Taking a Deeper Look at the Inverse Compositional Algorithm

Zhaoyang Lv[1,2]   Frank Dellaert[1]   James M. Rehg[1]   Andreas Geiger[2]

[1]Georgia Institute of Technology, Atlanta, United States

[2]Autonomous Vision Group, MPI-IS and University of Tübingen, Germany

{zhaoyang.lv, rehg}@gatech.edu   frank.dellaert@cc.gatech.edu   andreas.geiger@tue.mpg.de

Abstract

In this paper, we provide a modern synthesis of the classic inverse compositional algorithm for dense image alignment. We first discuss the assumptions made by this well-established technique, and subsequently propose to relax these assumptions by incorporating data-driven priors into this model. More specifically, we unroll a robust version of the inverse compositional algorithm and replace multiple components of this algorithm using more expressive models whose parameters we train in an end-to-end fashion from data. Our experiments on several challenging 3D rigid motion estimation tasks demonstrate the advantages of combining optimization with learning-based techniques, outperforming the classic inverse compositional algorithm as well as data-driven image-to-pose regression approaches.

1. Introduction

Since the seminal work by Lucas and Kanade [32], dense image alignment has become an ubiquitous tool in computer vision with many applications including stereo reconstruction [41], tracking [5, 42, 50], image registration [8, 29, 43], super-resolution [22] and SLAM [13, 16, 17]. In this paper, we provide a learning-based perspective on the Inverse Compositional algorithm, an efficient variant of the original Lucas-Kanade image registration technique. In particular, we lift some of the restrictive assumptions by parameterizing several components of the algorithm using neural networks and training the entire optimization process end-to-end. In order to put contributions into context, we will now briefly review the Lucas-Kanade algorithm, the Inverse Compositional algorithm, as well as the robust M-Estimator which form the basis for our model. More details can be found in the comprehensive reviews of Baker et al. [2, 3].

Lucas-Kanade Algorithm: The Lucas-Kanade algorithm minimizes the photometric error between a template and an image. Letting $\mathbf{T} : \Xi \to \mathbb{R}^{W \times H}$ and $\mathbf{I} : \Xi \to \mathbb{R}^{W \times H}$

denote the warped template and image, respectively[1], the Lucas-Kanade objective can be stated as follows

$$\min_{\boldsymbol{\xi}} \|\mathbf{I}(\boldsymbol{\xi}) - \mathbf{T}(\mathbf{0})\|_2^2 \tag{1}$$

where $\mathbf{I}(\boldsymbol{\xi})$ denotes image $\mathbf{I}$ transformed using warp parameters $\boldsymbol{\xi}$ and $\mathbf{T}(\mathbf{0}) = \mathbf{T}$ denotes the original template.

Minimizing (1) is a non-linear optimization task as the image $\mathbf{I}$ depends non-linearly on the warp parameters $\boldsymbol{\xi}$. The Lucas-Kanade algorithm therefore iteratively solves for the warp parameters $\boldsymbol{\xi}_{k+1} = \boldsymbol{\xi}_k \circ \Delta\boldsymbol{\xi}$. At every iteration k, the warp increment $\Delta\boldsymbol{\xi}$ is obtained by linearizing

$$\min_{\Delta\boldsymbol{\xi}} \|\mathbf{I}(\boldsymbol{\xi}_k + \Delta\boldsymbol{\xi}) - \mathbf{T}(\mathbf{0})\|_2^2 \tag{2}$$

using first-order Taylor expansion

$$\min_{\Delta\boldsymbol{\xi}} \left\| \mathbf{I}(\boldsymbol{\xi}_k) + \frac{\partial \mathbf{I}(\boldsymbol{\xi}_k)}{\partial \boldsymbol{\xi}} \Delta\boldsymbol{\xi} - \mathbf{T}(\mathbf{0}) \right\|_2^2 \tag{3}$$

Note that the "steepest descent image" $\partial\mathbf{I}(\boldsymbol{\xi}_k)/\partial\boldsymbol{\xi}$ needs to be recomputed at every iteration as it depends on $\boldsymbol{\xi}_k$.

Inverse Compositional Algorithm: The *inverse compositional* (IC) algorithm [3] avoids this by applying the warp increments $\Delta\boldsymbol{\xi}$ to the template instead of the image

$$\min_{\Delta\boldsymbol{\xi}} \|\mathbf{I}(\boldsymbol{\xi}_k) - \mathbf{T}(\Delta\boldsymbol{\xi})\|_2^2 \tag{4}$$

using the warp parameter update $\boldsymbol{\xi}_{k+1} = \boldsymbol{\xi}_k \circ (\Delta\boldsymbol{\xi})^{-1}$. In the corresponding linearized equation

$$\min_{\Delta\boldsymbol{\xi}} \left\| \mathbf{I}(\boldsymbol{\xi}_k) - \mathbf{T}(\mathbf{0}) - \frac{\partial \mathbf{T}(\mathbf{0})}{\partial \boldsymbol{\xi}} \Delta\boldsymbol{\xi} \right\|_2^2 \tag{5}$$

$\partial\mathbf{T}(\mathbf{0})/\partial\boldsymbol{\xi}$ does not depend on $\boldsymbol{\xi}_k$ and can thus be precomputed, resulting in a more efficient algorithm.

---

[1]The warping function $\mathcal{W}_{\boldsymbol{\xi}} : \mathbb{R}^2 \to \mathbb{R}^2$ might represent translation, affine 2D motion or (if depth is available) rigid or non-rigid 3D motion. To avoid clutter in the notation, we do not make $\mathcal{W}_{\boldsymbol{\xi}}$ explicit in our equations.

Robust M-Estimation: To handle outliers or ambiguities (e.g., multiple motions), robust estimation [19, 54] can be used. The robust version of the IC algorithm [2] has the following objective function

$$\min_{\Delta\boldsymbol{\xi}} \mathbf{r}_k(\Delta\boldsymbol{\xi})^T \mathbf{W} \mathbf{r}_k(\Delta\boldsymbol{\xi}) \tag{6}$$

where $\mathbf{r}_k(\Delta\boldsymbol{\xi}) = \mathbf{I}(\boldsymbol{\xi}_k) - \mathbf{T}(\Delta\boldsymbol{\xi})$ is the residual between image $\mathbf{I}$ and template $\mathbf{T}$ at the k'th iteration, and $\mathbf{W}$ is a diagonal weight matrix that depends on the residual[2] and is chosen based on the desired robust loss function ρ [54].

Optimization: The minimizer of (6) after linearization is obtained as the Gauss-Newton update step [6]

$$(\mathbf{J}^T\mathbf{W}\mathbf{J})\Delta\boldsymbol{\xi} = \mathbf{J}^T\mathbf{W}\mathbf{r}_k(\mathbf{0}) \tag{7}$$

where $\mathbf{J} = \partial\mathbf{T}(\mathbf{0})/\partial\boldsymbol{\xi}$ is the Jacobian of the template $\mathbf{T}(\mathbf{0})$ with respect to the warp parameters $\boldsymbol{\xi}$. As the approximate Hessian $\mathbf{J}^T\mathbf{W}\mathbf{J}$ easily becomes ill-conditioned, a damping term is added in practice. This results in the popular Levenberg–Marquardt (trust-region) update equation [35]:

$$\Delta\boldsymbol{\xi} = (\mathbf{J}^T\mathbf{W}\mathbf{J} + \lambda\,\text{diag}(\mathbf{J}^T\mathbf{W}\mathbf{J}))^{-1}\mathbf{J}^T\mathbf{W}\mathbf{r}_k(\mathbf{0}) \tag{8}$$

For different values of λ, the parameter update $\Delta\boldsymbol{\xi}$ varies between the Gauss-Newton direction and gradient descent. In practice, λ is chosen based on simple heuristics.

