Urban Semantic 3D Reconstruction from Multiview Satellite Imagery

Matthew J. Leotta, Chengjiang Long, Bastien Jacquet, Matthieu Zins, Dan Lipsa,
Jie Shan, Bo Xu, Zhixin Li, Xu Zhang, Shih-Fu Chang,
Matthew Purri, Jia Xue, Kristin Dana

{matt.leotta, chengjiang.long, bastien.jacquet, matthieu.zins, dan.lipsa}@kitware.com,
{jshan, xu1128, li2887}@purdue.edu, {xu.zhang, sc250}@columbia.edu,
{matthew.purri, jia.xue, kristin.dana}@rutgers.edu

Abstract

Methods for automated 3D urban modeling typically result in very dense point clouds or surface meshes derived from either overhead lidar or imagery (multiview stereo). Such models are very large and have no semantic separation of individual structures (i.e. *buildings, bridges) from the terrain. Furthermore, such dense models often appear "melted" and do not capture sharp edges. This paper demonstrates an end-to-end system for segmenting buildings and bridges from terrain and estimating simple, low polygon, textured mesh models of these structures. The approach uses multiview-stereo satellite imagery as a starting point, but this work focuses on segmentation methods and regularized 3D surface extraction. Our work is evaluated on the IARPA CORE3D public data set using the associated ground truth and metrics. A web-based application deployed on AWS runs the algorithms and provides visualization of the results. Both the algorithms and web application are provided as open source software as a resource for further research or product development.*

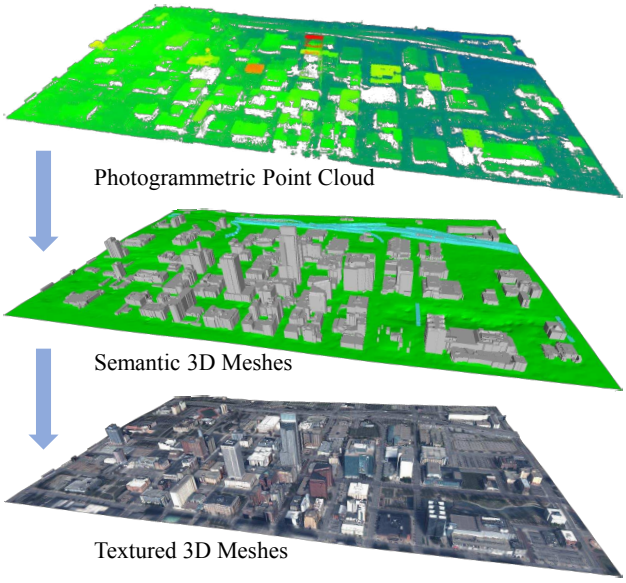


Figure 1. From satellite images and image-derived point clouds, our approach segments and models buildings (grey), bridges (cyan), and terrain (green) before texturing from the imagery.

1. Introduction

Accurate 3D geospatial mesh models of urban areas have uses in a variety of applications including navigation and city planning. There are several technologies for obtaining these 3D models. 3D models scanned with airborne lidar are accurate and precise, but collection productivity is limited by the required low flight height and narrow swath. Although recent Geiger-mode lidar can acquire data at a much higher productivity, its high noise and low penetration capability prevent it from being used for routine production at this time [35]. Moreover, lidar acquisition needs separate optical sensors to collect natural, true color images over the ground. The use of multiple sensors, operated either simultaneously or independently, can cause additional difficulties in data processing, such as precise data co-registration and handling the inconsistency between resolutions.

Multiview satellite images provide another means for reconstructing either point or surface models with photogrammetry. Compared to airborne lidar and images, satellite images can cover much larger areas from a highly stable platform. The high agility of modern satellite platforms and sensors enables multiview coverage in a targeted area within a short period of time. The ever increasing spatial resolution up to 0.3 meter or better is making satellite images comparable with aerial images. Furthermore, satellites can acquire images over areas that are aviation denied or inaccessible.

Point clouds derived by photogrammetry often have similar issues as lidar. Both may have holes and the size of the data is enormous relative to the level of detail provided.

A point cloud uses thousands of points to represent roof planes, whereas most buildings can be modeled with a relatively small number of primitive shapes. Point clouds also lack semantic segmentation of buildings into distinct object models that are separate from the ground terrain, and in many real world applications, surface meshes are desired rather than point clouds. While point clouds can be directly meshed (*e.g*. [15]), the resulting meshes tend to have a "melted" appearance and do not preserve sharp corners found in most buildings.

This paper presents an approach to create semantically segmented 3D building mesh models from commercial multiview satellite imagery. Our system is named *Danesfield* in reference to 3D modeling efforts during WWII [26]. Danesfield is evaluated on the IARPA CORE3D Public Data [6] consisting of DigitalGlobe WorldView 3 imagery over a few cities. Multiview stereo reconstruction is a part of our approach, but not the focus of this paper. We leverage an existing commercial system [30, 14] for satellite multiview stereo to produce a lidar-like point cloud. The focus of this paper is describing a system that starts with such an image-based point cloud, along with the source images, to produce detailed semantically segmented and textured meshes with sharp edges as illustrated in Figure 1.

An equally significant contribution of this paper is its focus on real world operation. Danesfield is able to handle areas at city scale covered by tens of satellite images. It is packaged in a Docker container and deployed on Amazon Web Services (AWS). It has a modern web-based interface with a map, job list, and 3D views to launch algorithms, monitor progress, and visualize the results. Registered users can select an area of interest (AOI) on a map on which to run the algorithms. The algorithms are run on the server and the final 3D textured model is rendered in the browser. All of the work described in this paper—including algorithms, server deployment, and web interface—is released under a permissive open source software licence[1].

