Deep Learning for Visual Odometry (Deep Learning 2022 Course)

Vladislav Trifonov [1] **Hekmat Taherinejad** [1] **Prateek Rajput** [1] **Vladimir Chernyy** [1] **Anna Akhmatova** [1] **Timur Bayburin** [1]

Abstract

This paper tests pre-processed data with respect to deblurring and transparent object segmentation on traditional pipeline for Visuao Odometry ie. ORB SLAM 2 and compares with the unprocesses data on KITTI and TUM sample sequences. We also tested a well known Deep Neural network by introducing a new coordconv layer to preserve the spacial information in the pipeline with introduction of a variable learning rate. Transparent object removal for pre-processing is an untested research area and a tricky task in itself but in combination with Visual Odometry has the potential to improve the final path greatly.

Video link: https://youtu.be/deKCT6b3C9M

Github repo: https://github.com/scalyvladimir/vo_exploring

1. Introduction

In robotics and computer vision, visual odometry is the process of determining the position and orientation of a robot by analyzing the associated camera images. It has been used in a wide variety of robotic applications, such as on the Mars Exploration Rovers or widely used in autonomous robots and self-driving cars. Current progress in robotics requires robust and quick way for positioning and localization of autonomous devices.

For a long time the problem of localization was based on geometrical approach in which linear systems were solved to get camera positions. In the last couple of years deep learning algorithm based on convolutional neural networks showed fairly good results in task of localization. Some modern approaches that combines learning and geometrical calculations achieve same results as current standard approach (ORB SLAM).

One of the challenges that indoor mobile robots face for visual odometry is processing the feature points that are located on transparent objects (e.g. glasses). If we can mitigate this effect we may be able to reach higher accuracy and better performance.

Another problem, that appear in visual odometry task is that datasets consist of blurry pictures. This problem comes naturally from the way how these pictures are captured (e.g. from moving cars and robots).

In this project we suggest to use pretrained networks to resolve problem mentioned above. We assume, that if we will pass pictures, in which transparent objects will be masked and/or pictures are deblurred, the results of localization will be improved, since it will be easier for ORB SLAM to capture key points.

An approach for visual odometry task which is based on deep learning rely on such dense maps as depth and/or optical flows. We suggest to use anther dense map which initially was introduced by Uber AI Labs group - CoordConv (Liu et al., 2018). Then, the model with the new input is fine-tuned with different ways to initializing weights for addition channels and modes of fine-tunning.

The main contributions of this project are as follows:

- Idea of masking transparent objects to enhance visual odometry task was introduced and checked.
- Idea of a deblurring input images for ORB SLAM was introduced and checked.
- The dense map based on pixel coordinates (CoordConv) was introduced for visual odometry task.
- Different modes for fine-tuning and weights initialization were performed with comparison of results.

[1]Skolkovo Institute of Science and Technology, Moscow, Russia. Correspondence to: Vladislav Trifonov <vladislav.trifonov@skoltech.ru>.

Final Projects of the Deep Learning 2022 Course, Skoltech, Moscow, Russian Federation, 2022.

2. Literature Review

2.1. Traditional Visual Odometry

I. Epipolar Geometry: Given an image pair, (I1,I2), the basic method for estimating the relative camera pose is solving fundamental oressential matrix, ***E***. When 2D-2D pixel correspondences (***p1, p2***) between the image pair are formed, epipolar constraint is employed for solving the essential matrix. Hence, relative pose, [***R,t***], can be recovered (Nistér, 2004) (Zhang, 1998), (Hartley, 1997).

$$p_2^T K^{-T} E K^{-1} p_1 = 0, \tag{1}$$

where $E = [t]R$; K is the camera intrinsics. Typically, the 2D-2D pixel correspondences are formed either by extracting and matching salient feature points in the images, or by computing optical flow. However, solving essential matrix for camera pose has some well-known issues.

- Scale ambiguity: translation recovered from essential matrix is up-to-scale.
- Pure rotation issue: recovering **R** becomes unsolvable if the camera motion is pure rotation.
- Unstable solution: the solution is unstable if the camera translation is small.

II. Perspective-n-Point (PnP): Pnp is a classic method for solving camera pose given 3D-2D correspondences. Suppose the observed 3D points of view 1 and the observed projection in view 2 (X1,p2) are given, PnP can be deployed for solving the camera pose by minimizing the reprojection error:

$$e = \sum_i ||K(RX_{(1,i)} + t) - p_{(2,i)}||_2 \tag{2}$$

In order to establish the 3D-2D correspondences, we need to (1) estimate the 3D scene structure, (2) match 3D points to 2D pixels by matching features.

2.2. Depth-Pose Learning without PoseNet

The central idea of existing self-supervised depth-pose learning methods is to learn two separated networks on the estimation of monocular depth and relative pose by enforcing geometric constraints on image pairs. This class of methods assume a consistent scale of depth and pose across all images. Such strong assumption still makes the learning problem difficult and leads to severely degraded performance, especially in long-sequence visual odometry applications and indoor environments, where the changes of relative pose across sequences are significantly remarkable (Zhao et al., 2020).

The scale-inconsistent issue naturally exists because the scales of the estimated depth and pose from neural networks are hard to measure. Also, the photometric error on the image plane supervises the depth in an implicit manner, which could suffer from data noise when large textureless regions exist. Furthermore, similar to two recent findings (Sattler et al., 2019), (Zhou et al., 2020) that CNN-based absolute pose estimation is difficult to generalize beyond image retrieval.

The authors of (Zhao et al., 2020) disentangles scale consistency at both training and inference. Instead of relying on CNN-based relative pose estimation (PoseNet), authors first predict optical flow and solve the fundamental matrix from the dense flow correspondence, thereby recovering relative camera pose. Then, authors sample over the inlier regions and use a differentiable triangulation module to reconstruct an up-to-scale 3D structure. Finally, depth error is directly computed after a scale adaptation from the predicted depth to the triangulated structure and reprojection error on depth and flow is measured to further enforce end-to-end joint training.

This approach borrows the advantage of traditional two-view geometry to acquire more direct, accurate and robust depth supervision in a self-supervised end-to-end manner,

Moreover, because this relative pose is directly solved from the optical flow, it simplifies the learning process and do not require the knowledge of correspondence to be learned from the PoseNet architecture, enabling the system to have better generalization ability.

2.3. Transformers

The authors of (Kuo et al., 2020) propose a dynamic attention-based visual odometry framework named (DAVO), It dynamically adjusts the attention weights on different semantic categories for different motion scenarios based on optical flow maps. These weighted semantic categories can then be used to generate attention maps that highlight the relative importance of different semantic regions in input frames for pose estimation. They performed experiments on the KITTI Visual Odometry and SLAM benchmarks to inspect the impacts of the dynamically adjusted weights on the accuracy of the evaluated trajectories.