Limitations: Despite its widespread utility, the IC method suffers from a number of important limitations. First, it assumes that the linearized residual leads to an update which iteratively reaches a good local optimum. However, this assumption is invalid in the presence of high-frequency textural information or noise in $\mathbf{I}$ or $\mathbf{T}$. Second, choosing a good robust loss function ρ is difficult as the true data/residual distribution is often unknown. Moreover, Equation (6) does not capture correlations or higher-order statistics in the inputs $\mathbf{I}$ and $\mathbf{T}$ as the residuals operate directly on the pixel values and the weight matrix $\mathbf{W}$ is diagonal. Finally, damping heuristics do not fully exploit the information available during optimization and thus lead to suboptimal solutions.

Contributions: In this paper, we propose to combine the best of both (optimization and learning-based) worlds by unrolling the robust IC algorithm into a more general parameterized feed-forward model which is trained end-to-end from data. In contrast to generic neural network estimators, this allows our algorithm to incorporate knowledge about the structure of the problem (e.g., family of warping functions, 3D geometry) as well as the advantages of a robust iterative estimation framework. At the same time, our approach relaxes the restrictive assumptions made in the original IC formulation [3] by incorporating trainable modules and learning the entire model end-to-end.

More specifically, we make the following contributions:

(A) We propose a **Two-View Feature Encoder** which replaces $\mathbf{I}, \mathbf{T}$ with feature representations $\mathbf{I}_\theta, \mathbf{T}_\theta$ that jointly encode information about the input image $\mathbf{I}$ and the template $\mathbf{T}$. This allows our model to exploit spatial as well as temporal correlations in the data.

(B) We propose a **Convolutional M-Estimator** that replaces $\mathbf{W}$ in (6) with a learned weight matrix $\mathbf{W}_\theta$ which encodes information about $\mathbf{I}, \mathbf{T}$ and $\mathbf{r}_k$ in a way such that the unrolled optimization algorithm ignores irrelevant or ambiguous information as well as outliers.

(C) We propose a **Trust Region Network** which replaces the damping matrix $\lambda\,\text{diag}(\mathbf{J}^T\mathbf{W}\mathbf{J})$ in (8) with a learned damping matrix $\text{diag}(\boldsymbol{\lambda}_\theta)$ whose diagonal entries $\boldsymbol{\lambda}_\theta$ are estimated from "residual volumes" which comprise residuals of a Levenberg-Marquardt update when applying a range of hypothetical λ values.

We demonstrate the advantages of combining the classical IC method with deep learning on the task of 3D rigid motion estimation using several challenging RGB-D datasets. We also provide an extensive ablation study about the relevance of each model component that we propose. Results on traditional affine 2D image alignment tasks are provided in the supplementary material. Our implementation is publicly accessible.[3]

2. Related Work

We are not the first to inject deep learning into an optimization pipeline. In this section, we first review classical methods, followed by direct pose regression techniques and related work on learning-based optimization.

Classical Methods: Direct methods [18, 21] that align images using the sum-of-square error objective (1) are prone to outliers and varying illuminations. Classical approaches address this problem by exploiting more robust objective functions [32, 37], heuristically chosen patch- [47] or gradient-based [29] features, and photometric calibration as a preprocessing step [16]. The most common approach is to use robust estimation (6) as in [7]. However, the selection of a good robust function ρ is challenging and traditional formulations assume that the same function applies to all pixels, ignoring correlations in the inputs. Moreover, the inversion of the linearized system (7) may still be ill-conditioned [1]. To overcome this problem, soft constraints in the form of damping terms (8) can be added to the objective [2,3]. However, this may bias the system to sub-optimal solutions.

This paper addresses these problems by relaxing the main assumptions of the Inverse Compositional (IC) algorithm [2, 3] using data-driven learning. More specifically, we propose to learn the feature representation (A), robust

---

[2]We omit this dependency to avoid clutter in the notation.

estimator (B) and damping (C) jointly to replace the traditional heuristic rules of classical algorithms.

Direct Pose Regression: A notably different approach to classical optimization techniques is to directly learn the entire mapping from the input to the warping parameters $\boldsymbol{\xi}$ from large amounts of data, spanning from early work using linear hyperplane approximation [23] to recent work using deep neural networks [9, 49, 51, 53, 58]. Prominent examples include single image to camera pose regression [25, 26], image-based 3D object pose estimation [34, 46] and relative pose prediction from two views [9, 49]. However, learning a direct mapping requires high-capacity models and large amounts of training data. Furthermore, obtaining pixel-accurate registrations remains difficult and the learned representations do not generalize well to new domains.

To improve accuracy, recent methods adopt cascaded networks [20, 52] and iterative feedback [24]. Lin et al. [31] combines the multi-step iterative spatial transformer network (STN) with the classical IC algorithm [2, 3] for aligning 2D images. Variants of this approach have recently been applied to various 3D tasks: Li et al. [30] proposes to iteratively align a 3D CAD model to an image. Zhou et al. [56] jointly train for depth, pose and optical flow.

Different from [31] and its variants which approximate the pseudo-inverse of the Jacobian implicitly using stacked convolutional layers, we exploit the structure of the optimization problem and explicitly solve the original robust objective (6) with learned modules using few parameters.

Learning-based Optimization: Recently, several methods have exploited the differentiable nature of iterative optimization algorithms by unrolling for a fixed number of iterations. Each iteration is treated as a layer in a neural network [12, 28, 36, 39, 40, 55]. In this section, we focus on the most related work which also tackles the least-squares optimization problem [13, 38, 48, 51]. We remark that most of these techniques can be considered special cases of our more general deep IC framework.

Wang et al. [51] address the 2D image tracking problem by learning an input representation using a two-stream Siamese network for the IC setting. In contrast to us, they exploit only spatial but not temporal correlations in the inputs (A), leverage a formulation which is not robust (B) and do not exploit trust-region optimization (C).

Clark et al. [13] propose to jointly learn depth and pose estimation by minimizing photometric error using the formulation in (4). In contrast to us, they do not learn feature representations (A) and neither employ a robust formulation (B) nor trust-region optimization (C).

Ranftl et al. [38] propose to learn robust weights for *sparse* feature correspondences and apply their model to fundamental matrix estimation. As they do not target direct image alignment, their problem setup is different from ours.

Besides, they neither learn input features (A) nor leverage trust-region optimization (C). Instead, they solve their optimization problem using singular value decomposition.

Concurrent to our work, Tang and Tan [48] propose a photometric Bundle Adjustment network by learning to align feature spaces for monocular reconstruction of a static scene. Different from us, they did not exploit temporal correlation in the inputs (A) and do not employ a robust formulation (B). While they propose to learn the damping parameters (C), in contrast to our trust-region volume formulation, they regress the damping parameters from the global average pooled residuals.

