computer graphics & vision

Inffeldgasse 16/2, A-8010 Graz
www.icg.tu-graz.ac.at

TUG

Robust Surface Reconstruction from Noisy Point Clouds using Graph Cuts

Christian Mostegel

Inst. for Computer Graphics and Vision
Graz University of Technology, Austria

Seminar/Project Report
Supervisor: Markus Rumpler
Graz, June 28, 2012

---

contact: Christian Mostegel morair@sbox.tugraz.at

Abstract

This work is concerned with the implementation and evaluation of a surface reconstruction approach, which is highly robust against outliers. The chosen approach has a range of appealing properties. First of all, it manages to incorporate a keypoint-based visibility constraint, which leads to a very high resilience to outliers. Secondly, it is based on a Delaunay triangulation, which makes the approach independent of the scene scale. Thirdly, the task is reduced to a binary labeling problem, which enables the efficient global optimization via graph cuts. The combination of these properties leads to a watertight and intersection-free surface mesh as output.
The main contribution of this work, aside from the highly adaptive implementation for scientific purposes, is an intensive evaluation of the chosen approach. A range of different combinations of visibility constraints and regularization terms is evaluated. Furthermore, the approach is compared to the very popular and robust Poisson reconstruction in the presence of outliers as well as Gaussian noise. The experiments are not limited to a visual comparison, but also evaluate the output accuracy on an outdoor dataset with a high-resolution ground truth mesh.
Furthermore, this work presents a novel way to estimate the 3D Gaussian scene noise from the median reprojection error. It is shown how this knowledge can be used to improve the result in a posterior smoothing step, when a high level of Gaussian scene noise is present.
Finally, this work proposes a simple regularization term, which is based on the minimum description length. The evaluation shows that this term, albeit its inexpensiveness, outperforms other popular regularization terms.

Keywords: *Surface reconstruction, 3D Delaunay triangulation, graph cuts, 3D scene noise estimation, pc2mesh*

Contents

1 Introduction

The field of computer vision can be split into two schools as proposed by David Marr 1982 [29]; the reconstruction and the recognition school. Image-based rendering and modeling are placed in the center of the first school. The idea is to gain understanding of the world which was projected on the image plane, as well as the relation between the images, through reconstruction.

The most common approach is to use a pipeline of feature detection/matching, sparse reconstruction and bundle adjustment to robustly find the relation between the cameras and the scene. This approach also yields a sparse reconstruction via a point cloud. Depending on the structure and the texture of the scene the density of the points varies gravely. This makes it, even for humans, difficult to interpret the scene which is presented. Thus, a lot of recent research is concerned with the densification of such a sparse reconstruction.

On the one hand, this dense point cloud might be enough for some applications, such as photo tourism [35], where it is only important that humans are able to interpret the scene. On the other hand, a simple point cloud (even dense) is not well-suited for automated tasks such as reverse engineering, prototyping or autonomous 3D path planning for MAVs (micro aerial vehicles). Consequently, there is a great need for the recovery of object surfaces from calibrated multi-view stereo.

This work fills the gap between a point cloud and a watertight representation via a surface mesh. This is achieved with energy minimization via graph cuts based on a Delaunay triangulation of the point cloud.

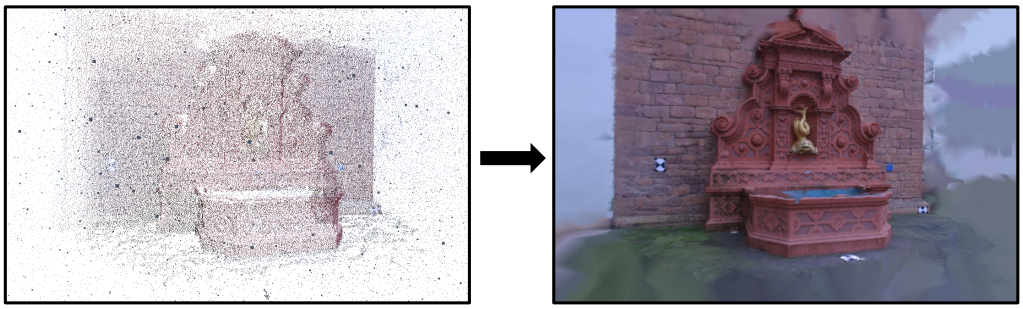


Figure 1: Transformation of a noisy point cloud with 100k inliers and 400k outliers to a watertight surface mesh.

The chosen approach is strongly based on the work of Labatut et al. [26, 27] for several reasons. Firstly, this approach is highly robust against outliers (see Figure 1) due to the strong feature of visibility which is incorporated in the energy formulation. Secondly, the memory usage of this approach does

not depend on the scale, but in practice, it is linear to the number of points in the point cloud. Thirdly, the approach achieves state-of-the-art performance on the dense multi-view stereo benchmark by Strecha et al. [37].

These nice properties, especially the outlier resilience, let aside the state-of-the-art performance, motivated us to implement and investigate the approaches of Labtut et al. [26, 27] on the way to a multi-view-stereo pipeline with a high accuracy mesh as output.

The following section provides an overview of the related work and explains the advantages of the chosen approach in detail. Section 3 gives details about the chosen approach as well as the contribution of this work. The experimental section (4) is split into three main parts. Firstly, it demonstrates the adaptability to scenes of various scale and point density. Secondly, the quality of different regularization terms with respect to accuracy is investigated on the "fountain" dataset of Strecha et al. [37]. Thirdly, the robustness against outliers, as well as Gaussian noise, is analyzed on the same dataset and directly compared to the very popular Poisson reconstruction [25].

2 Related Work

This section is split into three main parts. First of all, it outlines different concepts of 3D surface extraction. Secondly, possible data structures related to the surface extraction are discussed and, finally, the most relevant 3D Delaunay methods for the implemented graph cut approach are reviewed.

2.1 Concepts

For the following comparison, one has to keep in mind that some SfM (structure from motion) step has already been completed and found the camera positions and parameters robustly. Thus, the next step is to create a 3D model only based on images.

A lot of approaches are based on the concept of the visual hull [28] to extract an initial object surface for further optimization [22, 17, 24, 36, 38]. This concept uses the silhouette of objects to carve them into 3D space. Therefore, those methods cannot be applied to scenes where the silhouette cannot be extracted, e.g. building facades. Note that, for this initialization a good foreground/background segmentation is crucial and, consequently, those methods are not well-suited for natural scenes.

Another approach is to compute depth maps based on the images [21, 23, 9, 15]. Those approaches come with several problems. If only one view is used, the model is very likely to be incomplete. If all views are used

this approach very easily reaches its limits in matter of computing time and memory usage. Furthermore, a robust merging step is needed to obtain a unique result, where most approaches have problems with accuracy or completeness.

Other approaches try to obtain a semi-dense representation to overcome those limits. One of the most popular approaches on this matter is the PMVS (patch-based multi-view stereo) approach of Furukawa and Ponce [18]. It tries to densify the point cloud directly in the 3D space and has no need to calculate the depth of every pixel value, which even might not be possible due to missing texture. Firstly, it estimates small rectangular patches on the object surface based on feature matches. Secondly, it tries to iteratively construct patches in the neighborhood of the already constructed patches in an expansion procedure. Refinement and outlier removal are contained in all steps. Consequently, the approach is very robust, but the computing time is also considerable.

A point cloud, even a dense one, is still an incomplete representation without any real notion of objectness. The next challenge is to estimate a surface from the given points.

A very popular approach is the Poisson surface reconstruction [25], whose popularity is also due to the availability in many tools such as MeshLab [42]. It tries to fit a surface based on the gradient field imposed by samples and their normals. Opposed to most other approaches, the surface is not restricted to samples/measurements only, but can and will create new vertices for the mesh. As this approach originates more from the computer graphics domain, there is no room to incorporate visibility terms. This property together with the fact, that this reconstruction always yields closed surfaces, leads to the problem of unwanted bubbling effects. Thus, this approach needs a post processing, such as, either manually cropping of the scene, or heuristics as proposed in [19]. This approach suggests to remove all triangles that contain edges which have a length greater than 6 times the average edge length. Note that, such heuristics can easily lead to a very fragmented surface depending on the scene. Another drawback of this reconstruction is that it is not edge preserving as it is biased towards smooth solutions. Furthermore, the detail of the reconstruction is not inferred from the sample points, but has to be predefined by the user.

A carving based on keypoints seems to be more promising. The idea is to carve the object based on measurements on the surface of the object. This approach does not punish thin structures and preserves discontinuities. The ProForma approach of Pan et al. [31] solely relies on this information. It focuses on on-line model acquisition and has already been ported to mobile phones [32]. It initializes the model with a Delaunay triangulation and

subsequently removes triangles that are disproved by the rays from measurements to the camera centers. The disadvantage of this approach is that the outcome is not a surface mesh, but more a triangle soup. This means that the objects are also filled with triangles. As no global optimization is used, this approach is not very robust against outliers and the solution very often contains artifacts of thin triangles that were not intersected by any ray.

The work of Labatut et al. [26], on the other hand, seems to overcome all the problems mentioned above. It incorporates the visibility information of measurements, does not rely on the often not available visual hull information and preserves edges and fine structures. Moreover, it is very robust against outliers as the model is globally optimized via graph cuts. This results in a watertight surface mesh based on the existing measurements.

Very high accuracy can be achieved with variational optimization, as in [1, 22, 33, 20, 39]. This kind of approaches suffers either from a high and scale-dependent memory usage and/or a lack of robustness against outliers. As it is an iterative optimization scheme, it can easily get stuck on a bad local minimum, if the convexity of the optimization cannot be guaranteed. A reliable initial guess, which is very close to the optimum solution, is needed to overcome this problem.

In the very recent work of Vu et al. [39] it was shown, that state-of-the-art reconstruction accuracy as well as resilience to outliers can be achieved, in combining the keypoint based approach of Labatut et al. [26] with a posterior variational refinement step. This can be realized, as the approach of Labatut et al. [26] robustly computes a mesh which is very close to the optimum solution. Consequently, the chance of getting stuck on a bad local minimum with the variational refinement is very slim. Note that, as the resulting mesh of Labatut et al. [26] is already a good solution, for many applications this result might already be accurate enough. For these and other reasons we decided to strongly base this work on the approach of Labatut et al. [26, 27] on the way to a robust high accuracy 3D surface reconstruction.

2.2 Representations

Another important issue is the matter of how to represent the found object. Several approaches use a volumetric approach [24, 36, 38], where each voxel is assigned to be *object* or *empty*. This can be seen as the 3D analogue to the occupancy grid or bit map in 2D. This representation has the advantage that every point in space is either *empty* or belongs to an object. Hence, 2D algorithms such as the Monte Carlo localization for robots [11] can be easily adapted to 3D space. However, the biggest drawback of this representation is the mere memory consumption, which makes it impracticable for large scenes,

such as most outdoor scenes. Hornung and Kobbelt [24] try to overcome this problem by using a hierarchical voxel representation. At first, they use a coarse grid and then iteratively refine the grid around the estimated surface. Similarly, Sinha et al. [34] propose an adaptive tetrahedral mesh. This reduces the overall memory consumption while keeping the accuracy high. Nevertheless, it does not change the fact that the memory consumption is cubic to the scene size. Especially in the presence of many outliers, nearly all base-voxels have to be refined as the outliers are scattered across the scene and each base-voxel has a high probability to contain at least one outlier. The memory cost directly maps down to the computational cost of the graph cuts, which is used to extract the surface in most approaches.

With the Poisson reconstruction of Kazhdan et al. [25] the output is simply a surface mesh, which can be stored very efficiently. But unfortunately, the approach uses internally a voxel-like octree representation. Thus, the maximum resolution (in this case the octree depth) has to be predefined and drastically influences the run time. There is a quadratic relation between resolution and run time (as well as memory usage). Additionally, other post-processing steps are needed, because no visibility terms can be integrated in the reconstruction procedure.

A very efficient, representation is based on the sample points itself; namely the Delaunay triangulation. A Delaunay triangulation only consists of triangles between sample points. It can be shown [6] that the memory usage of such a triangulation for *generic* surfaces is linear to the number of points. A *generic* surface does not contain any spherical or cylindrical pieces. The computational complexity, in general, is $\Theta(n^2)$, but if the points are well distributed on the surface it can be shown [10] that it drops to $O(n\ log\ n)$.

The output of the Delaunay triangulation is a fully triangulated point set, which means that only the convex hull is visible from the outside. As it was shown in [2] the Delaunay triangulation contains a good approximation of the real surface, if it is sampled dense enough. Note that, this proof assumes ideal sampling, thus, a noise free environment. Opposed to the approaches that try to fit a surface in a continuous space, such as the Poisson reconstruction [25], the task can now be reduced to the search of the surface in a discrete set of triangles. Another very nice property is that a surface can also be extracted from a sparse representation, which is very hard to achieve for volumetric approaches.

2.3 Delaunay Methods

Since nearly three decades [7], researchers around the world have put a lot of afford in the task of finding the object surface in a Delaunay triangulation. As this triangulation originates from the computer graphics domain, the early research focused on finding the surface merely based on geometric properties of surfaces. E.g. Amenta and Bern [2] tried to use Voronoi filtering to extract the surface in the Crust algorithm. They build a Delaunay triangulation of sample points and a subset of the related Voronoi vertices, which they call *poles*. They extract the surface as triangles that are not connected to Voronoi vertices. The problem of this approach is the missing resilience to noise, because it assumes a noise free environment as well as sufficiently dense sampling.

This approach has been further improved and simplified with the Cocone algorithm [4]. Further affords have been made to improve the runtime with an approximate medial axis transform as well as to be more robust on realistic data with a weighted Voronoi diagram in the Power Crust approach [5].

With modifications the Power Crust [30], as well as the Cocone [13], can be made robust against noise. Nevertheless, those modifications focus solely on Gaussian noise and a recent evaluation in [27] shows that both approaches are very sensitive to outliers.

The methods above are very general approaches, that can be applied to a point cloud without any knowledge about the acquisition of the points. But as recent work [26, 27, 39, 31, 32] shows, the visibility of landmarks/keypoints is a very strong feature. The mentioned approaches use this feature to prove that some triangles cannot be part of the surface, otherwise the measurements/keypoints could not be visible from a specific camera. The approaches of Pan et al. [31, 32] solely rely on this feature without further optimization. Instead of an object surface this approach yields a set of triangles where most of the triangles lie inside the object. This is inefficient in terms of storage and makes further processing harder, because no real surface mesh was generated. The approaches in [26, 27, 39], on the other hand, use a global optimization via graph cuts to obtain a watertight mesh as output. This also gets rid of small artifacts which trouble the approach of Pan et al. [31]. Furthermore, the integration of the global optimization leads to a very high resilience to outliers.

3 Implemented Approach

The aim of this work was to implement the chosen approach, such that it provides a range of access possibilities in regard to visibility constraints as well as regularization terms and thusly allows for a thorough evaluation.

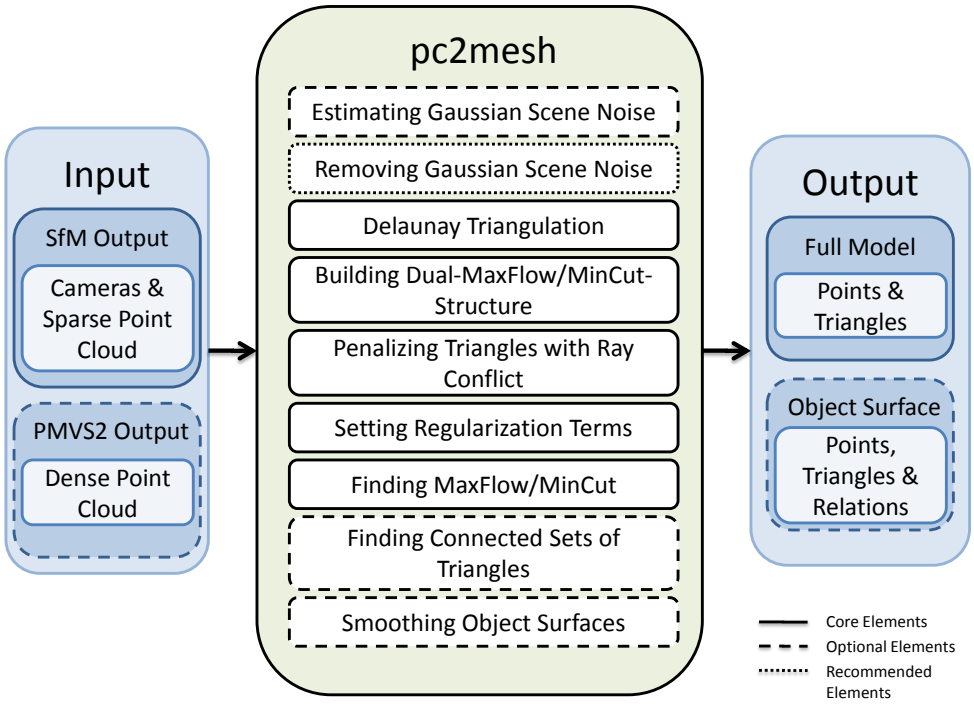


Figure 2: Reconstruction chain of the implemented approach. Elements with a solid border are the core elements of the approach. Elements with a dashed line with large spacing, on the other hand, stand for steps that might be skipped for the sake of performance. The one element with the dashed line with small spacing is recommended by the authors of this work to be added in a future version.

In Figure 2 one can see the main steps involved in the reconstruction process.

