Mosaic Maps: 2D Information from Perspective Data

Industrial Representative: R. Berry (AMRDEC)

Faculty: C. Breward (Oxford, UK), J. Fehribach (WPI), M. Leeser (Northeastern), J. Ockendon (Oxford, UK), C.P. Please (Southampton, UK), D. Schult (Colgate), J. Shen (Minnesota), B.S. Tilley (Olin College)

Students: J. Grand-Maison (McGill, Canada), R. Jain (NJIT), K. Maki (Delaware), G. Miller (New Mexico Tech/EMRTC), J. Phillips (McGill, Canada)

Summary Presentation: D. Schult 16 June 2006

Report Preparation: B.S. Tilley

1 Introduction

The US military has a need to find detailed maps of a specific region that is current to within a few hours. The current technology for this involves Unmanned Aerial Vehicles (UAV), which are portable, are easy to maneuver, and can provide images at standard video refresh rates. The mosaic software from the Aviation and Missile Research, Development, and Engineering Center (AMRDEC) stitches images viewed from above by the UAV camera (30° field-of-view) providing a composite map of an area of interest as shown in Figure 1. In general, mosaicking is the process of combining multiple images to form a single image, or image mosaic. The image mosaic process involves geometrically aligning single images to create a composite image of a larger area. AMRDEC's mosaic software provides an almost real-time, composite map of all the images seen by the UAV camera. This non-georeferenced map can be georeferenced using a satellite map of the area and either manually or automatically matching geometric features. The end result is that the UAV operator has a current, geo-referenced map, of all the area seen by the UAV camera. The operator can then zoom in on areas of interest contained on the mosaic map.

However, most UAVs in the field have a single camera whose focus lies along the ray 30° from the fore-aft plane of the UAV. This perspective is ideal for flying the aircraft, and AMRDEC would like to modify its mosaicking algorithm to use these perspective images in place of top-down images (i.e. 90° from the fore-aft plane of the UAV). Many small UAVs currently fielded by the military are equipped with a forward-looking camera. Adding a down-looking camera to fielded UAVs would be an easy solution that would facilitate AMRDEC's current mosaic software capabilities. However, to retrofit all the fielded systems would be very expensive.

AMRDEC's mosaic software uses a Real Time Mosaic Processor purchased from Sarnoff Acadia to generate a Mosaic. The Acadia processor forms a mosaic image by:

1. Selecting three or four points in one image;

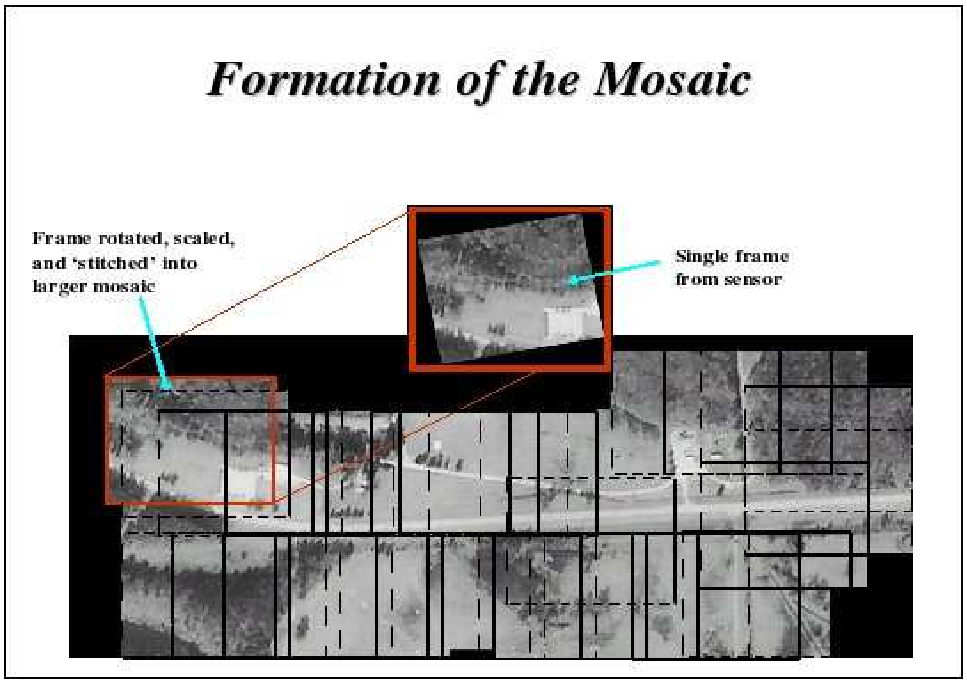

Figure 1: Example of mosaic image from a downward-looking camera generated by AMRDEC software. The goal during the workshop was to determine an algorithm that takes perspective images and generates a two-dimensional map.

2. Finding the same three or four points in a second image;
3. Determine the translation, rotation, scaling, and distortion between the two images by comparing changes in the points;
4. Geometrically correcting the second image;
5. Stitching the two images together by overlaying the three or four points;
6. Removing the overlap from the first image on the second image.

The algorithm currently uses the Canny edge/corner image processing scheme to identify numerous possible points for image matching. A Kalman filter is used to predict the change in position of these points between images. Three or four high confidence points are selected and used in the Mosaic process. The approach currently used does not require any information from the UAV's inertial instruments. This mosaic process continues connecting successive images to form a composite, real-time, map of the area seen by the UAV camera. This process works well for a downward looking camera. However, fielded UAVs such as Raven use a forward-looking camera and sometimes a side-looking wing camera. When a forward-looking camera looks at three-dimensional objects whose height is significant relative to the UAV's altitude, the higher objects are significantly distorted in the Mosaic

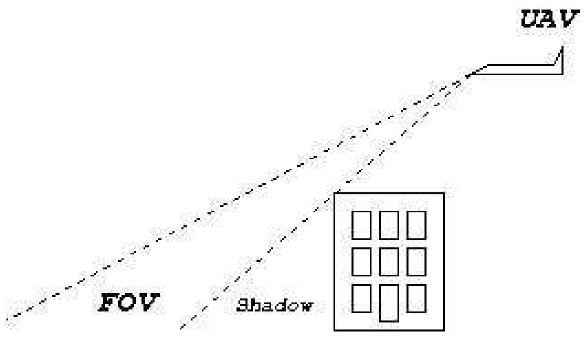

Figure 2: Cartoon of 'shadow' effect. Since the UAV's field-of-view (FOV) excludes the top-down view, there is a region near the structure that cannot be imaged if the UAV follows a linear trajectory.

image. This was first experienced when AMRDEC supported a test at the Air Force Research Laboratory in which their Mosaic ground station was used to process video from the BATCAM (Battlefield Air Targeting Camera Autonomous Micro-Air Vehicle) UAV. Since the BATCAM UAV contained a forward looking camera, the Mosaic image was significantly distorted.

The distortion is not due solely to perspective difficulties inherent in shifting a forward looking camera to an overhead view. The mosaicking process must find a translation, rotation, scaling and distortion between points on two images. With an overhead camera, any distortion due to differences in heights of the chosen tracking points is considered minimal. With a forward looking camera, however, objects of different heights require a different perspective shift. If the tracked points have different heights, the mosaicking transformation will warp the resulting image.

There are two significant problems with using perspective images to find two-dimensional data. The first is to identify regions of the image that are horizontal (i.e. parallel to the Earth's surface) and those that are vertical. Although the UAV is equipped with rate sensors (pitch, yaw, and roll), and altimeter, the precision of these instruments are not sufficiently precise to know the position of the UAV Hence, the relation of the current position to a given reference point becomes less reliable during the flight. In addition, the small UAV will experience significant body motion in flight due to turbulence and navigational response. Any algorithm that is developed should not depend on knowing an accurate position of the UAV.

