Real-time Mosaicing from Unconstrained Video Imagery for UAV Applications

Mateusz Brzeszcz
School of Engineering, Cranfield University, UK
Automatic Control, Electronics and Computer Science Department,
Silesian University of Technology, Poland

Toby P. Breckon
School of Engineering, Cranfield University, UK

Ken Wahren
Blue Bear Systems Research, Bedford, UK

Abstract

Here we investigate the problem of real-time video mosaicing and its application to UAV video imagery. The driving demand here is to improve overall situational awareness by providing the UAV operator with a real-time video mosaic - i.e. a panoramic image constructed in real-time from the incoming video frames captured from the UAV platform in flight. Specifically, in this work we use a feature point based image alignment by combining a state of the art feature point detector and a robust statistical selection methodology (RANSAC). Image alignment is then performed by online bundle adjustment supported by hardware accelerated visualization with quality enhancements and explicit task parallelization on modern CPU hardware. The video mosaic output is constructed solely from the input video frames with no additional constraining navigation, position (eg. GPS) or platform parameters. In this way it is highly suited to situations where GPS data may be unavailable or inaccurate. Real-time performance is further supported by novel developments in the use of specific frame sieve in order to avoid high data redundancy which is associated with temporally dense input video frames having significant spatial overlap. An evaluation of our approach is presented to illustrate overall robustness in video mosaic construction under a diverse range of conditions incuding varying illumination and presence of motion within the scene.

1 Introduction

Video imagery received from the UAV platforms usually lacks the general context in which it was captured. This limited view of the environment is often referred to by UAV operators as “viewing through a drinking straw” (i.e. a very limited view of a much wider environment). In order to improve the overall situational awareness we investigate how a video mosaic can be constructed, consisting of multiple spatially aligned video frames, to provide a wider spacial view of the environment within which the current sensor view can be superimposed. Furthermore, the limited bandwidth from the UAV platforms generally imposes a resolution constraint on the received video data. As a result when a specific object of interest is present within the scene the operator frequently zooms in the sensor to obtain a more detailed view at a higher resolution. This is at the expense of losing the more general wide angle view on the environment and subsequently being able to put this detailed information within a more general environmental context. Video mosaicing has the ability to retain the wide angle view into which this higher resolution detailed information can be embedded. This visualization capability could be combined with earlier work on vehicle [1], salient object [2] and people detection [3] to give an overall environment wide context of automatic threat or object detection within UAV imagery.

A range of prior work in this topic area exists, not only dealing with video mosaicing [4, 5] but also in the very closely related problem of panoramic stitching [6, 7]. The input to such a technique is a set of overlapping images (video frames) and the goal is to align them spatially and produce the larger output panoramic image (mosaic). However, when we examine these techniques in detail they are generally not the same. In the case of panoramic stitching the input is a set of unordered, high resolution, still images that overlap slightly. In the case of video mosaicing the input video frames are temporally dense (i.e. multiple frames per second (fps)) and in terms of spatial alignment have a large spatial overlap. This is caused by the fact that the camera movement between two consecutive video frames, within the environment, is usually relatively small and constrained.

In most cases video mosaicing algorithms have a real-time requirement and the problem is therefore most generally studied with a live video source - in our case an UAV imagery is used for this purpose. By contrast the panoramic stitching problem involving still images does not have a real-time constraint and thus regular approaches to this related problem focus mainly on the quality of the outputted composite (panoramic) image rather that real-time performance, visualization and presentation.

The image alignment and stitching problem solutions are usually based on two different image registration methods: feature based, where alignment of images is based on extracted feature points [6], and direct (pixel based), where images are aligned directly, i.e. by matching the individual image pixels [4]. Although Szeliski [8] suggests that the feature based algorithms are preferable for large image separation stitching applications and notably video mosaicing is not such a large separation problem, there are no definitive reasons for discarding the usage of feature based approaches for video mosaicing. It also has distinct advantages such as significant performance gains in the case of greater camera movement, greater inter-frame separation and additionally easy recognition of breaks within the video frame sequence continuity. Moreover, from empirical results a feature based approach appears to perform well in re-registration after a brake in continuity of the incoming video sequence.

Feature based image registration has already been archived for still image panoramas but the majority of prior work concentrates on the quality of the output rather than real-time presentation [6, 7]. However, many of the ideas and concepts presented these works can be adapted towards a real-time methodology and we will perform this non real-time to real-time transition as a part of the work presented here.

In this work we present a full pipeline for real-time video mosaicing, including the realization of real-time feature point detection and description [9], through the use of constrained real-time homography and bundle adjustment [10, 11, 12] supported by real-time image presentation and visualization [13]. Specifically we have developed novel approaches to support this real-time application: separation of bundle adjustment into specialized *pairwise bundle adjustment* and *global bundle adjustment* which significantly improves the processing time, the frame sieve concept to handle the large data redundancy which is associated with temporally and spatially

