May 19, 2015

Visualization of 3D Images from Multiple Texel Images Created from Fused Ladar/Digital Imagery

Cody C Killpack, *Utah State University*
Scott Budge

Visualization of 3D Images from Multiple Texel Images Created from Fused Ladar/Digital Imagery

Cody C. Killpack, Scott E. Budge

Center for Advanced Imaging Ladar, Utah State University,
Logan, UT 84322-4120, (435) 797-3433

ABSTRACT

The ability to create 3D models, using registered texel images (fused ladar and digital imagery), is an important topic in remote sensing. These models are automatically generated by matching multiple texel images into a single common reference frame. However, rendering a sequence of independently registered texel images often provides challenges. Although accurately registered, the model textures are often incorrectly overlapped and interwoven when using standard rendering techniques. Consequently, corrections must be done after all the primitives have been rendered, by determining the best texture for any viewable fragment in the model.

Determining the best texture is difficult, as each texel image remains independent after registration. The depth data is not merged to form a single 3D mesh, thus eliminating the possibility of generating a fused texture atlas. It is therefore necessary to determine which textures are overlapping and how to best combine them dynamically during the render process. The best texture for a particular pixel can be defined using 3D geometric criteria, in conjunction with a real-time, view-dependent ranking algorithm. As a result, overlapping texture fragments can now be hidden, exposed, or blended according to their computed measure of reliability.

Keywords: lidar, ladar, texel image, 3D texel model, dynamic texture, view-dependent rendering, render to texture

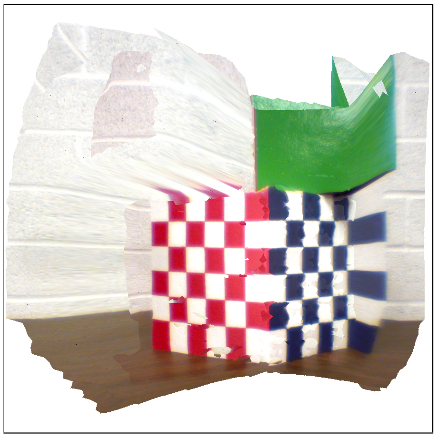
1. INTRODUCTION

The need for automatically generating 3D models exists in many technical fields of study. Creating a 3D model, with visual and dimensional accuracy, is an invaluable tool for disaster management, situational awareness, and even automatic target recognition (ATR). Building these models involves combining 3D data captured from various viewpoints around an object of interest. Once the images are matched with a registration algorithm, they can be viewed and explored in a vector graphics tool. The ability to apply and overlay accurate textures within this viewing program is an important step towards achieving a usable model.

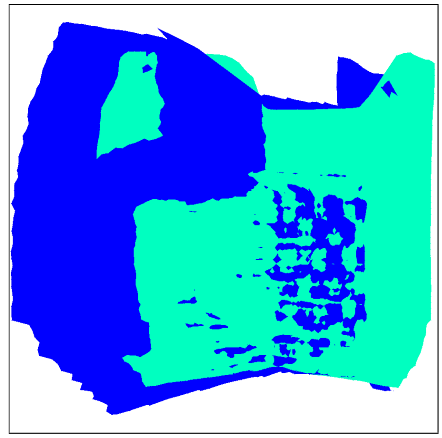
The Center for Advanced Imaging Ladar (CAIL) at Utah State University uses texel images (fused ladar and digital imagery) to create 3D models. Texel images are generated with a texel camera, which houses a co-axial ladar array and electro-optical (EO) camera. The data produced by a texel camera is fused at the pixel level.[1,2] Researchers at CAIL have exploited this data fusion to register multiple texel images into a single 3D model.[3] The transformation parameters computed during registration are used to match individual texel images, bringing them into a single reference frame (or common world coordinate system). The result is a 3D model, composed of independent 2.5D texel images.

Rendering these 3D texel models continues to be an important research focus. Rendering involves loading and viewing a 3D model. Numerous vector graphics tools provide the ability to magnify, transform, and explore a texel model in 3D space. However, viewing a sequence of independently registered texel images often provides challenges, and the problems that arise when using standard rendering techniques need to be corrected. Even when a 3D model is accurately registered, the individual textures assigned to it are often incorrectly overlapped and interwoven. This is caused by stretched mesh triangles, noise, and the default rendering methods used by typical vector graphics tools, as shown in Figure 1. In Figure 1(a), the problems with stretched triangles caused from different viewpoints are clearly illustrated. Figure 1(b) shows the same registered scene with the textures replaced by a color assigned to each image, indicating where the rendered textures originate.