3. Method

This section describes our model. A high-level overview over the proposed unrolled inverse-compositional algorithm is given in Fig. 1. Using the same notation as in Section 1, our goal is to minimize the error by warping the image towards the template, similar to (6):

$$\min_{\boldsymbol{\xi}} \ \mathbf{r}(\boldsymbol{\xi})^T \mathbf{W}_\theta \, \mathbf{r}(\boldsymbol{\xi}) \tag{9}$$

The difference to (6) is that in our formulation, the weight matrix $\mathbf{W}_\theta$ as well as the template $\mathbf{T}_\theta(\boldsymbol{\xi})$ and the image $\mathbf{I}_\theta(\boldsymbol{\xi})$ (and thus also the residual $\mathbf{r}(\boldsymbol{\xi}) = \mathbf{I}_\theta(\boldsymbol{\xi}) - \mathbf{T}_\theta(\mathbf{0})$) depend on the parameters of a learned model θ.

We exploit the inverse compositional algorithm to solve the non-linear optimization problem in (9), i.e., we linearize the objective and iteratively update the warp parameters $\boldsymbol{\xi}$:

$$\boldsymbol{\xi}_{k+1} = \boldsymbol{\xi}_k \circ (\Delta\boldsymbol{\xi})^{-1} \tag{10}$$

$$\Delta\boldsymbol{\xi} = \left(\mathbf{J}^T \mathbf{W}_\theta \mathbf{J} + \mathrm{diag}(\boldsymbol{\lambda}_\theta)\right)^{-1} \mathbf{J}^T \mathbf{W}_\theta \, \mathbf{r}_k \tag{11}$$

$$\mathbf{r}_k = \mathbf{I}_\theta(\boldsymbol{\xi}_k) - \mathbf{T}_\theta(\mathbf{0}) \tag{12}$$

$$\mathbf{J} = \partial\mathbf{T}_\theta(\mathbf{0})/\partial\boldsymbol{\xi} \tag{13}$$

starting from $\boldsymbol{\xi}_0 = \mathbf{0}$.

The most notable change from (8) is that the image features $(\mathbf{I}_\theta, \mathbf{T}_\theta)$, the weight matrix $(\mathbf{W}_\theta)$ and the damping factors $\boldsymbol{\lambda}_\theta$ are predicted by learned functions which have been collectively parameterized by $\theta = \{\theta_\mathbf{I}, \theta_\mathbf{W}, \theta_\boldsymbol{\lambda}\}$. Note that also the residual $\mathbf{r}_k$ as well as the Jacobian $\mathbf{J}$ implicitly depend on the parameters θ, though we have omitted this dependence here for notational clarity.

We will now provide details about these mappings.

(A) Two-View Feature Encoder: We use a fully convolutional neural network ϕ_θ to extract feature maps from the image $\mathbf{I}$ and the template $\mathbf{T}$:

$$\mathbf{I}_\theta = \phi_\theta([\mathbf{I}, \mathbf{T}]) \tag{14}$$

$$\mathbf{T}_\theta = \phi_\theta([\mathbf{T}, \mathbf{I}]) \tag{15}$$

Here, the operator $[\cdot, \cdot]$ indicates concatenation along the feature dimension. Instead of $\mathbf{I}$ and $\mathbf{T}$ we then feed $\mathbf{I}_\theta$
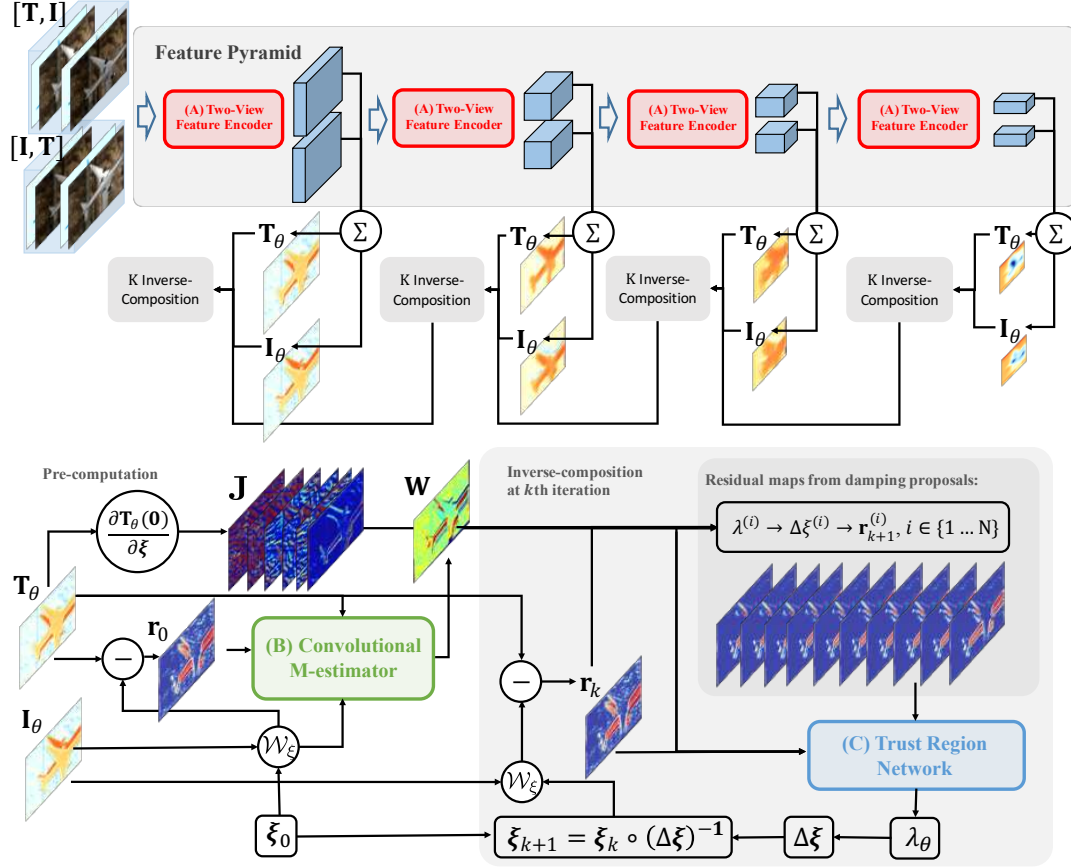
Figure 1: **High-level Overview of our Deep Inverse Compositional (IC) Algorithm.** We stack $[\mathbf{I}, \mathbf{T}]$ and $[\mathbf{T}, \mathbf{I}]$ as inputs to our **(A) Two-View Feature Encoder** pyramid which extracts 1-channel feature maps $\mathbf{T}_\theta$ and $\mathbf{I}_\theta$ at multiple scales using channel-wise summation. We then perform K IC steps at each scale using $\mathbf{T}_\theta$ and $\mathbf{I}_\theta$ as input. At the beginning of each scale, we pre-compute $\mathbf{W}$ using our **(B) Convolutional M-estimator**. For each of the K IC iterations, we compute the warped image $\mathbf{I}(\boldsymbol{\xi}_k)$ and $\mathbf{r}_k$. Subsequently, we sample N damping proposals $\lambda^{(i)}$ and compute the proposed residual maps $\mathbf{r}_{k+1}^{(i)}$. Our **(C) Trust Region Network** takes these residual maps as input and predicts λ_θ for the trust region update step.

