Mosaicing Scenes with a Quadcopter

Meghshyam G. Prasad & Sharat Chandran
Dept. of Computer Science and Engineering, IIT Bombay
{meghshyam, sharat}@cse.iitb.ac.in

Michael S. Brown
School of Computing, NUS
brown@comp.nus.edu.sg

Abstract

This paper focuses on a method of constructing panoramas from a quadcopter, and a new mosaicing sub-problem when the scene contains significant regions of vacant spaces. These vacant spaces yield little to no features to match input images and hence challenge existing mosaicing techniques. We describe a framework that is able to handle this unique input by leveraging the availability of the inertial measurement unit (IMU) data from the quadcopter.

Specifically, our method uses the imprecise IMU data accompanying a video to select a subset of images that contain interesting scene content. When the scene is such that this subset contains no vacant space, an appropriate panorama is effected; however, with featureless spaces, existing mosaicing methods do not work. In this paper, the subset is partitioned into multiple clusters. These subsets can now be stitched into a series of mini-panoramas, but a complete mosaic is not yet available. The gaps between these mini-panoramas represent regions of featureless spaces in the scene. Therefore, we once again use the IMU data together with a novel stereo reconstruction to determine appropriate portions of the images to complete the panorama. We demonstrate the efficacy of our approach on a number of input sequences that cannot be mosaiced by existing methods.

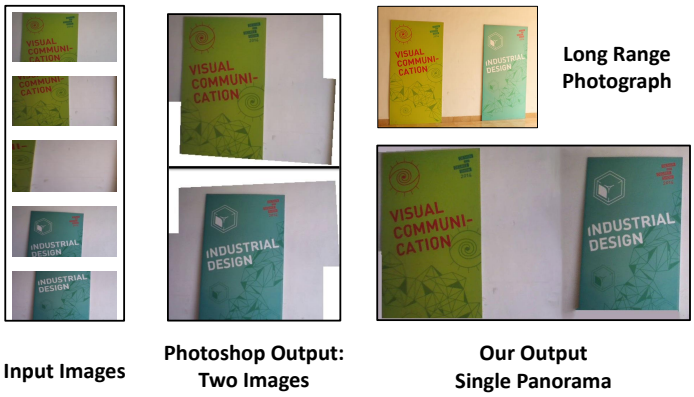

Figure 1: The long range photograph of a scene taken from an SLR camera is shown in top right. When such a scene is probed by a consumer-grade quadcopter, it results in the input images shown on the left (color balance is different from the SLR camera). The state of the art methods (middle column) are unable to make a *single* mosaic because the vacant space (more than two feet wide as seen in third picture on the left) does not seem to have any matchable features with subsequent input images. Our method handles this situation (bottom right) producing an orthographic view.

1. Introduction

Finding features and using them to align images to construct panoramas is one of the success stories of computer vision. Virtually all recent consumer cameras have this technology embedded. The success of these methods relies significantly on finding common features in the images that can be used to establish the appropriate warps to register the images together.

Problem Definition: There are scenes, however, that makes this challenging. One situation is when the scene needs to be probed in an *orthographic* view, and is not easily accessible. Murals on large urban architecture is an example. This suggests a 'close up orthographic view' by a moving camera fitted on, say a quadcopter. Another situation is when scene patterns and texture are repeated (too many similar features in, e.g., an art exhibition). This can make it challenging for a matching algorithm to find appropriate matches in large panoramas. A related situation is when a scene area simply does not contain features (too little, or no features, e.g. multiple adjoining posters in an conference event). Fig 1 shows an example of this case.

In such cases, state of the art methods are unsatisfactory. For example, the use of a moving quadcopter taking pictures suggests using a Structure from Motion (SfM) paradigm. However, based on our experiments with Bundler [17, 18] and VisualSFM [21], we see that the success of SfM depends strongly on "good" correspondences between input images, absent in large vacant (featureless) spaces. Specialized – state of the art – image stitching methods from [5, 7]

used in tuned software like Adobe Photoshop CS6 or AutoStitch *also* do not work as can be seen in Figure 1.

The goal in this paper is to create panoramic images of large scenes using a quadcopter in situations described above. From a vision perspective, we are excited about a new mosaicing problem containing large homogeneous vacant spaces. This results in scene regions that have no matches between many significant images, and therefore cannot be aligned using traditional mosaicing methods.

