LM-Reloc: Levenberg-Marquardt Based Direct Visual Relocalization

Lukas von Stumberg\textsuperscript{1,2*} Patrick Wenzel\textsuperscript{1,2*} Nan Yang\textsuperscript{1,2} Daniel Cremers\textsuperscript{1,2}
\textsuperscript{1} Technical University of Munich \textsuperscript{2} Artisense

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure1.png}
\caption{We propose LM-Reloc – a novel approach for visual relocalization based on direct image alignment. It consists of two deep neural networks: LM-Net, an encoder-decoder network for learning dense visual descriptors and a CorrPoseNet to bootstrap the direct image alignment. The final 6DoF relative pose estimate between image $I$ and $I'$ is obtained in a coarse-to-fine pyramid scheme leveraging the learned feature maps. The initialization for the direct image alignment is obtained by the CorrPoseNet.}
\end{figure}

Abstract

We present LM-Reloc – a novel approach for visual relocalization based on direct image alignment. In contrast to prior works that tackle the problem with a feature-based formulation, the proposed method does not rely on feature matching and RANSAC. Hence, the method can utilize not only corners but any region of the image with gradients. In particular, we propose a loss formulation inspired by the classical Levenberg-Marquardt algorithm to train LM-Net. The learned features significantly improve the robustness of direct image alignment, especially for relocalization across different conditions. To further improve the robustness of LM-Net against large image baselines, we propose a pose estimation network, CorrPoseNet, which regresses the relative pose to bootstrap the direct image alignment. Evaluations on the CARLA and Oxford RobotCar relocalization tracking benchmark show that our approach delivers more accurate results than previous state-of-the-art methods while being comparable in terms of robustness.

1. Introduction

Map-based relocalization, that is, to localize a camera within a pre-built reference map, is becoming more and more important for robotics \cite{6}, autonomous driving \cite{24,4} and AR/VR \cite{29}. Sequential-based approaches, which leverage the temporal structure of the scene provide more stable pose estimations and also deliver the positions in global coordinates compared to single image-based localization methods. The map is usually generated by either using LiDAR or visual Simultaneous Localization and Mapping (vSLAM) solutions. In this paper, we consider vSLAM maps due to the lower-cost visual sensors and the richer semantic information from the images. Feature-based methods \cite{14,7,22,23} and direct methods \cite{13,12,11,1} are two main lines of research for vSLAM.

Once a map is available, the problem of relocalizing within this map at any later point in time requires to deal with long-term changes in the environment. This makes a centimeter-accurate global localization challenging, especially in the presence of drastic lighting and appearance changes in the scene. For this task, feature-based methods are the most commonly used approaches to estimate the ego-pose and its orientation. This is mainly due to the advantage that features are more robust against changes in lighting/illumination in the scene.

However, feature-based methods can only utilize keypoints that have to be matched across the images before the pose estimation begins. Thus they ignore large parts of the
available information. Direct methods, in contrast, can take advantage of all image regions with sufficient gradients and as a result, are known to be more accurate on visual odometry benchmarks [41, 11, 39].

In this paper, we propose LM-Reloc, which applies direct techniques to the task of relocalization. LM-Reloc consists of LM-Net, CorrPoseNet, and a non-linear optimizer, which work seamlessly together to deliver reliable pose estimation without RANSAC and feature matching. In particular, we derive a loss formulation, which is specifically designed to work well with the Levenberg-Marquardt (LM) algorithm [16, 20]. We use a deep neural network, LM-Net, to train descriptors that are being fed to the direct image alignment algorithm. Using these features results in better robustness against bad initializations, large baselines, and against illumination changes.

While the robustness improvements gained with our loss formulation are sufficient in many cases, for very large baselines or strong rotations, some initialization can still be necessary. To this end, we propose a pose estimation network. Based on two images it directly regresses the 6DoF pose, which we utilize as initialization for LM-Net. The CorrPoseNet contains a correlation layer as proposed in [27], which ensures that the network can handle large displacements. The proposed CorrPoseNet displays a lot of synergies with LM-Net. Despite being quite robust, the predictions of the CorrPoseNet are not very accurate. Thus it is best used in conjunction with our LM-Net, resulting in very robust and accurate pose estimates.

