EUROGRAPHICS 2009 / P. Dutré and M. Stamminger
(Guest Editors)

Volume 28 (2009), Number 2

Completion and Reconstruction with Primitive Shapes

Ruwen Schnabel, Patrick Degener and Reinhard Klein†

Institut für Informatik II, Universität Bonn, Germany

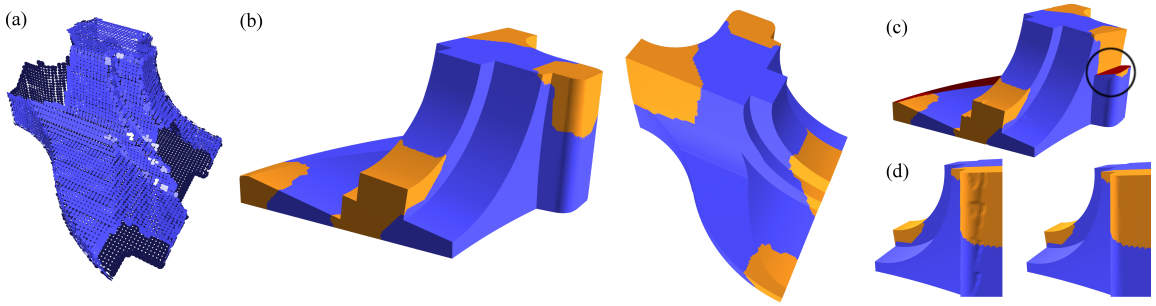

Figure 1: *Reconstruction of the fandisk model. Orange color signifies completed surface parts. (a) The input point-cloud with holes (b) Final result (c) Result without the connectivity enforcement algorithm of Sec. 5. The disconnected primitive highlighted in red cuts off part of the model. (d) Close-up views of result without consistent edge labels and final result (see Sec. 7)*

Abstract

We consider the problem of reconstruction from incomplete point-clouds. To find a closed mesh the reconstruction is guided by a set of primitive shapes which has been detected on the input point-cloud (e.g. planes, cylinders etc.). With this guidance we not only continue the surrounding structure into the holes but also synthesize plausible edges and corners from the primitives' intersections. To this end we give a surface energy functional that incorporates the primitive shapes in a guiding vector field. The discretized functional can be minimized with an efficient graph-cut algorithm. A novel greedy optimization strategy is proposed to minimize the functional under the constraint that surface parts corresponding to a given primitive must be connected. From the primitive shapes our method can also reconstruct an idealized model that is suitable for use in a CAD system.

Categories and Subject Descriptors (according to ACM CCS): Computer Graphics [I.3.5]: Curve, surface, solid, and object representations—

1. Introduction

Even despite considerable effort, data obtained with range scanners or stereo capture usually suffers from occluded or defective portions of objects that either could not be perceived during acquisition or have adverse material properties that hinder the scanning device. Nonetheless, a complete surface representation without holes is usually desired, and sometimes even required, for further processing or rendering. Therefore reconstruction algorithms must not only be able to recover the surface parts that have been captured, but must also synthesize plausible geometry in hole areas.

Previous work either uses general smoothness assumptions to derive the completing surface parts or relies on a database of suitable example cases from which a completing surface can be retrieved. We argue that for a large class of objects encountered in man-made environments, surface characteristics are well represented by a set of primitive shapes

† {schnabel,degener,rk}@cs.uni-bonn.de

(planes, spheres, cylinders, cones and tori). Besides global shape, these primitives also capture local surface differential properties. Moreover, their intersections naturally describe the structure of edges (see Fig. 1 and 2).

Thus, in this work we propose to exploit the information given by a set of shape primitives that has been detected on the input surface to automatically infer a closed reconstruction of an incomplete 3D model. The primitive shapes can be extended into the empty regions and serve as a guidance for hole-filling. Our algorithm differs from previous work in several important aspects:

- By using the primitive shapes as guidance for hole-filling, we find plausible completions that continue the geometric structure around the missing area. We can not only extend existing edges into the hole, but also derive the location of novel edges and corners in the synthesized surface from the primitives' intersections.
- Apart from filling of missing areas, our method also allows to use the guidance of primitive shapes to reconstruct a mesh that adheres to the primitives everywhere, i.e. not only in the holes. This results in an idealized noise free reconstruction that is composed only of the primitives and ignores fine surface detail (e.g. engravings). Such a reconstruction is suitable for processing in a CAD environment.
- Additionally our algorithm can also faithfully recover surface parts not approximated by primitives. Even fine details in the areas of the surface that are approximated by primitives can be recovered, while at the same time primitives do guide the completion of holes.

Our novel algorithm is based on an energy minimization approach that allows us to infer missing geometry from a set of surrounding primitives and reconstruct a closed, idealized 3D-model. In detail, our paper makes the following technical contributions:

- We derive a surface energy functional that incorporates the guidance given by the shape primitives. We propose a discretization that allows the application of an efficient graph-cut optimization algorithm.
- We give a novel greedy optimization strategy to minimize the above functional under the constraint that surface parts corresponding to a given primitive must be connected. This enables us to handle multiple holes in complex 3D models (see Fig. 1 (c)).
- We show that our method allows extraction of an idealized surface with sharp features. To this end we give an algorithm that resolves ambiguities arising from smooth transitions between primitives (see Fig. 1 (d)).

The proposed method is able to handle an arbitrary number of primitives that may have arbitrarily complex intersections which, to the best of our knowledge, was impossible previously. Finally, our method can also easily be adapted to the special conditions for completion of range-images or height-fields.

2. Previous work

The problem of hole-filling has been regarded from several different perspectives in previous work. Closely related to our setting is work on surface reconstruction and completion as well as the completion of range images. In the following the most relevant previous approaches from each area will be shortly reviewed.

Surface reconstruction involving fitted primitives has long been the standard in reverse engineering (see [BMV01]) but has also been considered in the graphics community [HDD*94] [JWB*06] [JKS08] [GSH*07]. While these methods usually support the reconstruction of sharp features at the intersection of primitives, they do not provide any means to infer larger regions of missing geometry from the information contained in the shapes.

Our method uses energy functionals similar to the general purpose reconstruction methods of Kazhdan et al. [KBH06], Hornung and Kobbelt [HK06] as well as Lempitsky and Boykov [LB07]. However, compared to their approaches, our method is the first to incorporate information from fitted primitives and to explicitly address the problem of surface completion.

