quad (4.26)$$

The angular distance of a neighborhood before PCT is given by

$$\begin{aligned} S_f(\theta) &= \alpha\, R_f \\ S_f(\theta) &= \alpha\, \theta\, (173.72 - 8.5\, \theta) \end{aligned} \quad (4.27)$$

where α is the angle which subtens the neighborhood. θ and α are in radians. The angular distance of a neighborhood after PCT is given by

$$\begin{aligned} S_p(\theta) &= \alpha\, R_p \\ S_p(\theta) &= \alpha\, C\tan(\theta) \end{aligned} \quad (4.28)$$

where again α is the angle which subtens the neighborhood. Since the PCT only moves points radially, α is the same before and after PCT.

In order to minimized the amount of circular compression or expansion due to the PCT, we want the difference between Eqs. (4.27) and (4.28) to be minimized. Equivalently, the square of the difference of Eqs. (4.27) and (4.28) can be minimized, giving:

$$\begin{aligned} h_i(C) &= [S_p(\theta_i) - S_f(\theta_i)]^2 \\ &= [\alpha_i\theta_i(173.72 - 8.5\theta_i) - C\alpha_i\tan(\theta_i)]^2 \end{aligned} \quad (4.29)$$

The subscript on h indicates the point being considered.

Eqs (4.26) and (4.29) give the amount of compression or expansion for one neighborhood when PCT-ed. However, we need to PCT two neighborhoods which will then be correlated. In order to minimize the compression and expansion for both neighborhoods we sum Eqs (4.26) and (4.29) for both neighborhoods and minimize. This gives:

$$\begin{aligned} & g_1(C) + g_2(C) + h_1(C) + h_2(C) \\ = \; & [R_f(\theta_1 + \Delta\theta_1) - R_p(\theta_1)]^2 + [R_f(\theta_2 + \Delta\theta_2) - R_p(\theta_2)]^2 + \\ & [S_f(\theta_1 + \Delta\theta_1) - S_p(\theta_1)]^2 + [S_f(\theta_2 + \Delta\theta_2) - S_p(\theta_2)]^2 \\ = \; & \left[(173.72\Delta\theta_1 - 17.0\theta_1\Delta\theta_1 - 8.5\Delta{\theta_1}^2) - C(\tan(\theta_1 + \Delta\theta_1) - \tan(\theta_1))\right]^2 + \\ & \left[(173.72\Delta\theta_2 - 17.0\theta_2\Delta\theta_2 - 8.5\Delta{\theta_2}^2) - C(\tan(\theta_2 + \Delta\theta_2) - \tan(\theta_2))\right]^2 + \\ & [\alpha_1\theta_1(173.72 - 8.5\theta_1) - C\alpha_1\tan(\theta_1)]^2 + \\ & [\alpha_2\theta_2(173.72 - 8.5\theta_2) - C\alpha_2\tan(\theta_2)]^2 \end{aligned}$$

We want the value of C which minimized this. This is just a parabola so the minimum is found by computing the derivative with respect to C, setting the result equal to zero and solving for C.

Appropriate values for α and $\Delta\theta$ still need to be found. In the discussion below, subscripts on $\Delta\theta$ and α will be dropped, but reference to $\Delta\theta$ and α will refer to all $\Delta\theta$'s and α's. One approach to PCT-ing a neighborhood would be to PCT every point in the neighborhood, effectly taking a square patch in WSI coordinates and putting it in PCT coordinates. If done in this way, $\Delta\theta$ and α are easily determined from the size of the neighborhood in WSI coordinates. However, as described in Section 4.1, using this method may result in pixels in the PCT neighborhood which have no value mapped to them as the PCT does not map pixels one to one. In order to prevent this, the mapping is done from PCT coordinates to WSI coordinates as described in Section 4.1. In this case $\Delta\theta$ and α would have to be determined from the neighborhood in PCT coordinates. But this cannot be done without first PCT-ing the neighborhood. But this cannot be done until $\Delta\theta$ and α are determined. In other words, there is a circular dependency. This dependency is eliminated by assuming that the values of $\Delta\theta$ and α for a square WSI neighborhood coordinates will result in the same minimizing value of C as would result using a square PCT neighborhood around the two points to be PCT-ed. So, $\Delta\theta$ and α are determined from square neighborhoods in WSI coordinates.

4.5 The Algorithm

At this point everything necessary to compute cloud base heights has been described. The complete algorithm is as follows:

- Compute the optical flow from WSI 1 [Chapter 3].
- Compute the optical flow from WSI 2 [Chapter 3].
- For every point, p_1, in image 1
  - Compute the epi-polar line for p_1 in image 2. [Section 2.2 Eqs. (2.17)–(2.21)].
  - For each point, p_2, along the epi-polar line
    - Perform PCT on the pixel values in neighborhoods around p_1 and p_2 [Section 4.1].
    - Perform PCT on the optical flow fields in neighborhoods around p_1 and p_2 [Section 4.3].
    - Perform a 4D correlation of the PCT-ed pixel and flow neighborhoods.
  - Find the maximum correlation value along the epi-polar line. Call this point p_2'.
  - Compute the epi-polar line for p_2' in image 1. [Section 2.2 Eqs. (2.17)–(2.21)].
  - For each point, p_3, along the epi-polar line
    - Perform PCT on the pixel values in neighborhoods around p_2' and p_3 [Section 4.1].
    - Perform PCT on the optical flow fields in neighborhoods around p_2' and p_3 [Section 4.3].
    - Perform a 4D correlation of the PCT-ed pixel and flow neighborhoods.

- Find the point of maximum correlation value along the epi-polar line. Call this point p_1'.
- Compute the cloud height using p_1 and p_2' as corresponding points. [Eq. (2.16)].
- Compute the distance between p_1 and p_1'. This is the confidence of point p_1.

4.6 The Effects of Non-Horizontal Cloud Shape and Motion

In the formulations in Sections 4.2 and 4.3.1 the assumption of constant height and motion was not necessary to show $y_1' - y_1 = y_2' - y_2$. So if cloud shape is not horizontal this result is unaffected. However, the same cannot be said of the x component.

The effect of non-horizontal shape can be examined by computing the difference between Eqs. (4.12) and (4.8). If $h' = h$ then this difference is zero. However, without this assumption we find

$$
\begin{aligned}
(x_1' - x_1) - (x_2' - x_2) &= \frac{(0.8)\ (235)}{\tan(65)} \left(\tan(\theta_1')\sin(\phi_1') - \tan(\theta_1)\sin(\phi_1)\right) - \\
&\quad \frac{(0.8)\ (235)}{\tan(65)} \left(\frac{h'\tan(\theta_1')\sin(\phi_1') - d}{h'} - \frac{h\tan(\theta_1)\sin(\phi_1) - d}{h}\right) \\
&= \frac{(0.8)\ (235)}{\tan(65)} \left(\tan(\theta_1')\sin(\phi_1') - \tan(\theta_1)\sin(\phi_1) - \right. \\
&\qquad \frac{h'\tan(\theta_1')\sin(\phi_1')}{h'} + \frac{h\tan(\theta_1)\sin(\phi_1)}{h} + \\
&\qquad \left. d\left(\frac{1}{h'} - \frac{1}{h}\right)\right) \\
&= \frac{(0.8)\ (235)}{\tan(65)}\, d\left(\frac{1}{h'} - \frac{1}{h}\right)
\end{aligned}
$$

By letting x_1' and x_1 be the x position of the same point at two separate times, the same procedure gives the potential error from non-horizontal motion.

Chapter 5

Real and Simulated Test Data

Cloud base heights have been computed for various synthetic cloud image sequences and for real WSI data collected under conditions designed to be similar to those that will actually be experienced in field studies. The generation of synthetic data and the collection of real data is described in this section.

5.1 Simulated Data

The simulated sequences were created with a cloud scene simulation model, developed by Cianciolo [6] which uses stochastic field generation techniques and knowledge of atmospheric structure and physics to model four-dimensional (3 spatial and 1 temporal) cloud scenes, represented by liquid water content (LWC) values. To this we attached a cloud density model to derive radiance fields from the LWC volumes. Synthetic images are then generated by projecting a cloud scene using a WSI camera model that is identical to an actual WSI. Since the cloud scene has a temporal component, it can be projected at a sequence of times, creating an image sequence that captures cloud evolution and motion.

The cloud scene simulation model is a fractal-based model which generates a time series of cloud fields. The sky is represented as a cube of data with each cell in the cube having some amount of liquid water. If the liquid water content is sufficiently high in a cell, then that cell is considered to be part of a cloud. By varying the distribution and amount of water in the cube, different types of clouds are produced. To generate a cloud field, the cloud cover, cloud type, cloud base and top, horizontal domain size, grid resolution and the atmospheric temperature, wind and humidity profiles, etc. are given as inputs. By tuning the parameters in the model, the morphology and texture of different cloud types including cumulus, stratus and cirrus can be simulated. Also, the input wind profile can be used to generate a time series of cloud fields. Figure 5.1 shows an example of a synthetic cloud field generated in this manner.

The simulated 3D cloud field is then used to form 2D WSI images. This is done by using a simple cloud illumination algorithm to generate the illumination effect of the cloud surface. The following assumptions are made.

- The incoming light is parallel.

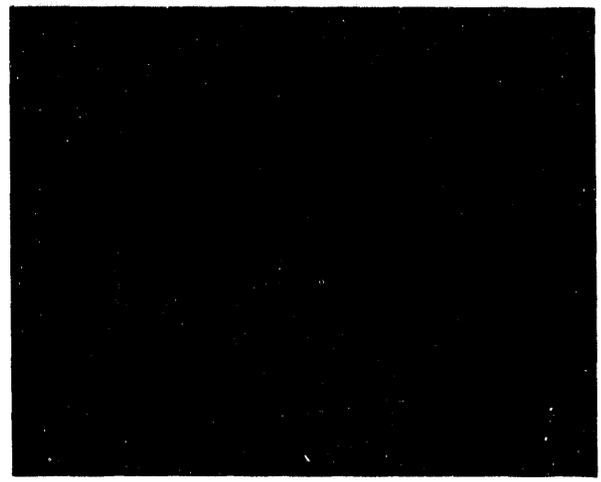


Figure 5.1: An example of a synthetic cloud field generated with the TASC fractal-based model.

- Illumination of the cloud surface is inversely proportional to the sum of the water content along the light path which is in the same direction as the incoming light.
- Light is scattered uniformly in all directions in the clear air and there is no attenuation effect in the clear air.
- There are no reflection effects on the cloud surface.

Because of the complicated optical properties of clouds in the atmosphere, these assumptions are a simplification of the real-world process. Figure 5.2 shows the illumination process of points on the cloud surface and their projection into two WSI cameras. The third assumption states that light is scattered uniformly in all directions, therefore, the gray level of that point in an image is the same regardless of the WSIs position.