The input of the work is at least a sparse point cloud with the visibility information of each point and the intrinsic and extrinsic camera parameters. In this case, this input data is provided by an SfM tool chain. Additionally, one can provide a densified version of the point cloud from the PMVS2 software [18].

The output is a set of watertight 3D meshes which are spanned between the existing input points. Optionally, one can also split the 3D model into connected sets of facets to remove small artifacts.

Further implemented features are scene noise estimation, noise generation (Gaussian and outliers), sub-sampling of the point cloud, posterior smoothing and accuracy analysis.

3.1 Delaunay Triangulation

The Delaunay triangulation is a very well researched triangulation of a point set. It is strongly related with the convex hull, the nearest-neighbor graph and the Voronoi diagram. The convex hull, as well as the nearest-neighbor graph, are subsets of the Delaunay triangulation. The Voronoi diagram, in fact, is the dual graph of the Delaunay triangulation if the point set P is in *general position*. In *general position* means that in the case of three dimension there are no five points that are co-spherical, i.e. that in the degenerated case, the triangulation is not uniquely defined. Especially, this means that the Delaunay triangulation is not well suited to model objects that contain ideal spherical or cylindrical sections. Anyway, algorithms [12] exists to find a unique solution even in such a degenerated case.

One of the most appealing properties for this kind of application is that the resulting triangulation is intersection free. Other nice properties are that it contains a good approximation of the real surface if the sampling is dense and accurate enough [2] and that for *generic* surfaces the number of simplices in the graph is linear to the number of points [6]. Our experiments show that in architectural scenes the number of tetrahedra is roughly 6 times and the number of facets roughly 12 times the number of points.

In this work we used the very robust and efficient implementation of the CGAL library [41]. It finds a unique solution for the Delaunay triangulation [12] and also caters for infinite cells and an infinite vertex. This is important for the construction of a complete dual graph for the energy minimization.

3.2 Energy Formulation

This section is very strongly based on the work of Labatut et al. [26, 27].

The fundamental idea of this approach is to use the measurements to disprove triangles in the Delaunay triangulation and induce a binary inside/outside labeling.

To achieve this, a pseudo-dual graph of the Delaunay triangulation is devised. In this dual graph the tetrahedra (or cells) become vertices and

the triangles (or facets) that separate the cells become oriented edges. This means for each facet two edges with opposite directions are inserted.

The resulting graph can be used for the binary labeling task. A cell is either labeled as *inside* or *outside*, and the surface is resulting implicitly as the set of facets that lie between cells with different labeling.

The overall energy of the surface S, which is to be minimized, can be written as:

$$E(S) = E_{vis}(S) + \alpha_{reg} \cdot E_{reg}(S) \tag{1}$$

where $E_{vis}(S)$ represents the data term and is the sum of the penalties from the ray conflicts across the surface. $E_{reg}(S)$ can be seen as a smoothness term and is the sum of all regularization penalties across the surface. α_{reg} is a constant parameter, which weights the relation between the two terms and determines the degree of smoothness. Note that, the regularization terms can be of any kind and also a combination of several such terms is possible and sometimes even reasonable for certain applications. See Section 4.1 for an extensive evaluation of these regularization terms.

Theoretically, it would also be possible to add other terms, such as photometric consistency, into the energy formulation. These terms could further improve the optimization, but also would add additional degrees of freedom, and thereby parameters which need to be tuned. But more importantly, these terms would add an additional computational cost to the approach, which can easily become very expensive in the case of the photometric consistency.

Figure 3 shows the most important concepts of this work and how the visibility constraint as well as the regularization can be integrated in the graph cut optimization. Note that, the graph cut representation is closely related to the Voronoi diagram, which is the dual graph of the Delaunay triangulation. The only difference is that in the graph cut representation the Voronoi diagram is augmented with two terminal vertices (source and sink). These vertices as well as the edges from and to them do not have a correspondence in the Delaunay triangulation, but are only used for the labeling task.

The remaining part of this subsection describes possible ways to incorporate the visibility constraint into the labeling task and explains the need for regularization as well as several ways to achieve it.

3.2.1 Data Term - The Visibility Constraint

The basic idea is to use the information of which point is visible in which camera to induce an inside/outside labeling. The task can be split into three, more or less, separate parts. First of all, one has to set connections (edges) from the source to the dual graph of the Delaunay triangulation. This will

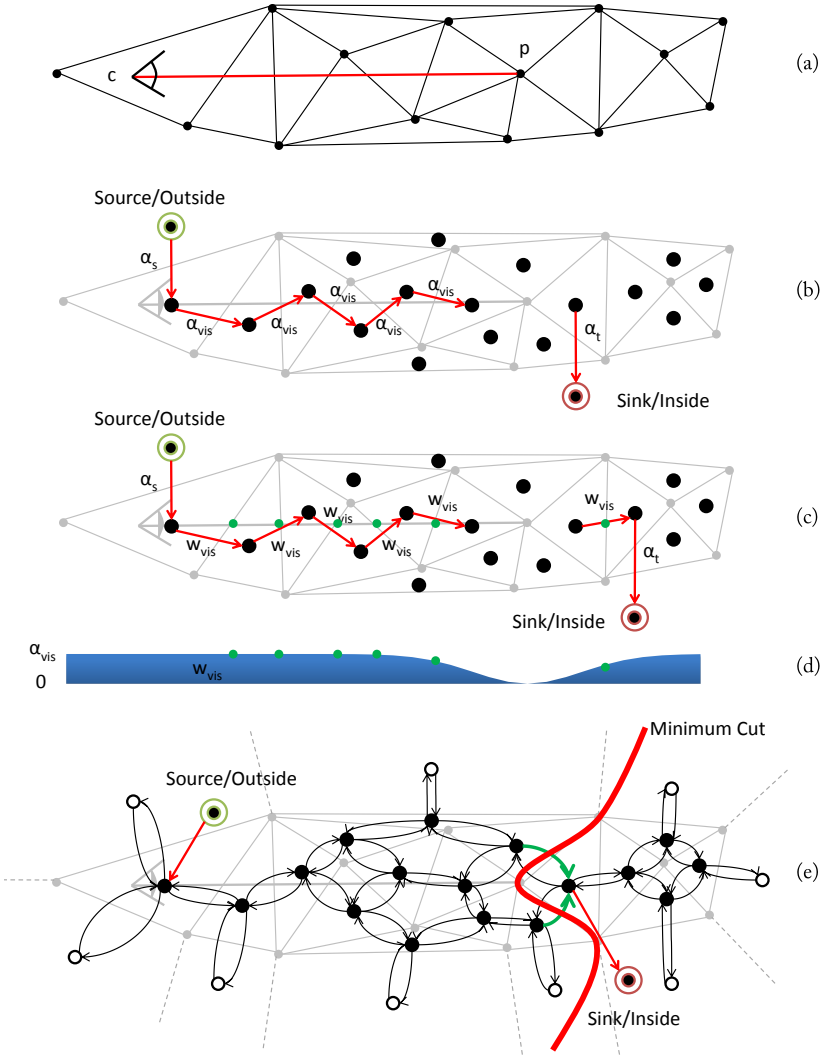


Figure 3: Graph cuts problem formulation: (a) Delaunay triangulation of the point cloud, with a camera center c and a point p which was visible from this camera. (b) Hard visibility weighting scheme as proposed in [26]. The large black dots are the graph cuts analogue to the tetrahedra in the triangulation and the red arrows specify the added weights. (c) Soft visibility weighting scheme as proposed in [27]. (d) The assigned facet weights of the soft version (c). The weights are dependent on the distance to the measurement as well as the width of estimated Gaussian scene noise σ. (e) Fully regularized flow graph of the hard version (b). All edges are part of the optimization. The thick red line is a minimum cut of the graph. The large not filled black dots correspond to the infinite cells in the Delaunay triangulation.

define parts certain parts of the graph as *outside*. Secondly, connections from the dual graph to the sink have to be created. These links can be seen as votes for the inside of the object. Finally, one wants to enforce the empty space constraint in penalizing ray conflicts. This is the most important part of the data term. The following paragraphs discuss these three tasks in detail.

Outside Labeling The first task is to create terminal links from the source to cells of the Delaunay triangulation. This means that we have to define certain cells which are to be labeled as *outside*.

For this task Labatut et al. [26, 27] simply suggest to define the cells the camera centers are in as *outside*. In our experiments we have discovered that under certain conditions this kind of outside definition may cause the scene to become unstable. It might happen that half of an object simply disappears, because there might only be one cell that is defined as outside. Thus, to stabilize the approach we propose to also define all infinite cells as outside. An infinite cell is not a physical cell, but a cell which is connected to the infinite vertex, which does not have a real location. Note that, an infinite cell always lies outside the convex hull of the point set. CGAL [41] defines multiple such infinite cells, one per facet on the convex hull. In our approach it would also be sufficient to simply define one such infinite cell, which then of course could have more than 4 neighbors.

Defining everything outside the convex hull as *outside*, does not harm the reconstruction as the surface of the object which is to be reconstructed has to be on or inside its convex hull. The only property that changes is, that the approach will now always return closed object surfaces. In practice, this turns out to be no problem at all, as it will only influence parts of the scene, which we have simply no information about, e.g., what is behind the wall?

This can even be an advantage for certain systems. Firstly, *inside* and *outside* are implicitly and unambiguously defined for further processing steps. Secondly, if one thinks of interactive or autonomous systems, facets which cannot be seen from any camera, can be cues that those regions have to be investigated further.

The definition of a cell as *outside* is done by setting weights from the source to this cell.

In a hard constraint as proposed in [26], an infinite weight is set. This means, that there is no way that this cell is to be labeled as inside. In a soft constraint as proposed in [27], a finite weight is added to the connection source to cell. This procures the question, what is gained by this definition? The answer is that cells, like the ones where the the images were taken in, can be labeled as inside. As this seems rather unintuitive, we decided against

it and further on only considered the hard constraint.

Inside Labeling The inside labeling, opposed to the outside labeling, should, in general, not be done with infinite weights. Such a labeling would weaken the graph cut energy minimization. The measurements are prone to contain outliers as well as Gaussian noise. If one would assign infinite weights, this noise would severely influence the outcome.

Thus, Labatut et al. [26, 27] propose finite weights. A simple constant value [26] can already get rid of outliers, but is not able to cope with Gaussian noise. Consequently, a soft visibility constraint was proposed in the more recent work of the same group [27]. In Figure 3 one can find a comparison of the two approaches. In the very recent work [39] of the same group, however, which mostly talks about a possible variational post-processing step, they seem to discard the soft visibility constraint and return to the constant version. This motivated us to investigate this topic further. In the experimental section one can find proof that the soft visibility, in fact, does not pay off in matters of accuracy.

The basic idea is that a pair of a 3D point and a camera, in which the point is visible in, can vote for the inside of the object. The simplest and, as it turns out, better way is to simply add a finite constant weight to the connection from the cell behind the measurement p to the sink as in Figure 3b. In [27] it was proposed to add the weight to the connection from the cell behind the first facet after the measurement to the sink instead. This was probably done to add some power to the soft visibility constraint. Anyway, it does not have a concrete theoretical basis and was probably devised empirically.

To investigate if the result can be improved with a better theoretical basis, we decided to try another approach. Instead of increasing only one sink-link per point-camera-pair, we allow the increase of the weights of all cells that are in the range of great confidence behind the measurement. Empirically, we decided this confidence range to be $\sigma_{est}/2$, where σ_{est} is the estimated width of the Gaussian scene noise. The calculation of σ_{est} described in detail in Subsection 3.4. The sink-links are also weighted with the confidence in the measurements. They are weighted in relation to the confidence in the facets behind the measurement. The added sink weights are then defined as $w_{sink} = 1 - w_{facet}$. w_{facet} corresponds to w_{vis} in the chart in Figure 3d.

Penalizing Ray conflicts The idea is to enforce the empty space criterion. This is achieved in penalizing facets that cannot exist because otherwise the measurement p could not have been obtained from camera c. Thus, one adds weights in the direction of the ray as it can be seen in Figure 3. Note that, it

is very important that the facets are updated in the right direction, as this direction defines the allowed flow from the source to the sink in the graph.

This feature is very powerful, but also comes at a significant computational cost. If implemented efficiently, the complexity of penalizing the ray conflicts is $O(n \cdot log(f))$, where n is the number of points and f the number of facets. Under the assumption of a *generic* surface it is simply $O(n \cdot log(n))$. An efficient implementation of the underlying structures and methods, such as the Delaunay triangulation or ray tracing, for the calculation of the energy terms is provided by the CGAL library [41].

Labatut et al. [26] first proposed to use a fixed penalty for each ray conflict. In order to be able to cope with Gaussian noise, they later decided [27] to introduce a soft version, which has a confidence parameter σ. The assigned weight in this version is dependent on the confidence in the measurement (estimated noise) as well as the distance d to the measurement:

$$w_{vis} = \alpha_{vis}(1 - e^{-d^2/2\sigma^2}) \tag{2}$$

In the hard version only facets in front of the measurement are penalized, whereas, in the soft version also the first facet behind the measurement is penalized.

As for the sink-links we decided that, instead of simply updating strictly one facet behind the measurement, it should be done for all facets in the range of great confidence.

3.2.2 Smoothness Term - Regularization

The need for regularization is combined with usage of the graph cut-based energy minimization. If only the penalties from the visibility constraint are used, the optimization can become unstable, and, in practice, always yields a surface with unpleasant hole-like spikes as can be seen in Figure 4.

This is due to the fact that facets only become available in the optimization if they have been assigned a penalty (capacity), otherwise they cannot be part of the solution. In the unregularized version all regularization parameters are set to zero and only the visibility constraint (the data term) remains for the optimization. This means, that a facet can only be part of the solution if there is measurement behind it that disproves it. Thus, one has to think of reasonable ways to regularize the optimization. The remaining part of this subsection will talk about possible ways to regularize the minimization.

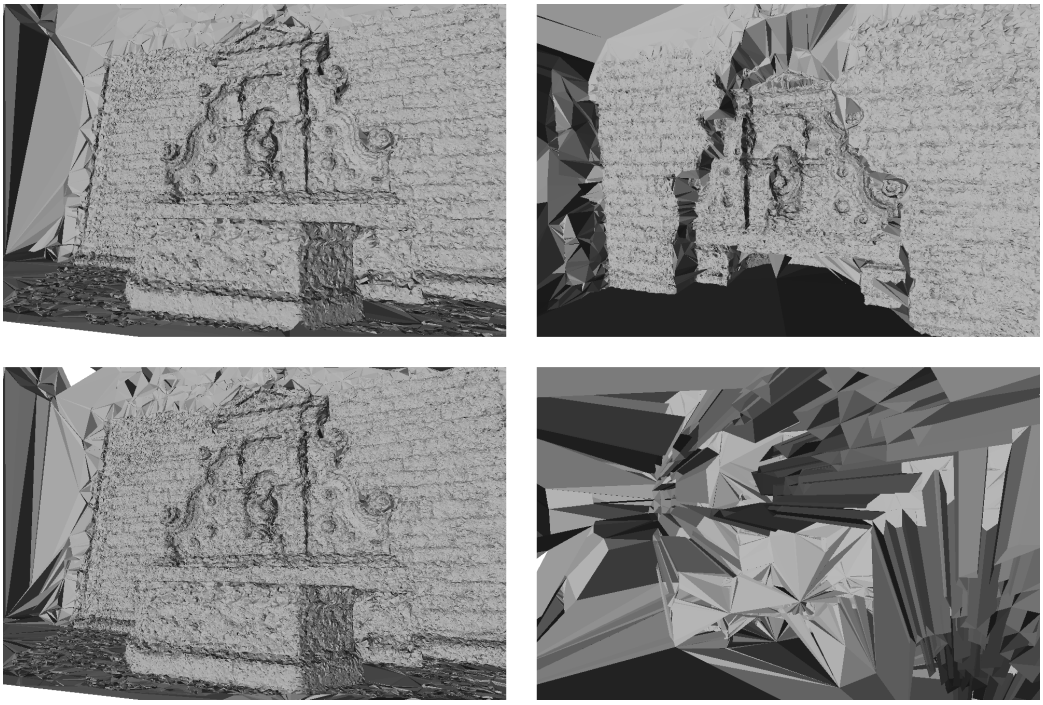


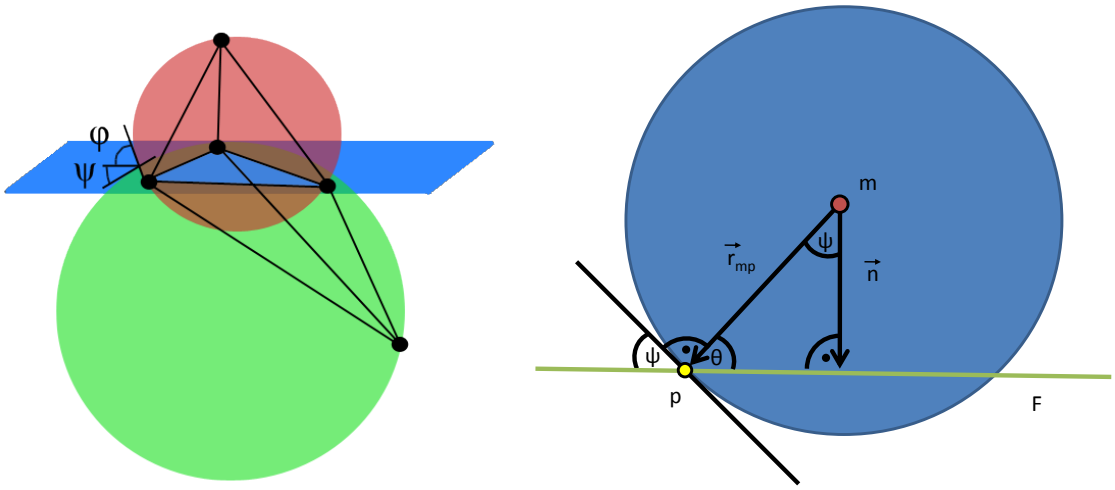
Figure 4: Comparison between regularized (top) and unregularized (bottom) energy minimization . "Unregularized" means that only the visibility constraint (the data term) was used for the optimization. The viewed object is the "fountain" of the dataset of Strecha et al. [37]. Left: Frontal view (from this viewpoint the original images were taken). Right: A view from behind the surface.

