Journal of Defense Modeling and Simulation
XX(X):1–17
©The Author(s) 2021
Reprints and permission:
sagepub.co.uk/journalsPermissions.nav
DOI: 10.1177/ToBeAssigned
www.sagepub.com/
SAGE

Structure from Motion with Planar Homography Estimation: A Real-Time Low-Bandwidth, High-Resolution Variant for Aerial Reconnaissance

Christian Arnold[1], Scott Nykl[1], Scott Graham[1], and Robert Leishman[1]

Abstract

We propose a new algorithm variant for Structure from Motion (SfM) to enable real-time image processing of scenes imaged by aerial drones. Our new SfM variant runs in real-time at 4 Hz equating to an 80x computation time speed-up compared to traditional SfM and is capable of a 90% size reduction of original video imagery, with an added benefit of presenting the original 2D video data as a 3D virtual model. This opens many potential applications for a real-time image processing that could make autonomous vision based navigation possible by completely replacing the need for a traditional live video feed. The 3D reconstruction that is generated comes with the added benefit of being able to generate a spatially accurate representation of a live environment that is precise enough to generate GPS coordinates from any given point on an imaged structure, even in a GPS denied environment.

Keywords

Structure from Motion (SfM); Planar Homography; Bandwidth Reduction; Real Time SfM

Introduction

Drones are ubiquitous in our modern world. From recreational hobbies to professional activities, they play an important role our daily lives. However, to be used remotely or even autonomously, drones require ever increasing data rates to remotely transmit live imagery to operators. This is a costly endeavor for the development of drones, and could put a lot of strain on future telecommunication systems if the amount of drones flying in the sky continues to increase at the rate currently projected[1 2]. While one solution is the development of better communication infrastructure, we propose an alternative to live streaming imagery to remote operators. Our approach uses a computer vision algorithm variant to transmit similar data for a dramatic decrease in package size.

Application of Real-Time Structure from Motion

Building from recent research on this same topic using Structure from Motion (SfM)[3], which demonstrated 49%-60% compression from the original imagery, we propose a new SfM variant that is both faster and more spatially accurate than traditional methods, and most importantly, capable of calculating imagery replacements in real-time at 4 Hz on our test laptop.

SfM is a set of computer vision algorithms commonly used to create static three-dimensional models, known as a reconstruction, from a scene using only an unordered array of detailed two-dimensional photographs. From these photographs, relative camera locations of each image are recovered, and through the use of Epipolar Geometry, depth information is extracted from the comparison of different perspectives. An example of the input and output of this process is illustrated by Figure 1.

Using a new computer vision technique combining Structure from Motion and Planar Homography Transforms, we propose an algorithm capable of transforming urban environments into living 3D virtual models that can be transmitted in lieu of real-time video for a fraction of the bandwidth. Our preliminary research, shows that we are able to create simplified 3D models of sensed buildings that consume only a tenth the size of the original video.

[1]Air Force Institute of Technology (AFIT), WPAFB, OH

Corresponding author:
Scott Nykl, AFIT, 2950 Hobson Way, Bldg 642, Room 203, WPAFB, OH 45433
Email: scott.nykl@afit.edu

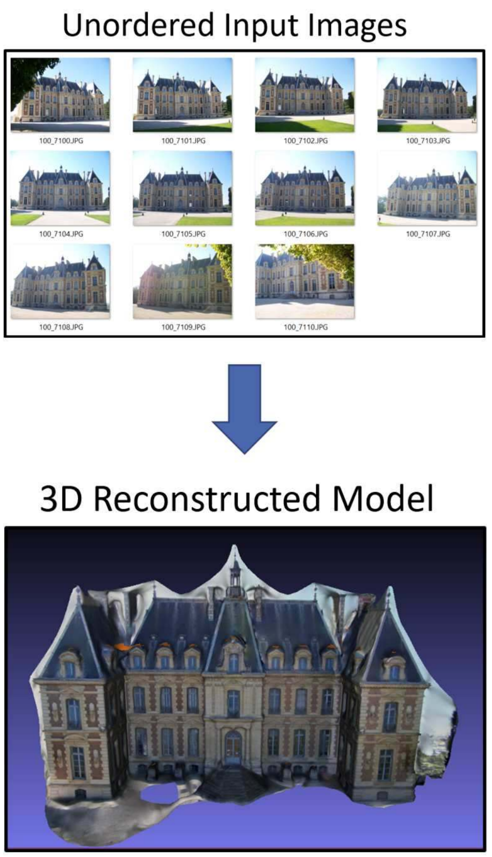


Figure 1. SfM Example: Input Unordered Images, Output Reconstruction

This research employed the AftrBurner 3D Visualization Engine created by Nykl et al[4]. It is an OpenGL-based[5] rendering engine with OpenCV[6] integration. This OpenGL / OpenCV combination enables rapid and realistic virtual image generation as well as state-of-the-art computer-vision processing. The strategic use of this visualization engine was vital to the development of the proposed method. A virtual environment enables repeatable analysis of information in a deterministic manner that conveniently eliminates many sources of error since the parameters of each experiment are explicitly specified and could be easily modified as needed. The virtual world also enabled a cost-effective testing environment, as no physical hardware other than a laptop was needed to conduct each experiment. Additionally, the experimental environment for each test was not subject to real-world problems like weather and logistics related to real world flight, as well as the regulatory process that would be necessary to accomplish these tasks.

Lastly, we encourage our readers to view our Youtube video on this algorithm at:
`https://youtu.be/Fs5-AaDO21k`[7]

Contributions

1. We propose a novel SfM variant that uses Planar Homography Estimation (PHE) to sense and visualize man-made structures more quickly and accurately than traditional SfM.
2. On a commodity laptop, our algorithm is capable of real-time computation at 4 Hz and offers compute times up to 80x faster than traditional SfM.
3. Requires less bandwidth to transmit a scene to a remote operator. Our experiments demonstrate a 90% data size reduction compared to traditional SfM.
4. Our generated 3D models more accurately preseve man-made feature locations on objects such as buildings. Our experiments show up to a 10x accuracy improvement of building window locations over traditional SfM.

Background

Technical Overview

Understanding how the Structure from Motion (SfM) process works requires in-depth knowledge of the entire digital image processing pipeline. In essence, SfM functionally “reverses” the action of taking a picture of a three-dimensional scene, so the best way to explain SfM is to first understand the process it is attempting to reverse.

The Projection Matrix

Beginning with process of taking an image, a camera can be mathematically represented by a *Projection Matrix* that encodes the *perspective* of a three-dimensional object onto a two-dimensional plane. A standard assumption made in many computer vision applications is that the camera can be approximated mathematically as an ideal pinhole camera, as referenced in Figure 2[8 9 10]. This assumption is then used to associate the data stored in pixel space with that of the camera through the means of an intrinsic camera matrix, $\boldsymbol{K}$, which, when joined with a rotation-translation matrix, forms the full *Projection Matrix*,

$$\boldsymbol{P} = \boldsymbol{K}[\boldsymbol{R}|T].$$

Next, a direct mapping can be made from 3-space, (x, y, z), to pixel space, (u, v). However, it is important to note that 3-space in the *camera frame* is slightly different than that of standard 3D graphical simulation engines. Instead of the z-axis representing the up-direction, as it commonly does in many 3D engines, it instead denotes distance from the camera aperture, where a positive z-value is a point that is in front of the camera as shown in Figure 2. The x and y vectors are then used to indicate the vertical and horizontal translations centered from the camera's frame. The relationship for mapping a 3D object to a 2D image is shown below:

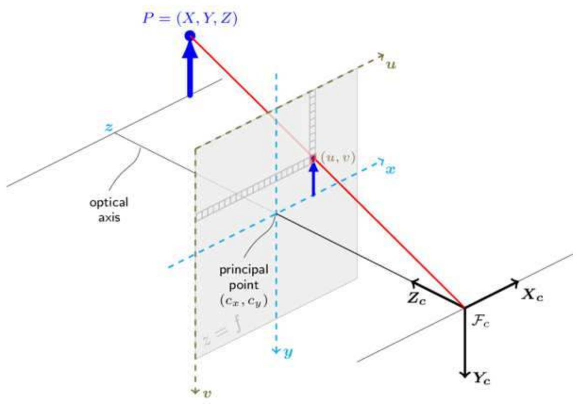


Figure 2. Pinhole Camera Model[11]

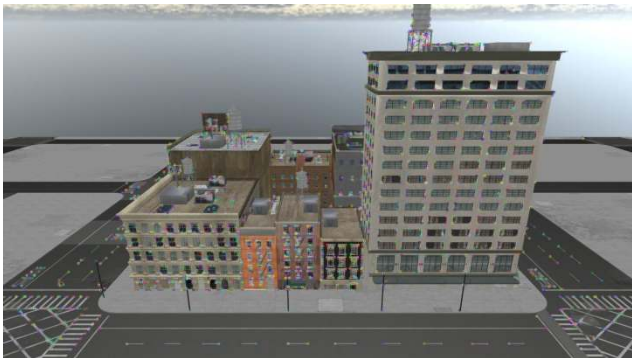


Figure 3. Visual representation of all features found in an image using the FAST feature detector

$$\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = [\boldsymbol{K}][\boldsymbol{R}|T] \begin{bmatrix} u \\ v \\ 1 \end{bmatrix}$$

Finding Image Features

Detecting specific and uniquely cross-identifiable features in images is paramount for SfM pipelines. Ultimately, feature matching is what allows for the recovery of camera position and orientation (pose)[12 13 14 15 16]. Other methods for recovering camera pose exist, they would be not be applicable to our intended application[17]. For SfM specifically, the detected features are the points that are triangulated between other images to create a sparse 3D reconstruction of the imaged scene, also known as a 3D point cloud.

