Generating Architectural Drawings from Photographs

Eric Chan, Holly Rushmeier

December 2018

Abstract

Current methods of creating accurate architectural drawings are costly or labor intensive. This project examines the application of Structure from Motion techniques to generating architectural plans, which has the potential to yield faster and more accurate drawings than current methods. In this report, we consider capture side optimizations in the Structure from Motion process, such as capture method, detail-increasing markers, state-of-the-art feature extraction methods and feature matching algorithms. We discuss techniques for extracting 2D architectural drawings from reconstructed point clouds, including filtering, object removal, wall-fitting, and interpolation. Finally, we construct and evaluate a pipeline utilizing the examined methods that is capable of automatically generating rough floor plans from imagery.

1 Introduction and Background

1.1 Architectural Drawings

Architectural record drawings, the most basic of which are the Plan, Section, and Elevation, have been used for centuries to communicate the structure and form of buildings. Historically, architects have been influenced by the great architecture of the past, often making record drawings in order to emulate forms and structures in their own work. More recently, record drawings have been used to preserve architectural heritage—the 1894 Survey of London and the 1933 Historic American Buildings Survey are two well known examples. Along with giving an accurate depiction of the building as it stands, these drawings give the fine measurements necessary for restoration and reconstruction.

Before the advent of modern imaging techniques, teams of surveyors might spend weeks measuring a single building by hand. In the last few years, laser scanners and other technology has been combined with hand measurements to drastically reduce the time required to generate drawings. However, the data must still be processed and painstakingly assembled by hand, and it must still be captured using specialized equipment operated by experts.
Photogrammetry, a method of taking measurements from photographs offers a potential solution. Using a technique called Structure from Motion, we have the potential to image a building and generate the corresponding architectural drawings, more accurately and quickly than ever before.

1.2 Structure from Motion

Structure from Motion (SfM) is a technique for reconstructing three-dimensional objects from a series of two-dimensional images. Similar to how stereo vision in animals works, structure from motion relies on viewing objects from different angles in order to estimate 3D locations for the cameras and detected features. Structure from Motion is generally broken down into four steps: Feature Extraction, Feature Matching, Camera Motion Estimation, and 3D Reconstruction using the motion estimation and features [4]. In sum, Structure From Motion allows us to generate a 3D point cloud corresponding to a structure, which can then be further processed into a 2D architectural plan.

1.3 Software

1.3.1 COLMAP

There are many different SfM pipelines, each of which utilizes different algorithms and techniques and have different advantages and disadvantages. COLMAP is an open-source, general use incremental SfM pipeline. According to the evaluations conducted by Bianco et al. [1], COLMAP performs well on a wide variety of scenarios and has the added benefit of being faster than most other state-of-the-art SfM pipelines. The open-source nature of the software combined with its excellent performance made it the SfM pipeline of choice for this project.

1.3.2 Point Cloud Library

We utilized the Point Cloud Library for manipulating and processing point clouds. PCL provides a framework for efficiently storing large collections of points, along with several common algorithms for filtering, feature estimation, surface reconstruction, registration, model fitting and segmentation [5].

2 Description of Primary Datasets

To test and evaluate different methods of generating architectural drawings, we captured image collections from existing buildings. In order to avoid overfitting to a particular type of structure or situation, we utilized four different datasets of varying scale, complexity, and detail.

2.0.1 Alameda House

The simplest dataset, the Alameda House, is a small, rectangular room with no interior walls or large obstructions. This building is currently under construc-
tion, and exposed woodwork, insulation, and tools result in detailed surfaces that allow for easy feature extraction and matching. Large windows on one side of the building are responsible for lighting the space. Although the building has no traditional furniture, tools, building materials, and other objects interrupt the clean surfaces of the building’s structural walls and a tall ladder stands in the middle of the room. Additionally, several people were present in the building during the capture, and their movement through the scene contributes to noise in the generated point cloud. Direct line of sight to every corner of the room allowed for the formation of complete loops. This dataset was generated using video frames taken with an iPhone X.