Area A very simple way of regularization is to penalize the area of the triangles. Thus, small triangles get a smaller penalty than large ones. This regularization can become very handy to remove large facets connected to a far-away outlier. But it turns out that the parameter of this term has to be set with care, as it tends to prefer large triangles close to surface. This means that if there exists a hypothesis of the surface that is made up of a lot of small triangles, it is very likely to be rough. Thus, a hypothesis with less triangles in the Delaunay triangulation is smoother and, consequently, minimizes the overall area of the surface mesh.

Beta Skeleton The beta skeleton is a more sophisticated regularization term, which is based on an idea of Amenta et al. [3]. The original concept was designed for two dimensions. The beta-skeleton is the set of edges, which have no other points in the "forbidden" region. The forbidden region is the

union of the two disks which have a radius of $\beta d(p_1, p_2)/2$, where p_1 and p_2 are the points at the end of the edge, $d(p_1, p_2)$ is the length of the edge and β is a constant. If the sampling is dense enough in relation to β, this approach is guaranteed to output the right reconstruction.

Labatut et al. [27] generalized this concept to the third dimension and incorporated it into the optimization. Instead of using such a hard constraint, they devised a soft approximation. They analyze the angle at which the circumscribing spheres of the two adjacent tetrahedra intersect with the plane of the facet. This angle is really better defined as the angle between the tangent plane at a point on the intersection circle and the plane of the surface, as depicted in Figure 5b.



(a) (b)

Figure 5: Beta skeleton generalization. (a) Intersection angles of the circumscribing sphere of two adjacent tetrahedra with the facet plane. The spheres intersect the facet plane with angles ϕ and ψ. (b) Cut through a circumscribing sphere at the plane that is normal to the facet plane, and goes through the sample point p and the sphere center m. The intersection angle of a sphere with a plane can be calculated from the normal $\vec{n}$ of the facet plane F and the normal of the tangent plane through sample point p, which corresponds to the radial vector $\vec{r}_{mp}$ from m to p.

The soft generalization of the beta skeleton is then defined as:

$$w_\beta = 1 - min\{cos(\phi), cos(\psi)\} \tag{3}$$

where ϕ and ψ are the intersection angles of the two adjacent tetrahedra with a facet plane as depicted in Figure 5a and w_β is the applied penalty.

For the calculation one can fit a tangent plane at any point on the intersection. The most convenient way is to simply choose one of the sample points that define the facet. Such a point p is per definition on the facet plane, as well as on the circumscribing sphere of the adjacent tetrahedra. Then one can use the radial vector $\vec{r}_{mp}$ from the sphere center m to p, which directly corresponds to the normal vector of the plane (see Figure 5b). Using the symmetry property of the cosine one can reduce the calculation of the term $cos(\psi)$ to the normed dot product of the two normals:

$$cos(\psi) = \left| \frac{\vec{r}_{mp} \cdot \vec{n}}{|\vec{r}_{mp}| \cdot |\vec{n}|} \right| \tag{4}$$

where $\vec{n}$ is the normal of the facet plane.

Constant The regularization with the beta skeleton, although theoretically well-founded, turns out to be inferior, in terms of accuracy, to the simplest of all regularization terms, a constant penalty. This might sound surprising, but also this has a solid theoretical basis, although not from the side of surface theory. It is based on one of basic concepts of information theory: The minimum description length (MDL) principle.

In a set of possible surface hypotheses the most simple one is to be preferred. In this case a surface is defined through its facets, therefore a simplest solution is the hypothesis that minimizes the overall number of facets.

As it is shown in the experimental section, of all tested regularization terms the MDL approach is the only approach that can increase the accuracy of the surface reconstruction compared to the unregularized optimization.

3.3 Global Optimization via Graph Cuts

Graph cuts are based on the theory of flow networks. A flow network is a graph of vertices and directed edges. There are two special vertices; the source and the sink. The source is the only vertex that can produce "flow" and the sink is the only one which can absorb it. This means for all other vertices that the amount flowing into a vertex equals the amount flowing out (cf. Kirchoff's current law). The flow of an edge is limited by its capacity.

The optimization tries to find the maximum flow from the source to the sink in the network. When the maximum flow is found, one is interested in the bottleneck of the graph, which is limiting the flow. This bottleneck is called minimum cut. Note that, there might be several possible minimum cuts in a graph. A cut partitions the graph into two disjoint sets of vertices. One is connected to the source and the other to the sink.

Many computer vision tasks can be formulated as energy optimization problems, e.g. labeling tasks, which can be efficiently solved with graph cuts.

In this case here, the capacity of an edge (facet) is the weighted sum of penalties from the visibility constraint and the regularization (cf. Equation 1). In increasing the weight/capacity of an edge, it gets more expensive to cut it, therefore the probability of being part of a minimum cut decreases. After finding the minimum cut all tetrahedra are either labeled as outside or inside. The found minimum cut is interpreted as the output surface, as it minimizes the energy in Equation 1. Figure 3e provides a graphical example.

Following in the footsteps of Labatut et al. [26, 27], this work uses the library of Boykov and Kolmogorov [8]. The theoretical complexity of this implementation is $O(v \cdot e^2 \cdot c)$, where v is the number of vertices (cells), e the number of edges (facets) and c the capacity of the minimum cut. This implementation is tuned for regular grid graphs, such as digital images, and has a worse theoretical complexity then most standard algorithms. Nevertheless, it turns out to be very fast in practice, also on this task.

3.4 Scene Noise Estimation

In the attempt to raise the soft visibility constraint to its full potential, it was necessary to estimate the scene noise to allow for the σ parameter to be tuned automatically.

There exists a lot of literature on the topic of denoising, but far less work on noise estimation. A widely used approach for image noise estimation is the mean absolute deviation (MAD) [14]. More recent work is concerned with the noise estimation from a single image with measured CCD camera response functions [16]. The scene noise, however, depends only to a fraction on the image noise. Scene noise can be induced nearly in every step of the whole reconstruction chain. The feature detector can have responses at slightly wrong positions, the feature matcher can match wrong features, the SfM algorithm can be off in matters of point locations and camera parameters. Thus, the image noise alone is hardly enough to estimate the scene noise.

Consequently, we thought of possible ways to estimate this parameter.

The first approach was to analyze the mean and median reprojection error, and then use this uncertainty as input for a computation of the median principle singular value of the intersection covariance matrix. The estimation was, unfortunately, rather poorly.

It was discovered that the second step did not have an effect on the relation between the real noise level and the estimated one. It simply relied

on the reprojection error and scaled it with a constant factor depending on the scene, but was still far from the real noise level.

In the analysis of the relation between artificially induced Gaussian noise and the mean/median reprojection error, it was discovered that this relation stayed constant, unless the noise was getting too large (meters in an outdoor scene).

If outliers were present, the relation could not be properly captured by the mean reprojection error, thus, we switched to the more reliable median reprojection error. The median happens to capture the right relation without outliers and is only slightly off in their presence.

This property is exploited in our approach to estimate the scene noise.

Firstly, a relatively small noise in comparison to the scene size is artificially induced. Secondly, the distance between the reprojected point (artificially perturbed by Gaussian noise) and the *ideal* reprojected point is interpreted as reprojection error. In this case, *ideal* reprojected point does not refer to the original 2D feature key point location, but to the reprojection of the original unperturbed 3D point. Finally, a constant factor k between the median reprojection error and the scene noise level can be extracted, as the noise was artificially induced with a known level. We assume, that k can be reliably estimated from a large enough random subset of points.

The estimated width of the Gaussian kernel σ which represents the scene noise level, can now be simply computed from the measured median reprojection error e_r as:

$$\sigma = k \cdot e_r \tag{5}$$

This approach is very good at estimating artificially induced noise (see experimental section 4). Unfortunately, it was not possible to evaluate the quality of the estimation in a real world example due to the unavailability of a dataset with known levels of noise.

This estimation cannot only be used to tune the σ parameter, but can also be used for better prior or posterior smoothing.

This approach, of course, needs knowledge about point correspondences in the images and the corresponding 3D points. As the PMVS2 does not output such information, we simply assume that the noise level on the sparse reconstruction is equal to the noise level of the densified PMVS2 point cloud.

3.5 Surface Object Extraction

The surface, in this case the minimum cut, can be found by searching for facets which separate cells with different labeling. This can easily be done in linear time.

We decided to experiment with a simple post processing step of the surface. Hence, it became necessary to extract what we call *surface objects*. A *surface object* is a set of facets that are connected via edges.

To achieve this a simple brute-force approach with a complexity $O(f^2)$ was taken, where f is the number of facets. The algorithm works as follows: It iterates over all facets, and looks if there exist connected sets of facets which this facet is connected to. If so, the sets are merged as they now are connected and the facet is added to the merged set. If no such set exists a new set is created.

This can clearly be improved in using the knowledge about neighbors from the Delaunay triangulation. A possible algorithm using this information to speedup the extraction process can be found in Algorithm 1.

Algorithm 1 Surface object extraction algorithm using neighborhood information

- *allsets = empty*

while *not all facets have been marked as assigned to a set* **do**

- *currentfacet = a facet from all facets that have not been marked as assigned*
- *neighborstack = empty*
- *push all neighbors of currentfacet into neighborstack*
- *currentset = empty*
- *add currentfacet to currentset*

while *neighborstack not empty* **do**

- *currentfacet = top facet of neighborstack*
- *pop neighborstack*
- *push all neighbors of currentfacet which are not marked as assigned into neighborstack and mark them as assigned*
- *add currentfacet to currentset*

end while

- *push currentset into allsets*

end while

A short runtime analysis of this algorithm: The outer WHILE-loop is called o times, where o is the total number of *surface objects*. The first facet can be selected in $O(f)$, where f is the number of facets. The inner WHILE-

loop is called at a maximum of f times. All other steps can be computed in constant time. Hence, this algorithm has complexity of $O(f \cdot o)$. Note that, o is, in general, very low compared to the number of facets, consequently, this would bring a significant speedup to the object extraction.

3.5.1 Simple Surface Smoothing

To evaluate the power of the soft visibility constraint opposed to smoothing, a posterior surface smoothing was applied. This surface smoothing takes the estimated noise σ into account. In the smoothing process a mean over a set of vertices is computed. The set consists of all vertices which are within a distance of 2σ from the vertex of interest and are connected to this vertex directly or across vertices within the specified distance. Note that, if the distance would be chosen too large, this would yield a skeleton of the real object.

3.5.2 Thoughts on Clean Surfaces

One of the ideas behind the *surface object* extraction was to be able to create a valid mesh in a half-edge structure to ease further processing. To this aim, an edge is only to be shared by two facets. As it turns out, the *surface object* is not yet suited for such a structure. As it happens, there is nothing in the graph cut approach that prohibits different *surface objects* to share a point or, more gravely, to share an edge. It is quite easy to detect such an ambiguous edge, as one just has to look for edges that are shared by more than two facets. The problem, that is to be faced afterwards, is more complex than it might seem at first. The easiest case would be that the ambiguous edge is shared across two *surface objects*. Then one could think of duplicating the edge and handing one of them to each *surface object*. Unfortunately, these ambiguous edges can also occur inside a single *surface object*. To make the problem even more complex, it can happen, that a single facet is made up of two or three such ambiguous edges. Furthermore, there is no restriction that only four facets share an ambiguous edge (which is mostly the case), but, in theory, it can be any multiply of 2. Uneven numbers are not possible because of the cell structure.

This makes a duplication of such edges very difficult, because it gets very hard to detect which facets should share an edge. To achieve this, one would have to have a strict inside/outside labeling, which is possible as only closed surfaces are returned by the implemented graph cut approach. Furthermore, it would be necessary to project the facets on the plane which has a normal parallel to the edge. On this plane together with the labeling it can be

determined which facets should share an edge.

After the removal of ambiguous edges, ambiguous points (connected peaks) can be cleaned using the neighborhood relation of facets that share this point. If now everything is cleaned up, one has to recheck the connectedness of the set. This cleansing of the surface can be very costly, and it is to be questioned if such an afford is needed for further processing/visualization steps.

4 Evaluation

The approach was tested on a range of datasets. Figures 6 and 7 display the results on different datasets. For each dataset the mesh was computed on the sparse point cloud of the SfM output as well as on the densified output of the PMVS2 approach [18]. Note that, the models are not textured, but the color of the triangles is an interpolation of the color of the input points. This leads to a shaggy appearance at regions with a low point density.

As a comparison the Poisson reconstruction [25] was chosen. This choice was made for several reason. First of all, the Poisson reconstruction yields very good results and it has been shown that it is a reasonable initialization for further refinements [18, 19]. Secondly, it is a very popular approach and is available in several tools such as MeshLab [42]. In this evaluation we used fixed parameters for the Poisson reconstruction. The parameters were chosen such that a good accuracy is achieved while the computing time is kept low. The parameters were chosen as follows: *octree depth* = 12, *solver divide* = 6, *samples per node* = 1, *surface offsetting* = 1.

Furthermore, an analysis of the effect of promising regularization terms and added noise was conducted. The effects are analyzed with respect to the ground truth of the "fountain" scene of the dataset of Strecha et al. [37]. This dataset appears to be the only available outdoor dataset with a ground truth mesh. The "fountain" scene was chosen specifically because it has a very complete ground truth, which is not the case for e.g. the "Herz-Jesu" scene, where important parts, such as hand rails, are missing.

Evaluation Strategy The chosen evaluation strategy is very similar to the one proposed by Strecha et al. [37]. It computes the depth maps at the ground truth camera positions and compares them to the depth maps that were generated from the ground truth mesh. The evaluation analyzes the depth error of the different reconstructions. As parts of the scene might be missing, a simple mean error would not yield objective results.

Thus, we analyze the spectrum of the error similar to Strecha et al. [37]. They use a fixed value t, and analyze the spectrum of the error in ten linear

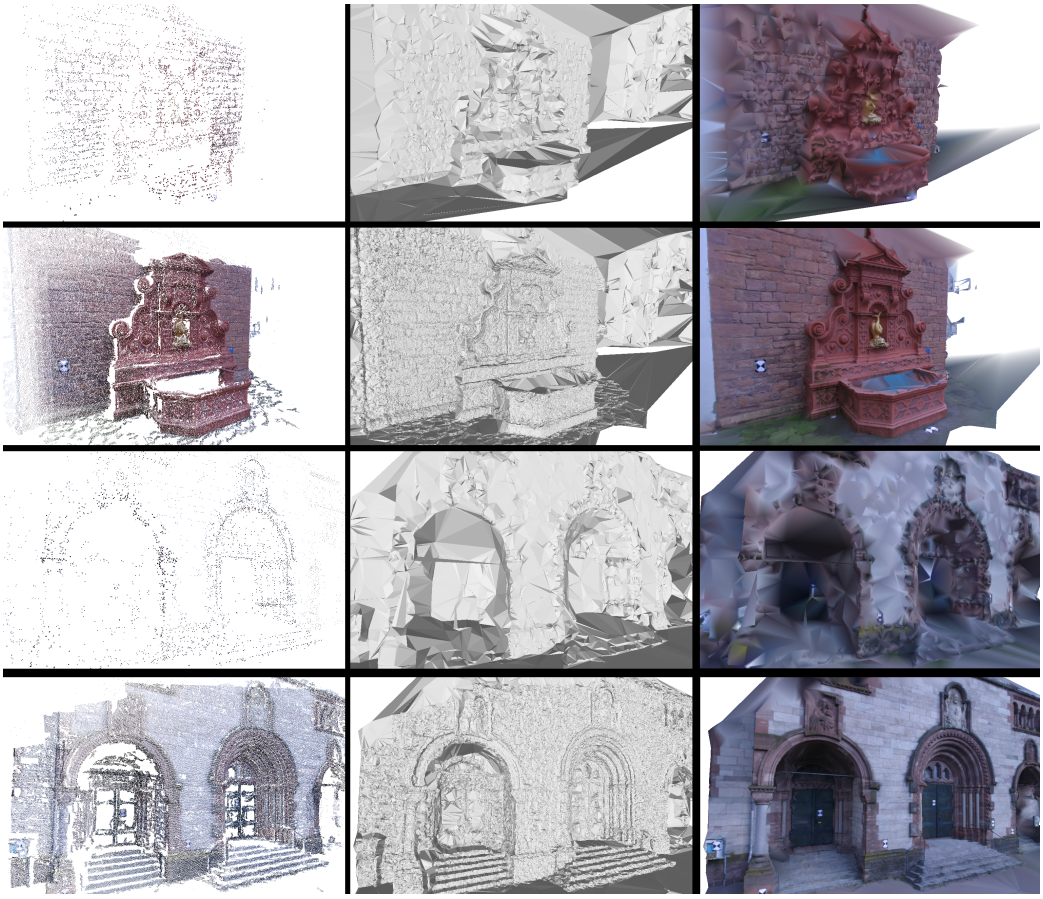


Figure 6: Results of the "fountain-P11" scene and "Herz-Jesu-P8" scene of the dataset of Strecha et al. [37]. From left to right: Point cloud, uncolored mesh, colored mesh.
Top: Reconstruction of the "fountain-P11" scene (11 images) from the SfM output (7123 points) and the PMVS2 [18] output (375 409 points).
Bottom: Reconstruction of the "Herz-Jesu-P8" scene (8 images) from the SfM output (4880 points) and the PMVS2 [18] output (298 993 points).

steps of t. This analysis is not very general, e.g., if a t of 1mm is chosen, then only the spectrum between 1 mm and 1 cm is analyzed.

