Multi-View Stereo by Temporal Nonparametric Fusion

Yuxin Hou   Juho Kannala   Arno Solin
Department of Computer Science
Aalto University, Finland
firstname.lastname@aalto.fi

Abstract

We propose a novel idea for depth estimation from multi-view image-pose pairs, where the model has capability to leverage information from previous latent-space encodings of the scene. This model uses pairs of images and poses, which are passed through an encoder–decoder model for disparity estimation. The novelty lies in soft-constraining the bottleneck layer by a nonparametric Gaussian process prior. We propose a pose-kernel structure that encourages similar poses to have resembling latent spaces. The flexibility of the Gaussian process (GP) prior provides adapting memory for fusing information from previous views. We train the encoder–decoder and the GP hyperparameters jointly end-to-end. In addition to a batch method, we derive a lightweight estimation scheme that circumvents standard pitfalls in scaling Gaussian process inference, and demonstrate how our scheme can run in real-time on smart devices.

(a) Reference frames

(b) Multi-view depth-estimation w/o GP
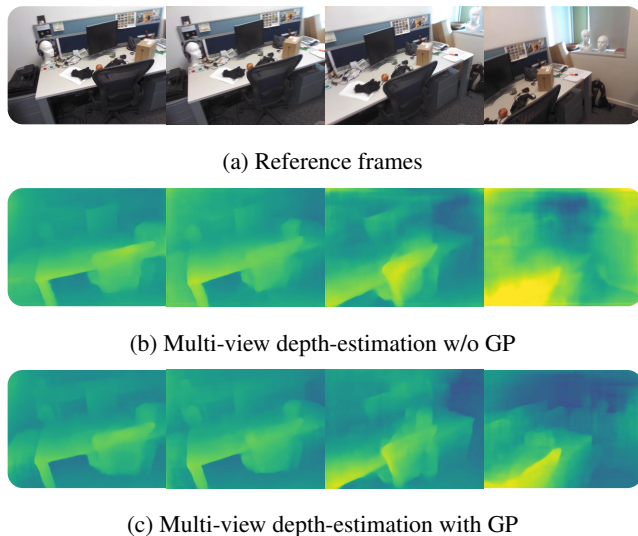

(c) Multi-view depth-estimation with GP

Figure 1. An example sequence of depth estimation results, where introducing information sharing in the latent space helps improving the depth maps by making them more stable and edges sharper.

1. Introduction

Multi-view stereo (MVS) refers to the problem of reconstructing 3D scene structure from multiple images with known camera poses and internal parameters. For example, estimation of depth maps from multiple video frames captured by a moving monocular video camera [34] is a variant of MVS when the motion is known. Other variants of the problem include depth estimation using conventional two-view stereo rigs [15] and image-based 3D modelling from image collections [8, 27]. MVS reconstructions have various applications. For instance, image-based 3D models can be used for measurement and visualization of large environments to aid design and planning [1], and depth estimation from stereo rigs or monocular videos benefits perception and simultaneous localization and mapping (SLAM) in the context of autonomous machines.

In this paper, we focus on depth map estimation for video frames captured by a monocular camera, whose motion is unconstrained but known. In practice, the motion could

be estimated using visual-inertial odometry techniques (see, *e.g.*, [29]), which are capable of providing high-precision camera poses in real-time with very small drift and are also commonly available in standard mobile platforms (*e.g.*, AR-Core on Android and ARKit on iOS).

Depth estimation from multiple video frames under varying and arbitrary motion is more challenging than depth estimation using a rigid two-view stereo rig, but there can be potential benefits in using a moving monocular camera instead of a fixed rig. Firstly, in small mobile devices the baseline between the two cameras of the rig can not be large and this limits the range of depth measurements. With a moving monocular camera the motion usually provides a larger baseline than the size of the device and thus measurement accuracy for distant regions can be improved. Secondly, when the camera is translating and rotating in a given space, it typically observes the same scene regions from multiple continuously varying viewpoints, and it would be beneficial

to be able to effectively fuse all this information for more robust and stable depth estimation.

In this work, we propose a new approach that combines a disparity estimation network, which has an encoder–decoder architecture and plane-sweep cost volume input as in [34], and a Gaussian process (GP, [24]) prior, which soft-constrains the bottleneck layer of the network to fuse information from video frames having similar poses. This is achieved by proposing a pose-kernel structure which encourages similar poses to have resembling latent space representations. The motivation behind the proposed approach is to efficiently improve fusion of information from overlapping views independently of their separation in time. That is, our pose-kernel can implicitly fuse information from all frames, which have overlapping fields of view, and without making the prediction of individual depth maps more time-consuming or without spending additional effort in the cost volume computation. In contrast to hard and heuristic view selection rules that are often applied in similar context [34, 36] our approach allows soft fusion of information via the latent representation. Our approach can be applied either in batch mode, where the fused result utilizes all available frames, or in online mode, where only previous frames affect the prediction of the current frame.

The contributions of this paper are as follows. *(i)* We propose a novel approach for multi-view stereo that passes information from previously reconstructed depth maps through a probabilistic prior in the latent space; *(ii)* For the non-parametric latent-space prior we propose a pose-kernel approach for encoding prior knowledge about effects of the relative camera pose between observation frames; *(iii)* We show that the CNN encoder–decoder structure and the GP hyperparameters can be trained jointly; *(iv)* We extend our method to an online scheme capable of running in real-time in smartphones/tablets.

To our knowledge this is the first paper to utilize GP priors for multi-view information fusion and also the first attempt at scalable online MVS on smartdevices.

2. Related work

MVS approaches can be categorized based on their output representations as follows: *(a)* volumetric reconstruction methods [18, 17, 13], *(b)* point cloud reconstruction methods [9, 37], and *(c)* depth map based methods [36]. In many cases, point cloud and depth map representations are finally converted to a triangular surface mesh for refinement [9, 19]. Volumetric voxel based approaches have shown good performance with small objects but are difficult to apply for large scenes due to their high memory load. Point cloud based approaches provide accurate reconstructions for textured scenes and objects but scenes with textureless surfaces and repeating patterns are challenging. In this work, we focus on multi-view depth estimation since

depth map based approaches are flexible and suitable for most use cases.

There has recently been plenty of progress in learning-based depth estimation approaches. Inspired by classical MVS methods [3], most attempts on learned MVS use plane-sweeping approaches to first compute a matching cost volume from nearby images and then regard depth estimation as a regression or multi-class classification problem, which is addressed by deep neural networks [11, 34, 36]. DeepTAM [39] computes the sum of absolute difference of patches between warped image pairs and use an adaptive narrow band strategy to increase the density of sampled planes. DeepMVS [11] proposed a patch matching network to extract features to aid in the comparison of patches. For feature aggregation, it considers both an intra-volume feature aggregation network and inter-volume aggregation network. MVDepthNet [34] computes the absolute difference directly without a supporting window to generate the cost volume, as the pixel-wise cost matching enable the volume to preserve detail information. MVSNet [36] proposes a variance-based cost metric and employ a 3D CNN to obtain a smooth cost volume automatically. DPSNet [12] concatenate warped features and use a series of 3D convolutions to learn the cost volume generation.