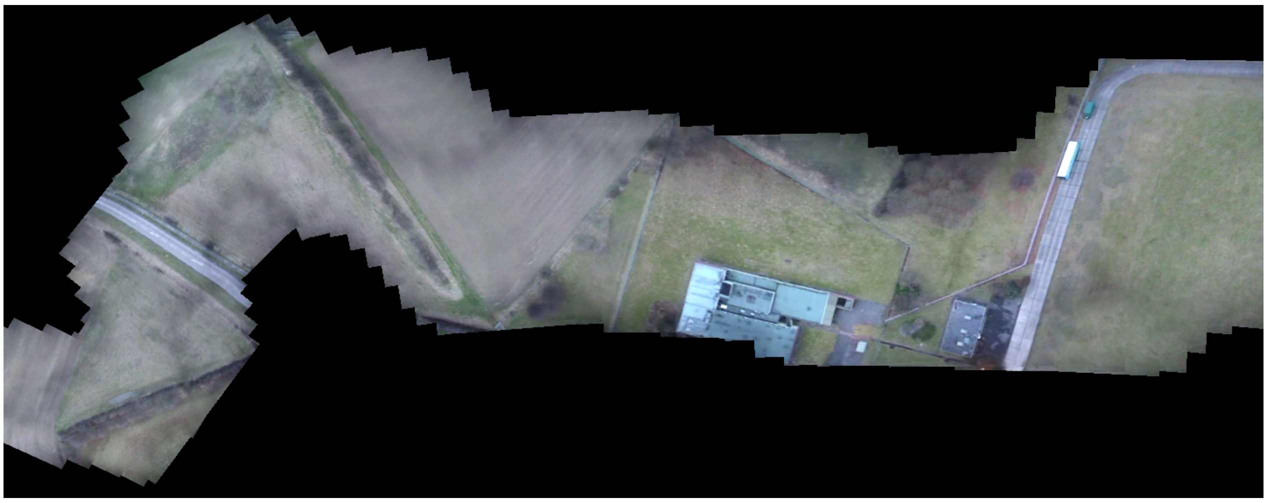


Figure 1: *An example of a mosaic constructed from a top-down UAV camera footage*

dense input video frames and hardware accelerated visualization with adapted visual enhancements. The approach is then tested on different types of UAV video imagery (top-down camera as well as angled view camera) using a ruggidized computer prepared for application in the UAV command and control. Figure 1 presents an example of a video mosaic constructed using our approach.

2 Feature Based Video Mosaic

2.1 Outline

In general the overall video mosaicing approach operates by identifying feature point matches within individual video frames (*SURF* feature points [9]). These feature points are used to derive a set of matches between consecutive frames. Afterwards the *RANSAC* [14] based methodology is applied for elimination of statistical outliers and rapid confirmation of the consistency of the detected set of matches. Based upon this set of identified match correspondences the *bundle adjustment* is used to estimate the image alignment.

We utilize a *3D visualization* [13] and a set of visual enhancements to produce the resulting video mosaic from captured frames and their estimated alignment. The *gain compensation* [6] is used to compensate for artefacts caused by the *Automatic Gain Control (AGC)* present on most modern cameras. Additionally blending is used to eliminate visible stitches between images that contribute to the final mosaic image.

Real-time performance is achieved by applying the *bundle adjustment* in *pairwise* and *global* manner. The *pairwise bundle adjustment* is responsible for relative frame to frame alignment for newly captured video frames. The *global bundle adjustment* accounts for the accumulated errors in the image registration providing the *globally optimal* image alignment. In order to manage the case where consecutive frame to frame match cannot be found we additionally implement aspects of *global search* within the panorama. The *key frame* based approach is used to handle the overlapping frames and redundant information removal.

2.2 Feature Point Detection and Matching

Feature Point Detection

The *SURF detector and descriptor* is used to extract the points of interest from the captured video frames. SURF is scale and rotation invariant feature point detector and descriptor. The scale and rotation invariance is essential, due to the fact that the input video footage can undergo scaling (i.e. zoom) as well as rotational transformations. Since the problem of feature detection and description has been widely studied previously, detailed benchmarks and evaluations are available [15, 16].

Matching

In the task of video mosaicing, as opposed to still image panoramic stitching, we can assume that the consecutive video frames overlap somewhat. There can be special cases when this assumption is broken (see Section 2.4) but this will be handled separately. This assumption simplifies the matching step, as we need to match the current video frame only with the previous one.

The SURF descriptor produces a 64 dimension feature vector and a similarity measure between such SURF feature vectors has been chosen to be measured as an Euclidean distance. The feature a from first image is considered a match to the feature b from the second image if the difference d (euclidean distance) between these features fulfils the following relationship

$$\frac{d}{d_{all}} < t \tag{1}$$

where d_{all} is the minimum difference over all possible differences between feature a and every feature from second image, excluding feature b. Finally t is a threshold value $t \in \{0...1\}$. This is the *nearest neighbour ratio matching strategy* [9, 17]. The threshold has been empirically chosen to be $t = 0.65$, with higher values found to result in a higher number of matches but with significant additional false erroneous matches.

RANSAC Sieve

Due to the presence of noise (i.e. false matches) in the extracted SURF feature matches the RANSAC sieve is used on the set of corresponding feature points to recognize the *statistical inliers*. Contrary to the common RANSAC application, the estimated statistical model is discarded and only the *statistical inliers* are passed to the next step which is the image alignment using the *bundle adjustment*. Essentially RANSAC is used to identify the *statistical outliers* and remove them from the global set of frame to frame SURF feature matches. The RANSAC procedure provides means for randomly sampling and consistency checking for the set of identified SURF feature point correspondences.

The model for RANSAC fitting has been chosen to be a *projective transform* (i.e. a *3x3 homography* matrix) that provides a projective mapping from one image to another [10]. It is obtained from four point correspondences by solving a set of linear equations using a *Direct Linear Transformation* [10]. Although the camera geometry in our case can be different, for example a non-linear constrained homography, here we use a generic linear transform as the projective model. The reason for that is the fact that the alternative computation of a more complicated model would be much more computationally demanding and would thus slow down the RANSAC procedure considerably. The implemented model has proven itself to be sufficient for the task of the inlier selection.

2.3 Frame Alignment

The frame alignment is realized by the bundle adjustment [11, 12, 10], which addresses the problem of optimizing the 3D structure of the reconstructed scene. In essence it is a large, sparse, geometric parameter estimation problem. The 2D positions on images constitute the measurement set, and the camera parameters with 3D coordinates of the feature points are the parameters being sought. The goal is to minimize the *reprojection error*, i.e. a sum of squares of euclidean distances of observed and predicted image features.

According to [12], the *Levenberg-Marquardt* algorithm has proven to be the best suited in solving this non-linear least-squares problem. It can be thought as a interpolation between the *gradient descent* and *Gauss-Newton* algorithms. Despite the high dimensionality of the problem, the lack of dependences among most of the estimated parameters (e.g. the 3D points do not influence each other) makes fast calculation possible because the structure of the problem is sparse.