As an improvement and to be able to capture the error in fine detail as well as large deviations, we propose a non-linear scaling of the spectrum. We propose a spacing to the power of 2, i.e. $2^i \cdot t$, with $i = 0, 1, ...9$. We chose the minimum spacing $t = 0.001$ m for the experiments. This way we can capture the quality of small details in mm as well as large deviations in the range of 0.5 m.

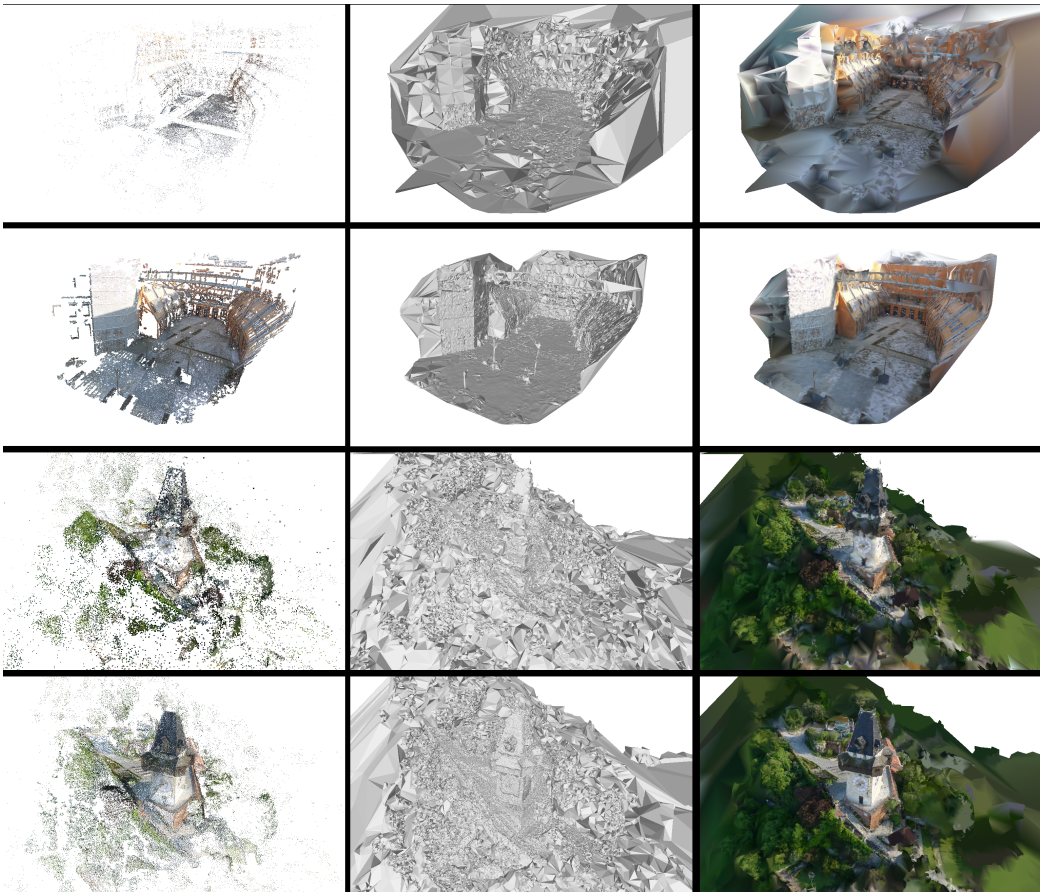


Figure 7: Results on two datasets of the Aerial Vision Group of the ICG institute. From left to right: Point cloud, uncolored mesh, colored mesh. Top: Concave scene of a courtyard the Inffeldgasse 10 building in Graz (104 images). SfM output (36 948 points), PMVS2 [18] output (318 540 points). Bottom: Large scale scene of the clock tower in Graz (421 images). SfM output (60 822 points), PMVS2 [18] output (277 429 points).

The color scheme for the visual comparison was also adapted in a similar manner to reflect this non-linear spacing. In contrast to Strecha et al. [37], we used the jet color map for the depth error, because humans can better distinguish colors then gray value differences. The colors range from dark blue (no error) to dark red (error larger than 0.5 m). The color black, means that there is no ground truth available for this part, whereas, white symbolizes that the query mesh does not model this part of the image, although it is modeled in the ground truth.

As it was not the aim to evaluate the SfM approach further up the recon-

struction chain, we used the ground truth positions of the cameras for the sparse reconstruction and the densification with the PMVS2 approach [18].

The remaining section is structured as follows: The evaluation of the effects of the regularization parameters in the case of soft and hard visibility constraints is described in Subsection 4.1. Subsection 4.2 covers the analysis of the robustness against noise. Not only the influence of outliers is evaluated, but also the effects of Gaussian noise. This section ends with an evaluation of the consumption of time and memory in Subsection 4.3.

4.1 Regularization Parameter Effects

This section will evaluate, which regularization term is better suited for constraining the smoothness of the final surface. It does not only evaluate it on a subjective basis as in [27], but compares the result mesh directly to the ground truth of the "fountain" dataset of Strecha et al. [37]. As it turns out, this is very important in regard to accuracy. Some meshes may look "better", as they are smooth approximations, but, in truth, are farther away from the real object surface. It was also discovered, that the regularization terms, show a very different behavior when hard or soft visibility constraints are used. Consequently, those two cases will be discussed separately, and then a comparison will draw conclusions from this evaluation.

4.1.1 Hard Visibility Constraint

In the case of the hard visibility constraint each ray conflict is penalized with a constant weight. The sink link weight is increased at the cell directly behind the measurement. In order to analyze the effect and quality of the different regularization terms, the *error percentage* above different thresholds is evaluated. The *error percentage* is the percentage of pixel in the depth map images which have a depth value difference above a specified threshold. We wish to see the effects of varying the parameters from very low values until the point of failure in non-linear steps. To get the effects into a comparable parameter range, the *area* parameter was additionally multiplied with a value of 0.0346918, such that the average area is 1.

In Figure 8 we show the changes of the percentage of pixels which are below a certain threshold. On the left the threshold is 0.016 m and on the right it is 0.512 m. The first threshold should capture the accuracy in a reasonable range and the second one large deviations. As one can see, no matter which term is concerned, the best accuracy is reached if the regularization parameters are kept small. It just has to be small enough not to take too much influence on the optimization process. The errors of the *constant* and

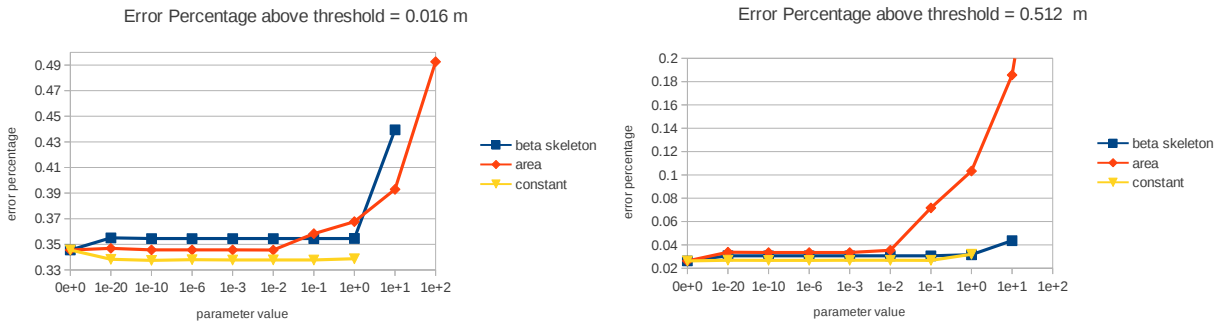


Figure 8: *Hard-Visibility Experiment* 4.1.1: Error percentage over varying parameters using a hard visibility constraint. When a line stops it means, that the reconstruction failed for the next parameter value.

the *beta skeleton* term stay on a constant level until a parameter value of 1. The *area* term takes effect earlier, but also shows a significantly more graceful ascent than the other two terms. In the diagram a disappearing line means, that the optimization process fails at the next value. Failing in this case means that it is cheaper to cut all sink links than to find a cut through the triangulation. This results in zero facets in the output mesh, which happens if the system is over-regularized.

Anyway, it seems to be the best idea in terms of accuracy to just add an "epsilon" of regularization to the system, and let the optimization be dominated by the visibility constraint. If the terms get too large and take a serious effect, it always diminishes the accuracy, and can even lead to a failure of the system.

Note that, a simple *constant* is the only of all tested terms, that improves the accuracy compared to the unregularized version. In the unregularized version all regularization parameters were set to zero and only the visibility constraint (the data term) remains for the optimization. This case is denoted in the charts as a parameter value of $0e+0$. The complex term of Labatut et al. [27], the *beta skeleton*, turns out to perform worst of all on fine details.

In Figure 9 the whole error histogram of all regularization terms is displayed. Note that, on the first bins, the *constant* term has a higher percentage than the others, and on the later bins it has a lower percentage. This can be interpreted that it has a higher accuracy and a lower overall error.

It is also apparent, that there is a great peak at about 1 cm for all regularization terms, although the higher bins have a far bigger range. This peak can be interpreted as the mean reconstruction accuracy.

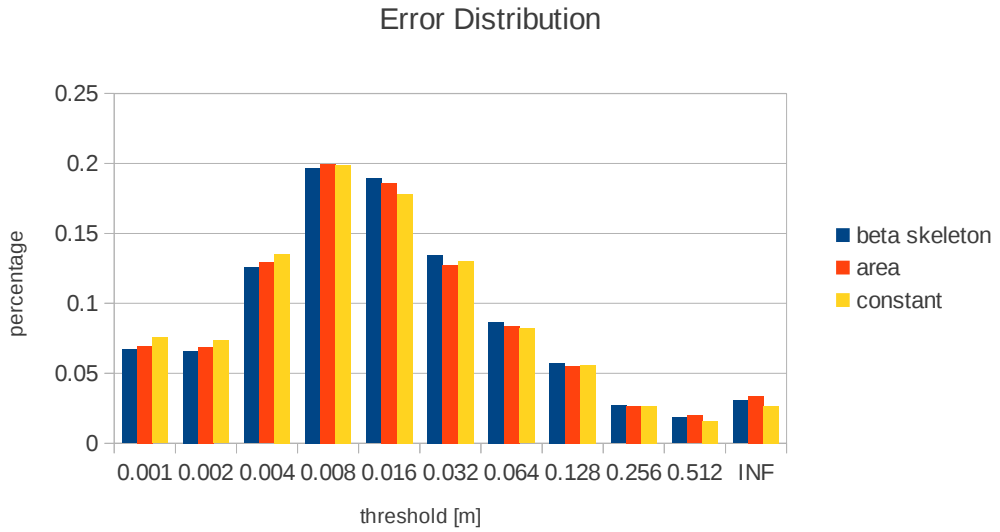


Figure 9: *Hard-Visibility Experiment* 4.1.1: Non-linear error histogram for a parameter value of 10^{-20} using a hard visibility constraint.

4.1.2 Soft-Visibility

The soft visibility constraint brings an additional degree of freedom. Its intended purpose is the integration of a term, which represents the Gaussian scene noise. The effect of the different regularization terms now also dependents on the chosen σ value, which is related to the width of the virtual kernel which caused the scene noise. Is the σ very small compared to the scene size, it has nearly no effect at all and behaves just equal to the hard visibility constraint. Thus, to analyze the effect of σ one has to set it to a range where it effects the optimization.

Sigma Variation The first conducted experiment varies the parameter of the *constant* term as it had the best accuracy in the case of the hard visibility constraint, while changing the σ parameter in decades. Figure 10 displays the cumulative error distribution for different parameter and σ values as well as the percentage of pixels with an error above a threshold of 0.016 m. With an increasing σ value the regularization term takes more effect on the optimization. Unfortunately, this effect turns out to decrease the accuracy of the surface mesh.

Another effect is visible in Figure 11. If the σ parameter is increased at a low constant regularization parameter, the number of artifact triangles at

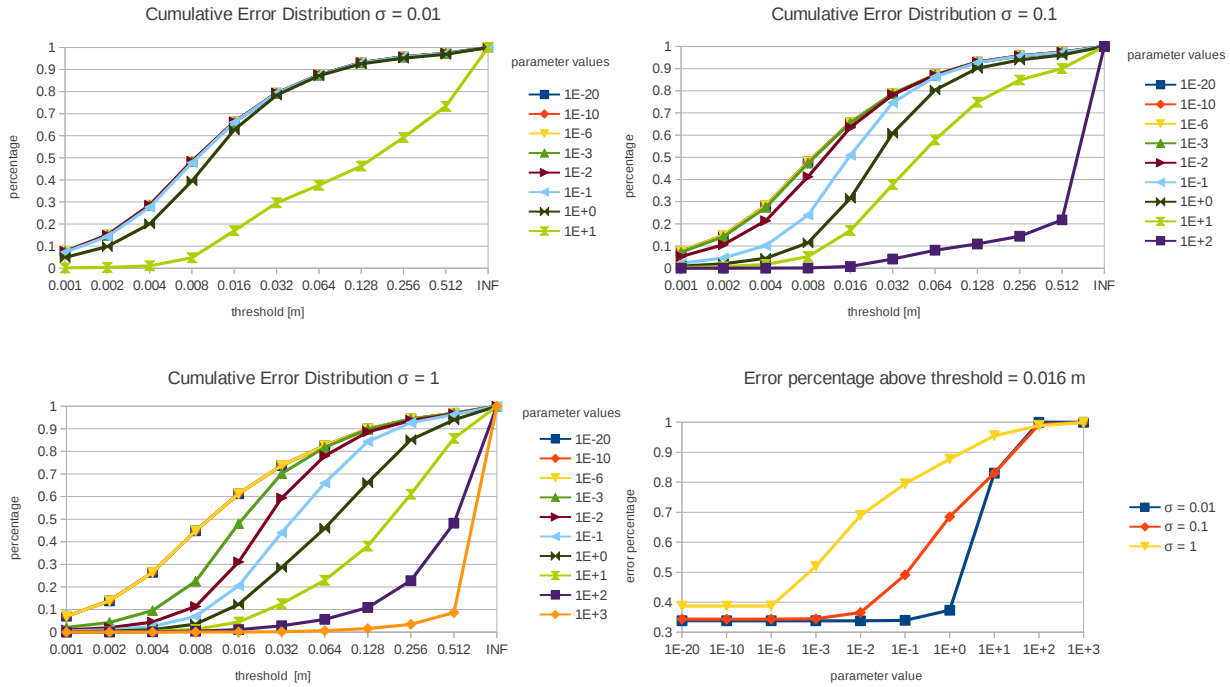


Figure 10: *Soft-Visibility Experiment* 4.1.2: Non-linear cumulative error distribution for varying parameter and σ values. Bottom right: Percentage of pixels with an error above the threshold = 0.016 m for different σ values.

non-convex parts of the surface rises.

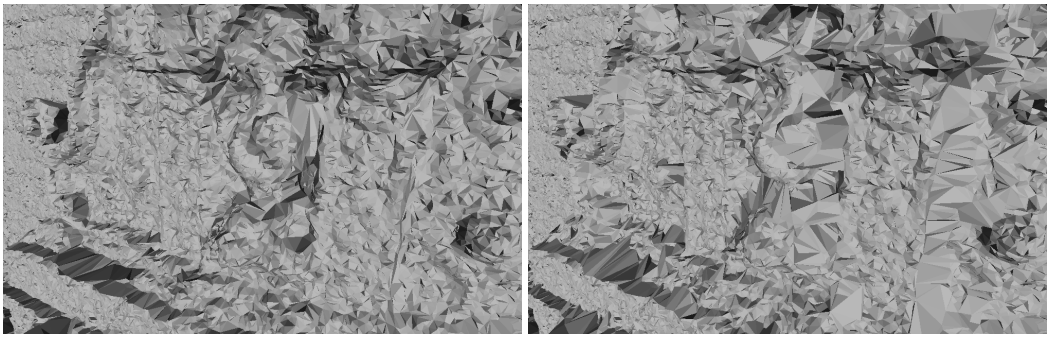
On the other hand, if the regularization parameters are set higher and, thus, can take more influence, the surface mesh gets increasingly more coarse, as it is visible in Figure 12.

Parameter Variation The aim of this experiment is to analyze the behavior of the different regularization terms at a constant σ. σ was set to 0.1 (10 cm) to see a clear effect of the parameter changes.

Figure 13 displays the changes of the surface from a central view-point, Figure 14 the depth error from a similar view-point and Figure 15 provides a closer look on the same changes from a different angle.

