Exploring the Impacts from Datasets to Monocular Depth Estimation (MDE) Models with MineNavi

Wang Xiangtong, Liang Binbin, Yang Menglong, and Li Wei,

Abstract—Current computer vision tasks based on deep learning require a huge amount of data with annotations for model training or testing, especially in some dense estimation tasks, such as optical flow segmentation and depth estimation. In practice, manual labeling for dense estimation tasks is very difficult or even impossible, and the scenes of the dataset are often restricted to a small range, which dramatically limits the development of the community. To overcome this deficiency, we propose a synthetic dataset generation method to obtain the expandable dataset without burdensome manual workforce. By this method, we construct a dataset called MineNavi containing video footages from first-perspective-view of the aircraft matched with accurate ground truth for depth estimation in aircraft navigation application. We also provide quantitative experiments to prove that pre-training via our MineNavi dataset can improve the performance of depth estimation model and speed up the convergence of the model on real scene data. Since the synthetic dataset has a similar effect to the real-world dataset in the training process of deep model, we also provide additional experiments with monocular depth estimation method to demonstrate the impact of various factors in our dataset such as lighting conditions and motion mode.

Index Terms—MineNavi, Synthetic Dataset, Monocular Depth Estimation

1 INTRODUCTION

In recent years, the machine learning based depth estimation methods, which heavily rely on the labeled dataset, have achieved satisfying performance. However, the scarcity of available labeled data, high costs of data acquisition and annotation, limit the quantity and variety of existing deep learning methods. Although the problem of data shortage can be partly solved by unsupervised learning methods with only sparse or even no annotated data, the ground-truth are still needed in experiments for evaluating or testing the generalization performance of the model. Thus, it is still of great significance to obtain a sufficient amount of images with accurate and dense depth information. The common data acquisition method in real world is not feasible for the depth estimation, especially for aircraft visual navigation because humans cannot manually label a pixel-wise annotation. Building a virtual world to generate synthetic datasets as the intermediate domain with the help of digital simulation technology may be the most feasible way for data generation and labeling at current stage. Since the newly released synthetic datasets[1], [2], [3], [4] are not flexible enough to suit for different needs, e.g., fixed resolution, limited scenes, low data diversity and huge volume, etc, it is difficult to apply them to the dense estimation task in the large scale environement, especially for the depth estimation in aircraft navigation. Therefore, in this paper, we propose a simple and expandable synthetic dataset generation method, and construct a custom dataset, which is called as MineNavi (Figure 1). This dataset generation method can not only solve the problem of high cost of real-world data acquisition, but also can narrow the gap between the training domain and the target domain by customizing the synthetic scene that similar with the target domain. Besides, different with conventional studies that adjust models in a fixed dataset to make them close to or superior to the state-of-the-art methods under certain evaluation metrics, we analyze the influences of the changes in datasets on the models. It is very significant because it can not only verify the generalization capabilities of the models to the environment, but also give guidance to construct real-world datasets. In addition, to explore the impact of the various dataset factors on depth estimation models, our constructed MineNavi dataset contains the dense depth information.
maps and surface normal vectors of objects. It will help us to observe the performance of depth estimation model under different factors of the dataset, such as the ego-motion camera, lighting and motion patterns, etc. Our experiments show that these variations on training sets may significantly affect the performance of the models.

Our contributions are as follows: firstly, we propose an open synthetic dataset generation method and construct MineNavi for the large-scale depth estimation applications. Secondly, we design experiments to report the performance of the baseline models pre-trained on the MineNavi dataset, and reveal the influence of various factors in datasets on depth estimation models. The source code is available.

2 RELATED WORK
2.1 Synthetic Datasets
Using synthetic data has a long history of application in computer vision community, it also is a significant data augmentation strategy for data driven models. On one hand, synthesis datasets solves the privacy issues involved in the data collection process partially. On the other hand, the low-level dense prediction tasks, such as optical flow or depth estimation, manual annotation is not feasible or even impossible, and synthetic datasets may be the only choice. It is a better idea to use existing scenes from an open-source 3D movie to render artificial annotated data for visual task. Using existing game scenes as data source is an ideal alternative, which decrease the cost of manual 3D modeling of scene and realize capturing the data in arbitrary perspectives. Building datasets through open source games can openly and freely allow users to perform differentiated processing for specific vision task and generate the datasets they need, and through rich plug-ins, the cost and difficulty of building data sets can be further reduced to a certain extent.