Here in our implementation we do not specifically assume a representation for the geometry used in the problem and thus use a general approach that can be used in a wide range of applications. We specify the projection function f that computes the estimated measurement vector, i.e. the position of the point in the camera plane, given the camera and 3D point parameters. The projection function f is given by the homography H_i of the camera i. For the estimated point $\mathbf{u}$ in the projective space, we can calculate its i-th camera coordinates, i.e. its position on the i-th image. These coordinates in terms of projective geometry are described by vector $\mathbf{u_i}$, which can be calculated by applying the homography to the $\mathbf{u}$ point.

$$\mathbf{u_i} = H_i \mathbf{u} \tag{2}$$

In order to calculate the 2D image coordinates (x_i, y_i) of this point, we need to transform the coordinates from the projective geometry to the image coordinate frame.

$$x_i = \frac{\mathbf{u_{ix}}}{\mathbf{u_{iz}}} \qquad y_i = \frac{\mathbf{u_{iy}}}{\mathbf{u_{iz}}} \tag{3}$$

These transformations are thoroughly described in [10].

2.4 Achieving Real-time Performance

Based on the presented outline (Sections 2.1 to 2.3) we furthermore outline key novel aspects of this work which facilitate overall real-time performance of this video mosaicing approach.

Pairwise and Global Bundle Adjustment

Due to the real-time requirements of this application we cannot afford to perform the bundle adjustment of all stored video frames every time a new frame is captured from the video source. However, it is still desirable to use the bundle adjustment globally in order to reduce the accumulated error which would be present if only pairwise image alignment would be used.

The image alignment has been divided into two parts: the pairwise bundle adjustment and the global bundle adjustment. Pairwise bundle adjustments occurs every input video frame. It takes only two frames, adjusting the second to align it optimally to the first one (i.e. it does not change the parameters of the first frame). It starts from the last globally adjusted video frame, and ends on the last input frame (the frame that has been most recently captured). Since it only takes two video frames at a time, it allows for a real-time placement of new frames relative to those which are already present and globally adjusted.

The bundle adjustment methodology was originally developed as a method of computing the parameters of a large structure and usually the linear homography was used to compute the image to image correspondence. However, due to the fact that non-linear representation can be used for the camera parameters the problem of image to image alignment has become non-linear. This is the reason why the bundle adjustment is used even for a pairwise alignment in our approach.

By constrast, *global image adjustment* is performed periodically. It adjusts all the images simultaneously taking into account the overall structure of the mosaic and thus it corrects accumulated errors. After the calculation is complete all the parameters of video frames that took part in the adjustment are updated.

Despite the fact that global bundle adjustment is slow (can take seconds when tens of video frames are present) it does not interfere with the rest of the system because in terms of the implementation it can be operated in a separate, parallel thread.

Both bundle adjustment methods (pairwise and global) require an initial estimate of the camera parameters. Actually all the video frames present in the mosaic are represented by previously estimated parameters (either estimated from previous pairwise bundle adjustment alone, or as estimated from prior global bundle adjustment if they have been present within the mosaic for a significant period of time). However, when a new video frame is captured from the video source, its camera parameters are unknown. In such a case we initialize these parameters with the parameters of the frame to which the captured frame matches (i.e. after the initialization the captured frame has the same parameters as the frame matched to it). This has proven itself to be a sufficient initial estimate. Empirically initializing the input frame with the parameters taken from the RANSAC sieve (after constraining and converting the estimated homography model to camera parameters used in the bundle adjustment) have not shown any improvements and occasionally caused a major misregistration of the video frame. RANSAC is thus used solely for rapidly confirming the presence of a suitable match and for eliminating the outliers from the image to image feature matchings dataset, whereas the pairwise bundle adjustment is actually used to compute the image registration of the new frame into the existing mosaic.

Image Mismatch Case

Occasionally the matching between the last video frame captured and the currently captured video frame cannot be found. There are many different reasons for this occurrence. For example a transmission break, a temporary camera malfunction, large scale movement of the UAV platform or a non distinctive video frame lacking suitable feature points. In such a case, the algorithm simply discards the input video frame and waits for the next. However, if it happens repeatedly then the algorithm starts to search for a global match - i.e. it tries to match the input video frame with one of all the currently stored ones within the video mosaic. This is based on the assumption that during the “outage period” of the image matching the UAV will have moved position within the environment and therefore it is reasonable to assume that a match may be found against any portion of the previously captured scene image information.

If the result of this search is successful then the new frame is aligned with the frame to which it matches. From this point the normal operation continues.

Actual determination if a suitable match between two video frames exists is made on a simple threshold basis. Firstly, there must be a sufficient amount of *statistical inliers* present in the set of feature point matches after the RANSAC sieve routine has been executed. Additionally, the *inlier* to *outlier* ratio must be greater than a set threshold.

Keyframes and Frame Sieve

The input video frames can be considered to be temporally dense with most of these video frames having a significant spatial overlap which results in a high data redundancy. The idea of key frames has been adopted to provide means of classification as to which portions of the image data are to be retained and which can be discarded due to spatial duplication.

Despite the fact that only a portion of the video stream is retained and contributes to final video mosaic initially all of the input video frames are pairwise aligned and displayed for visualization. Only after the next video frame (t-th) has been captured and aligned within the mosaic the decision about the previous one (($t-1$)-th) can be made (i.e. to discard or not to discard due to spatial redundancy). This decision is made in the concept of the *frame sieve* which essentially works on the identification of the key spatial frames within the image sequence.

There are two criteria which are checked in order to classify the frame, one being the percentage of area that is not common to this given frame and the last key frame (this makes a measure of the spatial distance between these frames). If this percentage archives a certain threshold, the frame is considered to be a key frame. A smaller threshold will result in more frames being retained. The other reason for a frame being labelled as a key frame is the attainment of a set temporal distance to the last key frame measured in terms of the frame index in the sequence.[1]

The frame sieve is composed of three main steps: 1) Non key frame removal; 2) Overlapping frame removal and 3) Oldest frame removal.

All the input frames are retained and displayed, key frames and non key frames. However, the frame sieve, a routine determining which frames are to be discarded, takes into account the type of the frame (key or not key) as it goes through its first step which is the removal of all non key frames except the most recent one. This assures that the most recent frame is available and displayed in addition to those which give the significant spatial coverage of the video mosaic environment.

