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

Supplementary Material

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

The supplementary material is structured as follows: we start with a number of visualizations that support the experimental analysis in Sec. 5 of the paper. First, we show larger and more detailed visualizations of the reconstructed large-scale 3D model, to give a more tangible impression of the model quality. We also visually depict the 2D evaluation procedure for the semantic labeling performance, complementing the quantitative results in Tab. 1 of the paper. In addition, we qualitatively contrast the octree and fixed-grid reconstructions in terms of geometric accuracy. Further, we visualize the comparison between the proposed voxel splitting scheme and the trivial strategy to refine wherever there is a non-zero data cost, *c.f*. Tab. 2 of the paper. Also in the context of that table, we show the detailed evolution of the memory and run-time consumption, per hierarchical refinement step of the proposed scheme.

The remainder of the supplementary material then gives additional technical details omitted in the paper: the exact feature set used to predict the class-conditional probabilities; the theoretical background and implementation details of the numerical scheme used to minimize the adaptive energy and several straightforward proofs for a number of claims that had to be omitted in the paper for lack of space.

1. Complementary Visualizations

To begin with, we present additional visualizations of the large-scale semantic 3D model. Fig. 1 shows an orthographic projection of our reconstruction along the vertical, juxtaposed with the aerial image of the same area from *Google Earth*. Fig. 2 shows an oblique view of the full scene; three oblique close-up views together with the most similar input images; and three street-level views together with the corresponding views from *Google Street View* (not used in our work). The comparisons demonstrate the correctness and high level-of-detail of our reconstruction.

[†] shared first authorship

2. Comparison with a Fixed Voxel Grid

As an add-on to Tab. 1 of the paper, we illustrate the validation of the semantic labeling accuracy. Fig. 3 shows the reconstructions obtained with the proposed scheme as well as with a fixed voxel grid, both at the same target resolution. Those reconstructions are then back-projected to one of the input cameras and compared to a ground truth semantic segmentation, as well as the raw 2D labeling from the MultiBoost classifier, which serves as input for our volumetric 3D modeling.

Complementary to the semantic validation, we qualitatively compare octree and fixed-grid reconstructions in terms of geometric accuracy. Fig. 4 highlights regions with perceptible differences, and simultaneously underlines that the two variants deliver nearly identical models. Differences in the reconstructions are rare and concern only small details.

3. Comparison of Splitting Schemes

This section complements Tab. 2 in the paper. It illustrates qualitatively how the proposed splitting criterion performs compared to a trivial splitting rule, in which all voxels with non-zero data cost are always subdivided (Fig. 5).

4. Performance Analysis per Refinement Level

When using a hierarchical scheme, the savings in memory consumption and computation time depend on the target resolution – the finer the discretization, the more wasteful a uniform voxel grid at that discretization interval. For completeness, we list memory consumption and run-time, as well as the corresponding gains, for different target resolutions. The analysis underlines the increasing benefit of the multi-resolution framework as one aims for more detailed and accurate models, *c.f*. discussion in Sec. 5 of the paper. Empirically, the gain grows by a factor 1.7-1.9 every time the target resolution is doubled, see Tab. 1.