and $\mathbf{T}_\theta$ to the residuals in (12) and to the Jacobian in (13). Note that we use the notation $\mathbf{I}_\theta(\boldsymbol{\xi}_k)$ in (12) to denote that the feature map $\mathbf{I}_\theta$ is warped by a warping function that is parameterized via $\boldsymbol{\xi}$. More formally, this can be stated as $\mathbf{I}_\theta(\boldsymbol{\xi}) = \mathbf{I}_\theta(\mathcal{W}_{\boldsymbol{\xi}}(\mathbf{x}))$, where $\mathbf{x} \in \mathbb{R}^2$ denote pixel locations and $\mathcal{W}_{\boldsymbol{\xi}} : \mathbb{R}^2 \rightarrow \mathbb{R}^2$ is a warping function that maps a pixel location to another pixel location. For instance, $\mathcal{W}_{\boldsymbol{\xi}}$ may represent the space of 2D translations or affine transformations. In our experiments, we will focus on challenging 3D rigid body motions, using RGB-D inputs and representing $\boldsymbol{\xi}$ as an element of the Euclidean group $\boldsymbol{\xi} \in \mathfrak{se}(3)$.

Note that compared to directly using the image $\mathbf{I}$ and $\mathbf{T}$ as input, our features capture high-order *spatial* correlations in the data, depending on the receptive field size of the convolutional network. Moreover, they also capture *temporal* information as they operate on both $\mathbf{I}$ and $\mathbf{T}$ as input.

(B) Convolutional M-Estimator: We parameterize the weight matrix $\mathbf{W}_\theta$ as a diagonal matrix whose elements are

determined by a fully convolutional network ψ_θ that operates on the feature maps and the residual:

$$\mathbf{W}_\theta = \mathrm{diag}(\psi_\theta(\mathbf{I}_\theta(\boldsymbol{\xi}_k), \mathbf{T}_\theta(\mathbf{0}), \mathbf{r}_k)) \qquad (16)$$

as is illustrated in Fig. 1. Note that this enables our algorithm to reason about relevant image information while capturing spatial-temporal correlations in the inputs which is not possible with classical M-Estimators. Furthermore, we do not restrict the implicit robust function ρ to a particular error model, but instead condition ρ itself on the input. This allows for learning more expressive noise models.

(C) Trust Region Network: For estimating the damping $\boldsymbol{\lambda}_\theta$ we use a fully-connected network as illustrated in Fig. 1. We first sample a set of scalar damping proposals λ_i on a logarithmic scale and compute the resulting Levenberg-Marquardt update step as

$$\Delta \boldsymbol{\xi}_i = (\mathbf{J}^T \mathbf{W} \mathbf{J} + \lambda_i \, \mathrm{diag}(\mathbf{J}^T \mathbf{W} \mathbf{J}))^{-1} \mathbf{J}^T \mathbf{W} \, \mathbf{r}_k(\mathbf{0}) \quad (17)$$

We stack the resulting N residual maps

$$\mathbf{r}_{k+1}^{(i)} = \mathbf{I}_\theta(\boldsymbol{\xi}_k \circ (\Delta\boldsymbol{\xi}_i)^{-1}) - \mathbf{T}_\theta(\mathbf{0}) \qquad (18)$$

into a single feature map, flatten it, and pass it to a fully connected neural network ν_θ that outputs the damping parameters $\boldsymbol{\lambda}_\theta$:

$$\boldsymbol{\lambda}_\theta = \nu_\theta \left(\mathbf{J}^T\mathbf{W}\mathbf{J}, \left[\mathbf{J}^T\mathbf{W}\mathbf{r}_{k+1}^{(1)}, \dots, \mathbf{J}^T\mathbf{W}\mathbf{r}_{k+1}^{(N)} \right] \right) \quad (19)$$

The intuition behind our trust region networks is that the residuals predicted using the Levenberg-Marquardt update comprise valuable information about the damping parameter itself. This is empirically confirmed by our experiments.

Coarse-to-Fine Estimation: To handle large motions, it is common practice to apply direct methods in a coarse-to-fine fashion. We apply our algorithm at four pyramid levels with three iterations each. Our entire model including coarse-to-fine estimation is illustrated in Fig. 1. We extract features at all four scales using a single convolutional neural network with spatial average pooling between pyramid levels. We start with $\boldsymbol{\xi} = \mathbf{0}$ at the coarsest pyramid level, perform 3 iterations of our deep IC algorithm, and proceed with the next level until we reach the original image resolution.

Training and Inference: For training and inference, we unroll the iterative algorithm in equations (10)-(13). We obtain the gradients of the resulting computation graph using auto-differentiation. Details about the network architectures which we use can be found in the supplementary material.

4. Experiments

We perform our experiments on the challenging task of 3D rigid body motion estimation from RGB-D inputs[4]. Apart from the simple scenario of purely static scenes where only the camera is moving, we also consider scenes where both the camera as well as objects are in motion, hence resulting in strong ambiguities. We cast this as a supervised learning problem: using the ground truth motion, we train the models to resolve these ambiguities by learning to focus either on the foreground or the background.

Warping Function: Given pixel $\mathbf{x} \in \mathbb{R}^2$, camera intrinsics $\mathbf{K}$ and depth $D(\mathbf{x})$, we define the warping $\mathcal{W}_{\boldsymbol{\xi}}(\mathbf{x})$ induced by rigid body transform $\mathbf{T}_{\boldsymbol{\xi}}$ with $\boldsymbol{\xi} \in \mathfrak{se}(3)$ as

$$\mathcal{W}_{\boldsymbol{\xi}}(\mathbf{x}) = \mathbf{K}\,\mathbf{T}_{\boldsymbol{\xi}}\,D(\mathbf{x})\mathbf{K}^{-1}\,\mathbf{x} \qquad (20)$$

Using the warped coordinates, compute $\mathbf{I}_\theta(\boldsymbol{\xi})$ via bilinear sampling from $\mathbf{I}_\theta$ and set the warped feature value to zero for all occluded areas (estimated via z-buffering).

---

[4]Additional experiments on classical affine 2D motion estimation tasks can be found in the supplementary material.

Training Objective: To balance the influences of translation and rotation we follow [30] and exploit the 3D End-Point-Error (EPE) as loss function. Let $\mathbf{p} = D(\mathbf{x})\mathbf{K}^{-1}\mathbf{x}$ denote the 3D point corresponding to pixel $\mathbf{x}$ in image $\mathbf{I}$ and let $\mathcal{P}$ denote the set of all such 3D points. We minimize the following loss function

$$\mathbb{L} = \frac{1}{|\mathcal{P}|} \sum_{l \in \mathcal{L}} \sum_{\mathbf{p} \in \mathcal{P}} \| \mathbf{T}_{gt}\,\mathbf{p} - \mathbf{T}(\boldsymbol{\xi}_l)\,\mathbf{p} \|_2^2 \qquad (21)$$

where $\mathcal{L}$ denotes the set of coarse-to-fine pyramid levels (we apply our loss at the final iteration of every pyramid level) and $\mathbf{T}_{gt}$ is the ground truth transformation.