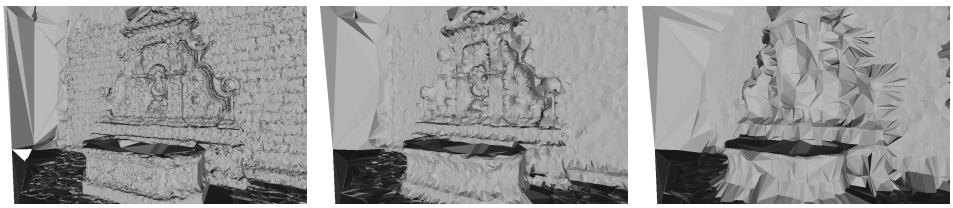
The visual inspection of the surface changes leads to the following two observations.

Firstly, the *area* term has a totally different behavior from the other two terms. It does not change the topology overly much, but instead crops the scene. It starts with the large triangles that have a low support, and then continuously shrinks the scene. If it gets too large, parts of the objects are



(a) $\sigma = 0.01$ (b) $\sigma = 1$

Figure 11: *Soft-Visibility Experiment* 4.1.2: With increasing σ the number of artifacts on the surface is rising if a small regularization parameter is used (here: 10^{-20}).



(a) $\sigma = 0.01$ (b) $\sigma = 0.1$ (c) $\sigma = 1$

Figure 12: *Soft-Visibility Experiment* 4.1.2: The effect of increasing the σ value at a fixed parameter of 1 for the *constant* regularization term.

missing and this leads to a fragmented reconstruction.

Secondly, the *constant* and the *beta skeleton* term make the model subsequently more coarse and simple. They reduce the overall number of facets. It might not be obvious from the mesh images, but the difference images in Figure 14 show that both approaches move the mesh further away from the real location of the surface. A large triangle, on e.g. the wall sections, lies *in front of* the small triangles. You could describe the process, as subsequently putting increasingly thicker cloth around the object. The gravity pulls it towards the object surface. Thus, a small parameter corresponds to a thin piece of cloth, which does not change the reconstruction very much. On the other hand, a large parameter corresponds to a thick cloth, which covers the details of the scene.

This behavior is only natural, as large triangles in the Delaunay triangulation are, in general, not located on the surface. The surface is more likely to be made of very small triangles, as the point cloud should have a higher density on the object surface. Thus, the only large triangles available for the

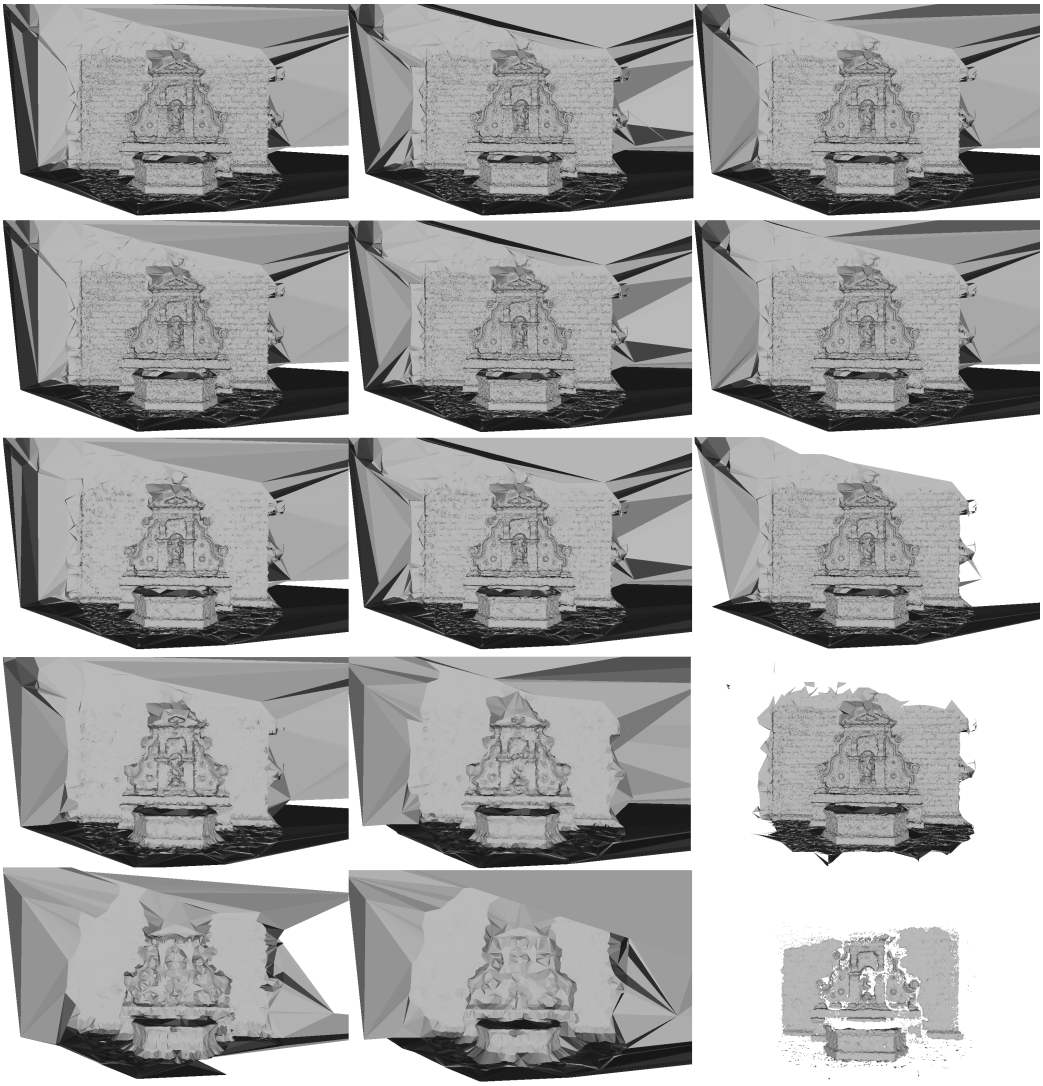


Figure 13: *Soft-Visibility Experiment* 4.1.2: Surface topology changes with increasing parameters from top to bottom. From left to right: *constant*, *beta skeleton* and *area* regularization terms.

optimization are the ones that connect to points farther away.

Figure 16 shows the effect of varying the parameters in terms of accuracy. It is apparent that higher parameter values cause the accuracy to decrease. Once again, the *constant* term turns out to be the only term that can increase the accuracy compared to the unregularized version. The *constant* and *beta skeleton* show a very similar decrease in accuracy, whereas the *area* term shows a totally different behavior.

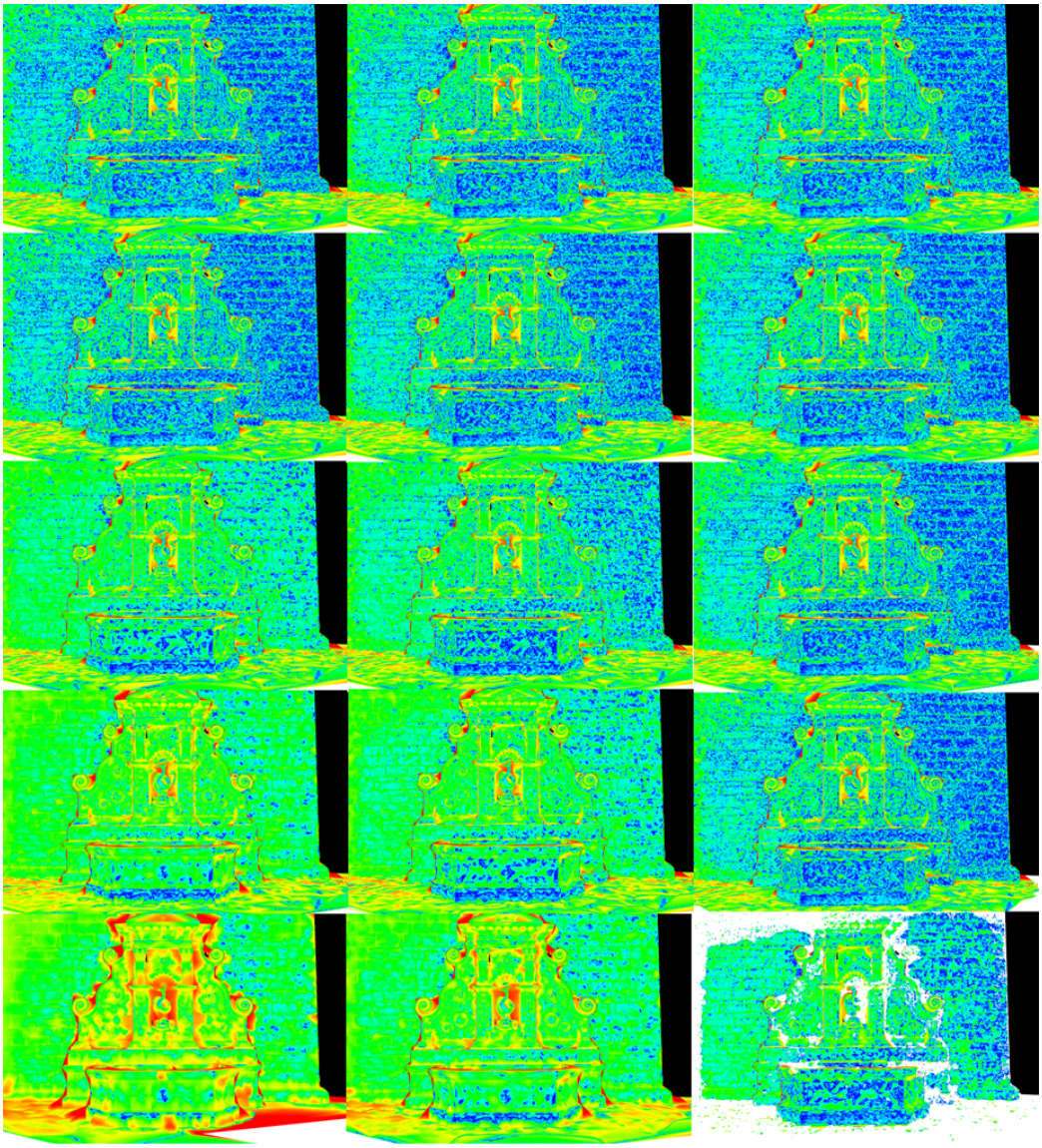


Figure 14: *Soft-Visibility Experiment* 4.1.2: Depth differences of camera 5 with increasing parameters from top to bottom. From left to right: *constant*, *beta skeleton* and *area* regularization terms. Blue means low error, red high error, black not covered by the ground truth and white not modeled by the query mesh.

Sink Link Variants In [26] Labatut et al. proposed to create a sink link directly from the cell behind the measurement and, thusly, vote for this cell to be inside. In [27], on the other hand, they proposed to create the link one cell later. Both approaches are depicted in Figure 3. Especially, the

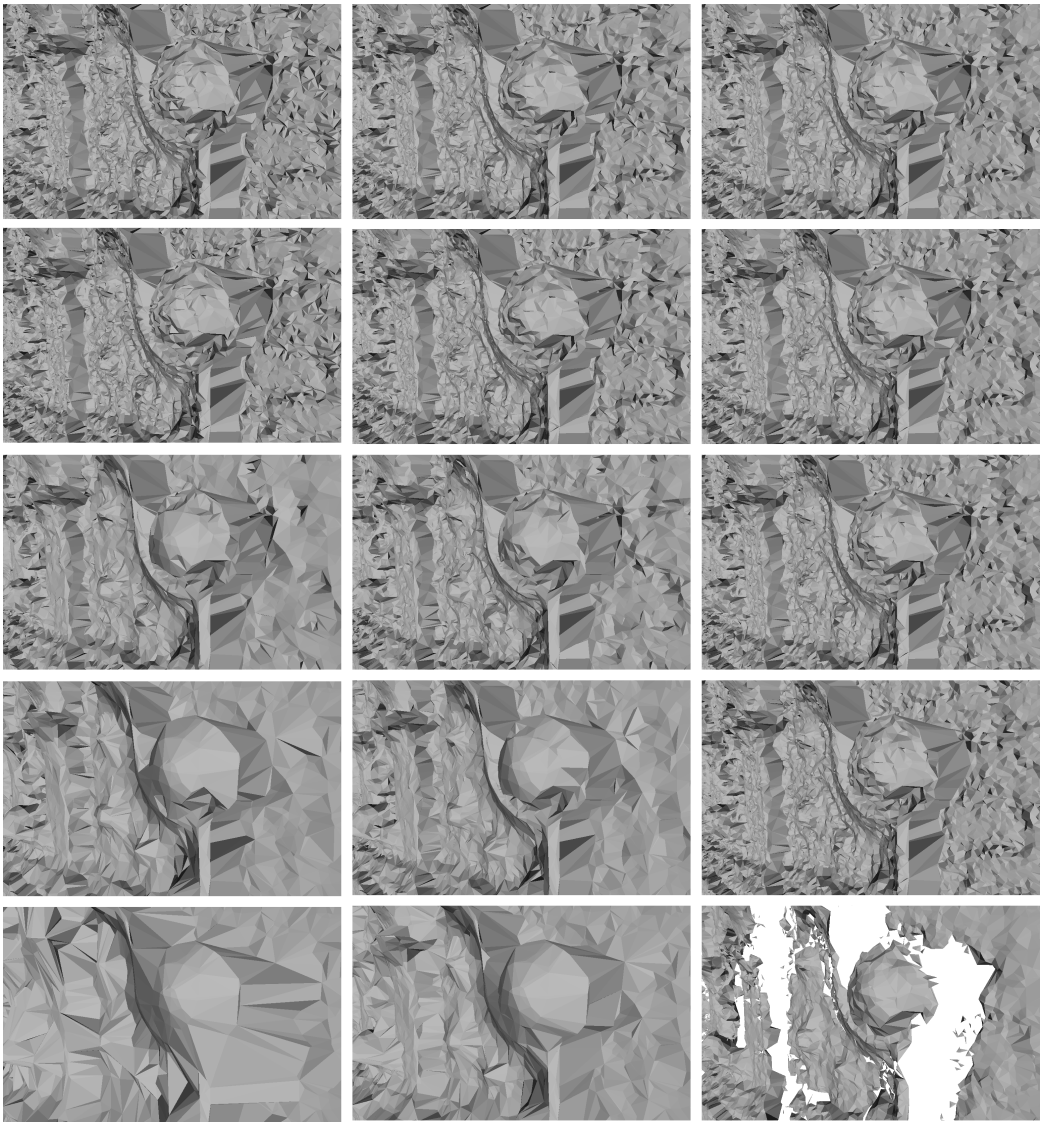


Figure 15: *Soft-Visibility Experiment* 4.1.2: Surface topology changes with increasing a parameter from top to bottom. From left to right: *constant*, *beta skeleton* and *area* regularization terms.

soft approach, which does not seems to be very intuitive, motivated us to investigate these strategies in terms of accuracy. Furthermore, we also tried to add multiple links which were weighted with one minus the facet weight in a region of great confidence. The region of great confidence was set to be within $\sigma_{est}/2$, where σ_{est} is the estimated width of the Gaussian blur kernel that represents the scene noise (See Section 3.4).

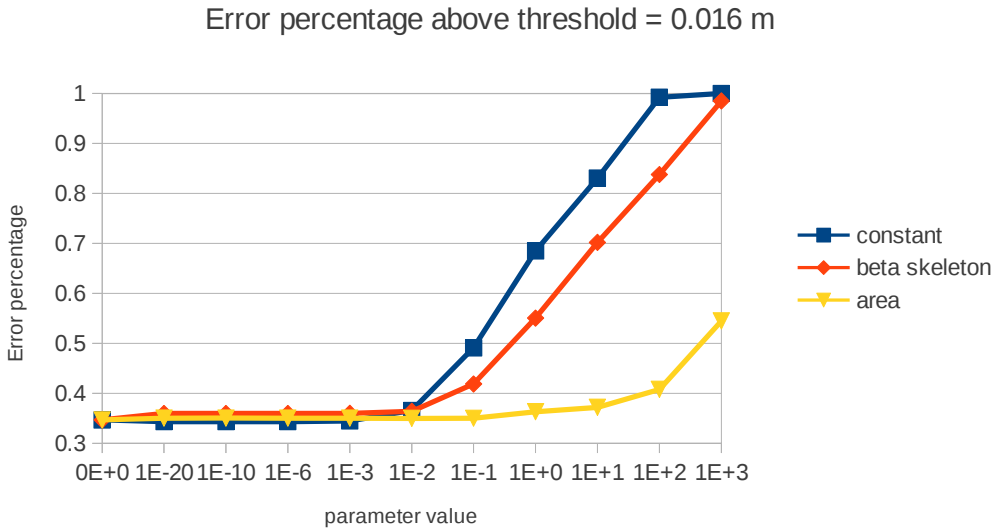


Figure 16: *Soft-Visibility Experiment* 4.1.2: Percentage of pixels with an error greater than a threshold of 0.016 m at increasing parameter values.

As an experiment, the behavior of all approaches to an increasing parameter of the *constant* regularization term were analyzed. The experiment showed, that the link creation process only influences the optimization mildly. From Figure 17 it is apparent, that the simplest approach with a link directly behind the measurement (*single direct link*) is the most accurate one. The approach, that created the link one cell later (*single late link*) is slightly less accurate and the approach with *multiple links* turned out to be the worst of the three. In fact, it turned out that the larger the region of great confidence was set, the worse the approach performed.