The second step erases all the frames that are overlapped by newer (temporally more recent) frames and thus are not visible to the user (due to the overlap in the display). Strictly speaking, this heuristic procedure is slightly more complicated. Frames are being investigated from the oldest to the newest. If a frame is partially covered by newer frames (the specific amount of coverage is specified by a threshold value), then frame is checked if it would leave gaps in the mosaic after it has been erased, i.e. if the older, non erased frames cover these gaps. When both tests are passed the frame is erased. This procedure has empirically proven itself to be reliable and suitable for application in the second step of the frame sieve.

[1]This assumes a constant video frame rate from a video source device.

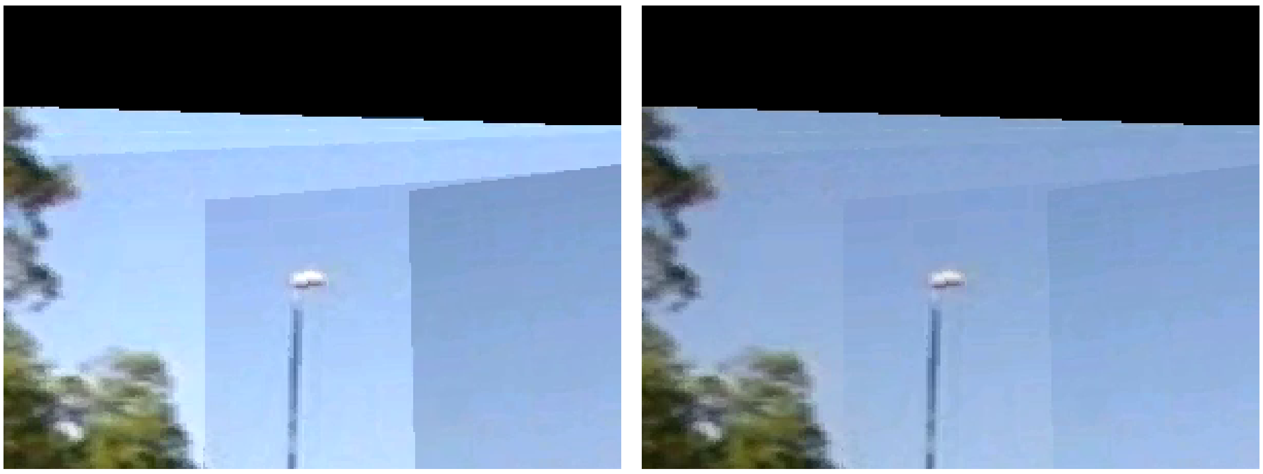

(a) Mosaic not compensated for the AGC (b) Mosaic with gain compensation

Figure 2: *The result of applying the gain compensation.*

The last and third step in the overall pipeline of frame sieve is to erase the oldest frames if the number of globally recorded frames exceeds a given threshold value.

2.5 Visualization

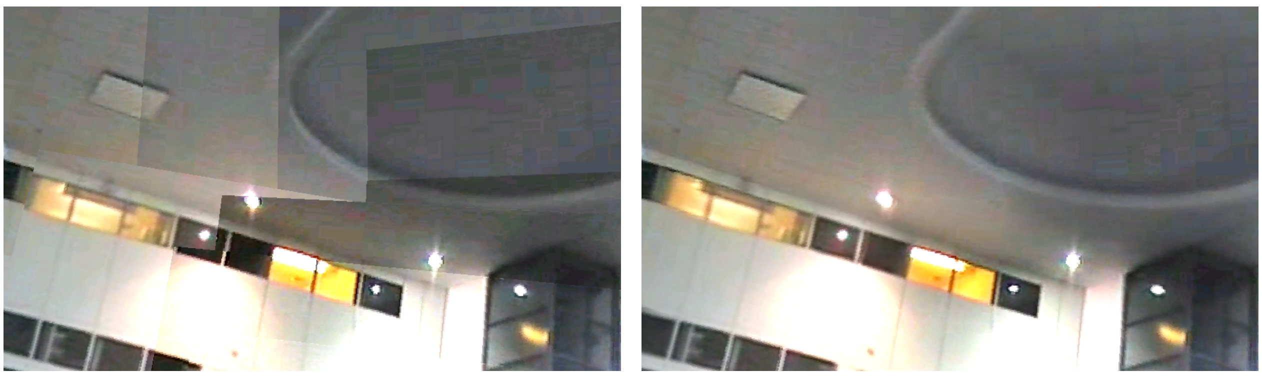
In most image stitching and video mosaicing approaches software visualization is used for output display [6, 4]. In order to meet real-time requirements we use hardware accelerated approach using OpenGL [13]. Stored video frames are represented as simple, textured 3D rectangles in a manner discriminated by the assumed geometry. Parameters stored in every video frame, estimated by the bundle adjustment, can be directly applied to the *ModelView* matrix [13] to arrange these graphics primitives appropriately. However, the major drawback of this approach is the lack of flexibility. The specific assumptions of OpenGL must be met and thus some solutions used widely in the prior visualization and rendering of stitched panoramas are not readily applicable [6]. Notably both *gain compensation* and *blending* must be handled within a real-time context compatible with this use of hardware accelerated visualization.

Gain Compensation

Most modern video cameras are equipped with *automatic gain control (AGC)*, which automatically adjusts the camera exposure to achieve automatically regulated level of image brightness and dynamic range based on the perceived light levels within the image. This feature can introduce undesirable effects in the case of video mosaicing because essentially the dynamic range of each video frame can vary depending on localised changes in lighting levels within the scene. An example of such a case is presented in Figure 2a where AGC decreased the overall camera gain during the camera movement. Correcting these differences is required to improve the overall quality of the output video mosaic and mitigate the effect of the gain compensation introduced by the AGC. An example of such correction is shown in the Figure 2b.

