

**Interpolation of Three-Dimensional (3-D) Light Detection
and Ranging (LiDAR) Point Cloud Data onto a Uniform
Upsampled Grid**

by Prudhvi Gurram, Shuowen Hu, and Alex Chan

ARL-TR-6616

September 2013

NOTICES

Disclaimers

The findings in this report are not to be construed as an official Department of the Army position unless so designated by other authorized documents.

Citation of manufacturer's or trade names does not constitute an official endorsement or approval of the use thereof.

Destroy this report when it is no longer needed. Do not return it to the originator.

Army Research Laboratory

Adelphi, MD 20783-1197

ARL-TR-6616

September 2013

**Interpolation of Three-Dimensional (3-D) Light Detection
and Ranging (LiDAR) Point Cloud Data onto a Uniform
Upsampled Grid**

Prudhvi Gurram, Shuowen Hu, and Alex Chan
Sensors and Electron Devices Directorate, ARL

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing the burden, to Department of Defense, Washington Headquarters Services, Directorate for Information Operations and Reports (0704-0188), 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to any penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.

PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.

1. REPORT DATE (DD-MM-YYYY) September 2013			2. REPORT TYPE Final		3. DATES COVERED (From - To) September 2012 to December 2012	
4. TITLE AND SUBTITLE Interpolation of Three-Dimensional (3-D) Light Detection and Ranging (LiDAR) Point Cloud Data onto a Uniform Upsampled Grid					5a. CONTRACT NUMBER	
					5b. GRANT NUMBER	
					5c. PROGRAM ELEMENT NUMBER	
6. AUTHOR(S) Prudhvi Gurram, Shuowen Hu, and Alex Chan					5d. PROJECT NUMBER	
					5e. TASK NUMBER	
					5f. WORK UNIT NUMBER	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) U.S. Army Research Laboratory ATTN: RDRL-SES-E 2800 Powder Mill Road Adelphi, MD 20783-1197					8. PERFORMING ORGANIZATION REPORT NUMBER ARL-TR-6616	
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)					10. SPONSOR/MONITOR'S ACRONYM(S)	
					11. SPONSOR/MONITOR'S REPORT NUMBER(S)	
12. DISTRIBUTION/AVAILABILITY STATEMENT Approved for public release; distribution unlimited.						
13. SUPPLEMENTARY NOTES						
14. ABSTRACT Airborne laser-scanning light detection and ranging (LiDAR) systems are used for remote sensing topology and bathymetry. The most common data collection technique used in LiDAR results in scanning data that form a non-uniformly sampled three-dimensional (3-D) point cloud. To interpret and further process the point cloud, these raw data are usually converted to digital elevation models (DEMs) in a uniform and upsampled raster format. For this, the elevation information from the available non-uniform point cloud is mapped onto the uniform grid points and the grid points with missing elevation information are filled by using interpolation techniques. In this report, a partial differential equations (PDE) based approach is proposed to perform this interpolation. Due to the desirable effects of using higher order PDEs, smoothness is maintained over homogeneous regions, while reducing the draping effects near the edges of distinctive objects in the scene. Simulation results are presented in this report to illustrate the advantages of the proposed algorithm.						
15. SUBJECT TERMS 3-D point cloud, LiDAR, uniform upsampling, image inpainting, Cahn-Hilliard PDE						
16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT UU	18. NUMBER OF PAGES 30	19a. NAME OF RESPONSIBLE PERSON Shuowen Hu	
a. REPORT Unclassified	b. ABSTRACT Unclassified	c. THIS PAGE Unclassified			19b. TELEPHONE NUMBER (Include area code) (301) 394-2526	

Contents

List of Figures	iv
1. Introduction	1
2. PDE Based Image Inpainting	3
3. Application to 3-D LiDAR Point Cloud	5
4. Simulation Results	7
5. Conclusion	14
6. References	15
Appendix. Code for the Uniform Grid Upsampling of 3-D LiDAR Point Cloud Data	17
List of Symbols, Abbreviations, and Acronyms	23
Distribution List	24

List of Figures

Figure 1. Points from 3-D LiDAR cloud placed on upsampled uniform grid.	6
Figure 2. Upsampling and inpainting of building A area.	8
Figure 3. Two large gaps of building B.	9
Figure 4. Upsampling and inpainting of building B - Gap 1.	10
Figure 5. Upsampling and inpainting of building B - Gap 2.	12
Figure 6. Upsampling and inpainting of a silo.	14

1. Introduction

Light detection and ranging (LiDAR, also known as LADAR) technology emerged soon after the introduction of the laser in the late 1950s, but it is only widely adopted recently for a myriad of applications in various engineering fields, as well as for use in the geospatial intelligence community and the military (1). Many types of LiDAR systems have been developed to date, such as two-dimensional (2-D) gated framing LIDAR, scanning three-dimensional (3-D) LiDAR, and more recently, flash 3-D LiDAR based on Geiger mode or linear mode avalanche photodiode (1). For this work, we focus on airborne scanning 3-D LiDAR, which has been widely used for applications that include sensing topology and bathymetry, as well as urban terrain mapping.

Typically, airborne scanning 3-D LiDAR systems employ pulsed laser light that is swept laterally by an oscillating scanner mirror, sampling at a rate between 10 and 200 Hz, depending on the specific sensor system. As the aircraft moves along its flight path, the linear scan produces a Z-shaped pattern (2). Measurements along the scan lines are discretized to points, typically by applying a threshold on the intensity, resulting in two to six returns per pulse measurement. Points are given a corrected geo-position using calibrated latitudinal and longitudinal coordinates by correlating time-stamped measurements from an inertial measurement unit and global positioning system sensor. The measurements are based on the time-of-flight principle, giving each return its distance from the sensor, and thus, allowing an elevation value to be computed. The resulting scanning data constitute a non-uniformly sampled (with respect to the x- and y-spatial directions) 3-D point cloud. The points from the Z-shaped scans are not evenly spaced, but are rather semi-randomly distributed due to flight path perturbations caused by wind, changes in aircraft heading and velocity, and other mechanical/environmental factors. Measured points may also overlap from adjacent flight paths, resulting in data with locally varying point densities. LiDAR sensors are usually mounted in the nadir-pointing orientation, but off-nadir pointing angles may occur either by design or due to perturbations in the flight path. For off-nadir pointing angles, “shadows” cast by tall objects or structures induce holes in the data where no points were collected. Due to these factors, LiDAR data can exhibit a high degree of local variability with respect to the sample spacing.

To interpret and further process the 3-D point cloud data, these data are often converted to digital elevation models (DEMs). DEMs are created either in raster format, where elevation information is available at every point on a uniform grid/raster, or as vector-based triangulated irregular network consisting of irregularly distributed nodes and lines with 3-D coordinates arranged in a network of non-overlapping triangles. The focus of this work is on rasterized DEMs, which are essential in various applications such as terrain modeling and hydrological modeling (3, 4). Several interpolation techniques have been developed to compute rasterized DEMs, including the

triangulated irregular network (TIN) interpolation (5), natural neighbor (NN) (6), ordinary kriging (7), and universal kriging (8). NN and TIN interpolation techniques are computationally efficient and most commonly employed in commercial software. The most prevalent LiDAR exploitation software is the Quick Terrain Modeler (QT Modeler) developed by Applied Imagery, which is widely used by the military and the intelligence community. To fill holes and generate a rasterized DEM (referred to as a digital surface model in the user's manual), QT Modeler applies adaptive triangulation, which is a modified TIN technique. QT Modeler's adaptive triangulation generates 3-D triangles across empty regions and then samples the elevation value of the triangle surface at the spatial coordinates of the empty cell to interpolate its elevation value (9).