We evaluate our approach on the relocalization tracking benchmark from [36], which contains scenes simulated using CARLA [9], as well as sequences from the Oxford RobotCar dataset [19]. Our LM-Net shows superior accuracy especially in terms of rotation while being competitive in terms of robustness.

We summarize our main contributions:

- LM-Reloc, a novel pipeline for visual relocalization based on direct image alignment, which consists of LM-Net, CorrPoseNet, and a non-linear optimizer.
- A novel loss formulation together with a point sampling strategy that is used to train LM-Net such that the resulting feature descriptors are optimally suited to work with the LM algorithm.
- Extensive evaluations on the CARLA and Oxford RobotCar relocalization tracking benchmark which show that the proposed approach achieves state-of-the-art relocalization accuracy without relying on feature matching or RANSAC.

2. Related Work

In this section, we review the main topics that are closely related to our work, including direct methods for visual localization and feature-based visual localization methods.

Direct methods for visual localization. In recent years, direct methods [13, 12, 11] for SLAM and visual odometry have seen a great progress. Unlike feature-based methods [14, 7, 22, 23] which firstly extracts keypoints as well as the corresponding descriptors, and then minimize the geometric errors, direct methods minimize the energy function based on the photometric constancy assumption without performing feature matching or RANSAC. By utilizing more points from the images, direct methods show higher accuracy than feature-based methods [39]. However, classical direct methods show lower robustness than feature-based methods when the photometric constancy assumption is violated due to, e.g., the lighting and weather changes which are typical for long-term localization [33]. In [2] and [25], the authors propose to use the handcrafted features to improve the robustness of direct methods against low light or global appearance changes. Some recent works [5, 18, 36] address the issue by using learned features from deep neural networks [15]. In [5] they train deep features using a Hinge-Loss based on the Lucas-Kanade method, however, in contrast to us, they estimate the optical flow instead of applying the features to the task of relocalization. The most related work to ours is GN-Net [36] which proposes a Gauss-Newton loss to learn deep features. By performing direct image alignment on the learned features, GN-Net can deliver reliable pose estimation between the images taken from different weather or season conditions. The proposed LM-Net further derives the loss formulation based on Levenberg-Marquardt to improve the robustness against bad initialization compared to the Gauss-Newton method. Inspired by D3VO [38], LM-Reloc also proposes a relative pose estimation network with a correlation layer [27] to regress a pose estimate which is used as the initialization for the optimization.

Feature-based visual localization. Most approaches for relocalization utilize feature detectors and descriptors, which can either be handcrafted, such as SIFT [17] or ORB [28], or especially in the context of drastic lighting and appearance changes can be learned. Recently, many descriptor learning methods have been proposed which follow a detect-and-describe paradigm, e.g., SuperPoint [8], D2-Net [10], or R2D2 [26]. Moreover, SuperGlue [32], a learning-based alternative to the matching step of feature-based methods has been proposed and yields significant performance improvements. For a complete relocalization pipeline the local pose refinement part has to be preceded by finding the closest image in a database given a query [3]. While some approaches [31, 30, 35] address the joint problem, in this work, we decouple these two tasks and only focus on the pose refinement part.
3. Method

In this work, we address the problem of computing the 6DoF pose $\xi \in SE(3)$ between two given images $I$ and $I'$. Furthermore, we assume that depths for a sparse set of points $P$ are available, e.g., by running a direct visual SLAM system such as DSO [11].

The overall pipeline of our approach is shown in Figure 1. It is composed of LM-Net, CorrPoseNet, and a non-linear optimizer using the LM algorithm. LM-Net is trained with a novel loss formulation designed to learn feature descriptors optimally suited for the LM algorithm. The encoder-decoder architecture takes as input a reference image $I$ as well as a target image $I'$. The network is trained end-to-end and will produce multi-scale feature maps $F_l$ and $F'_l$, where $l = 1, 2, 3, 4$ denotes the different levels of the feature pyramid. In order to obtain an initial pose estimate for the non-linear optimization, we propose CorrPoseNet, which takes $I$ and $I'$ as the inputs and regresses their relative pose. Finally, the multi-scale feature maps together with the depths obtained from DSO [11] form the non-linear energy function which is minimized using LM algorithm in a coarse-to-fine manner to obtain the final relative pose estimate. In the following, we will describe the individual components of our approach in more detail.