2.2 Bridging the Reality Gap
Models trained purely on synthetic data often suffer limited generalization caused by domain gap between two type of datasets. Therefore, the utilization and theoretical analysis of the synthetic data set are also very necessary. Domain Randomization (DR) is one of the most promising approaches to make straight-forward transfer learning from synthetic data to real world data. introduced structured domain randomization (SDR) to detection task, which imposes structure onto domain randomization (DR) in order to provide context, they have also shown that pre-training on SDR improves results from real data. represents the composition of a 3D scene with a scene graph and a probabilistic scene grammar, a common representation in computer graphics. Present works have also devoted significant study to domain adaptation (DA), e.g., the problem of adapting models trained on source domain to a previously unseen target domain. In our proposed MineNavi dataset, we implement DR through multiple lighting conditions rendering, switch shader in the same scene and scene replacement of the same path etc. Due to there is a little overlap between ImageNet and the dataset applied in large-scale depth estimation, MineNavi dataset can be ruled as a intermediate domain for multi-step domain adaptation.

2.3 Datasets Strategy on MDE Training
Although there is a domain gap between synthetic data and real data, most work is based on an assumption that the model trained on real data will perform some results that can be reproduced by training on synthetic data, which leads to the use of synthetic data as a proxy to explore the influence of factors in real data on the model. analyzed the multi-target detection task on different data with different illumination caused by weather conditions. analyzed the influence of background texture and lighting conditions on the task of detecting key points of the object, and concludes that more realistic lighting conditions are beneficial to the performance of the neural network in the task. detailed analysis of the virtual dataset in the optical flow estimation task not only on the impact of lighting, but also included the shape movement and texture of the object. While analysis on above datasets are fairly sufficient, what limits their performance is the lack of the the diversity among the benchmark datasets. To this end, makes a synthetic dataset based on CMU Visual Localization dataset and conducts extensive experimental evaluation on the proposed dataset with several learning-based algorithms. introduced a strategy that leverage the mixed datasets im multiple scene for MDE’s training. Collecting Large-scale datasets from the Internet is also a way to increase the data diversity, but it require a huge manual labor for data pre- and post-processing.

In this article, we analyze the depth estimation task of weather and lighting conditions, and focus on the influence of camera self-motion, texture and imaging quality on unsupervised monocular depth estimation. To the best of our knowledge, some factors that controlling data source has not been explored before in this context.

3 MineNavi: A Synthetic Dataset of Large Scale Scenes
Using MineCraft to construct a dataset is not a novel idea for computer vision community, we here to utilize it for depth estimation in aircraft visual navigation application. Our data generation method contains four steps: map loading, camera moving path setting, shader and lighting conditions setting, and ground-truth acquisition. shows the pipeline of data generation process.

Not only the environment features, such as the scene structure and lighting condition, affect the performance of depth estimation models, but also the particular dynamical parameters, such as moving targets in the environment and ego-motion of the aircraft, play important roles in benchmark datasets for models’ training and evaluation. Accordingly, each image frame of the dataset can be parameterized as:

\[ I [M, P^*(n), s, L(t, w)] \]  

(1)

where \( P^*(n) \) represents the 6 DoF camera motion paths, \( n \) is the quantified timestamp of the path, \( M \) is the map of the

1. https://github.com/xdr940/MineNavi
scene, $s$ is the shader that renders the world, $L(t, w)$ is the lighting condition, $t$ is the time in a day, and $w$ indicates the weather conditions.

### 3.1 Scene Construction

Although a lot of work\cite{2, 6} build the scenes based on 3D modeling software such as Blender and Maya, the construction of large-scale 3D scenes is still a relatively time-consuming and laborious task. Besides, the limited scenes diversity will lead to the over-fitting situation of the models. MineCraft community\cite{23} has extremely rich scene maps and users can freely build the required scene to generate specific dataset. Since the aircraft navigation is always involved in the large-scale scenes, the negative effects of the jagged features of objects in MineCraft can be ignored.

In order to increase the diversity of data, we use different shaders and lighting conditions. MineNavi dataset cooperates with the time and weather system according to different light and shadow styles to generate multiple style data. The construction in MineNavi based on the block is very simple and flexible. In order to build a more refined scene, users can use plug-ins to adjust the size of the block to achieve more complex objects (See Fig.3).

### 3.2 Camera Paths Setting

Base on previous study\cite{24, 25, 26}, we have found that the unsupervised monocular depth estimation methods are very sensitive to the camera motion in the training. Therefore, we develop different camera paths and generates corresponding datasets for experiments. Unlike lighting and other factors that can be quantified as a scalar, a moving camera has 6 continuous degrees of freedom. Therefore, for a training triplet, we propose a quantitative scalar $\lambda$, i.e., quasi-axis rate to generate datasets of the motion paths according to $\lambda$, and analyze the pros and cons of the data under different $\lambda$. The $\lambda$ can be formulated as:

\begin{align*}
\lambda(n) &= \frac{\bar{\phi}(n)\dot{t}(n)}{||\varphi(n)||||\dot{t}(n)||} \quad (2) \\
\bar{\phi}(n) &= \frac{\phi(n + 1) + \phi(t - 1)}{2} \quad (3) \\
\dot{t}(n) &= t(n + 1) - t(n - 1) \quad (4) \\
P^*(n) &= (\phi(n), t(n)) \quad (5)
\end{align*}

