Large-Scale Semantic 3D Reconstruction: an Adaptive Multi-Resolution Model for Multi-Class Volumetric Labeling

Maroš Bláha[†,1] Christoph Vogel[†,1,2] Audrey Richard[1] Jan D. Wegner[1]
Thomas Pock[2,3] Konrad Schindler[1]
[1] ETH Zurich [2] Graz University of Technology [3] AIT Austrian Institute of Technology

Abstract

*We propose an adaptive multi-resolution formulation of semantic 3D reconstruction. Given a set of images of a scene, semantic 3D reconstruction aims to densely reconstruct both the 3D shape of the scene and a segmentation into semantic object classes. Jointly reasoning about shape and class allows one to take into account class-specific shape priors (*e.g.*, building walls should be smooth and vertical, and vice versa smooth, vertical surfaces are likely to be building walls), leading to improved reconstruction results. So far, semantic 3D reconstruction methods have been limited to small scenes and low resolution, because of their large memory footprint and computational cost. To scale them up to large scenes, we propose a hierarchical scheme which refines the reconstruction only in regions that are likely to contain a surface, exploiting the fact that both high spatial resolution and high numerical precision are only required in those regions. Our scheme amounts to solving a sequence of convex optimizations while progressively removing constraints, in such a way that the energy, in each iteration, is the tightest possible approximation of the underlying energy at full resolution. In our experiments the method saves up to 98% memory and 95% computation time, without any loss of accuracy.*

1. Introduction

Geometric 3D reconstruction and semantic interpretation of the observed scene are two central themes of computer vision. It is rather obvious that the two problems are not independent: geometric shape is a powerful cue for semantic interpretation and vice versa. As an example, consider a simple concrete building wall: the observation that it is vertical rather than horizontal distinguishes it from a road of similar appearance; on the other hand the fact that it is a wall and not a tree crown tells us that it should be flat and vertical. More generally speaking, *jointly* addressing 3D reconstruction and semantic understanding can be expected to deliver at the same time better 3D geometry, via category-specific priors for surface shape, orientation and layout; and better segmentation into semantic object classes, aided by the underlying 3D shape and layout. Jointly inferring 3D geometry and semantics is a hard problem, and has only recently been tackled in a principled manner [17, 25, 36]. These works have shown promising results, but have high demands on computational resources, which limits their application to small volumes and/or a small number of images with limited resolution.

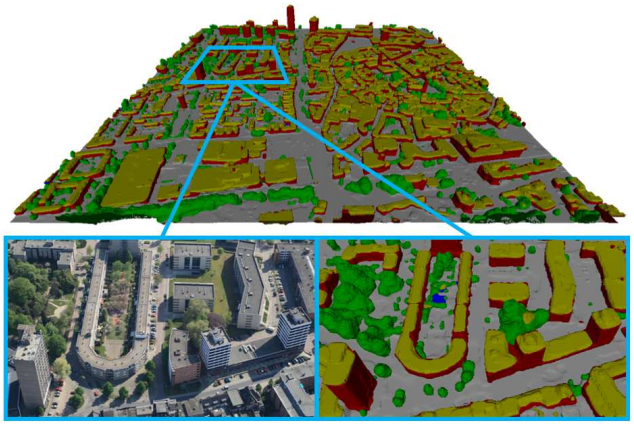


Figure 1: Semantic 3D model of the city of Enschede generated with the proposed adaptive multi-resolution approach.

We propose a method for joint 3D reconstruction and semantic labeling, which scales to much larger regions and image sets. Our target application is the generation of interpreted 3D city models from terrestrial and aerial images, *i.e.* we are faced with scenes that contain hundreds of buildings. Such models are needed for a wide range of tasks in planning, construction, navigation, *etc*. However, to this day they are generated interactively, which is slow and costly.

The core idea of our method is to reconstruct the scene with *variable volumetric resolution*. We exploit the fact that the observed surface constitutes only a 2D manifold in 3D space. Large regions of most scenes need not be modeled at

[†] shared first authorship

high resolution – mostly this concerns free space, but also parts that are under the ground, inside buildings, *etc*. Fine discretization and, likewise, high numerical precision are only required at voxels[1] close to the surface.

Our work builds on the convex energy formulation of [17]. That method has the favorable property that its complexity scales only with the number of voxels, but not with the number of observed pixels/rays. Starting from a coarse voxel grid, we solve a sequence of problems in which the solution is gradually refined only near the (predicted) surfaces. The adaptive refinement saves memory, which makes it possible to reconstruct much larger scenes at a given target resolution. At the same time it also runs much faster. On the one hand the energy function has a lower number of variables; on the other hand low frequencies of the solution are found at coarse discretization levels, and iterations at finer levels can focus on local refinements.

The contribution of this paper is an *adaptive multi-resolution framework for semantic 3D reconstruction*, which progressively refines a volumetric reconstruction only where necessary, via a sequence of convex optimization problems. To our knowledge it is the first formulation that supports multi-resolution optimization and adaptive refinement of the volumetric scene representation. As expected, such an adaptive approach exhibits significantly better asymptotic behavior: as the resolution increases, our method exhibits a quadratic (rather than cubic) increase in the number of voxels. In our experiments we observe gains up to a factor of 22 in speed and reduced memory consumption by a factor of 40. Both the geometric reconstruction and the semantic labeling are as accurate as with a fixed voxel discretization at the highest target resolution.

Our hierarchical model is a direct extension of the fixed-grid convex labeling method [17] and emerges naturally as the optimal adaptive extension of that scheme, *i.e*., under intuitive assumptions it delivers the tightest possible approximation of the energy at full grid resolution. Both models solve the same energy minimization, except that ours is subject to additional equality constraints on the primal variables, imposed by the spatial discretization.

2. Related Work

Large-scale 3D city reconstruction is an important application of computer vision, *e.g*. [15, 29, 26]. Research aiming at purely geometric surface reconstruction rarely uses volumetric representations, though, because of the high demands w.r.t. memory and computational resources. In this context [30] already used a preceding semantic labeling to improve geometry reconstruction, but not vice versa.