2.1 Saybrook Basement

A slightly more complex structure, the Saybrook Basement is a small, narrowing corridor with several wall depressions and large objects, such as vending machines. The section of basement imaged features large, flat surfaces with relatively few details along the walls and floors. The corridor is primarily artificially lit, and intense glare off of the semi-reflective floor and walls makes feature extraction and matching difficult. The scene contains a wide range of detail levels—the vending machine and stone walls contain many areas of sharp contrast, allowing for easy feature detection. However, the painted walls and floor offer few distinguishing features along their surfaces, preventing easy point generation along these surfaces. Because the corridor was long and continuous, the image collection did not form complete loops. This dataset was generated
using images taken with an iPhone X.

2.2 Saybrook Dining Hall

A larger scale structure, the Saybrook Dining Hall is a large room featuring several depressions, columns, and corners that obscure the line of sight of the camera. The dining hall features large flat surfaces, but wooden trim and detailing allows the extraction and matching of features. The dining hall is artificially lit, resulting in glare off of some glass surfaces, and the interior of the dining hall featured many tables and chairs, complicating the model. Because of the obstructions, and because we imaged only half of the dining hall, the images did not form complete loops. This dataset was generated from video frames taken with an iphone X.

2.3 Yale University Art Gallery (YUAG)

A large scale, complex indoor space, the Yale University Art Gallery’s modern art gallery forms a maze-like space divided into several individual rooms by interior walls. The gallery features smooth, uniformly lit walls with very few details; however, the paintings along the walls served as high detail areas and allowed for feature extraction and matching. In addition to the paintings, which protruded from the gallery walls, the space featured several statues and pedestals, as well as moderate pedestrian traffic. This dataset was generated from an image collection taken with a Canon 5D mkII and 15 mm wide-angle lens.

3 Discussion of Capture Side Optimizations

3.1 Photo vs. Video Capture

The point clouds that were later processed into 2D architectural drawings were generated using a structure from motion reconstruction of images. We tested two different methods of capturing these images: as individual photos and as continuous video sequences. From a quality standpoint, individual photos were significantly sharper and captured more detail. Because photos are nearly always taken with a stationary camera, the photos did not suffer from motion blur and were consistently sharp. In contrast, video frames, which were captured with a sweeping motion, tended to suffer from significant motion blur, obscuring fine details. Artifacts from video compression and limited video resolution further reduced the quality of the stills. When using exhaustive feature matching, individual photos tended to yield higher quality models, likely due to the higher number of detected and matched features.

However, video capture has a number of advantages over individual photo capture. We extracted video frames at a rate of 2 frames per second. The capture of videos in continuous sweeping motions at a high frame rate resulted in not only a much quicker capture of a building, but also a larger collection.
of images with which to perform the matching. For amateurs, video capture is significantly easier than photo capture, resulting in more detailed and complete models. Additionally, continuous video sequences allow for sequential feature matching, which dramatically speeds the reconstruction and in fact tends to generate more complete models. Sequential feature matching is discussed in more detail in section 3.4.3.

Our conclusion is that video capture, which is easier, faster, and potentially gives more complete models, outweighs the quality advantages of individual photo capture.

3.2 Detail-Increasing Markers

While some buildings, with walls covered in fine details, lend themselves to Structure from Motion reconstruction, others are constructed with blank walls with few distinguishing features. Modern-styled buildings, in particular, tend to utilize large, monochromatic surfaces that make feature extraction and matching difficult. Buildings with blank surfaces tend to generate sparse point clouds at best. At worst, large portions of the image collection may remain unmatched, resulting in a fragmented and incomplete reconstruction.

We experimented with placing distinctive, detail-increasing markers on low-detail surfaces. The addition of a few markers on a surface increases the number of extracted features, in some cases by a factor of four, resulting in greatly enhanced feature matching and reconstruction.

3.3 Feature Extraction

By default, COLMAP extracts Scale Invariant Feature Transform (SIFT) features. SIFT features are a class of local image features that are invariant to scaling, translation, and rotation and are robust under changing illumination and affine distortions or 3D projections. SIFT features are useful for object recognition when features may be partially obscured or otherwise affected by nearby clutter. In our case, these properties are especially useful when matching features in images taken from different angles, as they retain discriminative power even under varying vantage point [3].

An enhancement to standard SIFT features is Domain-Size Pooling. DSP-SIFT uses gradient orientation pooling across domain scales in the effort to make SIFT features more robust. DSP-SIFT is in theory more resistant than standard SIFT to blur, varying lighting, and transformations such as perspective projections and rotations.