Surface completion is traditionally part of surface reconstruction algorithms (see e.g. [CL96]), but has also received research attention in its own right. Most methods have been inspired by image inpainting approaches [BSCB00] [DCOY03] and can be attributed to one of two directions of research: (1) Methods based on level-set PDEs [DMGL02] [VCBS03] and energy minimization [CDD*04] (2) Example-based approaches [SACO04] [PMW*08] [WO02]. Methods in the first class focus on inferring smooth geometry in missing areas that also has smooth transitions to the existing surface parts. Therefore they cannot model the sharp features resulting from intersections of shape primitives. Moreover these methods cannot guarantee that for instance a hole in a cylindrical surface is also completed with the respective cylindrical geometry. The second class of methods is based on discovering (self-)similarity and regularity in or between models in order to infer the missing information from other fully captured surface parts. While example-based completion can achieve highly plausible reconstructions, it very much depends on the existence of suitable surface parts fitting into the missing area.

Podolak et al. [PR05] used a graph-cut based approach to resolve topological ambiguities of completing surfaces in 3D. In contrast to our approach no structural constraints other than smoothness can be imposed.

Mesh repair algorithms [Ju04] [Lie03] [BPK05] are able to fill small gaps in the input models but lack the capabilities to handle larger missing pieces and cannot propagate structure therein.

Range image completion has been studied by Fisher and colleagues in [SDF01] and [CLF02]. Similarly in spirit

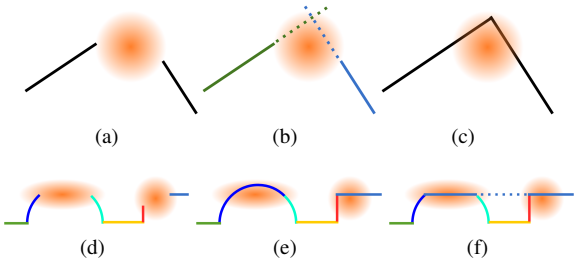

Figure 2: *(a) A hole is indicated by the orange area. (b) Planar primitives in the vicinity are extended. (c) Intersection define the completed surface. Bottom row: The effect of the connectivity constraint. (d) Primitives are color coded. (e) The desired reconstruction contains a completed circle. (f) If connectivity of the primitives is not enforced another reconstruction that cuts off the circle is also possible.*

to our approach, albeit limited to 2D, missing regions are filled by continuation of primitive shapes that reach a hole's boundary. However, only cases of two severed boundary edges are handled. Jia et al. propose a tensor voting approach to range image completion in [JT04]. Even though some discontinuities are preserved, intersections of surfaces are not explicitly handled.

3. Shape primitive guided completion

Given a point-cloud with oriented normals representing the potentially incomplete surface $\mathcal{S}^0$, we seek to reconstruct a closed surface $\mathcal{S}$ that propagates the geometric structure of the original surface into the missing areas. Our idea is to represent this structure by a set of shape primitives $\mathcal{P} = \{P_1, \ldots, P_n\}$ that has been detected on the incomplete surface $\mathcal{S}^0$, see Fig. 2. Shape primitives are given as implicit surfaces (planes, spheres, cylinders, cones and tori) of possibly infinite extent. Using this definition of structure, the completion problem can be put as follows: Find a suitable watertight surface $\mathcal{S}$ that approximates $\mathcal{S}^0$ and at any location adheres to at least one of the primitives $P_i \in \mathcal{P}$. We may later relax this condition if there exist areas of the input surface that cannot be represented by primitive shapes.

Since primitives are implicit surfaces of possibly infinite extent, it is possible that a primitive P_i is used to complete the surface in multiple disconnected regions that are far away from each other or dissimilar to the parts of $\mathcal{S}^0$ originally approximated by P_i. As demonstrated in the bottom row of Fig. 2 this can lead to undesirable completions or shortcuts. We therefore further require that all parts of the surface $\mathcal{S}$ following a primitive P_i should be connected.

While it is relatively easy to locally extend individual primitives into a hole region, intersections of multiple primitives can be highly complex and reconstruction becomes non-trivial. See Fig. 7 for an illustrating case where several planar primitives need to be intersected to form a complex rooftop geometry that has several sharp edge features. Our solution handles such complex intersections of an arbitrary number of primitives and gives plausible results.

In summary there are two characteristics that make up our primitive guided reconstruction: *Primitive adherence* makes the reconstructed surface follow the input primitives (see Sec. 4). Adherence is always in effect in hole areas, but surface parts that cannot be represented by primitive shapes at all as well as fine details can be handled using a traditional reconstruction method as outlined in Sec. 6. *Primitive connectivity* ensures that surface parts corresponding to a single primitive form a connected subset of the reconstructed surface (see Sec. 5) which, as argued above, is necessary for plausible reconstructions in case of multiple holes.

3.1. Shape detection

Before our algorithm can be applied, a set of shape primitives needs to be detected on the input surface. In our implementation we decompose the input point-cloud using the RANSAC-based approach of Schnabel et al. [SWK07], but methods as suggested by Cohen-Steiner et al. [CSAD04] or Wu and Kobbelt [WK05] would be applicable as well. An advantage of the employed method is its robustness with respect to noise in both positions and normals as demonstrated in the original paper. Even though the shape detection requires normal information, the correctness of normals close to sharp edges is not critical for the estimation of shape primitives. In our reconstruction, sharp features are deduced from the robustly fitted primitives directly, without relying on potentially noisy and incorrect normal information.

The shape detection provides us with an oriented set of primitive shapes $P_i \in \mathcal{P}$ where each primitive P_i is associated to a single connected support area $S_i \subset P_i$ that corresponds to the region of $\mathcal{S}^0$ approximated by P_i. While in practice shape primitives P_i approximate the original surface only up to a predefined tolerance, for the sake of simplicity the following discussion assumes that the surface $\mathcal{S}^0$ is given as the union of the support sets, i.e. $\mathcal{S}^0 = S_1 \cup S_2 \cup \ldots \cup S_n$. Thus we also have $S_i \subset \mathcal{S}^0$. We postpone the discussion of how details within the tolerance threshold and parts of the surface not covered by primitives can also be considered to Sec. 6.

4. Primitive adherence

In this section, we propose an energy functional that assigns any given closed surface $\mathcal{S}$ a cost according to how well the surface adheres to the given set of shape primitives $\mathcal{P}$. By minimizing this cost over the set of all closed surfaces we obtain a reconstruction $\mathcal{S}$ that is guided by $\mathcal{P}$ and satisfies the primitive adherence condition. For now we will disregard the connectivity condition and postpone the discussion of enforcing connectivity to Sec. 5.

