Australian Government
Department of Defence
Defence Science and
Technology Organisation

An overview of geolocation of airborne video using 3D models

Tristrom Cooke, Robert Whatmough, Nicholas Redding and Edwin El-Mahassni

Intelligence, Surveillance and Reconnaissance Division

Defence Science and Technology Organisation

DSTO–TR–2001

ABSTRACT

This is an overview report which describes a method for automatic geolocation of video from an airborne sensor. The approach described here uses positional information from three sources to compute refined coordinates in three dimensions for any feature in the video sequence. These three sources are: firstly, sensor-platform metadata describing the likely sensor footprint based on sensor-platform positional and attitudinal information; secondly, 3D information of a scene inherent in a video sequence collected from a moving platform; and thirdly, reference imagery of the region of interest that is geolocated and georectified such as aerial photography. The report describes the steps involved in this process, which have been successfully applied individually to two types of imagery (infrared MX-20 data, and high definition data from project Crystal View). Investigation into the final 2D registration stage and 3D registration with a CAD model is ongoing.

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE **JAN 2008**	2. REPORT TYPE	3. DATES COVERED **00-00-2008 to 00-00-2008**

4. TITLE AND SUBTITLE **An Overview of Geolocation of Airborne Video Using 3D Models**	5a. CONTRACT NUMBER
	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **Australian Government,Department of Defense,Defence Science and Technology Organisation,Australia,**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR’S ACRONYM(S)
	11. SPONSOR/MONITOR’S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES

14. ABSTRACT

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**	**Same as Report (SAR)**	**18**	

Published by

Defence Science and Technology Organisation
PO Box 1500
Edinburgh, South Australia 5111, Australia

Telephone: *(08) 8259 5555*
Facsimile: *(08) 8259 6567*

AR No. 014-096
January 2008

An overview of geolocation of airborne video using 3D models

EXECUTIVE SUMMARY

The information from airborne video sequences cannot be exploited fully from the imagery alone. The position in which the imagery was observed is also vital. Currently, this information is obtained from an embedded metadata stream, which provides platform position and sensor orientation data. This data is generally only sufficient for a crude estimate of the position of the imaged scene. More accurate information could be achieved by registering this data to existing well calibrated sources of imagery such as from an aerial survey.

A hindrance to the automatic registration of the video to the reference imagery is the difference in pose of the two sensors. In this case, the 3D characteristics of the scene would alter the observability and relative positions of each point in the image, so that the video cannot be related to the image by a simple linear transformation. This report provides an overview of the research conducted by DSTO into a system for automatic video registration. The following is a list of reports and conference papers related to this topic, some of which have been summarised in the current report:

- T.Cooke and R.Whatmough, "Detection and tracking of corner points for structure from motion," Technical Report, DSTO-TR-1759, August 2005.
- T.Cooke and R.Whatmough, "Evaluation of corner point detectors for structure from motion problems," Proceedings of DICTA, Cairns, December 2005.
- T.Cooke and R.Whatmough, "Using learning algorithms to improve corner detection," Proceedings of DICTA, Cairns, December 2005.
- R.Whatmough, "Combining shape from motion output with partial metadata," IEEE Conference on Advanced Video and Signal-based Surveillance, Sydney, November 2006.
- T.Cooke, R.Whatmough, N.Redding, G.Ewing and E.El-Mahassni, "On the extraction of 3D models from airborne video sensors for geolocation," Presented at DASP 2006, to appear in Digital Signal Processing.
- E.El-Mahassni and T.Cooke, "A survey on the suitability of some recent 3D surface reconstruction algorithms for airborne sensor imagery," DSTO-TR–2064, October 2007.
- E.El-Mahassni, "New robust matching cost functions for stereo vision," Proceedings of DICTA, Adelaide, December 2007.
- T.Cooke, "An empirical analysis of errors in structure from motion," Proceedings of DICTA, Adelaide, December 2007.

- T.Cooke, “Automatic extraction of 3D models from an airborne video sequence,” DSTO-TR-2095, January 2008.
- T.Cooke, R.Whatmough, N.Redding and E.El-Mahassni, “An overview of geolocation of airborne video using 3D models,” DSTO-TR-2001, January 2008.
- R.Whatmough, “Extracting the shape of a target from an image sequence with incomplete metadata,” DSTO-TR-2101, February 2008.
- R.Whatmough, “Error analysis of shape from motion extraction with incomplete metadata,” DSTO Technical Report in publication, DSTO-TR-2102, February 2008.
- R.Whatmough, “Registration of a Shape-From-Motion reconstruction to a geolocated 3-D model,” DSTO Technical Report in publication, DSTO-TR-2103, February 2008.

Authors

Tristrom Cooke

Information, Surveillance and Reconnaissance Division

This author obtained a B.Eng (Hons) in Electrical Engineering from the University of South Australia in 1992, a B.Sc (Hons) in Applied Mathematics from the University of Adelaide in 1995, and completed a PhD in Applied Mathematics (also at the University of Adelaide) in the area of thermo-elasticity at the end of 1998. He was then employed by CSSIP (CRC for Sensor Signals and Information Processing) until 2005, where he has mostly worked on projects relating to recognition of targets in both SAR and ISAR radar imagery. He is now full time employee in ISRD of DSTO, where he is working on structure from motion.

Robert Whatmough

Information, Surveillance and Reconnaissance Division

Robert Whatmough graduated from Monash University with the degree of Bachelor of Science with Honours in 1969. He joined the Weapons Research Establishment, which later became part of Defence Science and Technology Organisation, working on a mixture of scientific data processing problems including computer graphics and remote sensing. He was made Senior Research Scientist in 1986 and has since worked in the areas of image and video processing, including remote sensing and shape inference.

Nicholas J. Redding
Information, Surveillance and Reconnaissance Division

Nicholas J. Redding received a B.E. and Ph.D. in electrical engineering all from the University of Queensland, Brisbane, in 1986 and 1991, respectively. In 1988 he received a Research Scientist Fellowship from the Australian Defence Science and Technology Organisation (DSTO) and then joined DSTO in Adelaide as a Research Scientist after completing his Ph.D. in artificial neural networks in 1991. He was appointed a Senior Research Scientist in 1996. In 2000/2001 he was awarded a Defence Science Fellowship and was posted to the UKs Defence Evaluation and Research Agency. In 2004, he was appointed Head of the Image Analysis and Exploitation Group in DSTO, and received the DSTO Achievement Award for Excellence in Science and Technology. Since joining DSTO he has applied image processing and computer vision techniques to detection and classification problems in imagery from ionospheric sounders, synthetic aperture radar, and electro-optic still and video sensors.

Edwin El-Mahassni
Information, Surveillance and Reconnaissance Division

Edwin El-Mahassni received his M.Sc. degree from the Department of Mathematics, Macquarie University, Australia. He is currently a PhD candidate in the Department of Computing in the same university. He joined the Defence Science and Technology Organisation (DSTO), Australia in 2002 and has worked and published papers in the areas of signal and image processing, as well as, theoretical computer science.

Contents

Figures

1 Introduction

Exploitation of video from airborne sensor systems requires knowledge of the location of features in the sequence in real-world coordinates. This is particularly the case in surveillance and reconnaissance applications from airborne platforms such as unmanned aerial vehicles (UAVs). Typically, these sensor platforms will have some form of positional awareness of both the sensor, its viewing geometry, and its optical parameters such as focal length. These are used to compute the expected footprint that any video frame forms from its view of the ground. This data is usually referred to as the *metadata*, and in the better engineered sensor systems it will be incorporated with the video data as a synchronised private data stream. For many applications, the geolocation accuracy of the metadata from the sensor system is not sufficient, and determining accurate coordinates for a visible feature in a video sequence is a manually intensive activity that involves searching geolocated and georectified reference imagery for the corresponding feature from which to determine the feature's location. One answer to this problem is to re-engineer the sensor system to actively sense the location of the scene features using such things as a laser range finder or even an integrated LIDAR (LIght Detection And Ranging) which determine more parameters about the location that are matched with geographic data for more accurate geolocation. We take a different approach to the problem by automatically exploiting the 3D content inherent in the video sequence, because it has been collected from a moving platform, in conjunction with the geolocated and georectified reference imagery to compute more accurate geolocations for features in the imagery.

This report describes a system for accurate geolocation of objects in an airborne video scene, by automatically registering it to a geolocated reference image. While the registration of 2D images is a well studied problem, in the current application, registration of the video frames to an image is made non-trivial due to the real world being three dimensional. Differences in the pose of the two imaging sensors may result in very different images of structures with any appreciable 3D structure (such as buildings, which are the focus of the current report). Although an individual frame from a video contains no explicit depth information, such data may be inferred from the video sequence by exploiting changes in the image with a change in viewpoint produced by the movement of the platform. This may then be used to compensate for the difference in viewpoint between the video and the reference image, and allow a relatively straightforward 2D registration to geolocate the video in latitude and longitude. Theoretically, a point could also be located in height if terrain data for the area is available.

The proposed system for georegistration consists of using the sequence of frames from the video to construct a dense, textured 3D model of the area of interest. This model is then reprojected to appear as if it were imaged from the same direction as the reference imagery, therefore removing any 3D projection differences between the models (although differences in shadowing will still be present). Standard 2D registration techniques can then be applied to register the reprojected model (and hence the video sequence frames) to the geolocated reference imagery.

1.1 Process overview

Figure 1 shows a detailed flowchart of the suggested registration algorithm. The extraction of the dense, textured 3D model is accomplished by the first four steps. The first step is the detection and tracking of feature points throughout the relevant part of the video sequence. This step effectively reduces the amount of data required to be processed from 10^6 pixel values for each frame to a few hundred, which can be processed much more easily. Although no longer a feature of the current processing, it was originally envisioned that these corner points could be chosen to encapsulate all of the important information of the scene, and the rest of the imagery could be effectively discarded. This led to some significant research into improving and evaluating corner detectors. Section 2 describes the results of some of this research, as well as a brief description of the corner detector and tracker proposed for the automatic geolocation system. The output of this first stage is a

