LEGO: Learning Edge with Geometry all at Once by Watching Videos

Zhenheng Yang[1] Peng Wang[2] Yang Wang[2] Wei Xu[3] Ram Nevatia[1]

[1]University of Southern California [2]Baidu Research
[3]National Engineering Laboratory for Deep Learning Technology and Applications

Abstract

Learning to estimate 3D geometry in a single image by watching unlabeled videos via deep convolutional network is attracting significant attention. In this paper, we introduce a "3D as-smooth-as-possible (3D-ASAP)" prior inside the pipeline, which enables joint estimation of edges and 3D scene, yielding results with significant improvement in accuracy for fine detailed structures. Specifically, we define the 3D-ASAP prior by requiring that any two points recovered in 3D from an image should lie on an existing planar surface if no other cues provided. We design an unsupervised framework that Learns Edges and Geometry (depth, normal) all at Once (LEGO). The predicted edges are embedded into depth and surface normal smoothness terms, where pixels without edges in-between are constrained to satisfy the prior. In our framework, the predicted depths, normals and edges are forced to be consistent all the time. We conduct experiments on KITTI to evaluate our estimated geometry and CityScapes to perform edge evaluation. We show that in all of the tasks, i.e.*depth, normal and edge, our algorithm vastly outperforms other state-of-the-art (SOTA) algorithms, demonstrating the benefits of our approach.*

1. Introduction

Humans are highly competent in recovering the 3D geometry of observed natural scenes at very detailed level, even from a single image. Practically, being able to do detailed reconstruction for monocular images can be widely applied to many real-world applications such as augmented reality and robotics.

Recently, impressive progress [19, 64, 60] has been made to mimic detailed 3D reconstruction by training a deep network taking only unlabeled videos or stereo images as input and testing on monocular image, yielding even better depth estimation results than those of supervised methods [13] in outdoor scenarios. The core underlying idea is the supervision by view synthesis, where the frame of one view (source) is warped to another (target) based on the predicted depths and relative motions, and the photometric error between the warped and observed target frame is used to supervise the training. However, upon a closer look at the predicted results from [64] in Fig. 1(b) and [60] in Fig. 1(d), the estimated depths and normals (left and middle) are blurry and do not conform well to the scene geometry.

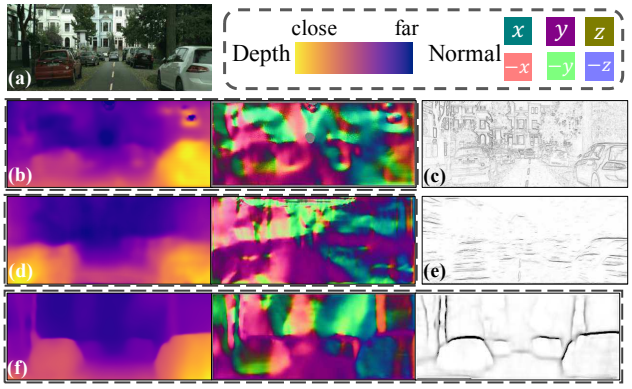


Figure 1: (a) Input image; (b) Depth and normal results by [64]; (c) Edges from image gradient; (d) Depth and normal results by [60]; (e) Unsupervised edge detection results by [37]; (f) Our unsupervised joint estimation of depth, normal and edge.

We argue that this is because the unsupervised learning pipelines are mostly optimizing the per-pixel photometric errors, while paying less attention to the geometrical edges. We use the term "geometrical edge" to include depth discontinuities and surface normal changes. This motivates us to jointly learn an edge representation with the geometry inside the pipeline, so that the two information reinforce each other. We come up with a framework that Learn Edge and Geometry all at Once (LEGO) with unsupversied learning. In our work, 3D geometry helps the model to discover mid-level edges by filtering out the internal edges inside the same surface (those from image gradient as shown in Fig. 1(c)). Conversely, the discovered edges can help the geometry estimation obtain long-range context awareness and non-local regularization, which pushes the model to generate results with fine details.

We formulate the interaction between the two by proposing a "as smooth as possible in 3D" (3D-ASAP) prior. It

requires all pixels recovered in 3D should lay in a same planar surface if no edge exists in-between, such that the edge and the geometrical smoothness are adversarial inside the learning pipeline, yielding consistent and visually satisfying results.

As shown in Fig. 1(f), the estimated depths and normals of LEGO have consistent structure with the 3D geometry. Compared to the results of SOTA unsupervised edge detection method [37] in Fig. 1(e), edge results generated by LEGO align well with the scene layout with fewer noises. The edges discovered in our pipeline is not necessarily semantic but geometrical, arguably alleviating the issues of confusing definition for supervised semantic edge predictions [39] that was questioned in [21].

We conducted extensive experiments over the public KITTI 2015 [18], CityScapes [10] and Make3D [45] datasets, and show that LEGO performs much better in depth prediction, especially when transferring the model cross different datasets (relatively 30% improvements over other SOTA methods). Additionally, LEGO achieves 20% improvement on normal estimation compared with [19], and 15% improvement on geometrical edges detection compared to previous unsupervised edge learning method [37]. Lastly, LEGO runs efficiently without much extra computation compared to [64, 60]. These demonstrate the efficiency and effectiveness of our approach. We plan to release our code upon the pulication of this paper.

2. Related Work

In this section, we briefly overview some traditional methods, and introduce current SOTA methods for unsupervised single view 3D geometry recovery and edge detection.

Structure from motion and single view geometry. Geometric based methods estimate 3D from a given video with feature matching, such as SFM [56], SLAM [41, 14] and DTAM [42], which could be effective and efficient in many cases. However, they can fail at where there is low texture, or drastic change of visual perspective *etc*.. More importantly, it can not extend to single view reconstruction. Specific rules are developed for single view geometry, such as computing vanishing point [20], following rules of BRDF [43, 26], or extract the scene layout with major plane and box representations [47, 50] *etc*.. These methods can only obtain sparse geometry representations, and some of them require certain assumptions (*e.g.* Lambertian, Manhattan world).

Supervised single view geometry via CNN. Deep neural networks (DCN) developed in recent years, *e.g.*VGG [49] and ResNet [34], provide strong feature representation. Dense geometry, i.e., pixel-wise depth and normal maps, can be readily estimated from a single image [54, 12, 34, 36, 16]. The learned CNN model shows significant improvement compared to other methods based on hand-crafted features [23, 32, 31]. Others tried to improve the estimation further by appending a conditional random field (CRF) [52, 38, 35]. Recently, Wang *et al.*[53] proposed a depth-normal regularization over large planar surfaces, which is formulated based on a dense CRF [28], yielding better results on both depth and normal predictions. However, all these methods require densely labeled ground truths, which are expensive to obtain in natural environments.