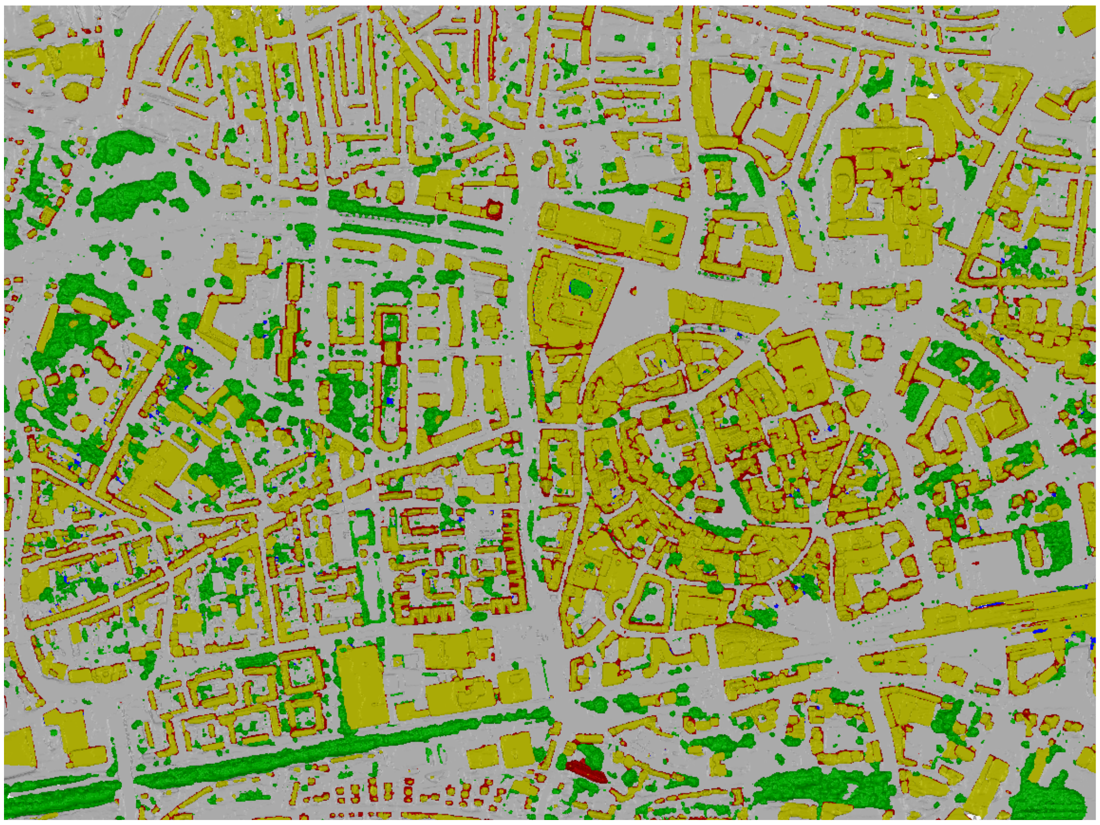
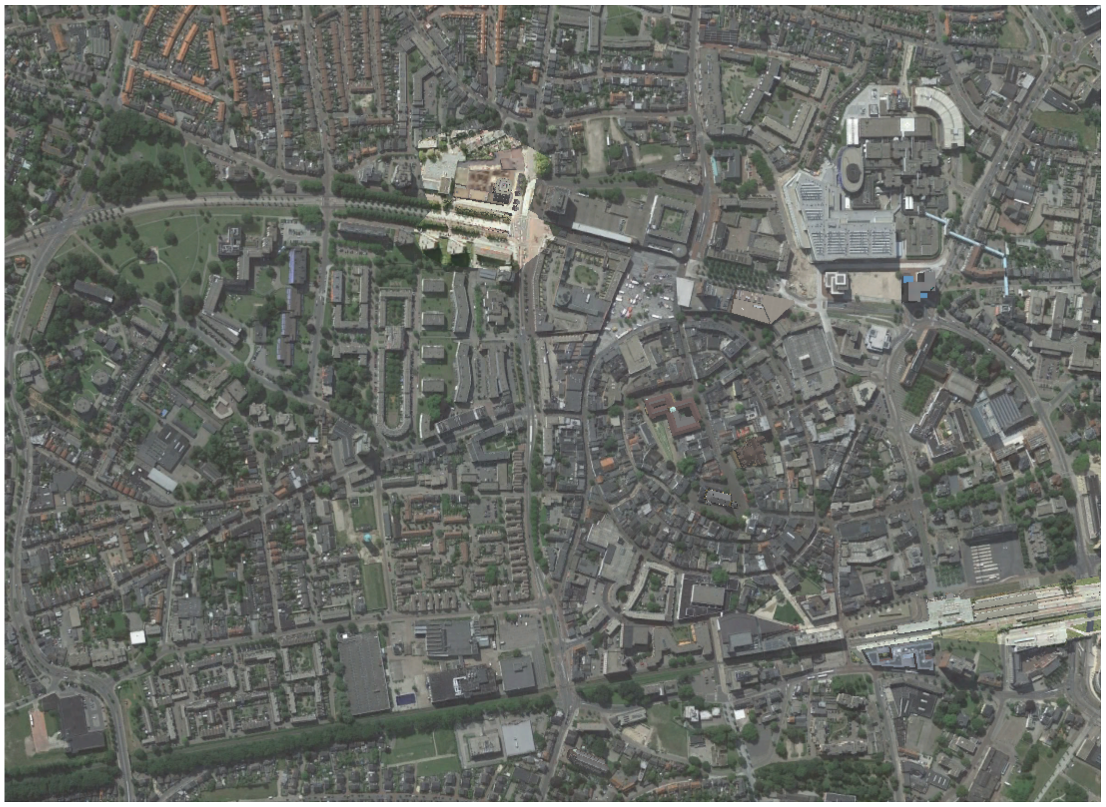



Figure 1. Additional Visualizations. *Top*: orthographic birds-eye view of the complete *Enschede* model. *Bottom*: independent *Google Earth* image of the same area. Colors indicate ground (*gray*), building (*red*), roof (*yellow*), vegetation (*green*) and clutter (*blue*).