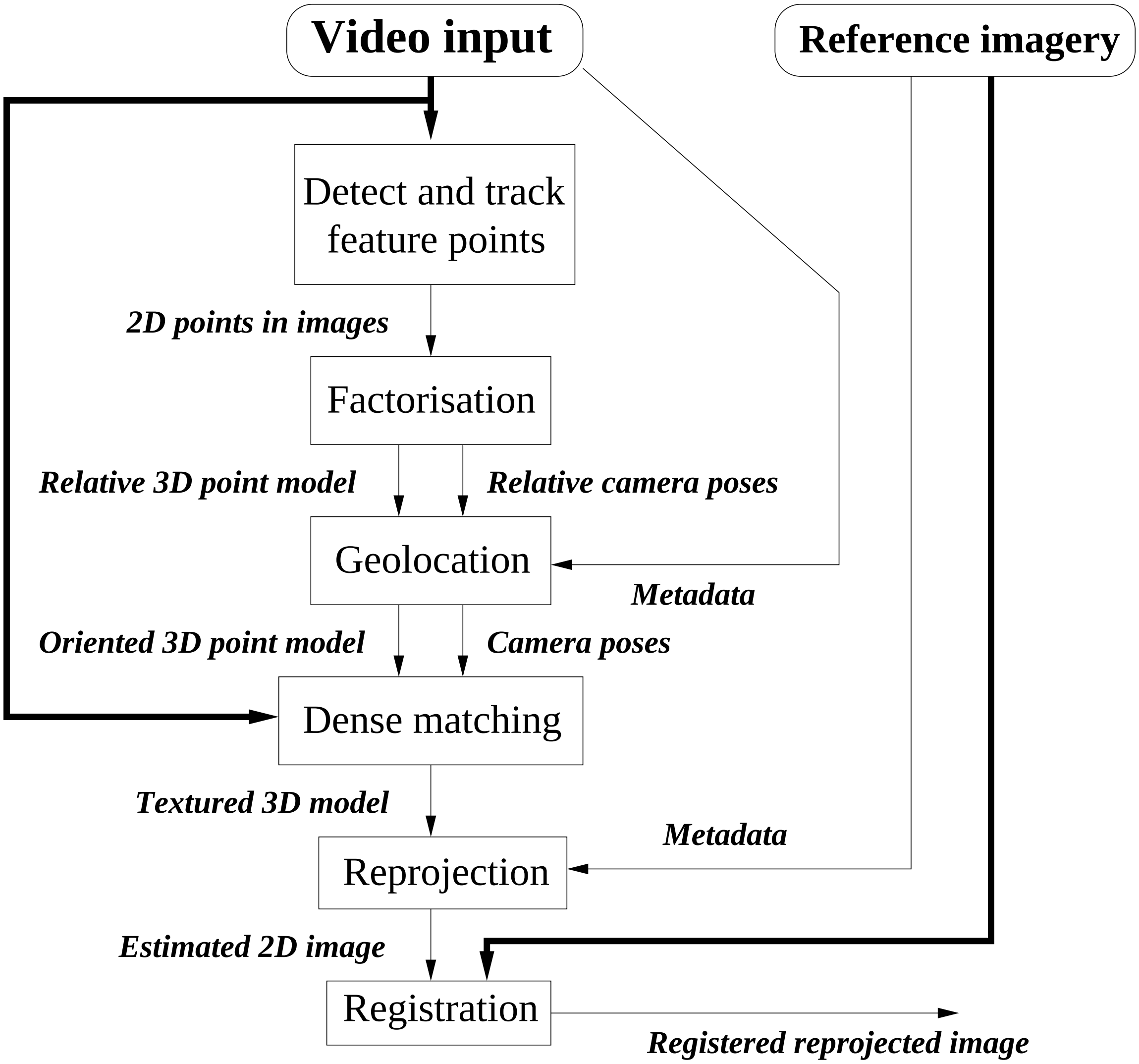

Figure 1: *Flowchart of steps required for registering disparate sources of imagery*

set of measurements $(x_i(t), y_i(t))$ of pixel locations for a set of corner points i throughout each of the video frames t.

The second stage of reconstructing the dense 3D model is referred to here as factorisation. This is the name of the fast singular value decomposition method used to construct a sparse 3D model from a complete set of (x, y) measurements of points in all of the image frames. In practice, this stage will also contain some additional processing to account for missing and faulty tracker data from the previous stage. A number of methods have been examined for filling in missing data, removing inconsistent points, and joining broken tracks. These methods have proven effective on the two test video sequences, and are described, along with the actual factorisation technique, in Section 3. The output of this stage is two matrices, P' of camera poses and Q' of coordinates of sparse model points, subject to unknown scaling, rotation, and possibly reflection and other distortion that cannot be found from the video frames alone. The corrected values for these matrices can be found by applying the equations

$$P = P'T \quad \text{and} \quad Q = T^{-1}Q' \tag{1}$$

where T is a small matrix, which is yet to be determined. By utilising metadata associated with the sensor and its platform, the value of T required for the equations (1) can be found by minimising the differences between P and values found from the metadata in a least squares sense. Then Q gives a correctly oriented and less distorted structure model and P gives camera poses more reliable than those from factorisation or metadata alone. A more detailed description of the geolocation process is given in Section 4.

The combination of the factorisation camera pose estimates with the metadata produces improved estimates for the camera poses in each of the frames. A geolocated estimate for the position of the scene may then be produced by footprint area calculation from the refined poses. The resulting estimate is still somewhat crude, which is why registration to ground-truthed imagery is deemed necessary. To avoid problems with registering a set of points with an image, the approach taken was to produce a textured model of the scene, which requires an estimate of the depth of every point within the image scene. Originally, this was to have been found by collecting groups of corner points into facets, and essentially constructing a wireframe model of the scene, onto which texture from the imagery could be mapped. This approach was tested, but there were a number of difficulties which could not be easily overcome, so instead a dense matching algorithm was applied, as described in Section 5. The dense matching implementation is based on a stereo image pair taken from the start and end of the sequence. These are assumed to be the most widely spaced in terms of camera geometry. Using the camera estimates from the geolocation step, the images are rotated and scaled so that the optical flow from one to the other, due to differences in depth, will be aligned in the vertical direction. The optical flow in the vertical direction is then found for each pixel in the image simultaneously by minimising the global match cost between the images, with a penalty term for discontinuities to produce a smoothed solution. A graph-cuts based method is used for this. The camera pose information then relates the optical flow to the height, so that a dense georectified 3D model of the scene is obtained.

Once the dense 3D model is obtained, standard techniques are available for adding texture from one of the images used to construct the model, and to reproject it to any direction required, including the direction of the reference ground-truth imagery. This process, referred to in the flowchart as reprojection, is briefly discussed to at the end of Section 5.

The final step of the process is to accurately determine the location of the scene in the video sequence by comparing the reprojected model with the geolocated reference imagery. The 2D image registration problem is considered to be a solved problem, and there are a number of approaches implemented within the ADSS framework that are likely to work well with the types of problems that are being dealt with. Section 6 gives a brief technical overview of one of the registration methods that was successful in registering a reprojected model calculated using the method outlined in this report, with a separate source georeferenced imagery. Some further work on registration to an existing 3D CAD model is expected to appear in a later report [20].

The georegistration that is the result of the complete process will have some degree of error associated with it. Errors will be introduced, or modified, at each stage of the process, but need to be quantified in some way so that the user of the system can have some confidence in the results. Section 7 describes a framework which has been developed for evaluating the error at each stage of the registration process. The framework has been shown to provide usable error estimates for a subset of the steps in the complete process. A more complete evaluation would require further work on registration and more cohesive integration of the stages within the process. A summary and some conclusions about the complete system are outlined in Section 8.

2 Detection and tracking of feature points

Detection and tracking of feature points is the first stage of the process for the automatic video geolocation task described in the introduction. There are two primary reasons for this step to be included. Firstly, it reduces the amount of data required to be processed from the billions of pixels in an image stream down to merely hundreds of tracked points in thousands, or even just tens of images if frame subsampling occurs. This is roughly a million fold reduction in the amount of data to process, which allows it to be processed in a reasonable amount of time. The second related reason is that points are easier to handle than are whole images for later camera and structure estimation modules. This is not just because of the relative size of the data. In order to determine the 3D position of a point in an image, it is necessary to find a correspondence to that point in at least one other image. A dense 3D model would require this for every point in the image which could, in theory, be computed using optical flow techniques. Most existing optical flow algorithms have some difficulty in determining the motion of areas of fairly uniform intensity, which is frequently a large proportion of the image. The more robust approach taken here effectively computes the optical flow only at selected points within the imagery, which can then be used to constrain the solution over the remainder of the image.

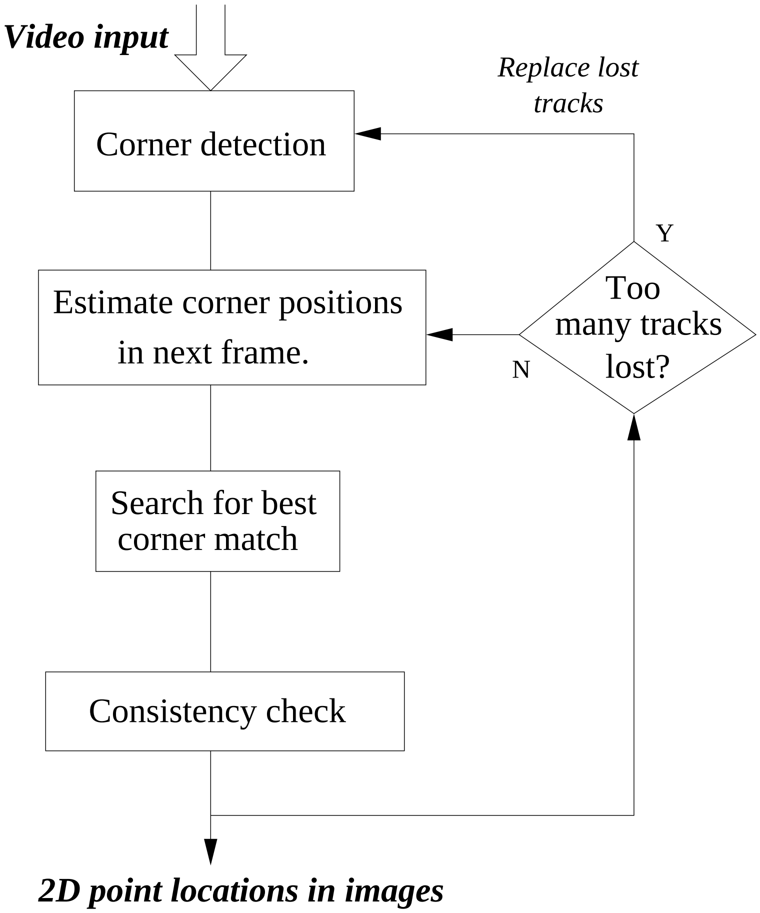
Video input
Replace lost tracks
Corner detection
Y
Estimate corner positions in next frame.
Too many tracks lost?
N
Search for best corner match
Consistency check
2D point locations in images

Figure 2: *A flow diagram of Stage 1: Detection and tracking of corner points*

Figure 2 shows the detailed structure of the detection and tracking. The recommended method is the KLT (Kanade, Lucas and Tomasi) method [15], which is one of the more commonly used video processing algorithms and makes up the first three steps of the flowchart.

The first step is to detect prominent points (variously referred to as corners or features) in an image from the video sequence. The KLT tracker has a default method, referred to as the Shi-Tomasi [12] detector, for doing this. Earlier in the development of this project, it was thought that using a more judicious choice of tracked features could allow a dense 3D model to be produced without any further reference to the video frames. This could be achieved by detecting the corners of building structures, and segmenting these corners into facets from which a 3D model could be built up. In support of this idea, algorithms for more accurately detecting corners were required. A DSTO report (DSTO-TR-1759) [2] was written by Cooke and Whatmough on the research in this area, much of which was summarised in two conference papers [3],[4]. The report contained technical information on seven different commonly used corner detectors from the literature, and a protocol under which the performance of the detectors could be empirically determined. The corner detectors were then compared and it was found that the Harris detector [9] was clearly the best of these, although the improvement compared to the Shi-Tomasi detector was not particularly great. Figure 3 shows some ROC (Receiver Operating Characteristic) performance curves for the common corner detectors, as well as a set of manually marked corner points which were used to assess the performance.

Within the corner detection report [2], a number of alternative detection algorithms were also proposed, all of which outperformed the Harris detector in some circumstances. Again the amount of improvement was not great (removing at most about 20 percent more false corners), and came at the expense of greater computational cost. The second conference paper [4] also suggests an improvement to the Harris detector by modifying the shape of the smoothing function which is convolved with the image. This modification was accomplished using a genetic algorithm to optimise performance, and resulted in small

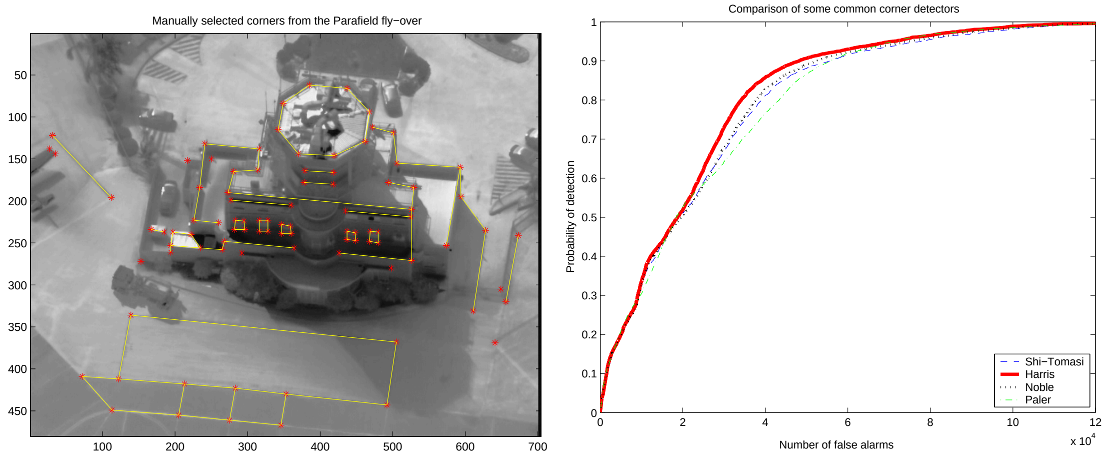

Figure 3: *An image of Parafield airport with corners marked, and some ROC curves showing the performance of some commonly used detectors*

to modest gains depending on the particular performance metric used. The link between the detection of corner points and tracking algorithms was also examined; some new and existing detectors were derived under the assumption that corners are in general those points in an image that are easiest to track.