Initial attempts to jointly perform geometric and semantic reconstruction started with depth maps [28], but later research, which aimed for truly 3-dimensional reconstruction from multiple views, switched to a volumetric representation [17, 2, 25, 36, 39], or in rare cases to meshes [9]. The common theme of these works is to allow interaction between 3D depth estimates and appearance-based labeling information, via class specific shape priors. Loosely speaking, the idea is to obtain at the same time a reconstruction with locally varying, class-specific regularization; and a semantic segmentation in 3D, which is then trivially consistent across all images. The model of [17] employs a discrete, tight, convex relaxation of the standard multi-label Markov random field problem [42] in 3D, at the cost of high memory consumption and computation time. Here, we use a similar energy and optimization scheme, but significantly reduce the run-time and memory consumption, while retaining the advantages of a joint model. [25] also jointly solve for class label and occupancy state, but model the data term with heuristically shortened ray potentials [32, 36]. Yet, the representation inherits the asymptotic dependency on the number of pixels in the input images. [25] also resort to an octree data structure to save memory, which is fixed in the beginning according to the ray potentials, contrary to our work, where it is adaptively refined. This is perhaps also the work that comes closest to ours in terms of large-scale urban modeling, but (like other semantic reconstruction research) it uses only street-level imagery, and thus only needs to cover the vicinity of the road network, whereas we reconstruct the complete scene.

Since the seminal work [13] volumetric reconstruction has evolved remarkably. Most methods compute a distance field or indicator function in the volumetric domain, either from images or by directly merging several 2.5D range scans. Once that representation has been established, the surface can be extracted as its zero level set, *e.g*. [33, 22].

Many volumetric techniques work with a regular partitioning of the volume of interest [43, 41, 32, 12, 23, 24, 36]. The data term per voxel is usually some sort of signed distance generated from stereo maps, *e.g*. [43, 41]. Beyond stereo depth, [12] propose to also exploit silhouette constraints as additional cue about occupied and empty space.

Going one step further, [32, 36] model, for each pixel in each image, the visibility along the full ray. Such a geometrically faithful model of visibility, however, leads to higher-order potentials per pixel, comprising all voxels intersected by the corresponding ray. Consequently the memory consumption is no longer proportional to the number of voxels, but depends on the number of ray-voxel intersections, which can be problematic for larger image sets and/or high-resolution images. In contrast, the memory footprint of our method (and of others that include visibility locally [43, 17]) is linear in the number of voxels, and thus can be reduced efficiently by adaptive discretization.

[27] deviate from a regular partitioning of the volume,

[1] Throughout the paper, the term *voxel* means a cube in any tesselation of 3-space. Different voxels do *not* necessarily have the same size.

and instead start from a Delaunay tetrahedralization of a 3D point cloud (from multi-view stereo). The tetrahedrons are then labeled empty or occupied, and the final surface is composed of triangles that are shared by tetrahedrons with different labels. The idea was extended by [20], who focus on visibility to also recover weakly supported objects.

In fact even the well-known PMVS multi-view stereo method [14] originally includes volumetric surface reconstruction from the estimated 3D points and normals. To that end, the Poisson reconstruction method [21] was adopted, which aligns the surface with a guidance vector field (given by the estimated normals). The octree representation of [21], was later combined [5] with a cascadic multigrid solver, *e.g.* [8, 16], leading to a significant speed-up. The framework is eminently suitable for large scale processing, but the least-squares nature inherited from the original Poisson formulation makes it susceptible to outliers. In contrast, our formulation can use robust error functions to handle noisy input. The price to pay is a more involved optimization problem instead of a simple linear system. We furthermore exploit that high precision is only needed at voxels close to the surface; representing large regions, that have a constant semantic label, with many voxels appears wasteful. A similar idea was utilized by [1] in the context of stitching images in the gradient domain. Contrary to prior work [21, 5, 10], our octree structure is not predetermined by the input data, but refined adaptively, such that we can exploit the per-class probabilities rather than only a minimal energy solution. Compared to refining all voxels with data, we can avoid many unnecessary splits that would otherwise be invoked by noise in the depth maps.

One can interpret our method as a combination of multigrid (coarse-to-fine) reconstruction on a volumetric pyramid [43, 41], and adaptive hierarchical refinement, *e.g.* [19]. We also refine selectively, and initialize the solver from previous results for faster convergence.

3. Method

To address 3D semantic segmentation and geometry reconstruction in a joint fashion, we follow the approach of [17]. The model employs an implicit volumetric representation, allowing for arbitrary but closed and oriented topology of the resulting surface. One limitation of that model is its huge memory consumption, which we address with our spatially adaptive scheme, without loss in quality.

3.1. Discrete Formulation

In [17] a bounding box of the region of interest is subdivided into regular and equally sized voxels $s \in \Omega$. The model then determines the likelihood that an individual voxel is in a certain state. The scene is described by a set of indicator functions $x_s^i \in [0,1]$, which are constant per voxel element s. As indicated by the respective function $(x^i = 1)$,

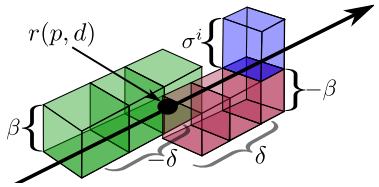


Figure 2: Contribution to the data term of the ray r through pixel p observing class i and depth d, *c.f.* Eqs. (3a,3b).

the voxels can take on a state (*i.e.* a class) i out of a predefined set $\mathcal{C} = \{0 \dots M-1\}$. For our urban scenario we consider a voxel to either be *freespace* ($i = 0$), or occupied with *building wall*, *roof*, *vegetation* or *ground*. Additionally we collect objects that are not explicitly modeled in an extra *clutter* state. A solution to the labeling problem is found by minimizing the energy:

$$E(\mathbf{x}) = \sum_{s \in \Omega} \sum_i \rho_s^i x_s^i + \sum_{i,j;i<j} \phi^{ij}(x_s^{ij} - x_s^{ji}), \quad (1)$$

subject to the following marginalization, normalization and non-negativity constraints:

$$x_s^i = \sum_j x_{s,k}^{ij}, \;\; x_s^i = \sum_j x_{s-e_k,k}^{ji}, \;\; k \in \{1,2,3\} \text{ and } \sum_i x_s^i = 1, \;\; x^{ij} \geq 0. \quad (2)$$

Here, $e_k \in \mathbb{R}^3$ denotes the k^{th} canonical unit vector. The convex and 1-homogeneous functions ϕ^{ij} locally penalize the transition from label i to label j. Intuitively, the variables x^{ij} can be interpreted as encoding the probability mass transferred from class i to class j as one moves from voxel s to its neighbor in direction k. Here, ϕ^{ij} acts as a class-specific geometric prior, which, given the local surface orientation $x_s^{ij} - x_s^{ji} \in [-1,1]^3$, can also take the direction of the boundary surface into account.