Unsupervised single view geometry. Motivated by traditional methods, videos, which are easier to obtain and hold richer 3D information. Motivated by traditional methods like SFM and DTAM, lots of CNN based methods are proposed to do single view geometry estimation with supervision from vieos, and yield impressive progress. Deep3D [57] learns to generate the right view from the given left view by supervision of stereo image pairs. In order to do back-propagation on depth values, the depth space is quantized and it is trained to select the right depth value. Concurrently, Garg *et al.*[17] applied the similar supervision from stereo pairs, while the depth is kept continuous. They apply Taylor expansion to approximate the gradient for depth. Godard *et al.*[19] extend Garg's work by including depth smoothness loss and left-right depth consistency. Zhou *et al.*[64] incoporated camera pose estimation into the training pipeline, which made depth learning possible from monocular videos. And they came up with an explainability mask to relieve the problem of moving object in rigid scenes. At the same time, Vijayanarasimhan *et al.*[51] proposed a network to include the modeling of rigid object motion. Most recently, Yang *et al.*[60] further induce normal representation, and proposed a dense depth-normal consistency within the pipeline, which not only better regularizes the predicted depths, but also learns to produce a normal estimation. However, as discussed in Sec. 1, the regularization is only applied locally and can be blocked by image gradient, yielding false geometrical discontinuities inside a smooth surface.

Non-local smoothness. Long range and non-local spatial regularization has been vastly explored in classical graphical models like CRF [33], where nodes beyond the neighboring are connected, and the smoothness in-between are learned with high-order CRF [61] or densely-connected CRF [29]. They show superior performance in detail recovery than those with local connections in multiple tasks, *e.g.*segmentation [28], image disparity [46] and image matting [9] *etc*. In addition, efficient solvers are also developed such as fast bilateral filter [4] or permutohedral lattice [1].

Although these methods run effectively and could combine with CNN as a post processing component [3, 63, 53, 55], they are not very efficient in learning and inference when combined with CNN, due to the iterative loop. To some extent, the non-local information from CRF overlaps with those multi-scale strategies [62, 7] proposed recently,

which yield comparable performance while are more effective. Thus, we adopt the latter strategy to learn the non-local smoothness inside the unsupervised pipeline, which is represented by geometrical edge in our case.

Edge detection. Learning edges from an image beyond low level methods such as Sobel or Canny [6] has long been explored via supervised learning [59, 27, 2, 11] along with the growth of semantic edge datasets [39, 21]. Recently, methods [5, 58, 25] have achieved outstanding performance by adopting supervisedly trained deep features.

As discussed, high-level edges can also be learned through non-local smoothness by implicit supervision. One recent work close to ours is [8]. They append a spatial domain transfer (DT) component after a CNN, which acts similar to a CRF for smoothness, and improves the results of semantic segmentation. However, their work is fully supervised with ground truth, and similar to CRF, the DT propagates to neighboring pixels every iteration which is not efficient. When no supervision is provided, Li *et al.*[37] proposed to use optical flow [44] to explicitly capture motion edge and use it as supervision for edge models.

Our method discovers geometrical edges in an unsupervised manner. In addition, we show that it is possible for the network to directly extract edge and smoothen the 3D geometry by enforcing a unified regularization, without appending extra components like [8]. We also show better performance than [37] in street-view cases.

3. Preliminaries

In order to make the paper self-contained, we first introduce the preliminaries for unsupervised depth and normal estimation proposed in [64, 60]. The core underlying idea is inverse warping from target view to source views with awareness of 3D geometry, and a depth-normal consistency, which we will elaborate in the following paragraphs.

View synthesis as depth supervision. From the multiple view geometry, we know that for a target view image I_t and a source view image I_s, given an estimated depth map D_t for I_t and an estimated transformation $T_{t\to s} \in \mathcal{SE}(3)$ from I_t to I_s, for any pixel p_t in I_t, the corresponding pixel p_s in I_s can be found through perspective projection, *i.e.*$p_s \sim \pi(p_t)$. Then, given such a matching relationship, a synthesized target view $\hat{I}_s$ can be generated from I_s through bilinear interpolation. Finally, by comparing the photometric error between the original target view I_t and the synthesized one $\hat{I}_s$. We can supervise the prediction of D_t and $T_{t\to s}$. Formally, given multiple source views $\mathcal{S} = \{I_s\}$ from a video sequences close to I_t, the photometric loss w.r.t. I_t can be formulated as,

$$\mathcal{L}_{vs}(D,\mathcal{T}) = \sum_s \sum_{p_t \in I_t} |I_t(p_t) - \hat{I}_s(p_t)|, \text{ s.t.} \forall p_t, D(p_t) > 0 \tag{1}$$

where $\mathcal{T}$ is the set of transformations between I_t to each of the source views in $\mathcal{S}$.

Regularization of depth. Nevertheless, supervision based solely on view synthesis is ambiguous, due to one pixel can match to many candidates. Thus, extra regularization is required to learn reasonable depth prediction. One common strategy proposed by previous works [19, 64] is to encourage the estimated depth to be locally similar when no significant image gradient exists. For instance, in [19], the regularization of depth D_t is formulated as:

$$\mathcal{L}_s(D_t, 2) = \sum_{p_t} \sum_{d \in x,y} \|\nabla_d^2 D_t(p_t)\|_1 e^{-\alpha|\nabla_d I(p_t)|} \tag{2}$$

where $\mathcal{L}_s(D, 2)$ is a spatial smoothness term that penalizes L1 norm of second-order gradients of depth along both x and y directions in 2D space. Here, the number 2 represents the 2nd order.

Regularization with depth-normal consistency. Yang *et al.* [60] claim that the smoothness in Eq. (2) is still a too weak constrain to generate a good scene structure, especially when visualized under normal representation, as shown in Fig. 1(b), the predicted normals from [64] varies on the surface of the ground. In their work, they further introduce a normal map N_t for I_t, and a depth-normal consistency energy between D_t and I_t is proposed,

$$\mathcal{C}_{p_t}(D_t, N_t) = \sum\nolimits_{p_j \in \mathcal{N}(p_t)} \omega_{jt} \|(\phi(p_j) - \phi(p_t))^T N(p_t)\|^2$$

where $\mathcal{N}(p_t)$ is a set of 8-neighbors of p_t. $\phi(p)$ is the back projected 3D point from 2D coordinate p. $\phi(p_j) - \phi(p_t)$ is a difference vector in 3D, and ω_{jt} weights the equation. Based on such an energy, they developed a differentiable depth-to-normal layer to estimate N_t given D_t, and a normal-to-depth layer to re-estimate D_t given N_t. By applying losses in Eq. (1) and Eq. (2), plus a first-order normal smoothness loss $\mathcal{L}_s(N_t, 1)$, N_t can be supervised and D_t can be better regularized with at least 8-neighbors. As shown in Fig. 1(d), their strategy yields better predicted depths and normals especially along surface regions. The depth and normal consistency same as in [60] is incorporated into LEGO.

