Unsupervised Learning of Depth and Ego-Motion from Video

Tinghui Zhou∗
UC Berkeley
Matthew Brown
Google
Noah Snavely
Google
David G. Lowe
Google

Abstract

We present an unsupervised learning framework for the task of monocular depth and camera motion estimation from unstructured video sequences. In common with recent work [10, 14, 16], we use an end-to-end learning approach with view synthesis as the supervisory signal. In contrast to the previous work, our method is completely unsupervised, requiring only monocular video sequences for training. Our method uses single-view depth and multi-view pose networks, with a loss based on warping nearby views to the target using the computed depth and pose. The networks are thus coupled by the loss during training, but can be applied independently at test time. Empirical evaluation on the KITTI dataset demonstrates the effectiveness of our approach: 1) monocular depth performs comparably with supervised methods that use either ground-truth pose or depth for training, and 2) pose estimation performs favorably compared to established SLAM systems under comparable input settings.

1. Introduction

Humans are remarkably capable of inferring ego-motion and the 3D structure of a scene even over short timescales. For instance, in navigating along a street, we can easily locate obstacles and react quickly to avoid them. Years of research in geometric computer vision has failed to recreate similar modeling capabilities for real-world scenes (e.g., where non-rigidity, occlusion and lack of texture are present). So why do humans excel at this task? One hypothesis is that we develop a rich, structural understanding of the world through our past visual experience that has largely consisted of moving around and observing vast numbers of scenes and developing consistent modeling of our observations. From millions of such observations, we have learned about the regularities of the world—roads are flat, buildings are straight, cars are supported by roads etc., and we can apply this knowledge when perceiving a new scene, even from a single monocular image.

∗The majority of the work was done while interning at Google.
truth. While imperfect geometry and/or pose estimation can cheat with reasonable synthesized views for certain types of scenes (e.g., textureless), the same model would fail miserably when presented with another set of scenes with more diverse layout and appearance structures. Thus, our goal is to formulate the entire view synthesis pipeline as the inference procedure of a convolutional neural network, so that by training the network on large-scale video data for the ‘meta’-task of view synthesis the network is forced to learn about intermediate tasks of depth and camera pose estimation in order to come up with a consistent explanation of the visual world. Empirical evaluation on the KITTI [15] benchmark demonstrates the effectiveness of our approach on both single-view depth and camera pose estimation. Our code will be made available at https://github.com/tinghuiz/SfMLearner.

2. Related work

Structure from motion The simultaneous estimation of structure and motion is a well studied problem with an established toolchain of techniques [12, 50, 38]. Whilst the traditional toolchain is effective and efficient in many cases, its reliance on accurate image correspondence can cause problems in areas of low texture, complex geometry/photometry, thin structures, and occlusions. To address these issues, several of the pipeline stages have been recently tackled using deep learning, e.g., feature matching [18], pose estimation [26], and stereo [10, 27, 53]. These learning-based techniques are attractive in that they are able to leverage external supervision during training, and potentially overcome the above issues when applied to test data.

Warping-based view synthesis One important application of geometric scene understanding is the task of novel view synthesis, where the goal is to synthesize the appearance of the scene seen from novel camera viewpoints. A classic paradigm for view synthesis is to first either estimate the underlying 3D geometry explicitly or establish pixel correspondence among input views, and then synthesize the novel views by compositing image patches from the input views (e.g., [4, 55, 43, 6, 9]). Recently, end-to-end learning has been applied to reconstruct novel views by transforming the input based on depth or flow, e.g., DeepStereo [10], Deep3D [51] and Appearance Flows [54]. In these methods, the underlying geometry is represented by quantized depth planes (DeepStereo), probabilistic disparity maps (Deep3D) and viewpoint-dependent flow fields (Appearance Flows), respectively. Unlike methods that directly map from input views to the target view (e.g., [45]), warping-based methods are forced to learn intermediate predictions of geometry and/or correspondence. In this work, we aim to distill such geometric reasoning capability from CNNs trained to perform warping-based view synthesis.

