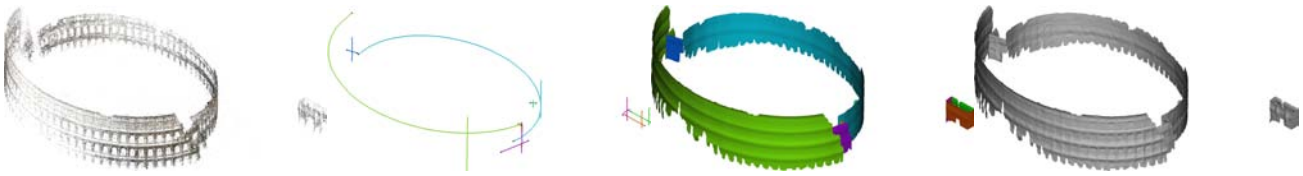
Schematic Surface Reconstruction

Changchang Wu[1,2], Sameer Agarwal[1,2], Brian Curless[1], and Steven M. Seitz[1,2]

[1]University of Washington [2]Google Inc.



(a) 200K SfM points (b) Schematic representation (c) Swept surfaces (d) Surface with displacements

Figure 1. Schematic surface reconstruction of the exterior of the Colosseum

Abstract

This paper introduces a schematic representation for architectural scenes together with robust algorithms for reconstruction from sparse 3D point cloud data. The schematic models architecture as a network of transport curves, approximating a floorplan, with associated profile curves, together comprising an interconnected set of swept surfaces. The representation is extremely concise, composed of a handful of planar curves, and easily interpretable by humans. The approach also provides a principled mechanism for interpolating a dense surface, and enables filling in holes in the data, by means of a pipeline that employs a global optimization over all parameters. By incorporating a displacement map on top of the schematic surface, it is possible to recover fine details. Experiments show the ability to reconstruct extremely clean and simple models from sparse structure-from-motion point clouds of complex architectural scenes.

1. Introduction

Technology for depth sensing is becoming commonplace, with the advent of modern structure-from-motion systems [11] and systems like Kinect [6]. Just by waving a camera or depth sensor around a room, you can now capture a wealth of 3D data. However, there is a huge difference between a cloud of 3D points and what an architect would call a "model." Architects prefer *schematic* representations such as floor and profile plans, which factor the scene into a set of planar lines and curves. The key property of schematics is that they are concise and easy to understand, yet provide an accurate and detailed representation of scene geometry.

This paper addresses the problem of converting sparse, structure-from-motion point clouds into schematic representations composed of planar curves. We focus particularly on the case of *architectural* scenes, and employ domain constraints that are tailored for this application. Compared to most prior work on surface reconstruction, our approach has the following advantages.

- **Understandability**: the representation is easily interpretable by humans, as it looks and behaves like a sketch (e.g., it can be easily edited).
- **Simplification**: Even complex 3D structures are representable with a handful of curves; the representation is highly compressed.
- **Completion**: the derived curves enable filling in holes and interpolating a dense surface from a sparse point cloud.
- **Fidelity**: by solving for a regularized height field on top of the schematic representation, we show that it's possible to recover the fine details present in the input.

Our schematic surface representation is composed of a network of planar *transport* and *profile* curves. The transport curve network captures the architectural notion of a *floor plan*, as the curves are all parallel to the ground plane. These curves can be grouped into one or more floor plans. The profile curves capture how the surface is vertically extruded, and allow for a wide range of both piecewise planar and curved surface shapes. This representation builds on classical work on swept surfaces and generalized cylinders [3], specialized to capture architectural constraints and generalized to enable modeling of complex scenes.

The main contribution of the paper is to introduce the schematic surface representation and robust algorithms for fitting this representation to sparse point could data. We

978-1-4673-1228-8/12/$31.00 ©2012 IEEE

focus primarily on point clouds derived from structure-from-motion (SfM) algorithms instead of Kinect or laser scans, as the former is more challenging due to greater sparsity and number of holes. However, we've tested the same algorithms on stereo and laser scan data, and they also work well on denser data.

The problem of fitting swept surfaces to point clouds has a long history in the computer vision literature, going back to Binford's work in the 1970's on reconstructing generalized cylinders from laser scans [3]. Almost all prior work in this area, however, focused on simple objects captured in a laboratory or other controlled environment, and models consisting of a single primitive or a few parts [9, 10, 14]. In contrast, we are the first to use collections of swept surfaces to model complex, large-scale architectural scenes. To scale to scenes of this complexity, we introduce a *network* of interconnected transport and profile curves, together with domain constraints (e.g., a ground plane). Our focus on SfM data, which is comparatively sparse and incomplete, also represents a departure from prior art (which focused primarily on dense range data or photographs). Another significant innovation is our use of 2D implicit functions (based on merging signed distance functions [4, 5]) to compute optimal, non-parametric representations of the profile and transport curves. While implicit techniques are popular for merging range scan data, they have not previously been used in the context of swept surface reconstruction. We also introduce a *global optimization* framework for refining the transport and profile curves simultaneously given data-fit and regularization terms subject to range constraints, using a quadratic program. A final innovation is our use of a displacement map on top of the swept surface, which captures fine details of the original point cloud, and is optimized using quadratic programming.

