Building Modeling Through Enclosure Reasoning

Adam Stambler and Daniel Huber
Carnegie Mellon University
5000 Forbes Avenue, Pittsburgh PA
adasta@gmail.com, dhuber@cmu.edu

Abstract

This paper introduces a method for automatically transforming a point cloud from a laser scanner into a volumetric 3D building model based on the new concept of enclosure reasoning. Rather than simply classifying and modeling building surfaces independently or with pairwise contextual relationships, this work introduces room, floor and building level reasoning. Enclosure reasoning premises that rooms are cycles of walls enclosing free interior space. These cycles should be of minimum description length (MDL) and obey the statistical priors expected for rooms. Floors and buildings then contain the best coverage of the mostly likely rooms. This allows the pipeline to generate higher fidelity models by performing modeling and recognition jointly over the entire building at once.

The complete pipeline takes raw, registered laser scan surveys of a single building. It extracts the most likely smooth architectural surfaces, locates the building, and generates wall hypotheses. The algorithm then optimizes the model by growing, merging, and pruning these hypotheses to generate the most likely rooms, floors, and building in the presence of significant clutter.

1. Introduction

Modeling existing buildings from point clouds has become an essential part of civil engineering. Building Information Models (BIMs) are used for everything from renovation planning and facility management to the creation of a facility's as-is blueprints [6, 7]. However, there is no complete, end-to-end automated process for generating the semantic, volumetric design models that are needed for those tasks. Instead, industry uses manual labour. It can take hundreds of hours from scanning a building with a terrestrial laser scanner (TLS), to registering the scans, to manually completing the BIM using the point clouds a reference.

This work seeks to automate this error-prone and laborious process. The first stage of the process, registration, has been largely solved in the literature and implemented in industry. Large scale point cloud reconstruction from web images [3], quad-rotors [5], or mobile mapping systems (like [22] have generated detailed, registered point cloud models of buildings.

This work takes a registered point cloud and automatically generates the building model with only the assumption that there is one complete building, both interior and exterior, present. The algorithm then segments surfaces, classifies them, and optimizes them into a semantic volumetric building model using occlusion and enclosure reasoning.

There has been prior work on modeling the exteriors and interiors of buildings, but much of it has taken the traditional 3D reconstruction route. Prior algorithms model and classify independently. There is an architectural surface detection, step and then all detections are used in the final BREP model regardless of if they make sense during modeling. This means that the final model is limited by to the locally visible surfaces of the building and cannot improve reconstructions by changing surface geometry or inferring existence of occluded surfaces. For building exteriors, this work falls under facade reconstruction. For example, [11][12] recognize exterior walls as planes and detect windows using hand-tuned rules. Their work then groups together the building surface polygons into a building volume. However, the building models are overly-simplified and the buildings are already segmented from their environment.

For interior modeling, there have been similar approaches. [1, 14, 16] use hard-coded rules on planar clustered point segments to find and classify building surfaces. They then use the polygon contour of these segments as their 3D model. Another approach to automatically generating indoor models has been [10], which uses a 2D histogram of the points on the floor plane to recognize walls. On each floor, all the points are projected on to the ground plane, and the most dense histogram cells are walls. All of these approaches use only hand-tuned features which are brittle to real world data.

More robust approaches, including ours, use machine learning to learn the appearance of architectural surfaces