Learning single-view 3D from registered 2D views Our work is closely related to a line of recent research on learning single-view 3D inference from registered 2D observations. Garg et al. [14] propose to learn a single-view depth estimation CNN using projection errors to a calibrated stereo twin for supervision. Concurrently, Deep3D [51] predicts a second stereo viewpoint from an input image using stereoscopic film footage as training data. A similar approach was taken by Godard et al. [16], with the addition of a left-right consistency constraint, and a better architecture design that led to impressive performance. Like our approach, these techniques only learn from image observations of the world, unlike methods that require explicit depth for training, e.g., [20, 42, 7, 27, 30].

These techniques bear some resemblance to direct methods for structure and motion estimation [22], where the camera parameters and scene depth are adjusted to minimize a pixel-based error function. However, rather than directly minimizing the error to obtain the estimation, the CNN-based methods only take a gradient step for each batch of input instances, which allows the network to learn an implicit prior from a large corpus of related imagery. Several authors have explored building differentiable rendering operations into their models that are trained in this way, e.g., [19, 29, 34].

While most of the above techniques (including ours) are mainly focused on inferring depth maps as the scene geometry output, recent work (e.g., [13, 41, 46, 52]) has also shown success in learning 3D volumetric representations from 2D observations based on similar principles of projective geometry. Fouhey et al. [11] further show that it is even possible to learn 3D inference without 3D labels (or registered 2D views) by utilizing scene regularity.

Unsupervised/Self-supervised learning from video Another line of related work to ours is visual representation learning from video, where the general goal is to design pretext tasks for learning generic visual features from video data that can later be re-purposed for other vision tasks such as object detection and semantic segmentation. Such pretext tasks include ego-motion estimation [2, 24], tracking [49], temporal coherence [17], temporal order verification [36], and object motion mask prediction [39]. While we focus on inferring the explicit scene geometry and ego-motion in this work, intuitively, the internal representation learned by the deep network (especially the single-view depth CNN) should capture some level of semantics that could generalize to other tasks as well.

 Concurrently to our work, Vijayanarasimhan et al. [48] independently propose a framework for joint training of depth, camera motion and scene motion from videos. While both methods are conceptually similar, ours is focused on the unsupervised aspect, whereas their framework adds the capability to incorporate supervision (e.g., depth, camera motion or scene motion). There are significant differences in how scene dynamics are modeled during training, in which they explicitly solve for object motion whereas our explainability mask discounts regions undergoing motion, occlusion and other factors.

3. Approach

Here we propose a framework for jointly training a single-view depth CNN and a camera pose estimation CNN from unlabeled video sequences. Despite being jointly trained, the depth model and the pose estimation model can be used independently during test-time inference. Training examples to our model consist of short image sequences of scenes captured by a moving camera. While our training procedure is robust to some degree of scene
motion, we assume that the scenes we are interested in are mostly rigid, i.e., the scene appearance change across different frames is dominated by the camera motion.

3.1. View synthesis as supervision

The key supervision signal for our depth and pose prediction CNNs comes from the task of novel view synthesis: given one input view of a scene, synthesize a new image of the scene seen from a different camera pose. We can synthesize a target view given a per-pixel depth in that image, plus the pose and visibility in a nearby view. As we will show next, this synthesis process can be implemented in a fully differentiable manner with CNNs as the geometry and pose estimation modules. Visibility can be handled, along with non-rigidity and other non-modeled factors, using an “explainability” mask, which we discuss later (Sec. 3.3).

Let us denote $< I_1, \ldots, I_N >$ as a training image sequence with one of the frames $I_t$ being the target view and the rest being the source views $I_s (1 \leq s \leq N, s \neq t)$. The view synthesis objective can be formulated as

$$L_{vs} = \sum_s \sum_p |I_t(p) - \hat{I}_s(p)| ,$$

where $p$ indexes over pixel coordinates, and $\hat{I}_s$ is the source view $I_s$ warped to the target coordinate frame based on a depth image-based rendering module [8] (described in Sec. 3.2), taking the predicted depth $\hat{D}_t$, the predicted $4 \times 4$ camera transformation matrix $\hat{T}_{t \rightarrow s}$ and the source view $I_s$ as input.