The second major problem with this camera orientation is "shadowing". Effectively, the UAV will not be able to image information that exists behind a structure. A cartoon of this effect is shown in Figure 1. As the UAV flies over a fixed structure, there is horizontal information that will never be viewed by the camera. Obtaining the information behind these structures will require either the use of a second camera[1] or multiple flyovers of shadowed regions. The workshop did not attempt to find solutions to this second problem.

There were two approaches which were pursued during the workshop. The first approach was to use information on the confidence points over multiple frames in the mosaicking algorithm to find the orientation of the edge (or line segment connecting the two points) with the Earth's surface. Tracking points whose edges are parallel to the Earth's surface

[1]Some UAVs are equipped with cameras on one of the wings.

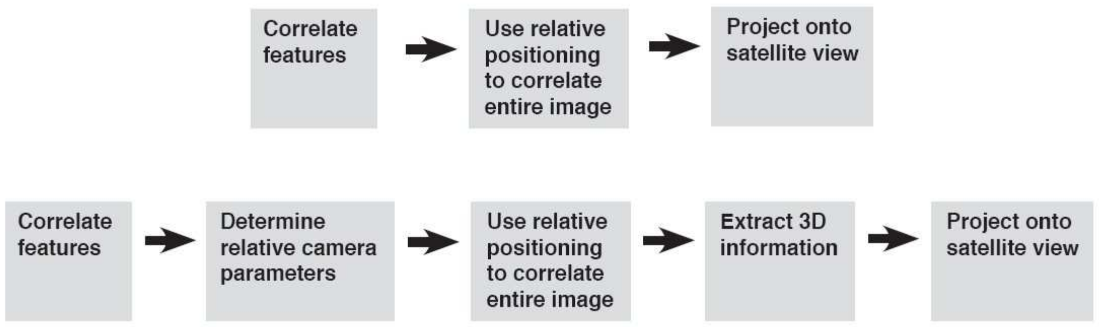
Correlate features
Use relative positioning to correlate entire image
Project onto satellite view
Correlate features
Determine relative camera parameters
Use relative positioning to correlate entire image
Extract 3D information
Project onto satellite view

Figure 3: Schematic diagram of each approach. The top approach does not attempt to store three-dimensional features of the scene, while the bottom approach attempts to recreate a discrete version of the environment. Note that each stage is prone to distortion and error, which can then propagate to the final mosaic.

identifies horizontal surfaces, which can then be used to correct the perspective. The second approach was to use multiple images to infer the three-dimensional geometry directly. From this construction, a two-dimensional projection can be readily determined. A schematic of both approaches is shown in Figure 3.

In both of these approaches, significant questions are still open. Firstly, the quality of the results needs to be explored. In Section 2, we discuss the different time and space scales that are present in the problem. An error analysis needs to be implemented to determine what are the bounds for the accuracy of the algorithm. The dominant error in both approaches (perspective) is present in the current top-down algorithm, and hence needs to be quantified. Secondly, the computational complexity for either approach needs to be determined. In both algorithms, the ability to determine real-time results based on the images remains to be explored.

The remainder of the report is organized as follows. In Section 2 we formulate the problem mathematically, and identify relevant parameters. In Section 3, we discuss how the current solution in the top-down algorithm needs to change for images in perspective data. In Section 4 we outline the mapping from the physical Earth plane to the image frame, and then identify geometrical models to describe how movement of the candidate points in the image frame relates to the geometry of the Earth frame. In Section 5 we discuss computer vision techniques to understand how to construct a three-dimensional world-view of the surroundings, and to find a two-dimensional projection from that. We conclude in Section 7.

2 Mathematical Preliminaries

In this section, we focus on the mathematical description of the problem, and the relevant dimensional scales. In Table 1 we outline the relevant length and time scales that appear in this problem. Note that curvature of the Earth scales on $H/R \approx 10^{-4}$, and hence we can assume that the reference surface of the Earth is a plane.

There are two reference frames which are outlined in Figure 4. The Earth frame (x, y, z) is the typical Cartesian frame, with y representing the distance normal from the reference frame. H denotes the UAV's height above this plane at any time t. The camera, however, takes images in its own reference frame (s, ϕ). The transformation from the Earth frame to the image frame is inherently nonlinear,

$$x(\phi, s) = H \cot(\phi) \tag{1}$$
$$z(\phi, s) = H\, s\, \csc(\phi)\ , \tag{2}$$

assuming that the UAV is flying parallel to the ground below.

Note that this transformation may be understood as a mapping of points by rays emanating from the UAV through points along the surface $y = f(x, z)$ to the reference plane $y = 0$. The goal of the workshop was to determine from a sequence of images in the image frame information about the surface $y = f(x, z)$ in the Earth frame. In Section 4, the image frame is used to determine information about the geometry of the surface in the Earth frame.

In order to use successive frames to gain information about the objects in the Earth frame, some care needs to be taken. From Table 1, we note that

Time Scales : $t_f/t_v \approx 0.2$ suggests that finding rates of change of geometrical variables (e.g. ϕ corresponding to different candidate points) through sequential images will be sensitive to changes in the UAV velocity.

Aspect Ratio: Given the typical flying altitude H, a characteristic feature height y_o, then the aspect ratio $\epsilon = \frac{y_o}{H} \sin\psi$, where ψ is the angle from vertical to the bottom of the imaged area, provides a measure of how perspective distorts the top-down measure of objects in the image. Note that a straight-down camera over a flat terrain ($\psi = 0$) no perspective distortion occurs, but increases near the edges of the image. Further features over a hilly terrain where $\epsilon = O(1)$, even for the top-down orientation, can lead to distortion in the mosaicked image. Note that $H \to \infty$ minimizes this error, but the resolution scale $x_p \to \infty$ in this limit.

There are two values which cannot be determined from an individual image. One is an absolute length scale, while the second is the pitch of the UAV with respect to the horizontal. The top-down view algorithm mitigates these by using scaling and rotation transformations on sequential images into the frame of the first image. Hence, although these quantities are unknown in all of the images, the current algorithm uses a common normalization for all of the images. Note, however, that if the three candidate points in the initial image do not lie in a plane parallel to the Earth, then the entire mosaic is distorted. This problem is exacerbated by images taken from a forward-looking camera, where the distortion can be far worse.