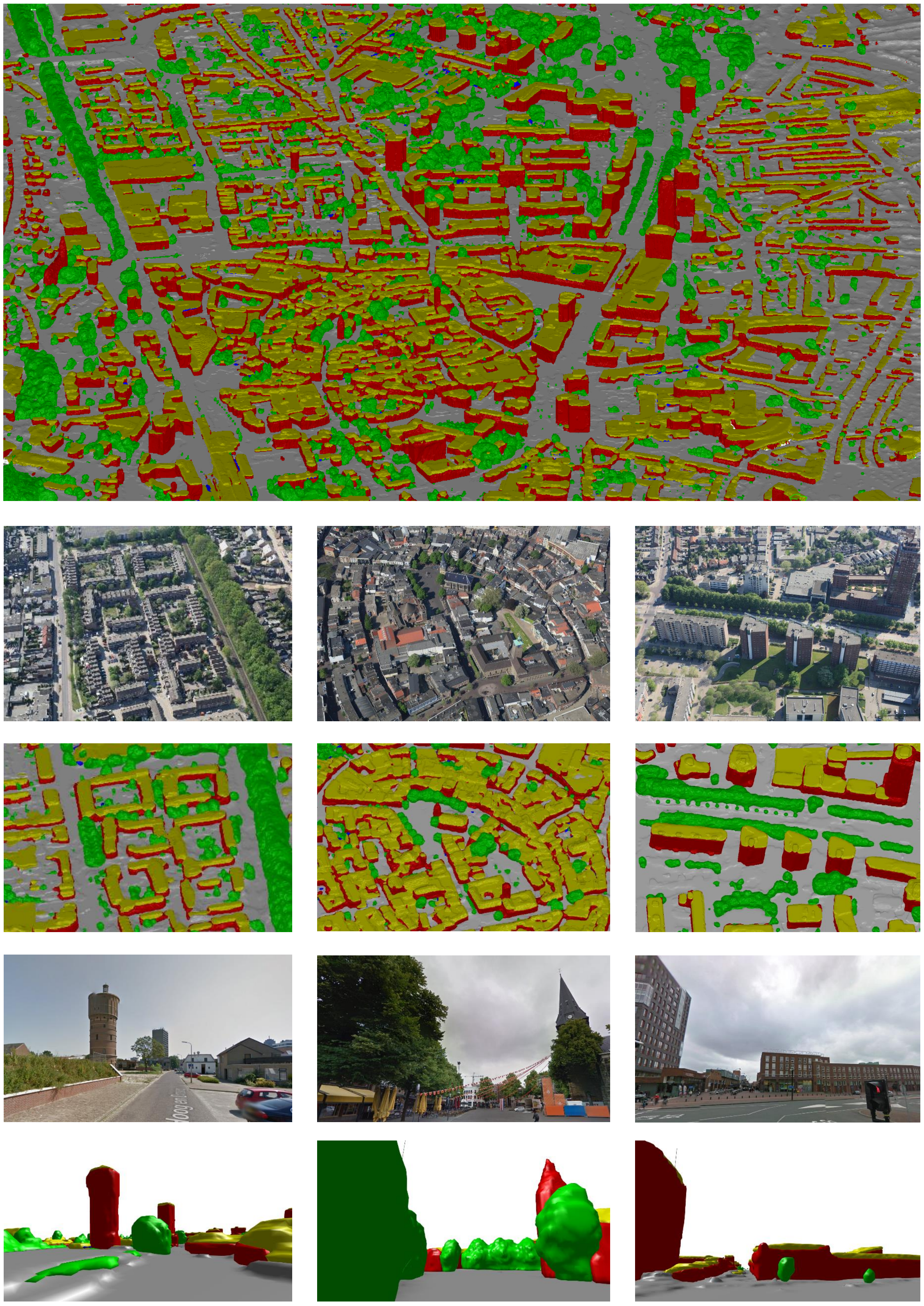


Figure 2. Additional Visualizations. *Big image*: oblique view of the *Enschede* model. *1st and 2nd rows*: aerial images from our input data set, and corresponding details from the semantic 3D model. *3rd and 4th rows*: independent *Google Street View* images, and corresponding details from the semantic 3D model. Colors indicate ground (*gray*), building (*red*), roof (*yellow*), vegetation (*green*) and clutter (*blue*).

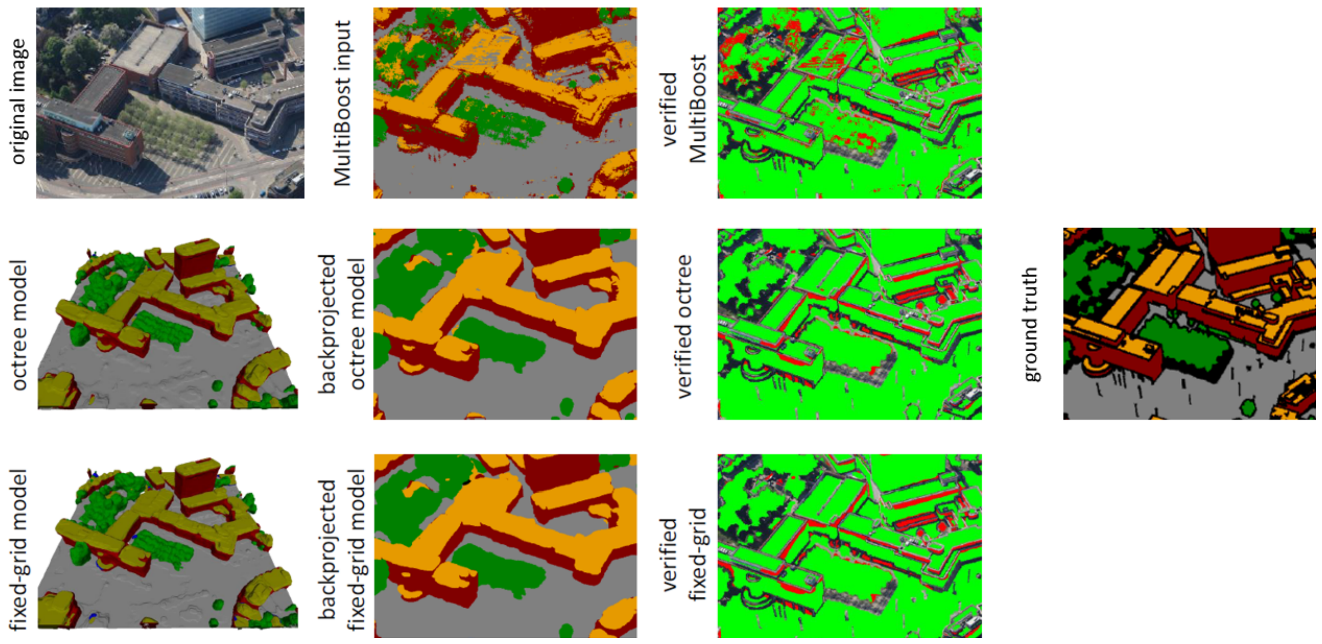


Figure 3. Comparison of the labeling accuracy. *Left*: input image and semantic 3D reconstructions. Colors indicate ground (*gray*), building (*red*), roof (*yellow*), vegetation (*green*) and clutter (*blue*). *Middle left*: resulting label images. *Middle right*: error plot (misclassified pixels are marked in red). *Right*: ground truth. Unlabeled image pixels (*black*) were excluded from the analysis.

