Deep Fundamental Matrix Estimation

René Ranftl and Vladlen Koltun

Intel Labs

Abstract. We present an approach to robust estimation of fundamental matrices from noisy data contaminated by outliers. The problem is cast as a series of weighted homogeneous least-squares problems, where robust weights are estimated using deep networks. The presented formulation acts directly on putative correspondences and thus fits into standard 3D vision pipelines that perform feature extraction, matching, and model fitting. The approach can be trained end-to-end and yields computationally efficient robust estimators. Our experiments indicate that the presented approach is able to train robust estimators that outperform classic approaches on real data by a significant margin.

1 Introduction

Deep learning has shown promising results on computer vision problems such as image categorization [20], image segmentation [24], and object detection [10]. Many problems that have been successfully tackled with deep learning share a common trait: The mapping from input to output is difficult to characterize by explicit mathematical modeling. This is especially true for the aforementioned applications, where even simple questions like what actually constitutes an object of a specific class cannot be answered in a simple way that lends itself to mathematical modeling [11]. Consequently, approaches such as deep learning, which are able to learn representations directly from large corpora of data are necessarily superior in these tasks.

On the other hand, certain computer vision problems, such as fundamental matrix estimation, can be defined in a precise mathematical way, provided that some assumptions are made about the data [12]. It is thus not surprising that these subfields have largely been spared by the recent surge in deep learning research.

However, being able to define a problem in a precise mathematical way doesn't necessarily mean that it can be easily solved. We argue that robust fundamental matrix estimation can be solved more accurately if the estimator can be adapted to the data at hand. For example, in an automotive scenario not all fundamental matrices are equally likely to occur. In fact, since the platform exhibits dominant forward or backward motion at all times, the space of fundamental matrices that can occur in this scenario is much smaller than the complete space of fundamental matrices. Another example is data that deviates from the common assumption of Gaussian inlier noise. Adapting model fitting approaches to different inlier noise distributions requires significant effort by an expert, but could be made much easier if the noise distribution can be learned from data.

In this work we present an approach that is able to learn a robust algorithm for fundamental matrix estimation from data. Our approach combines deep networks with a

well-defined algorithmic structure and can be trained end-to-end. In contrast to naive deep learning approaches to this problem, our approach disentangles local motion estimation and geometric model fitting, leading to simplified training problems and interpretable estimation pipelines. As such it can act as a drop-in replacement for applications where the RANSAC [7] family of algorithms is commonly employed [27, 35]. To achieve this, we formulate the robust estimation problem as a series of weighted homogeneous least-squares problems, where weights are estimated using deep networks.

Experiments on diverse real-world datasets indicate that the presented approach can significantly outperform RANSAC and its variants. Our experiments also show that estimators trained by the presented approach generalize across datasets. As a supporting result, we also show that the presented approach yields state-of-the-art accuracy in homography estimation.

2 Related Work

Robust fundamental matrix estimation, and more generally geometric model fitting, is a fundamental problem in computer vision that commonly arises in 3D processing tasks [12]. The common starting point is to first derive an estimator for outlier-free data. Various measures can then be taken to derive robust estimators that can deal with a certain amount of outliers.

Perhaps the most widely used approach for dealing with outliers is *RANdom SAmple Consensus* (RANSAC) [7], where one searches for a geometric model that has the most support in the form of inliers (defined based on some problem-specific point-to-model distance and a user-defined inlier threshold) using random sampling. There exists a vast amount of literature on variations of this basic idea [5, 30, 21, 39, 37, 36]. What most of these works have in common is the general structure of the algorithm. First, a set of points is sampled and a model is estimated using a non-robust baseline estimator. Second, the model is scored by evaluating a robust scoring function on all points and the model is accepted as the current best guess if its score is better than all previously scored models. This process is repeated until some stopping criterion is reached. A common weakness that is shared by sampling-based approaches is their dependence on the minimum number of data points required to unambiguously define a model. As the size of the minimal set increases, the probability of sampling at least one outlier rises exponentially. Note that RANSAC has been integrated into a deep scene coordinate regression pipeline [3] for camera localization. This approach uses finite differences to backpropagate through the non-robust base estimator and inherits the basic weaknesses of RANSAC.

Another line of work adopts the basic idea of consensus set maximization, but tackles optimization using globally optimal methods [22, 44]. Since the underlying optimization problem is NP-hard, these approaches are often prohibitively slow, degrading to exhaustive search in the worst case. While some progress has been made in speeding up globally optimal consensus set maximization [4], all known approaches are significantly slower than randomized algorithms and often lack the flexibility to tackle arbitrary geometric model fitting problems.

It is possible to directly robustify the base estimator using M-estimators [49, 8, 14, 50]. This line of work is most closely related to the presented approach, as it usually leads to a series of weighted least-squares problems. The major weakness of these approaches is that they require careful initialization and/or continuation procedures. Moreover, these approaches typically implicitly assume that the inliers are subject to Gaussian noise, which may not always be the case. In contrast, the presented approach doesn't make any assumptions on the inlier noise distributions, nor does it require extensive care to initialize the optimization, as both are learned from data.

There has been growing interest in applying deep learning to 3D processing tasks. DeTone et al. learned a neural network to directly regress from a pair of input images to a homography [6]. This work was later extended with an image-based loss to allow unsupervised training [28]. Agrawal et al. [1] estimate ego-motion using a neural network as a pre-training step for high-level tasks. PoseNet [17, 16] employs a convolutional network to estimate the pose of a given image for camera relocalization. The DeMoN architecture [41] provides, given two consecutive frames from a monocular camera, both an estimate of the depth of each pixel and an estimate of the motion between frames. A common characteristic of all these models is that they do not enforce the intrinsic structure of the problem, beyond their parametrization and training loss. As a consequence, a large amount of training data is needed and generalization performance is often a concern. A notable exception is the approach of Rocco et al. [31], which is modeled after the classical stages of feature extraction, matching, and model estimation. Note, however, that the model estimator again is a deep regressor that doesn't incorporate any geometric constraints.