In all of the ideas that follow, additional information about the scene is needed to close the mathematical formulations. Information about the altitude of the UAV and the orientation of the UAV to the ground plane is relevant for simplifying each of these algorithms. In

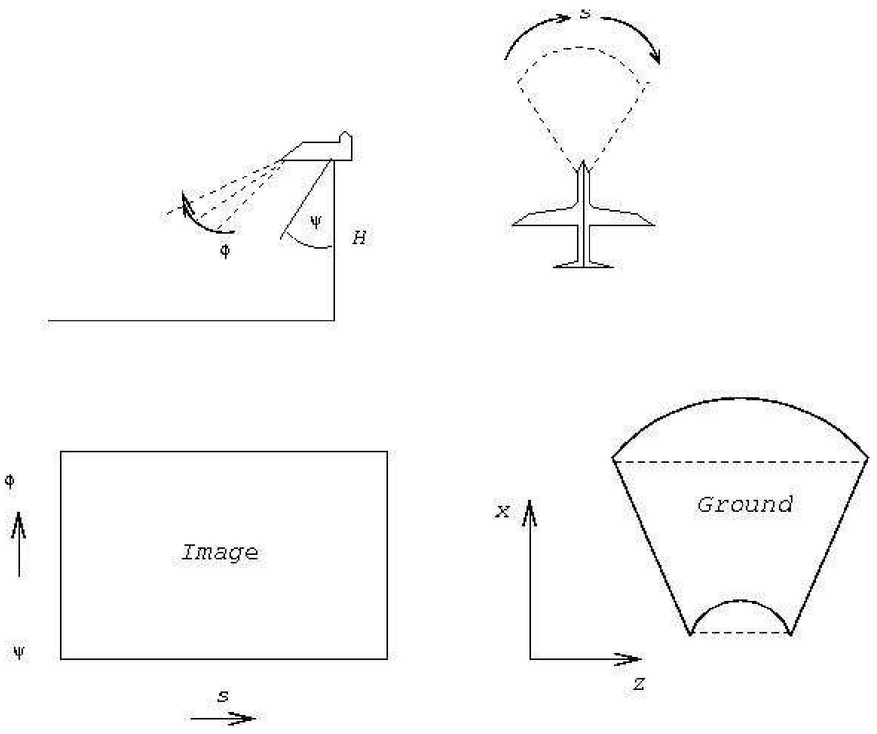

Figure 4: Schematic of transformations. Top-left: ϕ describes the angle at which the camera views the object. Top-right: angle s describes the transverse angle compared to the central axis of the UAV. Bottom-left: Image frame. Bottom-right: ground frame displayed in image. The dashed line corresponds to the ground truth when a pinhole-camera approximation is made (see Section 3).

the Section 7, we consider an error analysis based on the simplest of the models proposed to get an estimate of the required sampling rate of the instrumentation for these image techniques to work.

3 Current Algorithm and Perspective Modifications

Existing algorithms using Canny Edge Detection and Kalman Filter Tracking are sufficient for this application. AMRDEC currently uses such algorithms to track features from one image to the next. One potential improvement (if it is not already being done) which was discussed and implemented in a prototype Matlab script is to improve the initial guess for the Kalman filter. The typical Kalman Filter uses the current location of the feature as an initial guess for the location in the next image. We suggest using multiple previous images when available to predict the new positions. Multiple images allow us to estimate the velocity of the features so that a prediction of the location in the new image is much improved. Such a prediction may speed the Kalman filter by providing a ‘smart’ guess from which to start the search.

Variable (dimension)	*Description*	Value
H (m)	Vertical distance of UAV to Earth Plane	150
L (m)	Horizontal distance from the UAV to $\phi = \psi$.	150
R (m)	Radius of the Earth	6×10^6
a (m)	Aperature length of camera lens	0.1
N_p	Number of pixels	6000
x_o, y_o (m)	Characteristic structure length scale	3
$x_p = L/N_p$ (m)	Pixel resolution scale ($N_p \times N_p$)	0.1
V (m/s)	Characteristic horizontal velocity	20
$t_v = x_o/V$ (s)	Characteristic velocity time scale	0.15
t_f (s)	Frame-rate time scale	0.03
ψ (deg)	Reference angle with respect to vertical	$30 \le \psi \le 60$
ϕ (deg)	Coordinate angle from ψ	$0 \le \psi \le 30$

Table 1: Table of relevant dimensional quantities with typical values.

Once we have feature locations on two images, we must combine those images appropriately. Combining images is the heart of the mosaic process. Our algorithm is presumably similar to that used by AMRDEC for overhead camera systems. Given locations of three features on each of two overhead images, we transform the second image to correct for perspective change (warping) and translation (shifting) due to the movement of the camera. The transformation for the three points gives the position of each pixel in the new image in terms of coordinates from the old image. We interpolate the warped image values to the pixel locations of the old image's coordinate system–extended as necessary. This allows us to create a new larger image which overlays the two images. The coordinate system for the combined image is the same as for the initial image, allowing real time overlaying of a new image without jumping or jitter of the overall mosaic.

Overlaying the images can create "ghosting" if the top image is made partially transparent. We feel that this is an improvement over the current method of making the top image opaque. Keeping some information from previous images gives the viewer a good indication of the 3D position of objects. It also allows the viewer to see areas which later get covered as the plane flies over. Transparency levels of 20-30% seem to work reasonably well.

One problem with the mosaicking algorithm as described here is that it assumes that all features in both images lie in the same plane–the plane which passes through the three tracked features. Since the image is actually a 2D representation of 3D terrain, problems arise when the altitude of the three tracked features are disparate.

As an example, suppose the three tracked features are on level ground. A building between these points will appear warped by the transformation. But, if one of the tracked features is on the building and the other two are on the ground, the majority of the image will appear warped making the resulting combined image unreasonable. This is true for an overhead camera as well as a front facing camera, but it is much more problematic for a front facing camera since very little of the image lies in the same plane.

Thus, we suggest transforming the front facing image to an overhead one and making special efforts to ensure that the tracked features have the same altitude. One major focus of attention at the workshop was using multiple images (taken by a forward looking camera) to identify features with the same altitude.

Motivation

One key question of mosaicking is how to glue different frames in a robust way, so that no serious error propagation and amplification will occur during the process. Thus, one generally has to seek certain level of the so-called *ground truth* as the objective criterion.

It will be the most ideal to be able to completely reconstruct the ground terrain map from 2D image frames captured by a UAV camera. Several members of our team have been working hard to achieve this goal, as reported in other sections. This is, however, a highly nontrivial task since it involves too many unknown parameters, and is in essence an ill-posed inverse problem.

In this section, we focus on a very specific task: how to come up with an *objective* scaling rule so that the scaled image is proportional to the physical/ground size of its imaged area. In the following, we assume that the pinhole camera is the appropriate limit (see Figure 4).

The Computation

Our starting point is Figure 5, where the camera image square (or CCD plane) and the corresponding ground truth (the shaded trapezoidal area) are sketched. Here, we assume that the camera can be modeled by classical lens relations. The rectangular image on the CCD camera actually corresponds to a trapezoidal of area on the horizontal ground.

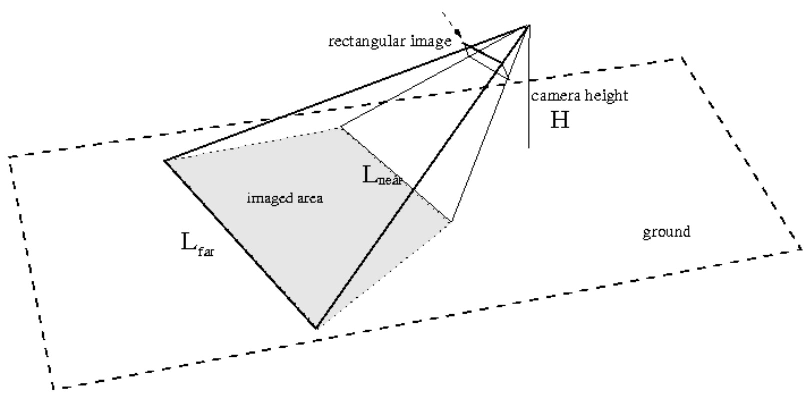

Figure 5: Overview of the setup

Let L_{near} and L_{far} denote the physical ground-truth lengths of the bottom and top lines of a captured image, respectively. (See Figure 5). Part of our goal is to figure out their ratio, without solving any complex vision problem.

The merit of our following approach resides in: although during the computation, various physical and camera constants are involved, the final formula is very clean, and only depends on the angle of the camera (or equivalently, the UAV).