4.1.3 Comparison and Conclusion

In a direct comparison of the both approaches (see Figure 18), the hard visibility performs slightly better than the soft version. This is because the σ parameter was set too high and some artifacts in non-convex regions of the surface appear. If the σ parameter is set to the estimated noise level ($\sigma = 0.000828666$), there is no difference between the two approaches. In fact, the hard version is the limiting case of the soft version with the σ parameter equal to zero. In such a scene with a very low noise level, the soft version cannot improve the accuracy, but easily worsen it.

Both approaches, the one with a hard visibility constraint and the soft

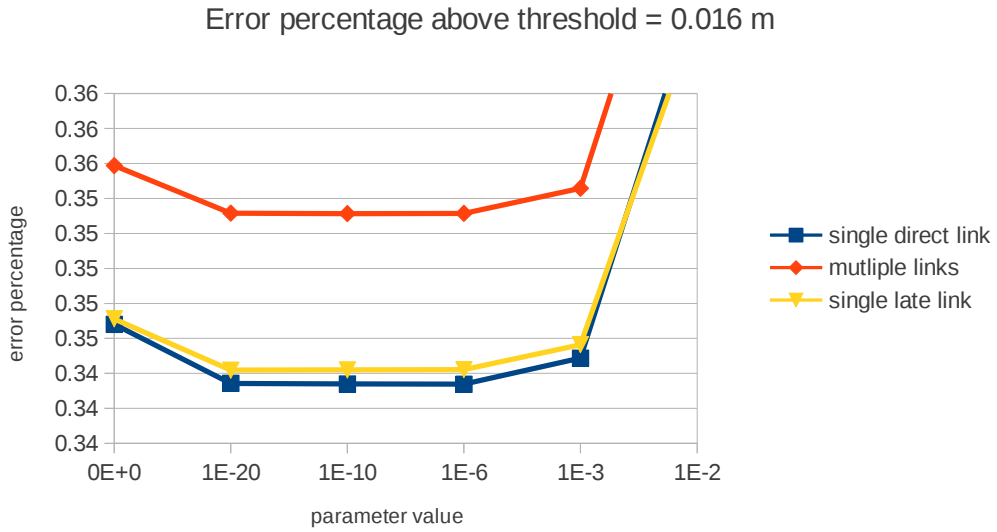


Figure 17: *Soft-Visibility Experiment* 4.1.2: Percentage of pixels with an error greater than threshold = 0.016 m at increasing parameters.

Figure 18: Error distribution histogram of the hard and soft version. For both versions, a parameter value of 10^{-20} for the *constant* regularization term was used and a single sink link was created behind the measurement. The σ parameter of the soft version was set to 0.1 to see a difference.

one, show that the best accuracy is reached, if the regularization parameters are kept small. Basically, they can be epsilon and are only needed to activate all edges in the graph.

On both approaches, the *constant* regularization term outperforms the other two terms. It behaves very similar to the *beta skeleton* term, but has a higher accuracy and no computational cost. It turns out, that the *constant* term is the only one to increase the accuracy in comparison to the unregularized version.

The *area* term has the property of not influencing the surface topology overly much, but instead crops the scene to the parts with the most support. If the corresponding parameter is set with care, it can get rid of large triangles that are connected to far-away outliers. If it is set too large, it can lead to a fragmented output mesh.

For all parameters it is true, that if they are set too high the optimization fails as it is then cheaper to label all cells as outside than to cut through the triangulation.

4.2 Noise Robustness

Nearly all fields of computer vision have a great need for robustness against noise. This is due to the imperfection of data, e.g. digital images have a limited degree of detail they can capture. Furthermore, all steps in the reconstruction chain can induce a certain amount of noise. One can, in general, split the noise which is present in this kind of point clouds into two types.

The first type is the Gaussian noise. In this case, it means that the position of samples in the point cloud is slightly wrong, compared to the real world location. Gaussian noise can originate from the imperfection of the original image or simply that the feature detector or the densification procedure can produce points that are slightly off the real position.

The second type are outliers. Opposed to the Gaussian noise, these samples do not have correspondences in the real world. They originate mostly from wrong feature matches between images. Although, one tries to get rid of outliers in nearly every step along the reconstruction chain, in practice, a small amount of outliers is always present.

To analyze the robustness of the approach against these two types of noise, we artificially added noise to the model. The remaining part of this subsection will give details about the strategy of adding noise and the measured performance of the implemented graph cut approach as well as a comparison to the Poisson reconstruction implemented in MeshLab [42], which had to be manually cropped to make a fair evaluation possible.

4.2.1 Outliers

It was observed that outliers mostly have a small number of 2D image correspondences. In practice, it turns out that most outliers have only two correspondences, far less have three, and there are nearly no outliers that have four or more correspondences.

Each added wrong correspondence is a wrong measurement, a wrong vote, thus, it is harmful for the reconstruction. Consequently, to make it harder rather than easier, we decided to assign three cameras on average to each outlier. In fact, the number of added cameras is equally distributed between 2, 3 and 4. The cameras themselves are chosen randomly as well.

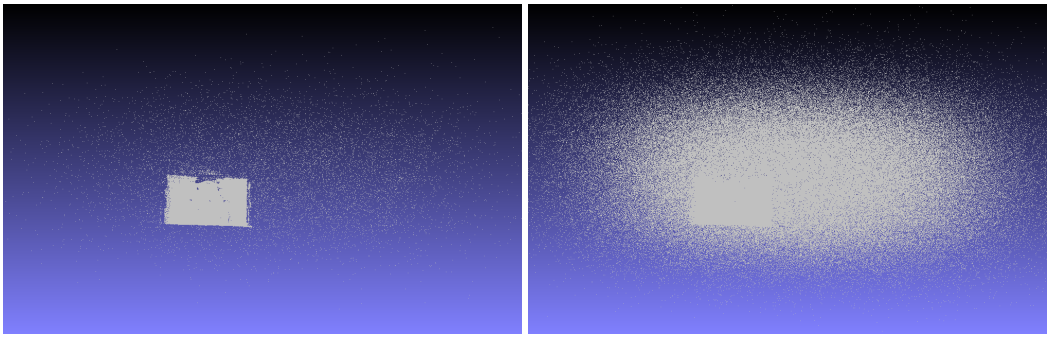
The outliers are added corresponding to the scene bounding box in pseudo uniform manner. Would the outliers be distributed uniformly within the bounding box, this would cause two problems. First of all, the outliers could not lie outside the bounding box, which is rather unrealistic. Secondly, the outliers would form a cube around the object. This cube would have very distinct faces, which could "rightfully" be detected as planes by the algorithm. To overcome both problems at once we propose to additionally induce a noise with a normal distribution to the outliers.

In this evaluation we distributed the outliers uniformly inside the scene bounding box and added a Gaussian noise with a kernel width of a fourth of the bounding box size. Like this, the outliers form some kind ellipsoid around the object which has a high density close to the object and fades out the farther away the points are from the bounding box.

Experiment For the evaluation of outlier robustness, only the hard version of the implemented graph cut approach is compared to the Poisson reconstruction. In this experiment we subsequently add more outliers to the original model of 100k points. Figure 19 depicts snapshots of the point cloud at 10% and 400% outliers. Note that, at 400% outliers only approximately 10k outliers are in the tight bounding box around the dense parts of the fountain model as the scene is very large.

The Poisson reconstruction was generated with MeshLab [42]. As the Poisson reconstruction needs the normals of the surface, those were calculated with MeshLab as well and flipped towards a frontal viewing direction.

In Figure 20 one can see the surface topology evolution in the presence of an increasing number of outliers. The Poisson reconstruction always yields closed bubble-like surfaces, thus, it is necessary to crop the scene to allow for a fair comparison. On the right side the cropped meshes are presented. We tried to keep the dense parts of the mesh intact, and remove only the parts which would lead to occlusions in the evaluation process.



(a) 10% outliers. (b) 400 % outliers.

Figure 19: *Outlier Experiment* 4.2.1: Snapshots of the scene at different levels of outliers.

The experiment shows that the implemented graph cut approach is highly robust against outliers. As it is visible on the left column of Figure 20, the surface of interest remains untouched in the presence of the increasing number of outliers. Solely, the surrounding areas, which only have a few inliers (if any), are influenced by the outliers. On the other hand, the Poisson reconstruction is severely effected by the outliers. With an increasing number of outliers the scene degenerates more and more.

Figure 21 covers the surface evolution in terms of accuracy. In the top row one can see the full cumulative error distribution. The Poisson reconstruction strongly loses accuracy with the increasing number of outliers at all thresholds, whereas, the implemented graph cut approach only suffers a slight loss at the higher thresholds. This is mainly due to the effects of the outliers in the regions of low support at the edges of the model.

Note that, the general gap of accuracy between the approaches is not important in this evaluation as the accuracy of the Poisson reconstruction could be improved with a larger tree depth. This evaluation is more concerned with the behavior of the approaches in the presence of outliers.

In the bottom row, one can see the percentage of pixels above the thresholds of 0.016 m and 0.512 m. Note that, the scale of the x-axis from 0.25 to 4 is an exponential one. Both approaches, show a linear ascent on the exponential scale at threshold of 0.016 m. This means, that they show a logarithmic behavior of the error to the number of outliers. Note that, the ascent of the error of the Poisson reconstruction is a decade higher than the ascent of the implemented graph cut approach (approximately 0.05 to 0.005). At a threshold of 0.512 m the Poisson reconstruction even has an exponential error curve, this means that the severe errors show a linear ascent to the number of outliers. The implemented graph cut approach, on the other

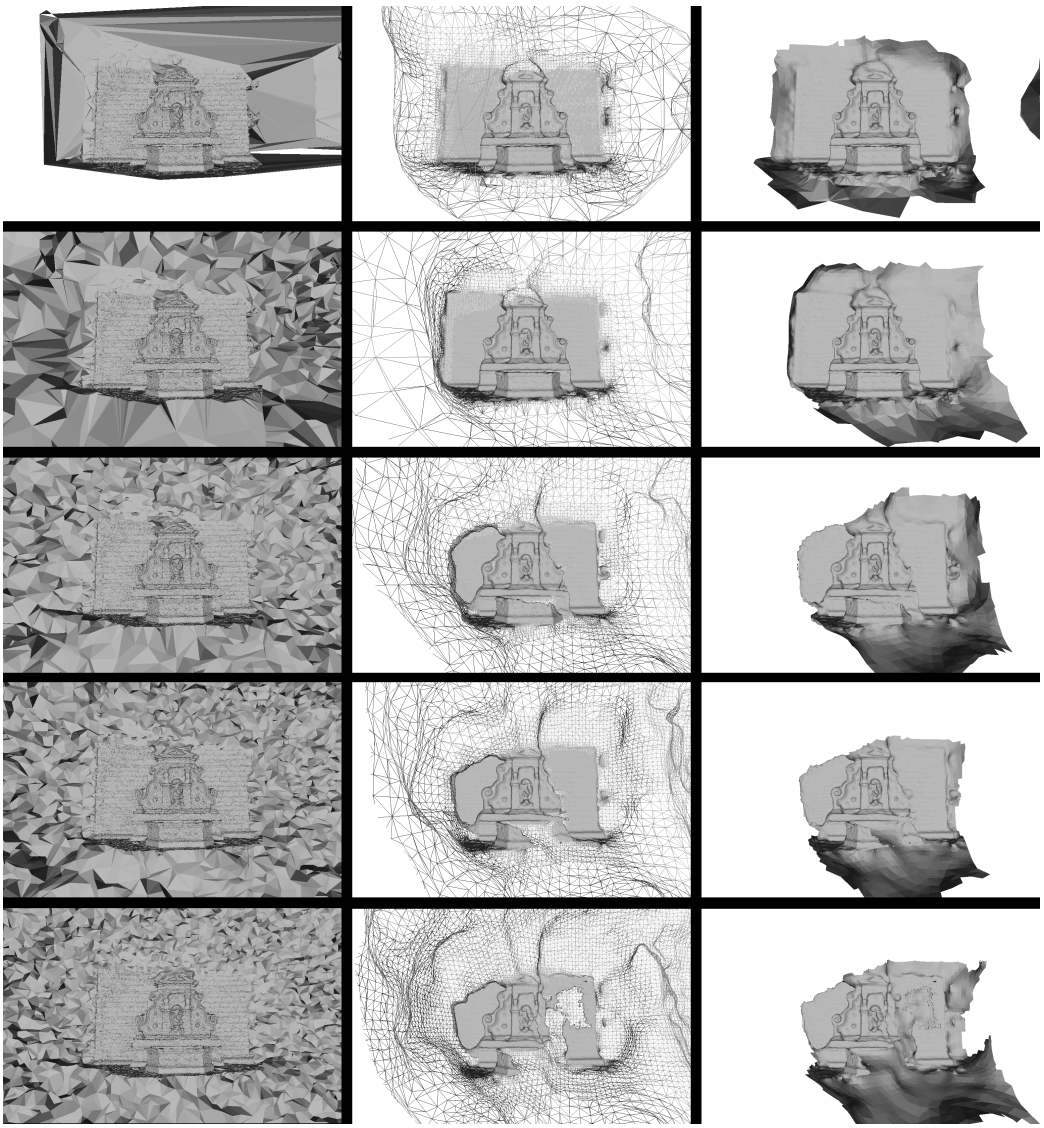


Figure 20: *Outlier Experiment* 4.2.1: Surface changes at an increasing number of outliers (from top to bottom). On the left: The implemented graph cut approach with a hard visibility term and a parameter of 10^{-20} for the *constant* regularization term. In the middle: Uncropped wire frame of the Poisson reconstruction. Note that, the bubbling effects would occlude the scene in the top most image. On the right: Manually cropped Poisson reconstruction, which was used for the accuracy evaluation.

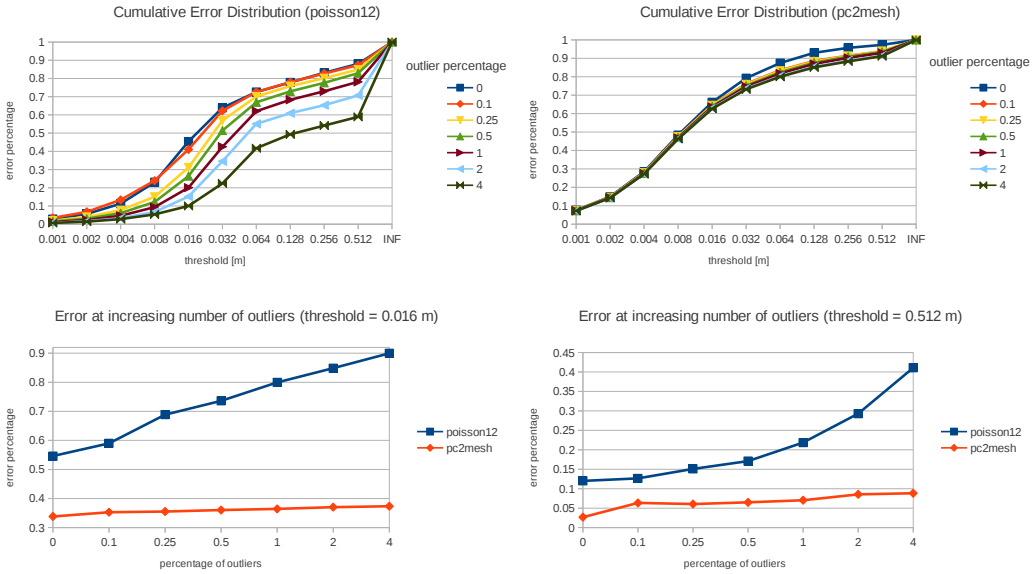


Figure 21: *Outlier Experiment* 4.2.1: Top: Cumulative error distribution of the Poisson reconstruction (left) and the implemented graph cut approach with a hard visibility term and a parameter of 10^{-20} for the *constant* regularization term (right). Bottom: Percentage of pixels of both approaches above a threshold of 0.016 m (left) and 0.512 m (right).

hand, is only vaguely influenced by the outliers.

4.2.2 Gaussian Noise

Adding Gaussian noise to the point cloud is far easier than adding outliers as no new points nor camera links have to be created. The Gaussian noise is generated in simply moving the 3D point in a random direction for a random distance. This is achieved in adding a sample of a normal distribution to each coordinate of the 3D point. The camera links and image measurements remain untouched.

Automated Sigma Estimation It is hard to evaluate the quality of the automated scene noise estimation, due to the unavailability of a dataset with known noise levels. The implemented graph cut approach assumes that the distance from the cameras to the 3D points is nearly in the same range for all points and cameras.

