Densifying SLAM for UAV navigation by fusion of monocular depth prediction

Yassine Habib, Panagiotis Papadakis, Cédric Le Barz, Antoine Fagette, Tiago Gonçalves, Cédric Buche

Densifying SLAM for UAV navigation by fusion of monocular depth prediction

Yassine Habib[1,2], Panagiotis Papadakis[2], Cédric Le Barz[1], Antoine Fagette[3], Tiago Gonçalves[1], Cédric Buche[4]

[1] ThereSIS Lab, Thales SIX GTS France, Palaiseau, France
[2] IMT Atlantique, Lab-STICC, UMR 6285, team RAMBO, Brest, France
[3] Thales Research & Technology Canada, Thales Digital Solutions, Montreal, Canada
[4] IRL CNRS CROSSING, Adelaide, Australia

***Abstract*—Simultaneous Localization and Mapping (SLAM) research has reached a level of maturity enabling systems to build autonomously an accurate sparse map of the environment while localizing themselves in that map. At the same time, the use of deep learning has recently brought great improvements in Monocular Depth Prediction (MDP). Some applications such as autonomous drone navigation and obstacle avoidance require dense structure information and cannot only rely on sparse SLAM representation. We propose to densify a state-of-the-art SLAM algorithm using deep learning-based dense MDP at keyframe rate. Towards this goal, we describe a scale recovery from SLAM landmarks by minimizing a depth error metric combined with a multi-view depth refinement using a volumetric approach. We conclude with experiments that attest the added value of our approach in terms of depth estimation.**

***Index Terms*—dense SLAM, monocular depth prediction, drone navigation**

I. Introduction

Autonomous navigation of a robot in an unknown area requires the perception and analysis of the surrounding environment. A way to achieve this is by reconstructing a 3D map which further serves at localizing the robot. In addition, real-time pose estimation enables applications such as autonomous drone navigation in GNSS-denied regions. This topic is mainly formulated as a maximum-a-posteriori estimation problem where the robot state is estimated from sensors' measurements. This subject is typically referred as Simultaneous Localization and Mapping (SLAM).

In this work we are particularly interested in UAV applications such as exploration of buildings for damage assessment [1]. Drone navigation does not need a high-resolution 3D mesh but rather a coarse, dense, metric representation. A voxel map like the one in Fig. 1 fits this need. For an embedded application, we focus on passive sensors, especially monocular inertial configurations. LiDAR and RGB-D sensors measure depth at the cost of power consumption and yet still suffer from sensor acquisition limitations. Stereo cameras also provide depth by stereo matching, but they require perfect calibration at all times and more volume because their range is limited by the baseline. The monocular camera is the most affordable solution as it requires less power and size. However, pixel depth recovery is much more difficult. The addition of an IMU sensor makes it possible to retrieve the metric information which considerably improves accuracy.

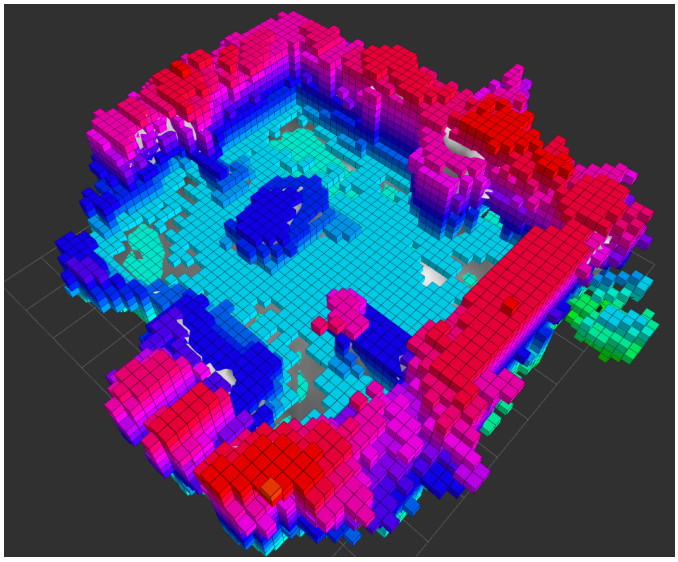

Fig. 1. Voxel map reconstructed using Voxblox on EuRoC [2] V1_01 scene from stereo matching.