$\phi(n)$ is the rotation angle of camera visual axis at time $n$, calculated from $R \in SO(3)$, and $t(n) = [x, y, z]$ is the camera position vector at time $n$. When $\lambda = 0$, the camera moves parallel to the visual axis, when $\lambda = 1$, the camera moves perpendicular to the visual axis.
In this section, we mainly talk about the principle of MDE models, a analysis method of ego-motion video for MDEs and the relevant dataset for the training and evaluation of MDEs in our experiments.

4.1 Deep Models for Monocular Depth Estimation

In order to reconstruct 3D geometry from the ego-motion video by self-supervised method, the main supervision signal comes from the view synthesis between consecutive frames. We aim to infer the intermediate variable, e.g. predicted depth as precisely as possible through this approach. This is an ill-posed problem because a single 2D image may be produced from a large number of distinct 3D scenes, thus it can only be resolved using prior knowledge about the appearance and motion of image sequences.

Assume that \( I_t \) and \( I_s(I_s \in \{I_{t-1}, I_{t+1}\}) \) are two consecutive frames from an unlabeled video, we want to establish the dense pixel correspondence between the two frames. Let \( p_i \) denotes the 2D homogeneous coordinate of a pixel in frame \( I_t \) and \( K \) denotes the intrinsic camera matrix. We can compute the corresponding point of \( p_i \) in frame \( I_s \) using the following equation,

\[
p_i = K \hat{T}_{t \rightarrow s} \hat{D}_{p_i} K^{-1} p_i
\]

where \( \hat{T} \) is the predicted depth map, and \( T_{t \rightarrow s} \) is the relative camera pose. We can get the synthesized image \( \hat{I}_s \), \( I_s = I_s(p_s) \) Thus, the photometric error can be formulated as:

\[
\mathcal{L}_p = \sum_{s \in \{t-1,t, t+1\}} \sum_p \rho(I_s(p), \hat{I}_s(p))
\]

where \( \rho(\cdot) \) is a function to measure the difference of pixels value in the two images. We use a combination of SSIM and L1 term as our photometric error function which is formulated as:

\[
\rho(I, \hat{I}) = \alpha \frac{1 - SSIM(I, \hat{I})}{2} + (1 - \alpha) ||I - \hat{I}||
\]

It is robust to illumination changes in a real-world scenario and we set \( \alpha = 0.85 \). In general, the photometric loss \( \mathcal{L}_p = \rho(I, \hat{I}) \).

However, the supervision signal just based on view synthesis is not informative enough in low-texture or homogeneous regions of the scene due to its ambiguities. Thus, additional regularization is required to learn reasonable depth prediction. A common strategy proposed by \[31\], \[32\], and it is formulated as:

\[
L_s = \sum_{Pr} \sum_{d \in x,y} ||\nabla_d \hat{D}(p_i)||_1 e^{-\alpha ||\nabla_d I_s(p_i)||}
\]

where \( \nabla \) is the first derivative along the spatial direction. \( L_s \) ensures that the edges in depth map are guided by the images. And overall objective function can be formulated as follows:

\[
\mathcal{L}_{final}^l = \mathcal{L}_p^l + \lambda L_s^l
\]

where \( l \) stands for multi-scale approach.

Unsupervised depth estimation includes monocular methods\[31\], \[32\], \[26\], \[25\], \[24\], \[33\], \[34\] usually contain
Fig. 5. For spatiotemporal feature learning, we build a 3d encoder and apply it into monodepth2 to build monodepth2-3D (Left) and monodepth2-3Ds (Right).

We have found that during the training, the depth estimation models are very sensitive to the distribution of SHPH. The reason behind is that if the photometric error value in the images is evenly distributed, the model can perform more stable during gradient back-propagation, thereby improving the training efficiency. Accordingly, the histogram of each photometric error map in the training sequence should be kept as even as possible, which also provides a basis for choosing a suitable training sequence.

4.3 Datasets

The depth estimation method that based on SfM is not suitable for some datasets due to the absence of successive image, large scene scale, or applicable camera motion. Therefore in our experiments, we use the follow datasets to make comparison and evaluate the feasibility of the dataset generation method.

- **KITTI** is widely used in monocular depth learning. In this study, we use KITTI Zhou split that contains image triplets, which static frames with an average optical flow of less than 1 pixel are removed. In our experiments, 40k triplets are used for training and 4k triplets for testing.

