# Incremental Dense Semantic Stereo Fusion for Large-Scale Semantic Scene Reconstruction

Vibhav Vineet* Ondrej Miksik* Morten Lidegaard Matthias Nießner Stuart Golodetz
Victor A. Prisacariu Olaf Kähler David W. Murray Shahram Izadi Patrick Pérez
Philip H. S. Torr

*Abstract*— **Our abilities in scene understanding, which allow us to perceive the 3D structure of our surroundings and intuitively recognise the objects we see, are things that we largely take for granted, but for robots, the task of understanding large scenes quickly remains extremely challenging. Recently, scene understanding approaches based on 3D reconstruction and semantic segmentation have become popular, but existing methods either do not scale, fail outdoors, provide only sparse reconstructions or are rather slow. In this paper, we build on a recent hash-based technique for large-scale fusion and an efficient mean-field inference algorithm for densely-connected CRFs to present what to our knowledge is the first system that can perform dense, large-scale, outdoor semantic reconstruction of a scene in (near) real time. We also present a 'semantic fusion' approach that allows us to handle dynamic objects more effectively than previous approaches. We demonstrate the effectiveness of our approach on the KITTI dataset, and provide qualitative and quantitative results showing high-quality dense reconstruction and labelling of a number of scenes.**

## I. Introduction

As we navigate the world, for example when driving a car from our home to the work place, we constantly perceive the 3D structure of the environment around us and recognise objects within it. Such capabilities help us in our everyday lives and allow us free and accurate movement even in unfamiliar places.

Building a system that can automatically perform incremental real-time dense large-scale reconstruction and semantic segmentation, as illustrated in Fig. 1, is a crucial prerequisite for a variety of applications, including robot navigation [1], [2], semantic mapping [3], [4], wearable and/or assistive technology [5], [6], and change detection [7]. However, despite the large body of literature motivated by such applications [3], [4], [8]–[12], most existing approaches suffer from a variety of limitations. Offline reconstruction methods can achieve impressive results at city scale [13] and beyond, but cannot be used in a real-time setting. Sparse online reconstructions [14]–[17] were historically favoured over dense ones due to their lower computational requirements and the difficulties of acquiring adequate input for dense methods, but sparse maps are not guaranteed to contain objects of interest (*e.g.* traffic lights, signs). Dense reconstructions working on a regular voxel grid [18]–[20] are limited to small volumes due to memory requirements. This has been addressed by approaches that use scalable data structures and stream data between GPU and CPU memory [21], [22], but they use Kinect-like cameras that only work indoors [9], [10]. Approaches working outdoors usually take significant time to run [4], [8], [11], [23], do not work incrementally [12] or rely on LIDAR data [24]. Existing systems also do not cope well with moving objects. Ideally, we believe a method should

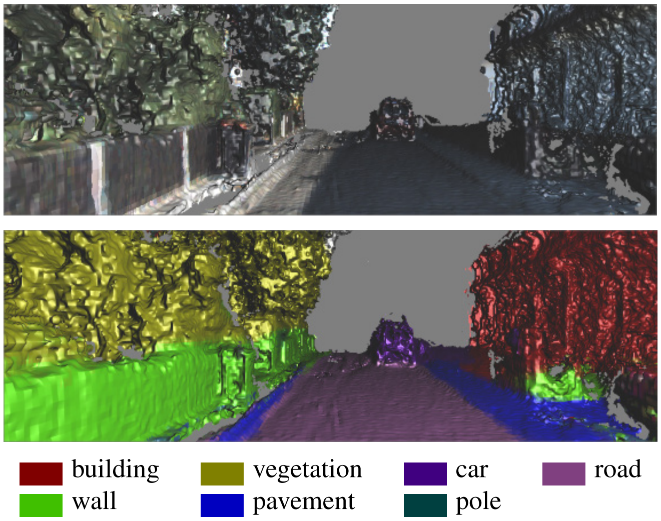

Fig. 1: Incremental reconstruction (top) and semantic segmentation (bottom) from our system, as seen from a moving platform on-the-fly (*i.e.* not a final mesh).

1) be able to incrementally build a dense semantic 3D map of any indoor or outdoor environment at any scale;
2) perform both tasks on-the-fly at real-time rates;
3) be amenable to handling moving objects.

In this paper, we propose an end-to-end system that can process the data incrementally and perform real-time dense stereo reconstruction and semantic segmentation of unbounded outdoor environments. The system outputs a per-voxel probability distribution instead of a single label (soft predictions are desirable in robotics, as the vision output is usually fed as input into other subsystems). Our system is also able to handle moving objects more effectively

* V. Vineet and O. Miksik assert joint first authorship.
{vibhav.vineet, ondra.miksik}@gmail.com
V. Vineet and M. Nießner are with Stanford, California, US.
O. Miksik, M. Lidegaard, S. Golodetz, V. Prisacariu, O. Kähler, D. Murray and P. Torr are with the University of Oxford, UK.
S. Izadi is with Microsoft Research, Redmond, Washington, US.
P. Pérez is with Technicolor R&I, Cesson Sévigné, FR.

This research was supported by Technicolor, EPSRC, the Leverhulme Trust and the ERC grant ERC-2012-AdG 321162-HELIOS. SG was funded via a Royal Society Brian Mercer Award for Innovation awarded to S. Hicks.

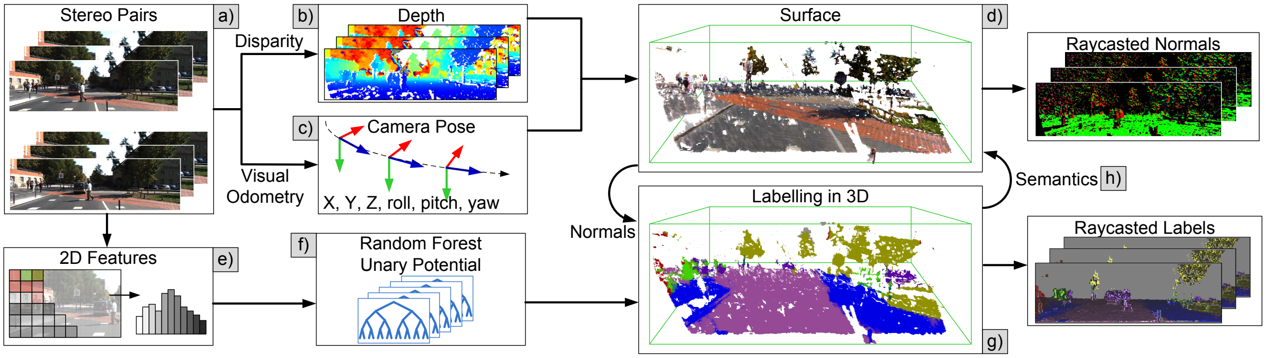

Fig. 2: Overview of our system: (a) given stereo image pairs, we (b) generate depth and (c) estimate 6 DoF camera pose using visual odometry in parallel. Next, we (d) fuse the depth into a common 3D map. We also (e) extract features, (f) evaluate unary potentials for each voxel and (g) perform inference over a densely-connected pairwise 3D random field to generate a high-quality labelling, which (h) controls fusion weights.

than prior approaches by incorporating knowledge of object classes into the reconstruction process. In order to achieve fast test times, we extensively use the computational power of modern GPUs.

