Deep Homography Estimation

Reported by

Paruchuri Ratan Chowdary

20011581

Outline:

1. Introduction:

Traditional homography estimation relies on feature matching and techniques like RANSAC to compute a transformation matrix mapping points between images. While effective in many cases, traditional methods have limitations, particularly in handling challenges like low-texture regions, occlusions, and significant viewpoint changes. They often require manual parameter tuning and may struggle with variations in lighting conditions.

In contrast, deep homography estimation, facilitated by neural networks, offers a more robust and adaptable approach. The end-to-end learning paradigm allows the model to automatically learn hierarchical features, making it more resilient to variations in scale, rotation, and viewpoint changes. Deep homography models excel in capturing complex patterns and handling occlusions, providing adaptability to diverse datasets without the need for extensive manual intervention. The report discusses variants of deep homography estimation, indicating modifications or different architectures that enhance the model's performance in specific scenarios or datasets. These variants showcase the flexibility of deep learning in addressing challenges associated with traditional homography methods.

2. Traditional Homography:

Feature Detection: Use a feature detector (e.g., SIFT, SURF, ORB) to identify distinctive key points in both images.

Homography Computation: Use the matched points to compute the homography matrix. The most common method is to use a least-squares algorithm. The homography matrix (H) is a 3x3 matrix that transforms points from the source image to the destination image.

The formula for the homography matrix (H) is:

$$s \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = H \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Here, (x, y) are the coordinates of a point in the source image, and (x', y') are the coordinates of the corresponding point in the destination image. 's' is a scaling factor.

Verification and Refinement: Use RANSAC (Random Sample Consensus), to robustly estimate the homography by iteratively fitting the model to a subset of correspondences.

3. Variants of Deep Homography Estimation:

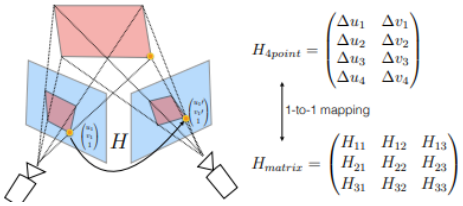
3.1 Deep Image Homography Estimation

On June 13, 2016, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich introduced a groundbreaking advancement in computer vision with the first-ever deep image homography estimation. Departing from traditional methods, their approach harnessed the power of convolutional neural networks (CNNs) to directly predict homography matrices between pairs of images. This novel technique eliminated the need for explicit feature matching, allowing the network to autonomously learn complex relationships between corresponding points during training.

The simplest way to parameterize a homography is with a 3x3 matrix and a fixed scale. However, if we unroll the 8 (or 9) parameters of the homography into a single vector, we'll quickly realize that we are mixing both rotational and translational terms. They found an alternate parameterization, one based on a single kind of location variable, namely the corner location, is more suitable for our deep homography estimation task. The 4-point parameterization has been used in traditional homography estimation methods, and they used it in modern deep manifestation of the homography estimation problem. Letting $\Delta u1 = u1^{1} - u1$ be the u-offset for the first corner, the 4-point parameterization represents a homography as follows:

$$H_{4point} = \begin{pmatrix} \Delta u_1 & \Delta v_1 \\ \Delta u_2 & \Delta v_2 \\ \Delta u_3 & \Delta v_3 \\ \Delta u_4 & \Delta v_4 \end{pmatrix}$$

Once the displacement of the four corners is known, one can easily convert H_{4point} to H matrix. This can be accomplished in several ways, for example one can use the normalized Direct Linear Transform (DLT) algorithm.

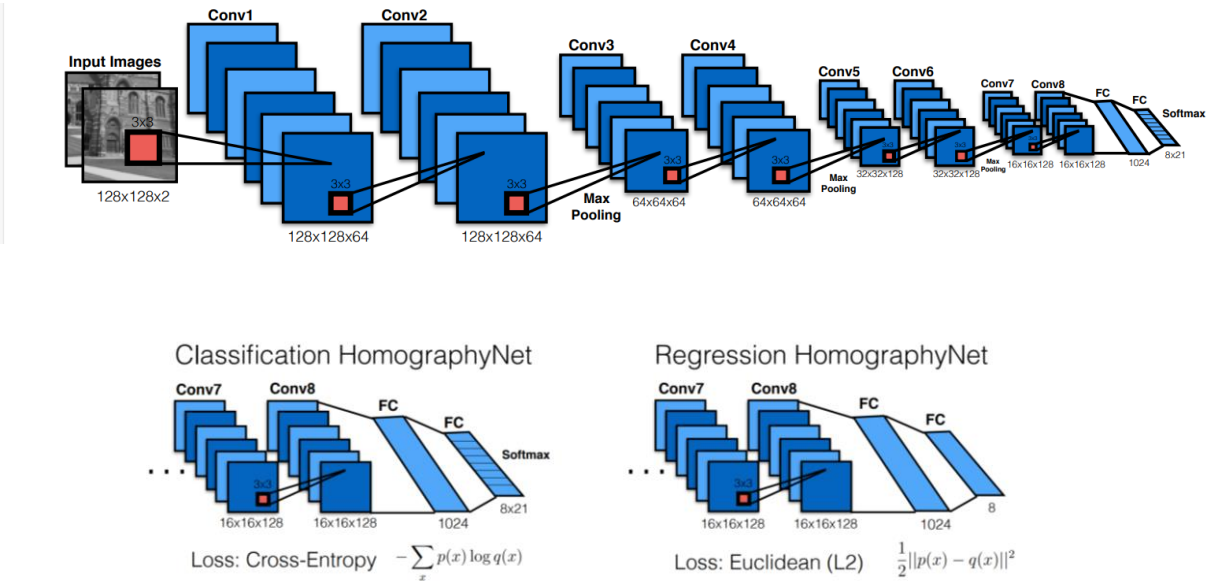


3.1.1 Convolution Neural Network Architecture:

The described neural network architecture is designed for both regression and classification tasks. The architecture closely resembles Oxford's VGG Net, employing 3x3 convolutional blocks with Batch Normalization and Rectified Linear Unit (ReLU) activations. Consisting of eight convolutional layers, a max-pooling layer with a 2x2 kernel and stride 2 follows every two convolutional layers. Following the convolutional layers are two fully connected layers, with the

first having 1024 units. A dropout layer with a probability of 0.5 is applied after the final convolutional layer and the first fully connected layer. For the **regression network**, the model directly produces eight real-valued numbers, representing the homography matrix parameters, and utilizes the Euclidean (L2) loss as the final layer during training. This architecture leverages the depth and hierarchical feature learning characteristics of VGG Net for effective homography estimation.