While TIN and NN techniques are computationally efficient, these techniques may result in draping effects wherever shadows occur and near the edges of objects like buildings and trees. Kriging methods are advanced geostatistical algorithms based on regionalized variable theory. However, kriging assumes that the spatial variation in the z-values is statistically homogeneous throughout the surface. While this assumption may hold in natural scenes for applications like soil-landscape and hydrological modeling, it is unlikely to be valid for dense urban scenes with buildings of all heights, as well as all sorts of trees and vehicles. Although LiDAR data for soil-landscape and hydrological modeling continue to be acquired, LiDAR data collections of urban scenes have increasingly been gaining importance and attention. In order to ameliorate the draping effects inherent to current interpolation techniques for urban LiDAR point clouds, we need to ensure that the edge information is well preserved during the interpolation procedure in order to compute the uniformly gridded DEMs.

In this report, we propose a partial differential equations (PDE) based approach to interpolate the 3-D point cloud onto a densely sampled (i.e., upsampled) uniform grid. First, all available LiDAR elevation data are used to populate an upsampling grid. Because the sample spacing (in x- and y- spatial directions) in the upsampling grid is much smaller than the average point cloud sample spacing, the populated upsampling grid is rather sparse. Since PDEs have been applied to automated image interpolation (10) in the past, we postulate that the same principle can be used for the interpolation of elevation information on a uniform grid from the sparsely populated upsampling grid by propagating the existing elevation data to compute the rest of the points on the upsampling grid. A second-order PDE-like heat diffusion equation can diffuse a value (elevation information here) across an area over time (11). However, this approach would work well only when the missing grid points are surrounded by homogeneous regions (areas of same elevation). It would fail when there is a sharp edge, for instance, in the LiDAR shadow regions along the edges of buildings and would lead to the same draping effects observable in linear interpolation techniques. This problem is addressed in this report by employing higher-order (e.g., fourth-order) PDEs (12), through which we can maintain smoothness in homogeneous regions while propagating the sharp edge information in the scene.

The rest of the report is organized as follows. In section 2, the concept of PDE-based image inpainting is introduced. Section 3 describes how this image inpainting technique is applied to perform uniform grid upsampling of LiDAR point cloud. Simulation results are presented in section 4. Finally, section 5 concludes the report with some remarks about the proposed method.

2. PDE Based Image Inpainting

Traditionally, image inpainting was done by skilled image restoration artists in museums to restore ancient paintings that had cracks or gaps due to aging. More recently, image inpainting has been widely used in digital image processing for applications such as image restoration and super-resolution. Image inpainting can be divided into two main categories: exemplar-based inpainting and PDE-based inpainting. Exemplar-based inpainting technique uses the whole image to find clues as to what constitutes the missing part of the image by checking for repetitive patterns and other clues. While they are extremely useful to inpaint textures in the images, these techniques require a lot of training data to learn what kind of patterns to look for in the images. On the other hand, PDE-based inpainting techniques do not use the whole image, but just the local regions around the missing parts of the image. Furthermore, PDE-based techniques do not need any kind of training process because they do not use any prior information regarding the images. A good review of the PDE based inpainting techniques was written by Chan and Vese (13).

The main principle of PDE-based techniques is to minimize the energy in the image with a constraint that the difference between the original image and inpainted image in non-inpainted areas is within a certain threshold. For example, if u denotes the original image (with no holes) over a 2-D domain Ω , u_0 denotes the observed image (with holes) over a 2-D observed domain D , $D \subset \Omega$, then PDE-based image inpainting algorithms minimize the energy in the image denoted by $E[u]$ over the domain Ω with a constraint that u and u_0 are as similar to each other as possible over the domain D (13). Mathematically it can be written as,

$$\min E[u] \quad \text{such that } E[u/u_0] \leq \sigma^2, \quad (1)$$

where σ^2 represents the white noise in the observed image. This problem can be written as an unconstrained optimization problem given by

$$\min E[u] + \lambda E[u/u_0], \quad (2)$$

where the Lagrange multiplier λ denotes the balance between energy minimization and image fitting. One can observe from this equation that PDE-based image inpainting produces an inpainted image with least energy and that best fits the observed image. In all PDE-based techniques, the data fitting model is considered to be

$$E[u/u_0] = \frac{1}{|D|} \int_D (u - u_0)^2 dx dy. \quad (3)$$

This is equivalent to determining the inpainted image that best fits the observed image in a least square error sense. The performance of the inpainting algorithm is crucially dependent on the choice of the energy functional used in equation 2. In the literature, there have been attempts to define this energy functional in different ways, which leads to different heat flow PDE equations. The choice of the energy functional essentially determines how the heat energy is diffused from the surrounding regions into the holes or missing parts of the image.

One of the first attempts made was to define the energy functional $E[u]$ using the Sobolev norm (14), i.e., $E[u] = \int_{\Omega} |\nabla u|^2 dx dy$. Substituting this energy functional and the data fitting model defined in equation 3 into equation 2, one can obtain the optimization problem as

$$\min \int_{\Omega} |\nabla u|^2 dx dy + \lambda \frac{1}{|D|} \int_D (u - u_0)^2 dx dy. \quad (4)$$

The above optimization problem can be solved by applying the Euler-Lagrange (E-L) system of equations defined for the extremum (minimum here) or stationary point of energy functionals, such as the one defined in equation 4. The definition and proof of the existence of Euler-Lagrange (E-L) equations can be found in reference 15. The E-L equations for the minimization problem in equation 4 leads to a Fourier heat diffusion equation, in which the PDE is given by

$$u_t = \Delta u + \lambda(u - u_0), \quad u(t = 0) = u_0. \quad (5)$$

Here, u_t is the diffused image or inpainted image at any time t , Δ represents the Laplacian operator, and u_0 is the observed image. This equation represents the diffusion of temperature across a metal over time. In case of image inpainting, the same equation may represent the diffusion of gray values across the image, i.e., from surrounding regions into the missing parts and holes. However, this energy functional and this PDE are effective only when the holes are present within and surrounded by homogeneous regions. This method also fails when there are sharp edges present around the missing parts of the image.

