VisIRNet: Deep Image Alignment for UAV-Taken Visible and Infrared Image Pairs

Sedat Özer, *Senior Member, IEEE*, and Alain P. Ndigande

***Abstract*— This article proposes a deep-learning-based solution for multimodal image alignment regarding unmanned aerial vehicle (UAV)-taken images. Many recently proposed state-of-the-art alignment techniques rely on using Lucas–Kanade (LK)-based solutions for a successful alignment. However, we show that we can achieve state-of-the-art results without using LK-based methods. Our approach carefully utilizes a two-branch-based convolutional neural network (CNN) based on feature embedding blocks. We propose two variants of our approach, where in the first variant (Model A), we directly predict the new coordinates of only the four corners of the image to be aligned; and in the second one (Model B), we predict the homography matrix directly. Applying alignment on the image corners forces the algorithm to match only those four corners as opposed to computing and matching many (key) points, since the latter may cause many outliers, yielding less accurate alignment. We test our proposed approach on four aerial datasets and obtain state-of-the-art results when compared to the existing recent deep LK-based architectures.**

***Index Terms*— Corner-matching, deep learning, image alignment, infrared image registration, Lukas–Kanade (LK) algorithms, multimodal image registration, unmanned aerial vehicle (UAV) image processing.**

I. Introduction

RECENT advancements in unmanned aerial vehicle (UAV) technologies, computing, and sensor technologies, allowed the use of UAVs for various earth observation applications. Many UAV systems are equipped with multiple cameras today, as cameras provide reasonable and relatively reliable information about the surrounding scene in the form of multiple images or image pairs. Such image pairs can be taken by different cameras, at different viewpoints, different modalities, or at different resolutions. In such situations, the same objects or the same features might appear at different coordinates on each image and, therefore, an image alignment (registration) step is needed before applying many other image-based computer vision applications such as image fusion, object detection, segmentation or object tracking as in [43], [44], and [45].

The infrared spectrum and visible spectrum may reflect different properties of the same scene. Consequently, images taken in those modalities, typically, differ from each other. On many digital cameras, the visible spectrum is captured and stored in the form of a red-green-blue (RGB) image model and a typical visible spectrum camera captures visible light ranging from approximately 400 to 700 nm in wavelength [6], [35]. Infrared cameras, on the other hand, capture wavelengths longer than those of visible light, falling between 700 and 10 000 nm [9]. Infrared images can be further categorized into different wavelength ranges as near-infrared (NIR), mid-infrared (MIR), and far-infrared (FIR) capturing different types of information in the spectrum [9], [10], [15], [16].

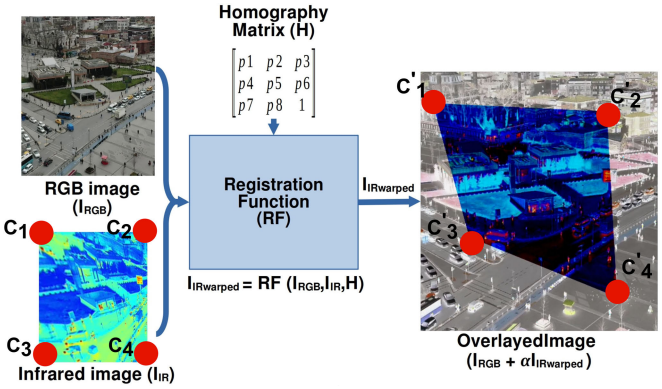

Fig. 1. Overview of the image alignment process is shown. On the left, input RGB, I_{RGB} (192 × 192 pixels) and IR, I_{IR} (128 × 128 pixels) images are shown. The I_{IR} is shown in pseudocolors. Both images are given as input to the registration stage where the transformation parameters represented by the homography matrix (H) are predicted. After the registration process, the I_{IR} is transformed (warped) onto the I_{RGB} space by locating the positions of c_1, c_2, c_3, and c_4 as c_1', c_2', c_3', and c_4'. The warped I_{IR} is overlayed (where $\alpha = 0.4$) on the I_{RGB}.

Image alignment is, essentially, the process of mapping the pixel coordinates from different coordinate system(s) into one common coordinate system. This problem is studied under different names including image registration and image alignment. We will also use the terms *alignment* and *registration* interchangeably in this article. Typically, alignment is done in the form of image pairs mapping from one image (source) onto the other one (target) [18]. Image alignment is a common problem that exists in many image-based applications where both the target and source images can be acquired by sensors using the same modality or using different modalities. There is a wide range of applications of image alignment in many fields including medical imaging [1], [21], UAV applications [22], [36], image stitching [8] and remote-sensing applications [5], [5], [29], [30], [37].

Manuscript received 4 August 2023; revised 16 December 2023 and 2 February 2024; accepted 11 February 2024. Date of publication 20 February 2024; date of current version 8 March 2024. This work was supported by the TÜBITAK under Project 118C356. *(Corresponding author: Sedat Özer.)*

The authors are with the Ozer Laboratory, Department of Computer Science, Ozyegin University, 34794 Istanbul, Turkey (e-mail: sedatist@gmail.com).

Digital Object Identifier 10.1109/TGRS.2024.3367986

Image alignment, in many cases, can be reduced to the problem of estimating the parameters of the perspective transformation between two images acquired by two separate cameras, where we assume that the cameras are located on the same UAV system. Fig. 1 summarizes such an image alignment process where the input consists of a higher resolution RGB image (e.g., 192 × 192 pixels) and a lower-resolution IR image (e.g., 128 × 128 pixels visualized in pseudocolors in the figure). The output of the registration algorithm is the registered (aligned) IR image on the RGB image's coordinate system. As perspective transformation [20] is typically enough for UAV setups containing nearby onboard cameras, our registration process uses a registration function based on the Homography (**H**) matrix. **H** contains eight unknown (projection) parameters and the goal of the registration process is to predict those eight unknown parameters, directly or indirectly.

In the relevant literature, registering RGB and IR image pairs is done by using both classical techniques (such as scale-invariant feature transform (SIFT) [33] along with the random sample consensus (RANSAC) [14] algorithm as in [3]) and by using more recent deep-learning-based techniques as in [7], [34], [52]. Classical techniques include feature-based [40], [50] and intensity-based [39] methods. Feature-based [40], [50] methods essentially find correspondences between the detected salient features from images [47]. Salient features are computed by using approaches such as SIFT [32], speeded-up robust features (SURF) [4], Harris Corner [19], and Shi-Tomas corner detectors [24] in each image. The features from both images are then matched to find the correspondences as in [41], [42], and [46], and to compute the transformation parameters in the form of a homography matrix. The RANSAC [46] algorithm is commonly used to compute the homography matrix that minimizes the total number of outliers in the literature. Intensity-based [39] methods compare intensity patterns in images via similarity metrics. By estimating the movement of each pixel, optical flow is computed and used to represent the overall motion parameters. In [2] and [13] uses Lucas–Kanade (LK)-based algorithms that take the initial parameters and iteratively estimate a small change in the parameters to minimize the error. A typical intensity-based registration technique essentially uses a form of similarity as its metric or as its registration criteria including mean squared error (MSE) [17], cross correlation [28], structural similarity index (SSIM), and a peak signal-to-noise ratio (PSNR) [51]. Such metrics are not sufficient when the source image and target image are acquired by different modalities. This can yield poor performance when such intensity-based methods are used.

Overall, such major classical approaches, typically, are based on finding and matching similar salient keypoints in image pairs, and therefore, they can yield unsatisfactory results in various multimodal registration applications.

Relevant deep alignment approaches use a form of keypoint matching, template matching, or LK-based approaches as in [7] and [27]. Those techniques typically consider multiple points or important regions in images to compute the homography matrix **H** which contains the transformation parameters. However, having the information of four matching points represented by their corresponding 2-D coordinates (x_i, y_i), where $i = 1, 2, 3, 4$ is sufficient to estimate **H**. Therefore, if found accurately, four matching image-corner points between the IR and RGB images would be enough to perform accurate registration between the IR and RGB images. While many techniques based on keypoint extraction can be employed to find matching keypoints between the images, we argue that the corner points on the borders of one image can also be considered as keypoints, and by using those corners of the image, we do not need to utilize any keypoint extraction step.

In this article, we propose a novel deep approach for registering IR and RGB image pairs, where instead of predicting the homography matrix directly, we predict the location of the four corner points of the entire image directly. This approach removes the additional iterative steps introduced by LK-based algorithms and eliminates the steps of computing and finding important keypoints. Our main contributions can be listed as follows.

1) We introduce a novel deep approach for alignment problems of IR images onto RGB images taken by UAVs, where the resolutions of the input images differ from each other.
2) We introduce a novel two-branch-based deep solution for registration without relying on the Lukas–Kanade-based iterative methods.
3) Instead of predicting the homography matrix directly, we predict the corresponding coordinates of the four corner points of the smaller image on the larger image.
4) We study and report the performance of our approach on multiple aerial datasets and present the state-of-the-art results.