2. Schematic Surface Modeling

Our schematic representation is inspired by architectural schematics, which are composed of horizontal and vertical plan views of the scene. Each plan view is composed of a set of inter-connected contours, delineating walls and other architectural elements. We introduce a surface modeling approach based on the same principles of describing scene content by inter-connected horizontal and vertical contours.

We assume an architectural scene is generated by a set of horizontal *transport* and vertical *profile* curves. More specifically, transport curves lie in planes parallel to the ground, while profile curves lie in planes that are orthogonal to the ground. We also require the transport curves to be simple and relatively low-curvature (but two transport curves may meet at a sharp corner).

The scene consists of a network of swept surfaces, each of which is generated by sweeping a profile curve along its associated transport curve. Multiple profiles

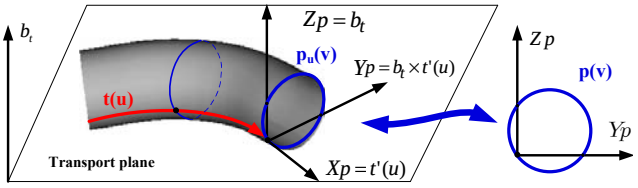


Figure 2. Swept surface representation. The transport curve $t(u)$ stays in a plane with normal b_t, while the profile curves $p(v)$ are in coordinate systems located at $t(u)$ and rotated to $[Xp, Yp, Zp]$.

are also allowed to share a common transport curve, for example, interleaving structures on building facades can be modeled by alternating profiles. The fine details, that are not captured by swept surfaces, are represented as displacements from the base surface.

2.1. Swept Surface

In this paper, we use the term *swept surface* to refer to the Planar Right Constant Generalized Cylinder [13]. Such surfaces are generated by sweeping a constant planar profile curve along a planar transport curve, with a constraint such that the profile curve is orthogonal to the transport curve directions. For example, a torus can be generated by sweeping a small profile circle along a large profile circle.

Let $t(u)$ be a transport curve with arc length (unit speed) parameterization. The binormal b_t of the transport curve is the ground plane normal that all transport curves share in architectural scenes. As illustrated in Figure 2, without loss of generality, we define the planar profile curve $p(v)$ in plane $X = 0$ and require that $p(v)$ goes through 0, and the rotation applied to a profile curve is given by

$$R(u) = [t'(u), b_t \times t'(u), b_t], \tag{1}$$

where $t'(u)$ provides the transport direction. The resulting swept surface is defined as

$$S(u, v) = t(u) + R(u)\, p(v). \tag{2}$$

This representation can express a large variety of shapes. In particular, planes, extruded surfaces, and surfaces of revolution are special cases of swept surfaces. The house-like object in Figure 4 is also a single swept surface. To handle more complex scenes, we extend the simple swept surface model to allow multiple profiles to share a common transport curve, which is parameterized as follows

$$S(u, v) = t(u) + R(u)\, p^{k(u)}(v), \tag{3}$$

where $k(u)$ is the piecewise constant function that chooses a profile curve $p^{k(u)}$ for each u.

Since architectural building are solid objects, we assume our swept surfaces non-self-intersecting. For any point $t(\mu)$ on the transport curve, its profile curve $S(\mu, v)$ must satisfy

$$||S(\mu, v) - t(\mu)|| \leq ||S(\mu, v) - t(u)|| \quad \text{for any} \quad u \neq \mu. \tag{4}$$

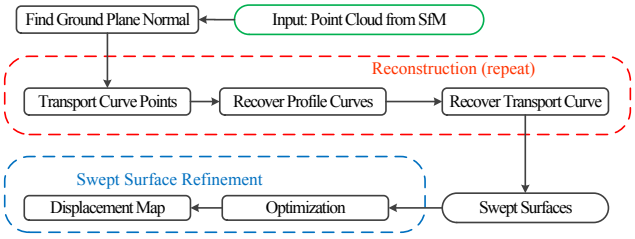


Figure 3. The diagram of our method. Swept surfaces are initialized by establishing transport and profile curves; The refinement stage improves the schematic surfaces via global optimization.

As a result, only the partial profile curves that satisfy the inequality are kept when profile curves intersect.

3. Swept Surface Reconstruction

As illustrated in Figure 3, our surface reconstruction first finds the ground plane normal, and then extracts multiple swept surfaces, of which the transport curves lie in horizontal planes. This section will introduce the basic reconstruction framework that initializes a set of swept surfaces, which will be further optimized by our refinement algorithm in Section 4. We encourage readers to watch our illustration video at the project website [2].

3.1. Preprocessing and Plane Intersections

Let x_i be an input 3D point, n_i its normal direction, c_{1i} and c_{2i} its two principal directions, and t'_i its transport direction. We estimate n_i, c_{1i} and c_{2i} with PCA on the SfM points within a distance threshold T_R. T_R is chosen such that the first quartile of the number of neighboring points across all inputs points is 100, which ensures for most points that the estimation of point normal and principal directions are robust to noise. The signed normal directions are chosen to point toward the SfM cameras and are globally consistent. To handle unknown sensor locations, the consistent signs can be obtained by using, for instance, the propagation method in [5].