4. Learning edge with geometry from videos

In this section, we introduce the 3D-ASAP prior w.r.t. geometrical edges, and how the edges can be learned jointly with 3D geometry.

4.1. 3D-ASAP prior

Firstly, the core assumption for 3D-ASAP is that for any surface in 3D $S \subset \mathbb{R}^3$, if there is no other cues provided visually, such as edges, S should be a single 3D planar surface. This prior is restrictive for large non-planar surface, but it fits well for street scene which we are mainly dealing with, where the dominant surfaces such as roads, building walls, are still planar. Formally, it should satisfy the following two conditions,

$$\beta \mathbf{x}_i + (1-\beta)\mathbf{x}_j \in S \;\; \forall \mathbf{x}_i, \mathbf{x}_j \in S, \beta \in [0,1] \tag{3}$$

which means any points on the line in-between two points $\mathbf{x}_i$ and $\mathbf{x}_j$ should be also inside the surface. Thus, given a target image $I_t \in \mathbb{R}^2$, which is a rasterized perspective projection from a set of continuous surfaces $\{S\}$, the estimated depth map D_t and normal map N_t should also approximately satisfy such a prior for each S. Specifically, for N_t, any two pixel in the image p_i and p_j, we favor the normal of the two points to be the same when p_i and p_j belong to the same S, which could be formulated as minimizing,

$$\mathcal{L}_N = \sum_{p_i \in I_t} \sum_{p_j \in I_t} \|N_t(p_i) - N_t(p_j)\|_1 \kappa(p_i, p_j) \tag{4}$$

where $\kappa(p_i, p_j)$ is a similarity affinity, which is 1 if p_i, p_j in the same S, and 0 otherwise. For D_t, we consider a triplet relationship, as indicated in Eq. (3). Given two different pixels p_i and p_j, we let any pixel p_k on the line in-between, lies in the same 3D line with p_i, p_j. Formally,

$$\mathcal{L}_D = \sum_{p_i} \sum_{p_j} \sum_{p_k \in l(p_i, p_j)} \|\mathbf{g}(p_i, p_j, p_k)\|_1 \kappa(p_i, p_k) \kappa(p_j, p_k) \tag{5}$$

$$\mathbf{g}(p_i, p_j, p_k) = \frac{D_t(p_i) - D_t(p_k)}{\phi(p_i) - \phi(p_k)} - \frac{D_t(p_j) - D_t(p_k)}{\phi(p_j) - \phi(p_k)},$$

where $\phi(p_i) = D_t(p_i)\mathbf{K}^{-1}h(p_i)$, the back projection function from 2D to 3D space, and $\mathbf{K}$ is the camera intrinsic and $h(p_i)$ is the homogeneous coordinates of p_i. $l(p_i, p_j)$ indicates a set of pixels on the line linking p_i and p_j.

Approximate with a multi-scale strategy. If $\kappa()$ is given, such as using image gradient, we can use these two energy functions to serve as non-local smoothness losses for the estimation of depths and normals. Nevertheless, it is impractical due to the large number of pixels in an image. One approximating solution is to drop the dense connection between one pixel with every other pixel to the connection of a set of pixels nearby. In our case, for each pixel p_i, to be compatible with network training, we choose to smoothen normals and depths with its $\mathcal{N} = 1, 2, 4, 8$ neighborhood along 3D x and y direction, yielding 16 neighbor pixels, which we found to be sufficiently good to avoid local context. Formally, let $p_i(x, y)$ be the pixel has an offset of (x, y) w.r.t. p_i, the energy for D_t and N_t are changed to be,

$$\mathcal{L}_N = \sum_{p_i} \sum_{x,y} \|N_t(p_i) - N_t(p_i(x, y))\|_1 \kappa(p_i, p_i(x, y)), \tag{6}$$

$$\mathcal{L}_{D_x} = \sum_{p_i} \sum_{x} \|\mathbf{g}(p_i, x)\|_1 \kappa(p_i, p_i(x)) \kappa(p_i, p_i(-x))$$

$$\mathbf{g}(p_i, x) = \frac{D_t(p_i(x)) - D_t(p_i)}{\phi(p_i(x)) - \phi(p_i)} - \frac{D_t(p_i) - D_t(p_i(-x))}{\phi(p_i) - \phi(p_i(-x))},$$

where $\mathcal{L}_{D_x}$ means the smoothness along x direction, and $p_i(x)$ is short for $p_i(x, 0)$, similar smoothness is also performed along y direction.

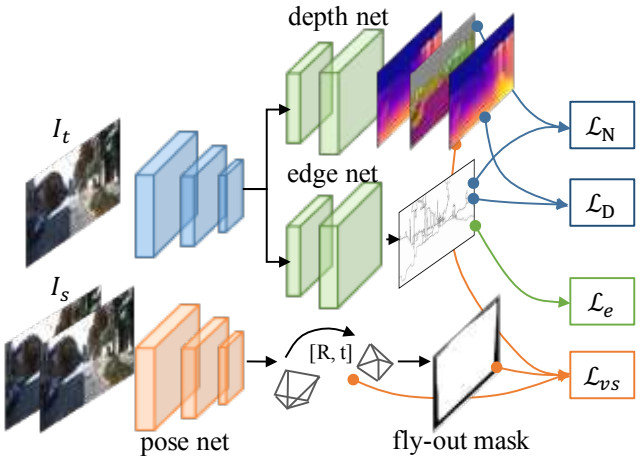


Figure 2: Our loss module consists of four parts: visual synthesis loss $\mathcal{L}_{vs}$, 3D-ASAP losses on depth and normal maps respectively ($\mathcal{L}_D$, $\mathcal{L}_N$), and edge loss $\mathcal{L}_e$. The same depth-normal consistency as in [60] has been used.

4.2. Parameterize and learn the geometrical edge

Given the energy loss proposed in Eq. (6), instead of using image gradient [60, 19], we jointly learn $\kappa()$ by estimating an edge map E_t for the target image. We have,

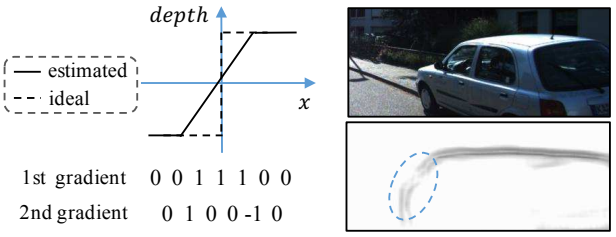
$$\kappa(p_i, p_j) = \exp\{-\max_{p_k \in l(p_i, p_j)} E_t(p_k)\}, \tag{7}$$

where $l(p_i, p_j)$ is the line between p_i and p_j including the end points. This indicates the intervening contour cue [48] for measuring the affinity between two pixels.

