Unsupervised Deep Homography: A Fast and Robust Homography Estimation Model

Ty Nguyen*, Steven W. Chen*, Shreyas S. Shivakumar, Camillo J. Taylor, Vijay Kumar

***Abstract*— Homography estimation between multiple aerial images can provide relative pose estimation for collaborative autonomous exploration and monitoring. The usage on a robotic system requires a fast and robust homography estimation algorithm. In this study, we propose an unsupervised learning algorithm that trains a Deep Convolutional Neural Network to estimate planar homographies. We compare the proposed algorithm to traditional feature-based and direct methods, as well as a corresponding supervised learning algorithm. Our empirical results demonstrate that compared to traditional approaches, the unsupervised algorithm achieves faster inference speed, while maintaining comparable or better accuracy and robustness to illumination variation. In addition, our unsupervised method has superior adaptability and performance compared to the corresponding supervised deep learning method. Our image dataset and a Tensorflow implementation of our work are available at** $https://github.com/tynguyen/unsupervisedDeepHomographyRAL2018$**.**

I. INTRODUCTION

A homography is a mapping between two images of a planar surface from different perspectives. They play an essential role in robotics and computer vision applications such as image mosaicing [1], monocular SLAM [2], 3D camera pose reconstruction [3] and virtual touring [4], [5]. For example, homographies are applicable in scenes viewed at a far distance by an arbitrary moving camera [6], which are the situations encountered in UAV imagery. However, to work well in the aerial multi-robot setting, the homography estimation algorithm needs to be reliable and fast.

The two traditional approaches for homography estimation are direct methods and feature-based methods [7]. Direct methods, such as the seminal Lucas-Kanade algorithm [8], use pixel-to-pixel matching by shifting or warping the images relative to each other and comparing the pixel intensity values using an error metric such as the sum of squared differences (SSD). They initialize a guess for the homography parameters and use a search or optimization technique such as gradient descent to minimize the error function [9]. The robustness of direct methods can be improved by using different performance criterion such as the enhanced correlation coefficient (ECC) [10], integrating feature-based methods with direct methods [11], or by representing the images in the Fourier domain [12]. In addition, the speed of direct methods can be increased by using efficient compositional image alignment schemes [13].

The authors are with GRASP Lab, University of Pennsylvania, Philadelphia, PA 19104, USA, {tynguyen, chenste, sshreyas, cjtaylor, kumar}@seas.upenn.edu.

*: The authors have equal contributions

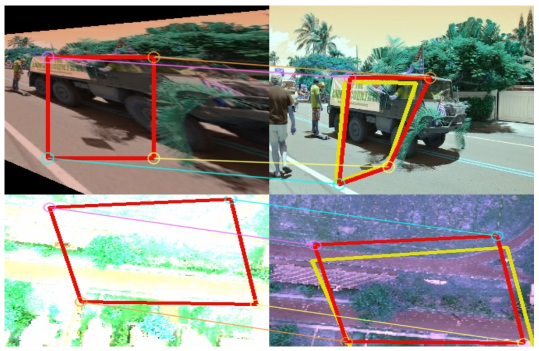


Fig. 1: Above: Synthetic data; Below: Real data; Homography estimation results from the unsupervised neural network. Red represents the ground truth correspondences, and yellow represents the estimated correspondences. These images depict an example of large levels of displacement and illumination shifts in which feature-based, direct and/or supervised learning methods fail.

The second approach are feature-based methods. These methods first extract keypoints in each image using local invariant features (e.g. Scale Invariant Feature Transform (SIFT) [14]). They then establish a correspondence between the two sets of keypoints using feature matching, and use RANSAC [15] to find the best homography estimate. While these methods have better performance than direct methods, they can be inaccurate when they fail to detect sufficient keypoints, or produce incorrect keypoint correspondences due to illumination and large viewpoint differences between the images [16]. In addition, these methods are significantly faster than direct methods but can still be slow due to the computation of the features, leading to the development of other feature types such as Oriented FAST and Rotated BRIEF (ORB) [17] which are more computationally efficient than SIFT, but have worse performance.

Inspired by the success of data-driven Deep Convolutional Neural Networks (CNN) in computer vision, there has been an emergence of CNN approaches to estimating optical flow [18], [19], [20], dense matching [21], [22], depth estimation [23], and homography estimation [24]. Most of these works, including the most relevant work on homography estimation, treat the estimation problem as a supervised learning task. These supervised approaches use ground truth labels, and as a result are limited to synthetic datasets where the ground truth can be generated for free, or require costly labeling of real-world data sets.

Our work develops an unsupervised, end-to-end, deep learning algorithm to estimate homographies. It improves upon these prior traditional and supervised learning methods by minimizing a pixel-wise intensity error metric that does not need ground truth data. Unlike the hand-crafted feature-based approaches, or the supervised approach that needs costly labels, our model is adaptive and can easily learn good features specific to different data sets. Furthermore, our framework has fast inference times since it is highly parallel. These adaptive and speed properties make our unsupervised networks especially suitable for real world robotic tasks, such as stitching UAV images.

We demonstrate that our unsupervised homography estimation algorithm has comparable or better accuracy, and better inference speed, than feature-based, direct, and supervised deep learning methods on synthetic and real-world UAV data sets. In addition, we demonstrate that it can handle large displacements ($\sim 65\%$ image overlap) with large illumination variation. Fig. 1 illustrates qualitative results on these data sets, where our unsupervised method is able to estimate the homography whereas the other approaches cannot.

Our unsupervised algorithm is a hybrid approach that combines the strengths of deep learning with the strengths of both traditional direct methods and feature-based methods. It is similar to feature-based methods because it also relies on features to compute the homography estimates, but it differs in that it learns the features rather than defining them. It is also similar to the direct methods because the error signal used to drive the network training is a pixel-wise error. However, rather than performing an online optimization process, it transfers the computation offline and "caches" the results through these learned features. Similar unsupervised deep learning approaches have been successful in computer vision tasks such as monocular depth and camera motion estimation [25], indicating that our framework can be scaled to tackle general nonlinear motions such as those encountered in optical flow.

