稠密重建

DeepFusion: Real-Time Dense 3D Reconstruction for Monocular SLAM using Single-View Depth and Gradient Predictions

Tristan Laidlow, Jan Czarnowski and Stefan Leutenegger

深度相机只能在室内小环境内使用；基于图像反投影误差的方法误差较大，且有尺度不确定性。

***Abstract*— While the keypoint-based maps created by sparse monocular Simultaneous Localisation and Mapping (SLAM) systems are useful for camera tracking, dense 3D reconstructions may be desired for many robotic tasks. Solutions involving depth cameras are limited in range and to indoor spaces, and dense reconstruction systems based on minimising the photometric error between frames are typically poorly constrained and suffer from scale ambiguity. To address these issues, we propose a 3D reconstruction system that leverages the output of a Convolutional Neural Network (CNN) to produce fully dense depth maps for keyframes that include metric scale.**

Our system, DeepFusion, is capable of producing real-time dense reconstructions on a GPU. It fuses the output of a semi-dense multiview stereo algorithm with the depth and gradient predictions of a CNN in a probabilistic fashion, using learned uncertainties produced by the network. While the network only needs to be run once per keyframe, we are able to optimise for the depth map with each new frame so as to constantly make use of new geometric constraints. Based on its performance on synthetic and real world datasets, we demonstrate that DeepFusion is capable of performing at least as well as other comparable systems.

Fig. 1: Fusing the depth predictions of a semi-dense multi-view stereo system with the depth and gradient predictions of a CNN allows for the creation of fully dense depth maps at scale. The above projected keyframe depth map was created by DeepFusion from only pose estimates and RGB images.

I. INTRODUCTION

One of the goals of Structure-from-Motion (SfM) and Visual Simultaneous Localisation and Mapping (SLAM) systems is the incremental creation of 3D scene reconstructions from moving cameras. Sparse SLAM systems such as MonoSLAM [1], PTAM [2] and ORB-SLAM [3] create sparse 3D maps of keypoints. While these features are useful for camera tracking, dense reconstructions may be preferred for safe robotic navigation, augmented reality and manipulation tasks. Dense reconstruction systems, such as DTAM [4] and REMODE [5], typically create dense scene representations by optimising for the depth values that minimise the photometric error over several frames. Unfortunately, depth estimation by minimising the photometric error is not well-constrained due the the presence of occlusions, and homogeneous or repeated texture. To combat this, dense systems often use a regulariser based on planar ([6]–[10]) or smoothness assumptions ([4], [5]). LSD-SLAM [11] tackles this issue by reconstructing only those points that have a strong image gradient, but can therefore only produce semi-dense reconstructions. Monocular reconstruction systems also suffer from an inherent scale ambiguity, as it is not possible to determine a camera's translation from image correspondences alone. ~~It is possible to address this scale ambiguity by fusing vision based measurements with readings from an inertial measurement unit (IMU); however, in situations with low accelerations (for example, when the camera is still), scale becomes practically unobservable when using low grade IMUs. One option to resolve these issues in monocular dense reconstruction is through the use of a depth camera (the approach taken by [12] and [13]), but depth cameras have limited range, consume more power, and do not typically work outdoors or in strong sunlight.~~

稠密重建多基于共面和平滑假设

LSD_SLAM实现了半稠密重建，但是基于强烈的图像梯度假设

这两点似乎每篇paper都要讲一遍

Recently, with the rapid growth of deep learning in computer vision, several data-driven approaches to dense reconstruction have been proposed. Presumably, the networks used in these systems are able to make predictions about the structure of the scene by leveraging some learned knowledge about observed objects and their spatial relationships. While many of these approaches advocate a completely end-to-end framework (for example, [14]–[19]), there has been some work demonstrating the benefit of combining both geometric constraints and learned priors. As shown in [20], geometric-based systems perform best on areas of high image gradients (usually on the edges of objects) but struggle with interior areas of low texture, whereas learning-based systems typically do reasonably well on interior points but blur the edges of objects. Despite the evidence to their complementary

The authors are with the Dyson Robotics Laboratory, Imperial College London, UK. Corresponding author: Tristan Laidlow, t.laidlow15@imperial.ac.uk

Research presented in this paper has been supported by Dyson Technology Ltd.

先用几何方法获得稀疏地图，将它输入CNN，得到稠密地图。

nature, however, the best approach to combining learning and geometry remains an open problem. For example, in [21], the authors create a dense monocular SLAM system based on DTAM, but use surface normal predictions from a CNN as a strong prior on the depth map rather than use a smoothness regulariser. Other approaches use geometric reconstructions as an input to a network and then either "fill in" any missing depth data [22] or refine them [23].