The data cost ρ_s^i combines evidence from depthmaps and semantic segmentation masks, and encodes the likelihood of label i at a certain voxel s.

The energy defined by Eqs. (1, 2) is a generalization of the standard primal LP-relaxation of the Markov Random Field energy. As noted in [42], the formulation in discrete space relaxes the need for a (w.r.t. the label space) metric regularizer ϕ, which is mandatory for the continuous case (*e.g.* [35, 38]).

3.2. Data Term

To define the data cost for a voxel at a certain grid resolution we again follow [17]. Consider a pixel p in one of the images, and let d denote the pixel's observed depth. The possible semantic classes are indexed with i. Now, let $r(p,\hat{d})$ be a function that maps a depth value $\hat{d}$ to a 3D point on the ray through p. Then the contribution of p to the energy at voxel s is:

$$\rho_s^i := \quad \sigma^i \text{ if } r(p, d+\delta) \in s \wedge i \neq 0\,, \text{ and} \tag{3a}$$

$$\rho_s^i := \begin{cases} \beta & \text{if } \exists\, \hat{d} : r(p,\hat{d}) \in s \wedge 0 < d - \hat{d} < \delta \wedge i \neq 0 \\ -\beta & \text{if } \exists\, \hat{d} : r(p,\hat{d}) \in s \wedge 0 < \hat{d} - d < \delta \wedge i \neq 0 \\ 0 & \text{otherwise.} \end{cases} \tag{3b}$$

The situation is depicted in Fig. 2. σ^i denotes the negative log-likelihoods of observing class i in the pixel. The σ^i are obtained from a MultiBoost classifier. Details can be found in the supplementary material.

Eq. (3b) is independent of the class label i and only considers voxels close to the surface, as predicted by the depth map. The data term in Eqs. (3a,3b) is given in form of a truncated L^1 norm, which penalizes the deviation of the reconstruction from the observed depth along a pixel's viewing ray. The parameters δ and β encode the truncation point and slope (weight) of the corresponding penalty. In other words, the underlying model assumes the inlier noise of the depthmaps to be exponentially distributed, *c.f.* [17]. Because we seek to minimize the energy (Eq. 1), the data cost prefers freespace for voxels in front of the observed depth. Assuming independence of the per-pixel observations, the final data costs per voxel can be accumulated over all rays.

Discussion. Accounting for visibility only locally near the observed depth is clearly an approximation, but it has the advantage that everything is encapsulated in the unary potentials. Modeling visibility along the full length of the rays leads to higher-order potentials which, for each pixel in each image, relate the depth observation to the occupancy of all voxels passed by the ray (either independently per view [25] or including multi-view constraints [32]). For a volume of $|\Omega|$ voxels, the less complicated first case already leads to $O(\sqrt[3]{|\Omega|})$ voxels per clique. In large-scale applications like ours, with hundreds of images of several Megapixels each, such a model faces serious memory issues.[2] In contrast, breaking the higher-order cliques down to local unary potentials eliminates the dependency on the number and the resolution of the input images, such that the memory consumption scales only with the number of voxels. Hence, the reduced number of voxels in our hierarchical model translates directly to a smaller memory footprint.

3.3. Class-Specific Geometric Priors

The functions ϕ^{ij} penalize class transitions in the volume, and are modeled as negative log-probabilities of the following form:

$$\phi^{ij}(y) = \psi^{ij}(y) + ||y||_2 T^{ij}. \tag{4}$$

The isotropic part T^{ij} contains the neighborhood statistics of the classes. The anisotropic part ψ^{ij} models the likelihood of a transition between classes i and j in a certain direction. Our task-specific choices are detailed in Sec. 5.

[2]Note that [25] propose heuristic ways to limit the influence region of a ray. That alternative could be analyzed in future work.

Note that ϕ^{ij} in Eq. (4) is 1-homogeneous, such that the area of the bounding surface element is implicitly considered in the finite difference scheme. The parametric form of ψ, or rather of its dual $\psi^*(p) = \iota_{W_\psi}$, is chosen to be the indicator function of a convex set W_ψ, the so-called *Wulff* shape. This choice leads to $\psi(y) = \sup_{n \in W_\psi} n^\mathsf{T} y$.

4. Hierarchical Algorithm

The described volumetric model for joint 3D reconstruction and semantic segmentation is rich, but memory-hungry and computationally expensive. To make it more scalable, we embed it in an octree and develop an optimal spatially adaptive refinement scheme. We start at a resolution level $l = L_0$ with a coarse 3D grid, minimize the energy, and then refine the discretization *only close to the surface*. We assert that the preliminary result at coarse discretization can not only serve as an initialization for the finer discretization, but also provides a good guess where one can expect surface transitions, and in this way guide the adaptive refinement. Data and regularization terms are updated for refined voxels, and the new energy is minimized, until the smallest voxels in the octree have the target resolution L_N. We point out the difference to standard surface refinement: in our volumetric multi-class scheme the connectivity can change at finer resolution levels, for instance a narrow street might open between two formerly connected buildings.

Loosely speaking, one can interpret our framework as a multi-grid method [8], where the solution at a coarse discretization of the domain is used as improved initial guess for the fine-grid relaxation. The multi-grid approach is a good match for our problem. Low frequency components of the solution are already found at coarse resolution. This greatly accelerates the computation, because at full resolution they span many voxels, thus gradient-based optimization would take many iterations to converge, *e.g.* [8, 16].

We first describe our data structure and then derive the hierarchical energy and optimization procedure.