II. Problem Formulation

We assume that images are obtained by a perspective pinhole camera and present points by homogeneous coordinates, so that a point $(u,v)^T$ is represented as $(u,v,1)^T$ and a point $(x,y,z)^T$ is equivalent to the point $(x/z,y/z,1)^T$. Suppose that $\mathbf{x}=(u,v,1)^T$ and $\mathbf{x}'=(u',v',1)^T$ are two points. A planar projective transformation or homography that maps $\mathbf{x}\leftrightarrow\mathbf{x}'$ is a linear transformation represented by a non-singular 3×3 matrix $\mathbf{H}$ such that:

$$\begin{bmatrix}u'\\v'\\1\end{bmatrix}=\begin{bmatrix}h_{11}&h_{12}&h_{13}\\h_{21}&h_{22}&h_{23}\\h_{31}&h_{32}&h_{33}\end{bmatrix}\begin{bmatrix}u\\v\\1\end{bmatrix}\quad \text{Or}\quad \mathbf{x}'=\mathbf{H}\mathbf{x} \tag{1}$$

Since $\mathbf{H}$ can be multiplied by an arbitrary non-zero scale factor without altering the projective transformation, only the ratio of the matrix elements is significant, leaving $\mathbf{H}$ eight independent ratios corresponding to eight degrees of freedom. This mapping equation can also represented by two equations:

$$u'=\frac{h_{11}u+h_{12}v+h_{13}}{h_{31}u+h_{32}v+h_{33}};v'=\frac{h_{21}u+h_{22}v+h_{23}}{h_{31}u+h_{32}v+h_{33}} \tag{2}$$

The problem of finding the homography induced by two images I^A and I^B is to find a homography $\mathbf{H}^{AB}$ such that Eqn. (1) holds for all points in the overlapping of the two images.

III. Supervised Deep Homography Model

The deep learning approach most similar to our work is the Deep Image Homography Estimation [24]. In this work, DeTone et al. use supervised learning to train a deep neural network on a synthetic data set. They use the 4-point homography parameterization $\mathbf{H}_{4pt}$ [26] rather than the conventional 3×3 parameterization $\mathbf{H}$. Suppose that $\mathbf{u}_k^A=(u_k^A,v_k^A,1)^T$ and $\mathbf{u}_k^B=(u_k^B,v_k^B,1)^T$ for $k=1,2,3,4$ are 4 fixed points in image I^A and I^B respectively, such that $\mathbf{u}_2^k=\mathbf{H}\mathbf{u}_1^k$. Let $\Delta u_k=u_k^B-u_k^A$, $\Delta v_k=v_k^B-v_k^A$. Then $\mathbf{H}_{4pt}$ is the 4×2 matrix of points $(\Delta u_k,\Delta v_k)$. Both parameterizations are equivalent since there is a one-to-one correspondence between them.

In a deep learning framework though, this parameterization is more suitable than the 3×3 parameterization $\mathbf{H}$ because $\mathbf{H}$ mixes the rotation, translation, scale, and shear components of the homography transformation. The rotation and shear components tend to have a much smaller magnitude than the translation component, and as a result although an error in their values can greatly impact $\mathbf{H}$, it will have a small effect on the $L2$ loss function of the elements of $\mathbf{H}$, which is detrimental for training the neural network. In addition, the high variance in the magnitude of the elements of the 3×3 homography matrix makes it difficult to enforce $\mathbf{H}$ to be non-singular. The 4-point parameterization does not suffer from these problems.

The network architecture is based on VGGNet [27], and is depicted in Fig. 2(a). The network input is a batch of image patch pairs. The patch pairs are generated by taking a full-sized image, cropping a square patch P^A at a random position p, perturbing the four corners of by a random value within $[-\rho,\rho]$ to generate a homography $\mathbf{H}^{AB}$, applying $(\mathbf{H}^{AB})^{-1}$ to the full-sized image, and then cropping a square patch P^B of the same size and at the same location as the patch P^A from the warped image. These image patches are used to avoid strange border effects near the edges during the synthetic data generation process, and to standardize the network input size. The applied homography $\mathbf{H}^{AB}$ is saved in the 4 point parameterization format, $\mathbf{H}^*_{4pt}$. The network outputs a 4 point parameterization estimate $\tilde{\mathbf{H}}_{4pt}$.

The error signal used for gradient backpropagation is the Euclidean $L2$ norm, denoted as $\mathbf{L}_H$, of the estimated 4-point homography $\tilde{\mathbf{H}}_{4pt}$ versus the ground truth $\mathbf{H}^*_{4pt}$:

$$\mathbf{L}_H=\frac{1}{2}||\tilde{\mathbf{H}}_{4pt}-\mathbf{H}^*_{4pt}||_2^2 \tag{3}$$