With the illumination of the cloud surface points determined, the final phase is to generate WSI images in a manner that simulates the real WSI cameras. For a WSI with a specified angular resolution, each non-occluded cloud surface point visible to the WSI is projected to a WSI image. Given a point on a cloud surface at zenith and azimuth θ and ϕ, respectively, the location of that point in an image is given by the following equations.

$$d = \theta(3.032 - 0.00259\theta) \quad (5.1)$$

$$x = 255 + 0.8d\sin(\phi) \quad (5.2)$$

$$y = 240 + d\cos(\phi) \quad (5.3)$$

d is the distance from the center of the image to a point. These equations, first shown in Section 2.2, were derived by Koehler [9] to describe the projection of the real WSI camera.

Since the spatial resolution of a WSI pixel decreases while the radial distance of the point to the WSI increases, and the data points are discrete, interpolation and smoothing are included in the projection. When the spatial resolution of the WSI is finer than that of

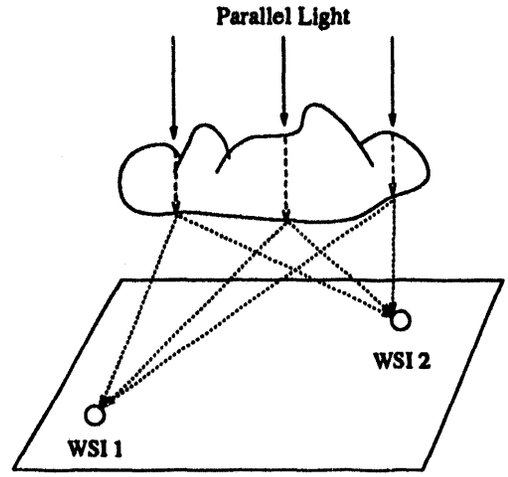


Figure 5.2: Light rays enter the cloud in parallel. The LWC is summed along each ray and each summation is used as the illumination of the point where the ray exits the cloud. The illumination is the same regardless of viewing position underneath the cloud.

the 3D cloud field, linear interpolation is used to fill the pixels in the WSI image. When the spatial resolution of WSI is more coarse than that of the 3D cloud field, the point on the cloud surface that projects closest to the WSI point is used.

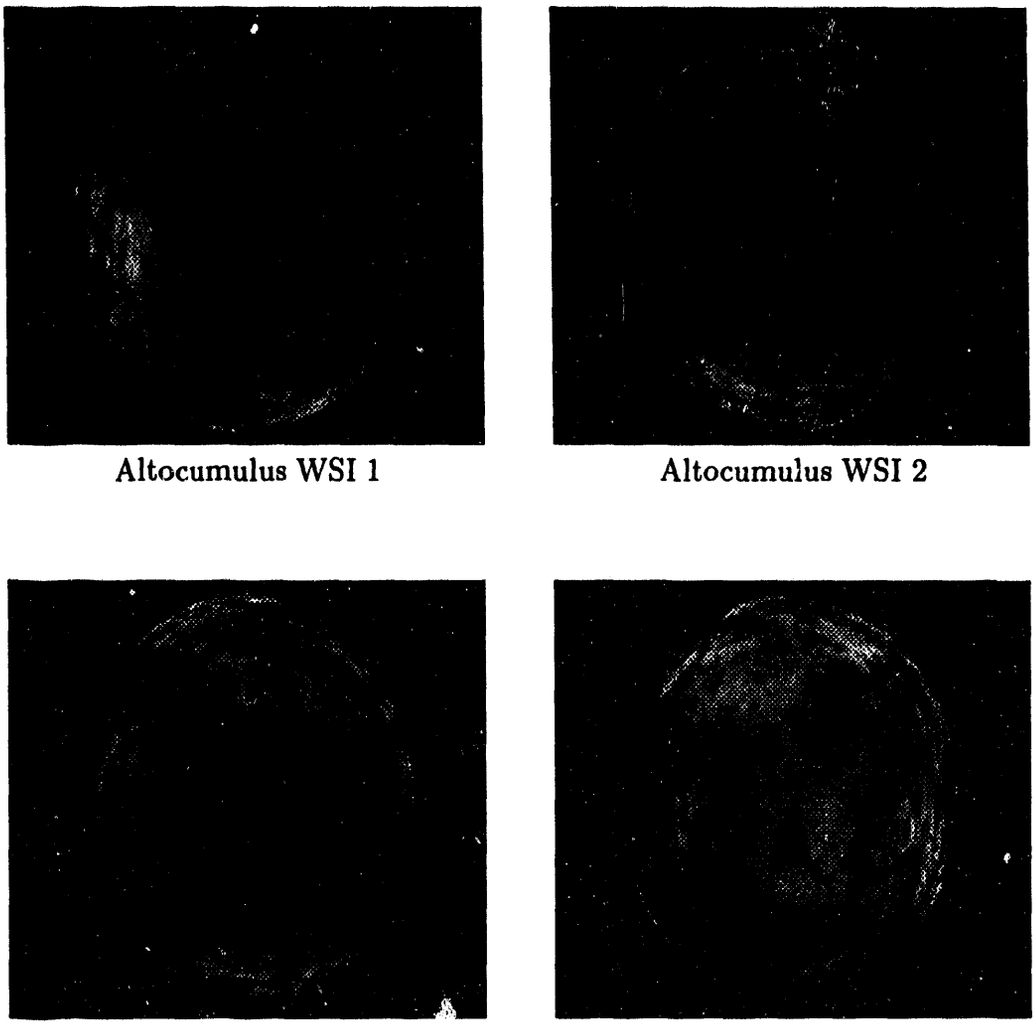
Figure 5.3 shows two examples of images produced from a synthetic cloud field. In each case two images are shown, one from each WSI camera viewing the cloud field. Figure 5.4 shows a single image of six other synthetic cloud fields.

Although the model and projection are much simpler than the real cloud projection in the atmosphere, the data set can used as an optimized case for the cloud base height calculation. Also, this allows calculation and, most importantly, *verification* of cloud base heights on realistic data. Moreover, the simulation can produce a wider range of cloud types than exist in our real imagery.

5.2 Whole-Sky Imager Data

The WSI camera which generated our existing data is the Whole Sky Imager developed by the Marine Physical Laboratory (MPL) at the Scripps Institution of Oceanography, as described Shields, Koehler and Karr [15]. Two of these cameras were used to collect data in a manner that is similar to their expected future use. This section will described the images in greater detail and describe the geometry of the site where the data was collected.

Two types of WSI images have been used in our experiments, Field and Cloud Decision. Field images are produced directly by the WSI while Cloud Decision images are the result of post, offsite processing of the Field images. The goal of the post processing is to label each pixel as either "sky", "opaque cloud", or "thin cloud". Having each pixel categorized as "cloud" or "not cloud" has obvious advantages when computing cloud base heights since sky points do not have to be considered, either when picking the points which have height



Altocumulus WSI 1 Altocumulus WSI 2

Altostratus WSI 1 Altostratus WSI 2

Figure 5.3: One frame of a synthetic altocumulus (top) and synthetic altostratus cloud scene as viewed from two WSI cameras. WSI 2 was south of WSI 1 so correspoinding pixels in the right image are shifted up relative to the left image.

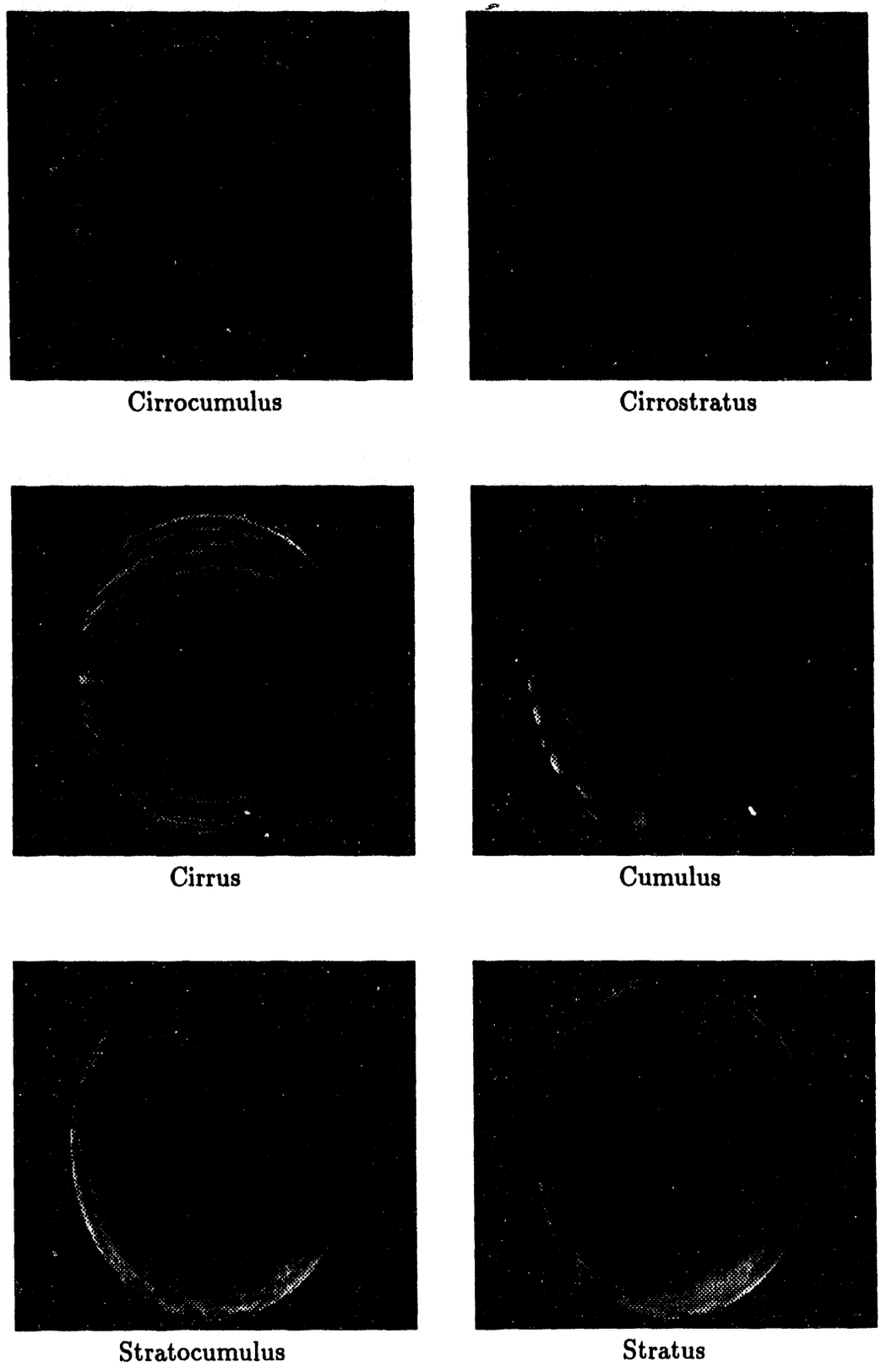


Figure 5.4: One frame of various synthetic cloud scenes.

computed or when finding the matching point in a corresponding image. However, the process of generating a Cloud Decision image requires human intervention and therefore can be prohibitive. However, to investigate whether the cost of human intervention is warranted, results were determined for both Field and Cloud Decision images. Below the process of generating Cloud Decision images will be described. This discussion is based upon the work of Johnson, Hering and Shields [7].