It is important to note that none of the aforementioned learning based MVS approaches has been demonstrated on a mobile platform. Indeed, most of the methods are heavy and it takes several seconds or even more to evaluate a single depth map with a powerful desktop GPU [11, 36]. The most light-weight model is [34], and therefore we use it as a baseline upon which we add our complementary contributions. The monocular depth estimation system in [34] uses a view selection rule, which selects frames that have enough angle or translation difference and then uses the selected frames for computing the cost volume. However, this kind of view selection can not use information from similar views in the more distant past, since the future motion is unknown and all past frames can not be stored. In contrast, our approach allows to utilize all past information in a computationally efficient manner. Also, our contribution is not competing with the various network architectures proposed recently [11, 39, 36, 34, 10] but complementary: The temporal coupling of latent representations has not been proposed earlier and could be combined also with other network architectures than [34], which we use in our experiments.

Another area of related work is depth map fusion which aims to integrate multiple depth maps into a unified scene representation and needs to deal with inconsistencies and redundancies in the process. For example, [21] defines three types of visibility relationships between predicted depth maps, determining the validity of estimation by detecting occlusions and free-space violations. Also, volumetric

approaches are widely used for fusion and reconstruction [22, 23]. Again, our method is complementary: it shares information implicitly in the latent space, and can be combined with a depth map fusion post-processing stage.

Finally, regarding the technical and methodological aspects of our work, we combine both deep neural networks and Gaussian process (GP) models. GPs are a probabilistic machine learning paradigm for encoding flexible priors over functions [24]. They have not been much used in this area of geometric computer vision. Though, GPs have been used in other latent variable modelling tasks in vision, where uncertainty quantification [14] plays a crucial role—including variational autoencoders with GP priors [4, 2] and GP based latent variable models for multi-view and view-invariant facial expression recognition [5, 6]. In [2] GPs are applied to face image modelling, where the GP kernel accounts for the pose, and in [33] they are used for 3D people tracking. The motivation for our work is in recent advances in real-time inference using GPs [26, 30] that make them applicable to online inference in smartphones.

3. Methods

Our multi-view stereo approach consist of two orthogonal parts. The first (vertical data flow in Fig. 2) is an CNN-powered MVS approach where the input frames are warped into a cost volume and then passed through an encoder–decoder model to produce the disparity (reciprocal of depth) map. The second part (horizontal data flow in Fig. 2) is for coupling each of the independent disparity prediction tasks, by passing information about the latent space (bottleneck layer encodings) over the camera trajectory. We will first go through the setup in the former (Sec. 3.1), and then focus on the latter (Secs. 3.2–3.4).

3.1. Network architecture

For the encoder and decoder, we build upon the straightforward model in [34]. Our framework only includes one encoder–decoder without change of architecture, so we can compare the results directly to check the impacts of Gaussian process prior. The output of the encoder–decoder is the continuous inverse depth (disparity) prediction. For each image-pose pair, we compute a cost volume of size $D \times H \times W$ and concatenate the reference RGB image as the input for the encoder. In this paper, we use an image size of 320×256, and $D = 64$ depth planes uniformly sampled in inverse depth from 0.5 m to 50 m. To compute the cost volume, we warp the neighbour frame via the fronto-parallel planes at fixed depths to the reference frame using the planar homography:

$$\mathbf{H} = \mathbf{K} \left(\mathbf{R} + \mathbf{t} \begin{pmatrix} 0 & 0 & \frac{1}{d_i} \end{pmatrix} \right) \mathbf{K}^{-1}, \tag{1}$$

where $\mathbf{K}$ is the known intrinsics matrix and the relative pose $(\mathbf{R}, \mathbf{t})$ is given in terms of a rotation matrix and translation
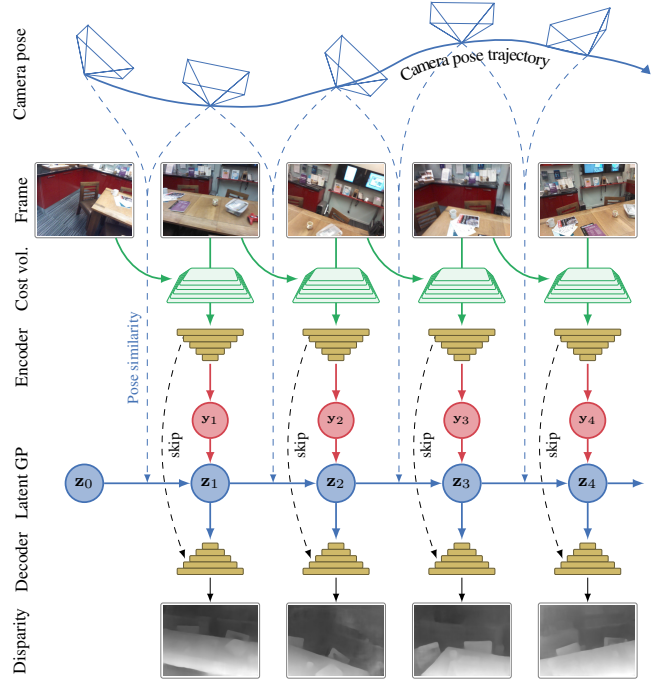

Figure 2. Illustrative sketch of our MVS approach. The camera poses and input frames are illustrated in the top rows. The current and previous (or a sequence of previous) frames are used for composing a cost volume, which is then passed through and encoder network. The novelty in our method is in doing Gaussian process inference on the latent-space encodings such that the GP prior is defined to be smooth in pose-difference. The GP prediction is finally passed through a decoder network which outputs disparity maps (bottom). This is the logic of the online variant of our method (the latent space graph is a directed graph / Markov chain). The batch variant could be illustrated in similar fashion, but with links between all latent nodes $\mathbf{z}_i$.

vector with respect to the neighbour frame. d_i denotes the depth value of the ith virtual plane. The absolute intensity difference between the warped neighbour frame and the reference frame is calculated as the cost for each pixel at every depth plane: $V(d_i) = \sum_{R,G,B} \tilde{I}_{d_i} - I_{\text{r}}$, where $\tilde{I}_{d_i}$ denotes the warped image via the depth plane at d_i and I_{r} denotes the reference frame.