We define the cost E of $\mathcal{S}$ by measuring the surface area

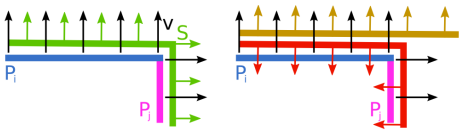

Figure 3: *The effect of different configurations on the energy term* $-\int_S \mathbf{H}(\langle n|v\rangle)dA$ *(see text).*

E_a of $\mathcal{S}$. To reward primitive adherence we do not take into account the area E_p of those surface parts that coincide with a primitive. A third term E_c avoids ambiguities in the solution and enforces the approximation of the original surface $\mathcal{S}^0$. Thus, denoting the surface normal of $\mathcal{S}$ by n, the functional responsible for primitive adherence that we want to minimize can be written as

$$\begin{aligned} E(\mathcal{S}) &= E_a(\mathcal{S}) - E_p(\mathcal{S}) + E_c(\mathcal{S}) \\ &= \int_{\mathcal{S}} dA - \int_{\mathcal{S}} \mathbf{H}(\langle n|v\rangle)dA + E_c(\mathcal{S}) \end{aligned} \tag{1}$$

where $v : \mathbb{R}^3 \rightarrow \mathbb{R}^3$ is a vector field derived from the shape primitives' normals and $\mathbf{H}$ denotes the Heaviside function

$$\mathbf{H}(x) := \begin{cases} 1 & x > 0 \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

Since the first term E_a in Eq. (1) is simply the surface area, minimizing E in the absence of, or far away from, primitive shapes results in a surface of minimal area.

Concerning the vector field v in the second term E_p, we compute for each primitive P_i the normal field $n_{P_i} : \mathbb{R}^3 \rightarrow \mathbb{R}^3$ which vanishes everywhere except on the surface of P_i. We define v as the sum over the normal fields n_{P_i} of all primitives. The idea behind this definition is illustrated in the left of Fig. 3. As shown in the figure, the vector field v (shown in black) is non-zero only on the shape primitives. Moreover, the term $\mathbf{H}(\langle n|v\rangle)$ evaluates to 1 iff the orientation of the surface normal field n (shown in green) and v coincide. For the configuration shown on the left of the figure both criteria are met. In this case, the second term cancels the area term resulting in zero cost. The sum of the terms increases if the surface does not follow the primitive shapes (yellow on the right) or if surface normal and vector field are not consistently oriented (red surface). Thus, minimizing the second term in the above energy aligns the surface with primitive shapes and enforces the correct orientation of $\mathcal{S}$.

Even though the first two terms of the cost function enforce adherence to the primitive shapes P_i, there are usually multiple minima, including the trivial empty surface. Therefore, the third term E_c of the function E is used to impose additional inside and outside constraints given by two sets $C_{in} \subset \mathbb{R}^3$ and $C_{out} \subset \mathbb{R}^3$. As a closed surface $\mathcal{S}$ divides the space into a well defined inside $\mathcal{S}_{in}$ and outside $\mathcal{S}_{out}$ (see Fig. 4(b)) and we require $C_{\{in,out\}}$ to be contained in the inside/outside $\mathcal{S}_{\{in,out\}}$ respectively. The actual choice of the sets $C_{\{in,out\}}$ is derived from the original surface $\mathcal{S}^0$ and detailed in Sec. 4.2. The third term in the cost function which integrates over all violated constraints is given by

$$E_c(\mathcal{S}) = \int_{C_{in} \setminus \mathcal{S}_{in}} \lambda dV + \int_{C_{out} \setminus \mathcal{S}_{out}} \lambda dV \tag{3}$$

where λ is a constant that must be chosen sufficiently large to avoid constraint violations, i.e. larger than the area of the reconstructed surface.

4.1. Discrete global minimization

While the cost function E enables an adequate formalization of the surface completion problem as sketched in Sec. 3, the multitude and structure of its local minima hinders an efficient global minimization by variational methods. To enable an efficient global optimization, we pursue an approach similar to Kolmogorov and Boykov [KB05] and formulate the surface completion problem defined in the previous section as a cut on a discrete volumetric graph. Fast graph-cut methods can then be used to compute a globally optimal solution in polynomial time.

We start by defining a volumetric graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where the set of vertices $\mathcal{V}$ consists of all voxels in a regular $3D$-grid that bounds the input surface S^0. In addition, $\mathcal{V}$ contains a special source vertex s and a sink vertex t. The edges in $\mathcal{E}$ connect each grid node v_{ijk} to its 26 neighbors in the grid. The definition of the volumetric graph is illustrated in Fig. 4 (a) for a $2D$ example.

On the edges of this graph we define a capacity function $\hat{E}$, that assigns each directed edge a cost. This function $\hat{E}$ can be regarded as a discrete counterpart to the continuous cost function E in the previous section. Given the graph $\mathcal{G}$ and the discrete cost function $\hat{E}$, an optimal partition of the vertices into two sets S_{in} and S_{out} - the cut - is computed by minimizing the costs of all edges from S_{in} to S_{out} subject to the constraints $s \in S_{in}$ and $t \in S_{out}$. With a suitable choice of the cost function $\hat{E}$, the discrete solution to the surface completion problem is then implicitly defined by the cut (S_{in}, S_{out}) (see Fig. 4 (b)).

Corresponding to the three terms of the continuous cost function E, we define the discrete edge costs assignment $\hat{E}$ as the sum of three functions

$$\hat{E} = \hat{E}_a - \hat{E}_p + \hat{E}_c \tag{4}$$

that constitute discrete measures of surface area, primitive adherence and constraint violations respectively. For the area costs $\hat{E}_a$ we resort to the weighting scheme given in [BK03] which provably converges to the continuous area term for a growing number of grid neighbors connected to each voxel.

To define a discrete analog of the second term in Eq. (1), for each directed edge $e \in \mathcal{E}$ the set of intersecting primitives is computed. If the orientation of any intersecting primitive P_i is consistent with the direction of e, we set $\hat{E}_p(e)$ to $\hat{E}_a(e)$ so that it cancels the area term's contribution (see Fig. 4(b)).