Nowadays, modern Visual SLAM systems are mature enough to provide accurate localization in a privileged context. Conventional state-of-the-art methods such as Basalt [3] and ORB-SLAM 3 [4] focus on maintaining an accurate sparse map ensuring a good localization and minimizing computation time. Building and maintaining a dense map requires considerably more calculation, but some methods investigate solutions to densify SLAM. Indicatively, Kimera [5] uses a dedicated thread to estimate dense depth from stereo images, then builds a voxel map with raycasting, and finally derives a 3D mesh by marching cubes. Nevertheless, this cannot be applied to monocular cameras which are subject to scale ambiguity.

On the other hand, Deep Neural Networks (DNN) have brought significant improvements in Monocular Depth Prediction (MDP), and most recent approaches tend to use them to

We would like to thank the ANRT (Association Nationale de la Recherche et de la Technologie), for its funding through the CIFRE grant 2019/1877.

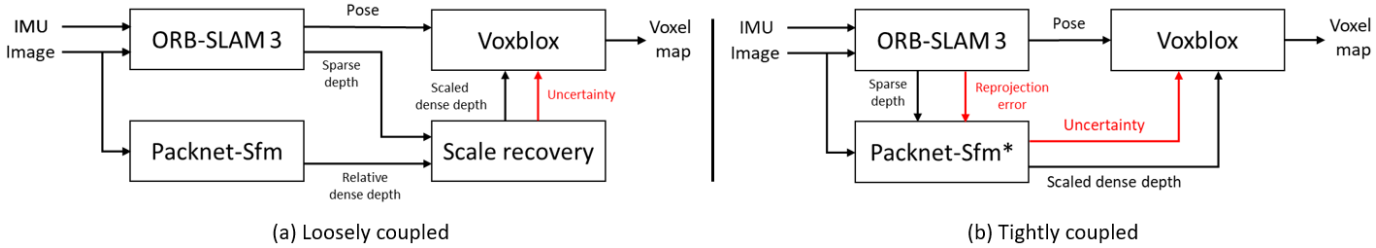

Fig. 2. Scheme of our proposed pipeline. (a) Loosely coupled: recover the scale of the predicted dense depth from the sparse depth estimated by SLAM. (b) Tightly coupled: predict absolute dense depth from a monocular image and the estimated sparse depth, requiring an adjustment of the Packnet-Sfm network. A voxel map is built with Voxblox by multi-view fusion and grouped raycasting. For both approaches, we further propose to account for the uncertainty of our final depth map (red arrows).

densify SLAM. CodeSLAM [6] introduced the use of Variational AutoEncoders (VAE) to infer depth, learning a compact depth representation from an RGB image and a SLAM-based sparse map. DeepFactors [7] and CodeMapping [8] implement the CodeSLAM VAE in a full SLAM system and further add depth uncertainty prediction and multi-view refinement via a factor graph. The latter also leverages available reprojection error to account for sparse depth uncertainty.

These MDP solutions are mainly based on supervised learning requiring a huge amount of image data with registered depth ground truth which is a tedious task. As a result, self-supervised methods gained interest allowing to train a network without ground truth. Indeed, based on multi-view geometry, training only needs stereo images [9] to reconstruct the left image from the inferred depth and the right image, after which the photometric loss is calculated for the reconstruction error. Some works [10]–[12] extend this approach to monocular sequences where relative pose between frames is known or jointly estimated. On the same idea, Packnet-Sfm [13] implements packing and unpacking blocks through 3D convolutions, claiming that it preserves dense geometric and appearance details as much as possible.

Thus, we choose to study the capacity of MDP to help densify SLAM. In this paper, we present our on-going work to provide drones with the capability to derive a dense and metric 3D map using SLAM. Section II describes our approach for SLAM map densification. Then, we report and analyze our initial results in Section III and finally, we discuss our perspectives for subsequent work in Section IV.

II. Our approach

We propose to densify SLAM sparse maps for UAV navigation in two stages. The global pipeline is illustrated in Fig. 2 (a). The first step consists in predicting a dense depth map using Packnet-Sfm and to scale it using ORB-SLAM 3 triangulated points (landmarks), as described in section II-A. At a second step, we aim to refine the scaled dense depth through multi-view refinement via volumetric fusion, as outlined in section II-B. The ORB-SLAM 3 tracking thread can run at a frame rate of 20 fps on an embedded system to provide localization. Local mapping also runs at the targeted keyframe rate (2-4 fps) if we limit the drone navigation to a reasonable speed, a realistic assumption when exploring a building. Therefore, we can add Packnet-Sfm to the local mapping thread and expect it to have a minimum impact on speed. We try to minimize the complexity of the fusion since we do not want to significantly drop the keyframe rate.