I is the RGB input received at time t, The architecture generates optical flow maps $(F_{t\to t}, F_{t+1\to t})$ by predicting the optical flow between consecutive input frames $(I_t \to I_t, I_{t+1} \to I_t)$, respectively. The defined Attention Module then highlights their order of significance and for each I_{t+1} produces an attention map A_{t+1}. This attention map is then applied to I_{t+1} and $F_{t+1\to t}$ by pixel-wise mul-

tiplication to generate a weighted RGB frame I'_{t+1} and a weighted flow map $F'_{t+1\to t}$, respectively.

$$I'_{t+1} = A_{t+1} \odot I_{t+1} \tag{3}$$

$$F'_{t+1\to t} = A_{t+1} \odot F_{t+1\to t}, \tag{4}$$

where $\odot$ denotes pixel-wise multiplication. The network then takes $I_t, F_{t\to t}, I'_{t+1}$, and $F'_{t+1\to t}$ as its inputs, encodes them by an eight-layer DCNN, and generates a three degree of freedom (3-DoF) translational motion estimation $\hat{\rho}_{t\to t+1}$ and another 3 -DoF rotational motion estimation $\hat{\varphi}_{t\to t+1}$ by two separate DCNN branches named TransNN and RotNN, respectively. The predicted relative pose $\hat{\chi}_{t\to t+1}$ is given by:

$$\begin{aligned}\hat{\chi}_{t\to t+1} &= (\hat{\rho}_{t\to t+1}, \hat{\varphi}_{t\to t+1}) \\ &= \mathcal{P}\left(I_t \oplus F_{t\to t} \oplus I'_{t+1} \oplus F'_{t+1\to t}\right),\end{aligned} \tag{5}$$

where $\mathcal{P}(\cdot)$ denotes PoseNN, $\oplus$ represents the channel-wise concatenation operator, and $\hat{\rho}$ and $\hat{\varphi}$ correspond to the predicted translational and rotational motions, respectively. Note that $F_{t\to t}$ is a zero map consisting of two channels. Finally, the entire trajectory is generated according to $\hat{\chi}_{t\to t+1} = (\hat{\rho}_{t\to t+1}, \hat{\varphi}_{t\to t+1})$ collected at different frame timestamps. The framework leverages two pairs of frames (I_t, I_{t+1}) and (I_t, I_{t-1}) as its inputs to predict $\hat{\chi}_{t\to t+1}$ and $\hat{\chi}_{t\to t-1}$ during the training phase for improving the representation learning of $\hat{\rho}$ and $\hat{\varphi}$. During the evaluation phase, only a single pair (I_t, I_{t+1}) is used by this framework.

3. Data

We used classical dataset KITTI. It contains of a suite of vision tasks built using an autonomous driving platform. The full benchmark contains many tasks such as stereo, optical flow, visual odometry, etc. This dataset contains the object detection dataset, including the monocular images and bounding boxes. For our task we used KITTI odometry dataset (Geiger et al., 2012).

The network for transparent detection was pre-trained of Trans10k dataset (Xie et al., 2020). It is a large-scale dataset for transparent object segmentation, named Trans10K, consisting of 10,428 images of real scenarios with carefully manual annotations, which are 10 times larger than the existing datasets.

For experiments with indoor environment we used sequences from TUM dataset (Schubert et al., 2018).

4. Method Description

4.1. ORB SLAM2

ORB-SLAM 2 is a versatile and accurate SLAM solution for Monocular, Stereo and RGB-D cameras. It is able to compute in real-time the camera trajectory and a sparse 3D reconstruction of the scene in a wide variety of environments, ranging from small hand-held sequences of a desk to a car driven around several city blocks. It is able to close large loops and perform global relocalisation in real-time and from wide baselines. It includes an automatic and robust initialization from planar and non-planar scenes. This method is provided by examples to run the SLAM system in the KITTI dataset as stereo or monocular, in the TUM dataset as RGB-D or monocular, and in the EuRoC dataset as stereo or monocular. For the benefit of our project we used this method to run our experiments on Kitti and TUM monocular sequences. Figure 1 shows the scheme of ORB SLAM 2.

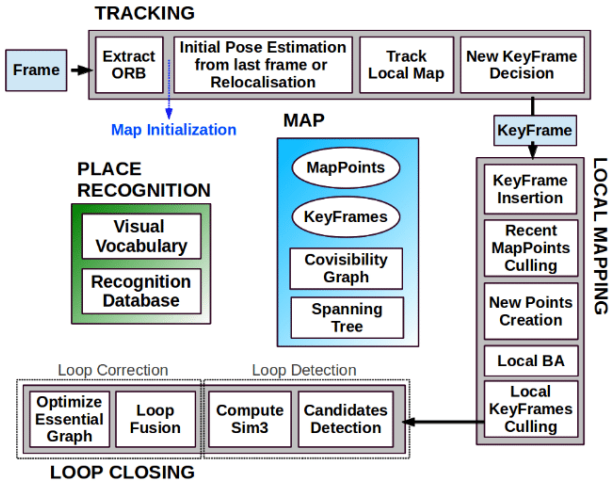

Figure 1. ORB SLAM 2 framework structure.

ORB-SLAM2 works on three tasks working simultaneously: tracking, local mapping & loop closing. In its tracking part, ORB-SLAM2 does frame-by-frame feature matching and compares them with a local map to find the exact camera location in real-time. It does a motion-only bundle adjustment so as to minimize error in placing each feature in its correct position, also called as minimizing reprojection error. Then comes the local mapping part. ORB-SLAM2 makes local maps and optimizes them using algorithms like ICP (Iterative Closest Point) and performs a local Bundle Adjustment so as to compute the most probable position of the camera. Finally, it uses pose-graph optimization to correct the accumulated drift and perform a loop closure. It's necessary to perform Bundle Adjustment once after loop

closure, so that robot is at the most probable location in the newly corrected map. After the addition of a keyframe to the map or performing a loop closure, ORB-SLAM2 can start a new thread that performs a Bundle adjustment on the full map so the location of each keyframe and points in it get a fine-tuned location value.

4.2. Deep Learning for Visual Odometry

As a solution for unsupervised monocular depth, we picked a network, proposed in (Bian et al., 2021) together with addition, described in (Liu et al., 2018), which we discuss later in below section.