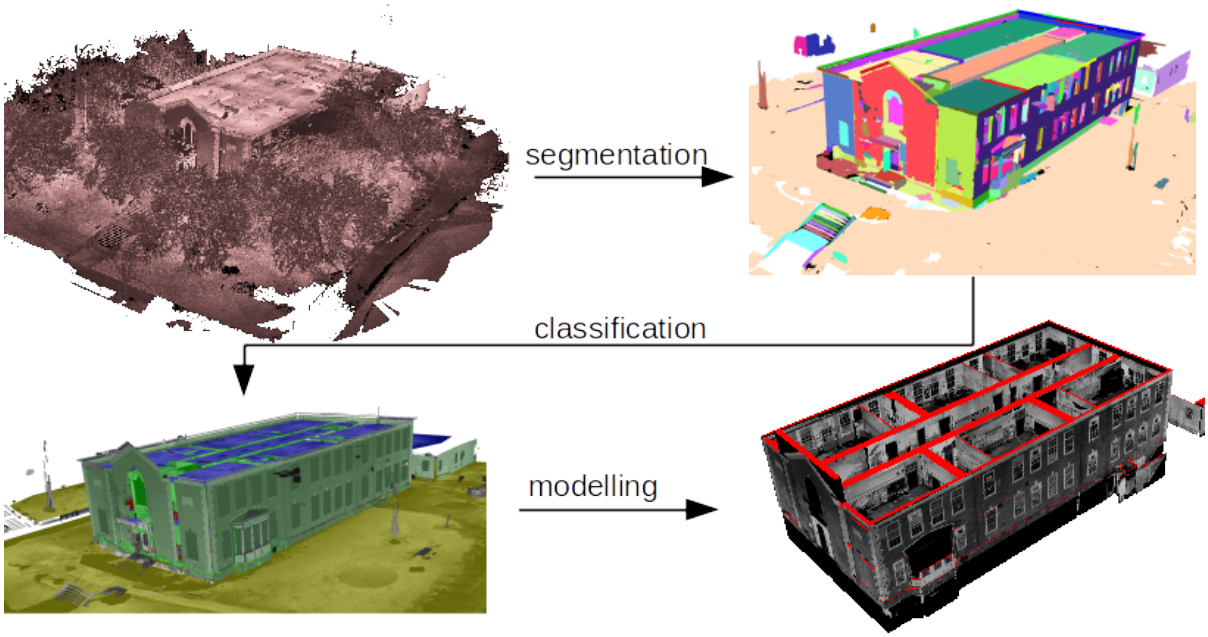

Figure 1. Visualization of building modeling pipeline for the school house dataset. The top left is the registered point cloud for the model. The top right is the point cloud segmented into dominant surfaces. All small segments have been removed for clarity. At the bottom left is a surface model with every surface coloured according to its class. Finally, the bottom right shows the final, volumetric building model textured with the point cloud intensity data.

and clutter from data. Xiong *et al*. have used this approach to successfully perform semantic labeling of building surfaces. In [20], a pairwise conditional random field with coplanarity constraints imposes contextual information while [21] introduces stacked learning for recognition. These authors propagate the structural information of the building during classification by imposing relatively local contextual reasoning. Surfaces must be pairwise coplanar or orthogonal. These constraints are analogous to ours, but not as powerful.

Our proposed enclosure constraint allows our pipeline to intertwine the recognition of the building with the modeling (Figure 1). Each surface is scored with a simple classifier. The surfaces then provide the seeds for the growth of parametric models of walls, doors, ceilings, and floors. This growth is restricted by knowledge of free space cleared by the scanners laser and by trained room shape priors. Our approach uses this constraint to directly extract the BIM primitives and regularize the dimensions of those primitives. To our knowledge, this is the first work to present a pipeline for volumetric building reconstruction using enclosure reasoning. The rest of the paper explains the scan preparation, wall hypothesis generation, and subsequent building optimization procedure.

2. Scan Preprocessing

The input to the pipeline is a registered set of laser scans covering both the interior and exterior of a building. These scans can be registered automatically using methods like [15] or [8] or by manual industry software. Our method assumes that input scans are in the original, organized point cloud directly from the scanner. We use these raw point clouds to reason about occlusions and volumetric occupancy within the scanning area. Additionally, the gestalt of each scan is used to estimate whether each surface is on outside or inside of the building.

With a merged point cloud, the only information in the data is the presence of a surface at a particular point. However, an organized point cloud from a particular scanner can be considered a ray originating from the scanner origin. Everything in the path of the laser from the scanner to the measured point is free space while everything after is unknown. All architectural elements will be ”unknown” at their core and this feature is used later in the pipeline. All scans are merged into a ray traced *volumetric knowledge grid (VKG)* where each voxel has been labelled as free, occupied, or unknown. This idea is in the same vein as evidence grids or occupancy maps in robotics [9], but in this representation we are explicitly interested in the unknown voxels rather than just the occupied voxels.