Using Features

A set of features is detected within each image. These features are then compared with the discovered features in other images to determine image pairs as shown in Figure 3. However, the feature-pairs detected between similar perspectives are not all correct, and outliers must be removed before triangulation can occur. One method is to use a symmetry test to ensure the features found in image pair symmetrically point to each other, as these features were matched independently of one another. The results of the symmetry test are shown in Figure 4.

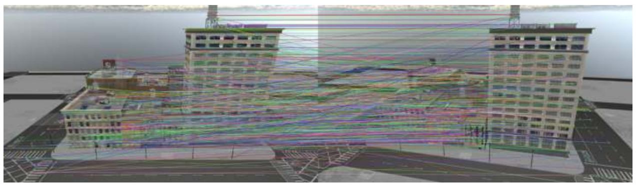


Figure 4. Paired image features between two different perspectives after Symmetry test

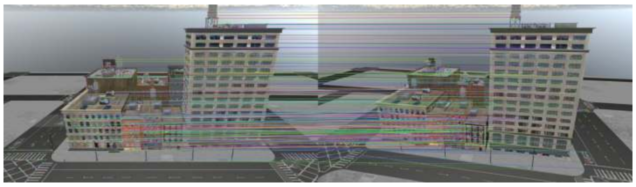


Figure 5. Removal of outliers after applying RANSAC

Random Sampling Consensus (RANSAC) to Find Essential Matrix

Even after the symmetry test, it is assumed that a considerable amount (approximately half) of all matched keypoints are considered outliers, which makes it difficult to determine a line of best fit when it comes to calculating the essential matrix[18]. However, the correct matrix can be approximated iteratively through Random Sampling Consensus (RANSAC) by testing a random sampling of possible essential matrices for a pair of perspective projections. As a result, an essential matrix is also estimated for the camera pairs which can be used to further remove outliers in the paired perspective projections as demonstrated in Figure 5. In reference to Figure 5, all features are still present from the older images, but only matches that have been deemed as inliers now have corresponding randomly colored lines drawn between them.

Extracting Depth Information

Epipolar Geometry is the mathematical constraint used to extract depth information from perspective pair images and is the next step in performing point triangulation of keypoints in SfM[19]. Typically used to recover the depth information of stereo cameras with known rotations and translations, SfM is able to adapt the concept by treating the two separate images as a stereo camera, so long as the pose between the cameras is known.

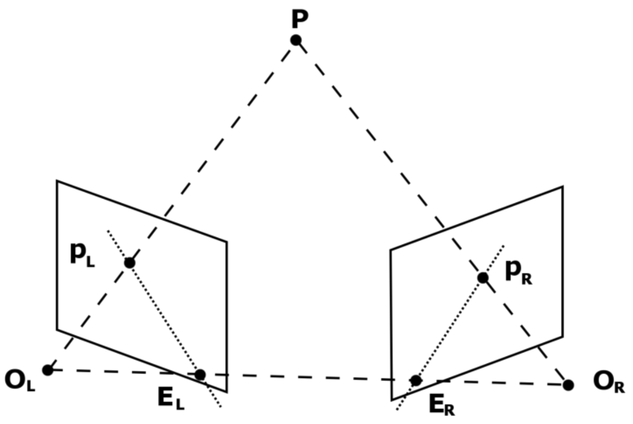


Figure 6. Epipolar Geometry

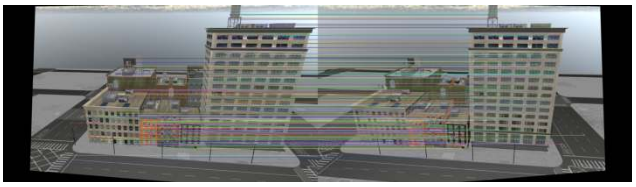


Figure 7. Perspective Rectification to create parallel Epipolar lines

A useful optimization for triangulating points using Epipolar lines is ensuring the lines are parallel, and mathematically aligning the corresponding points such that either the x-component or y-component is constant (row-aligned or column-aligned pixels). This process is known as *rectification* and the result of which is shown in Figure 7. The black borders and warped images embedded in the border visually demonstrate the effects of rectification. The exact process of this mathematical alignment is discussed in greater depth in Section *Homography Transform and Texturing*. The rectification process distorts the perspective pair of images by mathematically flattening the two 3D image planes through a common double-width aperture to produce perfectly parallel Epipolar lines. Once this is completed for each pair of images in the SfM toolset, 3D points can be triangulated. When the sparse reconstruction is complete in the form of a 3D point cloud from epipolar reprojection and triangulation, shown in Figure 8, with Figure 9 showing the truth model for reference.

Further SfM Steps

For the new method proposed by this paper, the remaining steps in SfM are not performed. Normally, after the sparse reconstruction is generated, as denser reconstruction is made[20]. Then the points in the dense reconstruction are efficiently connected to form a 3D mesh which is then

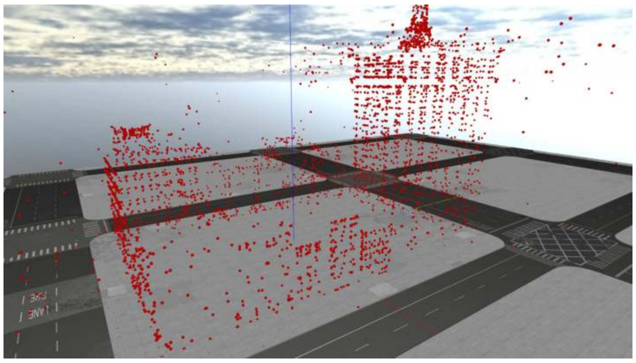


Figure 8. Point Cloud with only ground model superimposed

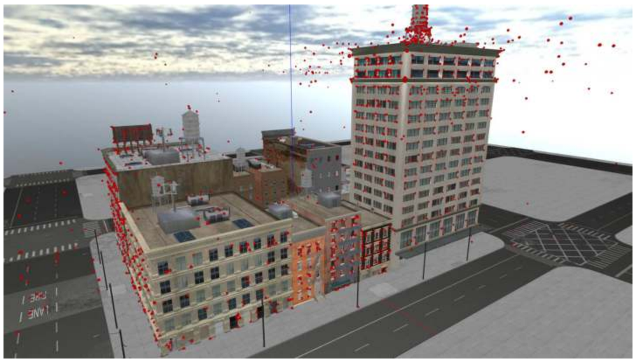


Figure 9. Point Cloud with full truth model superimposed

textured from the original source images to create the final reconstruction[21 22].

Model Size and Orientation

One notable artifact of the SfM process is that all camera poses recovered via RANSAC computations are *relative*, meaning that it is impossible recover the absolute positions of each camera without some external information. As such, the generated models typically have a random orientation and random scale applied to them. This means that these models will require external information if they are to be useful for aerial reconnaissance. Correcting for orientation could be as simple as storing magnetic orientation of the camera, as well as ensuring each picture captured is aligned with a vertical axis based on the direction of gravity. Scale and translation of the image would be a more difficult process, but models could be compared to existing computed geometry for landscapes. Another solution would be to simply tag each camera position with a GPS coordinate, but this approach lacks versatility in a GPS denied environment.

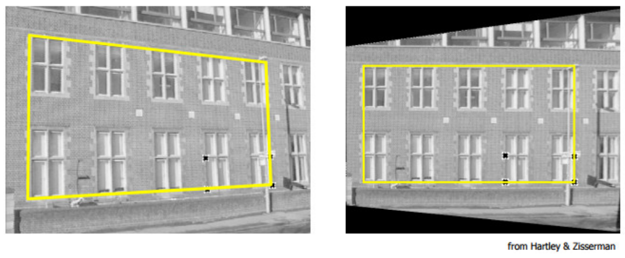


Figure 10. Perspective Removal of Imaged Building Facade[19]

Homography Transformation

Instead of creating a dense reconstruction, refined mesh, and texturing that mesh, our proposed method deviates from the traditional SfM pipeline by replacing these functions with plane detection and Homography transformation of the original imagery. We refer to the combination of these methods as Planar Homography Estimation (PHE), and replaces the model texturing phase of traditional SfM[23 24]. Homography is fundamental to producing visual perspective of a scene, and a Homography Transformation is a matrix transformation that enables the warping of perspectives in images. The Homography is defined as a transformation between two planes[19]:

$$s\begin{bmatrix}x'\\y'\\1\end{bmatrix} = H\begin{bmatrix}x\\y\\1\end{bmatrix} = \begin{bmatrix}h_{11} & h_{12} & h_{13}\\h_{21} & h_{22} & h_{23}\\h_{31} & h_{32} & h_{33}\end{bmatrix}\begin{bmatrix}x\\y\\1\end{bmatrix}.$$

A Homography transformation on a perspective projection image can be used to rectify portions of that image such that they align with a specfic geometry. An example of this process is shown in Figure 10 showing the removal of perspective in reference to the rectangular portion of the building. Notice how the outlined rectangle in the right image is now parallel to the viewer's image plane.

Related Work

This paper is a direct continuation of work performed by Roeber[3] who demonstrated the use of SfM as a method of data compression for aerial imagery. Roeber's results were extremely promising, showing a compression of nearly 60% compared to the original set of imagery. The largest two drawbacks in his approach were 1) the physical accuracy of the model, where distortion in the reconstruction was unintentionally introduced and 2) the large computational time required to create a 3D reconstruction from a set of captured images. Our work improves upon both of these shortcomings.

While plane finding was chosen as the primary method to detect planes within the sparse reconstruction, another approach involves attempting to match basic object primitives to the geometry present in the sparse reconstruction[25].