One problem with using a network as the final stage in a reconstruction pipeline is that an expensive network pass must be computed every time the underlying geometric information is updated, which may be unacceptably slow for real-time incremental systems. For this reason, a number of approaches that take network depth predictions and refine them with geometric constraints have been proposed. In [20], the authors compute a network depth prediction for each keyframe and update a semi-dense multiview stereo depth map with each new frame. The two depth estimates are then interpolated based on a set of tunable weights related to the image structure. Another approach [24] predicts surface normals and occlusion boundaries for each keyframe image and then attempts to fill in missing values in the output of a depth camera. Unfortunately, the solve time is too slow to be used on incremental SLAM systems. In [25], a CRF is used to refine the regression results. In [26], which greatly inspired our work, the authors were able to create accurate dense reconstructions using a fully-connected CRF with network predicted uncertainties from a sparse set of 3D points generated by a monocular SLAM system. Since they use the output of a monocular system to constrain the dense reconstructions, however, the resulting depth maps are ambiguous in scale. In our work, we use the idea of maintaining global consistency by linking neighbouring pixels in our reconstruction through depth gradient predictions and extend it to include the estimation of absolute scale. Perhaps the work that is most comparable to ours is CNN-SLAM [27], which uses a network to predict an at-scale depth map for each keyframe and then refines it through small baseline stereo constraints in real-time. The refinement in CNN-SLAM, however, is done on a per-pixel basis and therefore does not preserve global consistency.

In this paper, we propose DeepFusion, a 3D reconstruction system that is capable of producing dense depth maps at scale in real-time from RGB images and scale-ambiguous poses provided by a monocular SLAM system. We use network predicted depth gradients as a constraint on neighbouring pixels to ensure global consistency in our reconstructions, and learned uncertainties to fuse the different modalities in a probabilistic fashion. Please see Figure 1 for an example keyframe reconstruction.

II. METHOD

In this section, we describe our system for producing dense reconstructions at scale in real-time. Please see Figure 2 for an overview of the DeepFusion framework.

DeepFusion represents the observed geometry with a series of keyframe depth maps. With each new RGB image, the system obtains the pose from a monocular SLAM system (ORB-SLAM2 [3] in our implementation) and then updates the semi-dense depth estimates for the active keyframe using the method described in [28]. If the camera has translated more than λ_{trans} or had fewer than $\lambda_{inliers}$ inliers in the semi-dense estimation, a new keyframe is created.

先产生半稠密地图

To maintain a high frame rate, our network outputs are only generated once per keyframe. Using a CNN, we predict the log-depth, log-depth gradients and associated uncertainties from the new keyframe image. Like [26] and [29], we predict log-depths instead of depths or inverse depths because it is numerically better for network prediction (negative values are meaningful) and it has the convenient property that the difference between two log-depths (the gradient of the log-depth image) is the ratio of two depths, which is scale-invariant. We also choose to predict log-depth gradients in both the x- and y-directions on the image plane rather than surface normals in order to maintain the linearity of the optimisation problem, as this avoids the need for performing dot product and normalisation operations. Single-view depth prediction is a highly under-constrained problem, and in practice it seems easier for the network to make accurate predictions about fine-grained local geometry rather than absolute per-pixel depth. For this reason, we predict the absolute log-depth values and the log-depth gradients separately as we want separate uncertainties to reflect the difference in the network's ability at these two different tasks.

If a new keyframe is not created, then the current semi-dense depth map and network outputs are fused to update the current depth map.

A. Network Architecture

For our network, we use a U-Net [30] style architecture with the same dimensions as the one used in [31], except that we add three more identical decoders to predict log-depth uncertainties, log-depth gradients, and log-depth gradient uncertainties in addition to just the log-depths. All inputs and outputs have a resolution of 256x192.

In order to fuse the outputs of the network with the estimates coming from the semi-dense multiview stereo system, we require an uncertainty associated with each pixel in each of the log-depth and gradient images. To obtain this, we use the method described in [32] to have the network learn to predict both a mean and a variance with a maximum likelihood cost function:

$$\mathcal{L}_{\mathrm{NN}}(\boldsymbol{\theta}) = \sum_i \frac{\left(y_i - f_{\boldsymbol{\theta},i}(\mathbf{x})\right)^2}{\sigma_{\boldsymbol{\theta},i}(\mathbf{x})^2} + \log(\sigma_{\boldsymbol{\theta},i}(\mathbf{x})^2), \quad (1)$$

网络的输出是均值和方差。

where $\boldsymbol{\theta}$ is the set of network weights, $\mathbf{x}$ is the set of input pixels, y_i is the ground truth for pixel i, and $f_{\boldsymbol{\theta},i}(\mathbf{x})$ and $\sigma_{\boldsymbol{\theta},i}(\mathbf{x})^2$ are the network's predictions for the mean and variance, respectively. The total loss is the sum of this loss function for each of the output images.

Like [31], we train our network on the SceneNet RGB-D dataset [33], a dataset of 5 million rendered indoor scenes.