Our goal is to *incrementally* build dense large-scale semantic *outdoor* maps. We emphasise the *incremental* nature of our approach, as many methods employ post-processing techniques such as surface densification, texture mapping and tone matting, etc. to produce high-quality or visually-plausible meshes. However, in most robotics settings it is the actual output produced on-the-fly that matters (Fig. 1). This consideration motivates both our reconstruction pipeline and the system as a whole.

At the core of our system (Fig. 2) is a scalable fusion approach [22] that allows the reconstruction of high-quality surfaces in virtually unbounded scenes. It achieves this by replacing the fixed dense 3D volumetric representation of the standard formulations [18]–[20] with a hash-table-driven counterpart that ignores unoccupied space in the target environment. Furthermore, whilst the standard formulations are limited by the available GPU memory, [22] swaps/streams map data between device and host memories as needed. This is key for scalable dense reconstruction, and to our knowledge has thus far only been used in *indoor* environments.

Outdoor scenes present several challenges: 1) Kinect-like cameras are less effective outdoors, whilst LIDARs are often too large for "wearable robotics" or produce overly sparse point-clouds: we thus prefer to rely on stereo, which is suitable for both large robots and wearable glasses/headsets; 2) as a result, the estimated depth [25] is usually more noisy; 3) the depth range is much larger and 4) dynamically moving objects are much more common and the camera itself may move significantly between consecutive frames (*e.g.* if mounted on a car, etc.). All of this makes data association for ICP camera pose estimation (as used in [20], [22]) harder, so we replaced it with a more reliable visual odometry [16].

Our semantic segmentation pipeline extracts 2D features and evaluates unary potentials based on random forest classifier predictions. It transfers these into the 3D volume, where we define a densely-connected CRF. Volumetric CRFs reduce the computational burden, since multiple pixels usually correspond to the same voxel, and enforce temporal consistency, since we label actual 3D surfaces. In order to efficiently infer the approximate maximum posterior marginal (MPM) solution, we propose an online volumetric mean-field inference technique that incrementally refines the marginals of a voxel across iterations, and design a volumetric filter that is suitable for parallel implementation. This allows us to run inference each frame (a single mean-field update takes 2-6ms), so our dynamic energy landscape changes slowly and only a few mean-field update iterations are required at each time step. We use our semantic labels to reinforce the weights in the fusion step, thereby allowing us to handle moving objects more effectively than prior approaches (see §IV).

All parts of our system are implemented on a GPU, except for visual odometry and disparity estimation, but both are easily parallelisable and can hence be switched to the GPU.

## II. Related Work

### A. Reconstruction

Recently, [26] demonstrated large-scale semi-dense reconstruction using only a monocular camera. Early real-time dense approaches [18], [19] were able to estimate depth from monocular input, but their use of a regular voxel grid limited reconstruction to small volumes due to memory requirements. KinectFusion [20] directly sensed depth using active sensors and fused noisy depth measurements of the perceived scene over time to recover high-quality surfaces, but suffered from the same scalability issue. This drawback has since been removed by scalable approaches that use either a voxel hierarchy [21] or voxel block hashing [22] to avoid storing unnecessary data for free space, and stream individual trees in the hierarchy or voxel blocks between the GPU and CPU to allow scaling to unbounded scenes. The hashing approach has the advantage of supporting constant-time lookups of voxel blocks, whereas lookups even in a balanced hierarchy are logarithmic in the number of blocks.

### B. Semantic Segmentation

Many approaches have been proposed in this field [3], [4], [8]–[12], [23], [24]. A summary of the most relevant papers and key attributes for outdoor large-scale reconstruction is provided in Tab. I.

Hermans *et al.* [9] use a random forest classifier and a dense 2D CRF, transfer the resulting marginals into 3D and

TABLE I: Comparison with some related work: O = outdoor, C = camera only, I = incremental, SDT = sparse data structures, S = host-device streaming, RT = real-time, MV = moving objects

| Method | O | C | I | SDT | S | RT | MV |
|---|---|---|---|---|---|---|---|
| Sengupta *et al.* [4] | ✓ | ✓ | | | out only | | |
| Valentin *et al.* [12] | ✓ | ✓ | | | | | |
| Häne *et al.* [8] | ✓ | ✓ | | | | N/A | |
| Kundu *et al.* [11] | ✓ | ✓ | ✓ | ✓ | | | |
| Hermans *et al.* [9] | | | ✓ | | | ✓ | |
| Hu *et al.* [27] | ✓ | | ✓ | ✓ | | ✓ | |
| **Ours** | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |

solve a 3D CRF to refine the predictions. Other shortcomings aside (see Tab. I), a CPU implementation requires heuristic scheduling (frame-skipping, etc.) to maintain a near-real-time frame rate. Sengupta *et al.* [4] proposed an offline method, which uses label transfer from 2D to 3D with sampling in a reversed order, which is computationally very expensive. They support streaming from RAM (CPU implementation), but not back again, *i.e.* they always start from scratch. Similarly, Valentin *et al.* [12] define a CRF over a reconstructed mesh, leading to faster inference. However, their method is not incremental, *i.e.* they need to reconstruct the whole scene first and then label it. Kundu *et al.* [11] proposed an offline method (based on personal communication) to integrate sparse (monocular) reconstruction with 2D semantic labels into a CRF model to determine the structure and labelling of a scene. Whilst their results are visually appealing, they do appear slightly voxelated when viewed at close range. Other methods [8], [18], [19], [23] share similar issues, whilst Hu *et al.* [27] relies on LIDAR data. In contrast to [4], [11], [12], [23], our method provides soft predictions.

## III. Large-scale outdoor reconstruction

Our system relies on passive stereo cameras, so we need to estimate the depth data that we want to fuse into our reconstruction each frame. In order to fuse the depth data, we also need to know the current pose of the camera, so we run a camera pose tracker in parallel with our depth estimation process. The following subsections describe the three parts of our reconstruction system (depth estimation, camera pose estimation and large-scale fusion) in more detail.