Our curve reconstruction method employs the point cloud intersection with two types of planes: 1) *transport planes*: the planes perpendicular to the transport binormal, and 2) *profile planes*: the planes through a surface point x_i and perpendicular to its transport direction t'_i. The curves on the two types of planes are the transport curves and profile curves respectively. Given an input point cloud, we call the point subsets on the planes *transport slice* and *profile slice* accordingly. In practice, we keep the points within a small distance threshold $\frac{T_R}{8}$ for plane intersection, and skip planes that have fewer than 20 points.

3.2. Finding Ground Plane Normal

The model of swept surface enables us to automatically recover the transport binormal b_t from input point data. Rom and Medioni [10] prove that the principal directions of a swept surface $S(u, v)$ are $R(u)p'(v)$ and transport direction $t'(u)$, where $t'(u) \perp b_t$ is always true. Hence the transport direction t'_i of each point x_i corresponds to either c_{1i} or c_{2i}. Since transport directions t'_i lie in planes perpendicular to the ground plane normal b_t, every point should have one principal direction perpendicular to b_t. In practice, b_t should be perpendicular to one principal direction of the largest subset of points. With a simple thresholding[1] and by using boolean values as 1 or 0, the desired direction is defined by $\arg\max_b \sum_i (c_{1i} \perp b) \vee (c_{2i} \perp b)$.

We introduce two improvements to this simple binormal selection strategy to deal with planar structures and extruded surfaces. Planar structures (facades) have undefined principal directions, for which the plane normal is one of the many solutions in general. We suppress plane normals by not counting a point x_i for a candidate transport binormal b if $n_i \parallel b$. Another ambiguous case is a vertically extruded surface (e.g. cylinder). The ground plane normal and any direction parallel to the ground are equally good solutions for general swept surfaces. However, in the context of architectural scenes, we favor the direction of extrusion, so we also count a point if its normal is perpendicular to the binormal. The ground plane normal is given by

$$b_t = \arg\max_b \sum_i ((c_{1i} \perp b) \vee (c_{2i} \perp b) \vee (n_i \perp b)) \wedge (n_i \nparallel b), \tag{5}$$

which we solve via RANSAC. Within the sampling loop, we randomly choose two points i and j that satisfy $n_i \nparallel n_j$, and propose four candidate directions $c_{1i} \times c_{1j}$, $c_{1i} \times c_{2j}$, $c_{2i} \times c_{1j}$, and $c_{2i} \times c_{2j}$, because b_t is perpendicular to one principal direction of each point.

Having solved the ground plane normal, the planar transport direction for each point x_i is given by

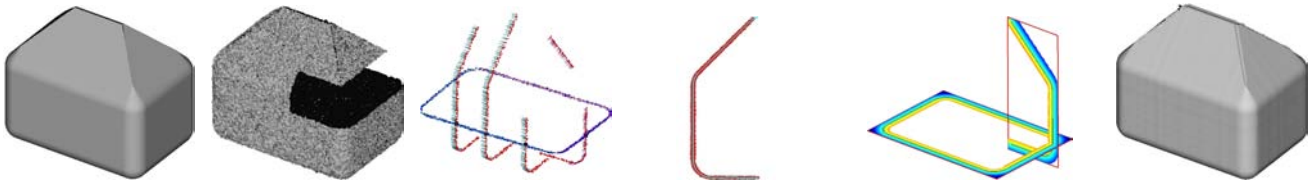
$$t'_i = \begin{cases} \frac{b_t \times n_i}{|b_t \times n_i|} & n_i \nparallel b_t \\ \text{Undefined} & \text{otherwise,} \end{cases}$$

because the transport directions are perpendicular to both surface normals and transport binormal. The rotation matrix of each point is defined as $R_i = [t'_i, b_t \times t'_i, b_t]$. It is not a problem to have some transport directions undefined, and we find that such regions can be still reconstructed from the profile planes defined by other points.

3.3. Choosing The Transport Curve Points

In order to define the transport curve and profile curve, we first need to pick a transport plane. The choices of $p(v)$ and $t(u)$ for a swept surface are not unique. Given any point q on $p(v)$, the same swept surface can also be generated by

[1] Two normalized vectors f_1 and f_2 are considered parallel if $|f_1 \cdot f_2| > 0.99$ or considered perpendicular if $|f_1 \cdot f_2| < 0.04$



(a) Synthetic surface (b) Corrupted input (c) Profile slices (d) Accumulated profile (e) Curve reconstruction (f) Our reconstruction

Figure 4. Swept surface reconstruction with missing data and noise. (a) We generate a synthetic swept surface with a size of $300 \times 200 \times 260$ and a mesh resolution of 2. (b) As input to our reconstruction, we corrupt the original surface by adding Gaussian noise (with standard deviation of 1) to each dimension of the original points and cropping out a small part. (c) The blue points are the transport points, and the four red curves are profile slice examples, which can be incomplete. The short line segments on the points indicate the transport directions. (d) The profile points accumulated by transforming the profile slices to a common profile plane. (e) The signed distance fields of the curves. (f) Our unoptimized reconstruction of the corrupted input is smooth and complete. Our reconstruction has a mean absolute error of 0.18 and RMS error of 0.61 when compared to ground truth.

using $p_q(v) = p(v) - q$ and $t_q(u) = t(u) + R(u)q$. As a result, we are free to choose any transport plane to define the transport curve for swept surface reconstruction.