Although the report on corner detection [2] described some interesting results, none of the tested detectors performed unambiguously better than the default detector. A later report (DSTO-TR-2095) [6] used an identical metric to measure the performance of two previously untested detectors: a phase congruency detector, and the FAST detector [10]. The FAST detector was found to detect corners better than Harris, and is much faster making it a suitable replacement for the Shi-Tomasi detector in the KLT stage.

After the detection of corners, to obtain consistent points in an image it is required to track them between frames. The tracking component of the flowchart in Figure 2 consists of two steps. The first is a position estimator, which estimates a region in which a corner from one image is likely to appear in the second. In video sequences there is generally not a lot of movement between frames. For this reason, the KLT tracker explicitly assumes that the corner will appear in the second frame within a small fixed neighbourhood of its position in the first frame. For fast motion or low frame rates, more complicated estimation techniques (such as a Kalman or Probabilistic Multi-Hypothesis Testing (PMHT) tracker) may be used, but these are unlikely to be needed for the current application.

Having guessed a rough position for the corner in the next frame, a more exact correspondence is then sought based on the image intensity values. The KLT method solves for translation between successive frames (or optical flow) by minimising the dissimilarity between small image windows around the detections in the two images. If the two image frames are given by I_1 and I_2, then the dissimilarity measure is

$$D(\tau) = \sum_{x=-N}^{N} \sum_{y=-N}^{N} w_{xy}(I_1(x,y) - I_2(x+\tau_1, y+\tau_2))^2. \tag{2}$$

A fast iterative algorithm is available to minimise this, which under the assumption that I_2 is an exact translation of I_1, involves solving the matrix equation

$$\left(\int \nabla I(\mathbf{x}) \nabla^T I(\mathbf{x}) w(\mathbf{x}) dx \right) \mathbf{d} = \mathbf{e}$$

where $I(\mathbf{x})$ is the image, ∇ is the gradient operator, defined as a column vector, $w(x)$ is some local weighting function (frequently a Gaussian, or a constant over some rectangular window), $\mathbf{d}$ is the displacement between the images, and $\mathbf{e}$ is a measure of the dissimilarity of the images. The image displacement can be found most accurately when the matrix on the left has large eigenvalues, which do not differ in scale too much. As a result, if λ_1 and λ_2 are the two eigenvalues, the best points for tracking have the largest values of

$$\min(\lambda_1, \lambda_2).$$

This method for choosing the points defines the Shi-Tomasi detector, and is the default corner detector for the tracker because, in some sense, it detects the points which are easiest to track, which should lead to fewer incorrect tracks. Other detectors (such as FAST) may be used here instead.

After the points have been detected, and the KLT tracker has estimated where they have moved to in the next frame, there will often be some mismatches. For instance, a point might move outside of the image range, or be obscured by other parts of the scene, in which case the tracker will determine the next best alternative. A consistency check then follows, which has two mechanisms for detecting a bad match. Firstly, the dissimilarity given by equation (2) will be relatively large for the poor matches, so good matches are required to have dissimilarities below some threshold. Secondly, since it is assumed that the frame rate of the video and motion of the platform are such that there is only a minimal change in camera position between frames, then the relationship between the positions of the points in the two frames should be approximately affine. Any points with high dissimilarity, or inconsistent with an affine model are then marked as bad, and are no longer tracked. If too many points are lost, then new tracks are initiated by applying the Shi-Tomasi detector to the new frame.

To date, detection and tracking algorithms have only been applied to sequences of a few thousand frames, where the building of interest is always in view. Acceptable results seem to be achieved using the proposed method, which has been implemented within the ADSS (Analysts' Detection Support System) infrastructure. In operational settings however, there are a number of complications. For instance, in very long sequences, the appearance of the corners in the images will change, which will tend to increase the drift in the tracked points. Furthermore, to increase the area surveyed, the sensor is unlikely to dwell on a target for an extended period, which will prevent the corners from being tracked successfully. The effect of these problems is not entirely clear, and would need to be assessed more clearly before operational deployment. It is expected that a user interface will be required to solicit user input on the subset of a video sequence to be processed as the simplest solution. However, as the available computing hardware increases in capability, the video sequence could be segmented regionally, on the basis of the metadata-derived footprint, and the video segments processed automatically.

3 Sparse 3D models

The proposed second step in automatic geolocation was referred to in the overview block diagram of Figure 1 as factorisation. This stage takes the tracked corner measurements from the previous step, and determines a set of 3D corner points and a series of camera positions which could produce these measurements under the assumption of a particular camera model. Due to limitations inherent in the measurement data from airborne imagery, there will usually be a number of parameters of the model which cannot be determined. This will include ambiguities in overall scale, translation and rotation of the model, difficulty in distinguishing between a model pointing towards or away from the camera, and a strong sensitivity to noise of the scaling of the model in depth. Due to these limitations, the resulting sparse 3D model is referred to as a relative 3D point model. Extra information in the form of metadata from the imaging platform is required to overcome these limitations, and this is handled by the subsequent geo-coding stage, described in Section 4.

The determination of a set of points in 3D and camera poses which are consistent with a set of tracked corners requires a model of the process by which a scene is captured as an image by the sensor. Many such models exist, some of which have been listed here from least to most complicated:

- **Orthographic:** The points are projected orthogonally onto a plane parallel to the image plane. The result is then scaled by an unknown constant, $\frac{\text{target distance}}{\text{focal length}}$. This model is suitable for views of a target from a large, fixed distance.
- **Scaled Orthographic or Weak Perspective:** This model differs from the orthographic model in that the target distance or focal length and scale may vary and the scale is unknown and different in each frame. It is suitable for views of a target from a large and varying distance.
- **Paraperspective:** The points are projected parallel to a line from camera to target centre onto a nearby plane parallel to the image plane, then scaled as in the Scaled Orthographic case. The model is suitable when the target distance varies and parallax changes are significant but perspective distortion is not.
- **Perspective:** The points are projected towards the camera centre onto the image plane, then scaled by a constant. This model is suitable for close targets but may lead to numerical difficulties for distant ones.

For airborne video imagery, the scaled perspective model is judged most suitable, because target distances clearly vary but fields of view are narrow and no convergence of parallel lines is evident. For wider views of target surroundings, the perspective model may be more appropriate. A more detailed discussion of camera models is available [18].

Having decided upon an appropriate camera model, Figure 4 now gives an overview of the process used to produce a sparse relative 3D point model and a set of relative camera poses from the 2D corner point measurements. There exists a very fast method for solving this problem, referred to as the factorisation method, which is based on the singular value decomposition. This is described in Subsection 3.1. In practice, this cannot be used directly to all of the data because it makes the unrealistic assumption that all of the features have been successfully tracked to all of the image frames. It can, however, be used to give an initial estimate which is obtained by finding a subset of features and a subset of image frames for which this assumption is true. It is beneficial if this core set of measurements is made as large as possible. A heuristic method for estimating the best subsets uses a greedy algorithm, which adds a frame or a feature so that the total number of measurements in this core matrix is increased by the largest amount. This is repeated until no further additions can increase the number of measurements.

The initialisation with factorisation will, in general, use only a small fraction of the total data available. As later steps in the processing chain require accurate camera pose estimates for widely separated frames, further processing is needed using more of the available data. A number of different methods for incorporating this data into the estimation have been considered. One commonly used technique is “hallucination” which estimates the missing corner measurements so that factorisation can be used on all of the

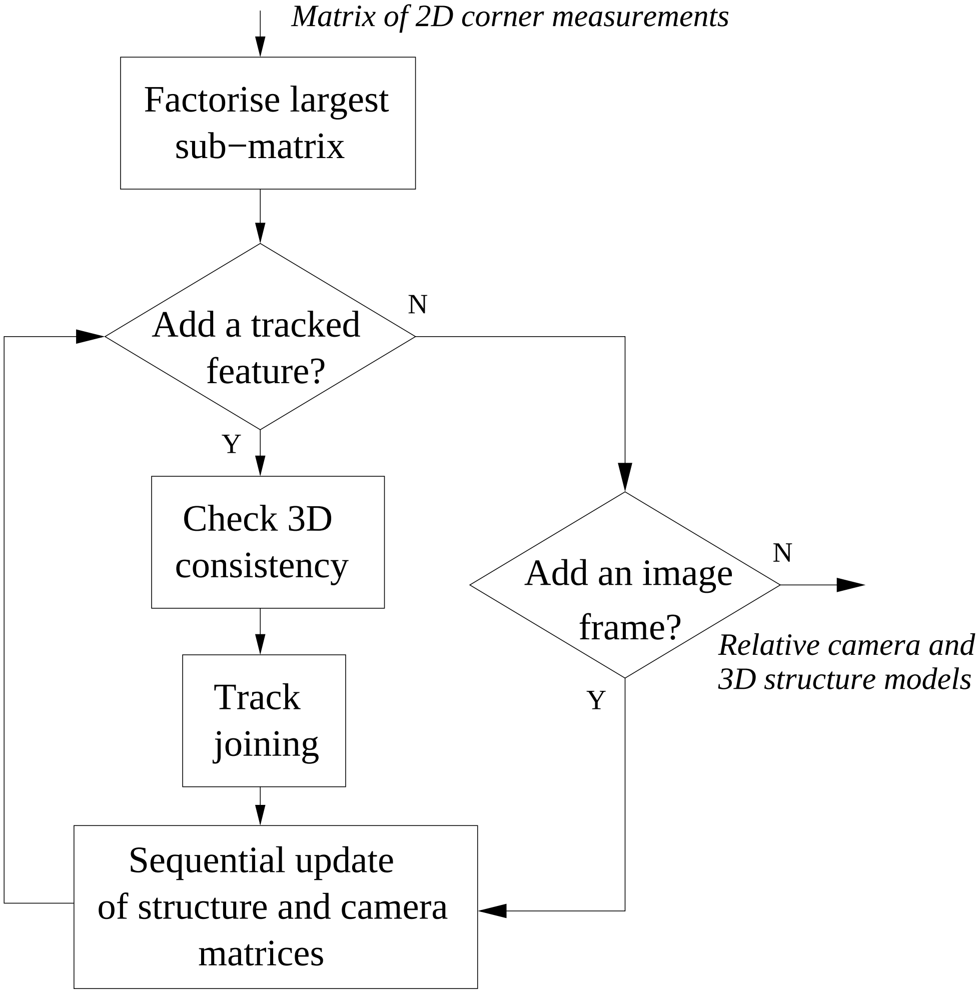
Matrix of 2D corner measurements
Factorise largest sub−matrix
Add a tracked feature?
N
Y
Check 3D consistency
Track joining
Sequential update of structure and camera matrices
Add an image frame?
N
Y
Relative camera and 3D structure models

Figure 4: *A flow diagram of Stage 2: Extracting a relative 3D point model from corner measurements*

data simultaneously. An overview of some hallucination methods is given in two reports (DSTO-TR-2101 and DSTO-TR-2095) [18], [6]. Although these techniques did not always work well, one of the methods which was based on extending the core part of the measurement matrix a corner or image frame at a time, was modified to give the proposed algorithm.