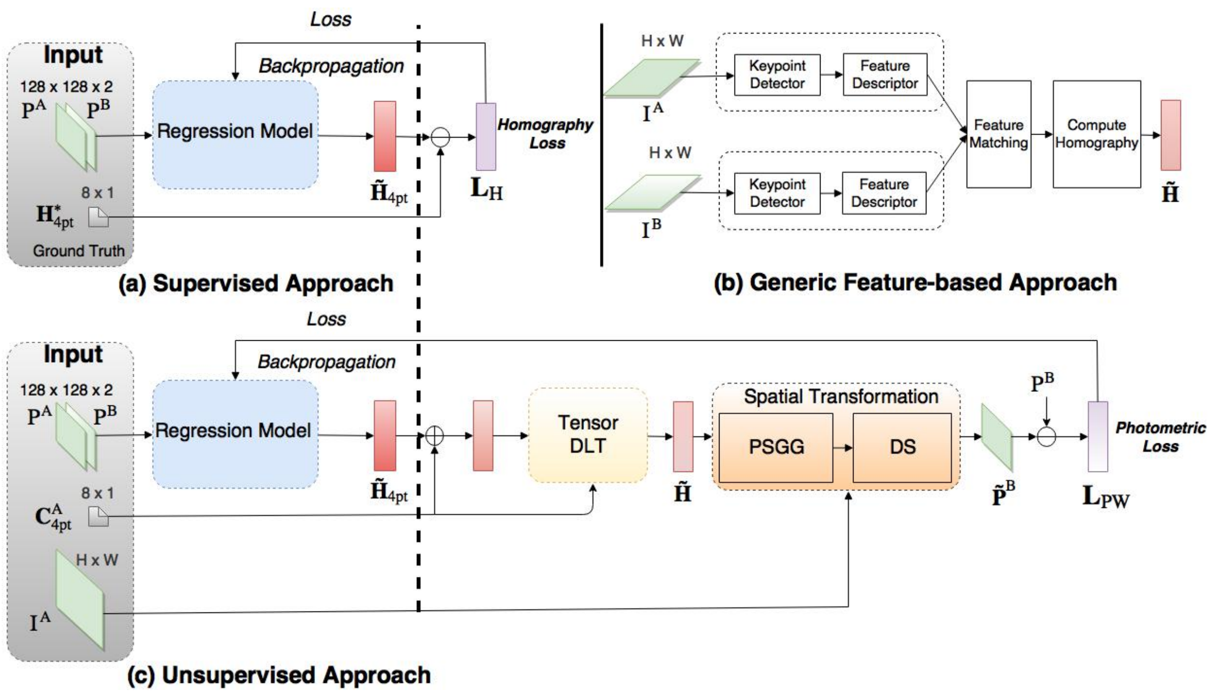


Fig. 2: Overview of homography estimation methods; (a) Benchmark supervised deep learning approach; (b) Feature-based methods; and (c) Our unsupervised method. DLT: direct linear transform; PSGG: parameterized sampling grid generator; DS: differentiable sampling.

IV. Unsupervised Deep Homography Model

While the supervised deep learning method has promising results, it is limited in real world applications since it requires ground truth labels. Drawing inspiration from traditional direct methods for homography estimation, we can define an analogous loss function. Given an image pair $I^A(\mathbf{x})$ and $I^B(\mathbf{x})$ with discrete pixel locations represented by homogeneous coordinates $\{\mathbf{x}_i = (x_i, y_i, 1)^T\}$, we want our network to output $\tilde{\mathbf{H}}_{4pt}$ that minimizes the average $L1$ pixel-wise photometric loss

$$\mathbf{L}_{PW} = \frac{1}{|\mathbf{x}_i|}\sum_{\mathbf{x}_i} |I^A(\mathscr{H}(\mathbf{x}_i)) - I^B(\mathbf{x_i})| \tag{4}$$

where $\tilde{\mathbf{H}}_{4pt}$ defines the homography transformation $\mathscr{H}(\mathbf{x}_i)$. We chose the $L1$ error versus the $L2$ error because previous work has observed that it is more suitable for image alignment problems [28], and empirically we found the network to be easier to train with the $L1$ error. This loss function is unsupervised since there is no ground truth label. Similar to the supervised case, we choose the 4-point parameterization which is more suitable than the 3×3 parameterization.

In order to compare our unsupervised deep learning algorithm with the supervised algorithm, we use the same VGGNet architecture to output the $\tilde{\mathbf{H}}_{4pt}$. Fig. 2(c) depicts our unsupervised learning model. The regression module represents the VGGNet architecture and is shared by both the supervised and unsupervised methods. Although we do not investigate other possible architectures, different regression models such as SqueezeNet [29] may yield better performance due to advantages in size and computation requirements. The second half of Fig. 2(c) represents the main contribution of this work, which consists of the differentiable layers that allow the network to be successfully trained with the loss function (4).

Using the pixel-wise photometric loss function yields additional training challenges. First, every operation, including the warping operation $\mathscr{H}(\mathbf{x}_i)$, must remain differentiable to allow the network to be trained via backpropagation. Second, since the error signal depends on differences in image intensity values rather than the differences in the homography parameters, training the deep network is not necessarily as easy or stable. Another implication of using a pixel-wise photometric loss function is the implied assumption that lighting and contrast between the input images remains consistent. In traditional direct methods such as ECC, this appearance variation problem is addressed by modifying the loss function or preprocessing the images. In our unsupervised algorithm, we standardize our images by the mean and variance of the intensities of all pixels in our training dataset, perform data augmentation by injecting random illumination shifts, and use the standard $L1$ photometric loss. We found that even without modifying the loss function, our deep neural network is still able to learn to be invariant to illumination changes.

A. Model Inputs

The input to our model consists of three parts. The first part is a 2-channel image of size $128 \times 128 \times 2$ which is the stack of P^A and P^B - two patches cropped from the two images I^A and I^B. The second part is the four corners in I^A, denoted as $\mathbf{C}_{4pt}^{A}$. Image I^{A} is also part of the input as it is necessary for warping.

B. Tensor Direct Linear Transform

We develop a Tensor Direct Linear Transform (Tensor DLT) layer to compute a differentiable mapping from the 4-point parameterization $\tilde{\mathbf{H}}_{\mathbf{4pt}}$ to $\tilde{\mathbf{H}}$, the 3×3 parameterization of homography. This layer essentially applies the DLT algorithm [30] to tensors, while remaining differentiable to allow backpropagation during training. As shown in Fig. 2(c), the input to this layer are the corresponding corners in the image pairs $\mathbf{C}_{4pt}^{A}$ and $\tilde{\mathbf{C}}_{4pt}^{B}$, and the output is the estimate of the 3×3 homography parameterization $\tilde{\mathbf{H}}$.

The DLT algorithm is used to solve for the homography matrix $\mathbf{H}$ given a set of four point correspondences [30]. Let $\mathbf{H}$ be the homography induced by a set of four 2D to 2D correspondences, $\mathbf{x}_i \leftrightarrow \mathbf{x}_i'$. According to the definition of a homography given in Eqn. (1), $\mathbf{x}_i' \sim \mathbf{H}\mathbf{x}_i$. This relation can also be expressed as $\mathbf{x}_i' \times \mathbf{H}\mathbf{x}_i = 0$.

Let $\mathbf{h}^{jT}$ be the j-th row of $\mathbf{H}$, then:

$$\mathbf{H}\mathbf{x}_i = \begin{bmatrix} \mathbf{h}^{1T}\mathbf{x}_i \\ \mathbf{h}^{2T}\mathbf{x}_i \\ \mathbf{h}^{3T}\mathbf{x}_i \end{bmatrix} = \begin{bmatrix} \mathbf{x}_i^T\mathbf{h}^1 \\ \mathbf{x}_i^T\mathbf{h}^2 \\ \mathbf{x}_i^T\mathbf{h}^3 \end{bmatrix} \tag{5}$$

where $\mathbf{h}^j$ is the column vector representation of $\mathbf{h}^{jT}$.