Table 1 shows that while DSP-SIFT typically resolves far more features than standard SIFT, resulting in more densely populated models, the processing time required rises dramatically. On the Alameda House dataset, DSP-SIFT generated nearly twice as many points, but required nearly five times as much processing time. Because the algorithms in section 4 do not require the data density advantages of DSP-SIFT features, we chose to use standard SIFT feature extraction for most tests.
3.4 Feature Matching Method

We compared the three commonly used methods of feature matching: Exhaustive, Vocabulary Tree, and Sequential Feature Matching.

3.4.1 Exhaustive Feature Matching

The simplest method of feature matching, the exhaustive feature matching method assumes no domain knowledge and compares every feature against every feature in other images. Exhaustive feature matching generally produces the most detailed point clouds, but because time grows exponentially with the size of the dataset, this method is impractical for collections of more than a few hundred individual images.

3.4.2 Vocabulary Tree Feature Matching

Vocabulary Tree Feature Matching is an optimization that speeds up feature matching by comparing features only against other features that are visually similar. By indexing features using a vocabulary tree, the matching algorithm can quickly narrow the search to features likely to be a match, making Vocabulary Tree Feature Matching ideal for large unordered datasets.

3.4.3 Sequential Feature Matching

Sequential feature matching assumes that images were taken in order, and it compares images only to the nearby images in the sequence. This greatly speeds feature matching when dealing with large image collections, since each image needs to be matched against only a small, constant subset of the total image collection. Sequential feature matching can make use of vocabulary trees to detect loops in images that are distant sequentially.

<table>
<thead>
<tr>
<th>Feature Matching Method</th>
<th>Matched Images</th>
<th>Total Matching Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Exhaustive</td>
<td>360</td>
<td>16m 16s 860ms</td>
</tr>
<tr>
<td>Vocabulary Tree</td>
<td>357</td>
<td>12m 4s 80ms</td>
</tr>
<tr>
<td>Sequential</td>
<td>358</td>
<td>2m 46s 800ms</td>
</tr>
</tbody>
</table>

Table 2: Comparison of Feature Matching Algorithms, tested on the Alameda House dataset.
A comparison of feature matching methods is given in table 2. Although the Exhaustive feature matching algorithm performed best, successfully matching the most images, the Vocabulary Tree and Sequential feature matching algorithms performed nearly as well, and with dramatic time savings.

4 Processing Side Optimizations

After capturing and reconstructing the 3D point cloud, a number of methods are required to generate 2D architectural drawings from the reconstructed 3D point clouds.

4.1 Filtering Methods

Figure 2: The Saybrook Dining Hall, before (left) and after (right) passthrough and statistical outlier filtering.

The generated 3D point cloud contains not only information on the walls of the building but points corresponding to furniture, people, floors, ceilings. Additionally, mistakes in feature matching generate extraneous noise—scattered points that do not correspond to real-life geometry. In order to extract the floor plan of a building, we must first filter out all of these external points. The following subsections detail different filtering techniques.

4.1.1 Passthrough Filter

Points corresponding to the floor and ceiling of the building must be excluded from the floor plan, which ideally contains only the vertical walls of the structure. Because the floor and ceiling generally reside at a constant elevation, the passthrough filter, which trims points outside of a certain vertical range, allows the simple elimination of the floor and ceiling. Using an initial passthrough filter allows us to efficiently remove information that may be damaging to successive steps.

4.1.2 K-Nearest Neighbors Filter

Measurement errors in matching contribute to extraneous points in the point cloud that do not correspond to real-life structural geometry. Because the den-
Figure 3: The YUAG, before and after passthrough and statistical outlier removal filtering. The passthrough filter removes points in the gallery ceiling while the statistical outlier remover filter cuts down on noise generated from measurement errors.

The density of this noise is generally lower than that of points corresponding to the structure, we can eliminate these random errors by using a K nearest neighbors filter. This filtering algorithm searches for neighbors using a $k$-d tree and removes points if they do not possess a minimum number of neighbors within a specified radius. One drawback of the K Nearest Neighbors Filter is that it is not adaptable to point clouds of different density; the same KNN filter that works well on a dense point cloud might remove important points from a less densely populated point cloud. Consequently, we have decided that the following filter, Statistical Outlier Removal Filtering, is a better option that is as effective while being more flexible.