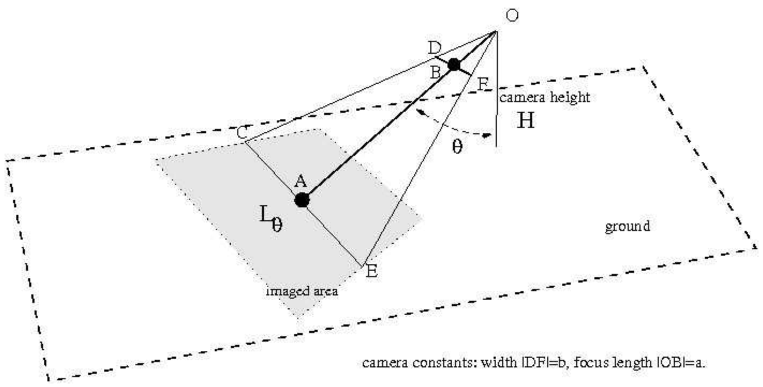

Figure 6: The physical/ground length L_θ imaged by an image line on the camera with (vertical) angle θ.

Instead of being restricted to L_{near} and L_{far} only, we will consider the physical length L_θ of a general image line with angle θ (See Figure 6).

First define two camera constants: the image width $b = |DF|$ and the focal length $a = |OB|$. Since the two triangles $\triangle OCE$ and $\triangle ODF$ are similar, we have

$$\frac{L_\theta}{b} = \frac{|OA|}{a}.$$

Suppose the height of the UAV is H. Then one has

$$H = |OAK| \cos\theta, \qquad \text{or} \quad |OA| = \frac{H}{\cos\theta}.$$

Therefore,

$$L_\theta = \frac{b}{a}\frac{H}{\cos\theta} = \frac{cH}{\cos\theta},$$

where $c = b/a$ is a camera constant which is not influenced by the positioning of the UAV.

In particular, we have

$$\frac{L_{far}}{L_{near}} = \frac{\cos\theta_{near}}{\cos\theta_{far}} \ ,$$

which is independent of the constants a, b, c, H.

In the laboratory example which was given to us (corresponding to the UAV system, see the left panel in Figure 7), the camera has viewing span of 30 degrees, and the central heading direction of 30 degree away from the vertical. Therefore,

$$\theta_{near} = \text{central angle} - \text{half of viewing span} = 30 - 15 = 15^\circ.$$

$$\theta_{far} = \text{central angle} - \text{half of viewing span} = 30 + 15 = 45^\circ.$$

Therefore, according to our prediction, we should have

$$\frac{L_{far}}{L_{near}} = \frac{\cos(\pi/12)}{\cos(\pi/4)} = 1.366.$$

On the right panel of Figure 7, we have scaled each horizontal image line according to our prediction formula. The results amazingly agree qualitatively well with those given to us from AMRDEC.

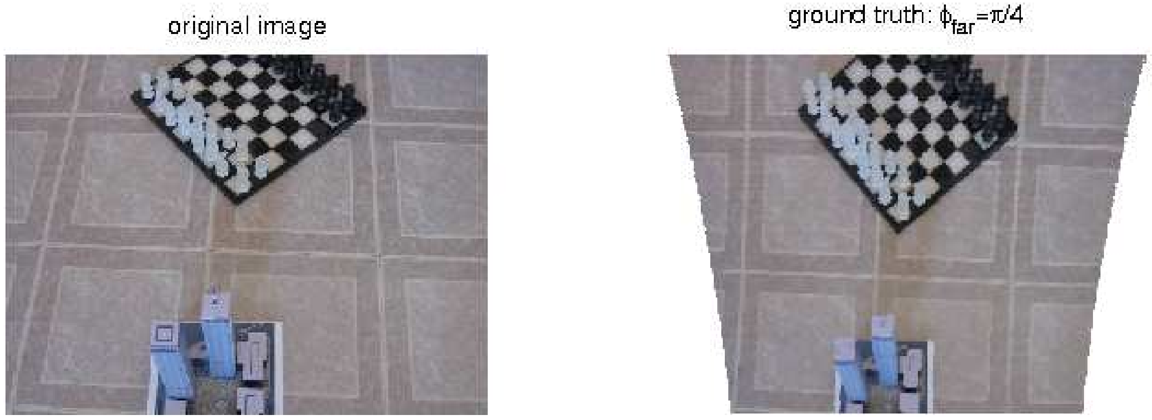

Figure 7: An example of image scaling according to our formula. It impressively automates the step which was previously *manually* done by the scientists at AMRDEC.

This is highly remarkable, since rectangular tiles of the floor have been faithfully scaled, without using any involved image processing tools or pattern recognition approaches. Note, however, that the camera orientation needs to be known in order to determine the appropriate scaling. This information cannot be found from the image directly.

4 Geometrical Models

4.1 Two-Dimensional Models

As a first attempt to affect a perspective shift, we guess an angle between the camera and the plane of the Earth and derive the transformation needed to shift to an overhead view. The transformation involves moving pixels of the image and interpolating between the moved points so as to create a new image with an approximate overhead view. The transformation should be exact for images of a two dimensional "planar Earth". In the following, we focus on the 2D case

In the 2-D case, we ignore the transverse direction. The plane of the Earth becomes a line of the Earth and we need to find the angle ϕ between that line and the line from the camera to an object in the image (see Figure 4.1). If we track K objects in the line of the Earth, we observe $K-1$ angles ϕ_k between objects k and $k+1$. If we track these objects across I images, we obtain $I(K-1)$ known angles. Along with the unknown origin angle ψ,

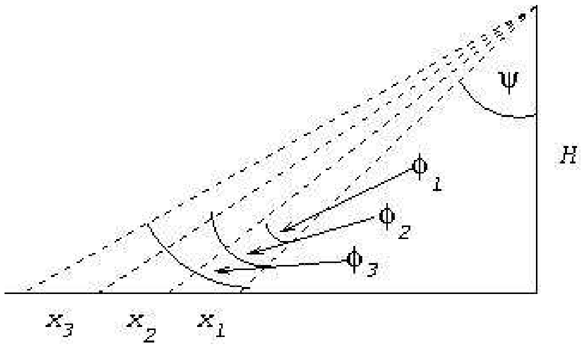

Figure 8: Schematic of point tracking. Notice that for each image, a range of angles $\phi_1, \phi_2, \ldots$ corresponds to horizontal distances $x_1, x_2, \ldots$.

each image requires that we obtain an unknown distance $\sqrt{L^2 + H^2}$ from camera to the first object x_1. Other unknowns are the actual distances x_k from object k to object $k+1$. Note that the x_k are fixed in space and do not change from image to image. If we assume that H is known, then we can relate one angle ϕ_k of each triangle with the length of one adjacent side and x_k. Thus we have $K-1$ equations from each image. With $I(K-1)$ equations, and $2I+K-1$ unknowns, we see that there is a trade-off between tracking more points and using more images. To obtain the origin of ψ with I images, one needs $K = (3I-1)/(I-1)$ objects (note $I > 1$). If we are given K objects we need $I = (K-1)/(K-3)$ images (note $K > 3$). Of special note is the case with two images where we must track 5 points to obtain the origin of ψ.

In the 3D case, we note that the K points are scattered throughout the image plane. This means that there exists $K(K-1)/2$ edges connecting each of these points, whose lengths can be determined. Thus, if the number of unknowns is given by the (x,z) ordinate of a feature, the height H and the UAV orientation θ_j) for each image (three Euler angles), the the number of images needed to close the problem is given by $I = 2(2K-1)/(K^2-K-8)$. Extensions to these ideas can be found in the Appendix.