### *A. Depth Estimation*

To estimate depth from each stereo pair, we first estimate disparity and then convert it to depth using the equation $z_i = bf/d_i$, in which $z_i$ and $d_i$ are (respectively) the depth and disparity for the $i$'th pixel, $b$ is the stereo camera baseline and $f$ is the camera's focal length. For disparity estimation, we use the approach of Geiger *et al.* [25], which forms a triangulation on a set of support points that can be robustly matched. This reduces matching ambiguities and allows efficient exploitation of the disparity via constraints on the search space without requiring any global optimization. As a result, the method can be easily parallelised.

### *B. Camera Pose Estimation*

To estimate camera pose, we use the FOVIS feature-based visual odometry method [16]. First, an input pair of images is preprocessed using a Gaussian smoothing filter and a three-level image pyramid is built (each level corresponds to one octave in scale space). Then, a set of sparse local features is extracted by using a FAST corner detector with an adaptively-chosen threshold to detect a sufficient number of features. The feature extraction step is usually "biased" using bucketing to ensure that features are uniformly distributed across space and scale.

To constrain the feature matching stage to local search windows, an initial rotation of the image plane is estimated to deal with small motions in 3D. The matching stage associates the extracted features with descriptors and features are matched using a mutual-consistency check. A robust estimate is performed either by finding a maximal clique in the graph or using RANSAC, and the final transformation is estimated on the inliers. Robustness is further increased by using "keyframes", which reduces drift when the camera viewpoint does not change significantly. This can be further improved by using a full SLAM with loop closures, but this is beyond the scope of this paper.

### *C. Large-Scale Fusion*

Traditionally, KinectFusion-based approaches have fused depth inside a full, dense, volumetric 3D representation, which severely limits the size of reconstruction that can be handled. However, in real-world scenarios, a large part of this volume only contains free space, which does not need to be densely stored. By focusing the representation on the useful parts of the scene, we can use memory much more efficiently, which in turn enables much larger environments to be reconstructed. This insight has acted as a catalyst for works such as the hash-based method of [22] and the octree technique of [21].

We adopt the hash-based fusion method [22], which allocates space for only those voxels that fall within a small distance of the perceived surfaces in the scene. This space is organised into small voxel blocks. As with other depth fusion approaches, the dense areas are represented using an approximate truncated signed distance function (TSDF) [28]. Access to individual voxel blocks is mediated by a hash table. Given a known camera pose (§III-B), we use the following fusion pipeline:

*a) Allocation:* We ensure that voxel blocks are allocated for each voxel visible in the depth image. This is done by (i) back-projecting all visible voxels to voxel block world coordinates; (ii) looking up each unique voxel block in the hash table to determine whether or not it is currently allocated and (iii) allocating any blocks that are currently unallocated.

*b) Integration:* We integrate the current depth and colour frames into the volumetric data structure, using the conventional sliding-average technique of [28].

*c) Host-device streaming:* Although current GPUs have several GB of device memory, it is generally not enough to store a full large-scale reconstruction. To this end, data is streamed between the device and host. We only keep parts that are in or near the frustum. To implement this approach, we actively swap parts of the map between device and host

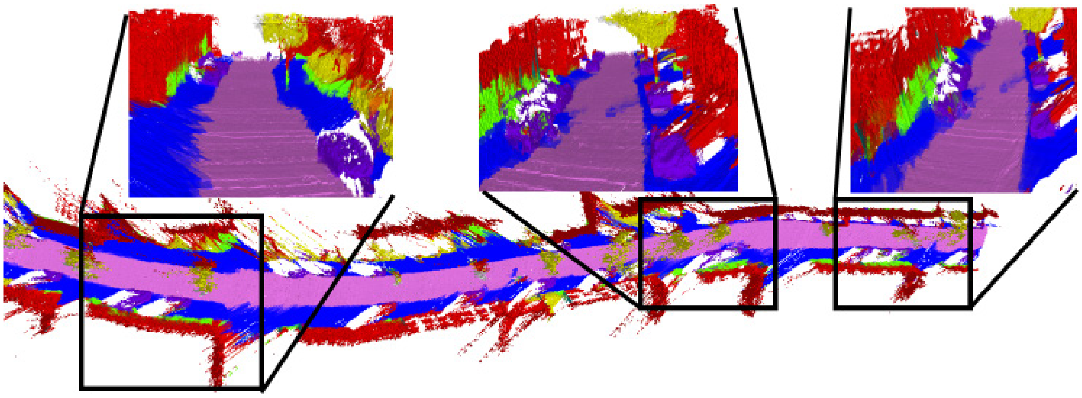

Fig. 3: Labelled mesh (output of our algorithm) for sequence 95 from the KITTI residential dataset, consisting of 268 stereo pairs. The close-up views show snapshots of the scene at several places along the route. See Fig. 1 for colour coding.

memory as they move in and out of view. Note that the scale of the reconstructions we can handle is still limited by host RAM in the current implementation. However, it would be simple to use the "swapping in and out" strategy between RAM and disk storage to achieve virtually unbounded reconstructions.

*d) Raycasting:* In every frame, the fused depth map is rendered from the current camera position.

## IV. Semantic Fusion

In the standard fusion approach, each voxel $i$ stores TSDF and colour measurements $\hat{T}^t_i$ and $\hat{C}^t_i$ at time $t$, together with weights $\hat{w}^t_{T,i}$ and $\hat{w}^t_{C,i}$ that capture our confidence in these measurements. These values are updated over time using the corresponding live TDSF and colour measurements $T^t_i$ and $C^t_i$, and some live weights $w^t_{T,i}$ and $w^t_{C,i}$ that can often be set to 1 to give simple running averages, *e.g.*:

$$\begin{aligned}\hat{w}^t_{T,i} &= \hat{w}^{t-1}_{T,i} + w^t_{T,i}\\ \hat{T}^t_i &= (\hat{w}^{t-1}_{T,i}\hat{T}^{t-1}_i + w^t_{T,i}T^t_i)/(\hat{w}^{t-1}_{T,i} + w^t_{T,i}) \qquad (1)\end{aligned}$$

This fusion step generally fails when there are moving objects in the scene, since static objects can become corrupted when we fuse in depth data from moving objects. This effect can be reduced by basing the live weights $w^t_{T,i}$ and $w^t_{C,i}$ on object class: by using higher weights for voxels that are labelled with moving object classes (*e.g.* car, pedestrian, etc.), we can speed up the process of fusing new data into our TSDF in places where the scene is more likely to be changing rapidly, which allows us to avoid being left with incorrect surfaces in places that briefly contained moving objects (note that the weights for voxels increase as we fuse in moving object data, and take some time to decrease again after the objects leave the voxels again). We call this adaptation of the original scheme "semantic fusion", and update our measurements using

$$\begin{aligned}\hat{w}^t_{T,i} &= \hat{w}^{t-1}_{T,i} + w_{\ell^t_i}\\ \hat{T}^t_i &= (\hat{w}^{t-1}_{T,i}\hat{T}^{t-1}_i + w_{\ell^t_i}T^t_i)/(\hat{w}^{t-1}_{T,i} + w_{\ell^t_i}), \qquad (2)\end{aligned}$$

in which $w_{\ell^t_i}$ is a per-class fixed weight corresponding to the the semantic label of voxel $i$ at time $t$.