In the encoder, there are five convolutional layers (a 7×7 filter for the first layer, a 5×5 filter for the second, and 3×3 filters for others). After encoding, we get a latent-space representation $\mathbf{y}$ of size $512 \times 8 \times 10$, which will be transformed by the GP model. Then decoder will take the transformed latent representation $\mathbf{z}$ as the input to generate a $1 \times H \times W$ prediction. There are four skip connections between the encoder and decoder and the inverse depth maps are predicted at four scales. All convolutional layers are followed by batch normalization and a ReLU function. The prediction layers using sigmoid function scaled by two to constrain the range of the predictions. To support arbitrary length of in-

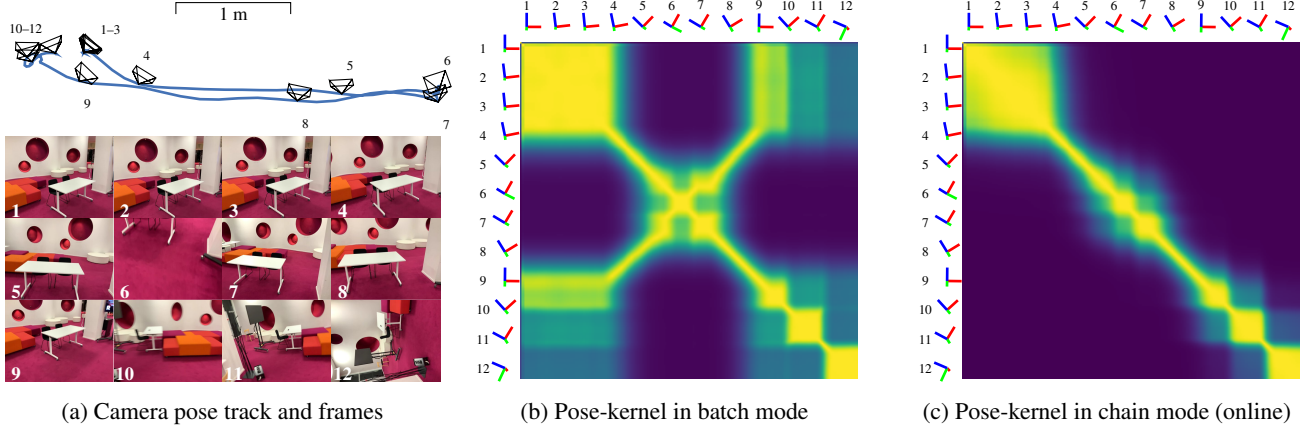(a) Camera pose track and frames (b) Pose-kernel in batch mode (c) Pose-kernel in chain mode (online)

Figure 3. (a) A continuous camera trajectory on the left with associated camera frames. In (b)–(c), the *a priori* pose-kernel covariance structures are shown as matrices (colormap: 0 ▇▇▇ γ^2). The kernel encodes information about how much similarity (or correlation) we expect certain views to have in their latent space. See, *e.g.*, the correlation between poses 1–4 and 9. In (b) this correlation is propagated over the entire track, while in (c) the long-range effects are suppressed. The small coordinate xyz-axes illustrate camera orientations.

puts, when there are more than one neighbour frame, we compute the cost volume with each neighbour image separately and then average the cost volumes before passing them to the encoder–decoder network.

During training, with a sequence of N input frames, we predict N depth maps by using the previous frame as the neighbour frame (except for the first frame that use the next frame as the neighbour frame), and use the mean of the L1 errors (at four scales) of all frames as the overall loss for training the model.

3.2. Pose-kernel Gaussian process prior

We seek to define a probabilistic prior on the latent space that would account for *a priori* knowledge that poses with close or overlapping field of view should produce more similar latent space encodings than poses far away from each other or poses with the camera pointing in opposite directions. This knowledge is to be encoded by a covariance function (kernel), and for this we need to define a distance measure or *metric* to define 'closeness' in pose-space.

To measure the distance between camera poses, we build upon the work by Mazzotti *et al.* [20] which considers measures of rigid body poses. We extend this work to be suitable for computer vision applications. Specifically, we propose the following pose-distance measure between two camera poses P_i and P_j:

$$D[P_i, P_j] = \sqrt{\|\mathbf{t}_i - \mathbf{t}_j\|^2 + \frac{2}{3}\,\mathrm{tr}(\mathbf{I} - \mathbf{R}_i^\top \mathbf{R}_j)}, \quad (2)$$

where the poses are defined as $P = \{\mathbf{t}, \mathbf{R}\}$ residing in $\mathbb{R}^3 \times$ SO(3), $\mathbf{I}$ is an identity matrix, and 'tr' denotes the matrix trace operator.

We define a covariance (kernel) function for the latent space bottleneck layer in Fig. 2. We design the prior for

the latent space processes such that they are stationary and both mean square continuous and once differentiable (see [24], Ch. 4) in pose-distance. This design choice is motivated by the fact that we expect the latent functions to model more structural than purely visual features, and that the we want the latent space to behave in a continuous and relatively smooth fashion. Choosing the covariance function structure from the so-called Matérn class [24] fulfils these requirements:

$$\kappa(P, P') = \gamma^2 \left(1 + \frac{\sqrt{3}\,D[P, P']}{\ell}\right) \exp\left(-\frac{\sqrt{3}\,D[P, P']}{\ell}\right). \tag{3}$$

This kernel encodes two arbitrary camera poses P and P' to 'nearness' or similarity in latent space values subject to the distance (in the sense of Eq. 2) of the camera poses. The tunable (learnable) hyperparameters γ^2 and ℓ define the characteristic magnitude and length-scale of the processes. Fig. 3 shows an example camera pose track and associated covariance matrix evaluated from Eq. (3) with unit hyperparameters.

