Veritatem Dies Aperit - Temporally Consistent Depth Prediction Enabled by a Multi-Task Geometric and Semantic Scene Understanding Approach

Amir Atapour-Abarghouei¹ Toby P. Breckon¹,²
¹Department of Computer Science – ²Department of Engineering
Durham University, UK
{amir.atapour-abarghouei,toby.breckon}@durham.ac.uk

Abstract

Robust geometric and semantic scene understanding is ever more important in many real-world applications such as autonomous driving and robotic navigation. In this paper, we propose a multi-task learning-based approach capable of jointly performing geometric and semantic scene understanding, namely depth prediction (monocular depth estimation and depth completion) and semantic scene segmentation. Within a single temporally constrained recurrent network, our approach uniquely takes advantage of a complex series of skip connections, adversarial training and the temporal constraint of sequential frame recurrence to produce consistent depth and semantic class labels simultaneously. Extensive experimental evaluation demonstrates the efficacy of our approach compared to other contemporary state-of-the-art techniques.

1. Introduction

As scene understanding grows in popularity due to its applicability in many areas of interest for industry and academia, scene depth has become ever more important as an integral part of this task. Whilst in many current autonomous driving solutions, imperfect stereo camera set-ups or expensive LiDAR sensors are used to capture depth, research has recently focused on refining estimated depth with corrupted or missing regions in post-processing, rendering it more useful in any downstream applications [6, 78, 84]. Moreover, monocular depth estimation has received significant attention within the research community as a cheap and innovative alternative to other more expensive and performance-limited technologies [8, 24, 29, 87].

Pixel-level image understanding, namely semantic segmentation, also plays an important role in many vision-based systems. Significant success has been achieved using Convolutional Neural Networks (CNN) in this field [10, 17, 53, 66, 70] and many others such as image classification [54], object detection [88] and alike in recent years.
be very effective when the input and output of a CNN share similar high-level spatial features \cite{60, 66, 73, 79}. We make use of a complex network of skip connections throughout the architecture to guarantee that no high-level spatial features are lost during training as the features are downsampled. In short, our main contributions are as follows:

- **Depth Prediction** - via a supervised multi-task model adversarially trained using complex skip connections that can predict depth (monocular depth estimation and depth completion) having been trained on high-quality synthetic training data \cite{67} (Section 3.2).
- **Semantic Segmentation** - via the same multi-task model, which is capable of performing the task of semantic scene segmentation as well as the aforementioned depth estimation/completion (Section 3.3).
- **Temporal Continuity** - temporal information is explicitly taken into account during training using both recurrent network feedback and gradients from a pre-trained frozen optical flow network.

This leads to a novel scene understanding approach capable of temporally consistent geometric depth prediction and semantic scene segmentation whilst outperforming prior work across the domains of monocular depth estimation \cite{8, 25, 29, 49, 83, 87}, completion \cite{9, 36, 50, 82} and semantic segmentation \cite{10, 17, 40, 52, 53, 59, 74, 75, 86}.

2. Related Work

We consider relevant prior work over three distinct areas, semantic segmentation (Section 2.1), monocular depth estimation (Section 2.2), and depth completion (Section 2.3).

2.1. Semantic Segmentation

Within the literature, promising results have been achieved using fully-convolutional networks \cite{53}, saved pooling indices \cite{10}, skip connections \cite{66}, multi-path refinement \cite{48}, spatial pyramid pooling \cite{85}, attention modules focusing on scale or channel \cite{18, 81} and others.

Temporal information in videos has also been used to improve segmentation accuracy or efficiency. \cite{26} proposes a spatio-temporal LSTM based on frame features for higher accuracy. Labels are propagated in \cite{58} using gated recurrent units. In \cite{27}, features from preceding frames are warped via flow vectors to reinforce the current frame features. On the other hand, \cite{69} reuses previous frame features to reduce computation. In \cite{89}, an optical flow network \cite{23} is used to propagate features from key frames to the current one. Similarly, \cite{77} uses an adaptive key frame scheduling policy to improve both accuracy and efficiency. Additionally, \cite{47} proposes an adaptive feature propagation module that employs spatially variant convolutions to fuse the frame features, thus further improving efficiency. Even though the main objective of this work is not semantic segmentation, it can be demonstrated that when the main objective (depth prediction) is performed alongside semantic segmentation, the results are superior to when the tasks are performed individually (Table 1).