- **FPV.** To test the performance of depth estimation models in the large-scale scenes, we selected the video data of the first person view (FPV) of the aircraft as another experimental dataset. These datasets cover most of the field scenes, including mountains, woods, lakes, snow mountains and cities. Different with previous uav datasets which captured by simple or even static camera motion, this dataset has diverse moving pattern that suitable for depth estimation model based on SfM. We divide the data into two datasets with the train set volume is 36k and 4k validation set.

- **MineNavi** datasets contain data samples according to different shaders, lighting conditions, camera motions, and image qualities (clear, low blur, middle blur, high blur). We set the field of views of the camera as 70 degree. The resolutions of the captured images are 1600 x 1200 pixels.
images are $1024 \times 768$. The matched depth map ranges from $0.1m$ to $576m$. Finally, we set more than 400 paths around a virtual scene of more than 16 square kilometers, and collected a total of more than 50,000 image sequences. The image scenes include various weather and lighting conditions, with matching depth values.

4.4 Evaluation Metrics

For depth estimation, we evaluate depth estimation models using three accuracy metrics ($\delta_1, \delta_2, \delta_3$) and four error metrics: absolute relative difference (AbsRel), square related difference (SeqRel), RMSE, log RMSE. Note that the accuracy metrics and AbsRel are scale-independent, i.e., the value does not change with the depth range of the dataset. And the others are scale-invariant, i.e., the results evaluated on the dataset with different depth ranges are in different order of magnitudes.

5 EXPERIMENTS

In this section, we verify the feasibility and credibility of the MineNavi dataset in the training of the depth estimation model, and explore the impact of dataset varieties on the unsupervised monocular depth estimation model. Thus, we will demonstrate that 1) the depth estimation model can improve generalization through pre-training on MineNavi. 2) it is desirable to exploit the influence of data to model caused by various factors of the dataset. We prepare monodepth2[35] and its two variants monodepth2-3D and monodepth2-3Ds as the test models on our proposed datasets. We also present using Sequential Heat-map of Photometric-error Histogram (SHPH) to verify weather an influence of data diversity on the performance, we collect two data samples: the first one contains 10k samples (MN-10K) rendered by high-sildurs shader with the sunny weather, and the second one contains 50K samples (MN-50K) rendered by three shaders (None, middle-sildurs, and high-sildurs) with various lighting conditions (morning, night, and sunny). For comparison, we also prepare the models that are trained from scratch and ImageNet[49]. Although the model pre-trained on ImageNet by classification task has structural difference compare with the model that trained on similar target task, it is still the most popular method in depth estimation task.

5.2 Generalization of MineNavi

We execute the models pre-training on MineNavi datasets with linear camera moving path. In order to evaluate the influence of data diversity on the performance, we collect two data samples: the first one contains 10k samples (MN-10K) rendered by high-sildurs shader with the sunny weather, and the second one contains 50K samples (MN-50K) rendered by three shaders (None, middle-sildurs, and high-sildurs) with various lighting conditions (morning, night, and sunny). For comparison, we also prepare the models that are trained from scratch and ImageNet[49]. Although the model pre-trained on ImageNet by classification task has structural difference compare with the model that trained on similar target task, it is still the most popular method in depth estimation task.

5.2.1 Fine-tune on KITTI

We conduct fine-tuning of the monodepth2 and monodepth2-3Ds pre-trained from above datasets on KITTI dataset for 10 epochs.

From TABLE 2 it can be seen that the performance of monodepth2 and monodepth2-3D pre-trained on ImageNet is better than that pre-trained on MN-10k and scratch, but worse than MN-50k. The MineNavi has a strong generalization capability compared to the KITTI. As mentioned before, MN-10k and MN-50k are only different in lighting condition and data volume. Therefore, the diversity of lighting conditions effectively improves the generalization capabilities of the models. Compared with the other datasets, the model of monodepth2-3Ds pre-trained on ImageNet has the better performance. This is mainly because excessive noises in KITTI, e.g., the moving objects deteriorate the robustness of the network performance of the shared encoder, but the large amount of data of ImageNet can make the model more robust[50]. Note that although MineNavi dataset is much smaller the ImageNet, it has competitive performance with ImageNet in depth estimation model training. We also provide qualitative result on Fig[10] and it shows the value in generalization of our MineNavi dataset.

5.2.2 Fine-tune on FPV

Since there is no ground-truth in the FPV dataset, we have to compute the distances between the models in different domains based on the loss value[40]. The closer the migration distance is, the better the pre-training dataset can be generalized to the target domain.

Compared the losses curve among of different pre-trained models that are fine-tuned on FPV in Fig[10], MineNavi pre-trained models converge faster than the others. The reason behind probably comes from that the MineNavi dataset is closer to the FPV dataset than ImageNet in terms...
of environment scenes. What’s more, compared with the ImageNet pre-trained model through the task of objects detection, the MineNavi pre-trained model through the task of depth estimation has learned geometric representation during the pre-training, which leads the model converge faster when the target task has structural similarity with source task. Note that, with the continuous expansion of the dataset, MineNavi can realize a more satisfied performance.