The method for calculating such gain compensation is detailed in [6]. The compensation works in terms of minimizing the error function - intensity differences between overlapping regions of the mosaic. The error function is defined as: -

$$e = \sum_{i=1}^{n}\sum_{j=1}^{n} N_{ij}((g_i I_{ij} - g_j I_{ji})^2 \frac{1}{\sigma_N^2} + (1 - g_i)^2 \frac{1}{\sigma_g^2}) \tag{4}$$



(a) Mosaic without the blending (b) Mosaic with the blending applied

Figure 3: *The result of applying the blending.*

where N_{ij} is number of pixels in image i that overlap with image j (note that N_{ij} does not necessarily equals N_{ji}), g_i is the searched gain parameter for image i while I_{ij} is the mean value of intensity of pixels in image i that overlap with image j. The σ are standard deviations of normalized intensity error and gain. Following from the prior work of [6], we choose these values to be $\sigma_N = 10.0$ and $\sigma_g = 0.1$. The $(1 - g_i)^2$ term has been added to keep the gain parameters close to unity without which it would result with a solution where $\mathbf{g} = 0$.

This gives a linear system of equations, which we solve by the Gaussian elimination method. This solution results in a recovery of the gain parameter vector $\mathbf{g}$ which contains the gain parameters for every video frame, i.e. g_1, g_2, ..., g_n. This is then applied to the visualization of the video mosaic as a texture parameters (separately for each frame at the time of rendering). The result of this gain compensation is shown in the Figure 2b.

Blending

The second visual enhancement used to improve the overall output quality is the video frame blending. It solves the problem of visible seams on the resulting image mosaic by blending the video frame border with the overlapped one. Brown and Lowe [6] suggest a multi-band blending methodology to merge the images in the composite panoramic image but such an approach is not readily possible to implement within real-time bounds using a hardware accelerated (OpenGL) visualization approach. Hence we use a much simpler approach by associating an *alpha channel* with each video frame. This channel specifies the opacity of a given part of the image. It is set to be completely opaque in the video frame centre with an increasing transparency towards the edges following a linear distribution. Despite the simplicity of the approach, the experimental results show that it is effective. In Figure 3 we can see the seams apparent within the mosaic prior to blending (Figure 3a) and an increase in the perceived quality of the mosaic post blending (Figure 3b).

3 Results

The main application of our approach was an aerial map construction from the UAV video imagery alone, i.e. no GPS or any other telemetry was used. The footage contained urbanised as well as more challenging (i.e. feature-sparse) rural areas with large uniform fields that contain lesser amounts of distinctive features to match.

Video mosaics presented in Figures 4 and 7 have been constructed from a top-down UAV camera. This is a typical setup for an aerial map construction. The bottom part of the mosaic in Figure 4 left presents a case of feature-sparse rural area which has been handled properly by our mosaicing approach. Some further tests have been performed using perspective angle view from the UAV camera and resulting video mosaics are presented in Figure 5. This shows that our approach can be successfully applied to a wide range of applications. Figure 6 presents the video mosaic at different stages as the video imagery is being received from the UAV platform (the red marquee shows the current position of the camera). This example clearly presents the gain in situational awareness of UAV platform operator by providing the wide angle video mosaic as opposed to the current view from the transmitted UAV video imagery alone.

The UAV used for this work was an electrically powered 3.6m wing-span fixed wing platform operating at an approximate altitude of 110m. The camera pod (containing both thermal and visible band cameras) was set to give a top-down planar view of the environment with perspective views obtained during banked turns. Control of the UAV was by both remote-control and autonomous auto-pilot for the collection of the aerial video imagery used in this work.

In terms of the computing equipment a portable quad-core *Intel Core i7* CPU based workstation with a *Nvidia GeForce 9800 GT* graphics card was used for primary development but secondary testing was also performed on a dual-core laptop where performance was also found to be satisfactory. Both test platforms showed the availability of ruggidized multi-core CPU hosts suitable for current operational scenarios for UAV command and control and can readily host real-time video mosaicing from live platform video input.

The evaluation and tests carried on the UAV video imagery has shown the real-time performance of the video mosaicing implementation. On average it takes about 75 ms for the program to process one video frame. This gives a frame rate of about 13 FPS, which is clearly within the bounds of real-time for any for the potential tasks this could be considered for. In cases of a video frame mismatch the performance drops due to the executed search for a global match, however it is still within the real-time bounds.

The performance of the visualization also meets the real-time requirement. The average processing time of the hardware accelerated (OpenGL) loop is 45 ms in the average case and 50 ms in the worst case. This gives 20 frames per second in the worst case, which is enough for visualization to be responsive for the user, i.e. the operations on the mosaic such as movement, zooming and rotation can be performed smoothly and without any perceived delay. Additionally, since the frame rate of the visualization is higher than the frame rate of the main processing loop, the presentation of all captured video frames is assured.

The global bundle adjustment processing time varies with the number of video frames registered on the image. Although it depends on the actual positioning of the video frames and on the number of matched feature points, for a low image count (about 20) the algorithm does not take more than half a second to globally align the video frames. However, when we consider larger mosaics, for example composed of 80 frames, the global bundle adjustment can take about three to five seconds but it is sufficient for the task of elimination of the accumulated errors. As this processing is carried out in parallel to the visualization and per-frame registration its effect on real-time performance is negligible.