We use pre-trained depth and pose CNNs from KITTI unlabeled videos. A brief description on how they work: given two adjacent frames (I_a, I_b) randomly sampled from a video, their depth maps (D_a, D_b) and relative 6-DoF camera pose P_{ab} are first estimated by the depth and pose CNNs, respectively. With the predicted depth and pose, we can synthesize the reference image I_a using the source image I_b by differentiable warping, which generates I'_a. Then the network is supervised by the photometric loss between the real I_a and the synthesized I'_a. To explicitly constrain the depth CNN to predict scale-consistent results on adjacent frames, a geometry consistency loss L_G proposed. To handle invalid cases such as static frames and dynamic objects, they use two masks. First, a M_s is introduced to reason the dynamics and occlusions by checking the depth consistency. Second, the auto-mask M_a to remove stationary points on image pairs where the camera is not moving.

The objective function is formulated as follows:

$$L = \alpha L_P^M + \beta L_S + \gamma L_G,$$

where L_P^M stands for the photometric loss L_P weighted by the mask M_s. L_S stands for the smoothness loss, and L_G is the geometric consistency loss. $[\alpha, \beta, \gamma]$ are the loss weighting terms. The loss is averaged over valid points, which are determined by M_a.

4.2.1. Photometric Loss

$$L_P = \frac{1}{|\mathcal{V}|} \sum_{p \in \mathcal{V}} (\lambda \left\| I_a(p) - I'_a(p) \right\|_1 + (1-\lambda) \frac{1 - \mathrm{SSIM}_{aa'}(p)}{2}), \tag{6}$$

where $\mathcal{V}$ stands for the set of valid points that are successfully projected from Ia to the image plane of I_b, and p stands for a generic point in $\mathcal{V}$. We choose L_1 loss due to its robustness to outliers. Besides, $\mathrm{SSIM}_{aa'}$ stands for the element-wise similarity between I_a and I'_a by:

$$\mathrm{SSIM(x, y)} = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)},$$

where x, y stands for two 3 by 3 patches around the central pixel. C_1 and C_2 are constants. μ and σ are local statistics of the image color, i.e., mean and variance, respectively. As it is for original paper, we used $C_1 = 0.0001$, $C_2 = 0.0009$, and $\lambda = 0.15$.

4.2.2. Smoothness Loss

Leveraging brightness constancy and spatial smoothness priors is ubiquitous in classical dense correspondence algorithms (Baker & Matthews, 2004). Previous works (Ranjan et al., 2019), (Yin & Shi, 2018), (Zhou et al., 2019) have used the photometric loss between the warped frame and the reference frame as an unsupervised loss function for network training. With the predicted depth D_a and pose P_{ab}, we synthesize $I_a^{'}$ by warping I_b, where differentiable warping (Jaderberg et al., 2015) is used. With the synthesized $I_a^{'}$ and the reference image I_a, we formulate the objective function.

As the photometric loss is not informative in lowtexture regions of the scene, existing work also incorporates a smoothness prior to regularize the estimated depth map. We adopt the edge-aware smoothness loss used in (Ranjan et al., 2019). Formally,

$$L_S = \sum_p (e^{-\bigtriangledown I_a(p)} * D_a(p)))^2 \tag{7}$$

where $\bigtriangledown$ is the first derivative along spatial directions.It ensures smoothness to be guided by image edges.

4.2.3. Geometric consistency Loss

To explicitly enforce geometry consistency, we constrain that the predicted D_a and D_b (related by Pab) conform the same 3D structure by penalizing their inconsistency.

Specifically, we propose a differentiable depth inconsistency operation to compute the pixel-wise inconsistency between two depth maps. Here, D_b^a is the synthesized depth for I_b, which is generated by D_a and pose P_{ab} with the underlying rigid transformation. $D_b^{'}$ is an interpolation of D_b for aligning and comparing with D_b^a. Given them, we compute the depth inconsistency map D_{diff} for each $pin\mathcal{V}$ as:

$$D_{diff}(p) = \frac{|D_b^a(p) - D_b^{'}(p)|}{D_b^a(a) + D_b^{'}(p)} \tag{8}$$

where we normalize depth differences by their summation. This works better than the absolute distance in practice

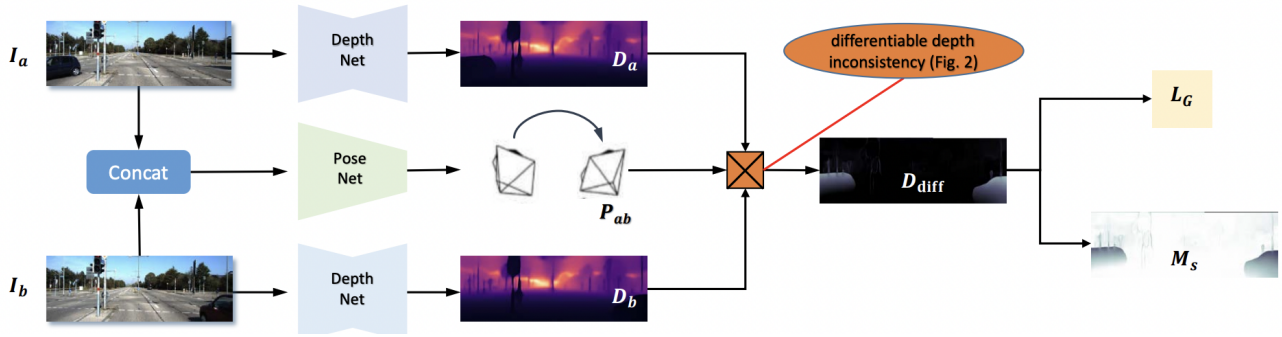

Figure 2. Illustration of the geometry consistency loss and M_s mask.

as it treats points at different absolute depths equally in optimization. Besides, the function is symmetric, and the outputs are naturally ranging from 0 to 1, which makes the training more stable.

With the inconsistency map, we define the proposed geometry consistency loss as:

$$L_P = \frac{1}{|\mathcal{V}|} \sum_{p \in \mathcal{V}} D_{diff}(p), \tag{9}$$

which minimizes the geometric inconsistency of predicted depths over two views. By minimizing the depth inconsistency between samples in a batch, we naturally propagate such consistency to the entire sequence: the depth of I_1 agrees with the depth of I_2 in a batch; the depth of I_2 agrees with the depth of I_3 in another training batch. Eventually, depths of Ii of a sequence should all agree with each other, leading to scale-consistent results over the entire sequence.

4.2.4. Masks

As moving objects and occlusions naturally violate the geometry consistency assumption, they will cause large depth inconsistency in our pre-computed D_{diff}. This encourages us to define the M_s as:

$$M_s = 1 - D_{diff} \tag{10}$$

where the M_s is in $[0, 1]$ and it attentively assign low weights for geometrically inconsistent pixels and high weights for consistent pixels.

