A voxel-based approach for imaging voids in three-dimensional point clouds

Katie N. Salvaggio, Carl Salvaggio, and Shea Hagstrom

Rochester Institute of Technology, 54 Lomb Memorial Drive, Rochester, NY, USA

ABSTRACT

With recent advances in technologies, reconstructions of three-dimensional (3D) point clouds from multi-view aerial imagery are readily obtainable. However, the fidelity of these point clouds has not been well studied, and voids often exist within the point cloud. Voids in the point cloud are present in texturally flat areas that failed to generate features during the initial stages of reconstruction, as well as areas where multiple views were not obtained during collection or a constant occlusion existed due to collection angles or overlapping scene. A method is presented for identifying the type of void present using a voxel-based approach to partition the 3D space. By using collection geometry and information derived from the point cloud, it is possible to detect unsampled voxels such that voids can be identified. A similar line-of-sight analysis can then be used to pinpoint locations at aircraft altitude at which the voids in the point clouds could theoretically be imaged, such that the new images can be included in the 3D reconstruction, with the goal of reducing the voids in the point cloud that are a result of lack of coverage. This method has been tested on high-frame-rate oblique aerial imagery captured over Rochester, NY.

Keywords: three-dimensional point clouds, voxel, computer vision, voids, holes, occlusion, multi-view imagery

1. INTRODUCTION

Advancements in modern computing have expanded the domains of research into areas previously considered to be too computationally intensive. One area that has benefited from such advancements and seen rapid development in the past decade is the automated generation of three-dimensional (3D) models from imagery. The photogrammetry community has been using aerial imagery for decades to develop topographic maps using stereo techniques, and more recently the computer vision community has developed an interest in phototourism, using Structure from Motion (SfM) techniques.[1]

The objective of scene reconstruction methodologies is to automatically extract as much 3D structure from the imagery as possible, in order to obtain a complete model of a scene. SfM techniques often result in a point cloud, containing individual 3D point locations for a model. In most cases, a point cloud will not suffice as a model as they are generally not directly usable in most 3D applications. The point clouds themselves are converted to some sort of surface model through a process referred to as surface reconstruction. The reconstruction of surfaces from points is not necessarily a straightforward problem, as point sampling and spacing is often non-uniform, positions (and normals if available) are noisy, and some regions are lacking data due to obscuration and accessibility limitations. The focus here is on the voids in the point clouds, areas that are lacking data.

Voids exist in point clouds where multiple views of an area were not included in the input imagery. Voids can also exist because of an occlusion or heavy shadow that blocked the area from clear view, a homogeneous surface that resulted in poor image correspondences, or an insufficient baseline that resulted in poor triangulation. Using a voxel-based approach, a method is presented for identifying the type of void. Collection geometry is used to build a voxel space, analyzing what the cameras can and cannot see such that the voxels can be classified as occupied, free, and unsampled. The voids are manifested as the unsampled voxels, where areas on the free-unsampled voxel boundaries are of particular interest. When determining potential image locations such that the voids are visible, it is important to consider the type of void. While areas that were occluded or did not originally have enough coverage will benefit from the inclusion of more imagery, voids that resulted from texturally flat

Further author information: (Send correspondence to K. N. Salvaggio)
E-mail: kns4906@rit.edu

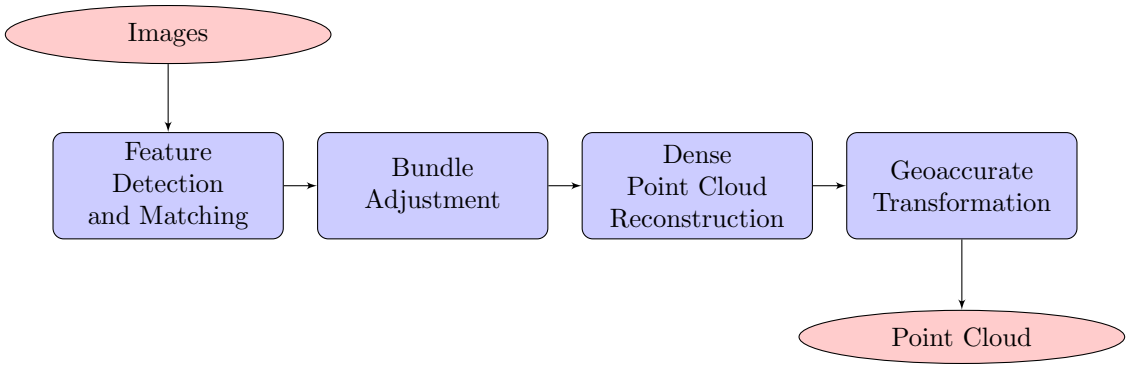


Figure 1. High level diagram of the work flow used to generate dense geoaccurate 3D reconstructions of a scene from 2D imagery.

areas likely will not. Therefore, it is important that these voids are removed from consideration so as not to direct an aircraft to collect repetitive imagery. A line-of-sight analysis is used both to distinguish the type of void and identify potential imaging locations.

This paper is organized as follows: Section 2 provides descriptions of the workflow used to generate point clouds from multi-view imagery (2.1), the voxelization process (2.2), the method used to identify voids that are a result of texturally flat areas (2.3), and the method used to find optimal locations from which to image the voids (2.4). The dataset is described briefly in Sec. 3, preliminary results are presented in Sec. 4, and conclusions and future work are presented in Sec. 5.

2. METHODOLOGY

2.1 Generation of a point cloud using SfM techniques

There are three steps in the automated 3D reconstruction pipeline used to obtain point clouds: (1) image-to-image correspondences are obtained using a feature detection and matching algorithm, (2) camera pose estimations are obtained using the correspondences and a bundle adjustment from a Structure from Motion (SfM) algorithm, and (3) dense 3D reconstructions in the form of a point cloud are obtained using the pose estimation and Multi-View Stereo (MVS) algorithms. A high-level diagram of the workflow used to obtain point clouds is shown in Fig. 1, including a post-processing transformation used to place the point clouds in a geoaccurate coordinate system.[2]

Feature extraction and matching is used to obtain image-to-image correspondences that can be used to estimate camera poses. One of the widely used feature descriptors, and the one employed in the workflow used here, is the scale-invariant feature transform (SIFT).[3] SIFT features are invariant to rotation and scale, and are robust to a range of affine transformations, noise, and some change in illumination. Matches are computed between feature descriptors across all possible image combinations to obtain correspondences.