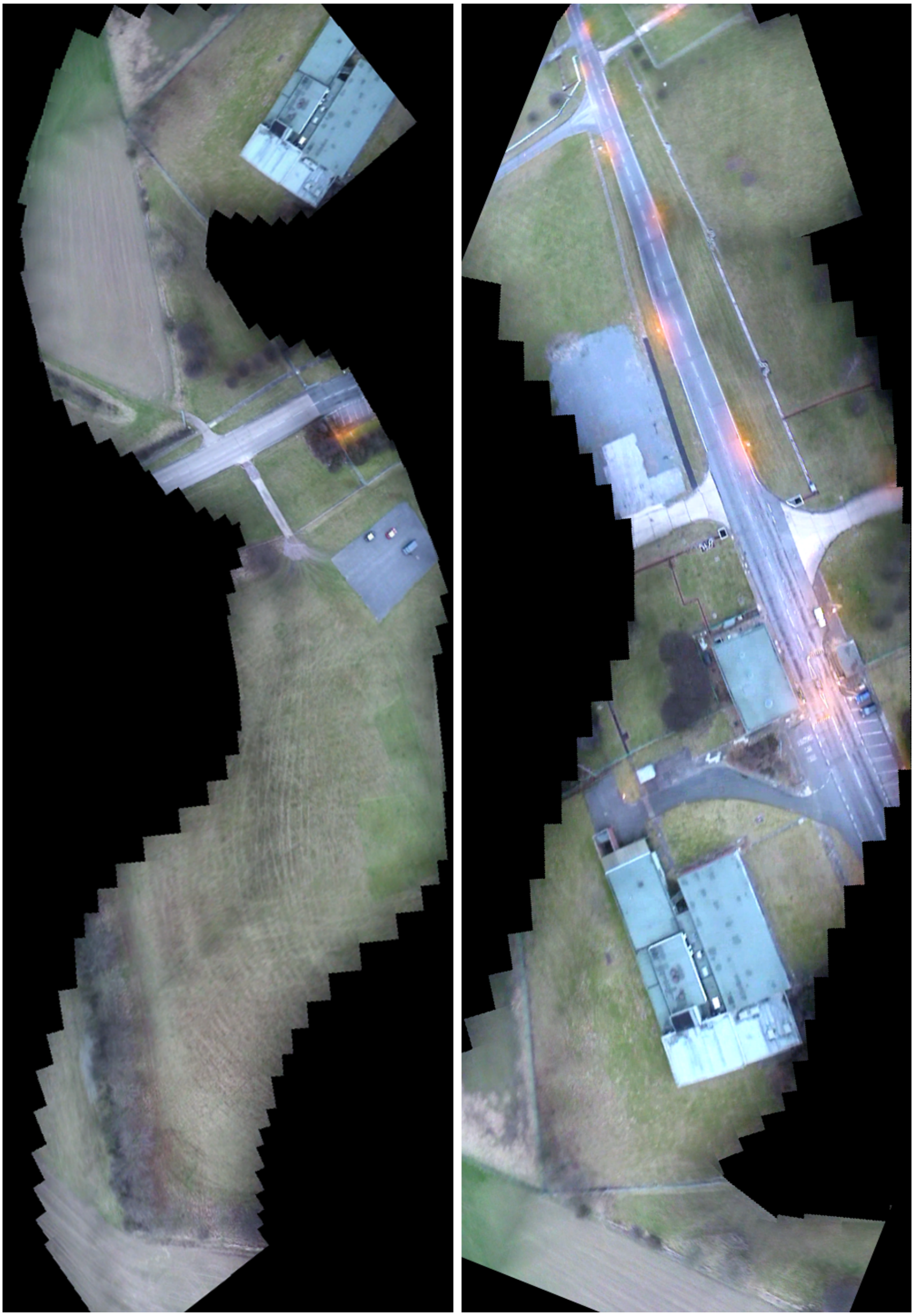


Figure 4: *Mosaics constructed from a top-down UAV camera footage*

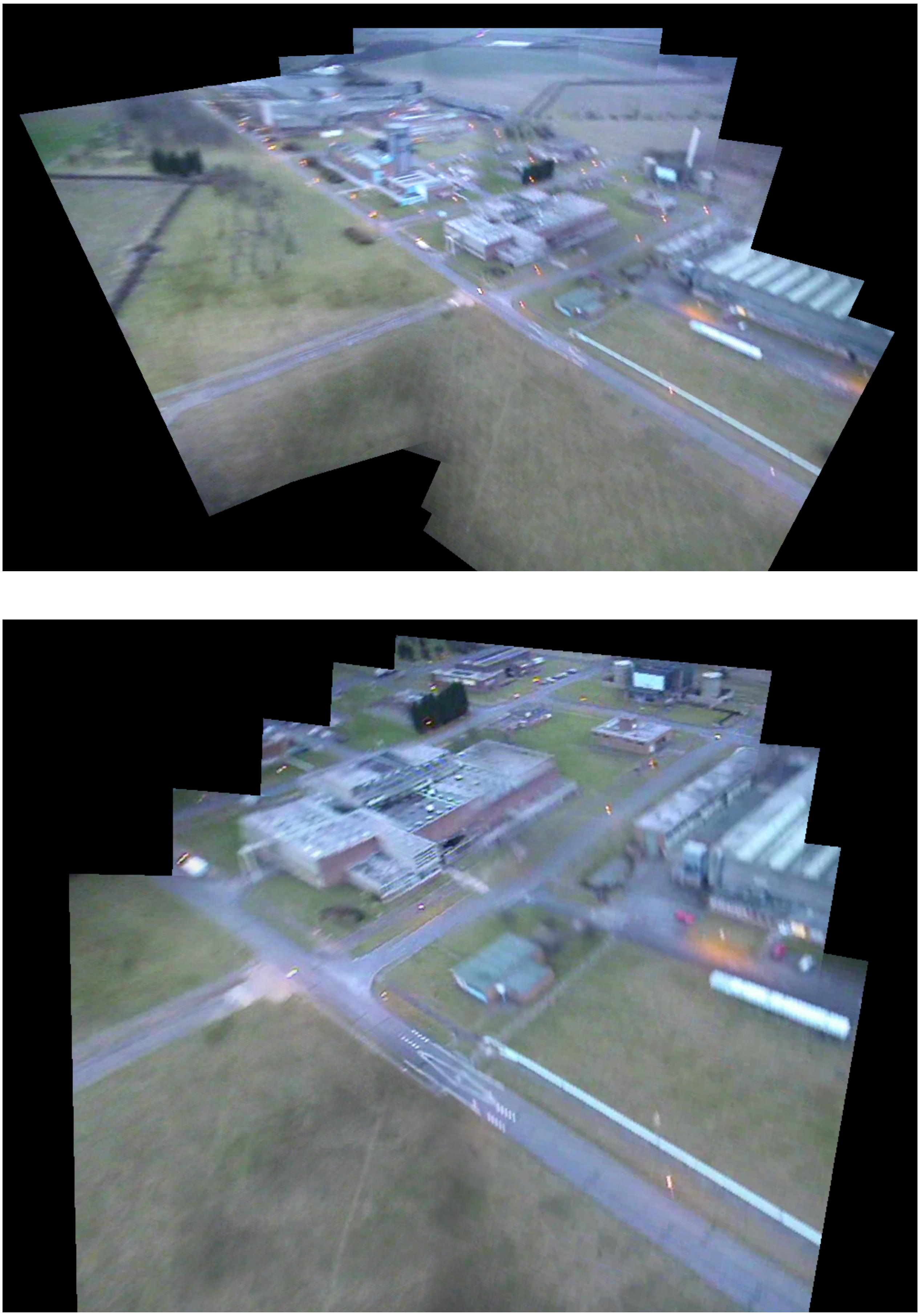


Figure 5: *Mosaics constructed from an angled UAV camera*

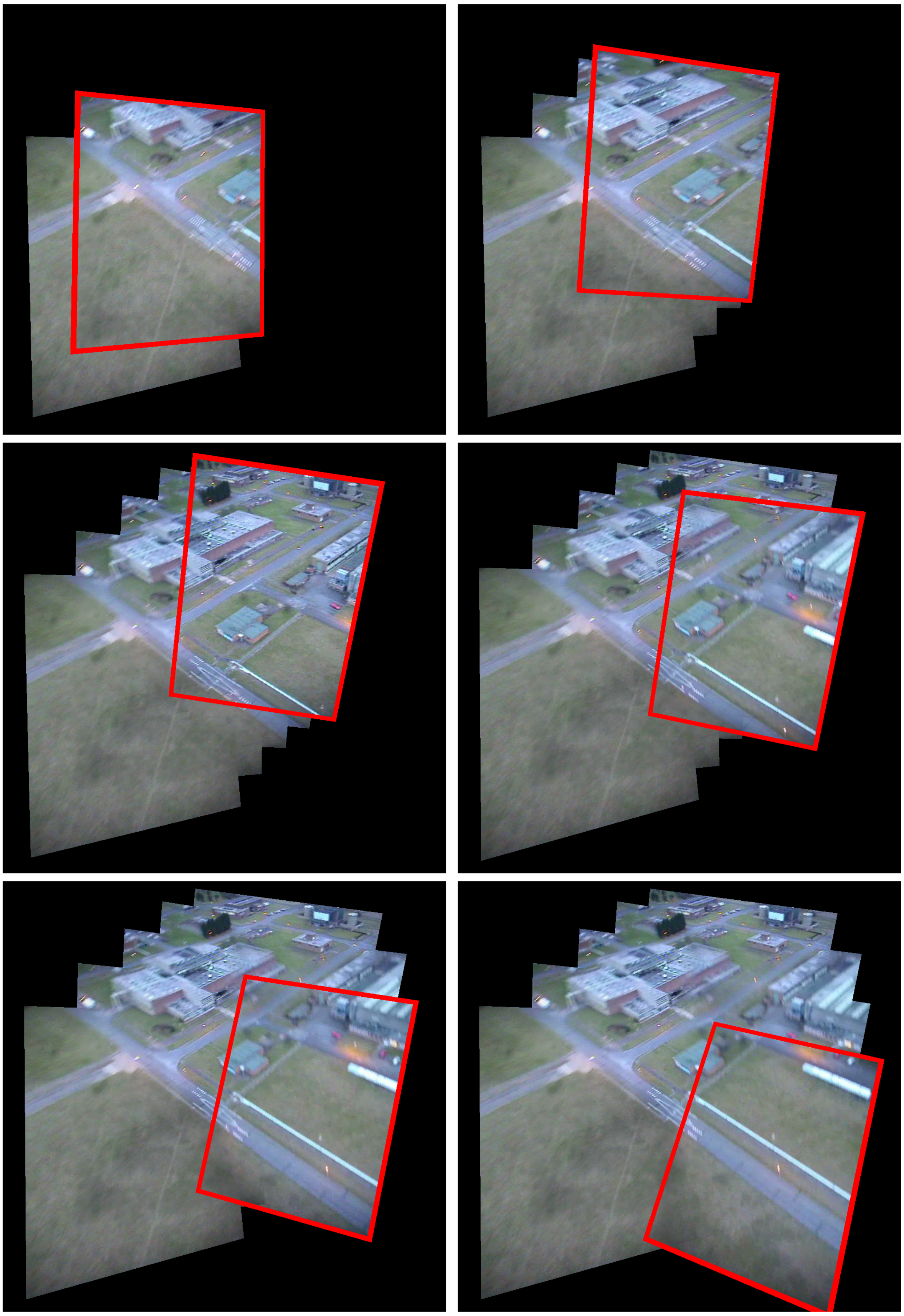


Figure 6: *Construction of the video mosaic as new frames are being received*

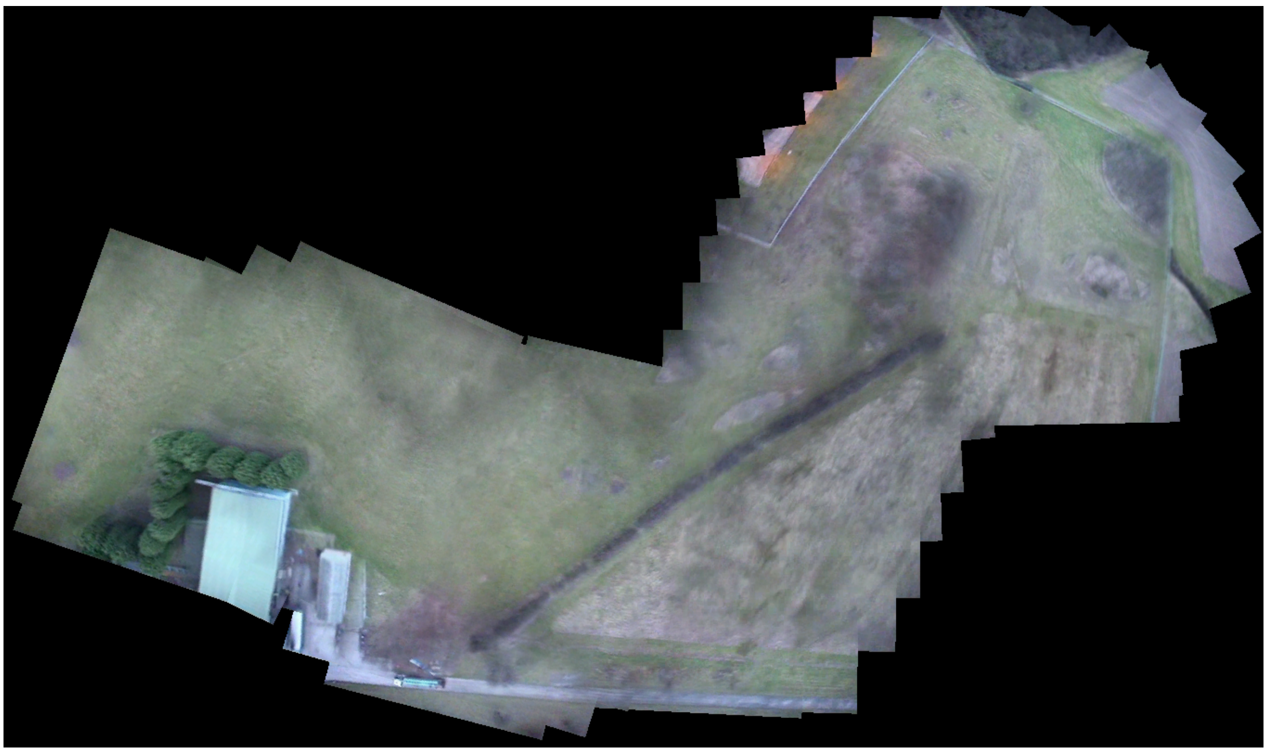


Figure 7: *Another example of a mosaic constructed from a top-down UAV camera footage*

4 Conclusions

In this paper we have presented a novel, feature point based approach for the task of real-time video mosaicing for UAV applications. The use of recent developments in the field of feature point extractors along with the methods known from the field of high quality panoramic stitching, hardware accelerated visualization and task parallelization are merged with novel heuristics - the frame sieve and two-tier (pairwise/global) bundle adjustment to allow us to robustly solve this demanding real-time task.

The evaluation of the presented system has confirmed that our system robustly constructs video mosaics in real-time. It can endure the presence of motion in the scene, varying lighting conditions and behaves well in a diverse range of environments. The presented solution is clearly real-time since it achieves the frame rate of 13 FPS in terms of processing the input video stream, and 20 FPS in terms of visualization. However, it requires a modern, parallelization capable CPU hardware to operate within these bounds.