We propose to mask out these regions by introducing a self-discovered mask (M_s) and adopting the auto-mask (M_a) by Monodepth2 (Godard et al., 2019). The proposed Ms computes weights (ranging from 0 to 1) for points in V by checking their depth consistency, and the Ma simply removes invalid points from V. The proposed two masks are readily integrated into the proposed learning framework.

First, to use M_a in our loss function, we remove invalid points in V that have $M_a(p) = 0$. When training the network, we only compute losses on the remaining valid points. Second, we use the proposed Ms to re-weight the photometric loss in (6) by:

$$L_P^M = \frac{1}{|\mathcal{V}|} \sum_{p \in \mathcal{V}} (M_s(p) * L_p(p)), \tag{11}$$

4.3. CoordnConv

Our main contribution for deep learning visual odometry is introduction of a CoordCond (Liu et al., 2018). The proposed CoordConv layer is an extension to the standard convolutional layer wherein convolution is conditioned on coordinates.

Convolution is equivariant, which implies that as each filter is applied to the input to generate the output, which mean that it losses context where each filter is. Introduction of two additional channels with pixel coordinates assist convolution by letting filters know where they are. Basically, allowing convolutional filters to see coordinates breaks translation equivariance

Authors found a little improvement for image classification task, but significant for object detection and for generative models. For some Atari games, like Ms. Pac-Man, CoordConv was useful for an agent to learn better policies.

This CoordConv is used instead of first convolutional layer in PoseNet. We use pre-trained network on result that were achieved in initial paper. Addition two channels force to add additional channels to weights of this layer either. We experimented with different variants for this procedure: 1) initialize additional-channels weights with zeros; 2) initialize them randomly; 3) reinitialize the whole weight randomly.

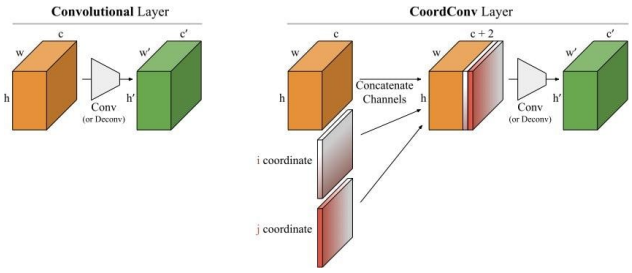

Figure 3. Comparison of 2D convolutional and CoordConv layers. (left) A standard convolutional layer maps from a representation block with shape h × w × c to a new representation of shape h' × w' × c'. (right) A CoordConv layer has the same functional signature, but accomplishes the mapping by first concatenating extra channels to the incoming representation. These channels contain hard-coded coordinates, the most basic version of which is one channel for the i coordinate and one for the j coordinate, as shown above.

We also used different modes for fine-tuning: 1) fine-tune all the weights of the PoseNet; 2) start to fine-tune weight of the first convolution, and unfreeze other layers of PoseNet with a delay. DepthNet is kept frozen for the whole time of fine-tuning.

4.4. Masking of transparent Objects

Following this paper (Xie et al., 2020). We used a pre-trained network to first create masks for our image sequences from the KITTI (sequence 8, 9 and 10) and TUM (freiburg1/xyz, freiburg1/rpy and freiburg3/long office household) datasets and then pre-processed the initial image sequences by masking out the detected areas from the original image while preserving dimensions. The same process is repeated for the deblurred sequences generated from the mentioned above datasets.

The network is pre-trained on Trans10k dataset which has 10,428 manually-labeled images with high degree of variability in terms of scale, pose, contrast, category, occlusion and transparency. Objects are divided into two categories, thing and stuff, where things are small and movable objects (e.g. bottle), while stuff are large and fixed (e.g. vitrine). Besides Trans10K, a boundary-aware object segmentation method, termed TransLab is used for network pre-training which is able to "Look at boundary" to improve transparent object segmentation.

For our method we create a Boolean tensor for the mask and use element wise multiplication with broadcasting to cover the transparent region with minimum pixel value on both the blurred and unblurred datasets to generate segmented images for both of them respectively.

4.5. Deblurring

Image deblurring has long been a topic of interest in computer vision researches. The purpose of deblurring is to restore a consistent base image with higher quality details and edge structure that was blurred because of camera fast motion, shake or defocus. As mentioned earlier, the main purpose of this work is to perform the task Visual Odometry on a highly dynamic system, and since it is a highly dynamic system, the probability of blurred images from the camera input is high. So in this section we are proposing a method to mitigate the blur effect on visual odometry task.

A learning-based method for deblurring is at the center of attention of many researchers. Early methods (Schuler et al., 2016) (Sun et al., 2015) (Xiao et al., 2016) replaced some modules or steps in the traditional framework with learned parameters to utilize external data. Recently, end-to-end trainable networks have been used in research to deblur images (Nah et al., 2016) and videos (Kim et al., 2017) (Su et al., 2016) (Wieschollek et al., 2017). They are, for example, Nah et al. achieved the most advanced results using multi scale convolutional neural networks (CNN). This process begins with a blurred image that is very rough and steadily restores the latent image with better resolution until it reaches maximum resolution. The system uses a multi scale mechanism in the conventional approach. When dealing with large fuzzy kernels, thick to thin pipelines are common (Cho & Lee, 2009).
In this project, we study this technique and perform deblurring using a Scale-recurrent Network (SRN-DeblurNet)(Tao et al., 2018b). It takes as input a sequence of blurry images downsampled from the input image at different scales, and produces a set of corresponding sharp images. The sharp one at the full resolution is the final output.

SRN-DeblurNet (Tao et al., 2018a) was compared with some of the previous image deblurring methods on well known GOPPRO(Nah et al., 2017) testing datasets. This dataset suffers from comlex blur due to large camera and object motion. The most used metrics that is used here for comparison are PSNR (Peak signal-to-noise ratio) and SSIM (Structural Similarity Index). In the following the comparison result is show and the end SRN-DeblurNet was chosen as deblurring approach to be used.

4.6. Metrics

For obtaining metrics we use Python package for the evaluation of odometry and SLAM. (Grupp, 2017). Package provides two metrics for evaluation. One of them is Absolute pose error (APE) and Relative pose error (RPE).

The absolute pose error (APE) is a metric for investigating the global consistency of a SLAM trajectory.