A. Scale recovery

For a keyframe I_k we define $\Omega_k \subset I_k$ as the subset of pixels for which ORB-SLAM 3 estimated the depth such that for $p \in \Omega_k$ the estimated depth is $D_p^k \in \mathbb{R}_+^*$. Likewise, Packnet-Sfm infers a dense depth map such that each pixel $p \in I_k$ has a predicted depth $\hat{D}_p^k$. Assuming that the predicted depth map is consistent, we obtain:

$$\exists \alpha_k \in \mathbb{R}, \forall p \in I_k, Z_p^k = \alpha_k \hat{D}_p^k \tag{1}$$

where Z_p^k is the ground truth and α_k the scale factor for this keyframe. When evaluating depth prediction, many authors use the ratio of the median predicted depth on the median of the ground truth, which will refer to as the GT-scale:

$$\alpha_k = \frac{med(\{Z_p^k, p \in I_k\})}{med(\{\hat{D}_p^k, p \in I_k\})} \tag{2}$$

Such a choice can be statistically insignificant for a small set of points, especially if the set is heterogeneous. Instead, we minimize the square relative error defined in [15]. This metric compensates for large errors in deep points by dividing them by the ground truth:

$$\hat{\alpha}_k = \min_{\alpha} \frac{1}{N} \sum_{p \in \Omega_k} \frac{\left\| \alpha \hat{D}_p^k - D_p^k \right\|^2}{D_p^k} \tag{3}$$

with N being the number of points in Ω_k. By developing the sum we easily get a positive polynomial of a degree 2 which is easy to minimize:

$$\hat{\alpha}_k = \min_{\alpha} \alpha^2 \sum_{p \in \Omega_k} \frac{\hat{D}_p^{k2}}{D_p^k} - 2\alpha \sum_{p \in \Omega_k} \hat{D}_p^k + \sum_{p \in \Omega_k} D_p^k \tag{4}$$

We later refer to the obtained scale as the SR-scale. While solving this equation is simple we stress that this still resides on a strong assumption related to the consistency of the predicted depth map.

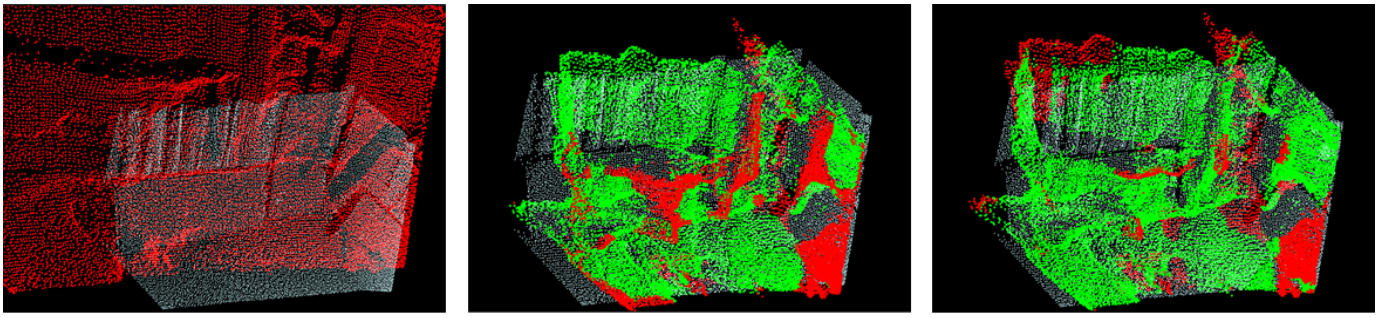

Fig. 3. Visualization on Camviz [14] of Packnet-Sfm [13] predicted depth map projected in 3D, with and without rescaling. Ground truth point cloud is white and each predicted point is green if $max(\frac{pred}{gt}, \frac{gt}{pred}) < 1.25$, red otherwise.