Practically, we parameterize the prediction of E_t using a decoder network, which decodes from a shared image encoder of depth network. Putting Eq. (7) back into Eq. (6), plus the photometric losses (Sec. 3), yields the loss function for both normal map N_t, depth map D_t and edge map E_t for regularization. As shown in Fig. 2, we show how different components contribute for different losses.

Overcoming the trivial solution. As we do not have direct supervision for E_t, training with Eq. (7) would result in a trivial solution by predicting every pixel as edge, which perfectly minimize the smoothness both on depths and normals. To resolve this, we add a regularization term with a simple L2 loss to favor no edge predictions, *i.e.*$\mathcal{L}_e(E) = \sum_{p_i} ||E_t(p_i)||^2$. Another potential way is to use cross-entropy as regularization. In our experiment, it does not work well and is very sensitive to the weighting balance. We think it is due to the edge map containing only sparse edges. For supervised learning, HED [58] adopts ground truth to balance positive and negative pixels for the cross-entropy, which is not available in our case.

Handling double edges during training. After training using the previous losses, we observe double-edge artifacts, as shown in Fig. 3(b). Unlike the ideal depth prediction, where depth across a boundary of discontinuity is a step jump (dashed line in Fig. 3(a)), the estimated depth



(a) Gradient of changing depth (b) Double-edge artifact

Figure 3: Double-edge issue in edge estimation.

Figure 4: Some part of the scene flies out as camera moves forward from I_t to I_{s2}. A fly-out mask is calculated from camera motion to filter out such regions.

changes smoothly across the object boundary (solid line). Thus, when computing the depth 3D-ASAP regularization term $\mathcal{L}_D$ with one neighborhood in Eq. (6) which is similar to a second-order gradient operation, a non-zero value is generated at both beginning and the end of depth changing. To minimize $\mathcal{L}_D$, the edge map E_t needs to predict a double edge to suppress both of the non-zero values.

We fix this issue by clipping the negative values in the computed gradient map from $\mathbf{g}(p_i, *)$ in Eq. (6), as for each boundary along x or y direction, second-order gradient will always have one positive and one negative value. Formally, we replace $\mathbf{g}(p_i, *)$ to $\mathbf{g}' = \max(\mathbf{g}(p_i, *), 0)$.

The architecture of the edge decoder network is set to be the same as the decoder of depth network, while we adopt nearest strategy for edge upsampling from low-scale to high-scale inside the network.

4.3. Overcoming invalid and local gradient

Fly-out mask for invalid gradient. Previous works [64, 60] have fixed the length of frame sequence to be 3, with the center frame as the target view (I_t) and the neighboring two frames as source view images (I_{s1}, I_{s2}). When doing view synthesis, possibly part of the corresponding pixels for target view is outside of the source view, yielding invalid gradient for those pixels. As shown in Fig. 4, we identify those pixel and mask out the invalid gradients.

Overcoming local gradient. Similar with gradient locality mentioned in [64], the spatial transform operation is based on bilinear interpolation which depends on only 4 neighboring pixels. Thus, loss based on multi-resolution is necessary for effective training. Same strategy is applied in our training pipeline, and in summary, our overall training loss could be written as,

$$\mathcal{L}(\{D_l\}, \{N_l\}, \{E_l\}, \mathcal{T}) = \sum_l \{\lambda_{vs}\mathcal{L}_{vs}(D_l, \mathcal{T}) + \lambda_d\mathcal{L}_D(D_l, E_l) + \lambda_n\mathcal{L}_N(N_t, E_t) + \lambda_e\mathcal{L}_e(E_t)\} \quad (8)$$

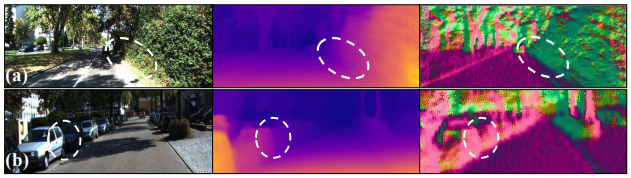


Figure 5: Depth and normal are complementary in geometrical edge discovery. (a) Across the edge between two intersecting planes (street and side wall), depth changes smoothly while normal varies drastically; (b) Across the edge between car sides, there is large depth change while normal is uniform.

where $\lambda_{vs}, \lambda_d, \lambda_n, \lambda_e$ are balancing factors that are tuned with a sampled validation set from training images.

Finally, in our experiments, we show it is important to have both smoothness over D_t and N_t. As illustrated in Fig. 5, depth and normal are complementary for discovering all the geometrical edges. More importantly, the learned edge are consistent with both depth and normal, yielding no perceptual confusion among different information.

5. Evaluation

In this section, we first describe the datasets and evaluation metrics used in our experiments. And then present comprehensive evaluation of LEGO on different tasks.

5.1. Implementation details

We adopt a DispNet [40] like achitecture for depth net and edge net. Regular DispNet is based on an encoder-decoder design with skip connections and multi-scale side outputs. Depth net and edge net share the same encoder while have separate decoder, which decodes depth and edge maps respectively. To avoid artifact grid output from decoder, the kernel size of decoder layers is set to be 4 and the input image is resized to be non-integer times of 64. All *conv* layers are followed by ReLU activation except for the top output layer, where we apply a sigmoid function to constrain the depth and edge prediction within a reasonable range. Batch normalization [22] is performed on all convolutional layers. To increase the receptive field size while maintaining the number of parameters, dilated convolution with a dilation of 2 is implemented.

During training, Adam optimizer [24] is applied with $\beta_1 = 0.9$, $\beta_2 = 0.999$, learning rate of 2×10^{-3} and batch size of 4. The balance between different losses is adjusted so that each loss component has loss value of similar scale. In practice, the loss weights are set as: $\lambda_{vs} = 1.0$, $\lambda_d = 2.0$, $\lambda_n = 0.01$, $\lambda_e = 0.15$ for KITTI dataset and $\lambda_{vs} = 1.0$, $\lambda_d = 4.0$, $\lambda_n = 0.05$, $\lambda_e = 0.12$ for Cityscapes dataset. All the hyper-parameters are tuned on the held-out validation set. The input monocular frame sequences are resized to 830×254 and the length of input sequence is set to be 3. The middle frame serves as the target frame and the neighboring two frames are used as source frames. The whole framework is implemented with Tensorflow platform. On a

single Titan X (Pascal) GPU, the framework occupies 4GB of memory with batch size of 4.