II. Related Work

Many recent techniques performing image alignment rely on deep learning. Convolutional neural networks (CNNs) form a pipeline of convolutional layers where filters learn unique features at distinct levels of the network. For example, DeTone et al. [12] proposed a deep image homography estimation network (DHN) that uses CNNs to learn meaningful features in both images and it directly predicts the eight affine transformation parameters of the homography matrix. Later, Le et al. [26] proposed using a series of networks to regress the homograph parameters in their approach. The latter networks in their proposed architecture aim to gradually improve the performance of the earlier networks. Their method builds on top of DHN [12]. Another work in [7] proposed incorporating the LK algorithm in the deep-learning pipeline.

Zhao et al. [52] used a CNN-based network and introduced a learning-based LK block. In their work, they designed modality-specific pipelines for both source and template images, respectively. At the end of each block, there is a unique feature construction function. Instead of using direct output feature maps, they constructed features based on Eigen values and Eigen vectors of the output feature maps. The features constructed from the source and template network channels have a similar learned representation. Transformation

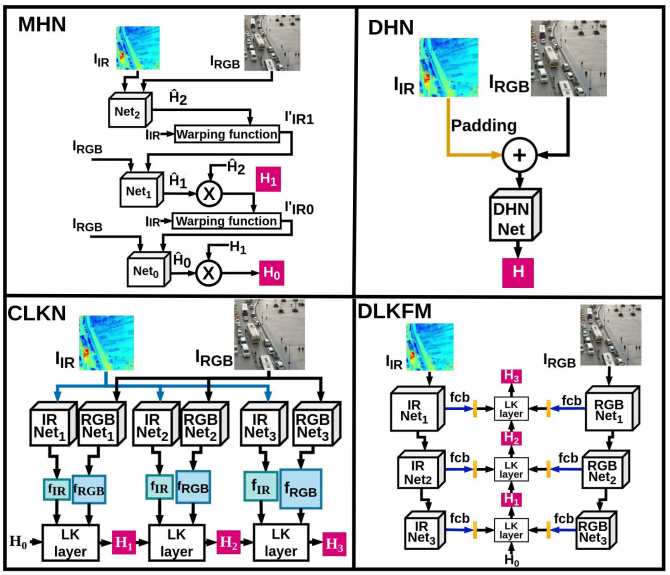

Fig. 2. Architectures of various recently proposed deep alignment algorithms including DHN [12], MHN [26], CLKN [7], and DLKFM [52]. While DHN and MHN predict the homography parameters **H**; CLKN and DLKFM rely on using LK-based iterative approach and they use feature maps at different resolutions. By doing so, they predict homography in steps $\mathbf{H_i}$ where each step aims to correct the previous prediction.

parameters found at a lower scale are given as input to the next level and the LK algorithm iterates until a certain threshold is reached. In another work, Deng et al. [11] utilized disentangled convolutional sparse coding to separate domain-specific and shared features of multimodal images for improved accuracy of registration. Multiscale generative adversarial networks (GANs) are also used to estimate homography parameters as in [34].

The architectural comparisons of the above-mentioned multiple networks are provided in Fig. 2. In DHN [12], the image to be transformed (it is noted as I_{IR} in the figure) is padded to have the same dimensions as the target image (I_{RGB}) and they are concatenated channel-wise. The concatenated images are given to the deep homography network (DHN) for the direct regression of the eight values of the homography matrix. On the other hand, multiscale homography estimation (MHN) [26] adapts using a series of networks (Net_i). The inputs for Net_2 are a concatenation of I_{IR} and I_{RGB}. For the succeeding levels, first, the warping function performs the projective inverse warping operation on the infrared image (I_{IR}) via the homography matrix which was predicted at the previous level. The resulting image (I'_{IR_i}) is first concatenated with I_{RGB} and then given as input to the Net_i. For the following levels, the current matrix and previously predicted matrices are multiplied to form the final prediction. This way MHN aims to learn to correct mistakes made in the earlier levels. Cascaded LK network (CLKN) [7] uses separate networks for each modality. They use levels of different scales in the form of feature pyramid networks and perform registration from the smallest to the largest. The homography matrix from the earlier LK-layer is given as input to the next. Deep LK feature maps (DLKFM) [52] also perform coarse to fine registration as shown in Fig. 2. It uses a special feature construction block called (**fcb**). The (**fcb**) block takes in the feature maps and transforms them into new features based on the Eigen vectors and covariance matrix. The constructed features capture principal information and the registration is performed on the constructed feature maps, thus, it aims to increase the accuracy of the LK-layer. Our approach uses separate feature embedding blocks to process each modality separately. It is trained to extract modality-specific features so that the output feature maps of different modalities can have similar feature representations.

III. Proposed Approach: VisIRNet

In our proposed approach, we aim at performing accurate, single, and multimodal image registration which is free of the iterative nature of LK-based algorithms. We name our network VisIRNet, where we aim to predict the location of the corners of the input image on the target image directly since having four matching points is sufficient to compute the homography parameters. In our proposed architecture, we assume that there are two input images with different resolutions. The overview of our architecture is given in Fig. 3. Our approach first processes two inputs separately by passing them through their respective feature embedding blocks and extracts representative features. Those features are then combined and given to the regression block as input. The goal of the regression block is to compute the transformation parameters accurately. The output of the regression block is eight-dimensional (which can represent the total number of homography parameters or the coordinates of the four corner points of the source image on the target image).

A. Preliminaries

1) Perspective Transformation: Here, by perspective transformation, we mean a linear transformation in the homogenous coordinate system which, in some sense, warps the source image onto the target image. The homography matrix consists of the transformation parameters needed for the perspective transformation. The elements of the 3×3 dimensional homography matrix represent the amount of rotation, translation, scaling, and skewing motions. Homography matrix **H** is defined as follows:

$$\mathbf{H} = \begin{bmatrix} p_1 & p_2 & p_3 \\ p_4 & p_5 & p_6 \\ p_7 & p_8 & 1 \end{bmatrix} \tag{1}$$

where the last element (p_9) is set to 1 to ensure the validity of conversion from homogeneous to the Cartesian coordinates. **Warping function** maps a set of coordinates $[(x_i, y_i), \ldots,]$ to another coordinate system via **H**. Let $c_i = (x_i, y_i)$ be the location of a point in the coordinates set C of the source image. Let $W(c, P)$ be the warping function that warps given coordinate c with parameter set P of **H** to the target image

$$c'_i = W(c_i, P). \tag{2}$$

The warping process is a linear transformation in a homogeneous coordinate system. Therefore, the Cartesian coordinates are first transformed into the homogeneous coordinate system

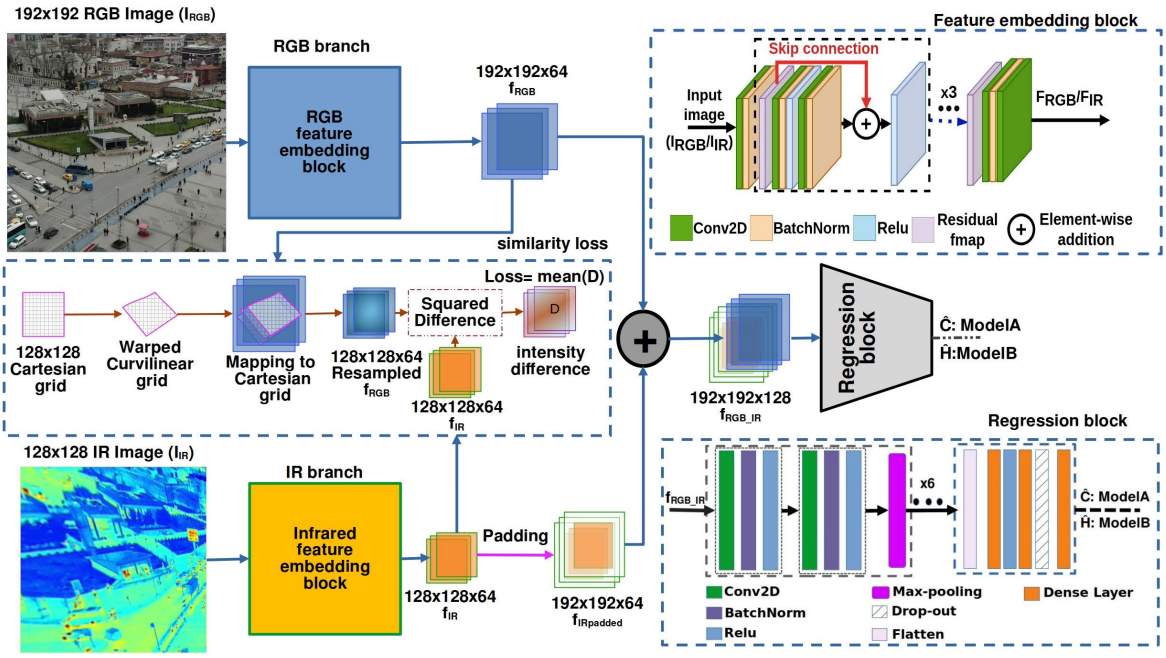

Fig. 3. Overview of our proposed network architecture. Two parallel branches including the RGB branch and IR branch (feature embedding blocks) extract the salient features for RGB and IR images, respectively. Those features are, then channel-wise concatenated and fed into the regression block for direct (Model B) or indirect (Model A) homography prediction, that is, the model can be trained for learning the homography matrix in Model B or to regress the corresponding coordinates of the four corners of the input IR image on the RGB image in Model A. The output is an eight-dimensional vector (for $\mathbf{H}$) if Model B is used; and it is an eight-dimensional vector where those eight values correspond to the (x, y) coordinates of the four corners of the IR image, if Model A is used. The details of the feature embedding block are given in the top corner of the figure (also see Table I). The details of the regression block are given in the lower right corner of the figure (also see Table II).