B. Multi-view depth refinement

In order to build a 3D map and improve the depth estimation, we need to perform a multi-view refinement. For this purpose, DeepFactors [7] and CodeMapping [8] use dense bundle adjustment (BA) and factor graph optimization. In [16], Rosinol et al. propose volumetric mapping to fuse depth maps which are weighed by a probabilistic uncertainty. In Kimera [5], bundled raycasting adapted from Voxblox [17] is used from multiple views with the estimated pose to build and maintain a voxel map. Indeed, thanks to stereo camera, a dense depth map can be estimated by stereo matching for each keyframe. Then, all points that belong to the same voxel are raycasted together which significantly speeds up the process. Thus, multi-view fusion is done at voxel level, and a mesh is eventually extracted by marching cubes.

We base our work on this last solution as we can obtain a dense depth map from previous steps. Basically, Voxblox constructs a voxel map where each voxel stores a weight and a distance to the nearest surface. A Truncated Signed Distance Function (TSDF) is defined and gives a distance to the surface boundary. Each voxel crossed by a ray and located up to a truncation distance sees its weight and distance updated. The weight function is fixed at 1 in front of the surface and decreases quadratically with the distance beyond the surface.

III. Experiments

We present here our initial results and observations by restricting our experiments in the scene V1_01 from EuRoC [2], since evaluation datasets that contain both trajectory and 3D structure ground truth are not easily accessible. We run ORB-SLAM 3 on this scene and export each keyframe with corresponding sparse map and timestamp. Dense depth maps are predicted from extracted frames using Packnet-Sfm, which was trained on the KITTI dataset [19]. Then, we obtain the ground truth for each keyframe by projecting the LiDAR point cloud into the image plane using ground truth camera pose and intrinsic parameters. Finally, we evaluate depth predictions using metrics defined in [15] as reported in Table I. Results from DeepFactors [7], Ma et al. [18] and CodeMapping [8] are also reported for reference.

Upon application of the recovered scale to the inferred dense depth as defined in (4), we observe a significant improvement as shown in Table I, divided in three groups. We first report ORB-SLAM 3 sparse depth evaluation for reference. It is evaluated on much less points than dense depth maps, thus the measure is more impacted by errors. Nonetheless, it confirms the relatively good accuracy of the depth estimation since $\delta_1 = 89.9\%$, referring to the percentage of estimated points that are within a 25% error range around the ground truth depth. The second group shows related results reporting only absolute difference and root mean square error. By leveraging reprojection error and multi-view depth refinement, CodeMapping attains excellent results at the cost of heavier computations. In the last group, we reveal Packnet-Sfm evaluation without scaling, with ground truth (GT) scaling, and with our proposition. Some qualitative results are shown in Fig. 3. We note that Packnet-Sfm fails to accurately infer some structures' depth, especially because the scale does not appear consistent on the image. However, it manages to segment many objects in the scene. We also note that the network fails to predict some planar surfaces or predicts them as if they were in an outdoor scene, with larger depth in the upper part of the image.

Finally, we tried using Voxblox from the dense depth maps obtained after scaling. These depth maps contain some errors which have large values that corrupt the voxel map during raycasting and which cannot be corrected by multi-view since they fall out of the truncation distance. We believe that improving the predicted dense depth map and filtering out these errors would solve this problem.

IV. Perspectives

Our first observation is about the dense depth map prediction quality. The model used here was trained on KITTI [19] which only includes outdoor scenes with mainly vehicles, street roads and trees. The network may have difficulty estimating square structures, objects close to the camera or the upper parts of the image because in outdoor images these parts correspond to the sky. We expect that a fine-tuning of the network on indoor scenes would greatly benefit the qualitative results. The HILTI dataset [20] provides interesting and

TABLE I
EVALUATION OF DEPTH PREDICTION ON EUROC [2] V1_01 SCENE. UNITS IN METERS. LIGHT†: WITHOUT MULTI-VIEW OR REPROJECTION ERROR.

	abs_diff	abs_rel	sq_rel	rmse	rmse_log	δ_1	δ_2	δ_3
ORB-SLAM 3 [4] (sparse)	0.283	0.147	0.315	0.612	0.218	0.899	0.946	0.973
DeepFactors [7]	0.842			1.050				
Ma et al. [18]	0.495			0.598				
CodeMapping [8] light†	0.280			0.435				
CodeMapping [8]	**0.192**			**0.381**				
Packnet-Sfm	6.209	2.637	21.756	7.171	1.244	0.014	0.047	0.123
Packnet-Sfm (GT-scale)	0.814	0.326	0.535	1.155	0.392	0.478	0.760	0.899
Packnet-Sfm (ORB-SLAM 3 SR-scale)	0.886	0.363	0.748	1.262	0.404	0.475	0.751	0.889

challenging indoor sequences with LiDAR data and grayscale images. However, colour images are more informative so we will not use it for training. Instead, we can use the TUM [21] dataset and also collect our own data with a calibrated monocular camera since no ground truth is required.