Let $\mathbf{x}_i' = (u_i', v_i', 1)^T$, then:

$$\mathbf{x}_i' \times \mathbf{H}\mathbf{x}_i = \begin{bmatrix} v_i'\mathbf{x}_i^T\mathbf{h}^3 - \mathbf{x}_i^T\mathbf{h}^2 \\ \mathbf{x}_i^T\mathbf{h}^1 - u_i'\mathbf{x}_i^T\mathbf{h}^3 \\ u_i'\mathbf{x}_i^T\mathbf{h}^2 - v_i'\mathbf{x}_i^T\mathbf{h}^1 \end{bmatrix} = 0 \tag{6}$$

This equation can be rewritten as:

$$\begin{bmatrix} 0_{3\times1}^T & -\mathbf{x}_i^T & v_i'\mathbf{x}_i^T \\ \mathbf{x}_i^T & 0_{3\times1}^T & -u_i'\mathbf{x}_i^T \\ -v_i'\mathbf{x}_i^T & u_i'\mathbf{x}_i^T & 0_{3\times1}^T \end{bmatrix} \begin{bmatrix} \mathbf{h}^1 \\ \mathbf{h}^2 \\ \mathbf{h}^3 \end{bmatrix} = 0. \tag{7}$$

which has the form $\mathbf{A}_i^{(3)}\mathbf{h} = \mathbf{0}$ for each $i = 1,2,3,4$ correspondence pair, where $\mathbf{A}_i^{(3)}$ is a 3×9 matrix, and $\mathbf{h}$ is a vector with 9 elements consisting of the entries of $\mathbf{H}$. Since the last row in $\mathbf{A}_i^{(3)}$ is dependent on the other rows, we are left with two linear equations $\mathbf{A}_i\mathbf{h} = \mathbf{0}$ where $\mathbf{A}_i$ is the first 2 rows of $\mathbf{A}_i^{(3)}$.

Given a set of 4 correspondences, we can create a system of equations to solve for $\mathbf{h}$ and thus $\mathbf{H}$. For each i, we can stack $\mathbf{A}_i$ to form $\mathbf{A}\mathbf{h} = \mathbf{0}$. Solving for $\mathbf{h}$ results in finding a vector in the null space of $\mathbf{A}$. One popular approach is singular value decomposition (SVD) [31], which is a differentiable operation. However, taking the gradients in SVD has high time complexity and has practical implementation issues [32]. An alternative solution to this problem is to make the assumption that the last element of $\mathbf{h}^3$, which is $\mathbf{H}_{33}$ is equal to 1 [33].

With this assumption and the fact that $\mathbf{x}_i = (u_i, v_i, 1)$, we can rewrite Eqn. (7) in the form $\hat{\mathbf{A}}_i\hat{\mathbf{h}} = \hat{\mathbf{b}}_i$ for each $i = 1,2,3,4$ correspondence points where $\hat{\mathbf{A}}_i$ is the 2×8 matrix representing the first 8 columns of $\mathbf{A}_i$,

$$\hat{\mathbf{A}}_i = \begin{bmatrix} 0 & 0 & 0 & -u_i & -v_i & -1 & v_i'u_i & v_i'v_i \\ u_i & v_i & 1 & 0 & 0 & 0 & -u_i'u_i & -u_i'v_i \end{bmatrix},$$

$\hat{\mathbf{b}}_i$ is a vector with 2 elements representing the last column of $\mathbf{A}_i$ subtracted from both sides of the equation,

$$\hat{\mathbf{b}}_i = [-v_i', u_i']^T,$$

and $\hat{\mathbf{h}}$ is a vector consisting of the first 8 elements of $\mathbf{h}$ (with $\mathbf{H}_{33}$ omitted).

By stacking these equations, we get:

$$\hat{\mathbf{A}}\hat{\mathbf{h}} = \hat{\mathbf{b}}, \tag{8}$$

Eqn. (8) has a desirable form because $\hat{\mathbf{h}}$, and thus $\mathbf{H}$, can be solved for using $\hat{\mathbf{A}}^{+}$, the pseudo-inverse of $\hat{\mathbf{A}}$. This operation is simple and differentiable with respect to the coordinates of $\mathbf{x}_i$ and $\mathbf{x}_i'$. In addition, the gradients are easier to calculate than for SVD.

This approach may still fail if the correspondence points are collinear: if three of the correspondence points are on the same line, then solving for $\mathbf{H}$ is undetermined. We alleviate this problem by first making the initial guess of $\mathbf{H}_{4pt}$ to be zero, implying that $\tilde{\mathbf{C}}_{4pt}^{B} \sim \mathbf{C}_{4pt}^{A}$. We then set a small learning rate such that after each training iteration, $\tilde{\mathbf{C}}_{4pt}^{B}$ does not move too far away from $\mathbf{C}_{4pt}^{A}$.

C. Spatial Transformation Layer

The next layer applies the 3×3 homography estimate $\tilde{\mathbf{H}}$ output by the Tensor DLT to the pixel coordinates $\mathbf{x}_i$ of image I^{A} in order to get warped coordinates $\mathscr{H}(\mathbf{x}_i)$. These warped coordinates are necessary in computing the photometric loss function in Eqn. (4) that will train our neural network. In addition to warping the coordinates, this layer must also be differentiable so that the error gradients can flow through via backpropagation. We thus extend the Spatial Transformer Layer introduced in [34] by applying it to homography transformations.

This layer performs an inverse warping in order to avoid holes in the warped image. This process consists of 3 steps: (1) Normalized inverse computation $\tilde{\mathbf{H}}_{inv}$ of the homography estimate; (2) Parameterized Sampling Grid Generator (PSGG); and (3) Differentiable Sampling (DS).

The first step, computing a normalized inverse, involves normalizing the height and width coordinates of images I^{A} and I^{B} into a range such that $-1 \le u_i, v_i \le 1$ and $-1 \le u_i', v_i' \le 1$. Thus given a 3×3 homography estimate $\tilde{\mathbf{H}}$, the inverse $\tilde{\mathbf{H}}_{inv}$ used for warping is computed as follows:

$$\tilde{\mathbf{H}}_{inv} = M^{-1}\tilde{\mathbf{H}}^{-1}M$$
$$\text{where } M = \begin{bmatrix} W'/2 & 0 & W'/2 \\ 0 & H'/2 & H'/2 \\ 0 & 0 & 1 \end{bmatrix}$$

with W' and H' are the width and height of the I_B.