In order to deal with this shortcoming, the Total Variational (TV) model was first introduced for image denoising by Rudin et al. (11), and later, used for image inpainting by Chan and Shen (10). In the TV model, the energy functional is given by the total variation of u , that is $|D(u)|$ over Ω , and the optimization problem is

$$\min \int_{\Omega} |D(u)|^2 dx dy + \lambda \frac{1}{|D|} \int_D (u - u_0)^2 dx dy. \quad (6)$$

The E-L system of equations for this problem yield an anisotropic or nonlinear diffusion equation as shown in

$$u_t = \nabla \cdot \left(\frac{\nabla u}{|\nabla u|} \right) + \lambda(u - u_0), \quad u(t = 0) = u_0, \quad (7)$$

where $\nabla \cdot$ represents the divergence operator. The TV model yields a second-order PDE and can handle the edges in the image to a certain extent and propagate them into the missing parts along with the homogenous regions.

There are other higher-order PDE models in the literature, which can handle edge information along with the diffusion of smooth regions, including the Mumford-Shah model (16) and Mumford-Shah-Euler model (17). There are two disadvantages with all these techniques. First, these PDE techniques cannot handle large gaps or holes in the image if they fall along the edges and produce a stair-casing effect over smooth regions. Second, they are computationally very expensive to implement.

In order to deal with the problem of large holes, inpainting of binary images using modified Cahn-Hilliard PDE equation was proposed by Bertozzi et al. (18), and later extended to grayscale images by Burger et al. (12). This new technique is called $TV - H^{-1}$ inpainting technique and the fourth-order PDE or the diffusion equation is given by

$$u_t = -\Delta \left(\nabla \cdot \left(\frac{\nabla u}{|\nabla u|} \right) \right) - \lambda(u - u_0), \quad u(t = 0) = u_0. \quad (8)$$

In order to discretize this PDE for numerical implementation, a convexity splitting scheme is used as shown in reference 12 by introducing two constants, C_1 and C_2 , where they are set to $C_1 > 1/\epsilon$ and $C_2 > \lambda$. The resulting discrete time-stepping scheme is given by

$$\frac{u_{k+1} - u_k}{\tau} + C_1 \Delta \Delta u_{k+1} + C_2 u_{k+1} = C_1 \Delta \Delta u_k - \Delta \left(\nabla \cdot \left(\frac{\nabla u}{|\nabla u|} \right) \right) + C_2 u_k - \lambda(u_k - u_0). \quad (9)$$

Here τ represents the time-differential over which the energy is being diffused. For complete details on the proof of existence of solution and convergence, please refer to the work of Burger et al. (12).

Based on the value of λ , one can balance between smoothness of the final image and fit between the final image and the observed image. If λ is set to a high value, the fit between the final image and the observed image is given more importance and the final image will look very similar to the observed image in the non-missing regions. On the other hand, if λ is set to a low value, the final image is very smooth since the minimization of the energy functional is given the priority. This option is useful when the noise in the observed image needs to be removed while it is being inpainted into the final image.

3. Application to 3-D LiDAR Point Cloud

In order to apply the $TV - H^{-1}$ inpainting technique to LiDAR data, we consider the (X, Y) coordinates in the data as the 2-D image domain, while the elevation or Z coordinate is

considered equivalent to the gray levels in an image. There are two tasks to be performed: (1) upsample the LiDAR point cloud and (2) rasterize the point cloud. First, a uniform grid or raster of 2-D points ((X, Y) coordinates) is defined over the area of interest. The upsampling of the data is performed by choosing the distance between the grid points to be at least half the average distance between the points in the dataset. Once the grid is generated, it is populated with the elevation information from the original point cloud using a nearest neighbor selection. For each datum in the point cloud, the nearest point on the (X, Y) grid is determined in terms of Euclidean distance and the elevation of the original point is assigned to the nearest grid point in the raster. After assigning the elevation information of all the cloud points to the uniform grid points, many empty cells (i.e., holes) will be present in the uniform grid. A real-world example of a uniform grid after the nearest neighbor interpolation is shown in figure 1. Here the white points represent the points of the uniform grid where the elevation information is present. The black regions in the figure represent the holes. These holes are filled by diffusing the homogeneous regions surrounding them into the holes and propagating edge information wherever necessary using $TV - H^{-1}$ PDE-based inpainting technique.

Points in region placed in upsampling-grid

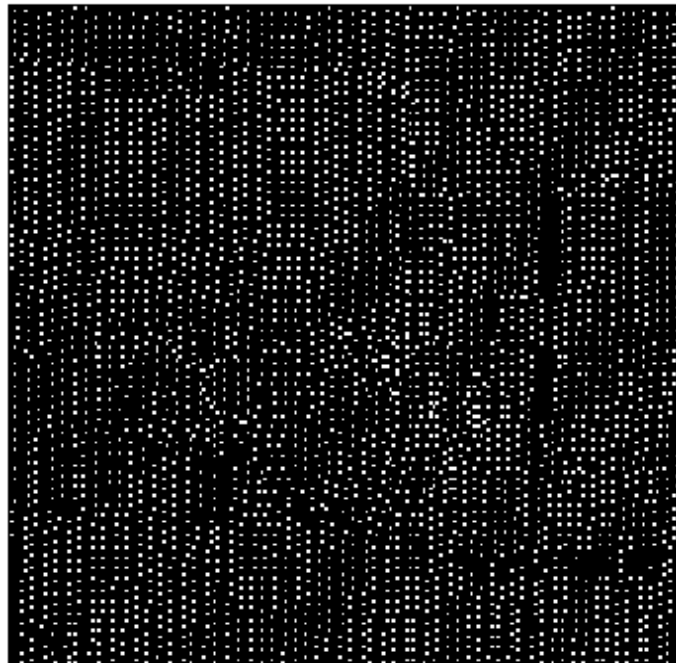


Figure 1. Points from 3-D LiDAR cloud placed on upsampled uniform grid.

4. Simulation Results

The airborne LiDAR data used in this work was collected by Merrick and Company over the city of Lubbock, TX, using a fixed-wing aircraft. All data and imagery shown in this work are unclassified. In this section, four sets of results are presented to show the effectiveness of the fourth-order PDE based inpainting to upsample and rasterize the LiDAR point cloud. The average distance between the 3-D points in the original data is 0.4 m. The l_2 error between the uniform grid point cloud with holes and the inpainted uniform grid point cloud is used as the termination criterion. When this error falls below a predefined threshold, the algorithm is considered as converged and the time-stepping process is then terminated.

The first dataset used for our simulation covers an area around a building (called building “A” in this report), which contains objects like buildings, trees, and shrubs. The original point cloud collected over the region is shown in figure 2a. The distance between grid points is set to 0.05 m i.e., the data are being upsampled by 8 times along both x and y axes. The balance parameter λ is set to 100. The upsampled point cloud on a uniform grid is illustrated in figure 2b. This figure shows the dense points on the upsampled grid, as well as the elevation information obtained after the inpainting process. In order to compare the inpainting technique to the standard interpolation technique used in QT Modeler, the surfaces generated by QT Modeler using standard internal interpolation of the original point cloud, as well as using the upsampled uniform grid point cloud, are presented in figures 2c and d, respectively. One can observe that the surface model obtained using upsampled point cloud is sharper than the one generated using original point cloud, especially on the roof of the buildings, as well as the drop between the tree and the shrub in the region.