2.2. Monocular Depth Estimation

Estimating depth from a single colour image is very desirable as unlike stereo correspondence \cite{68}, structure from motion \cite{16} and alike \cite{1, 71}, it leads to a system with reduced size, weight, power and computational requirements. For instance, \cite{11} employs sparse coding to estimate depth, while \cite{24, 25} generates depth from a two-scale network trained on RGB and depth. Other supervised models such as \cite{45, 46} have also achieved impressive results despite the scarcity of ground truth depth for supervision.

Recent work has led to the emergence of new techniques that calculate disparity by reconstructing corresponding views within a stereo correspondence framework without ground truth depth. The work by \cite{76} learns to generate the right view from the left image used as the input while producing an intermediary disparity map. Likewise, \cite{29} uses bilinear sampling \cite{39} and left/right consistency incorporated into training for better results. In \cite{87}, depth and camera motion are estimated by training depth and pose prediction networks, indirectly supervised via view synthesis. The model in \cite{44} is supervised by sparse ground truth depth and the model is then enforced within a stereo framework via an image alignment loss to output dense depth.

Additionally, contemporary supervised approaches such as \cite{8} have taken to using synthetic depth data to produce sharp and crisp depth outputs. In this work, we also utilize synthetic data \cite{67} in a directly supervised training framework to perform the task of monocular depth estimation.
Moreover, temporal information and semantic segmentation are essentially the same except for their output streams. These two modules are individually trained to perform the same tasks (Table 1). This has been made possible using a synthetic dataset [67] in which both ground truth depth and pixel-wise segmentation labels are made possible using a synthetic dataset [67] in which both ground truth depth and pixel-wise segmentation labels are available for video sequences of urban driving scenarios.

### 3. Proposed Approach

Our approach is designed to perform two tasks using a single joint model: depth estimation/completion (Section 3.2) and semantic segmentation (Section 3.3). By combining two separate objectives (predicting depth and pixel-wise class labels), our model learns to generate a complete depth image (Figure 2 (B) - right) which fuses the features and begins the decoding process and input streams are encoded, the middle stream (Figure 2 (B) - middle), which fuse the features and begins the decoding process, and finally the output streams (Figure 2 (B) - right), in which the results are generated.

As seen in Figure 2 (A), two of the inputs are RGB or RGB-D images (depending on whether monocular depth estimation to create depth, or depth completion to fill holes within an existing depth image, is the focus) from the current and previous time steps. The two input streams that decode these share their weights. The third input is the depth generated at the previous time step. The middle section of the network fuses and decodes the input features and finally the output streams produce the results (scene depth and segmentation). Every layer of the network contains two convolutions, batch normalization [37] and PReLU [31].

Following recent successes of approaches using skip connections [60, 66, 73, 79], we utilize a series of skip connections within our architecture (Figure 2 (B)). Our inputs and outputs, despite containing different types of information (RGB, depth and pixel-wise class labels), relate to consecutive frames from the same scene and therefore, share high-frequency information such as certain object boundaries, structures, geometry and alike, ensuring skip connections can be of significant value in improving the results. By combining two separate objectives (predicting depth and pixel-wise class labels) within our network, in which the input streams and middle streams are fully trained on both tasks, the results are better than when two separate networks are individually trained to perform the same tasks (Table 1).

Even though the entire network is trained as one entity, in our discussions, the parts of the network responsible for predicting depth will be referred to as \( G_1 \) and the portions involved in semantic segmentation \( G_2 \). These two modules are essentially the same except for their output streams.

### 3.2. Depth Estimation / Completion

We consider depth prediction as a supervised image-to-image translation problem, wherein an input RGB image (for depth estimation) or RGB-D image (with the depth channel containing holes for depth completion) is translated to a complete depth image. More formally, a generative model \( (G_1) \) approximates a mapping function that takes as its input an image \( x \) (RGB or RGB-D with holes) and outputs an image \( y \) (complete depth image) \( G_1 : x \rightarrow y \).

The initial solution would be to minimize the Euclidean distance between the pixel values of the output \( (G_1(x)) \) and the ground truth depth \( (y) \). This simple reconstruction mechanism forces the model to generate images that

### Table 1: Comparison of depth prediction and segmentation tasks performed in one single network and two separate networks.