In the **classification network**, a quantization scheme is employed, and the final layer includes a softmax activation. During training, the cross-entropy loss function is used. Despite inherent quantization errors, the network is capable of providing confidence scores for each corner produced by the method. Specifically, 21 quantization bins are chosen for each of the eight output dimensions, resulting in a final layer with 168 output neurons. This allows the network not only to classify corners but also to provide a measure of certainty or confidence in its predictions.

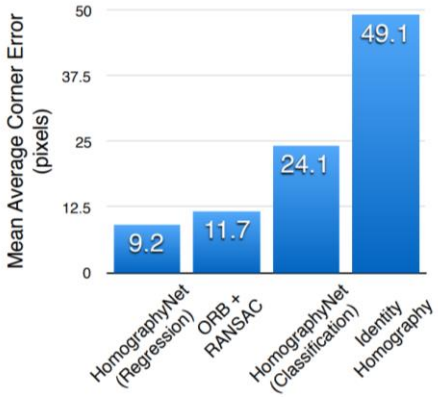


3.1.2 Data Generation for Deep Homography Estimation:

The authors introduced a method for generating an extensive dataset of training triplets (I_A, I_B, H^{AB}) from existing real image datasets like MS-COCO. The process begins by randomly cropping a square patch I_p, from a larger image I at position p, ensuring avoidance of image borders to prevent artifacts. Subsequently, the four corners of Patch A undergo random perturbations within the range [-ρ, ρ]. These perturbed corners define a homography H^{AB}. The inverse of this homography, $H^{BA} = (H^{AB})^{-1}$, is then applied to the original image I, resulting in a new image, I'. Another square patch, I'_p, is randomly cropped from I at position p. The grayscale representations of the two patches, I_p and I'_p, are stacked channel-wise to form a 2-channel image, which serves as input to their ConvNet. The 4-point parameterization of H^{AB} is utilized as the associated ground-truth training label, enabling the network to learn the homography transformation from the generated triplets effectively. This data generation pipeline allows for

the creation of a diverse and augmented dataset for training robust homography estimation models.

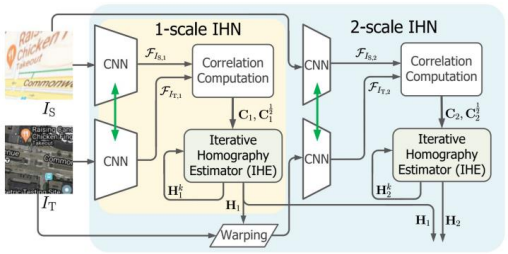
The metric used for evaluating test data is Mean Average Corner Error. To measure this metric, one first computes the L2 distance between the ground truth corner position and the estimated corner position. The error is averaged over the four corners of the image, and the mean is computed over the entire test set. Regression network outperformed traditional homography architectures such as ORB+RANSAC and classification network. The classification network can produce confidences and thus a meaningful way to visually debug the results.



3.2 Iterative Deep Homography Estimation

Homography estimation can be performed through two main approaches: Feature-based and Photometric-based. Photometric-based methods focus on estimating homography from pixel intensities. The Lucas-Kanade (LK) algorithm is a widely adopted photometric-based approach, iteratively estimating the residual homography using a pre-computed iterator. Subsequently, inspired by the LK algorithm, numerous recent methods have emerged, aiming to enhance estimation accuracy by cascading multiple VGG-style networks. This cascading is essentially a form of iteration that significantly improves the performance of homography estimation. However, it's worth noting that this iterative approach has its limitations, as it is confined to a fixed number of cascades, and increasing the number of network cascades does not necessarily guarantee improved performance.

The Iterative Deep Homography Network (IHN) represents a novel deep homography estimation architecture introduced in this work. In contrast to methods that achieve iterative refinement through network cascading or untrainable IC-LK iterator, IHN's iterator features tied weights and is fully trainable. The proposed IHN comes in two versions: (1) IHN for static scenes and (2) IHN-mov tailored for dynamic scenes incorporating moving objects. Both versions offer the flexibility to be configured in either 1-scale for efficiency or 2-scale for enhanced accuracy. The introduction of IHN-mov, which involves generating an inlier mask, serves to further enhance estimation accuracy, particularly in scenes with moving objects



(a) Iterative Homography Network (IHN)

3.2.1 Feature Extraction:

In the proposed Iterative Homography Network (IHN), feature maps of the source and target images are extracted using a Siamese CNN architecture. The building block of this architecture is a combination of 1 max-pooling layer with a stride of 2 and 2 residual blocks. The image data undergoes initial processing through a convolutional block with a 7x7 kernel. Subsequently, q basic units, each consisting of the described combination, are added to generate feature maps at a resolution of $1/(2^q) \times 1/(2^q)$. These feature maps are then reprojected using a linear convolutional layer with a 1x1 kernel. For the 1-scale IHN, the feature maps at 1/4 × 1/4 resolution are utilized, while the 2-scale IHN employs both 1/4 × 1/4 and 1/2 × 1/2 resolution feature maps.