SfM is a larger process that can be broken down into several smaller problems, with each step capable of being solved in many different approaches. Additionally, most freely available SfM pipelines are composed of several smaller programs independently developed and combined as one. No single pipeline contains the best method for generating an arbitrary 3D model from images, each comes with trade-offs. This is best highlighted by the different choices made by designers for different underlying algorithms that accomplish the same task. For instance, in contrast to the Delaunay triangulation method for mesh model creation, the Poisson Reconstruction method for generating a mesh model typically produces a model that is too detailed for aerial reconnaissance purposes, and pipelines using this method would have their defined use-cases specifically tailored to that type of modeling[26].

Another important decision in making a SfM pipeline is which feature/descriptor extractor to use, therefore different analyses of the performance of each is important in choosing which one to use. According to an analysis of various image matching (feature detection) techniques, based on evaluations of Scale-Invarient Feature Transform (SIFT), Speeded-Up Robust Features (SURF), and Oriented FAST and rotated BRIEF (ORB), ORB is the fastest algorithm, but performance is slightly worse than SIFT. ORB is likewise comparable to SURF in most scenarios[27]. These results are also supported by another study of the computation time in OpenCV presented by Raquet[28].

A noteworthy finding from the study is that ORB tends to concentrate on features in the center of the image, while key point detectors for SIFT and SURF are distributed throughout the image. Another noteworthy finding is that ORB has a tendency to degrade in its feature detection capability when images forming angles larger than 45 degrees are used, therefore it is important for this algorithm that the source images come from sets of images with small angular offsets between each image pair.

The SfM toolchain used for comparison against the SfM/PHE method proposed by this paper is a pipeline which combines the OpenMVG toolchain, used for feature matching, extraction, camera pose recovery, and sparse reconstruction, with OpenMVS, which generates and textures the final reconstruction model finishing the SfM pipeline[29],[30]. These two tools are freely available, and relatively straight-forward to use. These toolsets were chosen over the ones used by Roeber[3] because they can be seamlessly integrated through a python script, and can produce a full reconstruction without any manual input. However, the final model seems to be more prone to error, as the algorithm does not properly trim the model to the object of interest due to interference from the virtual

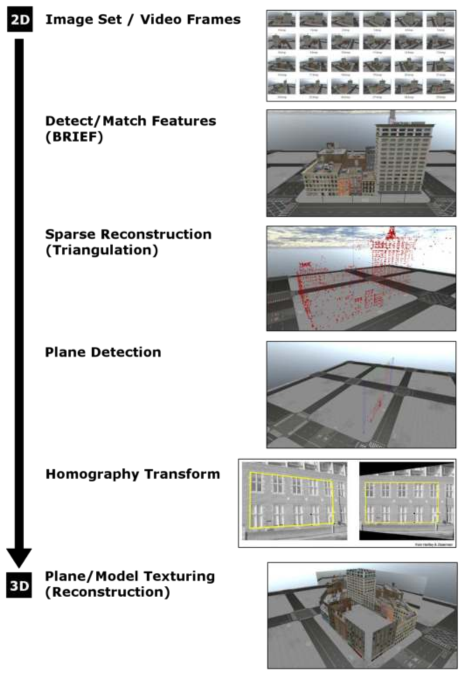

2D
Image Set / Video Frames
Detect/Match Features (BRIEF)
Sparse Reconstruction (Triangulation)
Plane Detection
Homography Transform
3D
Plane/Model Texturing (Reconstruction)

Figure 11. SfM with PHE Overview

world's skybox, which consists of similar textures rendered as a static box around the camera based on the configured view distance in the virtual world.

Methodology

Structure from Motion with Planar Homography Estimation

Structure from Motion (SfM) in its current implementation is limited by its computational complexity, requiring up to 30 minutes to process 90 images in one instance[3]. We have rebuilt an SfM pipeline with real-time computation as the primary requirement, with the aim to transmit the generated 3D models in lieu of the live video with the hope to capture the bandwidth savings of traditional SfM. For this new variant, we employed the AftrBurner Engine[4] to leverage its OpenGL-based rendering[5] as well as its integration with the OpenCV[6] library. Together this software platform can render virtual imagery, generate 3D models from the images, and quantify the reconstruction quality against the known truth model. All this can be done in a rapid, deterministic manner as shown in this paper's corresponding video[7].

To achieve these speed improvements, the sparse reconstruction is constructed in a similar manner as traditional SfM, but leverages the FAST feature detector and the BRIEF feature descriptor. Based on previous analyses, FAST and BRIEF show faster computation-time compared to other methods[27 28]. Next, we deviate from SfM by using a combined technique to detect planes within the sparse reconstruction and texture these planes through a process we call Planar Homography Estimation (PHE).

The planes are deteced via a RANSAC process, and then a homography matrix is computed that relates the 3D points of the detected plane projected onto an XY-plane with 2D coordinates of the features detected in the original images. The homography is then applied to one of the input images and this newly rectified image is textured across the plane. This process is repeated for each subsequent view. For an overview of the entire SfM with PHE process, reference Figure 11.

Operating Assumptions

Because the method is designed to run in real-time, the variant will not have access to future images, so it is designed to work sequentially by only processing the feature matches found in the current and most recent images. This has the benefit of cutting out the lengthy $O(n^2)$ process necessary to match an unordered list of images used in traditional SfM pipelines.

To facilitate ease of comparison and to eliminate a source of error in the new reconstruction, it is assumed the exact camera position is known in model space within the AftrBurner visualization engine. The errors in determining camera pose from a set of features is well-characterized. The process of making an estimation of the camera pose from the essential matrix is on the magnitude of single milliseconds so eliminating that calculation will not have a dramatic effect on the final computation time. Retaining access to the exact position of the camera also eliminates all issues with scaling and orientation of the reconstruction with respect to the virtual world, as all calculations can now be performed with the absolute translation and rotation. This further streamlines testing as a manual alignment of the reconstruction with the truth model is no longer required.

The proposed method is a proof of concept, as such, it is currently only designed to handle specific types of truth model geometry. The method assumes that each model contains vertical walls, common to that of man-made structures, specifically because this algorithm is purpose-built for imaging man-made objects. Aerial surveillance is commonly performed in urban environments, so the assumptions applied to this model would be applicable for a surveillance regime that specifically targets man-made structures at human scale. However, this limitation is self-imposed and not representative of the ultimate capability of the proposed method, but merely as an approach to limit the scope of initial evaluation. Even still, with this limitation, detail is not lost in the final reconstruction.

Finally, based on initial camera calibration testing, the virtual camera used in the AftrBurner engine is well characterized, and to save time between tests, the intrinsic parameters and distortion coefficients are manually stored in the source code.

Virtual Environment

When evaluating a new navigation-based algorithm, the ultimate goal is typically for real-world deployment; however, developing and testing in the real world brings a host of challenges including how to verify truth data, how to enforce flight path reproducibility, and how to close the loop for real-time decision making when data is only collected for post processing, that is, no live data. Since this work focuses on early algorithmic development, these challenges can be postponed by using a virtual simulation environment. Such an environment offers solutions to all aforementioned challenges at the cost of synthetically generated sensed data. The closer this generated data approximates real data, the more accurate and predictive the virtual environment becomes. One defense-oriented virtual solution that has been used in several instances is the AftrBurner Engine. Kim et al.[31] used AftrBurner to generate monocular imagery for an optical-flow based UAV navigation algorithm and was able to successfully use an embedded autopilot to navigate based on the virtual imagery. Lee et al.[32] employed AftrBurner to simulate stereo imagery for relative flight formation in the context of automated aerial refueling; his work presented results showing the predictive ability of stereo virtual imagery to be within centimeters of real stereo camera imagery capturing similar geometry. Parsons et al.[33] achieved similar results generating synthetic imagery for a vision-based proof of concept algorithm prior to several follow up test flights at Edwards AFB. The test flights confirmed Parson's algorithmic viability and predictive ability of AftrBurner. Johnson et al.[34] merged synthetic imagery with synthetic satellite ephemerides to build an extended Kalman filter (EKF) within AftrBurner for close-flight formation. Additionally, the AftrBurner Engine has been used to generate virtual imagery used to develop two US Patents[35,36] – one for an instrument to detect wake-turbulence during flight and another to drastically compress an automated aerial vehicle's remote sensing data stream.

In our case, a virtual environment is the preferred method to test this new SfM implementation. Being able to quickly and accurately generate image projections from any perspective with an intrinsically known truth greatly decreases development time and cost. Deterministically repeatable data collections enable homogenous comparisons across differing algorithmic implementations. There are many barriers to performing this type of experiment if limited to real-world hardware, such as obtaining licenses and

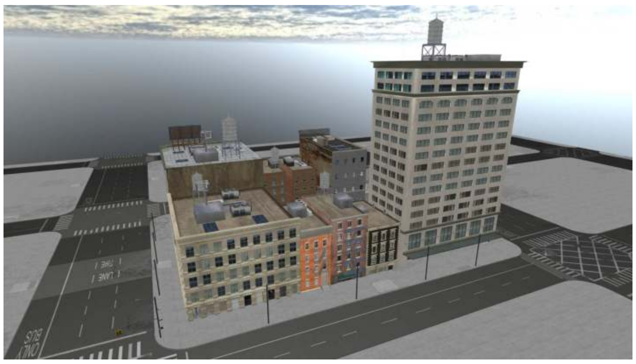


Figure 12. Truth Model rendered in AftrBurner

permissions to operate a UAV (Unmanned Aerial Vehicle) to collect aerial images, and errors introduced through poor navigation that would cause each flight to slightly differ, thereby harming reproducibility. Additionally, with a purely software solution, it is easier for others to verify the work demonstrated. Therefore it is more efficient to perform preliminary testing in a virtual-world to refine the process.