Note, however, that the underlying problem has both translational symmetries in the (x,z) (or ground) plane, in addition to rotational symmetries in the UAV orientation. These symmetries must be removed by a choice of origin in order to solve in terms of the features of the frame. A simple 2D example is presented below to demonstrate this effect.

Finding Horizontal Planes

These attempts to orient the camera depend on the tracked objects being in the plane (or line) of the Earth. How do we tell whether objects are actually above the Earth? What affect does the 3-D nature of the imaged objects have on perspective shifts and mosaicking? Our approach is to track points across multiple images and use the relative motion of these points to determine points which lie in the plane of the Earth. What we can actually measure is if the points lie in a plane parallel to the flight of the camera, so by focusing on points spread throughout the image, we should find a horizontal plane which represents the Earth well. These deviations are a source of distortion (see Discussion).

Viewing how the lengths between points change from image to image can help determine

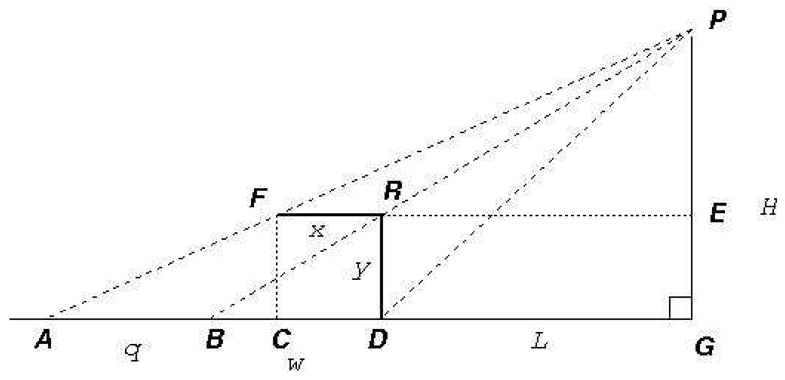

Figure 9: Idealization of plane building wall and roof **DRF** and how the lengths q and w are related to the angles ψ,θ, and ϕ.

which points lie in a plane parallel to the direction of camera movement. To determine the change in lengths, we analyze how the basic triangle from camera to object to ground changes during flight. Let H be the vertical distance from the camera to the plane of the Earth, L be the horizontal distance from camera to object. (See Figure 9). The distances q and w correspond to the lengths of the edges found in the image, and are assumed to be known up to a length scale. Initially, we assume that we know rates-of-change information about the angles, but this is not critical. We also assume that the height of the plane H remains fixed. This condition is not strictly true, and we discuss the impact of changes in H in the Discussion.

Note that triangles BDR and REP are similar which gives the relation

$$\frac{w}{y} = \frac{L}{H-y} \tag{3}$$

while similar triangles ACF and FEP gives

$$\frac{q+w-x}{y} = \frac{x+L}{H-y} \ . \tag{4}$$

Noting that x, y do not change in time, we can find how q, w change in time by taking the time derivatives of (3),(4), and assuming that $dH/dt = 0$, we find that

$$\frac{dw}{dt} = \frac{y}{H-y}\frac{dL}{dt} \tag{5}$$

$$\frac{dq}{dt} = \frac{y}{H-y}\frac{dL}{dt} - \frac{y}{H-y}\frac{dL}{dt} = 0 \ . \tag{6}$$

Thus, lengths of edges which represent surfaces parallel to the Earth reference plane do not grow in time. A discrete time version of this analysis returns the same result: for no changes in the vertical direction of the plane, the projection q will not vary. Note in that this formulation we assumed the heights of the roof were parallel to the ground plane. This need not be true.

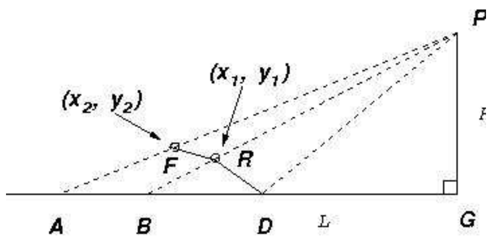

Figure 10: General surface analysis of Figure 9

An extension of this approach is shown in Figure 10. Here, we assume that the structure need not be rectilinear with respect to the ground plane. If again we assume that the orientation and height of the UAV remains fixed, then we find the following relations for q and w

$$\frac{q}{H} = \frac{x_2}{1-y_2} + \frac{x_1+l}{(1-y_1)(1-y_2)}(y_2-y_1) \tag{7}$$

$$\frac{w}{H} = \frac{x_1}{1-y_1} + \frac{y_1 l}{1-y_1} \tag{8}$$

where x_i, y_i are the x- and y-projections of the structure, scaled on the UAV altitude H, and $l = L/H$ is the scaled horizontal distance from the UAV to the first point D.

The ratio

$$\frac{dq/dl}{dw/dl} = \frac{y_2-y_1}{y_1(1-y_2)} = \frac{dq}{dw} ,$$

is a fixed quantity. If this related rate can be measured through the images, then we can reduce (7),(8) to

$$\frac{q - \frac{dq}{dw}w}{H} = \frac{x_2}{1-y_2} - \frac{dq}{dw}x_1 \tag{9}$$

$$\tag{10}$$

Unfortunately, we are two equations short to find the unknowns x_1, x_2, y_1, y_2. In the case where $x_1 = 0, y_1 = y_2$, then this reduces to our original formulation above. Note that no knowledge of the actual distance on the ground can be determined by this formulation. However, if the velocity information of the UAV is known up to some error, then y_1 is known in terms of $\Delta w = w^{(1)} - w^{(0)}$, the velocity, and the time step between frames. However, the ordinate x_1 cannot be evaluated unless information of the plane's orientation (i.e. the ψ angle) is known.

5 Mosaicking via 3D Reconstruction

The extraction of 3D information from 2D images is a well established area of computer vision research. The fundamentals of the mathematics involved can be found in [1, 2]. Chapters 24 and 25 of [3], describe algorithms for capturing 3D information from image sequences and multiple views. Practical examples of technologies that extract some (or all) the available 3D information include Google Earth, Google Maps, Microsoft Photosynth and Autostitch [4]. Note that the problem of panoramic stitching from images with unknown camera parameters solved in the last example is very similar to that of the mosaic.

If a good 3D model could be reconstructed from video images then the mosaicking problem would reduce to projecting the 3D model onto an appropriate image plane (i.e. the satellite view). In this section we review the ideas in the literature and show how they could be combined into an algorithm to solve the mosaic problem. We also discuss the computational expense of different approaches and make some suggestions about possible compromises. The reader should note that the technical details presented here are based on a cursory review of a large body of literature by a non-expert in the field. They are urged to consult the references directly for more accuracy, details and clarity.

A successful approach based on a complete or partial 3D reconstruction would also be a useful improvement to AMRDEC's existing algorithms for downward facing cameras, as it would remove the distortion reported for large roll or pitch angles or non-flat ground. It should also be able to deal well with data from a sideways facing camera.

5.1 Projective geometry

The problem of projecting a 3D model onto an image plane is straightforward and can easily be solved in realtime using off the shelf hardware for complicated 3D models, as can be seen in any modern video game, so we will not spend much time discussing it. However, it does serve as a useful starting point for the inverse problem - that of taking a 2D image and reconstructing the 3D model.