Each time through the loop in Figure 4, a set of measurements of a particular corner is eligible for addition if it was tracked for a large fraction of the currently selected subset of image frames. A check is then made to determine whether the track is consistent with being a fixed point in the scene. This is done by using the current camera pose estimates to estimate the 3D position of the new point. Then parts of the sequence where the corner measurements differ from their expected positions may be discarded. The expected positions, in frames where no measurement information is available, can also be used to determine whether it is appropriate to join tracks. More information on the consistency check and track joining is given in Subsection 3.2.

Having determined that a corner track is to be added, the final step is to update the camera poses and structure information on the basis of this new information. As only a relatively small amount of information is added each time a track is added, the resulting best solution (in a least squares sense) will be fairly close to the best solution without the new data. In this case, rather than solving the entire problem again using factorisation step, it is acceptable to improve the solution using a sequential update formula. Two methods for doing this are described in Subsection 3.3. The first method has been proven effective on several airborne sequences, where there is not a lot of missing data between frames. It is basically a single step in a steepest descent method for minimising the least square error between the measured camera positions and those estimated from the model. The second method is based on work by Shum et al. [13] and was found to be the most effective on another data set where occlusions were more significant, so may prove even more robust to missing data.

Following the addition of as many corner tracks and images as is sensible, the resulting relative 3D point model and camera pose information will be based on a large fraction of the data. A lot of the inconsistent data will have been thrown out, and the resulting estimates should be quite robust.

3.1 The factorisation method

Tomasi and Kanade [14] first noted that the problem of interpreting sets of feature coordinates in frames as different projections of coordinates in space can be treated as a matrix factorisation problem. This is done as follows.

Firstly, choose a weighted mean of the feature coordinates (preferably their centroid) as a reference point and subtract it from each point. Repeat this for each frame and consider it done for the (yet unknown) spatial positions, assuming for the moment that all features are found in all frames.

Let the orientation of the i^{th} image plane be set by specifying vectors

$$e_{i1} = (e_{i11}, e_{i12}, e_{i13})$$

$$e_{i2} = (e_{i21}, e_{i22}, e_{i23})$$

of length equal to the scaling factor S_i. Let the j^{th} feature point have relative target coordinates (x_j, y_j, z_j) and coordinates (u_{ij}, v_{ij}) in the i^{th} image. The scaled orthographic projection gives the two inner products

$$u_{ij} = e_{i11}x_j + e_{i12}y_j + e_{i13}z_j$$

$$v_{ij} = e_{i21}x_j + e_{i22}y_j + e_{i23}z_j$$

and the full set of projections can then be written in the form

$$\begin{bmatrix} u_{11} & u_{12} & u_{13} & u_{14} \\ u_{21} & u_{22} & u_{23} & u_{24} \\ u_{31} & u_{32} & u_{33} & u_{34} \\ v_{11} & v_{12} & v_{13} & v_{14} \\ v_{21} & v_{22} & v_{23} & v_{24} \\ v_{31} & v_{32} & v_{33} & v_{34} \end{bmatrix} = \begin{bmatrix} e_{111} & e_{112} & e_{113} \\ e_{211} & e_{212} & e_{213} \\ e_{311} & e_{312} & e_{313} \\ e_{121} & e_{122} & e_{123} \\ e_{221} & e_{222} & e_{223} \\ e_{321} & e_{322} & e_{323} \end{bmatrix} \begin{bmatrix} x_1 & x_2 & x_3 & x_4 \\ y_1 & y_2 & y_3 & y_4 \\ z_1 & z_2 & z_3 & z_4 \end{bmatrix}$$

(illustrated for the 3-image, 4-feature case). This may be written as $\mathbf{O} = \mathbf{PQ}$, where the observation matrix $\mathbf{O}$ is given, and assumed to be complete, and factors $\mathbf{P}$, $\mathbf{Q}$ are required.

Since $\mathbf{P}$ has three columns and $\mathbf{Q}$ has three rows, $\mathbf{O}$ must be of rank three or less (when errors of measurement are absent) and the singular-value decomposition of $\mathbf{O}$ can be used to find well-fitted factors $\mathbf{P}'$ and $\mathbf{Q}'$ of the right dimensions. The most general

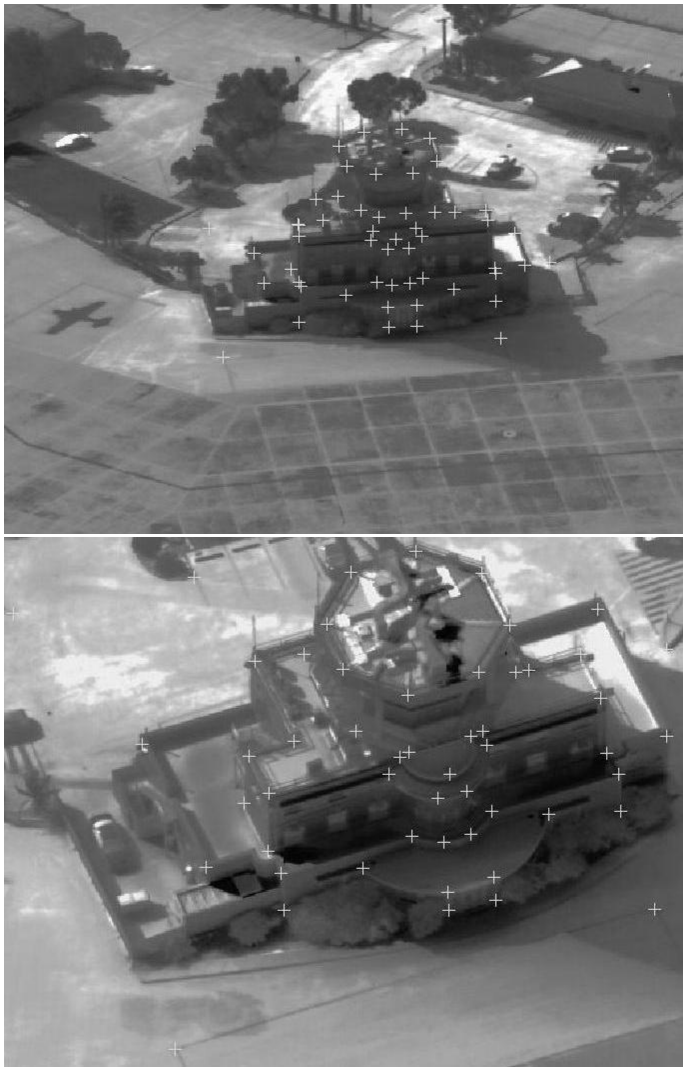

Figure 5: *First and last of 14 frames of an airport control tower.*

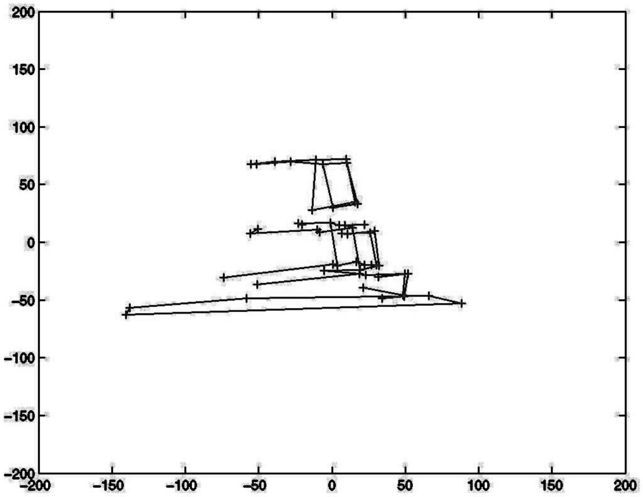

Figure 6: *Reconstruction of the airport control tower, oriented.*

factors are then $\mathbf{P}'\mathbf{T}$ and $\mathbf{T}^{-1}\mathbf{Q}'$ where $\mathbf{T}$ is any non-singular 3×3 transformation.

In the absence of outside information, $\mathbf{T}$ cannot be fully determined. It is constrained by the requirement that u_{ij} and v_{ij} are orthogonal and of equal (but unknown) length S_i for each frame number i, and by the further condition that $S_1 = 1$ (without which the constraint equations are singular). Under these conditions, $\mathbf{T}$ is known to within an arbitrary orthogonal transformation and the relative scales are known. There remain ambiguities in absolute scale, orientation of the target and a possible reflection of the target.

There is an alternative form of the factorisation method in which the reference point location is not subtracted from the observations, but entered through the projection equations. In this form, the observation matrix is of rank four, and the transformation $\mathbf{T}$, now 4×4, has further indeterminacies to allow for an arbitrary change of origin.

As an example, a 14 frame sequence of an airport control tower at Parafield, South Australia, is now considered. Each frame has been marked with 53 hand-selected features, whose coordinates were used to construct the observation matrix $\mathbf{O}$, to which factorisation was applied. Figure 5 shows the first and last frames of the sequence. Figure 6 shows a reconstruction of the points, on the ground or the near side of the building, obtained by the rank-three factorisation method, appropriately oriented and joined by selected lines to aid visualisation.

3.2 Dealing with poor tracking data

In practice, the observations from the feature detector and tracker of Section 2 are neither always accurate nor complete. For instance, if a feature appears next to another of similar appearance, the tracker might be lured by the wrong feature. Alternatively, a feature might move out of the image frame, or become obscured by parts of the target or by other nearby objects, in which case the tracker might latch onto the next best match or track could be lost completely. Either of these cases can cause problems with accurately determining 3D structure and camera poses, and need to be corrected for.

The problem of inaccuracy in measurements has already been partially compensated for as a part of the tracker. There, it was assumed that due to the small time between frames, that a global affine transformation could be used to relate the features from one frame to the next. A more refined approach is used here to throw out feature points which do not move as if they belonged to a rigid body (*i.e.* moving objects or points which slowly lose track). Two methods have been used successfully for removing inaccurate measurements. The first is based on RANSAC, as in Section 2, where factorisation is used to estimate the camera parameters for sets of three randomly selected tracks over three non-consecutive images which span the entire image sequence. The estimated camera parameters can then be used to determine the 3D locations of each of the tracked points, as long as they appear in at least two of the three frames. The expected positions of these points may then be compared with the observed positions to find the number of consistent points. The calculated camera poses with the largest number of consistent points is then the robust estimate, and the inconsistent points may be thrown away. This method was of particular use for Section 5, which uses stereo methods to perform dense matching, and so only requires the relative camera poses in two frames.

The second algorithm removes poor data on a sequential basis, and so is the method currently recommended for use in the steps outlined in Figure 4. From the factorisation step, an initial estimate of a subset of camera poses is available, and simple linear least squares can be used to estimate the 3D position of a point given a number of observations within this subset of frames. The expected positions of this 3D point in the image subset may then be calculated and compared with the measurements. All of the inconsistent measurements are discarded, and if there are too few measurements left, the entire track is thrown away. Otherwise, these points are kept and added to the measurement or observation matrix. At this stage, the frames for which there is no measurement data for this feature are compared with all of the frames for which there is data. If there exists another feature with observations consistent with the predicted positions of this feature, then it is likely that these observations belong to the same feature, and the measurements are merged into a single track. The camera positions and structure information are then updated on the basis of the added observations, as described in the next Subsection.

3.3 Sequential update

The factorisation method from subsection 3.1 obtains estimates for the 3D positions of features, as well as camera poses, from the 2D positions of all points in all frames. In practice, a full matrix of data is only available for a small subset of tracks and frames, from which an initial set of parameter estimates can be obtained. A more accurate result can be achieved by minimising the reprojection error over additional data, but methods for doing this must be able to handle missing data. For purposes of computational efficiency, it is also useful for the method to allow a fast sequential update of the parameter estimates as additional frame or track information is made available. This subsection describes two methods for sequentially updating the 3D location and pose estimates. The first of these appears to work well on several airborne video sequences and other toy simulations, and is the method currently recommended. The second approach is based on Shum's method [13] which appears to work better for scenarios where occlusion plays a much larger role.