The scene used for experiments represents a city scape, with buildings ranging from a simple tall building to a complex set of eleven buildings on a single block. This is designed to test a variety of scenarios that may be present in an urban environment with many structures close together with varied geometries and urban environments with buildings that maintain larger separation. All of these building are box-like, with some containing more complicated geometries, but all buildings imaged have vertical faces, which will be the target of the proposed SfM variant. This creates a varied test environment where the accuracies can be compared to that of traditional SfM, as well as to the original truth model. Figure 12 shows the geometry of the most complex truth model that will be tested. This truth model is similar to Roeber's demonstration of SfM as a data compression method[3], where the only difference is the ground model has been aligned with the building.

Lastly, the image collection apparatus is represented by a small UAV model flies around the test environment and captures virtual images of the truth model. The camera is simulated based on the pinhole camera model.

Input Data Collection

A circular flight path that orbits a central point was chosen as the method of data imagery collection with 30 evenly-spaced images. This serves the purpose of ensuring that every face of the model is imaged equally. This creates a best case scenario for creating a well-matched point cloud, which can be used to detect vertical walls, represented as vertical planes in the scene. The flight path is inspired by

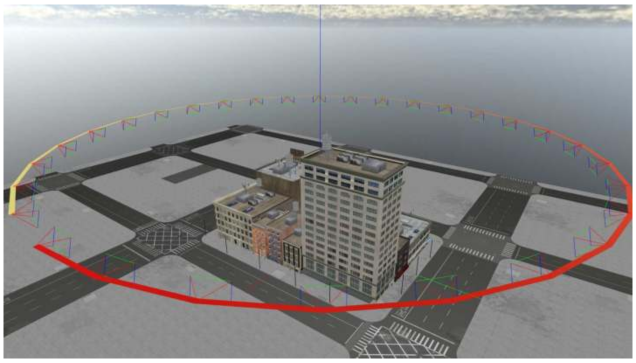


Figure 13. Standard orbital flight path with camera frustums

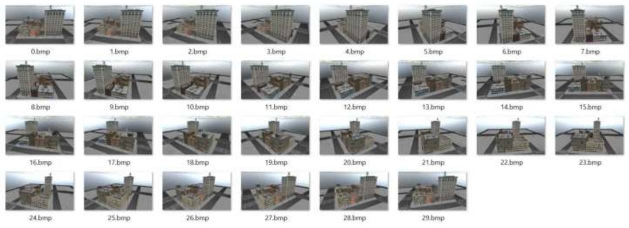


Figure 14. Example of images collected by virtual camera to be used for feature matching and plane texturing

the previous work of Roeber and Ekholm, who used similar flights paths to gather imagery of the truth model[3 37].

However, this flight path differs from Roeber in the sense that it is a purely circular orbit with no spiral pattern. The flight path is a predefined orbit with parameters for the altitude (meters), radius (meters), and the number of images collected. Additionally, the orbit requires a desired target, which in this case is defined as the point of surveillance for each captured image. Figure 13 shows a standard flightpath around a truth model, and each perspective projection is represented by a camera frustum originating from a point along the flight path. Figure 14 shows an example of captured images using the described orbit.

OpenCV Point Triangulation

From the row-aligned features, we triangulate those matches and create a sparse reconstruction in the form of a 3D point cloud. This is done through the cv::triangulatePoints function in the OpenCV library along with some additional logic to convert between the OpenCV reference frame and AftrBurner visualization engine. Point triangulation is a derivative of stereo imaging, as such it relies on the past work of Trucco and Verri[38], Hartley and Zisserman[19], Forsyth and Ponce[39], and Shapiro and Stockman[40]. Figure 15 shows the result of the point triangulation in the form of the sparse reconstruction. With this, we now have the general size and shape of the structure.

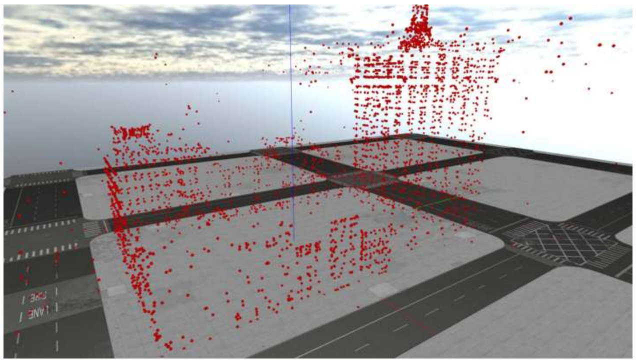


Figure 15. Reprojected Points without Model

Plane Finding

Using the sparse reconstruction, we approximate the geometry of the scene with large generalized vertical planes. To find planes within each feature matching set, using a unique RANSAC process, our algorithm randomly picks three starting points and counts how many points fit along that plane. Checks are made to avoid collinearity amongst the 3 points.

Other intrinsic errors are introduced in the process of extracting the 3D geometry of the scene, and the algorithm accounts for these variations through a threshold to determine coplanar and colinear points. These thresholds are dynamically computed based on given inputs for the size of the scene imaged. This is done by computing two vectors, V_1, V_2, from the three points, p_1, p_2, p_3, and comparing the magnitude of the cross product of these two vectors to ensure they are greater than the given colinear threshold, σ_{linear}. Ideally, σ_{linear} would be zero, but to account for small variances in the data, it is calculated with respect to the size of the detected plane. Mathematically, a good plane is determined as

$$\begin{aligned} V_1 &= p_1 - p_0 \\ V_2 &= p_2 - p_0 \\ \sigma_{linear} &= \tfrac{|V_1|+|V_1|}{2} \end{aligned}$$

where

$$|V_1 \times V_2| > \sigma_{linear}.$$

Once three phsyically separate points are chosen, then the number of points that are members of the chosen plane are counted. This is done by computing third vector, V_3, as the difference of p_3 and p_0. Ideally, a plane will always have a volume of zero, however, in practice it is highly unlikely than any experimental point will perfectly align with the rest of the plane. Therefore, the computed volume from three distinct vectors from a common origin can be used to verify that the tested point is near the randomly chosen

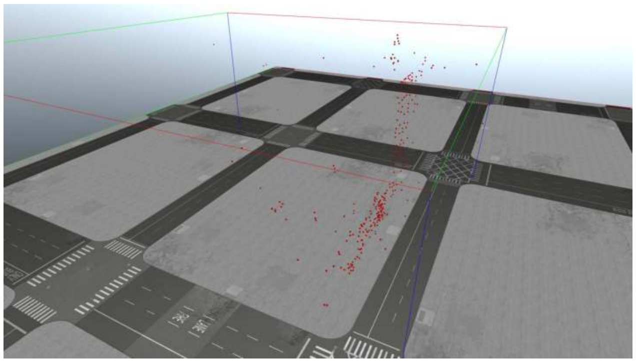


Figure 16. Point cloud of one perspective before plane detection

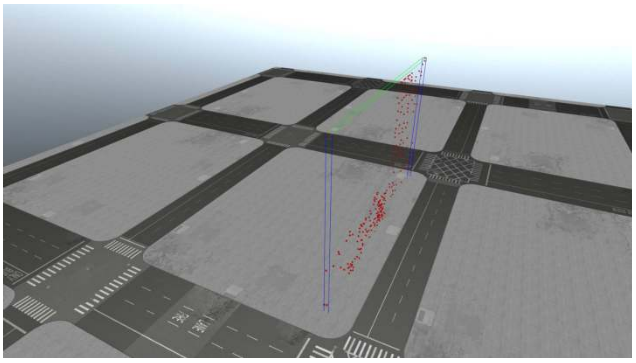


Figure 17. Point cloud of best plane detected from one perspective projection

plane. If that volume is within some σ_{planar} of zero, then it is considered a member of that plane.

$$V_3 = p_3 - p_0$$
$$\sigma_{planar} = |V_1 \times V_2|$$
$$(V_1 \times V_2) \cdot V_3 > \sigma_{planar}.$$

This RANSAC plane detection process is repeated for twice the number of 3D points present in the point cloud to ensure the accuracy of the detected plane. Each iteration checks for the plane with most members and stores the largest set as the *best plane*. Figures 16 and 17 show the results of this process. The bounding boxes for the point clouds are enabled and demonstrate how large the original point cloud is before the plane detection occurs.

Next, a check is made to ensure the detected plane is a vertical wall. This is done by comparing the z-value of the computed normal vector, N, of the plane (computed from the normalized $V_1 \times V_2$), and verifying that it is near 1. In the case of the method, a z-value greater than 0.95 in the normal vector is accepted.

Next, the displacement component of the plane, P, is computed as the maximum dot product of a member point, P_m, and the normal vector, satisfying the equation for a point that lies in a plane,

$$(A, B, C) = (N_x, N_y, N_z)$$
$$D = \max(p_m \cdot N)$$

$$Ax + By + Cz + D = 0.$$

Finally, the best plane detected by RANSAC is compared against previously detected planes from different perspectives to determine if it a duplicate. Further computations for a plane that has been previously detected in the scene is unnecessary unless the newer detected plane is better than previous ones. First the normal vectors for the candidate plane, C_n, and each previously detected planes, C_{n-1}, are compared by computing the magnitude of the combined vectors versus the individual magnitudes of each vector. If the magnitudes are within 90% of the length of the two vectors combined, then they are considered similar. Next, if the displacement values, C_d and D_d, are close to each other within some threshold, σ_{dist}, the two planes are deemed similar.

$$\sigma_{dist} = (\max(P_m) - \min(P_m)) \cdot 0.025$$
$$|C_n + D_n| > (|C_n| + |D_n|) \cdot 0.90$$
$$|C_d - D_d| < \sigma_{dist}.$$

However, if the candidate plane contains more 3D points it will be added to the list of detected planes and replace the previous plane that is similar to it.

In this implementation of the method, only one plane is detected between two perspective projection pairs. It could be altered to detect further smaller planes present in each pair, but for simplicity of testing, only the most prominent plane in each pair is needed.