The pinhole camera model is a simple model for cameras. It assumes that a (visible) point in 3D space will appear in a photo at the point where the line intersecting it with the optical center of the camera intersects the image plane, see figure 11. The parameters necessary to determine the image are the position of the camera, its orientation and its focal length (the minimum distance from the image plane to the optical center). Given that we are normally only looking at a rectangular subsection of the image plane, it is also necessary to know what rectangle of the plane is represented by the image, or, equivalently, the image coordinates of the nearest point of the image plane to the optical center. Note that depending on how a photo is cropped, this may not lie at the center of the image or even within the photo at all.

In practice, commercial cameras can include non-linear distortion, for example because the lens is not accurately modelled by a pinhole. Given the nature of this problem presented by AMRDEC, we will assume any significant distortion can be removed by pre-processing and that the pinhole model is a reasonable one.

Suppose that we have two images of the same scene by two different cameras. We can

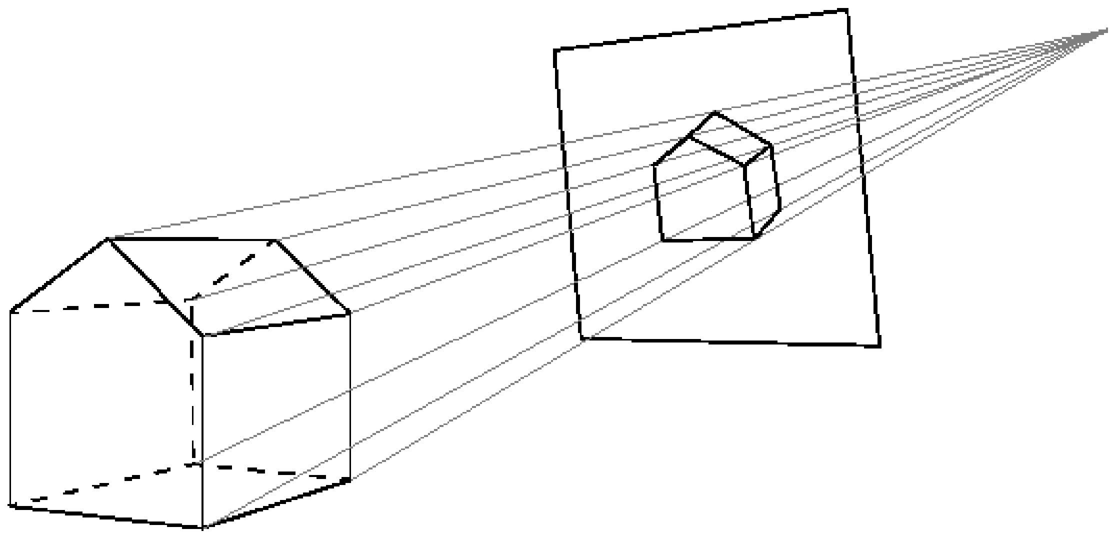

Figure 11: Image plane representation of a three-dimensional object with a pinhole camera model.

associate 3D coordinate frame $(x_i, y_i, z_i), i = 1, 2$ with each camera such that the optical center, C_i of each camera lies at the origin, the respective image planes are at $z_i = 1$ and x_i and y_i the match the image coordinates for any point that lies in the image plane. The points from the camera 1 coordinate system may be transformed to camera 2 using a rotation, R and a translation, $\mathbf{t}$, i.e. $\mathbf{x}' = R\mathbf{x} + \mathbf{t}$.

Figure 12 shows a point M appearing in both images with image coordinates (p_1, q_1) and (p_2, q_2) in the images of cameras 1 and 2 respectively and thus 3D coordinates $m_i = (p_i, q_i, 1)$. Note that the lines C_1M, C_2M, C_1C_2 are coplanar and use the above transformation formula to get the relationship:

$$m_1 \cdot (t \times Rm_2) = 0. \tag{11}$$

The cross product may be re-written as a matrix

$$t \times x = Tx \text{ where } T = \begin{bmatrix} 0 & -t_3 & t_2 \\ t_3 & 0 & -t_1 \\ -t_2 & t_1 & 0 \end{bmatrix} \tag{12}$$

Hence

$$m_1^t E m_2 = 0 \tag{13}$$

Where $E = TR$ is known as the "essential matrix". Any correlating point between two images produces a new linear equation in the entries of E. It is clear that, barring any degeneracy, eight points would provide a system of linear equations that could be solved

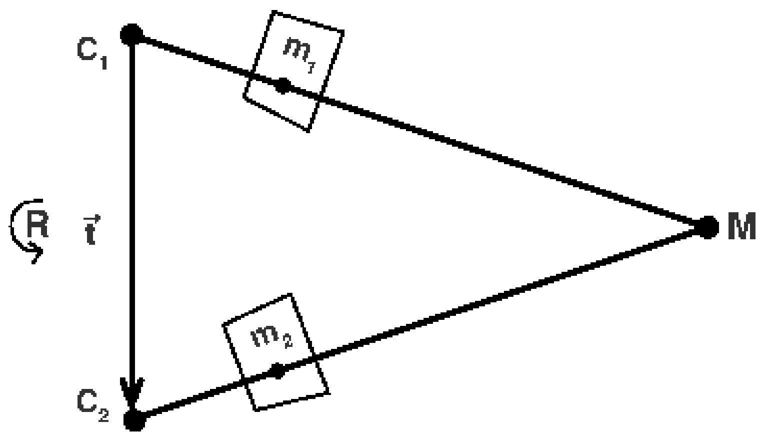

Figure 12: The image of a point in photos taken by two different cameras

to determine the entries of E.[2] Furthermore, it is possible to decompose E back into the translation and rotation, T and R. Thus, eight points of correlation between two photos are enough for us to determine (up to a scaling factor) the relative position and orientation of the two cameras that took the photos.

This is clearly simplistic and takes no account of how to cope with noisy data or false correlations. Practical applications take as many correlating points as possible and use a least squares approach to get the best possible fit. There are also approaches to eliminate false correlations. The reader is urged to consult chapter 7 of [1] for a thorough basic analysis.

Once the relative camera parameters are known, the position (up to a scale and a rotation) of any points correlated between the 2 images may be determined. Furthermore, the images can be "rectified". This means transforming one or both of the images so that their image planes are coplanar. A transformation between the coordinate systems of the two cameras now amounts to a simple translation in the xz-plane. It is standard to make the line C_1C_2 horizontal in this new coordinate system. The translation is now purely in the x direction and any further correlating points must lie on a horizontal line in the rectified image coordinate system. This makes the process of seeking more correlating points substantially easier.

[2]In fact, it is possible to show algebraically that because of the structure of E, only 5 points are needed, although that is of little practical use.

5.2 Algorithm

The process of mosaicking via 3D reconstruction may be broken down into substages, see the bottom scheme in Figure 3. Take two or more images, correlate several points between them and use the correlation of the points to determine the relative camera geometries for each image. Once the camera geometries have been established, try and map as much as possible of each image onto a 3D model. Finally, project the 3D model onto an appropriate image plane.

Eight or more correlated points are needed to lock down the camera geometries. If there are only small changes between each image, high quality correlations (i.e. lots of points, low false positives) between neighboring images are straightforward to achieve using the Kalman filtering techniques described above. The least squares process is very cheap and so it seems very reasonable to expect that this correlation could be achieved in real-time. However, given the ratio of the UAV velocity time scale and the sampling rate is not small, there may be some subtleties which were not considered at the workshop.