The image-to-image correspondences are then used to estimate the camera poses with a traditional bundle adjustment. A bundle adjustment, conceived in the photogrammetry community in the 1950s[4] and used more recently in the computer vision community, refers to a large non-linear least squares optimization problem. The objective of a bundle adjustment is to optimize the 3D structure and viewing parameters of a scene, such as the camera position, orientation, and calibration. Non-linear least squares algorithms are used to minimize the reprojection error between the observed and predicted image points, expressed as the sum of squares of a large number of non-linear real-valued functions.[5] The workflow presented here leverages Bundler,[6] written by Noah Snavely and built upon a software package available for generic sparse bundle adjustment written by Manolis Lourakis,[5] to obtain camera projection matrices. The camera projection matrices are equivalent to the photogrammetric ground-to-image function.

A cluster-based method is used in the last stage of the workflow to achieve scalability as the number of images to be used in the 3D reconstructions grows. The set of images is decomposed into clusters using view selection. A MVS algorithm can then be used to reconstruct dense 3D points for each individual cluster, and the resulting

solutions can be merged into a single model.[7] The Cluster-based Multi-View Stereo (CMVS)[7] and Patch-based Multi-View Stereo (PMVS)[8] algorithms, written by Yasutaka Furukawa, are the MVS algorithms used in this workflow to cluster image views and generate dense 3D reconstructions. Note that patch-based methods are only one type of MVS algorithm available, but they are simple and effective and suffice for the point-based rendering desired here.

Up to this point, the methods used reconstruct the scene up to a projective ambiguity (i.e. the scene reconstruction determined is within a projective transformation of a desired world-coordinate system). SfM algorithms are often designed to solve for an unknown number of uncalibrated cameras, whose images were obtained from unknown locations. In the problem being addressed here, it is assumed that the imagery has been captured on an airborne platform, equipped with a global positioning system (GPS) and an inertial navigation system (INS), such that the camera position and orientation is readily available. In this case, it is possible to estimate a mapping between the arbitrary coordinate system of the image-based reconstruction and the desired Earth-based coordinate system.[2,9] This mapping is performed as a post processing step, and the resulting output of the workflow is a dense 3D point cloud in a geoaccurate space.

2.2 Voxelization by line-of-sight

While a point cloud is a simple representation of a 3D model, it does not adequately capture all of the information available. Most notably, it is lacking any type of consideration of free space. If a camera is able to image a certain location, then there must be free space between the camera and that location. Extending this concept to a 3D reconstruction, if a camera is used to reconstruct a point, then the point was visible in that camera frame, and the ray between the camera and that point must therefore be unoccluded or free. This line-of-sight approach provides additional information about the reconstruction, allowing for identification of areas that were not seen, but is impossible to represent in the framework provided by a point cloud.

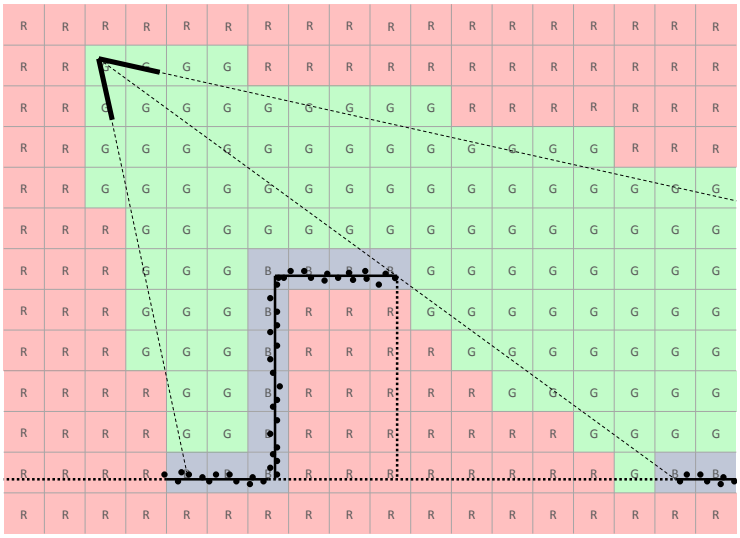


Figure 2. A 2-dimensional visualization of occupied, free, and unsampled voxels. Voxels that contain black points are occupied and denoted with blue (B). The camera is located in the top left and free voxels that rays pass through are denoted with green (G); three rays, denoted with dotted lines, are depicted for reference. Unsampled voxels are denoted with red (R). The scene depicts a rectangular surface, such as a building, on a ground surface; where surfaces visible in the image are denoted with a solid line, and non-visible surfaces with a dashed line.

Voxelization is a method for describing a scene that allows for complex overlapping geometries and can accommodate the idea of free space. Dividing the scene into small volumetric elements on a regular grid, where these 3D sub-volumes are known as voxels, provides a structure with which to better analyze the point cloud information. A ternary system is used in this approach to classify the voxels as occupied, free, and unsampled. Occupied voxels are voxels that contain reconstructed 3D points. Free voxels are those which lie on the rays

between the 3D points and the cameras that were used to reconstruct them. All other voxels are unsampled voxels because they have yet to be sampled and classified as occupied or free. A 2-dimensional (2D) visualization of occupied, free, and unsampled voxels constructed from a simple scene with a single camera is shown in Fig. 2.

Point cloud data is used to initialize the voxel space such that the entirety of the point cloud structure is contained within the space. The voxel space is described with an origin, element size, and number of elements in each direction, and is given by the bounding cuboid of the point cloud. The voxel space itself does not include the camera centers, and assumes that the space between the cameras and the voxel space is unoccluded. Consider an urban scene with the highest point at 300m imaged at an altitude of 3000m; storage requirements for the voxel space increase 10 times in the z-direction alone by including the space between the scene and the camera. The voxels in this space will be either free or non-classified, depending on whether rays from the camera pass through them, but they will not provide any useful information about the missing structure of the scene below. Using just the bounding cuboid of the point cloud (and not the bounding cuboid of the point cloud and camera centers) significantly reduces the amount of memory required for the voxel space when dealing with aerial datasets and speeds up processing time, without significant impact on the results.

For each point in the point cloud, the voxel containing it is identified and marked as occupied. Following this, ray tracing between the cameras used to reconstruct the point and the point itself is used to identify free voxels. This is accomplished using a fast voxel traversal algorithm[10] to identify the voxels intersected by the ray. Any remaining voxels are unsampled, thus resulting in the ternary system for classifying voxels as occupied, free, and unsampled.