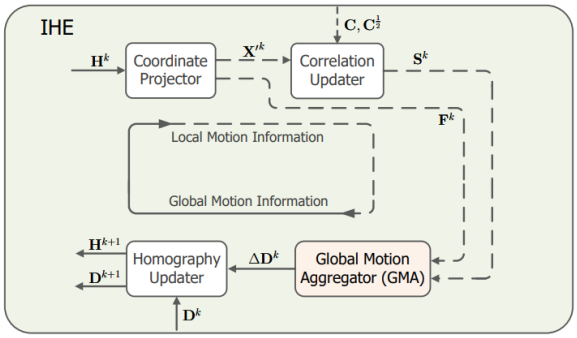
3.2.2 Correlation Computation:

In the proposed approach, the iterative refinement process involves the explicit computation of pairwise correlation to refine the homography estimation. The correlation is computed in the form of a correlation volume, determined by the equation below, where x_S and x_T represent the coordinate positions of the source and target feature maps:

$$\mathbf{C}(\mathbf{x}_\mathrm{S}, \mathbf{x}_\mathrm{T}) = \mathrm{ReLU}(\mathcal{F}_{I_\mathrm{S}}(\mathbf{x}_\mathrm{S})^\mathsf{T} \mathcal{F}_{I_\mathrm{T}}(\mathbf{x}_\mathrm{T})),$$

Here, the inputs F_{IS} and F_{IT} denote the source and target feature maps, respectively. The correlation volume C is a representation of the relationships between corresponding points in the feature maps. During each iteration, a fixed search window is sampled from the correlation volume by the correlation updater. Alternatively, the correlation volume can be computed on demand during the iteration, providing a potential reduction in space complexity.

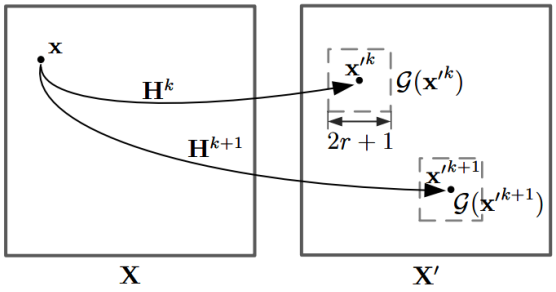
3.2.3 Iterative Homography Estimator:



(b) Iterative Homography Estimator (IHE)

The iterative homography estimator (IHE) is crafted with inspiration drawn from the IC-LK iterator. This component of the proposed framework takes the correlation volume as input and produces the estimated homography matrix, denoted as H. The design of IHE involves several stages, including the coordinate projector and the global motion aggregator. In the coordinate projector, local motion information is aggregated, and this process extends to the global motion aggregator, where local information is synthesized into a global homography estimation. The cycle continues as the global motion aggregator feeds back to the coordinate projector in the next iteration, where the global homography estimation is converted back to local coordinates for local information update.

3.2.4 Coordinate Projector and Correlation Updater:

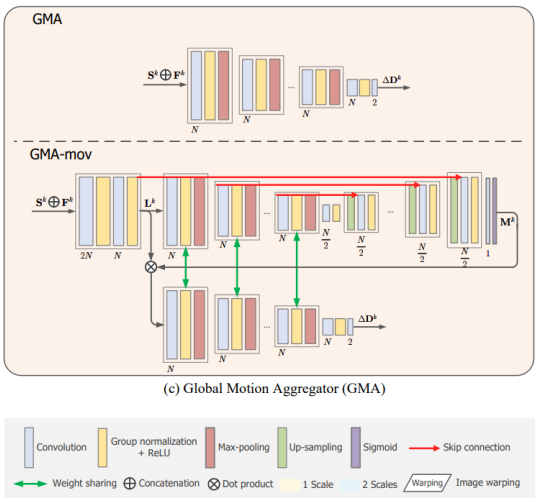


In the context of iteration k, the point-wise correspondence between the source feature map F_{IS} and the target feature map F_{IT} is established through the present homography matrix H^k. Let X denote the meshgrid coordinate set of F_{IS} and its corresponding set in F_{IT} is denoted as X'. For each coordinate position, denoted as x=(u,v) in X and x′=(u′,v′) in X′, the point-wise correspondence of X and X'^k is projected by H^k. To facilitate the learning of local motion information, the homography flow F^k, defined as $F_k=(X')^k-X$, is computed. Subsequently, the computed X'^k and the correlation volume are passed to the correlation updater. The correlation updater operates by sampling the correlation volume C using homography-projected coordinates X'^k and produces an updated correlation slice S^k

$$\mathbf{S}^k(\mathbf{x}) = \mathbf{C}(\mathbf{x}, \mathcal{G}_r(\mathbf{x}'^k)),$$

Where $G_r(x'^k)$) denotes a local square grid of fixed search radius r.

3.2.5 Global Motion Aggregator:



(c) Global Motion Aggregator (GMA)

The residual homography in iteration k is estimated by the global motion aggregator (GMA), with the homography matrix parameterized by the displacement vectors of the four corner points. In this process, GMA takes the concatenated correlation slice S^k and homography flow F^k as input. The GMA is structured with multiple basic units, each consisting of a 3x3 convolutional block, 1 group normalization + ReLU, and 1 max-pooling layer with a stride of 2. These basic units are continually added until the spatial resolution of the feature map reaches 2x2. Subsequently, a convolutional block projects the feature map into a 2x2x2 cube ΔD^k, representing the estimated residual displacement vectors of the four corner points.