5.2. Datasets and metrics

We conducted experiments on different tasks: depth estimation, normal estimation and edge detection. The performances are evaluated on three popular datasets: KITTI 2015, Cityscapes and Make3D, using corresponding metrics.

KITTI 2015. KITTI 2015 dataset provides videos in 200 street scenes captured by stereo RGB cameras, and sparse depth ground truth captured by Velodyne laser scanner. During training, 156 videos excluding test scenes are used, with the left and right videos treated independently. The training sequences are constructed with three consecutive frames, resulting in 40250 training samples. There are two test splits of KITTI 2015: the official test set consisting of 200 images (KITTI split) and the test split proposed in [13] consisting of 697 images (Eigen split). The official KITTI test split provides ground truth of better quality compared to Eigen split, where less than 5% pixels in the input image has ground truth depth values. LEGO is evaluated on both splits to better compare with other methods.

Cityscapes. Cityscapes is a city-scene dataset with ground truth for semantic segmentation. It contains 27 stereo videos, and provides pixel-wise semantic segmentation ground truth for 500 frames in validation split. Training sequences are constructed from 18 left-view videos of the training set, resulting in 69728 training samples. The semantic segmentation ground truth in 500 validation frames is used for the evaluation of edge detection. Details of using segmentation ground truth are described in Sec. 5.4.

Make3D. Make3D dataset contains no videos but 534 monocular image and depth ground truth pairs. Unstructured outdoor scenes, including bush, trees, residential buildings, *etc*. are captured in this dataset. Same as in [64, 19], the evaluation is performed on the test set of 134 images.

Metrics. The existing metrics of depth, normal and edge detection have been used for evaluation, as in [13], [15] and [2]. For depth and edge evaluation, we have used the code by [19] and [11] respectively. For normal evaluation, we implement the evaluation metrics in [15] and verify it by validating the results in [12]. The explanation of each metric used in our evaluation is specified in Tab. 1.

5.3. Depth and normal experiments

Experiment setup. The depth and surface normal experiments are conducted on KITTI 2015, Cityscapes and Make3D datasets. For KITTI 2015, the given depth ground truth is used for evaluting depth estimation, and the normal ground truth is computed from interpolated depth ground truth using depth-to-normal layer. Videos in Cityscapes dataset are captured by the cameras mounted on moving cars. Part of the car is captured in the videos hence the bottom part of the frames is cropped. As no ground truth depth is given in this dataset, we are using Cityscapes only for training. Images in Make3D dataset have different aspect ratio from KITTI or Cityscapes frames, the central part is cropped out for evaluation. For both depth and normal evaluation, only pixels with ground truth depth values are evaluated. One LEGO variant is generated by removing fly-out mask from the pipeline, LEGO (no fly-out), to explore the effectivenss of fly-out mask.

Table 1: From top row to bottom row: depth, normal and edge evaluation metrics.

Abs Rel: $\frac{1}{\lvert D\rvert}\sum_{d'\in D}\lvert d^*\text{–}d'\rvert/d^*$	Sq Rel: $\frac{1}{\lvert D\rvert}\sum_{d'\in D}\lvert\lvert d^*\text{–}d'\rvert\rvert^2/d^*$
RMSE: $\sqrt{\frac{1}{\lvert D\rvert}\sum_{d'\in D}\lvert\lvert d^*\text{–}d'\rvert\rvert^2}$	RMSE log: $\sqrt{\frac{1}{\lvert D\rvert}\sum_{d'\in D}\lvert\lvert \log d^*\text{–}\log d'\rvert\rvert^2}$
mean: $\frac{1}{\lvert N\rvert}\sum_{n'\in N}(n^*\cdot n')$	median: $median([(n^*\cdot n')]_{n'\in N})$
X°: % of $n'\in N$, $(n^*\cdot n')<X^\circ$	
ODS: optimal F1 for the dataset	OIS: optimal F1 for each image
AP: average precision	PR curve: precision-recall curve

The following evaluations are performed to present the depth and normal results: (1) depth estimation performance compared with SOTA methods; (2) normal estimation performance compared with SOTA methods; (3) generalization capability between different datasets.

Comparison with state-of-the-art. The model is trained on KITTI 2015 raw videos excluding frames of scenes in both test splits. Following the tradition of other methods [13, 64, 19], the maximum of depth estimation on KITTI split is capped at 80 meters and the same crop as in [13] is applied during evaluation on Eigen split.

Tab. 2 shows the comparison of LEGO variants and recent SOTA methods. LEGO outperforms all unsupervised methods [64, 30, 60] consistently on both test splits and performs comparably to the semi-supervised method [19]. It is also worth noting that on the metric of "Sq Rel", LEGO outperforms all other methods on KITTI split. This metric measures the ratio of square of prediction error over the ground truth value, and thus is sensitive to points where the depth values are away from the ground truth. The good performance under this metric indicates that LEGO produces consistent 3D scene layout and generates fewer outlier depth values.

The normal ground truth is generated by applying depth-to-normal layer on interpolated depth ground truth. As the depth ground truth point in Eigen split is very sparse (<5%), the interpolation incorporates extra noise and not suitable for normal evaluaton. The normal evaluation is performed only on KITTI split. The comparison of normal evaluations on KITTI split is presented in Tab. 3. The methods we have compared with include: (1) ground truth normal mean: mean value of ground truth normal over the image

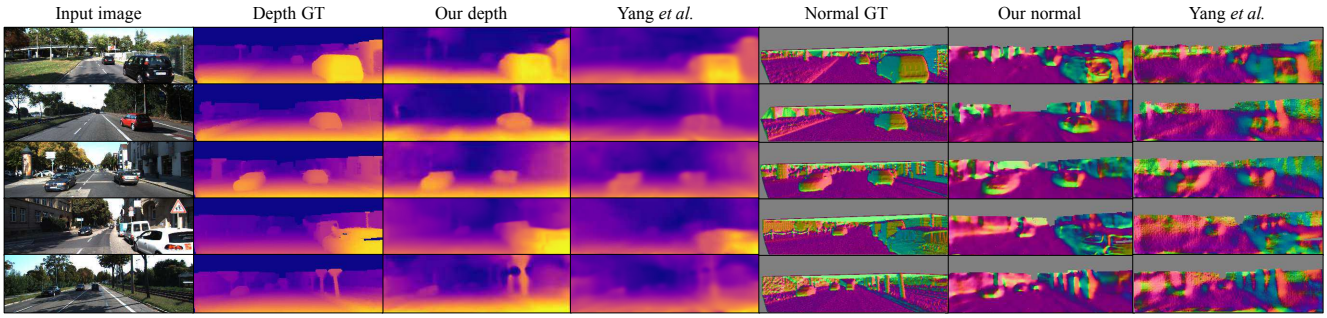


Figure 6: Visual comparison between Yang *et al.*[60] and LEGO results on KITTI test split. The depth and normal ground truths are interpolated and all images are reshaped for better visualization. For depths, LEGO results have noticeably shaper edges and the depth edges are well aligned with object boundaries. For surface normals, LEGO results have fewer artifacts and extract clear scene layout.