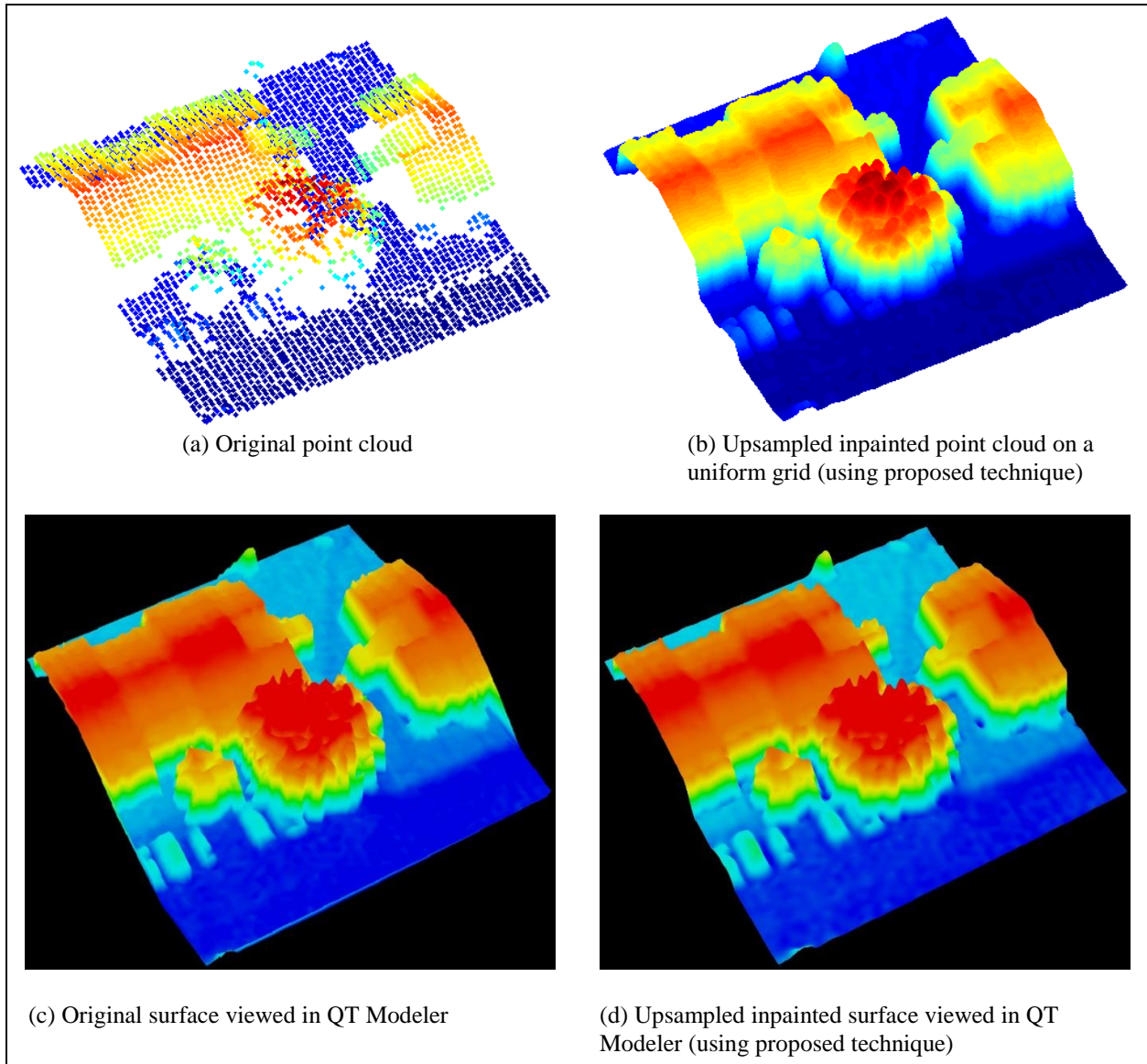


Figure 2. Upsampling and inpainting of building A area.

The second and third datasets used in this work focus on another building (referred to as building “B”). The original point cloud over the building B area (displayed in QT Modeler as a surface) is shown in figure 3. Two large gaps exist next to the building: Gap 1 is shown in figure 3a and Gap 2 in figure 3b. These gaps might be incurred through line-of-sight occlusion by the side of the building along the aircraft’s flight path or due to the presence of water that absorbed most of the LiDAR signal in those areas. To illustrate the benefit of the upsampling inpainting approach, sections of the building that include Gap 1 and Gap 2 were used in our simulations.

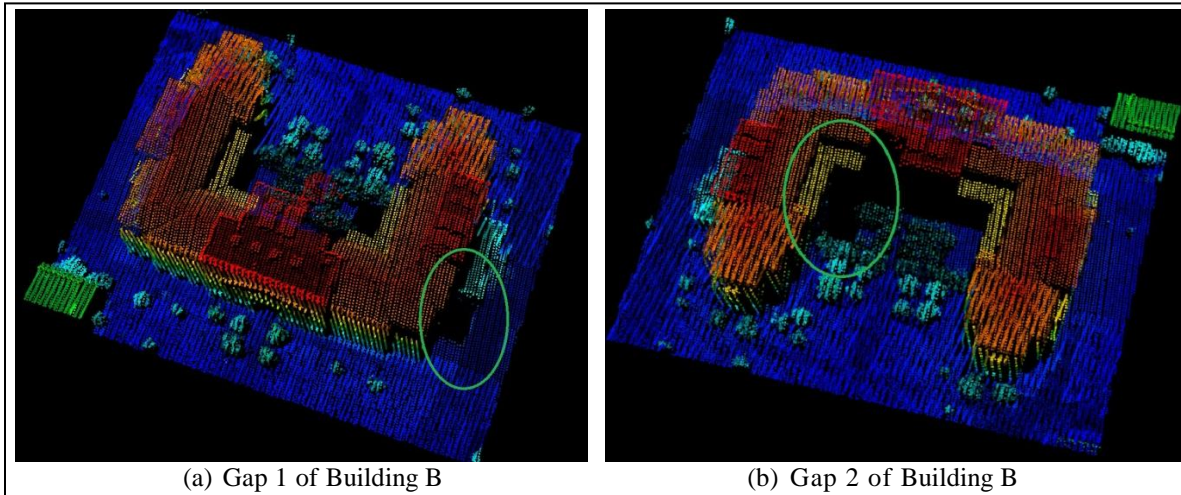


Figure 3. Two large gaps of building B.

The simulation results for Gap 1 of building B are presented in figure 4. The original point cloud collected over Gap 1 is shown in figure 4a. The distance between grid points is set to 0.1 m, i.e., the data are being upsampled by 4 times along both x and y axes. The balance parameter λ is set to 100. The upsampled point cloud on a uniform grid is illustrated in figure 4b. The surfaces generated by QT Modeler using standard internal interpolation of the original point cloud and using the upsampled uniform grid point cloud are presented in figures 4c and d, respectively. One can observe from figure 4c that TIN interpolation creates a flat surface with constant slope to connect the top of the building to the edge of the gap. On the other hand, the same wall in the upsampled inpainted surface in figure 4d has a much sharper interpolated slope, especially near the top of the building. This visually helps to improve the perception of the wall of the building and proves that the $TV - H^{-1}$ inpainting technique is able to propagate the information from the surrounding areas into the holes better than simple interpolation.