In contrast to these works, our approach directly operates on putative matches, independently of how these matches were obtained. Keypoint detection and matching remain an independent step. As a consequence, our approach can be used as a drop-in replacement in pipelines where RANSAC and similar algorithms are currently employed. We argue that such a modular approach to tackling 3D processing using deep learning is highly desirable, given the lack of large-scale datasets in this domain. It is much easier to learn different subparts of 3D reconstruction systems, such as feature matching [45, 34] and model estimation separately, as generating realistic training data for these subproblems becomes easier. Moreover, a modular approach leads to disentangled intermediate representations, which significantly enhances the interpretability of a pipeline.

Machine learning techniques have been applied to robustify and speed up optimization problems. Andrychowicz et al. [2] use neural networks to find update directions for gradient-descent algorithms. A framework to learn fixed-point iterations for point cloud registration is presented in [42]. These approaches are not directly applicable to fundamental matrix estimation, since gradient descent cannot be trivially applied.

3 Preliminaries

We refer to a single element of the input data of dimensionality d as a point $\mathbf{p}_i \in \mathbb{R}^d$. Let $\mathbf{P} \in \mathcal{P} = \mathbb{R}^{N \times d}$ be a collection of points of dimensionality d that contains N (not necessarily distinct) points. We use $(\mathbf{P})_i$ to refer to the i-th row of matrix $\mathbf{P}$. Note that

points can be either points in some metric space, or in the case of fundamental matrix and homography estimation point correspondences (e.g., we have $\mathbf{p}_i \in \mathbb{R}^4$ in this case by concatenating the two image coordinates of putative correspondences $\tilde{\mathbf{p}}_i \leftrightarrow \tilde{\mathbf{p}}'_i$).

In many geometric model fitting problems a homogeneous least-squares optimization problem arises:

$$\begin{aligned} \underset{\mathbf{x}}{\text{minimize}} \quad & \sum_{i=1}^{N} \left\| (\mathbf{A}(\mathbf{P}))_i \cdot \mathbf{x} \right\|^2 \\ \text{subject to} \quad & \|\mathbf{x}\| = 1, \end{aligned} \tag{1}$$

where $\mathbf{x} \in \mathbb{R}^{d'}$ defines the model parameters and $\mathbf{A} : \mathcal{P} \rightarrow \mathbb{R}^{kN \times d'}$ ($kN \geq d',\ k > 0$) is a problem-specific mapping of the data points.

Note that (1) admits a closed-form solution. Popular examples of algorithms where optimization problems of this form arise are the eight-point algorithm for fundamental matrix estimation [13], the Direct Linear Transform (DLT) [12], and general total least-squares fitting.

Consider hyperplane fitting as a simple example. Let $(\mathbf{n}^\top, c)^\top$ specify a hyperplane with normal $\mathbf{n}$ and intercept c. The goal of hyperplane fitting is to infer $(\mathbf{n}^\top, c)^\top$ from a set of points $\mathbf{P}$. To fit a hyperplane in a total least-squares sense, we have

$$\mathbf{A}(\mathbf{P}) \in \mathbb{R}^{N \times d}, \qquad (\mathbf{A}(\mathbf{P}))_i = \mathbf{p}_i^\top - \frac{1}{N} \sum_{j=1}^{N} \mathbf{p}_j^\top . \tag{2}$$

Solving (1) with this definition allows us to extract the plane using the model extraction function $g(\mathbf{x})$ that maps $\mathbf{x}$ to the model parameters:

$$g(\mathbf{x}) = \left(\mathbf{x}^\top, -\mathbf{x} \cdot \frac{1}{N} \sum_{i=1}^{N} \mathbf{p}_i \right)^\top = (\mathbf{n}^\top, c)^\top . \tag{3}$$

If the data is free of outliers, the least-squares solution will be close to the true solution (depending on the inlier noise distribution and the specific form of the problem). However, in practical applications the data does usually contain outliers. (Even worse, there may be more outliers than inliers.) Solving the estimation problem in a least-squares sense will yield wrong estimates even in the presence of a single outlier.

Much work has gone into finding robust approaches to geometric model fitting [7, 39, 30, 14]. One possible solution is to apply a robust loss function Φ to the residuals in (1). The resulting optimization problem does not admit a closed-form solution in general. A practical way to approximately solve the optimization problem is by solving a sequence of reweighted least-squares problems [38]:

$$\mathbf{x}^{j+1} = \underset{\mathbf{x}:\ \|\mathbf{x}\|=1}{\arg\min} \sum_{i=1}^{N} w(\mathbf{p}_i, \mathbf{x}^j) \left\| (\mathbf{A}(\mathbf{P}))_i \cdot \mathbf{x} \right\|^2 , \tag{4}$$

where the exact form of the weights w depends on Φ and the geometric model at hand.

Coming back to the hyperplane fitting example, assume that $w(\mathbf{p}_i, \mathbf{x}^j) = w_i = 1$ if $\mathbf{p}_i$ is an inlier and $w(\mathbf{p}_i, \mathbf{x}^j) = w_i = 0$ otherwise. It is clear that given these weights, the correct model can be recovered in a single iteration of (4) by setting

$$(\mathbf{A}(\mathbf{P}))_i = \mathbf{p}_i^\top - \frac{\sum_{j=1}^N w_j \mathbf{p}_j^\top}{\sum_{j=1}^N w_j}, \qquad g(\mathbf{x}) = \left(\mathbf{x}^\top, -\mathbf{x} \cdot \frac{\sum_{j=1}^N w_j \mathbf{p}_j}{\sum_{j=1}^N w_j} \right)^\top . \tag{5}$$

Knowing the weights in advance is a chicken-and-egg problem. On the one hand, if we knew the true model we could trivially separate inliers from outliers. On the other hand, if we knew which points are inliers we could directly recover the correct model. In what follows, we will show that in many instances the weights can be estimated reasonably well using a deep network with appropriate structure.