The second step (PSGG) creates a grid $G = \{G_i\}$ of the same size as the second image I^{B}. Each grid element $G_i = (u_i', v_i')$ corresponds to pixels of the second image I^{B}. Applying the inverse homography $\tilde{\mathbf{H}}_{inv}$ to these grid coordinates provides a grid of pixels in the first image I^{A}.

$$\begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} = \mathscr{H}_{inv}(G_i) = \tilde{\mathbf{H}}_{inv} \begin{bmatrix} u_i' \\ v_i' \\ 1 \end{bmatrix} \tag{9}$$

Based on the sampling points $\mathscr{H}_{inv}(G_i)$ computed from PSGG, the last step (DS) produces a sampled warped image V of size $H' \times W'$ with C channels, where $V(\mathbf{x}_i) = I^A(\mathscr{H}(\mathbf{x}_i))$.

The sampling kernel $k(\cdot)$ is applied to the grid $\mathscr{H}_{inv}(G_i)$ and the resulting image V is defined as

$$V_i^C = \sum_n^H \sum_m^W I_{nm}^c k(u_i - m; \Phi_u) k(v_i - n; \Phi_v), \quad \forall i \in [1...H'W'] \quad , \forall c \in [1..C] \tag{10}$$

where H, W are the height and width of the input image I^A, Φ_u and Φ_v are the parameters of $k(\cdot)$ defining the image interpolation. I_{nm}^c is the value at location (n,m) in channel c of the input image, and V_i^c is the value of the output pixel at location (u_i, v_i) in channel c. Here, we use bilinear interpolation such that the Eqn. (10) becomes

$$V_i^C = \sum_n^H \sum_m^W I_{nm}^c \max(0, 1 - |u_i - m|) \max(0, 1 - |v_i - n|) \tag{11}$$

To allow backpropagation of the loss function, gradients with respect to I and G for bilinear interpolation are defined as

$$\frac{\partial V_i^c}{\partial I_{nm}^c} = \sum_n^H \sum_m^W \max(0, 1 - |u_i - m|) \max(0, 1 - |v_i - n|) \tag{12}$$

$$\frac{\partial V_i^c}{\partial u^i} = \sum_n^H \sum_m^W I_{nm}^c \max(0, 1 - |v_i - n|) \begin{cases} 0 \text{ if } |m - u_i| \geq 1 \\ 1 \text{ if } m \geq u_i \\ -1 \text{ if } m < u_i \end{cases} \tag{13}$$

$$\frac{\partial V_i^c}{\partial v^i} = \sum_n^H \sum_m^W I_{nm}^c \max(0, 1 - |u_i - m|) \begin{cases} 0 \text{ if } |n - v_i| \geq 1 \\ 1 \text{ if } n \geq v_i \\ -1 \text{ if } n < v_i \end{cases} \tag{14}$$

This allows backpropagation of the loss gradients using the chain rule because $\frac{\partial u_i}{\partial h_{jk}}$ and $\frac{\partial v_i}{\partial h_{jk}}$ can be easily derived from Eqn. 2.

V. Evaluation Results

The intended use case for our algorithm is in estimating homographies for aerial multi-robot systems applications such as image mosaicing and collision avoidance. As a result, we demonstrate our unsupervised algorithm's accuracy, inference speed, and robustness to illumination variation relative to SIFT, ORB, ECC and the supervised deep learning method. We evaluate these methods on a synthetic dataset similar to the dataset used in [24], and on a real-world aerial image dataset. Since ORB's performance is inferior to that of SIFT, we only report ORB's performance in Fig. 4 and omit it in the remaining figures.

Both the supervised and unsupervised approaches use the VGGNet architecture to generate homography estimates $\tilde{\mathbf{H}}_{\mathbf{4pt}}$. The deep learning approaches are implemented in Tensorflow [35] using stochastic gradient descent with a batch size of 128, and an Adam Optimizer [36] with, $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\varepsilon = 10^{-8}$. We empirically chose the initial learning rate for the supervised algorithm and unsupervised algorithm to be 0.0005 and 0.0001 respectively.

The ECC direct method is a standard Python OpenCV implementation while the feature-based approaches are Python OpenCV implementations of SIFT RANSAC and ORB RANSAC. We found that in our synthetic dataset, using all detected features gives better performance, while in our aerial dataset, choosing the 50 best features is superior. These feature pairs are then used to calculate the homography using RANSAC with a threshold of 5 pixels. For the ECC method, we use identity matrix as the initialization and set 1000 as the maximum number of iterations.

A. Synthetic Data Results

This section analyzes the performance profile of the Unsupervised, Supervised, SIFT, and ECC methods on our synthetic dataset. We want to test how well our approach performs under illumination variation and large image displacement.

To account for illumination variation, we globally standardize our images based on the mean and variance of pixel intensities of all images in our training dataset. We additionally inject random color, brightness and gamma shifts during the training. We do not utilize any further preprocessing and use the $L1$ photometric loss function. To highlight the effect of displacement amount on each method, we break down the accuracy performance in terms of: 85% image overlap (small displacement), 75% image overlap (moderate displacement), and 65% image overlap (large displacement). We follow the synthetic data generation process on the MS-COCO dataset used in [24]. The amount of image overlap is controlled by the point perturbation parameter ρ. The evaluation metric is the 4pt-Homography RMSE from Eqn. (3) comparing the estimated homography to the ground truth homography.

We train the deep networks from scratch for $300,000$ iterations over ~ 30 hours, using two GPUs. This long training procedure only needs to be performed once, as the resulting model can be used as an initial pre-trained model for other data sets. We observed that the supervised model started overfitting after $150,000$ iterations so stopped training early. SIFT, ORB and ECC estimated homographies using the full images, while the deep learning methods are only given access to the small patches ($\sim 21\%$ pixels). This disadvantages our methods, and would result in better performance for the traditional methods, at the expense of slower running times.