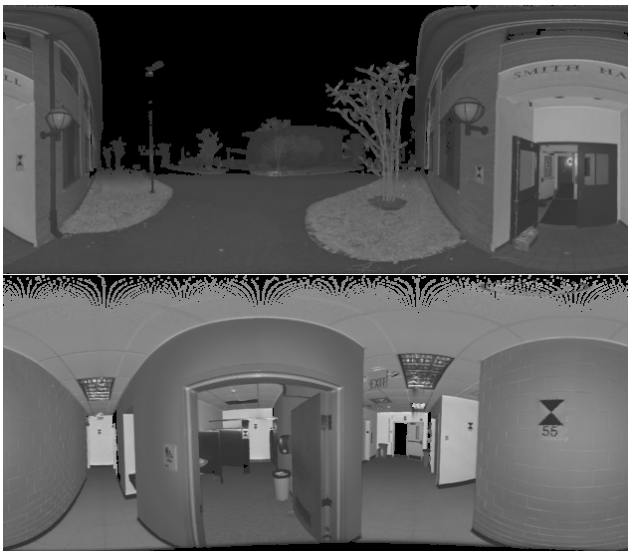

Figure 2. Two example spherical projections of laser scans from building A, coloured by laser intensity. The pipeline uses a histogram of the range image pixels classify if a scan was taken outside or inside.

Next, the algorithm uses a per-scan classifier to determine whether each scan was taken from the interior or the exterior of a building. This is used later to distinguish between surfaces on the inside and outside of the building. Surfaces observed from outside scanners are on the building exterior while surfaces on the interior are observed by inside scanners. For features we use a 2D histogram of azimuth angle by point range. In our implementation, the histogram is 2x9 bins with values ranging ranging from $[-\frac{\pi}{2}, \frac{\pi}{2}]$ radians and $[0, 15]$ meters. This range histogram is classified using a linear SVM.

The individual scans are then filtered for mixed pixels and downsampled to a 4 cm resolution. The scans are downsampled to focus later stage algorithms on the spatial frequency of interest: architectural structures whose size is greater than 8 cm (two times the sampling frequency). These are typically on the order of 1 m or greater.

3. Segmentation

After the scans are processed, they are compiled into a single point cloud and segmented into potential architectural surfaces. While most building surfaces are planar, our algorithm does not restrict itself to planar segments, only smooth, dominant surfaces. The segmentation algorithm is an iterative normal clustering algorithm based on the smoothness constraint of [13].

At each iteration, the algorithm orders all of the points according to their curvature in ascending order. It iterates through this list and uses the first unassigned point as both seed for growing a segment and as the segment's first root. This seed is also added to a segment seed stack. For each seed s_i on the stack, all points in within a radius R are compared against the current segment root point. If it is co-planar enough with the root, it is added to the segment. Something is coplanar if the angle between the surface normals is less than θ_n and the point-to-plane distance between the two points is less than ϵ_p. Additionally, if this neighbouring point has curvature less than ϵ_c, it is added to the seed stack. The segment is grown until there are no seeds on the seed stack and then a new segment is started with the next unused point from the curvature list. This continues until all unassigned points have a curvature greater than ϵ_c. Once the iteration completes, the normals of all the points in each segment are averaged and assigned to each segment point.

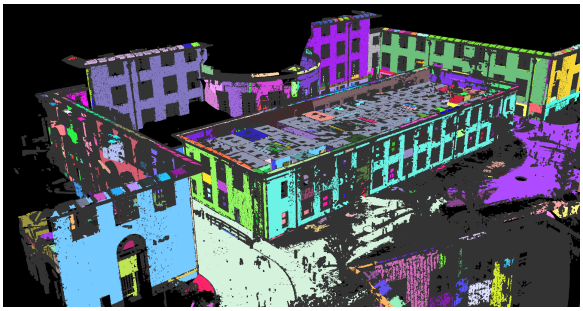

Figure 3. Segmentation Example: Exterior of building A segmented into architectural surfaces. Each color is a separate segment. Grey points do not belong to any segment. Each segment will become a architectural primitive hypothesis fed into the optimization algorithm.

This process repeats for n-1 iterations. However, at each iteration the neighbourhood radius R is increased while θ_n is decreased. Then, at the last iteration, the algorithm allows for smooth surfaces by changing the segment root point to the current seed point. In essence, this algorithm does purely planar clustering during the first n-1 stages. This clustering acts as a robust way of estimating the surface normals so that the last smooth surface clustering does not miss any sharp surface junctions.

4. Per Point Classification

Next, the pipeline performs per point classification on each scan individually. These per point classifications are later used to vote for the class of the extracted surfaces from the full building. The classification is accomplished by using a set of hand-selected features including the output of the inside/outside classifier, the local shape, the scanner viewpoint, and local segment shape.