Key Idea We propose to solve the vacant space problem by using an inexpensive off-the-shelf flying device, such as a quadcopter which can be assumed to contain an inertial measurement unit (IMU) that has spatial information. The proximity relationship that the resultant images have, can be used to significantly reduce the search space in finding matches. Further, the proximity relationship also allows, in principle, to vary the parameters involved in feature selection. For example, if there is reason to believe that two images are adjacent horizontally, one can choose to adjust thresholds in feature matching algorithm to hunt for otherwise elusive matching pairs.

We note that the IMU data can be also made available in other devices such as smartphones. An autonomous programmed quadcopter, however, is particularly enticing because of its ability to fly to areas that are accessible to the human eye, but inaccessible for the human to reach. Such areas do not lend themselves easily to high quality images.

A new challenge, however, presents itself. The IMU data, whether on an expensive smartphone or on an inexpensive quadcopter, cannot be relied exclusively, or sometimes at all. Our experiments indicate that the roll and pitch angles (depending on the distances involved) may be completely off, and so can the positional coordinates. This is a consequence of jerky, swift movements. Even if the inexpensive quadcopter is endowed with a GPS, indoor scenes present a challenge. Therefore, whenever there is overlap in feature space between images we use superior vision techniques (specifically, homography) to stitch i.e., to construct mini-panoramas. However, when vision is inappropriate, due to the present of vacant spaces, we use the IMU information to join mini-panoramas into the complete panorama.

Contributions The main technical contribution of this paper is that it improves the state of the art in mosaicing. We assume that the imagery is acquired by a quadcopter for the reasons mentioned above. Sending a battery of images from a quadcopter to an image mosaicing algorithm such as AutoStitch incapacitates the algorithm because of the sheer number of images. Sending a sampled version of images to a manageable number N of images, with $O(N^2)$ possible areas to match for features, also does not work since the sampled image contains vacant space. In this paper, we use the IMU information that lends itself to a graceful $O(N)$ algorithm. In summary we have a solution to a new problem, and a faster solution using the IMU data.

In this paper, we assume that the scene lies on a planar surface The quadcopter is programmed to have a viewing angle perpendicular to any desired planar structure. The standard homography computation is still not possible because of the vacant spaces. To overcome this, we reduce the mosaicing problem to the stereo problem and are thus able to complete the panorama.

2. Related Work

Panoramic image stitching (alternatively, image mosaicing) is a well-studied problem in the field of computer vision. Representative works include [14], [15], [8], [20] [6] [5]. A full discussion on related works is outside the scope of this paper, readers are referred to [19] for an excellent survey. Given the maturity of this area, there are various freeware as well as commercial software available for performing image stitching; most notable are AutoStitch [3], Microsoft´s Image Compositing Editor [13], and Adobe´s Photoshop [1].

All of these methods are based on a similar strategy of finding features in each image, matching these features between images, and then computing pairwise image warps to align them together. A bundle adjustment is often applied to globally refine the alignment. All of the aforementioned methods assume the imaged scene is planar or that the camera has been rotated carefully around its center of projection to avoid parallax.

Brown *et al*. [7] proposed a method that used invariant features located at Harris corners in discrete scale-space and oriented using a blurred local gradient for stitching. Eden *et al*. [9] were able to stitch images with large exposure difference as well as large scene motion into single HDR quality image without using any additional camera hardware.

All of the image mosaicing methods work only when there is an "intersection" in feature space of images to be stitched. When there are "gaps" (either physical or due to lack of features) between images to be stitched it is not clear how to perform the stitching. In this paper, we discuss how to use the available IMU data that accompanies our input images to help overcome these problems.

3. Methodology

The goal of this paper is to compute a panorama of a scene lying on planar surface that is, possibly, difficult to reach manually. The scene is assumed to have vacant spaces. A schematic for this problem is shown in Figure 2.

The method adopted is pictorially depicted in the overview shown in Figure 3 and is described in detail later on. In brief, we systematically acquire a video of the scene, reduce the input video to a manageable number of images,