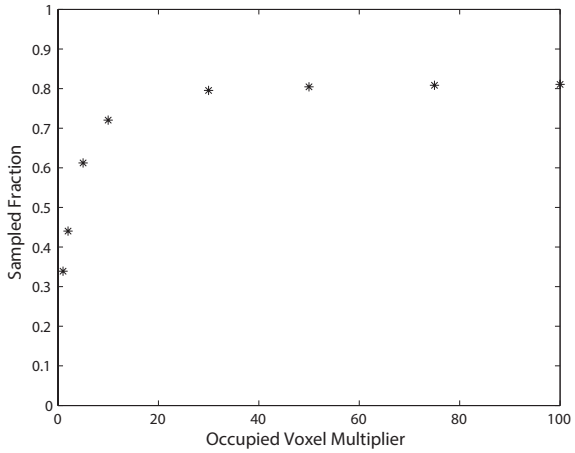
Now consider the fact that a voxel is not infinitesimally small, and therefore it is possible for the finite space occupied by a voxel to contain multiple points, multiple rays, or some combination of points and rays. This is related to the transmission of a voxel, where a voxel with more points would be considered less transmissive, or conversely more occluding, than a voxel with fewer points and more rays passing through it. For simplicity, the voxel space was built with a simple counter such that each voxel is initialized with the counter value set to a value of zero. When a point is added to a voxel, the counter is incremented, and when a ray passes through a voxel, the counter is decremented. Using this system, an occupied voxel is expressed with a positive value and a free voxel is expressed as a negative value. The more positive or negative the value of a counter in a voxel, the more likely the voxel is to be occupied or free. Unsampled voxels contain zero values.

Note that the counter will equalize to zero if the number of points and rays passing through it are the same. Such a voxel is equally likely to be occupied or free. In addition, a single ray will pass through numerous voxels, and therefore a voxel is more likely to be decremented than incremented. Preliminary results using this counting method lacked occupied voxels, despite using a densely populated point cloud. To mitigate this effect, a multiplicative factor was added to the counter increment, thereby incrementing voxels by larger amounts in order to make them less likely to become free. The effect of the occupied voxel multiplier is shown in Fig. 3. As the multiplier increases, the fraction of sampled voxels levels off and no longer changes as the multiplier is increased further. This suggests that voxels that contain points from the point cloud are treated as occupied, regardless of the number of rays that may pass through them, at multiplier values about 50. For the purposes of this research, an occupied voxel multiplier of 10 has been applied.

Finally, there are trade-offs to consider between the size of the voxels, the processing time, and the amount of data to store in memory. It might seem that smaller voxels provide an increased level of resolution in the voxel space, but this is only true if the dataset is sufficiently dense to support the higher resolution. The effect of this is examined in more detail in Fig. 4 and Fig. 10.

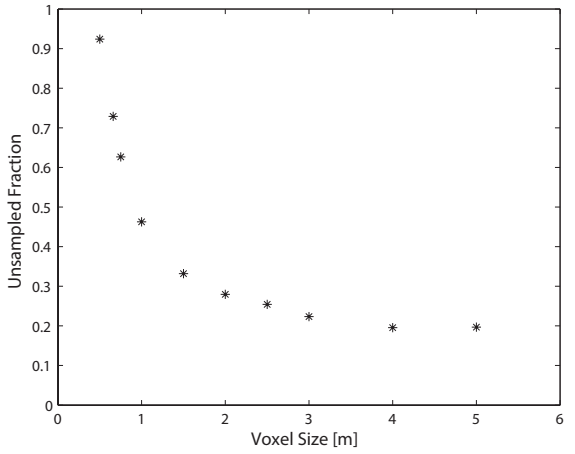
2.3 Identification of Voids in the Voxel Space

The most pertinent information in the voxel space lies in the boundaries between the different types of voxels. Voids are manifested as the unsampled voxels. Given that there are three different kinds of voxels (free, occupied, and unsampled), there are three types of boundaries to consider: the free-occupied boundary, the free-unsampled boundary, and the occupied-unsampled boundary. An illustration of the boundaries in the voxel space is shown in Fig. 5. A voxel face that is shared by a free voxel and an occupied voxel is indicative of known structure in the scene. Similarly a voxel face that is shared by a free voxel and an unsampled voxel is indicative of a



Occupied Voxel Multipler	Sampled Fraction
1	0.33914
2	0.44027
5	0.61225
10	0.72028
30	0.79552
50	0.80432
75	0.80827
100	0.81053

Figure 3. Plot of the fraction of sampled voxel faces versus the occupied voxel multiplier. Note that the sampled fraction levels off, and higher-valued multipliers can be used to ensure that voxels that contain points from the point cloud are treated as occupied.



Voxel Size [m]	Unsampled Fraction
0.5	0.92402
0.66	0.72868
0.75	0.62661
1	0.46273
1.5	0.33201
2	0.27972
2.5	0.25431
3	0.22344
4	0.19546
5	0.19693

Figure 4. Plot of the fraction of unsampled voxel faces versus the voxel size [m]. Note that the number of unsampled voxel faces rapidly increases as the voxel size decreases.

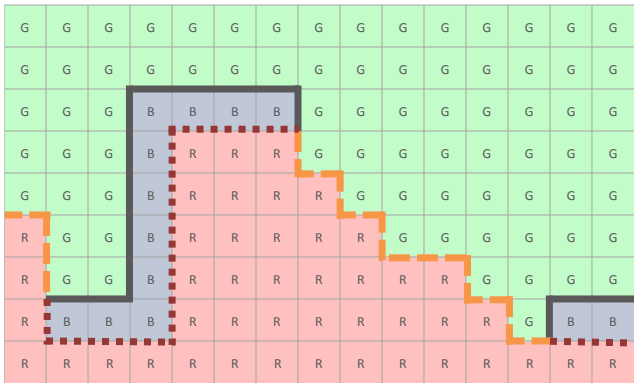


Figure 5. A 2-dimensional visualization of the voxel space with the three types of voxel boundaries marked. The free-occupied boundary is shown as a solid gray line, the free-unsampled boundary is shown as a dashed orange line, and the occupied-unsampled boundary is shown as a dotted red line. The voxel space itself is a subset of the voxel space depicted in Fig. 2.

void or missing structure in the scene. The occupied-unsampled boundary is theoretically complementary to the free-occupied boundary, where the free-occupied boundary will be on the side of the surface model oriented in the general direction of the cameras and the occupied-unsampled boundary will be on the side of the surface model oriented away from the general direction of the cameras. Inclusion of this third boundary does not gain any useful information and thus only two boundaries, the free-occupied boundary and the free-unsampled boundary, will be considered.

It is possible to use the voxel space and a line-of-sight analysis similar to the one used previously to glean more information from the voxel space regarding the voids themselves and optimal viewing locations. Up to this point there has been no distinction made regarding the type of void in the point cloud. Lack of coverage or constant occlusion of an area in the imagery will result in voids in the point cloud, as was stated previously, and the distinction between the two causes is inconsequential because inclusion of more imagery of those areas will be beneficial in both cases. However, it is important to distinguish voids that are a result of homogeneous areas that failed to generate features, because it is possible that these areas appeared in multiple images already, and inclusion of more images will not improve the reconstruction.