In Figure 22 one can see the relation between the induced noise and the median reprojection error. This median reprojection error was normed to

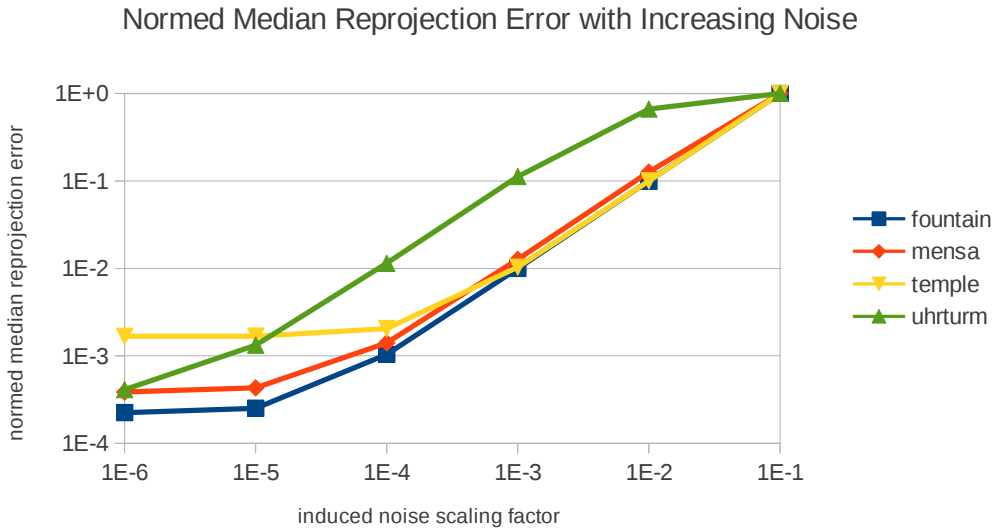


Figure 22: *Gaussian Noise Experiment* 4.2.2: Relation between median reprojection error and the induced noise level for different scenes. For a better comparison the median reprojection error has been normed to reach 1 for the highest considered noise level. The *induced noise scaling factor* s_σ defines the width of the Gaussian blur kernel σ in terms of the scene diameter: $\sigma = s_\sigma \cdot$ scene diameter.

reach 1 at the highest noise level. At a low level of induced noise the real scene noise is much larger than the induced noise and dominates the curves. At a certain point the induced noise starts taking over and from there on, the median reprojection shows a linear behavior over a long period. The "uhrturm" dataset has a huge scene around the object of interest. Thus, the induced noise is very large even at the smallest scaling factor. One can observe, that if the induced scene noise gets too large the linear behavior ceases. In this case, the noise gets so large that the 3D points even move behind the cameras, which causes the estimation to fail.

This work exploits the linear behavior between median reprojection error and artificially induced noise to estimate the real noise level. To this end, the ratio between median reprojection error and the induced noise is measured at a level of $\sigma = 0.001 \cdot$ scene size. Note that, opposed to the Figure 22, the reprojection error is measured between the ideal 2D reprojection point before applying noise and after applying noise. Consequently, the estimation of the ratio is independent of the real scene noise and only depends on the scene geometry.

It is assumed that the real noise influences the median reprojection error in the same way the artificial noise does. Under this assumption the median reprojection error can be directly used to estimate the real scene noise.

Soft Visibility Regularization Term Evaluation As the soft visibility constraint was supposed to be an improvement for Gaussian noise and the output of the PMVS2 approach [18] only contains a very low noise level, we also evaluated the quality of the terms at a higher artificially induced noise level of a few centimeters ($\sigma = 0.001 \cdot$ scene diameter $= 0.0368651$ m)

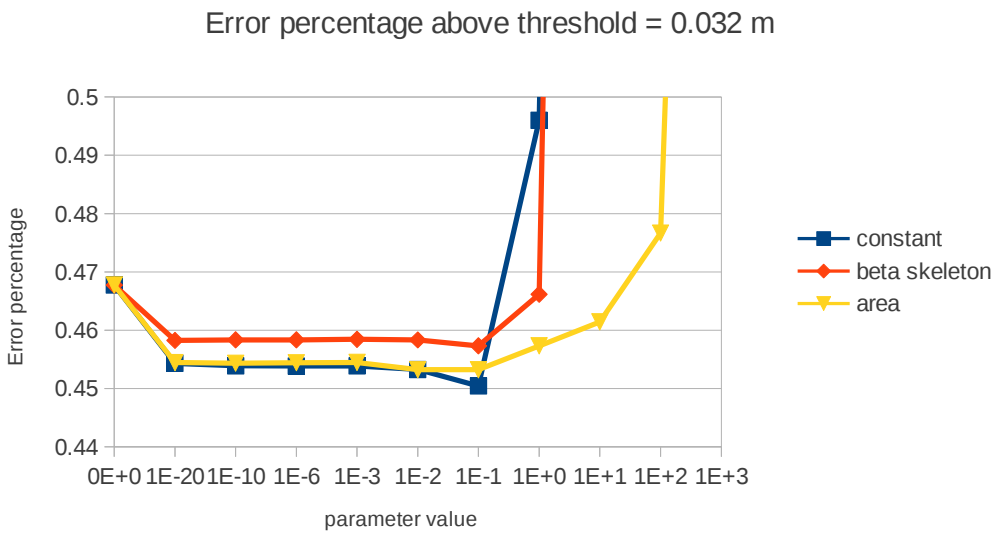


Figure 23: *Gaussian Noise Experiment* 4.2.2: Percentage of pixels with an error above the threshold of 0.032 m at an artificial noise level of $\sigma = 0.0368651$.

In Figure 23 one can see the error curve of the varying parameters below a threshold of 0.032 m. The experiments show that the terms show the same behavior in regard to quality as in the case of nearly no noise. The simple *constant* term outperforms the other two and the *beta skeleton* term turns out, once again, to have the least accuracy.

In this experiment, unlike the case of low noise, there appears to be a dip in the error curves around the parameter value of 0.1. It seems like, if the regularization terms influence the optimization slightly in the case of high Gaussian noise, they can improve the result. But it is also apparent, that the benefit of the terms is very unstable and if chosen a decade higher, can drastically worsen the accuracy. Furthermore, the experiment shows that in

the presence of higher Gaussian noise all terms can improve the accuracy compared to the unregularized version.

Performance Evaluation Four different approaches were chosen for comparison to evaluate the performance in the presence of noise.

First of all, we chose the soft visibility version with the regularization term and parameter that reached the highest score in the previous paragraph; a parameter of 0.1 for the *constant* regularization term. The σ parameter is automatically tuned to the estimated noise level.

Secondly, the hard visibility version is represented with a parameter of 10^{-20} for the *constant* regularization term.

Thirdly, we decided to apply simple posterior smoothing to the hard visibility version as described in 3.5.1. The aim was to evaluate the power of the soft visibility version, which should improve the performance in the presence of Gaussian noise, in comparison to a simple averaging smoothing step.

Finally, we compare our approaches to the Poisson reconstruction that is implemented in MeshLab [42]. To reach the full potential of the Poisson reconstruction we used the normals of the PMVS2 output as input for the Poisson reconstruction. Let it be noted at this point, that if the estimation of the normals of the Poisson reconstruction is bad in parts of the scene, this nearly always causes it to fail. Failing in this case means, that the bubbling effects, to which this approach tends, are not only restricted to the outside of the object, but basically split the object in several parts. In copying the normals from the PMVS2 output, the Poisson reconstruction has nearly ideal estimations of normals at all levels of noise (even at a sigma of 36 cm). As such good normals can hardly be realized in practice at higher levels of noise, this approach is denoted as "oracle poisson" in the diagrams.

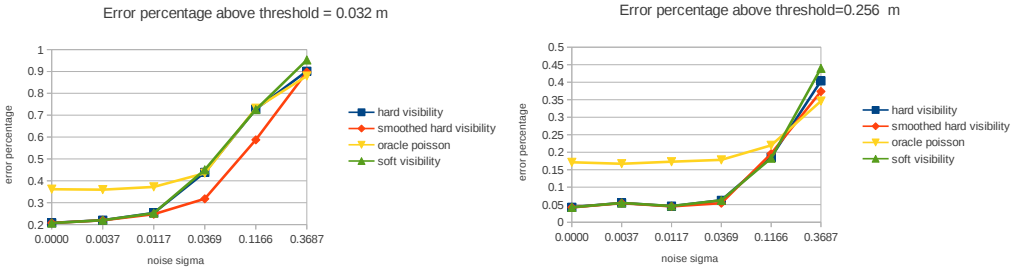


Figure 24: *Gaussian Noise Experiment* 4.2.2: Percentage of pixels that have a depth error above a threshold of 0.032 m (left) and 0.256 m (right) at an increasing width of the Gaussian noise kernel.

In Figure 24 one can see the error percentage of the different approaches at an exponentially increasing noise level. A pixel is treated as wrong if the difference to the ground truth is larger than 0.032 m or 0.256 m.

From this comparison, we reason several things. First of all, there is no significant performance difference between soft and hard visibility constraint. This is, of course, subject to setting the parameters in the soft version with utmost care.

Secondly, the initial gap in the accuracy between the implemented graph cut approach and the Poisson reconstruction is eliminated at higher levels of Gaussian noise (from a few centimeters onwards). This was to be expected, as it can interpolate between the sample points and the tendency of the Poisson reconstruction towards smooth surfaces pays off.

Thirdly, the chart shows that a posterior smoothing step can significantly improve the performance of the implemented graph cut approach at higher levels of noise. At a threshold of a few centimeters (0.0369 m and 0.1166 m) the error percentage can be reduced by approximately 14%.

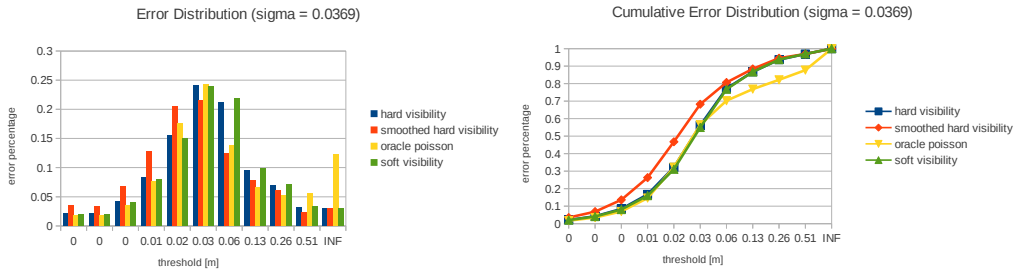


Figure 25: *Gaussian Noise Experiment* 4.2.2: Left: Error distribution histogram at a constant $\sigma = 0.0369$. Right: Cumulative error distribution curves at a constant $\sigma = 0.0369$.

Figure 25 shows the full error histogram and the corresponding cumulative curve at a constant noise level of $\sigma = 0.0369$. This noise level was chosen, because it is still possible to make out most of the details, but the effects of the noise are clearly visible.

Firstly, it is apparent that the hard version shows clearly a higher accuracy in fine details compared to the soft version. Secondly, the chart shows that the posterior smoothing step can drastically improve the performance. Thirdly, the histogram shows that also the Poisson reconstruction performs very well, and can equally well capture smaller details. On the other hand, the Poisson reconstruction suffers from a large infinite error, which is due to the bubbling effect of the approach. This behavior causes large errors at the edges of the model.

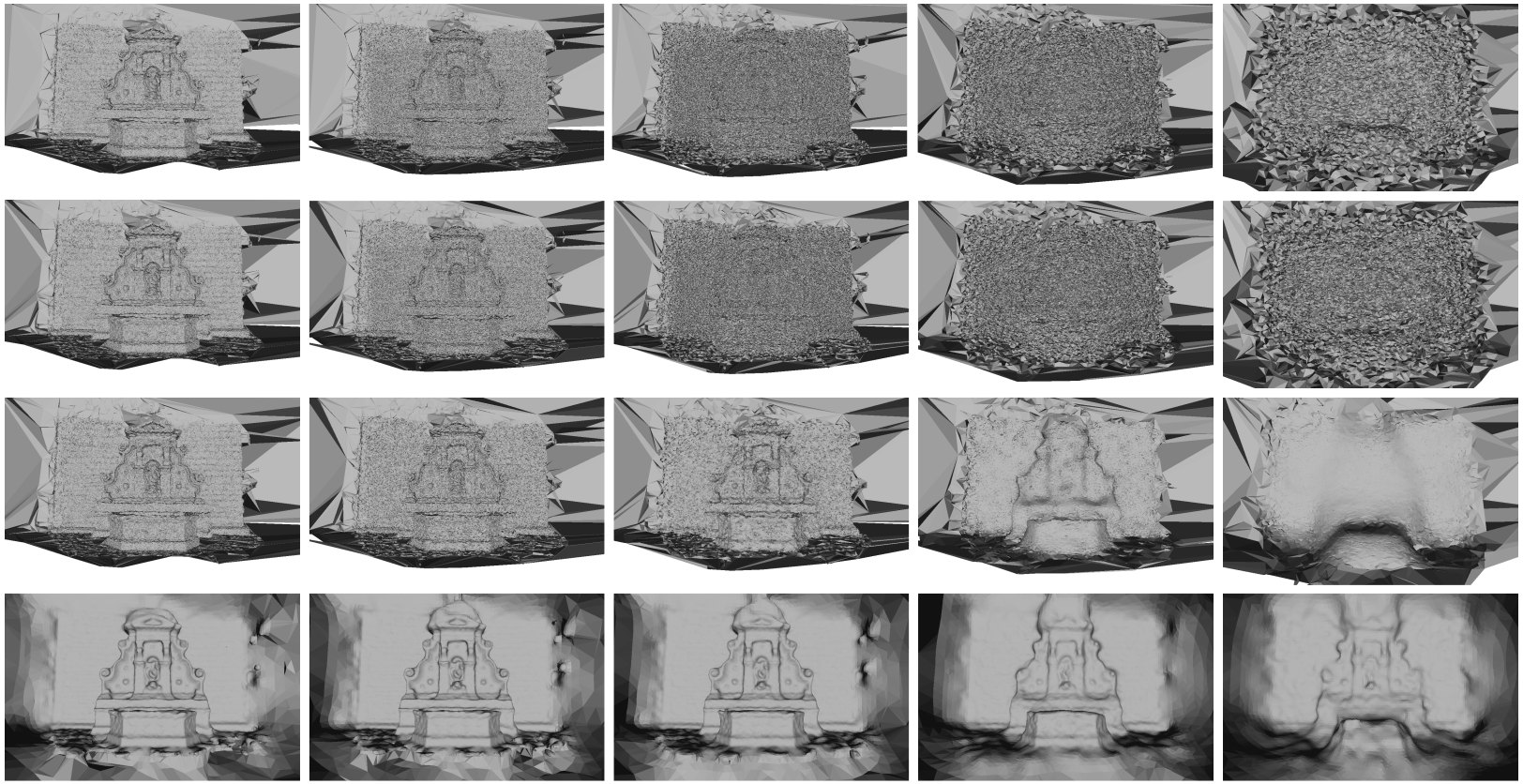


Figure 26: *Gaussian Noise Experiment* 4.2.2: 4 different approaches at an increasing Gaussian noise level (from left to right). From top to bottom: *soft visibility*, *hard visibility*, *smoothed hard visibility*, *oracle poisson*.

A visual comparison can be found in Figure 26. It is apparent, that there is no real difference between soft and hard version of the implemented graph cut approach at the chosen parameters. Visually, the Poisson reconstruction yields the best results, due to its interpolation and smoothing properties. But as discussed above, in matters of accuracy the Poisson reconstruction is inferior to the posterior smoothed version of the implemented graph cut approach. As one can see, the Poisson reconstruction is very good in the center of the model, but has severe problems at the margins where it already starts closing the bubble too early. Additionally, the Poisson reconstruction has problems in parts, where only a few feature points are present, such as the floor. These parts are not reconstructed properly, which leads to a decrease of the overall performance.

Conclusion We have shown that the accuracy of the implemented graph cut approach in the presence of Gaussian noise cannot be improved with a soft visibility constraint as proposed in [27]. The soft visibility constraint can be used to simplify the surface, but only at the cost of moving the surface further away from its true location. Instead of interpolation, the approach creates some kind of buffer around the true surface. A graphical explanation of this effect can be found in Figure 27.

Consequently, if for some reason the point cloud contains a high level of Gaussian noise, another step in the reconstruction chain is needed to get rid of it. We showed that a simple posterior smoothing step can significantly improve the accuracy in the case of a high level of Gaussian noise. But it may be possible to get similar improvements with a noise reduction step prior to the graph construction. The Poisson reconstruction, albeit other limitations, is better suited to compensate Gaussian noise, because it is not restricted to the given samples, but can interpolate between them. Nevertheless, we have shown that a simple additional smoothing step can overcome the lack of interpolation properties of the implemented graph cut approach.

4.3 Resource Consumption

The evaluation of the resource consumption is split into two parts. Firstly, the runtime is analyzed in relation to the number of points. Secondly, the memory usage of the most important data structures was captured at a fixed number of points.

It should be noted at this stage, that the main aim of this work was to provide an implementation, which is highly adaptive in regard to regularization terms and parameters. To cater for the adaptability, sufficient information for all possible regularization terms needs to be stored.