4.1.3 Statistical Outlier Removal Filter

Another filtering method for the elimination of outliers is the Statistical Outlier Removal filter. Using a $k$-d tree to find neighboring points, the filter determines the mean distance between a point and its 50 nearest neighbors. If a point is abnormally far away from its 50 nearest neighbors based on a Gaussian distribution, it is considered an outlier and is pruned from the cloud. The Statistical Outlier Removal Filter effectively removes noise from our models, and was adaptable over all datasets with minimal tuning.

4.1.4 Voxel Grid Filter

The raw point clouds, which sometimes consisted of several hundred thousand points, were often too large for efficient computation—algorithms such as Euclidean Clustering were inefficient on such large collections of points. In order to downsample the data, allowing for faster computation, we utilized a Voxel Grid Filter. The Voxel Grid Filter collapses points to the nearest location on a 3D voxel grid. By adjusting the size of these voxels, we could effectively control the number of points vs. the accuracy of point location.
4.2 Non-Structural Object Removal

Figure 4: Isolation and elimination of furniture, people, and other objects using Region Growing Segmentation. Individual objects are given unique colors.

Because our architectural drawings should represent the simplified layout of the building structure and not furniture, people, or other objects that may be present in the building, it is necessary to remove these objects from the point cloud. The following two sections detail different methods for extracting these objects.

4.2.1 Euclidean Cluster Segmentation

Euclidean cluster extraction is a method for segmenting the scene into objects by grouping points that are close together. Using a $k$-d tree representation of the input cloud, the Euclidean Clustering algorithm groups points by distance. Euclidean clustering effectively isolates objects that are far from structural walls, such as people and furniture.

4.2.2 Region Growing Segmentation

Another method for segmenting point clouds, region growing segmentation considers not only the distance between points but also the smoothness of a potential surface. The algorithm works by calculating normals for each point and grouping collections of points that form smooth, continuous surfaces. The advantage of Region Growing Segmentation over Euclidean Cluster Segmentation is that by considering curvature, Region Growing Segmentation can separate objects that are close spatially but disjoint in smoothness. For example, in our
tests, Region Growing Segmentation is more effective than Euclidean Cluster Extraction at isolating furniture and objects that are close to walls.

4.3 Wall Detection Methods

![Figure 5: Wall detection enables distortion correction. The raw point cloud, shown in white, suffers from error accumulation, has walls that are curved and at odd angles. The corrected and filtered plan, shown in red, aligns the walls more accurately.](image)

Three major factors contribute to point clouds that are far from idealized planes. Firstly, random noise due to measurement errors results in point clouds that are noisy and ill-defined. Secondly, objects such as light-fixtures and small scale details result in wall clouds with bumps protrusions. Finally, large structures that require many images to reconstruct suffer from distortion due to error accumulation. Because we would like the walls in our generated images to be sharp, simple, and straight, we require a method to correct for these factors. By detecting the general shape of the wall using wall detection techniques, we can compress the points into an idealized surface. The following sections detail different methods of wall detection.

4.3.1 Hough Transformation

The Hough transform is a technique for isolating and extracting features such as lines from images. The basic principle is a voting procedure–points contribute votes to possible lines based on their proximity to the line. The lines with the most votes can then be extracted. We implemented a modified version
of the classic Hough transform; instead of detecting lines in a 2D image, our
implementation detected planes in a 3D point cloud. The Hough transform
worked well for detecting large walls, but struggled to isolate smaller walls and
surfaces.

4.3.2 Random Sample Consensus (RANSAC)
Random Sample Consensus is another method for isolating walls by estimating
the parameters of surfaces. RANSAC is an iterative algorithm that grows and
refines a collection of points and model until the collection reaches a specified
size [2]. The advantage of RANSAC over other parameter estimation techniques
such as Ordinary Least Squares regression is a resistance to outliers—points that
are not close matches to the current model have no influence over the result.
Our implementation of RANSAC successively detected surfaces and removed the
detected points from the search space, continuing until fewer than 10% of the
raw point cloud remained. In our tests, RANSAC generally performed better
than the Hough transform, especially for smaller surfaces.