Both of the proposed methods for sequential update with missing data start with a large complete submatrix of observations. This submatrix may be chosen in a greedy manner by adding rows and columns which give the greatest increase in the number of measurements contained in the complete submatrix. Inaccurate data and merging broken tracks may then be accomplished using the method of Subsection 3.2, and a solution may be obtained for the filled sub-matrix using factorisation. The addition of a new track, or image frame, means that the new matrix will no longer be completely filled. However, since a good estimate of the solution is already available from factorisation, the best solution with the addition of a small amount of extra data will be very close to the original solution.

The first method for dealing with the missing data assumes that a single iteration of a gradient descent update formula gives sufficient improvement for accurate parameter estimation. The squared error between the observations and the measurements will be given by

$$Total\ error = \sum_i \sum_j w_{ij} \left(O_{ij} - \sum_k P_{ik} Q_{kj} - t_i \right) = \sum_i \sum_j E_{ij}^2,$$

where w_{ij} is one when a particular feature is observed in a given frame, and zero otherwise. E_{ij} is therefore a matrix of errors between the model and the observations, and is zero when there is no observation available for comparison. Minimising this total error with respect to the unknown quantities, using a steepest descent method, will then lead to the update formulae

$$\mathbf{P}' = \mathbf{P} + \lambda \mathbf{E} \mathbf{Q}^T, \qquad \mathbf{Q}' = \mathbf{Q} + \lambda \mathbf{P}^T \mathbf{E}, \qquad t_i' = t_i + \lambda \sum_j E_{ij},$$

where the step size λ may be chosen to satisfy the Wolfe criteria to guarantee convergence. Applying the above update formulae each time a new feature or a new image frame is considered will result in an estimate of the parameters which depends on most of the available data.

The second method for sequential update of missing data is based on Shum's method [13], which is an iterative procedure which fixes the unknown matrix **P** and solves for **Q**, using a simple linear least squares. Then **Q** is fixed, and **P** evaluated using the same method, and these steps are repeated until convergence is achieved. For a sequential update, the initial solution is expected to be fairly close, and only a few iterations should be necessary to update the solution. This method was one of those evaluated on the "dinosaur" data set, which contained a relatively small number of frames with a large number of occlusions, and no point appearing in more than half of the image frames. A comparison of methods for this data was presented as part of a technical report [6] and a conference paper [5], and found that Shum's method seemed to be the best of the tested methods. Although this may be more suitable than the gradient descent based method, it has not been as well tested on real airborne video sequences.

In both of the above sequential update schemes, the camera pose matrix $\mathbf{P}$ is not constrained in any way, which means that the camera vectors are unlikely to be orthogonal, as required for the scaled orthographic camera model. The poses can be de-skewed, as described in the metadata report [18]. Also, the resulting estimates will suffer from the same limitations inherent in the factorisation process, in that from the image data, it is only possible to determine the matrices $\mathbf{P}, \mathbf{Q}$, and the vector $\mathbf{t}$, accurately up to some linear transformation, which means that the resulting models only give useful relative information. The problem described in this report demands geo-referenced information, and this can only be achieved by incorporating additional information. The next Section describes how metadata from the image stream may be used to convert the relative 3D point model into an absolute model with rotation, orientation, translation and scaling of the camera poses and 3D feature positions in a georeferenced coordinate system.

4 Use of metadata for geolocation

The previous sections have described how a set of image frames from a video sequence can produce a sparse 3D model of the scene. The suggested method involves: detecting corner points, tracking them through the image sequence to produce a set of 2D observations, $\mathbf{O}$, and then finding a set of camera poses $\mathbf{P}$ and 3D point locations $\mathbf{Q}$ and translations $\mathbf{t}$ which can approximately replicate these observations under a scaled orthographic camera model. The resulting sets of parameters $\mathbf{P}, \mathbf{Q}, \mathbf{t}$ cannot be determined unambiguously. This section briefly describes a method for resolving ambiguities, and refining the parameter estimates, taking into account extra information that might be associated with the image stream. A more in depth discussion of this work will be available in a DSTO report (DSTO-TR-2101) [18].

Some airborne sensors are able to record information about themselves during imaging. Typically, the information is not available for every frame, and is not accurate enough for any two frames to allow a straightforward stereo reconstruction of the target. A method such as factorisation that uses all the available frames is still needed to perform the reconstruction, but the metadata can help to locate the target on a map and orient it to local coordinate axes (east, north and vertical).

The metadata is used to define the camera coordinate system (horizontal, vertical, distance) to the local coordinate system at the target, for at least some of the frames. The following information is needed to do this:

- The position of the camera (latitude, longitude and altitude of the platform).
- The orientation of the camera. (This could be the orientation of the platform given as yaw, pitch and roll, and the orientation of the camera relative to the platform given as pan, tilt and swing. The sensor system may have already done the calculations needed to combine these into overall yaw, pitch and roll values.)

- An extra quantity to define the range of the target. (This could be the range from a laser rangefinder, the elevation of the ground if it is level around the target, or digital map data if the terrain is more complex.)
- The focal length of the camera. (This may vary, either by lens adjustments or by switches from one camera to another. If not, it should be known as a sensor property and not be needed in the metadata.)
- The pixel sizes, horizontal and vertical. (These may be the same. They are a sensor property and are not needed in the metadata.)

The metadata is used by recognising that if the camera-to-local coordinate transformation is defined for frame i, so are the basis vectors e_{i1} and e_{i2} in the projection equations in matrix form. Let these vectors be assembled into a matrix P_M, like P, in which rows relating to frames with no metadata are indeterminate. We may now write

$$\tilde{P'}T = \tilde{P}_M$$

where the tilde indicates the omission of rows for frames with no metadata. This equation can be satisfied in the least-squares sense by taking

$$T = \left(\tilde{P'}^T \tilde{P'}\right)^{-1} \tilde{P'}^T \tilde{P}_M$$

so T is fully determined (except in degenerate cases) as far as feature location and metadata availability and accuracy allow. The resulting feature coordinates in $T^{-1}Q$ will be in the local coordinate system, defining the target orientation. The target location has already been determined from the metadata and cannot be improved unless better known objects are available nearby for comparison. The target size is so far based on relative scaling factors (with $S_1 = 1$), but can be corrected to match $\frac{\text{target distance}}{\text{focal length}}$ for one or more frames.

There is an alternative method for determining T. The factorisation method must find tentative values of P and Q while determining the scales S_i. Call them P'' and Q''. The correct values are now only a rotation and a possible reflection away, say by T''. Then take

$$T'' = \left(\tilde{P''}^T \tilde{P''}\right)^{-1} \tilde{P''}^T \tilde{P}_M$$

and take $P = P''T''$ and $Q = T''^{-1}Q''$. Whether this method is better than the direct method is still a matter for investigation.

5 Dense matching

From the previous steps, a set of points in the scene have been detected and tracked throughout the video sequence. A factorisation approach has then been used to produce a 3D model of the scene at these points, although the resulting model still has an arbitrary rotation and scaling. Metadata is then used to resolve these model ambiguities to produce a geolocated 3D point model. In order to reproject the scene to a different viewpoint, it is necessary to determine the 3D geometry of the entire scene in a procedure called dense matching.

Figure 7 shows a flowchart of the process of constructing a dense 3D model from two widely separated frames within the video sequence. From the knowledge of the camera poses for each of the two images, as obtained from Section 4, epipolar lines may be calculated. An epipolar line shows the vector normal to one image as it appears in the other image. This means that a point in one of the images must, if it appears in the other image, lie on the corresponding epipolar line, with the distance it moves along the line being related to the depth of that point in the scene. The first stage in the flowchart is

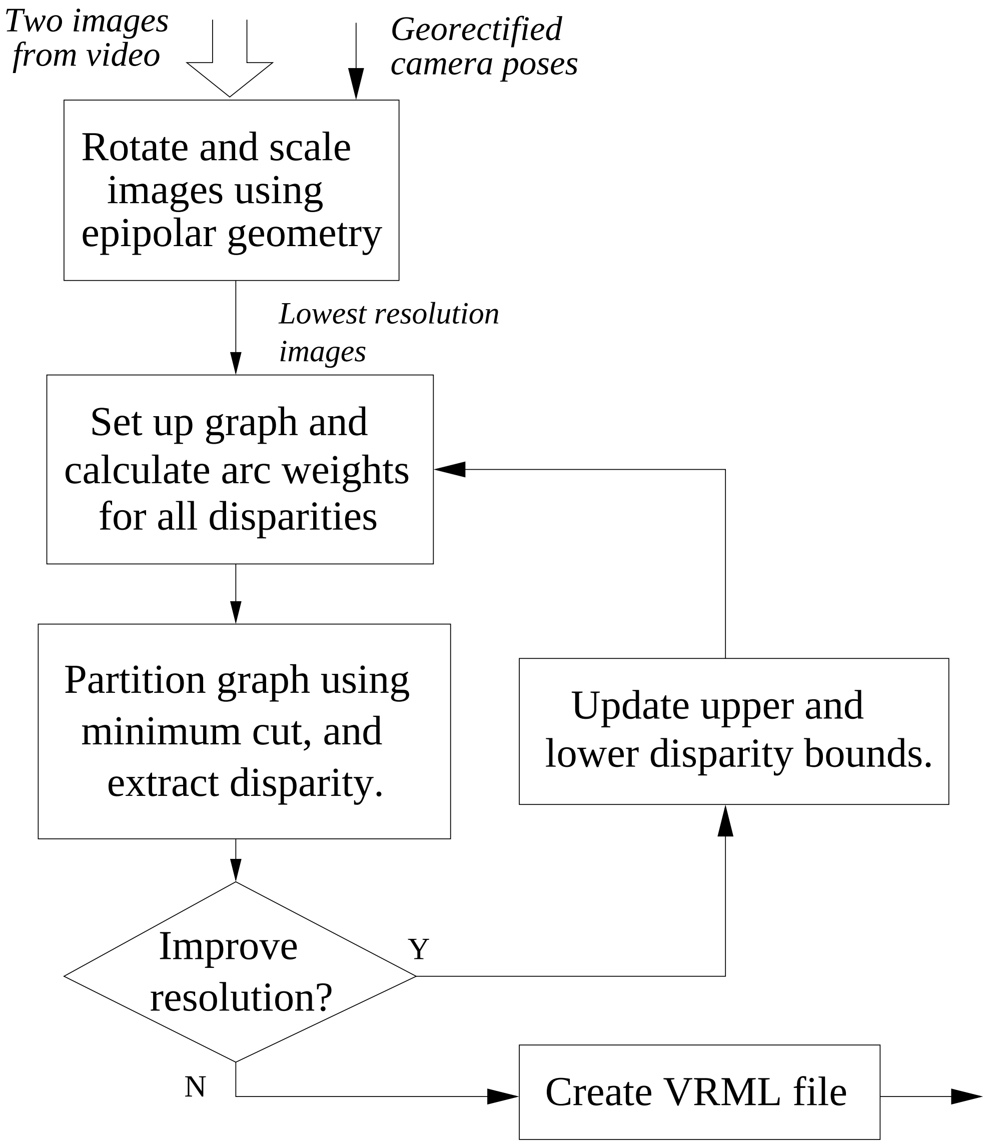
Two images from video
Georectified camera poses
Rotate and scale images using epipolar geometry
Lowest resolution images
Set up graph and calculate arc weights for all disparities
Partition graph using minimum cut, and extract disparity.
Update upper and lower disparity bounds.
Improve resolution?
Y
N
Create VRML file

Figure 7: *A flow diagram of Stage 4: Construction of a dense 3D model*

to rotate and scale each of the images so that the epipolar lines are vertical and equally spaced in both images. This means that in a rigid scene, the optical flow from one rotated image to the next is constrained to be vertical. The apparent motion of each point is by an amount called the disparity, from which the depth of each point in the scene may be found. Figure 8 shows two frames from either end of a short video sequence of the Parafield control tower, which have been appropriately rotated and scaled.