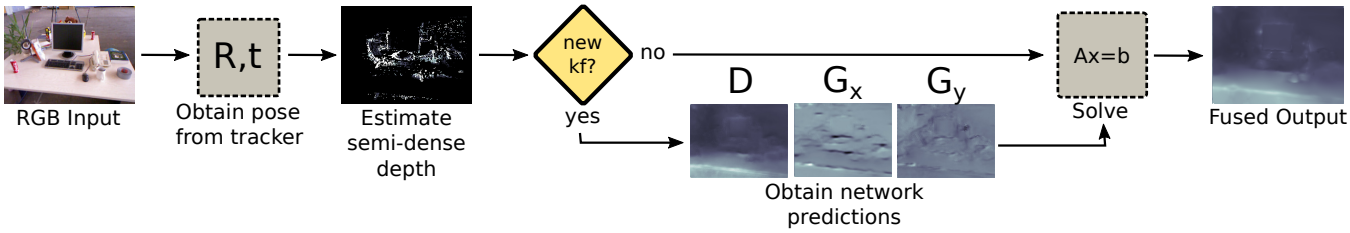

Fig. 2: The DeepFusion framework.

We train our network to predict log-depths that have been normalised by the focal length of the SceneNet camera. In the same manner as CNN-SLAM [27], we scale the log-depth predictions of the network by the focal length of the camera used at test time so we can recover absolute scale for images captured by cameras with different intrinsics. Since all predictions are done in logspace, the gradients (which represent depth ratios) and uncertainties do not need to be scaled.

B. Semi-Dense Estimation

For our semi-dense multiview stereo component, we implement the depth estimation method from [28]. For each pixel, $\mathbf{x}_i$, in the keyframe where there is sufficient texture, a search is made along the epipolar line for the depth value, $\mathbf{d}_{\text{semi},i}$ that minimises the sum of squared differences for five equally spaced points. If there is a current depth estimate for that pixel, the search is conducted over the interval $\mathbf{d}_{\text{semi},i} \pm 2\sigma_{\text{semi},i}$. Otherwise, the search is conducted over the entire epipolar line. Given a pixel in the keyframe, $\mathbf{x}_i$, and the estimated poses of the keyframe ($\boldsymbol{T}_{WC_0}$) and reference frame ($\boldsymbol{T}_{WC_1}$), the photometric error is given by:

$$e_i = I_1\left(\boldsymbol{\pi}\left(\mathbf{K}\,\boldsymbol{T}_{WC_1}^{-1}\,\boldsymbol{T}_{WC_0}\,\boldsymbol{\rho}(\mathbf{x}_i, \mathbf{d}_{\text{semi},i})\right)\right) - I_0(\mathbf{x}_i),$$

where $I_*(\cdot)$ is a scalar function that returns the intensity value of a given pixel, $\boldsymbol{\pi}(\cdot)$ is the projection and dehomogenisation function, $\boldsymbol{\rho}(\mathbf{x}_i, \mathbf{d}_{\text{semi},i})$ is the back-projection function that returns a homogeneous 3D point for pixel $\mathbf{x}_i$ with a depth $\mathbf{d}_{\text{semi},i}$, and $\mathbf{K}$ is the camera intrinsics matrix.

We approximate the Jacobian of the error function, $\mathbf{J}$, with finite differences:

$$\mathbf{J} \approx \frac{1}{\Delta \mathbf{d}_{\text{semi},i}} \Delta \mathbf{e},$$

where $\mathbf{e}$ is the 5x1 vector containing the photometric error associated with each of the five points. When searching for the minimum value along the epipolar line, even sized steps of 1 pixel length are taken. Once the minimum is found, we interpolate between two steps to find the optimal depth at sub-pixel resolution. The difference in the photometric error at the two endpoints of the interpolation is $\Delta \mathbf{e}$, and the difference between the depths at those points is $\Delta \mathbf{d}_{\text{semi},i}$.

We then approximate the uncertainty of each semi-dense measurement by:

$$\sigma_i^2 = (\mathbf{J}^T \mathbf{J})^{-1}. \tag{2}$$

The semi-dense depth estimates and uncertainties are then converted into logspace to match the network outputs.

C. Optimisation

To update the current depth prediction, we minimise the following cost function consisting of three terms with each new frame:

$$c(\mathbf{d}, s) = c_{\text{semi}}(\mathbf{d}, s) + c_{\text{net}}(\mathbf{d}) + c_{\text{grad}}(\mathbf{d}),$$

where $\mathbf{d}$ is the set of log-depth values to be estimated and s is the scale correction factor.

The semi-dense cost term imposes a unary constraint over the set of pixels where valid semi-dense log-depth values have been estimated:

$$c_{\text{semi}}(\mathbf{d}, s) = \mathbf{r}_{\text{semi}}^T(\mathbf{d}, s)\mathbf{R}_{\text{semi}}^{-1}\mathbf{r}_{\text{semi}}(\mathbf{d}, s),$$
$$\mathbf{r}_{\text{semi},i}(\mathbf{d}_i, s) = \ln \mathbf{d}_i - \ln s - \ln \mathbf{d}_{\text{semi},i},$$

where $\mathbf{R}_{\text{semi},i} = \sigma_i^2$, the uncertainty estimated by the approximation given in (2) and $\ln \mathbf{d}_{\text{semi},i}$ is the scale-ambiguous log-depth predicted by the semi-dense system for pixel i. Since the semi-dense log-depth estimates are calculated based on the poses provided by a monocular SLAM system, they have an arbitrary scale. Because the depth map we wish to estimate, $\mathbf{d}$, is to scale, we also need to solve for a scale correction factor, s. With the fully dense per-pixel cost term c_{net}, this scale correction factor becomes observable.