Fig. 3 displays the results of each method broken down by overlap and performance percentile. We break down the results by performance percentile to illustrate the various performance profiles of each method. Specifically, SIFT tends to do very well 60% of the time, but in the worst 40% of the time it performs very poorly, sometimes completely failing to detect enough features to estimate a homography. On the other hand, the deep learning methods tends to have much more consistent performance, which can be more desireable in practical applications such as using homographies for collision avoidance for aerial multi-robot systems. Both the

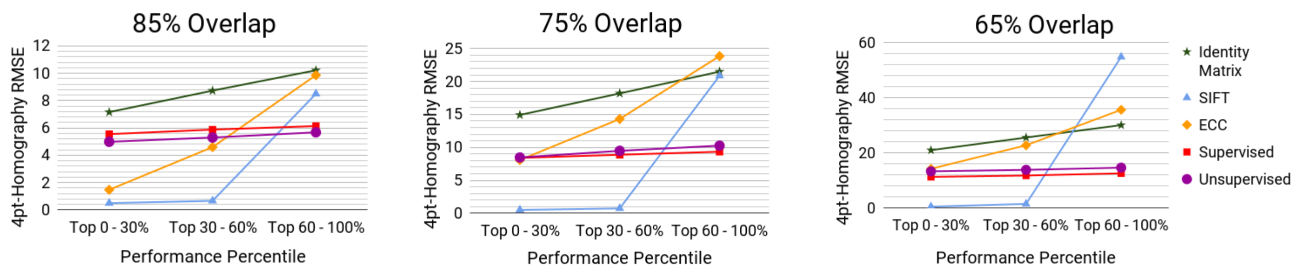


Fig. 3: Synthetic 4pt-Homography RMSE (lower is better). Unsupervised has comparable performance with the supervised method and performs better than the other approaches especially when the displacement is large.

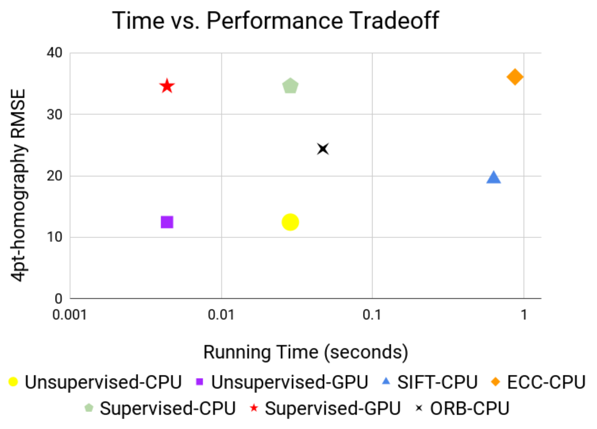


Fig. 4: Speed Versus Performance Tradeoff. Lower left is better. Suffixes GPU and CPU reflect the computational resource. All the feature-based methods are run on the CPU. The unsupervised network run on the GPU dominates all the other methods by having both the highest throughput and best performance.

learning methods and the feature-based methods outperform the direct method (ECC).

Interestingly, whereas direct method ECC has problems with illumination variation and large displacement, our unsupervised method is able to handle these scenarios even though it uses photometric loss functions. One potential hypothesis is that our method can be viewed as a hybrid between direct methods and feature based methods. The large receptive field of neural networks may allow it to handle large image displacement better than a direct method. In addition, whereas image gradients are used to update homography parameters in direct methods, with neural networks, these gradients are used to update network parameters which correspond to improving learned features. Finally, direct methods are an online optimization process that use gradients from a single pair of images, whereas training a deep network is an offline optimization process that averages gradients across multiple images. Injecting noise into this training process can further improve robustness to different appearance variations. Understanding the relationship between the neural network and photometric loss functions is an important direction for future work.

B. Aerial Dataset Results

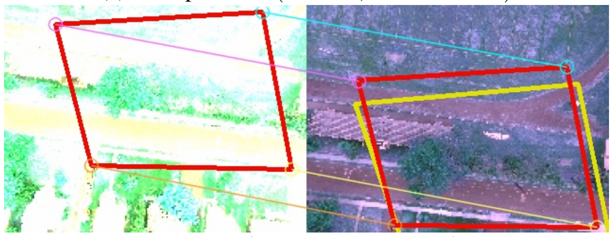
This section analyzes the performance profile of each method on a representative dataset of aerial imagery captured by a UAV. In addition to accuracy performance, an equally important consideration for real world application is inference speed. As a result, we also discuss the performance to speed tradeoffs of each method.

Our aerial dataset contains 350 image pairs resized to 240×320, captured by a DJI Phantom 3 Pro platform in Yardley, Pensylvania, USA in 2017. We divided it into 300 train and 50 test samples. We did not label the train set, but for evaluation purposes, we manually labeled the ground truth by picking 4 pairs of correspondences for each test sample. We also randomly inject illumination noise in both the training and testing sets. The evaluation metrics are the same for the synthetic data. To reduce training time, we finetune the neural networks on the aerial image data. Our unsupervised algorithm can directly use the aerial dataset image pairs. However, since we do not have ground truth homography labels, we have to perform a similar synthetic data generation process as in the synthetic dataset in order to finetune the supervised neural network. We fine tune both models over $150,000$ iterations for roughly 15 hours with data augmentation.

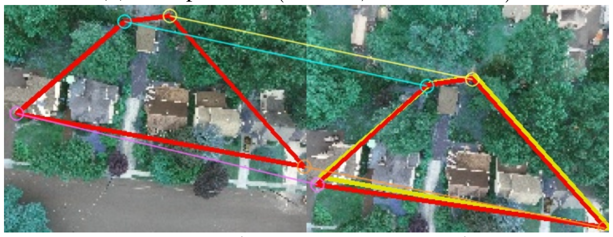
Fig. 6 displays the performance profile for the Unsupervised, Supervised, SIFT, and ECC methods. Fig 4 displays the speed and performance tradeoff for these methods, and additionally the featured based method ORB. The feature-based methods are tested on a 16-core Intel Xeon CPU, and the deep learning methods are tested on the same CPU and an NVIDIA Titan X GPU. The closer to the lower left hand corner, the better the performance and faster the runtime.

Both Figs. 6 and 4 demonstrate that our unsupervised algorithm has the best performance of all methods. In addition, Fig. 4 also shows that our unsupervised method on the GPU has both the best performance and the fastest inference times. SIFT has the second best performance after our unsupervised algorithm, but has a much slower runtime (approximately 200 times slower). ORB has a faster runtime than SIFT, but at the expense of poorer performance. The ECC direct method approach has the worst performance and runtime of all the methods. A qualitative example where both SIFT and ECC fail to deliver a good result while our method succeeds is illustrated in Fig. 5.