4 Deep Model Fitting

Our approach is inspired by the structure of (4). It can be thought of as an iteratively reweighted least-squares algorithm (IRLS) with a complex, learned reweighting function. Since we are learning weights from data, we expect that our algorithm is able to outperform general purpose approaches whenever one or more of the following assumptions are true. (1) The input data admits regularity in the inlier and outlier distributions that can be learned. An example would be an outlier distribution that is approximately uniform and sufficiently dissimilar to the inlier noise distribution. This is a mild assumption that in fact has been exploited in sampling-based approaches previously [39]. (2) The problem has useful side information that can be integrated into the reweighting function. An example would be matching scores or keypoint geometry. (3) The output space is a subset of the full space of model parameters. An example would be fundamental matrix estimation for a camera mounted on a car or a wheeled robot.

We will show in our experimental evaluation that our approach indeed is able to outperform generic baselines if regularity is present in the data, while being competitive when there is no apparent regularity in the data.

In the following we adopt the general structure of algorithm (4), but do not assume a simple form of the weight function w. Instead we parametrize it using a deep network and learn the network weights from data such that the overall algorithm leads to accurate estimates. Our approach can be understood as a meta-algorithm that learns a complex and problem-dependent version of the IRLS algorithm with an unknown cost function. We show that this approach can be used to easily integrate side information into the problem, which can enhance and robustify the estimates.

Model estimator. We first describe the fundamental building block of our approach, a version of (4) where the weights are parametrized by a deep network, and will discuss how the network can be trained end-to-end. We start by redefining the weight function as $w : \mathcal{P} \times \mathcal{S} \times \mathbb{R}^{d'} \to (\mathbb{R}_{>0})^N$, where $\mathbf{S} \in \mathcal{S} = \mathbb{R}^{N \times s}$ collects side information that may be available for each point. Note that this function is defined globally, thus individual points can influence each other. Since w can be a non-trivial function, we

parametrize it by a deep network with weights $\boldsymbol{\theta}$. With this parametrization, a single step in algorithm (4) becomes

$$\mathbf{x}^{j+1} = \underset{\mathbf{x}:\ \|\mathbf{x}\|=1}{\arg\min} \sum_{i=1}^{N} (w(\mathbf{P}, \mathbf{S}, \mathbf{x}^j; \boldsymbol{\theta}))_i \left\| (\mathbf{A}(\mathbf{P}))_i \cdot \mathbf{x} \right\|^2 . \tag{6}$$

The question is how to find a parametrization $\boldsymbol{\theta}$ that leads to robust and accurate estimates. We now drop the explicit dependence on the correspondences and side information for notational brevity and move to matrix form:

$$\mathbf{x}^{j+1} = \underset{\mathbf{x}:\ \|\mathbf{x}\|=1}{\arg\min} \left\| \mathbf{W}^j(\boldsymbol{\theta}) \mathbf{A} \mathbf{x} \right\|^2 , \tag{7}$$

where $(\mathbf{W}^j(\boldsymbol{\theta}))_{i,i} = \sqrt{w_i^j}$ collects the individual weights into a diagonal matrix.

Proposition 1 *Let $\mathbf{X} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^\top$ denote the singular value decomposition (SVD) of a matrix $\mathbf{X}$. The solution $\mathbf{x}^{j+1}$ of* (7) *is given by the right singular vector $\mathbf{v}_{d'}$ corresponding to the smallest singular value of the matrix $\mathbf{W}(\boldsymbol{\theta})\mathbf{A}$.*

This is a well-known fact as (7) is a homogeneous least-squares problem. For completeness, a derivation is given in supplementary material.

Proposition 1 implies that a solution to the model fitting problem is recovered as $g(f(\mathbf{W}(\boldsymbol{\theta})\mathbf{A}))$, where $f(\mathbf{X}) = \mathbf{v}_{d'}$ and $g(\mathbf{x})$ is an appropriate function that maps from the SVD to the parameters of the geometric model. An example was already given in (5) for the case of hyperplane fitting. We will provide an example for fundamental matrix estimation in Section 5.

In order to learn the weights $\boldsymbol{\theta}$ using gradient-based optimization, we need to be able to backpropagate the gradient through an SVD layer. Ionescu et al. [15] showed how this can be achieved using matrix calculus:

Proposition 2 *Assume that the singular vectors are ordered according to the magnitude of their singular values $\mathbf{V} = (\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_{d'})$ such that $\mathbf{v}_{d'}$ is the singular vector corresponding to the smallest singular value. We need to backpropagate through $g(f(\mathbf{X}))$ with $f(\mathbf{X}) = \mathbf{v}_{d'}$. The gradient of g with respect to the input $\mathbf{X}$ is given by*

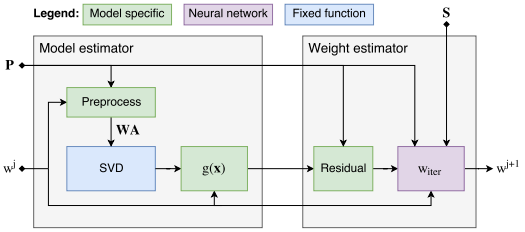
$$\frac{\partial g}{\partial \mathbf{X}} = \mathbf{U}\Big\{ 2\boldsymbol{\Sigma}\Big(\mathbf{K}^\top \circ \Big(\mathbf{V}^\top \frac{\partial g}{\partial \mathbf{v}_{d'}} \Big)_{sym} \Big) \Big\} \mathbf{V}^\top , \tag{8}$$

where

$$\mathbf{K}_{ij} = \begin{cases} \frac{1}{\sigma_i^2 - \sigma_j^2}, & \text{if } i \neq j \\ 0, & \text{otherwise} \end{cases} \tag{9}$$

and σ_i denotes the i-th singular value.

The structure of the gradient (8) follows as a special case of the derivations in [15].



Layer	# in	# out	L-ReLU+IN
1	–	64	✓
2	64	128	✓
3	128	1024	✓
4	1024	512	✓
5	512	256	✓
6	256	1	✗

Figure 1 & Table 1. Estimation module and network architecture. Left: The estimation module is composed of two parts. Given input points and a weighting, a model is estimated using weighted least-squares. In the second stage a new set of weights is generated given the points, their residuals with respect to the previously estimated model, and possibly side information. Right: The network architecture of w_{init} and w_{iter}. A checkmark in column L-ReLU+IN indicates that a leaky ReLU followed by instance normalization is applied to the output of the layer.