Local shape analysis is performed by using eigenvalue analysis of the covariance of the points around each point.

Given a point X_i, its covariance is calculated from the points within a neighbourhood N_i around X_i. If $\bar{X}$ is the mean location of that neighbourhood, then $C_i = \sum_{\forall j \in N_i}(X_j - \bar{X})(X_j - \bar{X})^T$ is the covariance. The eigenvector of the smallest eigenvalue of C_i is the local surface normal. This surface normal's sign is corrected to point back at the scanner and its Z component is used as a feature. The ordered eigenvalues of C_i, $\lambda_0 < \lambda_1 < \lambda_2$, are then used to describe the local shape. A proxy for curvature is defined as $\frac{\lambda_0}{\lambda_0+\lambda_1+\lambda_2}$. The algorithm also uses spectral features given by the difference of C_I's eigenvalues (similar to [18]). $\lambda_2 - \lambda_1$ is a measure of linearity, $\lambda_1 - \lambda_0$ is a measure of planarity, and λ_0 is a measure of sphericalness of the local neighborhood. This analysis is performed at multiple scales (0.1 m and 0.3 m in our implementation).

Next, the points are described by their location relative to the scanner or all the other points in the point cloud. The features are the difference in Z height relative to the scanner's position and the height of the point relative to a robust estimate of the minimum and maximum height of the points in the scan. Additionally, we look at the distribution of the points in the scan's point cloud along current point's normal. This results in two features. The fraction of points behind the current point along the normal and the distance of the current point to the farthest point behind it. If the point is clutter, there will likely be a structural surface behind the normal support it.

Finally, the algorithm segments each scan according to the segmentation algorithm described in Section 3 and uses the shape of the segment as a set of features. The total number of points in the segment are used as a proxy for its area. The height of the segment and curvature are used as a description of its shape.

These features, a total of 19, are used as inputs into a random forest classifier which produces probabilities of each point belonging to a surface class. In this work, we consider interior walls, exterior walls, ceilings, floors, and clutter. Clutter is used as a catch-all class to which points belong if they cannot be modeled (like stairs) or are not structural components of the building (like trees, furniture, artwork).

These per point classifications from each scan are then used to vote for class of each segment in the building point cloud. The classification scores for each point within each segment are averaged . This score becomes the segments classification score.

5. Building Localization

Next, the probabilistic classifications are used to orient, locate, and find the levels of the building within the point cloud. These steps are important for both the presentation of the final building model and for defining tight boundaries on which subsequent optimization is performed.

The orientation of the building is found using a Hough transform algorithm. Each interior wall surface normal votes for the transform that aligns itself to an X or Y axis. Each surface's vote is proportional to the classifier's confidence that the surface is a wall. The transform with the maximum vote is chosen. For all subsequent operations, this transformation is applied to the site.

The initial building location hypothesis is found by computing a robust, axis-aligned bounding box around the points that could belong to the building. Each building point is weighted by its classification score and all of the points are sorted along each axis. The top and bottom one percent of the points are trimmed away, and the range for each axis is extracted from the rest of the points. For additional safety, this bounding box is padded (by two meters in our implementation) to make sure that the whole building is included.

Finally, the levels of the building are found by using an architectural prior of a level typically being about four meters tall. Floor seeds are placed 4 meters apart along the building's z-axis. These seeds are refined by performing iterative weighted regression to the floors and ceilings of the building. This allows the level estimate to snap to the actual distance between the floors and ceilings of the particular building.

6. Model Creation

Now that the potential surfaces and location of the building have been identified, the building model hypothesis is created and optimized. This proceeds in three stages. First, the wall hypotheses are created for each detected surface. Second, these hypotheses are grown according to measured free and unknown space recorded in the site's volumetric knowledge grid. Third, the openings along each wall are detected. Finally, the building is optimized using the enclosure reasoning constraint.

6.1. Wall Hypothesis

An overcomplete set of wall hypotheses is generated from the segmented surfaces. These surfaces are selected using each surface's classification score. A wall hypothesis is seeded by every detected surface with a wall probability above a threshold λ_w. The lower λ_w the more overcomplete the set of wall hypotheses will be, which allows for lower-performing surface classifiers to be used. The hypothesis is seeded as a wall with zero thickness that extends from floor to ceiling on its building level.