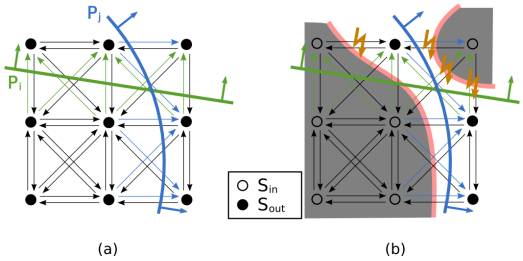

Figure 4: *(a) Graph construction and cost assignment: The colored edges intersect a primitive and match its orientation. On these edges $\hat{E}_p$ is set to cancel their area costs. (b) High costs result, if a cut does not follow primitives or fails to match their orientation (flashes mark edges with high costs).*

Formally, we set

$$\hat{E}_p(e) := \begin{cases} \hat{E}_a(e) & \text{a } P_i \text{ intersects } e \text{ and } \langle n_{P_i} | e \rangle > 0 \\ 0 & \text{otherwise} \end{cases}$$

where n_{P_i} denotes the normal of P_i at the intersection point. This choice of $\hat{E}_p$ does indeed mimic the behavior of the second term in the continuous formulation (1) in the following sense: For a cut (S_{in}, S_{out}) as shown in Figure 4 high cost results at edges, that are not intersected by any primitive or if the orientation of primitive and cut does not match. Therefore, a cut minimizing $\hat{E}_p(e)$ adheres to shape primitives and the orientation of the resulting surface S matches that of the followed primitives.

Just as in the continuous case we need to add a third term $\hat{E}_c$ to enforce certain inside and outside constraints derived from the input surface S^0. In the discrete setting, we assume that C_{in} and C_{out} are given as sets of vertices corresponding to voxels in the inside and outside respectively. In analogy to the continuous case $\hat{E}_c$ should be large if either $C_{in} \subset S_{in}$ or $C_{out} \subset S_{out}$ is violated. As by definition of the graph cut we have $s \in S_{in}$ and $t \in S_{out}$ it is sufficient to add edges with high costs connecting s to vertices in C_{in} and likewise edged connecting vertices in C_{out} to t. More precisely, the cost function $\hat{E}_c$ is defined to zero on all but these extra edges to which it assigns the high constant cost λ (in practice the maximal representable value of the employed data type is a viable choice). A violation of e.g. an inside constraint $v \in C_{in}$ will therefore result in a cut through this extra edge and thus in high overall costs. In the following section we will discuss the actual choice of the sets C_{in} and C_{out} .

The above discrete formulation of surface completion is closely related to the cut metric of Kolmogorov and Boykov [KB05] who proved for the $2D$ case that all functionals representable by cut metrics are of the form in Eq. (1). Although the construction used in the proof generalizes to $3D$, there are some ambiguities in the choice of the edge cost assignment. Therefore the choice of an optimal edge assignment is not clear, in particular if sharp features at primitive intersections are to be preserved. The discrete formulation presented here is directly adapted to primitive shapes and thus circumvents this problem. Moreover, it establishes an explicit edge to shape correspondence which is crucial for enforcing the connectivity constraint described in Sec. 5.

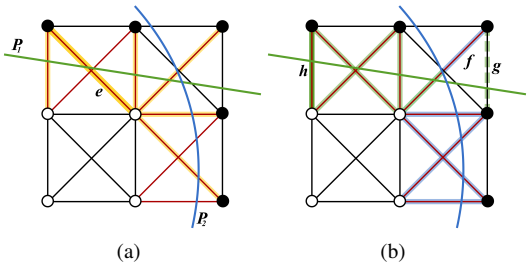

Figure 5: *Edge connectivity and edge/primitive correspondence. Cut-edges are depicted in red. (a) The cut-edges connected to edge e are highlighted. (b) Cut-edge f is intersected by both primitives. If h is part of the original support S_1^0 of P_1 and the cost of g had previously been increased, it is now reset because it is connected to h, see Sec. 5.*

In general, a drawback of the volumetric approach is the huge memory demand of the 3D-grid graph. However, recently Lempitsky and Boykov [LB07] proposed a hierarchical graph-cut approach that guarantees global optimality while operating only on a banded subset of the volume. Their technique is also applicable in our setting and we use it for completion of large models.

4.2. Placement of inside and outside constraints

So far, the original support of primitives has not been considered in the definition of $\hat{E}$. We therefore propose a simple scheme to derive constraints C_{in} and C_{out} from the support sets S_i of all primitives. In case of a single oriented primitive P_i, we start by computing the set of all directed edges E_{S_i}, that intersect the support S_i of P_i. We only consider those edges that run from the inside of P_i to the outside, i.e. that match the orientation of P_i. Then a natural choice for the inside constraints C_{in} consists of the set of voxels from which an edge in E_{S_i} emanates. Appropriate outside constraints can be defined in a similar fashion.

In the general case, defining inside and outside constraints in this way can lead to contradictions at primitive intersections. We therefore add a vertex v with an emanating edge in E_{S_i} to C_{in} only if it is not contained in E_{S_j} for any other primitive P_j.

5. Primitive connectivity

The above algorithm does not enforce connectivity of the areas $S_i \subset \mathcal{S}$ corresponding to primitive P_i. However, as outlined in Sec. 3 and illustrated in Figures 2 and 1 this is often crucial. From the shape detection we do have a connected region $S_i^0 \in \mathcal{S}^0$ for each primitive P_i where it is supported by the original surface. Thus we require the following: The

area S_i corresponding to P_i in the reconstructed surface $\mathcal{S}$ is a connected superset of S_i^0.

Unfortunately this connectivity constraint cannot be formulated as a graph-cut problem, see e.g. the work of Kolmogorov and Zabin [KZ04] who give a good characterization of functions minimizable by graph-cuts. Instead, in order to find the global optimum an involved combinatorial optimization is necessary which quickly becomes infeasible with a growing number of shape primitives. We therefore suggest a less complex but nonetheless effective iterative greedy optimization scheme.

This iterative optimization makes use of the graph structure described in the previous section. After the graph-cut, the reconstructed surface is implicitly defined by the set of cut-edges, i.e. all halfedges running from S_{in} to S_{out}. Since our graph construction provides an explicit edge/primitive correspondence, we can use the cut-edges to determine connectivity on the surface and to efficiently identify correspondence between surface parts and primitives. In fact we treat each cut-edge as a representative for a small local patch in the reconstructed surface. For each cut-edge we identify the set of intersecting shape primitives in order to establish surface/primitive correspondence, i.e. if a cut-edge is intersected by primitive P_i the respective surface patch is part of S_i, see Fig. 5 (b). We say two edges are connected (and therefore also their respective surface patches) if they share a common node in the graph, see Fig. 5 (a). With these definitions the connectivity condition can be restated as follows: The cut-edges corresponding to primitive P_i must form a connected superset of the cut-edges intersected by S_i^0. We call any cut-edge corresponding to P_i that does not belong to this connected set a violating edge.