For the sake of robustness, we choose the planes where the points have minimal noise and minimal curvature. By definition, the angle between b_t and surface normal n_i stays constant on a transport curve, such that $b_t \cdot n_i = (0,1,0)^T \cdot p'(v)$, so we measure the noise level of a given transport plane π_t by the variance of the angles $\sigma_\angle(\pi_t) = \text{stddev}\{\angle(b_t, n_i) | x_i \in \pi_t\}$. Low curvature points improve stability and are thus more suitable for our reconstruction. By projecting the point normals to the transport planes, we can estimate the transport curve curvature κ_i for each point, and we use $\sigma_c(\pi_t) = \text{rms}\{\kappa_i \,|\, x_i \in \pi_t\}$ to measure the overall curvature of a transport plane. Finally, we define the cost of choosing a plane π_t by $\sigma_c(\pi_t) + \sigma_\angle(\pi_t)$. We generate a list of transport planes with a small equal spacing of $\frac{T_R}{8}$, remove the ones that have below-average point counts, and choose the transport plane with the smallest cost.

To find the points that belong to a single transport curve, we obtain a transport slice by intersecting the transport plane with the input point cloud. One possible problem is that the plane may contain multiple transport curves. For example, the intersection with a torus generates one inner circle and one outer circle. Therefore, we look for maximum connected components from the transport slice and process each component as the points of a single transport curve. Here the connectivity is defined by using a simple threshold. Each low-curvature point is connected to another point in the same transport slice if they have similar normals and are within a distance range[2].

It is now possible to extract a transport curve from the points of a connected component; however, a single slice is often unreliable due to noise and missing data. Instead, we use the transport curve points to define a set of profile slices, robustly reconstruct their common profile curve, and fit the transport curve afterwards by using all the profile slice points. We will describe our reconstruction of profile curve and transport curve in Section 3.4 and 3.5, respectively.

3.4. Reconstructing The Profile Curve

The transport curve points define a set of profile planes. The profile slices on these planes share a common profile curve, but have different locations and rotations. Not all the points on a profile plane should be considered in the profile slice. First, the point normals should be perpendicular to the transport direction, so we verify this perpendicularity to filter out the points that do not belong to profile curves. Second, we filter the points according to the non-self-intersection assumption, and keep the partial slice that is close to each transport point. For example in Figure 4(c), the profile slices are halves of the profile plane intersections.

We transform the profile slices to the canonical profile plane for merging. Let π_p^i be the profile slice of a transport point x_i. Any point x_j on the profile slice can be transformed back to the canonical profile plane coordinate y_j^i by

$$y_j^i = R_i^{-1}(x_j - x_i). \tag{6}$$

Similarly the normal direction can be transformed to the profile plane coordinate as

$$n_j^i = R_i^{-1} n_j. \tag{7}$$

When y_j^i and n_j^i do not lie perfectly on the profile plane, we take their 2D projections. Like the merging of laser scan images [4], the merging of profile slices effectively handles noise and missing data on single profile slices.

Similar to the 3D volumetric approaches [5], we reconstruct the profile curve implicitly by computing a 2D signed distance field. Each oriented point (y_j^i, n_j^i) can be viewed as a small curve pieceh, so its depth function is

$$d_j^i(y) = (y - y_j^i) \cdot n_j^i. \tag{8}$$

[2]Specifically for a point x_i with $\kappa_i < \frac{1}{5T_R}$, we consider a point x_j connected to x_i if $n_i \cdot n_j > 0.95$ and x_j is within a rectangular range $|n_i \cdot (x_i - x_j)| < \frac{T_R}{4}$ and $|t_i' \cdot (x_i - x_j)| < 8T_R$

For convenience, we denote the distance along the tangent direction as $h_j^i(y) = |(y - y_j^i) \times n_j^i|$. We define the signed distance field as a weighted average of the individual depth functions with the following Gaussian weighting function

$$w_j^i(y) = \exp\Big(-\frac{d_j^i(y)^2}{2\sigma_1^2} - \frac{h_j^i(y)^2}{2\sigma_2^2}\Big), \tag{9}$$

which attenuates the weight of a point along depth direction and tangent direction. Our signed distance field is

$$D_p(y) = \Big(\sum_{(i,j)} d_j^i(y) w_j^i(y)\Big) \;/\; \Big(\sum_{(i,j)} w_j^i(y)\Big). \tag{10}$$

The iso-curve is then extracted as a list of 2D vertices by using the marching squares algorithm [8]. We choose $\sigma_1 = \frac{2T_R}{5}$ and $\sigma_2 = \frac{T_R}{5}$. The grid resolution of the distance field is chosen to be σ_2, and each point has an extent of $8\sigma_2$. We use the same weighting function to compute a smooth normal field $N_p(y)$ from n_j^i, which we will use to initialize normal directions in Section 4.1.