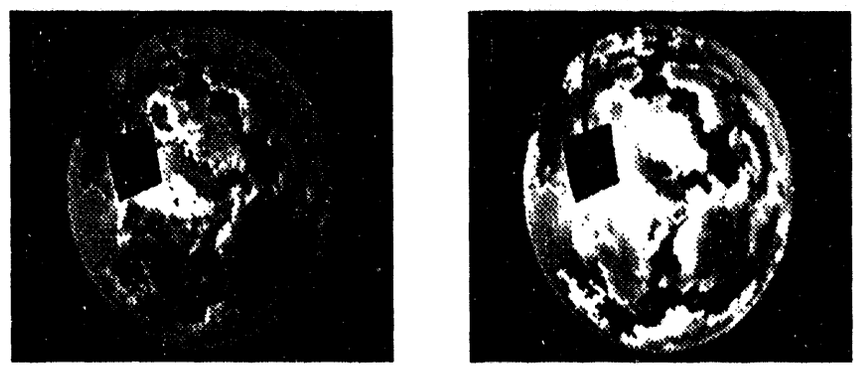
There are numerous methods that could be used to segment clouds from sky in WSI images. Two approaches include radiance thresholding and edge detection. However, problems with radiance thresholding include: 1) sky and radiances vary over large excursions; 2) threshold radiance requires continual analytic updates; 3) dark or shadowed clouds mimic background sky radiances. Problems with edge detection include: 1) clouds embedded in cloud background yield redundant boundaries; 2) fails as the scene approaches uniform overcast. However, cloud discrimination algorithms based on the ratio of blue and red radiance fields are effective since the blue/red spectral radiance ratios are near one for clouds and significantly higher for sky.

Toward this ratio approach, each minute the camera produces two image pairs for a total of four Field images in all. The first pair consists of an image taken with a red filter and one taken with a blue filter. The second pair is also a red-blue pair, but with a neutral density filter used as well. By "darkening" the images, the neutral density filter helps insure that the full range of sky/cloud brightness in the scene is accommodated. Each red-blue pair is then ratioed, threshold cutoff values for clouds determined, and the two ratio pairs combined to form the final Cloud Decision image. The threshold values are determined manually. Figure 5.2 shows an example of the four Field images and the resulting Cloud Decision image.

There are two problems with this Cloud Decision approach. First, the four Field images are not taken simultaneously, rather there are a few seconds between them. Depending upon the motion of the clouds, the difference between the four images may be appreciable. Second, the filters do not have the same optical properties, requiring that the red/blue pairs must first be geometrically corrected so that corresponding points appear at the same location in the images. Both of these problems result in ratio images that appear blurred.

While ratio images can be used for cloud discrimination, the results from using them for cloud base height calculation are expected to be adversely effected by the blurring. But Field images can be troublesome because the radiance can be so high that the image becomes saturated, removing texture. Section 6 will examine the accuracy obtained using only Field images, only Cloud Decision images, and Field images with Cloud Decision images as masks, indicating cloud and sky points.

Our real WSI data was taken in May, 1992, in White Sands, NM. In an attempt to simulate the eventual CART data, we separated two WSIs by 5.54 km, with a ceilometer located close to the midpoint between them. The intent of the ceilometer was to provide fiduciary points with which to check our algorithm. It was fired once a minute, in time with both WSIs. As a result, when the ceilometer reports the presence of clouds, simple geometry and knowledge of the camera location suffices to compute which pixel on the WSI images corresponds to that ceilometer report. The cloud base height computed at that point can then be compared to the ceilometer measurement. Figure 5.6 shows the geometry between the sites and the ceilometer.

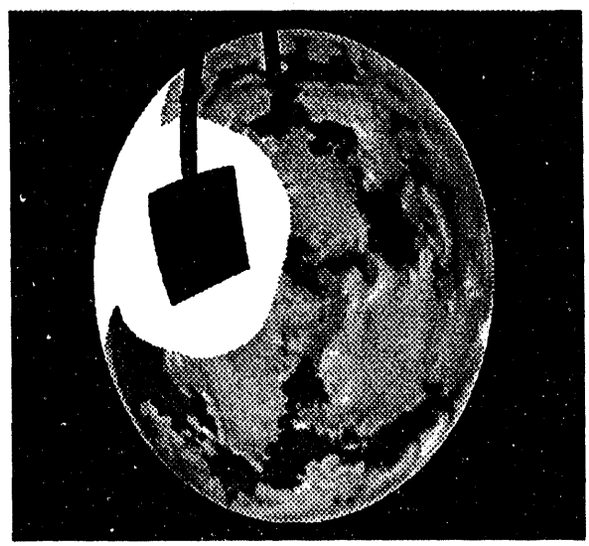


Blue

Red

Blue with ND filter

Red with ND filter



Cloud Decision

Figure 5.5: Four Field images and the resulting Cloud Decision image.

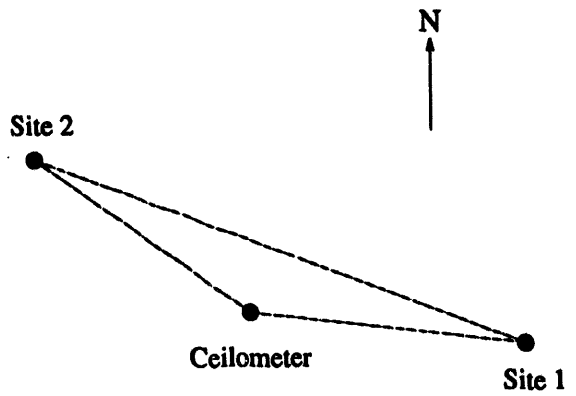


Figure 5.6: Relative geometry of the cameras and ceilometer at White Sands. There is a WSI camera at Site 1 and Site 2. The distance and bearing from Site 2 to Site 1 is 5.54 km and 125°, respectively. The distance and bearing from Site 2 to the ceilometer is 2.16 km and 133°, respectively.

Chapter 6

Performance Results

In this section results will be presented for both synthetic and real imagery. A number of issues are investigated, including:

- Should Field or Cloud Decision images be used?
- If Field images are used for finding corresponding points, does it help to use Cloud Decision images to identify cloud versus sky?
- What size correlation window should be used?
- How far off the epi-polar line should you look for corresponding points?
- How much does using the optical flow field help?

The geometry in the synthetic case is known exactly so it does not make sense to look off the epi-polar line for matches. The issue of Field verses Cloud Decision images does not apply to the synthetic data. But all the other issues are relevant to synthetic and real images. After showing results for the synthetic cloud data, each of these issues will be addressed.

In all cases, only points where the computed height was confident are considered when presenting results. Points with confidence values below 2.1 are considered confident. If the confidence threshold is set to a lower value (so only more confident points are considered), the number of confident points decreases, but the distribution of errors remains approximately the same.

For all histograms, the standard deviation of the data shown in the histogram is presented. However, the usefulness of standard deviation is questionable. If large errors are no worse than small errors, then the standard deviation is not a good indicator of performance since large errors have more impact on the standard deviation than do small errors. However, if large errors are in fact worse than small errors, then the standard deviation is an appropriate measure of performance. Standard deviation is presented for completeness.

Figure 6.1 shows the points where cloud height was computed correctly and incorrectly for four different synthetic images. White indicates that the computed height was too high. Black indicates too low. Dark gray indicates there was no cloud or that the computed height was not confident. Light gray indicates that the height was computed within 5% of the true height.

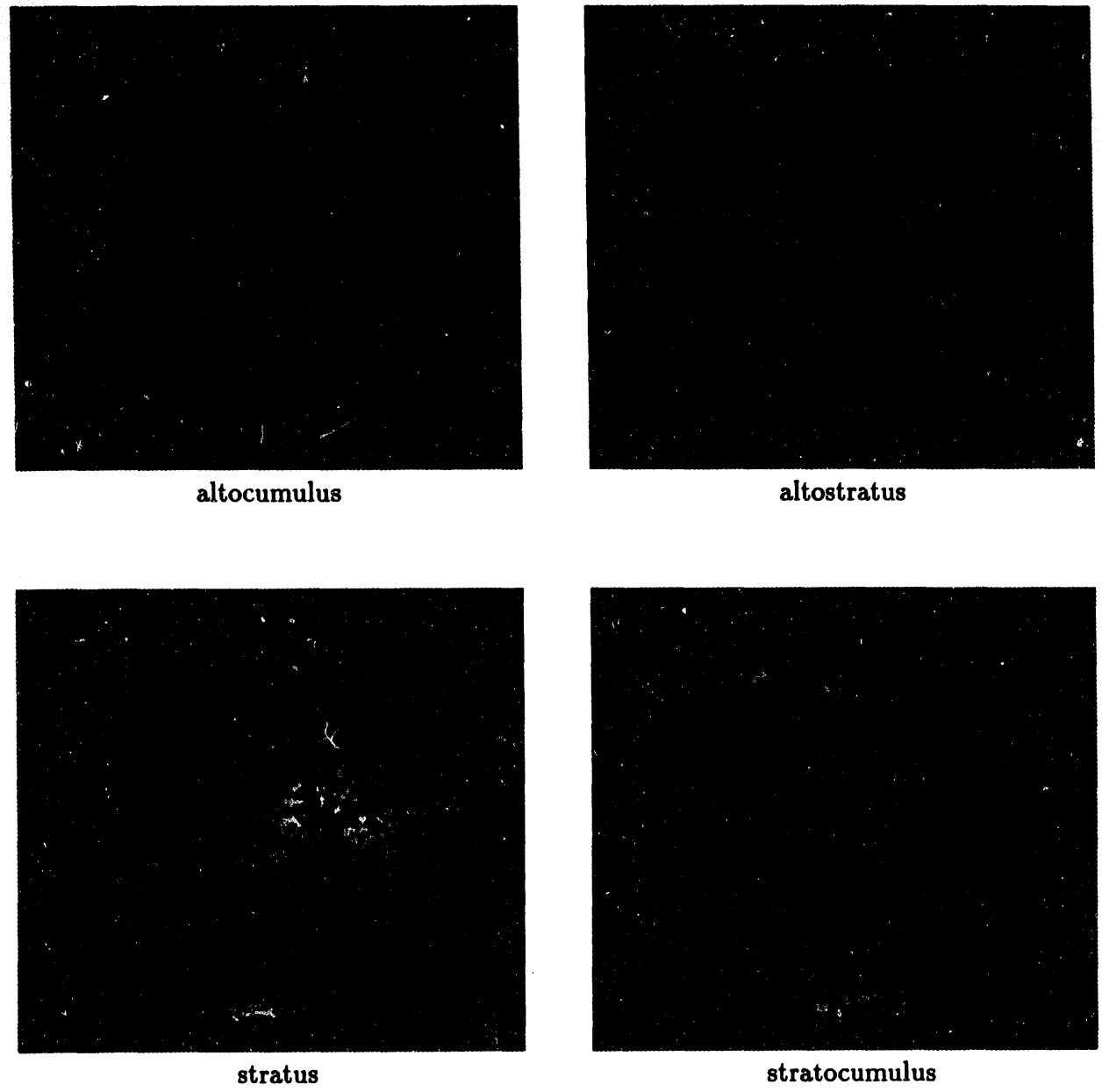


Figure 6.1: Accuracy of four synthetic cases. White indicates that the computed height was too high. Black indicates too low. Dark gray indicates there was no cloud or the that the computed height was not confident. Light gray indicates that the height was computed within 5% of the true height.