For scenes with moving objects, the GMA-mov variant is introduced. GMA-mov explicitly produces a mask M^k to weigh the matching inliers based on the homography transform assumption. It encodes point-wise local motion information into the feature map L^k using a single convolutional block. This L^k is then processed through multiple basic units, similar to GMA, to preliminarily extract the N×2×2 feature map containing global motion information. Unlike GMA, this global motion information is not directly used for residual homography estimation but rather for inlier mask prediction. The latter part of GMA-mov involves progressively up sampling the feature map with global motion information and combining it with local motion information through skip connections. The inlier mask M^k with the same size as L^k is predicted using a sigmoid function, and the dot product of M^k and L^k is sent into a structure similar to GMA to produce the final residual homography estimation. This approach allows GMA-mov to explicitly account for local motion information in scenes with moving objects, enhancing the accuracy of homography estimation.

3.2.6 Homography Updater:

This cube D captures the movement of the four corner points in an image. In each iteration (denoted as k), the displacement cube is updated as $D^{k+1} = D^k + \Delta D^k$.The updated displacement cube D^{k+1} is then used to calculate the homography matrix H^{k+1} using methods like the least square method or the direct linear transform. This updated homography matrix is then feedback into the coordinate projector in the next iteration. The process begins with an initial displacement cube D^0 set to zero, signifying an identical transformation.

3.2.7 Experimentation and Evaluation:

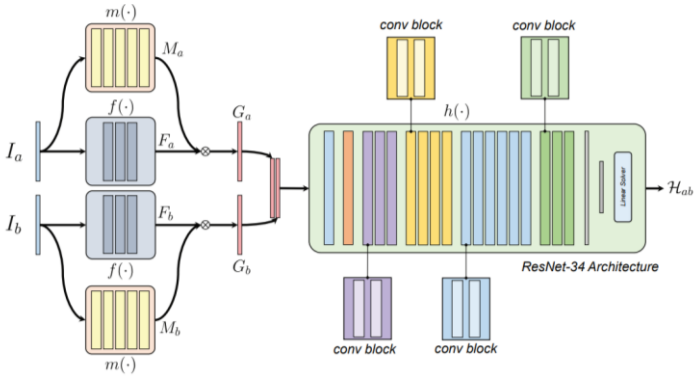
The performance of Iterative Homography Network (IHN) is assessed on various datasets encompassing both common and challenging scenarios. For static scenes, IHN is evaluated on the widely used MSCOCO dataset, demonstrating its effectiveness in handling common scenarios. Additionally, assessments are conducted on cross-modal datasets involving Google Earth and Google Map & Satellite imagery, showcasing the model's versatility across diverse datasets.

To address the challenges posed by dynamic scenes with moving objects, both IHN and its specialized variant, IHN-mov, are tested on a demanding scene generated from the SPID surveillance dataset. Dynamic scenes present complexities as foreground moving objects often occlude the background, violating the homography assumption and making accurate homography estimation challenging. In response to this scenario, IHN-mov is specifically designed to enhance homography estimation in the presence of occluding moving objects. Both IHN and IHN-mov outperformed other architectures with cascading and scaling.

3.3 Content-Aware Unsupervised Deep Homography Estimation

The process of estimating homography traditionally relies on matched image feature points like SIFT. This involves obtaining a set of feature correspondences and using methods like Direct Linear Transformation (DLT) coupled with RANSAC outlier rejection to calculate a homography matrix. However, accuracy can be compromised due to limited matched points or uneven feature distribution caused by factors like texture less regions, repetitive patterns, or lighting variations. Removing outlier points, such as those from non-dominant planes or dynamic objects, is crucial for better results. This makes feature-based homography estimation challenging for non-standard scenes.

To address these challenges, the authors propose an unsupervised solution utilizing a novel architecture equipped with content-aware learning. This approach is tailored for image pairs with a small baseline, a scenario commonly encountered in consecutive video frames, burst image sequences, or photos taken using dual-camera cellphones. The aim is to enhance homography estimation accuracy in situations where traditional methods struggle due to specific scene characteristics or setup configurations.



3.3.1 Network Structure:

The network is built upon convolutional neural networks. It takes two grayscale image patches Ia and Ib as input and produces a homography matrix Hab from I_a to I_b as output.

Feature Extractor:

The network is designed to autonomously learn intricate features from the input data to ensure resilient feature alignment. This is achieved by constructing a fully convolutional network (FCN) that operates on the input and generates a comprehensive feature map. When given inputs I_a and I_b, the feature extractor employs shared weights, processing both inputs to generate corresponding feature maps, denoted as F_a and F_b. This shared weight mechanism allows the network to extract and represent essential features from both inputs, facilitating alignment and comparison between the features extracted from these distinct input sources.

$$F_\beta = f(I_\beta), \quad \beta \in \{a, b\}$$

Mask Predictor:

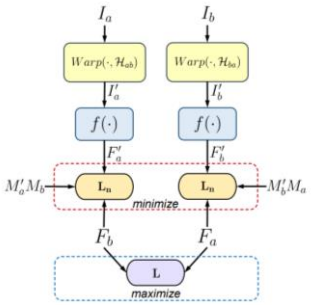
The construction involves creating a dedicated sub-network tasked with autonomously learning the positions of inliers within the data. This sub-network, denoted as m(·), is designed to generate an inlier probability map or mask. This map specifically highlights the content within the feature maps that significantly contribute to the homography estimation. With these generated masks, the features extracted by the primary feature extractor undergo further weighting. This process involves applying the masks to these features before they are fed into the homography estimator. Consequently, two new weighted feature maps, G_a and G_b, are obtained.

The masks serve a dual purpose within the network architecture. Firstly, they function as attention maps, directing focus towards the most relevant content in the feature maps. Secondly, they act as outlier rejecters, aiding in the identification and removal of elements that could potentially compromise the homography estimation process.

$$M_\beta = m(I_\beta), \quad G_\beta = F_\beta M_\beta, \quad \beta \in \{a, b\}$$

Homography estimator:
This process involves concatenating weighted feature maps, which is then passed through a ResNet-34-based homography estimator network. The network, comprising 34 layers of strided convolutions followed by global average pooling, generates fixed-size feature vectors regardless of input dimensions. From this, four 2D offset vectors (8 values total) are extracted to represent the transformation needed for alignment. These vectors allow straightforward derivation of an 8-degree-of-freedom homography matrix (H_{ab}) by solving a linear system.