In general our methodology employing the feature point based stitching can be successfully applied to various tasks concerning aerial video mosaicing and further development can introduce a specialisation into more specific branches such as aerial maps construction, visual positioning, UAV command and control enhancement or integration with automated object/threat detection [1, 2, 3].

References

[1] T.P. Breckon, S.E. Barnes, M.L. Eichner, and K. Wahren. “Autonomous Real-time Vehicle Detection from a Medium-Level UAV”. In *Proc. 24th International Unmanned Air Vehicle Systems*, pages 29.1–29.9. March 2009.

[2] J. Sokalski, T.P. Breckon, and I. Cowling. “Automatic Salient Object Detection in UAV Imagery”. In *Proc. 25th International Unmanned Air Vehicle Systems*, pages 11.1–11.12. April 2010.

[3] A. Gaszczak, T.P. Breckon, and J. Han. “Real-time people and vehicle detection from UAV imagery”. In *Proc. SPIE Intelligent Robots and Computer Vision XXVIII: Algorithms and Techniques*. (to appear) 2011.

[4] J. Robinson. “Collaborative vision and interactive mosaicing”. In *Vision, Video and Graphics (VVG)*. 2003.

[5] J. Robinson. “A simplex-based projective transform estimator”. In *International Conference on Visual Information Engineering (VIE)*, pages 290–293. 2003.

[6] M. Brown and D. Lowe. “Automatic Panoramic Image Stitching using Invariant Features”. *International Journal of Computer Vision*, 74(1):59–73, Aug. 2007.

[7] M. Brown and D. Lowe. “Recognising panoramas”. In *9th IEEE International Conference on Computer Vision*, volume 2, pages 1218–1225. October 2003.