5.3 Factors that Affect the Train of MDE

Due to the expandable characteristics of the MineNavi dataset, we can easily generate customized datasets with different variation factors to avoid the over-fitting. It also a helpful way to discover the impacts of factors of datasets on the models. Thus we conduct experiments to explore how the factors in dataset can affect MDE model, including the shader, lighting conditions, motion blur, ego-motion and velocity of training image sequence.

5.3.1 Impact of Shaders

The MineNavi dataset can generate the rendered image sequences sampled on the same path through different shaders, which allows us to quantitatively evaluate the impacts of the synthetic world design and the quality of other rendering parameters on the algorithm performance. We apply Sildurs to adjust the image rendering quality and build three datasets Raw, middle-sildurs and high-sildurs. All of them are captured in an identical scene with linear camera motion and collected for 10k samples. The only differences among them are shader setting: Raw is rendered by no

4. https://sildurs-shaders.github.io/
TABLE 1
Quantitative results in MN datasets. First six model are trained in supervised and the rest are unsupervised (monodepth2).

| Model         | Dataset       | AbsRel | SqRel | RMS  | RMSlog | <1.25 | <1.25² | <1.25³ |
|---------------|---------------|--------|-------|------|--------|-------|--------|--------|
| ResNet18      | MN-2K(middle) | 0.198  | 1.318 | 5.679| 0.268  | 0.731 | 0.923  | 0.972  |
| 3D-ResNet18   | MN-2K(middle) | 0.194  | 1.197 | 5.453| 0.259  | 0.749 | 0.932  | 0.973  |
| ResNet18      | MN-2K(high)  | 0.207  | 1.479 | 5.832| 0.274  | 0.721 | 0.920  | 0.968  |
| 3D-ResNet18   | MN-2K(high)  | 0.181  | 1.159 | 5.372| 0.250  | 0.761 | 0.935  | 0.976  |
| ResNet18      | MN-10K(high) | 0.142  | 0.707 | 4.455| 0.191  | 0.846 | 0.961  | 0.986  |
| 3D-ResNet18   | MN-10K(high) | 0.129  | 0.675 | 4.384| 0.171  | 0.861 | 0.966  | 0.987  |
| monodepth2    | MN-2K(high)  | 0.212  | 1.426 | 7.054| 0.295  | 0.706 | 0.902  | 0.959  |
| monodepth2-3D | MN-2K(high)  | 0.245  | 1.833 | 5.919| 0.273  | 0.750 | 0.925  | 0.965  |
| monodepth2    | MN-10K(high) | 0.170  | 0.974 | 5.782| 0.211  | 0.798 | 0.941  | 0.977  |
| monodepth2-3D | MN-10K(high) | 0.165  | 1.170 | 5.965| 0.211  | 0.800 | 0.942  | 0.977  |
| monodepth2-3Ds| MN-10K(high) | 0.160  | 0.991 | 5.899| 0.208  | 0.809 | 0.943  | 0.977  |

Fig. 9. Fine-tuning curves of three test models on KITTI. Solid curves denote accuracy ($\delta \leq 1.25$) metric of depth estimation and dash curves denote training loss.

TABLE 2
Quantity Results in KITTI with different pre-trained datasets. The best result are bolded and the second best are underlined.

| Models       | Pre-trained Datasets | Error ↓ | Accuracy ↑ |
|--------------|----------------------|---------|-------------|
| monodepth2   | scratch              | 0.144   | 0.864       |
|              | ImageNet             | 0.135   | 0.864       |
|              | MN-10k               | 0.138   | 0.864       |
|              | MN-50k               | 0.130   | 0.864       |
| monodepth2-3D| scratch             | 0.170   | 0.864       |
|              | ImageNet             | 0.163   | 0.864       |
|              | MN-10k               | 0.167   | 0.864       |
|              | MN-50k               | 0.153   | 0.864       |
| monodepth2-3Ds| scratch            | 0.161   | 0.864       |
|              | ImageNet             | 0.158   | 0.864       |
|              | MN-10k               | 0.165   | 0.864       |
|              | MN-50k               | 0.158   | 0.864       |

shader, middle-sildurs uses sildurs with middle performance and high-sildurs uses high-performance shader. We apply random initial weights encoder to monodepth2 and train it on above three datasets. We use cross-evaluation on each trained model, i.e., evaluate every model on all datasets. The qualitative results are shown in TABLE 3.