3.4.1 Clustering Profile Slices

It is common in real scenes for different profiles to share a continuous transport curve. For example in Figure 5, there are two slightly different shapes along the corridor. Without accounting for the differences, we may end up with doubled profile curves. We address this problem by recovering multiple profile curves along a single transport curve, via a simple clustering procedure.

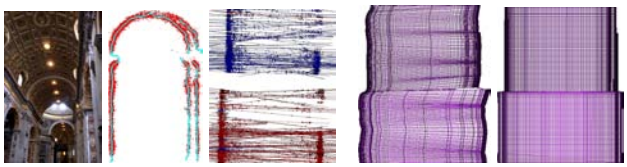
We identify clusters of consistent profile slices and recover a profile curve for each cluster as follows. We repeatedly pick a single profile slice as a seed to define a new cluster. All the profile slices that match with the seed curve are assigned to the cluster. Given a profile curve with distance field $D_p(y)$, we match a slice π_p according an inlier ratio:

$$|\,\{q \,|\, q \in \pi_p, D_p(q) < \sigma_1\}\,| \;/\; |\,\pi_p\,| \tag{11}$$

with a threshold $T_S = 90\%$. The profile curve for the cluster is then updated by using all the profile slices of the cluster. New clusters are added until the overall inlier ratio reaches a threshold $T_C = 90\%$. Our experiments produce one or two profile curves for most transport curves.

The seed slice selection is based on a score that evaluates the number of points covered by the single slice curve reconstruction. Given a set of 2D profile points R, we use the function $W(R, y) = \sum_R w_j^i(y)$ to approximate the number of points at y, and define the density score of a slice π as $Q(\pi, R) = \sum_{y \in \pi} W(R, y)/W(\pi, y)$, where the weight $1/W(\pi, y)$ avoids multiple counting of the same locations. The slice with the highest score is chosen as the seed.

When multiple clusters of profile slices are detected, we substitute Equation 3 for Equation 2 for the swept surface reconstruction. The transport curve shared by multiple profile curves maintains the connectivity between multiple segments of surfaces.



(a) Image (b) Profile (c) Clusters (d) Unoptimized (e) Optimized

Figure 5. Swept surface refinement on one corridor of St Peter's Basilica. (b) The accumulated profile points. (c) The top-down view of two profile slice clusters in red and blue, (d-e) The unoptimized swept surface and the optimized swept surface.

3.5. Reconstructing The Transport Curve

The reconstructed profile curve allows us to robustly recover the transport curve. Equation 2 can be rewritten to obtain the transformation from surface to the transport plane: $t(u) = S(u, v) - R(u)\,p(v)$, where $S(u, v)$ and $R(u)$ are the position and the rotation at each point, and the profile curve $p(v)$ is already reconstructed. Given a point x_j, we estimate the corresponding curve point p_j^i on $p(v)$ by intersecting line $(y - y_j^i) \cdot Zp = 0$ with the curve[3]. Each point x_j on profile slice π_p^i is transformed to

$$z_j^i = x_j - R_j\, p_j^i, \tag{12}$$

where the rotation of R_j is used rather than R_i to take the rotation of each point into account. The normal direction of the transformed point on the transport plane can be defined according to its transport direction as

$$t_j' \times b_t. \tag{13}$$

Consequently, the accumulated 3D points are transformed to a set of oriented points in the transport plane.

To ensure that the transformations are consistent within each profile slices, the above two transformations require a sign correction by $t_j' \leftarrow \text{sign}(t_j' \cdot t_i')t_j'$ for each $x_j \in \pi_p^i$. One example of such a correction is the corridor in Figure 5, where the transport directions on two sides are initially opposite due to opposite normals. This is also why the orientation is defined according to t_j' instead of n_j.

Similar to the reconstruction of profile curves, signed distance field $D_t(z)$ and normal field $N_t(z)$ are constructed from the accumulated points on the transport plane. The transport curve is extracted as the iso-curves of $D_t(z)$, and $N_t(z)$ gives a smoothed normal field. In the case of multiple profile curves sharing the transport curve,

[3]This intersection is reliable only if the curve is not close to horizontal, so we use only the subset of accumulated points that satisfy $\angle(n_j^i, Zp) > 30^\circ$ for transport curve reconstruction.

we assign for each transport curve vertex the cluster that produce the highest density, which is defined as the sum of the Gaussian weight of each point at the vertex.

To deal with possible holes in the initial chosen transport plane, we search for additional profile slices to complete the missing part of the transport curve. Starting from the set of accumulated 3D points, we examine the profile slices of their neighboring points, which are now not limited to the initial transport plane. Although these transport points are not on the original transport plane, the reconstructed profile curve allows us to transform the new profile slices to the canonical profile plane coordinate and to the transport plane. Next, we match the transformed profile slices with the known profile curve by using the inlier ratio defined in Equation 11, and keep the ones that match the profile curve for transport curve reconstruction.