Forming a texture atlas (stitching textures together) is impractical, due to the independent nature of the underlying data sets. Consequently, corrections must be done after all the primitives (triangles) have been



(a) Textured 3D Texel Model



(b) Colored 3D Texel Model

Figure 1. A registered 3D texel model with noise and distortion present. The individual texel images, or scans, were taken from two slightly different perspectives. (a) The combination of these two texel images in an independently registered and textured sequence. (b) The same scene found in (a) but with a unique color assigned to each image, rather than a texture.

rendered, by determining the best texture for any given area of the model. Triangle based multi-texturing techniques have been proposed, and define the best texture for a given area as a triangle labeling problem.[4]

The "accuracy" of a texture for a particular area of the model can be defined using a 3D geometric criterion. Additionally, this criterion must be used in conjunction with a real-time, view-dependent ranking algorithm, as the virtual camera perspective also influences the suitability of a texture. The concept of view-dependent texture mapping has been proposed, along with examples of real-time applications.[5,6] Well known 3D geometric criteria have also been demonstrated, such as the angle between a triangle normal and the viewing frustum, distance to the camera, surface area of a triangle face, and even photo-consistency metrics.[4,7–9] These methods have all been explored and ranked.[10] A combination of these metrics, referred to as the "reliability" of the texture, would allow overlapping texture fragments (i.e. pixels) to be hidden, exposed, or blended according to their computed measure of reliability. Methods for blending these textures have been proposed.[11,12] Current techniques determining the visibility and rank of a texture triangle can also be found.[12,13]

The problem addressed in this paper is unique because each texel image remains independent after registration. The depth data has not been merged, thus eliminating the need for a fused texture atlas. Therefore, the solution must determine which textures are overlapping, and how to best combine them dynamically while rendering. The OpenGL Shading Language (GLSL) enables textures to be combined on a fragment level (per-pixel) during the render process. Furthermore, the accuracy of a texture triangle can be interpolated, which results in greater uniform interaction among overlapping textures. As a result, the best texture assignment for a given area of the 3D texel model can be done on a per-fragment basis. This is achievable in real-time for applications that use texel images obtained from framing ladar sensors.

The remainder of this paper is presented as follows. Details concerning triangle ranking and 3D geometric criteria are given in Section 2. A brief introduction into generating synthetic ladar data, which is used to prove the feasibility of the rendering algorithm presented in this paper, is also given. Section 3 discusses the rendering algorithm in it's entirety. Current implementation and programming techniques are covered in detail. The results can be found in Section 4. Section 5 concludes the paper, giving insight into the limitations of the rendering algorithm presented. Future work is also discussed.

2. ACCUMULATED TEXTURE RELIABILITY

An estimation of accumulated texture reliability for a single texel image, or scan, is done via 3D geometric criteria. The reliability of a texture represents an accumulated rankable value, which is computed using three

Algorithm 1 Computing texture reliability for a single texel image.

1. Compute the *normal vector accuracy* for each triangle in the mesh. This is done *once* for each texel image in the scan set.
2. Render a single texel image. For each primitive in the image:
   (a) Compute the *point of view confidence.*
   (b) Interpolate *range confidence.*
   (c) Compute *accumulated texture reliability.*
   (d) Interpolate the pixel color.
   (e) Store corresponding pixel color and reliability data for future pipeline processing.

---

specific criteria discussed in this section. The ability to determine rank among overlapping textures, based on their estimated reliability values, is a critical step for proper texel model rendering. Not only must these reliability ratings be accurate, but they must also adapt dynamically to satisfy changes in virtual camera perspective. For convenience, the values assigned by these criteria are clamped, falling between the range $[0.0, 1.0]$. A reliability rating of zero represents a discardable fragment (essentially unusable for rendering), while a reliability rating of one represents an ideal fragment. In general, a wide range of reliabilities will be found across the surface of a texel image mesh; however, only a limited number of these will ever be considered perfectly reliable.

The steps for computing the accumulated texture reliability of a single texel image, viewed from the current virtual camera position, are summarized in Algorithm 1. This process is repeated until all visible scans in a texel model have been rendered. Any scan located on the backside of a 3D texel model should not be processed. Once complete, 2D textures (containing color and reliability information) for every scan in the model will be available for use in the second stage of the rendering pipeline. This process is known as *rendering to texture.* As a result, a complicated 3D sorting problem has been resolved with a simple 2D texture solution. Additionally, transparency sorting and object intersection algorithms will no longer be needed for proper model rendering.