Triangulation

Performance issues aside, the natural approach now is to use the rectified images to attempt to correlate as many points as possible (remembering that we only need to search along a line to find potential correlations). These points can then be triangulated and each triangle “warped” onto the 3D mesh. Much more detail about this process, along with some impressive results and further references can be found in chapter 24 of [3]. Note that this step could be expensive for complicated scenes, but we have no data about the costs of this. However, there are many possibilities for optimization. If the change in camera position is small between consecutive images and the large distance from the camera to any features means that one could restrict the search area for matching features to a subset of the epipolar line. Depending on the objectives of the mosaicking process, the algorithm could be focused on newly revealed areas, ignoring areas that have previously been reconstructed (and where no movement is detected).

Area correlation

An alternative approach is to use an area correlation technique ([1], Section 6.4). Rather than matching specific image features, start with a pair of rectified images and measure optimal horizontal shift of a particular sub-region of one image to maximize the correlation with the other image. This horizontal shift is known as the disparity. The smaller the distance to a feature, the greater the disparity, thus the average distance to a sub-region can be determined from the optimal disparity measurement resulting from the correlation process. With small enough sub-regions and accurate enough data, this could be used to build up an accurate 3D map (in fact, it basically reduces to a fairly inefficient form of feature matching). However, the advantage of the correlation technique is that the computational cost of the correlation process can be controlled by choosing larger, more coarsely grained, sub-regions; the trade-off being that the coarser graining smooths out 3D model. For the

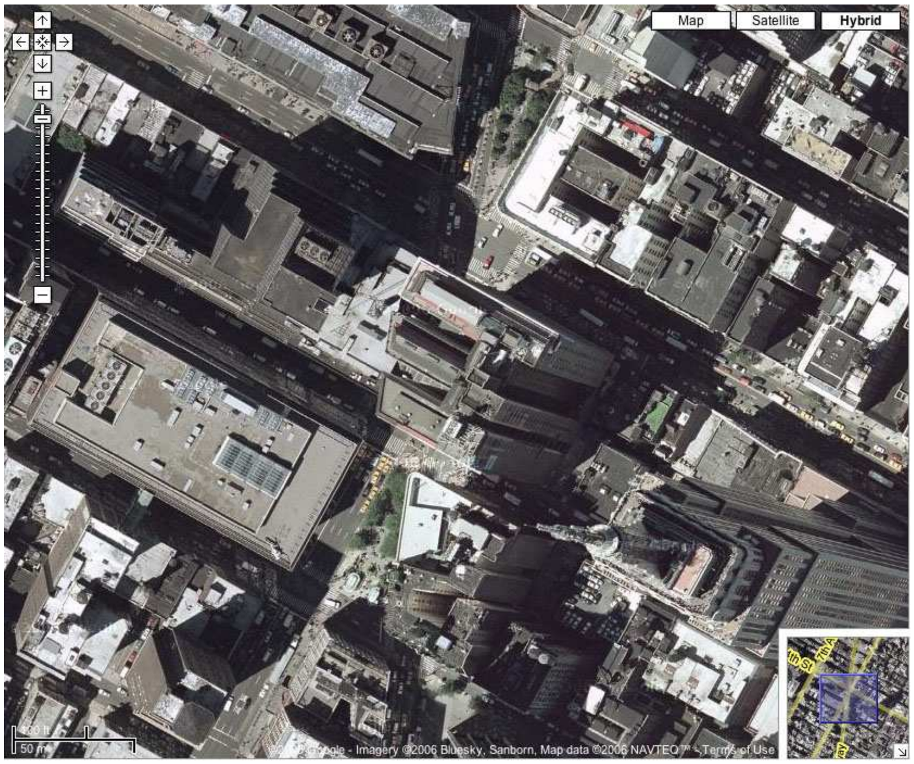
Map
Satellite
Hybrid
100 ft
50 m
©2006 Google - Imagery ©2006 Bluesky, Sanborn, Map data ©2006 NAVTEQ™ - Terms of Use

Figure 13: Broadway and 34th, New York, NY, http://maps.google.com

mosaicking problem, the sub-region size could be chosen to be an order of magnitude larger than buildings, thus producing an approximation of the topography.

Once the topography is established, we can map every pixel of the image to a satellite view as if it that pixel is at the topographic distance. The perspective of non-horizontal features in the resulting image will be distorted, but all features will be placed at or close to their correct geographic position. We call this the “Google Maps” effect. Figure 13 shows a clip of Google Maps where photos from three different perspectives have been stitched together. Note that in the mosaicking application a mixture of of perspective distortions like this would only be seen if the UAV revisited an area from a different direction. An examination of Google Maps should provide a good indication of whether this kind of perspective distortion would be too distracting to an operator.

6 Related Work

While much research has been conducted into mosaicking two dimensional images to form a larger map, and more work has been done in reconstructing three dimensional information from two dimensional data, there is less research into constructing a two dimensional map from images containing three dimensional information.

A survey article on image registration methods [5] breaks down the problem along similar lines that we considered. The four basic steps are feature detection, feature matching, mapping function design, and image translation and resampling. They further divide the problem into area based and feature based approaches. We take a feature based approach to image mosaicking. This survey article covers a very broad range of applications including remote sensing, medical imaging, computer vision and registering a scene to a model. Issues in remote sensing include many that are not relevant to the problem we studied, including occlusion of larger areas due to cloud cover. Also, most remote sensing approaches assume that the sensor is far enough away from the scene that 3D artifacts can be ignored.

Recently, a number of approaches have looked at mosaicking images from handheld cameras [6, 7, 8]. The paper most relevant to our approach looks at mosaicking images with parallax [6]. In this paper, the authors are looking at different views taken by a handheld camera, so they cannot exploit information such as the speed of the plane. The scenes used as examples are indoor scenes, and do not have the same issues of shadows and terrain differences evident in our problem. Still, there are many similarities. The authors make use of two heuristics. Their first heuristic uses features and ”planar plus parallax” transformations and is similar to the approaches that we studied. Their second approach uses all the pixels in the scene, and approximates the scene as a set of small planar patches. Other researchers [7] use a similar approach and information from a pair of cameras to separate an image into planar sub-scenes, and mosaic each sub-scene individually. We discarded these area based approaches as being to computationally expensive for the kinds of images being acquired from a UAV.

The problem we are investigating lies between remote sensing from satellite images and mosaicking images taken from handheld cameras on the ground.

Relevant references: [9] [10], [11].

Another approach to this project is to attempt to create a three dimensional model of the imaged objects. This is much more ambitious than 2-D image compilation. First the relative position and orientation of the camera must be inferred from multiple images and tracked points. Once these camera orientations are known, we can choose edges and points to construct a 3-D model of the objects on the ground. Another alternative is to break up the image into small (approximately planar) regions, track points in those regions to find their orientation, and connect the regions to create a topography of the area. Each of these steps have been done in other applications, but often with human intervention at critical points in the creation process. Algorithm speed is of course also important and it is not clear whether this approach would be feasible for a real time application.

7 Discussion

In this report, we have discussed several different methods for determining satellite-view data from perspective images. These methods can be categorized into trying to solve for the satellite view directly or to construct a three-dimensional view of the objects and then find their projections on the ground surface. Note that the former algorithm will require fewer computational resources than the 3D reconstruction, but it also provides less information about the environment. Further, both of these methods are subject to distortion (as is the current algorithm), and the perspective images on which they are based amplify the distortion errors.

In the process of developing these methods, the current top-down view algorithm is better understood and its errors are quantified. The current algorithm removes two symmetries that are found in the full three-dimensional problem. The first fixes the relevant angle ψ from vertical (since the ideal view is directly downward), while the mosaicking algorithm fixes the length scale based on the first image. Since the UAV is subject to significant movement both vertically and in terms of its heading, then this information, in some form, is needed to convert the feature information of the image into spatial information of the environment.