The remainder of this paper is organized as follows. Section 2 discusses related work in 3D reconstruction from satellite images and image-derived point clouds. Section 3 describes the technical approach and algorithm details. Section 4 provides results of experiments on the CORE3D public data including comparisons with some alternative methods. Sections 5 discusses the web application and its deployment on cloud infrastructure. Lastly, conclusions are offered in Section 6.

2. Related Work

There has been decades long effort in automated 3D reconstruction from both photogrammetry and computer vision communities. The first step towards 3D reconstruction using images is to generate point clouds through image matching. Using the IARPA CORE3D dataset, [10] intends to handle images collected at different dates with significant lighting and vegetation differences. Unlike previous work, *e.g*., [23], the authors sort all images into pairs and carry out image matching pair by pair. The resultant Digital Surface Models (DSMs) from each pair are then incrementally fused through the k-means clustering technique. The advantage of the method is two fold—it does not need global bundle adjustment and the pairwise DSM can be added to the fusion process once it is available. However, there is no comparative evaluation with reference to ground truth. Using the same CORE3D images, Gong and Fritsch [11] propose a pipeline for processing the benchmark data to digital surface models. Suitable image pairs are selected according to the incidence angle and capture date. A bias-corrected RPC (Rational Polynomial Coefficient) model is then applied to generate epipolar images. The subsequent image matching is carried out by using the tSGM method, the core algorithm of SURE, a software product from nFrames. tSGM is an enhanced version of Semi Global Matching (SGM [32]). The resultant point cloud is filtered to generate a DSM [11]. The final DSM can present small structures comparable with the ones from airborne lidar. Gong and Fritsch [11] conclude that landcover (especially vegetation) changes among different seasons can not be handled easily. Similarly, point cloud filtering needs to be fine tuned to assure a quality DSM. The performance of the entire pipeline needs to be further evaluated by testing more varying urban areas [11]. After identifying some common but challenging difficulties in working with satellite images due to seasonal appearance differences and scene changes, [4] present a lightweight solution based on recent convolutional neural network (CNN) models. It recognizes the need for reliable image pair selection in multiview 3D reconstruction. To promote future studies, the authors present a novel large-scale public data set for semantic stereo with multiview, multi-band, incidental satellite images [4] [6].

Once the point clouds are created they will follow a procedure similar to the one applied to lidar point clouds for 3D reconstruction. According to [2], surface reconstruction is a process by which a 3D object is inferred, or reconstructed, from a collection of discrete points. It states that reconstruction is ill-posed unless certain priors are utilized. Depending on the scenes to be recovered, a variety of cues or priors should be utilized, including geometric shape, smoothness, symmetry, topology, structures, primitive relations etc. The quickly growing data volume from different types of sensors means that dealing with city scale scenes will become more common. Recovering such large-scale scenes is a stimulating scientific challenge, especially when this needs to be done in an online environment [2]. As a conceptual and feasibility study, Haene *et al*. [12] propose a

[1] `https://github.com/kitware/danesfield-app`

mathematical framework to formulate and solve a joint segmentation and dense reconstruction problem. They argue that knowing either the semantic class or the surface direction of a shape would help inferring the other. Tests on a few simple data sets with limited complexity demonstrate somewhat consistent results. The framework developed by [22] treats polygonal surface reconstruction from point clouds as a binary labeling problem. They seek an optimal combination of the intersected planes under manifold and watertight constraints. However, the work is mostly based on separated, individual buildings with a rather clean neighborhood, which leaves open the scalability of the approach and the inclusion of geometric primitives other than planes. To handle poor-quality data, missing data, noise and outliers in point clouds, Zhang *et al*. [43] first consolidate classified building points and reconstruct the buildings from the consolidated point clouds. However, the method is unable to explicitly create large-scale urban building models with geometric 3D-primitives like planes, cylinders, spheres or cones. In a similar study, authors of [44] attempt to use 3D CNN for classifying large scale point clouds. However, its performance is very much dependant how well the 3D CNN is trained, especially when the urban roofs have varieties of colors, shapes and sizes. In addition, no wireframe or boundary reconstruction based on geometric primitives is achieved. [9] use stereo pairs of satellite images to simultaneously retrieve geometry and semantics of cities. Although several large scale cities are tested and the computation time is reasonable (from a few to tens of minutes), buildings can only be modeled by piece-wise flat planes and there is no texture mapping applied.

3. Approach

The flowchart in Figure 2 gives an overview of our approach. Co-collected panchromatic (PAN) and multispectral (MSI) WorldView 3 (WV3) image pairs are fed into *Intersect Dimension* [30, 14], a commercial multiview stereo software system developed by Raytheon. That software provides the bundle adjusted RPC camera models and a dense point cloud (see Figure 1, top). The point cloud and imagery are then normalized by various pre-processing or normalization steps described in Section 3.1. In brief, after the point cloud is rendered into a DSM, a digital terrain model (DTM) is estimated and subtracted from the DSM to create a normalized digital surface model (nDSM). The MSI images are corrected to top of atmosphere reflectance and orthorectified using the DSM. A normalized difference vegetation index (NDVI) is also computed from these orthorectified images.

The next step is semantic segmentation of the orthorectified imagery into categories of building, elevated road/bridge, and background as described in Section 3.2. Multiple deep networks are compared for this step and

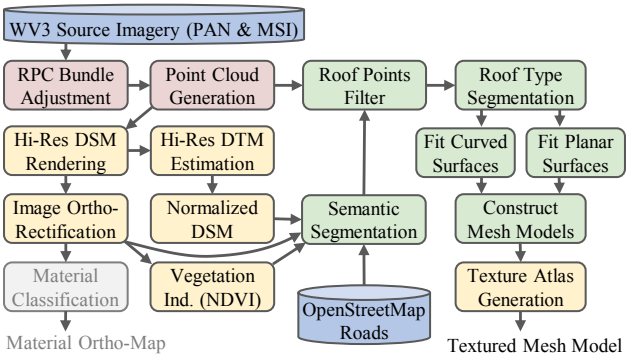


Figure 2. Processing stages of the proposed system. Data sources are shown in blue; the commercial satellite MVS product [30] is in red; standard pre- and post-processing steps are in yellow; and the novel algorithm contributions are in green. Material classification is also part of the complete system but covered in a separate paper (Purri *et al*. [27]).