		Runtime [sec]				Memory [GB]				mean ratio	gain factor
	Scene	1	2	3	4	1	2	3	4		
1 Split	Octree	160	161	29	39	0.03	0.03	0.01	0.01	–	
	Grid	461	473	121	113	0.11	0.11	0.03	0.03	–	
	Ratio	2.9	2.9	4.2	2.9	3.7	3.7	3.0	3.0	3.3	
2 Splits	Octree	834	827	213	201	0.12	0.12	0.03	0.03	–	
	Grid	4452	4398	1131	1124	0.85	0.85	0.21	0.21	–	
	Ratio	5.3	5.3	5.3	5.6	7.1	7.1	7.0	7.0	6.2	1.9
3 Splits	Octree	4134	3977	1183	1028	0.58	0.56	0.16	0.15	–	
	Grid	39521	39044	9873	9906	6.78	6.78	1.70	1.70	–	
	Ratio	9.6	9.8	8.3	9.6	11.7	12.1	10.6	11.3	10.4	1.7
4 Splits	Octree	19883	19672	5488	4984	2.68	2.62	0.74	0.69	–	
	Grid	430545	416771	91982	92893	54.25	54.25	13.56	13.56	–	
	Ratio	21.7	21.2	16.8	18.6	20.2	20.7	18.3	19.7	19.7	1.9
5 Splits	Octree	–	–	25390	21533	–	–	3.32	2.70	–	
	Grid	–	–	–	–	–	–	108.50	108.50	–	
	Ratio	–	–	–	–	–	–	32.7	40.2	36.5	1.9

Table 1. Gain in run-time and memory footprint for different target resolutions, respectively refinement levels. Note, because of the low memory consumption at coarse discretizations, we show an additional decimal place compared to the table in the main paper.

5. Features of the Semantic Input Data

To estimate the pixelwise class-conditional probabilities in the input images, we train a multiclass boosting classifier [1]. Per pixel we extract a 94-dimensional feature vector from the intensity image and the corresponding depth map. Appearance features are simply RGB values in a 5×5 pixel window, for a total of 75 features. Additionally, we convert the depth map to a 3D point cloud in the scene coordinate system and extract 19 local geometry features [4], derived from the height, the 3D structure tensor (*i.e.* eigenvalues), the local tangent plane, and the point distribution in a vertical column. See Tab. 2 for a complete list.

Type	Feature definition
Height features	Height (z-component) Height variance
Eigenvalue features	Anisotropy $(\lambda_1 - \lambda_3)/\lambda_1$ Planarity $(\lambda_2 - \lambda_3)/\lambda_1$ Sphericity λ_3/λ_1 Linearity $(\lambda_1 - \lambda_2)/\lambda_1$
Local tangent plane features	Vertical component of plane normal Deviation angle of plane normal from vertical Variance of deviation angles Distance from point to local plane Variance of point-to-plane distances
Features based on histogram of signed z-differences to other points in a vertical column	# of bins above mean frequency # of bins below mean frequency Difference: # above bins - # below bins # of local frequency maxima Average distance between local maxima Sum of positive values Sum of negative values # of elements in zero-bin

Table 2. Geometric features [4] used for semantic segmentation (together with RGB values).

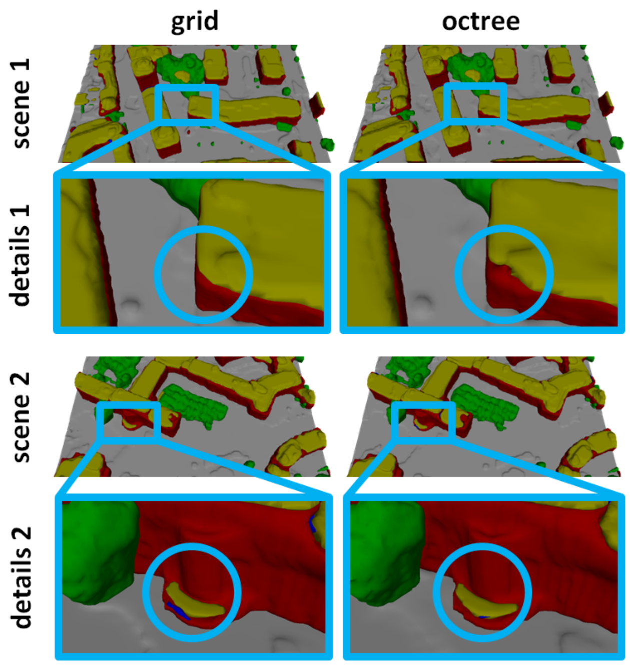


Figure 4. Qualitative comparison between octree and fixed-grid reconstructions in terms of geometric accuracy. Differences are marked with blue circles.

6. Numerical Scheme of the Discrete Energy in the Octree

This section provides a brief overview of our optimization framework. At each octree level l we minimize the adapted energy function E_l, as imposed by the current octree discretization. Recall that voxels of different sizes coexist in the octree. The energy (Eqs. 14, 15 in the paper) was defined as:

$$E_l(\mathbf{x}_l) = \sum_{s\in\Omega^l}\sum_i \rho_s^i x_s^i + \sum_{i,j;i<j} \Phi_l^{ij}(x_s^{ij} - x_s^{ji}). \quad \text{(A.1)}$$

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

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

Here $\mathcal{N}_{-e_k}(s)$ denotes the neighborhood of the voxel s in the direction $-e_k$, which can consist of a voxel of the same size, multiple smaller voxels, or (part of) one bigger voxel. To minimize the convex problem in Eqs. (A.1, A.2), we introduce Lagrange multipliers for the constraints to convert the problem to primal-dual form. The dual variables are:

- ν_s for $\sum_i x_s^i = 1$,
- $\lambda_{s,k}^i$ for $x_s^i = \sum_j x_{s,k}^{ij}$, and
- $\theta_{s,\bar{s}}^i$ for the “per-face” constraints $x_s^i = \sum_j x_{\bar{s},k}^{ji}$, which exist for all $s, \bar{s} \in \mathcal{N}_{-e_k}(s)$.