Note that the idea of view synthesis as supervision has also been recently explored for learning single-view depth estimation [14, 16] and multi-view stereo [10]. However, to the best of our knowledge, all previous work requires posed image sets during training (and testing too in the case of DeepStereo), while our framework can be applied to standard videos without pose information. Furthermore, it predicts the poses as part of the learning framework. See Figure 2 for an illustration of our learning pipeline for depth and pose estimation.

3.2. Differentiable depth image-based rendering

As indicated in Eq. 1, a key component of our learning framework is a differentiable depth image-based renderer that reconstructs the target view $I_t$ by sampling pixels from a source view $I_s$ based on the predicted depth map $\hat{D}_t$ and the relative pose $\hat{T}_{t \rightarrow s}$.

Let $p_t$ denote the homogeneous coordinates of a pixel in the target view, and $K$ denote the camera intrinsics matrix. We can obtain $p_t$’s projected coordinates onto the source view $p_s$ by

$$p_s \sim K\hat{T}_{t \rightarrow s}\hat{D}_t(p_t)K^{-1}p_t .$$

Notice that the projected coordinates $p_s$ are continuous values. To obtain $I_s(p_s)$ for populating the value of $\hat{I}_s(p_t)$ (see Figure 3), we then use the differentiable bilinear sampling mechanism proposed in the spatial transformer networks [23] that linearly interpolates the values of the 4-pixel neighbors (top-left, top-right, bottom-left, and bottom-right) of $p_s$ to approximate $I_s(p_s)$, i.e.,

$$\hat{I}_s(p_t) = I_s(p_s) = \sum_{i \in \{t, b, l, r\}} w^i I_s(p^i),$$

where $w^i$ is linearly proportional to the spatial proximity between $p_t$ and $p^i$, and $\sum_{i \in \{t, b, l, r\}} w^i = 1$. A similar strategy is used in [54] for learning to directly warp between different views, while here the coordinates for pixel warping are obtained through projective geometry that enables the factorization of depth and camera pose.

3.3. Modeling the model limitation

Note that when applied to monocular videos the above view synthesis formulation implicitly assumes 1) the scene is static without moving objects; 2) there is no occlusion/disocclusion between the target view and the source views; 3) the surface is Lambertian so that the photo-consistency error is meaningful. If any of these assumptions are violated in a training sequence, the gradients could be corrupted and potentially inhibit training. To improve the robustness of our learning pipeline to these factors, we additionally train a explainability prediction network (jointly and simultaneously with the depth and pose networks) that outputs a per-pixel soft mask $\hat{E}_s$ for each target-source pair, indicating the

\[\text{In practice, the CNN estimates the Euler angles and the 3D translation vector, which are then converted to the transformation matrix.}\]
network’s belief in where direct view synthesis will be successfully modeled for each target pixel. Based on the predicted \( \hat{E}_s \), the view synthesis objective is weighted correspondingly by

\[
\mathcal{L}_{vs} = \sum_{i \in S} \sum_{p} |\hat{E}_s(p)| |I_s(p) - \hat{I}_s(p)| .
\] (3)

Since we do not have direct supervision for \( \hat{E}_s \), training with the above loss would result in a trivial solution of the network always predicting \( \hat{E}_s \) to be zero, which perfectly minimizes the loss. To resolve this, we add a regularization term \( \mathcal{L}_{reg}(\hat{E}_s) \) that encourages nonzero predictions by minimizing the cross-entropy loss with constant label 1 at each pixel location. In other words, the network is encouraged to minimize the view synthesis objective, but allowed a certain amount of slack for discounting the factors not considered by the model.

3.4. Overcoming the gradient locality

One remaining issue with the above learning pipeline is that the gradients are mainly derived from the pixel intensity difference between \( I(p_s) \) and the four neighbors of \( I(p_s) \), which would inhibit training if the correct \( p_s \) (projected using the ground-truth depth and pose) is located in a low-texture region or far from the current estimation. This is a well known issue in motion estimation [3].

Empirically, we found two strategies to be effective for overcoming this issue: 1) using a convolutional encoder-decoder architecture with a small bottleneck for the depth network that implicitly constrains the output to be globally smooth and facilitates gradients to propagate from meaningful regions to nearby regions; 2) explicit multi-scale and smoothness loss (e.g., as in [14, 16]) that allows gradients to be derived from larger spatial regions directly. We adopt the second strategy in this work as it is less sensitive to architectural choices. For smoothness, we minimize the \( L_1 \) norm of the second-order gradients for the predicted depth maps (similar to [48]).

