Self-Supervised Shape Alignment for Sports Field Registration

Feng Shi, Paul Marchwica, Juan Camilo Gamboa Higuera, Mike Jamieson, Mehrsan Javan, Parthipan Siva
SLiQ Labs, Sportlogiq, Montreal, QC., Canada
{feng, paul, camilo, mikej, mehrsan, parthipan}@sportlogiq.com

Abstract

This paper presents an end-to-end self-supervised learning approach for cross-modality image registration and homography estimation, with a particular emphasis on registering sports field templates onto broadcast videos as a practical application. Rather then using any pairwise labelled data for training, we propose a self-supervised data mining method to train the registration network with a natural image and its edge map. Using an iterative estimation process controlled by a score regression network (SRN) to measure the registration error, the network can learn to estimate any homography transformation regardless of how misaligned the image and the template is. We further show the benefits of using pretrained weights to finetune the network for sports field calibration with few training data. We demonstrate the effectiveness of our proposed method by applying it to real-world sports broadcast videos where we achieve state-of-the-art results and real-time processing.

1. Introduction

Image registration and homography estimation is a well studied computer vision problem with applications in image mosaic, SLAM, camera calibration, and sports field registration. Homography estimation can be categorized into single-modality homography estimation (e.g. natural image to natural image as in image mosaic [5] and SLAM [27]) or cross-modality homography estimation (e.g. edge template to natural image as in sports field registration [7, 20, 24] and robotics [19, 30]). Recently deep convolutional neural networks (CNNs) have been used to achieve some remarkable results [11] for single-modality homography estimation. However, using CNNs to address the cross-modality homography estimation problem has not been well explored. A potential reason, as suggested by [15], is that CNNs are strongly biased towards recognising texture which dominates natural images, rather than shapes which dominate edge templates.

Figure 1 illustrates our proposed method. Given a cross-modality image pair $I_A$ and $E_B$, they are fed into our homography regression network to estimate the homography transform between $I_A$ and $E_B$. We show that our approach can be run iteratively to further refine the initial homography estimate. Furthermore, inspired by [11], we also propose a method to automatically generate cross-modality training data from natural image datasets.

To the best of our knowledge, the method presented here is the first approach for cross-modality homography estimation. This method is ideal for scenarios where texture or colour information is not available as in the case of edge templates. To demonstrate the effectiveness of our method, we apply our method to sports field registration [7, 20, 24]. Unlike others, we directly estimate the homography between the field template and the image of the sports field with players without applying any pre-processing on the image. In summary, our paper has following contributions:

- An end-to-end training approach to perform self-supervised cross-modality homography estimation between natural images and their corresponding edge templates.
- A carefully designed unsupervised training strategy to train the proposed homography estimation network even with a limited training set.
- A score regression network to estimate the alignment error and control the number of required iterations during homography estimation and refinement process, in particular for applications in sports field registration.

2. Related work

Single-modality homography estimation – Homography estimation is a fundamental task in computer vision, with a variety of applications such as image mosaicing [5], SLAM [27], camera calibration [36, 3, 6], template-based tracking [9], and sports field registration [7, 20, 24]. Methods to estimate homography transformations include dense direct approaches [26, 13, 9] and sparse feature-based methods [5, 39, 27]. Both types of approaches are limited by the quality of local features [38], which depends on illumination conditions and the presence of textures, as well as the...
robustness of the objective function estimator [3, 9]. Recently, deep learning methods have also been proposed for more robust homography estimation. DeTone et al. [11] propose to train a network to regress the homography with 4-points parameters through self-supervision data mining. Nguyen et al. [28] propose a similar method using a pixel-wise photometric loss as the training objective. Both methods focus on improving the robustness and inference speed of homography estimation while maintaining an accuracy that is comparable to traditional methods.

Cross-modality homography estimation – Few methods have explored the particular issues of homography estimation between images of different modalities [37] e.g. synthetic templates, segmented images, edge maps, etc. Mutual information maximization has been demonstrated to successfully align multi-modal images [37, 9], but it requires a set of effective features. Other learning methods have been proposed to deal with choice of features. For example, Rocco et al. [32] propose a method that estimates the geometric transformations using a thin-plate spline model. However, training only with natural images shows feature bias with texture cues [15], which will not generalize well to the multi-modal setting. Introducing synthetic data may help in learning extracted features that use shape information. Geirhos et al. [15] demonstrate learning a shape-based representation by training models with images in which the texture information is randomly replaced via style transfer. Radenovic et al. [31] use edge maps to generate data for learning representations for sketch-based image retrieval. In our work, we propose a training scheme to learn shape representations that are suited for image registration through homography estimation.