It is possible to determine whether the cameras were able to see the voxels on the free-unsampled boundary by using a similar line-of-sight analysis as the one that was used to create the voxel space. Using the center of the voxels and the camera centers, it is possible to trace rays from the voxels to each camera. It is assumed that the content of the voxels is homogeneous, and for the purposes of this initial analysis, partial transmission of voxels is ignored and voxels are assumed to be either fully opaque (occupied) or fully transmissive (free). If the voxels along the ray are all free, then it is assumed that the camera was able to image the area. Similarly, if there is an occupied voxel along the ray, then it is assumed that the camera was unable to image the area. It is unknown whether there is structure within the unsampled voxels, but for the purposes of this analysis, it is assumed that unsampled voxels are also opaque so as to avoid overestimation of the number of cameras that imaged a point. Voxels centers that were seen by many cameras can then be removed from the set of voxels of interest. Benefits of this analysis will be discussed in Section 4.

2.4 Exploitation of Voxel Space Information for Flight Planning Purposes

Now that voids that are a result of lack of coverage and occlusion have been identified, the next step is to determine a method for filling in these surfaces. Due to the nature of surface reconstruction, it is impossible to obtain a model free of unsampled voxels, as there will always be unsampled voxels under the model (e.g. in the interior areas of buildings). The objective then is to obtain a model with minimal unsampled surfaces (e.g. voxels that lie on the free-unsampled boundaries). It is believed that the inclusion of imagery that covers the voids in a reconstruction will result in a point cloud with fewer voids that are a result of lack of coverage in

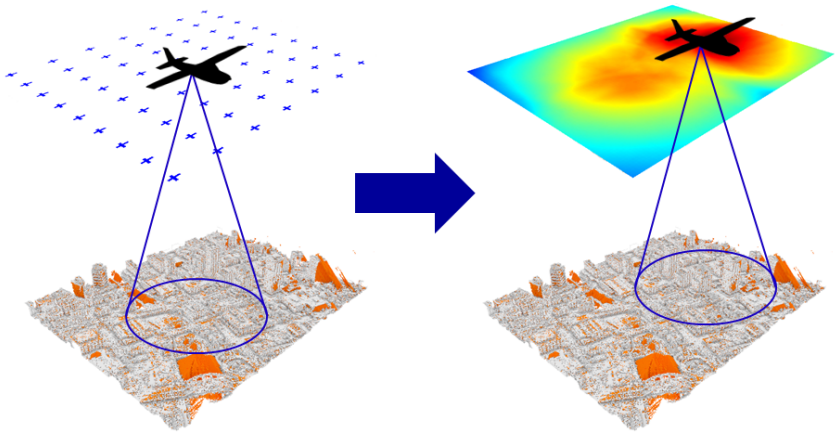


Figure 6. An illustration of the process used to obtain imaging locations. A regular grid of potential positions for the aircraft are shown above the model (left). At each location the number of voxels on the free-unsampled boundary that are visible is computed, which is used to make the heat map (right), where locations shown in red can see more voxels than those shown in blue.

the imagery. Therefore it becomes important to identify potential camera locations that could provide optimal views of the void areas, particularly those that have not been seen previously.

While yet another line-of-sight analysis will prove useful here, identifying these optimal imaging locations is not a straightforward problem. Assuming the problem is constrained to imaging from an airborne platform, the position and pointing angle of the aircraft are all variables that need to be considered. The solution space is therefore large, encompassing an infinite number of combinations of position and pointing angle. For proof of concept, the problem will be constrained initially to a single aircraft altitude and pointing angle, and the locations will be quantized. Quantization of the locations of the aircraft at flying altitude should have little impact on the results, provided that the distance between locations is small, given that positioning of an aircraft in flight is less than exact depending on weather related conditions.

Given an aircraft location and pointing angle, it is possible to determine which boundary voxels a camera placed in that location would be able to see. This is accomplished by leveraging the same type of line-of-sight analysis used previously. Again occupied and unsampled voxels are treated as opaque while free voxels are treated as transmissive, and the camera is assumed to have a circular field-of-view. Repeating this process in different locations at altitude allows for the development of a heat map that can be used to determine optimal imaging locations. An illustration of this process is shown in Fig. 6.

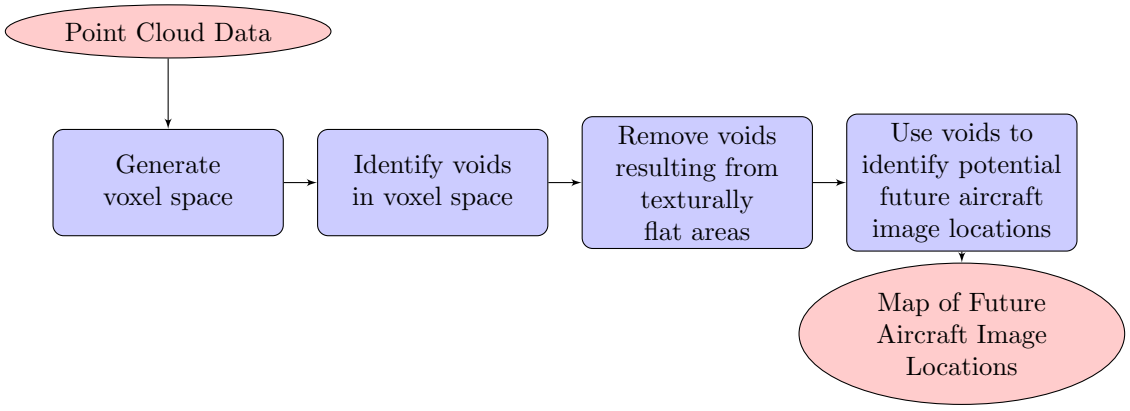


Figure 7. High level diagram of the work flow used to obtain future aircraft image locations from point cloud data.

Finally, an end-to-end work flow diagram of the procedure used to obtain potential aircraft image locations from point cloud data is shown in Figure 7. Using the point cloud as input, a voxel space is generated using the procedure from Sec. 2.2, voids are identified and voids that are a result of texturally flat areas are removed from consideration using the procedure described in Sec. 2.3, and the voids are used to identify potential image locations as described in Sec. 2.4.

3. DATA

Real-world data, collected over downtown Rochester, NY by the Exelis Wide Area Motion Imagery (WAMI) system, was used to demonstrate the voxel-based approach presented here. A sample image sequence highlighting three downtown buildings in Rochester, NY is shown in Fig. 8, where the images have been cropped to just the area of interest. A point cloud derived from a sequence of 48 WAMI images using the 3D workflow is shown in Fig. 9.