A schematic overview of the model estimator is shown in Figure 1. The block takes as input the points $\mathbf{P}$ and a set of weights w. It constructs the matrix $\mathbf{W}(\boldsymbol{\theta})\mathbf{A}$ as a preprocessing step, applies a singular value decomposition, and then performs the model extraction step $g(\mathbf{x})$ that yields an estimate of the geometric model given the input weights. The estimated model can then be used to estimate a new set of weights, based on the input points, side information, and the residuals of the currently estimated model.

Weight estimator. To accurately estimate weights, the estimator needs to fulfill two requirements: It has to be equivariant to permutation of the input data and it has to be able to process an arbitrary number N of input points. The first requirement arises since the data presented to the weight estimator does not exhibit any natural ordering. Thus the function approximator needs to integrate global information in a way that is independent of the actual ordering of the input data points. The second requirement arises from the fact that in most applications we do not know the number of input points a priori.

To build a deep network that fulfills both requirements, we adopt the idea presented in [29] and [48] for processing unordered sets of points using deep networks. The key idea of these works is simple: In order to make a network equivariant to permutation of the input data, every operation in the network itself has to be equivariant to permutation. This is especially relevant for layers that operate across multiple data points. It can be shown that global average and max-pooling along dimension N fulfill this property. We adopt the general structure of [29] with a small modification: Instead of a single pooling layer that integrates global information, we perform instance normalization [40] after each layer:

$$(I(\mathbf{h}))_i = \frac{\mathbf{h}_i - \mu(\mathbf{h})}{\sqrt{\sigma^2(\mathbf{h}) + \epsilon}}, \tag{10}$$

where $\mathbf{h}_i$ is the feature vector corresponding to point i and the mean $\mu(\mathbf{h})$ as well as the variance $\sigma^2(\mathbf{h})$ are computed along dimension N. This operation integrates the distribution of global information across points and normalization into a single step. Since

Algorithm 1 Forward pass

Construct $\mathbf{A}(\mathbf{P})$
$w^0 \leftarrow \mathrm{softmax}(w_{init}(\mathbf{P}, \mathbf{S}))$ ▷ Initial weights
for $j = 0$ **to** D **do**
    $\mathbf{X} \leftarrow \mathrm{diag}(\mathbf{w}^j)\mathbf{A}$
    $\mathbf{U}, \mathbf{\Sigma}, \mathbf{V} \leftarrow \mathrm{svd}(\mathbf{X})$
    Extract $\mathbf{x}^{j+1} = \mathbf{v}^{d'}$ from $\mathbf{V}$ ▷ Solution to (7)
    Compute residuals to construct $\mathbf{r}$ from $g(\mathbf{x}^{j+1})$
    $\mathbf{w}^{j+1} \leftarrow \mathrm{softmax}(w_{iter}(\mathbf{P}, \mathbf{S}, \mathbf{r}, \mathbf{w}^j))$
end for
return $g(\mathbf{x}^D)$

instance normalization is entirely composed of permutation equivariant operations, the overall network is equivariant to permutations of the input points. We found that this modification improves stability during training, especially in the high noise regime. A similar observation was made independently in concurrent work on essential matrix estimation [46]. We conjecture that for data with low signal-to-noise ratio, it is crucial to have multiple operations in the network that integrate data globally. This is in contrast to the original PointNet architecture [29], where a single global integration step in the form of a pooling layer is proposed.

An overview of the architecture is shown in Table 1. It consists of repeated application of a linear layer (acting independently for each point), followed by a leaky ReLU activation function [26] and the instance normalization module that enables global communication between points. In order to produce strictly positive weights, the error estimator is followed by a softmax. We experimented with different output activations and found that the softmax activation leads to initializations that are close to the least-squares estimate.

We define two networks: $w_{init}(\mathbf{P}, \mathbf{S})$ to compute an initial set of weights and a network $w_{iter}(\mathbf{P}, \mathbf{S}, \mathbf{r}, \mathbf{w}^j)$ to update the weights after a model estimation step, where $\mathbf{r}$ denotes the geometric residuals of the current estimate, $(\mathbf{r})_i = r(\mathbf{p}_i, g(\mathbf{x}^j))$.

Architecture. The complete architecture consists of an input weight estimator w_{init}, repeated application of the estimation module, and a geometric model estimator on the final weights. In practice we found that five consecutive estimation modules strike a good balance between accuracy and speed. An overview of a complete forward pass is shown in Algorithm 1.

We implement the network in PyTorch. In all applications that follow we use Adamax [18] with an initial learning rate of 10^{-3} and a batch size of 16. We reduce the learning rate every 10 epochs by a factor of 0.8 and train for a total of 100 epochs.

5 Fundamental Matrix Estimation

For a complete forward pass, the following problem-dependant components need to be specified: The preprocessing step $\mathbf{A}(\mathbf{P})$, the model extractor $g(\mathbf{x})$, and the residual

$r(\mathbf{p}_i, \mathbf{x})$. Note that all of these quantities also need to be specified for RANSAC-type algorithms, since they specialize the general meta-algorithm to the specific problem instance. While our exposition focuses on fundamental matrix estimation, our approach can handle other types of problems that are based on homogeneous least-squares. We use homography estimation as an additional example. Note that other types of estimators which are not based on homogeneous least-squares could be integrated as long as they are differentiable. In addition we need to specify a training loss to be able to train the pipeline. We will show that the loss can be directly derived from the residual function r.