The error histograms of these four cases and four other ones are shown in Figure 6.2. For each confidently computed height, it is subtracted from the true height (known because this is synthetic imagery). The resulting differences are shown in the normalized histogram. As expected, the peaks are centered at zero.

In order to test the accuracy of our computed heights with real images a ceilometer was used to measure the cloud height for a point on a cloud. Cloud heights were computed for all the points in a 5×5 neighborhood around the point where the ceilometer was pointing. This is done for all the ceilometer hits in one day and results for the entire day are shown together. Unfortunately, the range of the ceilometer was only 3800 meters, so clouds above 3800 meters were not detectable by the ceilometer. Since only points around ceilometer hits are processed for these results, all clouds in these experiments were below 3800 meters. The lower the cloud is in the sky, the greater its shape varies between two WSI cameras. Therefore, these results show the performance of the algorithm on the hardest of cases, i.e., low clouds. It is very reasonable to expect that results will improve if clouds at all heights are considered.

The first issue when working with real images is whether Field images give better or worse results than Cloud Decision images. Figure 6.3 compares the results using Cloud Decision images and Field images for two separate days. For May 4 it is clear that Field images give better results. For May 5 Cloud Decision images give slightly better results.

The advantage of Field images is that they do not require extra computation and human intervention whereas the Cloud Decision images do. However, the quality control of the Field images is not as good as with Cloud Decision images. The images for one of the cameras on May 5 were washed out, making matching points between images difficult. Even though the Cloud Decision images have segmented the clouds from the sky, the texture in the clouds has been diminished in the process. Since there is usually more texture available when finding corresponding points and Field images are easier to acquire, they are the type of image of choice.

Even though Cloud Decision images give worse results than Field images, they may still be useful since they distinguish sky from cloud. While using Field images, the Cloud Decision images can be used as masking images, indicating points that are sky and can therefore not be a corresponding point of a cloud point. The results of using Field images with Cloud Decision images as a mask verses no mask are shown in Figure 6.4. It is clear that for May 4, masking hurts performance. This results because the set of points that went into the Masking histogram differs from the set of points in the No Masking histogram.

That is, for a given day, the set of points that have a height computed can differ between Cloud Decision and Field images, or between Field images with masking and Field image without masking. For each ceilometer hit, all points in a 5×5 neighborhood around the hit are considered as possible cloud points. If it is a cloud point, then it's height is computed. For Cloud Decision images it is straightforward to determined if the point is labeled as cloud, but for Field images it is not as easy. Currently simple thresholding is used. This results in some cloud points being eliminated and some sky points considered as cloud. Errors in labeling clouds are made in both the Cloud Decision and Field images, and it is because of these errors that differences result in the set of points that go into the histograms. When presenting histograms, we could only use the points common to all the sets of data, but we are more concerned with the percentage of correct answers given the entire set of data for

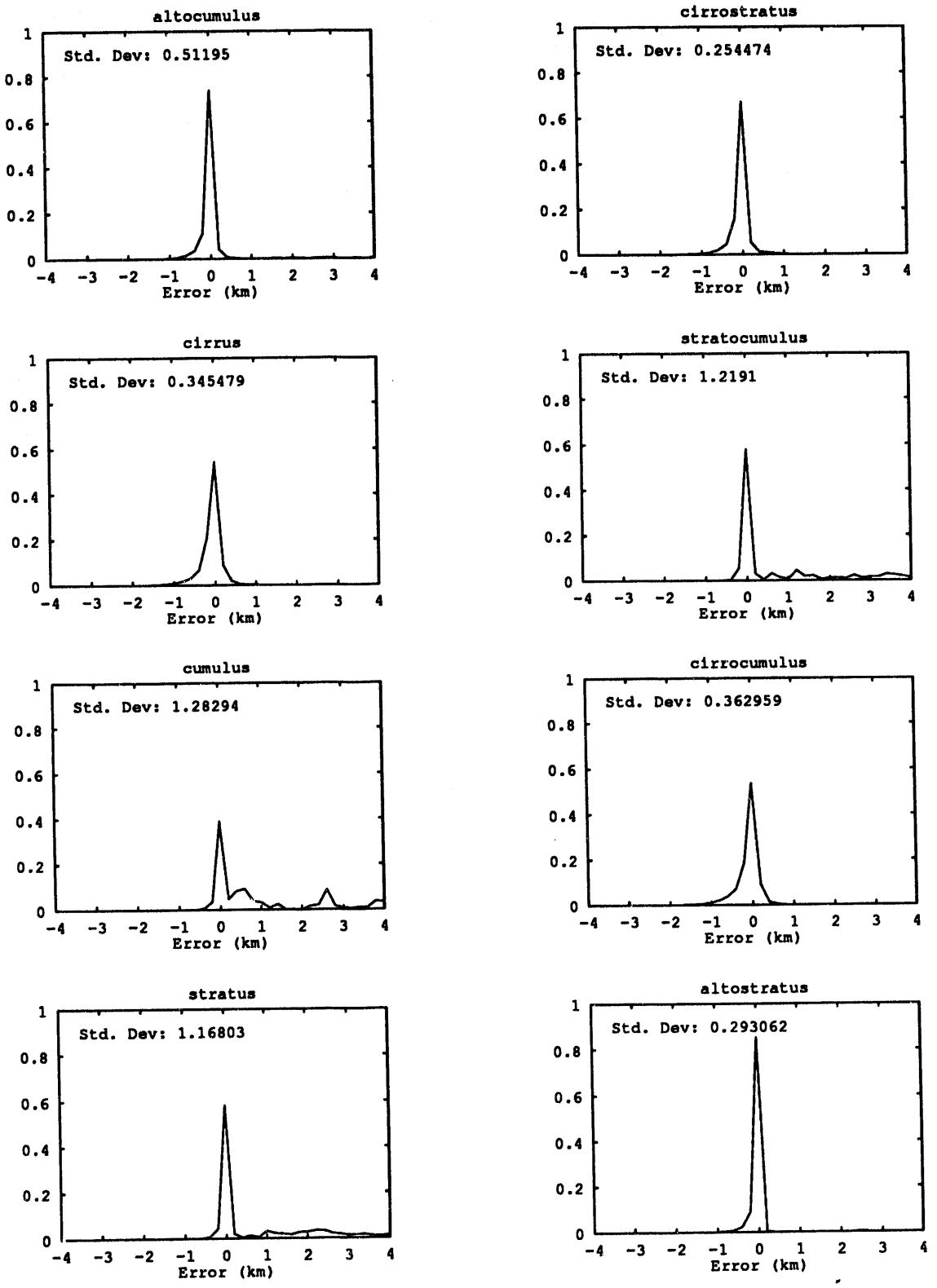


Figure 6.2: Normalized error histograms for eight different synthetic cases.

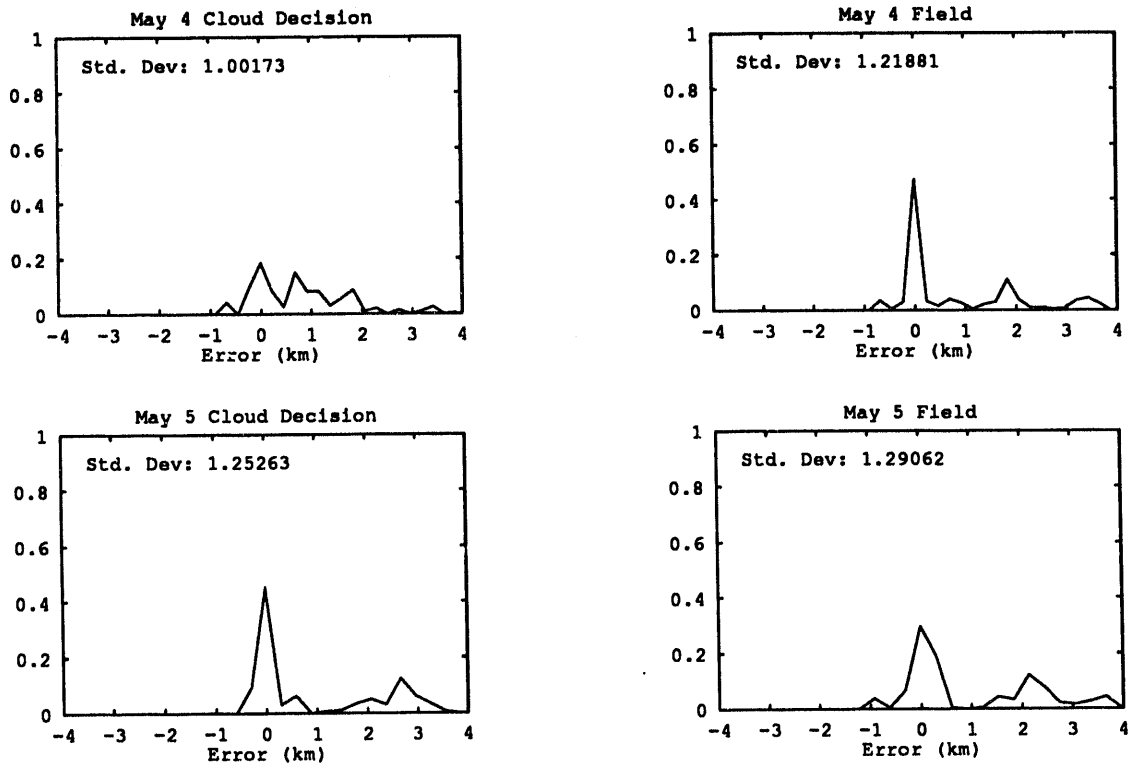


Figure 6.3: Normalized error histograms using Cloud Decision images (left) and Field images (right). Each row shows corresponding histograms.