The network depth cost term imposes an additional unary constraint over all pixels of the fused depth map:

$$c_{\text{net}}(\mathbf{d}) = \mathbf{r}_{\text{net}}^T(\mathbf{d})\mathbf{R}_{\text{net}}^{-1}\mathbf{r}_{\text{net}}(\mathbf{d}),$$
$$\mathbf{r}_{\text{net},i}(\mathbf{d}_i) = \ln \mathbf{d}_i - \ln \mathbf{d}_{\text{net},i},$$

where $\mathbf{R}_{\text{net},i} = \sigma_{\boldsymbol{\theta},\text{depth},i}^2(\mathbf{x})$, the uncertainty predicted by the network (see (1)) and $\ln \mathbf{d}_{\text{net},i}$ is the log-depth prediction for pixel i made by the network. While the network may have a high uncertainty about the absolute depth at any pixel (as the uncertainty is conditioned on the image, it will depend on how close the image is to the training data), this cost term provides a weak prior on the absolute scale of the scene and allows us to estimate the scale correction factor s.

In order to maintain global consistency while fusing together the semi-dense and network depth values, we include an additional cost term that imposes pairwise constraints between a given pixel and each of its four neighbours:

$$\begin{aligned} c_{\text{grad}}(\mathbf{d}) &= \mathbf{r}_{\text{grad,x}}^T(\mathbf{d})\mathbf{R}_{\text{grad,x}}^{-1}\mathbf{r}_{\text{grad,x}}(\mathbf{d}) \\ &\quad + \mathbf{r}_{\text{grad,y}}^T(\mathbf{d})\mathbf{R}_{\text{grad,y}}^{-1}\mathbf{r}_{\text{grad,y}}(\mathbf{d}) \\ \mathbf{r}_{\text{grad,x},i}(\mathbf{d}_{i+1}, \mathbf{d}_i) &= \ln \mathbf{d}_{i+1} - \ln \mathbf{d}_i - \mathbf{g}_{\text{x},i} \\ \mathbf{r}_{\text{grad,y},i}(\mathbf{d}_{i+\text{W}}, \mathbf{d}_i) &= \ln \mathbf{d}_{i+\text{W}} - \ln \mathbf{d}_i - \mathbf{g}_{\text{y},i} \end{aligned}$$

where W is the width of the image, $\mathbf{g}_{\mathrm{x},i}$ and $\mathbf{g}_{\mathrm{y},i}$ are the log-depth gradients in the x- and y-directions predicted by the network, and $\mathbf{R}_{\mathrm{grad,x},i} = \sigma^2_{\boldsymbol{\theta},\mathrm{grad,x},i}(\mathbf{x})$ and $\mathbf{R}_{\mathrm{grad,y},i} = \sigma^2_{\boldsymbol{\theta},\mathrm{grad,y},i}(\mathbf{x})$, the associated predicted uncertainties (see (1)).

The semi-dense depth estimates can be very noisy and the network predictions can have extreme outliers which have a significant impact on the final reconstruction. For this reason, we use the Huber loss function on each of the cost terms.

We solve the system using the Opt [34] optimisation framework. With Opt, we define an energy function for each term in our cost function which are then automatically compiled into GPU optimisation kernels. We compute 10 Gauss-Newton iterations, alternating between solving for the depth and scale. With this setup, we are capable of solving the system for each new frame in real-time.

III. Experimental Results

A. Qualitative Results

Figure 3 shows some qualitative results from selected keyframes on the ICL-NUIM [35] and TUM RGB-D [36] datasets. Comparing the network depth predictions with the final fused depth maps shows that including geometric constraints from the semi-dense depth estimation and pairwise pixel constraints from the gradient predictions produces depth maps that are more globally consistent and have fewer blurring artifacts.

Figure 4 shows the network output for sample images in the SceneNet RGB-D dataset [33]. The uncertainties associated with the gradient images are clearly largest on areas of high image gradient suggesting that the network has learned that these regions tend to correspond with depth discontinuities or other rapid changes in the depth gradient.

B. Reconstruction Evaluation

We evaluate our reconstruction pipeline by comparing it to the results obtained by CNN-SLAM [27], a state-of-the-art system that fuses together network predictions and geometric constraints to produce fully dense depth maps. Following the evaluation procedure of CNN-SLAM, we also compare our results to the depth maps produced by a sparse feature-based monocular system (ORB-SLAM2 [3]), a semi-dense geometric system (LSD-SLAM [11]), a fully dense reconstruction method (REMODE [5]), and a pure deep learning approach (Laina, et al. [16]).

As there was no open-source version of CNN-SLAM available at the time of writing, we evaluate DeepFusion on the same sequences used in [27] and compare with their reported results. The sequences used for the comparison come from two different datasets: the synthetic ICL-NUIM RGB-D dataset [35] and the real world TUM RGB-D SLAM dataset [36]. The ICL-NUIM dataset provides rendered depth maps as a ground truth comparison and the TUM RGB-D dataset approximates this with Kinect depth camera images.