4.1. Octree-Structure

In the octree we distinguish inner and leaf nodes. The former hold the parent-child relations of the tree, whereas the latter store the variables needed to minimize the energy. Inner nodes are designed to consume as little memory as possible. They each contain a 32-bit index for the eight children, of which 1 bit is used to indicate whether the child is a leaf or inner node; and one 32-bit index to the parent, of which 5 bits are used to store the depth (octree level). Although more sophisticated implementations exist, *e.g.* [31], this simple structure proved sufficient for our application. A leaf voxel, on the other hand, has to store a number of floats, which is quadratic in the number $|\mathcal{C}|$ of classes. In our case ($|\mathcal{C}| = 6$) we need 181 floats. Hence, our octree consumes approximately 99% of the memory in its leafs, which shows

that the overhead introduced by the adaptive data structure is negligible.

4.2. Discrete Energy in the Octree

Other than in the regular voxel grid $\Omega^{L_N} := \Omega$, voxels of different sizes coexist in the refined volume Ω^l at resolution level $l \in \{L_0, \dots, L_N\}$. Our derivation of the corresponding generalized energy starts from three desired properties: **(i)** Elements form a hierarchy defined by an octree. **(ii)** Each voxel, independent of its resolution, holds the same set of variables. **(iii)** The energy can only decrease if the discretization is refined from Ω^l to Ω^{l+1}:

$$E_l(\mathbf{x}_l^*) \geq E_{l+1}(\mathcal{A}_{l,l+1}\mathbf{x}_l^*) \geq E_{l+1}(\mathbf{x}_{l+1}^*). \tag{5}$$

Here, we have defined the linear operator $\mathcal{A}_{l,l+1}$ to lift the vectorized set of primal variables $\mathbf{x}^l := \big(\mathbf{x}(s)\big)_{s\in\Omega^l}$,

$$\mathbf{x}(s) := ((x^i(s))_{i=0\dots M-1}, (x_k^{ij}(s))_{i,j=0\dots M-1,\ k=1,2,3})^\mathsf{T}, \tag{6}$$

to the refined discretization at level $l+1$. While the second inequality in (5) follows immediately from the optimality of $\mathbf{x}^*$, the first one defines the relationship between solutions at coarser and finer levels. In case of equality, we can observe that minimizing our energy w.r.t. the reduced variable set at coarser level corresponds to minimizing the energy of its lifted version in the refined discretization.

In the light of **(i)**, any proper *prolongation* must fulfill: $\mathcal{A}_{l+1,l+2}\mathcal{A}_{l,l+1} = \mathcal{A}_{l,l+2}$. Then, with the choice $E_l(\mathbf{x}_l) := E(\mathcal{A}_{l,L_N}\mathbf{x}_l)$, equality in the first part of **(iii)** holds:

$$\begin{aligned} E(\mathcal{A}_{l,L_N}\mathbf{x}_l) = E_l(\mathbf{x}_l) \geq E_{l+1}(\mathcal{A}_{l,l+1}\mathbf{x}_l) = \\ E(\mathcal{A}_{l+1,L_N}\mathcal{A}_{l,l+1}\mathbf{x}_l) = E(\mathcal{A}_{l,L_N}\mathbf{x}_l) \end{aligned} \tag{7}$$

Prolongation Operator. Because of the hierarchical structure it is sufficient to specify mappings only for a single coarse parent voxel s and one of its descendants $\bar{s}$. We further assemble the operator from two parts, which individually lift indicator and transition variables:

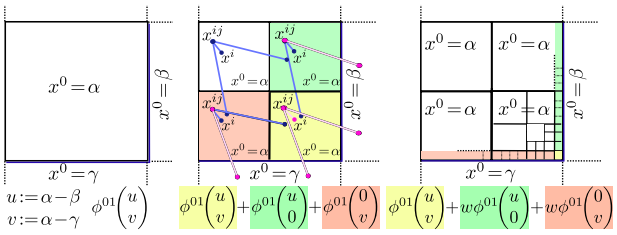
$$\mathcal{A} := \left[(\mathcal{A}^I)^\mathsf{T}; (\mathcal{A}^{IJ})^\mathsf{T}\right]^\mathsf{T}. \tag{8}$$

We start with the former:

$$\begin{aligned} &\mathcal{A}^I_{l,L}(s,\bar{s}) := \left[A^I_{l,L}|\mathbf{0}\right],\ A^I_{l,L} \in \mathbb{R}^{M\times M}, \mathbf{0} \in \mathbb{R}^{M\times 3M^2}, \\ &A^I_{l,L}(i,j) = 1 \text{ iff } i = j \text{ and } 0 \text{ else.} \end{aligned} \tag{9}$$

The operator is already specified for general $L \geq l$. Then, the data energy of a labeling $\mathbf{x}_s^i$ for a coarse voxel s at level l becomes: $\sum_{\bar{s}\in\Omega^{L_N}\cap s,\, i} \rho_{\bar{s}}^i A^I_{l,L_N}(s,\bar{s}) x_s^i = \sum_i \rho_s^i x_s^i$. In accordance with (1), we abbreviated the data term for the coarse voxel with ρ_s, summing over all its descendants.

To define the prolongation of the transition variables $x^{ij}(s)$, we first analyze the splitting of a single voxel. The situation is illustrated in Fig. 3, for simplicity restricted to a



(a) (b) (c)

Figure 3: Adaptive regularizer (2D case).

2 label, 2D case. After splitting the coarse voxel (Fig. 3a), its refined version has to fulfill the constraints from (2). All inner constraints (Fig. 3b, blue lines) can be fulfilled by setting $x_k^{ii} = x^i$ and $x_k^{ij} = 0$ else, which also avoids a penalty from the regularizer. Voxel with non-zero transitions are only found at the boundary (Fig. 3b, pink lines). Depending on the location at the border of the coarse voxel, different components of the argument of the regularizer ϕ can be set to 0 (Fig. 3b). Further, after additional splits (Fig. 3c), the same functional forms occur with different frequency. This motivates the choice of a level-dependent regularizer. For a voxel at level l we use the *weighted sum of functions that occur at its border after maximal refinement*. Let $\partial_{e_k} s$ be the boundary of s in direction e_k. We can now define our lifting of the transition variables from a parent voxel $s \in \Omega^l$ to $\bar{s} \in \Omega^L \cap s$:

$$\begin{aligned} &\mathcal{A}^{IJ}_{l,L}(s,\bar{s}) := \left[B^I_{l,L}|B^{IJ}_{l,L}\right],\ B^I_{l,L} \in \mathbb{R}^{3M^2\times M},\ B^{IJ}_{l,L} \in \mathbb{R}^{3M^2\times 3M^2} \\ &B^I_{l,L}((i,i,k),(i)) = 1 \text{ iff } \partial_{e_k}\bar{s} \not\subset \partial_{e_k} s \text{ and } 0 \text{ else} \\ &B^{IJ}_{l,L}((i,j,k),(i,j,k)) = 1 \text{ iff } \partial_{e_k}\bar{s} \subset \partial_{e_k} s \text{ and } 0 \text{ else.} \end{aligned} \tag{10}$$

Feasibility is preserved by construction and both conditions for (7) are fulfilled (proof in the supplementary material).