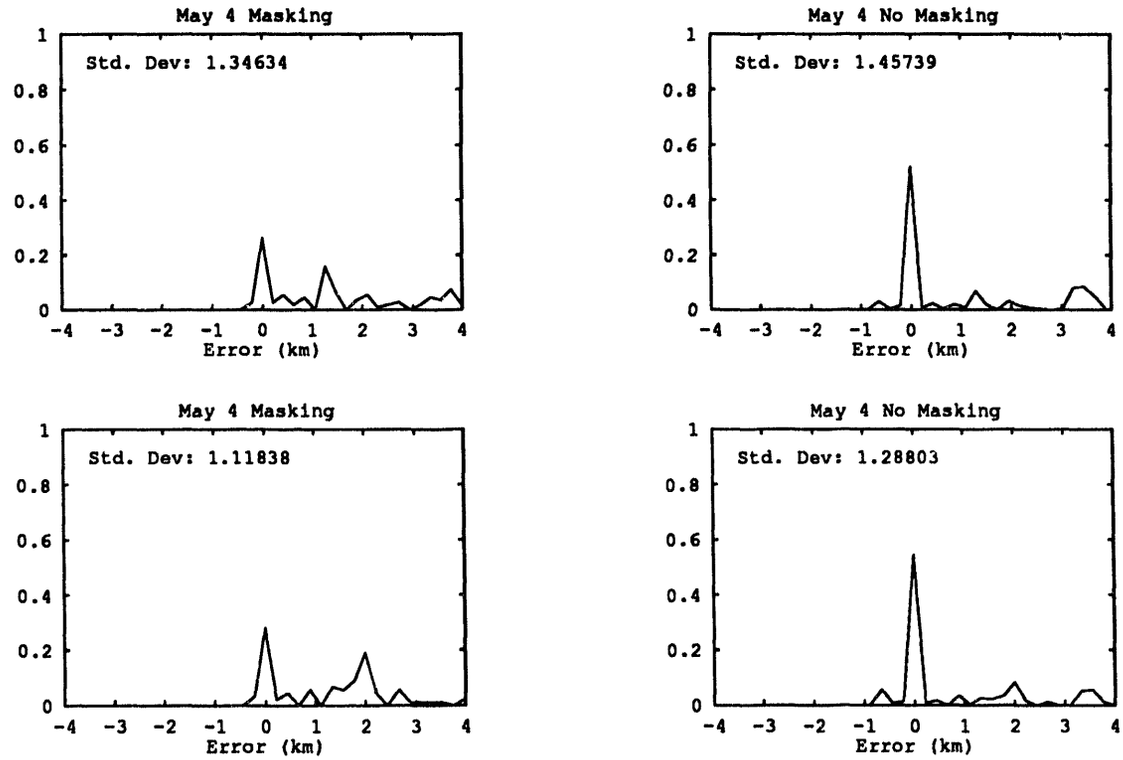


Figure 6.4: Normalized error histograms using Field images with Cloud Decision images as masks (left) and Field images without a mask (right). Each row shows corresponding histograms. The May 4 data was used in every run shown here, but the parameters of the runs were different for each row. Specifically, flow was used for the runs shown in the first row but not used for the runs shown in the second row.

the day.

The size of the neighborhood used to do the correlation when trying to find matching points can vary anywhere from one to the size of the image. Smaller neighborhood sizes require less computation, but often do not perform as well as larger sizes. In order to determine the optimal value, results are computed for various correlation sizes. This is shown in Figure 6.5. After a size of about 19×19 little is gained by using a larger window.

Since there is uncertainty in the relative positioning of the WSI cameras, it is possible that the matching point is not on the epi-polar line. Therefore it may be beneficial to look some small distance away from the epi-polar line for matching points. Figure 6.6 shows the results when additional points near the epi-polar line are considered as possible matching points. Clearly, nothing is gained by examining points off the epi-polar line. So, if the correct matching point is in fact off the line, the point on the epi-polar line that best matches must be very close to the true matching point. Since wider epi-polar lines require more computation, an epi-polar line width of 1 is used.

Finally, the importance of using optical flow was tested. Figure 6.7 shows results with and without using flow to help find corresponding points. The benefits of using flow, while there, are not as significant as one would initially expect. This tells us that, while the human visual systems needs the additional help of flow to find corresponding points, the computer gets most of the matching information from the texture of the clouds and not from the variations in the flow field.

In this chapter we have shown the following:

- Field images should be used rather than Cloud Decision images
- Using Cloud Decision images as masks results in fewer cloud points being considered and resulted in worse performance.
- A 19×19 correlation window should be used
- Looking for matching points off the epi-polar lines does not increase performance
- This usefulness of optical flow for finding corresponding points is minimal.

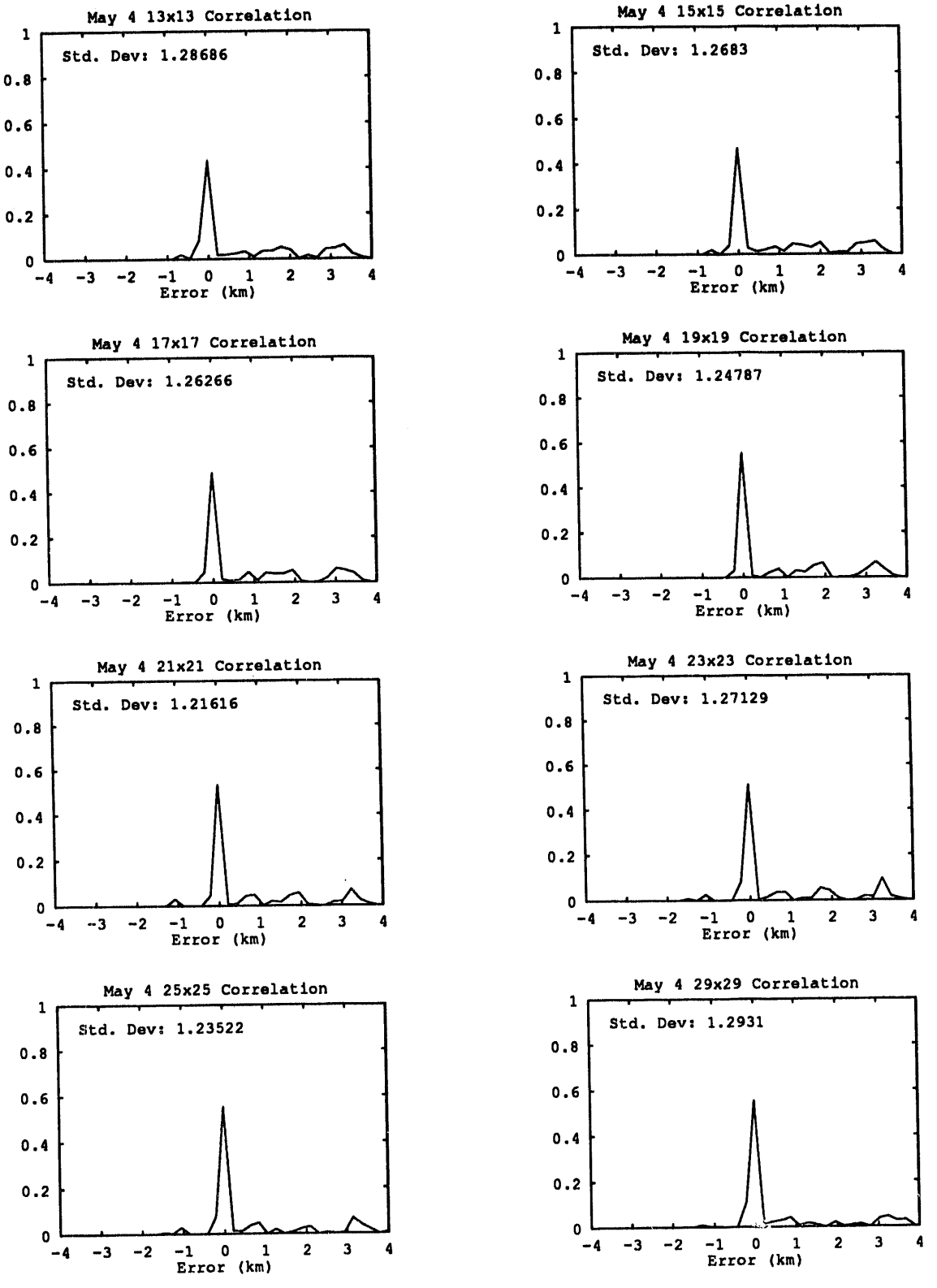


Figure 6.5: Normalized error histograms for different correlation neighborhood sizes.

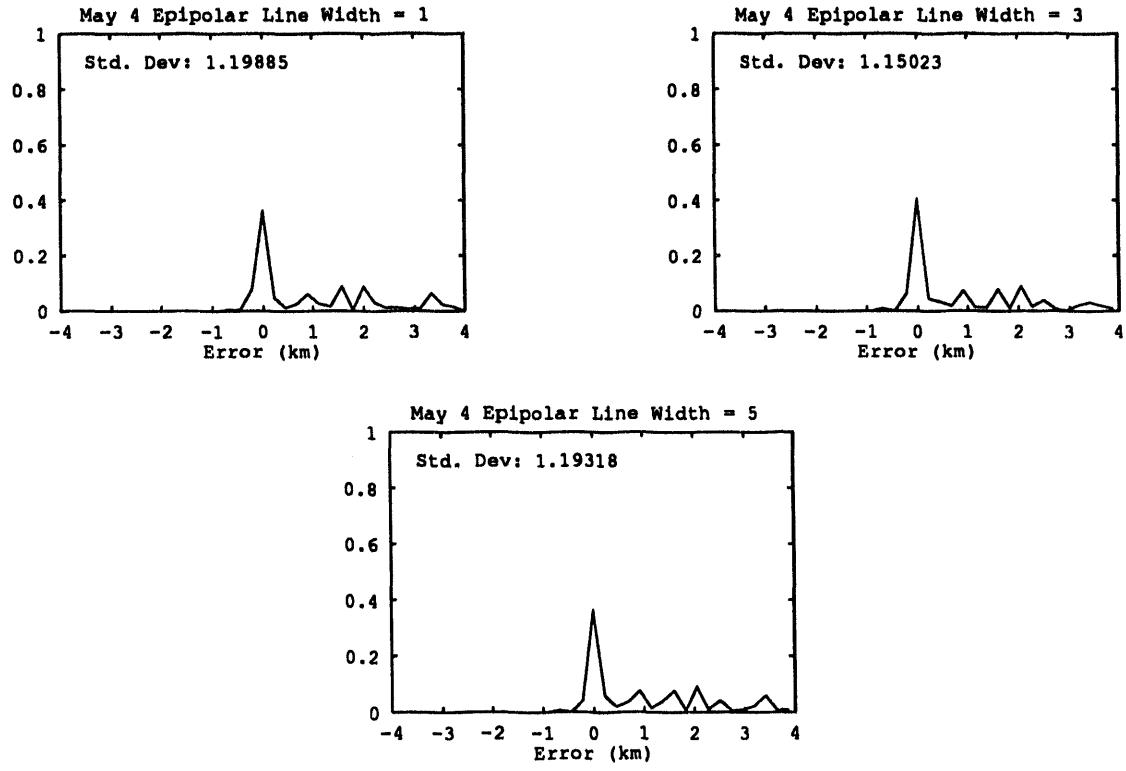


Figure 6.6: Normalized error histograms for different epi-polar line widths.