3.6. Network of Swept Surfaces

Given the recovered profile and transport curve, the mesh of the swept surface is easily generated by placing a transformed version of the profile curve at each transport curve vertex. Given the generated full mesh, we first trim the mesh by self-intersection according to Equation 4. Second, we crop the upper/lower vertices of a mesh to define a vertical boundary, so the surface does not extend to where there are no input points. Along each profile curve on the mesh, we start from the two ends of the curve, and keep trimming the vertices until there are input points within distance $\frac{T_R}{2}$.

A collection of swept surfaces is produced by repeating the single swept surface reconstruction on multiple connected components and on multiple transport planes. When one swept surface is extracted, the points within distance T_R to the surface are marked as covered, and excluded from being used by other swept surfaces. When there are multiple connected component on a transport plane, we process them in the order of decreasing sizes. After processing of one transport plane, we continue searching for new transport planes to model as many points as possible.

The schematic representation of a scene is defined by the network of swept surfaces, where the connectivity between swept surfaces is defined according to the transport curves. We find the transport planes that intersect most surfaces, and use the transport curves on such planes to define a floorplan. Figure 6 shows two examples of computed floorplans. Given a transport curve and one of its two ends, the closest transport curve within a threshold $4T_R$ gives a connected neighbor surface. We modify the horizontal boundaries of the meshes to be where neighbor surfaces intersect. Since all swept surfaces share the same transport binormal, mesh trimming is computed efficiently as curve intersections on each plane. When two transport curves approximately form a right corner, we extend the mesh along the transport directions of end vertices and perform the same intersection.

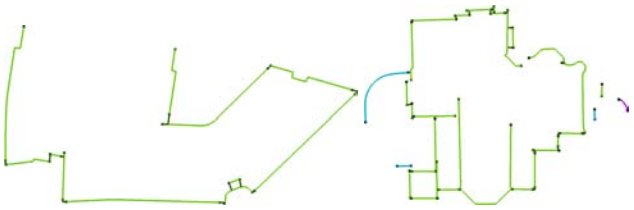


Figure 6. The reconstructed floorplans of the Allen Center and the Uris Library (see Figure 7 for their input points). The transport curves are color-coded by the common transport planes.

4. Swept Surface Refinement

In this section, we optimize the swept surfaces to better fit the data through two improvements. First, we introduce a global optimization to jointly refine the profile and transport curves based on data-fit and smoothness objectives. Second, we introduce a displacement map on top of the swept surface to recover fine-grained details.

4.1. Swept Surface Optimization

Up until this point, we have described a multi-stage procedure to reconstruct swept surfaces. We treat the result as an initialization which we now refine by jointly optimizing over the profile and transport curves to obtain an optimal trade-off of data-fit quality with smoothness objectives. We now introduce the set of objectives to optimize.

Let $S(u, v)$ be the initially reconstructed swept surface, and S_d (with $p_d(v)$ and $t_d(u)$) the optimized surface. We refine the swept surface to fit the subset of 3D points within distance T_R. Let (u_i, v_i) be the closest mesh vertex parameter for each x_i, the data-fitting cost is defined by the distances from the surface to its nearby 3D points:

$$E_{data} = \sum |(x_i - S_d(u_i, v_i)) \cdot N_s(u_i, v_i)|^2,$$

where $N_s(u, v) = R(u)N_p(v)$ is the normal direction for error estimation. Because the transport curve derivative directions are sensitive to the small errors in curve positions, we initially define $R(u) = [N_t(u) \times b_t, N_t(u), b_t]$ from the smooth curve normal field, and then adopt $R(u) = [t'(u), b_t \times t'(u), b_t]$ after the first optimization iteration.

Second, we expect the curve derivatives to be perpendicular to the curve normal fields, so we define a tangent fitting cost similar to the objective of Poisson reconstruction [7]

$$E_{tangent} = \sum (|p'_d(v) \cdot N_p(v)|^2 + |t'_d(u) \cdot N_t(u)|^2).$$

Here the derivatives are estimated from the discretized vertices with respect to constant speed parameterization.

Third, we optimize the smoothness of the swept surfaces by penalizing the second order derivatives of the curves

$$E_{smooth} = \sum (||p''_d(v)||^2 + ||t''_d(u)||^2).$$

In summary, the profile curve p_d and transport curve t_d are jointly optimized by minimizing the energy function

$$E_{sweep} = E_{data} + \lambda_n E_{tangent} + \lambda_s E_{smooth}. \quad (14)$$

We parameterize the curves by using a displacement variable on each curve vertex: $p_d(v) = p(v) + \beta(v)N_p(v)$ and $t_d(u) = t(u) + \psi(u)N_t(u)$, where the displacements are bounded in $[-T_R, T_R]$. This surface parameterization has much fewer parameters compared to typical mesh fitting, and thus enables efficient optimization. We approximate each error term by a linear function of β and ψ, and use Mosek [1] to solve the quadratic optimization. Note that the optimization requires the constraint $\beta(v_0) = 0$ for the vertex $p(v_0) = (0, 0, 0)$ to pick a unique one out of the family of solutions.

In each optimization iteration, we use the current curves to obtain $N_s(u, v)$, $N_t(u)$ and $N_p(v)$, then find the optimal displacement $\beta(v)$ and $\psi(u)$ to compute the optimized swept surface. A post-processing step is done after the optimization to remove any self-intersection in the curves. Experiments show that the errors in the initial transport curve may cause the optimization to get stuck in local minima. To improve the convergence, we add auxiliary variables $r(u) = t'_d(u)$, optimize only $r(u)$ in the first pass to smoothen the second order derivatives, and then jointly optimize all parameters for subsequent iterations.