Computing accumulated texture reliability begins with *normal vector accuracy.* Using the normal vector orientation of a triangle face within a texel image mesh is a suitable method for determining accuracy. Triangle accuracies are constant values, and are computed using the original acquisition perspective of a texel image. Normal vectors oriented toward the acquisition center of projection (COP) usually correspond to triangles with smaller surface areas, which result in less texture interpolation (i.e. pixelation). Triangles whose normal vectors are rotated away from the frustum viewpoint are usually less accurate, and often correspond to stretched and noisy areas of the mesh. Using the cosine similarity, given by

$$\cos(\theta) = \frac{\mathbf{a} \cdot \mathbf{N}}{\|\mathbf{a}\|\|\mathbf{N}\|}, \tag{1}$$

between a triangle normal vector $\mathbf{N}$ and the acquisition COP vector $\mathbf{a}$ provides a suitable measure of accuracy. Figure 2 helps illustrate this principle. As the angular difference between these two vectors approaches zero degrees, the computed accuracy draws closer to one. As the angular difference increases, the accuracy level decreases, and eventually reaches zero (90 degree difference). The acquisition COP vector is not constant, which often results in accuracy degradation for triangles located further away from the center of the texel image. Even a perfectly flat surface can only have ideal accuracies when the triangle normal vectors and the acquisition COP vector are identically oriented (the image center).

Although normal vector orientation assigns an accuracy value for every triangle in a texel image mesh, this value is constant, and only represents the maximum possible accuracy for any given triangle. To account for deviations in virtual perspective, numerical values known as *confidences* must also be assigned to every texel image triangle. When used together, confidence and accuracy values define triangle reliability when the virtual camera deviates from the acquisition perspective of a texel image. Not surprisingly, as the perspective view of

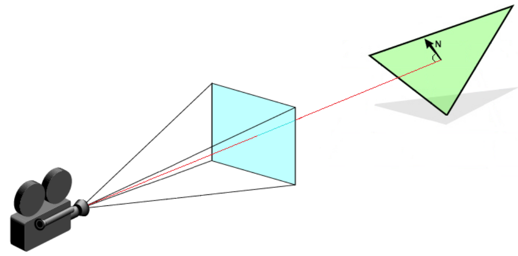


Figure 2. Triangle Normal Vector Accuracy

a triangle changes, the confidence of it's texture will begin to degrade. Consequently, the most reliable view of any triangle, is the view from which it was originally acquired. Any deviation from this acquisition perspective begins to decrease a triangle's confidence, thus changing it's overall reliability.

Point of view confidence is the correlation between the current view (virtual camera) and the acquisition view (real texel camera). When these two perspectives align, the POV confidence of any triangle in the mesh is said to be *ideal*. Not surprisingly, cosine similarity provides a reliable and inexpensive way to calculate these confidence values in real-time. A comparison is done between the acquisition COP vector and the orientation of the triangle mesh from the current point of view. The measure of confidence is computed using the normalized dot product between these two vectors. Angle differences greater than 90 degrees are ignored, and will not be rendered.