3.3.2 Triplet Loss for Robust Homography Estimation:



With the homography matrix H_{ab} estimated, authors warp image Ia to I'_a and then further extract its feature map as F'_a . Intuitively, if the homography matrix H_{ab} is accurate enough, F'_a should be well aligned with F_b, causing a low loss between them.

$$\mathbf{L_n}(I'_a, I_b) = \frac{\sum_i M'_a M_b \cdot ||F'_a - F_b||_1}{\sum_i M'_a M_b}$$

The fact that the original images I_a and I_b are mis-aligned is encoded as below equation,

$$\mathbf{L}(I_a, I_b) = ||F_a - F_b||_1$$

They swap the features of Ia and Ib and produce another homography matrix H_{ba}. Also add a constraint that enforces H_{ab} and H_{ba} to be inverse.

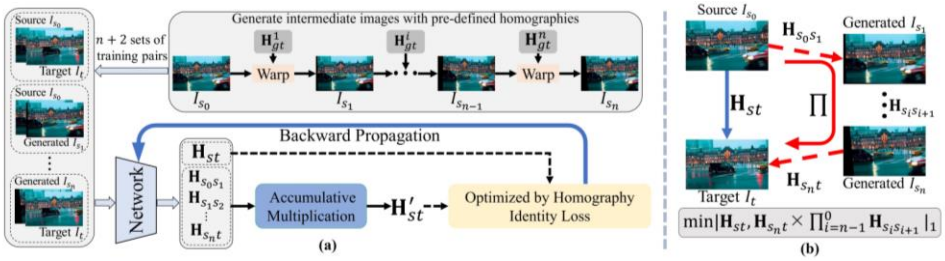
$$\min_{m,f,h} \mathbf{L_n}(I'_a, I_b) + \mathbf{L_n}(I'_b, I_a) - \lambda \mathbf{L}(I_a, I_b) + \mu ||\mathcal{H}_{ab}\mathcal{H}_{ba} - \mathcal{I}||_2^2$$

3.3.3 Evaluation
The proposed dataset for comprehensive homography evaluation is a significant contribution, addressing the absence of dedicated datasets in this domain. Covering five distinct categories—Regular, Low-Texture, Low-Light, Small-Foregrounds, and Large-Foregrounds—the dataset encapsulates a wide array of real-world challenges encountered in image alignment. This diversity provides a robust platform for evaluating and benchmarking homography estimation algorithms across scenarios with varying texture, lighting conditions, and foreground sizes. The content aware unsupervised neural network proposed by authors outperforms all other architectures both quantitatively and qualitatively.

3.4. Semi-supervised Deep Large-baseline Homography Estimation with Progressive Equivalence Constraint

The supervised approaches in homography estimation rely on synthetic image pairs for training due to the scarcity of real-world image pairs with ground truth labels. However, these methods often lack realistic scene parallax, limiting their ability to generalize effectively. On the other hand, unsupervised methods adopt label-free training strategies aiming to minimize the photometric distance between warped source images and target images. While they perform well in small-baseline scenes with low non-overlap rates (below 10%), their efficacy diminishes in large-baseline scenarios (20% to 50% non-overlap rates) due to substantial appearance and viewpoint changes, leading to numerous out-of-boundary pixels in the warped images. Our proposed method addresses both scenarios.

The authors introduce a progressive estimation strategy that tackles large-baseline homography estimation by breaking down the large-baseline into several intermediate ones, enabling a more manageable estimation process. Additionally, they propose a semi-supervised homography identity loss that guides the network to prioritize optimizing the homography itself. To facilitate robust learning, they introduce a comprehensive large-scale dataset comprising diverse scenes specifically tailored for large baseline homography learning.



The introduced progressive strategy for addressing large baseline homography estimation involves breaking down the substantial baseline into several intermediate stages by introducing intermediate images. These intermediate images (I_{si}) are generated using a predefined set of homographies. To circumvent the pitfalls of accumulating degenerate solutions from intermediate homographies, non-identity matrices are randomly sampled from the predefined set. This process results in the creation of n intermediate images, generating n + 2 sets of image pairs: (I_{si}, I_{si+1}), (I_{sn}, I_t), and (I_s, I_t).

The primary objective is to train a neural network, denoted as f_θ with parameters θ, which predicts the homography matrix relating each intermediate image (I_{si}) to its subsequent counterpart (I_{si+1}), the last intermediate image (I_{sn}) to the target image (I_t), and the initial image (I_s) to the target image (I_t), respectively. The crucial aspect is that the multiplication of all intermediate homographies should yield an equivalence to the overall homography (H_{st}) relating the initial image to the target image.

3.4.1 Homography Identity Loss:

The correspondence between each set of image pairs can be established via the mapping provided by the corresponding homography matrix. Denoting X_{si} as the meshgrid coordinate set of I_{si} and X_t as that of It, the point-wise correspondence between X_{si} and its subsequent counterpart X_{si+1} is determined by the intermediate homography $H_{si,si+1}$ using the equation $X_{si+1} = H_{si,si+1}X_{si}$. The corresponding identity equation can be expressed as

$$\prod_{i=n-1}^{0} \mathbf{H}_{s_i s_{i+1}} = \mathbf{H}_{s_n t}^{-1} \times \mathbf{H}_{st}.$$

In our optimization process, the objective isn't to minimize the distance between the warped source image and the target image. Instead, our focus lies in minimizing the error between the estimated homographies based on Equation 1. This approach is pursued in an unsupervised manner, where the primary goal is to reduce the discrepancies between the estimated homographies across the sequence of intermediate images, ultimately ensuring a more accurate and consistent estimation of the transformation matrices without explicit ground truth supervision.