As proposed by [27], we measure the percentage of estimated depth values that are within 10% of the corresponding ground truth depth values in order to evaluate both the reconstruction accuracy and density.

The results are presented in Table I. While DeepFusion and CNN-SLAM have approximately the same performance overall, this performance is not evenly distributed over the two datasets. DeepFusion performs the best on four out of the six ICL-NUIM sequences, whereas CNN-SLAM performs the best on two out of the three TUM RGB-D sequences. The reason for this most likely has to do with the training data used by the two systems. Our network is trained on the synthetic SceneNet dataset [33], whereas CNN-SLAM uses the network described in [16] which is trained on the Kinect-captured NYUv2 [37]. In addition, two of the TUM RGB-D sequences used in the comparison consist of the camera moving over flat planes covered in posters, with seemingly little semantic information for the networks to leverage. These results speak to the importance of keeping the network's training data as close as possible to the domain in which the system will be deployed. While both DeepFusion and CNN-SLAM clearly outperform the geometry-only systems, the learning-based method proposed by Laina, et al. [16] also does well, performing the best on two of the nine sequences. This demonstrates the power of learning-based approaches, but ultimately the systems that can make use of both modalities perform the best overall.

C. Scale Optimisation Evaluation

While we include the network predicted log-depth values in our optimisation problem to solve for the absolute scale of the reconstruction, there are alternative methods. For instance, in [21], the authors first predict a scale-ambiguous reconstruction and then scale their reconstruction by finding a least-squares fit with a network predicted depth map. The advantage of our method is that we are able to use the relative uncertainties between the semi-dense estimates and network predictions when performing our fusion.

To demonstrate that our method produces better results, we implemented a version of DeepFusion that optimises for a scale-ambiguous depth map using only the semi-dense and gradient terms of the cost function and then finds a least-squares fit with the network predicted depth and measured its performance on sequences used for our reconstruction evaluation. The results are presented in Table II. In seven of the nine sequences, our method outperforms the least-squares post-processing method.

D. Global Consistency Evaluation

One of the primary differences between DeepFusion and CNN-SLAM [27] is that DeepFusion includes a pairwise constraint on neighbouring pixels in order to enforce global consistency on the optimised depth maps.

To show that including these constraints does in fact help produce more accurate reconstructions, we implement a version of DeepFusion that optimises for a depth map with absolute scale using only the network predicted log-depths, the semi-dense log-depth estimates, and their associated uncertainties. We evaluate the performance of this version on the sequences used in the reconstruction evaluation. The results are presented in Table III. In seven of the nine

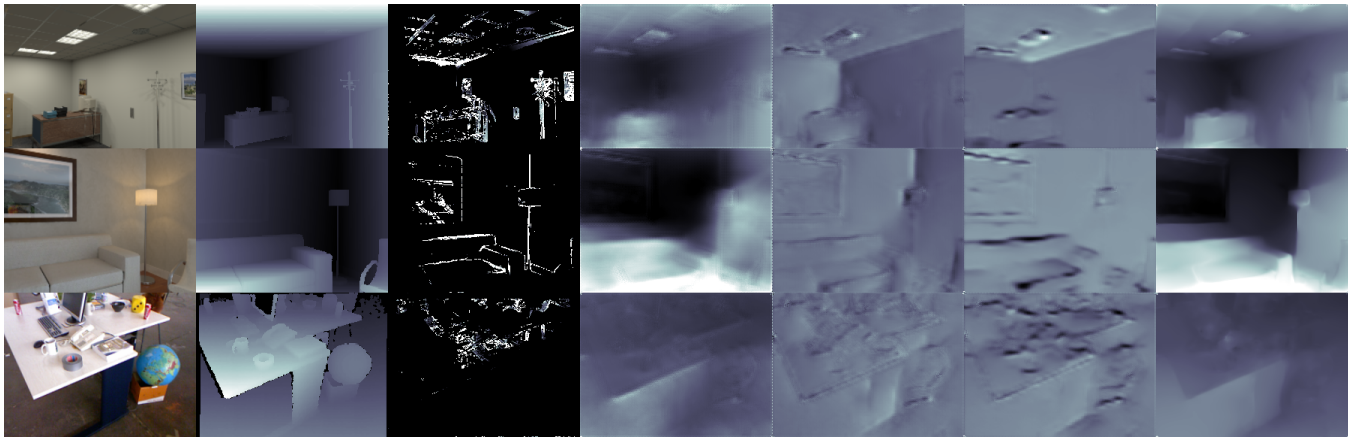

Fig. 3: Qualitative results for selected keyframes on the ICL-NUIM Office2 (top), ICL-NUIM LivingRoom1 (middle) and TUM RGB-D fr2_desk (bottom) sequences. From left to right: input image, ground truth depth, semi-dense depth estimate, network depth prediction, network depth gradient prediction in the x-direction, network depth gradient prediction in the y-direction, and the optimised depth map.