It shows that as the training scenes rendered gradually improve, the performance of the depth estimation model improves consequently. Besides, compared with a model that is trained on less-texture data and tested on rendered data, the model that is trained on rendered data and tested on less-texture data brings a worse result. It is consistent with the fact that the rendering performance will promote the robust of the model during the training.

TABLE 3
The performance of the accuracy of the generated dataset under different rendering shaders. Here AbsRel and $\delta \leq 1.25$ are used as error and accuracy indicators. The best result in each row has been underlined and the optimal result has been bolded.

| Train Sets (AbsRel, $\delta$) | Test Datasets |
|------------------------------|---------------|
| raw                          | scratch       | ImageNet     | MN-10k       | MN-50k       | raw      |
| scratch                      | 0.143         | 0.326        | 0.511        | 0.314        | 0.777    |
| middle-sildurs               | 0.143         | 0.326        | 0.511        | 0.314        | 0.777    |
| high-sildurs                 | 0.143         | 0.326        | 0.511        | 0.314        | 0.777    |
5.3.2 Lighting Conditions

Previous study [35], [52] show that during the depth estimation model training, the low-texture areas caused by insufficient lighting or overexposure will produce problem pixels in depth estimation.

To further explore the impact of lighting conditions in data to the depth estimation model, we apply the models with random initial weights and train them on five datasets (See Fig. [11]) under different lighting conditions: morning, noon, afternoon, night and rainy day. Quantified results on APE are shown in Fig. [12]. We can observe that in the lighting conditions at morning and noon, three test models achieve similar results. However, as the lighting in training data is getting dim (afternoon, night, rainy), three models deteriorate significantly. This can be attributed primarily to that the adequate lighting makes the color between pixels more diverse, and the error map is close to the uniform distribution. Note that at the time of afternoon, the models performance dropped dramatically, even worse than night that with dimmer lighting condition, we suspect that the reason behind this is there are too much problematic pixels in captured images caused by lens flare, which strongest in afternoon compared with the other lighting conditions. SHPH results on the collected sequences under different lighting conditions and different camera moving paths are shown in the row3 of Fig. [11]. It can be clearly seen that the clear lighting conditions bring the even distribution of the SHPH.
TABLE 4  
Quantity Results in MineNavi with different lights

| Models       | dataset | Error ↓  | Accuracy ↑ |
|--------------|---------|----------|------------|
|              |         | AbsRel   | SqRel      | RMS        | RMSlog     | < 1.25    | < 1.25<25 | < 1.25<25 |
| monodepth2   | morning | 0.276    | 8.491     | 36.672     | 0.289      | 0.694     | 0.885     | 0.947     |
|              | noon    | 0.276    | 11.284    | 49.412     | 0.321      | 0.665     | 0.858     | 0.931     |
|              | afternoon | 0.463   | 26.176    | 67.480     | 0.534      | 0.375     | 0.647     | 0.803     |
|              | night   | 0.391    | 18.674    | 57.397     | 0.432      | 0.470     | 0.748     | 0.875     |
|              | rain    | 0.547    | 30.597    | 54.880     | 0.520      | 0.395     | 0.671     | 0.818     |
| monodepth2-3D| morning | 0.243    | 7.469     | 36.867     | 0.263      | 0.737     | 0.903     | 0.957     |
|              | noon    | 0.297    | 11.108    | 46.373     | 0.324      | 0.622     | 0.852     | 0.936     |
|              | afternoon | 0.374  | 13.967    | 45.221     | 0.391      | 0.478     | 0.779     | 0.910     |
|              | night   | 0.359    | 15.065    | 49.514     | 0.392      | 0.490     | 0.780     | 0.907     |
|              | rain    | 0.481    | 24.776    | 57.464     | 0.492      | 0.386     | 0.673     | 0.836     |
| monodepth2-3Ds | morning | 0.277    | 10.086    | 41.467     | 0.290      | 0.697     | 0.883     | 0.950     |
|              | noon    | 0.290    | 11.167    | 46.460     | 0.315      | 0.656     | 0.865     | 0.936     |
|              | afternoon | 0.311  | 11.901    | 43.584     | 0.322      | 0.618     | 0.859     | 0.942     |
|              | night   | 0.351    | 14.853    | 52.417     | 0.384      | 0.507     | 0.790     | 0.906     |
|              | rain    | 0.479    | 26.123    | 67.677     | 0.535      | 0.349     | 0.621     | 0.793     |

Table 6 shows the performance of two variants of monodepth2 in MineNavi datasets with different motion blur. It can be seen that the two models are trained on the motion-blurred dataset, and the performance is significantly better than the dataset without blurred.