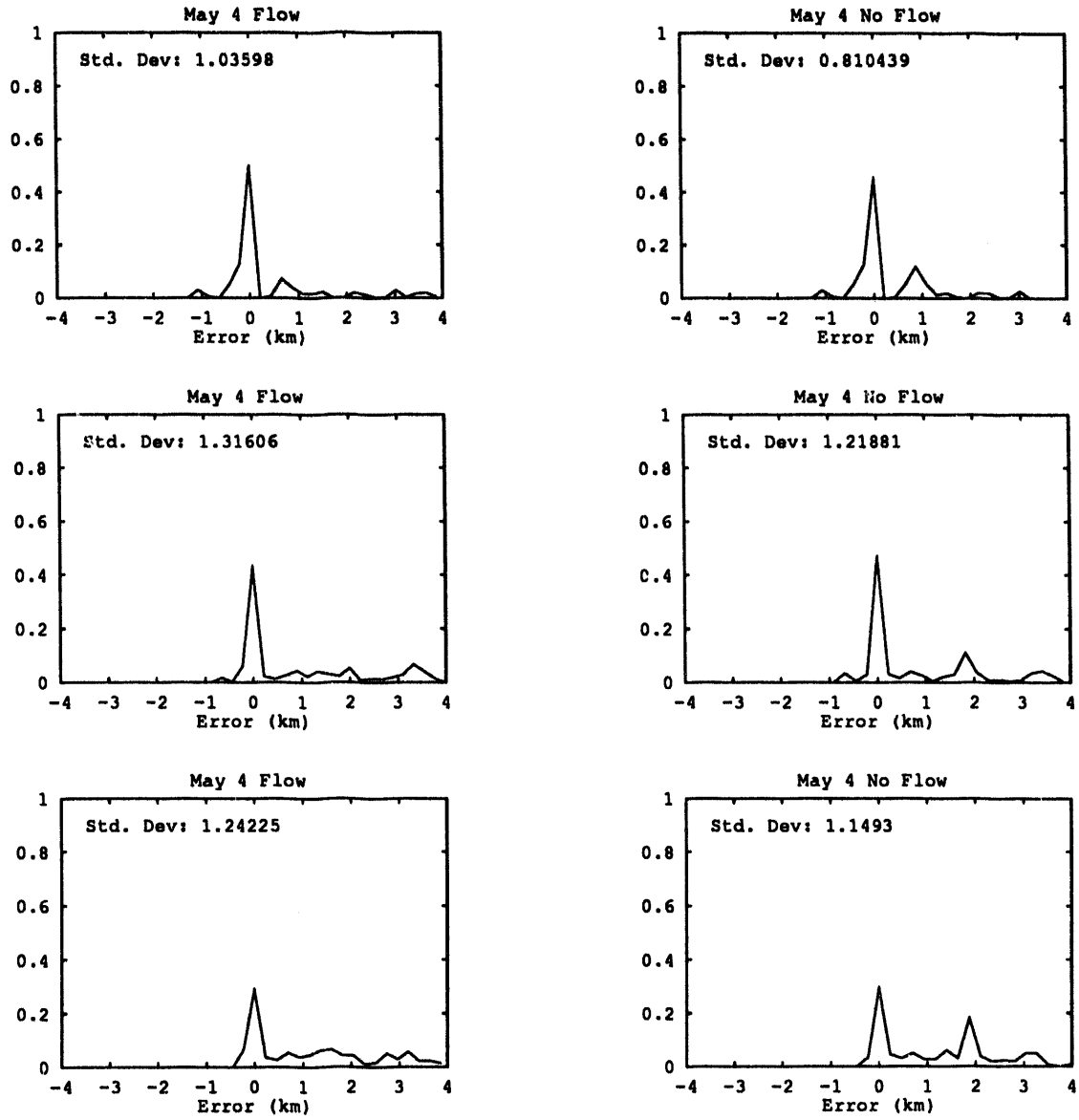


Figure 6.7: Normalized error histograms using optical flow (left) and not using optical flow (right). Each row shows corresponding histograms. The May 4 data was used in every run shown here, but the parameters of the runs were different for each row. Specifically, the first row used no masking and a correlation window size of 29×29. The second row used no masking and a correlation window size of 9×9. The third row used masking and a correlation window size of 9×9.

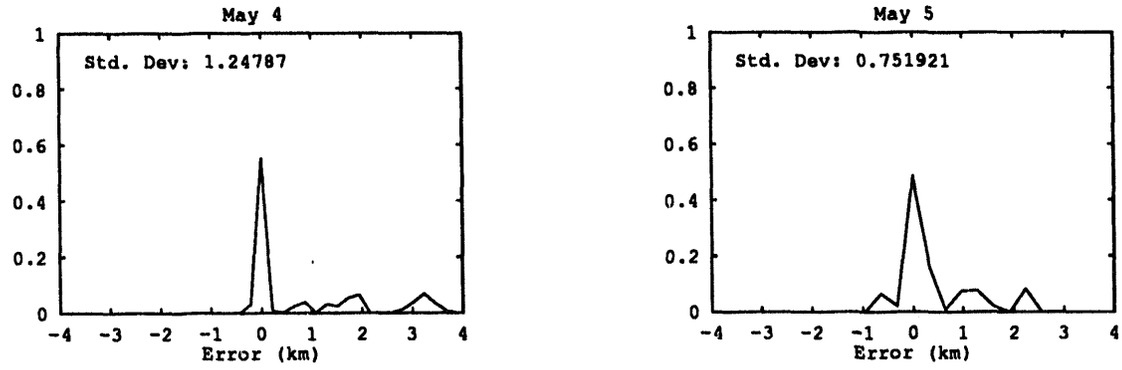


Figure 6.8: Normalized error histograms using the best setting of all the parameters.

Chapter 7

Concluding Remarks

We have demonstrated how paired data from widely separated whole-sky imagers can be fused to extract cloud base heights, an important cloud property and one which could not be recovered from either imager alone. An important feature of our approach, one that will help it to generalize to the incorporation of other data sources, is its ability to measure its own confidence in the determined base heights.

A motivating factor for this research was to exploit the ability of the WSI imager to provide temporally dense sequences of images. This would allow us to use cloud dynamics to help find corresponding points between images. However, the usefulness of using optical flow was not as great as was expected. But this only says there is sufficient texture in the images to find corresponding points, and that the additional information of optical flow is not strictly necessary. It was also concluded that, given the expense of obtaining Cloud Decision images and their diminished texture, Field images should be used. We also found that looking off the epi-polar line does not improve performance and that a 19×19 correlation window is a reasonable choice.

We presented results on both synthetic data and particularly challenging real data. Recall that due to the limitations of the ceilometer, only the hardest case clouds were used in the results. With this data, cloud base heights are computed to within 5% of the correct height approximately 50% of the time.

Appendix A

Open Issues and Future Work

In this section open issues will be briefly reviewed and future work discussed. This is not intented to be an exhaustive description, rather a terse summary of the issues. For further information the authors should be contacted. Unresolved issues with derivations will be discussed first, followed by unresolved issued with the interpretation of results. Finally future research directions will be discussed.

A.1 Future Work

Future efforts would be most usefully devoted to improving the confidence metric. The confidence metric described in this work is a measure of how likely it is that a cloud point is visible in both cameras. Other metrics can be sensitive to how large and/or unique the best correlation value is along the epi-polar line. If the best match is weak or there are other points with correlation values of similar strength, then the confidence that the correct matching point has been found should be low. This could be formulated using a Bayesain model where corresponding points along the epi-polar line are found using the maximum *a posteriori* (MAP) probability criterion. In this case, the *a posterior* probability can be obtained for each possible match along the epi-polar line. A low MAP probability indicates that the posterior probability distribution has a broad, low peak (as would occur for sky points), or that the there are many strong matches along the epi-polar line. With this approach, the confidence would be low in these cases, as desired.

Other future and concurrent work investigates the extraction of other properties of interest (particularly fractional cloud cover and aspect ratio) from paired WSI images, and fusing of WSI images with satellite imagery in order to determine cloud top structure as well.

A.2 Derivation Issues

In Section 4.4 an assumption was required in order to find the parameters of PCT which minimized the amount of compression and/or expansion of neighborhood induced by the PCT. There was a circular dependency that prevented knowing the size of the neighborhood in WSI coordinates that would map to a specific size neighborhood in PCT coordinates. It

was assumed that the parameters that minimized the amount of expansion and/or compression was independent of the size of the neighborhood in WSI coordinates. For example, the optimal parameters for a 5×5 neighborhood would be the same for a 9×9 neighborhood. This assumption may not be valid. The assumption could be avoided if the size of the neighborhood were not part of the final optimization equation.

Below we present some preliminary thoughts on an improvement to the equation to be optimize. These improvements make assumptions unnecessary in some cases and possibly true otherwise.

Currently, the expansion and compression of a neighborhood is measured by subtracting the size of the neighborhood before and after PCT. But consider a 1×1 neighborhood in WSI coordinates that becomes a 5×5 neighborhood after PCT. In this case the neighborhood expanded by a factor of 5 in both dimensions and the PCT parameter formulation will consider the expansion in both dimension as $5-1 = 4$. Now consider a 5×5 neighborhood in WSI coordinates that becomes a 25×25 neighborhood after PCT. Again, the neighborhood expanded by a factor of 5 in both dimensions, but the PCT parameter formulation will consider the expansion in both dimension as $25 - 5 = 20$. This is clearly not correct since in both cases each pixel was copied five times. This can be corrected by considering the ratio of the of the sizes of the neighborhoods rather than the difference. With this approach, the expansion would be seen as five for both cases.

If this new approach is used, we want the the ratio of the sizes to be 1.0. By taking the ratio, subtracting 1 and squaring the result (to eliminate possible negative values), we have a formulation which can be minimized to give the optimal PCT parameters. In this formulation, the angular size of the neighborhood drops out, so our assumption is valid for the angular dimension of the neighborhood. However, the radial size of the neighborhood does not cancel. But it may be the case that the assumption is valid in this case.