We perform robust fundamental matrix estimation based on the normalized 8-point algorithm [13]. We rescale all coordinates to the interval $[-1, 1]^2$ and define the preprocessing function

$$(\mathbf{A}(\mathbf{P}))_i = \text{vec}\left(\mathbf{T}\hat{\mathbf{p}}_i(\mathbf{T}'\hat{\mathbf{p}}'_i)^\top\right), \tag{11}$$

where $\hat{\mathbf{p}}_i = ((\mathbf{p}_i)_1, (\mathbf{p}_i)_2, 1)^\top$ and $\hat{\mathbf{p}}'_i = ((\mathbf{p}_i)_3, (\mathbf{p}_i)_4, 1)^\top$ are homogenous coordinates of the correspondences in the left and right image respectively, and $\mathbf{T}$, $\mathbf{T}'$ are normalization matrices that robustly center and scale the data [13] based on the estimated weights. We further define the model extractor as

$$g(\mathbf{x}) = \underset{\mathbf{F}:\ \det(\mathbf{F})=0}{\arg\min} \left\|\mathbf{F} - \mathbf{T}^\top(\mathbf{x})_{3\times 3}\mathbf{T}'\right\|_F, \tag{12}$$

where $\mathbf{F}$ denotes the fundamental matrix. The model extractor explicitly enforces rank deficiency of the solution by projecting to the set of rank-deficient matrices. It is well-known that this projection can be carried out in closed formed by setting the smallest singular value of the full-rank solution to zero [13]. We use the symmetric epipolar distance as the residual function:

$$r(\mathbf{p}_i, \mathbf{F}) = \left|\hat{\mathbf{p}}_i^\top \mathbf{F}\hat{\mathbf{p}}'_i\right| \left(\frac{1}{\|\mathbf{F}^\top\hat{\mathbf{p}}_i\|_2} + \frac{1}{\left\|\mathbf{F}\hat{\mathbf{p}}'_i\right\|_2}\right). \tag{13}$$

Fundamental matrices cannot be easily compared directly due to their structure. We opt to compare them based on how they act on a given set of correspondences. To this end we generate virtual pairs of correspondences that are inliers to the groundtruth epipolar geometry by generating a grid of points in both images and reprojecting the points to the groundtruth epipolar lines. This results in virtual, noise-free inlier correspondences $\mathbf{p}_i^{gt}$ that can be used to define a geometrically meaningful loss. This can be understood as sampling the groundtruth epipolar geometry in image space. We define the training loss as

$$\mathcal{L} = \frac{1}{N_{gt}} \sum_{j=0}^{D} \sum_{i=1}^{N_{gt}} \min\left(r(\mathbf{p}_i^{gt}, g(\mathbf{x}^j)), \gamma\right). \tag{14}$$

Clamping the residuals ensures that hard problem instances in the training set do not dominate the training loss. We set $\gamma = 0.5$ for all experiments. Note that since the network admits interpretable intermediate representations, we attach a loss to all D intermediate results.

To faciliate efficient batch training, we constrain the number of keypoints per image pair to 1000, by randomly sampling a set of keypoints if the detected number is larger. We replicate random keypoints if the number of detected keypoints is smaller than 1000.

At test time we evaluate the estimated solution and perform a final, non-robust model fitting step to the 20 points with smallest residual error in order to correct for small inaccuracies in the estimated weights.

6 Experiments

In order to show that our approach is able to exploit regularity in data when it is present, while providing competitive performance on generic fundamental matrix estimation problems, we conduct experiments on datasets of varying regularity: (1) The Tanks and Temples dataset, which depicts images of medium-scale scenes taken from a hand-held camera [19]. This dataset presents a large-baseline scenario with generic camera extrinsics, but exhibits some regularity (particularly in the intrinsics) as all sequences in the dataset are acquired by two cameras. (2) The KITTI odometry dataset, which consists of consecutive frames in a driving scenario [9]. This datasets exhibits high regularity, with small baselines and epipolar geometries that are dominated by forward motion. (3) An unstructured SfM dataset, with images taken from community photo collections [43]. This dataset represents the most general case of fundamental matrix estimation, where image pairs are taken from arbitrary cameras over large baselines. We will show that our approach is still able to learn a robust estimator that performs as well as or better than classic sampling-based approaches in the most general case, while offering the possibility to specialize if regularity is present in the data.

Tanks and Temples. The Tanks and Temples dataset consists of medium-scale image sequences taken from a hand-held camera [19]. We use the sequences *Family*, *Francis*, *Horse*, and *Lighthouse* for training. We use *M60* for validation and evaluate on the three remaining 'Intermediate' sequences: *Panther*, *Playground*, and *Train*. (The train/val/test split was done in alphabetical order, by sequence name.) We reconstruct the sequences using the COLMAP SfM pipeline [35] to derive groundtruth camera poses and corresponding fundamental matrices. We use SIFT [25] to extract putative correspondences between all pairs of frames in a sequence and discard pairs which have less than 20

Table 2. Performance on the Tanks and Temples dataset for different numbers of iterations D. $D = 5^*$ does not use any side information (the only inputs to the network are the x-y coordinates of the putative matches). *Direct reg.* is a network that directly regresses to the fundamental matrix.

	% Inliers	F-score	Mean	Median	Min	Max	Time [ms]
$D = 1$	42.30	44.80	3.45	1.00	0.08	1912.67	7
$D = 3$	44.91	47.25	1.98	0.82	0.08	566.70	18
$D = 5$	45.02	46.99	2.04	0.83	0.11	285.36	26
$D = 5^*$	44.60	46.42	2.23	0.84	0.10	391.64	26
Direct reg.	4.42	9.14	16.67	11.96	0.83	386.15	3

matches within one pixel of the groundtruth epipolar lines. The resulting dataset is composed of challenging wide-baseline pairs. An example is shown in Figure 3 (rightmost column). Note that the SfM pipeline reasons globally about the consistency of 3D points and cameras, leading to accurate estimates with an average reprojection error below one pixel [35]. We generate two datasets: A default dataset were the correspondences were prefiltered using a ratio test with a cut-off of 0.8. Unless otherwise stated we train and test on this filtered dataset. The ratio test is a commonly employed technique in SIFT matching and can lead to greatly improved inlier ratios, but might lead to a sparse set of candidate correspondences. We generate a second significantly harder dataset without this pre-filtering step to test the robustness of our approach in the high noise regime.