5.3.3 Impact of Motion Blur

The motions of cameras will also affect the stability of the SHPH. As shown in the Fig.13, it can be seen that the distribution of the photometric error map gradually even with the increase motion blur. In our experiment, four datasets with different motion blur are built. The quantified results of monodepth2 are shown in the TABLE5. The motion blur has a great impact on the SHPH, we suspect that it is an effective way to overcome the noise and introduce the robustness by adding a certain motion blur in sequences. This is reflected in SHPH that appropriate motion blur can make the SHPH more stable, which leads the view synthesis of depth estimation model easier (see Fig.13).

Besides, we also introduce vary lighting conditions into experiments. As shown in the Table 4, it can be seen that in the variant models of monodepth2, the darker the lighting conditions, the worse the performance, which is consistent with the results of the experiments. Note that at the time of afternoon, the models performance dropped dramatically, even worse than night that with dimmer lighting condition, We suspect that the reason behind this is there are too much problematic pixels in captured images caused by lens flare, which strongest in afternoon compared with the other lighting conditions. It also suggest that the scene fidelity of MineNavi.

Fig. 13. At close to the ground (<70cm) a histogram statistical result of the inter-frame error and its sequence photometric error heat map.

| Train Sets (AbsRel \( \delta_1 \)) | Test Datasets |
|-----------------------------------|---------------|
| None                              | low blur      |
| low blur                          | middle blur   |
| middle blur                       | high blur     |
| None                              | low blur      |
| low blur                          | middle blur   |
| middle blur                       | high blur     |
| None                              | low blur      |
| low blur                          | middle blur   |
| middle blur                       | high blur     |
| None                              | low blur      |
| low blur                          | middle blur   |
| middle blur                       | high blur     |

Table 7 shows the the performance of two variants of monodepth2 in MineNavi datasets with different lights. The best result in each row is underlined and the optimal result is bolded.
the difficulty of data collection in some dense estimation generation method can provide a solution to overcome the 6DoF paths of the camera’s ego-motion. This dataset, which includes a large-scale scene with low cost but infinite volume, including surface normals, depth, and semantic information, can greatly affect the performance of the model. It is essential because if the density of sampling camera is too low, the photometric differences between two adjacent frames are bigger, making the model difficult to train. Fig. 14 shows the qualitative results of the models that vary in velocity of training sequence and encoder.

### 5.3.5 Velocity of Training Sequence

We find that the training sequence vary in sample frequency can greatly affect the performance of the model. It is essential because if the density of sampling camera is too low, the photometric differences between two adjacent frames are bigger, making the model difficult to train. Fig. 14 shows the qualitative results of the models that vary in velocity of training sequence and encoder.

### 6 Conclusion

This paper proposes a method to construct a synthetic dataset, which includes a large-scale scene with low cost but infinite volume, including surface normals, depth, and the 6DoF paths of the camera’s ego-motion. This dataset generation method can provide a solution to overcome the difficulty of data collection in some dense estimation tasks. For depth estimation task in aircraft navigation, we construct several datasets. According to the experimental results, our proposed dataset generation method can perform as an intermediate domain for depth estimation. The data-to-model experiments reveal that future work should not only focus on the innovation of the models, but also pay more attention to the factors in the dataset that affect the models.

### References

[1] N. Mayer, E. Ilg, P. Hauser, P. Fischer, D. Cremers, A. Dosovitskiy, and T. Brox, “A large dataset to train convolutional networks for 3D object recognition,” in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 4040-4048.

[2] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black, “A naturalistic open source movie for optical flow evaluation,” in European Conf. on Computer Vision (ECCV), 2012.

[3] A. Antonini, W. Guerra, V. Murrali, T. Sayre-McCord, and S. Karaman, “The blackbird dataset: A large-scale dataset for uav perception in aggressive flight,” CoRR, 2018.

[4] Y. Zhang, S. Song, E. Yumer, M. Savva, J.-Y. Lee, H. Jin, and T. Funkhouser, “Physically-based rendering for indoor scene understanding using convolutional neural networks,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 5287-5295.

[5] G. Ros, L. Sellart, J. Materzynska, D. Vazquez, and A. M. Lopez, “The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes,” in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 3234-3243.

[6] A. Gaidon, Q. Wang, Y. Cabon, and E. Vugrinovich, “Virtualworlds as proxy for multi-object tracking analysis,” in 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.

[7] J. Tremblay, A. Prakash, D. Acuna, M. Brophy, V. Jampani, C. Anil, T. To, E. Cameracci, S. Boochoon, and S. Birchfield, “Training deep networks with synthetic data: Bridging the reality gap by domain randomization,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 4969-4977.

[8] W. Guerra, E. Tal, V. Murrali, G. Ryou, and S. Karaman, “Flightgoggles: Photorealistic sensor simulation for perception-driven robotics using photogrammetry and virtual reality,” in 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, 2019, pp. 6941-6948.