The main component in the construction of a dense 3D model is the estimation of the disparity at each point in the image. Two DSTO reports describe different methods for accomplishing this task. The first report (DSTO-TR-2095) [6] contains a description of some original research on sparse methods, as well as a number of dense matching methods. In this context, sparse methods are those which find the disparity at a small number of points within the image and interpolate for the remaining points. Dense matching methods which explicitly estimate the disparity at each point, usually with some added terms to enforce smoothness, monotonicity or some other desirable feature of the solution.

The second report (DSTO-TR-2064) [7] evaluates and compares a number of algorithms

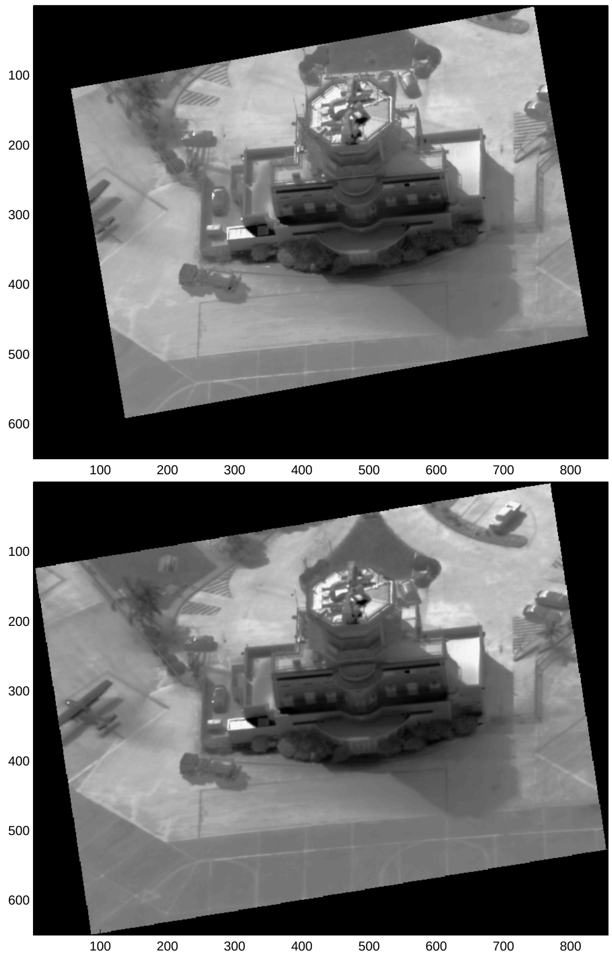

Figure 8: *Images from the Parafield airport sequence, rotated so that the epipolar lines are vertical*

from the literature for dense matching. It also examines the effect of cost functions, which are terms used by the dense matching methods to measure the similarity of the intensity distributions between neighbourhoods in two different images. Two new variations of these cost functions were also examined in more depth in a conference paper [8].

Of the various methods considered in the above reports, the particular method recommended here is a dense matching scheme, based on graph-cuts. This method relates the minimisation of a 2D matching cost function to the problem of finding the maximum flow through an undirected cyclic graph. To understand this fully, an explanation of the relationship between the maximum flow and the minimum cut of a graph is required, as described in the next paragraph.

Figure 9 shows an example of an undirected graph. The graph consists of nodes, represented by circles, and arcs or links which connect two circles together. Each of the links has a number associated with it called the capacity, which defines the maximum amount of "flow" which that link can carry. The graph also contains two special nodes, s the source node, and f the sink node. A commonly considered problem in graph theory is to find the maximum flow that the graph can support from the source node to the sink node. A simple way of solving such a problem is to find a path from s to f which is not at capacity, which is referred to as an augmenting path, and is indicated in bold in the example figures. The flow along the augmenting path is increased until one of the links reaches capacity, which is indicated as a dashed line. Then a new augmenting path is found and the process is repeated until no more flow can be added, as shown in the last graph of the example. The resulting graph has partitioned the nodes into two sets: those connected by links with spare capacity to the source, and those connected to the sink. This partition can be defined by a cut through the graph, and it turns out that if the cost of a cut is defined to be the sum of the capacities of the links that it cuts, then this particular cut has the minimum possible cost for this particular graph. A fast method for solving the maximum flow algorithm is described by Boykov, Veksler and Zabih [1] and their code, made freely available for research purposes, has been modified for testing the graph-cuts algorithm.

Returning to disparity estimation, one naive method for determining how far a point has moved from one image to the next might involve extracting a small square of imagery about the point in the first image, and then measuring the similarity (given by a match cost based on the correlation) with similar squares of imagery from columns in the second image. The match with the smallest match cost would then be assumed to be the correct match. Such a method is equivalent to finding the minimum cut in the simple graph on the left hand side of Figure 10, where the link capacities correspond to the match costs C, l is the lower bound on the disparity for that pixel, and u is the corresponding upper bound. The position of the cut along the graph indicates the best disparity. In fact, all of the disparities could be found simultaneously by creating a single large graph, consisting all of the graphs for the individual pixels in parallel. Each of these graphs could also assign the upper and lower disparity bounds separately for each pixel.

The problem with the above naive method is that due to image noise, there are usually many feasible matches of a given point along a line in the other image, and so the resulting disparity map is extremely noisy. One way to reduce this noise is to add penalty terms for discontinuities. This can be done by adding horizontal links between the graphs for the

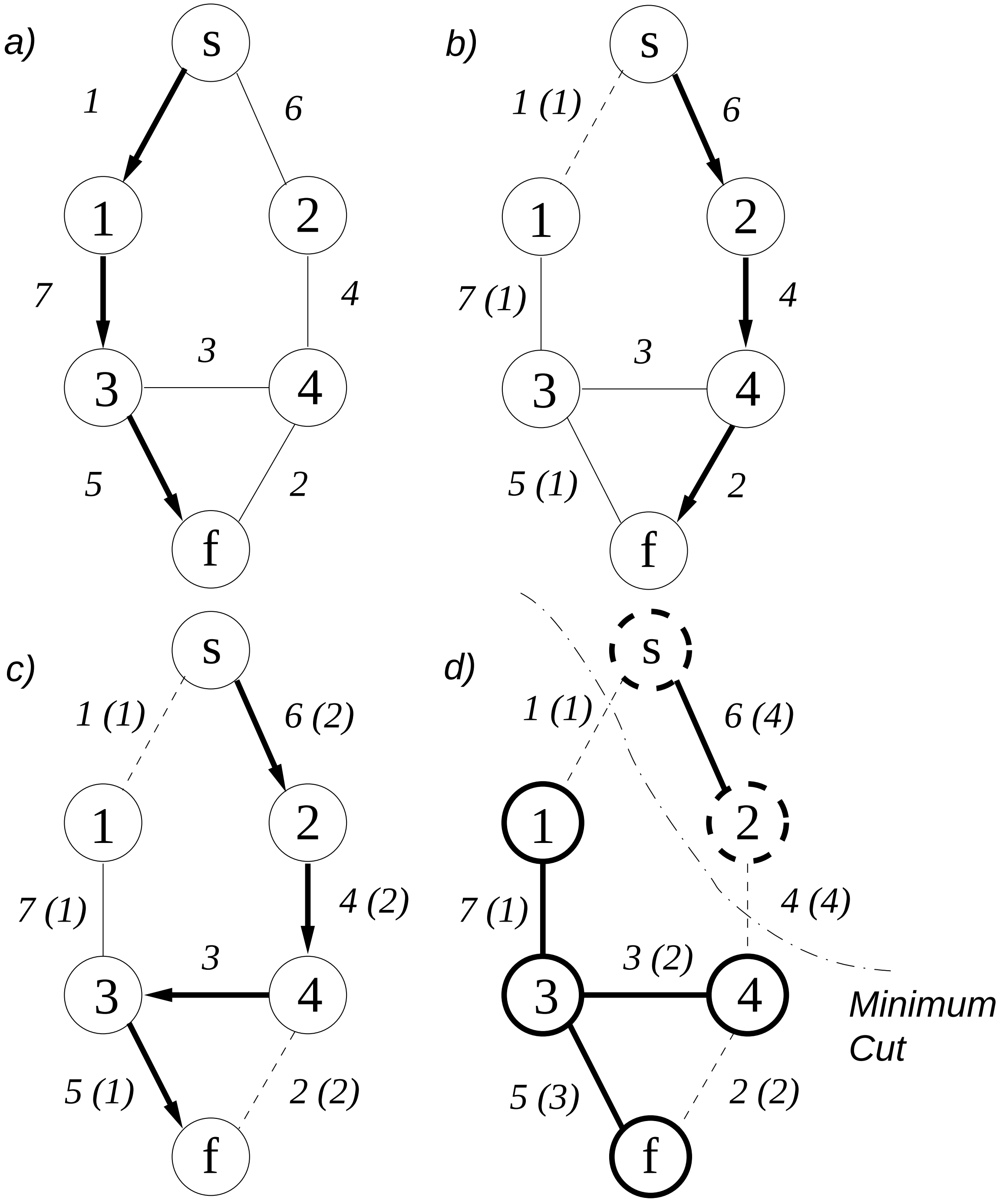

Figure 9: *Solving for the maximum flow / minimum-cut in an example undirected graph*

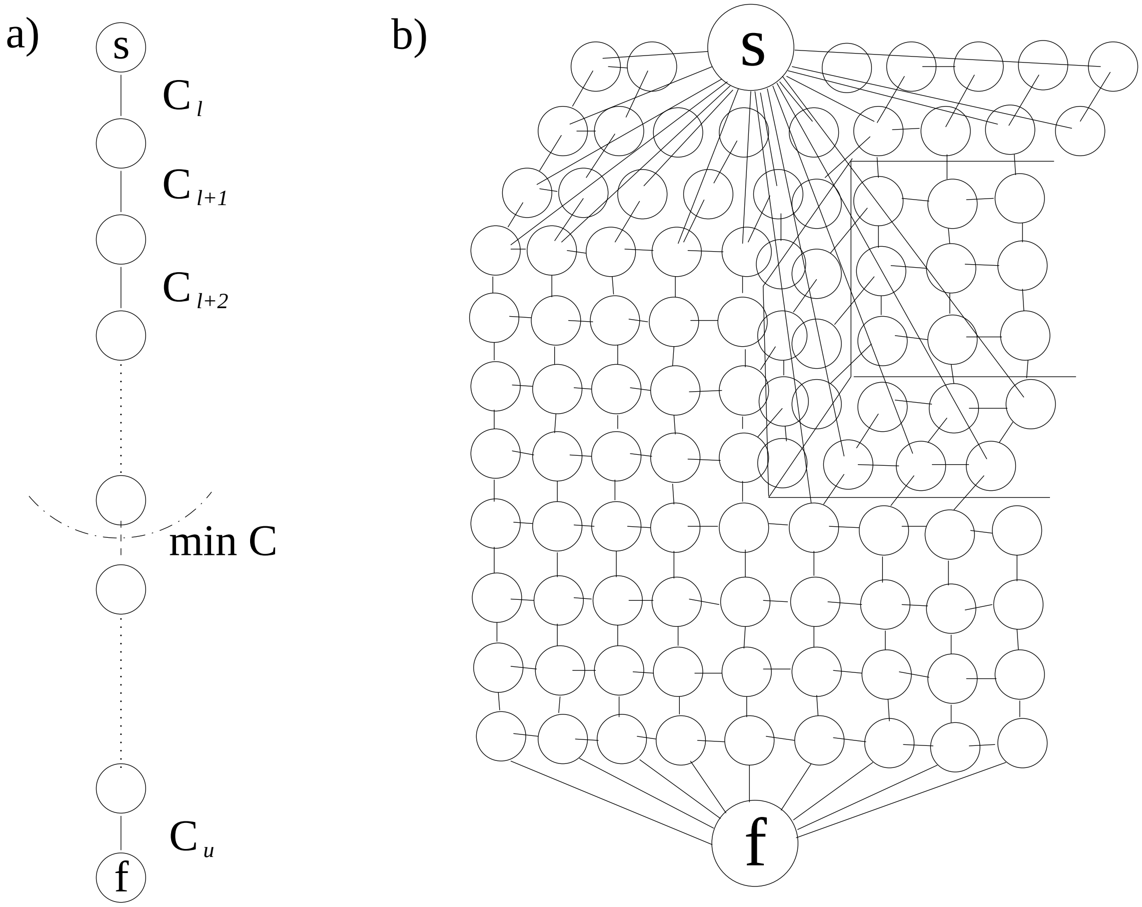

Figure 10: *Some examples of graphs for calculating the disparity between two images*

individual pixels, as shown on the right hand side of Figure 10. This works because if two neighbouring pixels have different disparities, then the minimum cut would have to cut one or more of these horizontal links, therefore increasing the cut cost. Good performance on several images has been found when the horizontal link capacities are set to be a fixed constant. It might be possible, however, to choose the capacities to be based on the edge strength, to make it more likely that a discontinuity occurs at an edge within the image. This has not been extensively tested though.