The RANSAC plane finding method used for our proposed variant is but one alternative. Another common approach involves attempting to match basic object primitives to the geometry present in the sparse reconstruction[25].

Homography Transform and Texturing

Next, we create a texture to apply to the newly discovered planes. We use the cv::findHomography function call to compute the homography transform between two sets of points[41], however, the point sets need to be represented as a 2D array of xy-points in an image. The first set can be directly derived from previous computations where the 2D feature keypoints were computed in the image, but the second set must come from the 3D points. In this instance, the 2D feature keypoints represent the previously warped perspective, and the 3D points can be altered to represent a perfect planar projection of a desired area in an image. In this case, the desired area is the vertical face of a building, described previously as the detected plane in the sparse reconstruction.

For this the 3D points from the plane can be mathematically reprojected as 2D members of a rectified xy-plane, P. First the x and y vectors for the plane are computed. Because the planes represent vertical walls, for this purpose they are assumed to have an up direction of

$$P_y = (0, 0, 1),$$

thus the x-component can be computed as the cross product of the normal vector of the plane and the y-component,

$$P_x = P_n \times P_y.$$

Next, the corners of the 2D projection are derived from the 3-space min and max points present in the detected plane, where the bottom left and top right corners, $C_{BottomLeft}$ and $C_{TopRight}$, of the 2D plane is computed as

$$C_{BottomLeft} = (P_x \cdot P_{min}, P_y \cdot P_{min})$$
$$C_{TopRight} = (P_x \cdot P_{max}, P_y \cdot P_{max}).$$

It is unknown exactly how the plane is oriented in 3-space, so a simple sign modifier for the x and y components of each 2D points is computed as

$$M_x = \frac{C_{TopRight_x} - C_{BottomLeft_x}}{|C_{TopRight_x} - C_{BottomLeft_x}|}$$
$$M_y = \frac{C_{BottomLeft_y} - C_{TopRight_y}}{|C_{BottomLeft_y} - C_{TopRight_y}|},$$

which is then applied to the respective components of the corners and to the 2D projection of the remaining points in the detected plane. The 2D projection for each point, p, in the detected plane, P, is

$$p_x = M_x \cdot (P_x \cdot P_m)$$
$$p_y = M_y \cdot (P_y \cdot P_m)$$

Next, the scale and offset for the projected points need to be computed as they exist in a different reference frame from that of the original perspective projection pair. This is done by computing vectors, v_o and v_p, for the *original* keypoints, k, and the *projected* 3D points, p, of the detected plane.

$$v_o = k_0 - k_1$$
$$v_p = p_0 - p_1$$

where the scale, s, and the offset, o, are

$$s = \frac{|v_o|}{|v_p|}$$
$$o = k_0 - s \cdot p_0$$

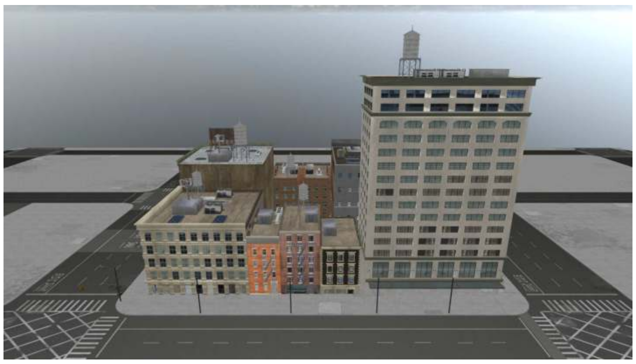


Figure 18. Before homography transformation

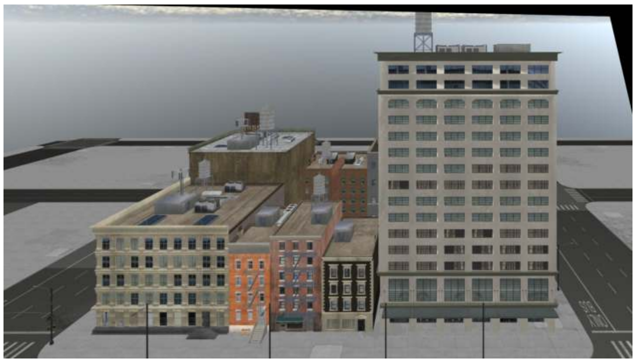


Figure 19. After homography transformation

The scale and offset can then be applied to all the points in the detected plane. At this point, the two sets of points are ready to be used to calculate the homography matrices for the left and right perspective projections.

Then we apply the computed homography to the original images, which warps the perspective of the original image as a fully-rectified texture. Figures 18 and 19 shows the results of the homography transformation to visually align the detected plane with the viewer's image plane. The black borders and warped images embedded in the border visually demonstrate the effects of rectification.

Finally, the image is then cropped to the size of the detected plane and used as a texture for the quadrilateral object in the reconstruction.

Plane Refinement

If a better plane is discovered that occupies the same location as another plane, it must be replaced. A simple method that only replaces the reconstruction quadrilateral, d, with the candidate quadrilateral, c, when they overlap is used.

A similar check as the plane detector to find similar planes among the plane that is to be added to the reconstruction model,

$$\begin{gathered}\epsilon_{dist} = (\max(P_m) - \min(P_m)) \cdot 0.025 \\ |c_n + d_n| > (|c_n| + |d_n|) \cdot 0.90 \\ |c_D - d_D| < \sigma_{dist}.\end{gathered}$$

The above equation determines if a candidate plane and an existing plane share similar normal vectors and displacement components. If this is the case, then an additional check is performed to verify that the planes themselves overlap.

$$\begin{gathered}x_\chi = c_x \cdot d_x - c_{width} - d_{width} \\ y_\chi = c_y \cdot d_y - c_{height} - c_{height}\end{gathered}$$

If either x_χ or y_χ are less than zero, then the two planes intersect and the algorithm defaults to the candidate plane replacing the current plane in the reconstruction.

Once all the perspective projection pairs have been processed, the reconstruction for SfM with PHE is complete.

Generating Traditional SfM Reconstruction for Comparison

For an evualation of the efficacy of using SfM with PHE, a traditional SfM Reconstruction is also generated to be used for comparison. This is done using the OpenMVG and OpenMVS pipelines linked to each other through a python script[29],[30]. The exact same input images are used in this pipeline, using the same intrinsic camera parameters.

Care is taken to ensure that the alignment of the traditional SfM reconstruction matches the truth model for the best possible spatial comparison.

Evaluation Criteria

Multiple comparisons are made for each data set. First a data size comparison between the storage size of the input images, the size of traditional SfM reconstructions, and the size of the SfM with PHE reconstructions will be done. This allows us understand bandwidth transmission needs for real-time UAV scanning on target.

Next, a computation time analysis is performed, measuring the length of time required to complete a traditional SfM reconstruction and a new SfM PHE reconstruction. This allows us to understand the size, weight, power, and cost requirements to run this algorithm on a UAV.

Finally, both the traditional and PHE reconstructions are assessed for their spatial accuracy by using virtual georeference points, which has also been used in related works[3 42 43]. The georeference points are manually chosen key points in the truth model, and the same points are mapped similarly to both reconstructions. A sum-of-squared differences is used as the error metric, where a lower number corresponds to better spatial accuracy. For this experiement, the XYZ error is tested.

Results and Discussion

Hardware/Software Environment

The experiments were conducted on a Thinkpad P51 laptop similar to the hardware used by Roeber[3]. The laptop employed an Intel i7-7820HQ CPU clocked at 2.90GHz, with physical 4 Core, 8 Logical Processors using Intel Hyperthreading, and 16 GB RAM. OpenMVG and OpenMVS support multi-threaded processing, so they will have a small advantage over the SfM PHE method which currently does not implement multi-threaded computations. However, a future implementation of this method could be capable of multi-threading. Traditional SfM also uses smaller texture sizes for various parts of the reconstruction geometry, whereas SfM with PHE still uses full size images.

Datasets

Each dataset assumed only one circular orbit for image collection in contrast to Roeber's pervious work where multiple helixical orbits collected more imagery than our experiments[3]. The datasets generated in these experiments are similar to an actual reconnaissance flight path. Because of this, noticeable degradation in the model accuracy of the traditional SfM reconstructions was observed. Work done by Roeber demonstrated that the exact flight paths did not have much of an effect on the final model.

Each flight path collected images along a fixed radius around each model. For instance the Arab House flight path is an orbit around a smaller building at a lower altitude – effectively testing what the algorithm might produce when a surveillance platform is used at the same elevation as the observed structure. Whereas the Tall Building and the City Block models demonstrate a flight path from a reconnaissance platform flying high above a city. No specific flight path was chosen because it was the best one for that environment, but merely chosen based on the best ability to capture the entire object in one image.

Geopoints were collected manually by finding easily identifiable features between the truth model and reconstructions. Typically, this meant various windows and building corners present in the scene. An example of this process is shown in Figures 20 and 21. Yellow and orange points are visible in these pictures, with yellow indicating the geopoints for the reconstruction, and orange for the truth model. In this example, the orange geopoints are very difficult to see as they line up near perfectly with the yellow geopoints points.