APE is based on the absolute relative pose between two

poses $P_{ref,i}, P_{est,i} \in \mathrm{SE}(3)$ at timestamp i:

$$\begin{aligned} E_i &= P_{est,i} \ominus P_{ref,i} \\ &= P_{ref,i}^{-1} P_{est,i} \in \mathrm{SE}(3) \end{aligned} \tag{12}$$

where $\ominus$ is the inverse compositional operator, which takes two poses and gives the relative pose. (Lu & Milios, 1997)

These different pose relations can be calculate with APE:

Translation part of the error:

$$APE_i = \|\mathrm{trans}(E_i)\| \tag{13}$$

Rotation angle error:

$$APE_i = |(\mathrm{angle}(\log_{\mathrm{SO}(3)}(\mathrm{rot}(E_i)))| \tag{14}$$

where: $\log_{\mathrm{SO}(3)}(\cdot)$ is the inverse of $\exp_{\mathrm{SO}(3)}(\cdot)$ (Rodrigues formula)

Rotation part of the error:

$$APE_i = \|\mathrm{rot}(E_i) - I_{3\times3}\|_F \tag{15}$$

Full transformation error:

$$APE_i = \|E_i - I_{4\times4}\|_F \tag{16}$$

Then, different statistics can be calculated on the APEs of all timestamps, e.g. the RMSE:

$$\mathrm{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N} APE_i^2} \tag{17}$$

The relative pose error (RPE) is a metric for investigating the local consistency of a SLAM trajectory.

RPE compares the relative poses along the estimated and the reference trajectory. This is based on the delta pose difference:

$$\begin{aligned} E_{i,j} &= \delta_{est_{i,j}} \ominus \delta_{ref_{i,j}} \\ &= (P_{ref,i}^{-1} P_{ref,j})^{-1} (P_{est,i}^{-1} P_{est,j}) \in \mathrm{SE}(3) \end{aligned} \tag{18}$$

Pose relations for RPE uses the same formulas as APE with relative to $E_{i,j}$.

Metrics for all experiments acquired with $SE(3)$ Umeyama alignment (Umeyama, 1991) and scale correction apply which are common use for monocular SLAM.

5. Experiments

5.1. Preprocessing Input Data for ORB SLAM2

Preprocessing the input sequences for the ORB SLAM2 was done for such sequences from TUM as "freiburg1 rpy" (fr1/rpy), "freiburg1 xyz" (fr1/xyz), "freiburg3 long office household" (fr3/long) both directly and after deblurring.

Figure 4. Sample images before(above) after deblurring from TUM sequence fr3/long office household. The datasets mostly have blurred images while the camera rotates

For KITTI sequences we used: seq08, seq09, seq10. These sequences were chosen since this is a classical split in a deep learning visual odometry community: seq08 is used for training, seq09 and seq10 for the validation. This allows us to compare results with fine-tuning of the network with the CoordConv. This topic with regards to Visual Odometry is quite new and not much research is available moreover all the models we found take ground truth labels which depend on the training dataset used.

Apart from the examples that worked well for masking there

Figure 5. Sample images after deblurring and transparent object segmentation from KITTI seq 8 (above) and TUM sequence fr3/long office household (below). The two datasets are chosen for outdoor and indoor VO respectively

were also many that didn't as can be seen from the examples below specifically for the KITTI dataset, the network sometimes masked the sky which strongly affected the path in ORB SLAM pipeline and thus the metrics could not be calculated for that as it completely lost track in some cases a sample of such an image is shown below:

5.2. Deep Learning Visual Odometry with CoordConv

As was stated before, we used pre-trained Unsupervised Scale-consistent Depth Learning network (Bian et al., 2021). We changed the first convolutional layer of PoseNet to the CoordConv. This change forces us to add addition channels to the pretrained weight of first convolutional layer of PoseNet so it will match the dimension of the CoordConv.

Figure 6. Sample images after deblurring and transparent object segmentation from KITTI seq 8 (above) where the segmentation is improper and starts masking out the sky

In total we make possible six different experiments with regard to weight initialization and fin-tuning mode. For the weight initialization we set up different options: initialize additional weights with zeros, initialize them randomly, re-initialize the whole weight of the layer randomly. Furthermore, we set up different modes for fine-tuning: fine-tune the whole PoseNet; initially fine-tune only the first layer, then unfreeze the whole PoseNet. For the whole process of fine-tuning out DepthNet is kept frozen.

Authors of source paper had used the encoder composed out of Resnet blocks which were pre-trained on the Imagenet for both DepthNet and PoseNet. Since we did not want to retrain nets entirely and we use Adam optimizer, first steps of training should have been with learning rate close to zero. This keep save the pre-trained weights from huge and not valid changes in the very beginning.

We also experimented with the step learning rate scheduler with the warmup. Trying on different epoch size and amount of epochs showed that better to use a setup of the authors from the source paper with epoch size equal to 1000 steps. After one epoch (1000 steps) learning rate comes to the base value equal to 1e-5.

The computational complexity of the experiments did not allow us to conduct the whole set of experiments and find the best hyper-parameters of it. On GPU Tesla V100-SXM2-16GB 50 epochs of fine-tuning only the PoseNet (DepthNet is kept frozen) took almost 15 hours. This is explained by the fact that encoders of the networks from the source paper is based on the Resnet50 blocks.

6. Results

6.1. Preprocessing Input Data for ORB SLAM2

Segmentation and masking out transparent objects on original and deblurred sequences showed better results for the TUM dataset in general with little inconsistencies. For example on TUM sequence "fr1/rpy" got significant benefit from masking out transparent objects, same as for "fr3/long". But for sequence "fr1/xyz" results got worse.

For KITTI sequences masking process led to unstable ORB-SLAM2 results. ORB-SLAM2 lost trajectories after first frames and was unable to provide results. Main reason is masking out major part of the sky which as we researched has feature points to maintain trajectory (Figure 6). ORB SLAM halted mid process and the final results could not be obtained, for future results we suggest fine tuning the segmentation net separately on the particular dataset with a lot of similar images for it to work properly and maybe get reliable results also in the outdoor environment.

	fr1/rpy			fr1/xyz			fr3/long		
	$APE_t(m)$	$RPE_t(m)$	$RPE_r(°)$	$APE_t(m)$	$RPE_t(m)$	$RPE_r(°)$	$APE_t(m)$	$RPE_t(m)$	$RPE_r(°)$
o	0.024508	0.018725	1.444773	**0.005638**	**0.008801**	**0.544655**	0.031001	0.007968	0.366255
s	0.022390	**0.013729**	1.280872	0.006539	0.010742	0.647860	**0.025041**	0.007128	**0.360615**
d	**0.018767**	0.017813	**1.127863**	0.008820	0.013572	0.696928	0.034174	0.007442	0.336747
d+s	**0.018613**	0.065011	2.190500	0.007662	0.012044	0.710171	0.025793	**0.006862**	0.336524