The above graph-cuts algorithm is able to be applied directly to the two rotated images at their original resolution. A difficulty is that because the disparities are completely unknown, the graph to be processed will be extremely large. For an $M \times N$ image, this means there are N possible disparities to consider at each point, so MN^2 nodes and about $5MN^2$ arcs. Fast maximum flow methods for generalised graphs would therefore require $\mathcal{O}(M^3N^6)$ operations to produce a solution, which is prohibitive even for modest size images. A way to reduce this cost is to solve the problem in a hierarchical framework, as indicated by the loop in the flowchart in Figure 7. Here, graph-cuts is applied at the worst resolution to obtain a low resolution estimate for the disparity. This means that at the next best resolution, each pixel will have a smaller range of possible disparities, and so the graph required to be processed will be much smaller. The complexity of the algorithm in this framework for an $M \times N$ image will be at worst $\mathcal{O}(M^3 N^3 \log N)$ which is much more easily achievable.

The final step is to turn the dense map of disparities into a format for reprojection. A standard file format for displaying 3D models is VRML (Virtual Reality Markup Language). Each world consists of a number of shapes, which may be collections of geometric primitives. Each shape can have its own position and orientation, and may be given its own texture, and material type (which affects how it reflects, refracts or emits light). The models generated in this section are based on only two camera views and can, in general, be adequately represented by an elevation model where each (x, y) position in the image can be associated with a single depth coordinate. In this case, the scene can be represented by a single primitive "ElevationGrid" which is defined in the following way

```
#VRML V2.0 utf8
  Shape {
    appearance Appearance {

      material Material {
        ambientIntensity 0
        diffuseColor  1.0 1.0 1.0
        emissiveColor 1.0 1.0 1.0
      }

      texture ImageTexture {
        url "im.jpg"
      }
    }

    geometry ElevationGrid {
      xDimension 163
      zDimension 214
      xSpacing   1.0
      zSpacing   1.0
      height [
               ......
              ]
    }
  }
```

The above file specifies a single shape according to the VRML 2.0 specification. The appearance has been chosen so there is no ambient light because small variations in the surface texture were found to cause large shadows giving an unpleasant crinkly effect. Instead, the shadows (at least those generated by the VRML) are eliminated by assuming that the surface of the shape is emitting white light. The shape is also textured using the file "im.jpg" which is one of the frames used to construct the 3D model. The surface itself is of type 'ElevationGrid' with the specified dimensions, and the height profile (represented by dots in the above example) will be an array of numbers describing the height at each pixel location. The resulting VRML file can be examined on any VRML2.0 compatible viewer such as "Blaxxun Contract", which will allow the scene to be tilted, panned and zoomed to achieve the required view of the model. Such a viewer will also allow the image to be reprojected to any other view, such as that seen from a set of reference imagery. Figure 11 shows two such reprojections of a dense 3D model of the Parafield control tower which was obtained using the graph-cuts algorithm. The smearing behind the tower seen in the top view is because this area is obscured in the first image. The reprojected model is used in the final step of the process, where the estimated location of the object on the earth is further refined by registering the reprojected model against the reference imagery. This is described in Section 6.

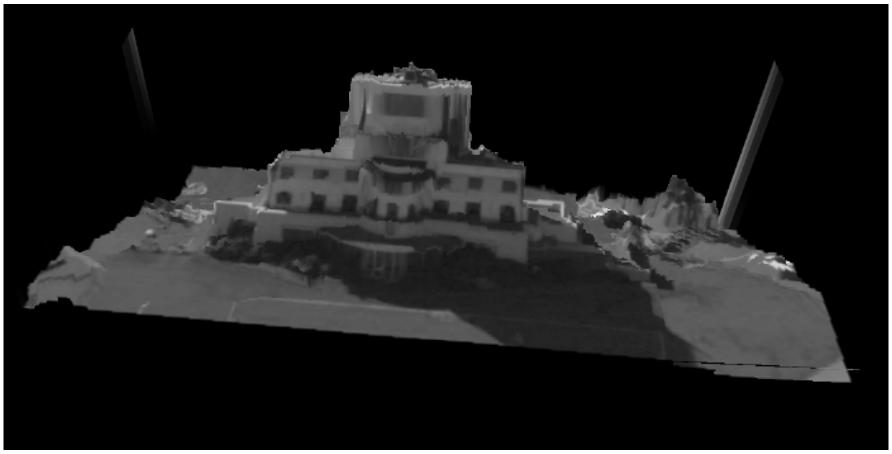

Figure 11: *A novel view of an automatically extracted VRML model of Parafield control tower.*

6 Registration

Despite the fact that metadata may be available concerning the position of the sensor and the pose of the camera, the location of an object in that imagery may still only be accurate to a few hundred metres. To obtain this position more precisely, it is necessary to register the data with higher quality information such as that available in some well georeferenced imagery. One great difficulty in directly comparing images from the two sensors is that they were taken from different looking angles, and so due to the 3D structure of the scene, the images differ by more than just a simple affine transformation. The previous four sections have described a way by which a dense 3D model of the scene may be generated, and a new image produced which appears to have been taken from the same position as the sensor from the reference model. If all of the previous steps are performed successfully, then the reprojected image from Section 5 should now differ only in a translation from the reference imagery.

The process of 2D registration is a well studied area, with many of the standard algorithms likely to produce accurate estimates for the required 2D translation. A proper evaluation of registration techniques for this application would likely require better integration of all of the previous steps, which are currently implemented to be only semi-automatic. This section does describe a method which has been used to successfully register a 3D model, produced from a video sequence, to a separate reference image. This method has not been tested on a large number of images, so while it provides a proof of concept, it can't be guaranteed to perform robustly.

There are two main categories of methods for registering two images: area based and feature based. The area based methods compare the pixel intensities over areas of the image to produce a cost function which is maximised with respect to the warping function. Feature based methods, however, extract notable features from the image to form a sparse model, and then attempt to find the correspondences between features using techniques such as RANSAC. The method demonstrated here is line feature based.

The first step in line based registration is the detection stage. In this example, the statistical hypothesis Hough transform detector, described in [6], was used. Lines were detected in both the reference and reprojected images, and then were clustered. Assuming that the two images differ mostly by a translation, corresponding lines should be roughly parallel, limiting the potential matches. RANSAC is then used to select three potential matches, from which a linear warp function is computed. If the calculated translation

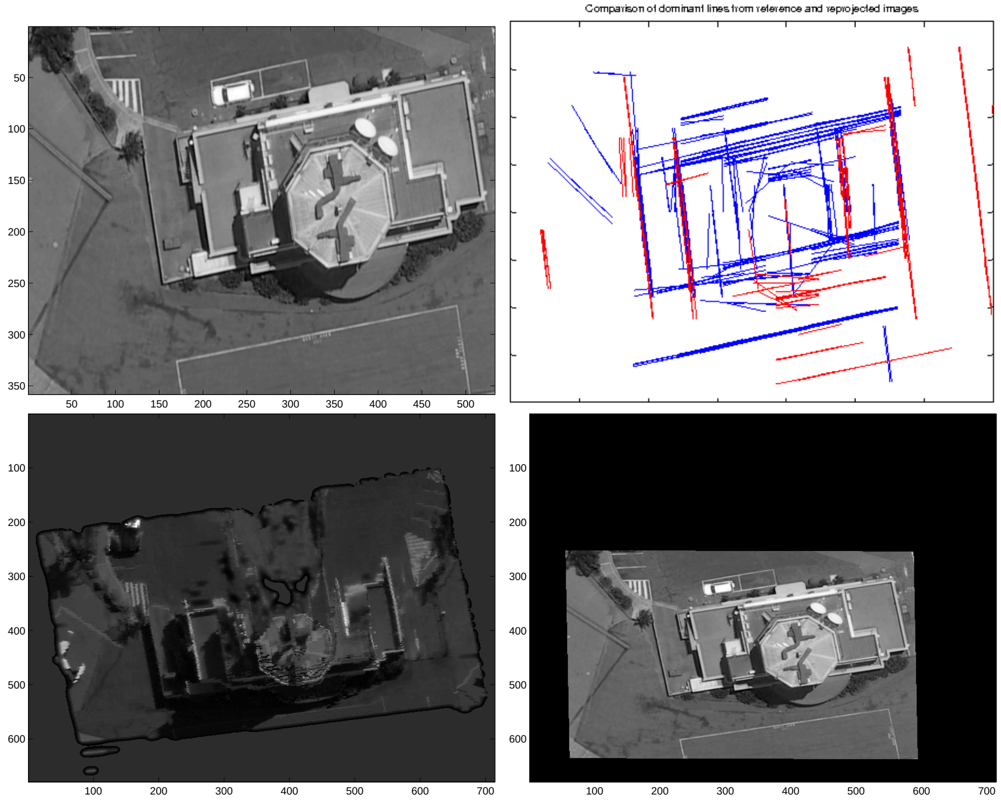

Figure 12: *Example of automatic registration between a reference image and a reprojection of a 3D model.*

has sufficiently small rotation, scaling and skew, then the transformation is plausible. A symmetric measure for the degree of overlap between the line models is then calculated. After many runs, the plausible measure with the highest overlap is considered the correct registration.

Figure 12 illustrates the registration process. The diagrams in the left column show the reference image and a reprojection of the 3D model in Figure 11 onto the horizontal plane. The line diagram shows the matching clusters of lines following the registration step, and the final figure shows the reference image in the same geo-coordinates as the reprojection. Visual observation of several points on the building indicate that, for this example, points from the original video data would be correctly geolocated to within 1 m in latitude and longitude. Height coordinates may also be estimated when the local topography is available, but it is not known for this data how accurately that could be achieved. Further work on registration in 3D (*e.g.* to existing CAD models) is to be described in a later report [20].

7 Error analysis

The previous sections describe a method by which raw video data with a metadata stream may be automatically registered to accurately geo-referenced survey imagery. This allows the geolocation of individual points within the video stream. The accuracy of this geolocation will depend on a number of factors. The quality of the video, will affect the accuracy with which consistent points within the scene can be detected, and the ease with which they are tracked between frames. The scheduling of the sensor and the airborne platform will change the effective angle of view over which the scene is imaged, which will in turn affect the accuracy with which depth can be estimated. Also, accuracy of the metadata will affect the amount of rotation, translation and scaling still present in the model reprojection, which will again affect the geolocation. This resulting accuracy will affect the ability with which the information can be exploited, and so it is important that some estimate for this accuracy be determined.

Two separate reports [19],[6] have dealt with the estimation of errors in the system described by the previous sections. The error estimation in both reports focuses on the factorisation and geolocation using Monte-Carlo-like simulations. The second report also provides a framework by which the errors may be propagated through the other steps to provide an overall system measure for the error. A brief summary of this framework was also published as part of a conference paper [5].