Sharp corners in the transport curve are common in man-made scenes, but cause singularities in tangents and normals. We therefore downweight the tangent fitting error in areas where $\angle(N_t(u), t'(u)) < 45^\circ$ for transport curves and where there is an alternation of profiles when multiple profile curves share a common transport curve.

4.2. Displacement Map

To model the fine-scale geometric details, we introduce a scalar displacement map on top of the swept surface representation. The displaced swept surface is defined by

$$S_d(u, v) = S(u, v) + d(u, v)\ N_s(u, v), \quad (15)$$

where $d(u, v)$ is the displacement function. Similar to the joint optimization of curves, we consider both the fitting of input point data and the mesh smoothness

$$E_{disp} = E_{data} + \lambda_d E_{mesh}. \quad (16)$$

Given the optimized swept surface and its smooth normal N_s, we penalize big jumps along the normal directions by

$$E_{mesh} = \sum (|S_u \cdot N_s|^2 + |S_v \cdot N_s|^2), \quad (17)$$

where S_u and S_v are the two partial derivatives of the displaced swept surface. Like the previous optimization, we bound the displacements by $[-T_R, T_R]$, and solve the displacement map with Mosek quadratic optimization.

Figure of datasets	1	7(a)	7(b)	7(c)
# of SfM points	200K	514K	133K	689K
# of curve vertices	2K	3K	2K	7K
# of mesh vertices	141K	373K	97K	415K
Time on swept surfaces	3.0	12	1.2	10
Time on optimization	0.13	0.58	0.12	0.61
Time on displacements	0.38	1.6	0.25	1.2

Table 1. The statistics of our reconstruction and the timing of each step in minutes. The experiments are conducted on a PC with Intel Xeon X5680 3.33Ghz CPU and 12GB Ram.

5. Experiments and Discussions

We present schematic surface reconstructions on SfM points of complex architectural scenes: the Colosseum in Figure 1, St. Peter's Basilica, the Allen Center, and the Uris Library in Figure 7. Due to space constraints, we place additional results and a comparison with Poisson surface reconstruction at our project website [2].

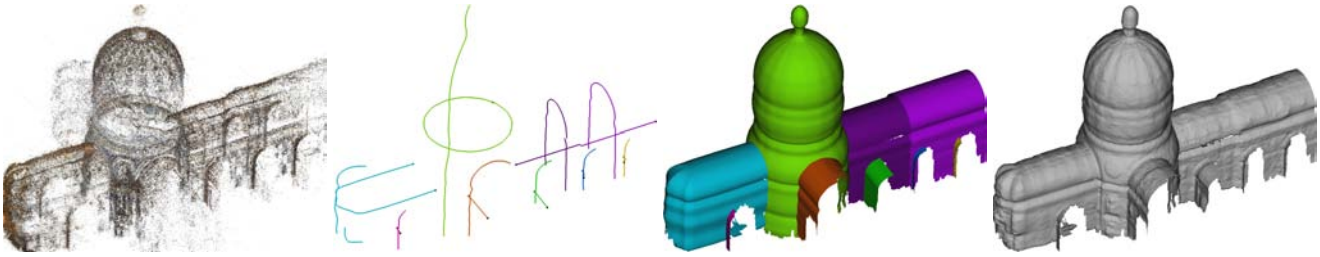
We ran all experiments with the same settings. All 3D scale parameters are proportional to T_R, which is chosen automatically according to point density. The resolution of the signed distance field $\sigma_2 = \frac{T_R}{5}$ is chosen such that a 10×10 grid within the radius on a planar surface yields one point in every cell. A uniforming 3D binning voxel grid with resolution T_R is used for nearest point queries. The optimization parameters are chosen experimentally for very smooth surfaces and then kept constant: $\lambda_s = 100, \lambda_n = 16, \lambda_d = 4$.

Experiments demonstrate the robustness of our method, which consistently detects the correct ground normal, produces clean curves and surfaces, and preserves details via the displacement maps. Table 1 shows the conciseness of our representation, which requires 2 orders of magnitude fewer parameters compared to the original point cloud.

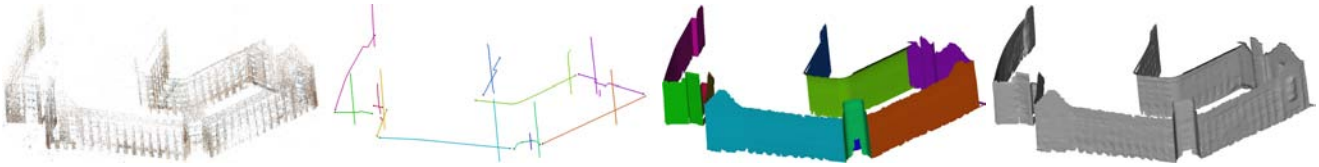
The point density in SfM models is often more than sufficient to recover high quality schematics. Indeed, we've found that a small fraction of SfM points (1% for the Colosseum, and 10% for the others) is sufficient for schematic reconstruction (see our website [2] for details). Our method also works reasonably well for dense MVS points, and a choice of slightly larger T_R can furthur improve the reconstruction by downweighting the details in MVS points.