Table 2: Monocular depth evaluation results on KITTI split (upper part) and Eigen split(lower part). All methods use KITTI dataset for traning if not specially noted. Results of [64] on KITTI test split are generated by training their released model on KITTI dataset. *CS* denotes the method trained on Cityscapes and then finetuned KITTI data. *PP* and *R* denote post processing and ResNet respectively.

Method	Test data	Supervision	Lower the better			
			Abs Rel	Sq Rel	RMSE	RMSE log
Train set mean	KITTI split	Depth	0.398	5.519	8.632	0.405
Godard *et al.*[19]		Pose	0.124	1.388	6.125	0.217
Vij.. *et al.*[51]			-	-	-	0.340
Zhou *et al.*[64]			0.216	2.255	7.422	0.299
Yang *et al.*[60]			0.165	1.360	6.641	0.267
LEGO (no fly-out)			0.157	1.303	6.223	0.241
LEGO			0.154	1.272	6.012	0.230
Godard *et al.*[19]+CS	KITTI split	Pose	0.104	1.070	5.417	0.188
Godard *et al.*[19]+CS+PP+R		Pose	0.097	0.896	5.093	0.176
LEGO+CS			0.142	1.237	5.846	0.225
Train set mean	Eigen split	Depth	0.403	5.530	8.709	0.403
Kuz.. *et al.*[30] supervised		Depth	0.122	0.763	4.815	0.194
Kuz.. *et al.*[30] unsupervised		Pose	0.308	9.367	8.700	0.367
Kuz.. *et al.*[30] combined		Pose+Depth	0.113	0.741	4.621	0.189
Godard *et al.*[19]		Pose	0.148	1.344	5.927	0.247
Zhou *et al.*[64]			0.208	1.768	6.856	0.283
Yang *et al.*[60]			0.182	1.481	6.501	0.267
LEGO (no fly-out)			0.170	1.382	6.321	0.255
LEGO			0.162	1.352	6.276	0.252
Godard *et al.*[19]+CS	Eigen split	Pose	0.124	1.076	5.311	0.219
Godard *et al.*[19]+CS+PP+R		Pose	0.114	0.898	4.935	0.206
Zhou *et al.*[64]+CS			0.198	1.836	6.565	0.275
LEGO+CS			0.159	1.345	6.254	0.247

size; (2) pre-defined scene: based on the observation that KITTI is a street scene dataset, the image is divided into 4 parts by connecting the center and 4 corners, approximating the scene with road in the bottom part, buildings on the two sides and sky at the top; (3) normal results generated by applying depth-to-normal layer on depth maps from some baseline methods [19, 64, 60].

LEGO outperforms all baseline methods by a large margin. Note that LEGO has inferior results compared to [19] on depth results while still outperforms on normals. One possible reason is that the depth is only evaluated on pixels with ground truth values, while the normal direction of each pixel is computed based on neighboring points, which

Table 3: Normal evaluation results on KITTI test split.

Method	Mean	Median	11.25°	22.5°	30°
Ground truth normal mean	72.39	64.72	0.031	0.134	0.243
Pre-defined scene	63.52	58.93	0.067	0.196	0.302
Zhou *et al.*[64]	50.47	39.16	0.125	0.303	0.425
Godard *et al.*[19]	39.28	29.37	0.158	0.412	0.496
Yang *et al.*[60]	47.52	33.98	0.149	0.369	0.473
LEGO (no-flyout)	39.29	28.14	0.226	0.421	0.508
LEGO	**36.13**	**25.94**	**0.241**	**0.473**	**0.542**

Table 4: Depth evaluation results with model trained on a different dataset. Note that [19] leverages pose ground truth during training.

Methods	Train/Test dataset	Lower the better			
		Abs Rel	Sq Rel	RMSE	RMSE log
Godard *et al.*[19]	CS/K	0.699	10.06	14.44	0.542
Zhou *et al.*[64]		0.275	2.883	7.684	0.382
LEGO		**0.201**	**1.650**	**6.788**	**0.278**
Godard *et al.*[19]	CS/Make3D	0.535	11.99	11.51	-
Zhou *et al.*[64]		0.383	**5.321**	10.47	0.478
LEGO		**0.352**	7.731	**7.194**	0.346
Kuznietsov *et al.*[30]	K/Make3D	0.421	-	8.237	0.190

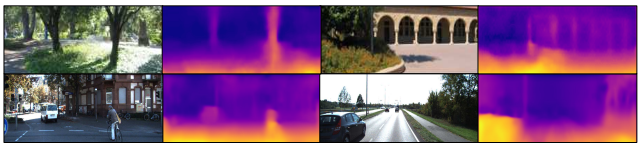


Figure 7: Depth test results by the model trained on a different dataset. Top row: trained on Cityscapes, tested on Make3D. Second row: trained on Cityscapes, tested on KITTI 2015.

indicates that LEGO may produce depth and normal that are more consistent with the scene layout. Compared to LEGO (no fly-out), LEGO experiences larger performance improvement in normal results compared to depth evaluation.

Qualitative results are shown in Fig. 6. Compared with [60], LEGO generates smoother depth and normal outputs within the same surface while still preserving clear geometrical edges.

Generalization capability. Generalizing to data unseen during training is an important property for unsupervised geometry estimation as there may not be enough data for certain scenes. The generalization capability of LEGO is tested by training on one dataset and testing on another dataset. Specifically, to compare with previous methods,

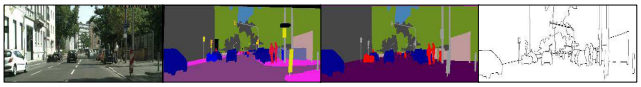


Figure 8: Display of the process of geometric edge ground truth generation. From left to right: RGB image, original segmentation ground truth, combined segmentation result, edge ground truth.

two experiments have been conducted: (1) pipeline trained on Cityscapes dataset (CS) and tested on KITTI dataset (K); (2) pipeline trained on Cityscapes and evaluated on Make3D dataset (Make3D). The comparison results are shown in Tab. 4.

Under both settings, LEGO achieves state-of-the-art performance. When transferring from Cityscapes to KITTI, it outperforms other methods by a large margin. One potential explanation is that compared to supervised or semi-supervised methods, LEGO has less risk of overfitting. Compared to other unsupervised methods, our novel 3D-ASAP regularization encourages the network to learn the structural layout information jointly and thus the trained model is more robust to scene changes. Some visualization examples of the generalization results are shown in Fig. 7.