In a first experiment, we train our network with varying depths D and use the descriptor matching score as well as the ratio of best to second best match as side information. We report the average percentage of inliers (correspondences with epipolar distance below one pixel), the F1-score (where positives are defined as correspondences with an epipolar distance below one pixel with respect to the groundtruth epipolar line), and the mean and median epipolar distance to groundtruth matches. We additionally report the minimum and maximum errors incurred over the dataset. The results are summarized in Table 2. It can be observed that with a larger number of iterations, more accurate results are found. The setting $D = 1$ corresponds to only applying the neural network followed by a single weighted least-squares estimate. Using three steps of iterative refinement ($D = 3$) considerably improves the average inlier count as well as the overall accuracy. The network with five steps of iterative refinement ($D = 5$) performs comparably to three steps in most measures, but is more robust in the worst case. We thus use this architecture for all further evaluations.

We additionally evaluate the influence of side information. This is shown in the $D = 5^*$ setting, where only the locations of the putative correspondences are passed to the neural networks. Removing the side information leads to a small but noticeable drop in average accuracy. Finally, *Direct reg.* shows the result of an unstructured neural network that directly regresses from correspondences and side information to the coefficients of the fundamental matrix. The architecture resembles the $D = 1$ setting, with the weighted least-squares layer replaced by a pooling layer followed by three fully-connected layers. Details of the architecture can be found in the supplementary material. It can be seen that the unstructured network leads to considerably worse results. This highlights the importance of modeling the problem structure. We additionally report average execution times in milliseconds for the different architectures as measured on an NVIDIA Titan X GPU.

Table 3 compares our approach ($D = 5$) to RANSAC [7], Least Median of Squares (LMEDS) [32], MLESAC [39], and USAC [30]. Note that USAC is a state-of-the-art robust estimation pipeline. Similarly to our approach, it can leverage matching quality as additional side information to perform guided sampling. For fair comparison, we thus provide the matching scores (side information) to USAC. For RANSAC, LMEDS, and MLESAC we used the eight-point algorithm [13] as the base estimator, whereas USAC used the seven-point algorithm. For the baseline methods we performed a grid-search over hyperparameters on the training set.

Table 3. Results on the Tanks and Temples dataset. We evaluate two scenarios: moderate noise, where the putative correspondences were prefiltered using the ratio test, and high noise, without the ratio test. Our approach outperforms the baselines in both scenarios.

	Tanks and Temples – with ratio test				*Tanks and Temples – without ratio test*			
	% Inliers	F-score	Mean	Median	% Inliers	F-score	Mean	Median
RANSAC	42.61	42.99	**1.83**	1.09	2.98	10.99	122.14	79.28
LMEDS	42.96	40.57	2.41	1.14	1.57	4.78	120.63	108.72
MLESAC	41.89	42.39	2.04	1.08	2.13	8.28	131.11	93.04
USAC	42.76	43.55	3.72	1.24	4.45	23.55	46.32	8.52
Ours	**45.02**	**46.99**	2.04	**0.83**	**5.62**	**26.92**	**36.81**	**7.82**

As shown in Table 3, our approach outperforms all the baselines on both datasets. The difference is particularly striking on the dataset that does not include the ratio test. This datasets features very high outlier ratios (80%+), pushing the sampling-based approaches beyond their breakdown point in most cases. USAC and our approach perform considerably better than the other baselines on this dataset, which highlights the importance of using side information to guide the estimates.

KITTI odometry dataset. The KITTI odometry dataset [9] consists of 22 distinct driving sequences, eleven of which have publicly available groundtruth odometry. We follow the same protocol as on the previous dataset and use the ratio test to pre-filter the putative correspondences. We train our network on sequences *00* to *05* and use sequences *06* to *10* for testing.

Table 4 summarizes the results on this dataset. We show the results of two different models: One that was trained on the KITTI training set (*Ours tr. on KITTI*) and one that was trained on Tanks and Temples (*Ours tr. on T&T*). The results indicate that our approach is able to learn a model that is more accurate when specialized to the dataset. It is interesting to note that the model that was trained on the Tanks and Temples dataset generalizes to the KITTI dataset, even though the datasets are very different in terms of

Table 4. Results on the KITTI benchmark for different inlier thresholds. We evaluate a model that was trained on the KITTI training set as well as a model that was trained on Tanks and Temples, in order to show both ability to take advantage of regularities in the data and ability to learn an estimator that generalizes across datasets.

	@ 0.1px		*@ 1px*			
	% Inliers	F-score	% Inliers	F-score	Mean	Median
RANSAC	21.85	13.84	84.96	75.65	0.35	0.32
LMEDS	20.01	13.34	84.23	75.44	0.37	0.35
MLESAC	18.60	12.54	84.48	75.15	0.39	0.36
USAC	21.43	13.90	85.13	75.70	0.35	0.32
Ours tr. on T&T	21.00	13.31	84.81	75.08	0.39	0.33
Ours tr. on KITTI	**24.61**	**14.65**	**85.87**	**75.77**	**0.32**	**0.29**

the camera geometry and the range of fundamental matrices that can occur. The model that was trained on Tanks and Temples performs comparably to the baseline approaches, which indicates that a robust estimator that is applicable across datasets can be learned.

Community photo collections. We additionally show experiments on general community photo collection datasets [43]. This dataset admits no obvious regularity, with images taken from distinct cameras from a large variety of positions.

We use the sequence *Gendarmenmarkt* from this dataset for training, and use the sequence *Roman Forum* for testing. We reconstruct both sequences using COLMAP [35]. We randomly sample 10,000 image pairs from *Gendarmenmarkt* that contain at least 20 matches that are within 1 pixel of the groundtruth epipolar line to generate the training set. We randomly sample 1000 image pairs from *Roman Forum* to generate the test set. Table 5 summarizes the results on this dataset. It can be seen that our approach performs as well as or better than the baselines according to most measures. Since there is no apparent regularity in the data, this highlights the ability of our approach to learn a general-purpose estimator, while still being able to exploit regularity to its advantage if it is present in the data. We present qualitative results on this dataset in Figure 3.

Homography estimation. As a supporting contribution, we show that our approach leads to state-of-the-art results on the task of homography estimation. We use the DLT as a base estimator and provide exact expressions for $\mathbf{A}(\mathbf{P})$, $g(\mathbf{x})$, and the residuals in the supplementary material.