In order to share the temporal information between frames in the sequence, we assign *independent* GP priors to all values in $\mathbf{z}_i$, and consider the encoder outputs $\mathbf{y}_i$ to be noise-corrupted versions of the 'ideal' latent space encodings (see Fig. 2). This inference problem can be stated as the following GP regression model:

$$\begin{aligned} z_j(t) &\sim \mathrm{GP}(0, \kappa(P[t], P[t'])), \\ y_{j,i} &= z_j(t_i) + \varepsilon_{j,i}, \quad \varepsilon_{j,i} \sim \mathrm{N}(0, \sigma^2), \end{aligned} \tag{4}$$

where $z_j(t)$, $j = 1, 2, \ldots, (512 \times 8 \times 10)$, are the values of the latent function $\mathbf{z}$ at time t. The noise variance σ^2 is a parameter of the likelihood model, and thus the third and final free parameter to be learned.
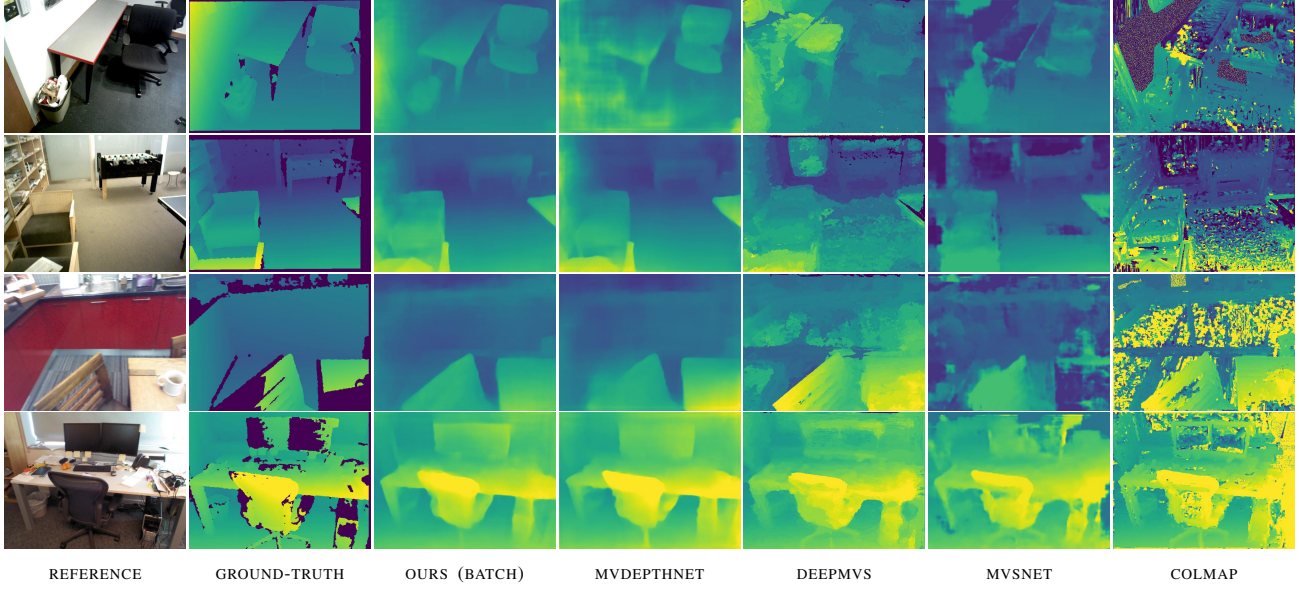
| REFERENCE | GROUND-TRUTH | OURS (BATCH) | MVDEPTHNET | DEEPMVS | MVSNET | COLMAP |

Figure 4. Qualitative comparisons on the SUN3D and 7SCENES.

3.3. Latent-state batch estimation

We first consider a batch solution for solving the inference problem in Eq. (4) for an unordered set of image-pose pairs. Because the likelihood is Gaussian and all the GPs share the same poses at which the covariance function is evaluated, we may solve all the $512 \times 8 \times 10$ GP regression problems with one matrix inversion. This is due to the posterior covariance only being a function of the input poses, not values of learnt representations of images (*i.e.*, $\mathbf{y}$ does not appear in the posterior variance terms in Eq. 5). The posterior mean and covariance will be given by [24]:

$$
\begin{aligned}
\mathbb{E}[\mathbf{Z} \,|\, \{(P_i, \mathbf{y}_i)\}_{i=1}^N] &= \mathbf{C}\,(\mathbf{C} + \sigma^2\,\mathbf{I})^{-1}\,\mathbf{Y}, \\
\mathbb{V}[\mathbf{Z} \,|\, \{(P_i, \mathbf{y}_i)\}_{i=1}^N] &= \mathrm{diag}(\mathbf{C} - \mathbf{C}\,(\mathbf{C} + \sigma^2\,\mathbf{I})^{-1}\,\mathbf{C}),
\end{aligned} \tag{5}
$$

where $\mathbf{Z} = (\mathbf{z}_1\,\mathbf{z}_2\,\dots\,\mathbf{z}_N)^\top$ are stacked latent space encodings, $\mathbf{Y} = (\mathbf{y}_1\,\mathbf{y}_2\,\dots\,\mathbf{y}_N)^\top$ are outputs from the encoder, and the covariance matrix $\mathbf{C}_{i,j} = \kappa(P_i, P_j)$ (see Fig. 3b for an example). The posterior mean $\mathbb{E}[\mathbf{z}_i \,|\, \{(P_i, \mathbf{y}_i)\}_{i=1}^N]$ is then passed through the decoder to output the predicted disparity map.

This batch scheme considers all the inter-connected poses in the sequence, making it powerful. The downside is that the matrix $\mathbf{C}$ grows with the number of input frames/poses, N, and the inference requires inverting the matrix—which scales cubically in the matrix size. This scheme is thus applicable only to sequences with up to some hundreds of frames.

3.4. Online estimation

In the case that the image-pose pairs have a natural ordering—as in a real-time application context—we may relax our model to a directed graph (*i.e.*, Markov chain, see Fig. 2 for the chain). In this case the GP inference problem can be solved in state-space form (see [26, 25]) with a *constant* computational and memory complexity per pose/frame. The inference can be solved *exactly* without approximations by the following procedure [26].

For state-space GP inference, the covariance function (GP prior) is converted into a dynamical model. The initial (prior) state is chosen as the steady-state corresponding to the Matérn covariance function (Eq. 3): $\mathbf{z}_0 \sim \mathrm{N}(\boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0)$, where $\boldsymbol{\mu}_0 = \mathbf{0}$ and $\boldsymbol{\Sigma}_0 = \mathrm{diag}(\gamma^2, 3\gamma^2/\ell^2)$. We jointly infer the posterior of all the independent GPs, such that the mean $\boldsymbol{\mu}_i$ is a matrix of size $2 \times (512 \cdot 8 \cdot 10)$, where the columns are the time-marginal means for the independent GPs and the two-dimensional state comes from the Matérn model being once mean square differentiable. The covariance matrix is shared between all the independent GPs, $\boldsymbol{\Sigma}_i \in \mathbb{R}^{2 \times 2}$. This makes the inference fast.