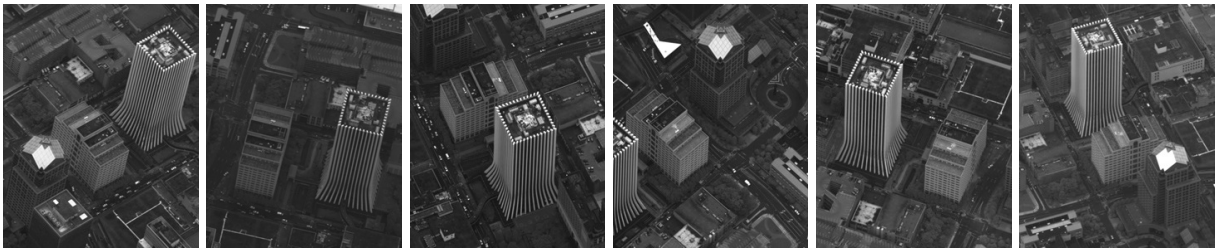


Figure 8. Sample image sequence from Exelis' Widea Area Motion Imagery (WAMI) system over an area in downtown Rochester, NY. Images have been cropped for display.

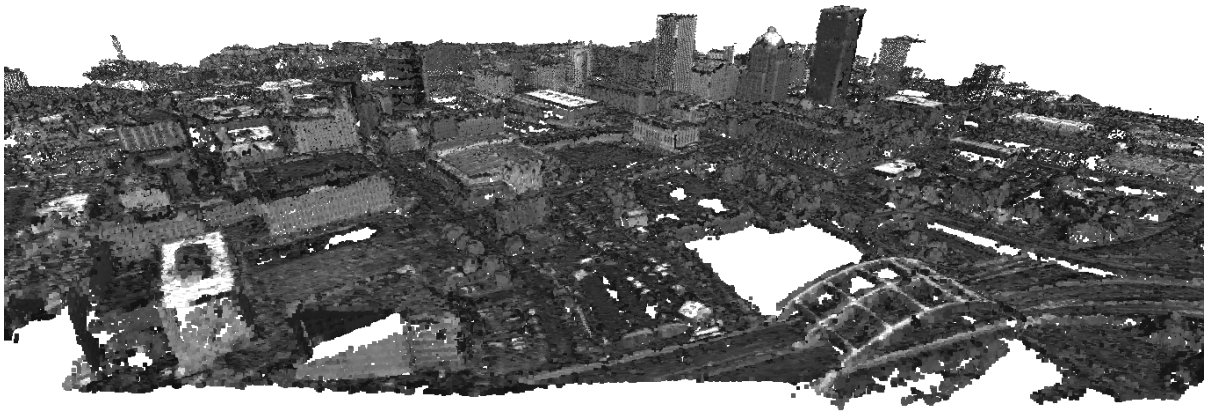


Figure 9. Resulting point cloud data from a WAMI image sequence, using the 3D workflow from Fig. 1.

4. RESULTS

Though the voxel space concept was presented previously,[11] the topic is worth revisiting as updated visualizations have been developed. The effect of the resolution of the voxel space on the unsampled regions is shown in Fig. 10, where resolutions range from 0.5m to 4.0m on a side. At 4.0m voxel resolution, the model appears blocky. More details appear in the model at resolutions of 2.5m, 2.0m, and 1.5m, and while the number of unsampled surfaces appears to increase, the model itself does not appear to be compromised. At 1.0m resolution, more surfaces of the model appear to be unsampled and some spurious unsampled voxels begin to appear in what should be free space around the model. As the voxel size decreases further, these spurious unsampled voxels

increase in number to the point that it becomes obvious at 0.5m that the point cloud data cannot support such fine voxel resolutions. A voxel resolution of 2.0m is used in all subsequent analysis because it provides adequate levels of detail in the structure of the model and does not contain spurious unsampled voxels, as can be seen in Fig. 10.

A portion of the downtown area of Rochester, NY is used here for analysis; the scene includes several high-rise buildings, many low-rise buildings, the Genesee River and multiple bridges. An oblique image, captured with the Exelis' WAMI system, is shown in Fig. 11 as a reference.

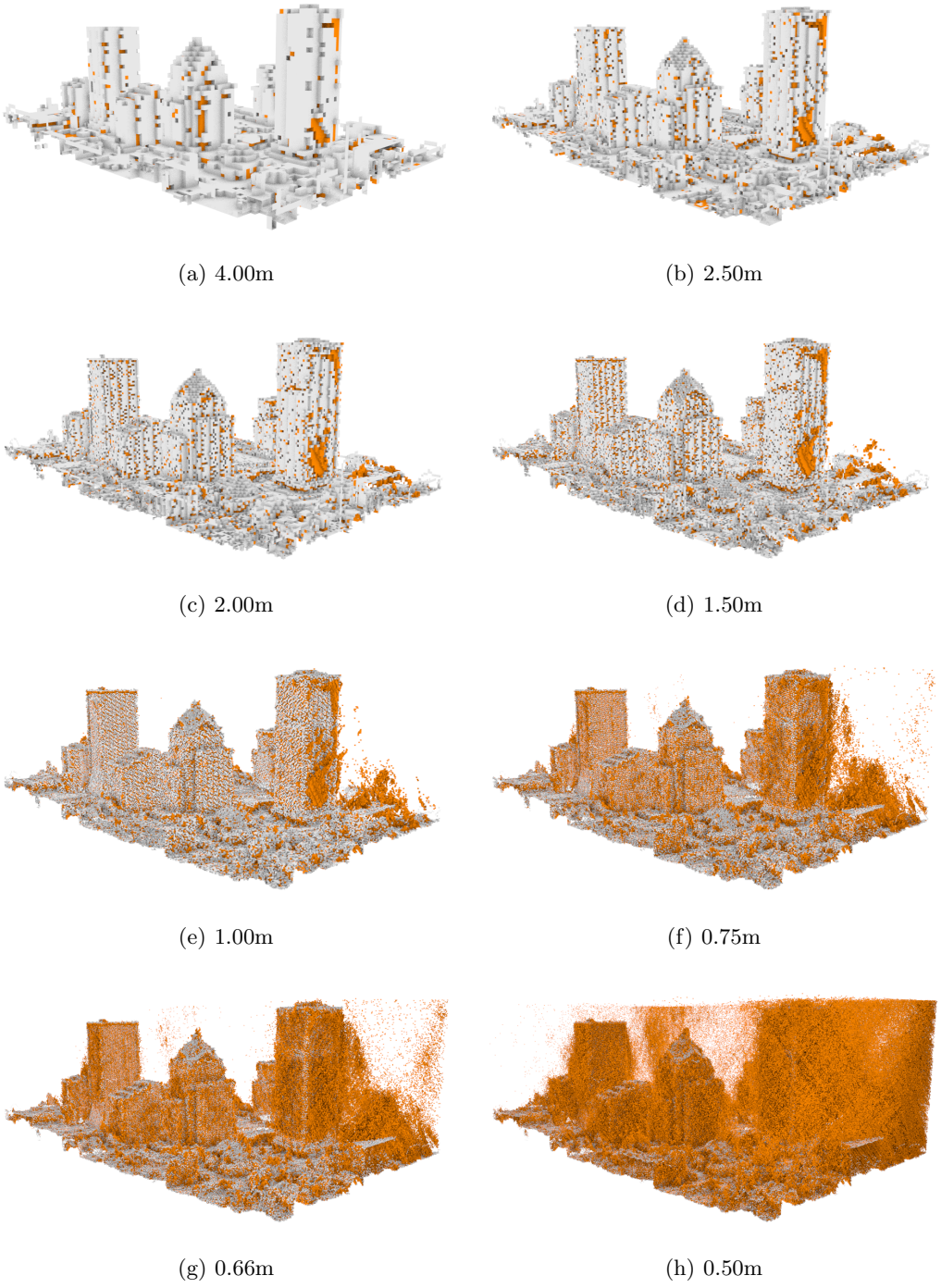
Using the point cloud shown in Fig. 9, a voxel space was built with 2.0m voxel resolution and an occupied voxel multiplier of 10 using the procedure described in Section 2.2. Using the procedure described in Section 2.3, centers of missing voxels on the free-unsampled boundary were identified and rays were cast to determine how many cameras had a clear line-of-sight to each particular center. Initial results had no consideration of camera pointing or field-of-view, and as a result overestimated the number of cameras that could see each center. It is important not to overestimate how many cameras could see an unsampled voxel, because unsampled voxels that are seen by a number of cameras that is over a certain threshold will be removed from consideration when computing the maps of potential aircraft imaging locations. If the number of cameras is overestimated, more unsampled voxels will be removed from consideration, including those that were identified erroneously, and this will impact the final result of the potential image location mapping.