Adaptive regularization. Our regularizer, $\Phi_l^{ij}(x_s^{ij} - x_s^{ji})$, depends on the resolution of a voxel and is of the form:

$$\Phi_l(z) := \phi(z) + \sum_{k=1}^{3} w_e^l \phi(z - z^\mathsf{T} e_k e_k) + w_f^l \phi(z^\mathsf{T} e_k e_k). \tag{11}$$

At faces we measure $\phi(z^\mathsf{T} e_k e_k)$, at edges $\phi(z - z^\mathsf{T} e_k e_k)$ for some direction e_k, $k = 1, 2, 3$ and in the corner we get $\phi(z)$. The weights reflect the occurrence of grid-level voxels at the boundary of the enclosing parent voxel (*c.f.* Fig. 3c):

$$w_e^l := 2^{L_N - l} - 1 \text{ and } w_f^l := (w_e^l)^2. \tag{12}$$

All our (an-)isotropic regularizers are of the form $\phi(z) := \sup_{n\in W} n^\mathsf{T} z$, since $T^{ij}||n||_2 = \sup_{n:||n||_2\leq T^{ij}} n^\mathsf{T} z$. Equation (11) is then equivalent to:

$$\begin{aligned} &\Phi_l(z) := \sup_{n\in W^l} n^\mathsf{T} z, \text{ with} \\ &W^l := W \oplus \sum_{k=1}^{3} w_e^l P_{H_k}(W) \oplus w_f^l P_{L_k}(W), \end{aligned} \tag{13}$$

where W^l is the Minkowski sum of the respective sets and P denotes a projection onto the plane $H_k := \{x \in \mathbb{R}^3 | x^\mathsf{T} e_k = 0\}$, respectively the line $L_k := \{se_k | s \in \mathbb{R}\}$.

Numerical scheme. Equipped with prolongation operator, scale-dependent regularizer Φ_l^{ij} and data term, our energy for an arbitrary hierarchical discretization Ω^l of 3-space becomes:

$$E_l(\mathbf{x}_l) = \sum_{s \in \Omega^l} \sum_i \rho_s^i x_s^i + \sum_{i,j;i<j} \Phi_l^{ij}(x_s^{ij} - x_s^{ji}). \quad (14)$$

Introducing the set $\mathcal{N}_{-e_k}(s)$ to collect the neighborhood of s in direction $-e_k$, we can denote a new set of constraints:

$$x_s^i = \sum_j x_{s,k}^{ij}, \;\; x_s^i = \sum_j x_{\bar{s},k}^{ji}, \forall \bar{s} \in \mathcal{N}_{-e_k}(s),$$
$$k \in \{1,2,3\} \;\text{ and }\; \sum_i x_s^i = 1, \;\; x^{ij} \geq 0. \quad (15)$$

The energy (14) is convex. To solve it, we introduce Lagrange multipliers for the constraints (15), convert the problem to primal-dual form, and apply the method of [11]. The prolongation operator defines a weighting of the different constraints. This is helpful for pre-conditioning [34], which is essential because of the large size differences between voxels in our hierarchical framework.

Our numerical scheme requires us to project onto shapes that are Minkowski sums of convex sets. For that several alternatives exist. In case the Wulff shapes are given explicitly in the form of a triangular mesh, one can pre-compute (13) for each level in polynomial time in an offline step [3]. In our case the sets are simple, in the sense that the projection onto each Wulff shape can be performed in closed form. Thus, we utilize Eq. 11. If memory consumption is not an issue, a simple way is to maintain separate dual variables for the individual sets. In contrast, a Dykstra-like projection scheme [6] avoids storing additional variables altogether, at the cost of a small increase in computation time. In the supplementary we give an outline of the numerical scheme, including a derivation of this projection.

Optimality. The adaptive energy E_l as introduced above can be expressed equivalently by augmenting the basic non-adaptive energy (1, 2) with the following constraints:

$$\forall i, \forall s \in \Omega^l : \{x_{\bar{s}}^i = x_s^i | \forall \bar{s} \in \Omega \cap s\} \text{ and } \quad (16)$$
$$\forall i,j, \forall s \in \Omega^l : \{x_{\bar{s},k}^{ij} = x_{s,k}^{ij} | \forall \bar{s} \in \Omega \cap s \wedge \partial_{e_k} \bar{s} \subset \partial_{e_k} s\}.$$

I.e., one can interpret our hierarchical refinement scheme as solving the same problem, subject to additional equality constraints imposed by the variable discretization of the volume. This equivalence proves the optimality of the proposed scheme under the assumptions **(i)**, **(ii)** and **(iii)**.

We point out that the first set of constraints in (16) is not sufficient to derive the multi-resolution scheme (counterexample in the supplementary material). Without additional equality constraints on the x^{ij}, the optimality condition **(iii)** would require the introduction of additional variables for each Wulff shape in (13). We abandon that idea at this point, but if memory consumption is not an issue, a tighter bound on the energy can perhaps be achieved.

Splitting criteria. Ideally our algorithm would in every step refine exactly all those voxels which intersect with the true surface. This is a chicken-and-egg problem, so we have to rely on the preliminary solution at the coarser level to predict these voxels. In practice we simply identify neighboring voxel pairs that are assigned to different classes:

$$\exists k, \bar{s} \in \mathcal{N}_{e_k}(s) : \arg\max_i x_s^i \neq \arg\max_i x_{\bar{s}}^i \quad (17)$$

and divide both voxels s and $\bar{s}$. Moreover, we also require the resolution of adjacent voxels to differ by at most one level, and split voxels accordingly. After refinement, we use the lifting scheme (8) to initialize the newly introduced (primal and dual) variables at the finer resolution.