Our final objective becomes

\[
\mathcal{L}_{final} = \sum_{l} \mathcal{L}_{vs}^l + \lambda_s \mathcal{L}_{smooth}^l + \lambda_e \sum_{s} \mathcal{L}_{reg}(\hat{E}_s^l) ,
\] (4)

where \( l \) indexes over different image scales, \( s \) indexes over source images, and \( \lambda_s \) and \( \lambda_e \) are the weighting for the depth smoothness loss and the explainability regularization, respectively.

3.5. Network architecture

**Single-view depth** For single-view depth prediction, we adopt the DispNet architecture proposed in [35] that is mainly based on an encoder-decoder design with skip connections and multi-scale side predictions (see Figure 4). All conv layers are followed by ReLU activation except for the prediction layers, where we use \( 1/(\alpha \cdot \text{sigmoid}(x) + \beta) \) with \( \alpha = 10 \) and \( \beta = 0.1 \) to constrain the predicted depth to be always positive within a reasonable range. We also experimented with using multiple views as input to the depth network, but did not find this to improve the results. This is in line with the observations in [47], where optical flow constraints need to be enforced to utilize multiple views effectively.
**Pose** The input to the pose estimation network is the target view concatenated with all the source views (along the color channels), and the outputs are the relative poses between the target view and each of the source views. The network consists of 7 stride-2 convolutions followed by a $1 \times 1$ convolution with $6 \times (N - 1)$ output channels (corresponding to 3 Euler angles and 3-D translation for each source view). Finally, global average pooling is applied to aggregate predictions at all spatial locations. All conv layers are followed by ReLU except for the last layer where no nonlinear activation is applied.

**Explainability mask** The explainability prediction network shares the first five feature encoding layers with the pose network, followed by 5 deconvolution layers with multi-scale side predictions. All conv/deconv layers are followed by ReLU except for the prediction layers with no nonlinear activation. The number of output channels for each prediction layer is $2 \times (N - 1)$, with every two channels normalized by $softmax$ to obtain the explainability prediction for the corresponding source-target pair (the second channel after normalization is $E_\alpha$ and used in computing the loss in Eq. 3).

### 4. Experiments

Here we evaluate the performance of our system, and compare with prior approaches on single-view depth as well as ego-motion estimation. We mainly use the KITTI dataset [15] for benchmarking, but also use the Make3D dataset [42] for evaluating cross-dataset generalization ability.

**Training Details** We implemented the system using the publically available TensorFlow [1] framework. For all the experiments, we set $\lambda = 0.5/l$ (l is the downsampling factor for the corresponding scale) and $\lambda_e = 0.2$. During training, we used batch normalization [21] for all the layers except for the output layers, and the Adam [28] optimizer with $\beta_1 = 0.9, \beta_2 = 0.999$, learning rate of 0.0002 and mini-batch size of 4. The training typically converges after about 150K iterations. All the experiments are performed with image sequences captured with a monocular camera. We resize the images to $128 \times 416$ during training, but both the depth and pose networks can be run fully-convolutionally for images of arbitrary size at test time.

#### 4.1. Single-view depth estimation

We train our system on the split provided by [7], and exclude all the frames from the testing scenes as well as static sequences with mean optical flow magnitude less than 1 pixel for training. We fix the length of image sequences to be 3 frames, and treat the central frame as the target view and the $\pm 1$ frames as the source views. We use images captured by both color cameras, but treated them independently when forming training sequences. This results in a total of 44,540 sequences, out of which we use 40,109 for training and 4,431 for validation.

To the best of our knowledge, no previous systems exist that learn single-view depth estimation in an unsupervised manner from monocular videos. Nonetheless, here we provide comparison with prior methods with depth supervision [7] and recent methods that use calibrated stereo images (i.e. with pose supervision) for training [14, 16]. Since the depth predicted by our method is defined up to a scale factor, for evaluation we multiply the predicted depth maps by a scalar $\hat{s}$ that matches the median with the ground-truth, i.e. $\hat{s} = median(D_{gt})/median(D_{pred})$.