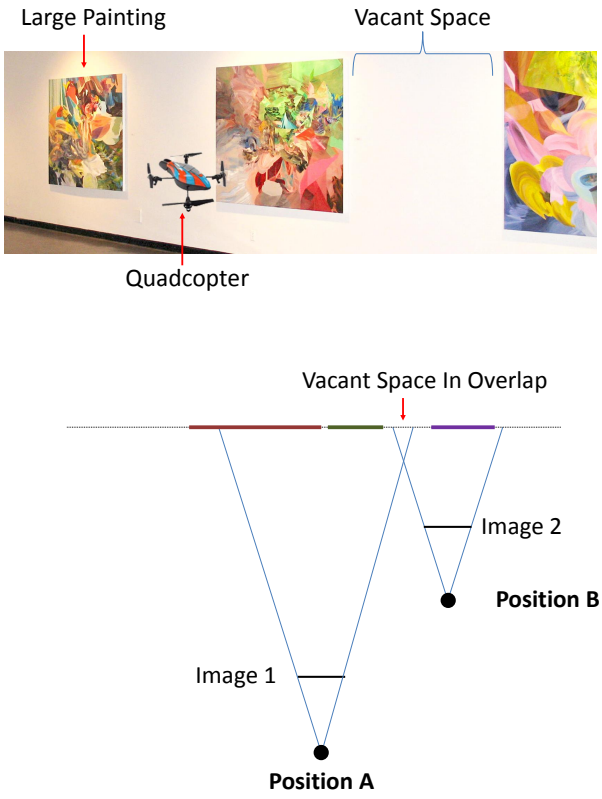

Figure 2: Problem definition. (Top) Vacant spaces are encountered in various scenes. When individual portions are captured by a quadcopter, how does one create the complete mosaic given that common features are either not available as in this example, or confusing (see Fig. 10)? (Bottom) Simplified reduction of the problem.

and finally combine the images acquired from different positions into a mosaic.

3.1. Video acquisition

We first dispatch the quadcopter to as close to the scene as possible. The corners of a rectangular area of interest are provided to the quadcopter, and it is programmed to traverse the area in a raster scan fashion. There are various control aspects involved in sending a quadcopter; in outdoor areas, the quadcopter is impacted by wind and it might lose its way. The control aspects of the quadcopter is beyond the scope of this paper. Note that trying to create a mosaic in an incremental linear fashion by combining adjacent frames is prone to loss of two-dimensional spatial proximal information. It is also computationally overwhelming.

The quadcopter returns with a video of the scene. Images extracted from a short video of about a minute or more overwhelms existing mosaicing software, such as AutoStitch or Adobe Photoshop[1]

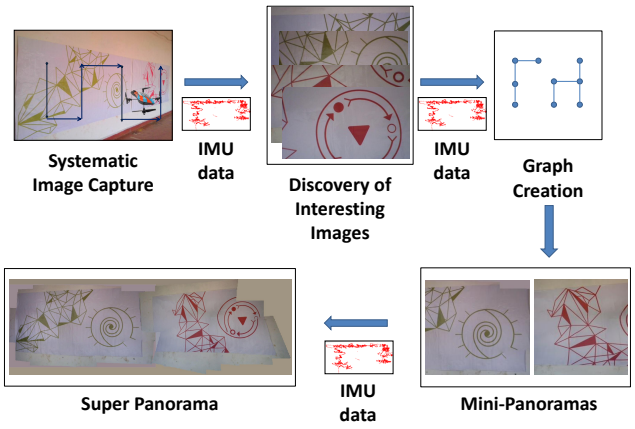

Figure 3: Overview: Input imagery is systematically acquired (top left) by a quadcopter. In the next step, interesting images are found by clustering the video into regions based on positional data. A graph is constructed using proximal images. For each connected component in a graph, standard stitching techniques are used to create mini-panoramas which are then joined together into super panorama again using the IMU data.

3.2. Selecting interesting images

Our goal in this step is to reduce the amount of input data and produce a set of interesting images. In other words, we wish to convert a video into an album of images. The key difference between our problem and standard albumization [2, 12] is the use of positional information. A standard quadcopter has an IMU that, after calibration, may give reasonable information of positions. Using positional information it is possible to cluster the images, and sort the images into an $m \times n$ grid. The number of cluster centers is automatically determined using the agglomerative bottom up hierarchical clustering method [16], with the additional requirement that the whole scene (represented by the positional data) is covered.

Clustering Details We assume that each IMU data position corresponds to an image of definite fixed dimensions. Consider each position of the IMU data to be a leaf node. Two nodes are greedily combined based on the closest Euclidean distance, and replaced with an internal node; the position of the internal node is set to be the centroid of the two nodes, and each internal node now corresponds to a virtual image of the same size taken by a virtual quadcopter. The algorithm recursively merges all the nodes till we end up

[1] In the rest of this paper, we use AutoStitch to indicate state of the art stitching programs such as AutoStitch, Photoshop, etc.