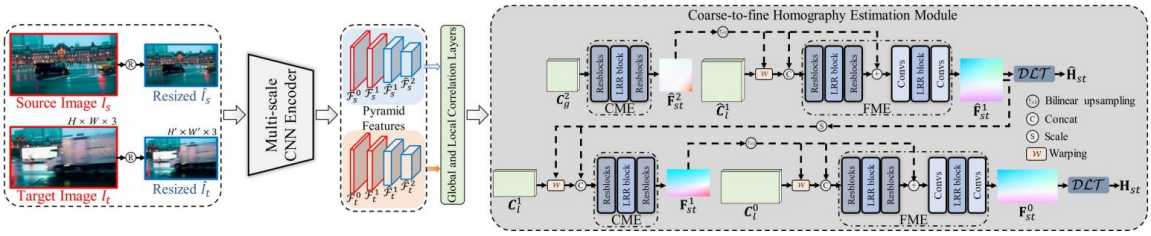
$$\mathcal{L}_{unsup} = \sum_{i=0}^{n-1} \lambda_i |(\mathbf{H}_{s_{i+1}t}^{-1} \times \mathbf{H}_{st}) - \prod_{j=i}^{0} \mathbf{H}_{gt}^{j+1}|_1,$$

Since the intermediate images are generated through the pre-defined homographies, so we can estimate the homographies of intermediate images in a supervised way. The supervised objective function is formulated as,

$$\mathcal{L}_{sup} = \sum_{i=0}^{n-1} |\mathbf{H}_{s_i s_{i+1}} - \mathbf{H}_{gt}^{i+1}|_1,$$

Our final semi-supervised homography identity loss L_{HIL} combines the L_{sup} and L_{unsup} as $L_{HIL} = L_{unsup} + \lambda_w L_{sup}$, where the λ_w is a weighting factor.

3.4.2 Multi-scale Homography Estimation Network:



There are two challenges for the large-baseline homography estimation: 1) the overlap rate between two images is low, and 2) the receptive field of CNN-based models is limited. To overcome these problems, they designed a multi-scale homography estimation network. It comprises four sequential residual blocks and integrated max-pooling layers. This specialized

architecture enables the extraction of multi-scale features from pairs of input images, denoted as I_s and I_t. The encoder structure is designed to capture information at varying resolutions. Specifically, higher-level pyramid layers encapsulate broader, global details, while lower-level ones focus on finer, more localized information. The topmost layer extracts the most comprehensive, global information, pivotal for generating a global correlation map. These pyramid layers also facilitate the estimation of homographies in a coarse-to-fine manner.

Feature Correlation:
The feature correlation process involves assessing the similarity between extracted features from the images. Among these features, $F^{\wedge 2}$ contains notably more global information than the others. Therefore, we utilize the global correlation layer to express the pairwise resemblance between spatial positions within the source feature $F^{\hat{}2}_s$ and the target feature $F^{\wedge 2}$. For other feature maps like F^0, F^1, and $F^{\hat{}1}$, a local correlation layer evaluates feature similarities, yielding C^0_l, C^1_l, and $C^{\hat{}1}_l$.

$$\mathbf{C}_g^2(\mathbf{x}_s, \mathbf{x}_t)) = \hat{\mathcal{F}}_s^2(\mathbf{x}_s)^\top \hat{\mathcal{F}}_t^2(\mathbf{x}_t),$$

Coarse-to-fine Homography Estimation Module:

The estimation of homographies, crucial for understanding the transformation between I_s and I_t, is facilitated through the utilization of feature correlations. This involves employing two coarse motion estimators (CME) and two fine motion estimators (FME) to extract both global and local motion information from the correlations. The relative motion between the images, represented by a homography, is computed as $F_{st} = X_t - X_s$ to aid in learning motion information. This information is further utilized to solve the Direct Linear Transform (DLT), resulting in a unique homography. The feature C^2_g is inputted into the CME to estimate the coarse homography flow $F^{\hat{}2}_{st}$. Subsequently, $F^{\hat{}1}_{st}$ is obtained by incorporating $F^{\hat{}2}_{st}$ and $C^{\hat{}1}_l$ as input through the FME. The process is iterated for generating homography flows F^1_{st} and F^0_{st} based on their respective features and correlations. Finally, we convert the flows into the corresponding homographies by solving the DLT.

3.4.3 Evaluation:
The authors have introduced a substantial dataset designed specifically for large baseline homography estimation, recognizing the absence of a dedicated dataset tailored for this specific task. This dataset encompasses five distinct categories: regular (RE-L), low-texture (LT-L), low-light (LL-L), small-foregrounds (SF-L), and large-foregrounds (LF-L) scenes.

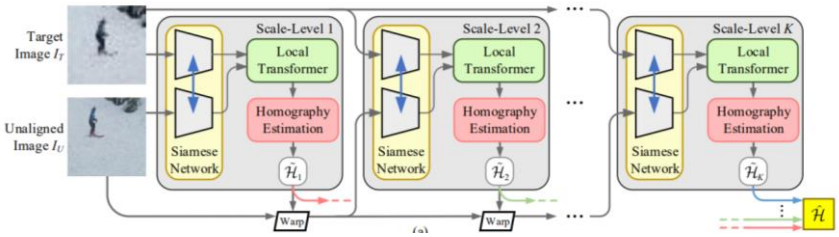
Carefully curated from real-world scenes, the image pairs within the dataset were selected to ensure a controlled average non-overlap rate ranging between 20% and 50%. For the evaluation phase, each image pair underwent a meticulous manual labeling process. Specifically, more than 6 uniformly distributed matching points were meticulously identified and labeled within each evaluation image pair. The semi supervised architecture that authors proposed outperformed other architectures in all categories of dataset.