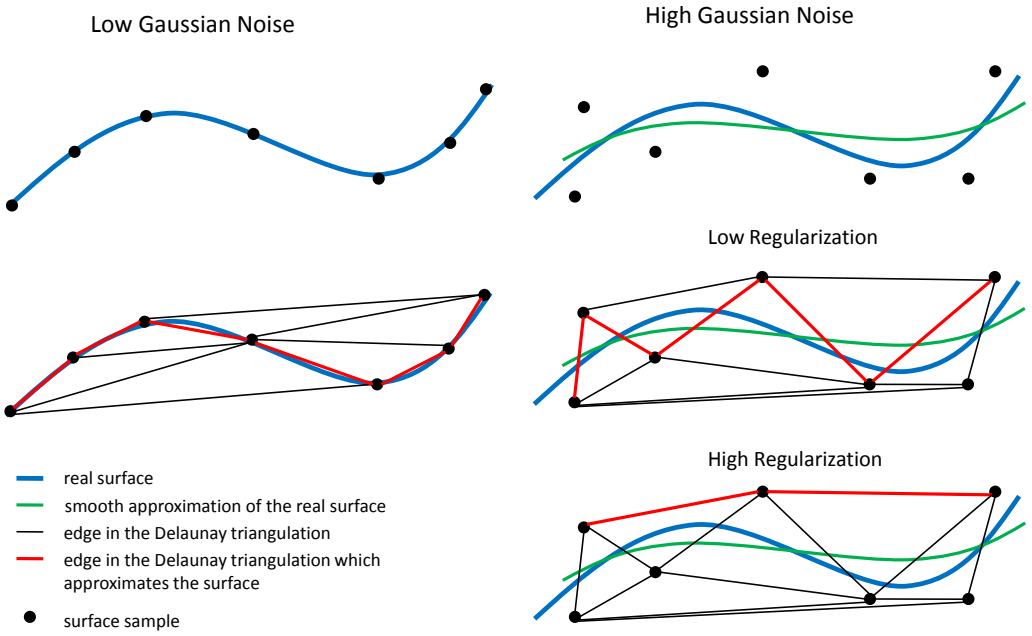


Figure 27: Graphical explanation for the fact that Delaunay triangulation based approaches have problems with Gaussian noise. As one can see on the left side, the Delaunay triangulation contains a good approximation of the surface if the noise is low compared to the sampling density. The right side displays the case of a high level of Gaussian noise. The output surface has to be a subset of the Delaunay triangles in 3D, in this two-dimensional case a subset of Delaunay edges. If the noise level is too high compared to the sampling density, the Delaunay triangulation does very likely not contain a good approximation of the surface. Instead of a smooth interpolation, which would maximize the probability of finding a good approximation, the Delaunay based approaches try to find a surface in the discrete set of facets in the triangulation. This leads to either a wavy/spiky surface, if a low regularization is used, or a simpler/smoother surface which, unfortunately, is far away from the real surface.

To facilitate experiments with the parameters, the whole state of the graph structure is serialized to the file system with the aid of the boost libraries [40]. To achieve such a serialization the data structure has to be kept simple and pointers should only be used with utmost care.

4.3.1 Runtime Analysis

The runtime was analyzed on the "fountain" scene of Strecha et al. [37]. The experiments were conducted on a notebook of the type Acer TravelMate 8572G. Its CPU is a Intel Core i5, dual core processor @ 2.4 GHz, it has 4GB of RAM and a GeForce GT330M for graphics. Note that, the current implementation operates on a single core only. The used operating system is the 64-bit version of Ubuntu 11.10.

On the specified machine a surface reconstruction of the "fountain" scene (with 100k points and 11 cameras) takes 386.131 seconds.

In Figure 28 on can see the runtime evolution with an increasing number of points and a pie chart showing the percental consumption of the total runtime.

For the analysis the program was split into four parts. The first one is the *cell graph construction*, which consists of the construction of Delaunay triangulation and the corresponding pseudo-dual graph for the energy minimization. The second one, called *visibility terms*, is the ray tracing task, which updates all facets with a ray conflict and the sink links. The third one (*max flow min cut*) is the energy minimization task via graph cuts and the minimum cut extraction. The last one, *I/O and data conversion*, is the rest of the application. This part is mainly dominated by I/O bound tasks, such as input file reading and de/serialization of the internal structures.

It is apparent that nearly all parts of the implementation show a close-to-linear behavior to the number of points. Solely, the steepness of the *I/O and data conversion* part, which is dominated by I/O tasks, seems to rise slightly with the increasing number of points.

It is apparent, that the most expensive part is the ray tracing task. It consumes more than 90% of the overall runtime. The energy minimization itself is very cheap compared to all other parts with less than 1%. Note that, the ray tracing was also computed on the CPU.

Optimization Suggestions The most expensive task is clearly the ray tracing. One could drastically improve the overall performance of the approach in porting the ray tracing task to the GPU. This task can be highly paralyzed as thousands of rays have to be traced independently. On the other hand, care has to be taken in regard to race conditions on the weight data. Independent rays may want to update the weight of the same facet. Thus, it might be hard to reach linear speedup without linearly increasing the memory of the facet weights.

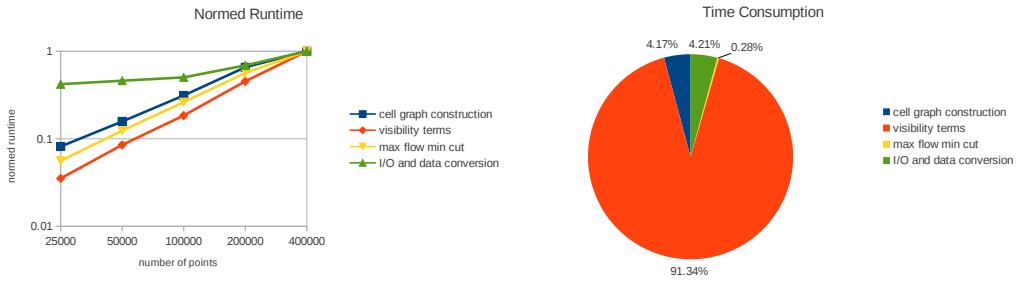


Figure 28: *Resource Consumption Experiment* 4.3: Left: Normed runtime of the different parts of the implementation. The runtime was normed to reach 1 at the highest value. Right: Percental consumption of the total runtime.

4.3.2 Memory Usage

The memory usage was analyzed on the "Herz-Jesu-P8" dataset of Strecha et al. [37]. The dataset contains 8 images of a building facade. The combination of SfM output and PMVS2 output yields 299 900 points. The Delaunay triangulation contains 3 628 962 facets and 1 814 463 cells. The memory usage was measured with the System Monitor of the Ubuntu operating system. To determine the usage of certain parts of the program, the program was sent into an infinite loop at different program states, before and after the construction of certain data structures. Table 1 contains the results of this experiment.

Data Structure	memory usage (in GB)
Points & Cameras	0.6
Delaunay Triangulation	0.1
Graph Vertices	0.1
Graph Facets	0.4
Graph Cells	0.2
Ray Casting Triangles	0.3
Overall Peak	1.8

Table 1: *Resource Consumption Experiment* 4.3: Approximate memory usage of different data structures of this implementation at 300k input points.

The peak memory is approximately 6 times higher than in the implementation of Labatut et al. [27], which report a usage of only 309MB for a model with 360k points. It is apparent that one of the largest data structures are the *Points & Cameras*, which is already provided by the I3D library. It contains the SfM points, links and cameras as well as the PMVS2 points and

links. This data structure alone is 2 times bigger than the overall memory usage of Labatut et al. [27].

Optimization Suggestions This implementation carries a lot of overhead in the shape of data, which is need for regularization terms. As it turns out, all those different regularization terms are inferior to a simple *constant* regularization. Thus, the memory usage of the graph structure can be reduced by approximately 20% in removing the unnecessary overhead. But this would only mean a reduction of approximately 150MB in the case above. In the current implementation, data had to be duplicated to make use of different frameworks, e.g. the facets had to be transformed to triangles to be used for the ray tracing task. In removing this redundancy one can easily gain another 15% in total.

5 Conclusion and Future Work

In this work we implemented and evaluated a keypoint-based approach for robust surface reconstruction from 3D point clouds using Delaunay triangulation, visibility information and a global optimization via graph cuts.

The implementation provides a wide range of modes in regard to possible regularization terms, experimental features (e.g. noise generation), evaluation and output generation.

We evaluated the robustness against Gaussian noise as well as outliers. In the quest for a better noise reduction, we presented a way to estimate the Gaussian noise in a 3D scene. We demonstrated that the knowledge about the scene noise can be used to improve the surface reconstruction in a simple posterior smoothing step. This smoothing does only take effect if the ratio between scene noise and sampling density is large enough.

We have compared the most promising regularization terms; visually and analytically. Furthermore, the effect of these terms was analyzed in the combination with a hard and a soft visibility constraint. The soft constraint puts a lower penalty on ray conflicts closer to the corresponding measurement, in order to express the expected uncertainty of the measurement's location. The hard constraint, on the other hand, penalizes all ray conflicts equally.

The first evaluated regularization term penalizes the *area* of triangles. This term can be used to crop the scene and reduce it to the parts with the most support. If it is set too high, this will lead to a fragmented reconstruction.

The second evaluated term is the *beta skeleton* term proposed by Labatut et al. [27]. It wants to enforce large circumscribing spheres around the

tetrahedra adjacent to a facet.

Finally, we propose a simple *constant* regularization term, which is well-founded on the theory of the minimum description length.

The *beta skeleton* and the *constant* term can simplify/smooth the model if a soft visibility constraint is used. This simplification comes at the cost of moving the approximation further away from the real surface towards the cameras. This results in a thickened version of the original object.

In the case of a hard visibility constraint, only the *area* term shows the same behavior as in the case of the soft constraint, whereas varying the parameters of the other two terms has nearly no effect.

It was discovered that the complex *beta skeleton* term is inferior to the simple *constant* term. In fact, it was shown that the *constant* term is the only term, out of all tested ones, that has the power to increase the accuracy compared to the unregularized version (only visibility constraint without regularization).

Our experiments showed that the highest accuracy can be achieved with the hard visibility constraint and the *constant* regularization term with a very low parameter value (epsilon). It activates all edges in the graph without influencing the visibility constraint overly much.

It was discovered that the implemented graph cut approach can only compensate for Gaussian noise with an additional smoothing step. On the other hand, we came to the conclusion that it has a very high resilience to outliers, far higher than the compared Poisson reconstruction [25].

In future work, we hope to improve the resource efficiency in removing unnecessary overhead and porting the ray tracing task to the GPU. Additionally, the overall computing time of the reconstruction chain could be improved if a different densification step was implemented. The PMVS2 approach [18] does a lot of outlier removal, which is very expensive. As the implemented graph cut approach can easily handle a high degree of outliers, a cheaper and less robust densification step, might yield the same quality at a far lower cost. Furthermore, it is our aim to develop a post-processing step which directly uses the input images to photometrically refine the extracted surface meshes to achieve more detail and even higher accuracy.

References

[1] P. Alliez, D. Cohen-Steiner, Y. Tong, M. Desbrun, *Voronoi-based Variational Reconstruction of Unoriented Point Sets*, Eurographics Symposium on Geometry Processing, 2007. 5

[2] N. Amenta, M. Bern, *Surface reconstruction by Voronoi filtering*, SCG, 1998. 6, 7, 9

[3] N. Amenta, M. Bern, D. Eppstein, *The crust and the β-skeleton: Combinatorial curve reconstruction*, Graphical Models and Image Processing, 1998. 15

[4] N. Amenta, S. Choi, T. Dey, N. Leekha, *A simple algorithm for homeomorphic surface reconstruction*, SCG, 2000. 7

[5] N. Amenta, S. Choi, R. Kolluri, *The Power Crust*, Proceedings of 6th ACM Symposium on Solid Modeling, 2001. 7

[6] D. Attali, J.-D. Boissonnat, *A Linear Bound on the Complexity of the Delaunay Triangulation of Points on Polyhedral Surfaces*, Discrete and Computational Geometry, 2004. 6, 9

[7] J.-D. Boissonnat, *Geometric structures for three-dimensional shape representation*, ACM Transactions on Graphics, 1984. 7

[8] Y. Boykov, V. Kolmogorov, *An Experimental Comparison of Min-Cut/Max-Flow Algorithms for Energy Minimization in Computer Vision*, PAMI, 2004. 18

[9] N. Campbell, G. Vogiatzis, C. Hernndez, R. Cipolla, *Multiple Hypotheses Depth-Maps for Multi-View Stereo* ECCV, 2008. 3

[10] F. Cazals and J. Giesen. *Delaunay based Surface Reconstruction algorithms: Ideas and Algorithm*, http://hal.inria.fr/inria-00070610 (checked May 2012), Research Report 5393, INRIA, 2004. 6

[11] F. Dellaert, D. Fox, W. Burgard, S. Thrun, *Monte Carlo Localization for Mobile Robots*, ICRA, 1999. 5

[12] O. Devillers , M. Teillaud. *Perturbations and vertex removal in a 3D Delaunay triangulation*, ACM-SIAM Sympos. Discrete Algorithms (SODA), 2003. 9

[13] T. Dey, S. Goswami, *Provable surface reconstruction from noisy samples*, Computational Geometry: Theory and Applications, 2006. 7

[14] D. Donoho. *De-noising by soft-thresholding*, IEEE Transactions on Information Theory, 1995. 18

[15] J.-M. Frahm, P. Georgel, D. Gallup, T. Johnson, R. Raguram, C. Wu, Y.-H. Jen, E. Dunn, B. Clipp, S. Lazebnik, M. Pollefeys, *Building Rome on a Cloudless Day*, ECCV, 2010. 3

[16] W. Freeman, R. Szeliski, S. Kang , *Noise Estimation from a Single Image*, CVPR, 2006. 18

[17] Y. Furukawa, J. Ponce, *Carved Visual Hulls for Image-Based Modeling*, ECCV, 2006. 3

[18] Y. Furukawa, J. Ponce *Accurate, Dense, and Robust Multi-View Stereopsis*, CVPR, 2007. 4, 8, 22, 23, 24, 25, 41, 50

[19] Y. Furukawa, J. Ponce, *Accurate, Dense, and Robust Multi-View Stereopsis*, PAMI, 2010. 4, 22

[20] G. Graber, T. Pock, H. Bischof, *Online 3D reconstruction using Convex Optimization*, ICCV, 2011. 5

[21] M. Goesele, N. Snavely, B. Curless, H. Hoppe, S. Seitz, *Multi-View Stereo for Community Photo Collections* ICCV, 2007. 3

[22] C. Hernndez, F. Schmitt, *Silhouette and stereo fusion for 3D object modeling*, Computer Vision and Image Understanding, December 2004. 3, 5

[23] C. Hernndez, G. Vogiatzis, R. Cipolla, *Probabilistic visibility for multi-view stereo*, CVPR, 2007. 3

[24] A. Hornung, L. Kobbelt, *Hierarchical Volumetric Multi-view Stereo Reconstruction of Manifold Surfaces based on Dual Graph Embedding* CVPR, 2006. 3, 5, 6

[25] M. Kazhdan , M. Bolitho, H. Hoppe, *Poisson Surface Reconstruction*, Eurographics Symposium on Geometry Processing, 2006. 3, 4, 6, 22, 50

[26] P. Labatut, J.-P. Pons, R. Keriven, *Efficient Multi-View Reconstruction of Large-Scale Scenes using Interest Points, Delaunay Triangulation and Graph Cuts*, ICCV, 2007. 2, 3, 5, 7, 9, 11, 12, 13, 14, 18, 31

[27] P. Labatut, J.-P. Pons, R. Keriven, *Robust and efficient surface reconstruction from range data*, CGF, Volume 28, Issue8 , December 2009. 2, 3, 5, 7, 9, 11, 12, 13, 14, 16, 18, 25, 26, 31, 45, 48, 49

[28] A. Laurentini, *The Visual Hull Concept for Silhouette-Based Image Understanding*, PAMI, 1994. 3

[29] D. Marr, *Vision. A Computational Investigation into the Human Representation and Processing of Visual Information*, W.H. Freeman and Company,1982. 2

[30] B. Mederos, N. Amenta, L. Velho, L. H. de Figueiredo, *Surface Reconstruction from Noisy Point Clouds*, Eurographics Symposium on Geometry Processing, 2005. 7

[31] Q. Pan, G. Reitmayr, T. Drummond, *ProFORMA: Probabilistic Feature-based On-line Rapid Model Acquisition*, BMVC, 2009. 4, 7

[32] Q. Pan, C.Arth, G. Reitmayr, E. Rosten, T. Drummond, *Rapid Scene Reconstruction on Mobile Phones from Panoramic Images*, ISMAR, 2011. 4, 7

[33] J.-P. Pons, R. Keriven, O. Faugeras, *Multi-View Stereo Reconstruction and Scene Flow Estimation with a Global Image-Based Matching Score*, IJCV, 2007. 5

[34] S. Sinha, P. Mordohai, M. Pollefeys, *Multi-View Stereo via Graph Cuts on the Dual of an Adaptive Tetrahedral Mesh*, ICCV, 2007. 6

[35] N. Snavely, S. Seitz, R. Szeliski, *Photo tourism: Exploring photo collections in 3D*, ACM Transactions on Graphics (SIGGRAPH Proceedings), 25(3), 835-846, 2006. 2

[36] J. Starck, A. Hilton, G.Miller, *Volumetric stereo with silhouette and feature constraints*, BMVC, 2006. 3, 5

[37] C. Strecha, W. von Hansen, L. Van Gool, P. Fua, U. Thoennessen, *On Benchmarking Camera Calibration and Multi-View Stereo for High Resolution Imagery*, CVPR, 2008. 3, 15, 22, 23, 24, 25, 47, 48

[38] G. Vogiatzis, C. Hernndez, P. Torr, R. Cipolla, *Multi-view Stereo via Volumetric Graph-cuts and Occlusion Robust Photo-Consistency*, PAMI, 2007. 3, 5

[39] H. Vu, P. Labatut, J.-P. Pons and R. Keriven, *High Accuracy and Visibility-Consistent Dense Multi-view Stereo comments* PAMI, 2012. 5, 7, 13

[40] Set of libraries, *Boost C++ libraries*, http://www.boost.org (checked May 2012). 46

[41] Open source project, *Computational Geometry Algorithms Library*, http://www.cgal.org (checked May 2012). 9, 12, 14

[42] Open source project, *MeshLab*, supported by the 3D-CoForm project, http://meshlab.sourceforge.net (checked May 2012). 4, 22, 35, 36, 42