This approach temporarily decreases the smoothness of the surface of affected voxels, but it allows us to preserve moving objects in a scene and avoids corruption of static objects. An example showing the way in which our semantic fusion approach is able to handle dynamically-moving objects is shown in Figure 6.

## V. Volumetric CRF and Mean-field Inference

### A. Model

We begin by defining a random field over random variables $\mathcal{X} = \{X_1, ..., X_N\}$, conditioned on the 3D surface $\mathbf{D}$. We assume that each discrete random variable $X_i$ is associated with a voxel $\mathcal{V} \in \{1, ..., N\}$ in the 3D reconstruction volume and takes a label $l_i$ from a finite label set $\mathcal{L} = \{l_1, ..., l_L\}$, corresponding to different object classes such as car, building or road. We formulate the problem of assigning object labels to the voxels as one of solving a volumetric, densely-connected, pairwise Conditional Random Field (CRF).

We define this CRF over the voxels in the current view frustum. Since our volumetric reconstruction is dynamically changing as new observations are captured, we have to deal with a dynamic energy function that keeps on changing in each iteration. Our CRF can be expressed as

$$\begin{aligned}P(\mathbf{X}|\mathbf{D}) &= \frac{1}{Z(\mathbf{D})}\exp(-E(\mathbf{X}|\mathbf{D}))\\ E(\mathbf{X}|\mathbf{D}) &= \sum_{i\in\mathcal{V}}\psi_u(X_i) + \sum_{i<j\in\mathcal{V}}\psi_p(X_i, X_j), \qquad (3)\end{aligned}$$

in which $E(\mathbf{X}|\mathbf{D})$ is the energy associated with a configuration $\mathbf{X}$, conditioned on the volumetric data $\mathbf{D}$, $Z(\mathbf{D}) = \sum_{\mathbf{X}'}\exp(-E(\mathbf{X}'|\mathbf{D}))$ is the (data-dependent) partition function and $\psi_u(\cdot)$ and $\psi_p(\cdot,\cdot)$ are the unary potential and pairwise potential functions, respectively, both implicitly conditioned on the data $\mathbf{D}$.

*Unary potentials:* Unary potential terms $\psi_u(\cdot)$ correspond to the cost of voxel $i$ taking an object label $l \in \mathcal{L}$. In order to evaluate the per-voxel unary potentials, we first train per-pixel object class models derived from TextonForest [29] using a set of per-pixel ground truth training images [4]. We use the 17-dimensional filter bank suggested by Shotton *et al.* [29], and follow Ladický *et al.* [30] by adding colour, histogram of oriented gradients (HOG), and pixel location features. At test time, we evaluate unary potentials in the image domain and then project them onto the voxels using the current camera pose and average them over time.

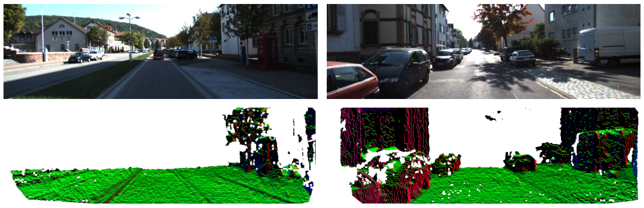

Fig. 4: An example of the normals we generate from the TSDF surfaces. These provide a lot of information about surface orientation and curvature that we use in pairwise potentials.

*Pairwise potentials:* The pairwise potential function $\psi_p(\cdot,\cdot)$ enforces consistency over pairs of random variables and thus generally leads to a smooth output. In our application, we use the weighted Potts model, which takes the form $\psi_{ij}(l,l') = \lambda_{ij}(\mathbf{f}_i,\mathbf{f}_j)[l \neq l']$, where [.] is the Iverson bracket (1 iff the condition in the square bracket is satisfied and 0 otherwise) and $\mathbf{f}_i, \mathbf{f}_j$ are the 3D features extracted from data $\mathbf{D}$ at the $i^{th}$ and $j^{th}$ voxels (respectively).

In the 2D segmentation domain, the cost $\lambda_{ij}$ of assigning different labels to neighbouring pixels is generally chosen such that it preserves image edges. Inspired by these edge-preserving smoothness costs, we make $\lambda_{ij}$ a weighted combination of Gaussian kernels (with unit covariance matrix) that depend on appearance and depth features:

$$\begin{aligned}\lambda_{ij} &= \sum_{m=1}^{M} \theta^m \lambda_{ij}^m(\mathbf{f}_i,\mathbf{f}_j) \\ &= \theta_{\mathrm{p}}^m e^{-\|\mathbf{p}_i-\mathbf{p}_j\|_2^2} + \theta_{\mathrm{a}}^m e^{-\|\mathbf{a}_i-\mathbf{a}_j\|_2^2} + \theta_{\mathrm{n}}^m e^{-\|\mathbf{n}_i-\mathbf{n}_j\|_2^2}\end{aligned} \tag{4}$$

Here, $\mathbf{p}_i$, $\mathbf{a}_i$ and $\mathbf{n}_i$ are respectively the 3D world coordinate position, RGB appearance, and surface normal vector of the reconstructed surface at voxel $i$, and $\theta_{\mathbf{p}}$, $\theta_{\mathbf{a}}$ and $\theta_{\mathbf{n}}$ are parameters obtained by cross-validation. Note that surface normals are calculated using the TSDF values [20]. In general, we obtain high-quality normals (see Fig. 4), which helps in achieving very smooth output.

### B. Efficient Mean-Field Inference

One of the most popular approaches for multi-label CRF inference has been graph-cuts based $\alpha$-expansion [31], which finds the maximum a posteriori (MAP) solution. However, graph-cuts leads to slow inference and is not easily parallelisable. Given the form of the energy function defined above, we follow the mean-field based optimization method, a filter-based variant that has been shown to be very efficient for densely-connected CRFs in 2D image segmentation [32], [33].