Figure 13 shows a flow diagram, as it appeared in one of the reports [6], of the errors in the measurements and derived quantities as they are processed. The errors in corner measurement were not explicitly modelled, but were empirically estimated from the differences between the observed corner positions, and those estimated from the relative 3D point model produced by the factorisation step. One hundred different instances of the measurement matrix were then produced, and fed into the factorisation stage to produce one hundred relative 3D point models. The variability between these models allowed estimates for the errors in camera poses and relative structure to be determined. Errors within the fully oriented model were similarly obtained by perturbation of the incoming

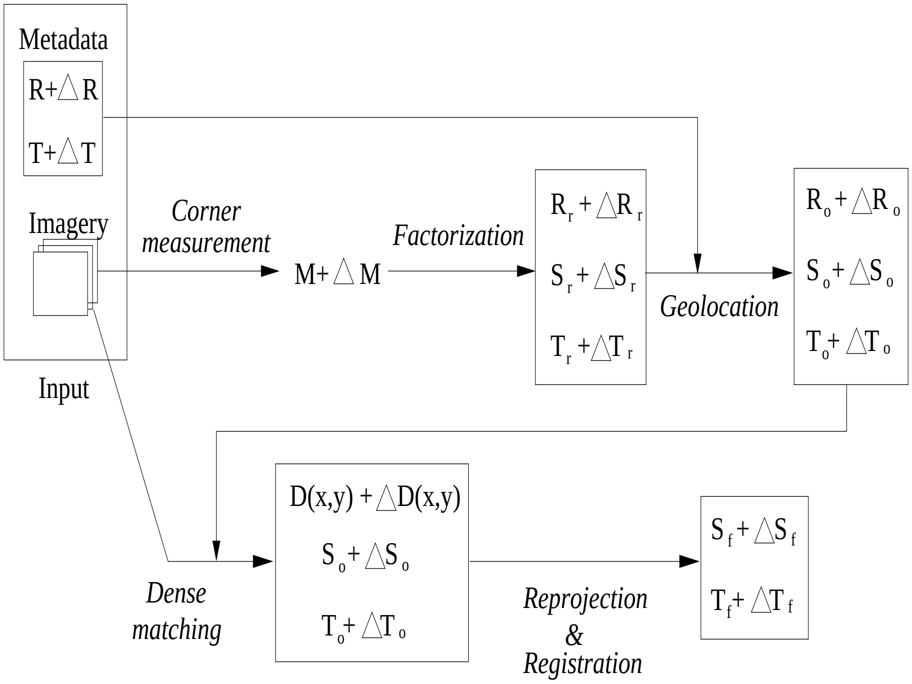

Figure 13: *Flow diagram showing the errors throughout the video registration process*

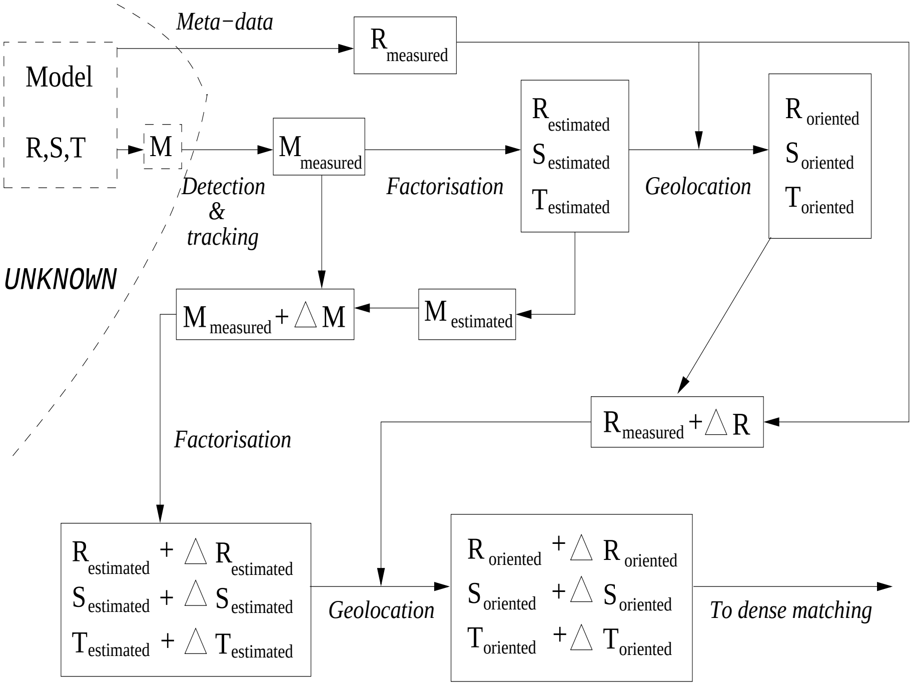
Meta−data
Model
R,S,T
M
UNKNOWN
$R_{measured}$
$M_{measured}$
Detection & tracking
Factorisation
$R_{estimated}$
$S_{estimated}$
$T_{estimated}$
Geolocation
$R_{oriented}$
$S_{oriented}$
$T_{oriented}$
$M_{measured}+\triangle M$
$M_{estimated}$
$R_{measured}+\triangle R$
Factorisation
$R_{estimated}+\triangle R_{estimated}$
$S_{estimated}+\triangle S_{estimated}$
$T_{estimated}+\triangle T_{estimated}$
Geolocation
$R_{oriented}+\triangle R_{oriented}$
$S_{oriented}+\triangle S_{oriented}$
$T_{oriented}+\triangle T_{oriented}$
To dense matching

Figure 14: *Flow diagram of the error estimation stage prior to dense matching*

metadata (based either on known sensor error characteristics, or estimated errors in zoom factor and view angle by comparison with the poses estimated from factorization). This Monte-Carlo-like simulation, illustrated in Figure 14, results in estimates for the error in 3D position for each of the tracked corner points in the video sequence. The error to this point has been shown to give close to the expected results in some numerical simulations based on tracked data points from an airborne video sequence.

Following from the geolocation is dense matching. As a result of the strong dependence on the imagery and the scene, for which there are no general models, this stage does not seem amenable to any form of error analysis. Therefore, a heuristic for incorporating the error from this stage is to extend the range of the errors in depth until it encloses the dense match surface. After reprojection of the model, and the associated errors, comes the final 2D registration. Although Section 6 described a method which has produced accurate registration on some imagery, it has not been fully tested and may need modifications, or even a completely different method, to deal automatically and robustly with different scenarios. Therefore a concrete method for dealing with errors in this module has not yet been developed. The report [6] provided some general ideas that may prove of use in the error analysis when a robust registration method is eventually selected.

8 Summary and Conclusions

This report has described a system for the accurate geolocation of points on an object in a video sequence. Each of the individual steps comprising the system have been tested on at least one fly-over sequence of Parafield control tower, which gives a proof of concept. The system as a whole has not, as yet, been comprehensively tested end to end, nor tested on a wide set of data. As such, there are likely to be many implementation issues to be addressed, and also some potential research problems which might need to be solved for certain types of imagery.

The complete algorithm consists of five main stages. The first, the detection and tracking of feature points, is mature and not expected to cause any problems in general video imagery. The next step is to determine a set of camera poses and 3D point positions which are consistent with the observed feature locations. Firstly, in situations where the object being imaged is large compared to the distance from the camera (*e.g.* terrain), the scaled orthographic camera model will not be sufficiently accurate, and full perspective must be used. This will affect practically all of the remaining steps. Also, there are potential difficulties for long video sequences, where the object of interest may not be in the field of view for the entire sequence. This may be partially handled by the track joining algorithm, but it is not certain to produce robust results.

The fourth stage creates a dense 3D model of the scene, and reprojects it so that it appears to be taken from the same angle as the sensor taking the reference image. The dense matching has been successfully tested on synthetic imagery, infrared data from the MX20 camera, and HDTV data from project Crystal View. Frequently, dense matching algorithms appear to be strongly dependent on a choice of parameters, and although this was not a problem with the tested data, it is possible that difficulties may appear when a wider range of imagery is considered. There is also a difficulty with choice of video images, since the method is based on two images chosen from the entire sequence. Currently, these are selected to be the start and end frames of the sequence, as these generally have the widest baseline, which improves accuracy. There are, however, other factors that come into play, such as the size of the object within the image and the fraction of the building that can be seen in both images. Also, due to obscuration, different parts of the building might benefit from using different frames to perform the reconstruction. The general topic of combining more than just two images also needs to be addressed. In addition, since dense matching is an area of ongoing research, it is possible that a method better than graph-cuts might be available. Scharstein and Szeliski [11] describe and evaluate a large number of these, some of which have been considered for the airborne video problem in a DSTO report (DSTO-TR-2064) [7]. The authors also update a website for the comparison of new algorithms, and it is clear that there are many possibilities in this regard that have not yet been considered for the current application.

The final stage of the process takes the reprojected imagery, which should now resemble the reference imagery, and registers the two to accurately geolocate the model extracted from the video imagery. A line-based feature matching algorithm has been shown to give good registration on some imagery, but further effort may be required to produce a more robust solution. Research into registration methods related to 3D CAD models is ongoing.

Finally, a framework for the evaluation of errors in the complete video registration system has been defined. The estimated errors to the end of the georegistration stage have been shown to be similar to the known errors in some simulations based on real data. An assessment of the final georegistration error awaits more comprehensive study of the 2D registration stage and more complete integration of the individual subsystems.

References

1. Y.Boykov, O.Veksler and R.Zabih, “Fast approximate energy minimization via graph cuts,” IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol.23, No.11, pp.1222–1239, 2001.

2. T.Cooke and R.Whatmough, “Detection and tracking of corner points for structure from motion,” Technical Report, DSTO-TR-1759, August 2005.

3. T.Cooke and R.Whatmough, “Evaluation of corner point detectors for structure from motion problems,” DICTA 2005.

4. T.Cooke and R.Whatmough, “Using learning algorithms to improve corner detection,” DICTA 2005.

5. T.Cooke, “An empirical analysis of errors in structure from motion,” DICTA 2007.

6. T.Cooke, “Automatic extraction of 3D models from an airborne video sequence,” DSTO-TR-2095, January 2008.

7. E.El-Mahassni and T.Cooke, “A survey on the suitability of some recent 3D surface reconstruction algorithms for airborne sensor imagery,” DSTO-TR–2064, October 2007.

8. E.El-Mahassni, “New robust matching cost functions for stereo vision,” DICTA 2007.

9. C.Harris and M.Stephens, “A combined corner and edge detector,” Proceedings of the 4th Alvey Vision Conference, University of Manchester, pp.147–151, 1988.

10. E.Rosten and T.Drummond, “Machine learning for high-speed corner detection,” European Conference on Computer Vision, May 2006.

11. D.Scharstein and R.Szeliski, “A taxonomy and evaluation of dense two-frame stereo correspondence algorithms,” International Journal of Computer Vision, Vol.47, pp.7–42, 2002.

12. J.Shi and C.Tomasi, “Good features to track,” Proceedings of the IEEE Conference of Computer Vision and Pattern Recognition (CVPR’94), Seattle, June 1994.

13. H.Shum, K.Ikeuchi and R.Reddy, “Principal Component Analysis with missing data and its application to polyhedral object modeling”, IEEE Transactions in Pattern Analysis and Machine Intelligence, Vol.17, No.9, pp.854–867, 1995.

14. C.Tomasi and T.Kanade, “Shape and motion without depth”, Proceedings 3rd International Conference on Computer Vision, IEEE, pp.91–95, 1990.

15. C.Tomasi and T.Kanade, “Shape and motion from image streams: A factorization method – Part 3: Detection and tracking of point features,” Carnegie Mellon University Technical Report CMU-CS-91-132, April 1991.

16. C.Tomasi and T.Kanade, “Shape and motion from image streams: A factorisation method,” International Journal of Computer Vision, Vol.9, No.2, pp.137–154, 1992.

17. R.Whatmough, “Combining shape from motion output with partial metadata,” IEEE Conference on Advanced Video and Signal-based Surveillance, Sydney, November 2006.

18. R.Whatmough, “Extracting the shape of a target from an image sequence with incomplete metadata,” DSTO-TR-2101, 2008.

19. R.Whatmough, “Error analysis of shape from motion extraction with incomplete metadata,” DSTO-TR-2102, 2008.

20. R.Whatmough, “Registration of a Shape-From-Motion reconstruction to a geolocated 3-D model,” DSTO-TR-2103, 2008.