Implementation: We use the Sobel operator to compute the gradients in $\mathbf{T}$ and analytically derive $\mathbf{J}$. We calculate the matrix inverse on the CPU since we observed that inverting a small dense matrices $\mathbf{H} \in \mathbb{R}^{6 \times 6}$ is significantly faster on the CPU than on the GPU. In all our experiments we use $N = 10$ damping proposals for our Trust Region Network, sampled uniformly in logscale between $[10^{-5}, 10^5]$. We use four coarse-to-fine pyramid levels with three iterations each. We implemented our model and the baselines in PyTorch. All experiments start with a fixed learning rate of 0.0005 using ADAM [27]. We train a total of 30 epochs, reducing the learning rate at epoch [5,10,15].

4.1. Datasets

We systematically train and evaluate our method on four datasets which we now briefly describe.

MovingObjects3D: For the purpose of systematically evaluating highly varying object motions, we downloaded six categories of 3D models from ShapeNet [10]. For each object category, we rendered 200 video sequences with 100 frames in each sequence using Blender. We use data rendered from the categories 'boat' and 'motorbike' as test set and data from categories 'aeroplane', 'bicycle', 'bus', 'car' as training set. From the training set we use the first 95% of the videos for training and the remaining 5% for validation. In total, we obtain 75K images for training, 5K images for validation, and 25K for testing. We further subsample the sequences using sampling intervals $\{1, 2, 4\}$ in order to obtain small, medium and large motion subsets.

For each rendered sequence, we randomly select one 3D object model within the chosen category and stage it in a static 3D cuboid room with random wall textures and four point light sources. We randomly choose the camera viewpoint, point light source position and object trajectory, see supplementary material for details. This ensures diversity in object motions, textures and illumination. The videos also contain frames where the object is only partially visible. We exclude all frames where the entire object is not visible.

BundleFusion: To evaluate camera motion estimation in a static environment, we use the eight publicly released

scenes from BundleFusion[5] [14] which provide fully synchronized RGB-D sequences. We hold out 'copyroom' and 'office2' scenes for test and split the remaining scenes into training (first 95% of each trajectory) and validation (last 5%). We use the released camera trajectories as ground truth. We subsampled frames at intervals $\{2, 4, 8\}$ to increase motion magnitudes and hence the level of difficulty.

DynamicBundleFusion: To further evaluate camera motion estimation under heavy occlusion and motion ambiguity, we use the DynamicBundleFusion dataset [33] which augments the scenes from BundleFusion with non-rigidly moving human subjects as distractors. We use the same training, validation and test split as above. We train and evaluate frames subsampled at intervals $\{1, 2, 5\}$ due to the increased difficulty of this task.

TUM RGB-D SLAM: We evaluate our camera motion estimates on the TUM RGB-D SLAM dataset [45]. We hold out 'fr1/360', 'fr1/desk', 'fr2/desk' and 'fr2/pioneer_360' for testing and split the remaining trajectories into training (first 95% of each trajectory) and validation (last 5%). We randomly sample frame intervals from {1,2,4,8}. All images are resized to 160×120 pixels and depth values outside the range [0.5m, 5.0m] are considered invalid.

4.2. Baselines

We implemented the following baselines.

ICP: We use classical *Point-to-Plane ICP* [11] and *Point-to-Point-ICP* [4] implemented in Open3D [57]. To examine the effect of ambiguity between the foreground and background in the object motion estimation task, we also evaluate a version for which we provide the ground truth instance segmentation to both methods. Note that this is an upper bound to the performance achievable by ICP methods. We thus call these the *Oracle* ICP methods.

RGB-D Visual Odometry: We compare to the RGB-D visual odometry method [44] implemented in Open3D [57] on TUM RGBD SLAM datasets for visual odometry.

Direct Pose Regression: We compare three different variants that directly predict the mapping $f : \mathbf{I}, \mathbf{T} \to \boldsymbol{\xi}$. All three networks use Conv1-6 encoder layers from FlowNet-Simple [15] as two-view regression backbones. We use spatial average pooling after the last feature layer followed by a fully-connected layer to regress $\boldsymbol{\xi}$. All three CNN baselines are trained using the loss in (21).
- **PoseCNN**: A feed-forward CNN that directly predicts $\boldsymbol{\xi}$.
- **IC-PoseCNN**: A PoseCNN with iterative refinement using the IC algorithm, similar to [31] and [30]. We noticed that training becomes unstable and performance saturates

when increasing the number of iterations. For all our experiments, we thus used three iterations.
- **Cascaded-PoseCNN**: A cascaded network with three iterations, similar to IC-PoseCNN but with independent weights for each iteration.

Learning-based Optimization: We implemented the following related algorithms within our deep IC framework. For all methods, we use the same number of iterations, training loss and learning rate as used for our method.
- **DeepLK-6DoF**: We implemented a variant of DeepLK [50] which predicts the 3D transformation $\boldsymbol{\xi} \in \mathfrak{se}(3)$ instead of translation and scale prediction in their original 2D task. We use Gauss-Newton as the default optimization for this approach and no Convolutional M-Estimator. A comparison of this approach with our method when using only the two-view feature network (A) shows the benefits of our two-view feature encoder.
- **IC-FC-LS-Net**: We also implemented LS-Net [13] within our IC framework with the following differences to the original paper. First, we do not estimate or refine depth. Second, we do not use a separate network to provide an initial pose estimation. Third, we replace their LSTM layers with three fully connected (FC) layers which take the flattened $\mathbf{J}^T \mathbf{W} \mathbf{J}$ and $\mathbf{J}^T \mathbf{W} \mathbf{r}_k$ as input.

Ablation Study: We use (A), (B), (C) to refer to our contributions in Sec.1. We set $\mathbf{W}$ to the identity matrix when the Convolutional M-Estimator (B) is not used and use Gauss-Newton optimization in the absence of the Trust Region Network (C). We consider the following configurations:
- **Ours (A)+(B)+(C):** Our proposed method with shared weights. We perform coarse-to-fine iterations on four pyramid levels with three IC iterations at each level. We use shared weights for all iterations in (B) and (C).
- **Ours (A)+(B)+(C) (No WS):** A version of our method without shared weights. All settings are the same as above except that the network for (B) and (C) have independent weight parameters at each coarse-to-fine scale.
- **Ours (A)+(B)+(C) (K iterations/scale):** The same network as the default setting with shared weights, except that we change the inner iteration number K.
- **No Learning:** Vanilla coarse-to-fine IC alignment minimizing photometric error (4) without learned modules.