In the mean-field framework, we approximate the true distribution $P(\mathbf{X})$ by a family of $Q(\mathbf{X})$ distributions that factorize as the product of all components' marginals (components are independent) $Q(\mathbf{X}) = \prod_i Q_i(x_i)$. The mean-field inference then attempts to minimize the KL-divergence $D_{\mathrm{KL}}(Q||P)$ between the tractable distribution $Q$ and true distribution $P$. Under this assumption, the fixed point solution of the KL-divergence leads to the following mean-field update for all $j \neq i$ (refer to [34] for more details):

$$\begin{aligned}Q_i(X_i = l) =& \frac{1}{Z_i}\exp\{-\psi_u(X_i) - \\ & \sum_{l'\in\mathcal{L}}\sum_{j\neq i} Q_j(X_j = l')\psi_p(X_i,X_j)\}\end{aligned} \tag{5}$$

where $Z_i = \sum_{X_i=l\in\mathcal{L}} \exp\{-\psi_u(X_i) - \sum_{l'\in\mathcal{L}}\sum_{j\neq i} Q_j( X_j = l')\psi_p(X_i,X_j)\}$ is a constant normalizing the marginal at voxel $i$. The complexity of the mean-field update for the volumetric data is $\mathcal{O}(N^2)$.

Next, we discuss our online volumetric mean-field approach, which has been adapted from the 2D filtering-based mean-field approach we described above. Although this *online* mean-field approach has previously been applied in 2D [35], we believe this is the first time it has been applied in a 3D setting.

### C. Volumetric filtering-based mean-field

The most time-consuming step in the mean-field inference is the pairwise update, whose complexity is $\mathcal{O}(N^2)$. Now we will discuss how we reduce this complexity to $\mathcal{O}(N)$ for pairwise potentials taking the form of a weighted combination of Gaussian kernels. Our work is motivated by [32], [33], who show that fast approximate MPM inference can be achieved by applying cross bilateral filtering techniques.

First, we show why the mean-field update from Eq. 5 can be interpreted as filtering. To this end, we apply the transformation

$$\tilde{Q}_i^{(m)}(l) = \sum_{j\neq i} \lambda^m(\mathbf{f}_i,\mathbf{f}_j)Q_j(l) = [G^m \otimes Q(l)](\mathbf{f}_i) - Q_i(l), \tag{6}$$

in which $G^m$ is the Gaussian kernel corresponding to the $m^{th}$ component and $\otimes$ is the convolution operator. Since $\sum_{j\neq i} Q_j(x_j = l')\psi_p(x_i,x_j)$ can be written as $\sum_m w^{(m)}\tilde{Q}_i^{(m)}(l')$, and approximate Gaussian convolution is $\mathcal{O}(N)$, parallel updates can be efficiently approximated in $\mathcal{O}(MNL)$ time for the Potts model. The algorithm is run for a fixed number of iterations, and the MPM solution extracted by choosing $X_i \in \operatorname{argmax}_l Q_i(x_i = l)$ from soft predictions at the final iteration. We use high-dimensional filtering on the 3D volumetric data, where the filtering is a simple extension of the 2D permutohedral lattice-based filtering shown in [32] to 3D.

### D. Online mean-field

Given unlimited computation, one might run multiple update iterations until convergence. However, in our online system, we assume that the next frame's updates to the volume (and thus to the energy function) are not too radical, and so we can make the assumption that the $Q_i$ distributions can be temporally propagated from one frame to the next, rather than re-initialized (*e.g.* to uniform) at each frame. Thus, running even a *single iteration* of mean-field updates per frame effectively allows us to amortize an otherwise expensive inference operation over multiple frames and maintain real-time speeds.

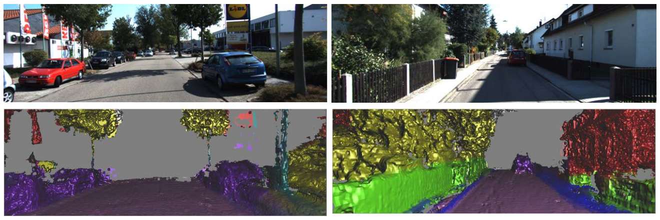

Fig. 5: Our approach not only reconstructs and labels entire outdoor scenes that include roads, pavements and buildings, but also accurately recovers thin objects such as lamp posts and trees. See Fig. 1 for colour coding.

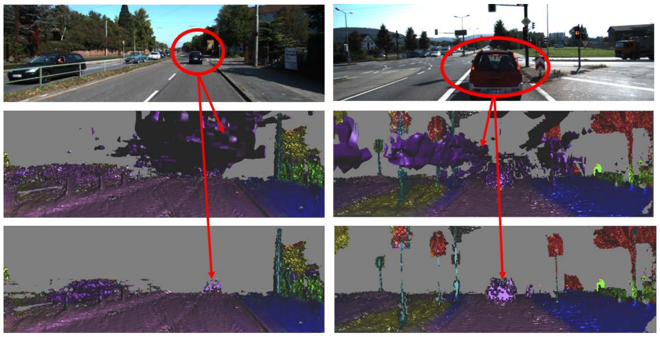

Fig. 6: Our semantic fusion technique enables us to avoid corrupting a static scene with data from moving objects. First row: input image; second row: reconstructed scene without semantic fusion; third row: reconstructed scene with semantic fusion. Note the way in which semantic fusion helps suppress the trail of spurious voxels that moving objects would normally leave behind. See Fig. 1 for colour coding.

As described above, the output of the classifier responses is used to update the unary potentials, which will, over several frames, impact the final segmentation that results from the online mean-field inference. However, to speed up convergence, rather than simply propagating the $Q_i^{t-1}$s from the previous frame, we instead provide the next iteration of mean-field updates with a weighted combination of $Q_i^{t-1}$ and the classifier prediction $P_u(x_i = l \mid \mathbf{D})$. We thus use

$$\bar{Q}_i^{t-1}(l) = \gamma Q_i^{t-1}(l) + (1-\gamma) P_u(X_i = l \mid \mathbf{D}) \qquad (7)$$

in place of $Q_i^{t-1}$, where $\gamma$ is a weighting parameter.

## VI. Experiments

We demonstrate the effectiveness of our approach for both 3D semantic segmentation and reconstruction. We evaluate our system on the KITTI dataset [36], which contains a variety of outdoor sequences, including a city, road and campus. All sequences were captured at a resolution of $1241 \times 376$ pixels using stereo cameras (with baseline 0.54m) mounted on the roof of a car. The car was also equipped with a Velodyne HDL-64E laser scanner (LIDAR). The KITTI dataset is very challenging since it contains many moving objects such as cars, pedestrians and bikes, and numerous changes in lighting conditions.