To tackle the problem of scale consistency, a possible solution consists in computing a scale map that would recover scale locally. A way to achieve this with minimum computation would be to segment the depth map and retrieve scale independently in each cluster as described in II-A. However, our approach is loosely coupled and relies on an ad-hoc formulation. Considering a tightly coupled method would greatly benefit the results. Indeed, we could adapt Packnet-Sfm network to also process ORB-SLAM 3 landmarks as input (Fig. 2 (b)). The sparse map, containing the metric information, would allow to directly infer a metric dense depth map. A depth error term can be appended to the loss, comparing the sparse set of points in the input and the prediction. Setting a dedicated learning rate would allow to increase the contribution of this term after few epochs, when the network is good enough at predicting relative dense depth. This way, we aim to leverage DNN capacity to model how to diffuse the sparse estimated depth over the image.

On the other hand, regarding Voxblox multi-view fusion, weights propagated along rays have to be adapted to deal with low confidence rays. ORB-SLAM 3 sparse depth points have a good confidence since they are estimated by BA based on epipolar geometry. The idea is therefore to compute an uncertainty map on the scaled dense depth, where the confidence is greater the closer it is to a sparse depth. We measure the tracked points confidence by their reprojection error and define an ad-hoc way to diffuse it to neighbouring pixels. Thus, we propose to use a Gaussian Mixture where each triangulated point of coordinates $q_i \in \Omega_k$ defines a Gaussian of mean $\mu_i = q_i$ and covariance $\Sigma_i = diag(R, R)$. Here, we define $R \in \mathbb{R}_+^*$ as a fixed radius to diffuse the weights around each landmark. The Gaussian Mixture is defined by the equation:

$$\forall x \in I_k, p(x|\pi_i, \mu, \Sigma) = \sum_{i=1}^{N} \pi_i p(x, \mu_i, \Sigma_i) \tag{5}$$

where π_i is the mixing coefficient. We leverage reprojection error ρ_i calculated by ORB-SLAM 3 and the softmax function as follows:

$$\pi_i = \frac{e^{-\rho_i}}{\sum_{j=1}^{N} e^{-\rho_j}} \qquad \text{such that} \quad \sum_{i=1}^{N} \pi_i = 1 \tag{6}$$

Thus we can now deduce the confidence map c as a function of the Gaussian Mixture density f:

$$\forall x \in I_k, f(x) = \sum_{i=1}^{N} \frac{\pi_i}{2\pi|\Sigma_i|^{\frac{1}{2}}} e^{-\frac{1}{2}(x-\mu_i)^T \Sigma_i^{-1}(x-\mu_i)} \tag{7}$$

$$\forall x \in I_k, c(x) = \max(c_{min}, f(x)) \tag{8}$$

c_{min} is a positive scalar used to keep a minimum confidence because we do not want to get rid of predicted points that are too far from any landmark. We plan to leverage this uncertainty to replace the constant weight fixed at 1 in front of the surface. Thus, the contribution of uncertain predictions are minimized or even discarded. In addition, we did not account for local and global mapping update of ORB-SLAM 3 covisibility graph. Indeed, the updates correct the position of landmarks, especially at the beginning or during loop closure. Leveraging these updates should allow to limit or discard a previous ray, particularly when a landmark had a significant change.

Finally, we discussed uncertainty calculation on our loosely coupled approach, but it could also be extended to the tightly coupled proposition as illustrated on Fig. 2 (b). Indeed, we could also append ORB-SLAM 3 reprojection error information to guide the network in predicting an uncertainty map.

V. CONCLUSION

In this paper, we presented our ongoing work on the joint use of ORB-SLAM 3 sparse depth and Packnet-Sfm predicted dense depth to produce a 3D dense metric map. As a first step, we proposed a simple scaling recovery solution with promising results which can serve as a basis for multi-view volumetric depth fusion based on grouped raycasting and confidence map calculation. We presented future works and perspectives, including Packnet-Sfm customization to propose a tightly coupled approach. Finally, the measurement of 3D metrics as presented in [5], [22] would allow a relevant analysis of the accuracy and completeness of the 3D mapping.