The basic idea of our connectivity enforcing algorithm is in each iteration to greedily set the cost of all violating cut-edges equal to the cost given by $\hat{E}_a$. Then the graph-cut is re-run with the increased cost on the violating edges. This means that the newly reconstructed surface will avoid the now costly edges and prefer cheaper edges corresponding to other primitives. In order to remedy some of the greedy decisions of earlier iterations, the cost of graph edges whose cost had previously increased but are now connected to non-violating cut-edges is reset in each iteration, see Fig. 5 (b). The procedure is repeated until the set of cut-edges does not change between iterations.

In summary, the iterative optimization consists of the following main steps: (1) Compute the reconstruction using the graph-cut algorithm (2) Detect violations of the connectivity constraint by inspection of the cut edges. Since S_i must contain S_i^0 the non-violating edges can be identified by a graph traversal visiting all cut-edges connected to S_i^0. (3) Increase cost of violating edges in the graph and reset cost of revalidated edges (4) Reiterate until the set of cut-edges converges. Although convergence cannot be guaranteed in general, we found that in practice all of our test cases converged in less than fifteen iterations.

6. Reconstruction of detail

In the previous sections we have presented an algorithm that uses guidance from primitives for hole-filling and gives an idealized reconstruction of the input surface that adheres to the primitives everywhere. Such a reconstruction is often useful if the model is to be used in CAD systems or if the input data was corrupted by many outliers and noise. However, high quality scans may contain valuable detail geometry, e.g. engravings, or models may contain parts that cannot by approximated by primitives at all, e.g. a small statue mounted on a wall. Depending on the application it may be desirable to recover these features as well and in this section we outline how we can seamlessly combine our primitive based reconstruction with the one given by Lempitsky and Boykov [LB07] to this end.

Lempitsky and Boykov define a smooth vector field u from a set of input points with oriented normals (please see the original paper for details). In our setting we can either use the entire point-cloud for computation of u or, if we are certain that the input model is in principle well represented by primitives, we can use only the points associated to the primitives and ignore any remaining points as noise and outliers. Thus, given the vector field u the detail preserving reconstruction functional is stated as follows:

$$E(\mathcal{S}) = E_a(\mathcal{S}) - E_p(\mathcal{S}) - \lambda E_u(\mathcal{S}) \tag{5}$$

where E_a and E_p are as in Eq. (1) and

$$E_u(\mathcal{S}) = \int_{\mathcal{S}} \langle n|u \rangle \, dA \tag{6}$$

In contrast to E_p we can apply the divergence theorem to Eq. (6) because u is smooth, such that

$$E_u(\mathcal{S}) = \int_{\mathcal{S}_{in}} div(u) dV \tag{7}$$

Note that we no longer include the $E_c(\mathcal{S})$ from Eq. (1) since adherence to the original surface is now ensured by the term $E_u(\mathcal{S})$. Except for the missing in/out constraints, the graph construction of Sec. 4.1 remains unchanged. However additionally, $div(u)$ is evaluated on the grid nodes and depending on the sign s- or t-links are added with a cost proportional to the divergence, please refer to [KB05] for details.

7. Surface extraction

After application of the algorithms of the previous sections, all voxels have been classified as either inside or outside and the final task is to extract the resulting surface mesh. Of course it is possible to extract a mesh using the standard marching cubes algorithm [LC87], but this would not faithfully recover the shape primitives and does not capture any sharp features at the intersections. Instead, we can exploit the

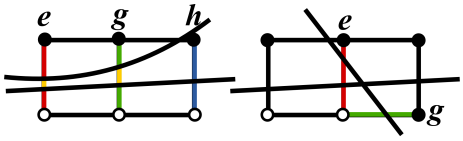

Figure 6: *Illustration of the different cases for the smoothness term V in eq. (8). Left: $V_{e,g}$ is of type (2) and will give the distance between the primitives along edge e, while $V_{g,h}$ is of type (3) and gives the distance between the intersections of g. Right: $V_{e,g}$ is of type (3) and evaluates to zero because the primitives intersect within the cube face.*

tight coupling between cut-edges and primitives discussed in the previous sections to employ the extended marching cubes algorithm of Kobbelt et al. [KBSS01] for recovery of shape primitives as well as sharp features.

In order to recover sharp features, the extended marching cubes requires for each cube edge intersected by the surface not only the point of intersection but also the surface normal at that position. This information is easily obtained in our setting: If a cut-edge is labeled with a shape primitive P_i then the point of intersection as well as the normal are computed using P_i. If a cut-edge is not associated with any primitive then the midpoint of the edge is taken as intersection point and the normal is ignored.

However, it can happen that a single cut-edge is labeled with two or more primitives, see e.g. Fig. 5. This is usually the case near primitive intersections or locations where primitives come close to each other, see also Fig. 1. These situations need to be disambiguated in order to achieve high quality results. For the disambiguation only the cut-edges that will actually be inspected by the extended marching cubes algorithm need to be considered, i.e. only the axis aligned edges of the cubes and no diagonals. We will denote this set of cube edges by $\mathcal{E}_c$. We will also need a neighborhood relation $\mathcal{N} \subset \mathcal{E}_c \times \mathcal{E}_c$ between the edges in $\mathcal{E}_c$ which is different from the one defined in Sec. 5 as this time the diagonal edges are missing. In the following, two edges in $\mathcal{E}_c$ are said to be neighbors if they are adjacent to a common cube face.

7.1. Consistent edge labeling

Our algorithm for derivation of consistent edge labels is based on the following observation: Neighboring edges should have different primitive labels only if the two primitives come very close or intersect in the space between the edges. Otherwise continuation of a primitive should be preferred. Such label dependent neighbor relations lead to a well studied class of energy functions of the following type[†]

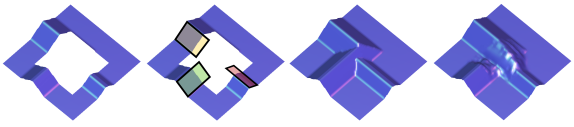

Figure 7: *(a) Height field with missing area. (b) Planar primitives are detected on the ridges (c) Our result (d) Completion result obtained by [JT04]*

(see [SZS*08] for a survey):

$$C(\mathbf{f}) = \sum_{e \in \mathcal{E}_c} D_e(f_e) + \sum_{e,g \in \mathcal{N}} V_{e,g}(f_e, f_g) \qquad (8)$$

where $\mathcal{E}_c$ is a set of sites, in our case the set of edges introduced above. $\mathbf{f}$ is a labeling of the sites, i.e. a mapping from $\mathcal{E}_c$ to $\mathcal{P}$. Thus, $\mathbf{f}$ assigns each edge a primitive from $\mathcal{P}$. $D_e(f_e)$ is a data cost term that specifies the cost of assigning label f_e to edge e and is used to restrict the set of possible labels for the edge e. $V_{e,g}$ measures the cost of assigning the labels f_e, f_g to the adjacent edges e, g and is responsible for ensuring the above conditions on shape changes between edges. The term $D_e(i)$ vanishes if e is among the set of edges associated to primitive P_i, otherwise we set $D_e(i) = \infty$. $V_{e,g}(i,j)$ vanishes if $i = j$, otherwise we distinguish the following cases (see also Fig. 6):

(1) One of the labels is invalid for the respective edge, i.e. $e \notin S_i$ or $g \notin S_j$. In this situation the cost of $V_{e,g}$ is meaningless as the data term D_e will be infinite and we simply set $V_{e,g}$ to zero.