Similar to [16], we also experimented with first pre-training the system on the larger Cityscapes dataset [5] (sample predictions are shown in Figure 5), and then fine-tune on KITTI, which results in slight performance improvement.

**KITTI** Here we evaluate the single-view depth performance on the 697 images from the test split of [7]. As shown in Table 1, our unsupervised method performs comparably with several supervised methods (e.g. Eigen et al. [7] and Garg et al. [14]), but falls short of concurrent work by Godard et al. [16] that uses calibrated stereo images (i.e. with pose supervision) with left-right cycle consistency loss for training. For future work, it would be interesting to see if incorporating the similar cycle consistency loss into our framework could further improve the results. Figure 6...
provides examples of visual comparison between our results and some supervised baselines over a variety of examples. One can see that although trained in an unsupervised manner, our results are comparable to that of the supervised baselines, and sometimes preserve the depth boundaries and thin structures such as trees and street lights better.

We show sample predictions made by our initial Cityscapes model and the final model (pre-trained on Cityscapes and then fine-tuned on KITTI) in Figure 7. Due to the domain gap between the two datasets, our Cityscapes model sometimes has difficulty in recovering the complete shape of the car/bushes, and mistakes them with distant objects.

We also performed an ablation study of the explainability modeling (see Table 1), which turns out only offering a modest performance boost. This is likely because 1) most of the KITTI scenes are static without significant scene motions, and 2) the occlusion/visibility effects only occur in small regions in sequences across a short time span (3-frames), which make the explainability modeling less essential to the success of training. Nonetheless, our explainability prediction network does seem to capture the factors like scene motion and visibility well (see Sec. 4.3), and could potentially be more important for other more challenging datasets.

**Make3D** To evaluate the generalization ability of our single-view depth model, we directly apply our model trained on Cityscapes + KITTI to the Make3D dataset unseen during training. While there still remains a significant performance gap between our method and others supervised using Make3D ground-truth depth (see Table 2), our predictions are able to capture the
1. Single-view depth results on the KITTI dataset [15] using the split of Eigen et al. [7] (Baseline numbers taken from [16]). For training, K = KITTI, and CS = Cityscapes [5]. All methods we compare with use some form of supervision (either ground-truth depth or calibrated camera pose) during training. Note: results from Garg et al. [14] are capped at 50m depth, so we break these out separately in the lower part of the table.

<table>
<thead>
<tr>
<th>Method</th>
<th>Dataset</th>
<th>Supervision</th>
<th>Error metric</th>
<th>Accuracy metric</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>Depth Pose</td>
<td>Abs Rel Sq Rel RMSE RMSE log</td>
</tr>
<tr>
<td>Train set mean</td>
<td>K</td>
<td>✓</td>
<td>0.403 5.530</td>
<td>8.709 0.403</td>
</tr>
<tr>
<td>Eigen et al. [7]</td>
<td>K</td>
<td>✓</td>
<td>0.214 1.605</td>
<td>6.563 0.292</td>
</tr>
<tr>
<td>Eigen et al. [7]</td>
<td>K</td>
<td>✓</td>
<td>0.203 1.548</td>
<td>6.307 0.282</td>
</tr>
<tr>
<td>Liu et al. [32]</td>
<td>K</td>
<td>✓</td>
<td>0.202 1.614</td>
<td>6.523 0.275</td>
</tr>
<tr>
<td>Godard et al. [16]</td>
<td>K</td>
<td>✓</td>
<td>0.148 1.344</td>
<td>5.927 0.247</td>
</tr>
<tr>
<td>Godard et al. [16]</td>
<td>CS + K</td>
<td>✓</td>
<td>0.124 1.076</td>
<td>5.311 0.219</td>
</tr>
<tr>
<td>Ours (w/o explainability)</td>
<td>K</td>
<td>✓</td>
<td>0.221 2.226</td>
<td>7.527 0.294</td>
</tr>
<tr>
<td>Ours</td>
<td>K</td>
<td>✓</td>
<td>0.208 1.768</td>
<td>6.856 0.283</td>
</tr>
<tr>
<td>Ours</td>
<td>CS</td>
<td></td>
<td>0.267 2.686</td>
<td>7.580 0.334</td>
</tr>
<tr>
<td>Ours</td>
<td>CS + K</td>
<td></td>
<td>0.198 1.836</td>
<td>6.565 0.275</td>
</tr>
<tr>
<td>Garg et al. [14] cap 50m</td>
<td>K</td>
<td>✓</td>
<td>0.169 1.080</td>
<td>5.104 0.273</td>
</tr>
<tr>
<td>Ours (w/o explainability) cap 50m</td>
<td>K</td>
<td>✓</td>
<td>0.208 1.551</td>
<td>5.452 0.273</td>
</tr>
<tr>
<td>Ours cap 50m</td>
<td>K</td>
<td>✓</td>
<td>0.201 1.391</td>
<td>5.181 0.264</td>
</tr>
<tr>
<td>Ours cap 50m</td>
<td>CS</td>
<td></td>
<td>0.260 2.232</td>
<td>6.148 0.321</td>
</tr>
<tr>
<td>Ours cap 50m</td>
<td>CS + K</td>
<td></td>
<td>0.190 1.436</td>
<td>4.975 0.258</td>
</tr>
</tbody>
</table>