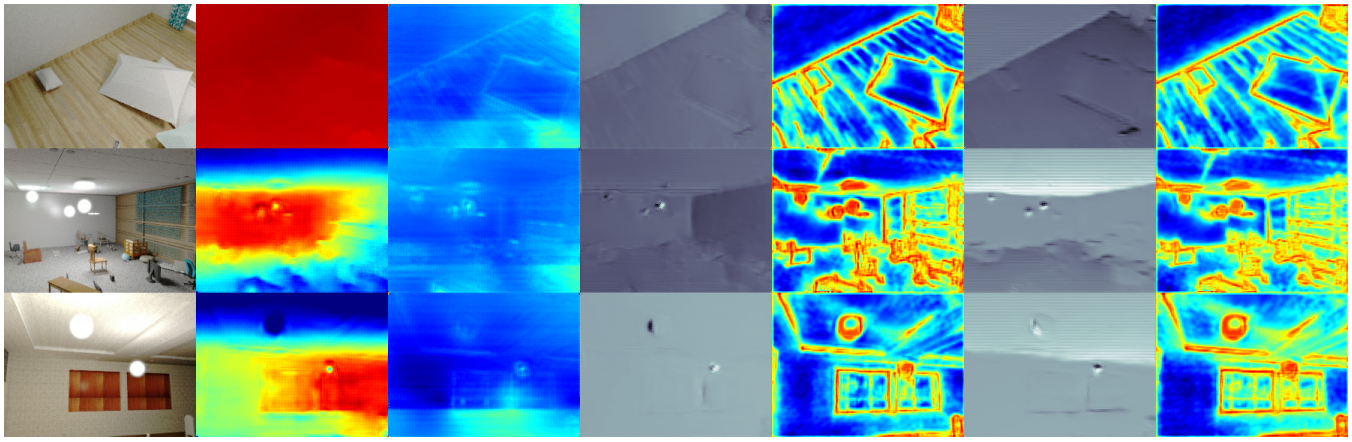

Fig. 4: Example network predictions on sample images from the SceneNet RGB-D dataset. From left to right: input image, log-depth prediction, log-depth uncertainty prediction, log-depth gradient prediction in the x-direction, log-depth gradient in the x-direction uncertainty prediction, log-depth gradient prediction in the y-direction, log-depth gradient in the y-direction uncertainty prediction.

TABLE I: Comparison of reconstruction accuracy in terms of percentage of correct depth values (within 10% of ground truth) on ICL-NUIM and TUM RGB-D datasets (TUM/seq1: *fr3/long_office_household*, TUM/seq2: *fr3_nostructure_texture_near_withloop*, TUM/seq3: *fr3/structure_texture_far*). LSD-SLAM (BS) is LSD-SLAM bootstrapped with a ground truth depth map, and REMODE uses LSD-SLAM (BS) poses and keyframes.

Sequence	**DeepFusion**	**CNN-SLAM** [27]	**LSD-SLAM (BS)** [11]	**LSD-SLAM** [11]	**ORB-SLAM** [3]	**Laina** [16]	**REMODE** [5]
ICL/office0	**21.090**	19.410	0.603	0.335	0.018	17.194	4.479
ICL/office1	**37.420**	29.150	4.759	0.038	0.023	20.838	3.132
ICL/office2	30.180	**37.226**	1.435	0.078	0.040	30.639	16.708
ICL/living0	**24.223**	12.840	1.443	0.360	0.027	15.008	4.479
ICL/living1	**14.001**	13.038	3.030	0.057	0.021	11.449	2.427
ICL/living2	25.235	26.560	1.807	0.167	0.014	**33.010**	8.681
TUM/seq1	8.069	12.477	3.797	0.086	0.031	**12.982**	9.548
TUM/seq2	14.774	**24.077**	3.966	0.882	0.059	15.412	12.651
TUM/seq3	27.200	**27.396**	6.449	0.035	0.027	9.450	6.739
Avg.	**22.466**	22.464	3.032	0.226	0.029	18.452	7.649

TABLE II: Comparison of scale estimation methods in DeepFusion in terms of percentage of correct depth values (within 10% of ground truth) on ICL-NUIM and TUM RGB-D datasets (TUM/seq1: *fr3/long_office_household*, TUM/seq2: *fr3_nostructure_texture_near_withloop*, TUM/seq3: *fr3/structure_texture_far*). "Least Squares for Scale Estimation" shows the results when using least squares to align a scale-ambiguous estimation with the network depth prediction to estimate a scaled depth map for each keyframe.

Sequence	DeepFusion	Least Squares for Scale Estimation
ICL/office0	**21.090**	18.135
ICL/office1	**37.420**	26.415
ICL/office2	30.180	**31.359**
ICL/living0	**24.223**	23.861
ICL/living1	**14.001**	10.372
ICL/living2	**25.235**	22.082
TUM/seq1	8.069	**9.690**
TUM/seq2	**14.774**	14.490
TUM/seq3	**27.200**	24.047

TABLE III: Analysis on the importance of pairwise constraints on reconstruction accuracy in terms of percentage of correct depth values (within 10% of ground truth) on ICL-NUIM and TUM RGB-D datasets (TUM/seq1: *fr3/long_office_household*, TUM/seq2: *fr3_nostructure_texture_near_withloop*, TUM/seq3: *fr3/structure_texture_far*). "No Pairwise Constraints" shows the results when fusing together only the semi-dense and network log-depth values.