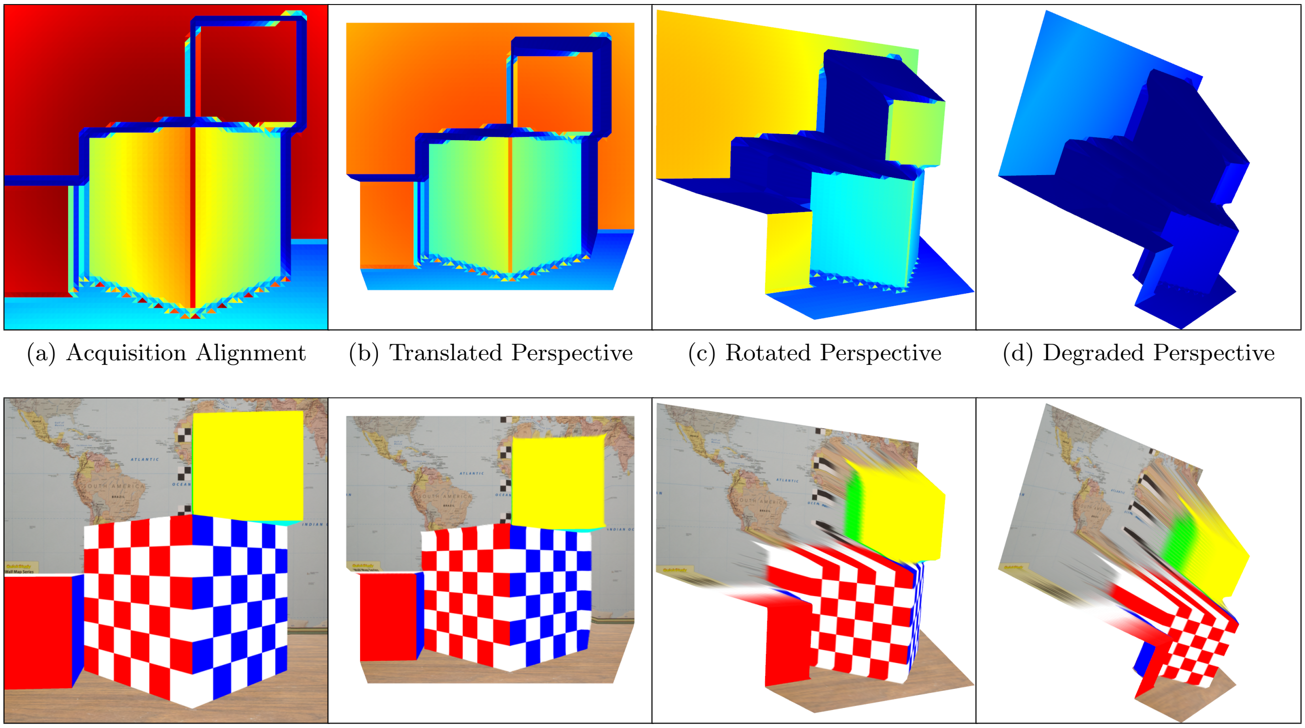
Range confidence describes the correlation between the virtual and acquisition distances of a given triangle. The original acquisition distance (meters) is compared with the range, or distance, between rendered mesh triangles and the virtual camera COP. Vertex coordinates are commonly captured in metric units, and therefore, the rendered scene is required to use the same coordinate system for clipping plane placement and camera movement. These confidence values change as the virtual camera zooms in and out of a rendered texel image.

The accumulation of triangle normal vector accuracy, POV confidence, and range confidence constitutes the *accumulated texture reliability* measure of a mesh. The three measures are combined by taking the product of the measures. Accumulated reliability values represent the overall accuracy of any triangle for a given distance and orientation in virtual 3D space. These values are then interpolated on a per-fragment (pixel) level at render time using GLSL shaders. The accumulated reliability for a fragment is highest when it's virtual and real-world acquisition frustums perfectly coincide. Overlapping fragment areas of a texel model can now be accurately ranked and weighed using their individual reliability ratings. To ease alpha blending and computation, accumulated reliability values are computed as percentages, clamped between $[0.0, 1.0]$.

Figure 3 illustrates accumulated accuracy by showing a synthetically generated texel image from various viewpoints in virtual space. Using synthetic data facilitates visualizing and evaluating imaged-based algorithms. The synthetic data sets shown in this paper mimic the real-world scenes and texel camera characteristics found in the CAIL laboratory. To aid in visualizing the reliability of the texel image mesh, a heat map has been applied. Cooler colors represent poorly rated triangles and warmer colors indicate triangles with higher reliability.

3. TEXEL MODEL RENDERING

Once the individual 2.5D scans of a texel model have been rendered and evaluated for reliability, the resulting 2D texture outputs can be sorted and blended together on a per-pixel basis. For a texel model composed of n scans, there may be up to n overlapping fragments for any given area. An overview of the model rendering pipeline can be seen in Figure 4. Fragments which match the rendering background color, or those which have a reliability of zero, will be discarded. Once a list of valid fragments is obtained, they must be sorted according to their accumulated texture reliability ratings. The top ranked fragments are then extracted for processing, as shown in Algorithm 2. These fragments can be blended if they fall within a certain threshold, or simply the best fragment can be rendered. Displaying the best fragment works well with synthetic data, however real-world data benefits from moderate blending, as a result of higher noise levels and misregistration errors.