by adding the extra z dimension to the 2-D Cartesian pixel coordinates. Let c_i be the pixel with x_i, y_i coordinates. Homogeneous coordinate of c_i can be represented by setting $z-$axis to 1, that is, $c_{h_i} = (x_i, y_i, 1)$. Once we have the homography matrix, we warp any given ith pixel location c_i represented by (x_i, y_i) to its warped version c_i^{warped} on the other image's Cartesian coordinate as follows:

$$c_{h_i}^{\text{warped}} = W(c_i, P) \Longleftrightarrow \begin{bmatrix} x_i' \\ y_i' \\ z_i' \end{bmatrix} = \begin{bmatrix} p_1 & p_2 & p_3 \\ p_4 & p_5 & p_6 \\ p_7 & p_8 & 1 \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} \quad (3)$$

where x_i', y_i', z_i', are warped homogeneous coordinates of c_i^{warped} which can be converted to Cartesian coordinates by simply division by the z_i' value. Therefore, we can obtain the final warped 2-D pixel coordinates in Cartesian coordinates as follows: $c_i' = (x_i', y_i')$, where

$$x_i^{\text{warped}} = \frac{x_i'}{z_i'} \Longleftrightarrow \frac{p_1 x_i + p_2 y_i + p_3}{p_7 x_i + p_8 y_i + 1} \quad (4)$$

$$y_i^{\text{warped}} = \frac{y_i'}{z_i'} \Longleftrightarrow \frac{p_4 x_i + p_5 y_i + p_6}{p_7 x_i + p_8 y_i + 1}. \quad (5)$$

B. Network Structure

Our proposed network is composed of multimodal feature embedding blocks (MMFEB) and a regression block (see Fig. 3). The regression block is responsible for predicting the eight homography matrix parameters directly or indirectly. In this article, we study the performance of two variants of our proposed model and we call them Model A and Model B. Model A predicts the coordinates of the corner points while its variant, Model B, predicts the direct homography parameters. In Model A, four corners are enough to find the homography matrix. Therefore, the last layer has eight neurons for the four (x, y) corner components for Model A, or the eight unknown homography parameters for Model B.

1) Multimodal Feature Embedding Backbone: MMFEB is responsible for producing a combined representative feature set formed of fine-level features for both of the input images. The network then will use that combined representative feature set to transform the source image onto the target image. We adapt the idea of giving RGB and infrared modalities separate branches as in [52]. We use two identical networks (branches) with the same structure but with different parameters for RGB and infrared images, respectively. Therefore, the multimodal feature embedding block has two parallel branches with identical architectures (however, they do not share parameters), namely the RGB branch and the infrared branch. We first train the multimodal feature embedding backbone by using average similarity loss $\mathcal{L}_{\text{sim}}$ (see 6). To compute the similarity loss, we first generate a 128×128 rectilinear grid, representing locations in the infrared coordinate system as in spatial transformers [23]. Then, we use the ground-truth homography matrix to warp the grid onto the RGB coordinate system resulting in a warped curvilinear grid representing projected locations. We use bilinear interpolation [25], [38] to sample those warped locations on the RGB feature maps (f_{RGB}). After that, we can compute the similarity loss between IR feature maps and resampled RGB feature maps. Algorithm 1 provides the algorithmic details of calculating the similarity loss for the feature embedding block.

MMFEB is trained by using the $\mathcal{L}_{\text{sim}}$ (see 6) which is detailed in Algorithm 3. Steps for training the MMFEB are given in Algorithm 1. The regression block is trained with homography loss ($\mathcal{L}_2^H$) in combination with average corner

TABLE I

LAYER-BY-LAYER DETAILS OF THE FEATURE EMBEDDING BLOCK. THERE ARE ALSO SKIP CONNECTIONS BETWEEN THE LAYERS IN THIS ARCHITECTURE AS SHOWN IN FIG. 3

Layer	Filter Number	Filter-dims	Stride	Padding	Repetition
Conv2D	64	3x3	1	SAME	x1
BN	-	-	-	-	
Conv2D	64	3x3	1	SAME	
BN	-	-	-	-	
Relu	-	-	-	-	x3
Conv2D	64	3x3	1	SAME	
BN	-	-	-	-	
Relu	-	-	-	-	
Conv2D	64	3x3	1	SAME	
BN	-	-	-	-	x1
Conv2D	64	3x3	1	SAME	

error ($\mathcal{L}_{\text{Ace}}$) (see the "average corner error (ACE)" section for the definitions of $\mathcal{L}_{\text{Ace}}$), yielding the total loss $\mathcal{L}$ to train our model. Table I summarizes the structure of our used MMFEB.

2) Regression Block: The second main stage of our pipeline is the regression block which is responsible for making the final prediction. The prediction can be the four corner locations if Model A; or the unknown parameters of the homography matrix if Model B. f_{RGB} and f_{IR} are the feature maps extracted by passing the RGB image and infrared image through their respective feature embedding blocks in the feature embedding block. Note that f_{RGB} and f_{IR} have different dimensions. Therefore, we apply zero-padding to the lower dimensional feature maps (f_{IR}) so that we can bring its dimensions to the dimensions of f_{RGB}, resulting f_{IRpadded}. We concatenate (channel-wise) f_{IRpadded} to f_{RGB} feature maps coming from infrared and RGB feature embedding blocks and use that as input for the regression block.

The architecture for the regression block is further divided into two subparts as shown in Fig. 3. The first part is composed of six levels. Apart from the last level, each level is composed of two sublevels followed by a max-pooling layer. Sublevel is a convolution layer followed by a batch normalization layer followed by a relu activation function. Sublevels m and n of a level l are identical in terms of the filters used, kernel size, stride, and padding used for level l. The sixth level does not have a max-pooling layer. The second part has two 1024-dense layers with relu as an activation function followed by a dropout layer and an eight-dense output layer for eight parameters of the homography matrix or corner components. Feature maps from the previous part are flattened and given to the second part where homography matrix parameters or corner components are predicted according to the model used. Table II gives detailed information for the first and the second parts of the regression head.

Algorithm 1 Training Steps of the MMFEB

Inputs: $I^*_{\text{RGB}}, I^*_{IR}$ ▷ * indicates whole training set
for $e \leftarrow 0$ to $epochs$ **do**
  for $batch \leftarrow 0$ to $datasetSize/batchsize$ **do**
    $I_{\text{RGB}} = I^*_{\text{RGB}}[batch]$
    $I_{\text{IR}} = I^*_{\text{IR}}[batch]$
    $H \leftarrow groundTruthHomography$
    $f_{\text{IR}} \leftarrow RGBbranch(I_{\text{RGB}})$
    $f_{\text{RGB}} \leftarrow IRbranch(I_{\text{IR}})$
    $simLoss \leftarrow \mathcal{L}_{sim}(f_{\text{IR}}, f_{\text{RGB}}, H)$
    $Backprop(simLoss)$ Using $AdamOptimizer$
  end for
end for

TABLE II

LAYER-BY-LAYER DETAILS OF THE REGRESSION BLOCK AS SHOWN IN FIGURE 3. LEVELS INDICATED BY L ARE GROUPS OF CONV2D + BATCHNORMALIZATION + RELU. THE CONV2D LAYERS IN EACH LEVEL HAVE THE SAME CHARACTERISTICS AND FILTER DIMENSIONS. THE NUMBER OF USED FILTERS INCREASES AS WE GET DEEPER IN THE ARCHITECTURE

Level	Number of filters / Units	Filter-dims	Stride	Padding	Activation
L1	32	3x3	1	SAME	
max-pool	-	2x2	2	SAME	
L2	64	3x3	1	SAME	
max-pool	-	2x2	2	SAME	
L3	64	3x3	1	SAME	
max-pool	-	2x2	2	SAME	
L4	128	3x3	1	SAME	
max-pool	-	2x2	2	SAME	
L5	128	3x3	1	SAME	
max-pool	-	2x2	2	SAME	
L6	256	3x3	1	SAME	
Flatten					
Dense	1024	-	-		Relu
Dense	1024	-	-		Linear
Dropout	20%	-	-		-
Dense	8	-	-		Linear