[8] R. Szeliski. “Image alignment and stitching: a tutorial”. *Foundations and Trends in Compututer Graphics and Vision*, 2(1):1–104, 2006.

[9] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool. “Speeded-Up Robust Features (SURF)”. *Computer Vision and Image Understanding*, 110(3):346–359, 2008.

[10] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, New York, NY, USA, 2003. ISBN 0521540518.

[11] B. Triggs, P. McLauchlan, R. Hartley, and A. Fitzgibbon. “Bundle adjustment — a modern synthesis”. *Vision Algorithms: Theory and Practice*, 1883:153–177, 1999.

[12] M. Lourakis and A. Argyros. “SBA: A Software Package for Generic Sparse Bundle Adjustment”. *ACM Transactions on Mathematical Software*, 36(1):1–30, 2009.

[13] M. Woo, J. Neider, T. Davis, and D. Shreiner. *OpenGL Programming Guide: The Official Guide to Learning OpenGL, Version 1.2*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1999. ISBN 0201604582.

[14] M. Fischler and R. Bolles. “Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography”. *Communications of the ACM*, 24(6):381–395, 1981.

[15] C. Schmid, R. Mohr, and C. Bauckhage. “Evaluation of Interest Point Detectors”. *International Journal of Computer Vision*, 37(2):151–172, 2000. ISSN 0920-5691.

[16] L. Juan and O. Gwun. “A comparison of SIFT, PCA-SIFT and SURF”. *International Journal of Image Processing*, 3(4):143–152, July/August 2009.

[17] D. Lowe. “Distinctive image features from scale-invariant keypoints”. *International Journal of Computer Vision*, 60(2):91–110, 2004.