We follow the evaluation protocol defined in [6], which is based on the MS-COCO dataset [23]. For each image we extract random patches of size 256×256 pixels. We generate a groundtruth homography by perturbing the corners of the reference patch uniformly at random by up to 64 pixels. The inverse homography is used to warp the reference image and a second (warped) patch is extracted from the same location. We refer to [6] for further details on the dataset generation. We generate 10,000 training pairs and use ORB [33] and the Hamming distance for matching. We discard pairs with less than 100 matches and use a total of 500 points to allow batching. We use matching scores as side information and clamp the maximal residual in (14) to $\gamma = 64$.

We compare our approach to a deep network for homography estimation [6] (HNet), RANSAC followed by a non-linear refinement stage, and USAC. To train the baseline

	@ 0.1px		*@ 1px*			
	% Inliers	F-score	% Inliers	F-score	Mean	Median
RANSAC	49.55	40.80	67.52	59.12	2.29	1.21
LMEDS	**51.74**	41.87	67.85	59.38	2.50	1.16
MLESAC	48.07	40.01	67.40	58.64	**1.45**	1.17
USAC	51.21	41.87	66.65	58.93	2.94	1.22
Ours	51.41	**43.28**	**68.31**	**60.67**	1.51	**1.02**

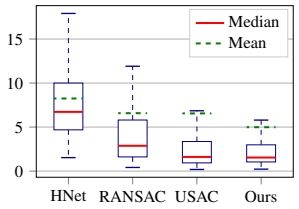


Table 5 & Figure 2. Left: Performance of fundamental matrix estimation on *Roman Forum*. This datasets exhibits a wide range of camera geometries and motions. Our approach leads to an estimator that is competitive with the baselines. Right: Performance on the homography estimation task in terms of average corner error.

network, we follow the exact protocol described in [6]. The test set consists of 1000 images from the MS-COCO test set that where generated in the same way as the training set with the exception that we do not discard pairs with less than 100 matches.

The results are summarized in Figure 2. We report statistics of the average corner error of the estimated homographies. Note that our result of HNet is slightly better than what was reported by the authors (avg. error 8.0 pixels vs 9.2 pixels in [6]). Our approach outperforms both HNet and the SAC baselines.

7 Conclusion

We have presented a method for learning robust fundamental matrix estimators from data. Our experiments indicate that the learned estimators are robust and accurate on a variety of datasets. Our approach enables data-driven specialization of estimators to certain scenarios, such as ones encountered in autonomous driving. Our experiments indicate that general robust estimators that are competitive with the state of the art can be learned directly from data, alleviating the need for extensive modeling of error statistics.

We view the presented approach as a step towards modular SLAM and SfM systems that combine the power of deep networks with mathematically sound geometric modeling. In addition to the presented problem instances, our approach is directly applicable to other problems in multiple-view geometry that are based on the Direct Linear Transform, such as triangulation or the PnP problem [13]. Furthermore, the general scheme of the algorithm may be applicable to other problems where IRLS is employed [47].

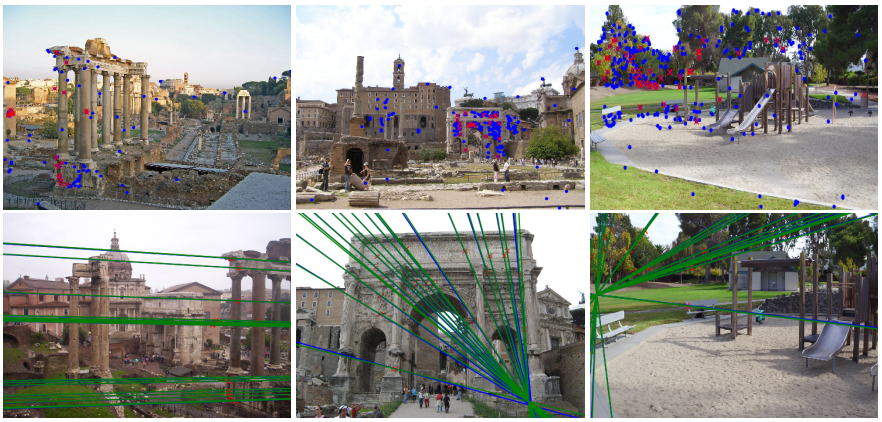


Figure 3. Image pairs from *Roman Forum* (first and second column) and *Tanks and Temples* (last column). Top row: First image with inliers (red) and outliers (blue). Bottom row: Epipolar lines of a random subset of inliers in the second image. We show the epipolar lines of our estimate (green) and of the groundtruth (blue). Images have been scaled for visualization.