Sequence	DeepFusion	No Pairwise Constraints
ICL/office0	**21.090**	16.641
ICL/office1	**37.420**	24.633
ICL/office2	30.180	**30.899**
ICL/living0	**24.223**	21.643
ICL/living1	**14.001**	12.774
ICL/living2	**25.235**	21.772
TUM/seq1	8.069	**9.469**
TUM/seq2	**14.774**	14.187
TUM/seq3	**27.200**	23.584

sequences, our method outperforms the method that does not enforce global consistency.

E. Timing Evaluation

To demonstrate the real-time capability of our system, we show the approximate runtimes of each of our components in Table IV. These runtimes were based on our implementation using an Intel Core i7-5820K CPU and a GeForce GTX 980 GPU.

TABLE IV: Approximate timing information for key components in the DeepFusion system.

	Semi-Dense	Optimisation	Network Prediction
Mean	16ms	33ms	45ms
Min	1ms	26ms	44ms
Max	43ms	47ms	47ms

IV. Conclusion

We have presented DeepFusion, a system capable of producing dense 3D reconstructions at scale in real-time. By formulating a cost function that includes per-pixel losses based on network depth predictions and sparse semi-dense depth estimates with pairwise constraints from network depth gradient predictions we are able to estimate both the shape of the observed scene and its absolute scale. By predicting both the per-pixel mean and variance, we are able to obtain uncertainties for all network outputs and fuse them together with the geometric constraints in a probabilistic fashion. Since DeepFusion only requires the network to be run once per keyframe, we are able to maintain real-time capability.

Through a series of experiments on synthetic and real datasets, we demonstrate that DeepFusion performs at least as well as other comparable systems. Furthermore, through a series of ablation studies, we demonstrate the value of estimating the scale of the depth maps by including the network depth output as a per-pixel constraint in the optimisation and using pairwise constraints to enforce global consistency.

In future work, we will attempt to improve DeepFusion by more thoroughly investigating the impact that certain design choices have on the performance of the algorithm. For example, as was noted in our comparison with CNN-SLAM, the choice of training data and its similarity to the test environment likely has a significant impact on the quality of the final reconstruction. Which training dataset to use and how to ensure that the predicted uncertainty reflects differences between training and testing environments are open questions.

Related to the network predictions and their associated uncertainties is the question of how to handle extreme outliers that may be produced by the network. In DeepFusion, this was handled by using a robust cost function (in particular, a Huber loss function) in the optimisation. Whether other cost functions that penalise extreme outliers less severely (such as Tukey) would result in better performance remains to be seen.

Finally, as with all other keyframe-based SLAM systems, there are many trade-offs that need to be considered when choosing keyframe selection criteria. How these trade-offs are impacted when a geometry-based depth estimation is coupled with a learning-based prediction should be investigated.

References

[1] A. J. Davison, N. D. Molton, I. Reid, and O. Stasse, "MonoSLAM: Real-Time Single Camera SLAM," *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 29, no. 6, pp. 1052–1067, 2007.

[2] G. Klein and D. W. Murray, "Parallel Tracking and Mapping on a Camera Phone," in *Proceedings of the International Symposium on Mixed and Augmented Reality (ISMAR)*, 2009.

[3] R. Mur-Artal and J. D. Tardós, "ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras," *IEEE Transactions on Robotics (T-RO)*, vol. 33, no. 5, pp. 1255–1262, 2017.

[4] R. A. Newcombe, S. Lovegrove, and A. J. Davison, "DTAM: Dense Tracking and Mapping in Real-Time," in *Proceedings of the International Conference on Computer Vision (ICCV)*, 2011.

[5] M. Pizzoli, C. Forster, and D. Scaramuzza, "REMODE: Probabilistic, monocular dense reconstruction in real time," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2014.

[6] A. Flint, D. W. Murray, and I. Reid, "Manhattan Scene Understanding Using Monocular, Stereo, and 3D Features," in *Proceedings of the International Conference on Computer Vision (ICCV)*, 2011.

[7] V. Pradeep, C. Rhemann, S. Izadi, C. Zach, M. Bleyer, and S. Bathiche, "MonoFusion: Real-time 3D reconstruction of small scenes with a single web camera," in *Proceedings of the International Symposium on Mixed and Augmented Reality (ISMAR)*, 2013, pp. 83–88.

[8] A. Concha, W. Hussain, L. Montano, and J. Civera, "Manhattan and piecewise-planar constraints for dense monocular mapping," in *Proceedings of Robotics: Science and Systems (RSS)*, 2014.

[9] A. Concha and J. Civera, "Using superpixels in monocular SLAM," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2014, pp. 365–372.