Following the derivation in [26], we define an evolution operator (which has the behaviour of the Matérn)

$$
\boldsymbol{\Phi}_i = \exp\!\left[\begin{pmatrix} 0 & 1 \\ -3/\ell^2 & -2\sqrt{3}/\ell \end{pmatrix} \Delta P_i \right], \tag{6}
$$

where the pose difference $\Delta P_i = D[P_i, P_{i-1}]$ is the pose-distance between consecutive poses. This gives us the predictive latent space values $\mathbf{z}_i \,|\, \mathbf{y}_{1:i-1} \sim \mathrm{N}(\bar{\boldsymbol{\mu}}_i, \bar{\boldsymbol{\Sigma}}_i)$, where the mean and covariance are propagated by:

$$
\bar{\boldsymbol{\mu}}_i = \boldsymbol{\Phi}_i\,\boldsymbol{\mu}_{i-1}, \qquad \bar{\boldsymbol{\Sigma}}_i = \boldsymbol{\Phi}_i\,\boldsymbol{\Sigma}_{i-1}\,\boldsymbol{\Phi}_i^\top + \mathbf{Q}_i, \tag{7}
$$

where $\mathbf{Q}_i = \boldsymbol{\Sigma}_0 - \boldsymbol{\Phi}_i\,\boldsymbol{\Sigma}_0\,\boldsymbol{\Phi}_i^\top$. The posterior mean and covariance is then given by conditioning on the encoder output

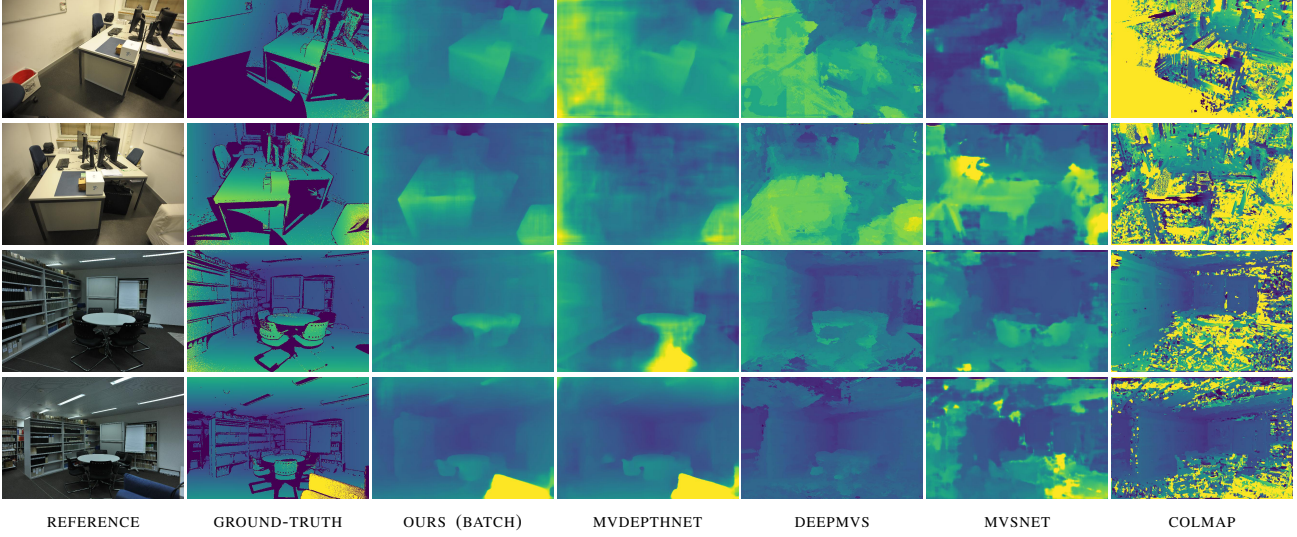| REFERENCE | GROUND-TRUTH | OURS (BATCH) | MVDEPTHNET | DEEPMVS | MVSNET | COLMAP |

Figure 5. Qualitative comparisons on the ETH3D dataset.

$\mathbf{y}_i$ of the current step:

$$\boldsymbol{\mu}_i = \bar{\boldsymbol{\mu}}_i + \mathbf{k}_i \, (\mathbf{y}_i^\top - \mathbf{h}^\top \bar{\boldsymbol{\mu}}_i), \quad \boldsymbol{\Sigma}_i = \bar{\boldsymbol{\Sigma}}_i - \mathbf{k}_i \, \mathbf{h}^\top \bar{\boldsymbol{\Sigma}}_i, \quad (8)$$

where $\mathbf{k}_i = \bar{\boldsymbol{\Sigma}}_i \, \mathbf{h} / (\mathbf{h}^\top \bar{\boldsymbol{\Sigma}}_i \, \mathbf{h} + \sigma^2)$ and the observation model $\mathbf{h} = (1 \; 0)^\top$. The posterior latent space encodings $\mathbf{z}_i \mid \mathbf{y}_{1:i} \sim N(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$ conditioned on all image-pose pairs up till the current are then passed through the decoder to produce the disparity prediction. Due to overloaded notation (the state-space model tracks both the latent space values and their derivatives), it is actually $\mathbf{h}^\top \boldsymbol{\mu}_i$ that is passed to the decoder.

4. Experiments

We train our model with the same data as in DeMoN [32]. The training data set includes short sequences from real-world data sets SUN3D [35], RGBD [31], MVS (includes CITYWALL and ACHTECK-TURM [7]), and a synthesized data set SCENES11 [32]. There are 92,558 training samples and each training sample consists of a three-view sequence with ground-truth depth maps and camera poses. The resolution of the input images is 320×256. All data in our training set are also used in the training set of MVDepth-Net, but the size of our training set is much smaller, so the improvement of the performance should not be explained by our training set. We load the MVDepthNet pretrained model as the starting point of training. We jointly train the encoder, decoder, and the GP hyperparameters on a desktop worstation (NVIDIA GTX 1080 Ti, i7-7820X CPU, and 63 GB memory) using the Adam solver [16] with $\beta_1 = 0.9$ and $\beta_2 = 0.999$, and a learning rate of 10^{-4}. The model was implemented in PyTorch and trained with 46k iterations with a mini-batch size of four. During training, we use the batch GP scheme (Sec. 3.3). After training, the GP hyperparameters were $\gamma^2 = 13.82$, $\ell = 1.098$, and $\sigma^2 = 1.443$.

4.1. Evaluation

We evaluate our method on four sequences picked randomly from the indoor data set 7SCENES [28] (*office-01*, *office-04*, *redkitchen-01*, *redkitchen-02*). The 7SCENES data set can be regarded as an ideal evaluation data set as none of models are trained with 7SCENES, the results can reveal the generalization abilities of models. Moreover, sequences in 7SCENES generally contain different viewpoints in the same room, so there are many neighbour views that share similar scenes, which is applicable for studying the impact of our fusion scheme. Four sequences from SUN3D (*mit_46_6lounge*, *mit_dorm_mcc_eflr6*, *mit_32_g725*, *mit_w85g*) and two sequences from ETH3D (*kicker*, *office*) are in evaluation of predicted depth maps. Altogether, there are 951 views in the evaluation set.