3.5 LocalTrans: A Multiscale Local Transformer Network for Cross-Resolution Homography Estimation

The rapid evolution of multiscale gigapixel photography introduces an immersive visual experience by amalgamating numerous high-resolution local views into a single ultra-high-resolution image. However, this process creates a considerable resolution gap, termed cross-resolution, between the high and low-resolution views.

Addressing this challenge in traditional homography estimation, the authors introduce a novel solution named LocalTrans, a multiscale local transformer network. This innovative approach leverages transformer structures to explicitly capture correspondence between inputs. Their design incorporates a unique local transformer layer within a multiscale framework, capable of effectively capturing correspondence with both long and short-range attention. Experimental results indicate that this proposed structure outperforms global attention mechanisms commonly used in similar contexts. Moreover, the LocalTrans architecture demonstrates significantly improved processing speeds and reduced GPU memory requirements compared to standard transformer structures, offering an efficient solution for addressing cross-resolution challenges in homography estimation.

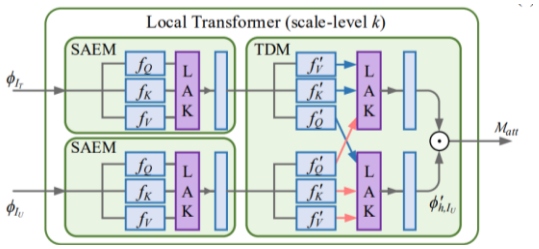
3.5.1 Multiscale Local Transformer structure:



The LocalTrans network begins by employing a deep siamese network, which entails an image encoder with shared weights, to extract features φ_{IT} and φ_{IU} from the target image I_T and the unaligned image I_U, respectively. This network architecture includes two convolutional layers followed by a 2×2 max-pooling layer, constituting a basic block. By adjusting the block number within the deep siamese network, features with varying scales are constructed in the multiscale structure, generating feature maps $\varphi^{(k)}_I$ of shape $H_k \times W_k$ in each scale level k. Uniquely diverging from existing deep homography methods that typically concatenate images or features, this approach explicitly establishes correspondences between the extracted features φ_{IT} and φ_{IU} using a transformer structure. Within each scale level, a homography estimation module is employed to estimate a homography matrix H_k based on the attention map and feature maps.

Subsequently, the unaligned image I_U is warped according to the estimated homography matrix H_k before being passed to the subsequent scale level for further processing and refinement.

3.5.2 Transformer Structure:



The inputs to the transformer consist of two features, $\varphi^{(k)}_{IT}$ and $\varphi^{(k)}_{IU}$, generated by the deep Siamese network. To leverage internal relations within the feature maps and capture correspondences across these features, two modules, the Self-Attention Encoder Module (SAEM) and the Transformer Decoder Module (TDM), are employed.

The SAEM initiates the process by applying three 1×1 convolution layers $f_Q(\varphi_I)$, $f_K(\varphi_I)$, and $f_V(\varphi_I)$—without activation functions to encode the input image feature into $\varphi_{Q,I}$, $\varphi_{K,I}$ and $\varphi_{V,I}$. By encoding the self-attention result $\varphi_{h,I}$, the network accentuates attention towards the edges with similar features to the center pixel, a characteristic more pronounced in attention maps at higher scale levels. The final high-level feature $\varphi_{s,I}$, is derived by encoding $\varphi_{h,I}$ using a 1×1 convolution layer.

$$\mathcal{M}_s = \sigma\left(\frac{\phi_{Q,I} \odot \phi_{K,I}}{\sqrt{C}}\right),$$
$$\phi_{h,I} = \mathcal{M}_s \otimes \phi_{V,I},$$

On the other hand, the TDM involves two iterations of cross-attention mechanisms. In the first iteration, $\varphi_{s,I}$ undergoes encoding to $\varphi'_{Q,I}$, $\varphi'_{K,I}$, and $\varphi'_{V,I}$, similar to the SAEM, yet distinct in conducting cross-attention between features from the target image I_T and the unaligned image I_U. The cross-attention mechanism produces cross-attention maps, $M_{IU\to IT}$ and $M_{IT\to IU}$, along with attention-aware features, $\varphi'_{h,IT}$ and $\varphi'_{h,IU}$.

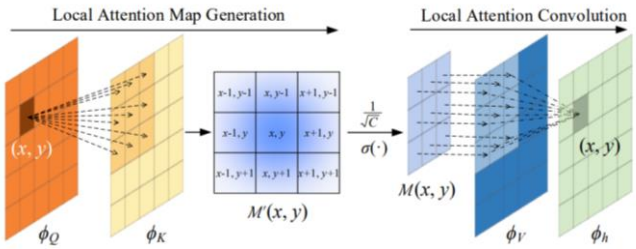
$$\mathcal{M}_{I_U\to I_T} = \sigma\left(\frac{\phi'_{Q,I_U} \odot \phi'_{K,I_T}}{\sqrt{C}}\right),$$
$$\phi'_{h,I_T} = \mathcal{M}_{I_U\to I_T} \otimes \phi'_{V,I_T},$$
$$\mathcal{M}_{I_T\to I_U} = \sigma\left(\frac{\phi'_{Q,I_T} \odot \phi'_{K,I_U}}{\sqrt{C}}\right),$$
$$\phi'_{h,I_U} = \mathcal{M}_{I_T\to I_U} \otimes \phi'_{V,I_U},$$

The second iteration follows by encoding $\varphi'_{h,IT}$ and $\varphi'_{h,IU}$ into $\varphi'_{s,IT}$ and $\varphi'_{s,IU}$ utilizing two 1×1 convolution layers. Subsequently, an attention map, M_{att}, is computed using $\varphi'_{s,IT}$ and $\varphi'_{s,IU}$ as inputs. This attention map is instrumental in estimating the homography matrix, utilizing the

information gleaned from the cross-attention mechanisms to refine and determine the transformation between the target and unaligned images.