3.1. Direct Image Alignment with Levenberg-Marquardt

In order to optimize the pose $\xi$ (consisting of rotation matrix $R$ and translation $t$), we minimize the feature-metric error:

$$E(\xi) = \sum_{p \in P} \left\| F'_l(p') - F_l(p) \right\|_\gamma,$$

(1)

where $\| \cdot \|_\gamma$ is the Huber norm and $p'$ is the point projected onto the target image $I'$ using the depths and the pose:

$$p' = \Pi \left( (R \Pi^{-1}(p, d_p) + t) \right).$$

(2)

This energy function is first minimized on the coarsest pyramid level 1, whose feature maps $F_1$ have a size of $(w/8, h/8)$, yielding a rough pose estimate. The estimate is refined by further minimizing the energy function on the subsequent pyramid levels 2, 3, and 4, where $F_4$ has the size of the original image $(w, h)$. In the following, we provide details of the minimization performed in every level and for simplicity we will denote $F_1$ as $F$ from now on.

Minimization is performed using the Levenberg-Marquardt algorithm. In each iteration we compute the update $\delta \in \mathbb{R}^6$ in the Lie algebra $se(3)$ as follows: Using the residual vector $r \in \mathbb{R}^6$, the Huber weight matrix $W \in \mathbb{R}^{n \times n}$, and the Jacobian of the residual vector with respect to the pose $J \in \mathbb{R}^{n \times 6}$, we compute the Gauss-Newton system:

$$H = J^T W J$$

and

$$b = -J^T W r.$$  

(3)

The damped system can be obtained with either Levenberg’s formula [16]:

$$H' = H + \lambda I$$

or the Marquardt’s formula [20]:

$$H' = H + \lambda \text{diag}(H)$$

(4)

(5)

depending on the specific application.

The update $\delta$ and the pose $\xi^i$ in the iteration $i$ are computed as:

$$\delta = H'^{-1} b$$

and

$$\xi^i = \delta \boxplus \xi^{i-1},$$

(6)

where $\boxplus : se(3) \times SE(3) \rightarrow SE(3)$ is defined as in [11].

The parameter $\lambda$ can be seen as an interpolation factor between gradient descent and the Gauss-Newton algorithm. When $\lambda$ is high the algorithm behaves like gradient descent with a small step size, and when it is low it is equivalent to the Gauss-Newton algorithm. In practice, we start with a relatively large $\lambda$ and multiply it by 0.5 after a successful iteration, and by 4 after a failed iteration [11].

Figure 2 shows the typical behaviour of the algorithm. In the beginning the initial pose is inaccurate, resulting in projected point positions, which are a couple of pixels away from the correct location. $\lambda$ will be high meaning that the algorithm will behave similar to gradient descent. After a couple of iterations, the pose got more accurate, and the projected points are in a closer vicinity to the correct location. By now, $\lambda$ has probably decreased, so the algorithm will behave more similar to the Gauss-Newton algorithm. Now we expect the algorithm to converge quickly.

3.2. Loss Formulation for Levenberg-Marquardt

The key contribution of this work is LM-Net which provides feature maps $F$ that improve the convergence behaviour of the LM algorithm and, in the meantime, are invariant to different conditions. We train our network in a Siamese fashion based on ground-truth pixel correspondences.

In this section, $p$ denotes a reference point (located on image $I$) and the ground-truth correspondence (located on image $I'$) is $p'_{gt}$. For the loss functions explained below we further categorize $p'$ into $p'_{neg}$, $p'_{gt}$, and $p'_{ung}$, which is realized by using different negative correspondence sampling. Our loss formulation is inspired by the typical behaviour of the Levenberg-Marquardt algorithm explained in the previous section (see Figure 2). For a point, we distinguish four cases which can happen during the optimization:

1. The point is at the correct location ($p'_{gt}$).
2. The point is an outlier (\( p'_{\text{neg}} \)).

3. The point is relatively far from the correct solution (\( p'_{\text{cor}} \)).

4. The point is very close to the correct solution (\( p'_{\text{cor}} \)).

In the following we will derive a loss function for each of the 4 cases:

1. The point is already at the correct location. In this case we would like the residual to be as small as possible, in the best case 0.

\[
E_{\text{pos}} = \| F'(p'_g) - F(p) \|^2
\]  

2. The point is an outlier or the pose estimate is completely wrong. In this case the projected point position can be at a completely different location than the correct correspondence. In this scenario we would like the residual of this pixel to be very large to reflect this, and potentially reject a wrong update. To enforce this property we sample a negative correspondences \( p'_{\text{neg}} \) uniformly across the whole image, and compute

\[
E_{\text{neg}} = \max \left( M - \| F'(p'_{\text{neg}}) - F(p) \|^2, 0 \right) \tag{8}
\]

where \( M \) is the margin how large we would like the energy of a wrong correspondence to be. In practice, we set it to 1.