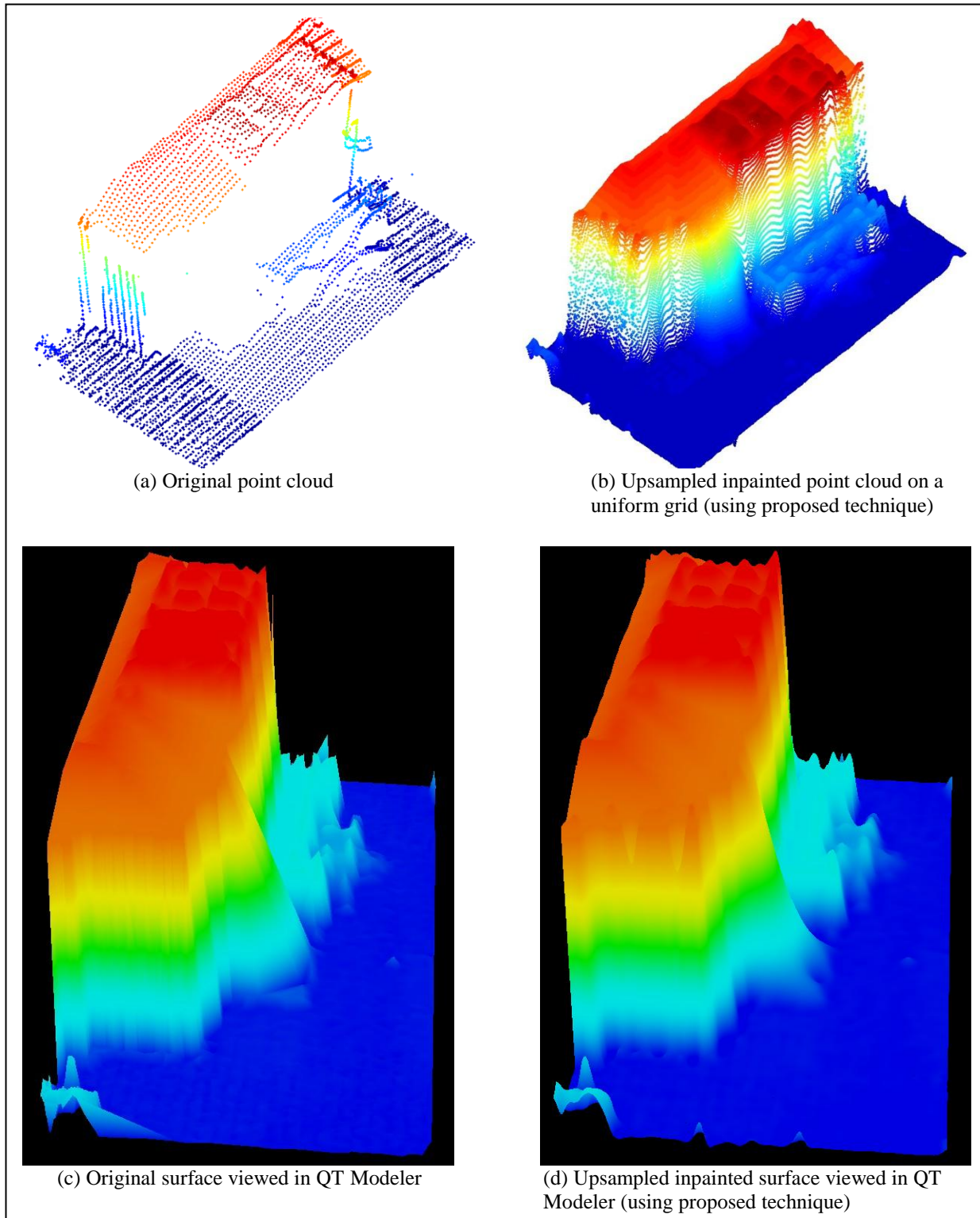


Figure 4. Upsampling and inpainting of building B - Gap 1.

The simulation results for Gap 2 of building B are presented in figure 5. The original point cloud collected over Gap 2 is shown in figure 5a. The distance between grid points is set to 0.2 m, i.e., the data are being upsampled by 2 times along both x and y axes. The balance parameter λ is again set to 100. The upsampled point cloud on a uniform grid is illustrated in figure 5b. The surfaces generated by QT Modeler using standard internal interpolation of the original point cloud and using the upsampled uniform grid point cloud are presented in figures 5c and 5d, respectively. The piece-wise linear interpolation of original point cloud produced spurious slant surfaces from the top of the building to the ground level that cover the hole in the image. However, the PDE-based inpainting technique generates a more vertical wall from the top of the building to the ground level and propagates the edge information from the surrounding shrubs in to the hole.