(2) Both labels are valid on both edges. If shapes are approximately tangent, the cost should be lowest at the point where both primitives are closest to each other. If the shapes intersect within the cube we let $V_{e,g}$ vanish, otherwise we let

$$V_{e,g}(i,j) = \min(\|I_e(i) - I_e(j)\|, \|I_g(i) - I_g(j)\|) \qquad (9)$$

where $I_e(i)$ denotes the point of intersection of e and P_i. See Fig. 6 on the left.

(3) One edge is valid for both labels while the other can only be assigned one of the labels. Here the same arguments hold as in case (2) and $V_{e,g}$ vanishes if the primitives intersect, otherwise we let $V_{e,g}(i,j) = \|I_h(i) - I_h(j)\|$ where $h \in \{e,g\}$ is the edge intersected by both primitives. See Fig. 6 on the right.

We use the graph-cut based algorithm of Boykov et al. [BVZ01] for optimization of Eq. (8). While this method is quite efficient, performance can be increased by optimizing connected component of ambiguous edges separately.

8. Height-fields

Our algorithm can directly be applied to geometry derived from height-fields, but the generated completion might not be representable as the graph of a function over the ground.

[†] Interestingly, the discretization in Sec. 4.1 can also be interpreted as a realization of such a functional, but the motivation from a geometric point of view is much more intuitive. See [KB05] and [KZ04] for an in-depth discussion of the relationship.

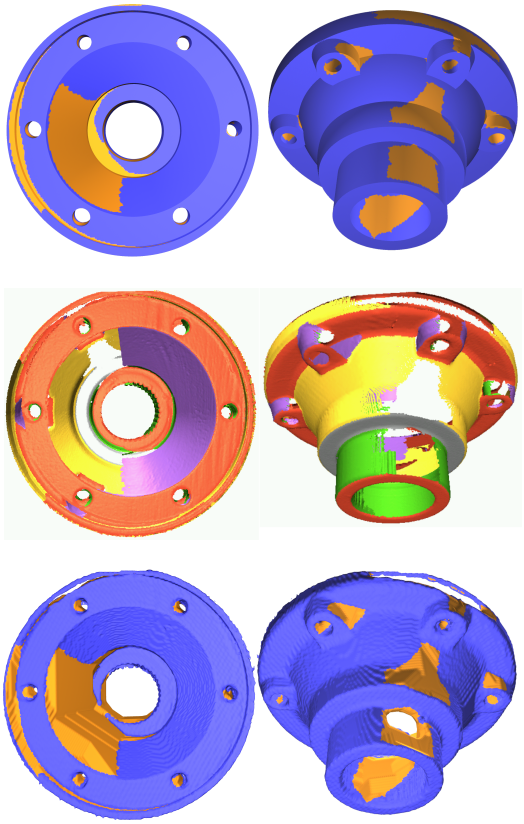

Figure 8: *Completion of the carter model. Orange color signifies completed surface parts. Top row: Our final result with sharp features. Middle row: The input point-cloud with holes. Primitive types are colored as follows: plane/red, sphere/yellow, cylinder/green, cone/purple, grey/torus. Bottom row: The result with the algorithm of [LB07].*

To ensure this we extend the graph-cut construction given in Sec. 4.1. We only need to prevent cuts that produce switches from outside to inside in direction of the z-axis. To this end it suffices to add maximal weight to each directed edge $(v_{i,j,h}, v_{i,j,h-1})$ connecting a node with its neighbor below. This follows from the graph rules given in [KZ04], a similar construction was also used by Rubinstein et al. [RSA08].

For completion of height-fields we also allow the detection of primitives on depth discontinuities by insertion of additional point samples on the surfaces implicitly spanned by these discontinuities. Shapes on depth discontinuities will allow propagation of the discontinuities into the empty region during hole-filling (see e.g. Fig. 10).

9. Experimental results

To illustrate the ability of our method to handle complex intersections of primitives we show a synthetic example of an incomplete height field in Fig. 7. The result of the range-image completion method proposed by Jia and Tang [JT04]

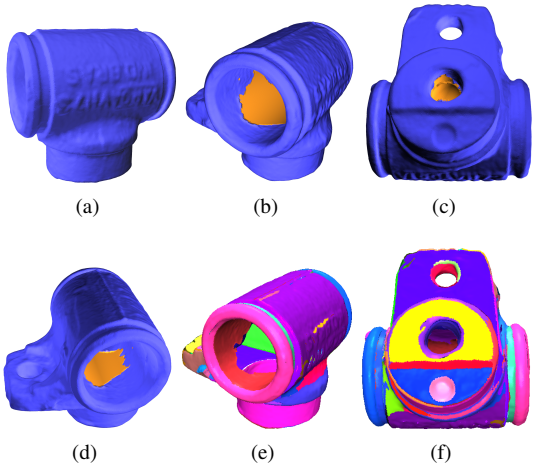

Figure 9: *Completion of the master cylinder. (a)-(d): The final result using the detail reconstruction of Sec. 6 (e)-(f): Input point-cloud. Areas corresponding to different primitives rendered in random colors.*

is contrasted to the one of our approach. Our method produces a plausible result even in this complex case, while the other algorithm fails to reconstruct any sharp features and does not prolong the planar surface parts of the height-field.