Discussion

Comparisons on the reconstructed models' data size, computation time, and spatial accuracy were made. For every reconstruction with and without PHE, the size of each model was significantly smaller than the original

Table 1. Storage Size Comparison

Scene	Dataset	Trad. SfM	Imprvmnt	SfM w/ PHE	Imprvmnt
Tall Building	60,887 KB	5,106 KB	91.61%	1,370 KB	97.74%
City Block	77,096 KB	9,444 KB	87.75%	5,527 KB	92.83%
Arab House	86,317 KB	3,535 KB	95.90%	9,552 KB	88.93%

Table 2. Computation Time Comparison

Scene	Traditional SfM	SfM with PHE	Speed Up
Tall Building	282.7s	7.045s	40.12x
City Block	783.1s	9.842s	79.56x
Arab House	495.3s	5.973s	80.92x

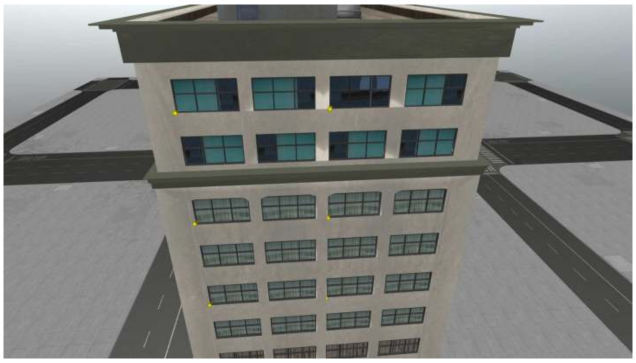


Figure 20. Example of geopoints on truth model, notice yellow dots in low left corners of windows

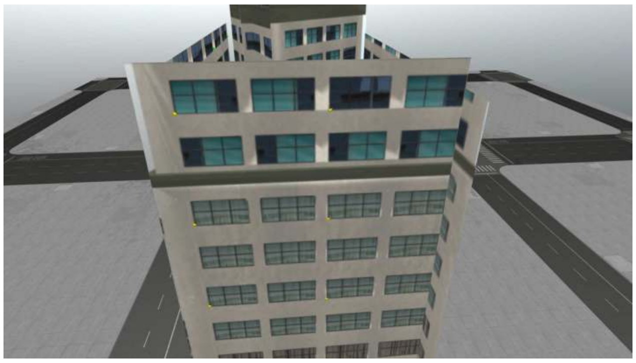


Figure 21. Example of geopoints on PHE reconstruction

image set containing 30 different perspectives. However, PHE differed from traditional SfM in that it performed its task much faster, achieving significant speedups, up to 80x that of traditional SfM. Based on visual inspection, the traditional SfM appeared to be less spatially accurate than PHE, however, for instance where PHE failed to properly find a plane, or where objects behind the plane were included in the texture, PHE accuracy fell. Also, a metric analysis was not performed for the Arab House due to the absence of easily identifiable features to measure on the traditional SfM.

The Arab House proved to be the most difficult for both algorithms to create reconstructions as the virtual model was very simplistic and contained many repeated features, but the PHE reconstruction approximated the underlying geometry better than the traditional SfM reconstruction. Additionally, several of the reconstructions built with traditional SfM had to be recomputed, due to very large distortions present in the reconstruction. This is likely due to the automated algorithm that chooses what is the most prominent object in the scene, and it is likely that the static environmental background was causing these errors. This is supported by Roeber's previous work, where the background was intentionally left blank due to errors it introduced.

Because this algorithm is only designed to work for vertical walls of man-made structures, the roofs of these structures are noticeably missing. This was an intentional design decision when building the algorithm , and because an implementation was not made that estimates the full geometry of an object based on its point cloud. This is a separate component to the problem that is left as future work.

Tall Building

The singular tall building shown in Figure 22 represents a best case for this specific implentation of PHE, as it contains a singular cuboid geometry. Future implementations could include better imaging algorithms to trim planes to the specific geometry of complex objects.

PHE was able to achieve the greatest compression with the Tall Building, compressing the final reconstruction 97.74% smaller than the original 30 image data set, all while doing it in approximately 7 seconds. If use for a video feed is assumed, then for this instance, PHE would be able to process the scene at 4 frames per second.

Even though the full geometry is not represented by the PHE reconstruction, what was represented matched the truth model nearly perfectly, showing little to no error, see Figure 26. More error was likely introduced through the manual process of collecting the geopoints than due to inaccuracies in the reconstruction. In contrast, the traditional SfM reconstruction had a significant lean present in the model, in addition to the model appearing very rough in general. This observation is supported empirically by Figure 25 and through visual inspection of Figures 24 and 23.

For a more in-depth view of the Tall Building model and the resulting reconstruction, please reference timestamps 3:22, 3:31, and 3:40 in our video[7].

City Block

The city block represents a typical use case for aerial surveillance on man-made objects. It contains many buildings with various and complex geometry. In its current implementation, PHE reconstructions might not be fully adequate to replace the video feed for aerial reconnaissance, showing its limitations to only find simple rectangular planes to texture. However, for the areas that are represented in the reconstruction, they are more accurate than that of traditional SfM, which is prone to greater distortions. Additionally, the PHE reconstruction might also be limited by the nature of the virtual model, which contains fewer physical features for the feature matcher to recognize. So the limitations present here may not fully represent its capability with real-world imagery.

None the less, the storage size of the model was still smaller than both the original data set and the traditional SfM reconstruction, PHE was computationally faster as well. This is expected as the reconstruction is composed of 4-6 planes with homographied textures. Similar distortions are present in the traditional SfM reconstruction as shown in Figure 27 and supported empirically by the graphs in Figures 28 and 29, whereas the PHE reconstruction had no such distortions present.

For a more in-depth view of the City Block model and the resulting reconstruction, please reference timestamps 3:25, 3:33, and 4:00 in our video[7].

Arab House

The arab style house demonstrates how PHE might perform with images taken from street level, where the geometry of the roof may not be known. Typically, this is also accomplished easily by traditional SfM algorithms, but in this instance, the model textures were so similar that they resulted in many inaccuracies in the reconstruction process, as such, only a partial model was recovered.

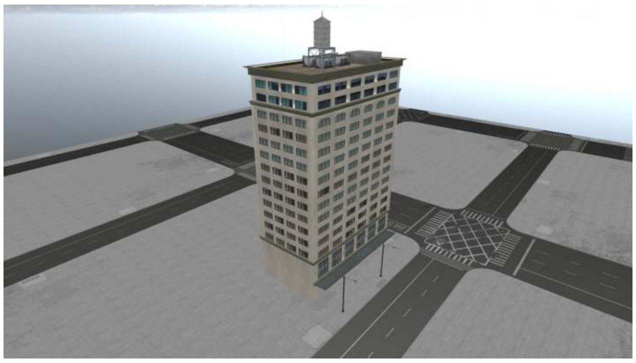


Figure 22. Tall Building Scene with only the truth model visible

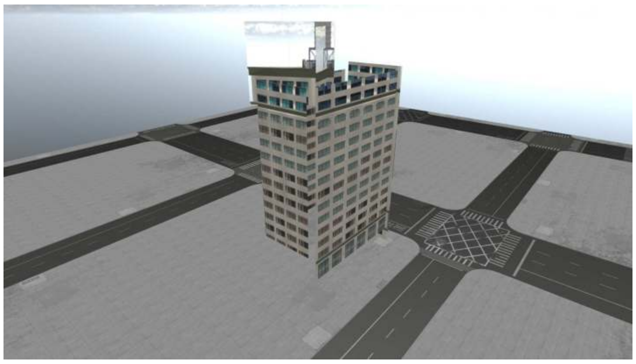


Figure 23. Tall Building Scene with only the PHE model visible

Interestingly, the size of the traditional SfM reconstruction was smaller than that of the PHE reconstruction, however, this may be attributed to the incomplete reconstruction that was generated. Even though the PHE reconstruction was larger than the traditional SfM reconstruction, it was still significantly smaller than the data set, and the computation time of PHE was still much faster than traditional SfM.

For a more in-depth view of the Arab House model and the resulting reconstruction, please reference timestamps 3:28, 3:35, and 4:16 in our video[7].

Conclusions and Future Work

Conclusions

This paper contributes 1) a novel algorithm – Structure from Motion with Planar Homography Estimation (SfM/PHE). This algorithm leverages the intrinsic properties common to many man-made structures, such as vertical walls, to more rapidly and accurately create a texturized 3D model of an environment. As quantified in Table 2, our algorithm is 2) capable of running at 4Hz, approximately 80x faster than traditional SfM approaches using a commodity laptop. 3) Our algorithm outputs a texture-mapped, low-polygon

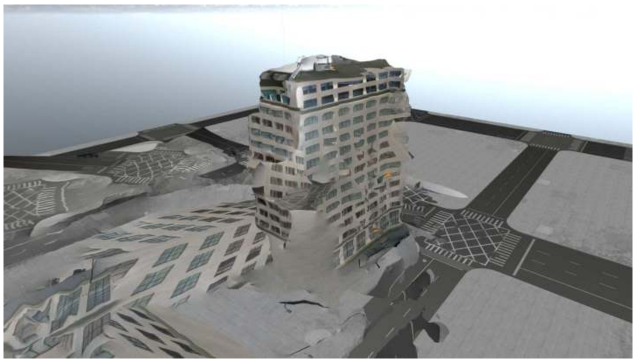
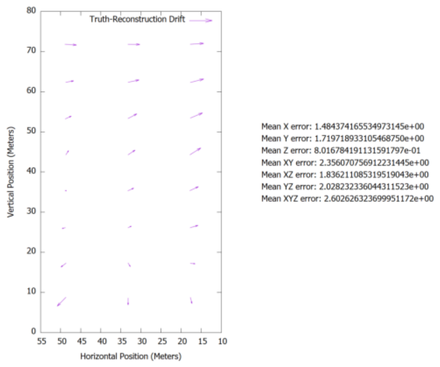
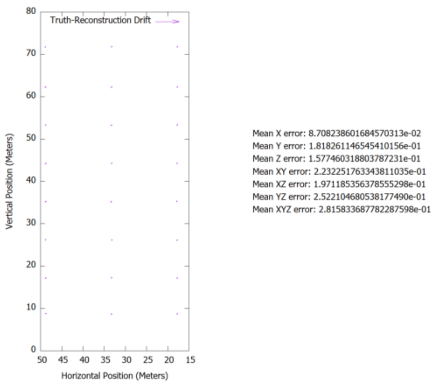


Figure 24. Tall Building Scene with only the traditional SfM model visible



```
Mean X error: 1.48437416553497314 5e+00
```

Figure 25. Tall Building Metric Analysis of Spatial Accuracy of X-face between truth model and traditional SfM



```
Mean X error: 8.70823860168457031 3e-02
Mean Y error: 1.81826114654541015 6e-01
Mean Z error: 1.57746031880378723 1e-01
Mean XY error: 2.23225176334381103 5e-01
Mean XZ error: 1.97118535637855529 8e-01
Mean YZ error: 2.52210468053817749 0e-01
Mean XYZ error: 2.81583368778228759 8e-01
```

Figure 26. Tall Building Metric Analysis of Spatial Accuracy of X-face between truth model and PHE

3D model of the scene in lieu of transmitting an entire image set. As shown in Table 1, this achieves compression ratios 88.9%-97.7% higher than traditional SfM. Finally, 4) the textureized 3D models are as accurate or in our presented cases, up to 10x more accurate than traditional SfM approaches. Our spatial analysis shown in 28 vs 29 and 25 vs 26, shows the accuracy of features lying on our generated 3D model is approximately an order of magnitude more accurate than a traditional SfM approach. The accuracy is high enough that a remote operator could simply click a location on the 3D model, acquire the corresponding global 3D coordinate, and submit that 3D coordinate for intelligence purposes or additional mission vectoring.