Sports field registration – Early works on registering sports broadcast images [29, 17, 16] rely on local feature matching and key-frame seeking over a video. These methods typically assume the parameters to be estimated are initialized so that the transformation is close to the identity, re-using the solution from previous frames in subsequent video frames. Recently, [35, 7, 20, 24] address these limitations by learning a model that can predict good initial parameters that can be subsequently refined. These methods rely on learning the representation between image and sports field. Homayounfar et al. [20] use a deep network to perform semantic segmentation on broadcast images, which are used to estimate the parameters of the field and camera pose on a Markov Random Field with geometric priors. Sharma et al. [35] use edge images as the input representation, generating a synthetic dictionary of edge map / homography pairs for retrieval-based homography estimation. Chen and Little [7] build a camera pose database with synthetic data, and treat the problem as query-based approach with field markings from segmented image. Citraro et al. [8] use a keypoint approach to perform camera pose estimation. The method heavily depends on the players’ location. Sha et al. [34] propose an end-to-end method with both segmentation and STN [23]. The state-of-the-art method in [24] proposes a two-stage method with two regression networks: one for initial estimation and another for geometric error estimation.

3. Method

We take a similar approach as [11], with the exception that our network estimates homographies between colour images and full or partial edge maps. Many problems, such as sports field registration [24, 11] and medical image registration [4] require matching across different modalities. To this end, instead of taking two images \([I_A, I_B]\) as the network inputs, we propose to compute the edge \(E_B\) of image \(I_B\), and feed it with image \(I_A\) as inputs to our homography regression network. We follow the same spirit of deep optical flow network [12, 22] in using image warping and cost volume. However, instead of finding pixel-to-pixel displacement, our method tries to learn the alignment between
an image and an edge image.

Given an image \( I_A \) and an edge image \( E_B \), the task is to estimate their homography \( H \) with 4-points regression network. In practice, we iteratively apply image warping based on the network output of last estimated homography to find optimal alignment between them.

\[ \text{Pts}_{\text{ref}} = [(1023, 144), (256, 144), (1023, 575), (256, 575)] \] (1)

which are 4 corners of a rectangle centred in the image with a patch size of \( 768 \times 432 \), as shown in Figure 4 (left). We train the regression network to output their corresponding four-points in the edge image \( E_B \).

The regression network is shown in Figure 2. The two inputs \( I_A \) and \( E_B \) are fed into two separate processing streams \( A \) and \( B \). Their outputs are concatenated as a cost volume, which goes through two ResNet building blocks, \( \text{conv}4_x \) and \( \text{conv}5_x \). A linear regression layer is followed to output 4 points.

Score regression network – Although iterative refinement improves alignment for inputs with large initial displacement, each iteration has extra computational cost. For real-time applications, we can improve speed by stopping the refinement process early once the homography provides ‘good enough’ alignment between image and edge map.

Therefore, we add a Score Regression Network (SRN) that estimates the quality of the homography output. The ground truth score is calculated based on the intersection-over-union (IoU) of the perturbed image and the ground truth one. Since we would like to distinguish among IoU values close to 1.0, we use \( \text{IoU}^3 \) as the ground truth score for the SRN. Training loss for the SRN is the mean squared error between the output score and the ground truth score. As shown in Figure 2, the SRN network is similar to the regression network, with a score sub-branch \( C \).

3.1. Architecture

As shown in Figure 2, we create two separate, non-identical processing streams (\( A \) and \( B \)) for input image \( I_A \) and edge image \( E_B \). We merge them at a later stage, and then split them again into two separate branches, a 4-points homography regression branch (\( D \)) and a score regression branch (\( C \)).