4.3. Results and Discussion

Table 1 and Table 2 summarize our main results. For each dataset, we evaluate the method separately for three different motion magnitudes {Small, Medium, Large}. In Table 1, {Small, Medium, Large} correspond to frames sampled from the original videos at intervals $\{1, 2, 4\}$. In Table 2, [Small, Medium, Large] correspond to frame intervals $\{2, 4, 8\}$ on BundleFusion and $\{1, 2, 5\}$ on DynamicBundleFusion. We show the following metrics/statistics:

---

[5]

	Model Descriptions	3D EPE (cm) ↓ on Validation/Test			Θ (Deg) ↓ / t (cm) ↓ / (t < 5 (cm) & Θ < 5°) ↑ on Test		
		Small	Medium	Large	Small	Medium	Large
ICP	Point-Plane ICP [11]	4.88/4.28	10.13/8.74	20.24/17.43	4.32/10.54/66.23%	8.29/20.90/33.4%	15.40/40.45/11.2%
	Point-Point ICP [4]	5.02/4.38	10.33/9.06	20.43/17.68	4.04/10.51/70.0%	7.89/20.15/33.8%	15.33/40.52/11.1%
	Oracle Point-Plane ICP [11]	3.91/3.31	10.68/9.63	22.53/19.98	**2.74**/9.75/**78.6%**	8.31/19.72/**40.3%**	16.64/41.40/**16.4%**
	Oracle Point-Point ICP [4]	4.34/3.99	11.83/10.29	21.27/26.13	3.81/10.11/75.0%	9.30/26.41/39.1%	19.53/62.30/14.1%
DPR	PoseCNN	5.18/4.60	10.43/9.20	20.08/17.74	3.91/10.51/69.8%	7.89/21.10/34.7%	15.34/40.54/11.0%
	IC-PoseCNN	5.14/4.56	10.40/9.13	19.80/17.31	3.93/10.49/70.1%	7.90/21.01/34.8%	15.31/40.51/11.1%
	Cascaded-PoseCNN	5.24/4.68	10.43/9.21	20.32/17.30	3.90/10.50/70.0%	7.90/21.11/34.6%	15.32/40.60/10.8%
Learning Optim	No learning	11.66/11.26	21.85/22.95	37.01/38.88	4.29/15.90/66.8%	8.33/31.80/32.2%	15.78/55.70/10.44%
	IC-FC-LS-Net, adapted from [13]	4.96/4.62	10.49/9.21	20.31/17.34	3.94/10.49/70.0%	7.90/21.20/34.5%	15.33/40.63/11.2%
	DeepLK-6DoF, adapted from [50]	4.41/3.75	9.05/7.54	18.46/15.33	4.03/10.25/68.1%	7.96/20.34/33.6%	15.43/39.56/10.6%
	Ours: (A)	4.35/3.66	8.80/7.23	18.28/15.06	4.09/10.19/68.6%	8.00/20.28/32.9%	15.37/39.49/10.9%
	Ours: (A)+(B)	4.33/3.26	8.84/7.30	18.14/15.04	4.02/10.11/69.2%	7.96/20.26/33.4%	15.35/39.56/10.9%
	Ours: (A)+(B)+(C)	**3.58**/2.91	**7.30**/5.94	**15.48**/12.96	3.74/**9.73**/74.5%	7.41/**19.60**/38.2%	14.71/**38.39**/12.9%
	Ours: (A)+(B)+(C) (No WS)	3.62/**2.89**	7.54/6.08	16.00/12.98	3.69/**9.73**/74.9%	**7.37**/19.74/38.4%	**14.65**/38.69/12.8%
	Ours: (A)+(B)+(C) ($K = 1$)	4.12/3.37	8.64/7.08	17.67/14.92	3.85/9.95/71.2%	7.80/20.13/35.5%	15.21/39.30/11.6%
	Ours: (A)+(B)+(C) ($K = 5$)	3.60/2.92	7.49/6.09	16.06/13.01	3.68/9.77/74.4%	7.46/19.69/37.8%	14.76/38.65/12.4%

Table 1: **Quantitative Evaluation on MovingObjects3D.** We evaluate the average 3D EPE, angular error in Θ (Euler angles), translation error **t** and success ratios **t** < 5 & Θ < 5° for three different motion magnitudes {Small, Medium, Large} which correspond to frames sampled from the original videos using frame intervals {1, 2, 4}.

	Model Descriptions	3D EPE (cm) ↓ Validation/Test on BundleFusion [14]			3D EPE (cm) ↓ Validation/Test on DynamicBundleFusion [33]			Model Size (K)	Inference Time (ms)
		Small	Medium	Large	Small	Medium	Large		
ICP	Point-Plane ICP [11]	2.81/2.01	6.85/4.52	16.20/11.11	1.26/0.97	2.64/2.09	10.38/4.89	-	310
	Point-Point ICP [4]	3.62/2.48	8.17/5.72	17.23/12.58	1.42/1.08	3.40/2.42	13.09/7.43	-	402
DPR	PoseCNN	4.76/3.41	9.78/6.85	17.90/13.31	2.20/1.65	4.22/3.19	13.90/8.24	19544	5.7
	IC-PoseCNN	4.32/3.26	8.98/6.52	16.30/12.81	2.21/1.66	4.24/3.18	12.99/8.05	19544	14.1
	Cascaded-PoseCNN	4.46/3.41	9.38/6.81	16.50/13.02	2.20/1.60	4.13/3.15	12.97/8.15	58632	14.1
Learning Optim	No learning	4.52/3.35	8.64/6.30	17.06/12.51	4.89/3.39	5.64/4.69	13.88/8.58	-	1.5
	IC-FC-LS-Net, adapted from [13]	4.04/3.03	9.06/6.85	17.15/13.32	2.33/1.80	4.43/3.45	13.84/8.35	674	1.7
	DeepLK-6DoF, adapted from [50]	4.09/2.99	8.14/5.84	16.81/12.27	2.15/1.72	3.78/3.12	12.73/7.22	596	2.2
	Ours: (A)	3.59/2.65	7.68/5.46	16.56/11.92	2.01/1.65	3.68/2.96	12.68/7.11	597	2.2
	Ours: (A)+(B)	2.42/1.75	5.39/3.47	12.59/8.40	2.22/1.70	3.61/2.97	12.57/6.88	622	2.4
	Ours: (A)+(B)+(C)	2.27/**1.48**	**5.11/3.09**	**12.16**/7.84	1.09/0.74	2.15/1.54	9.78/4.64	662	7.6
	Ours: (A)+(B)+(C) (No WS)	**2.14**/1.52	5.15/3.10	12.26/**7.81**	**0.93/0.61**	**1.86/1.32**	**8.88/3.82**	883	7.6

Table 2: **Quantitative Evaluation on BundleFusion and DynamicBundleFusion.** In BundleFusion [14], the motion magnitudes {Small, Medium, Large} correspond to frame intervals {2, 4, 8}. In DynamicBundleFusion [33], the motion magnitudes {Small, Medium, Large} correspond to frame intervals {1, 2, 5} (we reduce the intervals due to the increased difficulty).

	mRPE: θ (Deg) ↓ / t (cm) ↓			
	KF 1	KF 2	KF 4	KF 8
RGBD VO [44]	0.55/1.03	1.39/2.81	3.99/5.95	9.20/13.83
Ours: (A)	0.53/1.17	0.97/2.63	2.87/6.89	7.63/12.16
Ours: (A)+(B)	0.51/1.14	0.87/2.44	2.60/6.56	7.30/11.21
Ours: (A)+(B)+(C)	**0.45/0.69**	**0.63/1.14**	**1.10/2.09**	**3.76/5.88**

Table 3: **Results on TUM RGB-D Dataset [45].** This table shows the mean relative pose error (mRPE) on our test split of the TUM RGB-D Dataset [45]. KF denotes the size of the key frame intervals. Please refer to the supplementary materials for a detailed evaluation of individual trajectories.