The dual variables are arranged into a vector $(\boldsymbol{\nu}_l, \boldsymbol{\lambda}_l, \boldsymbol{\theta}_l)^\mathsf{T}$. The same is done for the primal variables $(\boldsymbol{x})_l^\mathsf{T} = (\boldsymbol{x}_l^i, \boldsymbol{x}_l^{ij})^\mathsf{T}$. Then the energy can be written in its primal-

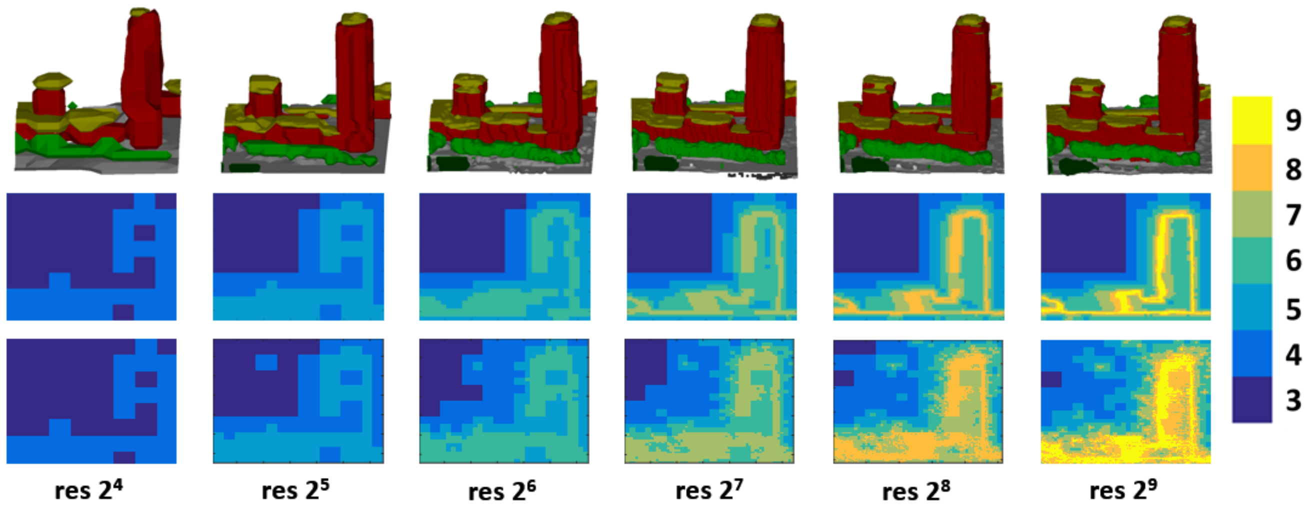


Figure 5. Comparison of the evolution using the proposed splitting criterion and naive splitting (*c.f.* Fig. 7 in the paper). *Top*: intermediate reconstructions throughout the refinement. Both shape and semantic labels gradually emerge in a coarse-to-fine manner. *Middle*: vertical slice through the scene, with color-coded octree depth (or equivalently, voxel size), using the proposed splitting criterion. *Bottom*: same, but with naive splitting criterion.

dual form E_l^{pd}:

$$\min_{\mathbf{x}_l} \max_{\boldsymbol{\nu}_l,\boldsymbol{\lambda}_l,\boldsymbol{\theta}_l} E_l^{pd}(\mathbf{x}_l;\boldsymbol{\nu}_l,\boldsymbol{\lambda}_l,\boldsymbol{\theta}_l) = \begin{pmatrix}\boldsymbol{\nu}_l\\ \boldsymbol{\lambda}_l\\ \boldsymbol{\theta}_l\end{pmatrix} \mathcal{M}_l \begin{pmatrix}\boldsymbol{x}_l^i\\ \boldsymbol{x}_l^{ij}\end{pmatrix} + \sum_{s\in\Omega^l}\sum_i \rho_s^i x_s^i + \sum_{i<j}\Phi_l^{ij}(x_s^{ij}-x_s^{ji}) + \iota_{\geq 0}(x_s^{ij}) + \iota_{\geq 0}(x_s^{ji})$$

$$\text{with } \mathcal{M}_l := \mathcal{M}_{L_N}\mathcal{A}_{l,L_N}. \tag{A.3}$$

Here we used $\mathcal{M}_{L_N}$ as the matrix encoding the full constraint set at maximal (grid) resolution. The level-dependent multipliers $\boldsymbol{\nu}_l, \boldsymbol{\lambda}_l$ and $\boldsymbol{\theta}_l$ follow by combining constraints that occur multiple times, and eliminating redundant ones (*e.g.* inside a coarse voxel). In this form we can apply the primal-dual algorithm [3] to find a saddle-point for E_l^{pd}, which in our case reads:

$$\begin{pmatrix}\boldsymbol{\nu}_l\\ \boldsymbol{\lambda}_l\\ \boldsymbol{\theta}_l\end{pmatrix}^{n+1} = \begin{pmatrix}\boldsymbol{\nu}_l\\ \boldsymbol{\lambda}_l\\ \boldsymbol{\theta}_l\end{pmatrix}^{n} + \sigma\mathbf{M}_l\mathbf{x}_l^n$$

$$\mathbf{x}_l^{n+1} = (I+\partial G)^{-1}\left(\mathbf{x}_l^n - \tau\mathbf{M}_l^\intercal\left(2\begin{pmatrix}\boldsymbol{\nu}_l\\ \boldsymbol{\lambda}_l\\ \boldsymbol{\theta}_l\end{pmatrix}^{n+1} - \begin{pmatrix}\boldsymbol{\nu}_l\\ \boldsymbol{\lambda}_l\\ \boldsymbol{\theta}_l\end{pmatrix}^{n}\right)\right)$$

$$\text{with } G(\mathbf{x}_l) := \sum_{s\in\Omega^l}\sum_i \rho_s^i x_s^i + \sum_{i<j}\Phi_l^{ij}(x_s^{ij}-x_s^{ji}) + \iota_{\geq 0}(x_s^{ij}) + \iota_{\geq 0}(x_s^{ji}). \tag{A.4}$$

Overrelaxation $2(\boldsymbol{\nu}_l,\boldsymbol{\lambda}_l,\boldsymbol{\theta}_l)^{n+1} - (\boldsymbol{\nu}_l,\boldsymbol{\lambda}_l,\boldsymbol{\theta}_l)^n$ on the dual variables proves to be especially memory efficient. In practice only the update of the indicator functions x^i affects multiple voxels. Thus, it is sufficient to only keep track of $(x_s^i)^n$. The respective descent step for the transition variables $\boldsymbol{x}^{ij}$ can be updated on-the-fly, without requiring additional storage.