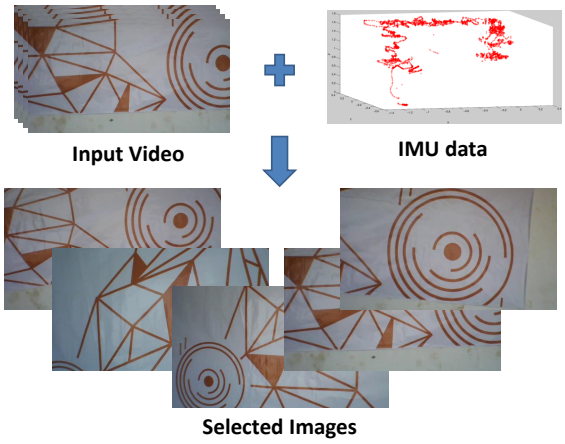

Figure 4: We align the image stream with the IMU data, and then transform the video into a set of interesting images with a clustering algorithm.

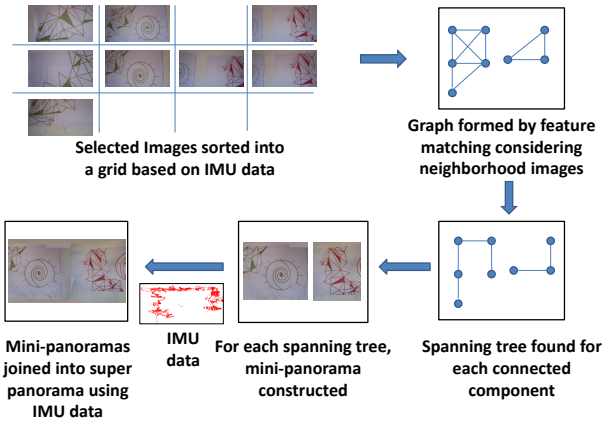

Figure 5: Interesting images acquired are segmented and individual (mini-panoramas) are constructed. These are then later combined into the desired super-panoramas using the IMU data.

with a root. In the next phase, we produce cluster centers; a set of nodes is considered for being the output as cluster centers if the union of these nodes completely spatially cover the scene. From the bottom-up construction, it is clear that the root will represent a single position, and thus a single virtual image, and will not cover the scene. At the other extreme, the set of all leaf nodes *will* cover the scene. To resolve this, during the calibration phase, we pre-decide the minimum distance between two center of projection to have least overlap. This is used as the threshold in the clustering algorithm. Once cluster centers are found, we pick the leaf node which is closest to the cluster center to find a real image. This process is schematically shown in Figure 4.

Remarks: If we had no IMU information, one may consider selecting a set of interesting images using any appearance based method such as optical flow or feature selection. However, due to the jerky and uneven motion of the quadcopter, such measures do not prove to be sufficient.

In practice, the number of cluster centers and thus images for the scenes we have covered is now within the capacity of AutoStitch. As mentioned in the introduction, as long as there are sufficiently varying and "matchable" features, AutoStitch is able to perform a reasonable result. However, if there are very few features in overlapping region of two images, then the output is not acceptable. This situation will arise when there is vacant space in the imagery.

Time complexity AutoStitch has not been designed to use positional information. As a result if there are N input images, the program has to consider possible matches in approximately $O(N^2)$ set of areas. Our program is able to mosaic in an $O(N)$ fashion.

Mini-Panoramas Specifically, we assume at this point that the interesting photos are available in the form of a $m \times n$ grid. First, we find SURF [4] features for each image in a grid. Next, we use Best of Nearest Neighbor matcher (from the OpenCV library) with Random Sample Consensus (RANSAC) [10] to find geometrically consistent matches between neighborhood images inside grid. We create a graph with images being nodes, and add an edge between two nodes if there are sufficient matches. We have to recall at this point that if there are "vacant spaces" there will not be enough features for successful matches; the graph will end up with multiple (disconnected) components. We next compute multiple spanning trees for the various components. Given a spanning tree, the center of the spanning tree is a node from which the distance to all other nodes is minimal [11]. Next we calculate the homography of each image with respect to spanning tree center. Finally, for each spanning tree, we stitch all pictures within the spanning tree to create a mini-panorama using the computed homographies by warping all images with reference to the image at spanning tree center. The spanning tree is an $O(N)$ structure. The process is described in Figure 5.

3.3. Super-panorama

In this section, we consider the situation when programs like AutoStitch fail. We assume that the output of the previous step has resulted in multiple spanning trees where each spanning tree center corresponds to a specific depth, since we have stitched all images by taking the spanning tree center as a reference. Individual panoramas for each spanning tree termed mini-panoramas have been created. A super-panorama must be created from mini-panoramas; these usually correspond to different depths for at least two reasons.

First, it is invariably difficult, if not impossible, to control a quadcopter to be at the exact depth even in indoor scenes. The aerodynamics and the thrust produced tends to make the quadcopter drift. Second, it might also be necessary to let the quadcopter probe and come closer to the scene so as to get a "good picture".