We use four common error metrics: *(i)* L1, *(ii)* L1-rel, *(iii)* L1-inv, and *(iv)* sc-inv. The three L1 metrics are mean absolute difference, mean absolute relative difference, and mean absolute difference in inverse depth, respectively. They are given as L1 $= \frac{1}{n} \sum_i |d_i - \hat{d}_i|$, L1-rel $= \frac{1}{n} \sum_i |d_i - \hat{d}_i| / \hat{d}_i$, and L1-inv $= \frac{1}{n} \sum_i |d_i^{-1} - \hat{d}_i^{-1}|$, where d_i (meters) is the predicted depth value, $\hat{d}_i$ (meters) is the ground-truth value, n is the number of pixels for which the depth is available. The scale-invariant metric is sc-inv $= \left(\frac{1}{n} \sum_i z_i^2 - 1/n^2 (\sum_i z_i)^2 \right)^{1/2}$, where $z_i = \log d_i - \log \hat{d}_i$. L1-rel normalizes the error, L1-inv puts more importance to close-range depth values, and sc-inv is a scale-invariant metric.

We compare our method with three state-of-the-art CNN-based MVS methods (MVSNet [36], DeepMVS [11], and MVDepthNet [34]), and one traditional MVS method

Table 1. Comparison results between COLMAP, MVSNet, DeepMVS, MVDepthNet, and our method. We outperform other methods in most of the data sets and error metrics (smaller better).

	COLMAP	MVSNet	DeepMVS	MVDepthNet	Ours (online)	Ours (batch)
SUN3D						
L1-rel	0.8169	0.3971	0.4196	0.1147	0.1064	**0.1010**
L1-inv	0.5356	0.1204	0.1103	0.0610	0.0548	**0.0512**
sc-inv	0.8117	0.3355	0.3288	0.1320	0.1268	**0.1220**
L1	1.6324	0.6538	0.9923	0.2631	0.2512	**0.2386**
7SCENES						
L1-rel	0.5923	0.2789	0.2198	0.1972	0.1706	**0.1583**
L1-inv	0.4160	0.1201	0.0946	0.1064	0.0931	**0.0884**
sc-inv	0.4553	0.2570	0.2258	0.1611	0.1490	**0.1458**
L1	1.0659	0.4971	0.4183	0.3807	0.3187	**0.2947**
ETH3D						
L1-rel	0.5574	0.4706	0.4124	0.2569	0.2354	**0.2291**
L1-inv	0.4307	0.1901	0.3380	0.1366	0.1227	**0.1066**
sc-inv	0.5595	0.4555	0.4661	0.2667	0.2561	**0.2517**
L1	0.6440	0.9567	0.5684	0.5979	0.5417	**0.5374**

Ground-truth

Without GP

With GP

Figure 6. 3D reconstruction on 7SCENES by TSDF Fusion [38]. The results are fused from 25 depth maps.

(COLMAP, [27]), because all these methods are available to image sequences. For COLMAP, we use the ground-truth poses to produce dense models directly. For MVS-Net, 192 depth labels based on ground-truth depth are used. For COLMAP, MVSNet, and DeepMVS, to get good results, four neighbour images are assigned for each reference image, while MVDepthNet and our method only use one previous frame that has enough angle difference ($>15°$) or baseline translation (>0.1 m) as the neighbour frame.

As shown in Table 1, our methods, both the online and batch version, outperform other methods on all evaluation sets/metrics. Compared with the original MVDepthNet, the performance gets improved on all data sets after introducing the GP prior. These results underline, that sharing information across different poses always seems beneficial. As expected, the online estimation results are slightly worse than the batch estimation, because the online method only leverages the frames in the past. All models are trained with similar scenes, except for MVSNet which is trained with the DTU dataset that has much smaller scale of depth ranges; as our test sequences have larger ranges, the depth labels might become too sparse for the model, explaining its failed predictions. As also noted in the original publications, running COLMAP and DeepMVS is slow (orders of magnitude slower than the other methods). In comparison to MVDepthNet, as the GP inference only adds the cost of some comparably small matrix calculations which is small in comparison to the network evaluations, the improvements come at almost no cost.

Fig. 4 and 5 show qualitative comparison results. Patch-based methods like DeepMVS and COLMAP more easily to suffer from textureless regions and are more noisy. Compared to MVDepthNet, introducing the GP prior helps to obtain more stable depth maps with sharper edges. Fig. 6
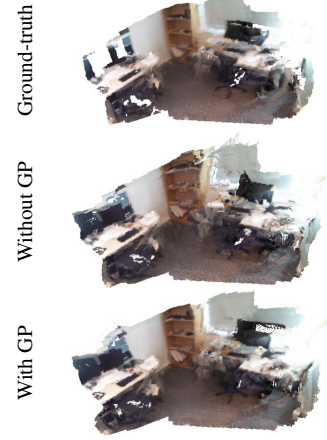
reveals the temporal consistency of our method and proves that it is supplementary to traditional fusion methods.

4.2. Ablation studies

We have conducted several ablation studies for the design choices in our method.

Number of neighbour frames. MVS methods typically use more than two input frames to reduce the noise in the cost volume. Our method can also use more than just a pair of inputs. Table 2 shows the results on *redkitchen-02*, where we compare our method to MVDepthNet and MVSNet. The use of more input frames improves all methods, but does not change the conclusions. Without the GP prior, even using five frames is inferior to our method with only two frames.

Neighbour selection. Strict view selection rules are required in many methods to obtain good predictions, because the cost volume breaks down if there is not enough baseline between views. We study robustness by decreasing the threshold of translation when selecting the neighbour frame in the SUN3D and 7SCENES sequences. In Table 3, the error metrics increase more without using GP priors, which signals that the GP is beneficial in cases where the camera does not move much.

Choice of kernel function. In addition to the Matérn kernel, we experiment with the exponential kernel [24]: $\kappa(P, P') = \gamma^2 \exp(-D[P, P']/\ell)$. The exponential kernel does not encode any smoothness (not differentiable), which makes it too flexible for the task as can be read from the error metrics in Table 4. If we ignore the pose information in the kernel, and only use the temporal difference (TD) instead, $D[i, j] = |i - j|$, the GP can be seen as a low-pass filter. We experimented with the TD in the Matérn ker-

Table 2. Ablation experiment: Performance comparison w.r.t. different number of input frames.