3. The predicted pose is relatively far away from the optimum, meaning that the projected point position will be a couple of pixels away from the correct location. As this typically happens during the beginning of the optimization we assume that \( \lambda \) will be relatively large and the algorithm behaves similar to gradient descent. In this case we want that the gradient of this point is oriented in the direction of the correct solution, so that the point has a positive influence on the update step.

For computing a loss function to enforce this property we sample a random negative correspondence \( p'_{\text{cor}} \) in a relatively large vicinity around the correct solution (in our experiments we use 5 pixels distance). Starting from this negative correspondence \( p'_{\text{cor}} \) we first compute the \( 2 \times 2 \) Gauss-Newton system for this individual point, similarly to how it is done for optical flow estimation using Lucas-Kanade:

\[
r_p(p, p_{\text{cor}}') = F'(p_{\text{cor}}') - F(p)
\]  

\[
J_p = \frac{dF'(p'_{\text{cor}})}{dp'_{\text{cor}}} \quad \text{and} \quad H_p = J_p^T J_p \quad \text{and} \quad b_p = J_p^T r_p \tag{10}
\]

We compute the damped system using a relatively large fixed \( \lambda_f \), as well as the optical flow step\(^*\)

\[
H'_p = H_p + \lambda_f I \quad \text{and} \quad p_{\text{after}} = p_{\text{cor}}' + H_p'^{-1} b_p. \tag{11}
\]

In order for this point to have a useful contribution to the direct image alignment, this update step should move in the correct direction by at least \( \delta \). We enforce this using a Gradient-Descent loss function which is small only if the distance to the correct correspondence after the update is smaller than before the update:

\[
E_{\text{GD}} = \max \left( \| p_{\text{after}} - p_{\text{gt}}' \|^2 - \| p_{\text{cor}}' - p_{\text{gt}}' \|^2 + \delta, 0 \right) \tag{12}
\]

In practice, we choose \( \lambda_f = 2.0 \) and \( \delta = 0.1 \).

4. The predicted pose is very close to the optimum, yielding a projected point position in very close proximity of the correct correspondence, and typically \( \lambda \) will be very small, so the update will mostly be a Gauss-Newton step. In this case we would like the algorithm to converge as quickly as possible, with subpixel accuracy. We enforce this using the Gauss-Newton loss [36]. To compute it we first sample a random negative correspondence \( p'_{\text{cor}} \), in a 1-pixel vicinity

\(^*\)Here we use Equation (4) instead of Equation (5) since we find it more stable for training LM-Net.
around the correct location. Then we use Equations (9) and (10), replacing $p_{P'}$ with $p_{P'}$ to obtain the Gauss-Newton system formed by $H_p$ and $b_p$. We compute the updated pixel location:

$$p_{after}' = p_{target}' + (H_p + \epsilon I)^{-1}b_p$$

Note that in contrast to the computation of the LM-Loss (Equation (12)), in this case $\epsilon$ is just added to ensure invertibility and therefore $\epsilon$ is much smaller than the $\lambda$ used above. The Gauss-Newton loss is computed with:

$$E_{GN} = \frac{1}{2}(p_{after}' - p_{gt})^TH_p(p_{after}' - p_{gt})$$

$$+ \log(2\pi) - \frac{1}{2} \log(|H_p|)$$

Note how all our 4 loss components use a different way to sample the involved points, depicted also in Figure 2. With the derivation above we argue that each loss component is important to achieve optimal performance and we demonstrate this in the results section. Note that the Gauss-Newton systems computed for the GD-Loss and the GN-Loss are very relevant for the application of direct image alignment. In fact the full Gauss-Newton system containing all points (Equation (3)), can be computed from these individual Gauss-Newton systems (Equation (10)) by simply summing them up and multiplying them with the derivative with respect to the pose [36].