<table>
<thead>
<tr>
<th>Method</th>
<th>Abs. Rel.</th>
<th>Sq. Rel.</th>
<th>RMSE</th>
<th>RMSE log</th>
<th>Depth Accuracy (higher, better)</th>
<th>Segmentation (higher, better)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>( \sigma &lt; 1.25 )</td>
<td>( \sigma &lt; 1.25^2 )</td>
<td>( \sigma &lt; 1.25^3 )</td>
<td>( \sigma &lt; 1.25^3 )</td>
<td>Accuracy</td>
<td>IoU</td>
</tr>
<tr>
<td>Two Models</td>
<td>0.245</td>
<td>1.513</td>
<td>6.323</td>
<td>0.274</td>
<td>0.803</td>
<td>0.856</td>
</tr>
<tr>
<td>One Model</td>
<td>0.208</td>
<td>1.402</td>
<td>6.026</td>
<td>0.269</td>
<td>0.836</td>
<td>0.901</td>
</tr>
</tbody>
</table>

Figure 2: Comparing the results of the approach on synthetic test set when the model is trained with and without temporal consistency. **RGB**: input colour image; **GTD**: Ground Truth Depth; **GTS**: Ground Truth Segmentation; **TS**: Temporal Segmentation; **TD**: Temporal Depth; **ND**: Non-Temporal Segmentation.
Experiments with an L input RGB-D image in which the depth channel is y where:

\[ \mathcal{L}_{rec} = \| G_1(x) - y \|_1, \]

where \( x \) is the input image, \( G_1(x) \) is the output and \( y \) the ground truth. For depth completion, however, the input \( x \) is a four-channel RGB-D image with the depth containing holes that would occur during depth sensing. Since we use synthetic data [67], we only have access to hole-free pixel-perfect ground truth depth. While one could naively cut out random sections of the depth image to simulate holes, as other approaches have done [62, 80], we opt for creating realistic and semantically meaningful holes with characteristics of those found in real-world images [6]. A separate model is thus created and tasked with predicting where holes would be by means of pixel-wise segmentation. A number of stereo images (30,000) [28] are used to train the hole prediction model by calculating the disparity using Semi-Global Matching [34] and generating a hole mask (\( M \)) which indicates which image regions contain holes. The left RGB image is used as the input and the generated mask as the ground truth label, with cross-entropy as the loss function.

When our main model is being trained to perform depth completion, the hole mask generated by the hole prediction network is employed to create the depth channel of the input RGB-D image. Subsequently, the reconstruction loss is:

\[ \mathcal{L}_{rec} = \| (1 - M) \odot G_1(x) - (1 - M) \odot y \|_1, \]

where \( \odot \) is the element-wise product operation and \( x \) the input RGB-D image in which the depth channel is \( y \odot M \). Experiments with an L2 loss returned similar results.

However, the sole use of a reconstruction loss would lead to blurry outputs since monocular depth estimation and depth completion are multi-modal problems, i.e., several plausible depth outputs can correctly correspond to a region of an RGB image. This multi-modality results in the generative model (\( G_1 \)) averaging all possible modes rather than selecting one, leading to blurring effects in the output. To prevent this, adversarial training [30] has become prevalent within the literature [8, 22, 38, 62, 80] since it forces the model to select a mode from the distribution resulting in better quality outputs. In this vein, our depth generation model (\( G_1 \)) takes \( x \) as its input and produces fake samples \( G_1(x) = \tilde{y} \) while a discriminator (\( D \)) is adversarially trained to distinguish fake samples \( \tilde{y} \) from ground truth samples \( y \). The adversarial loss is thus as follows:

\[ \mathcal{L}_{adv} = \min_{G_1} \max_{D} \mathbb{E}_{x,y \sim \mathbb{P}_d(x,y)} \left[ \log D(x, y) \right] + \mathbb{E}_{x \sim \mathbb{P}_d(x)} \left[ \log (1 - D(x, G_1(x))) \right], \]

where \( \mathbb{P}_d \) is the data distribution defined by \( \tilde{y} = G_1(x) \), with \( x \) being the generator input and \( y \) the ground truth.