A.3 Results Interpretation Issues

In the results chapter an explanation was given as to why the results using Field images with Cloud Decision images as masks can give worse error histograms. The reason was that different sets of cloud points went into each histogram. But even if the same cloud points go into corresponding histograms, masking does not improve results. One would expect that masking would help at least slightly since the clouds are segmented from the sky; so you cannot accidently match a cloud point to a sky point, as you can when using Field images. Why masking does not help more, or why cloud points are never matched to sky points when using Field images without masks is worth further investigation.

For all the results presented, the confidence threshold was set at 2.1. So only the cloud base heights with a confidence value less than 2.1 are shown in the error histograms. If the confidence is set lower or slightly higher, the error histograms do not change significantly. The number of confidently computed cloud base heights decreases as the confidence threshold decreases, but the mix of height errors remains the same. An explanation for why this is true is worth further investigation.

Appendix B

A Guide to the Software

There are two software packages needed for computing cloud base heights, `cloud_of` and `cbh`. `cloud_of` computes the optical flow and `cbh` computes the cloud base heights. Each package will be described below. Then a third package for doing the pseudo-Cartesian transformation (PCT) of a WSI image is described. The code for doing PCT is incorporated into `cbh`. The package described below is a stand-alone version, supplied for possible interest. Finally, parameters within the code that may need to be changed depending upon the cameras used and the geometry of the site are discussed. In all cases, images must be in VIFF format [11].

B.1 `cloud_of`

`cloud_of` computes the optical flow for an image sequence using Burt's hierarchical pyramid algorithm. While this program exists in support of `cbh`, and has its default parameters set accordingly, it is actually a general purpose optical flow algorithm and can be used for other purposes. This is why there is such flexibility in the command line arguments. For use with `cbh` the only parameters that need to be specified are "-i" and "-o". For all the other parameters, appropriate default values will be used if no other values are provided on the command line. The command-line parameters are as follows. Optional parameters are indicated with square brackets; the default values shown in parentheses.

-i	Image Base Name
-o	Output flow
[-m]	Masking Image 1 (null)
[-f]	First Frame (0)
[-t]	Image Type (0-Non Cloud; 1-Field; 2-Cloud Decision) (2)
[-n]	Number of Frames of Flow (1)
[-s]	Compute flow every sth pixels (1)
[-vt]	Vector Type (0-3D Spatiotemporal; 1-2D Displacement) (1)
[-p]	Pyramid Generation Size (3)
[-l]	Maximum Levels in the pyramids (100)
[-c]	Correlation Size (5)

Input Image Names (-i -f): The base name of the input images. The images must be in the files *base_namexxx*.xv, *base_namexxx+1*.xv, etc., where *xxx* is the three-digit frame

number specified by the "-f" parameter. Note that if the frame number is less than 100, the number in the file names must be padded with zeros.

Output Image Name (-o): The "-o" specifies the file that the flow is to be written to.

Pyramid Generation (-p): Two versions of the algorithm are available, "-p 3" and "-p 5". The -p argument specifies the size of the neighborhood used around a point to generate the next level of the pyramid.

Correlation Size (-c): The "-c" parameter specifies the size of the neighborhood around a point used when correlating that neighborhood against the next image in the sequence.

Sampling (-s): If flow is not needed at every point then the "-s" argument can be used so that flow is computed every sth pixel. Note that if the flow field is going to be used with cbh, the "-s" argument must be 1. Flow vectors are not computed for pixels that are skipped over and are not in the outputted flow file.

Number of Frames of Flow (-n): The "-n" specifies the number of frames of flow that is to be generated. All the flow files will be put in the file specified by "-o". Note that if the flow field is going to be used with cbh, the "-n" argument must be 1.

Image Type (-t): The "image type", -t, argument specifies the type of image being passed in and hence where flow should be computed. 0 indicates any type of image and that flow should be computed at every pixel. 1 indicates that flow should be computed at every nonzero pixel. This is what should be specified for a Field WSI image. 2 indicates that this is a Cloud Decision image and that flow should only be computed on clouds.

Masking (-m): If flow is being computed for Field images, then a Cloud Decision image can be used as a mask. A masking image is specified with the "-m" parameter. It will be assumed that the masking image is a cloud decision image and flow will only be computed for cloud points. Note that the mask name is the complete file name, not the base name. So it will also be assumed that only one frame of flow is desired since only one mask can be specified. If a masking image is used the image type should be 2, i.e., "-t 2" should be specified. Points in the halo around the occluder will not be considered even though those points are visible in the field image.

Vector Type (-vt): The "vector type", -vt, argument specifies the type of flow field to generate. 0 indicates that a spatiotemporal flow field (a three component vector field, dx, dy and dt, representing the instantaneous motion at a point) should be written. 1 indicates that a displacement flow field (a two component vector field, dx and dy, representing the displacement of a point from one image to the next) should be written. Note that if the flow field is going to be used with cbh, the -vt argument must be 1.

Number of Levels in the Pyramid (-l): A standard, non hierarchical algorithm is achieved by setting the maximum levels of the pyramid to 1 ("-l 1"). If the hierarchical algorithm is desired simply set this parameter to some large number, 100 for example.

Flow Field File Format: Below is a listing of the contents of the resulting flow fields. If more than one flow field is requested, i.e. "-n" greater than 1, all the flow fields are put in one file with one header at the start of the file. The header is the same for files containing flow fields of spatiotemporal vectors or displacement vectors.

Type	Size	Description
int	4 bytes	Number of rows.
int	4 bytes	Number of columns.
int	4 bytes	Number of frames of flow in the file.
int	4 bytes	Sampling, flow computed every sampling-th pixel in each dimension.
int	4 bytes	Offset, the number of pixels in from the edge of the image that the flow field starts.
int	4 bytes	First frame number. File number of the first frame which actually had flow computed. Not used by `cbh`.
float or char	variable	Flow field vectors

If the sampling is greater than 1, only the pixels that had flow computed are stored.

For spatiotemporal flow fields, for each pixel which had flow computed, three vector components, dx, dy, and dt, are stored in that order as floats (4 bytes). The vector field is stored in row-major order. Every vector is a unit vector.

For displacement flow fields, for each pixel which had flow computed, two vector components, dx and dy are stored in that order as characters. The vector field is stored in row-major order.

Examples:

`cloud_of -i cirrus -f 0 -o flow1 -m cirrus000.cd.xv -t 2 -vt 0`
will compute the optical flow for the image sequence with base name `cirrus`. The file names of the images will be in `cirrus000.xv` and `cirrus0001.xv`. The cloud decision image `cirrus000.cd.xv` is a mask. Since a mask image is specified, the image type is specified as 2. The flow field will be written to the file `flow1`. Since this flow is to be used with cbh, the vector type is set to 0.

`cloud_of -i cirrus -f 0 -o flow2 -n 10 -p 5 -l 100 -w 15 -vt 1`
will generate 10 displacement vector flow fields and put it in the file flow2. A 5 by 5 pyramid generation scheme will be used. Since the number of maximum levels is set to a large number the entire pyramid will be generated. A neighborhood size of 15 by 15 will be used when doing correlation.

B.2 cbh

`cbh` computes the cloud heights for points visible in Cloud Image 1. This is done by knowing the geometry between two WSI cameras and finding corresponding points between Cloud Image 1 and Cloud Image 2. In order to help find the corresponding points, two optical flow fields, one for Cloud Image 1 and one for Cloud Image 2, are also passed in. The

only parameters that need to be specified are "-cloud1", "-cloud2", "-flow1", "-flow2" and, possibly, "-mask1" and "-mask2". For all the other parameters, appropriate default values will be used. This is assuming that the relative geometry between the cameras has been set in the file `cbh.pane`. If not, then "-dist" and "-dir" must also be specified. The command-line parameters are as follows. Optional parameters are indicated with square brackets; the default values shown in parentheses.

-cloud1	Cloud Image 1
-cloud2	Cloud Image 2
-flow1	Flow Field 1
-flow2	Flow Field 2
-o	Base Name for computed height and confidence images
[-mask1]	Masking Image 1 (null)
[-mask2]	Masking Image 2 (null)
[-cp]	Check Point File Base Name (null)
[-dir]	Direction of 2nd WSI with respect to 1st (125)
[-dist]	Distance Between WSI Sites (5540)
[-flow_wt]	Flow weight
	(3 * pixel_wt + flow_wt = 1.0) (0.166667)
[-pixel_wt]	Pixels weight
	(3 * pixel_wt + flow_wt = 1.0) (0.5)
[-c]	Correlation Size (19)
[-e]	Distance off the epi-polar line to look for matching points (0)
[-ci]	Generate a confidence image (true)
[-s]	Compute heights every sth pixels (1)
[-x]	Left corner of a region to be processed (0)
[-y]	Upper corner of a region to be processed (0)
[-w]	Width of a region to be processed (-1)
[-h]	Height of a region to be processed (-1)
[-t]	Image Type (1-Field; 2-Cloud Decision) (1)

Input Images (-cloud1 -cloud2): A pair of WSI images is specified with the "-cloud1" and "-cloud2" parameters.

Output Images (-o): <Output Image>.xv is a float image with the computed height at each pixel. The unit of the computed heights is the same as the unit of the "-dist" parameter. If "-ci 1" is specified then <Output Image>.conf.xv will be created. <Output Image>.conf.xv is the confidence of the computed height at each pixel. A value of -1 in the confidence image indicates that no height was computed for that point.

Image Type (-t): The "image type", -t, argument specifies the type of image being passed in. 1 indicates a Field image and 2 indicates a Cloud Decision image.

Masking Images (-mask1 -mask2:) If Field images are specified with -cloud prameters, then Cloud Decision images can be use as masks. Two masking images are passed in,

one for each cloud image. The masking image is assumed to be a Cloud Decision image and is used to distinguish cloud points from sky points. If masking images are used the image type should be 2, i.e., "-t 2" should be specified. Cloud Decision images have a "halo" around the occluder where clouds are not visible. If masking images are used, points in the halo will not be considered even though those points are visible in the field image.

Confidence Image (-ci): If "-ci 1" is specified then a confidence image will be generated. For every pixel in <Output Image>.xv, its confidence will be the corresponding pixel value in <Output Image>.conf.xv. This confidence image is generated as follows. First, for every point, p, in cloud image 1, a matching point, p', is found in cloud image 2 and a height computed. Then for point p' in image 2, the best matching point, p" in cloud image 1 is found. Obviously, in the ideal case, p and p" are the same. The distance between these two points is the confidence for point p.

Sampling (-s): If heights are not needed at every point then the "-s" argument can be used so that heights are computed every sth pixel. Pixels that were skipped over will not be in the output images. The output images will be 1/s-th the size of the input images.

Camera Positioning (-dir -dist): The relative positioning of camera 2 relative to camera 1 is specified by the distance, "-dist", and the direction, "-dir", between them. The distance is in whatever units the cloud height is desired in and the direction is in degrees measured from North.