For both voxel labelling and reconstruction, we show our results on both static and dynamic scenes. This enables us to properly evaluate how well our approach handles motion. For static scenes, we used the dataset of Sengupta *et al.* [4], which consists of 45 training and 25 test images that are labelled with the following classes: road, building, vehicle, pedestrian, pavement, tree, sky, signage, post/pole and wall/fence. For dynamic scenes, we manually annotated sequences from the KITTI dataset that contained many moving objects. We compare the timings and accuracy achieved by our voxel-labelling approach against two baselines, Ladický *et al.* [30] and Sengupta *et al.* [4]. To evaluate our reconstruction results, we compare them with the depth data generated using Geiger *et al.*'s approach [25], using LIDAR data from the Velodyne scanner as ground truth. To perform qualitative and quantitative evaluation, we back-project the voxel labels and reconstructed surfaces onto the camera's image plane, ignoring those that are farther than 25 metres from the camera.

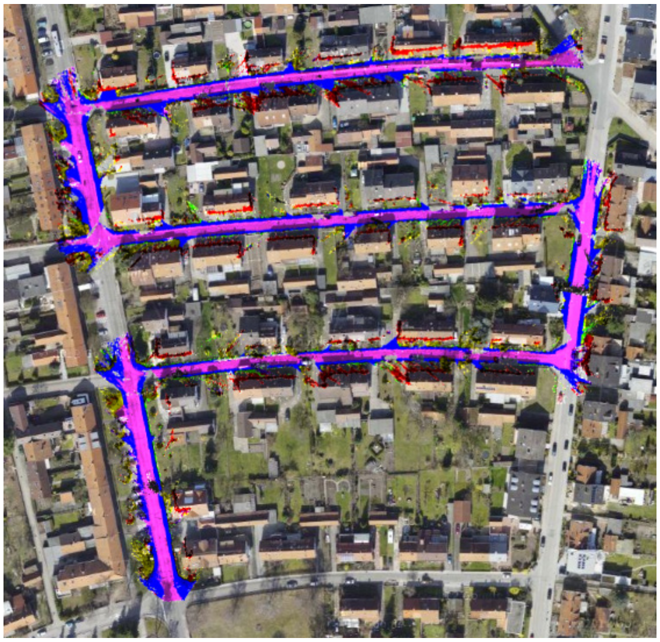

Fig. 7: A high-quality mesh recovered from the long (1000 images) sequence 5 of the KITTI odometry dataset, superimposed over the corresponding Google Earth image. This shows the ability of our method to reconstruct and label large scenes. See Fig. 1 for colour coding.

### A. Qualitative KITTI Results

First, we show some qualitative results for our semantic reconstruction approach. In Fig. 5, we highlight the ability of our approach not only to reconstruct and label entire outdoor scenes that include roads, pavements and buildings, but also to accurately recover thin objects such as lamp posts and trees. In Fig. 6, we show the advantages of our semantic fusion approach in handling moving objects (in this case, a car). Note in particular that with semantic fusion turned on, the static scene is far less corrupted by moving objects than it would be otherwise. Fig. 7 shows a high-quality mesh recovered from a long KITTI sequence (1000 images), superimposed over the corresponding Google Earth image. This shows the ability of our method to reconstruct and label large scenes. In Fig. 8, we show a close-up view of a semantic model produced using our method, in which the arrows indicate the image locations and their corresponding positions in the 3D model, and colours indicate the object labels. This shows that even though our approach is an incremental one, we are able to achieve smooth surfaces for outdoor scenes.

TABLE II: Quantitative results for our segmantic segmentation approach on the KITTI dataset. We compare global accuracy and intersection/union on both (a) static and (b) moving scenes. For static scenes, we compare our approach without semantic fusion [Ours(1)] against the state-of-the-art approaches of Ladický *et al.* [30] and Sengupta *et al.* [4]. For moving scenes, we compare our approach with semantic fusion [Ours(2)] against [Ours(1)] and [30].

(a) Static

| | Global Accuracy | | | Intersection/Union | | |
|---|---|---|---|---|---|---|
| Class | [30] | [4] | Ours(1) | [30] | [4] | Ours(1) |
| building | 97.0 | 96.1 | **97.2** | 86.1 | 83.8 | **88.3** |
| vegetation | 93.4 | 86.9 | **94.1** | 82.8 | 74.3 | **83.2** |
| car | 93.9 | 88.5 | **94.1** | 78.0 | 63.5 | **79.5** |
| road | 98.3 | 97.8 | **98.7** | 94.3 | **96.3** | 94.7 |
| wall | **48.5** | 46.1 | 47.8 | **47.5** | 45.2 | 46.3 |
| pavement | 91.3 | 46.1 | **91.8** | 73.4 | 68.4 | **73.8** |
| pole | 49.3 | 38.2 | **51.4** | 39.5 | 28.9 | **41.7** |
| Average | 81.7 | 71.4 | **82.2** | 71.7 | 65.8 | **72.5** |

(b) Moving

| | Global Accuracy | | | Intersection/Union | | |
|---|---|---|---|---|---|---|
| Class | [30] | Ours(1) | Ours(2) | [30] | Ours(1) | Ours(2) |
| building | 90.9 | 89.1 | **93.1** | 82.1 | 81.9 | **82.7** |
| vegetation | 89.2 | 66.9 | **92.1** | 77.6 | 64.3 | **79.0** |
| car | 92.1 | 78.5 | **94.3** | 72.0 | 56.4 | **77.5** |
| road | **98.6** | 87.8 | 97.7 | 91.3 | 86.3 | **92.1** |
| wall | 46.7 | 42.1 | **48.1** | 49.5 | 42.2 | **50.3** |
| pavement | 93.3 | 84.5 | **94.8** | 72.4 | 63.4 | **75.8** |
| pole | 46.2 | 36.7 | **47.4** | 34.1 | 24.6 | **36.7** |
| Average | 79.6 | 69.4 | **81.1** | 68.4 | 59.9 | **70.6** |

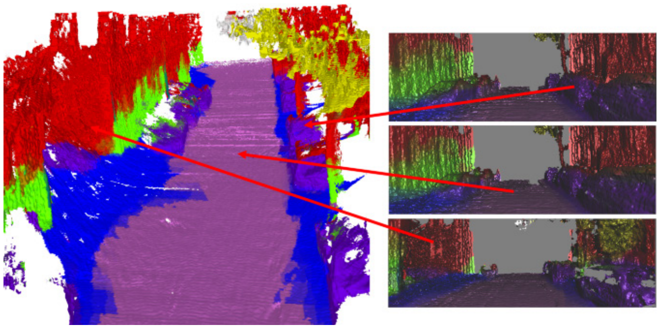

Fig. 8: A close-up view of a semantic model produced using our method, in which the arrows indicate the image locations and their corresponding positions in the 3D model, and colours indicate the object labels. This shows that even though our approach is an incremental one, we are able to achieve smooth surfaces for outdoor scenes. See Fig. 1 for colour coding.