$$\mathcal{M}_{att} = \phi'_{s,I_T} \odot \phi'_{s,I_U}.$$

3.5.3 Local Attention Kernel:



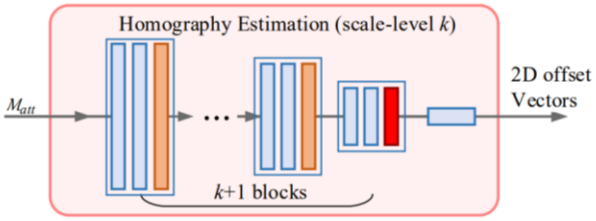
The proposed Local Attention Kernel (LAK) is introduced to expedite the transformer structure by capturing correspondences within a localized range, drawing inspiration from conventional 2D deconvolution (transposed convolution) and convolution techniques.

The generation of the local attention map starts by describing the correspondences between features φ_Q and φ_K within a specific local range—defined as a squared window with a radius of r. For an element positioned at x = (x, y) in φ_Q, its interaction with elements in a local radius r within φ_K, designated as u ∈ N(x), is queried. This interaction generates a local correspondence map M', which serves as the preliminary local attention map. The final attention map M is,

$$\mathcal{M} = \sigma(\frac{\mathcal{M}'}{\sqrt{C}}).$$

The local attention convolution operation involves utilizing the resultant local attention map, M, along with the feature φ_V, to derive a high-level feature φ_h. This process harnesses the information captured in the refined local attention map to obtain enhanced high-level features, facilitating more focused and localized correspondences between the input features.

3.5.4 Histogram Estimation Module:



In each scale-level, the homography estimation module utilizes the attention map $M_{att.}$ The feature undergoes processing within k + 1 convolution blocks, where each block incorporates two 3×3 convolution layers and one max-pooling layer. In the final block, the max-pooling layer is substituted with an average-pooling layer, producing a feature of dimensions $C \times 1 \times 1$. Following this, a 1D convolution layer with a kernel size of 1 is applied, culminating in an 8-dimensional vector signifying the 2D offsets of the four corner points within scale-level k.

It's important to note that H_i within each scale-level denotes the homography matrix at full resolution. Hence, the final homography matrix, H, is computed directly by aggregating the estimated homography matrices, H_i, from each level. This accumulation process is represented as $H = H_1 \times H_2 \times \cdots \times H_K$, resulting in the determination of the comprehensive homography matrix considering information from all scale levels.

3.5.5 Evaluation of LocalTrans:

The evaluation of the proposed LocalTrans network encompasses two distinct settings. Firstly, in the common data setting, assessments are conducted using the MS-COCO dataset, a common choice for numerous deep homography estimation methodologies. Secondly, the evaluation extends to the cross-resolution setting, where the target image possesses a lower resolution compared to other images.

For the cross-resolution setting, two different types of datasets are employed. The first dataset involves synthesized cross-resolution data, wherein the target images are downscaled using bicubic interpolation, specifically with factors of 4× and 8×. Additionally, the evaluation incorporates datasets related to optical zoom-in cross-resolution scenarios and cross-resolution stereo datasets, further diversifying the assessment scenarios. These datasets introduce scenarios where the resolution disparities between images are significant, reflecting real-world scenarios where images might have different resolutions due to various factors like optical zoom or stereo image capturing. LocalTrans outperforms all other architectures on the provided test dataset.

4. Conclusion:

The advancements in deep homography estimation, as detailed in this report, mark a significant transition in computer vision methodologies. Moving from traditional feature matching and RANSAC-based approaches, the field has embraced deep learning techniques, each tailored to overcome specific challenges inherent in conventional methods.

The initial breakthrough in deep homography estimation was led by DeTone, Malisiewicz, and Rabinovich, who introduced a method using convolutional neural networks. This approach, focusing on directly predicting homography matrices, eliminated the need for explicit feature matching. It demonstrated the potential of deep learning in adapting to various imaging conditions and scenarios, showcasing a significant improvement over traditional method.

Further developments in this domain include the Iterative Deep Homography Estimation, which refines the homography estimates through cascading VGG-style networks. This method excels in both static and dynamic scenes, offering specialized variants for each and enhancing the model's versatility and accuracy. Another notable advancement is the Content-Aware Unsupervised Deep Homography Estimation, specifically designed for image pairs with small baselines. This method addresses the limitations of feature-based homography estimation, especially in non-standard scenes, by employing a unique network structure that focuses on content-aware learning.

The Semi-supervised Deep Large-baseline Homography Estimation with Progressive Equivalence Constraint presents a novel approach to large-baseline homography estimation. By breaking down the large baseline into several intermediate stages, this method manages to tackle the challenges posed by substantial appearance and viewpoint changes, often encountered in large-baseline scenarios. It represents a significant step towards bridging the gap between supervised and unsupervised methods in homography estimation.

Lastly, the introduction of LocalTrans, a Multiscale Local Transformer Network, addresses the challenges of cross-resolution homography estimation. This method combines the advantages of multiscale approaches with local transformer networks, offering efficient and accurate estimation in scenarios with significant resolution disparities between images.

In summary, the integration of deep learning into homography estimation reflects the ongoing evolution in computer vision. These advancements not only provide more accurate, efficient, and robust solutions but also open new avenues for application in various challenging environments

5. References:

1. Deep Image Homography Estimation
2. Iterative Deep Homography Estimation
3. Content-Aware Unsupervised Deep Homography Estimation
4. Semi-supervised Deep Large-baseline Homography Estimation with Progressive Equivalence Constraint
5. LocalTrans: A Multiscale Local Transformer Network for Cross-Resolution Homography Estimation