The proximal step can be solved locally for each voxel. While the update of the indicator functions x_s^i is simple (per voxel s and label i subtract ρ_s^i), the update of the transition functions x_s^{ij} is more involved. We apply a variant of Dykstra's algorithm [2], described below in Sec. 6.1. The algorithm converges as soon as the step sizes τ, σ fulfill $\tau\sigma|\mathbf{M}_l|^2 < 1$, where we used $|\mathbf{M}_l|$ to denote the operator norm of $\mathbf{M}_l$. In practice, instead of using constant step sizes, we use the pre-conditioning scheme of [6] to accelerate convergence. After refinement, the previous solution is lifted to the next level according to the prolongation operator $\mathcal{A}_{l,l+1}$.

Practical issues. Recall that the data term for a voxel s at any level l ($s \in \Omega^l$) is computed from the data term at the finest resolution L_N via $\rho_s^i := \sum_{\bar{s}\in\Omega^{L_N}\cap s}\rho_{\bar{s}}^i A_{l,L_N}^I(s,\bar{s})$. To limit memory consumption, one can thus tile the grid Ω and compute the unary terms independently and in parallel. During refinement the data costs can then be loaded and constructed in parts. To accelerate the inference, we assign a small value σ_s^{sky} to voxels s, if they are likely to belong to the freespace. These are all voxels that were not assigned any data cost according to Eq. (3) from the paper, and whose depth is lower than the observed one in all images.

6.1. Proxmap for the Minkowski sum of convex sets

While most parts of the primal-dual optimization stated above are straight-forward, the proximity step w.r.t. the transition variables is a bit more involved. We must solve the following sub-problem per voxel:

$$\arg\min_{x^{ij},x^{ji}} \frac{1}{2}||x^{ij} - \overline{x^{ij}}||^2 + \frac{1}{2}||x^{ji} - \overline{x^{ji}}||^2 + \sum_{k:W_k\in\mathcal{W}} \sup_{w\in W_k} w^\mathsf{T}(x^{ij} - x^{ji}) + \iota_{\geq 0}(x^{ij}) + \iota_{\geq 0}(x^{ji}) \tag{A.5}$$

Here, the Minkowski sum, $\mathcal{W}$, is assumed to consist of K different Wulff shapes $W_1, \ldots, W_K \in \mathcal{W}$. Introducing auxiliary variables $\{y_k, z_k\}_{k=0}^K$ and additional Lagrange multipliers $\{\mu_k, \lambda_k\}_{k=0}^K$, we can decouple the argument within the regularizer:

$$\min_{x^{ij},x^{ji},y_k,z_k} \max_{\mu_k,\lambda_k} \frac{1}{2}||x^{ij} - \overline{x^{ij}}||^2 + \frac{1}{2}||x^{ji} - \overline{x^{ji}}||^2 + \sum_{k:W_k\in\mathcal{W}} \sup_{w\in W_k} w^\mathsf{T}(y_k - z_k) + \iota_{\geq 0}(y_0) + \iota_{\geq 0}(z_0) - \sum_{k=0}^{K} \lambda_k^\mathsf{T}(x^{ij} - y_k) - \mu_k^\mathsf{T}(x^{ji} - z_k) \tag{A.6}$$

Optimality w.r.t. x^{ij}, x^{ji} implies:

$$x^{ij} = \overline{x^{ij}} + \sum_{k=0}^{K} \lambda_k \quad \text{and} \quad x^{ji} = \overline{x^{ji}} + \sum_{k=0}^{K} \mu_k. \tag{A.7}$$

This leads to:

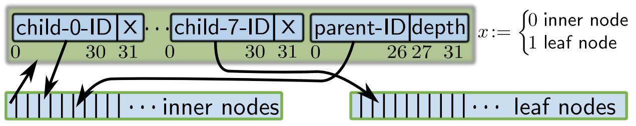
$$\min_{y_k,z_k} \max_{\mu_k,\lambda_k} \frac{-1}{2}||\sum_{k=0}^{K} \lambda_k - \overline{x^{ij}}||^2 + \frac{-1}{2}||\sum_{k=0}^{K} \mu_k - \overline{x^{ji}}||^2 + \sum_{k:W_k\in\mathcal{W}} \sup_{w\in W_k} w^\mathsf{T}(y_k - z_k) + \iota_{\geq 0}(y_0) + \iota_{\geq 0}(z_0) + \sum_{k=0}^{K} \lambda_k^\mathsf{T} y_k + \mu_k^\mathsf{T} z_k \,. \tag{A.8}$$

Using the fact that the support function of a set is the conjugate of its indicator function, we add $\pm \sum_{k=1}^K \lambda_k^\mathsf{T} z_k$ and "dualize" the regularizer and positivity constraints:

$$\max_{\mu_k,\lambda_k} \min_{z_k} \frac{-1}{2}||\sum_{k=0}^{K} \lambda_k - \overline{x^{ij}}||^2 + \frac{-1}{2}||\sum_{k=0}^{K} \mu_k - \overline{x^{ji}}||^2 - \sum_{k:W_k\in\mathcal{W}} \iota_{W_k}(-\lambda_k) - \iota_{\leq 0}(-\lambda_0) - \iota_{\leq 0}(-\mu_0) + \sum_{k=1}^{K} (\lambda_k + \mu_k)^\mathsf{T} z_k. \tag{A.9}$$