We use ResNet [18] as the network backbone and adopt two strategies to improve the efficiency. First, since the input edge image \( E_B \) has only 1 channel with a lot of zero pixels, we use a shallow network (3-layers, \( B \) as in Figure 2) with few channels for edge input processing stream. Second, we use a relatively deep network (1 input layer and 2 stacked ResNet building blocks, \( A \) as in Figure 2) for image input processing stream. We also observed that several iterative refinements can improve homography accuracy if there are large displacements between two inputs. Thus, we design our method in a way to improve the running speed with multiple iterations. To this end, for an input pair, we only need to run the whole network with image input processing stream \( A \) once. If more iterations are needed, we apply the output homography to warp the input edge image \( E_B \) and reuse the features of stream \( A \).

Regression network – We follow the recent trends [3, 11, 24] to represent the homography, \( H \), with 4-points parameters. However, instead of using point offsets [11] or normalized coordinates [24], we use 4 fixed points from the reference image \( I_A \). Specifically, considering an image of size \( 1280 \times 720 \), we use:

\[ \text{Pts}_{\text{ref}} = [(1023, 144), (256, 144), (1023, 575), (256, 575)] \] (1)

3.2. Training sample generation

As shown in Figure 3 (for simplicity, we omit the SRN score branch), for each training image \( I_A \), we first calculate its Canny edge \( E_A^{(0)} \), and feed them as one training sample. In this case, the expected network output is the four points \( \text{Pts}_{\text{ref}}^{(0)} \) given in Equation 1, scaled by the network input size. Then, we randomly perturb \( \text{Pts}_{\text{ref}}^{(0)} \) into reference points \( \text{Pts}_{\text{ref}}^{(k)} \) and use them to calculate a homography \( H(k) \) and a perspective-transformed edge image \( E_A^{(k)} \). The perturbation process is repeated several times per image to create multiple training samples. Including the non-warped training samples (edge \( E_A^{(0)} \) ) in the training data is a key element of the success of the method, since it helps the network to learn the visual correspondence of the edge features.

We generate 7 warped edge maps per image, for a total of \( N = 8 \) training samples per training image. The perturba-
Figure 3. Illustration of our training data from one image. For each training image we calculate its edge image using Canny edge detection. Then, we stack the grayscale image with the edge image channel-wise. We calculate several homographies based on perturbing 4 fixed points and use them to get warped edge images. We stack them with the original grayscale image and feed them into the network to train.

Algorithm 1, data augmentation – Our first method is similar to [11]. However, instead of cropping two overlapped images within the same image border, we apply a large perturbation to a single image, as shown on the left of Figure 4. Specifically, we use the original image and apply a homography to warp its edge into the second training sample. The homography is decided by perturbing each point of \( Pts_{\text{ref}} \) (Equation 1) with up to a scale of \((0.35 \times 768, 0.35 \times 432)\) in \((x, y)\) dimension. Among all 7 transformed training samples, the perturbation is also performed to simulate camera translation and zooming.

Algorithm 2, data augmentation – Our second approach is as shown on the right of Figure 4. In this approach we aim to avoid the warping outside of the image border. Given an image \( I_A \) with network input size of 256 \( \times \) 256, we calculate 7 homographies by using 4-points perturbation. We first select a square inside the image with a random location. The size of the square is decided by a range of \((0.7 \times 256, 0.92 \times 256)\). Then, we perturb the 4 corners of the square in a range of \([-20, 20]\) to obtain the homography. The perturbation is designed to ensure that warped image remains within the original image border. The images are finally warped from \( I_A \) with the homographies. After generating all 8 images, we randomly choose one image \( I_A^{(k)} \) to create 8 training samples by combining it with all edge images calculated from all 8 images. The ground truth 4-points are calculated from the homographies.

### 3.3. Homography inference

**Iterative refinement** – The network aligns image and edges well if the initial displacement between the two modalities is modest. For larger displacements, the output homography tends to be less precise. Therefore at test we use an iterative refinement approach where each new iteration is refined with the output homography from the last iteration. The initial regression pass estimates the coarse homography between the input image \( I_A \) and edge map \( E_B \). The output homography is used to perform perspective transformation on \( E_B \). Then, we feed the warped \( E_B \) and \( I_A \) as network input for the next iteration. We repeat the same process for subsequent iterations until the score from SRN is larger than a threshold or it reaches the maximal number of iterations.