Next, each wall is greedily grown into a full volumetric wall by maximizing its wall score $S(W)$. The wall score consists of two parts: $S(W) = S_v(W) + S_a(W)$. A wall is judged by how much its volume is wall-like (first term) and how much of its surface area has been classified as being a wall (second term).

The intuition of the volume score is that walls cannot exist in space that was cleared by laser beams. However,

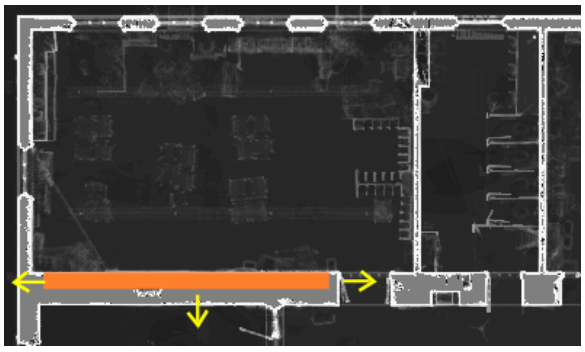

Figure 4. Wall Growth: This example illustrates how the algorithm determines wall extents when there are no supporting points. It shows a part of the occupancy map of a floor in the school house building. Grey indicates an unknown region, white indicates occupied space, and black is known free. A detected wall, indicated in orange, will grow thicker and longer to explain the unknown volume in the VKG. It will not grow into the free space of the room. Arrows denote the direction of growth.

if the space was occluded or occupied by wall points, then the wall primitive should be extended to fill that space. The volumetric composition of a wall is determined by querying the VKG of the building. However, if that void is a detected architectural opening like a window or door then the free volume penalty should be removed. If $V(W, X)$ computes the volume of the wall that is of type X = ***U**nknown, **F**ree, **O**ccupied*, and ***D**etected opening* then this is captured in the scoring function:

$$S_v(W) = \frac{V(W,U) + V(W,\mathrm{O}) - V(W,\mathrm{F}) + V(W,\mathrm{D})}{V(W)}$$

The intuition for the surface area score $S_a(W)$ is that better walls will have more observable wall surfaces. If $A(W)$ computes the surface area of the wall and $p_w(a)$ is the probability that element a is a wall, then the score is:

$$S_a(W) = \frac{\int_W p_w(a)\Delta a}{A(W)}$$

The $p_w(a)$ is given by the probability outputted by the segment class scores.

This approach is in contrast to prior work, which connects wall surfaces with no regard to free space constraints [16]. Their approach assumes that wall surfaces that are close enough should be merged or joined together at perpendicular edges.

Once the walls have been grown according to the VKG, the pose and size of the wall are refined based on the supporting point cloud to improve dimensional accuracy. The VKG is computed at a lower resolution than the original point cloud due to memory constraints, and the initially

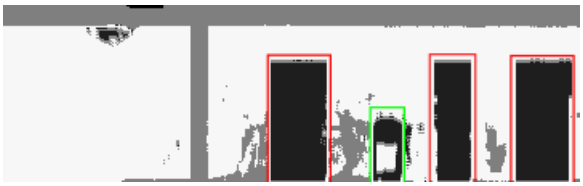

Figure 5. Opening detection. Openings are detected using 2D image recognition techniques applied to the occupancy map of wall surfaces. In the occupancy map, white = occupied, black = free space, and grey = occluded/unknown. The red boxes indicate detected doorways. The green box denotes a detractor opening belonging to a water fountain cubbyhole.

estimated dimensions could have as much this resolution in quantization error. The refinement step considers all of the points in contact with the wall and performs iterative weighted least squares optimization. Each point is weighted according to its distance to the current wall estimate and its normal's angle to the wall's corresponding surface point. With linear walls, this means that we perform a regression for the supporting set of parallel and perpendicular wall surfaces.