(a) Acquisition Alignment (b) Translated Perspective (c) Rotated Perspective (d) Degraded Perspective

(e) (f) (g) (h)

Figure 3. An example of *accumulated texture reliability* is given against a single synthetic texel image, shown from a variety of virtual perspectives. (a) Acquisition alignment, which equals the maximum possible reliability for a triangle mesh. (b) Degraded accumulated reliability levels as the virtual perspective is translated back and to the left. (c) The effect on reliability due to rotation. (d) Severely degraded perspective. The perspective is so degraded that triangles with reliability ratings of zero will be ignored by the rendering algorithm. For reference, the companion texture for (a)–(d) can be seen below each in (e)–(h), respectively.

Algorithm 2 Processing overlapped textures for a 3D texel model.

1. Extract valid pixels within a vertical column.
2. Eliminate non-range compliant pixels.
3. Sort remaining pixels according to their accumulated reliability ratings.
4. Process the final pixel group: show best pixel or perform blending.
5. Repeat steps 1-4 for all pixels across the 2D canvas.

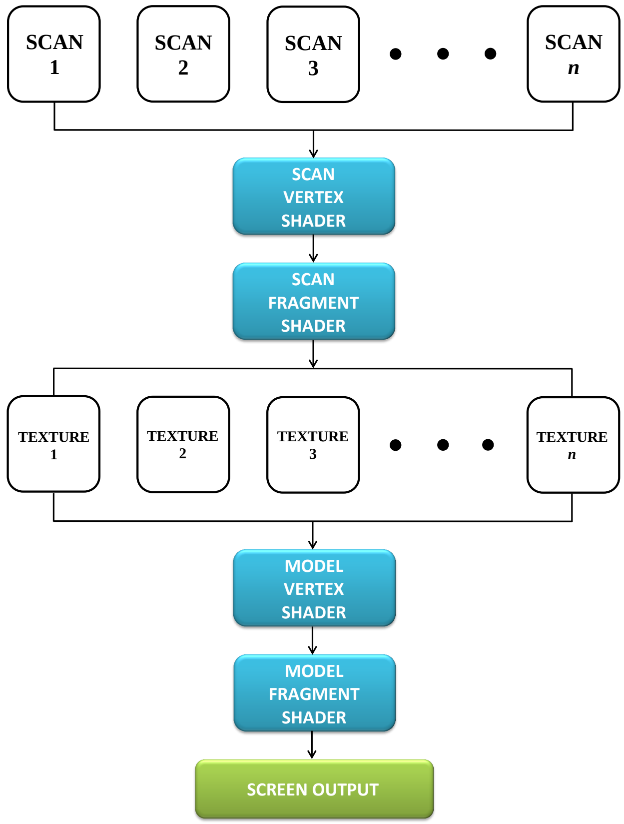
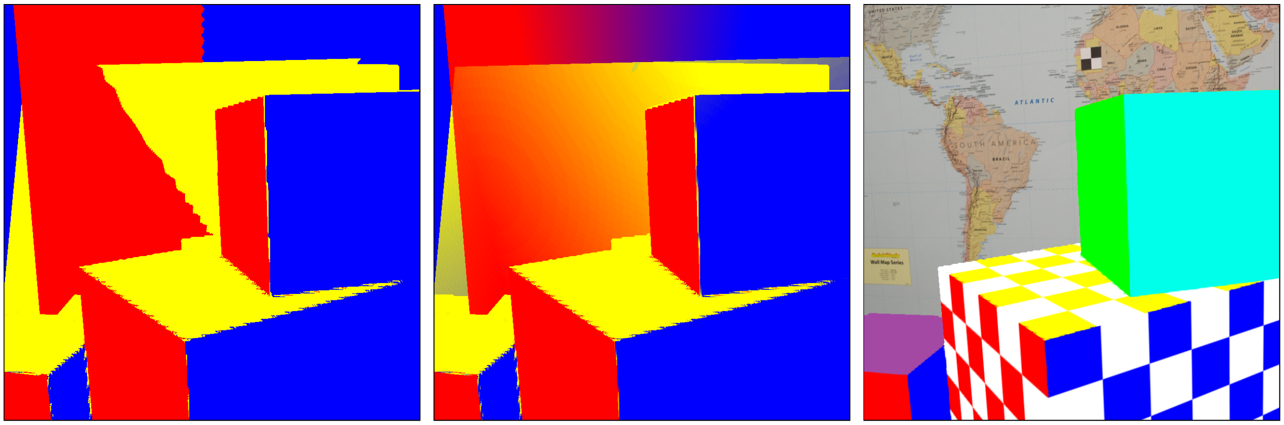