The UAV currently has instrumentation that determines heading and velocity over time, but to a relatively low degree of accuracy. However, even if this information is known to some error ϵ_I, it can be constructive to interpreting the image data for the mosaicking algorithm. For example, in the two-dimensional problem of finding ordinates (x_i, y_i) along the centerline of the image, we note that if the angle ψ is known to within some error $\alpha\epsilon_I$, and the altitude H is know up to some error $\beta\epsilon_I$, then from (3), (4),

$$y = \frac{wH}{H\tan\psi + w}\ ,\ x = \frac{q[H-y]}{H}\ .$$

which is accurate up to $O(\epsilon)$. To correct for the instrumentation error, a sequence of measurements $(x^{(i)}, y^{(i)})$ can be calculated and standard statistical methods can be used to determine better approximations for the true values of (x, y).

Appendix: Three-Dimensional Geometrical Equations

From the geometrical viewpoint a useful paradigm problem is that of locating points P_k with $k = 1, 2, \ldots, K$ on a flat earth from a series of images $(i = 1, \ldots, I)$ of these points taken a camera moving arbitrarily above the earth. The calculations are simplest to motivate by restricting the problem to 2-D. There are two frames of reference that were considered to study the problem. The first, shown in figure 14 is fixed in the earth. It was however is easier to think of the camera as being at the center of a "planetarium" with the points P_k producing stars on the spherical surface that move as i changes, these points have polar angle $\theta_k^{(i)}$ with respect to some axis as shown in figure 15.

The key question we ask is "for a given number of points K how large must I be for us to be able to find the P_k from the $\theta_k^{(i)}$?"

We observe $(K-1)$ values of $\theta_k^{(i)}$ $(k = 2\ldots, K)$ in each image but cannot observe the unknowns of the height $H^{(i)}$ and the orientation $\theta_0^{(i)}$ of the camera (in the planetarium frame these corresponds to unknowns $\rho^{(i)}$ and $\psi^{(i)}$). Thus by applying simple geometry we have the $K-1$ equations

$$P_k = H^{(i)}\tan(\theta_1^{(i)} + \theta_k^{(i)}) \qquad\qquad k = 2, \ldots K$$

From these $I(K-1)$ equations we have to determine the $K + 2I$ unknowns P_k, θ_1^i and H^i. In determining these we know that there is a scale invariance so we can arbitrarily set the

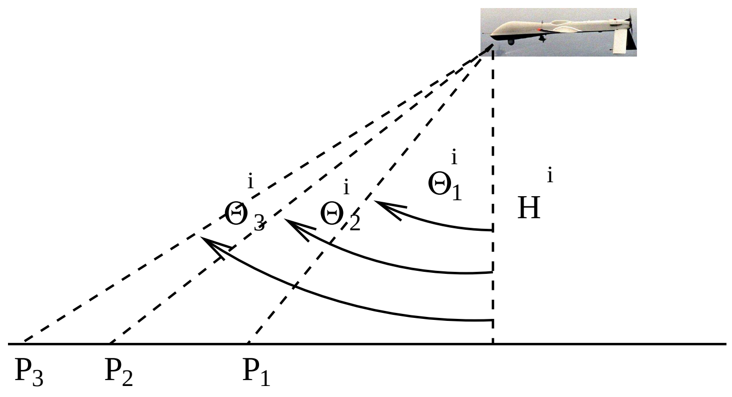

Figure 14:

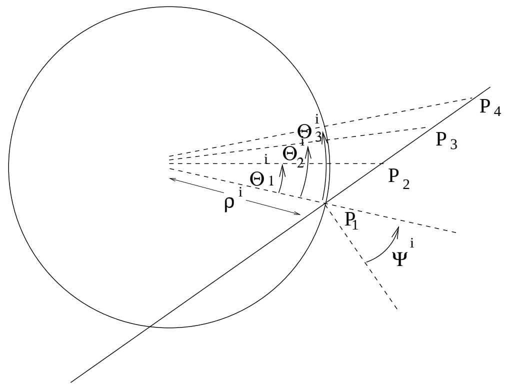

Figure 15:

origin of the axes to be $P_1 = 0$ and set a length scale by taking $P_2 = 1$. For the solution to be determined we therefore require

$$I(K-1) \geq K + 2I - 2 \qquad \text{or} \qquad I \geq \frac{K-2}{K-3}$$

The previous naive argument can be generalized to more practical situations. First for the case where we consider a 2D problem but the earth is no longer taken to be flat, so that the P_k are not collinear. We take the $\theta_k^{(i)}$ to be the measured polar angles between the points P_1 and P_k (we can consider these 2D positions as complex variables) referred to an aircraft whose position (again complex) is $z^{(i)}$. Hence

$$\theta_k^{(i)} = \arg\left(\frac{z^{(i)} - P_k}{z^{(i)} - P_1}\right) \qquad k = 2, \ldots, K, \quad i = 1, \ldots, I$$

There are therefore $I(K-1)$ such nonlinear algebraic equations. These equations must be solved for the unknown $2I$ coordinate components of the $z^{(i)}$ and the $2K-4$ coordinate components of the P_k (note that P_1 can arbitrarily be assigned to $(0,0)$ and P_2 to $(1,0)$ in order to assign the coordinate system in the earth). Hence there may be sufficient equations if

$$I \geq \frac{K-2}{K-3}$$

Note that if we impose the additional requirement that the P_k be collinear (the earth is flat) then we have $I(K-1)$ equations for $2I + K - 2$ unknowns which retrieves the results in (7).

References

[1] Faugeras, O. (1993) *Three-Dimensional computer vision : a geometric viewpoint*. MIT Press.

[2] Emanuele Trucco, A. V. (1998) *Introductory techniques for 3-D computer vision*. Prentice Hall.

[3] Nikos Pargios, O. F., Yunmei Chen (ed.) (2006) *Handbook of mathematical models in computer vision*. Springer.

[4] Lowe, D., http://www.cs.ubc.ca/ mbrown/panorama/panorama.html.

[5] Zitova, B. and Flusser, J. (2003) Image registration methods: a survey. *Image and Vision Computing*, **21**, 977–1000.

[6] Dornaika, F. and Chung, R. (2004) Mosaicking images with parallax. *Signal Processing: Image Communication*, **19**, 771–786.

[7] Gorges, N., M. Hanheide, W. C., Bauckhage, C., Sagerer, G., and Kittler, J. (2004) Mosaics from arbitrary stereo video sequences. *PATTERN RECOGNITION LECTURE NOTES IN COMPUTER SCIENCE*, **3175**, 342–349.

[8] Cheung, M. T. and Chung, R. (2003) Video mosaicking for arbitrary scene imaged under arbitrary camera motion. *COMPUTER ANALYSIS OF IMAGES AND PATTERNS, LECTURE NOTES IN COMPUTER SCIENCE*, **2756**, 254–261.

[9] Hsu, C.-T., Cheng, T.-H., Beuker, R. A., and Horng, J.-K. (2000) Feature-based video mosaic. *International Coference on Image Processing (ICIP)*, vol. 2, pp. 887–890.

[10] Lu, S., Yan, L., and Zhao, H. (2005) Parallel-perspective stereo mosaics of video images from unmanned aerial vehicle remote sensing system. *IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, vol. 6, pp. 3860– 3863.

[11] Majumdar, J., Vinay, S., and Selvi, S. (2004) Registration and mosaicing for images obtained from UAV. *Signal Processing and Communications (SPCOM)*, pp. 198– 203.