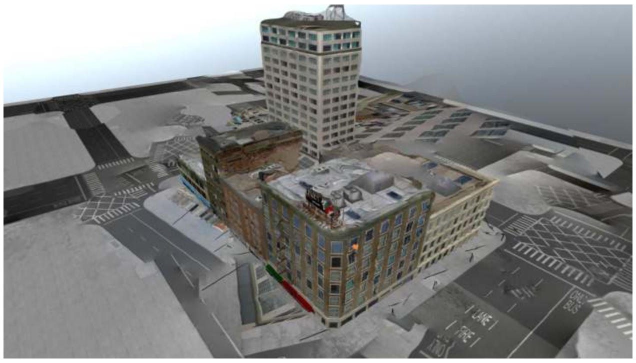
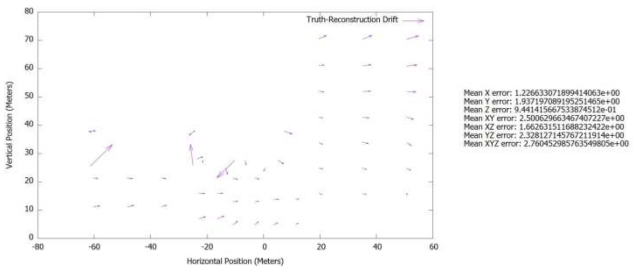
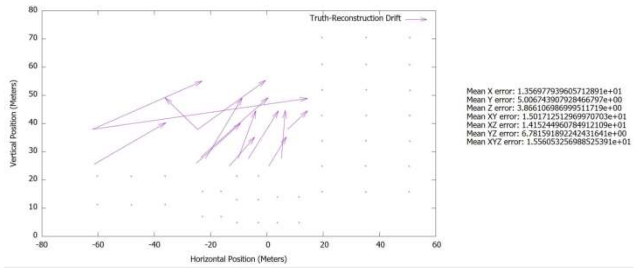


Figure 27. City Block Scene with only the traditional SfM model visible



```
Mean X error: 1.22663307189941406 3e+00
Mean Y error: 1.93719708919525146 5e+00
Mean Z error: 9.44141566753387451 2e-01
Mean XY error: 2.50062966346740722 7e+00
Mean XZ error: 1.66263151168823242 2e+00
Mean YZ error: 2.32812714576721191 4e+00
Mean XYZ error: 2.76045298576354980 5e+00
```

Figure 28. City Block Metric Analysis of Spatial Accuracy of X-face between truth model and traditional SfM



```
Mean X error: 1.35697793960571289 1e+01
Mean Y error: 5.00674390792846679 7e+00
Mean Z error: 3.86610698699951171 9e+00
Mean XY error: 1.50171251296997070 3e+01
Mean XZ error: 1.41524496078491210 9e+01
Mean YZ error: 6.78159189224243164 1e+00
Mean XYZ error: 1.55605325698852539 1e+01
```

Figure 29. City Block Metric Analysis of Spatial Accuracy of X-face between truth model and PHE

This system was built with computational speed in mind, and we were able to achieve operation at 4 Hz providing data similar to traditional SfM algorithms at a fraction of the computation time and space. This demonstrates its potential for use to augment or replace video imagery. This work enables ability to pre-process imagery into a smaller 3D model that is transmitted from a UAV to remote operators.

The speed at which this method was developed would not be possible without the strategic use of virtual environments and a graphical visualization engine to test our new method. The use of a virtual environment enabled the repeatable analysis of information in a deterministic manner, eliminating many sources of error. The virtual world also enabled a cost-effective testing environment, as no physical hardware other than a laptop was needed to conduct each experiment. Additionally, the experimental environment for each test was not subject to real-world problems such as weather and logistical flight coordination to collect aerial photography, as well as the regulatory process that would be necessary to accomplish these tasks.

Many metrics were used to measure the effectiveness for SfM with PHE versus traditional SfM and a standard data set with no size reduction to simulate the requirements for a video feed. From the data collected, SfM with PHE shows promise, but requires further development before it can be used as an alternative to traditional SfM or replace video imagery. However, it does come with the benefits of creating a 3D model that reflects greater spatial accuracy of the scene versus traditional SfM, and is able to create a reconstruction in a timely manner that would be acceptable for aerial reconnaissance, depending on the application. The model created is so accurate, it could also be used to generate GPS coordinates for specific features of the building imaged.

Future Work

There is still much work to do, and many different areas are ripe for improvement. To improve spatial accuracy, further exploration of image processing algorithms to detect the edges of a rectified face of a building could be used. Ideas such as using Hough lines and pixel comparisons between different frames were explored as part of this research, but ultimately not included as an adequate working model was not achieved in a timely manner. Other exploration could look into using the images parallax as a means to determine what parts of the image belong to the detected plane, and what parts of the image lie in front of or behind the rectified plane.

To improve the computational time required for this method further, multithreading strategic areas of the algorithm that perform repeatative tasks could be beneficial, but was not explored as part of this research. Furthermore, it should be possible to accelerate computations via a graphical processing unit in some areas of SfM with PHE. This is possible due to many sections in the algorithm where calculations are performed on a pixel-by-pixel level, and would benefit greatly from vectorized execution.

More varied flight paths should be tested in the future, like a point-to-point flight path that can specifically mimic data collection in an urban canyon as the drone flies over streets between buildings.

An analysis of the energy required to run this algorithm was not performed, which would be important if this is designed to run on a remote device that is reliant on a finite energy supply. However, evidenced by the short computation times, it is unlikely that SfM with PHE would have a dramatic energy usage for an aerial surveillance platform. Furthermore, energy usage could be improved via intelligent implementation of accelerations by low-energy graphical processing units such as an Nvidia Jetson TX-series.

Additionally, SfM with PHE could be improved via mipmapping, reducing the sizes of textures, as the majority of the reconstruction storage size is due to the size of the images. There are arguments to be made that if reducing the image resolution is possible, then it should also be possible with the source imagery. While this is true, having access to the 3D aspect of the reconstruction means that much less texture information is required to convey the same meaning to an intelligence analyst as traditional 2D images.

Further aims to make the algorithm more robust, beyond just identifying and reconstructing vertical walls. SfM with PHE should be capable of handling a variety of geometries, and more importantly, completing the geometry of imaged objects, such that roofs and building corners. Included in this should be determining the feasibility of long-range sensory imagery for SfM, where the remote sensor is hundreds of kilometers from its target, like that of a satellite. Some imaging regimes will not involve a remote asset that is able to get in close with the target object.

Finally, real-world tests must be performed once much of the above is explored or implemented. While traditional SfM has proven effective for real-world objects, and there is sufficient reason to believe that SfM with PHE would perform similarly, it is also important to fully characterize the algorithms behavior and make potential tweaks to handle instances under non-ideal conditions, such as excessive shadows or glare.

References

1. Shannon CE. Communication in the Presence of Noise. *Proceedings of the IRE* 1949; 37(1): 10–21. DOI:10.1109/JRPROC.1949.232969.
2. McMillan J. Why auction the spectrum? *Telecommunications Policy* 1995; 19(3): 191–199. DOI:10.1016/0308-5961(94)00021-J.
3. Roeber J, Nykl S and Graham S. Assessment of Structure from Motion for Reconnaissance Augmentation and Bandwidth Usage Reduction. *Journal of Defense Modeling and Simulation* 2019; 1(1): 1–13. DOI:10.1177/1548512919844021. https://doi.org/10.1177/1548512919844021.