### 3.3. CorrPoseNet

In order to deal with the large baselines between the images, we propose CorrPoseNet to regress the relative pose between two images $I$ and $I'$, which serves as the initialization of LM optimization. As our network shall work even in cases of large baselines and strong rotations, we utilize the correlation layer proposed in [27] which is known to boost the performance of affine image transformation and optical flow [21] estimation for large displacements, but has not been applied to pose estimation before.

Our network first computes deep features $f_{corr}$, $f'_{corr} \in \mathbb{R}^{h \times w \times c}$ from both images individually using multiple strided convolutions with ReLU activations in between. Then the correlation layer correlates each pixel from the normalized source features with each pixel from the normalized target features yielding the correlation map $c \in \mathbb{R}^{h \times w \times (h \times w)}$:

$$c(i, j, (i', j')) = f_{corr}(i, j)Tf'_{corr}(i', j')$$

The correlation map is then normalized in the channel dimension and fed into 2 convolutional layers each followed by batch norm and ReLU. Finally we regress the Euler angle $\eta^euler$ and translation $t$ using a fully connected layer. More details on the architecture are shown in the supplementary material.

We train CorrPoseNet from scratch with image pairs and groundtruth poses $t_{gt}$ and $R_{gt}$. We utilize an L2-loss working directly on Euler angles and translation:

$$E = ||t - t_{gt}||_2 + \lambda ||\eta^euler - \eta^euler_{gt}||_2,$$

where $\lambda$ is the weight, which we set to 10 in practice.

As the distribution of groundtruth poses in the Oxford training data is limited we apply the following data augmentation. We first generate dense depths for all training images using a state-of-the-art dense stereo matching algorithm [40]. The resulting depths are then used to warp the images to a different pose sampled from a uniform distribution. In detail, we first warp the depth image to the random target pose, then inpaint the depth image using the OpenCV implementation of Navier Stokes, and finally warp our image to the target pose using this depth map. Note that the dense depths are only necessary for training, not for evaluation. We show an ablation study on the usage of correlation layers and the proposed data augmentation in the supplementary material.

### 4. Experiments

We evaluate our method on the relocalization tracking benchmark proposed in [36], which contains images created with the CARLA simulator [9], and scenes from the Oxford RobotCar dataset [19]. We train our method on the respective datasets from scratch. LM-Net is trained using the Adam optimizer with a learning rate of $10^{-6}$ and for CorrPoseNet we use a learning rate of $10^{-4}$. For both networks we choose hyperparameters and epoch based on the results on the validation data. Our networks use the same hyperparameters for all experiments except where stated otherwise; the direct image alignment code is slightly adapted for Oxford RobotCar, mainly to improve performance when the ego-vehicle is standing still.

As the original relocalization tracking benchmark [36] does not include validation data on Oxford RobotCar we have manually aligned two new sequences, namely 2015-04-07-09-06-25 and 2015-05-19-14-06-38, and extend the benchmark with these sequences as validation data. **Evaluation metrics:** We evaluate the predicted translation $t_{est}$ and rotation $R_{est}$ against the ground-truth $t_{gt}$ and $R_{gt}$ according to Equations (17) and (18).

$$t_\Delta = ||t_{est} - t_{gt}||_2$$

$$R_\Delta = \arccos \left( \frac{\text{trace}(R_{est}^{-1}R_{gt}) - 1}{2} \right)$$

In this section, we plot the cumulative translation and rotation error until 0.5m and 0.5°, respectively. For quantitative results we compute the area under curve (AUC) of these
cumulative curves in percent, which we denote as $t_{AUC}$ for translation and $R_{AUC}$ for rotation from now on.

We evaluate the following direct methods:

**Ours:** The full LM-Reloc approach consisting of CorrPoseNet, LM-Net features and direct image alignment based on Levenberg-Marquardt. The depths used for the image alignment are estimated with the stereo version [37] of DSO [11].

**Ours (w/o CorrPoseNet):** For a more fair comparison to GN-Net we use identity as initialization for the direct image alignment instead of CorrPoseNet. This enables a direct comparison between the two loss formulations.

**GN-Net [36]:** In this work, we have also improved the parameters of the direct image alignment pipeline based on DSO [11]. Thus we have re-evaluated GN-Net with this improved pipeline to make the comparison as fair as possible. These re-evaluated results are better than the results computed in the original GN-Net paper.