5.4. Edge experiments

Experiment setup. The geometrical edge detection performance is evaluated on Cityscapes dataset. Cityscapes contains a validation set of 500 images with pixel-wise semantic segmentation annotation. The edge ground truth is generated from the segmentation ground truth. Some geometrically connected categories such as "ground" and "road", "fence" and "guard rail", "pole" and "traffic sign" are combined and the geometrical edges are extracted from the boundaries of these combined categories. Fig. 8 shows how the ground truth edge ground truth is generated. More details are provided in the supplementary material.

As there has not been previous work that reported edge detection performance on Cityscapes, we compare with unsupervised edge learning [37] and some other baselines we build. The results of [37] are generated by traning their public model on Cityscapes videos. Different from [37] which randomly samples training data, we do not apply any sampling to make the number of training samples comparable to our method. Other baseline methods include: (1) modification of Zhou *et al.*[64] method by adding an edge detection network to the model (Zhou *et al.*[64]+edge net); (2) apply the pre-trained Structured Edge detector (SE) [11] on depth and normal output from [60] (SE-D/SE-N); (3) apply the pre-trained holistically-nested edge detector (HED) [58] edge detector on depth and normal results from [60] (HED-D/HED-N).

Ablation study. Two LEGO variants are generated by applying geometrical edge in only depth or normal smoothness term (LEGO (d-edge) and LEGO (n-edge)). We explore the effect of depth and normal complementing each other in geometrical edge detection.

Methods	ODS	OIS	AP
Li *et al.*[36]	0.623	0.663	0.557
HED-D	0.457	0.503	0.467
HED-N	0.614	0.642	0.571
SE-D	0.502	0.513	0.536
SE-N	0.621	0.647	0.579
Zhou *et al.*[61]+edge net	0.621	0.637	0.637
LEGO (d-edge)	0.602	0.615	0.594
LEGO (n-edge)	0.679	0.718	0.692
LEGO	**0.710**	**0.731**	**0.729**

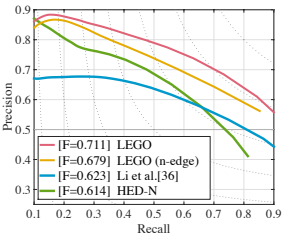


Figure 9: Edge evaluation results on Cityscapes.

Figure 10: Edge detection results on Cityscapes dataset. From top to bottom: input image, unsupervised edge by Li *et al.*[37], HED-N, our results, edge ground truth. All detection visualization results are before the process of non-maximum suppression).

Comparison with other methods. LEGO is compared with re-trained [37] and general edge detection (SE [11], HED [58]) results applied on depth/normal output. The quantitative and qualitative results are presented in Fig. 9 and Fig. 10. LEGO outperforms other methods by a large margin on all metrics. In visualization results as in Fig. 10, predictions by LEGO preserve the object boundaries and ignore trivial edges within a surface like lane marking. Compared to the edge generated from normal (HED-N), LEGO estimations are well aligned with ground truth edges.

6. Conclusion

In this paper, we proposed LEGO, an unsupervised framework for joint depth, normal and edge learning. A novel 3D-ASAP prior is proposed to better regularize the learning of scene layout. This regularization jointly considers the three important descriptors of 3D scene and improves the results on all tasks: depth, normal and edge estimation. We conducted comprehensive experiments to present the performance of LEGO. On KITTI dataset, LEGO achieves SOTA performance on both depth and normal evaluation. For edge evaluation, LEGO outperformes the other methods by a large margin on Cityscapes dataset.

References

[1] A. Adams, J. Baek, and M. A. Davis. Fast high-dimensional filtering using the permutohedral lattice. In *Computer Graphics Forum*, volume 29, pages 753–762. Wiley Online Library, 2010.

[2] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik. Contour detection and hierarchical image segmentation. *TPAMI*, 33(5):898–916, 2011.

[3] A. Arnab, S. Jayasumana, S. Zheng, and P. H. Torr. Higher order conditional random fields in deep neural networks. In *ECCV*, 2016.

[4] J. T. Barron and B. Poole. The fast bilateral solver. In *ECCV*, 2016.

[5] G. Bertasius, J. Shi, and L. Torresani. High-for-low and low-for-high: Efficient boundary detection from deep object features and its applications to high-level vision. In *ICCV*, pages 504–512, 2015.

[6] J. Canny. A computational approach to edge detection. *TPAMI*, pages 679–698, 1986.

[7] L. Chen, G. Papandreou, F. Schroff, and H. Adam. Rethinking atrous convolution for semantic image segmentation. *CoRR*, abs/1706.05587, 2017.

[8] L.-C. Chen, J. T. Barron, G. Papandreou, K. Murphy, and A. L. Yuille. Semantic image segmentation with task-specific edge detection using cnns and a discriminatively trained domain transform. In *CVPR*, pages 4545–4554, 2016.

[9] X. Chen, D. Zou, S. Zhiying Zhou, Q. Zhao, and P. Tan. Image matting with local and nonlocal smooth priors. In *CVPR*, pages 1902–1907, 2013.

[10] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele. The cityscapes dataset for semantic urban scene understanding. In *CVPR*, 2016.

[11] P. Dollár and C. L. Zitnick. Structured forests for fast edge detection. In *ICCV*, 2013.

[12] D. Eigen and R. Fergus. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In *ICCV*, 2015.

[13] D. Eigen, C. Puhrsch, and R. Fergus. Depth map prediction from a single image using a multi-scale deep network. In *NIPS*, 2014.

[14] J. Engel, T. Schöps, and D. Cremers. Lsd-slam: Large-scale direct monocular slam. In *ECCV*, 2014.

[15] D. F. Fouhey, A. Gupta, and M. Hebert. Data-driven 3d primitives for single image understanding. In *ICCV*, 2013.

[16] C. Gan, B. Gong, K. Liu, H. Su, and L. Guibas. Geometry-guided cnn for self-supervised video representation learning. In *CVPR*, 2018.

[17] R. Garg, V. K. B. G, and I. D. Reid. Unsupervised CNN for single view depth estimation: Geometry to the rescue. *ECCV*, 2016.

[18] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *CVPR*, 2012.

[19] C. Godard, O. Mac Aodha, and G. J. Brostow. Unsupervised monocular depth estimation with left-right consistency. In *CVPR*, 2017.

[20] D. Hoiem, A. A. Efros, and M. Hebert. Recovering surface layout from an image. In *ICCV*, 2007.

[21] X. Hou, A. Yuille, and C. Koch. Boundary detection benchmarking: Beyond f-measures. In *CVPR*, pages 2123–2130, 2013.