Additionally, a smoothing term [29, 32] is utilized to encourage the model to generate more locally-smooth depth outputs. Output depth gradients (\( \partial G_1(x) \)) are penalized using L1 regularization, and an edge-aware weighting term based on input image gradients (\( \partial x \)) is used since image gradients are stronger where depth discontinuities are most likely found. The smoothing loss is therefore as follows:

\[ \mathcal{L}_s = \| \partial G_1(x) \|_1 \| \partial x \|_1, \]

where \( x \) is the input and \( G_1(x) \) the depth output. The gradients are summed over vertical and horizontal axes.

Table 2: Numerical results with different components of loss. T: Temporal training; T: Non-Temporal training; R: Reconstruction loss; A: Adversarial loss; SC: Skip Connections; S: Smoothing loss; OF: Optical Flow.

<table>
<thead>
<tr>
<th>Method</th>
<th>Abs. Rel.</th>
<th>Sq. Rel.</th>
<th>RMSE</th>
<th>RMSE log</th>
<th>( \sigma &lt; 1.25 )</th>
<th>( \sigma &lt; 1.25^2 )</th>
<th>( \sigma &lt; 1.25^3 )</th>
<th>Accuracy</th>
<th>IoU</th>
</tr>
</thead>
<tbody>
<tr>
<td>T/R</td>
<td>0.991</td>
<td>1.964</td>
<td>7.393</td>
<td>0.402</td>
<td>0.598</td>
<td>0.684</td>
<td>0.698</td>
<td>0.156</td>
<td>0.335</td>
</tr>
<tr>
<td>T/R/A</td>
<td>0.851</td>
<td>1.798</td>
<td>6.826</td>
<td>0.368</td>
<td>0.692</td>
<td>0.750</td>
<td>0.778</td>
<td>0.341</td>
<td>0.435</td>
</tr>
<tr>
<td>T/R/A/SC</td>
<td>0.655</td>
<td>1.616</td>
<td>6.473</td>
<td>0.278</td>
<td>0.753</td>
<td>0.812</td>
<td>0.838</td>
<td>0.669</td>
<td>0.738</td>
</tr>
<tr>
<td>T/R/A/SC/S</td>
<td>0.412</td>
<td>1.573</td>
<td>6.256</td>
<td>0.258</td>
<td>0.793</td>
<td>0.875</td>
<td>0.887</td>
<td>0.693</td>
<td>0.741</td>
</tr>
<tr>
<td>N/R/A/SC/S</td>
<td>0.534</td>
<td>1.602</td>
<td>6.469</td>
<td>0.275</td>
<td>0.758</td>
<td>0.820</td>
<td>0.856</td>
<td>0.614</td>
<td>0.681</td>
</tr>
<tr>
<td>T/R/A/SC/S/OF</td>
<td>0.208</td>
<td>1.402</td>
<td>6.026</td>
<td>0.269</td>
<td>0.836</td>
<td>0.901</td>
<td>0.926</td>
<td>0.748</td>
<td>0.764</td>
</tr>
</tbody>
</table>

Figure 4: Comparing the performance of the approach with differing components of the loss function removed.
our pre-trained optical flow network \cite{65}, which utilizes a coarse-to-fine spatial pyramid to learn residual flow at each scale, into our pipeline to explicitly enforce consistency in the presence of camera/scene motion. At each time step \( n \), the flow between the ground truth depth frames \( n \) and \( n-1 \) is estimated using our pre-trained optical flow network \cite{65} as well as the flow between generated outputs from the same frames. The gradients from the optical flow network \( F \) are used to train the generator \( G_1 \) to capture motion information and temporal continuity by minimizing the End Point Error (EPE) between the produced flows. Hence, the last component of our loss function is:

\[
\mathcal{L}_{\text{V}} = \| F(G_1(x_n), G_1(x_{n-1})) - F(y_n, y_{n-1}) \|_2, \quad (5)
\]

where \( x \) and \( y \) are input and ground truth depth images respectively and \( n \) the time step. While we utilize ground truth depth as inputs to the optical flow network, colour images can also be equally viable inputs. However, since our training data contains noisy environmental elements (e.g., lighting variations, rain, etc.), using the sharp and clean depth images leads to more desirable results.