5. Evaluation

We test our algorithm on a real world data set from the city of Enschede [37]. All experiments are run on a machine with 64 GB of RAM and a hexa *Intel Core i7* CPU.

Data Set and Input Data. We follow a current trend in image-based mapping and exploit oblique aerial imagery in addition to classical nadir photographs. This mitigates visibility problems such as foreshortening or occlusion. In total, the data set comprises of 510 images acquired in the *Maltese cross* configuration (for each position a nadir image and four oblique views to the north, south, east and west).

Our method requires two types of input data: oriented images in order to generate depthmaps, and training data for statistical learning. The images were oriented with VisualSFM [40], and depthmaps were generated with semi-global matching [18, 7]. For semantic labeling we train a MultiBoost classifier [4] on a few hand-labeled images. Details about the employed image features can be found in the supplementary material. With the classifier we then predict per-pixel log-likelihoods for all possible classes. Fig. 4 illustrates our input data at a glance.

Label-Specific Geometric Priors. We employ two forms of Wulff shapes (Sec. 3.3). The first supports flat, horizontal structures in the 3D model and applies to the following label transitions: *ground–freespace*, *ground–building*, *ground–vegetation*, *building–roof*, *roof–freespace*. The second one prefers vertical boundaries at the transitions *building–freespace* and *building–vegetation*. Parameters for the Wulff shapes are either learned from existing city models or set empirically, see [17] for details.

Comparison to Fixed Voxel-Grid. We go on to compare our hierarchical model with a fixed voxel grid of the same

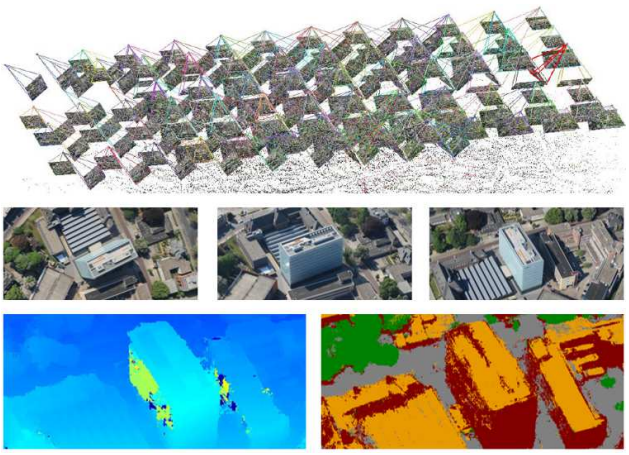


Figure 4: Input data for our method: oriented images (*top*), cutouts from 1 nadir and 2 oblique views (*middle*), depthmap and class probability map (*bottom*).

Data set	Error measure	Octree	Grid	MB
Scene 1	Overall acc. [%]	92.8	92.3	89.1
	Average acc. [%]	92.2	91.7	87.0
Scene 2	Overall acc. [%]	83.9	83.4	82.5
	Average acc. [%]	80.6	79.9	81.4

Table 1: Quantitative verification of our results with the grid model and the MultiBoost input data from [4].

target resolution. The fixed grid requires 600 iterations to converge. In our multi-resolution procedure, we run 200 iterations at every level, then refine all voxels that fulfill the splitting criterion, and run the next 200 iterations. When the first voxels have reached the target resolution L_N we run 100 iterations, conditionally split voxels that are not yet at minimal size, and finally run another 100 iterations.

One problem we face is the lack of 3D ground truth. To quantitatively check the correctness of the results, we use the following procedure: we select two representative images from our data set and manually label them to obtain a semantic ground truth. For the corresponding scene parts, we then run semantic 3D reconstruction, back-project the result to the images, and compare them to the ground-truth labeling in terms of *overall accuracy* and *average accuracy*.

Tab. 1 summarizes the outcomes of the comparison. The differences between adaptive and non-adaptive reconstruction are vanishingly small (< 0.7 percent points) and mostly due to aliasing. The comparison for one of the two scenes is illustrated in Fig. 5. The classification maps from the octree and the full grid are almost indistinguishable, which underlines that the two methods give virtually the same results. We conclude that our refinement scheme is valid and does not lead to any loss in accuracy compared to the full voxel grid. Labels back-projected from the semantic 3D reconstruction are less noisy than the raw classifier output. However, the reconstruction (both adaptive and non-adaptive) introduces a systematic error at sharp 3D boundaries, best

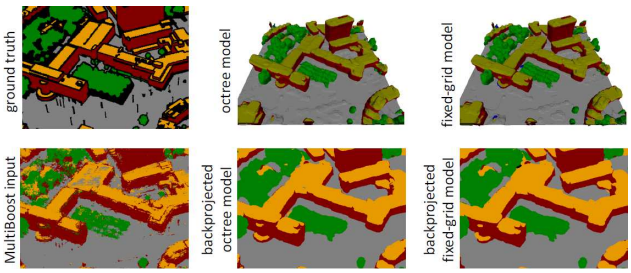


Figure 5: Comparison of the labeling accuracy. Colors indicate ground (*gray*), building (*red*), roof (*yellow*), vegetation (*green*) and clutter (*blue*).

Figure 6: *Left:* Two images from the Enschede dataset. *Middle left:* Semantic 3D models (Scene 3 & 4). *Middle right:* Back-projected models overlayed on the images. *Right*: Pure volumetric 3D models [41]. Note errors such as deformed and fragmented buildings or flattened vegetation.

visible along transitions between *building walls* and *roofs*. This bias originates from our data term, which forces the voxels behind the observed depth (in ray direction) to be occupied. This fattening effect was also observed in [36], and it was shown that complete ray potentials can remedy the problem, at the cost of much higher memory consumption. In spite of the fattening, the back-projected 3D models are still (slightly) more correct than the MultiBoost results.

The gains are larger in the opposite direction, *i.e.* the semantic information significantly improves the 3D surface shape. Fig. 6 illustrates exemplary cases where our class-specific priors lead to superior 3D models compared to a generic regularization of the surface area [41]. Unfortunately that effect is hard to quantify.

Performance Analysis. We go on to measure how much memory and computation time we save by adaptively refining the reconstruction only where needed. As a baseline, we run the non-adaptive method at full target resolution. Even at 0.4 m voxel size the storage requirements of the baseline limit the comparison to four smaller subsets of our dataset.