Algorithm 2 Training Step of the Regression Block

let **M** be RegressionBlock
Inputs: $I^*_{\text{RGB}}, I^*_{\text{IR}}$ ▷ * indicates whole training set
for $e \leftarrow 0$ to $epochs$ **do**
  for $batch \leftarrow 0$ to $datasetSize/batchsize$ **do**
    $I_{\text{RGB}} = I^*_{\text{RGB}}[batch]$
    $I_{\text{IR}} = I^*_{\text{IR}}[batch]$
    $H \leftarrow groundTruthHomography$
    $f_{\text{IR}} \leftarrow RGBbranch(I_{\text{RGB}})$
    $f_{\text{RGB}} \leftarrow IRbranch(I_{\text{IR}})$
Ensure: $f_{\text{RGB}}.shape = 192 \times 192 \times 64$
Ensure: $f_{\text{IR}}.shape = 128 \times 128 \times 64$
    $f_{IRpadded} = zeroPadd(f_{\text{IR}})$
    $f_{\text{RGB\_IR}} = concat(f_{\text{RGB}}, f_{IRpadded})$
    $\hat{H} = \mathbf{M}(f_{\text{RGB\_IR}})$
    $Loss = \mathcal{L}(\hat{H}, H)$
    $Backprop(Loss)$ using $AdamOptimizer$
  end for
end for

C. Loss

While MMFEB uses *similarity loss*, we used two loss terms based on the corner error and homography for the regression head.

1) Similarity Loss: The similarity loss is used to train MMFEB and is defined as follows:

$$\mathcal{L}_{\text{sim}} = \frac{1}{x * y} \sum_{x=0}^{n} \sum_{y=0}^{n} \left(f'_{\text{RGB}}(x, y) - f_{\text{IR}}(x, y) \right)^2 \quad (6)$$

where $f_{\text{IR/RGB}}(x, y)$ is the value at (x, y) location for respective image feature maps. $f'_{\text{RGB}}(x, y)$ is the value at (x, y) location on the resampled RGB feature maps. Note that the (x, y) is a location on the coordinate system constrained by the infrared image height and width. The algorithmic details of the similarity loss are provided in Algorithm 3.

2) L_2 Homography Loss Term: Model B is trained to predict the values of the elements of the homography matrix. Therefore, its output is the eight elements of a 3×3 matrix

Algorithm 3 Computing the $\mathcal{L}_{sim}$ loss

$f_{\text{RGB}} \leftarrow RGBbranch(I_{\text{RGB}})$
$f_{\text{IR}} \leftarrow IRbranch(I_{\text{IR}})$
$H \leftarrow groundTruthHomography$
$grid_{n\times n} \leftarrow 2x2gridwithIrdimensions$
Ensure: $warpedGrid = warpGrid(grid_{n\times n}, H^{-1})$
$f'_{\text{RGB}} = BilinearSampler(f_{\text{RGB}}, warpedGrid)$
$\mathcal{L}_{sim} \leftarrow 0$
for $i = 0, i \leq n \times n$ **do**
  $Ir_i = f_{\text{IR}}[i]$
  $Rgb_i = f'_{\text{RGB}}[i]$
  $P_{diff} \leftarrow Ir_i - Rgb_i$
  $\mathcal{L}_{sim} \leftarrow \mathcal{L}_{sim} + P^2_{diff}$
end for
$\mathcal{L}_{sim} \leftarrow \mathcal{L}_{sim}/(n \times n)$

TABLE III
SUMMARY OF THE USED DATASETS IN OUR EXPERIMENTS IS GIVEN. THE TRAINING AND TEST DATASETS ARE GENERATED AS EXPLAINED IN SECTION IV

Dataset	Modality	Training set	Test Set
SkyData	RGB + Infrared	27700	7990
MSCOCO [31]	Single modality (RGB)	82600	6400
Google Maps	RGB + map (vector)	8800	888
Google Earth	RGB + RGB	8750	850
VEDAI [48]	RGB + Infrared	8722	3738

(where the ninth element is set to 1). The homography-based loss term: L_2^H is defined as follows: let $[p_i$: *(for* $i = 1, 2, 3, 4, 5, 6, 7, 8$*), 1]* be the elements of a 3×3 **H** ground-truth homography matrix. Similarly, let $[\hat{p}_i$: *(for* $i = 1, 2, 3, 4, 5, 6, 7, 8$*), 1]* be elements of 3×3 $\hat{H}$, the predicted homography matrix. Then, $\mathcal{L}_2^H = (1/8)\sum_{i=1}^{8}(p_i - \hat{p}_i)^2$, where $\mathcal{L}_2^H$ represents the homography loss based on the L_2 distance.

3) Average Corner Error: Ace is computed as the average sum of squared differences between the predicted and ground-truth locations of the corner points. For Model B, we use the predicted homography matrix to transform the four corners of the infrared image onto the coordinate system of the RGB image, and together with ground-truth locations we compute $\mathcal{L}_{\text{Ace}}$. Let e_i be a corner at the (x_i, y_i) coordinates on the infrared image and let e'_i be its warped equivalent on the RGB coordinate space such that $e'_i = W(e_i, P)$ where W is the warping function

$$\mathcal{L}_{\text{Ace}} = \frac{1}{4}\sum_{i=1}^{4} D\left(e_i, e'_i\right)^2 = \frac{1}{4}\sum_{i=1}^{4}\left(W(e_i, P) - W\left(e_i, \hat{P}\right)\right)^2 \quad (7)$$

where D is defined as $D(e_i, e'_i) = W(e_i, P) - W(e_i, \hat{P})$, and where P and $\hat{P}$ are ground truth and predicted vectorized homography matrices, respectively. The total loss for ModelB, then, is computed as $\mathcal{L} = \mathcal{L}_2^H + \gamma\mathcal{L}\text{Ace}$, where γ is weight factor (a hyperparameter).

In ModelB, we predict the x and y locations of the four corner points, instead of computing the homography matrix. This makes it possible for the network to learn to predict exact locations (landmarks) instead of focusing on one solution. As shown in our experiments (see Fig. 4 for qualitative and Fig. 5 for quantitative results), Model A converges faster and yields better results while minimizing outliers. We use a slightly modified version of $\mathcal{L}_{\text{Ace}}$ for Model A such that $\hat{e}_i$ becomes the ground-truth corner coordinate in RGB coordinate space. For Model A, $\mathcal{L}_{\text{Ace}}$ is defined as follows:

$$\mathcal{L}_{\text{Ace}} = \frac{1}{4}\sum_{i=1}^{4}\left(e_i - \hat{e}_i\right)^2. \quad (8)$$

In addition to these loss functions, we also used additional loss functions in the MMFEB block during our ablation study. Those functions are $\mathcal{L}_{\text{MAE}}$ and $\mathcal{L}_{\text{SSIM}}$. They are briefly defined below

$$\mathcal{L}_{\text{MAE}} = \frac{1}{x * y}\sum_{x=0}^{n}\sum_{y=0}^{n}\left|f'_{\text{RGB}}(x, y) - f_{\text{IR}}(x, y)\right| \quad (9)$$

$$\mathcal{L}_{\text{SSIM}} = 1 - \text{SSIM}\left(f'_{\text{RGB}}(x, y), f_{\text{IR}}(x, y)\right) \quad (10)$$

where SSIM is used as also used and defined in [43].

IV. EXPERIMENTS

In this section, we describe our experimental procedures, used datasets, and our metrics. Below we describe our used datasets.

A. Datasets

In our experiments, we use Skydata[1] containing RGB and IR image pairs, MSCOCO [31], Google-Maps, and Google-Earth (as taken from DLKFM [52]), VEDAI [48] datasets. Refer to Table III for more details about the used datasets in our experiments. SkyData is originally a video-based dataset that provides each frame of the videos in image format.

B. Generating the Training and Test Sets

To train the algorithms, we need unregistered and registered (ground truth) image pairs. For SkyData, we randomly select m frame pairs for each video sequence.

For each dataset that we use, we generate the training and test sets as follows.

1) Select a registered image pair at higher resolutions.
2) Sample (crop) regions around the center of the image to get smaller patches of 192×192 pixels. This process is done in parallel for visible and infrared images.
3) If the extracted patches are not sufficiently aligned, manually align them.
4) For each pair, select a subset of the IR image, by randomly selecting four distinct locations on the image.
5) Find perspective transformation parameters that map those randomly chosen points to the following fixed locations: (0,0), $(n-1,0)$, $(n-1, n-1)$, $(0,n-1)$ so that they can correspond to the corners of the unregistered IR image patch, where we assume that the unregistered I_{IR} is $n \times n$ dimensional (in our experiments n is set to 128). This process creates an unregistered infrared patch (from the already registered ground truth) that needs to be placed back to its true position.
6) Use those four initially selected points as the ground-truth corners for the registered image.