Within the final decoder used exclusively for depth prediction, outputs are produced at four scales, following \cite{29}. Each scale output is twice the spatial resolution of its previous scale. The overall depth loss is therefore the sum of losses calculated at every scale:

\[
\mathcal{L}_{\text{depth}} = \sum_{c=1}^{C} (\lambda_{\text{rec}} \mathcal{L}_{\text{rec}} + \lambda_{\text{adv}} \mathcal{L}_{\text{adv}} + \lambda_{s} \mathcal{L}_{s} + \lambda_{V} \mathcal{L}_{\text{V}}). \quad (6)
\]

The weighting coefficients \( \lambda \) are empirically selected (Section 3.4). These loss components, used to optimize depth fidelity, are used alongside the semantic segmentation loss, explained in Section 3.3.

### 3.3. Semantic Segmentation

As semantic segmentation is not the primary focus of our approach, but only used to enforce deeper and better representation learning within our model, we opt for a simple and efficient fully-supervised training procedure for our segmentation \( G_2 \). The RGB or RGB-D image is used as the input and the network outputs class labels. Pixel-wise softmax with cross-entropy is used as the loss function, with the loss summed over all the pixels within a batch:

\[
\mathcal{L}_{\text{seg}} = -\log(P_i(G_2(x))\mid x), \quad (8)
\]

where \( G_2(x) \) denotes the network output for the segmentation task, \( a_k(x) \) is the feature activation for channel \( k \), \( K \) is the number of classes, \( P_k(x) \) is the approximated maximum function and \( i \) is the ground truth label for image pixels. The loss is summed for all pixels within the images.

Finally, since the entire network is trained as one unit, the joint loss function is as follows:

\[
\mathcal{L} = \mathcal{L}_{\text{depth}} + \lambda_{\text{seg}} \mathcal{L}_{\text{seg}}. \quad (9)
\]

with coefficients selected empirically (Section 3.4).

### 3.4. Implementation Details

Synthetic data \cite{67} consisting of RGB, depth and class labels are used for training. The discriminator follows the architecture of \cite{64}, and the optical flow network \cite{65} is pre-trained on the KITTI dataset \cite{56}. Experiments with the Sintel dataset \cite{15} returned similar, albeit slightly inferior, results. The discriminator uses convolution-BatchNorm-leaky ReLU (slope = 0.2) modules. The dataset \cite{67} contains numerous sequences some spanning thousands of frames. However, a feedback network taking in high-resolution images \((512 \times 128)\) back-propagating over thousands of time steps is intractable to train. Empirically, we found training over sequences of 10 frames offers a reasonable trade-off between accuracy and training efficiency. Mini-batches are loaded in as tensors containing two sequences of 10 frames each, resulting in roughly 10,000 batches overall. All implementation is done in PyTorch \cite{61}, with Adam \cite{32} providing the best optimization (\( \beta_1 = \)
Figure 6: Results of our approach applied to KITTI [2, 56]. RGB: input colour image; GTD: Ground Truth Depth; MDE: Monocular Depth Estimation; GTS: Ground Truth Segmentation; GS: Generated Segmentation.

Figure 7: Our results on locally captured data. SD: Depth via Stereo Correspondence; DC: Depth Completion; MDE: Monocular Depth Estimation; S: Semantic Segmentation. 0.5, $\beta_2 = 0.999$, $\alpha = 0.0002$). The weighting coefficients in the loss function are empirically chosen to be $\lambda_{rec}=1000$, $\lambda_{adv}=100$, $\lambda_s=10$, $\lambda_V=1$, $\lambda_{seg}=10$.

4. Experimental Results

We assess our approach using ablation studies and both qualitative and quantitative comparisons with state-of-the-art methods applied to publicly available datasets [2, 14, 20, 28, 56]. We also utilize our own synthetic test set and data captured locally to further evaluate the approach.

4.1. Ablation Studies

A crucial part of our work is demonstrating that every component of the approach is integral to the overall performance. We train our model to perform two tasks based on the assumption that the network is forced to learn more about the scene if different objectives are to be accomplished. We demonstrate this by training one model performing both tasks and two separate models focusing on each and conducting tests on randomly selected synthetic sequences [67]. As seen in Table 1 both tasks (monocular depth estimation and semantic segmentation) perform better when the model is trained on both. Moreover, since the segmentation pipeline does not receive any explicit temporal supervision (from the optical flow network) and its temporal continuity is only enforced by the input and middle streams trained by the depth pipeline, when the two pipelines are disentangled, the segmentation results become far worse than the depth results (Table 1).

Figure 3 depicts the quality of the outputs when the model is a feedback network trained temporally compared to our model when the output depth from the previous time step is not used as the input during training. We can clearly see that both depth and segmentation results are of higher fidelity when temporal information is used during training.