Table 1: APE translation part error (APE_t), RPE translation (RPE_t) and rotation (RPE_r) part errors (lower is better) for TUM data sequences. Experiments were conducted on "freiburg1 rpy" (fr1/rpy), "freiburg1 xyz" (fr1/xyz), "freiburg3 long office household" (fr3/long). Ground truth data compared to sequences: o - original, s - segmented, d - deblurred, $d + s$ - deblurred and segmented.

	seq08			seq09			seq10		
	$APE_t(m)$	$RPE_t(m)$	$RPE_r(°)$	$APE_t(m)$	$RPE_t(m)$	$RPE_r(°)$	$APE_t(m)$	$RPE_t(m)$	$RPE_r(°)$
o	**37.757439**	**0.467190**	**0.053345**	**31.126846**	0.635913	**0.053478**	**3.502739**	**0.060417**	**0.059422**
d	38.701056	0.487801	**0.053924**	42.403437	**0.334358**	**0.053853**	5.366431	0.095038	0.072026

Table 2: APE translation part error (APE_t), RPE translation (RPE_t) and rotation (RPE_r) part errors (lower is better) for KITTI data sequences. Experiments were conducted on sequences 08, 09 and 10. Ground truth data compared to sequences: o - original, d - deblurred. (ORB-SLAM2 on segmented sequence loses trajectory)

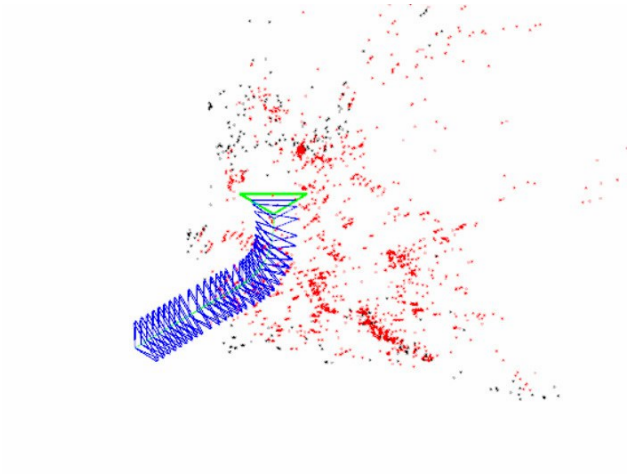

Figure 7. Sample sequence from KITTI on which the ORB SLAM pipeline stops performing, this is due to a continuous sequence of images (even a small sequencial set) that has the sky masked out

Figure 8. Sample image of road during sunny day and its reflective ability. Segmentation network masks road as transparent (reflective) object.

Result trajectories are shown on Figure 9 and 10.

Deblurring tends to increase images quality and improve feature points extraction, but in conducted experiments hard to say whether deblurring improves or decreases results. For example, in sequence 09 of KITTI dataset the RPE_t decreased significantly while APE_t increased.

6.2. Deep Learning for Visual Odometry

Introduction a CoordConv instead of the first convolutional layer into the PoseNet did not provide any gain in metrics. As you can see at the table 3, both metrics for both sequences are greater than the ones that were achieved in the source paper. However, this results is not representative and one should consider loss plots (Figure 12, Figure 13). for evaluation of the CoordConv existence.

We consider the fine-tuning regime for our net. For this, we used a smaller learning rate (1e-5 instead of 1e-4 from

	seq09		seq10	
	t_err, %	r_err, (deg/100m)	t_err, %	r_err, (deg/100m)
source paper	**7.31**	**3.05**	**7.79**	**4.90**
random, first_all	86.94	24.92	147.39	22.77
zeros, whole	133.56	12.73	162.55	18.52
random, whole	92.36	25.25	147.66	22.45
allrandom, first_all	86.54	26.15	147.28	23.90
zeros, first_all	87.08	25.09	147.54	22.70

Table 3: Translation and rotation errors for the deep learning visual odometry. Random, zeros, and allrandom - initialize addition channel randomly, with zeros, initalize entire weight randomly. Whole, first_all - fine-tune the whole net, initially fine-tune the first layer, then the whole net.Translation and rotation errors for the deep learning visual odometry. Random, zeros, and allrandom - initialize addition channel randomly, with zeros, initalize entire weight randomly. Whole, first_all - fine-tune the whole net, initially fine-tune the first layer, then the whole net.

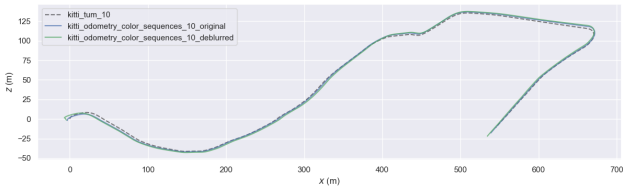

Figure 9. Acquired trajectories with ORB-SLAM2 (KITTI seq10), dotted - groundtruth, blue - ORB-SLAM2 on original sequence, green - on deblurred sequence.

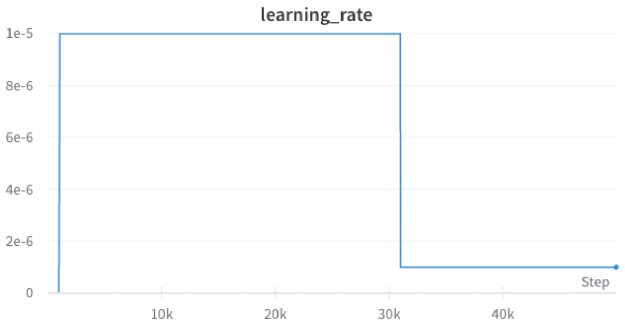

Figure 11. Learning rate during training.

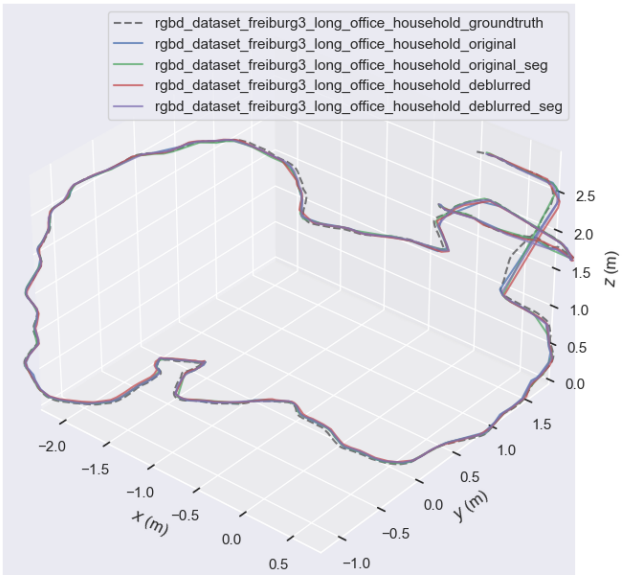

Figure 10. Acquired trajectories with ORB-SLAM2 (TUM fr3/long), dotted - groundtruth, blue - ORB-SLAM2 on original sequence, green - segmented, red - deblurred, violet - deblurred and segmented.

the source code) and a partially training for our net in some scenarios. After first epoch for which we intentionally set an extremely low learning, total loss is getting lower, because of the drops in photometric loss. This results are expected since DepthNet is frozen and PoseNet only learns.