[9] A. Antonini, W. Guerra, V. Murrali, T. Sayre-McCord, and S. Karaman, “The blackbird dataset: A large-scale dataset for uav perception in aggressive flight,” in International Symposium on Experimental Robotics. Springer, 2018, pp. 130-139.

[10] Q. Wang, J. Cao, W. Lin, and Y. Yuan, “Learning from synthetic data for crowd counting in the wild,” in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 8198-8207.

[11] S. R. Richter, V. Vineet, S. Roth, and V. Kolkin, “Playing for data: Ground truth from computer games,” in European conference on computer vision. Springer, 2016, pp. 102-118.

[12] W. H. Guss, B. Houghton, N. Topin, P. Wang, C. Codel, M. Veloso, and R. Salakhutdinov, “MineNet: A large-scale dataset of minecraft demonstrations,” in JICAR, 2019.

[13] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel, “Domain randomization for transferring deep neural networks from simulation to the real world,” in 2017 IEEE/RSJ international conference on intelligent robots and systems (IROS). IEEE, 2017, pp. 23-30.

[14] A. Prakash, S. Boochoon, M. Brophy, D. Acuna, E. Cameracci, G. State, O. Shapira, and S. Birchfield, “Structured domain randomization: Bridging the reality gap by context-aware synthetic data,” in 2019 International Conference on Robotics and Automation (ICRA). IEEE, 2019, pp. 7249-7255.

[15] A. Kar, A. Prakash, M.-Y. Liu, E. Cameracci, J. Yuan, M. Ruscina, D. Acuna, A. Torralba, and S. Fidler, “MetaSim: Learning to generate synthetic datasets,” in Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019, pp. 4551-4560.

[16] M. Wang and W. Deng, “Deep visual domain adaptation: A survey,” Neurocomputing, vol. 312, pp. 135-153, 2018.
J. Watson, O. Mac Aodha, V. Prisacariu, G. Brostow, and M. Firz.

Z. Yin and J. Shi, "Geonet: Unsupervised learning of dense depth, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020.

Y. Zou, Z. Luo, and J.-B. Huang, "Df-net: Unsupervised joint learning of depth and flow using cross-task consistency," in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 4521–4530.

MineCraftMaps, "minecraftmaps," https://www.minecraftmaps.com/11.

J. Bian, Z. Li, N. Wang, H. Zhan, C. Shen, M.-M. Cheng, and I. Reid, "Unsupervised scale-consistent depth and ego-motion learning from monocular video," Advances in neural information processing systems, vol. 32, pp. 35–45, 2019.

Y. Zou, Z. Luo, and J.-B. Huang, "Df-net: Unsupervised joint learning of depth and flow using cross-task consistency," in Proceedings of the European conference on computer vision (ECCV), 2018, pp. 56–53.

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 (CVPR), 2020.

Z. Yin and J. Shi, "Geonet: Unsupervised learning of dense depth, optical flow and camera pose," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 1983–1992.

J. Watson, O. Mac Aodha, V. Prisacariu, G. Brostow, and M. Firman, "The temporal opportunist: Self-supervised multi-frame monocular depth," in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 1164–1174.

M. Klingner, J.-A. Termøhlen, J. Mikolajczyk, and T. Fingscheidt, "Self-supervised monocular depth estimation: Solving the dynamic object problem by semantic guidance," in European Conference on Computer Vision. Springer, 2020, pp. 582–600.

Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," IEEE transactions on image processing, vol. 13, no. 4, pp. 600–612, 2004.

T. Zhou and M. Brown, "Unsupervised learning of depth and ego-motion from video," in CVPR, 2017.

C. Godard and M. Aodha, "Unsupervised monocular depth estimation with left-right consistency," in CVPR, 2017.

A. Ranjan, V. Janpani, L. Balles, K. Kim, D. Sun, J. Wulff, and M. J. Black, "Competitive collaboration: Joint unsupervised learning of depth, camera motion, optical flow and motion segmentation," in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 12240–12249.

Z. Yang, P. Wang, Y. Wang, W. Xu, and R. Nevatia, "Lego: Learning edge with geometry all at once by watching videos," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 225–234.

C. Godard, O. Mac Aodha, M. Firman, and G. Brostow, "Digging into self-supervised monocular depth estimation," in CVPR, 2019.

N. Li, F. Chang, and C. Liu, "Spatial-temporal cascade autoencoder for video anomaly detection in crowded scenes," IEEE Transactions on Multimedia, vol. 23, pp. 203–215, 2020.

J. Li, X. Liu, W. Zhang, M. Zhang, J. Song, and N. Sebe, "Spatial-temporal attention networks for action recognition and detection," IEEE Transactions on Multimedia, vol. 22, no. 11, pp. 2990–3001, 2020.

J. Hur and S. Roth, “Self-supervised multi-frame monocular scene flow,” in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 2684–2694.