Additionally, our depth prediction pipeline uses several loss functions. We employ the same test sequences to evaluate our model trained as different components are removed. Table 2 demonstrates the network temporally trained with all the loss components (T/R/A/SC/S/OF) outperforms models trained without specific ones. Qualitatively, we can see in Figure 4 that the results are far better when the network is fully trained with all the components. Specifically, the set of skip connections used in the network make a significant difference in the quality of the outputs.

4.2. Semantic Segmentation

Segmentation is not the focus of this work and is mainly used to boost the performance of depth prediction. However, we extensively evaluate our segmentation pipeline which outperforms several well-known comparators. We utilize Cityscapes [20] and CamVid [14] test sets for our performance evaluation despite the fact that our model is solely trained on synthetic data and without any domain adaptation should not be expected to perform well on naturally sensed real-world data. The effective performance of our segmentation points to the generalization capabilities of our model. When tested on CamVid [14], our approach produces better results compared to well-established techniques such as [10, 40, 59, 75] despite the lower quality.

<table>
<thead>
<tr>
<th>Method</th>
<th>PSNR</th>
<th>SSIM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Holes</td>
<td>33.73</td>
<td>0.372</td>
</tr>
<tr>
<td>ICA [82]</td>
<td>31.01</td>
<td>0.488</td>
</tr>
<tr>
<td>GIF [50]</td>
<td>44.57</td>
<td>0.972</td>
</tr>
<tr>
<td>FDF [9]</td>
<td>46.13</td>
<td>0.986</td>
</tr>
<tr>
<td>Ours</td>
<td><strong>47.45</strong></td>
<td><strong>0.991</strong></td>
</tr>
</tbody>
</table>

Table 5: Structural integrity analysis post depth completion.
Figure 8: Comparison of various completion methods applied to the synthetic test set. RGB: input colour image; GTD: Ground Truth Depth; DH: Depth Holes; FDF: Fourier based Depth Filling [9]; GLC: Global and Local Completion [36]; ICA: Inpainting with Contextual Attention [82]; GIF: Guided Inpainting and Filtering [50].

<table>
<thead>
<tr>
<th>Method</th>
<th>Error Metrics (lower, better)</th>
<th>Accuracy Metrics (higher, better)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Abs. Rel.</td>
<td>Sq. Rel.</td>
</tr>
<tr>
<td>Train Set Mean</td>
<td>0.403</td>
<td>0.530</td>
</tr>
<tr>
<td>Eigen et al.</td>
<td>0.203</td>
<td>1.548</td>
</tr>
<tr>
<td>Liu et al.</td>
<td>0.202</td>
<td>1.614</td>
</tr>
<tr>
<td>Zhou et al.</td>
<td>0.208</td>
<td>1.768</td>
</tr>
<tr>
<td>Godard et al.</td>
<td>0.148</td>
<td>1.344</td>
</tr>
<tr>
<td>Zhan et al.</td>
<td><strong>0.144</strong></td>
<td>1.391</td>
</tr>
<tr>
<td>Our Approach</td>
<td>0.193</td>
<td>1.438</td>
</tr>
</tbody>
</table>

Table 6: Numerical comparison of monocular depth estimation over the KITTI [28] data split in [25]. All comparators are trained and tested on the same dataset (KITTI [28]) while our approach is trained on [67] and tested using [25].

of the input images as seen in Table 4. As for Cityscapes [20], the test set does not contain video sequences, but our temporal model still outperforms approaches such as [17] [52] [53] [74] [86], as demonstrated in Table 3.

Examples of the segmentation results over both datasets are seen in Figure 5. Additionally, we also use the KITTI semantic segmentation data [2] in our tests and as shown in Figure 6, our approach produces high fidelity semantic class labels despite including no domain adaptation.

4.3. Depth Completion

Evaluation for depth completion ideally requires dense ground truth scene depth. However, no such dataset exists for urban driving scenarios, which is why we utilize randomly selected previously unseen synthetic data with available dense depth images to assess the results. Our model generates full scene depth and the predicted depth values for the missing regions of the depth image are subsequently blended in with the known regions of the image using [63]. Figure 8 shows a comparison of our results against other contemporary approaches [9] [36] [50] [82]. As seen from the enlarged sections, our approach produces minimal artefacts (blurring, streaking, etc.) compared to the other techniques. To evaluate the structural integrity of the results post completion, we also numerically assess the performance of our approach and the comparators. As seen in Table 5, our approach quantitatively outperforms the comparators as well.