For a fair comparison, we cut the bounding box for the non-adaptive method such that it tightly encloses the data (whereas our octree implementation always covers a cubic volume). Since the city of Enschede is flat, this favors the non-adaptive method. In rough terrain or non-topographic applications the gains will be even higher.

Scene	Runtime@0.4 m [sec] 1	2	3	4	Memory@0.4 m [GB] 1	2	3	4	Memory@0.2 m [GB] 3	4
Octree	19883	19672	5488	4984	2.7	2.6	0.7	0.7	3.3	2.7
Grid	430545	416771	91982	92893	54.3	54.3	13.6	13.6	108.5	108.5
Octree (naive)	43174	43845	10603	11343	6.5	6.8	1.7	1.9	—	—
Ratio (Grid)	**21.7**	21.2	16.8	18.6	20.1	**20.9**	19.4	19.4	32.9	**40.2**
Ratio (Octree naive)	2.2	2.2	1.9	2.3	2.4	2.6	2.4	2.7	—	—

Table 2: Comparison of run-time and memory footprint of our method (*Octree*), [17] (*Grid*), and a *naive Octree*. Maximum gains for processing time and memory consumption per refinement level are shown in bold. The target *Grids* feature a resolution of *512 x 512 x 256* (Scene 1 and 2) and *256 x 256 x 256* (Scene 3 and 4) at 0.4 m.

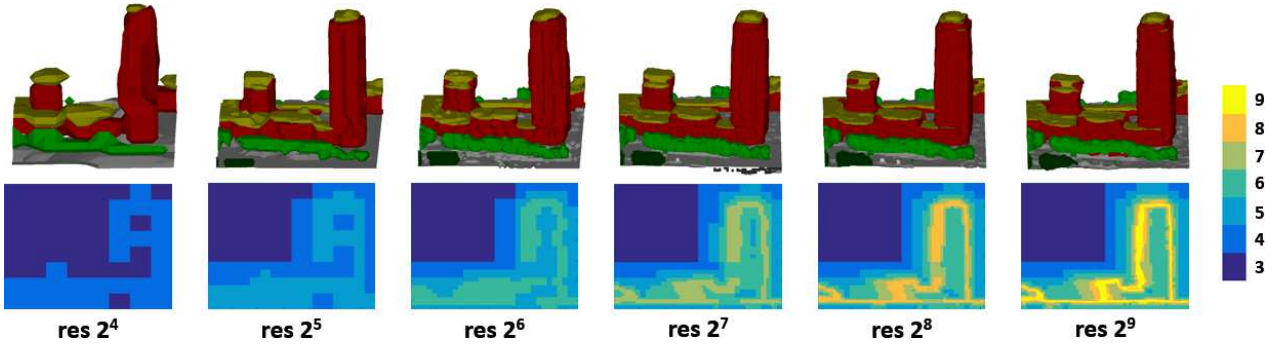


Figure 7: Evolution of the multi-scale semantic 3D model over five refinement steps. *Top*: Reconstructions at the corresponding refinement levels. Both shape and semantic labels gradually emerge in a coarse-to-fine manner. *Bottom*: Vertical slice through the scene, with color-coded voxel size (respectively, depth in the octree).

The hierarchical scheme starts with voxels of size 13.5 m, and does 5 refinements to reach a target resolution of 0.4 m. The results are summarized in Tab. 2. In all scenes, the adaptive computation saves around 95% of both memory and computation time. To quantify the effect of the proposed splitting criterion (Sec. 4) we further contrast it with a simpler adaptive procedure which naively splits any voxel with non-zero data cost (Tab. 2). Our method, which takes into account the class likelihoods, uses around $2.5\times$ less memory and is more than $2\times$ faster. For the two smaller scenes 3 and 4 we refine one level further to a target resolution of 0.2 m. At that resolution the baseline would require >108 GB of memory, 33–$40\times$ more than our adaptive scheme. We only had 64 GB of RAM available, so we could not compare computation time. Across our experiments, we observe an empirical gain of 1.9^N over N refinements, for both processing time and memory consumption.

Fig. 7 illustrates the evolution of our adaptive scheme over 5 refinement steps. The top row shows the intermediate reconstruction, gradually improving in terms of accuracy and detail. The bottom row shows a vertical slice through the volume, with voxels color-coded according to their size, respectively refinement level. Colors range from blue (coarse, $13.5\,\mathrm{m}^3$ voxels) to yellow (fine, $0.2\,\mathrm{m}^3$ voxels). One can clearly see the splitting near surfaces (class boundaries), while voxels in homogeneous areas like freespace or the inside of buildings remain big.

Large-Scale City Reconstruction. Finally, we proceed to our target application of large-scale city reconstruction. We process the whole data set of 510 images and reconstruct an extensive semantic model of the city of Enschede ($3\mathrm{km}^2$) with a target resolution of 0.8 m, respectively $\frac{1}{2048}$ of the bounding volume, see Fig. 1. Our adaptive scheme requires a moderate 27.9 GB of memory, and completed the reconstruction in 40 hours on one PC. The same resolution ($2048\times2048\times128$) would require 434 GB of memory without a multi-resolution scheme.

6. Conclusion

We have proposed an adaptive multi-resolution processing scheme for (joint) semantic 3D reconstruction. The method makes it possible to process much larger scenes than was previously possible with volumetric reconstruction schemes, without any loss in quality of the results.

In future work, we plan to extend the scheme to irregular discretizations, such as the recently popular Delaunay tetrahedralizations, so as to adapt even better to the data at hand. Moreover, our basic idea is generic and not limited to semantic 3D reconstruction. We would like to explore other applications where it may be useful to embed the convex relaxation scheme in an adaptive multi-resolution grid.

Acknowledgements. We thank Christian Häne and Marc Pollefeys for source code and discussions. This work was supported by SNF grant 200021_157101.

References

[1] A. Agarwala. Efficient gradient-domain compositing using quadtrees. *SIGGRAPH 2007*.

[2] Y. Bao, M. Chandraker, Y. Lin, and S. Savarese. Dense object reconstruction using semantic priors. *CVPR 2013*.