Figure 4. Texel Model Rendering Pipeline

A visual illustration of real-time, dynamic texture assignment is shown using a synthetically generated data set in Figure 5. In the Figure, the model shown is composed of three individual 2.5D texel images, taken from different perspectives, which have been separately rendered into 2D textures. As a result, the final texture assignment is simplified, by using these 2D images stacked on top of each other. The pixels within every column of the texture stack are sorted according to reliability. This enables the best fragments to be brought forward or possibly blended. The blue areas will be rendered from pixels in image 1, the red areas from image 2, and the yellow areas from image 3. The effect of blending is shown in Figure 5(b), which illustrates how the sharp boundaries in reliability between images 1, 2, and 3 can be blended to reduce artifacts at the boundary. Figure 5(c) is the final rendered scene.

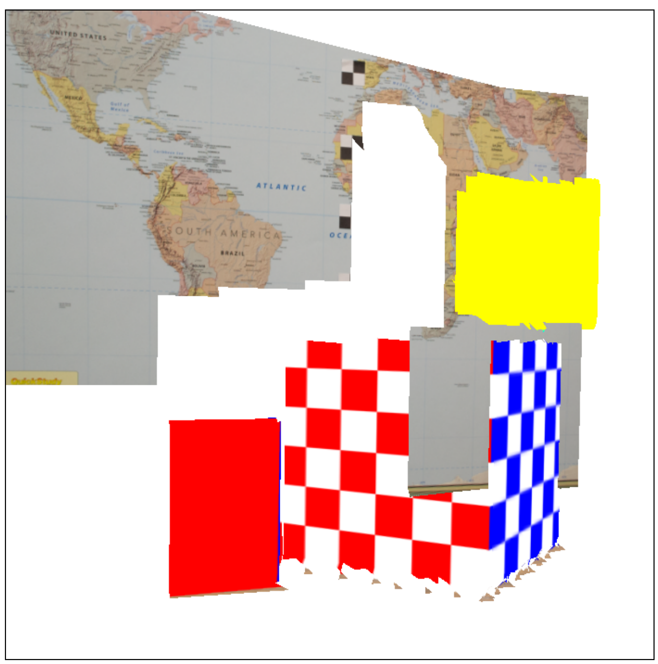
Although using a rendered stack of 2D images simplifies model texturing, doing so eliminates any inherited visibility properties. Hidden surface removal (HSR) is used to determine which surfaces should not be visible from a particular viewpoint. This process is sometimes called hiding, and is often a necessary step to render images correctly. Texel model rendering is no exception, as evident in Figure 6. In this example, portions of a texel image with higher reliability can be erroneously seen through portions of a texture closer to the virtual camera. Specifically, a small section of the world map in the background can be seen through the checkered cube. To prevent these windowing artifacts, range compliance must be used when comparing overlapping textures.

Fragments should only be processed if they are within a reasonable distance from each other (i.e. they are range compliant). Assessing range compliance is done by comparing the virtual distances between all overlapping fragments and the virtual camera COP. The distance threshold is taken from acquisition camera noise characteristics: any fragments located beyond two times the standard noise deviation should be ignored. Checking for range compliance ensures that only valid overlapping textures can be ranked, sorted, blended, and displayed.



(a) Best Texture Map (b) Blended Texture Map (c) Final Texture

Figure 5. A visual demonstration of real-time, dynamic texture assignment using a synthetically generated 3D model created from three texel images. (a) Sections of the image rendered from the best reliability measures in image 1 (red), 2 (blue) and 3 (yellow). (b) Blended reliability. (c) The final rendered image using colors from pixels selected according to the reliability map given in (a).



(a)

Figure 6. The necessity of range compliance is demonstrated. Under certain circumstances, portions of imagery with higher reliability ratings can be erroneously seen through textures that are closer to the virtual camera.