OpenStreetMap road network vector data is used to separate building and elevated road/bridge categories.

The image segmentation results are then mapped back to the point cloud and used to select the building roof points from the original point cloud (likewise for bridge surface points). Points are further segmented by roof geometry type in Section 3.3. Specifically, a PointNet [28, 29] architecture classifies each point in the subset of roof points into flat, sloped, cylindrical, and spherical categories. The curved roof surfaces, which are considerably less common, are fit first with local cylinder and sphere models as dictated by the segmentation result. The algorithm further segments the remaining planar points into local planar patches based on position, estimated surface normal, and color.

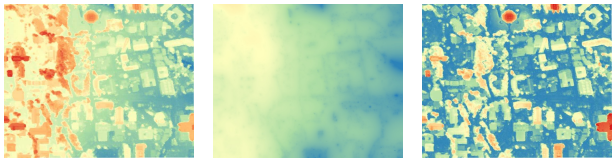
In Section 3.4, planar patches are intersected to find roof edges and peaks with additional regularization constraints applied to form complex roof geometries. The roof geometry is extruded downward to form complete building meshes. A triangulated regular grid on the estimated DTM forms the base terrain mesh.

Lastly, we apply texture mapping algorithms (see Section 3.5) to build a texture atlas for the building, bridge, and terrain meshes and to populate the texture map with texture projected from the source images. Special care is given to handle occlusion reasoning during texture map generation.

3.1. Normalization of Input Data

Multiple forms of data normalization are used to preprocess the imagery and point cloud data into aligned, standard coordinates. The first normalization applies to the height of the geometry. We convert the point cloud into a digital surface model (DSM) to provide a map of absolute height. However, *relative* height above the terrain is a more useful input to semantic segmentation. Thus, we estimate a bare earth digital terrain model (DTM) from the DSM by adopting the cloth simulation idea from Zhang *et*

al. [45] to the 2.5D image domain. A normalized DSM (nDSM) gives the relative height of structures above the terrain: $\text{nDSM} = \text{DSM} - \text{DTM}$. Figure 3 shows an example of a DSM, estimated DTM, and nDSM. The nDSM gives all structures a relative height above a common Z=0 ground plane—useful for building segmentation. The resultant bare earth DTM also gives rise to a terrain mesh via triangulation on a regular grid. The terrain mesh is a 3D ground layer on which final building and bridge models are added. See Figure 1, center green layer.



(a) DSM (b) DTM (c) nDSM

Figure 3. Input DSM (a) is filtered with [45] to estimate a DTM (b). The nDSM (c) is DSM - DTM.

The second normalization occurs in image radiometry. Since images are taken at different times from different viewing angles under different illumination angles, their intensity must be normalized to achieve reliable semantic segmentation and realistic, consistent texture mapping. For spectral band λ, the normalized intensity, L_λ, is computed from the input pixel value, I_λ, as

$$L_\lambda = \frac{G_\lambda A_\lambda I_\lambda}{B_\lambda} + O_\lambda, \tag{1}$$

where the G_λ and O_λ are the absolute radiometric calibration gains and offsets respectively. These band-dependent adjustment factors are provided in the DigitalGlobe WV3 specification. The A_λ and B_λ are the abscal factor and the effective bandwidth. These values are image specific and can be found in the image metadata (IMD) file provided with each image. The radiometrically calibrated image is further converted to top of atmosphere reflectance, ρ_λ as

$$\rho_\lambda = \frac{L_\lambda d^2 \pi}{E_\lambda \cos(\theta_s)}, \tag{2}$$

where L_λ is the band-dependent at-sensor radiance from equation (1), d is the Earth-Sun distance, E_λ is the band-dependent solar irradiance, and θ_s is the solar zenith angle.

A third form of normalization is image registration into a common orthorectified space such that each pixel in each image represents the same world location. Such alignment is nontrivial for images taken from different viewing angles in complex 3D environments. Our approach projects each image onto the high resolution DSM to orthorectify not only with the terrain but also using all building surfaces. With complex DSM geometry, areas of occlusion will occur in orthorecified results, but these can be masked out in further processing.

3.2. Semantic Segmentation for Buildings

The task of semantic segmentation is to segment the scene into semantic categories like ground, buildings, vegetation, roads, *etc*. In particular, our primary objective is to create masks to separate buildings and elevated roads from the background for 3D reconstruction of these two classes. Our approach adopts a simple binary segmentation of *structures* versus background and then uses matching with OpenStreetMap road data to further segment *structures* into buildings and elevated roads/bridges. Trees and other vegetation are considered background, even when elevated above the ground.

Semantic segmentation fuses input from orthorectified MSI and nDSM. Furthermore, we found the normalized difference vegetation index (NDVI) to be a useful input as well. NDVI is derived directly from the red and near-IR bands of MSI as $NDVI = \frac{NIR-Red}{NIR+Red}$. This normalized value correlates strongly with the presence of vegetation [37]. We compute NDVI for each orthorectified MSI and average them.

Multiple methods are compared for semantic segmentation. As a baseline we consider a simple approach of applying thresholds with morphological cleanup and hysteresis within connected components to both the nDSM and NDVI images. In short, pixels are *structures* if above the ground (nDSM > 4 meters with hysteresis down to 2 meters) and not vegetation (NDVI < 0.1 with hysteresis up to 0.2).