References

1. Agrawal, P., Carreira, J., Malik, J.: Learning to see by moving. In: ICCV (2015)
2. Andrychowicz, M., Denil, M., Colmenarejo, S.G., Hoffman, M.W., Pfau, D., Schaul, T., de Freitas, N.: Learning to learn by gradient descent by gradient descent. In: NIPS (2016)
3. Brachmann, E., Krull, A., Nowozin, S., Shotton, J., Michel, F., Gumhold, S., Rother, C.: DSAC - differentiable RANSAC for camera localization. In: CVPR (2017)
4. Chin, T., Purkait, P., Eriksson, A.P., Suter, D.: Efficient globally optimal consensus maximisation with tree search. IEEE Transactions on Pattern Anaysis and Machine Intelligence **39**(4), 758–772 (2017)
5. Chum, O., Matas, J.: Matching with PROSAC - progressive sample consensus. In: CVPR (2005)
6. DeTone, D., Malisiewicz, T., Rabinovich, A.: Deep image homography estimation. arXiv:1606.03798 (2016)
7. Fischler, M.A., Bolles, R.C.: Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. Communications of the ACM **24**(6), 381–395 (1981)
8. Fitzgibbon, A.W.: Robust registration of 2D and 3D point sets. Image and Vision Computing **21**(13-14) (2003)
9. Geiger, A., Lenz, P., Urtasun, R.: Are we ready for autonomous driving? The KITTI vision benchmark suite. In: CVPR (2012)
10. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: CVPR (2014)
11. Grabner, H., Gall, J., Gool, L.J.V.: What makes a chair a chair? In: CVPR (2011)
12. Hartley, R., Zisserman, A.: Multiple view geometry in computer vision. Cambridge University Press (2000)
13. Hartley, R.I.: In defense of the eight-point algorithm. IEEE Transactions on Pattern Anaysis and Machine Intelligence **19**(6), 580–593 (1997)
14. Hoseinnezhad, R., Bab-Hadiashar, A.: An M-estimator for high breakdown robust estimation in computer vision. Computer Vision and Image Understanding **115**(8), 1145–1156 (2011)
15. Ionescu, C., Vantzos, O., Sminchisescu, C.: Matrix backpropagation for deep networks with structured layers. In: ICCV (2015)
16. Kendall, A., Cipolla, R.: Geometric loss functions for camera pose regression with deep learning. In: CVPR (2017)
17. Kendall, A., Grimes, M., Cipolla, R.: PoseNet: A convolutional network for real-time 6-DOF camera relocalization. In: ICCV (2015)
18. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: ICLR (2015)
19. Knapitsch, A., Park, J., Zhou, Q.Y., Koltun, V.: Tanks and temples: Benchmarking large-scale scene reconstruction. ACM Transactions on Graphics **36**(4) (2017)
20. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional neural networks. In: NIPS (2012)
21. Lebeda, K., Matas, J., Chum, O.: Fixing the locally optimized RANSAC. In: BMVC (2012)
22. Li, H.: Consensus set maximization with guaranteed global optimality for robust geometry estimation. In: ICCV (2009)
23. Lin, T., Maire, M., Belongie, S.J., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft COCO: Common objects in context. In: ECCV (2014)
24. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: CVPR (2015)
25. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision **60**(2), 91–110 (2004)

26. Maas, A.L., Hannun, A.Y., Ng, A.Y.: Rectifier nonlinearities improve neural network acoustic models. In: ICML Workshops (2013)
27. Mur-Artal, R., Montiel, J.M.M., Tardós, J.D.: ORB-SLAM: A versatile and accurate monocular SLAM system. IEEE Transactions on Robotics **31**(5), 1147–1163 (2015)
28. Nguyen, T., Chen, S.W., Shivakumar, S.S., Taylor, C.J., Kumar, V.: Unsupervised deep homography: A fast and robust homography estimation model. arXiv:1709.03966 (2017)
29. Qi, C.R., Su, H., Mo, K., Guibas, L.J.: PointNet: Deep learning on point sets for 3D classification and segmentation. In: CVPR (2016)
30. Raguram, R., Chum, O., Pollefeys, M., Matas, J., Frahm, J.: USAC: A universal framework for random sample consensus. IEEE Transactions on Pattern Anaysis and Machine Intelligence **35**(8), 2022–2038 (2013)
31. Rocco, I., Arandjelovic, R., Sivic, J.: Convolutional neural network architecture for geometric matching. In: CVPR (2017)
32. Rousseeuw, P.J.: Least median of squares regression. Journal of the American Statistical Association **79**(388), 871–880 (1984)
33. Rublee, E., Rabaud, V., Konolige, K., Bradski, G.R.: ORB: An efficient alternative to SIFT or SURF. In: ICCV (2011)
34. Savinov, N., Seki, A., Ladicky, L., Sattler, T., Pollefeys, M.: Quad-networks: Unsupervised learning to rank for interest point detection. In: CVPR (2017)
35. Schönberger, J.L., Frahm, J.M.: Structure-from-motion revisited. In: CVPR (2016)
36. Tennakoon, R.B., Bab-Hadiashar, A., Cao, Z., Hoseinnezhad, R., Suter, D.: Robust model fitting using higher than minimal subset sampling. IEEE Transactions on Pattern Anaysis and Machine Intelligence **38**(2), 350–362 (2016)
37. Torr, P.H.S.: Bayesian model estimation and selection for epipolar geometry and generic manifold fitting. International Journal of Computer Vision **50**(1), 35–61 (2002)
38. Torr, P.H.S., Murray, D.W.: The development and comparison of robust methods for estimating the fundamental matrix. International Journal of Computer Vision **24**(3), 271–300 (1997)
39. Torr, P.H.S., Zisserman, A.: MLESAC: A new robust estimator with application to estimating image geometry. Computer Vision and Image Understanding **78**(1), 138–156 (2000)
40. Ulyanov, D., Vedaldi, A., Lempitsky, V.S.: Improved texture networks: Maximizing quality and diversity in feed-forward stylization and texture synthesis. In: CVPR (2017)
41. Ummenhofer, B., Zhou, H., Uhrig, J., Mayer, N., Ilg, E., Dosovitskiy, A., Brox, T.: DeMoN: Depth and motion network for learning monocular stereo. In: CVPR (2017)
42. Vongkulbhisal, J., De la Torre, F., Costeira, J.P.: Discriminative optimization: Theory and applications to point cloud registration. In: CVPR (2017)
43. Wilson, K., Snavely, N.: Robust global translations with 1DSfM. In: ECCV (2014)
44. Yang, J., Li, H., Jia, Y.: Optimal essential matrix estimation via inlier-set maximization. In: ECCV (2014)
45. Yi, K.M., Trulls, E., Lepetit, V., Fua, P.: LIFT: Learned invariant feature transform. In: ECCV (2016)
46. Yi, K.M., Trulls, E., Ono, Y., Lepetit, V., Salzmann, M., Fua, P.: Learning to find good correspondences. In: CVPR (2018)
47. Zach, C.: Robust bundle adjustment revisited. In: ECCV (2014)
48. Zaheer, M., Kottur, S., Ravanbakhsh, S., Póczos, B., Salakhutdinov, R., Smola, A.J.: Deep sets. In: NIPS (2017)
49. Zhang, Z.: Determining the epipolar geometry and its uncertainty: A review. International Journal of Computer Vision **27**(2), 161–195 (1998)
50. Zhou, Q., Park, J., Koltun, V.: Fast global registration. In: ECCV (2016)