A super-panorama is done using a two step process. Assume two trees in the forest corresponding to area A and area B of the scene (see Figure 6). Assume that a mini-

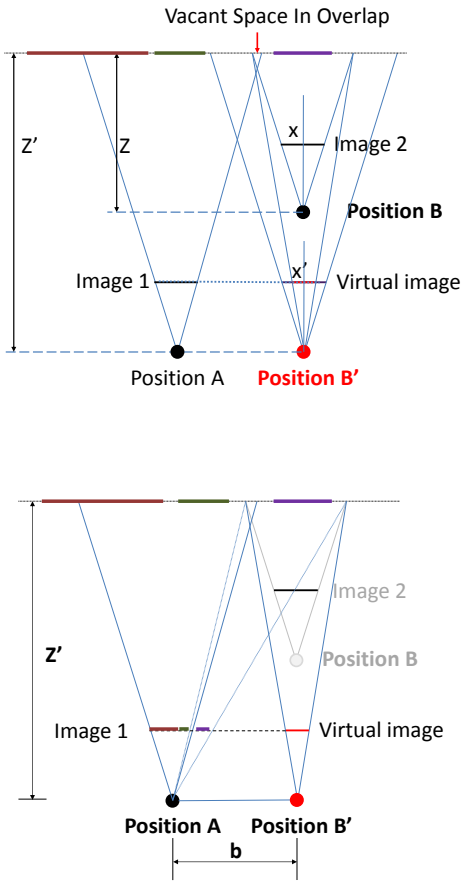

Figure 6: (Top) The virtual picture as seen from position B' is computed using Equation 1 from the real picture taken from B. (Bottom) Using the stereo disparity, calculated from the baseline width b and depth Z', it is possible to depict the composite scene obtained from both A and B' (from the view point of A).

panorama is created from these two areas, and the depth of the planar surface from the camera is more for A, than for B. We then take the mini-panorama image captured at B, and 'move' it to a new location B' whose depth (from the imaged surface) is the same as that of A. The resulting image will be smaller; the images are related by the equation

$$\frac{\mathbf{x}'}{\mathbf{x}} = \frac{\mathbf{Z}}{\mathbf{Z}'} \tag{1}$$

Figure 7: Candidate images (from different depths) for a super-panorama. For context, see Fig. 10. These images are 'reference' images (spanning tree center) of individual mini-panoramas.

where **x** (respectively **x'**) represents a pixel location of the image in B (respectively, B') and **Z** (respectively **Z'**) represents the depth of the images surface from B (respectively, B').

In order to form a super-panorama from the depth of A, we can now treat the resulting images from A (unchanged) and B (computed from Equation 1) forming a simplistic stereo pair at the depth of position A. Using the stereo disparity formula we can "place," from the view point of A, the image captured from B', thereby creating a super-panorama. (We could as well present the entire scene from the viewpoint of B' (since it is at the same depth); we prefer these pictures to the one that one may be created from the depth of B.)

Example: Consider two images (shown in Fig. 7) taken from the positions (1.37, 0.85, 1.6), and (3.75, 0.98, 1.4). As their depths are different, we (virtually) move the second image by shrinking the second image by a factor $\frac{1.4}{1.6}$, i.e., to 87% of its original size. With both images at the same depth (1.6m), the disparity of the second image is

$$\begin{aligned} \text{disparity}_x &= (3.75 - 1.37)f/1.6 = 839 \\ \text{disparity}_y &= (0.85 - 0.98)f/1.6 = -45 \end{aligned}$$

where f is focal length of the quadcopter camera in pixel units.

3.4. Summary: Use of the IMU

The IMU data is used primarily for two purposes:

1. **Selection and ordering of images:** We use the IMU data to select representative images from the video and arrange them into rectangular grid according to the 'spatial' neighborhood. It also disambiguates situations when multiple images that are spatially distant, but have similar, repeated features.

2. **Super-panorama:** Whenever there are no features in the overlap region of two images, we use the IMU data to find the relative position of one mini-panorama with respect to another.

4. Experiments and Results

All our experiments have been completed with the inexpensive consumer quadcopter called a Parrot's AR Drone 2.0. The imagery acquired were from actual graffiti painted on large walls as well as posters in an exhibition. We have used the ROS based ARDrone Autonomy Driver to communicate with the drone. For the purpose of showing the efficacy of this paper, we also took a picture of the scene from a distance with a smartphone camera to better understand the scene.

We have implemented our algorithm in C++ using the OpenCV library. Experiments were performed on a PC with Intel Core i7 processor(@3.4GHz) and 8GB RAM. Please visit `http://goo.gl/sYvoVP` for datasets and code related to the paper.