4. Nykl S, Mourning C, Leitch M et al. An overview of the steamie educational game engine. In *Frontiers in Education Conference, 2008. FIE 2008. 38th Annual*. IEEE, pp. F3B–21.
5. Shreiner D and Group TKOAW. *OpenGL Programming Guide: The Official Guide to Learning OpenGL, Versions 3.0 and 3.1*. 7th ed. Addison-Wesley Professional, 2009. ISBN 0321552628.
6. Bradski G. The OpenCV Library. *Dr Dobb's Journal of Software Tools* 2000; URL http://opencv.org/.
7. Arnold C, Nykl S, Graham S et al. Real-time structure from motion with planar homography estimation. https://www.youtube.com/watch?v=Fs5-AaDO21k.
8. Szeliski R. *Computer Vision : Algorithms and Applications*. Springer, 2011. ISBN 1848829345. DOI:10.1007/978-1-84882-935-0. URL http://research.microsoft.com/en-us/um/people/szeliski/book/drafts/szelski{_}20080330am{_}draft.pdf. arXiv:1011.1669v3.
9. Jones AD. Manual of photogrammetry, eds c.c. slama, c. theurer and s.w. hendrikson, american society of photogrammetry, falls church, va., 1980, fourth edition, 180 × 260mm, xvi and 1056 pages (with index), 72 tables, 866 figures. isbn 0 937294 01 2. *Cartography* 1982; 12(4): 258–258. DOI:10.1080/00690805.1982.10438226.
10. Tsai RY. An efficient and accurate camera calibration technique for 3D machine vision. In *Proceedings of IEEE Computer Vision and Pattern Recognition Conference*. ISBN 0818607211, pp. 364–374.
11. Opencv 2.4.13.7 documentation, 2018. http://docs.opencv.org/modules/refman.html.
12. Rosten E and Drummond T. Machine learning for high-speed corner detection. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 3951 LNCS. ISBN 3540338322, pp. 430–443. DOI:10.1007/11744023_34.
13. Moravec H. Obstacle avoidance and navigation in the real world by a seeing robot rover. *tech report CMU-RI-TR-80-03* 1980; : 175DOI:ADA092604. URL https://www.ri.cmu.edu/publication{_}view.html?pub{_}id=22.
14. Harris C and Stephens M. A Combined Corner and Edge Detector. In *Procedings of the Alvey Vision Conference 1988*. ISBN 0364-5134 (Print)\r0364-5134 (Linking), pp. 23.1–23.6. DOI:10.5244/C.2.23. URL http://www.bmva.org/bmvc/1988/avc-88-023.html. 0804.1469.
15. Jianbo Shi and Tomasi. Good features to track. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition CVPR-94*. ISBN 0-8186-5825-8, pp. 593–600. DOI:10.1109/CVPR.1994.323794. URL http://ieeexplore.ieee.org/document/323794/. 0410079.
16. Lowe D. Distinctive Image Features from. *International Journal of Computer Vision* 2004; 60(2): 91–110. DOI: 10.1023/B:VISI.0000029664.99615.94. 0112017.
17. Tseng HY, Wu PC, Yang MH et al. Direct 3D pose estimation of a planar target. In *2016 IEEE Winter Conference on Applications of Computer Vision, WACV 2016*. ISBN 9781509006410, pp. 1–9. DOI:10.1109/WACV.2016.7477640.
18. Longuet-higgins HC. A computer algorithm for reconstructing a scene from two projections. *Nature* 1981; 293(5828): 133–135. DOI:10.1038/293133a0.
19. Hartley R and Zisserman A. Introduction. A Tour of Multiple View Geometry. *Multiple View Geometry in Computer Vision* 2004; : 1–21.
20. Furukawa Y and Ponce J. Accurate, dense, and robust multiview stereopsis. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 2010; 32(8): 1362–1376. DOI: 10.1109/TPAMI.2009.161.
21. Delaunay PB. Sur la sphere vide. *Bulletin of Academy of Sciences of the USSR* 1934; 12(6): 793–800. DOI: 10.1051/jphysrad:01951001207073500. URL http://galiulin.narod.ru/delaunay{_}.pdf.
22. Waechter M, Moehrle N and Goesele M. Let there be color! Large-scale texturing of 3D reconstructions. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 8693 LNCS. ISBN 978-3-319-10601-4, pp. 836–850. DOI:10.1007/978-3-319-10602-1_54.
23. Szpak ZL, Chojnacki W, Eriksson A et al. Sampson distance based joint estimation of multiple homographies with uncalibrated cameras. *Computer Vision and Image Understanding* 2014; 125: 200–213. DOI:10.1016/j.cviu.2014.04.008.
24. Simon G and Berger MO. Pose Estimation for Planar Structures. *IEEE Computer Graphics and Applications* 2002; 22(6): 46–53. DOI:10.1109/MCG.2002.1046628.
25. Garcia S. *Fitting primitive shapes to point clouds for robotic grasping*. PhD Thesis, Skolan för datavetenskap och kommunikation, Kungliga Tekniska högskolan, 2009. DOI:ISSN-1653-5715. URL http://kiosk.nada.kth.se/utbildning/grukth/exjobb/rapportlistor/2009/rapporter09/garcia{_}sergio{_}09131.pdf.
26. Kazhdan M and Hoppe H. Screened poisson surface reconstruction. *ACM Transactions on Graphics* 2013; 32(3): 1–13. DOI:10.1145/2487228.2487237. URL http://dl.acm.org/citation.cfm?doid=2487228.2487237. 1006.4903.
27. Karami E, Prasad S and Shehata M. Image Matching Using SIFT , SURF , BRIEF and ORB : Performance Comparison for Distorted Images. *Newfoundland Electrical and Computer Engineering Conference, IEEE, Newfoundland and Labrador Section* 2015; : 3–6DOI:10.13140/RG.2.1.

1558.3762. 1710.02726.
28. Raquet J. Vision navigation with opencv, 2018.
29. Open mvg library, 2018. https://github.com/openMVG/openMVG.
30. Open mvs library, 2018. https://github.com/cdcseacave/openMVS.
31. Kim[O] K, Leishman[F] R and **Nykl S**. Virtual Testbed for Monocular Visual Navigation of Small Unmanned Aircraft Systems. *The Journal of Defense Modeling and Simulation* 2020; : 1–19DOI:https://doi.org/10.1177/1548512920954545. URL: https://doi.org/10.1177/1548512920954545.
32. Lee[A] A, Dallmann[A] W, **Nykl S** et al. Long Range Pose Estimation for Aerial Refueling Approaches Using Deep Neural Networks. *AIAA: Journal of Aerospace Information Systems* 2020; 17(11): 634–646. DOI:10.2514/1.I010842. URL: https://doi.org/10.2514/1.I010842.
33. Parsons[A] C, Paulson[A] Z, **Nykl S** et al. Analysis of Simulated Imagery for Real-Time Vision-Based Automated Aerial Refueling. *AIAA: Journal of Aerospace Information Systems* 2019; 16(3): 77–93. DOI:10.2514/1.I010658. URL: https://doi.org/10.2514/1.I010658.
34. Johnson[A] DT, **Nykl S** and Raquet[F] J. Combining Stereo Vision and Inertial Navigation for Automated Aerial Refueling. *AIAA: Journal of Guidance, Control, and Dynamics* 2017; 40(9): 2250–2259. DOI:10.2514/1.G002648.
35. **Nykl** S and Mourning C. Wake Turbulence Analyzer for Real-Time Visualization, Detection, and Avoidance, 2017.
36. Mourning C and **Nykl** S. Reduction of Sensor Captured Data Streamed to an Operator, 2017.
37. Ekholm J. *3-D Scene Reconstruction from Aerial Imagery*. Master's Thesis, Air Force Institute of Technology, 2012.
38. Trucco E and Verri A. Introductory techniques for 3-D computer vision, 1998. DOI:10.1103/PhysRevLett.103.010404. URL http://cseweb.ucsd.edu/classes/fa12/cse252A-a/hw4/trucco{_}verri{_}pp178-194.pdf. arXiv:1011.1669v3.
39. Forsyth D and Ponce J. Tracking with Non-Linear Dynamic Models. In *Computer Vision - A modern approach*. Pearson, 2003. p. 398.
40. Shapiro L and Stockman G. *Computer vision and image processing*, volume 3. Academic Press, 1992. ISBN 0130307963. DOI:10.1525/jer.2008.3.1.toc. URL http://www.amazon.com/Computer-Vision-Linda-G-Shapiro/dp/0130307963.
41. Open cv library, 2018. http://www.drdobbs.com/open-source/the-opencv-library/184404319.
42. Turner D, Lucieer A and Watson C. An automated technique for generating georectified mosaics from ultra-high resolution Unmanned Aerial Vehicle (UAV) imagery, based on Structure from Motion (SFM) point clouds. *Remote Sensing* 2012; 4(5): 1392–1410. DOI:10.3390/rs4051392.
43. Koutsoudis A, Vidmar B, Ioannakis G et al. Multi-image 3D reconstruction data evaluation. *Journal of Cultural Heritage* 2014; 15(1): 73–79. DOI:10.1016/j.culher.2012.12.003. arXiv:1011.1669v3.

Authors

Christian Arnold, MSc is a military officer in the United States Air Force. He is interested in 3D computer graphics, computer vision, and space-based applications of computer systems. He received his M.S. Computer Science at the Air Force Institute of Technology in 2019. He received his B.S. Computer Engineering and B.S. Computer Science from the US Air Force Academy in 2017.

Scott Nykl, PhD is an Assistant Professor of Computer Science at the Air Force Institute of Technology. His areas of interest are Real Time 3D Computer Graphics, computer vision, sensor fusion, parallel processing, Interactive Virtual Worlds, and computer networking. Throughout his graduate studies, he authored a 3D Visualization Engine and created avionics-related visualizations earning several national awards including mention in Forbe's "The Greatest Young Inventors In America". He was a National Collegiate Inventor's Competition finalist as well as the Institution of Navigation (ION) graduate student award recipient.

Scott Graham, PhD is an Associate Professor of Computer Engineering at the Air Force Institute of Technology. He received the PhD degree in electrical engineering from the University of Illinois at Urbana-Champaign in 2004. He is interested in the intersection between real physical systems and the computers that control them. Specific areas of interest include cyber physical systems security, embedded computing, computer communication networks, and vehicular cyber security.

Robert Leishman, PhD is the Director of the Autonomy and Navigation Technology ()ANT) Center and a Research Assistant Professor at the Air Force Institute of Technology. His areas of interest surround topics relating to autonomy of small unmanned vehicles and GPS-denied navigation, as well as the interesting intersection of the two. He received a Ph.D. in Mechanical Engineering from Brigham Young University in 2013 with support from the DoD SMART Scholarship.