Table 1. Results on the Make3D dataset [42]. Similar to ours, Godard et al. [16] do not utilize any of the Make3D data during training, and directly apply the model trained on KITTI+Cityscapes to the test set. Following the evaluation protocol of [16], the errors are only computed where depth is less than 70 meters in a central image crop.

<table>
<thead>
<tr>
<th>Method</th>
<th>Supervision</th>
<th>Error metric</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Depth Pose</td>
</tr>
<tr>
<td>Train set mean</td>
<td>✓</td>
<td>0.876 13.98</td>
</tr>
<tr>
<td>Karsch et al. [23]</td>
<td>✓</td>
<td>0.428 5.079</td>
</tr>
<tr>
<td>Liu et al. [31]</td>
<td>✓</td>
<td>0.475 6.562</td>
</tr>
<tr>
<td>Laina et al. [31]</td>
<td>✓</td>
<td>0.204 1.840</td>
</tr>
<tr>
<td>Godard et al. [16]</td>
<td>✓</td>
<td>0.544 10.94</td>
</tr>
<tr>
<td>Ours</td>
<td></td>
<td>0.383 5.321</td>
</tr>
</tbody>
</table>

Table 2. Similar to ours, Godard et al. [16] do not utilize any of the Make3D data during training, and directly apply the model trained on KITTI+Cityscapes to the test set. Following the evaluation protocol of [16], the errors are only computed where depth is less than 70 meters in a central image crop.

Figure 8. Our sample predictions on the Make3D dataset. Note that our model is trained on KITTI + Cityscapes only, and directly tested on Make3D.

4.2. Pose estimation

To evaluate the performance of our pose estimation network, we applied our system to the official KITTI odometry split (containing 11 driving sequences with ground truth odometry obtained through the IMU/GPS readings, which we use for evaluation purpose only), and used sequences 00-08 for training and 09-10 for testing. In this experiment, we fix the length of input image sequences to our system to 5 frames. We compare our ego-motion estimation with two variants of monocular ORB-SLAM [37] (a well-established SLAM system): 1) ORB-SLAM (full), which recovers odometry using all frames of the driving sequence (i.e., allowing loop closure and re-localization), and 2) ORB-SLAM (short), which runs on 5-frame snippets (same as our input setting). Another baseline we compare with is the dataset mean of car motion (using ground-truth odometry) for 5-frame snippets. To resolve scale ambiguity during evaluation, we first optimize the scaling factor for the predictions made by each method to best align with the ground truth, and then measure the Absolute Trajectory Error (ATE) [37] as the metric. ATE is computed on 5-frame snippets and averaged over the full sequence. As shown in Table 3 and Fig. 9, our method outperforms both baselines (mean odometry and ORB-SLAM (short)) that share the same input setting as ours, but falls short of ORB-SLAM (full), which leverages whole sequences (1501 for seq. 09 and 1201 for seq. 10) for loop closure and re-localization.