6.2. Opening Detection

Without explicit opening detection and modeling, the VKG-driven wall growth would have difficulty extending walls past door openings or large windows, and it would penalize walls with interior windows. By default, the algorithm does not overcommit to the existence of a wall without supporting points or occlusion, which is normally an advantage unless the missing points are due to a door or window embedded in the wall. We therefore, explicitly detect and model the openings and compensate for the penalty in wall score that would be incurred. The grown walls surfaces are used as search surfaces for finding salient architectural openings like windows or door-sized rectangular openings. First, adjacent coplanar wall surfaces within one meter of each other are merged into a single surface in order to find openings between two walls. These surfaces are then intersected with the VKG to create a 2D occupancy map along the wall. This map shows if the wall surface was detected by the laser, was free space, or was occluded by another object (Figure 6.2). A Hough transform is used to search over the occupancy map for rectangular openings. Non-maximum suppression is used to select the largest detections with over 75% of opening's boundary known as free. This opening is then added as an architectural opening, and the wall is kept as joined.

6.3. Optimization

At this point in the pipeline, the algorithm assumes that all architectural primitives (walls, openings, levels) are

identified and that the set is overcomplete. There may be misclassified primitives, and some walls may have multiple hypotheses because they have been observed from multiple sides. The pipeline's job is now to regularize the dimensions of the primitives while pruning away the redundant or incorrect ones. This optimization is formulated as finding the best room covering for all floors of the building with a penalty for using more rooms or walls than necessary. Thus, the building score is a weighted sum of all room scores plus a minimum description length (MDL) term. The objective function is:

$$\max_{B} S(B) = \sum_{\forall R \in B} (A(R)S(R)) + \lambda_0 \exp(-\frac{||W||}{||R||})$$

where $||W||$ is the number of walls in the building and $||R||$ is the number of rooms. The "best" rooms are cycles of wall primitives that enclose the most likely rooms and whose walls have the highest wall score. If there are N walls and K openings within a room, then the room score $S(R)$ is:

$$S(R) = \lambda_1 P(R) + \lambda_2 e^{-|N+K-5|} + \frac{\lambda_3}{N} \sum_{\forall W \in R} S(W)$$

The first term, $P(R)$, represents the probability of the room's shape. This probability is modelled as a set of exponential distributions learned from over 38,000 room floor plans from the Understanding Indoor Environments Dataset [4, 19]. $P(R)$ is the product of the independent probability of the room's area, perimeter to area ratio, aspect ratio, and convexity. Convexity is defined as $\frac{A(B_R)-A(R)}{A(R)}$, where $A(B_R)$ is the 2D oriented bounding box of the room, while $A(R)$ is the area of the room's polygon. The second term enforces a minimum description length for the number of walls and openings used to explain the room. The room will score less if has more than one door and four walls. The last term is simply the average score of the walls composing the room. This term encourages room hypotheses consisting of volumetric primitives that look more "wall-like."

The pipeline optimizes the building using simulated annealing over the set of rooms and their walls. The initial rooms for the simulated annealing are found by finding the free space connected components of each level (Figure 7). Each room's initial wall set is the smallest set of walls that enclose the room. Next, the optimization proceeds with each step having the following operations: walls can be joined, rooms can swap walls in or out, and rooms can be merged. Only the wall primitives that are a part of rooms are kept in the final model.

category / dataset	A	B	C
Scan interior	0.91	0.80	0.98
Wall	0.62/0.82	0.74 / 0.73	0.91/0.46
Ceiling	0.83/0.94	0.91 / 0.87	0.98/0.66
Floor	0.94/0.96	0.41 / 0.82	0.97/0.58
Ground	0.96 /0.93	0.9 / 0.06	–
Exterior Wall	0.87/0.37	0.83 / 0.6	–
Clutter	0.71/0.79	0.67 / 68	0.6/0.81

Table 1. Classification performance. The first row contains scan level interior/exterior classification accuracy. The remaining rows list the precision and recall of the per-point classification of each building point. Raw exterior scans for building C were unavailable, so classification results are omitted.

7. Discussion

This paper proposes a complete building modelling pipeline built upon enclosure reasoning. This algorithm's output can be used as a simplified view of a building or as the basis for further refined manual or automatic modeling in a BIM pipeline. The quantitative results are listed in Table 1 while qualitative results are shown in Figure 1 and Figure 7. We have conducted experiments using data from the Scan-To-BIM repository, which was designed for testing building modelling algorithms (http://scantobim.ri.cmu.edu/). In total, the repository consists of over 570 annotated survey scans. We utilize three full buildings from the repository: A) a two story office building, B) a two story school house, and C) three story old college building. Additionally, we utilize partial building data sets consisting of single floors of buildings or exterior scans taken from the street for training the scan exterior/interior recognition and the point cloud segment recognition. Initial test results are reported for all three full buildings. The classification experiments were performed by leaving that building out of the training set and using the rest of the dataset as training data.