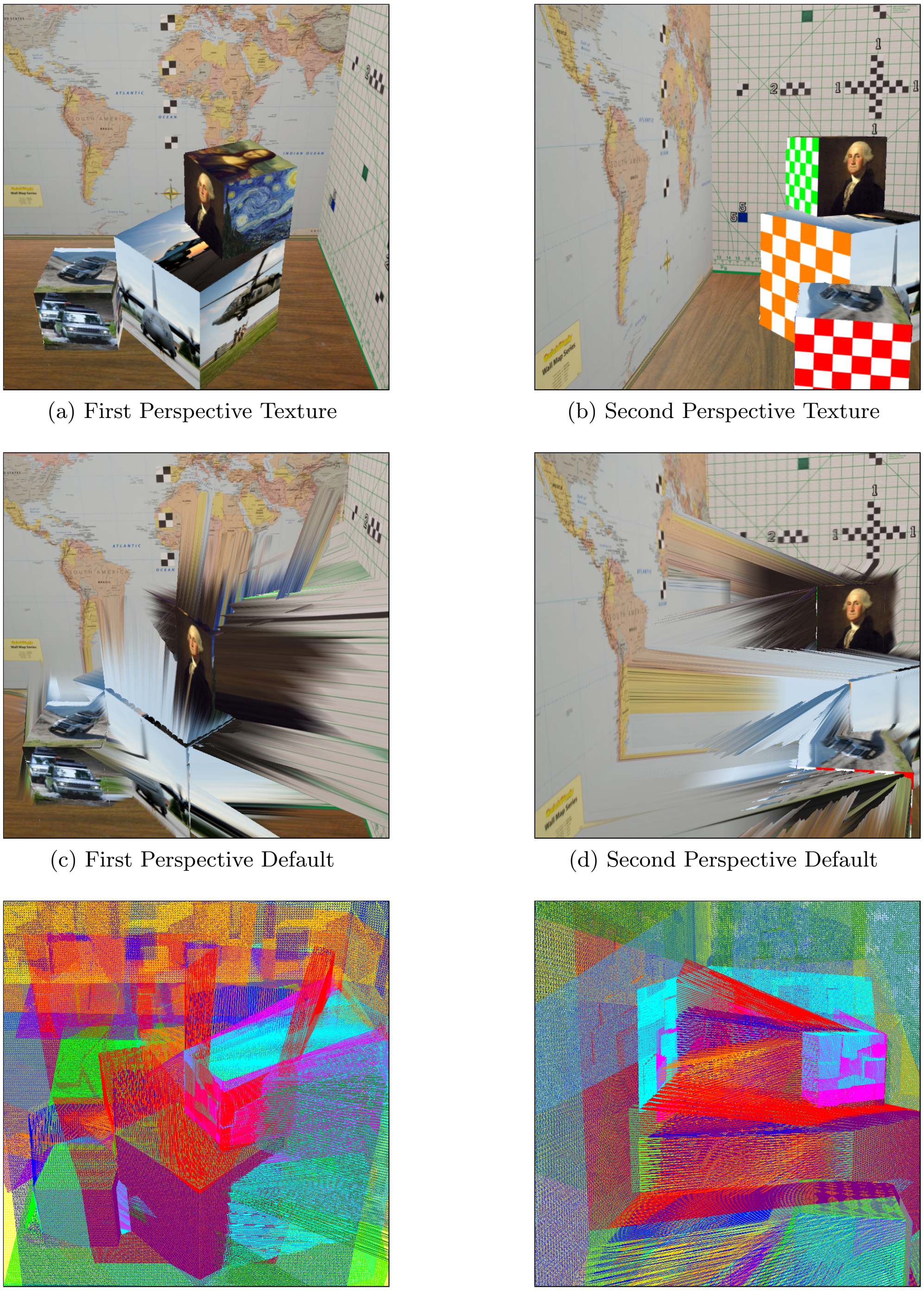
4. RESULTS

Final algorithm performance is demonstrated using a synthetic data set, shown in Figure 7. Real-time, dynamic texturing can be seen as the model is explored from a variety of perspectives. The model shown in Figure 7 is composed of 10 different 2.5D texel images, which have been captured from various viewpoints around the scene using a synthetic texel image generator. The advantages of using real-time, dynamic texture assignment can be seen in Figures 7(a) and 7(b). Although the perspectives of these two images are dramatically different, the model textures have adapted to accommodate virtual camera orientation and weighted texture reliability. For comparison, the original default rendering of these two model perspectives is given in Figures 7(c) and (d), respectively. The underlying triangle meshes are also given in Figures 7(e) and 7(f). As shown from these Figures, the problems with stretched triangles and overlapped textures have been visibly reduced using the method proposed in this paper.