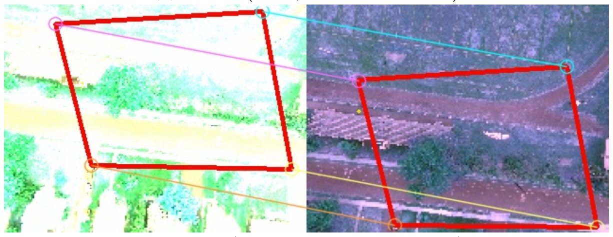
(a) Unsupervised $(Success, RMSE = 15.6)$



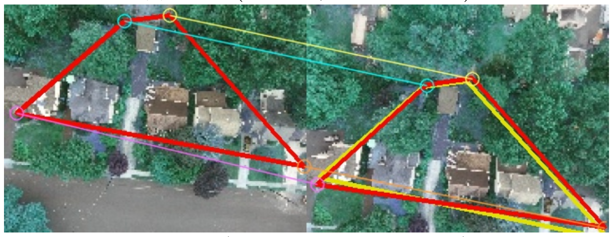
(b) Unsupervised $(Success, RMSE = 4.50)$



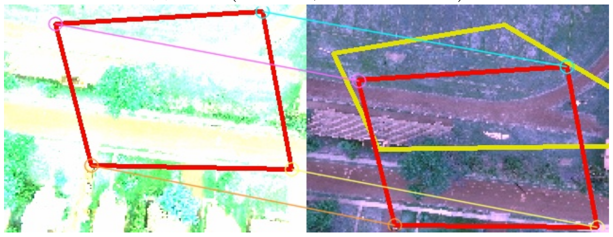
(c) SIFT $(Fail, RMSE = 105.2)$



(d) SIFT $(Success, RMSE = 6.06)$



(e) ECC$(Success, RMSE = 66.4)$



(f) ECC$(Success, RMSE = 48.10)$

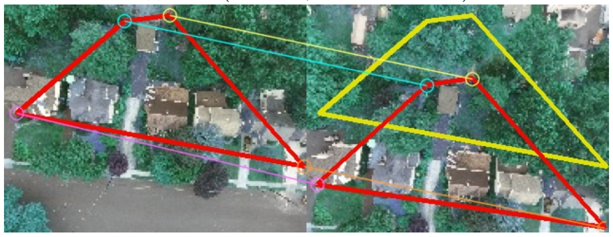


Fig. 5: Qualitative visualization of estimation methods on aerial dataset. Left: hard case, right: moderate case. ECC performs better than SIFT in the case of small displacement, but performs worse than SIFT in case of large displacement. Unsupervised network outperforms both SIFT and ECC approaches. Supervised network is omitted due to limited space and its poor performance on this dataset.

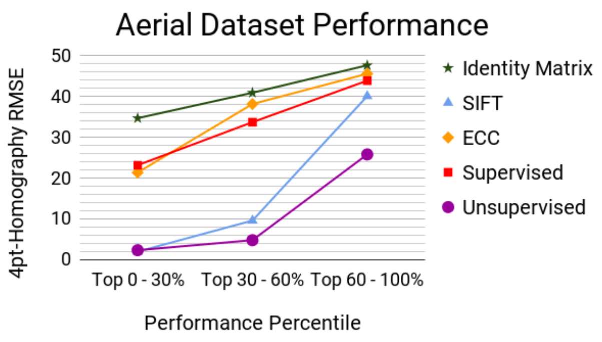


Fig. 6: 4pt-homography RMSE on aerial images (lower is better). Unsupervised outperforms other approaches significantly.

One of the most interesting results is that while the supervised and unsupervised approaches performed comparably on the synthetic data, the supervised approach had drastically poorer performance on the aerial image dataset. This shift is due to the fact that ground truth labels are not available for our aerial dataset. The generalization gap from synthetic (train) to real (test) data is an important problem in machine learning. The best practical approach is to additionally fine-tune the model on the new distribution of data. In a robotic field experiment, this can be achieved by flying the UAV to collect a few sample images and fine-tuning on those images. However, this fine-tuning is only possible with our unsupervised algorithm. Our aerial dataset results highlight the fact that even though synthetic data can be generated from real images, a pair of synthetic images is still very different from a pair of real images. These results demonstrate that the independence of our unsupervised algorithm from expensive ground truth labels has large practical implications for real-world performance.

VI. Conclusions

We have introduced an unsupervised algorithm that trains a deep neural network to estimate planar homographies. Our approach outperforms the corresponding supervised network on both synthetic and real-world datasets, demonstrating the superiority of unsupervised learning in image warping problems. Our approach achieves faster inference speed, while maintaining comparable or better accuracy than feature-based and direct methods. We demonstrate that the unsupervised approach is able to handle large displacements and large illumination variations that are typically challenging for direct approaches that use the same photometric loss function. The speed and adaptive nature of our algorithm makes it especially useful in aerial multi-robot applications that can exploit parallel computation.

In this work, we do not investigate robustness against occlusion, leaving it as future work. However, as suggested in [24], we could potentially address this issue by using data augmentation techniques such as artificially inserting random occluding shapes into the training images. Another direction for future work is investigating different improvements to achieve sub-pixel accuracy in the top 30% performance percentile.

Finally, our approach is easily scalable to more general warping motions. Our findings provide additional evidence for applying deep learning methods, specifically unsupervised learning, to various robotic perception problems such as stereo depth estimation, or visual odometry. Our insights on estimating homographies with unsupervised deep neural network approaches provide an initial step in a structured progression of applying these methods to larger problems.

VII. Acknowledgements

We gratefully acknowledge the support of ARL grants W911NF-08-2-0004 and W911NF-10-2-0016, ARO grant W911NF-13-1-0350, N00014-14-1-0510, N00014-09-1-1051, N00014-11-1-0725, N00014-15-1-2115 and N00014-09-1-103, DARPA grants HR001151626/HR0011516850 USDA grant 2015-67021-23857 NSF grants IIS-1138847, IIS-1426840 CNS-1446592 CNS-1521617 and IIS-1328805, Qualcomm Research, United Technologies, and TerraSwarm, one of six centers of STARnet, a Semiconductor Research Corporation program sponsored by MARCO and DARPA. We would also like to thank Aerial Applications for the UAV data set.