**Baseline methods:** Additionally, we evaluate against current state-of-the-art indirect methods, namely SuperGlue [32], R2D2 [26], SuperPoint [8], and D2-Net [10]. For these methods, we estimate the relative pose using the models provided by the authors and the OpenCV implementation of solvePnP/Tranesac. We have tuned the parameters of RANSAC on the validation data and used 1000 iterations and a reprojection error threshold of 3 for all methods. For estimating depth values at keypoint locations we use OpenCV stereo matching. It would be possible to achieve a higher accuracy by using SfM and MVS solutions such as COLMAP [34]. However, one important disadvantage of these approaches is, that building a map is rather time consuming and computationally expensive, whereas all other approaches evaluated on the benchmark [36] are able to create the map close to real-time, enabling applications like long-term loop-closure and map-merging.

### 4.1. CARLA Relocalization Benchmark

Figure 3 depicts the results on the test data of the CARLA benchmark. For all methods we show the cumulative error plot for translation in meters and rotation in degree. It can be seen that our method is more accurate than the state-of-the-art while performing similarly in terms of robustness. We also show the AUC for the two Figures in Table 1. Compared to GN-Net it can be seen that our new loss formulation significantly improves the results.

| Method                  | $t_{AUC}$ | $R_{AUC}$ |
|-------------------------|-----------|-----------|
| Ours                    | **80.65** | **77.83** |
| SuperGlue [32]          | 78.99     | 59.31     |
| R2D2 [26]               | 73.47     | 54.42     |
| SuperPoint [8]          | 72.76     | 53.38     |
| D2-Net [10]             | 47.62     | 16.47     |
| Ours (w/o CorrPoseNet)  | 63.88     | 61.9      |
| GN-Net [36]             | 43.72     | 44.08     |
Table 2: Results on the Oxford RobotCar relocalization tracking benchmark [36]. We compare LM-Net (Ours) against other state-of-the-art methods (SuperGlue, R2D2, SuperPoint, and D2-Net). As can be seen from the results, our method almost consistently outperforms other SOTA approaches in terms of rotation AUC whilst achieving comparable results on translation AUC.

| Sequence             | Ours             | SuperGlue [32]  | R2D2 [26]        | SuperPoint [8] | D2-Net [10] |
|----------------------|------------------|-----------------|------------------|----------------|-------------|
|                      | $t_{AUC}$        | $R_{AUC}$       | $t_{AUC}$        | $R_{AUC}$       | $t_{AUC}$   |
| Sunny-Overcast       | 79.83            | 55.48           | 81.01            | 52.83           | 80.86       |
| Sunny-Rainy          | 71.54            | 43.7            | 75.58            | 45.99           | 74.84       |
| Sunny-Snowy          | 59.69            | 44.06           | 63.57            | 43.64           | 62.92       |
| Overcast-Rainy       | 80.54            | 63.7            | 79.99            | 61.64           | 81.29       |
| Overcast-Snowy       | 55.38            | 47.88           | 57.67            | 47.16           | 55.39       |
| Rainy-Snowy          | 68.57            | 41.67           | 69.91            | 39.87           | 71.79       |

Table 3: This table shows the results on the Oxford RobotCar relocalization tracking benchmark test data against GN-Net. Thanks to our LM-based loss formulation we consistently outperform GN-Net on all sequences.

| Sequence             | Ours (w/o CorrPoseNet) | GN-Net [36] |
|----------------------|-------------------------|-------------|
|                      | $t_{AUC}$ | $R_{AUC}$ | $t_{AUC}$ | $R_{AUC}$ |
| Sunny-Overcast       | 79.61     | 55.45     | 73.53    | 49.31    |
| Sunny-Rainy          | 70.46     | 42.86     | 64.58    | 37.27    |
| Sunny-Snowy          | 59.7      | 44.17     | 55.27    | 41.36    |
| Overcast-Rainy       | 79.67     | 63.08     | 75.72    | 60.13    |
| Overcast-Snowy       | 54.94     | 47.19     | 51.34    | 42.91    |
| Rainy-Snowy          | 66.23     | 39.93     | 62.63    | 36.2     |

Figure 4: This plot shows our ablation study for removing different loss parts on the CARLA relocalization tracking benchmark. Without the GD-loss the achieved robustness is reduced, whereas removing the GN-loss leads to decreased accuracy. Using our full loss formulation yields a large improvement.