4.1. Selecting Images

In our first experiment, we wanted to ensure that the selection of images done was comprehensive and useful. We sent the drone to image an outdoor scene with no vacant space. This experiment was conducted in an outdoor environment. We note here that there were approximately 3000 images in the raw video. AutoStitch and Photoshop were unable to cope when fed with this large number of frames.

One way to produce some sort of mosaic was to simply reduce the amount of data given to AutoStitch. Figure 8(a) shows uniformly (time) sampled images from the video. When these sampled images are given to AutoStitch or to Adobe Photoshop, we find (Figure 8(b)) that these programs are able to produce some output, but the results are not satisfactory.

Instead of feeding time-sampled images, we ran our selection algorithm (Section 3.2) on the video which resulted in $N = 5$ images. Though the number of input images in the video is large, the total distance covered by the quadcopter in this duration (of around 90 seconds) is small; thus the number of distinct images returned by the algorithm shows a dramatic reduction. Figure 8(c) shows examples of selected images. Many of the images are similar to the time sampled version; however, the occasional differences are enough to make AutoStitch work. The results are shown in Figure 8(d).

In summary, this experiment provides evidence to show that (a) our selection algorithm is reasonable and (b) our stitching results are comparable to that of AutoStitch for the kind of scenes considered.

4.2. Indoor Imagery with Vacant Spaces

Our next selection of experiments were conducted in an indoor environment.

The input stream had about 4300 images. The selection algorithm pruned the video into $N = 5$ images. A sample of the selected images are seen in Figure 1.

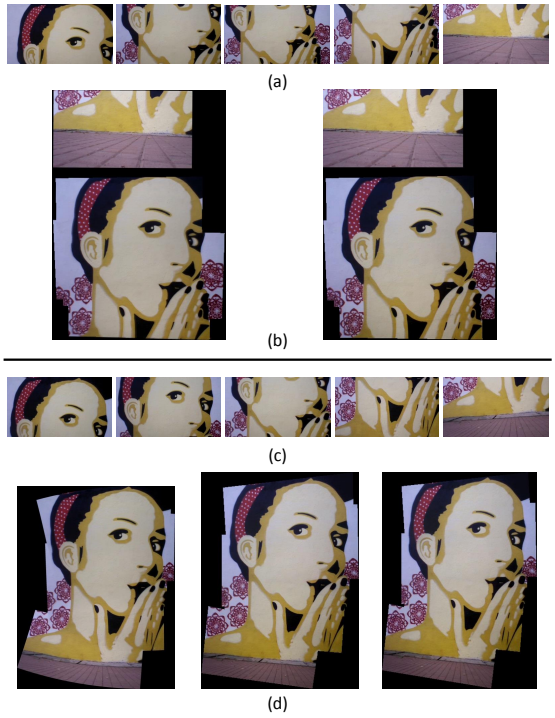

Figure 8: (a) Uniformly sampled images from an outdoor video expedition. (b) Output of the state of the art photo stitchers (left:AutoStitch, right:Adobe Photoshop CS6) on uniformly time sampled images. As time sampled images do not guarantee coverage of the scene, the panorama is broken. The top portions (see (b) & (d)) do not belong at the right place (c) Salient image selection from the set of approximately 3000 images using positional information. (d) When salient images are given to AutoStitch (left) and Photoshop (middle), we can create a panoramic mosaic (since there are no vacant spaces). We also show the result from our stitching algorithm (bottom right).

There were two disconnected components in the resulting graph. AutoStitch was unable to produce any reasonable output as seen in Figure 1. The scene, captured from a distance is also shown. We see a better orthographic view of the posters in our result shown in Figure 1(right bottom).

In an another experiment, the input stream had about 9000 input images. The selection algorithm pruned the video into $N = 13$ images. A sample of the selected images are seen in Figure 9(a). The scene as captured by a smartphone can also be seen, as well as the outputs of the state of the art stitchers. Note that AutoStitch is only able to stitch the upper half of the scene. Our result Figure 9(e) clearly stands out in comparison.