Another synthetic example is given in Fig. 1. We manually removed several parts of the fandisk point-cloud and applied our algorithm. In this Figure, the effects of both the connectivity enforcing as well as the consistent edge labeling can be observed. In (c) the reconstruction result without connectivity enforcement is depicted. On the right side of the model a part of the surface has been cut off by a disconnected primitive. Our iterative connectivity enforcement algorithm successfully increases the cost of the violating surface parts and arrives at the final solution shown in (b). A part of the fandisk where a cylinder and a plane are almost tangential is shown in (d). Without the consistent edge labeling the transition between the primitives appears bumpy in the reconstruction ((d) left). This is because the edges considered in the marching cubes algorithm are associated arbitrarily with one of the two shapes. Our edge labeling method on the other hand finds a smooth transition ((d) right).

Results for a real-world case are shown in Fig. 8. Nine range scans of the carter model were registered into an incomplete point-cloud. 38 shape primitives were detected on this point-cloud (see second row). In the final reconstruction shown in the first row all holes were successfully filled using the shape primitives. Sharp features were faithfully recovered on the entire model. For comparison we also show in the bottom row the result obtained with our implementation of the algorithm of Lempitsky and Boykov [LB07], which fills holes with minimal surfaces but does not use any shape primitive guidance. While the algorithm reconstructs a watertight surface, several holes are closed in an undesired

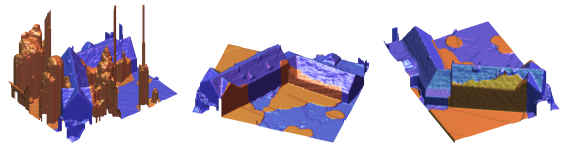

Figure 10: *Removal of defective and undesired data from an aerial heightfield (highlighted in orange). Our algorithm infers missing features such as dormers or walls by propagating surrounding structure represented as primitive shapes.*

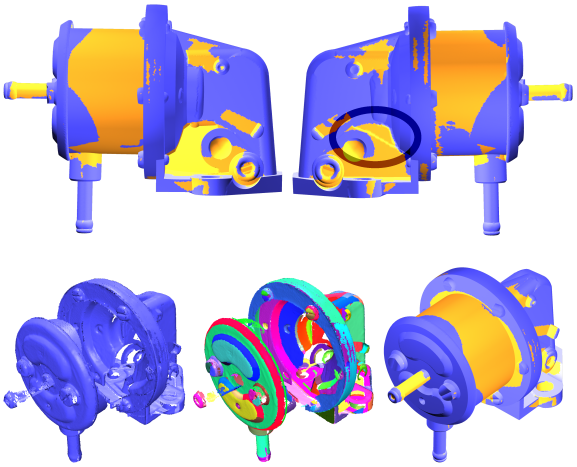

Figure 11: *Reconstruction of the oil-pump from 9 scans. Top: Final result. Bottom from left to right: Input point-cloud. Input point-cloud colored by primitives. Final result.*

manner. Clearly, our method strongly benefits from the additional guidance provided by the primitives and is therefore able to find the correct reconstruction.

To demonstrate a reconstruction using the detail preserving variant of our method given in Sec. 6 we have applied our algorithm to the master cylinder model depicted in Fig. 9. This model contains many elements that cannot, or only very roughly, be approximated by shape primitives, e.g. the engraved writing. However the holes contained in the input model are well suited for completion with primitives. As can be seen in the images, small detail is well preserved on the original surface while at the same time the holes are plausibly filled using the primitive information.

A case of height field completion is given in Fig. 10. We manually removed occluding vegetation and defective elevation data from this aerial reconstruction. The missing geometry of the houses and floor were successfully reconstructed. The roof gable and the dormer were correctly inferred from the surrounding primitives.

In Fig. 11 we show a reconstruction result from 9 scans of the oil-pump model. The point-cloud has several large holes and is decomposed into 171 primitives. With the guidance

model	T_s	$\|\mathcal{P}\|$	T_r	$\|V\|$
fandisk	0.27	22	1:11	193x113x199
carter	5.4	38	1:54	181x199x167
master cylinder	5.9	60	5:40	389x349x398
house	6.5	51	8:13	509x381x201
oil-pump	18.6	171	9:29	317x265x265

Table 1: *Timings for shape detection in seconds (T_s), number of detected primitives ($|\mathcal{P}|$), timings for reconstruction in minutes (T_r), virtual size of volume ($|V|$)*

of the primitives our algorithm finds a very plausible completion despite the significant amount of missing geometry. However, some limitations of our approach become apparent as well. In the marked area of the top right image our algorithm cannot follow the cylindrical shape of the protruding element because no primitive could be detected in the region. On the other side of the model this is not the case and the protrusion is handled correctly. Since our method has no concept of symmetry it cannot deduce the correct reconstruction from the information on the opposite side. Also tiny artifacts may occur at primitive boundaries if primitives are slightly misaligned and resulting gaps are filled with minimal surfaces (see e.g. backside of oil-pump in accompanying video). This could be alleviated by applying a refitting algorithm as suggested in [JKS08].

Finally, we give some timings of our method in Tab. 1. We also give the size of the virtual grid used for the graph-cut. Although the worst-case complexity is cubic with respect to the grid resolution, the actual amount of allocated nodes is however far less than the full grid since we use only a banded subset of the volume. On our examples we observed band sizes between only 10% and 15% of the actual grid size and decreasing fractions for higher resolutions. In general the choice of grid resolution is not critical as long as voxels are small enough to separate between inside and outside areas of the volume. The precision of the reconstructed surface is hardly affected by the voxel resolution since mesh vertices are positioned exactly on the primitives. Primitive intersections are faithfully recovered due to the extended marching cubes algorithm.

10. Conclusion

We have proposed a novel method for reconstruction of 3D-models that is guided by a set of primitive shapes and uses this guidance to complete missing parts of the input geometry. We have shown that our algorithm performs well on various involved examples with many holes on which previous methods fail to deliver correct results. While we currently employ only a small set of implicit surfaces, which is nonetheless sufficient in many cases, our method could also be extended to more complex primitives such as e.g. NURBS. However, regardless of the nature of the primitives, our algorithm is always limited to reconstructions deducible from the set of primitives that have been detected in the

vicinity of the holes and in the future we plan to research a combination of our approach with methods based on self-similarity such as symmetry detection or texture synthesis, which should enable the completion of a whole new class of challenging scenarios.

Acknowledgements

We would like to thank the AIM@SHAPE project for providing the carter, master cylinder and oil pump models. The fandisk model is courtesy of Hughes Hoppe. Many thanks to Leo Jia for providing the comparison data and to Freek Stulp for his helpfulness and discussions.

References

[BK03] BOYKOV Y., KOLMOGOROV V.: Computing geodesics and minimal surfaces via graph cuts. In *ICCV* (2003), pp. 26–33.