4.2. Oxford RobotCar Relocalization Benchmark

We compare to the state-of-the-art indirect methods on the 6 test sequence pairs consisting of the sequences 2015-02-24-12-32-19 (sunny), 2015-03-17-11-08-44 (overcast), 2014-12-05-11-09-10 (rainy), and 2015-02-03-08-45-10 (snowy). In Table 2, we show the area under curve until 0.5 meters / 0.5 degrees for all methods. It can be seen that our method clearly outperforms the state-of-the-art in terms of rotation accuracy, while being competitive in terms of translation error. It should be noted that the ground-truth for these sequences was generated using ICP alignment of the 2D-LiDAR data accumulated for 60 meters. We have computed that the average root mean square error of the ICP alignment is 16 centimeters. Therefore, especially the ground-truth translations have limited accuracy. As can be seen from Figure 3, the accuracy improvements our method provides are especially visible in the range below 0.15 meters which is hard to measure on this dataset. The rotation error of LiDAR alignment is lower than the translational one, which is why we clearly observe the improvements of our method on the rotations.

In Table 3, we compare LM-Net without the CorrPoseNet to GN-Net. Due to our novel loss formulation LM-Net outperforms the competitor on all sequences significantly.

4.3. Ablation Studies

We evaluate LM-Net on the CARLA validation data with and without the various losses (Figure 4). Compared to a normal contrastive loss, the given loss formulation is a large improvement. As expected, $E_{GD}$ (green line) mainly improves the robustness, whereas $E_{GN}$ (blue line) improves the accuracy. Only when used together (our method) we achieve large robustness and large accuracy, confirming our theoretical derivation in Section 3.
4.4. Qualitative Results

To demonstrate the accuracy of our approach in practice, we show qualitative results on the Oxford RobotCar dataset. We track the snowy test sequence 2015-02-03-08-45-10 using Stereo DSO [37] and at the same time perform relocalization against the sunny reference map 2015-02-24-12-32-19. Relocalization between the current keyframe and the closest map image is performed using LM-Net. Initially, we give the algorithm the first corresponding map image (which would in practice be provided by an image retrieval approach such as NetVLAD [3]). Afterwards we find the closest map image for each keyframe using the previous solution for the transformation between the map and the current SLAM world $T_{w,m}$. We visualize the current point cloud (blue) and the point cloud from the map (grey) overlayed using the smoothed $T_{w,m}$ (Figure 5). The point clouds will align only if the relocalization is accurate. As can be seen in Figure 5, the lane markings, poles, and buildings between the reference and query map align well, hence qualitatively showing the high relocalization accuracy of our method. We recommend watching the video at https://vision.in.tum.de/lm-reloc. In Figure 6 we show example images from the benchmark.

5. Conclusion

We have presented LM-Reloc as a novel approach for direct visual localization. In order to estimate the relative 6DoF pose between two images from different conditions, our approach performs direct image alignment on the trained features from LM-Net without relying on feature matching or RANSAC. In particular, with the loss function designed seamlessly for the Levenberg-Marquart algorithm, LM-Net provides deep feature maps that coin the characteristics of direct image alignment and are also invariant to changes in lighting and appearance of the scene. The experiments on the CARLA and Oxford RobotCar relocalization tracking benchmark exhibit the state-of-the-art performance of our approach. In addition, the ablation studies also show the effectiveness of the different components of LM-Reloc.
[39] N. Yang, R. Wang, X. Gao, and D. Cremers. Challenges in monocular visual odometry: Photometric calibration, motion bias, and rolling shutter effect. RA-L, 3(4):2878–2885, 2018.

[40] F. Zhang, V. Prisacariu, R. Yang, and P. H. Torr. Ga-net: Guided aggregation net for end-to-end stereo matching. In CVPR, pages 185–194, 2019.

[41] X. Zheng, Z. Moratto, M. Li, and A. I. Mourikis. Photometric patch-based visual-inertial odometry. In ICRA, pages 3264–3271, 2017.
LM-Reloc: Levenberg-Marquardt Based Direct Visual Relocalization: 
Supplementary Material

Lukas von Stumberg$^{1,2*}$  Patrick Wenzel$^{1,2*}$  Nan Yang$^{1,2}$  Daniel Cremers$^{1,2}$
$^1$ Technical University of Munich  $^2$ Artisense

1. Video

As mentioned in the paper, we provide a video of the qualitative relocalization demo, which is available at https://vision.in.tum.de/lm-reloc.