[1] www.skydatachallenge.com

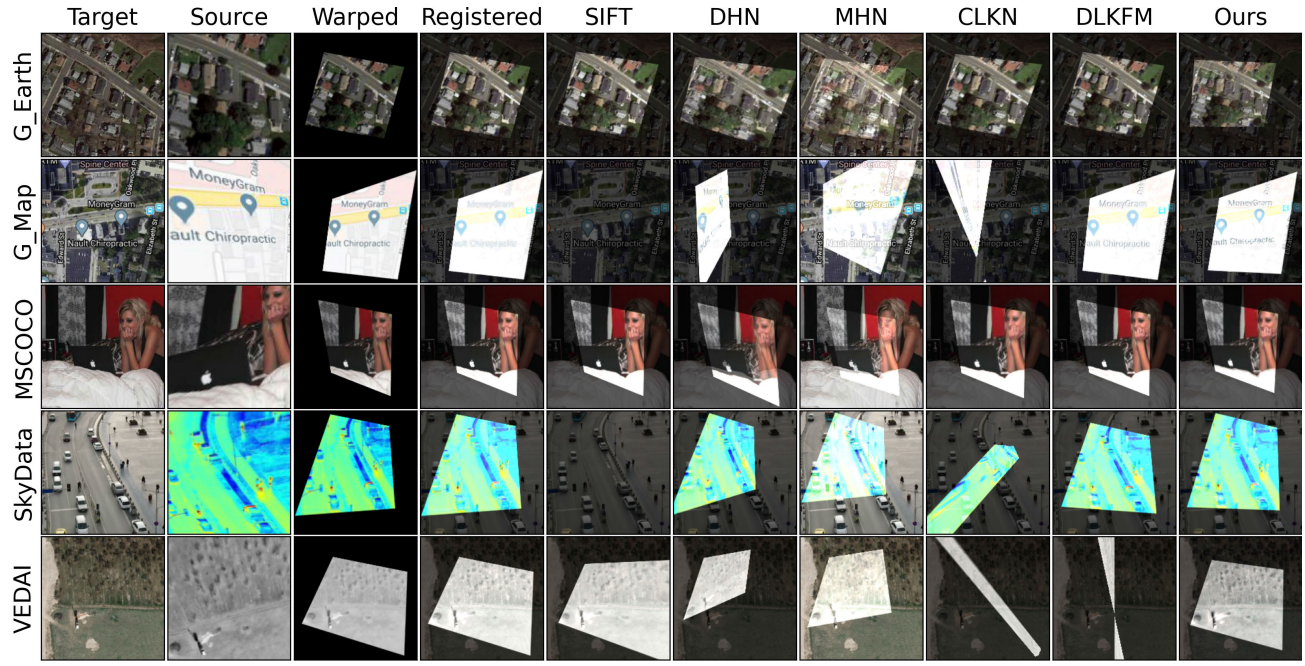

Fig. 4. Qualitative results on sample image pairs taken from different datasets. The first two columns show the input image pairs for the algorithms. The target image is 192 × 192 pixels and the source image is 128 × 128 pixels (which covers a scene that is a subset of the target image). The third column shows the ground-truth version (192 × 192 pixels) of the source image on the coordinate system of the target image after being warped. The fourth column shows the ground truth (warped) where the source image is overlayed on the target image (192 × 192 pixels). The remaining six columns show the overlayed results (192 × 192 pixels), after applying for registration with the algorithms in the order of SIFT, DHN, MHN, CLKN, DLKFM, and our approach, respectively. Visually, each algorithm's result can be compared to the image in the fourth column.

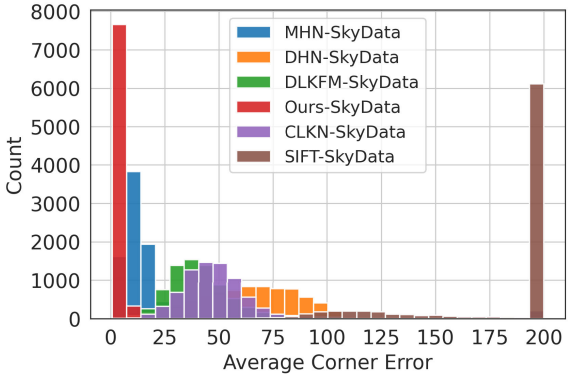

Fig. 5. ACE distribution versus count of image pairs for different models is shown on the test set of Skydata.

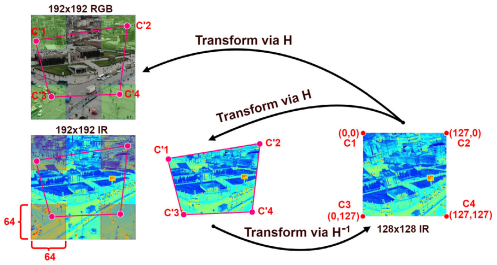

Fig. 6. How to select initial corner points on the registered image pairs and how to generate the training data. First, a random image patch is taken from the originally registered IR image. Then, the random corners of that patch are transformed into fixed coordinates and after that, the **H** matrix (and its inverse) performing that transformation is computed.

7) Repeat process k times to create k different image pairs. This newly created dataset is then split into training and test sets.
8) The RGB images are used as the target set and the transformed infrared patches are used as the source set (for both training and testing).

This process is done on randomly selected registered pairs for each dataset. Fig. 6 also illustrates this process on a pair of RGB and IR images. The list of all the used datasets and their details are summarized in Table III.

C. Evaluation Metrics

As shown in Table IV, we quantitatively evaluate the performance of our models using Ace and homography error. We compute each algorithm's result distribution in terms of quantiles, mean, standard deviation, and min-max values for a given test set. Quartiles are a set of descriptive statistics that summarize central tendency and variability of data [49]. Quartiles are a specific type of quantiles that divide the data into four equal parts. The three quartiles are denoted as Q1, Q2 (which is also known as the median), and Q3. The 25% (Q1), 50% (Q2), and 75% (Q3) percentiles indicate that k% of the data falls below the kth quartile (the bottom right illustration in Fig. 7 also illustrates these terms). To find quartiles, we first sort elements in the data being analyzed in ascending order. The first quartile is the number of samples that fall below the dataset size*(1/4) element. Likewise, the second quartile is the count of elements that fall below dataset size * (2/4) and the third quartile is dataset size * (3/4)th element in the sorted dataset. The samples that fall out of (Q1−1.5 *

TABLE IV

COMPARATIVE RESULTS OF ALGORITHMS ON EACH DATASET IN TERMS OF ACE. BEST RESULTS ARE SHOWN IN BOLD. THE RESULTS ILLUSTRATE THAT AVERAGE TRADITIONAL SIFT PERFORMED ON AVERAGE IN DATASETS OF SINGLE OR CLOSE MODALITIES. IN CASES WHERE ENOUGH PAIRS WERE NOT FOUND SIFT IS UNABLE TO ESTIMATE HOMOGRAPHY MATRIX. WE ASSIGN A CONSTANT 10000.0 AS THE ERROR. LEARNING-BASED ALGORITHMS DHN AND MHN DIRECTLY PREDICT HOMOGRAPHY MATRIX WITHOUT LEARNING COMMON REPRESENTATION ALSO SUFFERS ESPECIALLY ON DATASETS SUCH AS SKYDATA AND GOOGLE MAPS. THIS IS DUE TO THE MODELS BEING UNABLE TO CREATE MEANINGFUL CORRESPONDENCES FOR INPUT AND TARGET IMAGES AS A RESULT OF THE MODALITY DIFFERENCE LEVEL. NOTE THAT OUR APPROACH HAS A SMALL STANDARD DEVIATION AS OPPOSED TO LK-BASED APPROACHES. LK TECHNIQUES OFTEN SIGNIFICANTLY DEVIATE FROM THE SOLUTION DEPENDING ON NUMBER OF ITERATIONS THEY ARE RUN AND INITIAL PARAMETERS THEY RECEIVE.