The classification results are shown in Table 1. Although the surface classification results are imperfect, the algorithm is still able to cope with poor classification results because of the regularization provided by the modeling. The algorithm is still successful at detecting the type of surface because the other shape and pose features compensate. Additionally, the optimization step takes this redundant set of surface primitives and generates less than half the number of volumetric primitives for the final model. However, these models still need to be evaluated for dimensional accuracy.

Additionally, the preliminary results show some failure modes for the algorithm. One big failure mode is when the segmentation fails to identify a wall segment. This occurs when the walls are not smooth, are primarily composed of windows, or they are totally obscured. The only way for

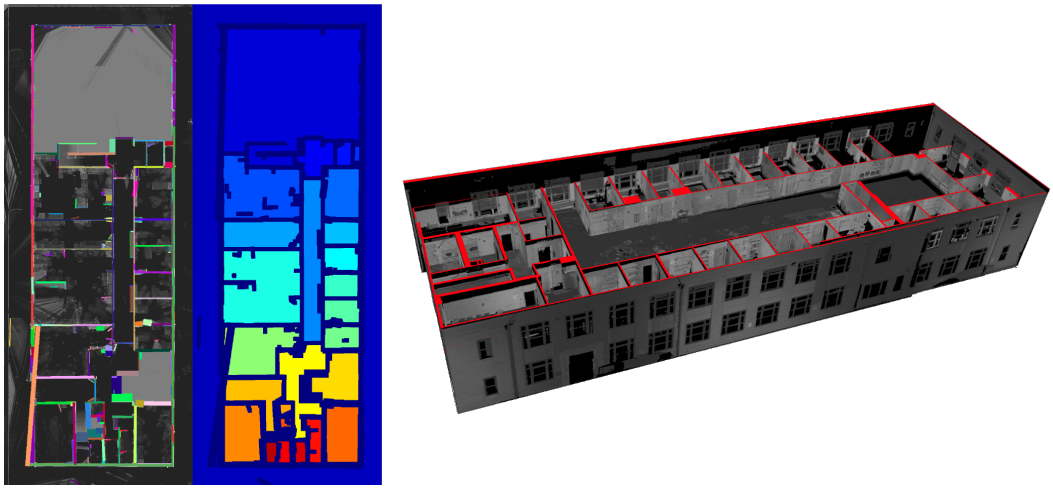

Figure 6. To find the room hypotheses for a floor, the wall hypotheses are independently and greedily grown. (Left) Each wall hypothesis, is colored a randomly on the occupancy map. Then the room hypotheses (center) are created by finding connected components of free space on a 2D floor plan. These components are shown along with the morphologically expanded wall hypothesis separating potential rooms. In this example, dark blue indicates building walls. All other colors represent the initial rooms found. Their adjacent wall primitives are used to score the rooms. (Right) These hypotheses are optimized into the final volumetric model.

the modelling to recover is for the wall to be recognized and grown from the opposing side. Another failure mode is closed door detection. The algorithm searches for doorway openings, not doors. If the door was closed while scanning, the opening detector does not find it. Future algorithms can overcome this deficiency by using more sophisticated window/door detectors rather than salient opening detectors. Techniques like sliding window search for doors on texture mapped intensity and deviation images of walls are potential approaches.

8. Future Work

The presented pipeline suggests many avenues for future research. It creates a model of all dominant architectural components and seeks to enforce a sensible reconstruction. However, this notion of sensibility is severely constrained by the amount of publicly available building data. It is limited to 2D room shape priors learned from floor plans. Ideally, the likelihood of full building reconstruction should be evaluated through 3D statistical models of building construction. Future research needs more laser surveys and needs to investigate the use of other modalities, like imagery, to generate improved statistical models. For example, this might include evaluating the likelihood of a wall or window given its room's detected function. Another important avenue of improvement is in the recognition and exact identification of building components like moulding, furnaces, or furniture. Adan et al. [2] automatically recognize windows using a trained classifier, but it only works with fully open, rectangular windows and does not say anything about how the window is constructed. Human modellers identify the exact manufacturer and model number of the component. [17], for example, does preliminary work with building moulding identification, but it remains an open problem to perform the large-scale 3D matching required to match each component to hundreds of thousands of possible part numbers.