### 3.4. Implementation details

For the backbone of our homography network we use ResNet [18] (tested with ResNet-18 and ResNet-50). The
input size is $256 \times 256$. We use smooth-L1 loss for 4-points regression and mean squared error loss for SRN score. The network is trained jointly with both losses and all networks are trained from scratch.

**Hyperparameters** – We train our networks with Adam optimizer [25], default parameters $\beta_1 = 0.9$, $\beta_2 = 0.999$ and an initial learning rate of 0.03. We decrease the learning rate by a factor of 10 after 5 and 10 epochs, and stop training after 15 epochs. We use 2 GPUs to train the network with a batch size of 256 for ResNet-18 and 128 for ResNet-50. For each epoch, since we use 8x number of images, our network takes more time to train.

We use OpenCV Canny edge detector to calculate edge images. For both training and testing, we calculate maximal image pixel value, and take its 0.3 and 0.7 as lower and upper Canny detector threshold. We apply a $5 \times 5$ Gaussian filter on image before calculating edge.

### 4. Experiments

**Evaluation protocols** – Our first experiment follows the similar evaluation method as [11]. We use ImageNet [10] training images to train our network and use all ImageNet test images to create 100,000 pairs of images as homography testing data. For each image, we use Algorithm 1 (Section 3.4) to create a ground truth homography and warp the image. The perturbation for each point is set to a random value of $[0.1 \times 768, 0.32 \times 768]$ in x-axis and $[0.1 \times 432, 0.32 \times 432]$ in y-axis. We fix the number of refinement iterations at 4.

We compare our network performance with two baselines, ORB [33] and AKAZE [14] detector. For both baselines, we use OpenCV implementation, and the homography is estimated with the robust feature matching RANSAC method. In the cases where the network or either of the baseline methods fails, we output identity matrix as homography. We run the baseline methods with the original image size. For network, we keep the original image $I_A$, and calculate Canny edge maps $E_B$ from warped image $I_B$. We resize them to $256 \times 256$ and feed them as network input. Note, when calculating Canny edges, a proper mask is applied to remove the border edges due to the warping out of the image border, because we do not want the network to take advantage of the border edges.

**Results** – Figure 5 shows our testing results over 100,000 testing data. Again we follow [11] in reporting mean average corner errors over 4 points. The homography network performs better than both feature-matching baselines, even though the baselines operate on a single modality. ResNet-50 has smaller errors than ResNet-18. As expected, the network gives better results with more refinement iterations, especially from the first iteration to the second one. This may be because the initial iteration have solved the largest image-edge displacement, and thus alignment is mostly complete after the second run.

![Figure 5. Homography estimation comparison with Mean Average Corner Errors (logarithmic scale). Bars in blue denote ResNet-18 results while orange ones are ResNet-50 results](image1)

![Figure 6. Hybrid homography results (logarithmic scale)](image2)

**Hybrid homography estimation** – Since the first iteration of the network solves the largest errors, one may ask if it can lead to better results when combined with traditional ORB or AKAZE detector. Therefore, we develop a hybrid method. We run the ResNet-18 network with $I_A$ and $E_B$ first. Then we take its output to warp the image $I_B$, and use the result image and $I_A$ to estimate final homography with feature detectors. To provide a baseline comparison, we also run ORB twice and AKAZE twice.

The results of hybrid methods are shown on Figure 6. Both hybrid methods give very good results, especially network-AKAZE. This is a very interesting observation and may give us a robust method in homography estimation for various applications, such as wide baseline matching or cross-examination with both network and feature matching. Note: our method is not limited to cross-modality, and an image-to-image model can be trained with same approach.

**Illustration of network iterations** – To demonstrate that the network actually performs shape matching over several iterations, we run our ResNet-50 network on some challenging homography estimation image pairs from [1]. The results are shown in Figure 7. The “Boston” image pair (first row) show a large translation and relatively small overlap between the views in the first two images, and it takes 4 iterations for the network to find the optimal shape matching. The “Boat” image pair (second row) on the other hand have
Figure 7. Edge alignment visualization of our method on homography estimation dataset. The first two columns are image pairs (a and b), and columns 3 to 6 are results of inference iterations 1 to 4, respectively. The edges of image b gradually match the image a over the iterations even in the presence of large translation (Row 1) as well as rotation and scale change (Row 2). The bottom row shows a failed case where the edges match the image with local maxima. Best viewed in color.