### B. Quantitative KITTI Results

*Semantic Segmentation:* Next, we quantitatively evaluate the speed and accuracy of our mean-field-based volumetric labelling approach. Mean-field updates take roughly 20ms. Although the timings change as a function *e.g.* of the number of visible voxels, in all tests we performed we observed real-time performance. We assess the overall percentage of correctly-labelled voxels (global accuracy) and the intersection/union (I/U) score defined in terms of the true/false positives/negatives for a given class, *i.e.* TP/(TP+FP+FN).

Quantitative results for static scenes are shown in Tab. II(a). In comparison to the 2D approach of Ladický *et al.* [30], we achieve a $0.49\%$ improvement in global accuracy and a $0.84\%$ improvement in I/U score. We also significantly improve upon the 3D approach of Sengupta *et al.* [4], achieving a $10.8\%$ improvement in global accuracy and a $6.7\%$ improvement in I/U. More importantly, our approach achieves encouraging improvements in global accuracy and I/U for thin objects (*e.g.* poles).

In Tab. II(b), we evaluate the accuracy of our labellings on sequences containing many moving cars. We observe that our non-semantic fusion approach reduces accuracy by over $10\%$ in comparison to [30]; however, our semantic fusion approach improves overall accuracy by $1.5\%$. For cars, we observe an improvement of $2.2\%$ in global accuracy and $5.5\%$ in I/U. Note that our semantic fusion approach significantly improves both the global accuracy and I/U of our method, in both cases by over $10\%$. The improvements for cars are even more significant, highlighting the importance of using semantic fusion for scenes containing moving objects.

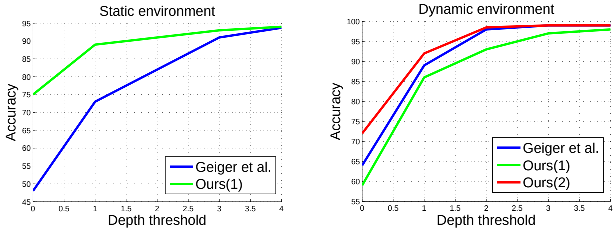

Fig. 9: Quantitative results for depth evaluation for static (left) and moving (right) scenes.

*Reconstruction:* Next, we quantitatively evaluate the efficiency and accuracy of our reconstruction approach. Camera tracking takes roughly 20ms, stereo estimation takes around 40ms (on our 12 core systems) and fusion takes 14ms. In order to evaluate accuracy, we follow the approach of Sengupta *et al.* [4], who measure the number of pixels whose distance (in terms of depth) from the ground truth (in our case the Velodyne data) after projection to the image plane is less than a fixed threshold.

Quantitative results for depth evaluation are summarised in Fig. 9 for both static and dynamic scenes. We observe that for static scenes, our non-semantic fusion approach itself achieves almost $90\%$ and $95\%$ accuracy when the thresholds are 1m and 4m respectively. We therefore achieve an improvement of almost $20\%$ over the initial depth estimated using the stereo output from Geiger *et al.*'s approach [25]. However, for sequences in which there are many moving objects, non-semantic fusion does not perform that well and leads to a decrease in accuracy of almost $5\%$ compared to Geiger *et al.*'s method. By contrast, our semantic fusion approach achieves an almost $5\%$ improvement in accuracy.

We would like to highlight that the real-time aspect of our semantic reconstruction pipeline does not include the feature evaluation time. However, features can be implemented on GPU to provide real-time performance, as shown in [37].

### C. Other Qualitative Results

Finally, we show additional qualitative results on four new, challenging sequences that we captured using a head-mounted stereo camera. Fig. 10 shows the final smooth semantic reconstructions obtained by running our mean-field inference procedure. The images clearly indicate the sharp boundaries that we manage to achieve between different conflicting semantic classes. For example, observe the extremely accurate boundary between the pavement and the road in the sequence in the third column. More results are provided in the supplementary video.

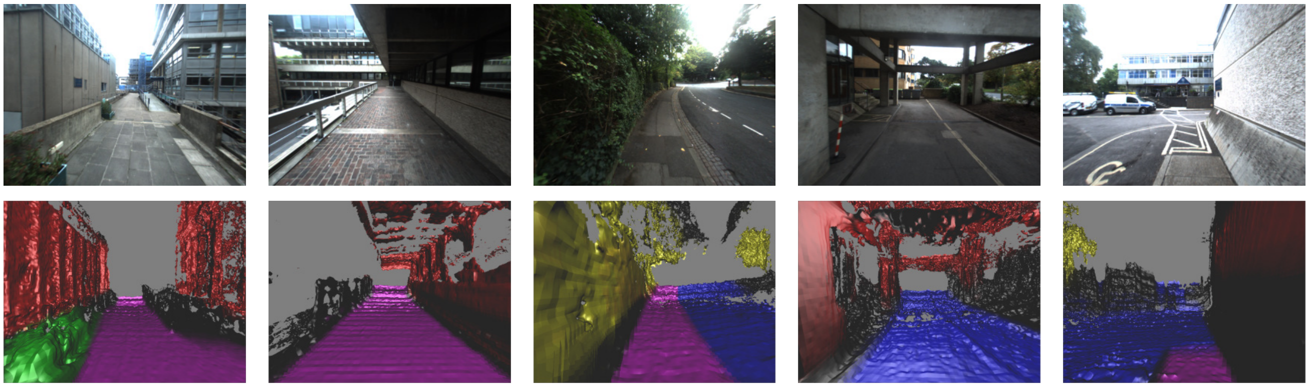

Fig. 10: Final labelling surfaces for four reconstructed sequences (the last two columns belong to the same sequence). See Fig. 1 for colour coding.

## VII. Conclusion

We have presented a robust and accurate approach for incremental dense large-scale semantic reconstruction of outdoor environments in real time from a stereo camera. At the core of our algorithm is a hash-based fusion approach for 3D reconstruction and a volumetric mean-field inference approach for object labelling. By performing reconstruction and recognition in tandem, we capture the synergy between the two tasks. By harnessing the processing power of modern GPUs, we can perform semantic reconstruction at real-time rates, even for large-scale environments. We have demonstrated our system's effectiveness for both high-quality dense reconstruction and scene labelling on the KITTI dataset.

Our paper offers many interesting avenues for further work. One area that we would like to explore is the enforcement of object-specific shape priors for 3D reconstruction. Currently, feature generation and learning of the class models have been done in an offline fashion. We would like to implement the online aspects of these tasks on GPU.