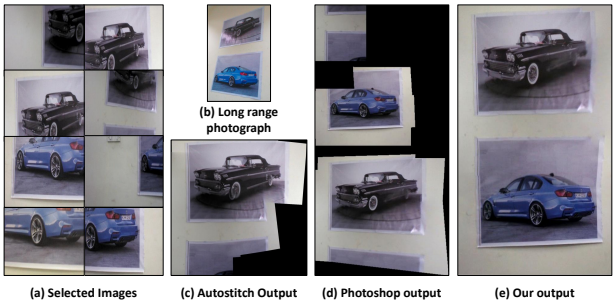

Figure 9: (a) Salient images from the quadcopter video using our selection algorithm of (b) an indoor scene. This long range photograph has been captured separately by a smartphone camera only for context. Notice a significant vertical vacant space (around three feet) in the imagery. (c) Output of AutoStitch – only the upper half of the scene is output. (d) Output of Adobe Photoshop CS6 – the vacant space posed a problem to the feature matching algorithm, so instead of a mosaic, individual pieces were output as mini-panoramas (e) Our output on the selected images. We are able to present the scene in high fidelity in an orthographic view.

4.3. Outdoor Imagery with Vacant Spaces

Our next set of experiments were conducted in an outdoor environment. The input stream had about 12000 images. The selection algorithm pruned the video into $N = 30$ images. A sample of the selected images are seen in Figure 10(a). The scene as captured by a smartphone can be seen in Figure 10(b). Figures 10(c), (d) and (e) shows the comparison of outputs of state of the art stitchers with the output of our algorithm. Note that AutoStitch is getting confused by too many matching features. Please see supplementary material for results on other indoor as well as outdoor datasets.

4.4. Analysis

The performance of our algorithm as a function of the scene, as well comparison with other software is summarized in Table 1. It can be seen that, whenever there is vacant spaces between adjacent images, AutoStitch produces only one component, presumably the largest. Adobe Photoshop outputs all disconnected components. Sometimes due to the lack of spatial proximity information, the resulting images (or components) are disconnected instead of being mosaiced (unlike AutoStitch). In contrast, in all cases, our algorithm successfully uses proximity information which results in a reduced number of mini-panoramas.

As expected the number of selected images in our saliency algorithm varies based on environment considerations (outdoor/indoor), the average depth from the scene, and the total scene area.

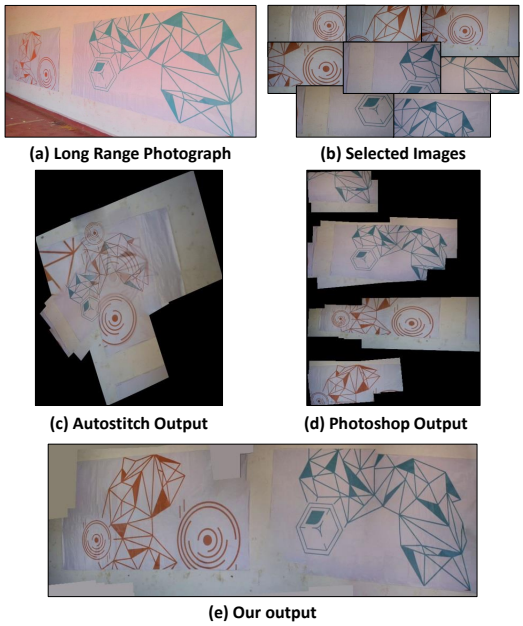

Figure 10: (a) An outdoor scene captured by a standard camera in an exhibition. The approach to the area is normally cordoned off and one needs permission to get a quadcopter to take the picture. Notice a significant gap (more than two feet) between the two posters. (b) Salient images from the quadcopter video using our selection algorithm. (c) Output of AutoStitch on the selected images. The mosaic is not reasonable presumably because of the confusion in features. (d) Output of Adobe Photoshop CS6 on the selected images. The vacant space posed a problem to the feature matching algorithm, so instead of a mosaic, individual pieces were output as mini-panoramas (e) Our output on the selected images. We are able to join two posters (separated by vacant space) using the IMU data.

5. Concluding remarks

In this paper, we have described a method of imaging large scenes using a quadcopter enabling close orthographic views. We also defined a new problem, that of computing a mosaic of a planar scene with vacant spaces. Vacant space relates to images in an input stream where there are not enough features for traditional mosaicing algorithms to estimate geometric warps to align the images.

Our solution to this problem is to use an autonomous quadcopter which is capable of taking pictures. The quadcopter has an inertial measurement unit that is capable of outputting reasonable spatial locations, but unreliable roll, yaw and pitch. Using this positional information, our algorithm selects an "interesting" subset of the video imagery. Whenever there is overlap in feature space in the subset, we