[10] ——, "DPPTAM: Dense Piecewise Planar Tracking and Mapping from a monocular sequence," in *Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2015.

[11] J. Engel, T. Schoeps, and D. Cremers, "LSD-SLAM: Large-scale direct monocular SLAM," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2014.

[12] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohli, J. Shotton, S. Hodges, and A. Fitzgibbon, "KinectFusion: Real-Time Dense Surface Mapping and Tracking," in *Proceedings of the International Symposium on Mixed and Augmented Reality (ISMAR)*, 2011.

[13] C. Kerl, J. Sturm, and D. Cremers, "Dense visual SLAM for RGB-D cameras," in *Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems (IROS)*, 2013.

[14] D. Eigen and R. Fergus, "Predicting Depth, Surface Normals and Semantic Labels with a Common Multi-Scale Convolutional Architecture," in *Proceedings of the International Conference on Computer Vision (ICCV)*, 2015.

[15] B. Ummenhofer, H. Zhou, J. Uhrig, N. Mayer, E. Ilg, A. Dosovitskiy, and T. Brox, "DeMoN: Depth and motion network for learning monocular stereo," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[16] I. Laina, C. Rupprecht, V. Belagiannis, F. Tombari, and N. Navab, "Deeper depth prediction with fully convolutional residual networks," in *Proceedings of the International Conference on 3D Vision (3DV)*, 2016.

[17] F. Liu, C. Shen, and G. Lin, "Deep Convolutional Neural Fields for Depth Estimation from a Single Image," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[18] 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 *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[19] T. Dharmasiri, A. Spek, and T. Drummond, "Joint prediction of depths, normals and surface curvature from RGB images using cnns," in *Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems (IROS)*, 2017.

[20] J. M. Fácil, A. Concha, L. Montesano, and J. Civera, "Single-view and multi-view depth fusion," *IEEE Robotics and Automation Letters*, vol. 2, no. 4, pp. 1994–2001, 2017.

[21] C. S. Weerasekera, Y. Latif, R. Garg, and I. Reid, "Dense monocular reconstruction using surface normals," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2017.

[22] N. Yang, R. Wang, J. Stückler, and D. Cremers, "Deep virtual stereo odometry: Leveraging deep depth prediction for monocular direct sparse odometry," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.

[23] H. Zhou, B. Ummenhofer, and T. Brox, "DeepTAM: Deep tracking and mapping," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.

[24] Y. Zhang and T. Funkhouser, "Deep depth completion of a single rgb-d image," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[25] X. Yin, X. Wang, X. Du, and Q. Chen, "Scale recovery for monocular visual odometry using depth estimated with deep convolutional neural fields," in *Proceedings of the International Conference on Computer Vision (ICCV)*, 2017.

[26] C. S. Weerasekera, T. Dharmasiri, R. Garg, T. Drummond, and I. Reid, "Just-in-time reconstruction: Inpainting sparse maps using single view depth predictors as priors," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2018.

[27] K. Tateno, F. Tombari, I. Laina, and N. Navab, "CNN-SLAM: Real-time dense monocular slam with learned depth prediction," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[28] J. Engel, J. Sturm, and D. Cremers, "Semi-dense visual odometry for a monocular camera," in *Proceedings of the International Conference on Computer Vision (ICCV)*, 2013.

[29] D. Eigen, C. Puhrsch, and R. Fergus, "Depth Map Prediction from a Single Image using a Multi-Scale Deep Network," in *Neural Information Processing Systems (NIPS)*, 2014.

[30] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," in *Proceedings of the International Conference on Medical Image Computing and Computer Assisted Intervention (MICCAI)*, 2015.

[31] M. Bloesch, J. Czarnowski, R. Clark, S. Leutenegger, and A. J. Davison, "CodeSLAM — learning a compact, optimisable representation for dense visual SLAM," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[32] A. Kendall and Y. Gal, "What uncertainties do we need in bayesian deep learning for computer vision?" in *Neural Information Processing Systems (NIPS)*, 2017.

[33] J. McCormac, A. Handa, S. Leutenegger, and A. J. Davison, "SceneNet RGB-D: Can 5M synthetic images beat generic ImageNet pre-training on indoor segmentation?" in *Proceedings of the International Conference on Computer Vision (ICCV)*, 2017.

[34] Z. DeVito, M. Mara, M. Zollöfer, G. Bernstein, C. Theobalt, P. Hanrahan, M. Fisher, and M. Nießner, "Opt: A domain specific language for non-linear least squares optimization in graphics and imaging," in *ACM Transactions on Graphics*, 2017.

[35] A. Handa, T. Whelan, J. B. McDonald, and A. J. Davison, "A Benchmark for RGB-D Visual Odometry, 3D Reconstruction and SLAM," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2014. [Online]. Available: http://www.doc.ic.ac.uk/ ahanda/VaFRIC/iclnuim.html

[36] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A Benchmark for the Evaluation of RGB-D SLAM Systems," in *Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems (IROS)*, 2012.

[37] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus, "Indoor segmentation and support inference from RGBD images," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2012.