REFERENCES

[1] G.-J. M. Kruijff, F. Pirri, M. Gianni, P. Papadakis, M. Pizzoli, A. Sinha, V. Tretyakov, T. Linder, E. Pianese, S. Corrao, F. Priori, S. Febrini, and S. Angeletti, "Rescue robots at earthquake-hit mirandola, italy: A field report," in *IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, 2012.

[2] M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, M. W. Achtelik, and R. Siegwart, "The euroc micro aerial vehicle datasets," *The International Journal of Robotics Research (IJRR)*, vol. 35, no. 10, pp. 1157–1163, 2016.

[3] V. Usenko, N. Demmel, D. Schubert, J. Stückler, and D. Cremers, "Visual-inertial mapping with non-linear factor recovery," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 422–429, 2020.

[4] C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. M. Montiel, and J. D. Tardós, "Orb-slam3: An accurate open-source library for visual, visual–inertial, and multimap slam," *IEEE Transactions on Robotics*, vol. 37, no. 6, pp. 1874–1890, 2021.

[5] A. Rosinol, M. Abate, Y. Chang, and L. Carlone, "Kimera: an open-source library for real-time metric-semantic localization and mapping," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2020, pp. 1689–1696.

[6] M. Bloesch, J. Czarnowski, R. Clark, S. Leutenegger, and A. J. Davison, "Codeslam—learning a compact, optimisable representation for dense visual slam," in *IEEE conference on computer vision and pattern recognition*, 2018, pp. 2560–2568.

[7] J. Czarnowski, T. Laidlow, R. Clark, and A. J. Davison, "Deepfactors: Real-time probabilistic dense monocular slam," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 721–728, 2020.

[8] H. Matsuki, R. Scona, J. Czarnowski, and A. J. Davison, "Codemapping: Real-time dense mapping for sparse slam using compact scene representations," *IEEE Robotics and Automation Letters*, vol. 6, no. 4, pp. 7105–7112, 2021.

[9] R. Garg, V. K. Bg, G. Carneiro, and I. Reid, "Unsupervised cnn for single view depth estimation: Geometry to the rescue," in *European conference on computer vision*. Springer, 2016, pp. 740–756.

[10] T. Zhou, M. Brown, N. Snavely, and D. G. Lowe, "Unsupervised learning of depth and ego-motion from video," in *IEEE conference on computer vision and pattern recognition (CVPR)*, 2017, pp. 1851–1858.

[11] 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 *IEEE conference on computer vision and pattern recognition (CVPR)*, 2017, pp. 5038–5047.

[12] C. Godard, O. Mac Aodha, and G. J. Brostow, "Unsupervised monocular depth estimation with left-right consistency," in *IEEE conference on computer vision and pattern recognition (CVPR)*, 2017, pp. 270–279.

[13] V. Guizilini, R. Ambrus, S. Pillai, A. Raventos, and A. Gaidon, "3d packing for self-supervised monocular depth estimation," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2020, pp. 2485–2494.

[14] T. R. Institute, "Camviz," https://github.com/TRI-ML/camviz, 2021.

[15] D. Eigen, C. Puhrsch, and R. Fergus, "Depth map prediction from a single image using a multi-scale deep network," *Advances in neural information processing systems*, vol. 27, p. 2366–2374, 2014.

[16] A. Rosinol, J. J. Leonard, and L. Carlone, "Probabilistic volumetric fusion for dense monocular slam," *arXiv preprint arXiv:2210.01276*, 2022.

[17] H. Oleynikova, Z. Taylor, M. Fehr, R. Siegwart, and J. Nieto, "Voxblox: Incremental 3d euclidean signed distance fields for on-board mav planning," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017, pp. 1366–1373.

[18] F. Ma and S. Karaman, "Sparse-to-dense: Depth prediction from sparse depth samples and a single image," in *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 4796–4803.

[19] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[20] M. Helmberger, K. Morin, N. Kumar, D. Wang, Y. Yue, G. Cioffi, and D. Scaramuzza, "The hilti slam challenge dataset," 2021.

[21] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of rgb-d slam systems," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2012, pp. 573–580.

[22] E. P. Örnek, S. Mudgal, J. Wald, Y. Wang, N. Navab, and F. Tombari, "From 2d to 3d: Re-thinking benchmarking of monocular depth prediction," *arXiv preprint arXiv:2203.08122*, 2022.