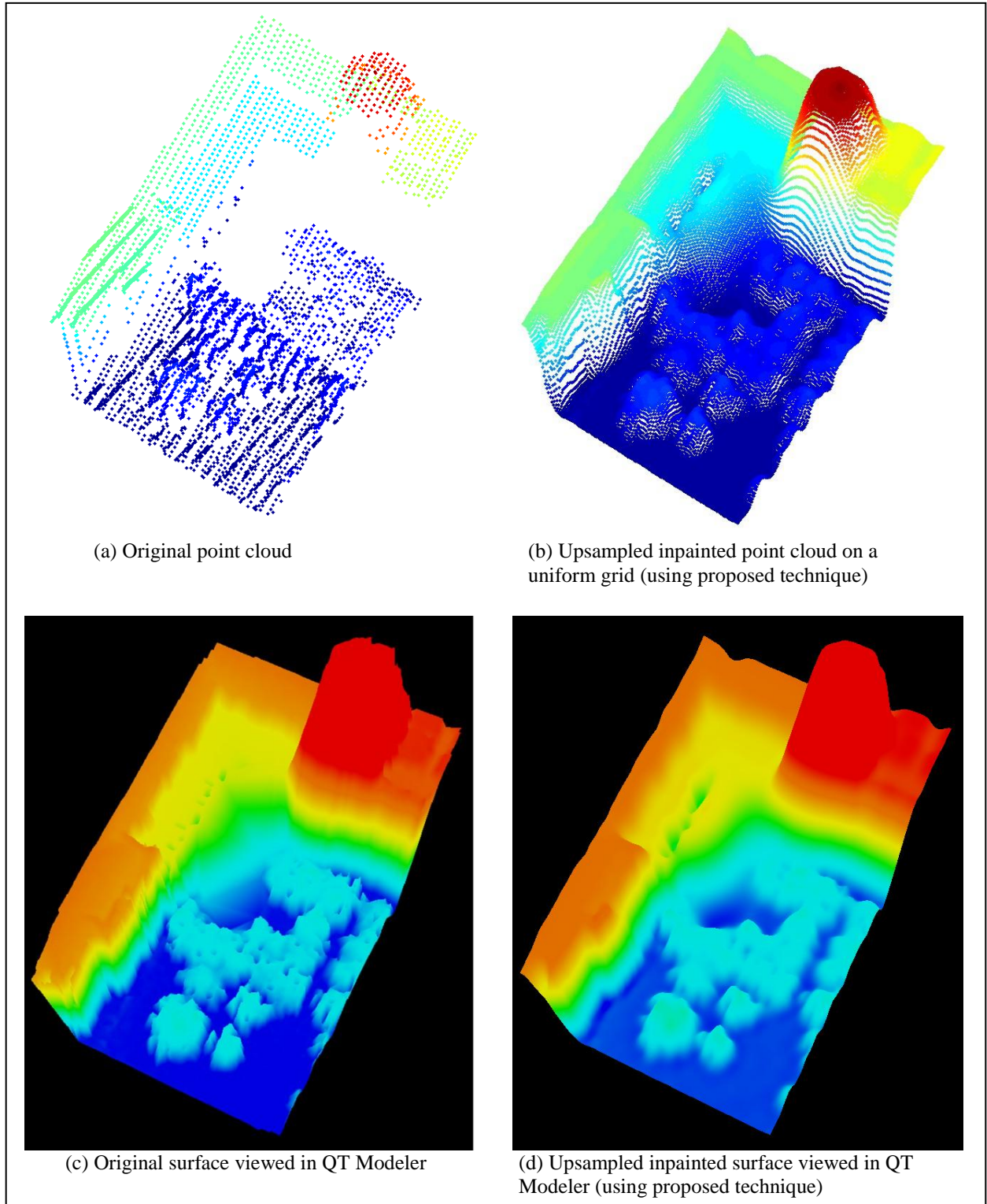


Figure 5. Upsampling and inpainting of building B - Gap 2.

The fourth and final dataset used in this work covers the region around a silo, which allows us to demonstrate the effects of the balance parameter λ . Using PDE-based techniques, not only can we obtain sharp edges, but we can also smooth the noisy data points on curved surfaces. For this experiment, λ is set to 10, i.e., priority is being given to the minimization of the energy functional or smooth diffusion of the homogeneous regions over the fitting between the inpainted point cloud and original point cloud. The results for the silo region are presented in figure 6. The original point cloud collected over the silo region is shown in figure 6a. The distance between grid points is set to 0.1 m, i.e., the data are being upsampled by 4 times along both x and y axes. The upsampled point cloud on a uniform grid is illustrated in figure 6b. The surfaces generated by QT Modeler using standard internal interpolation of the original point cloud and using the upsampled uniform grid point cloud are presented in figures 6c and d, respectively. It is clear that the TIN interpolation of the original point cloud generated jagged structures on the top of the silo, but the PDE-based inpainting technique produced a smooth hemispherical surface, while maintaining flat vertical wall around the silo.

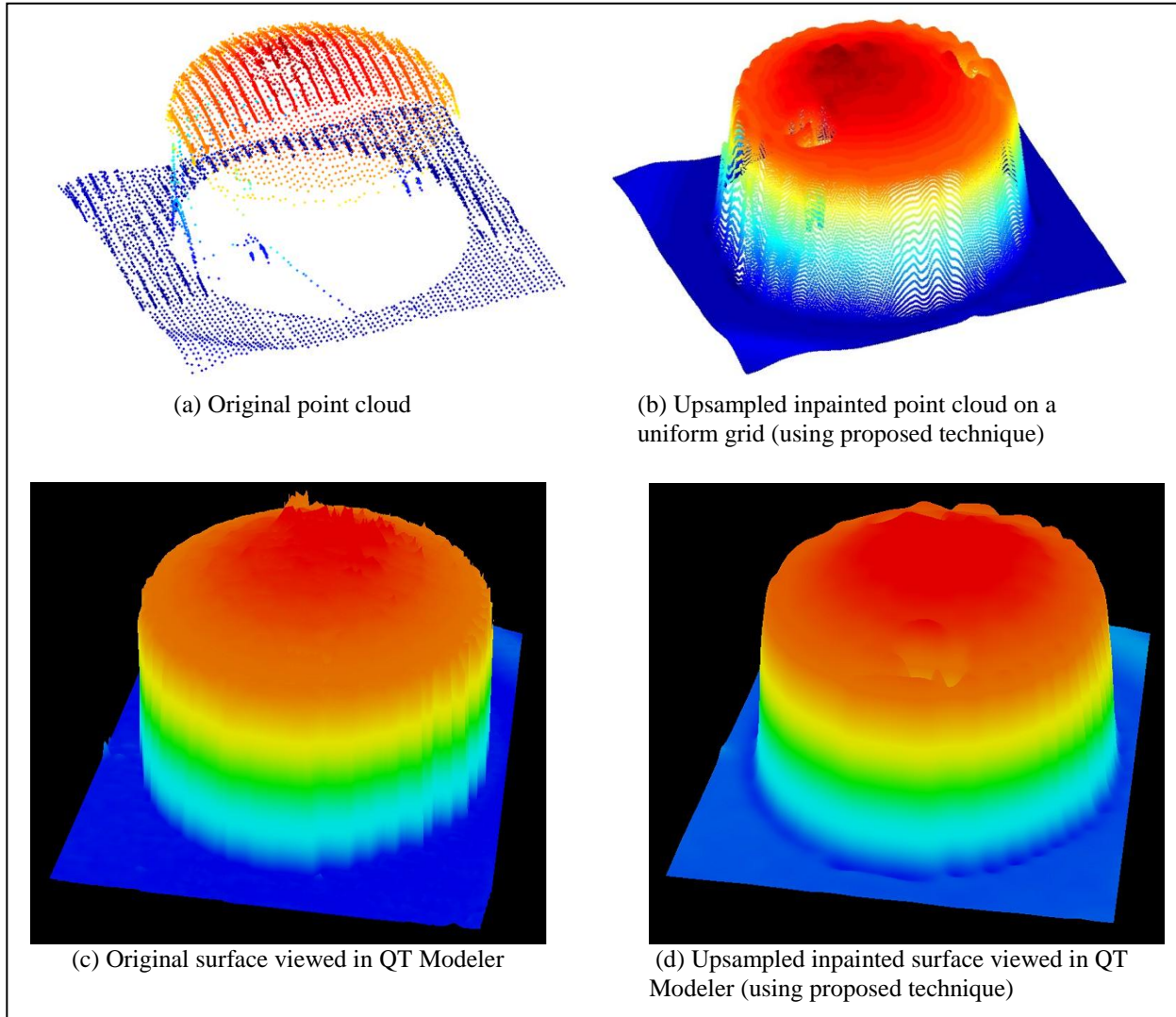


Figure 6. Upsampling and inpainting of a silo.

5. Conclusion

In this report, a fourth-order PDE based inpainting technique is proposed and evaluated to upsample 3-D LiDAR point cloud on to a uniform grid. This technique, called $TV - H^{-1}$ inpainting technique, is based on modified Cahn-Hilliard gradient flow equation. Our simulation results have shown that this technique reduces the draping effects that are usually generated by the standard piece-wise linear interpolation technique used for upsampling the LiDAR point cloud. This technique also ensures smoothness over homogeneous regions, while propagating edges over large holes.