5. CONCLUSION

Standard rendering techniques are not suitable for texturing 3D texel models. Due to the independent nature of the underlying data sets, corrections to model textures must be done at render time. The most reliable texture, for a given perspective, is better defined using 3D geometric criteria in conjunction with a real-time, view-dependent ranking algorithm. The method described in this paper has shown promise in correctly rendering 3D texel models. Improved results were observed when using higher resolution texel models.

Unfortunately, misregistration errors can lead to poor rendering performance. Although blending techniques help mitigate these problems, blurred and clouded texture areas can appear as a result. In addition, stretched and noisy triangles continue to present numerous challenges when rendering texel models. This is especially obvious when working with sharp and abrupt edges within a model. Future work may investigate solutions beyond triangle-based rendering. Additionally, better blending and alternative image-based processing techniques warrant further investigation.



(a) First Perspective Texture

(b) Second Perspective Texture

(c) First Perspective Default

(d) Second Perspective Default

(e) First Perspective Mesh

(f) Second Perspective Mesh

Figure 7. Dynamic texture assignment is shown using a synthetically generated 3D texel model. Figures (a) and (b) are the same scene viewed from two different perspectives. For comparison, the original default rendering of these two model perspectives is given in Figures (c) and (d), respectively. The underlying triangle meshes for these two model perspectives are shown in Figures (e) and (f).

REFERENCES

[1] Pack, R. T., Israelsen, P., and Sealy, K., "A co-boresighted synchronized ladar/EO imager for creating 3D images of dynamic scenes," in [*Laser Radar Technology and Applications X*], Kamerman, G. W., ed., **5791**, 42–50, SPIE (May 2005).

[2] Budge, S. E. and Badamikar, N. S., "Calibration method for texel images created from fused lidar and digital camera images," *Opt. Eng.* **52**, 103101 (Oct. 2013).

[3] Budge, S. E., Badamikar, N. S., and Xie, X., "Automatic registration of fused lidar/digital imagery (texel images) for three-dimensional image creation," *Opt. Eng.* **54**, 031105 (Mar. 2015).

[4] Lempitsky, V. and Ivanov, D., "Seamless mosaicing of image-based texture maps," in [*Computer Vision and Pattern Recognition, 2007. CVPR '07. IEEE Conference on*], 1–6 (June 2007).

[5] Debevec, P. E., Taylor, C. J., and Malik, J., "Modeling and rendering architecture from photographs: A hybrid geometry- and image-based approach," in [*Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*], *SIGGRAPH '96*, 11–20, ACM, New York, NY, USA (1996).

[6] Debevec, P., Yu, Y., and Boshokov, G., "Efficient view-dependent image-based rendering with projective texture-mapping," tech. rep., EECS Department, University of California, Berkeley (1998).

[7] Debevec, P., Taylor, C., Malik, J., Levin, G., Borshukov, G., and Yu, Y., "Image-based modeling and rendering of architecture with interactive photogrammetry and view-dependent texture mapping," in [*Circuits and Systems, 1998. ISCAS '98. Proceedings of the 1998 IEEE International Symposium on*], **5**, 514–517 vol.5 (May 1998).

[8] Buehler, C., Bosse, M., McMillan, L., Gortler, S., and Cohen, M., "Unstructured lumigraph rendering," in [*Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*], *SIGGRAPH '01*, 425–432, ACM, New York, NY, USA (2001).

[9] Alj, Y., Boisson, G., Bordes, P., Pressigout, M., and Morin, L., "Space carving mvd sequences for modeling natural 3d scenes," in [*Three-Dimensional Image Processing (3DIP) and Applications*], **8290**, 829005–829005–15, SPIE (2012).

[10] Alj, Y., Boisson, G., Bordes, P., Pressigout, M., and Morin, L., "Multi-texturing 3D models: How to choose the best texture?," in [*3D Imaging (IC3D), 2012 International Conference on*], 1–8 (Dec 2012).

[11] Iiyama, M., Kakusho, K., and Minoh, M., "Super-resolution texture mapping from multiple view images," in [*Pattern Recognition (ICPR), 2010 20th International Conference on*], 1820–1823 (Aug 2010).

[12] Wang, L., Kang, S. B., Szeliski, R., and Shum, H.-Y., "Optimal texture map reconstruction from multiple views," in [*Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*], **1**, I–347–I–354 vol.1 (2001).

[13] Yu, Y., "Efficient visibility processing for projective texture-mapping," *J. Computers & Graphics* **23**(2), 245–253 (1999).