4.4 Wall Interpolation

![Image](image.png)

(a) SY Basement  (b) SY Dining Hall

Figure 8: Representation of hole filling algorithm. Detected points are shown in white, interpolated points are shown in red.

Despite the optimizations discussed above, sparse point clouds necessitated data interpolation. Using the estimated surface parameters, we were able to fill in areas of the model that would likely be continuations of the surface, but were missing points. Figure 8a shows our scheme in action. The hole filling scheme effectively interpolates data across small holes and smooths small-scale details, resulting in a simpler, higher level representation of walls.

5 Discussion and Evaluation of Current Methods

Using the techniques discussed above, we created a pipeline for automatically generating architectural plans from 2D image collections. An evaluation of the performance of these methods on the datasets is given in the following section. Ground truth plans were measured by hand and modeled with Solidworks.
Figure 9: Comparison between ground truth and generated plans for all four datasets.
5.1 Distortion

Our pipeline, which combined the methods detailed in sections 3 and 4, successfully generated architectural plans that matched the general geometry of the test structures. For large surfaces, distortion was effectively controlled; the wall fitting methods produced walls that were straight and aligned at right angles. However, the wall fitting methods struggled to detect the smaller surfaces in more complex models, and thus were less effective at generating clean lines in complex models. For instance, our algorithms failed to isolate several walls in the YUAG, so the plan, shown in figure 9, exhibits walls suffering from high distortion. The same methods, however, worked well with structures composed of larger, flatter surfaces, such as the SY Basement. In general the algorithm effectively controls drift and generates models with accurate proportions. When comparing the relative wall dimensions, the generated plans were accurate to within 6”, even over large distances. More advanced filtering and wall detection could yield further accuracy increases.

5.2 Completeness

Surfaces with high detail produced dense point clouds that enabled accurate wall extraction. Surfaces with lesser detail were often capable of being interpolated, and still resulted in complete surfaces. However, there was one key scenario in which the algorithms struggled: windows. In the generated plan for the Alameda House, shown in figure 9, our algorithms fail to reconstruct parts of the eastern wall, which is made up of floor-to-ceiling windows in real life. Similarly, the pipeline struggles on the southern wall of the YUAG, which is also blanketed in floor-to-ceiling windows. Structure from motion algorithms generally have difficulty reconstructing transparent surfaces, so it is unsurprising but unfortunate that we see similar results with our algorithms.

5.3 Resistance to Furniture and Non-structural Elements

For the most part, our algorithms were very successful in eliminating furniture, people, and other objects, and in smoothing unnecessary details. The SY Dining Hall plan, shown in Figure 9, showcases excellent removal of tables, chairs, and lighting fixtures. The generated plan for the Alameda House, in Figure 9, illuminates a tricky example. Our algorithms failed to eliminate a 12 ft ladder, which shows up in the top left corner of the room. The temporary ladder, which should not have been present in the plan, produced a similar point cloud signature to a permanent column. Future work could focus more specifically on differentiating temporary objects from structural elements.

6 Summary and Next Steps

This report examines methods for generating architectural plans from images, which could potentially yield faster and more accurate drawings than current
methods. We discuss several capture side optimizations in the Structure from Motion process, and we examine methods for distilling raw point clouds to 2D architectural drawings. Finally, we tested and evaluated a pipeline using the considered techniques. Our current pipeline is capable of generating accurate albeit rough floor plans.

There are a number of improvements that could be made in order to improve the quality of future plans. From a capture side perspective, several ideas could be explored to improve the speed and accuracy of matching. For instance, if each image was tagged with an approximate location, a feature matching algorithm would only have to consider nearby images. Such an improvement would greatly increase the matching speed on large image collections, and would reduce noise in the generated point cloud caused by false matches.

From a processing side perspective, we should investigate possible improvements to our current wall detection methods. In doing so, we would be able to further correct for distortions and generate cleaner models, especially for complex structures in which current methods struggle, such as the YUAG.

Additionally, future researchers could consider using synthetic benchmarks to evaluate generated plan accuracy. Bianco et al. utilize 3D modelling software to create both target models and simulated input images [1]. By testing our pipeline on a virtual structure, we could more effectively control variables such as lighting and detail, and evaluate the results against ground truth plans of infinite precision.

References