The latter summand $\min_{z_k} \sum_{k=1}^K (\lambda_k + \mu_k)^\mathsf{T} z_k$ requires $\lambda_k = -\mu_k$, so we end up with:

$$\min_{\mu_0,\lambda_k} \frac{1}{2}||\sum_{k=0}^{K} \lambda_k - \overline{x^{ij}}||^2 + \frac{1}{2}||\sum_{k=0}^{K} \lambda_k + \overline{x^{ji}}||^2 + \sum_{k:W_k\in\mathcal{W}} \iota_{W_k}(-\lambda_k) + \iota_{\geq 0}(\lambda_0) + \iota_{\geq 0}(\mu_0). \tag{A.10}$$


child-0-ID
X
child-7-ID
X
parent-ID
depth
0
30
31
0
30
31
0
26
27
31
$x := \begin{cases} 0 \text{ inner node} \\ 1 \text{ leaf node} \end{cases}$
inner nodes
leaf nodes
Figure 6. A simple octree structure. Inner and leaf nodes are kept in two separate arrays. Inner nodes hold pointers to their children, the data is only stored in the leaf nodes.

In that form, a certain number of iterations of block coordinate descent on the dual variables $\{\lambda_k\}_{k=0\ldots K}, \mu_0$ and (A.7) will deliver the solution to the the proximal step (A.5).

6.2. Octree structure

As mentioned in section 4.1 of the paper, the employed octree data structure is rather simple (Fig. 6). A quick test (always splitting each voxel) revealed an overhead of a factor 1.6 in computation time, when compared to the grid structure. For the uniform grid, the neighborhood relation of adjacent voxels is directly accessible. In our octree implementation we use a simple recursive procedure. Yet, more sophisticated octree models exist [5], which are specifically tailored for fast neighbor retrieval. Constructing a dual octree [5] promises a significant speedup with similar memory usage. In future work we plan to investigate whether our implementation could benefit from this kind of data structure.

7. Proofs

In this section we formally derive a number of claims which were stated without proof in the paper.

7.1. Prolongation operator

We first show that property **(iii)** in section 4.2 is fulfilled with equality:

$$E_l(\mathbf{x}_l) = E_{l+1}(\mathcal{A}_{l,l+1}\mathbf{x}_l) \tag{A.11}$$

According to Eq. (7) in the paper two properties need to be fulfilled:

$$E_l(\mathbf{x}_l) = E(\mathcal{A}_{l,L_N}\mathbf{x}_l) \text{ and} \tag{A.12}$$
$$\mathcal{A}_{l+1,l+2}\mathcal{A}_{l,l+1} = \mathcal{A}_{l,l+2} \,, \tag{A.13}$$

with L_N the maximal resolution in the octree. First, note that the resolution-dependent energy was defined according to (A.12). To prove (A.13) we first repeat the definition of $\mathcal{A}$:

$$\mathcal{A} := \left[(\mathcal{A}^I)^\mathsf{T}; (\mathcal{A}^{IJ})^\mathsf{T}\right]^\mathsf{T}. \tag{A.14}$$

Again the hierarchical structure allows us to specify mappings only for a single coarse parent voxel s and one of its children $\bar{s}$. The full operator $\mathcal{A}$ in (A.14) is simply a concatenation of its parts. Indicator variables are transformed according to:

$$\mathcal{A}^I_{l,L}(s,\bar{s}) := \left[A^I_{l,L}|\mathbf{0}_{l,L}\right],\ A^I_{l,L}\in\mathbb{R}^{M\times M}, \mathbf{0}_{l,L}\in\mathbb{R}^{M\times 3M^2},$$
$$A^I_{l,L}(i,j) = \begin{cases}1 & \text{iff } i=j\\ 0 & \text{otherwise.}\end{cases} \tag{A.15}$$

Recall that $\partial_{e_k} s$ denotes the boundary of voxel s in direction e_k. The second part of $\mathcal{A}$, the prolongation of the transition variables, is given by:

$$\mathcal{A}^{IJ}_{l,L}(s,\bar{s}) := \left[B^I_{l,L}|B^{IJ}_{l,L}\right]$$
$$B^I_{l,L}\in\mathbb{R}^{3M^2\times M}$$
$$B^{IJ}_{l,L}\in\mathbb{R}^{3M^2\times 3M^2}$$
$$B^I_{l,L}((i,i,k),(i)) = \begin{cases}1 & \text{iff } \partial_{e_k}\bar{s}\not\subset\partial_{e_k}s\\ 0 & \text{else}\end{cases} \tag{A.16}$$
$$B^{IJ}_{l,L}((i,j,k),(i,j,k)) = \begin{cases}1 & \text{iff } \partial_{e_k}\bar{s}\subset\partial_{e_k}s\\ 0 & \text{else.}\end{cases}$$

To show (A.13), we denote a parent voxel $s\in\Omega^l$, one of its children $\bar{s}\in\Omega^{l+1}\cap s$ and a "grand-child" of s with $\bar{\bar{s}}\in\Omega^{l+2}\cap\bar{s}$, and abbreviate $l_{01} := \{l, l+1\}$, $l_{12} := \{l+1, l+2\}$, $l_{02} := \{l, l+2\}$.