(a) SkyData Results

	mean	std	min	25%	50%	75%	max
MHN [26]	14.5	67.29	1.19	8.11	11.36	15.4	4296.21
DHN [12]	77.96	854.59	4.1	48.34	65.95	82.19	76119.41
DLKFM [52]	93.4	2894.7	7.32	32.08	40.73	51.95	258091.03
Ours	**3.83**	**1.77**	**0.78**	**2.64**	**3.46**	**4.61**	**19.19**
CLKN [7]	77.31	862.85	5.47	38.66	47.72	57.74	73661.8
SIFT [33]	43477.63	201670.37	2.13	232.88	1275.96	1285.6	100000.0

(b) VEDAI [48] Results

	mean	std	min	25%	50%	75%	max
DLKFM [52]	382.4	3363.72	10.93	101.43	120.09	187.6	189375.0
Ours	**24.76**	**4.77**	4.09	21.59	**24.88**	**28.03**	**42.77**
MHN [26]	374.53	5519.56	8.74	56.4	103.68	141.99	319559.19
SIFT [33]	40221.01	195588.26	**0.11**	**2.07**	88.96	1264.86	100000.0
DHN [12]	163.87	427.16	41.59	128.8	137.43	146.43	19138.92
CLKN [7]	99.49	709.1	2.45	37.97	52.29	75.13	30289.16

(c) GoogleEarth Results

	mean	std	min	25%	50%	75%	max
DHN [12]	1073.9	25214.41	8.85	52.1	67.17	83.78	733889.5
Ours	**10.91**	**3.61**	4.19	8.25	10.5	**12.9**	**31.13**
SIFT [33]	1334.41	34297.53	0.51	**2.63**	8.45	108.25	100000.0
DLKFM [52]	27.65	121.65	**0.36**	7.41	18.5	28.96	2733.34
MHN [26]	118.86	257.47	12.62	45.06	60.91	106.44	4552.5
CLKN [7]	14.58	36.57	0.38	3.15	**6.51**	14.51	730.46

(d) GoogleMaps Results

	mean	std	min	25%	50%	75%	max
MHN [26]	319.68	1003.02	9.48	40.68	73.54	208.8	12910.34
SIFT [33]	178912.9	382210.28	48.79	1273.04	1281.86	1292.6	100000.0
CLKN [7]	123.96	453.65	13.22	56.43	66.9	80.77	8496.15
DHN [12]	131.27	410.38	12.28	30.27	43.08	115.07	9866.02
Ours	**9.57**	**4.15**	2.6	**6.82**	**8.68**	**11.28**	**33.67**
DLKFM [52]	77.78	183.6	**0.47**	20.27	38.76	66.28	3251.74

(e) MSCOCO [31] Results

	mean	std	min	25%	50%	75%	max
DLKFM [52]	67.16	2515.37	**0.06**	0.44	8.31	31.12	200374.44
Ours	**3.67**	**2.45**	0.64	2.28	2.99	4.13	**27.14**
CLKN [7]	6.45	8.96	0.1	1.68	3.86	8.01	280.96
DHN [12]	622.38	7493.6	3.0	141.93	194.88	383.56	580642.19
SIFT [33]	3308.58	57236.2	0.07	**0.37**	**0.65**	**1.38**	100000.0
MHN [26]	15.5	7.31	1.84	10.17	14.39	19.32	90.29

IQR and Q3+1.5IQR) where IQR is interquartile range, are considered outliers. The box plot as in Fig. 7 illustrates the above-mentioned description visually.

Table V shows an ablation study on using different loss functions in each block in our architecture. The used metric in the table is Ace and the best values are shown in bold. The

TABLE V

ABLATION STUDY ON USING DIFFERENT COMBINATIONS OF LOSS FUNCTIONS ON TWO DIFFERENT DATASETS. THE LOSS FUNCTIONS SHOWN IN EACH ROW ARE USED FOR THE MMFEB BLOCK, AND THE LOSS FUNCTIONS SHOWN IN EACH COLUMN ($\mathcal{L}_{\text{ACE}}$ AND L_2^H) ARE USED FOR THE REGRESSION BLOCK IN OUR MODEL. BEST RESULTS ARE SHOWN IN BOLD. ACE IS THE METRIC USED TO COMPUTE THE RESULTS FOR EACH LOSS FUNCTION COMBINATION. THE LAST COLUMN SHOWS THE AVERAGE ACE VALUE FOR EACH LOSS FUNCTION USED IN THE MMFEB BLOCK. ON AVERAGE, $\mathcal{L}_{\text{SIM}}$ YIELDED THE BEST RESULTS

	SkyData		VEDAI		Average
	$\mathcal{L}_{Ace}$	L_2^H	$\mathcal{L}_{Ace}$	L_2^H	
$\mathcal{L}_{sim}$	18.6	21.6	19.1	**128.3**	**36.85**
$\mathcal{L}_{MAE}$	**18.5**	35.8	**18.5**	178.1	62.7
$\mathcal{L}_{SSIM}$	**18.5**	**20.1**	19.1	134.2	47.9

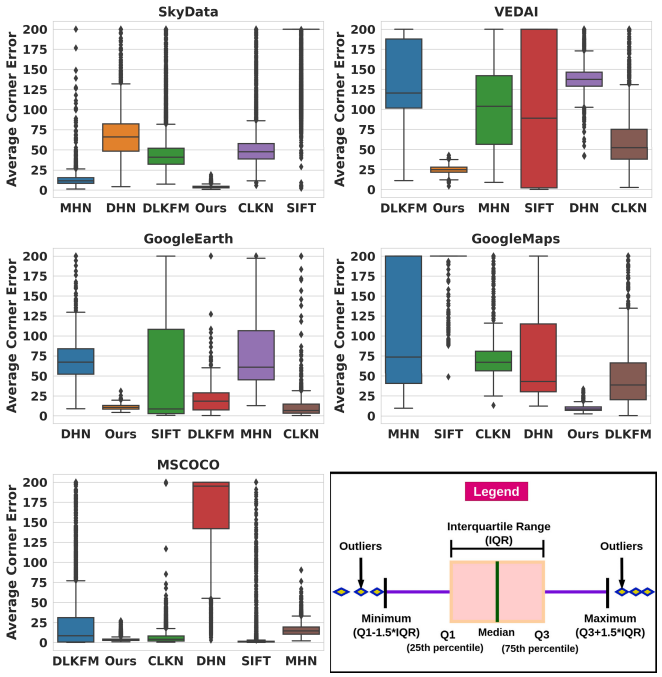

Fig. 7. Each plot shows the ACE for the algorithms including MHN, DHN, DLKFM, CLKN, SIFT, and ours for a different dataset. The legends used in the plots are also given in the lower right corner of the figure.

loss functions in each row are used to train the MMFEB block including $\mathcal{L}_{\text{sim}}$, $\mathcal{L}_{\text{MAE}}$, and $\mathcal{L}_{\text{SSIM}}$. The loss functions used for the regression block are $\mathcal{L}_{\text{Ace}}$ and L_2^H. In the table, the last column shows the average error for both $\mathcal{L}_{\text{Ace}}$ and L_2^H (over two datasets including SkyData and VEDAI) for each of the used loss functions in the MMFEB block.

Next, we provide experimental results on the effect of the hyperparameters that we studied for both Model A and Model B. Table VI summarizes those results. In particular, we studied the effect of using different loss functions (L_1, L_2, and L_{Ace}) and using different batch sizes for both models. All the experiments were done on the SkyDataset. The best results are shown in bold. Overall, Model B showed promising results achieving better results when compared to Model A. Therefore, for the rest of our experiments, we kept using Model B only.

TABLE VI

COMPARISON OF THE RESULTS OF MODEL A AND MODEL B AT VARIOUS HYPERPARAMETERS INCLUDING BATCH SIZES AND LOSS FUNCTIONS. THE TOP TABLE SHOWS THE HOMOGRAPHY ERROR IN (A), WHILE THE BOTTOM TABLE SHOWS THE RESULTS AS ACE IN (B). NOTE THAT THE RESULTS ILLUSTRATING LOW HOMOGRAPHY ERROR DOES NOT NECESSARILY IMPLY SMALL REGISTRATION ERROR. WE STUDY THE EFFECT OF BACTH SIZES AND LOSS FUNCTIONS. DIRECTLY PREDICTING HOMOGRAPHY MATRIX WORK BUT IT DOES NOT MINIMIZE THE REGISTRATION ERROR AS PREDICTING DIRECT CORNERS IN OUR EXPERIMENTS. THESE RESULTS ARE OBTAINED ON SKYDATA DATASET

(a)

Model	BatchS	loss	mean	std	min	25%	50%	75%	max
ModelB	8	L1	**0.6**	0.44	0.03	0.32	**0.49**	**0.76**	8.38
ModelB	8	L2	1.84	4.49	**0.0**	0.37	0.89	2.03	219.33
ModelA	8	Ace	2.49	4.15	**0.0**	0.48	1.23	2.85	63.65
ModelB	16	L1	**0.6**	**0.4**	0.03	0.34	0.51	**0.76**	**6.08**
ModelB	16	L2	1.94	4.05	**0.0**	**0.3**	0.8	2.02	92.81
ModelA	16	Ace	2.14	4.1	**0.0**	0.4	1.04	2.42	176.51
ModelB	32	L1	0.7	0.46	0.02	0.4	0.61	0.9	8.17
ModelB	32	L2	2.79	5.3	**0.0**	0.5	1.31	3.08	149.63
ModelA	32	Ace	2.98	4.71	**0.0**	0.57	1.51	3.5	81.88
ModelB	64	L1	0.73	0.49	0.03	0.39	0.61	0.93	7.03
ModelB	64	L2	3.62	5.94	**0.0**	0.71	1.82	4.09	107.96
ModelA	64	Ace	3.65	5.17	**0.0**	0.91	2.22	4.57	157.6