Metric / Methods	2 frames			3 frames			5 frames		
	MVDepthNet	MVSNet	Ours	MVDepthNet	MVSNet	Ours	MVDepthNet	MVSNet	Ours
L1-rel	0.2009	0.3159	**0.1615**	0.1897	0.2665	**0.1460**	0.1734	0.2758	**0.1429**
L1-inv	0.1161	0.1435	**0.0979**	0.1064	0.1244	**0.0881**	0.1028	0.1195	**0.0850**
sc-inv	0.1866	0.3250	**0.1729**	0.1809	0.2902	**0.1598**	0.1766	0.2809	**0.1587**
L1	0.4238	0.6036	**0.3386**	0.3922	0.5133	**0.3066**	0.3619	0.5116	**0.2964**

Table 3. Performance comparison w.r.t. thresholds of translation.

Metric / Methods	$t_{\min} = 0.1$ m		$t_{\min} = 0.05$ m	
	w/o GP	Ours	w/o GP	Ours
L1-rel	0.1474	0.1238	0.1535	0.1262
L1-inv	0.0790	0.0660	0.0828	0.0669
sc-inv	0.1436	0.1315	0.1487	0.1334
L1	0.3098	0.2609	0.3242	0.2664

Table 4. Performance comparison w.r.t. different kernels.

Metric / Methods	L1-rel	L1-inv	sc-inv	L1
Matérn	0.1298	0.0683	0.1384	0.2769
Exponential	0.1376	0.0703	0.1417	0.2846
TD kernel	0.1450	0.0745	0.1457	0.3041
w/o GP	0.1538	0.0824	0.1507	0.3265

nel, which gives better results than not using a GP, but performed worse than both the GPs that use the pose-distance.

4.3. Online experiments with iOS

To demonstrate the practical value of our MVS scheme, we ported our implementation to an iOS app. The online GP scheme and cost volume construction were implemented in C++ with wrappers in Objective-C, while the app itself was implemented in Swift. More specifically, the homography warping for cost volume construction leverages OpenCV, and the real-time GP is implemented using the Eigen matrix library. The trained PyTorch model was converted to a CoreML model through ONNX. The camera poses are captured by Apple ARKit. Fig. 7 shows a screenshot of the app in action, where we have set the refresh rate to ∼1 Hz. Note that the model was not trained with any iOS data, nor any data from the environment the app was tested in.

5. Discussion and conclusion

In this paper, we proposed a novel idea for MVS that enables the model to leverage multi-view information but keep the frame structure simple and time-efficient at the same time. Our pose-kernel measures the 'closeness' between frames and encodes this prior information using a Gaussian process in the latent space. In the experiments, we showed that this method clearly advances the state-of-the-art. Our proposed model consistently improves the accuracy of estimated depth maps when appended to the baseline disparity network of [34] and this holds independently of the number of views used for computing individual cost volumes. In addition, as the proposed model fuses information in the latent space, it is complementary with depth map fusion techniques, such as [38], which can fuse information from multiple depth maps estimated using our approach. In fact, besides improving individual depth map predictions, our latent-space GP prior leads to improved result also when
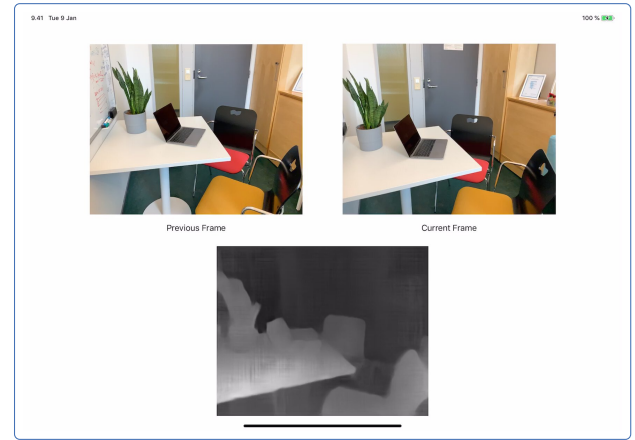

Figure 7. Screenshot of our disparity estimation method running on an Apple iPad Pro (11-inch, late-2018 model). The previous and current frames are side-by-side on the top. The predicted disparity (corresponding to the current frame) is visualized on the bottom. The pose information comes from Apple's ARKit API.

combined with a subsequent depth map fusion stage.

One possible limitation of our method is that wrong predictions might also be propagated forward because of the fusion in the latent space. We do not employ any outlier rejection rules like traditional depth fusion methods. The same applies to occlusion. Even though we recognize this concern, we did not notice any problems in robustness while experimenting with our online app implementation. Still, introducing confidence measures to penalize wrong predictions could improve the method in the future.

Codes and material available on the project page: https://aaltoml.github.io/GP-MVS.

References

[1] Acute3D, A Bentley Systems Company. https://www.acute3d.com/. 1

[2] Francesco Paolo Casale, Adrian Dalca, Luca Saglietti, Jennifer Listgarten, and Nicolo Fusi. Gaussian process prior variational autoencoders. In *Advances in Neural Information Processing Systems (NIPS)*, pages 10369–10380. Curran Associates, Inc., 2018. 3

[3] Robert T. Collins. A space-sweep approach to true multi-image matching. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 358–363, 1996. 2

[4] Stefanos Eleftheriadis, Ognjen Rudovic, Marc Peter Deisenroth, and Maja Pantic. Variational Gaussian process auto-encoder for ordinal prediction of facial action units. In *Asian Conference on Computer Vision (ACCV)*, pages 154–170. Springer, 2016. 3

[5] Stefanos Eleftheriadis, Ognjen Rudovic, and Maja Pantic. Discriminative shared Gaussian processes for multiview and view-invariant facial expression recognition. *IEEE Transactions on Image Processing*, 24(1):189–204, 2015. 3

[6] Stefanos Eleftheriadis, Ognjen Rudovic, and Maja Pantic. Multi-conditional latent variable model for joint facial action unit detection. In *IEEE International Conference on Computer Vision (ICCV)*, pages 3792–3800, 2015. 3

[7] Simon Fuhrmann, Fabian Langguth, and Michael Goesele. MVE – A multi-view reconstruction environment. In *Eurographics Workshop on Graphics and Cultural Heritage (GCH)*, pages 11–18, 2014. 6

[8] Yasutaka Furukawa, Brian Curless, Steven M. Seitz, and Richard Szeliski. Towards Internet-scale multi-view stereo. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1434–1441, 2010. 1

[9] Yasutaka Furukawa and Jean Ponce. Accurate, dense, and robust multiview stereopsis. *IEEE Transactions on Pattern Analysis Machine Intelligence*, 32(8):1362–1376, 2010. 2

[10] Yuxin Hou, Arno Solin, and Juho Kannala. Unstructured multi-view depth estimation using mask-based multiplane representation. In *Scandinavian Conference on Image Analysis (SCIA)*, pages 54–66. Springer, 2019. 2