For better understanding of our pose estimation results, we allow evaluation ORB-SLAM (full) we break down the trajectory of the full sequence into 5-frame snippets with the reference coordinate frame adjusted to the central frame of each snippet.

For evaluating ORB-SLAM (full) we break down the trajectory of the full sequence into 5-frame snippets with the reference coordinate frame adjusted to the central frame of each snippet.
show in Figure 9 the ATE curve with varying amount of side-
rotation by the car between the beginning and the end of a se-
sequence. Figure 9 suggests that our method is significantly bet-
ter than ORB-SLAM (short) when side rotation is small, and is comparable to ORB-SLAM (full) across the entire spectrum.

Table 3. Absolute Trajectory Error (ATE) on the KITTI odome-
try split averaged over all 5-frame snippets (lower is better). Our
method outperforms baselines with the same input setting, but falls short of ORB-SLAM (full) that uses strictly more data.

<table>
<thead>
<tr>
<th>Method</th>
<th>Seq. 09</th>
<th>Seq. 10</th>
</tr>
</thead>
<tbody>
<tr>
<td>ORB-SLAM (full)</td>
<td>0.014 ± 0.008</td>
<td>0.012 ± 0.011</td>
</tr>
<tr>
<td>ORB-SLAM (short)</td>
<td>0.064 ± 0.141</td>
<td>0.064 ± 0.130</td>
</tr>
<tr>
<td>Mean Odom.</td>
<td>0.032 ± 0.026</td>
<td>0.028 ± 0.023</td>
</tr>
<tr>
<td>Ours</td>
<td>0.021 ± 0.017</td>
<td>0.020 ± 0.015</td>
</tr>
</tbody>
</table>

Figure 9. Absolute Trajectory Error (ATE) at different left/right
turning magnitude (coordinate difference in the side-direction be-
tween the start and ending frame of a testing sequence). Our
method performs significantly better than ORB-SLAM (short) when side rotation is small, and is comparable with ORB-SLAM (full) across the entire spectrum.

4.3. Visualizing the explainability prediction

We visualize example explainability masks predicted by our
network in Figure 10. The first three rows suggest that the network has learned to identify dynamic objects in the scene as unexplain-
able by our model, and similarly, rows 4–5 are examples of ob-
jects that disappear from the frame in subsequent views. The last
two rows demonstrate the potential downside of explainability-
weighted loss: the depth CNN has low confidence in predicting thin structures well, and tends to mask them as unexplainable.

5. Discussion

We have presented an end-to-end learning pipeline that utilizes
the task of view synthesis for supervision of single-view depth and camera pose estimation. The system is trained on unlabeled
videos, and yet performs comparably with approaches that require ground-truth depth or pose for training. Despite good performance on the benchmark evaluation, our method is by no means close to solving the general problem of unsupervised learning of 3D scene structure inference. A number of major challenges are yet to be addressed: 1) our current framework does not explicitly estimate scene dynamics and occlusions (although they are implicitly taken into account by the explainability masks), both of which are critical factors in 3D scene understanding. Direct modeling of scene dynamics through motion segmentation (e.g. [48, 40]) could be a potential solution; 2) our framework assumes the camera intrinsics are given, which forbids the use of random Internet videos with un-
known camera types/calibration — we plan to address this in future work; 3) depth maps are a simplified representation of the under-
lying 3D scene. It would be interesting to extend our framework to learn full 3D volumetric representations (e.g. [46]).

Another interesting area for future work would be to investi-
gate in more detail the representation learned by our system. In
particular, the pose network likely uses some form of image cor-
respondence in estimating the camera motion, whereas the depth estimation network likely recognizes common structural features of scenes and objects. It would be interesting to probe these, and investigate the extent to which our network already performs, or could be re-purposed to perform, tasks such as object detection and semantic segmentation.

Acknowledgments: We thank our colleagues, Sudheendra Vijaya-
narasimhan, Susanna Ricco, Cordelia Schmid, Rahul Sukthankar, and Ka-
terina Fragkiadaki for their help. We also thank the anonymous reviewers for their valuable comments. TZ would like to thank Shubham Tulsiani for helpful discussions, and Clement Godard for sharing the evaluation code. This work is also partially funded by Intel/NSF VEC award IIS-1539099.
References