- **3D End-Point-Error (3D EPE):** This metric is defined in (21). We only evaluate errors on the rigidly moving objects, i.e., the moving objects in MovingObjects3D and the rigid background mask in DynamicBundleFusion.
- **Object rotation and translation:** We evaluate 3D rotation using the norm of Euler angles Θ, translation **t** in cm and the success ratio (**t** < 5 (cm) & Θ < 5°), in Table 1.
- **Relative Pose Error**: We follow the TUM RGBD metric [45] using relative axis angle θ and translation **t** in cm.
- **Model Weight:** The number of learnable parameters.
- **Inference speed:** The forward execution time for an image-pair of size 160×120, using GTX 1080 Ti.

Comparison to baseline methods: Compared to all baseline methods (Table 1 row 1-10 and Table 2 row 1-8), our full model ((A)+(B)+(C)) achieves the overall best performance across different motion magnitudes and datasets while maintaining fast inference speed. Compared to ICP methods (ICP rows in Table 1 and Table 2) and classical method (No Learning in Table 1 and Table 2), our method achieves better performance without instance information at runtime. Compared to RGB-D visual odometry [44], our method works particularly well in unseen scenarios and

Figure 2: **Qualitative Results on MovingObjects3D.** Visualization of the warped image $\mathbf{I}(\boldsymbol{\xi})$ using the ground truth object motion $\boldsymbol{\xi}^{\text{GT}}$ (third row) and the object motion $\boldsymbol{\xi}^{\star}$ estimated using our method (last row) on the MovingObjects3D *validation* (left two) and *test* sets (right three). In $\mathbf{I}$, we plot the instance boundary in red and the instance boundary of $\mathbf{T}$ in green as comparison. Note the difficulty of the task (truncation, independent background object) and the high quality of our alignments. Black regions in the warped image are due to truncation or occlusion.

in the presence of large motions (Table 3). Besides, note that our model can achieve better performance with a significantly smaller number of weight parameters compared to direct image-to-pose regression (DRP rows in Table 1 and Table 2). Fig. 2 shows a qualitative comparison of our method on the MovingObject3D dataset by visualizing $\mathbf{I}(\boldsymbol{\xi})$. Note that our method achieves excellent two-view image alignments, despite large motion, heavy occlusion and varying illumination of the scene.

Ablation Discussion: Across all ablation variants and datasets, our model achieves the best performance by combining all three proposed modules ((A)+(B)+(C)). This demonstrates that all components are relevant for robust learning-based optimization. Note that the influence of the proposed modules may vary according to the properties of the specific task and dataset. For example, in the presence of noisy depth estimates from real data, learning a robust M-estimator (B) (Table 2 row 10 in BundleFusion) provide significant improvements. In the presence of heavy occlusions and motion ambiguities, learning the trust region step (C) helps to find better local optima which results in large improvements in our experiments, observed when estimating object motion (Table 1 row 13) and when estimating cam-

era motion in dynamic scenes (Table 2 row 11 in DynamicBundleFusion). We find our trust region network (C) to be highly effective for learning accurate relative motion in the presence of large variations in motion magnitude(Table 3).

5. Conclusion

We have taken a deeper look at the inverse compositional algorithm by rephrasing it as a neural network with three trainable submodules which allows for relaxing some of the most significant restrictions of the original formulation. Experiments on the challenging task of relative rigid motion estimation from two RGB-D frames demonstrate that our method achieves more accurate results compared to both classical IC algorithms as well as data hungry direct image-to-pose regression techniques. Although our data-driven IC method can better handle challenges in large object motions, heavy occlusions and varying illuminations, solving those challenges in the wild remains a subject for future research. To extend the current method to real-world environments with complex entangled motions, possible future directions include exploring multiple motion hypotheses, multi-view constraints and various motion models which will capture a broader family of computer vision tasks.

References

[1] P. Anandan. A computational framework and an algorithm for the measurement of visual motion. *International Journal of Computer Vision (IJCV)*, 2(3):283–310, 1989. 2

[2] S. Baker, R. Gross, I. Matthews, and T. Ishikawa. Lucas-kanade 20 years on: A unifying framework: Part 2. Technical report, Carnegie Mellon University, 2003. 1, 2, 3

[3] S. Baker and I. Matthews. Lucas-kanade 20 years on: A unifying framework: Part 1. Technical report, Carnegie Mellon University, Pittsburgh, PA, July 2002. 1, 2, 3

[4] P. J. Besl and N. D. McKay. A method for registration of 3-d shapes. *IEEE Trans. Pattern Anal. Machine Intell.*, 14(2):239–256, Feb. 1992. 6, 7

[5] A. Bibi, M. Mueller, and B. Ghanem. Target response adaptation for correlation filter tracking. In *Proc. of the European Conf. on Computer Vision (ECCV)*, 2016. 1

[6] A. Björck. *Numerical Methods for Least Squares Problems*. Siam Philadelphia, 1996. 2

[7] M. J. Black and P. Anandan. The robust estimation of multiple motions: Parametric and piecewise-smooth flow fields. *Computer Vision and Image Understanding (CVIU)*, 63(1):75–104, 1996. 2

[8] Brown, Matthew, Lowe, and David. Automatic panoramic image stitching using invariant features. *International Journal of Computer Vision (IJCV)*, 74(1):59–73, August 2007. 1

[9] A. Byravan and D. Fox. SE3-Nets: Learning rigid body motion using deep neural networks. In *Proc. IEEE International Conf. on Robotics and Automation (ICRA)*, pages 173–180. IEEE, 2017. 3

[10] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, J. Xiao, L. Yi, and F. Yu. ShapeNet: An Information-Rich 3D Model Repository. Technical Report arXiv:1512.03012 [cs.GR], Stanford University — Princeton University — Toyota Technological Institute at Chicago, 2015. 5

[11] Y. Chen and G. Medioni. Object modelling by registration of multiple range images. *Image and Vision Computing*, 10(3):145–155, 1992. 6, 7

[12] I. Cherabier, J. Schönberger, M. Oswald, M. Pollefeys, and A. Geiger. Learning priors for semantic 3d reconstruction. In *Proc. of the European Conf. on Computer Vision (ECCV)*, 2018. 3

[13] R. Clark, M. Bloesch, J. Czarnowski, S. Leutenegger, and A. J. Davison. Learning to solve nonlinear least squares for monocular stereo. In *European Conf. on Computer Vision (ECCV)*, Sept. 2018. 1, 3, 6, 7

[14] A. Dai, M. Nießner, M. Zollöfer, S. Izadi, and C. Theobalt. BundleFusion: real-time globally consistent 3D reconstruction using on-the-fly surface re-integration. *ACM Transactions on Graphics 2017 (TOG)*, 2017. 6, 7

[15] A. Dosovitskiy, P. Fischer, E. Ilg, P. Haeusser, C. Hazirbas, V. Golkov, P. v.d. Smagt, D. Cremers, and T. Brox. Flownet: Learning optical flow with convolutional networks. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*, 2015. 6