(b)

Model	BatchS	Loss	Mean	Std	Min	25%	50%	75%	Max
ModelB	8	L1	21.38	44.01	2.53	8.65	12.57	18.27	235.99
ModelB	8	L2	25.15	119.95	2.30	11.83	13.11	15.53	893.01
ModelA	8	Ace	3.96	**1.64**	0.93	2.84	3.68	4.72	19.59
ModelB	16	L1	25.83	108.53	1.66	8.02	10.97	16.27	743.54
ModelB	16	L2	31.27	143.28	2.29	10.52	13.65	18.47	1180.64
ModelA	16	Ace	**3.83**	1.77	**0.78**	**2.64**	**3.46**	**4.61**	19.19
ModelB	32	L1	25.17	101.33	1.87	6.08	8.80	16.97	803.25
ModelB	32	L2	6.50	4.11	2.05	5.43	6.40	7.35	**15.07**
ModelA	32	Ace	5.32	1.87	1.35	4.04	5.02	6.22	22.21
ModelB	64	L1	30.60	177.66	3.41	11.07	11.65	12.41	1298.48
ModelB	64	L2	31.86	135.19	1.74	8.10	13.46	19.23	981.56
ModelA	64	Ace	5.01	2.01	0.95	3.62	4.65	6.0	18.94

Fig. 7 uses a box plot, also known as a box-and-whisker plot, to display the distribution of ACE for different datasets and different models. It provides a summary of key statistical measures such as the minimum, first quartile (Q1), median (Q2), third quartile (Q3), and maximum. The length of the box indicates the spread of the middle 50% of the data. The line inside the box represents the median (Q2). The whiskers extend from the box and represent the variability of the data beyond the quartiles, in our case, they represent $Q1-1.5*\text{IQR}$ and $Q3 + 1.5 * \text{IQR}$. Individual data points that lie outside the whiskers are considered outliers and are plotted with diamonds. The figure compares the results for six algorithms on five different datasets.

Fig. 5 shows the performance of six methods (SIFT, DHN, MHN, CLKN, DLKFM, and Ours) on the SkyDataV1 dataset, in terms of ACE. Skydata has RGB and infrared image pairs. In this figure, we aim to show that feature-based registration techniques such as SIFT perform poorly, whereas methods that leverage neural networks and learn representations are superior.

Fig. 4 gives detailed qualitative results of our experiments. Each row represents a sample taken from a different dataset. The columns represent inputs and results for different approaches. Target is (192 × 192) (first column) and source (second column) (128 × 128) are input image pairs. Warped (third column) is the ground-truth projection of the source to the coordinate system of the target image and Registered (fourth column) is the warped image overlayed on the target image as shown. Columns 5–10 show the registered and overlayed results for SIFT, DHN, MHN, CLKN, DLKFM, and Ours (Model A) for the given input pair. While almost all algorithms relatively well on Google Earth pair (which provides similar modalities for both target and source images), when the modalities are significantly different, as in the SkyData, Google Maps, and VEDAI pairs, the figure shows that SIFT, CLKN, MHN, DHN, and DLKFM algorithms can struggle for aligning them and they may not converge to any useful result near the ground truth (see SIFT and CLKN results), while our approach converges to the ground truth by yielding small ACE error for each of those sample pairs.

Table IV illustrates the results of using different approaches for each dataset, separately. In Table IV(e), the MSCOCO results being a single modality dataset, SIFT performs relatively better but there are cases where the algorithm could not find homography due to insufficient pairs. Google earth in (c) also has RGB image pairs but from different seasons. The SIFT algorithm is still able to pick enough salient features, therefore, the performance is still reasonable. (d) Google maps, (a) SkyData and (b) VEDAI have pairs of significant modality difference. Deep-learning-based approaches were able to perform registration often with a high number of outliers. Our approach was able to perform registration on both single and multimodal image pairs, specifically, we were able to keep the max error minimum as opposed to LK-based approaches.

V. CONCLUSION AND DISCUSSION

In this article, we introduce a novel image alignment algorithm that we call VisIRNet. VisIRNet has two branches and does not have any stage to compute keypoints. Our experimental results show that our proposed algorithm performs state-of-the-art results when it is compared to the LK-based deep approaches.

Our method's main advantages can be listed as follows.

1) Number of iterations during inference: The above-mentioned LK-based methods (after the training stage), also iterate a number of times during the inference stage, and at each iteration, they try to minimize the loss. However, those methods are not guaranteed to converge to the optimal solution and often number of iterations, chosen as a hyperparameter, is an arbitrary number during the inference stage. Such iterative approaches introduce uncertainty for the processing time, as convergence can happen after the first iteration in some situations and after the last iteration in other situations during inference. Such uncertainty also affects the real-time processing of images, as they can introduce varying frame-per-second values. Our method uses a single pass during inference to make it more applicable to real-time applications.

2) Dependence on the initial $\mathbf{H}$ estimate: In addition to the above-mentioned difference, the LK-based algorithms require an initial estimate of the homography matrix and the performance (and number of iterations required for convergence)

directly depend on the initial estimate of **H** and, therefore, it is typically given as input (hyperparameter). While we also have initialization of the weights in our architecture, we do not need an initial estimate of the homography matrix within the architecture as input.

Image alignment on image pairs taken by different onboard cameras on UAVs is a challenging and important topic for various applications. When the images to be aligned are acquired by different modalities, the classic approaches, such as SIFT and RANSAC combination, can yield insufficient results. Deep-learning techniques can be more reliable in such situations as our results demonstrate. LK-based deep techniques have recently shown promise, however, we demonstrate with our approach (VisIRNet) that without designing any LK-based block, and by focusing only on the four corner points, we can sufficiently train deep architectures for image alignment.

ACKNOWLEDGMENT

This article has been produced benefiting from the 2232 International Fellowship for Outstanding Researchers Program of TUBITAK (Project No: 118C356). However, the entire responsibility of the article belongs to the owner of the article.

REFERENCES

[1] F. Alam, S. Ur Rahman, A. Din, and F. Qayum, "Medical image registration: Classification, applications and issues," *J. Postgraduate Med. Inst.*, vol. 32, pp. 300–3007, 2018.

[2] S. Baker and I. Matthews, "Lucas–Kanade 20 years on: A unifying framework," *Int. J. Comput. Vis.*, vol. 56, no. 3, pp. 221–255, 2004.

[3] D. Barath and Z. Kukelova, "Relative pose from SIFT features," pp. 1–16, 2022. [Online]. Available: https://arxiv.org/abs/2203.07930, doi: 10.48550/ARXIV.2203.07930.

[4] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, "SURF: Speeded up robust features," *Comput. Vis. Image Understand.*, vol. 110, no. 3, pp. 346–359, 2006.

[5] Y. Bentoutou, N. Taleb, K. Kpalma, and J. Ronsin, "An automatic image registration for applications in remote sensing," *IEEE Trans. Geosci. Remote Sens.*, vol. 43, no. 9, pp. 2127–2137, Sep. 2005.

[6] D. Carreres-Prieto, J. T. García, F. Cerdán-Cartagena, and J. Suardiaz-Muro, "Performing calibration of transmittance by single RGB-LED within the visible spectrum," *Sensors*, vol. 20, no. 12, p. 3492, Jun. 2020.

[7] C.-H. Chang, C.-N. Chou, and E. Y. Chang, "CLKN: Cascaded Lucas–Kanade networks for image alignment," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 3777–3785.

[8] S. Chen, F. Yu, and X. Zhu, "Real-time registration in image stitching under the microscope," in *Proc. IEEE 9th Int. Conf. Softw. Eng. Service Sci. (ICSESS)*, Nov. 2018, pp. 907–911.

[9] C. Cucci, A. Casini, P. Marcello, and L. Stefani, "Extending hyperspectral imaging from vis to nir spectral regions: A novel scanner for the in-depth analysis of polychrome surfaces," *Proc. SPIE*, vol. 8790, pp. 1–9, May 2013.

[10] J. Delaney, M. Thoury, J. Zeibel, P. Ricciardi, K. Morales, and K. Dooley, "Visible and infrared imaging spectroscopy of paintings and improved reflectography," *Heritage Sci.*, vol. 4, pp. 1–10, Mar. 2016, doi: 10.1186/s40494-016-0075-4.

[11] X. Deng, E. Liu, S. Li, Y. Duan, and M. Xu, "Interpretable multi-modal image registration network based on disentangled convolutional sparse coding," *IEEE Trans. Image Process.*, vol. 32, pp. 1078–1091, 2023.