Dataset	Number of Images in video	Approx. planar area covered	Number of selected images	AutoStitch: # Components	Photoshop: # Components	Our algorithm: #mini-panoramas	Remarks
Lady	3000	60 sqft.	5	1	1	1	As there are enough features in the intersection of selected images, AutoStitch, Photoshop as well as our algorithm produces the panorama correctly.
Indoor exhibition	4300	40 sqft.	5	1	2	2	As there is vacant space between the two posters, AutoStitch produces only one panoramic image. Photoshop outputs two posters as two disconnected components; these correspond to our mini-panoramas.
Cars	9000	60 sqft.	13	1	3	3	As there is vacant space between the two visuals, AutoStitch produces only one panoramic image, the black vehicle. In the case of Photoshop, two of the three disconnected components represents two partial visuals, while the third component is the intersection between the two cars – this portion contains featureless space.
Outdoor exhibition	12000	80 sqft.	30	1	4	2	AutoStitch is confused by the replicated features in the two posters which are sometimes proximal and sometimes geographically distant. A single panorama is produced, but the output is incorrect. We use the IMU data for arranging the images in spatial neighborhood; we have fewer mini-panoramas. Photoshop is not able to produce a super-panorama and the number of disconnected components in Photoshop's output is larger than the number of mini-panoramas.

Table 1: Quantitative summary of results.

stitch images using traditional vision-based methods avoiding the erroneous roll based warping. At other times, we use positional information, and reduce the resulting problem of computing a complete panorama to the stereo problem of merging mini-panoramas. Our method works on both indoor and outdoor scenes.

Controlling a consumer-focused inexpensive quadcopter can be problematic; for instance the quadcopter could have severe yaw and roll. For future work, vision based algorithms to control such quadcopters might be quite useful.

References

[1] Adobe Photoshop CS6. http://www.adobe.com/products/photoshop. [Online; accessed 23-January-2016]. 2

[2] A. Aner and J. R. Kender. Video summaries through mosaic-based shot and scene clustering. In *ECCV*, 2002. 3

[3] AutoStitch. http://matthewalunbrown.com/autostitch/autostitch.html. [Online; accessed 23-January-2016]. 2

[4] H. Bay, T. Tuytelaars, and L. V. Gool. SURF: Speeded Up Robust Features. In *ECCV*, 2006. 4

[5] M. Brown and D. Lowe. Recognising panoramas. In *ICCV*, 2003. 1, 2

[6] M. Brown and D. G. Lowe. Automatic panoramic image stitching using invariant features. *International Journal of Computer Vision*, 74(1):59–73, 2007. 2

[7] M. Brown, R. Szeliski, and S. Winder. Multi-image matching using multi-scale oriented patches. In *CVPR*, 2005. 1, 2

[8] D. Capel and A. Zisserman. Automated mosaicing with super-resolution zoom. In *CVPR*, 1998. 2

[9] A. Eden, M. Uyttendaele, and R. Szeliski. Seamless Image Stitching of Scenes with Large Motions and Exposure Differences. In *CVPR*, 2006. 2

[10] M. A. Fischler and R. C. Bolles. Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. *Commun. ACM*, 24(6):381–395, June 1981. 4

[11] W. Kocay and D. L. Kreher. *Graphs, Algorithms, and Optimization*. Discrete Mathematics and Its Applications. Chapman and Hall, 2004. 4

[12] Y. Lee, J. Ghosh, and K. Grauman. Discovering important people and objects for egocentric video summarization. In *CVPR*, 2012. 3

[13] Microsoft Image Composite Editor. http://research.microsoft.com/en-us/um/redmond/groups/ivm/ICE/. [Online; accessed 23-January-2016]. 2

[14] D. Milgram. Computer methods for creating photomosaics. *IEEE Transactions on Computers*, C-24(11):1113–1119, 1975. 2

[15] D. Milgram. Adaptive techniques for photomosaicking. *IEEE Transactions on Computers*, 26(11):1175–1180, 1977. 2

[16] L. Rokach and O. Maimon. Clustering methods. *Data mining and knowledge discovery handbook*, pages 321–352, 2005. 3

[17] N. Snavely, S. M. Seitz, and R. Szeliski. Photo Tourism: Exploring image collections in 3D. *ACM Transactions on Graphics*, 25:835–846, 2006. 1

[18] N. Snavely, S. M. Seitz, and R. Szeliski. Modeling the World from Internet Photo Collections. *International Journal of Computer Vision*, 80:189–210, 2007. 1

[19] R. Szeliski. Image alignment and stitching: A tutorial. Technical report, MSR-TR-2004-92, Microsoft Research, 2004, 2005. 2

[20] R. Szeliski and H.-Y. Shum. Creating Full View Panoramic Image Mosaics and Environment Maps. In *SIGGRAPH*, SIGGRAPH '97, pages 251–258, 1997. 2

[21] C. Wu. Towards Linear-Time Incremental Structure from Motion. In *International Conference on 3D Vision*, pages 127–134, 2013. 1