Figure 8. Edge alignment visualization of our method on homography estimation dataset. The first two columns are image pairs (a and b), and columns 3 to 6 are results of inference iterations 1 to 4, respectively. The edges of image b gradually match the image a over the iterations even in the presence of large translation (Row 1) as well as rotation and scale change (Row 2). The bottom row shows a failed case where the edges match the image with local maxima. Best viewed in color.

5. Sports field registration

In the previous section, we demonstrate that the network can be trained to align images based on correspondences in underlying shape information. This type of network could potentially be applied to many registration problems where only partial shape information is available. Here, we demonstrate its application on sports field registration [20, 24]. We hope that this will shed some light on future work in this area.

5.1. Problem statement

As shown in Figure 8, the sports field registration can be addressed as shape matching. Given an image, we want to map its location on the field template. In fact, the relationship between an image and the template can be represented with a homography \( H(k) \). Let us assume that we have the ground truth homography between an image and a template, we want to train the network to output the homography with 4-points method. To this end, we can use the ground truth homography to perform perspective transformation on the template, and treat the warped template (Figure 8, bottom left) as foreground image edge with the background players and non-field portions removed. Thus, we could use the similar method as last section to learn the regression network.
recent studies in deep learning and computer vision achieve impressive results \([7, 20, 24, 34]\) in sports field registration. Among them, \([24]\) is closely related to us. It relies on two decoupled regression networks. Interestingly, its second error network also stacks the image with the warped template as input. However, we approach the problem from a very different perspective. Instead of specifically learning to minimize errors with only one perturbation, we train the network to learn shape matching with the ground truth sample and several perturbed noise samples. As a result, our network only needs up to 4 iterations to achieve the optimal results during inference, while \([24]\) often runs the optimization for 400 iterations.

### 5.2. Sports registration network

We use similar method as \([24]\) in using two decoupled regression networks. We train a coarse model \(M(0)\) (without SRN branch) to find approximate homography \(H(0)\) with an input pair of image and full template (the top image as shown in Figure 8). Then, we train a second model \(M(1)\) to do several refinements based on \(H(0)\).

We feed image and full field template as input to train model \(M(0)\). We perturb the full template with few pixels (less than 20) to avoid always training the network with the same one. For model \(M(1)\), we prepare our training data in the same way as Algorithm 1. Specifically, we use one warped template (decided by ground truth homography) and 7 warped templates with perturbation. Each of the 8 templates is fed with a copy of the field image, forming 8 training samples. During inference, we first use model \(M(0)\) with an input pair of image and full template to estimate the initial homography. We then perform the iterations based on initial estimate. Note, to perform the inference with video, we could skip the model \(M(0)\) for subsequent frames, and use the homography from last frame to do the refinement.

### 5.3. Experiments

To evaluate our method, we follow state-of-the-art methods \([20, 24, 34, 7]\) in using WorldCup soccer dataset \([20]\), Hockey dataset \([20]\) and Volleyball dataset \([7]\).

**WorldCup soccer dataset** – The WorldCup soccer dataset is very small, which has only 209 training images and 186 testing images. We follow \([24]\) to split 209 images into 170 training data and 39 validation data. The testing is evaluated with the best model based on the performance on 39 validation images. To increase the data, the training/validation images are horizontally flipped.

To reduce the risk of overfitting on such small data, in addition to using ResNet-18 with fewer parameters, we develop a method to increase the randomness from the training data. For each training image, we use Algorithm 2 (Section 3.2) to generate an extra 7 images. We shuffle all generated images plus the original ground truth data and use Algorithm 1 (Section 3.2) to create 8 training samples per image. We use all of them as a training epoch, and perform such operation for each epoch. Note, with data augmentation, each epoch includes around 20K training samples, and the samples are different in each epoch.

**Volleyball dataset** – The Volleyball dataset \([7]\) is collected from the volleyball action recognition dataset of Ibrahim et al. \([21]\). The dataset includes total 47 games, with 10 images per game. We follow \([7]\) in choosing 24 games for training/validation and 23 games for testing. We follow the same training/testing strategies as WorldCup soccer dataset in using 22 games as training data and 2 games as validation data to find the optimal model for testing.