We also consider multiple deep neural network structures for semantic segmentation, including: GoogLeNet [36], DenseUNet (a combination of U-Net [31] and DenseNet [13]), and PSPNet [46].

The GoogLeNet [36] based semantic segmentation network takes RGB (a subset of MSI bands), nDSM, and NDVI as input and outputs a high-resolution building mask. To keep a high resolution of the output mask, the pool1, pool3 and pool5 layers are removed from the original GoogLeNet model. All the fully connected layers are also removed. The final layer is an 1×1 convolutional layer with 2 channels followed by Softmax function as a binary classifier for each pixel in the final feature map. Therefore, the output mask is 1/4 the width and 1/4 the height of the input.

The U-Net [31] architecture is considered because it requires very few annotated images and achieves very good performance in other semantic segmentation domains. Inspired by DenseNet [13], we propose a modified version of U-Net, which we call "DenseUNet" (see Figure 4), to replace the original conv1, conv2, conv3, conv4 and center in U-Net with 5 dense blocks, and also replace the Max-pooling layer in U-Net with a transition layer to connect two consecutive dense blocks. Our proposed DenseUNet combines dense blocks that perform iterative concatenation of feature maps with the advantages of the U-Net architecture for semantic segmentation. Different from GoogLeNet,

the output mask is the same width and the same height of the input.

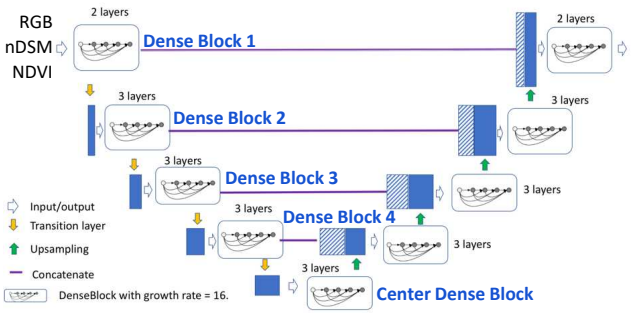


Figure 4. The architecture of DenseUNet.

Lastly, we consider PSPNet [46] because it outperforms other deep neural networks on multiple semantic segmentation benchmarks like ImageNet Scene Parsing Challenge Dataset, PASCAL VOC Dataset, ADE20K Dataset and Cityspace Dataset. The Pyramid Pooling Module in PSPNet adopts the average pooling at four scale levels, *i.e.*, 1x1, 2x2, 3x3, and 6x6 (kernel size, stride), so that it is good at extracting both global and sub-regional contextual features, which are fused as the global contextual prior for pixel-level scene parsing and semantic segmentation under complicated scenes. Like DenseUNet, the output mask shares the same width and height with the input.

An additional post-process step is applied to seperate elevated roads and bridges from building masks. We query OpenStreetMap (OSM) for road vectors with a "bridge" label and rasterize them with a width of 10 meters. Buildings in the mask that overlap with the rasterized road are removed, and the rasterized bridges are used as the mask for the bridge and elevated road class.

Final segmentation masks are mapped back to the point cloud to extract only the subset of points on building roofs for further processing, and likewise for bridges.

3.3. Roof Shape Segmentation

To discover the geometric shape of the roof, we propose shape segmentation to recognize different primitive shapes in the point cloud. Each roof may locally consist of different types of surfaces (flat, sloped, cylindrical and spherical). The proposed shape segmentation method assigns a shape label to *each point* in the cloud. To determine each label, the algorithm needs to consider the overall shape of the roof (global information) and the location of the point within the roof (local information).

We adopt PointNet [28] to classify the points. A PointNet module processes each point with a multi-layer perception network to get a local feature. A symmetric function (*e.g.* element-wise max pooling) is applied to all the local features to get a global feature. The local and the global features are aggregated for use in shape prediction.

A challenge existing in training this network is obtaining balanced training data. There are often far more planar roofs than curved roofs in actual buildings, and the model is easily biased toward planes. Simulation of additional curved roofs (spherical and cylindrical) to balance the training data is critical. We initially synthesized training data by sampling points from ideal spheres and cylinders with added Gaussian noise. However, this performs poorly (See Section 4.2) because the simulated noise statistics differ from noise in real data. Instead, we propose to synthesize curved surfaces by starting with planar roof patches sampled from the real data and "bending" the data into cylindrical or spherical sections. The original noise characteristics of the planar roof are mostly preserved in the synthesized roof, resulting in better performance.

3.4. Building Reconstruction

Building reconstruction uses multiple geometric primitives to form a 3D watertight boundary representation for buildings. We 1) assign the points to specify shape instances with RANSAC, 2) find the boundary of each shape instance, 3) refine the boundary and guarantee the continuity of the roof by checking the topology of the building roof.

To extract a shape instance from the point cloud, points in each shape type (as segmented in Section 3.3) are handled separately. We iteratively apply RANSAC [41, 7] to the point cloud. It detects an instance of the shape, fits the parameters, removes the inliers from further consideration, and continues to find the next object in the remaining points. Multiple cues of position, normal, and color in hypothesis validation are used to determine planar surfaces. For each inlier, we compute a score as the product of point-plane distance, angular difference of point and plane normals, and color difference to the average color of inliers. The weights are averaged over all points and compared to a threshold to decide whether to accept each hypothesis.