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

[13] B. Duvenhage, J. P. Delport, and J. de Villiers, "Implementation of the Lucas–Kanade image registration algorithm on a GPU for 3D computational platform stabilisation," in *Proc. 7th Int. Conf. Comput. Graph., Virtual Reality, Visualisation Interact. Afr.*, Jun. 2010, pp. 83–90.

[14] M. A. Fischler and R. 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, 1981.

[15] G. Fox, "The brewing industry and the opportunities for real-time quality analysis using infrared spectroscopy," *Appl. Sci.*, vol. 10, no. 2, p. 616, Jan. 2020.

[16] R. Gade and T. B. Moeslund, "Thermal cameras and applications: A survey," *Mach. Vis. Appl.*, vol. 25, no. 1, pp. 245–262, 2014.

[17] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*. Reading, MA, USA: Prentice-Hall, 2008.

[18] A. A. Goshtasby, *Image Registration—Principles, Tools and Methods* (Advances in Computer Vision and Pattern Recognition). Cham, Switzerland: Springer, 2012.

[19] C. Harris and M. Stephens, "A combined corner and edge detector," in *Proc. Alvey Vis. Conf.*, Manchester, U.K., 1988, pp. 147–151.

[20] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge, U.K.: Cambridge Univ. Press, 2003.

[21] D. L. G. Hill, P. G. Batchelor, M. Holden, and D. J. Hawkes, "Medical image registration," *Phys. Med. Biol.*, vol. 46, no. 3, p. R1, Mar. 2001.

[22] S.-M. Huang, C.-C. Huang, and C.-C. Chou, "Image registration among UAV image sequence and Google satellite image under quality mismatch," in *Proc. 12th Int. Conf. ITS Telecommun.*, Nov. 2012, pp. 311–315.

[23] M. Jaderberg, K. Simonyan, A. Zisserman, and K. Kavukcuoglu, "Spatial transformer networks," 2015, *arXiv:1506.02025*.

[24] L. Juranek, J. Stastny, and V. Skorpil, "Effect of low-pass filters as a Shi-Tomasi corner detector's window functions," in *Proc. 41st Int. Conf. Telecommun. Signal Process. (TSP)*, Jul. 2018, pp. 1–5.

[25] E. J. Kirkland, *Bilinear Interpolation*. Boston, MA, USA: Springer, 2010, pp. 261–263.

[26] H. Le, F. Liu, S. Zhang, and A. Agarwala, "Deep homography estimation for dynamic scenes," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 7649–7658.

[27] R. Lei et al., "Deep global feature-based template matching for fast multi-modal image registration," in *Proc. IEEE Int. Geosci. Remote Sens. Symp. IGARSS*, Jul. 2021, pp. 5457–5460.

[28] J.-C. Yoo and T. H. Han, "Fast normalized cross-correlation," *Circuits, Syst. Signal Process.*, vol. 28, no. 6, pp. 819–843, Dec. 2009.

[29] Y. F. L. Fuyu, "Summarization of sift-based remote sensing image registration techniques," *Remote Sens. Natural Resour.*, vol. 28, no. 2, p. 14, 2016.

[30] Z.-L. Song, S. Li, and T. F. George, "Remote sensing image registration approach based on a retrofitted SIFT algorithm and lissajous-curve trajectories," *Opt. Exp.*, vol. 18, no. 2, p. 513, Jan. 2010.

[31] T.-Y. Lin et al., "Microsoft COCO: Common objects in context," 2014, *arXiv:1405.0312*.

[32] D. G. Lowe, "Object recognition from local scale-invariant features," in *Proc. 7th IEEE Int. Conf. Comput. Vis.*, 1999, pp. 1150–1157.

[33] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vis.*, vol. 60, no. 2, pp. 91–110, Nov. 2004.

[34] Y. Luo, X. Wang, Y. Wu, and C. Shu, "Infrared and visible image homography estimation using multiscale generative adversarial network," *Electronics*, vol. 12, no. 4, p. 788, Feb. 2023.

[35] M. Magnusson, J. Sigurdsson, S. E. Armansson, M. O. Ulfarsson, H. Deborah, and J. R. Sveinsson, "Creating RGB images from hyperspectral images using a color matching function," in *Proc. IEEE Int. Geosci. Remote Sens. Symp.*, Sep. 2020, pp. 2045–2048.

[36] M. I. McCartney, S. Zein-Sabatto, and M. Malkani, "Image registration for sequence of visual images captured by UAV," in *Proc. IEEE Symp. Comput. Intell. Multimedia Signal Vis. Process.*, Mar. 2009, pp. 91–97.

[37] V. Mochalov, O. Grigorieva, D. Zhukov, A. Markov, and A. Saidov, "Remote sensing image processing based on modified fuzzy algorithm," in *Artificial Intelligence and Bioinspired Computational Methods*, R. Silhavy, Ed. Cham, Switzerland: Springer, 2020, pp. 563–572.

[38] P. Monasse, "Extraction of the level lines of a bilinear image," *Image Process. Line*, vol. 9, pp. 205–219, Aug. 2019, doi: 10.5201/ipol.2019.269.

[39] Y. Mumtaz Ahmad, S. Sahran, A. Adam, and S. Osman, "Linear intensity-based image registration," *Int. J. Adv. Comput. Sci. Appl.*, vol. 9, pp. 211–217, Jan. 2018.

[40] P. N. Pournami and G. S. Shaj, "Threshold accepting approach for image registration," *UACEE Int. J. Comput. Sci. Appl.*, vol. 2, no. 2, pp. 89–92, 2012.

[41] S. Ozer, "Similarity domains machine for scale-invariant and sparse shape modeling," *IEEE Trans. Image Process.*, vol. 28, no. 2, pp. 534–545, Feb. 2019.

[42] S. Özer, "Feature matching with similarity domains network," in *Proc. 28th Signal Process. Commun. Appl. Conf. (SIU)*, Oct. 2020, pp. 1–4.

[43] S. Özer, M. Ege, and M. A. Özkanoglu, "SiameseFuse: A computationally efficient and a not-so-deep network to fuse visible and infrared images," *Pattern Recognit.*, vol. 129, Sep. 2022, Art. no. 108712.

[44] S. Ozer, H. E. Ilhan, M. A. Ozkanoglu, and H. A. Cirpan, "Offloading deep learning powered vision tasks from UAV to 5G edge server with denoising," *IEEE Trans. Veh. Technol.*, vol. 72, no. 6, pp. 8035–8048, 2023, doi: 10.1109/TVT.2023.3243529.

[45] M. A. Özkanoğlu and S. Ozer, "InfraGAN: A GAN architecture to transfer visible images to infrared domain," *Pattern Recognit. Lett.*, vol. 155, pp. 69–76, Mar. 2022.

[46] R. Raguram and J. M. M. F. Pollefeys, "A comparative analysis of RANSAC techniques leading to adaptive real-time random sample consensus," in *Proc. 10th Eur. Conf. Comput. Vis.*, Oct. 2008, pp. 500–513.

[47] L. Ray, "2-D and 3-D image registration for medical, remote sensing, and industrial applications," *J. Electron. Imag.*, vol. 14, p. 9901, Jul. 2005.

[48] S. Razakarivony and F. Jurie, "Vehicle detection in aerial imagery: A small target detection benchmark," *J. Vis. Commun. Image Represent.*, vol. 34, pp. 187–203, Jan. 2016.

[49] J. A. Rice, *Mathematical Statistics and Data Analysis*, 3rd ed. Belmont, CA, USA: Duxbury Press, 2007.

[50] S. T. Vijay and P. N. Pournami, "Feature based image registration using heuristic nearest neighbour search," in *Proc. 22nd Int. Comput. Sci. Eng. Conf. (ICSEC)*, Nov. 2018, pp. 1–3.

[51] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 600–612, Apr. 2004.

[52] Y. Zhao, X. Huang, and Z. Zhang, "Deep Lucas–Kanade homography for multimodal image alignment," 2104, *arXiv:2104.11693*.

Sedat Özer (Senior Member, IEEE) received the M.Sc. degree from the University of Massachusetts, Dartmouth, MA, USA, and the Ph.D. degree from Rutgers University, Piscataway, NJ, USA.

He has worked as a Research Associate in various institutions including University of Virginia, Charlottesville, VA, USA, and Massachusetts Institute of Technology, Cambridge, MA, USA. He is currently an Assistant Professor with the Department of Computer Science, Ozyegin University, İstanbul, Turkiye. His research interests include pattern analysis, remote sensing, object detection and segmentation, object tracking, visual data analysis, geometric and explainable AI algorithms, and explainable fusion algorithms.

Dr. Özer was a recipient of the TUBITAK's International Outstanding Research Fellow.

Alain P. Ndigande received the B.Eng. degree from Kocaeli University, İzmit, Turkey, in 2022. He is currently pursuing the M.Sc. degree with Ozyegin University, İstanbul, Türkiye.

His research interests include deep learning, image registration, and remote sensing.