**Hockey dataset** – The Hockey dataset \([20, 24]\) has a large variety of data with 1.67M images. In this test, we only randomly sample 3000 ground truth images from the dataset and combine them with the synthetic training data. We col-
Table 1. Comparison with state-of-the-art results. Best results are shown in bold. Ours$^1$ represents the results of pretrain model finetuned with our homography network model, and Ours$^2$ are from no pretrain model.

<table>
<thead>
<tr>
<th>Method</th>
<th>Whole IoU</th>
<th>Part IoU</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>mean</td>
<td>median</td>
</tr>
<tr>
<td>Soccer</td>
<td></td>
<td></td>
</tr>
<tr>
<td>[20]</td>
<td>83</td>
<td>–</td>
</tr>
<tr>
<td>[8]</td>
<td>90.5</td>
<td>91.8</td>
</tr>
<tr>
<td>[34]</td>
<td>88.3</td>
<td>92.1</td>
</tr>
<tr>
<td>[24]</td>
<td>89.8</td>
<td>92.9</td>
</tr>
<tr>
<td>Ours$^1$</td>
<td>92.76</td>
<td>94.1</td>
</tr>
<tr>
<td>Ours$^2$</td>
<td>93.16</td>
<td>94.87</td>
</tr>
<tr>
<td>Hockey</td>
<td></td>
<td></td>
</tr>
<tr>
<td>[20]</td>
<td>82</td>
<td>–</td>
</tr>
<tr>
<td>[24]</td>
<td>96.2</td>
<td>97.0</td>
</tr>
<tr>
<td>Ours</td>
<td>95.49</td>
<td>96.39</td>
</tr>
<tr>
<td>Volleyball</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ours</td>
<td>96.03</td>
<td>97.29</td>
</tr>
</tbody>
</table>

Note: [8] also reports better results with manually annotated player locations as keypoints for homography computation. One very important observation is pretrain model performs better than no pretrain model. In addition, it takes 420 training epochs to find best pretrain model with validation data vs. 1600 training epochs for no pretrain model. This demonstrates the effectiveness of our method and its potential usage on transfer learning on limited data.

The Volleyball dataset shows camera views covering most of the field. Our method performs better than the baseline method [7] with almost perfect results on IoU$_{part}$. For Hockey dataset we achieve better performance on IoU$_{part}$, and a little worse on IoU$_{whole}$ than [24]. Note, instead of using 1.67M training data as [24], we only use 3K broadcasting data plus synthetic data.

We also evaluate our SRN model on [24]. We first run [24] code [2] on WorldCup dataset with its default 400 iterations, and then use our SRN model to decide the output homography with a score threshold of 0.97. Our method reduces the iterations to 299.4 without loss of accuracy with mean IoU$_{part}$ = 95.3 and mean IoU$_{whole}$ = 90.0.

**Inference** – For evaluation, we split the model into two models, a 4-points regression model and a SRN model. For up to 4 iterations per image, we only run the image stream branch, A in Figure 2. The Nvidia TensorRT is used for accessing the features. The 4-points model runs first, and its output homography is used to warp the template as the input for the edge stream branch, B in Figure 2, to evaluate the score from the SRN model. Our method can achieve over 100fps running on an Nvidia Tesla T4 GPU using FP16.

### 6. Conclusions

We presented a new method to train an end-to-end CNN to perform cross-modality homography estimation. Our training approach does not require any labelled data and we have shown the benefits of combining our method with traditional feature based registration methods to achieve better results. Testing on sports field registration datasets shows the effectiveness of the end-to-end self-supervised network to achieve state of the art results for cross-modality image registration. Experimental results indicate that our method outperforms state-of-the-art registration techniques using only a small number of labelled samples of about 240 images.

### Acknowledgement

We thank Wei Jiang and Kwang Moo Yi from the University of British Columbia for helping and providing code for evaluating the experimental results. We also thank Jim Little from the University of British Columbia for providing the Volleyball dataset.
References


[29] Jens Puwein, Remo Ziegler, Julia Vogel, and Marc Pollefeys. Robust Multi-view Camera Calibration for Wide-baseline