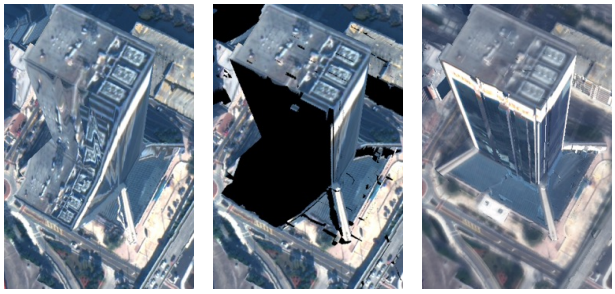
An alpha-shape hull [3, 33] of the inliers of each shape instance is used to estimate the boundary of that shape. However, due to noise on the point cloud as well as the fitting error, the boundary of shapes in one roof may not be continuous, especially near the ridge of the roof, and the shape boundary itself could be irregular. To make a solid roof model, we apply a hierarchical roof topology tree [40, 39] which considers the topology of the roof as a tree structure and helps build solid roof ridges and regular roof boundaries. Given the refined boundary of each shape instance, we use Delaunay triangulation [8] to construct the triangular meshes for texture mapping. We extrude the roof down to lower roof levels and, ultimately, to the ground (DTM), to form the facades. After these steps, we group all corresponding building components including roof surfaces, walls, and bottom surfaces to form a solid building with boundary representation.

3.5. Texture Mapping

The task of texture mapping consists of generating textures for our reconstructed 3D models from the satellite images. We adapt the work of Pages *et al*. [24, 25] from human body to urban 3D models and large satellite images.

While significant work exists in optimal mesh UV unwrapping [20, 19, 21], we take a simpler approach by creating seams at all plane boundaries. This avoids texture distortion but results in many disjoint rectangular patches and some of a patches of more complex shape. Optimal texture packing is a complex problem, but some very efficient heuristics exist. In our case, we rotate faces so the longest edge is horizontal and pack in order of increasing height.

Occlusion is an important issue to consider for satellite images; some parts of the scene are hidden or shadowed. Naïvely projecting images results in occluded areas filled with the wrong texture as shown in Figure 5. Likewise, selecting pixels from regions in shadow results in poor texture. Both shadows and occlusion are easy to detect with a Z-buffer depth test. Occlusion testing uses an RPC camera for depth tests while shadow testing uses a virtual affine camera aligned with the sun angle. Using shadow and occlusion masks allows the algorithm to select which image is best for sampling texture at each surface location.



(a) naïve (b) mask occlusions (c) combine images

Figure 5. Texture map occlusion detection and filling

4. Experimental Results

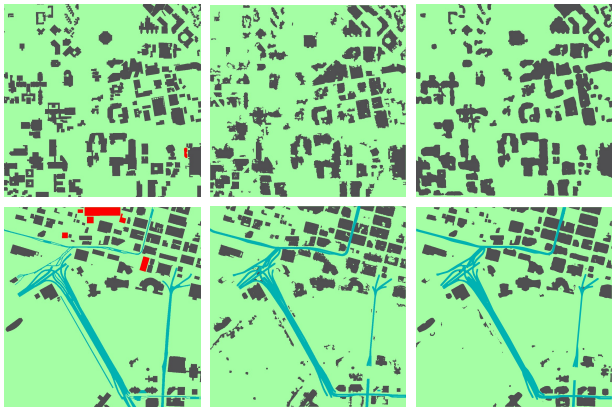
Danesfield was evaluated on the IARPA CORE3D Public Data [6], which consists of multiview WV3 PAN and MSI image pairs for three U.S. Cities: San Diego, CA (46 views); Jacksonville, FL (29 views); and Omaha, NE (45 views). Ground truth DSM and class labels (building/elevated road/background) are provided for only two areas of interest: the UCSD campus (1 km^2) and downtown Jacksonville (2 km^2). Ground truth is derived from high resolution airborne lidar and manual annotations. We evaluate our system quantitatively on the CORE3D ground truth AOIs using metrics proposed by Bosch *et al*. [5]. We quantitatively compare building segmentation methods in Section 4.1 and roof shape segmentation in Section 4.2 before demonstrating overall system performance in Section 4.3. We also provide additional qualitative results on other areas covered by the imagery.

4.1. Building Semantic Segmentation

We compare three deep learning architectures, GoogLeNet, DenseUNet and PSPNet as well as the baseline threshold method on two CORE3D AOIs. The training data is collected from regions that do not overlap with the test AOIs. The OpenStreetMap building mask is used as a training mask. Note that OpenStreetMap may contain annotation errors. However, the final segmentation models seem to work reasonably well. The results are summarized in Table 1 and Figure 6.

	UCSD			Jacksonville		
Method	P	R	IoU	P	R	IoU
Threshold	0.69	0.90	0.64	0.83	0.83	0.71
GoogLeNet	0.84	0.89	0.75	0.79	0.80	0.66
DenseUNet	0.87	0.85	0.75	0.63	0.86	0.57
PSPNet	0.82	0.90	0.75	0.76	0.87	0.69

Table 1. Comparison of building segmentation performance across methods on CORE3D UCSD and Jacksonville AOIs. Scores are precision (P), recall (R), and intersection over union (IoU).



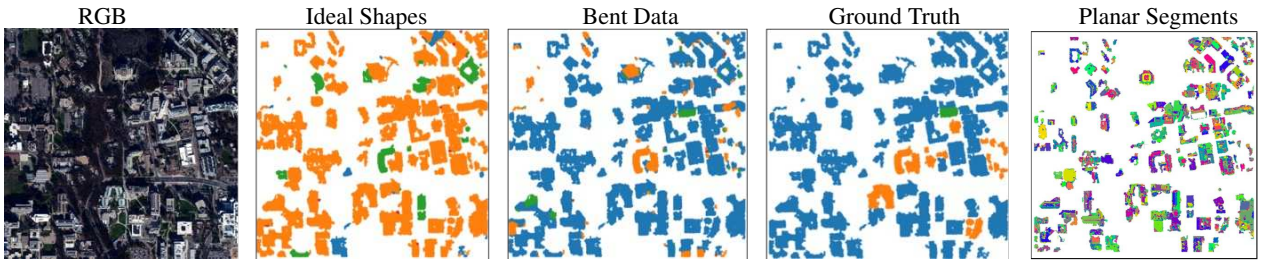
Ground Truth Threshold PSPNet

Figure 6. Visualization of semantic segmentation on both UCSD (top) and Jacksonville (bottom). Red areas are ignored in scoring.