Our algorithm has been tailored specifically for common architectural scenes. The single transport binormal and the piecewise constant model may be over-simplified for other type of scenes. While our approach works reasonably well for some objects (e.g. cups, bottles), more complex surfaces will break down into a collection of small pieces.

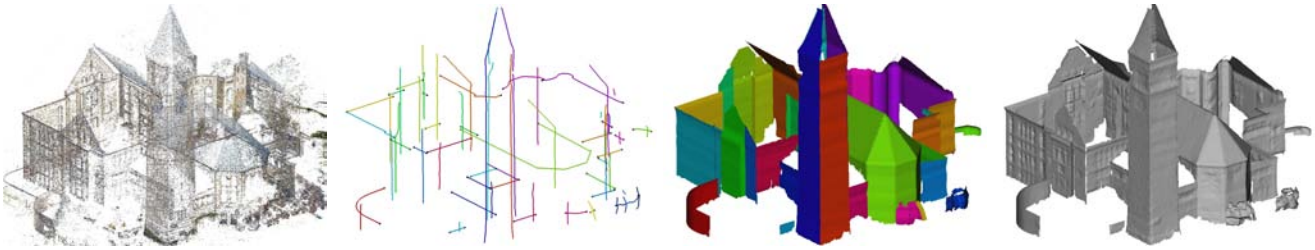
One limitation of our method is the uniformly applied scale T_R, which limits the processing of sparse regions. SfM does not produce enough points for poorly textured regions or places where the camera viewing angles are too oblique (e.g. the side corridors in Figure 7(a)), for



(a) St Peter's Basilica Reconstruction



(b) Allen Center Reconstruction



(c) Uris Library Reconstruction

Figure 7. Schematic surface reconstruction. The figures from left to right are input SfM points, schematic representation curves, swept surfaces, and swept surfaces with optimized displacements. The curves and swept surfaces are color-coded by the indices of the swept surfaces.

which our method produces holes. In the future, we would like to explore multi-scale approaches, symmetries or other high-level regularities to improve the accuracy and completeness of the reconstruction. In addition, our current choice of surface boundaries relies on the existence of input points, but this can be disturbed by outliers and missing data. In the future, we would like to explore more architectural priors to regularize the boundaries.

Acknowledgements We thank the authors of the PhotoCity project [12] for providing the Allen Center and Uris Library datasets. This work was supported in part by NSF grants IIS-0811878 and IIS-0963657, SPAWAR, the University of Washington Animation Research Labs, Intel, Microsoft, and Google.

References

[1] MOSEK optimization software, http://www.mosek.com. 7

[2] Schematic surface reconstruction project website. http://grail.cs.washington.edu/projects/schematic. 3, 7

[3] T. Binford and G. Agin. Computer description of curved objects. In *IJCAI*, pages 629–640, 1973. 1, 2

[4] B. Curless and M. Levoy. A volumetric method for building complex models from range images. In *SIGGRAPH*, pages 303–312, 1996. 2, 4

[5] H. Hoppe, T. DeRose, T. Duchamp, J. Mcdonald, and W. Stuetzle. Surface reconstruction from unorganized points. In *SIGGRAPH*, pages 71–78, 1992. 2, 3, 4

[6] S. Izadi, D. Kim, O. Hilliges, D. Molyneaux, R. A. Newcombe, P. Kohli, J. Shotton, S. Hodges, D. Freeman, A. J. Davison, and A. W. Fitzgibbon. Kinectfusion: real-time 3D reconstruction and interaction using a moving depth camera. In *UIST*, pages 559–568, 2011. 1

[7] M. Kazhdan, M. Bolitho, and H. Hoppe. Poisson surface reconstruction. In *SGP*, pages 61–70, 2006. 6

[8] W. E. Lorensen and H. E. Cline. Marching cubes: A high resolution 3D surface construction algorithm. In *SIGGRAPH*, pages 163–169, 1987. 5

[9] A. Pentland. Recognition by parts. In *ICCV*, 1987. 2

[10] H. Rom and G. Medioni. Part decomposition and description of 3D shapes. In *ICPR*, pages 629–632, 1994. 2, 3

[11] N. Snavely, S. M. Seitz, and R. Szeliski. Photo tourism: Exploring photo collections in 3D. In *SIGGRAPH*, pages 835–846, 2006. 1

[12] K. Tuite, N. Snavely, D.-y. Hsiao, N. Tabing, and Z. Popovic. Photocity: training experts at large-scale image acquisition through a competitive game. In *CHI*, 2011. 8

[13] F. Ulupinar and R. Nevatia. Shape from contour: Straight homogeneous generalized cylinders and constant cross section generalized cylinders. *IEEE TPAMI*, 17:120–135, 1995. 2

[14] M. Zerroug and R. Nevatia. Part-based 3D descriptions of complex objects from a single image. *IEEE TPAMI*, 21:835–848, 1999. 2