Therefore it becomes important to consider the pointing of the camera and field-of-view in determining whether a camera could see a particular voxel. A central circular field-of-view was selected for simplicity, such that it did not include areas outside of the sensor array. Selecting a narrow circular region also limits the geometric distortion and is computationally efficient. The circular nature made it possible to implement this with a simple dot product to determine the angle between the camera axis and the ray of interest; if the ray is within an angle of tolerance, the ray tracing algorithm is used, otherwise it is assumed that the point was not seen by the camera. Results of each technique are shown in Fig. 12. Note that by limiting the camera field-of-view, there was a reduction in the number of cameras that saw the voxel centers on the outer edge of the point cloud. The bright area is centered around the fixed stare point, as expected.

At least some part of the river was visible in all of the images in the reconstruction, however the river does not reconstruct well and is therefore represented in the voxel space as a void area. Figure 12 shows that the river was seen by many cameras, indicated by the brightness. As was stated previously, it would not be beneficial to include more images of such an area, as it likely will not reconstruct. In this case, it becomes important to filter the voxels on the free-unsampled boundary based on the number of cameras that likely saw them. The dense reconstruction algorithm used in the workflow requires that at least 3 images be used for reconstruction, therefore it is certain that voxels that have only been seen by 2 or fewer cameras were not reconstructed due to lack of coverage.

A camera located at an altitude of 10,000ft with a nadir pointing angle was used to determine potential imaging locations. The footprint of the voxel space at altitude was broken up into 10x10m blocks, and the number of unsampled voxel centers that could be seen from a central location in each block was calculated. The unsampled voxels at the free-unsampled boundaries and the results of the potential imaging location calculations, displayed as a heat map where red values indicate that more unsampled voxels would be seen from that point, are shown in Fig. 13. Results are shown for unsampled voxels that were seen by two or fewer, three or fewer, and four or fewer cameras; these portions of unsampled voxels are subsets of the results presented in Fig. 12(b). Note that the circular portion in the lower left, including part of the river, has been suppressed from consideration as many of those unsampled voxels were included in other imagery. This is also reflected in the heat maps for potential imaging locations, as not as many unsampled voxels are seen from locations in that corner. The highest concentration of unsampled voxels is in the top right of the scene, and that also corresponds to where the most voxels are potentially seen from the aircraft. This portion of the scene included numerous buildings that were missing eastern facing sides, thus there was a large concentration of unsampled voxels. The location in each heat map where the most unsampled voxels can be viewed from is denoted with an X; note how the location shifts as more unsampled centers are included.

Despite these maps being in the preliminary stages of development, the potential for application is undeniable. The maps presented here only vary the X and Y location parameters, while holding the altitude and pointing



(a) 4.00m (b) 2.50m

(c) 2.00m (d) 1.50m

(e) 1.00m (f) 0.75m

(g) 0.66m (h) 0.50m

Figure 10. The effect of voxel resolution on unsampled regions. Surface voxels are shown in grey and unsampled voxels in orange. There are few extraneous unsampled voxels at voxel sizes above 1.50m, though the number of unsampled voxels on the buildings increases as the voxel size decreases. Additionally, spurious unsampled voxels begin to appear around the model in areas that should be free space as the voxel size decreases. At 0.50m, it is obvious that the voxel space cannot support this resolution.