As we can see, all three deep learning architectures perform similarly and much better than the threshold baseline in term of both precision and IoU on UCSD. The threshold baseline works marginally better than deep learning based methods on Jacksonville due to our training data being mostly collected from rural regions with its distribution significantly different from that of Jacksonville which contains elevated roads and skyscrapers.

4.2. Roof Shape Segmentation

To evaluate the performance of the proposed roof shape segmentation algorithm, we manually annotate the roof type



(a) UCSD Campus - Roof type accuracy improves from 10.5% (ideal) to 87.1% (bent).

(b) Downtown Jacksonville - Roof type accuracy improves from 13.1% (ideal) to 95.9% (bent).

Figure 7. Roof type point cloud segmentation results comparing training on ideal synthetic shapes to real planar data bent to curved shapes. Classes are flat (blue), sloped (orange), cylindrical (green), and spherical (red). The last column shows the subsequent segmentation of planar points into specific plane instances (shown in random colors).

label for all the buildings in UCSD Campus and Downtown Jacksonville. Four types of roofs—flat, sloped, cylindrical, and spherical—are considered (the 4th column in Figure 7). We compare a model trained with synthesized point clouds using our bending method (Section 3.3) to a model trained with points sampled from ideal shape with Gaussian noise. Each shape has 300 point clouds for training (1,200 for each training set). PointNet++ [29] is chosen as the training model.

We first run the cluster extraction method in PCL [1] to separate isolated point clouds into different clusters based on the Euclidean distance. Each cluster is sent to the segmentation model to assign a shape label to each point in the point cloud. The predicted results are shown in Figure 7. The model learned with ideal shape always recognizes planar roofs as sloped roofs. The error is due to attached structures on top of a flat roof and the reconstruction error at the edge of the roof. The model trained with our synthesized roof achieves a much better result. It is even able to recognize a cylindrical roof in UCSD campus (marked in green).

4.3. Overall Reconstruction Performance

Performance of the final 3D models with reference to the CORE3D ground truth [6] and metrics [5] is given in Table 2. Qualitative results, showing both semantic segments and texture, are shown in Figure 8. The scores indicate that our current system emphasizes correctness over completeness.

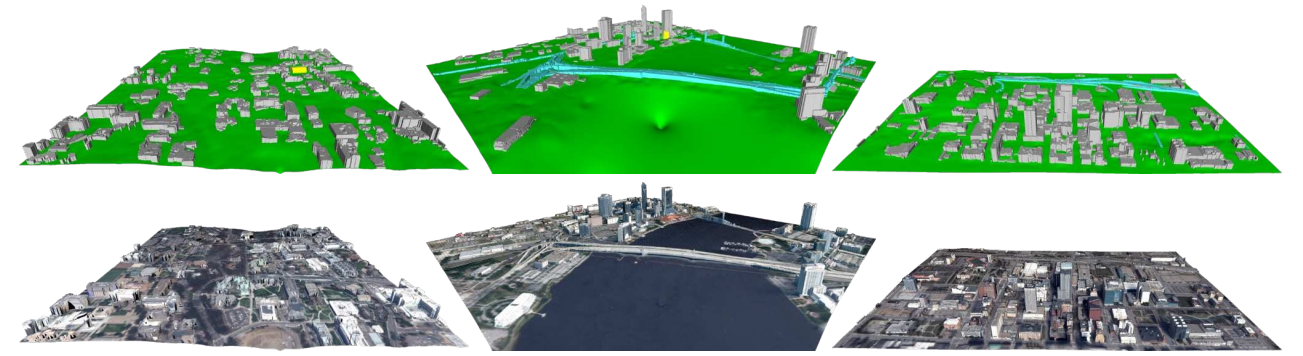
Metric	UCSD	Jacksonville
2D Correctness	0.91	0.9
2D Completeness	0.75	0.71
3D Correctness	0.88	0.91
3D Completeness	0.75	0.75
Geolocation Error (m)	1.58	2.24
Z-RMSE (m)	1.29	0.6
H-RMSE (m)	1.8	2.06
Run Time (hr/km^2)	4.8	2.28

Table 2. Quantitative metrics [5] for the complete system.

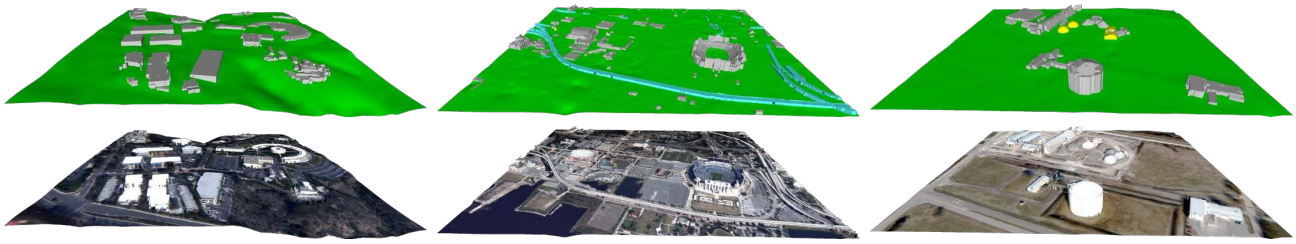
5. Web-based Deployment

The Danesfield web client is a modern web application based on Vue.js [42]. The user interface utilizes Vuetify [38] UI library and embraces Material Design with its UI/UX patterns. The application leverages GeoJS [18] for geospatial map related visualization of base maps, raster, and vector data, and uses VTK.js [17] for 3D model visualization. A screen capture of the user interface is shown in Figure 9. The left pane allows for selection of data to process or visualize. The center shows a raster segmentation image for the selected AOI overlaid on a map. The right shows an interactive 3D view with the final texture mapped model.