6. References

1. McManamon, P. F. Review of Ladar: A Historic, Yet Emerging, Sensor Technology with Rich Phenomenology. *Optical Engineering* **2012**, 51 (6).
2. Wehr, A. LiDAR: *Airborne and Terrestrial Sensors*; CRC Press, 2008.
3. Guo, Q.; Li, W.; Yu, H.; Alvarez, O. Effects of Topographic Variability and Lidar Sampling Density on Several Dem Interpolation Methods. *Photogrammetric Engineering and Remote Sensing* **2010**, 76 (6).
4. Walker, J. P.; Willgoose, G. R. On the Effect of Digital Elevation Model Accuracy on Hydrology and Geomorphology. *Water Resources Research* **1999**, 35 (7), 2259–2268.
5. Polis, M. F.; McKeown, D. M. Iterative Tin Generation from Digital Elevation Models. *Proceedings of Computer Vision and Pattern Recognition* **1992**, 787–790.
6. Barnett, V. *Interpreting Multivariate Data*; John Wiley, 1981.
7. Cressie, N. Spatial Prediction and Ordinary Kriging. *Mathematical Geology* **1988**, 20 (4), 405–421.
8. Zimmerman, D.; Pavlik, C.; Ruggles, A.; Armstrong, M. P. An Experimental Comparison of Ordinary and Universal Kriging and Inverse Distance Weighting. *Mathematical Geology* **1999**, 31 (4), 375–390.
9. Quick Terrain Modeler: Version 7, User’s Manual, Applied Imagery, 2011.
10. Chan, T. F.; Shen, J. Variational Image Inpainting. *Comm. Pure Applied Math* **2005**, 58, 579–619.
11. Rudin, L. I.; Osher, S.; Fatemi, E. Nonlinear Total Variation Based Noise Removal Algorithms. *Physica D* **1992**, 60 (1–4), 259–268.
12. Burger, M.; He, L.; Schonlieb, C.-B. Cahn-Hilliard Inpainting and a Generalization for Grayvalue Images. *SIAM J. Imaging Sci.* **2009**, 2 (4), 1129–1167.
13. Chan, T. S.; Shen, J.; Vese, L. Variational PDE Models in Image Processing. *Notices of the AMS* **2003**, 50 (1), 14–26.
14. Phillips, D. L. A Technique for the Numerical Solution of Certain Integral Equations of the First Kind. *Journal of ACM* **1962**, 9 (1), 84–97.
15. Fox, C. *An Introduction to Calculus of Variations*; Dover Books on Mathematics, 1987.

16. Tsai, A.; Yezzi, A.; Willsky, A. S. Curve Evolution Implementation of the Mumford-Shah Functional for Image Segmentation, Denoising, Interpolation, and Magnification. *IEEE Transactions on Image Processing* **2001**, *10* (8), 1169–1186.
17. Esedoglu, S.; Shen, J. Digital Inpainting Based on the Mumford-Shah-Euler Image Model. *European Journal on Applied Mathematics* **2002**, *13*, 353–370.
18. Bertozzi, A.; Esedoglu, S.; Gillette, A. Inpainting of Binary Images Using Cahn-Hilliard Equation. *IEEE Transactions on Image Processing* **2007**, *16* (1), 285–291.

Appendix. Code for the Uniform Grid Upsampling of 3-D LiDAR Point Cloud Data

The following is the code used for the uniform grid upsampling of 3-D LiDAR point cloud data.

```
clear all;close all;clc

% READING FILE
% NOTE: Easier to remove all header information (.xyz - .txt file), and read
in numeric data
fid=fopen('504_building.txt');
% read numeric data
points=textscan(fid,'%f %f %f %d %d %d %d %d %d %f %d %d %f');
X=points{1};
Y=points{2};
Z=points{3};
clear points
fclose(fid);

X0=max(X)-min(X) % calculating horizontal span of point cloud (m)
Y0=max(Y)-min(Y) % calculating vertical span of point cloud (m)
% truncating to smaller region within point cloud using X Y utm coordinates
ind=find((Y>3.719447*10^6) & (Y<3.719497*10^6) & (X>2.32715*10^5) & (X<2.32742*10^5
));
I=X(ind);
J=Y(ind);
E=Z(ind);

s=5; % plotting subsampled version of point cloud
figure
plot3k({X(1:s:end) Y(1:s:end) Z(1:s:end)}); axis equal;
set(gca,'xtick',[]);
set(gca,'ytick',[]);
set(gca,'ztick',[]);
set(gcf,'color',[1 1 1]);
xlabel('X');
ylabel('Y');
title('LiDAR Point Cloud');

figure
plot3k({I J E}); axis equal
set(gca,'xtick',[]);
set(gca,'ytick',[]);
set(gca,'ztick',[]);
set(gcf,'color',[1 1 1]);
xlabel('X');
ylabel('Y');
title('Region of LiDAR Point Cloud');

X0=max(I)-min(I);
Y0=max(J)-min(J);
```

```

Ivec=min(I):(max(I)-min(I)):max(I);
Jvec=min(J):(max(J)-min(J)):max(J);

% find minimum distance between points (wrt X and Y, NOT Z coordinates)
% NOTE: D IS NxN MATRIX - MEMORY ISSUE DEPENDING ON SIZE OF POINT CLOUD
% NICE TO FIRST EXAMINE DISTRIBUTION OF DISTANCES BETWEEN ALL POINTS
D=L2_distance([I J]',[I J]');
Dpos=D(D~=0);
N=length(Dpos);
minD=min(min(Dpos))
INF=1e6;
D(D==0)=INF; % set distance between a point and itself to be INF
minDvec=min(D); % find distance between every pair of closest points
[N X]=hist(minDvec,100);
figure
stem(X,N);
xlabel('Distance between every pair of closest points');

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% define spacing (minD or smaller) %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
spacing=0.2; % spacing can be defined by looking at histogram of using minD
LX=max(I)-min(I);
LY=max(J)-min(J);
% form grid for x and y based on minimum spacing between points
gridXvec=min(I):spacing:max(I);
gridYvec=min(J):spacing:max(J);
% making grid even
if mod(length(gridXvec),2)==1
    gridXvec=[gridXvec gridXvec(end)+spacing];
end
if mod(length(gridYvec),2)==1
    gridYvec=[gridYvec gridYvec(end)+spacing];
end
[gridXmatr gridYmatr]=meshgrid(gridXvec,gridYvec);
[r c]=size(gridXmatr);

% let gridX and gridY be centers of each cell in high resolution grid
gridX=reshape(gridXmatr,1,r*c)+spacing/2;
gridY=reshape(gridYmatr,1,r*c)+spacing/2;
D=L2_distance([gridX' gridY']',[I J]');
% find the closest cell center to each of the points
[val ind]=min(D,[],1);
clear D;

Zvals=zeros(1,r*c);
Zvals(ind)=E;
I = find(Zvals~=0);
figure
plot3k({gridX(I)' gridY(I)' Zvals(I)'}); axis equal
set(gca,'xtick',[]);
set(gca,'ytick',[]);
set(gca,'ztick',[]);
set(gcf,'color',[1 1 1]);
xlabel('X');
ylabel('Y');