2. Network Architecture

CorrPoseNet. The CorrPoseNet takes 2 images (I and I') as the input and outputs the relative pose $R, t$ between those images. The overall network architecture of the CorrPoseNet is depicted in Figure 1. The convolutional blocks consist of in total 9 convolutional layers followed by ReLU activations. The architectural details of the convolutional blocks are listed in Table 1. The correlation layer which takes the output of the convolutional blocks as input is described in the main paper. The correlation layer is followed by the regression block which regresses the relative pose. The layers of the regression block are listed in Table 2. The output of the network is the rotation $R$ as Euler angles and translation $t$.

Table 1: Network architecture and parameters of the convolutional blocks. k denotes kernel size, s stride, and p padding.

| Convolutional blocks | layer | in-chns | out-chns | k | s | p | activation |
|----------------------|-------|---------|----------|----|---|---|------------|
| conv0                | 3     | 16      | 16       | 2  | 3 |   | ReLU       |
| conv1                | 16    | 32      | 5        | 2  | 2 |   | ReLU       |
| conv2                | 32    | 64      | 3        | 2  | 1 |   | ReLU       |
| conv3                | 64    | 64      | 3        | 1  | 0 |   | ReLU       |
| conv4                | 64    | 128     | 3        | 2  | 2 |   | ReLU       |
| conv5                | 128   | 128     | 3        | 1  | 1 |   | ReLU       |
| conv6                | 128   | 256     | 3        | 2  | 1 |   | ReLU       |
| conv7                | 256   | 256     | 3        | 1  | 1 |   | ReLU       |

LM-Net. We adopt U-Net [1] as the encoder of LM-Net. However, we change the decoder part of the architecture in the following way. Starting from the coarsest level, we upsample (with bilinear interpolation) the feature maps by 2 and concatenate those feature maps with the feature map of the higher level. This is followed by 1 × 1 convolutional filters. This procedure is repeated 4 times. This results in the feature pyramid maps as described in Table 3.

3. Ablation Study Correlation Layer

We demonstrate the impact of the Correlation layer in the proposed CorrPoseNet. We compare it to a simpler pose estimation network where the correlation and regression layers are replaced with two 1 × 1 convolutions with 3 output-channels each, which directly regress rotation and Euler angles. This simpler PoseNet has one more convo-
Table 2: Network architecture and parameters of the regression block. $k$ denotes kernel size, $s$ stride, and $p$ padding. $N_c = 256$ denotes the input channels for the CARLA model, and $N_o = 260$ denotes the input channels for the Oxford model, respectively.

| Regression block | in-chns | out-chns | k | s | p | activation |
|------------------|---------|----------|---|---|---|------------|
| conv0            | $N_c$ / $N_o$ | 128      | 7 | 1 | 0 | ReLU       |
| BN               | 128     | 128      | - | - | - |            |
| conv1            | 128     | 64       | 5 | 1 | 0 | ReLU       |
| BN               | 64      | 64       | - | - | - |            |
| FC               | 2304    | 6        | - | - | - |            |

Table 3: Output of the decoder of LM-Net. $H$, and $W$ denote height and width of the feature maps.

| Decoder layer | Output size     |
|---------------|-----------------|
| $F_1$         | $16 \times H/8 \times W/8$ |
| $F_2$         | $16 \times H/4 \times W/4$ |
| $F_3$         | $16 \times H/2 \times W/2$ |
| $F_4$         | $16 \times H \times W$     |

4. Ablation Study Oxford Data Augmentation

We show the impact of the data augmentation for the Oxford RobotCar Relocalization benchmark, where we warp the images to different poses using dense depths in Table 4. It can be seen that the proposed augmentation improves translation and rotation error on the validation data.

References

[1] O. Ronneberger, P. Fischer, and T. Brox. U-Net: Convolutional Networks for Biomedical Image Segmentation. In MICCAI, 2015.

[2] L. von Stumberg, P. Wenzel, Q. Khan, and D. Cremers. GNNet: The Gauss-Newton Loss for Multi-Weather Relocalization. RA-L, 5(2):890–897, 2020.