$$\mathcal{A}_{l_{12}}(\bar{s},\bar{\bar{s}})\mathcal{A}_{l_{01}}(s,\bar{s}) =$$
$$\left[\begin{array}{c|c} A^I_{l_{12}}A^I_{l_{01}} & \mathbf{0}\\ \hline B^I_{l_{12}}A^I_{l_{01}} + B^{IJ}_{l_{12}}B^I_{l_{01}} & B^{IJ}_{l_{12}}B^{IJ}_{l_{01}}\end{array}\right] := \left[\begin{array}{c|c} C^I_{l_{02}} & \mathbf{0}\\ \hline D^I_{l_{02}} & D^{IJ}_{l_{02}}\end{array}\right]$$
$$C^I_{l_{02}}(i,j) = \begin{cases}1 & \text{iff } i=j\\ 0 & \text{else.}\end{cases} \tag{A.17}$$
$$D^I_{l_{02}}((i,i,k),(i)) = \begin{cases}1 \text{ iff } \partial_{e_k}\bar{\bar{s}}\not\subset\partial_{e_k}\bar{s}\ \vee\\ \qquad(\partial_{e_k}\bar{\bar{s}}\subset\partial_{e_k}\bar{s}\wedge\partial_{e_k}\bar{s}\not\subset\partial_{e_k}s)\\ 0 \text{ else}\end{cases}$$
$$D^{IJ}_{l_{02}}((i,j,k),(i,j,k)) = \begin{cases}1 \text{ iff } \partial_{e_k}\bar{s}\subset\partial_{e_k}s\wedge\partial_{e_k}\bar{\bar{s}}\subset\partial_{e_k}\bar{s}\\ 0 \text{ else.}\end{cases}$$

In its last form the equivalence to the definition of $\mathcal{A}^I_{l_{02}}$ and $\mathcal{A}^{IJ}_{l_{02}}$ (*c.f*. Eqs. (9,10) in the paper) is apparent:

$$C^I_{l_{02}}(i,j) = \begin{cases}1 & \text{iff } i=j\\ 0 & \text{else}\end{cases}$$
$$D^I_{l_{02}}((i,i,k),(i)) = \begin{cases}1 & \text{iff } \partial_{e_k}\bar{\bar{s}}\not\subset\partial_{e_k}s\\ 0 & \text{else}\end{cases} \tag{A.18}$$
$$D^{IJ}_{l_{02}}((i,j,k),(i,j,k)) = \begin{cases}1 & \text{iff } \partial_{e_k}\bar{\bar{s}}\subset\partial_{e_k}s\\ 0 & \text{else.}\end{cases}$$

This concludes the proof.

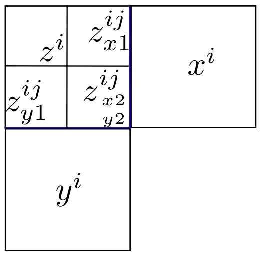


Figure 7. Configuration of the counterexample. (*c.f*. Sec. 7.2).

7.2. Equality constraints on the transition variables

In this section we show that, by only demanding equality in the indicator functions, our scheme does not derive from the "as-tight-as-possible" principle alone. We give the following 2D counterexample with three labels, and three pixels arranged in a clockwise rotated L-shape configuration (*c.f*. Fig. 7). The central one holds variables z, its right neighbor x and the pixel at the bottom variables y. We set the data term and regularizer as follows:

$$\rho^i_z := (0, 0.999, 0)$$
$$\rho^i_x := (0, 0, 0.999)$$
$$\rho^i_y := (0.999, 0, 0) \tag{A.19}$$
$$\phi^{01}(z) = \phi^{12}(z) = 0.5||z||_2$$
$$\phi^{02}(z) = 0.625||z||_2.$$

Assuming one level difference to the maximal resolution, an optimal solution is:

$$x = y = z := (\frac{1}{3}, \frac{1}{3}, \frac{1}{3}),$$
$$\text{with } z^{ij} = 0\ \forall\, i\neq j \tag{A.20}$$

with an energy of 0.999. Here, any feasible assignment with $z^{ij} = 0\,\forall i\neq j$ is optimal. In contrast, introducing transition variables (and handling constraints) at the level of the grid, the optimal solution becomes:

$$z := (\frac{1}{2}, 0, \frac{1}{2}) \quad , \quad x := (\frac{1}{2}, \frac{1}{2}, 0) \quad , \quad y := (0, \frac{1}{2}, \frac{1}{2})$$
with
$$z^{00}_{x_1} = z^{21}_{x_1} = z^{01}_{y_1} = z^{11}_{y_1} = 0.5,$$
$$z^{00}_{x_2} = z^{21}_{x_2} = z^{01}_{y_2} = z^{22}_{y_2} = 0.45, \tag{A.21}$$
$$z^{01}_{x_2} = z^{20}_{x_2} = z^{02}_{y_2} = z^{21}_{y_2} = 0.05, \text{ and}$$
$$z^{ij}_* = 0 \text{ for all other } i,j \text{ and } * = x_1, x_2, y_1, y_2.$$

Although the solution appears significantly different, the energy of 0.994 is only slightly lower. To prove optimality one could check the respective Lagrangian—we found both solutions with a Matlab program, and omit further tedious details at this point.

Discussion. The analysis above shows that boundary transitions at coarse voxels could be modeled more accurately by maintaining additional pseudo-marginals x^{ij} for each voxel. Aiming for a low memory footprint, such a procedure does not appear very reasonable. A situation as depicted in the example above should better be resolved by refining the affected voxels, in order to allow for a more accurate transition boundary. Recall that we require the resolution of adjacent voxels to differ by at most one level, and split a voxel whenever its neighborhood contains any voxels with a different dominant label. Both conditions will eventually lead to the splitting of affected regions.

References

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

[2] 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. 6

[3] A. Chambolle and T. Pock. A first-order primal-dual algorithm for convex problems with applications to imaging. *JMIV*, 2011. 6

[4] N. Chehata, L. Guo, and C. Mallet. Airborne lidar feature selection for urban classification using random forests. *IntArchPhRS*, 2009. 4, 5

[5] T. Lewiner, V. Mello, A. Peixoto, S. Pesco, and H. Lopes. Fast generation of pointerless octree duals. In *Symposium on Geometry Processing 2010 (Computer Graphics Forum)*, 2010. 7

[6] T. Pock and A. Chambolle. Diagonal preconditioning for first order primal-dual algorithms in convex optimization. In *ICCV*, 2011. 6