The Danesfield back-end builds on the Girder [16] data management platform to manage the source imagery and computed products. It leverages Celery [34] for distributed job management. The entire system was deployed to a single GPU-enabled AWS instance (p3.2xlarge) providing eight virtual CPUs, 61GB of main memory, and one



UCSD Campus – 0.97 km^2 | Downtown Jacksonville – 2.04 km^2 | Downtown Omaha – 1.36 km^2



Primordial Genetics – 0.35 km^2 San Diego, CA | TIAA Bank Field – 1.64 km^2 Jacksonville, FL | Watco Omaha Terminal – 0.15 km^2 Omaha, NE

Figure 8. Final 3D mesh results several regions in the IARPA CORE3D dataset. Meshes on top are colored by semantic category: green for terrain, cyan for elevated roadways, yellow for curved roof buildings, and gray for flat roof buildings. Bottom meshes are textured.

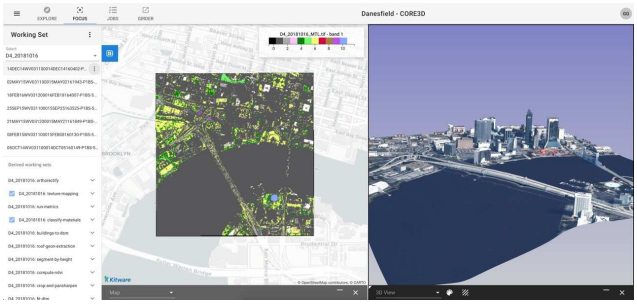


Figure 9. Web-based Danesfield application running on AWS.

NVIDIA Tesla V100 with 16GB of GPU memory at a current cost of $3.06 USD per hour. On average Danesfield processes one km^2 of data (30-50 WV3 images at about 0.3m/pixel) in 3.26 hours, so the total operating cost is just under $10 USD per km^2. The majority of the processing time is spent in the point could generation stage.

In the future our system could be made considerably less expensive to operate by splitting the processing across multiple AWS instances that are brought up and down on demand and running only the GPU jobs on the GPU instance. The p3.2xlarge instance is much more expensive than others due to the GPU that is only needed by deep learning steps.

6. Conclusion

We presented an algorithmic pipeline to create semantic 3D models from multiview satellite images. Once the photogrammetric point clouds are created, geometric, radiometric and terrain normalization processes enable accurate semantic segmentation over the scene. The subsequent 3D reconstruction is based on a model- and data-driven (learning and non-learning) coupled approach, while the former determines the roof manifold type and the latter forms the exact geometry.

The entire algorithmic pipeline has been integrated into a complete, operational system. Evaluation by independent users on additional non-public WV3 data has demonstrated that our system generalized beyond the CORE3D public data. With the exception of point cloud generation, the entire algorithmic pipeline, deployment framework, and web application are open source (Apache 2.0 license). We aim to adapt existing open multiview stereo software to the satellite imagery domain to establish a completely end-to-end open source pipeline in the future. There is room for improvement in various areas, but the current system provides the community a firm foundation to build future research or even commercial products.

This research is based upon work supported by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), via Department of Interior/Interior Business Center (DOI/IBC) contract number D17PC00286. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the ODNI, IARPA, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon.

References

[1] M. Alexa, J. Behr, D. Cohen-Or, S. Fleishman, D. Levin, and C. T. Silva. Computing and rendering point set surfaces. *IEEE Transactions on Visualization and Computer Graphics*, 9(1):3–15, Jan 2003.

[2] M. Berger, A. Tagliasacchi, L. M. Seversky, P. Alliez, G. Guennebaud, J. A. Levine, A. Sharf, and C. T. Silva. A survey of surface reconstruction from point clouds. In *Computer Graphics Forum*, volume 36, pages 301–329. Wiley Online Library, 2017.

[3] F. Bernardini and C. L. Bajaj. Sampling and reconstructing manifolds using alpha-shapes. In *Canadian Conference on Computational Geometry*, 1997.

[4] M. Bosch, K. Foster, G. Christie, S. Wang, G. D. Hager, and M. Z. Brown. Semantic stereo for incidental satellite images. *CoRR*, abs/1811.08739, 2018.

[5] M. Bosch, A. Leichtman, D. Chilcott, H. Goldberg, and M. Brown. Metric evaluation pipeline for 3d modeling of urban scenes. *International Archives of the Photogrammetry, Remote Sensing & Spatial Information Sciences*, 42, 2017.

[6] M. Brown, H. Goldberg, K. Foster, A. Leichtman, S. Wang, S. Hagstrom, M. Bosch, and S. Almes. Large-scale public lidar and satellite image data set for urban semantic labeling. In *Laser Radar Technology and Applications XXIII*, volume 10636. International Society for Optics and Photonics, 2018.

[7] S. Choi, T. Kim, and W. Yu. Performance evaluation of ransac family. In *Proceedings of the British Machine Vision Conference 2009*, volume 24, 01 2009.

[8] B. Delaunay et al. Sur la sphere vide. *Izv. Akad. Nauk SSSR, Otdelenie Matematicheskii i Estestvennyka Nauk*, 7(793-800):1–2, 1934.

[9] L. Duan and F. Lafarge. Towards large-scale city reconstruction from satellites. In *European Conference on Computer Vision*, pages 89–104. Springer, 2016.

[10] G. Facciolo, C. de Franchis, and E. Meinhardt. Automatic 3d reconstruction from multi-date satellite images. *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 1542–1551, 2017.

[11] K. Gong and D. Fritsch. Point cloud and digital surface model generation from high resolution multiple view stereo satellite imagery. *International Archives of the Photogrammetry, Remote Sensing & Spatial Information Sciences*, 42(2), 2018.