[11] Po-Han Huang, Kevin Matzen, Johannes Kopf, Narendra Ahuja, and Jia-Bin Huang. DeepMVS: Learning multi-view stereopsis. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2821–2830, 2018. 2, 6

[12] Sunghoon Im, Hae-Gon Jeon, Stephen Lin, and In So Kweon. DPSNet: End-to-end deep plane sweep stereo. In *International Conference on Learning Representations (ICLR)*, 2019. 2

[13] Abhishek Kar, Christian Häne, and Jitendra Malik. Learning a multi-view stereo machine. In *Advances in Neural Information Processing Systems (NIPS)*, pages 365–376, 2017. 2

[14] Alex Kendall and Yarin Gal. What uncertainties do we need in Bayesian deep learning for computer vision? In *Advances in Neural Information Processing Systems (NIPS)*, pages 5574–5584. Curran Associates, Inc., 2017. 3

[15] Alex Kendall, Hayk Martirosyan, Saumitro Dasgupta, and Peter Henry. End-to-end learning of geometry and context for deep stereo regression. In *IEEE International Conference on Computer Vision (ICCV)*, pages 66–75, 2017. 1

[16] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 6

[17] Kalin Kolev, Maria Klodt, Thomas Brox, and Daniel Cremers. Continuous global optimization in multiview 3D reconstruction. *International Journal of Computer Vision*, 84(1):80–96, 2009. 2

[18] Kiriakos N. Kutulakos and Steven M. Seitz. A theory of shape by space carving. *International Journal of Computer Vision*, 38(3):199–218, 2000. 2

[19] Florent Lafarge, Renaud Keriven, Mathieu Brédif, and Hoang-Hiep Vu. A hybrid multiview stereo algorithm for modeling urban scenes. *IEEE Transactions on Pattern Analysis Machine Intelligence*, 35(1):5–17, 2013. 2

[20] Claudio Mazzotti, Nicola Sancisi, and Vincenzo Parenti-Castelli. A measure of the distance between two rigid-body poses based on the use of platonic solids. In *RO-MANSY 21-Robot Design, Dynamics and Control*, pages 81–89. Springer, 2016. 4

[21] Paul Merrell, Amir Akbarzadeh, Liang Wang, Philippos Mordohai, Jan-Michael Frahm, Ruigang Yang, David Nistér, and Marc Pollefeys. Real-time visibility-based fusion of depth maps. In *IEEE International Conference on Computer Vision (ICCV)*, pages 1–8, 2007. 2

[22] Richard A. Newcombe, Shahram Izadi, Otmar Hilliges, David Molyneaux, David Kim, Andrew J Davison, Pushmeet Kohi, Jamie Shotton, Steve Hodges, and Andrew Fitzgibbon. KinectFusion: Real-time dense surface mapping and tracking. In *International Symposium on Mixed and Augmented Reality*, pages 127–136, 2011. 3

[23] Matthias Nießner, Michael Zollhöfer, Shahram Izadi, and Marc Stamminger. Real-time 3D reconstruction at scale using voxel hashing. *ACM Transactions on Graphics (ToG)*, 2013. 3

[24] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006. 2, 3, 4, 5, 7

[25] Simo Särkkä and Arno Solin. *Applied Stochastic Differential Equations*. Cambridge University Press, Cambridge, UK, 2019. 5

[26] Simo Särkkä, Arno Solin, and Jouni Hartikainen. Spatiotemporal learning via infinite-dimensional Bayesian filtering and smoothing. *IEEE Signal Processing Magazine*, 30(4):51–61, 2013. 3, 5

[27] Johannes L. Schönberger, Enliang Zheng, Jan-Michael Frahm, and Marc Pollefeys. Pixelwise view selection for unstructured multi-view stereo. In *European Conference on Computer Vision (ECCV)*, pages 501–518, 2016. 1, 7

[28] Jamie Shotton, Ben Glocker, Christopher Zach, Shahram Izadi, Antonio Criminisi, and Andrew Fitzgibbon. Scene coordinate regression forests for camera relocalization in RGB-D images. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2930–2937, 2013. 6

[29] Arno Solin, Santiago Cortes, Esa Rahtu, and Juho Kannala. PIVO: Probabilistic inertial-visual odometry for occlusion-

robust navigation. In *Winter Conference on Applications of Computer Vision (WACV)*, pages 616–625, 2018. 1

[30] Arno Solin, James Hensman, and Richard E. Turner. Infinite-horizon gaussian processes. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 3486–3495. Curran Associates, Inc., 2018. 3

[31] Jürgen Sturm, Nikolas Engelhard, Felix Endres, Wolfram Burgard, and Daniel Cremers. A benchmark for the evaluation of RGB-D SLAM systems. In *International Conference on Intelligent Robots and Systems (IROS)*, pages 573–580, 2012. 6

[32] Benjamin Ummenhofer, Huizhong Zhou, Jonas Uhrig, Nikolaus Mayer, Eddy Ilg, Alexey Dosovitskiy, and Thomas Brox. DeMoN: Depth and motion network for learning monocular stereo. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5038–5047, 2017. 6

[33] Raquel Urtasun, David J. Fleet, and Pascal Fua. 3D people tracking with Gaussian process dynamical models. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 238–245, 2006. 3

[34] Kaixuan Wang and Shaojie Shen. MVDepthNet: Real-time multiview depth estimation neural network. In *International Conference on 3D Vision (3DV)*, pages 248–257. IEEE, 2018. 1, 2, 3, 6, 8

[35] Jianxiong Xiao, Andrew Owens, and Antonio Torralba. SUN3D: A database of big spaces reconstructed using SfM and object labels. In *IEEE International Conference on Computer Vision (ICCV)*, pages 1625–1632, 2013. 6

[36] Yao Yao, Zixin Luo, Shiwei Li, Tian Fang, and Long Quan. MVSNet: Depth inference for unstructured multi-view stereo. In *European Conference on Computer Vision (ECCV)*, pages 767–783, 2018. 2, 6

[37] Markus Ylimäki, Juho Kannala, Jukka Holappa, Sami S. Brandt, and Janne Heikkilä. Fast and accurate multi-view reconstruction by multi-stage prioritised matching. *IET Computer Vision*, 9(4):576–587, 2015. 2

[38] Andy Zeng, Shuran Song, Matthias Nießner, Matthew Fisher, Jianxiong Xiao, and Thomas Funkhouser. 3dmatch: Learning local geometric descriptors from rgb-d reconstructions. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 7, 8

[39] Huizhong Zhou, Benjamin Ummenhofer, and Thomas Brox. DeepTAM: Deep tracking and mapping. In *European Conference on Computer Vision (ECCV)*, pages 822–838, 2018. 2