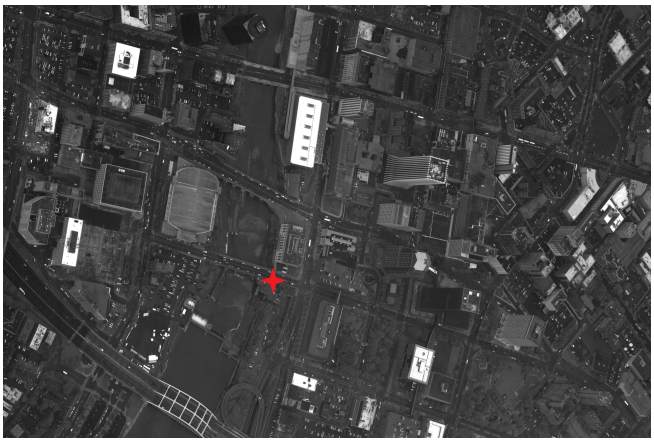
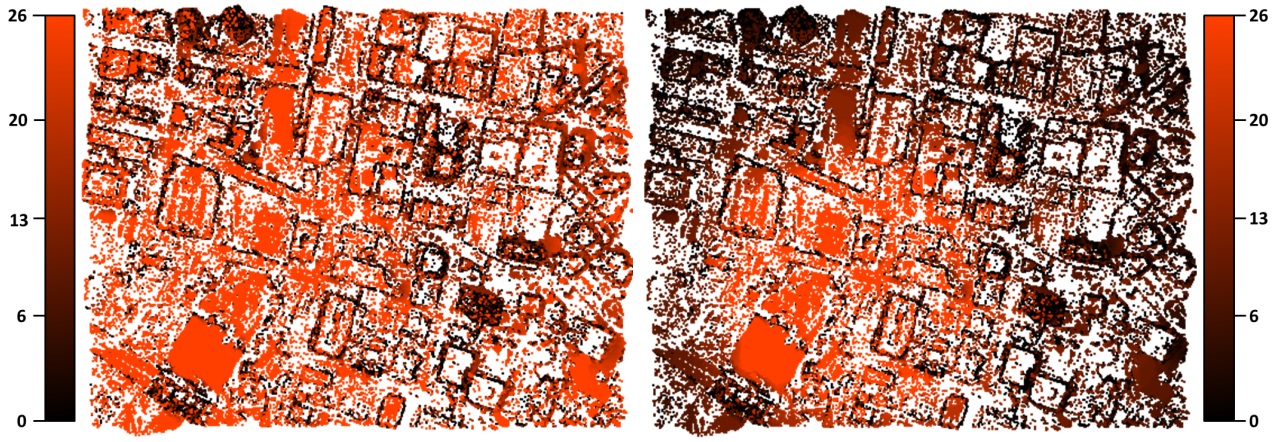
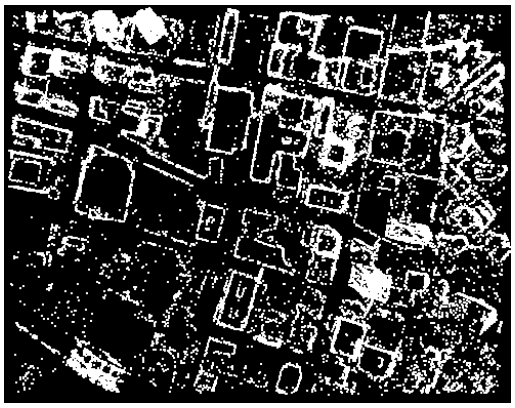


Figure 11. An oblique image over an area in downtown Rochester, NY captured with Exelis' Wide Area Motion Imagery (WAMI) system. The image was collected as part of a circular flight pattern with a fixed stare point, which has been marked in red. This image is provided to serve as a point of reference for the point cloud results

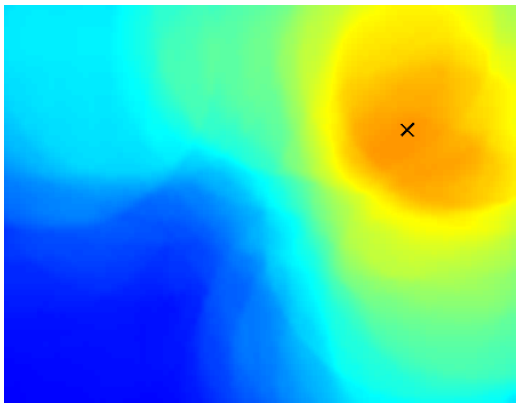


(a) No consideration of camera pointing or field-of-view (b) Limiting camera to circular central field-of-view

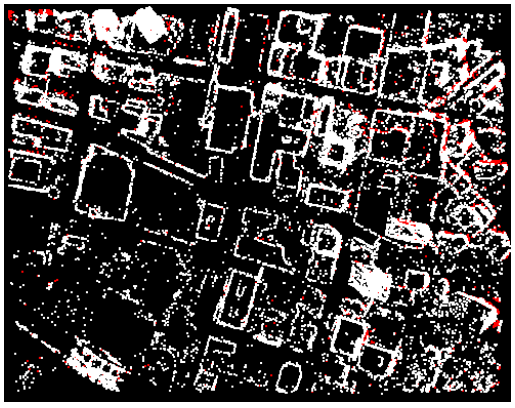
Figure 12. Missing centers of the voxel space, where the shading indicates how many cameras could potentially see the missing voxel. Those points that are black were seen by no cameras, and those that are the brightest were seen by all 26 cameras used in the PMVS reconstruction, all others were seen by some subset as indicated by the scale bars. The visibility of those centers in (a) was computed with no pointing or field-of-view constraints from the cameras; the fact that most of the points are bright indicates an overestimation in the number of cameras that could see each center. The visibility of those centers in (b) was computed with camera pointing information, and the camera was limited to a circular central field-of-view; note the reduction in brightness of the centers on the outer edges of the scene.



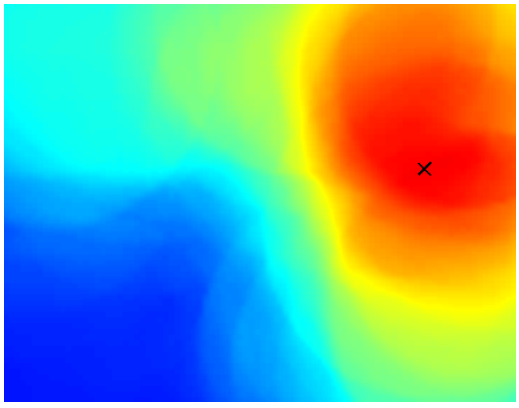
(a) Missing voxels seen by ≤ 2 cameras



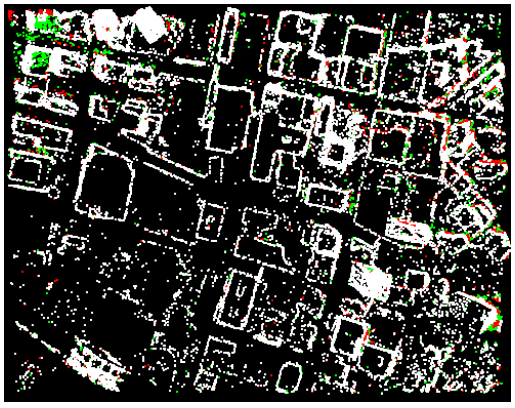
(b) Future aircraft image locations ≤ 2 cameras