9. Acknowledgements

This material is based upon work supported by the National Science Foundation (NSF) under Grant No. 0856558 and by the Pennsylvania Infrastructure Technology Alliance. We thank Quantapoint, Inc., University of Arkansas, Autodesk, and Carnegie Mellon for providing experimental data. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the NSF.

References

[1] J. B. A. Budroni. Toward automatic reconstruction of interiors from laser data. In *Proceedings of 3D-ARCH*, Tento,Italy, 2009.

[2] A. Adan and D. Huber. 3D Reconstruction of Interior Wall Surfaces Under Occlusion and Clutter. *3D Imaging, Modeling, Processing,*, 2011.

[3] S. Agarwal, N. Snavely, I. Simon, S. M. Seitz, and R. Szeliski. Building Rome in a day. *2009 IEEE 12th International Conference on Computer Vision*, pages 72–79, Sept. 2009.

[4] P. J. Alper Aydemir and J. Folkesson. What can we learn from 38,000 rooms? reasoning about unexplored space in in-

door environments. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2012.

[5] H. Bischof. Dense reconstruction on-the-fly. *CVPR*, 2012.

[6] GSA. Gsa bim guide for 3d imaging, Jan. 2009.

[7] GSA. Gsa bim guide for energy performance, Jan. 2009.

[8] Z. Kang, J. Li, L. Zhang, Q. Zhao, and S. Zlatanova. Automatic Registration of Terrestrial Laser Scanning Point Clouds using Panoramic Reflectance Images. *Sensors (Basel, Switzerland)*, 9(4):2621–46, Jan. 2009.

[9] H. P. Moravec. Robot spatial perception by stereoscopic vision and 3d evidence grids. Technical report, Carnegie Mellon University, 1996.

[10] B. Okorn, X. Xiong, B. Akinci, and D. Huber. Toward automated modeling of floor plans. In *Proceedings of the Symposium on 3D Data Processing, Visualization and Transmission*, volume 2, 2010.

[11] S. Pu. Automatic building modeling from terrestrial laser scanning. *Advances In 3d Geoinformation Systems*, 2008.

[12] S. Pu. *Knowledge based building facade reconstruction from laser point clouds and images*. NCG,, Delft, the Netherlands, 2010.

[13] T. Rabbani, F. van Den Heuvel, and G. Vosselmann. Segmentation of point clouds using smoothness constraint. *International archives of photogrammetry, remote sensing and spatial information sciences*, 36(5):248–253, 2006.

[14] V. Sanchez and A. Zakhor. Planar 3D Modeling OF Building Interiors From Point Cloud Data. In *IEEE International Conference on Image Processing*, 2012.

[15] P. W. Theiler and K. Schindler. Automatic registration of terrestrial laser scanner point clouds using natural planar surfaces. *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, I-3:173–178, 2012.

[16] E. Valero, A. Adán, and C. Cerrada. Automatic method for building indoor boundary models from dense point clouds collected by laser scanners. *Sensors (Basel, Switzerland)*, 12(12):16099–115, Jan. 2012.

[17] E. Valero, A. Adan, D. Huber, and C. Cerrada. Detection, modeling, and classification of moldings for automated reverse engineering of buildings from 3D data. *ISARC*, pages 3–8, 2011.

[18] N. Vandapel, D. Munoz, and M. Hebert. Onboard contextual classification of 3-D point clouds with learned high-order Markov Random Fields. *2009 IEEE International Conference on Robotics and Automation*, pages 2009–2016, May 2009.

[19] E. Whiting, J. Battat, and S. Teller. Generating a topological model of multi-building environments from floorplans. In *CAAD (Computer-Aided Architectural Design) Futures 2007*, pages 115–28, Sydney,Australia, July 2007.

[20] X. Xiong. Using context to create semantic 3D models of indoor environments. *Proc. BMVC*, pages 1–11, 2010.

[21] X. Xiong, A. Adan, B. Akinci, and D. Huber. Automatic creation of semantically rich 3D building models from laser scanner data. *Automation in Construction*, 31:325–337, May 2013.

[22] J. Zhang and S. Singh. Loam: Lidar odometry and mapping in real-time. July 2014.