[12] C. Häne, C. Zach, A. Cohen, and M. Pollefeys. Dense semantic 3d reconstruction. *IEEE transactions on pattern analysis and machine intelligence*, 39(9):1730–1743, 2017.

[13] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.

[14] A. Kalinski. 3d models from 2d imagery: Raytheon's intersect dimension. Geospatial Solutions, August 2014. Online; accessed 3 Mar. 2019.

[15] M. Kazhdan, M. Bolitho, and H. Hoppe. Poisson surface reconstruction. In *Proceedings of the fourth Eurographics symposium on Geometry processing*, volume 7, 2006.

[16] Kitware. Girder: a data management platform. `https://girder.readthedocs.io/en/stable/`. Online; accessed 4 Mar. 2019.

[17] Kitware. Visualize your data with vtk.js. `https://kitware.github.io/vtk-js/index.html`. Online; accessed 4 Mar. 2019.

[18] Kitware and Epidemico. GeoJS—a javascript library for visualizing geospatial data in a browser. `https://opengeoscience.github.io/geojs/`. Online; accessed 4 Mar. 2019.

[19] B. Lévy. Constrained texture mapping for polygonal meshes. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '01, pages 417–424, 2001.

[20] B. Lévy and J.-L. Mallet. Non-distorted texture mapping for sheared triangulated meshes. In *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '98, pages 343–352, 1998.

[21] B. Lévy, S. Petitjean, N. Ray, and J. Maillot. Least squares conformal maps for automatic texture atlas generation. *ACM Trans. Graph.*, 21(3):362–371, July 2002.

[22] L. Nan and P. Wonka. Polyfit: Polygonal surface reconstruction from point clouds. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[23] O. C. Ozcanli, Y. Dong, J. L. Mundy, H. Webb, R. Hammoud, and V. Tom. A comparison of stereo and multiview 3-d reconstruction using cross-sensor satellite imagery. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2015.

[24] R. Pagés, S. Arnaldo, F. Morán, and D. Berjón. Composition of texture atlases for 3d mesh multi-texturing. In *Eurographics Italian Chapter Conference*, 2010.

[25] R. Pagés, D. Berjón, F. Morán Burgos, and N. García. Seamless, static multi-texturing of 3d meshes. *Computer Graphics Forum*, 34, 10 2014.

[26] A. W. Pearson. Allied military model making during world war ii. *Cartography and Geographic Information Science*, 29(3):227–242, 2002.

[27] M. Purri, J. Xue, K. Dana, M. Leotta, D. Lipsa, Z. Li, B. Xu, and J. Shan. Material segmentation of multi-view satellite imagery. *CoRR/Arxiv*, 2019.

[28] C. R. Qi, H. Su, K. Mo, and L. J. Guibas. PointNet: Deep Learning on Point Sets for 3d Classification and Segmentation. In *CVPR*, 2016.

[29] C. R. Qi, L. Yi, H. Su, and L. J. Guibas. PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, 2017.

[30] Raytheon. Raytheon analytics: Intersect dimension. `https://www.raytheon.com/capabilities/products/analytics`. Online; accessed 3 Mar. 2019.

[31] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[32] M. Rothermel, K. Wenzel, D. Fritsch, and N. Haala. Sure: Photogrammetric surface reconstruction from imagery. In *Proceedings LC3D Workshop, Berlin*, volume 8, page 2, 2012.

[33] A. Sampath and J. Shan. Building boundary tracing and regularization from airborne lidar point clouds. *Photogrammetric Engineering & Remote Sensing*, 73(7):805–812, 2007.

[34] A. Solem and contributors. Celery: Distributed task queue. `http://www.celeryproject.org/`. Online; accessed 4 Mar. 2019.

[35] J. M. Stoker, Q. A. Abdullah, A. Nayegandhi, and J. Winehouse. Evaluation of single photon and geiger mode lidar for the 3d elevation program. *Remote Sensing*, 8(9), 2016.

[36] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

[37] C. J. Tucker. Red and photographic infrared linear combinations for monitoring vegetation. *Remote sensing of Environment*, 8(2):127–150, 1979.

[38] Vuetify, LLC. Vue.js material design component framework—vuetify.js. `https://vuetifyjs.com/en/`. Online; accessed 4 Mar. 2019.

[39] B. Xiong, S. O. Elberink, and G. Vosselman. A graph edit dictionary for correcting errors in roof topology graphs reconstructed from point clouds. *ISPRS Journal of photogrammetry and remote sensing*, 93:227–242, 2014.

[40] B. Xu, W. Jiang, and L. Li. Hrtt: A hierarchical roof topology structure for robust building roof reconstruction from point clouds. *Remote Sensing*, 9(4):354, 2017.

[41] B. Xu, W. Jiang, J. Shan, J. Zhang, and L. Li. Investigation on the weighted ransac approaches for building roof plane segmentation from lidar point clouds. *Remote Sensing*, 8(1), 2016.

[42] E. You. Vue.js : The progressive javascript framework. `https://vuejs.org/`. Online; accessed 4 Mar. 2019.

[43] L. Zhang, Z. Li, A. Li, and F. Liu. Large-scale urban point cloud labeling and reconstruction. *ISPRS Journal of Photogrammetry and Remote Sensing*, 138:86–100, 2018.

[44] L. Zhang and L. Zhang. Deep learning-based classification and reconstruction of residential scenes from large-scale point clouds. *IEEE Transactions on Geoscience and Remote Sensing*, 56(4):1887–1897, 2018.

[45] W. Zhang, J. Qi, P. Wan, H. Wang, D. Xie, X. Wang, and G. Yan. An easy-to-use airborne lidar data filtering method based on cloth simulation. *Remote Sensing*, 8(6):501, 2016.

[46] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia. Pyramid scene parsing network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2881–2890, 2017.