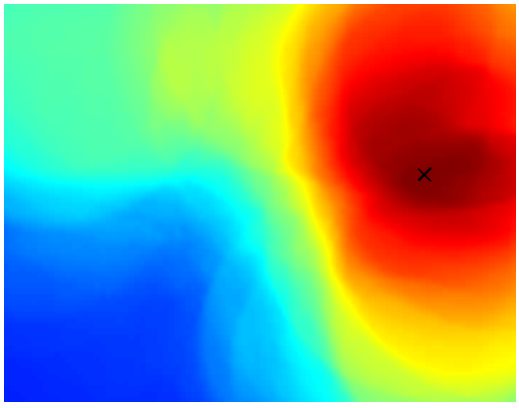
(c) Missing voxels seen by ≤ 3 cameras



(d) Future aircraft image locations ≤ 3 cameras



(e) Missing voxels seen by ≤ 4 cameras



(f) Future aircraft image locations ≤ 4 cameras

Figure 13. Unsampled voxels at free-unsampled boundaries that were seen by 2 or fewer, 3 or fewer, and 4 or fewer are shown in (a), (c), and (e) respectively. Corresponding heat maps of future aircraft image locations are shown in (b), (d), and (f), where a nadir pointing angle has been specified. The location at which the most unsampled voxels will be seen is marked with an X. Unsampled voxels at free-unsampled boundaries that were seen by more cameras than indicated were suppressed from consideration when generating the heat maps of potential aircraft image locations. To emphasize the differences, the unsampled voxels that were seen by 2 or fewer cameras is shown in white, those seen by 3 cameras are shown in red, and those seen by 4 cameras are shown in green. While there is not a large change between the three different cases, the inclusion of more unsampled voxels causes a shift in the identified aircraft location in the heat maps.

direction constant. In total, there are six potential variables to consider and determining the best methods for searching is a non-trivial problem. The maps themselves will prove to be useful for flight planning purposes, particularly if flying additional image collections, as the imagery obtained can be used to supplement an existing dataset. In addition, the aircraft locations of interest could be used to search some sort of database to determine if such an image already exists. This work is based on the assumption that inclusion of such imagery in a reconstruction will result in a point cloud with fewer voids in the imaged area.

5. CONCLUSIONS AND FUTURE WORK

Voids in point clouds derived from multi-view imagery exist as a result of texturally flat areas, obscuration of portions of the scene, and lack of coverage. A voxel-based approach has been presented to partition the 3D space, taking advantage of the idea of free space, such that voids can be easily identified. Once the potential areas of interest are determined, using the free-unsampled boundaries in the voxel space, a line-of-sight analysis is used to determine whether the void results from texturally flat areas, which are then removed as potential areas of interest. And finally, another line-of-sight analysis is used to create a map of potential imaging locations that can be used to obtain images covering the areas of interest in the point cloud that were likely the result of lack of coverage. This work is based on the assumption that inclusion of such images in the reconstruction would result in a point cloud with fewer voids. This method was demonstrated on point clouds derived from high frame rate, oblique, aerial imagery.

The simple line-of-sight analysis used to determine whether missing voxels were visible from each camera proved useful. However it is possible that a single threshold is not the best practice, particularly if trying to eliminate inclusion of voids that were a result of texturally flat areas in an image as there is other information available. Future endeavors could make use of the imagery itself, by reprojecting back to the camera frame and analyzing the local image textures. Multiple images with homogeneous texture could be used as another metric to determine whether a point should be included for further analysis or not. In addition, the angle between the cameras that could see a point may be useful. Images that are separated by small angles will result in poor triangulation of a point because of an insufficient base-to-height ratio. Images that are separated by large angles will result in features that are dissimilar, and therefore will likely fail to match in initial stages of reconstruction. In both cases, the likelihood of reconstruction of such a point in the scene is small, even if it was seen by 3 or more cameras. Thus these portions of the voxel space that were seen by 3 or more cameras with particularly small or large angular disparity should not be eliminated from the set of unsampled voxels used to develop flight maps because they likely could be reconstructed with the inclusion of more imagery.

Other areas for future work include using the voxel faces, rather than just the centers of the voxels for analysis. The current process does not consider whether a vertical or horizontal boundary is present, but this could be important when determining whether the face was visible from a camera. More specifically, a camera with a clear path to a voxel face will be counted as being seen by the camera, but this might not be the case depending on the angle between the ray and the surface, particularly at grazing angles.

The most obvious area for future work is the development of the heat maps for purposes of flight planning. Results presented here only considered a single altitude and single pointing angle. Ideally, a methodology will be developed that can consider multiple altitudes and pointing angles, combining the information in such a way that the best locations and pointing angles can be determined. This would move toward the establishment of a fully-automated system that can, possibly in real-time, provide guidance to in-flight systems to collect this missing data to provide void-free geometric reconstructions.

ACKNOWLEDGMENTS

This research is based upon work supported by the National Science Foundation (NSF) under award No. NSF 10-608, Accelerating Innovation Research and by support from Exelis, Inc., Geospatial Systems Division. The authors wish to thank Exelis, Inc. for providing the WAMI data used in conjunction with this project.

REFERENCES

1. S. Agarwal, N. Snavely, I. Simon, S. M. Seitz, and R. Szeliski, "Building Rome in a day," in *Computer Vision*, pp. 72–79, IEEE, 2009.
2. D. J. Walvoord, A. J. Rossi, B. D. Paul, B. Brower, and M. F. Pellechia, "Geoaccurate three-dimensional reconstruction via image-based geometry," in *SPIE Defense, Security, and Sensing*, pp. 874706–874706, SPIE, 2013.
3. D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision* **60**(2), pp. 91–110, 2004.
4. D. Brown, "A solution to the general problem of multiple station analytical stereo triangulation," tech. rep., RCA-MTP, 1958.
5. M. I. Lourakis and A. A. Argyros, "SBA: A software package for generic sparse bundle adjustment," *ACM Transactions on Mathematical Software* **36**(1), p. 2, 2009.
6. N. Snavely, S. M. Seitz, and R. Szeliski, "Photo tourism: Exploring photo collections in 3D," in *ACM Transactions on Graphics*, **25**(3), pp. 835–846, ACM, 2006.
7. Y. Furukawa, B. Curless, S. M. Seitz, and R. Szeliski, "Towards internet-scale multi-view stereo," in *IEEE Trans. on Pattern Analysis and Machine Intelligence*, **32**(8), pp. 1362–1376, 2010.
8. Y. Furukawa and J. Ponce, "Accurate, dense, and robust multi-view stereopsis," *IEEE Trans. on Pattern Analysis and Machine Intelligence* **32**(8), pp. 1362–1376, 2010.
9. R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, Cambridge University Press, 2nd ed., 2003.
10. J. Amanatides, A. Woo, *et al.*, "A fast voxel traversal algorithm for ray tracing," in *Proceedings of EUROGRAPHICS*, **87**, pp. 3–10, 1987.
11. K. N. Salvaggio and C. Salvaggio, "Automated identification of voids in three-dimensional point clouds," in *SPIE Optical Engineering+ Applications*, pp. 88660H–88660H–12, International Society for Optics and Photonics, 2013.