For the validation loss the results is not that much straightforward: photometric loss is getting lower, but geometry consistency loss in general is getting bigger. However, total validation loss is also decrease.

Also, the losses of the fine-tuning the PoseNet when the additional layers were initialized with the zeros and with fine-tuning the whole net show that there is indeed a lack of training. This net was train the lest amount of epochs (36) and achieves the worst results. The learning rate for the most of the experiments one can see on Figure 11.

Although the metrics are worse, the training process is obviously underway by which we make a conclusion, that initial assumption that introduction of the CoordConv is require only the fine-tuning is wrong. The PoseNet with the CoordConv requires more precise experiments with more accurate choosing of hyper-parameters.

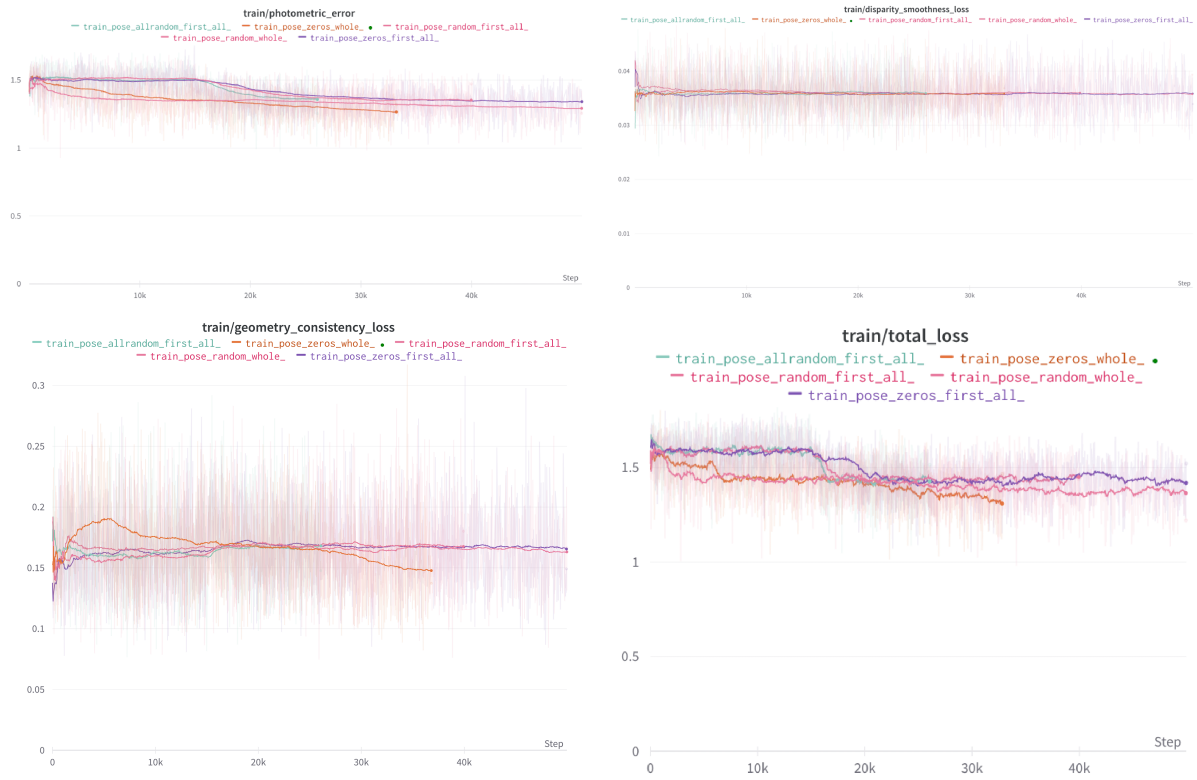

Figure 12. Train loss. Random, zeros, and allrandom - initialize addition channel randomly, with zeros, initialize entire weight randomly. Whole, first_all - fine-tune the whole net, initially fine-tune the first layer, then the whole net.

7. Conclusion and Prospects

From our experiments with deblurring we found that the ORB SLAM pipeline takes an inordinate about of time to work and also the metrics reported are deteriorated in general so there is no gain for both KITTI and TUM dataset sequences. For segmentation we can see significant increase for indoor datasets for TUM sequences fr1/rpy and fr1/xyz while it decreases a little for fr3/long. The combination of deblurring with segmentation also showed similar trend on these datasets. For the KITTI sequence with regards to segmentation the ORB SLAM haults and thus we have no metrics, in future we propose to use fine tuning with the particular outdoor datasets or some unsupervised methods or segmentation. Our research suggests that the sky actually plays an important role in determining the path and is or important for reference. If this problem in segmentation of sequences can be solved this could lead to interesting results,

The CoordConv layer showed the significant results for some task in the source paper (Liu et al., 2018). The results of this project shows that the training with the CoordCond is underway, but the any solid conslusions can be made. For now future work for experiments with CoordCOnv in this task are as follows:

- Conclude current fine-tune experiments for a longer time with higher learning rate.
- Consider the new weights for losses.
- Retrain the whole network entirely with the introduction of the CoordConv.

Figure 13. Validation loss. Random, zeros, and allrandom - initialize addition channel randomly, with zeros, initialize entire weight randomly. Whole, first_all - fine-tune the whole net, initially fine-tune the first layer, then the whole net.

References

Baker, S. and Matthews, I. Lucas-kanade 20 years on: A unifying framework. *International journal of computer vision*, 56(3):221–255, 2004.

Bian, J.-W., Zhan, H., Wang, N., Li, Z., Zhang, L., Shen, C., Cheng, M.-M., and Reid, I. Unsupervised scale-consistent depth learning from video. *International Journal of Computer Vision*, 129(9):2548–2564, 2021.

Cho, S. and Lee, S. Fast motion deblurring. *ACM Trans. Graph.*, 28(5):1–8, December 2009. ISSN 0730-0301. doi: 10.1145/1618452.1618491. URL https://doi.org/10.1145/1618452.1618491.

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

Godard, C., Mac Aodha, O., Firman, M., and Brostow, G. J. Digging into self-supervised monocular depth estimation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 3828–3838, 2019.

Grupp, M. evo: Python package for the evaluation of odometry and slam. https://github.com/MichaelGrupp/evo, 2017.