## References

[1] H. Dahlkamp, A. Kaehler, D. Stavens, S. Thrun, and G. Bradski, “Self-supervised Monocular Road Detection in Desert Terrain,” in *Robotics: Science and Systems*, 2006.

[2] C. Urmson et al., “Autonomous driving in urban environments: Boss and the urban challenge,” *JFR*, 2008.

[3] S. Sengupta, P. Sturgess, L. Ladický, and P. H. S. Torr, “Automatic Dense Visual Semantic Mapping from Street-Level Imagery,” in *IROS*, 2012.

[4] S. Sengupta, E. Greveson, A. Shahrokni, and P. H. S. Torr, “Urban 3D Semantic Modelling Using Stereo Vision,” in *ICRA*, 2013.

[5] Google, “ATAP Project Tango  Google,” 2014. [Online]. Available: http://www.google.com/atap/projecttango/

[6] S. L. Hicks, I. Wilson, L. Muhammed, J. Worsfold, S. M. Downes, and C. Kennard, “A Depth-Based Head-Mounted Visual Display to Aid Navigation in Partially Sighted Individuals,” *PLoS ONE*, 2013.

[7] A. Taneja, L. Ballan, and M. Pollefeys, “City-Scale Change Detection in Cadastral 3D Models using Images,” in *CVPR*, 2013.

[8] C. Häne, C. Zach, A. Cohen, R. Angst, and M. Pollefeys, “Joint 3D Scene Reconstruction and Class Segmentation,” in *CVPR*, 2013.

[9] A. Hermans, G. Floros, and B. Leibe, “Dense 3D Semantic Mapping of Indoor Scenes from RGB-D Images,” in *ICRA*, 2014.

[10] H. S. Koppula, A. Anand, T. Joachims, and A. Saxena, “Semantic Labeling of 3D Point Clouds for Indoor Scenes,” in *NIPS*, 2011.

[11] A. Kundu, Y. Li, F. Dellaert, F. Li, and J. M. Rehg, “Joint Semantic Segmentation and 3D Reconstruction from Monocular Video,” in *ECCV*, 2014.

[12] J. P. C. Valentin, S. Sengupta, J. Warrell, A. Shahrokni, and P. H. S. Torr, “Mesh Based Semantic Modelling for Indoor and Outdoor Scenes,” in *CVPR*, 2013.

[13] S. Agarwal, Y. Furukawa, N. Snavely, I. Simon, B. Curless, S. M. Seitz, and R. Szeliski, “Building Rome in a Day,” *CACM*, 2011.

[14] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, “MonoSLAM: Real-Time Single Camera SLAM,” *PAMI*, vol. 29, no. 6, 2007.

[15] G. Klein and D. Murray, “Parallel Tracking and Mapping for Small AR Workspaces,” in *ISMAR*, 2007.

[16] A. S. Huang, A. Bachrach, P. Henry, M. Krainin, D. Maturana, D. Fox, and N. Roy, “Visual Odometry and Mapping for Autonomous Flight Using an RGB-D Camera,” in *ISRR*, 2011.

[17] C. Forster, M. Pizzoli, and D. Scaramuzza, “SVO: Fast Semi-Direct Monocular Visual Odometry,” in *ICRA*, 2014.

[18] J. Stühmer, S. Gumhold, and D. Cremers, “Real-time dense geometry from a handheld camera,” in *DAGM*, 2010.

[19] R. A. Newcombe, S. J. Lovegrove, and A. J. Davison, “DTAM: Dense Tracking and Mapping in Real-Time,” in *ICCV*, 2011.

[20] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohli, J. Shotton, S. Hodges, and A. Fitzgibbon, “KinectFusion: Real-Time Dense Surface Mapping and Tracking,” in *ISMAR*, 2011.

[21] J. Chen, D. Bautembach, and S. Izadi, “Scalable Real-time Volumetric Surface Reconstruction,” *TOG*, vol. 32, no. 4, 2013.

[22] M. Nießner, M. Zollhöfer, S. Izadi, and M. Stamminger, “Real-time 3D Reconstruction at Scale using Voxel Hashing,” *TOG*, 2013.

[23] G. Floros and B. Leibe, “Joint 2D-3D Temporally Consistent Semantic Segmentation of Street Scenes,” in *CVPR*, 2012.

[24] D. Munoz, J. A. Bagnell, N. Vandapel, and M. Hebert, “Contextual Classification with Functional Max-Margin Markov Networks,” in *CVPR*, 2009.

[25] A. Geiger, M. Roser, and R. Urtasun, “Efficient Large-Scale Stereo Matching,” in *ACCV*, 2010.

[26] J. Engel, T. Schöps, and D. Cremers, “LSD-SLAM: Large-Scale Direct Monocular SLAM,” in *ECCV*, 2014.

[27] H. Hu, D. Munoz, J. A. Bagnell, and M. Hebert, “Efficient 3-d scene analysis from streaming data,” in *ICRA*, 2013.

[28] B. Curless and M. Levoy, “A Volumetric Method for Building Complex Models from Range Images,” in *SIGGRAPH*, 1996.

[29] J. Shotton, M. Johnson, and R. Cipolla, “Semantic texton forests for image categorization and segmentation,” in *2008 IEEE CVPR, 24-26 June 2008, Anchorage, Alaska, USA*, 2008.

[30] L. Ladický, C. Russell, P. Kohli, and P. H. S. Torr, “Associative Hierarchical Random Fields,” *PAMI*, vol. 36, no. 6, 2014.

[31] Y. Boykov, O. Veksler, and R. Zabih, “Fast Approximate Energy Minimization via Graph Cuts,” *PAMI*, vol. 23, no. 11, 2001.

[32] P. Krähenbühl and V. Koltun, “Efficient Inference in Fully Connected CRFs with Gaussian Edge Potentials,” in *NIPS*, 2011.

[33] V. Vineet, J. Warrell, and P. H. S. Torr, “Filter-based Mean-Field Inference for Random Fields with Higher-Order Terms and Product Label-Spaces,” in *ECCV*, 2012.

[34] D. Koller and N. Friedman, *Probabilistic Graphical Models - Principles and Techniques*. MIT Press, 2009.

[35] C. Medrano, J. Herrero, J. Martnez, and C. Orrite, “Mean field approach for tracking similar objects,” in *CVIU*, 2009.

[36] A. Geiger, P. Lenz, and R. Urtasun, “Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite,” in *CVPR*, 2012.

[37] V. A. Prisacariu and I. Reid, “fasthog - a real-time gpu implementation of hog,” in *University of Oxford, Tech. Report*, 2009.