While blending [63] might work well for colour images with a connected missing region, significant quantities of small and large holes in depth images can lead to undesirable artefacts such as stitch mark or burning effects post blending. Examples of artefacts can be seen in Figure 7, which demonstrates the results of the approach applied to locally captured data. This is further discussed in Section 5.

4.4. Monocular Depth Estimation

As the main focus of our model, our monocular depth estimation model is evaluated against contemporary state-of-the-art approaches [8] [25] [29] [49] [83] [87]. Following the conventions of the literature, we use the data split suggested in [25] as the test set. These images are selected from random sequences and do not follow a temporally sequential pattern, while our full approach requires video sequences as its input. As a result, we apply our approach to all the sequences from which the images are chosen but the evaluation itself is only performed on the 697 test images.

For numerical assessment, the generated depth is corrected for the differences in focal length between the training [67] and testing data [28]. As seen in Table 6, our approach outperforms [25] [49] [87] across all metrics and stays
Figure 9: Comparing the results of the approach against [87, 29, 44, 8]. Images have been adjusted for better visualization.

RGB: input colour image; GTD: Ground Truth Depth; DEV: Depth and Ego-motion from Video [87]; LRC: Left-Right Consistency [29]; SSE: Semi-supervised Estimation [44]; EST: Estimation via Style Transfer [8]; GS: Generated Segmentation.

It is important to note that all of these comparators are trained on the same dataset as the one used for testing [28] while our approach is trained on synthetic data [67] without domain adaptation and has not seen a single image from [28]. Additionally, none of the other comparators is capable of producing temporally consistent outputs as all of them operate on a frame level. As this cannot be readily illustrated via still images within Figures 8 and 9, we kindly invite the reader to view the supplementary video material accompanying the paper.

We also assess our model using the data split of KITTI [56] and qualitatively evaluate the results, since the ground truth images in [56] are of higher quality than the laser data and provide CAD models as replacements for the cars in the scene. As shown in Figure 6, our method produces sharp and crisp depth outputs with segmentation results in which object boundaries and thin structures are well preserved.

5. Limitations and Future Work

Even though our approach can generate temporally consistent depth and segmentation by utilizing a feedback network, this can lead to error propagation, i.e., when an erroneous output is generated at one time step, the invalid values will continually propagate to future frames. This can be resolved by exploring the use of 3D convolutions or regularization terms aimed at penalizing propagated invalid outputs. Moreover, as mentioned in Section 4.3, blending the depth output into the known regions of the depth [63] produces undesirable artefacts in the results. This can be rectified by incorporating the blending operation into the training procedure. In other words, the blending itself will take place before the supervisory signal is back-propagated through the network during training, which would force the network to learn these artefacts, removing any need for post-processing. As for our segmentation component, no explicit temporal consistency enforcement or class balancing is performed, which has lead to frame-to-frame flickering and lower accuracy with unbalanced classes (e.g., pedestrians, cyclists). By improving segmentation, the entire model can benefit from a performance boost. Most of all, the use of domain adaptation [8, 35] can significantly improve all results since despite its generalization capabilities, the model is only trained on synthetic data and should not be expected to perform just as well on naturally-sensed real-world images.

6. Conclusion

We propose a multi-task model capable of performing depth prediction and semantic segmentation in a temporally consistent manner using a feedback network that takes as its recurrent input the output generated at the previous time step. Using a series of dense skip connections, we ensure that no high-frequency spatial information is lost during feature down-sampling within the training process. We consider the task of depth prediction within the areas of depth completion and monocular depth estimation, and therefore train models based on both objectives within the depth prediction component. Using extensive experimentation, we demonstrate that our model achieves much better results when it performs depth prediction and segmentation at the same time compared to two separate networks performing the same tasks. The use of skip connections is also shown to be significantly effective in improving the results for both depth prediction and segmentation tasks. Although certain isolated issues remain, experimental evaluation demonstrates the efficacy of our approach compared to contemporary state-of-the-art methods tackling the same problem domains [17, 29, 40, 53, 82, 83, 87].

We kindly invite the readers to refer to the video: https://vimeo.com/325161805 for more information and larger improved-quality result images.