[BMV01] BENKÖ P., MARTIN R. R., VÁRADY T.: Algorithms for reverse engineering boundary representation models. *Computer-Aided Design 33*, 11 (2001), 839–851.

[BPK05] BISCHOFF S., PAVIC D., KOBBELT L.: Automatic restoration of polygon models. *ACM Trans. Graph. 24*, 4 (2005), 1332–1352.

[BSCB00] BERTALMIO M., SAPIRO G., CASELLES V., BALLESTER C.: Image inpainting. In *SIGGRAPH* (2000), pp. 417–424.

[BVZ01] BOYKOV Y., VEKSLER O., ZABIH R.: Fast approximate energy minimization via graph cuts. *IEEE Trans. Pattern Anal. Mach. Intell. 23*, 11 (2001), 1222–1239.

[CDD*04] CLARENZ U., DIEWALD U., DZIUK G., RUMPF M., RUSU R.: A finite element method for surface restoration with smooth boundary conditions. *CAGD* (2004), 427–445.

[CL96] CURLESS B., LEVOY M.: A volumetric method for building complex models from range images. In *SIGGRAPH* (1996), pp. 303–312.

[CLF02] CASTELLANI U., LIVATINO S., FISHER R. B.: Improving environment modelling by edge occlusion surface completion. *3DPVT* (2002), 672–675.

[CSAD04] COHEN-STEINER D., ALLIEZ P., DESBRUN M.: Variational shape approximation. *ACM Trans. on Graph. 23*, 3 (Aug. 2004), 905–914.

[DCOY03] DRORI I., COHEN-OR D., YESHURUN H.: Fragment-based image completion. *ACM Trans. Graph. 22*, 3 (2003), 303–312.

[DMGL02] DAVIS J., MARSCHNER S. R., GARR M., LEVOY M.: Filling holes in complex surfaces using volumetric diffusion. In *3DPVT* (2002), pp. 428–861.

[GSH*07] GAL R., SHAMIR A., HASSNER T., PAULY M., COHEN-OR D.: Surface reconstruction using local shape priors. In *Eurographics SGP* (2007), pp. 253–262.

[HDD*94] HOPPE H., DEROSE T., DUCHAMP T., HALSTEAD M., JIN H., MCDONALD J., SCHWEITZER J., STUETZLE W.: Piecewise smooth surface reconstruction. In *SIGGRAPH* (1994), ACM, pp. 295–302.

[HK06] HORNUNG A., KOBBELT L.: Robust reconstruction of watertight 3D models from non-uniformly sampled point clouds without normal information. In *Eurographics SGP* (2006), pp. 41–50.

[JKS08] JENKE P., KRÜCKEBERG B., STRASSER W.: Surface reconstruction from fitted shape primitives. In *Vision, Modeling and Visualization (VMV '08)* (2008).

[JT04] JIA J., TANG C.-K.: Inference of segmented color and texture description by tensor voting. *IEEE Trans. Pattern Anal. Mach. Intell. 26*, 6 (2004), 771–786.

[Ju04] JU T.: Robust repair of polygonal models. *ACM Trans. Graph. 23*, 3 (2004), 888–895.

[JWB*06] JENKE P., WAND M., BOKELOH M., SCHILLING A., STRASSER W.: Bayesian point cloud reconstruction. *Comput. Graph. Forum 25*, 3 (2006), 379–388.

[KB05] KOLMOGOROV V., BOYKOV Y.: What metrics can be approximated by geo-cuts, or global optimization of length/area and flux. In *ICCV* (2005), pp. 564–571.

[KBH06] KAZHDAN M., BOLITHO M., HOPPE H.: Poisson surface reconstruction. In *Eurographics SGP* (2006), pp. 61–70.

[KBSS01] KOBBELT L. P., BOTSCH M., SCHWANECKE U., SEIDEL H.-P.: Feature sensitive surface extraction from volume data. In *SIGGRAPH* (2001), pp. 57–66.

[KZ04] KOLMOGOROV V., ZABIN R.: What energy functions can be minimized via graph cuts? *IEEE Trans. on Pattern Anal. and Mach. Intell. 26*, 2 (2004), 147–159.

[LB07] LEMPITSKY V. S., BOYKOV Y.: Global optimization for shape fitting. In *CVPR* (2007), pp. 1–8.

[LC87] LORENSEN W. E., CLINE H. E.: Marching cubes: A high resolution 3d surface construction algorithm. *SIGGRAPH 21*, 4 (1987), 163–169.

[Lie03] LIEPA P.: Filling holes in meshes. In *Eurographics SGP* (2003), pp. 200–205.

[PMW*08] PAULY M., MITRA N. J., WALLNER J., POTTMANN H., GUIBAS L.: Discovering structural regularity in 3D geometry. *ACM Trans. on Graph. 27*, 3 (2008), 1–11.

[PR05] PODOLAK J., RUSINKIEWICZ S.: Atomic Volumes for Mesh Completion. In *Eurographics SGP* (2005), pp. 33–41.

[RSA08] RUBINSTEIN M., SHAMIR A., AVIDAN S.: Improved seam carving for video retargeting. *ACM Trans. Graph. 27*, 3 (2008), 1–9.

[SACO04] SHARF A., ALEXA M., COHEN-OR D.: Context-based surface completion. *ACM Trans. on Graph. 23*, 3 (Aug. 2004), 878–887.

[SDF01] STULP F., DELL'ACQUA F., FISHER R.: Reconstruction of surfaces behind occlusions in range images. In *3D Digital Imaging and Modeling* (2001), pp. 232–239.

[SWK07] SCHNABEL R., WAHL R., KLEIN R.: Efficient ransac for point-cloud shape detection. *Computer Graphics Forum 26*, 2 (2007), 214–226.

[SZS*08] SZELISKI R., ZABIH R., SCHARSTEIN D., VEKSLER O., KOLMOGOROV V., AGARWALA A., TAPPEN M., ROTHER C.: A comparative study of energy minimization methods for markov random fields with smoothness-based priors. *IEEE Trans. Pattern Anal. Mach. Intell. 30*, 6 (2008), 1068–1080.

[VCBS03] VERDERA J., CASELLES V., BERTALMÍO M., SAPIRO G.: Inpainting surface holes. In *ICIP (2)* (2003), pp. 903–906.

[WK05] WU J., KOBBELT L.: Structure recovery via hybrid variational surface approximation. *Computer Graphics Forum 24*, 3 (2005), 277–284.

[WO02] WANG J., OLIVEIRA M. M.: Improved scene reconstruction from range images. *Computer Graphics Forum 21*, 3 (Sept. 2002), 521–530.