```

```

title('Region of LiDAR Point Cloud');

X=reshape(gridX,r,c);
Y=reshape(gridY,r,c);
Z=reshape(Zvals,r,c);
mask = zeros(size(Z));
I = find(Z==0);
mask(I) = 1;
figure
colormap(gray(256));
imagesc(flipud(Z)); axis image;
set(gca,'xtick',[]);
set(gca,'ytick',[]);
title('Points in region placed in upsampling-grid');

g=Z;
[m,n]=size(g);
g=double(g);
gmax = max(max(g));
g = g./max(max(g));
clims=[0 1];
figure
imagesc(g,clims); axis image; axis off; colormap(gray);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% Definition of PARAMETERS:%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
h1=1;
h2=1;
dt=100;
epsilon = 0.01;
ep2 = epsilon^2;
lpower=2;
lambda0 = 10^lpower;
c1=1/epsilon;
c2=lambda0;
approxerr=10^(-21);
% Definition of the inpainting domain:
u0 = mask;
lambda = ones(m,n)*lambda0;
[j,k] = find(u0==1);
sd = size(j,1);
for i=1:sd
    lambda(j(i),k(i)) = 0;
end
clear j k sd
figure
imagesc(u0); axis image; axis off; colormap(gray)

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% Part of this code was adopted from the code written by the authors
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% of Ref. 12
% Diagonalize the Laplace Operator by: Lu + uL => D QuQ + QuQ D,
% where Q is nonsingular, the matrix of eigenvectors of L
% and D is a diagonal matrix. We have to compute QuQ.
% This we can do in a fast way by using the fft-transform:
u0 = g;
% Initialization of u and its Fourier transform:
u=u0;

```

```

ubar=fft2(u);
lu0bar=fft2(lambda.*u);
lubar=lu0bar;
curv=zeros(m,n);
%Definition of the Laplace operator with periodic boundary conditions:
L1=1/(h1^2)*(diag(-2*ones(m,1)) + diag(ones(m-1,1),1) + diag(ones(m-1,1),-
1)...
+ diag(ones(1,1),m-1)+ diag(ones(1,1),-m+1));
L2=1/(h2^2)*(diag(-2*ones(n,1)) + diag(ones(n-1,1),1) + diag(ones(n-1,1),-
1)...
+ diag(ones(1,1),n-1)+ diag(ones(1,1),-n+1));
% Computation of the eigenvalues of L1:
for j=1:m
    eigv1(j) = 2*(cos(2*(j-1)*pi/m)-1);
    Lambda1(j,j)=eigv1(j);
end
% Computation of the eigenvalues of L2:
for j=1:n
    eigv2(j) = 2*(cos(2*(j-1)*pi/n)-1);
    Lambda2(j,j)=eigv2(j);
end
Denominator=1/h1^2.*Lambda1*ones(m,n) + 1/h2^2.*ones(m,n)*Lambda2;
% Now we can write the above equation in much simpler way and compute
% the solution ubar
figure
it=0;
err=1;
tic
while err > approxerr
    err
    % Computation of the tv-seminorm:
    % estimate derivatives
    ux = (u(:, [2:n n])-u(:, [1 1:n-1]))/2;
    uy = (u([2:m m], :)-u([1 1:m-1], :))/2;
    uxx = u(:, [2:n n])+u(:, [1 1:n-1])-2*u;
    uyy = u([2:m m], :)+u([1 1:m-1], :)-2*u;
    Dp = u([2:m m], [2:n n])+u([1 1:m-1], [1 1:n-1]);
    Dm = u([1 1:m-1], [2:n n])+u([2:m m], [1 1:n-1]);
    uxy = (Dp-Dm)/4;
    % compute flow
    Num = uxx.*(ep2+uy.^2)-2*ux.*uy.*uxy+uyy.*(ep2+ux.^2);
    Den = (ep2+ux.^2+uy.^2).^(3/2);
    curv = Num./Den;
    lapcurvbar = Denominator.*fft2(curv);
    ubar = ((1+c2*dt+dt*c1*Denominator.^2).*ubar - dt*lapcurvbar ...
+ dt*(lu0bar-lubar))./(1+dt*c2+dt*c1*Denominator.^2);
    unew = real(ifft2(ubar));
    %error in l^2 norm:
    err= sum(sum((unew-u).^2))/(m*n);
    u=unew;
    lubar=fft2(lambda.*u);
    it=it+1;
end
duration=toc;
imagesc(u); axis image; axis off; colormap(gray);
pause(0.01);
disp(['u is inpainted in ' num2str(it) ' iterations and ' num2str(duration)

```

```
'CPU seconds']])

% Displaying the upsampled 3D point cloud on a uniform grid
u = u*gmax/max(max(u));
I = X;
J = Y;
E = u;
figure
plot3k({I J E}); axis equal
set(gca, 'xtick', []);
set(gca, 'ytick', []);
set(gca, 'ztick', []);
set(gcf, 'color', [1 1 1]);
xlabel('X');
ylabel('Y');
title('Region of LiDAR Point Cloud');
```

INTENTIONALLY LEFT BLANK.

List of Symbols, Abbreviations, and Acronyms

2-D	two-dimensional
3-D	three-dimensional
DEMs	digital elevation models
E-L	Euler-Lagrange
LiDAR	light detection and ranging
NN	natural neighbor
PDEs	partial differential equations
QT Modeler	Quick Terrain Modeler
TIN	triangulated irregular network
TV	Total Variational

NO. OF COPIES	ORGANIZATION
1 (PDF)	ADMNSTR DEFNS TECHL INFO CTR ATTN DTIC OCP
1 (PDF)	GOVT PRINTG OFC A MALHOTRA
4 (PDFS)	US ARMY CERDEC NVESD ATTN L GRACEFFO ATTN M GROENERT ATTN J HILGER ATTN J WRIGHT
2 (PDFS)	US ARMY AMRDEC ATTN RDMR WDG I J MILLS ATTN RDMR WDG S D WAAGEN
2 (PDFS)	US ARMY RSRCH OFFICE ATTN RDRL ROI C L DAI ATTN RDRL ROI M J LAVERY
18 (PDFS)	US ARMY RSRCH LAB ATTN IMAL HRA MAIL & RECORDS MGMT ATTN RDRL CIO LL TECHL LIB ATTN RDRL SE P PERCONTI ATTN RDRL SES J EICKE ATTN RDRL SES M D'ONOFRIO ATTN RDRL SES N NASRABADI ATTN RDRL SES E R RAO ATTN RDRL SES E A CHAN ATTN RDRL SES E H KWON ATTN RDRL SES E S YOUNG ATTN RDRL SES E J DAMMANN ATTN RDRL SES E D ROSARIO ATTN RDRL SES E H BRANDT ATTN RDRL SES E S HU ATTN RDRL SES E M THIELKE ATTN RDRL SES E P RAUSS ATTN RDRL SES E P GURRAM ATTN RDRL SES E C REALE