References

[1] M. Brown, D. G. Lowe, *et al.*, "Recognising panoramas." in *ICCV*, vol. 3, 2003, p. 1218.

[2] M. Shridhar and K.-Y. Neo, "Monocular slam for real-time applications on mobile platforms," 2015.

[3] Z. Zhang and A. R. Hanson, "3d reconstruction based on homography mapping," *Proc. ARPA96*, pp. 1007–1012, 1996.

[4] Z. Pan, X. Fang, J. Shi, and D. Xu, "Easy tour: a new image-based virtual tour system," in *Proceedings of the 2004 ACM SIGGRAPH international conference on Virtual Reality continuum and its applications in industry*. ACM, 2004, pp. 467–471.

[5] C.-Y. Tang, Y.-L. Wu, P.-C. Hu, H.-C. Lin, and W.-C. Chen, "Self-calibration for metric 3d reconstruction using homography." in *MVA*, 2007, pp. 86–89.

[6] D. Capel, "Image mosaicing," in *Image Mosaicing and Super-resolution*. Springer, 2004, pp. 47–79.

[7] R. Szeliski, "Image alignment and stitching: A tutorial," *Foundations and Trends® in Computer Graphics and Vision*, vol. 2, no. 1, pp. 1–104, 2006.

[8] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in *Proceedings of the 7th International Joint Conference on Artificial Intelligence - Volume 2*, ser. IJCAI'81. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1981, pp. 674–679.

[9] S. Baker and I. Matthews, "Lucas-kanade 20 years on: A unifying framework," *International journal of computer vision*, vol. 56, no. 3, pp. 221–255, 2004.

[10] G. D. Evangelidis and E. Z. Psarakis, "Parametric image alignment using enhanced correlation coefficient maximization," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 10, pp. 1858–1865, 2008.

[11] Q. Yan, Y. Xu, X. Yang, and T. Nguyen, "Heask: Robust homography estimation based on appearance similarity and keypoint correspondences," *Pattern Recognition*, vol. 47, no. 1, pp. 368–387, 2014.

[12] S. Lucey, R. Navarathna, A. B. Ashraf, and S. Sridharan, "Fourier lucas-kanade algorithm," *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 6, pp. 1383–1396, 2013.

[13] E. Muñoz, P. Márquez-Neila, and L. Baumela, "Rationalizing efficient compositional image alignment," *International Journal of Computer Vision*, vol. 112, no. 3, pp. 354–372, 2015.

[14] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.

[15] M. A. Fischler and R. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Commun. ACM*, vol. 24, no. 6, pp. 381–395, June 1981.

[16] F.-l. Wu and X.-y. Fang, "An improved ransac homography algorithm for feature based image mosaic," in *Proceedings of the 7th WSEAS International Conference on Signal Processing, Computational Geometry & Artificial Vision*. World Scientific and Engineering Academy and Society (WSEAS), 2007, pp. 202–207.

[17] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "Orb: An efficient alternative to sift or surf," in *Computer Vision (ICCV), 2011 IEEE international conference on*. IEEE, 2011, pp. 2564–2571.

[18] P. Weinzaepfel, J. Revaud, Z. Harchaoui, and C. Schmid, "Deepflow: Large displacement optical flow with deep matching," in *Proceedings of the IEEE International Conference on Computer Vision*, 2013, pp. 1385–1392.

[19] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox, "Flownet 2.0: Evolution of optical flow estimation with deep networks," *arXiv preprint arXiv:1612.01925*, 2016.

[20] P. Fischer, A. Dosovitskiy, E. Ilg, P. Häusser, C. Hazırbaş, V. Golkov, P. van der Smagt, D. Cremers, and T. Brox, "Flownet: Learning optical flow with convolutional networks," *arXiv preprint arXiv:1504.06852*, 2015.

[21] J. Revaud, P. Weinzaepfel, Z. Harchaoui, and C. Schmid, "Deepmatching: Hierarchical deformable dense matching," *International Journal of Computer Vision*, vol. 120, no. 3, pp. 300–323, 2016.

[22] H. Altwaijry, A. Veit, S. J. Belongie, and C. Tech, "Learning to detect and match keypoints with deep architectures." in *BMVC*, 2016.

[23] D. Eigen, C. Puhrsch, and R. Fergus, "Depth map prediction from a single image using a multi-scale deep network," in *Advances in neural information processing systems*, 2014, pp. 2366–2374.

[24] D. DeTone, T. Malisiewicz, and A. Rabinovich, "Deep image homography estimation," *arXiv preprint arXiv:1606.03798*, 2016.

[25] T. Zhou, M. Brown, N. Snavely, and D. G. Lowe, "Unsupervised learning of depth and ego-motion from video," *arXiv preprint arXiv:1704.07813*, 2017.

[26] S. Baker, A. Datta, and T. Kanade, "Parameterizing homographies," *Technical Report CMU-RI-TR-06-11*, 2006.

[27] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[28] H. Zhao, O. Gallo, I. Frosio, and J. Kautz, "Is l2 a good loss function for neural networks for image processing?" *ArXiv e-prints*, vol. 1511, 2015.

[29] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, "Squeezenet: Alexnet-level accuracy with 50x fewer parameters and¡ 0.5 mb model size," *arXiv preprint arXiv:1602.07360*, 2016.

[30] R. I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed. Cambridge University Press, ISBN: 0521540518, 2004.

[31] G. H. Golub and C. Reinsch, "Singular value decomposition and least squares solutions," *Numerische mathematik*, vol. 14, no. 5, pp. 403–420, 1970.

[32] T. Papadopoulo and M. I. Lourakis, "Estimating the jacobian of the singular value decomposition: Theory and applications," in *European Conference on Computer Vision*. Springer, 2000, pp. 554–570.

[33] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[34] M. Jaderberg, K. Simonyan, A. Zisserman, *et al.*, "Spatial transformer networks," in *Advances in Neural Information Processing Systems*, 2015, pp. 2017–2025.

[35] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, *et al.*, "Tensorflow: Large-scale machine learning on heterogeneous distributed systems," *arXiv preprint arXiv:1603.04467*, 2016.

[36] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.