[3] H. Bekker and J. Roerdink. An efficient algorithm to calculate the Minkowski sum of convex 3d polyhedra. *ICCS'01*.

[4] D. Benbouzid, R. Busa-Fekete, N. Casagrande, F.-D. Collin, and B. Kégl. MULTIBOOST: a multi-purpose boosting package. *JMLR*, 13(1), 2012.

[5] M. Bolitho, M. Kazhdan, R. Burns, and H. Hoppe. Multilevel streaming for out-of-core surface reconstruction. *Eurographics 2007*.

[6] J. P. Boyle and R. L. Dykstra. A method for finding projections onto the intersection of convex sets in Hilbert spaces. *Lecture Notes in Statistics*, 1986.

[7] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.

[8] W. L. Briggs, V. E. Henson, and S. F. McCormick. *A Multigrid Tutorial (2nd Ed.)*. Society for Industrial and Applied Mathematics, 2000.

[9] R. Cabezas, J. Straub, and J. Fisher III. Semantically-aware aerial reconstruction from multi-modal data. *ICCV 2015*.

[10] F. Calakli and G. Taubin. SSD: Smooth signed distance surface reconstruction. *Computer Graphics Forum*, 30(7), 2011.

[11] A. Chambolle and T. Pock. A first-order primal-dual algorithm for convex problems with applications to imaging. *JMIV*, 40(1), 2011.

[12] D. Cremers and K. Kolev. Multiview stereo and silhouette consistency via convex functionals over convex domains. *PAMI*, 33(6), 2011.

[13] B. Curless and M. Levoy. A volumetric method for building complex models from range images. *SIGGRAPH*, 1996.

[14] Y. Furukawa and J. Ponce. Accurate, dense, and robust multiview stereopsis. *PAMI*, 32(8), 2010.

[15] D. Gallup, J.-M. Frahm, and M. Pollefeys. Piecewise planar and non-planar stereo for urban scene reconstruction. *CVPR 2010*.

[16] E. Grinspun, P. Krysl, and P. Schröder. CHARMS: A Simple Framework for Adaptive Simulation. *SIGGRAPH*, 2002.

[17] C. Häne, C. Zach, A. Cohen, R. Angst, and M. Pollefeys. Joint 3d scene reconstruction and class segmentation. *CVPR 2013*.

[18] H. Hirschmüller. Stereo processing by semiglobal matching and mutual information. *PAMI*, 30(2), 2008.

[19] A. Hornung and L. Kobbelt. Hierarchical volumetric multiview stereo reconstruction of manifold surfaces based on dual graph embedding. *CVPR 2006*.

[20] M. Jancosek and T. Pajdla. Multi-view reconstruction preserving weakly-supported surfaces. *CVPR 2011*.

[21] M. Kazhdan, M. Bolitho, and H. Hoppe. Poisson surface reconstruction. *Eurographics 2006*.

[22] M. Kazhdan, A. Klein, K. Dalal, and H. Hoppe. Unconstrained isosurface extraction on arbitrary octrees. *Eurographics 2007*.

[23] K. Kolev, T. Brox, and D. Cremers. Fast joint estimation of silhouettes and dense 3D geometry from multiple images. *PAMI*, 34(3), 2012.

[24] I. Kostrikov, E. Horbert, and B. Leibe. Probabilistic labeling cost for high-accuracy multi-view reconstruction. *CVPR'14*.

[25] A. Kundu, Y. Li, F. Dellaert, F. Li, and J. Rehg. Joint semantic segmentation and 3d reconstruction from monocular video. *ECCV 2014*.

[26] G. Kuschk and D. Cremers. Fast and accurate large-scale stereo reconstruction using variational methods. *ICCV Workshop on Big Data in 3D Computer Vision, 2013*.

[27] P. Labatut, J.-P. Pons, and R. Keriven. Efficient Multi-View Reconstruction of Large-Scale Scenes using Interest Points, Delaunay Triangulation and Graph Cuts. *ICCV 2007*.

[28] Ľ. Ladický, P. Sturgess, C. Russell, S. Sengupta, Y. Bastanlar, W. Clocksin, and P. Torr. Joint optimisation for object class segmentation and dense stereo reconstruction. *BMVC 2010*.

[29] F. Lafarge, R. Keriven, M. Bredif, and H. H. Vu. A hybrid multi-view stereo algorithm for modeling urban scenes. *PAMI*, 35(1), 2013.

[30] F. Lafarge and C. Mallet. Creating large-scale city models from 3D-point clouds: a robust approach with hybrid representation. *IJCV*, 99(1), 2012.

[31] T. Lewiner, V. Mello, A. Peixoto, S. Pesco, and H. Lopes. Fast generation of pointerless octree duals. *Symposium on Geometry Processing 2010*.

[32] S. Liu and D. B. Cooper. Ray markov random fields for image-based 3d modeling: Model and efficient inference. *CVPR 2010*.

[33] W. E. Lorensen and H. E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. *SIGGRAPH'87*.

[34] T. Pock and A. Chambolle. Diagonal preconditioning for first order primal-dual algorithms in convex optimization. *ICCV 2011*.

[35] T. Pock, A. Chambolle, D. Cremers, and H. Bischof. A convex relaxation approach for computing minimal partitions. *CVPR 2009*.

[36] N. Savinov, Ľ. Ladický, C. Häne, and M. Pollefeys. Discrete optimization of ray potentials for semantic 3d reconstruction. *CVPR 2015*.

[37] Slagboom en Peeters Aerial Survey. http://www.slagboomenpeeters.com/3d.htm.

[38] E. Strekalovskiy, B. Goldlücke, and D. Cremers. Tight convex relaxations for vector-valued labeling problems. *ICCV 2011*.

[39] V. Vineet et al. Incremental dense semantic stereo fusion for large-scale semantic scene reconstruction. *ICRA 2015*.

[40] C. Wu. VisualSFM: A visual structure from motion system, 2011.

[41] C. Zach. Fast and high quality fusion of depth maps. *3DV'08*.

[42] C. Zach, C. Häne, and M. Pollefeys. What is optimized in convex relaxations for multilabel problems: Connecting discrete and continuously inspired MAP inference. *PAMI*, 2014.

[43] C. Zach, T. Pock, and H. Bischof. A globally optimal algorithm for robust TV-L1 range image integration. *ICCV'07*.