[22] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, 2015.

[23] K. Karsch, C. Liu, and S. B. Kang. Depth transfer: Depth extraction from video using non-parametric sampling. *IEEE transactions on pattern analysis and machine intelligence*, 36(11):2144–2158, 2014.

[24] D. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[25] I. Kokkinos. Pushing the boundaries of boundary detection using deep learning. *ICLR*, 2016.

[26] N. Kong and M. J. Black. Intrinsic depth: Improving depth transfer with intrinsic images. In *ICCV*, 2015.

[27] S. Konishi, A. L. Yuille, J. M. Coughlan, and S. C. Zhu. Statistical edge detection: Learning and evaluating edge cues. *TPAMI*, 25(1):57–74, 2003.

[28] P. Krähenbühl and V. Koltun. Efficient inference in fully connected crfs with gaussian edge potentials. *NIPS*, 2012.

[29] P. Krähenbühl and V. Koltun. Parameter learning and convergent inference for dense random fields. In *ICML*, 2013.

[30] Y. Kuznietsov, J. Stuckler, and B. Leibe. Semi-supervised deep learning for monocular depth map prediction. In *CVPR*, 2017.

[31] B. L. Ladicky, Zeisl, M. Pollefeys, et al. Discriminatively trained dense surface normal estimation. In *ECCV*, 2014.

[32] L. Ladicky, J. Shi, and M. Pollefeys. Pulling things out of perspective. In *CVPR*, 2014.

[33] J. Lafferty, A. McCallum, and F. C. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML*, 2001.

[34] I. Laina, C. Rupprecht, V. Belagiannis, F. Tombari, and N. Navab. Deeper depth prediction with fully convolutional residual networks. In *3D Vision (3DV), 2016 Fourth International Conference on*, pages 239–248. IEEE, 2016.

[35] B. Li, C. Shen, Y. Dai, A. van den Hengel, and M. He. Depth and surface normal estimation from monocular images using regression on deep features and hierarchical crfs. In *CVPR*, 2015.

[36] J. Li, R. Klein, and A. Yao. A two-streamed network for estimating fine-scaled depth maps from single rgb images. In *ICCV*, 2017.

[37] Y. Li, M. Paluri, J. M. Rehg, and P. Dollár. Unsupervised learning of edges. In *CVPR*, 2016.

[38] F. Liu, C. Shen, and G. Lin. Deep convolutional neural fields for depth estimation from a single image. In *CVPR*, June 2015.

[39] D. Martin, C. Fowlkes, D. Tal, and J. Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *ICCV*, volume 2, pages 416–423, July 2001.

[40] N. Mayer, E. Ilg, P. Hausser, P. Fischer, D. Cremers, A. Dosovitskiy, and T. Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *CVPR*, 2016.

[41] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos. Orb-slam: a versatile and accurate monocular slam system. *IEEE Transactions on Robotics*, 31(5):1147–1163, 2015.

[42] R. A. Newcombe, S. Lovegrove, and A. J. Davison. DTAM: dense tracking and mapping in real-time. In *ICCV*, 2011.

[43] E. Prados and O. Faugeras. Shape from shading. *Handbook of mathematical models in computer vision*, pages 375–388, 2006.

[44] J. Revaud, P. Weinzaepfel, Z. Harchaoui, and C. Schmid. Epicflow: Edge-preserving interpolation of correspondences for optical flow. In *CVPR*, pages 1164–1172, 2015.

[45] A. Saxena, S. H. Chung, and A. Y. Ng. Learning depth from single monocular images. In *NIPS*, pages 1161–1168, 2006.

[46] D. Scharstein and C. Pal. Learning conditional random fields for stereo. In *CVPR*, 2007.

[47] A. G. Schwing, S. Fidler, M. Pollefeys, and R. Urtasun. Box in the box: Joint 3d layout and object reasoning from single images. In *ICCV*, 2013.

[48] J. Shi and J. Malik. Normalized cuts and image segmentation. *TPAMI*, 22(8):888–905, 2000.

[49] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv:1409.1556*, 2014.

[50] F. Srajer, A. G. Schwing, M. Pollefeys, and T. Pajdla. Match box: Indoor image matching via box-like scene estimation. In *3DV*, 2014.

[51] S. Vijayanarasimhan, S. Ricco, C. Schmid, R. Sukthankar, and K. Fragkiadaki. Sfm-net: Learning of structure and motion from video. *CoRR*, abs/1704.07804, 2017.

[52] P. Wang, X. Shen, Z. Lin, S. Cohen, B. L. Price, and A. L. Yuille. Towards unified depth and semantic prediction from a single image. In *CVPR*, 2015.

[53] P. Wang, X. Shen, B. Russell, S. Cohen, B. L. Price, and A. L. Yuille. SURGE: surface regularized geometry estimation from a single image. In *NIPS*, 2016.

[54] X. Wang, D. Fouhey, and A. Gupta. Designing deep networks for surface normal estimation. In *CVPR*, 2015.

[55] Y. Wang, Y. Yang, Z. Yang, L. Zhao, and W. Xu. Occlusion aware unsupervised learning of optical flow. *CVPR*, 2018.

[56] C. Wu et al. Visualsfm: A visual structure from motion system (2011). *URL http://www. cs. washington. edu/homes/ccwu/vsfm*, 14, 2011.

[57] J. Xie, R. Girshick, and A. Farhadi. Deep3d: Fully automatic 2d-to-3d video conversion with deep convolutional neural networks. In *ECCV*, 2016.

[58] S. Xie and Z. Tu. Holistically-nested edge detection. In *ICCV*, pages 1395–1403, 2015.

[59] K. Yamaguchi, T. Hazan, D. McAllester, and R. Urtasun. Continuous markov random fields for robust stereo estimation. In *ECCV*, pages 45–58. Springer, 2012.

[60] Z. Yang, P. Wang, W. Xu, L. Zhao, and N. Ram. Unsupervised learning of geometry from videos with edge-aware depth-normal consistency. In *AAAI*, 2018.

[61] N. Ye, W. S. Lee, H. L. Chieu, and D. Wu. Conditional random fields with high-order features for sequence labeling. In *NIPS*, pages 2196–2204, 2009.

[62] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia. Pyramid scene parsing network. *CVPR*, 2016.

[63] S. Zheng, S. Jayasumana, B. Romera-Paredes, V. Vineet, Z. Su, D. Du, C. Huang, and P. Torr. Conditional random fields as recurrent neural networks. In *International Conference on Computer Vision (ICCV)*, 2015.

[64] T. Zhou, M. Brown, N. Snavely, and D. G. Lowe. Unsupervised learning of depth and ego-motion from video. In *CVPR*, 2017.