Flow and Pixel Weighting (-flow_wt -pixel_wt): The weighting for the optical flow vector components and pixel values is specified with the "-flow_wt" and "-pixel_wt" parameters, respectively. The flow weight is applied to all three components of the optical flow vectors. The sum of the weights must equal 1.0, so $3 * flow\_wt + pixel\_wt = 1.0$.

Correlation Size (-c): The "-c" parameter specifies the size of the neighborhood around a point which is used when correlating that neighborhood against the other image.

Epi-polar Line Size (-e): The number of pixels to look off the epi-polar line for matching points is specified by the "-e" parameter.

Checkpoint Files (-cp): Since a run can take a rather long time the routine periodically writes a checkpoint if a file is specified with the -cp argument. If a checkpoint file is specified and the run is being started for the first time make sure that the files <Check Point File>.cp and <Check Point File>.conf.cp do not exist. If starting the routine from a checkpoint simply give the same command that was first used to start the run. The existence of the checkpoint files will be checked. If they exist they will be read in and the run will continue from the checkpoint.

Subwindows (-x -y -w -h): If heights are desired for part of the points in Cloud Image 1, the upper left corner of the region is specified by the -x and -y parameters. The width and height of the region are specified by -w and -h, respectively.

Examples:

```
cloud_of -i wsi1_ -f 100 -o flow1
cloud_of -i wsi2_ -f 100 -o flow2
cbh -cloud1 wsi1_100.xv -cloud2 wsi2_100.xv -flow1 wsi1_100.flow
    -flow2 wsi2_100.flow -dir 90 -dist 5540 -o cbh
```

The two `cloud_of` commands will compute the optical flow fields for the two sequences with base names `wsi1` and `wsi2`. The resulting two flow fields in `flow1` and `flow2` will be used by the `cbh` command.

The `cbh` command will compute the cloud height for each point in `wsi1_100.xv` and put the computed heights in `cbh.xv` and the confidences in `cbh.conf.xv`. The direction (from north in degrees) and distance (in any units) of WSI2 from WSI1 are 90 and 5540, respectively. The computed heights are in the same units as the -dist argument.

For the following examples, the computation of the flow fields will not be shown.

```
cbh -cloud1 wsi1_100.xv -cloud2 wsi2_100.xv -flow1 wsi1_100.flow
    -flow2 wsi2_100.flow -mask1 wsi1_100.cd.xv -mask2 wsi2_100.cd.xv
    -e 2 -ci 1 -s 2 -dir 90 -dist 5540 -o cbh
```

will compute the cloud height for each point in `wsi1_100.xv` and put the computed heights in `cbh.xv` and the confidences in `cbh.conf.xv`.

The cloud decision image `wsi1_100.cd.xv` is used as a mask; so only points that are on a cloud in that image will have a height calculated. The cloud decision image `wsi2_100.cd.xv` acts as a mask image for `wsi2_100.xv`; again, only points that are on clouds will be considered as potential matching points for the cloud points in `wsi1_100.cd.xv`.

All points within two pixels of the epi-polar line will be considered as possible matching points.

The confidence image will be written to `cbh.conf.xv`.

Heights and confidences will only be computed every second pixel.

```
cbh -cloud1 wsi1_100.xv -cloud2 wsi2_100.xv -flow1 wsi1_100.flow
    -flow2 wsi2_100.flow -x 150 -y 235 -w 10 -h 10
```

will compute the cloud heights for a 10 by 10 neighborhood whose upper left had corner is located at point 150, 235 in the image.

The default direction and distance between the cameras will be used.

B.3 pct

pct takes a Whole Sky Imager image and converts it from fisheye coordinates to pseudo-Cartesian coordinates. This is done by changing the distance between each point and the center of the image so that the distance from the center varies with the tangent of the zenith angle. By doing this the projective effects of the camera are eliminated. That is, clouds near the edge of the image are expanded, counteracting the compression at large zenith angles. Further, the motion of clouds will be uniform across the image; the speed of clouds will no longer be slow near the edge and faster near the center.

It must be assumed that clouds are horizontal. In areas where this assumption is violated, the expansion will not be correct and the corrected motion will not necessarily be correct.

This algorithm was developed by Koehler and Shields [9].

Command-line parameters are:

-i	Input File
-o	Output File

Input Image (-i): A WSI image to be PCT-ed.

Output Images (-o): The PCT-ed image.

B.4 Parameters Within the Code

There are two reasons why the `cbh`, `cloud_of` or `pct` code may need changing: 1) the geometry between the cameras does not match the default values used by `cbh`; 2) the type of images produced by the WSI cameras change.

For the first case, the default values for the "-dir" and "-dist" parameters in the pane file for `cbh` need to be changed. Ghostwriter is then run and the code recompiled. This is not absolutely necessary since the default values can be overridden on the command line.

For the second case, all the parameters of the WSI images are specified by "#defines" in the file `defs.h` and all the code that relates points in the sky to where they appear in the image is in the file `wsi.c`. There is a `wsi.c` file that could require changing for `pct` and `cbh`. There are "#defines" that could require changing for `pct`, `cloud_of` and `cbh`. The #defines that should be checked with any new camera are:

#define	Description
MAXROWS	Maximum number of rows of an image
MAXCOLS	Maximum number of columns of an image
WSI_ROW_SIZE	Row size of a WSI image
WSI_COL_SIZE	Column size of a WSI image
MID_X	X location of the center pixel
MID_Y	Y location of the center pixel
INNER_WSI_AREA	Only pixels within this distance from the center of the image have a height computed
INNER_ZENITH_AREA	Only pixels withing this zenith have a height computed (effectively the same as INNER_WSI_AREA)
IMAGE_RADIUS	When PCTing an image, this is the size of the final image

If Cloud Decision images are used the following values should also be checked:

#define	Description
HALO_VALUE	Gray level of pixels in the occluder halo
SKY_VALUE	All pixels with gray levels less than this are considered sky

If only the algorithm for generated Cloud Decision images changed that it is possible that only the **#defines** will need to change. But with a new camera it is very likely that the how the camera projects points into the image will change. In this case the `wsi.c` files will have to be examined and the functions modified to match to the new camera. See the code for more information.

References

[1] ALLMEN, M. C., AND KEGELMEYER, W. P. A method for the objective evaluation of optical flow. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (1993). (submitted to).

[2] ANANDAN, P. A computational framework and an algorithm for the measurement of visual motion. *Int. J. Computer Vision 2* (1989), 283–310.

[3] BRADBURY, D. L., AND FUJITA, T. Computation of height and velocity of clouds from dual, whole-sky, time-lapse picture sequences. SMRP Research Paper 90, Department of the Geophysical Sciences, University of Chicago, March 1968.

[4] BURT, P. J. The pyramid as a structure for efficient computation. In *Multiresolution Image Processing and Analysis*, A. Rosenfeld, Ed. Springer, Berlin, 1984, pp. 7–35.

[5] CESS, R., ET AL. Interpretation of cloud-climate feedback as produced by 14 atmospheric general circulation models. *Science 245* (1989), 513–516.

[6] CIANCIOLO, M. E., AND RASMUSSEN, R. G. Cloud scene simulation modeling. In *Proceedings of the 1991 Cloud Impacts on DOD Systems Conference* (1992).

[7] JOHNSON, R., HERING, W., AND SHIELDS, J. Automated visibility & cloud cover measurements with a solid-state imaging system. Technical Note GL-TR-89-0061, Marine Physical Laboratory, SCRIPPS Institution of Oceanography, San Diego, CA, 92151-6400, March 1989.

[8] KOEHLER, T. L. Modelling the geometric properties of the WSI instrument. Tech. Rep. AV89-070t, Marine Physical Lab, Scripps Institution of Oceanography, July, 1989.

[9] KOEHLER, T. L., AND SHIELDS, J. E. Factors influencing the development of a short term CFARC prediction technique based on WSI imagery. Tech. Rep. 223, Marine Physical Lab, Scripps Institution of Oceanography, November, 1990.

[10] LYONS, R. D. Computation of height and velocity of clouds over Barbados from a whole-sky camera network. SMRP Research Report 95, Department of Geophysical Sciences, University of Chicago, August 1971.

[11] RASURE, J., AND WILLIAMS, C. An integrated visual language and software development environment. *Journal of Visual Languages and Computing 2* (1991), 217–246.

[12] ROCKS, J. K. The whole sky sensor: Phase 1 final report. DTIC AD-A179 271, February 1987.

[13] ROSSOW, W., ET AL. ISCCP cloud algorithm intercomparison. *Journal of Climate and Meteorology 24* (1985), 887–903.

[14] SHIELDS, J., KOEHLER, T., AND KARR, M. Automated cloud cover & visibility systems for real time applications. Technical Note 217, Marine Physical Laboratory, SCRIPPS Institution of Oceanography, San Diego, CA, 92151-6400, January 1990.

[15] SHIELDS, J., KOEHLER, T., AND KARR, M. Automated cloud cover & visibility systems for real time applications. Technical Note 217, Marine Physical Laboratory, SCRIPPS Institution of Oceanography, San Diego, CA, 92151-6400, January 1990.

[16] URAS, S., GIROSI, F., VERRI, A., AND TORRE, V. A computational approach to motion perception. *Biological Cybernetics 60* (1988), 79–87.

UNLIMITED RELEASE
INITIAL DISTRIBUTION

MS 0735	D.A. Arvizu, 6200
	Attn: C.P. Cameron, 6215
	C.E. Tyner, 6216
	D.E. Hasti, 6218
MS 0722	D. Emgi, 6904
MS 9001	J.C. Crawford, 8000
	Attn: D.L. Crawford, 1900
	E.E. Ives, 5200
	J.B. Wright, 5300
	R.J. Detry, 8200
	W.J.McLean, 8300
	L.A. Hiles, 8400
	P.N. Smith, 8500
	L.A. West, 8600
	R.C. Wayne, 8700
	M.T. Dyer, 8800
MS 9004	M. E. John, 8100
MS 9103	G.A. Thomas, 8111
MS 9201	R.M. Wheeler, 8112
MS 9031	J.S. Swearengen, 8113
MS 9201	R.J. Gallagher, 8114
MS 9104	M.H. Rogers, 8115
MS 9202	R.L. Bierbaum, 8116
MS 9056	J. Vitko, Jr., 8102
MS 9056	L.R. Thorne, 8102
MS 9056	C.H. Sun, 8102
MS 9056	M. Lapp, 8102
MS 9214	L.M. Napolitano, 8117
MS 9214	M.C. Allmen, 8117 (15)
MS 9214	W.P. Kegelmeyer, Jr., 8117 (10)
MS 9014	K.A. Buch, 8351
MS 9021	Technical Communications for OSTI (10)
MS 9021	Technical Communications/Technical Library Processes, 7141
MS 0899	Technical Library Processes (4)
MS 9018	Central Technical Files (3)

END

DATE
FILMED
5/16/94