Hartley, R. I. In defense of the eight-point algorithm. *IEEE Transactions on pattern analysis and machine intelligence*, 19(6):580–593, 1997.

Jaderberg, M., Simonyan, K., Zisserman, A., et al. Spatial transformer networks. *Advances in neural information processing systems*, 28, 2015.

Kim, T. H., Lee, K. M., Schölkopf, B., and Hirsch, M. Online video deblurring via dynamic temporal blending network. In *Proceedings IEEE International Conference on Computer Vision (ICCV)*, pp. 4038–4047, Piscataway, NJ, USA, October 2017. IEEE. URL http://openaccess.thecvf.com/content_

ICCV_2017/papers/Kim_Online_Video_Deblurring_ICCV_2017_paper.pdf.

Kuo, X.-Y., Liu, C., Lin, K.-C., and Lee, C.-Y. Dynamic attention-based visual odometry. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pp. 36–37, 2020.

Liu, R., Lehman, J., Molino, P., Petroski Such, F., Frank, E., Sergeev, A., and Yosinski, J. An intriguing failing of convolutional neural networks and the coordconv solution. *Advances in neural information processing systems*, 31, 2018.

Lu, F. and Milios, E. Globally consistent range scan alignment for environment mapping. *Autonomous robots*, 4 (4):333–349, 1997.

Nah, S., Kim, T. H., and Lee, K. M. Deep multi-scale convolutional neural network for dynamic scene deblurring. *CoRR*, abs/1612.02177, 2016. URL http://arxiv.org/abs/1612.02177.

Nah, S., Kim, T. H., and Lee, K. M. Deep multi-scale convolutional neural network for dynamic scene deblurring. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

Nistér, D. An efficient solution to the five-point relative pose problem. *IEEE transactions on pattern analysis and machine intelligence*, 26(6):756–770, 2004.

Ranjan, A., Jampani, V., Balles, L., Kim, K., Sun, D., Wulff, J., and Black, M. J. 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*, pp. 12240–12249, 2019.

Sattler, T., Zhou, Q., Pollefeys, M., and Leal-Taixe, L. Understanding the limitations of cnn-based absolute camera pose regression. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 3302–3312, 2019.

Schubert, D., Goll, T., Demmel, N., Usenko, V., Stückler, J., and Cremers, D. The tum vi benchmark for evaluating visual-inertial odometry. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1680–1687. IEEE, 2018.

Schuler, C. J., Hirsch, M., Harmeling, S., and Schölkopf, B. Learning to deblur. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(7):1439–1451, 2016. doi: 10.1109/TPAMI.2015.2481418.

Su, S., Delbracio, M., Wang, J., Sapiro, G., Heidrich, W., and Wang, O. Deep video deblurring. *CoRR*, abs/1611.08387, 2016. URL http://arxiv.org/abs/1611.08387.

Sun, J., Cao, W., Xu, Z., and Ponce, J. Learning a convolutional neural network for non-uniform motion blur removal. *CoRR*, abs/1503.00593, 2015. URL http://arxiv.org/abs/1503.00593.

Tao, X., Gao, H., Shen, X., Wang, J., and Jia, J. Scale-recurrent network for deep image deblurring. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018a.

Tao, X., Gao, H., Wang, Y., Shen, X., Wang, J., and Jia, J. Scale-recurrent network for deep image deblurring. *CoRR*, abs/1802.01770, 2018b. URL http://arxiv.org/abs/1802.01770.

Umeyama, S. Least-squares estimation of transformation parameters between two point patterns. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 13(04): 376–380, 1991.

Wieschollek, P., Hirsch, M., Schölkopf, B., and Lensch, H. P. A. Learning blind motion deblurring. *CoRR*, abs/1708.04208, 2017. URL http://arxiv.org/abs/1708.04208.

Xiao, L., Wang, J., Heidrich, W., and Hirsch, M. Learning high-order filters for efficient blind deconvolution of document photographs. In *Computer Vision - ECCV 2016*, volume Lecture Notes in Computer Science, LNCS 9907, Part III, pp. 734–749. Springer, October 2016.

Xie, E., Wang, W., Wang, W., Ding, M., Shen, C., and Luo, P. Segmenting transparent objects in the wild. *arXiv preprint arXiv:2003.13948*, 2020.

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

Zhang, Z. Determining the epipolar geometry and its uncertainty: A review. *International journal of computer vision*, 27(2):161–195, 1998.

Zhao, W., Liu, S., Shu, Y., and Liu, Y.-J. Towards better generalization: Joint depth-pose learning without posenet. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9151–9161, 2020.

Zhou, J., Wang, Y., Qin, K., and Zeng, W. Moving indoor: Unsupervised video depth learning in challenging environments. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 8618–8627, 2019.

Zhou, Q., Sattler, T., Pollefeys, M., and Leal-Taixe, L. To learn or not to learn: Visual localization from essential matrices. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3319–3326. IEEE, 2020.

8. Repositories

In this section we brought all the repositories that we used during the project.

https://github.com/JiawangBian/SC-SfMLearner-Release

https://github.com/anastasiia-kornilova/ORB_SLAM2.git

https://github.com/jiangsutx/SRN-Deblur.git

https://github.com/MichaelGrupp/evo.git

https://github.com/xieenze/Segment_Transparent_Objects

Team member's contributions

Vladimir Chernyy

- Maitaining github repository
- Developing training + evaluation scripts
- Running fine-tuning on remote cluster
- Adjusting existing codebase to run on WANDB instead of TensorBoard
- Preparing part of report on PoseNet architecture

Vladislav Trifonov

- Literature review
- Choosing the deep learning visual odoemtry net
- Introduce the CoordConv
- Introduce the regimes for fine-tune and weight initialization
- Set up the scripts expereiments
- Assisted in the experiments with deep learning visual odometry

Anna Akhmatova

- Evaluation and preparation results for the required metrics
- Researching the literature on the evaluation process
- Writing part about metric(evaluation) and part about loss-functions and masks of the report

Hekmat Taherinejad

- Running preprocessing and deblurring on the datasets
- Building and running ORB-SLAM2 on the original and pre-processed datasets
- Contribution in github repository completion
- Developing the presentation slides
- Preparing report on Deblurring and ORB-SLAM2

Prateek Rajput

- Researching papers for supervised and unsupervised transparent object and dynamic object detection architectures
- Generating masks for transparent objects and segmenting them from the image sequences

- Compiling datasets for all the permutations of preprocessed data
- Writing literature survey on architectures and segmentation plus conclusion parts of the report
- Preparing presentation and figures for the video recording

Timur Bayburin

- Metrics and trajectories visualisation
- Datasets management
- GitHub: script for metrics, packages integration
- Report formatting (tables, figures)
- Report section 6