[16] J. Engel, V. Koltun, and D. Cremers. Direct sparse odometry. *IEEE Trans. Pattern Anal. Machine Intell.*, 40:611–625, 2018. 1, 2

[17] J. Engel, T. Schöps, and D. Cremers. LSD-SLAM: Large-scale direct monocular SLAM. In *European Conf. on Computer Vision (ECCV)*, September 2014. 1

[18] B. K. Horn and E. Weldon. Direct methods for recovering motion. *Intl. J. of Computer Vision*, 2(1):51–76, 1988. 2

[19] P. J. Huber. Robust estimation of a location parameter. *The Annals of Mathematical Statistics*, 35(1):73–101, 03 1964. 2

[20] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox. FlowNet 2.0: evolution of optical flow estimation with deep networks. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2017. 3

[21] M. Irani and P. Anandan. About direct methods. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV) Workshops*, 2000. 2

[22] M. Irani and S. Peleg. Improving resolution by image registration. *Graphical Model and Image Processing (CVGIP)*, 53(3):231–239, 1991. 1

[23] F. Jurie and M. Dhome. Hyperplane approximation for template matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(7):996–1000, 2002. 3

[24] A. Kanazawa, M. J. Black, D. W. Jacobs, and J. Malik. End-to-end recovery of human shape and pose. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2018. 3

[25] A. Kendall, M. Grimes, and R. Cipolla. Posenet: A convolutional network for real-time 6-dof camera relocalization. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*, 2015. 3

[26] A. Kendall, H. Martirosyan, S. Dasgupta, and P. Henry. End-to-end learning of geometry and context for deep stereo regression. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*, 2017. 3

[27] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *Proc. of the International Conf. on Learning Representations (ICLR)*, 2015. 5

[28] E. Kobler, T. Klatzer, K. Hammernik, and T. Pock. Variational networks: Connecting variational methods and deep learning. In *Proc. of the German Conference on Pattern Recognition (GCPR)*, 2017. 3

[29] A. Levin, A. Zomet, S. Peleg, and Y. Weiss. Seamless image stitching in the gradient domain. In *Proc. of the European Conf. on Computer Vision (ECCV)*, 2004. 1, 2

[30] Y. Li, G. Wang, X. Ji, Y. Xiang, and D. Fox. Deepim: Deep iterative matching for 6d pose estimation. In *Proc. of the European Conf. on Computer Vision (ECCV)*, 2018. 3, 5, 6

[31] C.-H. Lin and S. Lucey. Inverse compositional spatial transformer networks. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2017. 3, 6

[32] B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proc. of the International Joint Conf. on Artificial Intelligence (IJCAI)*, 1981. 1, 2

[33] Z. Lv, K. Kim, A. Troccoli, D. Sun, J. Rehg, and J. Kautz. Learning rigidity in dynamic scenes with a moving camera

for 3d motion field estimation. In *European Conf. on Computer Vision (ECCV)*, 2018. 6, 7

[34] F. Manhardt, W. Kehl, N. Navab, and F. Tombari. Deep model-based 6d pose refinement in rgb. In *Proc. of the European Conf. on Computer Vision (ECCV)*, September 2018. 3

[35] D. W. Marquardt. An algorithm for least-squares estimation of nonlinear parameters. *Journal of the Society for Industrial and Applied Mathematics (SIAM)*, 11(2):431–441, 1963. 2

[36] T. Meinhardt, M. Möller, C. Hazirbas, and D. Cremers. Learning proximal operators: Using denoising networks for regularizing inverse imaging problems. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*, 2017. 3

[37] R. A. Newcombe, S. Lovegrove, and A. J. Davison. DTAM: dense tracking and mapping in real-time. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*, 2011. 2

[38] R. Ranftl and V. Koltun. Deep fundamental matrix estimation. In *Proc. of the European Conf. on Computer Vision (ECCV)*, 2018. 3

[39] R. Ranftl and T. Pock. A deep variational model for image segmentation. In *Proc. of the German Conference on Pattern Recognition (GCPR)*, 2014. 3

[40] G. Riegler, M. Rüther, and H. Bischof. Atgv-net: Accurate depth super-resolution. In *Proc. of the European Conf. on Computer Vision (ECCV)*, 2016. 3

[41] D. Scharstein and R. Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International Journal of Computer Vision (IJCV)*, 47:7–42, 2002. 1

[42] J. Shi and C. Tomasi. Good features to track. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 1994. 1

[43] H.-Y. Shum and R. Szeliski. Systems and experiment paper: Construction of panoramic image mosaics with global and local alignment. *International Journal of Computer Vision (IJCV)*, 36(2):101–130, February 2000. 1

[44] F. Steinbrücker, J. Sturm, and D. Cremers. Real-time visual odometry from dense rgb-d images. In *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*, pages 719–722. IEEE, 2011. 6, 7

[45] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers. A benchmark for the evaluation of rgb-d slam systems. In *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, Oct. 2012. 6, 7

[46] M. Sundermeyer, Z.-C. Marton, M. Durner, M. Brucker, and R. Triebel. Implicit 3d orientation learning for 6d object detection from rgb images. In *Proc. of the European Conf. on Computer Vision (ECCV)*, September 2018. 3

[47] R. Szeliski. Image alignment and stitching: A tutorial. *Foundations and Trends in Computer Graphics and Vision*, 2(1):1–104, 2007. 2

[48] C. Tang and P. Tan. Ba-net: Dense bundle adjustment network. In *Intl. Conf. on Learning Representations (ICLR)*, 2019. 3

[49] B. Ummenhofer, H. Zhou, J. Uhrig, N. Mayer, E. Ilg, A. Dosovitskiy, and T. Brox. Demon: Depth and motion network for learning monocular stereo. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2017. 3

[50] C. Wang, H. K. Galoogahi, C.-H. Lin, and S. Lucey. Deep-lk for efficient adaptive object tracking. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2018. 1, 6, 7

[51] C. Wang, J. Miguel Buenaposada, R. Zhu, and S. Lucey. Learning depth from monocular videos using direct methods. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 3

[52] S.-E. Wei, V. Ramakrishna, T. Kanade, and Y. Sheikh. Convolutional pose machines. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 4724–4732, 2016. 3

[53] Z. Yin and J. Shi. GeoNet: Unsupervised Learning of Dense Depth, Optical Flow and Camera Pose. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2018. 3

[54] Z. Zhang, P. Robotique, and P. Robotvis. Parameter estimation techniques: a tutorial with application to conic fitting. *Image and Vision Computing (IVC)*, 15:59–76, 1997. 2

[55] S. Zheng, S. Jayasumana, B. Romera-Paredes, V. Vineet, Z. Su, D. Du, C. Huang, and P. H. S. Torr. Conditional random fields as recurrent neural networks. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*, 2015. 3

[56] H. Zhou, B. Ummenhofer, and T. Brox. Deeptam: Deep tracking and mapping. In *Proc. of the European Conf. on Computer Vision (ECCV)*, 2018. 3

[57] Q.-Y. Zhou, J. Park, and V. Koltun. Open3D: A modern library for 3D data processing. *arXiv:1801.09847*, 2018. 6

[58] T. Zhou, M. Brown, N. Snavely, and D. G. Lowe. Unsupervised learning of depth and ego-motion from video. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2017. 3