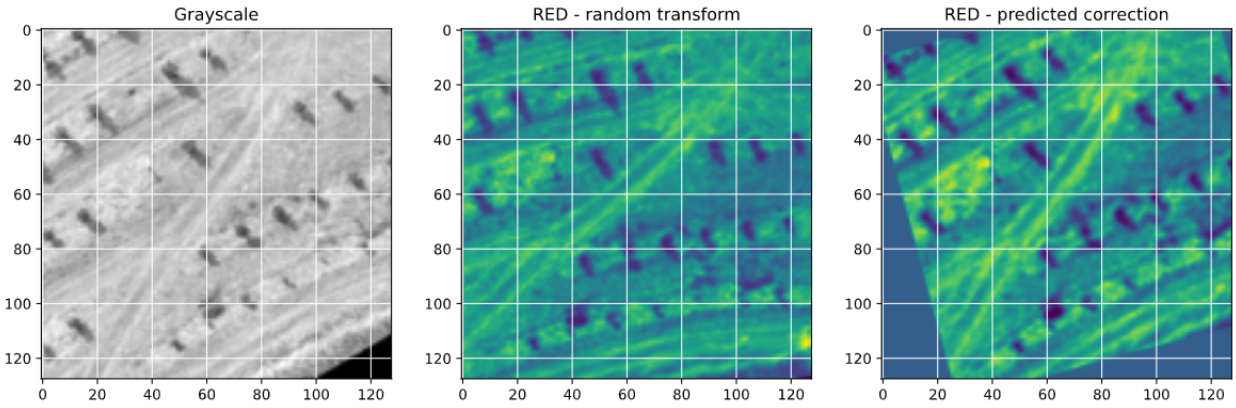
Computer Vision Engineer Assignment

Multimodal Image Registration

Jan Marais



Summary

The task is to align a narrowband RED image with its corresponding RGB image. The approach taken in this project is to stack the misaligned pair channel wise and feed it to a CNN to predict the horizontal and vertical offsets of the corners. From these offsets we can calculate the affine transformation that will transform the RED image so that its corners are in the adjusted locations and thus aligned with the RGB image.

The CNN is trained and evaluated on artificially misaligned images for which we know the ground truth. The mean average corner error (MACE) after the estimated transformation is applied on the validation set is 11 pixels, compared to the MACE of the misaligned pair of 11. Substantial improvements are likely to be gained by training on more data, larger images and optimised hyperparameters.

What follows in this report is, firstly, a discussion on why this approach was chosen compared to other possible solutions. Then we describe the approach in more detail followed by a discussion of the results and then conclude by proposing ways of improving it.

Introduction

Image registration is a common task in computer vision applications. There are many approaches that can be used. The standard approach is to locate corresponding points in both images and find the transform so that these points are aligned. Instead of detecting points, one can also search for the transform that maximises some image similarity metric.

The challenging part of our task is to deal with different modalities of the images. This makes it difficult to determine local and global similarities between the images. To overcome this one often has to resort to ad hoc pre or post processing pipelines to make the image pair look more similar or to describe them invariant to the modality. It is hard to devise such pipelines that are applicable to the complete set of input images.

The more modern approach is to learn parts of the registration pipeline (or the whole) from the data. The benefit of this approach is that one does not need to handcraft image features and descriptors since this is done implicitly by the learned model. In addition, a CNN provides a lot of flexibility in modelling various input-output systems and thus this approach can potentially be used with various modalities, scenes and transformations with minimal overhead.

There are also downsides to using a CNN to learn the optimal transformation is. Firstly, it is heavily dependent on quality and large amounts of training data. It also requires heavy computing resources. Both of these points implies that it comes with a lot more development and maintenance resources than the classical approaches. Lastly, one can't avoid manually tuning the system - although the CNNs can automate the feature descriptions, one needs to consider a plethora of model and optimiser hyperparameters in order to find the optimal way of doing so.

Nevertheless, given we have enough compute and data, the learned CNN is most likely going to give the best results with the added benefit of scalability to other datasets.

On a personal note: in a previous Aerobotics application I demonstrated the classical approach to image registration and thought it would be a good idea to try the modern approach this time around.

Approach

Here we discuss how a CNN was developed and used to align a pair of RGB-RED images. We describe how an artificial dataset was created for training the model, the major modelling design decisions and how predictions are made.

Data Generation

The aim is to have many examples of misaligned RGB and RED image pairs for which we know the true transformation. To create this, we apply the following algorithm:

For each survey:

a. Load full RGB and aligned RED image.
b. Resize RGB to RED dimensions (H, W).
c. Apply random rotation to both images at the same angle.
d. For *i* in [1,2,...,*images_per_survey*]:
    i. Sample a random point in (H, W).
    ii. Crop a square of random size *sz* (in predetermined range) with this point as its center from the RGB image. Save this RGB crop.
    iii. Sample a random rotation angle within a predetermined range and rotate the RED image by this angle.
    iv. Sample random vertical and horizontal translations and translate the RED image by these distances.
    v. Crop a square of size *sz* at the point of *(c.i)* from the transformed RED image and save this cropped image.

We end up with examples of this form: (RGB crop, transformed RED crop, transformation parameters). By transforming before cropping we minimise the loss of pixel data caused by

the transformation. We also reject crops with too much empty space. The random sizing of the crops helps the model to deal with different levels of zoom.

The current model was trained on a dataset generated from the training surveys using 1000 images per survey, *sz* in (300, 600), angle in (-15, 15), translation in 10% of size. We restricted the experiment to small affine transformations to match the use case, but one can also experiment with perspective and dense transforms.

Modelling

The developed CNN takes a grayscale conversion of the cropped RGB image and the crop of the misaligned RED image, stacked channel wise as input. Grayscale was used for simplicity and size and resource constraints. Also chosen for the same reason, we set the input image size to 128x128.

Instead of explicitly predicting the transformation parameters, we output 8 scalars parameterising the horizontal and vertical shift the corners of the RED image needs to undergo to be aligned with the RGB image. This representation was chosen because then all the output values are in the same range.

A standard ResNet18 architecture is used for the CNN, but instead of a fully connected layer at the end, one more convolutional layer was added of kernel size (1, 1) and stride 2, to output a 2x2x2 tensor to match the corner offsets:

dx1	dx2
dx4	dx3

dy1	dy2
dy4	dy3

This format was chosen to preserve spatial awareness of the outputs and reduce the number of parameters. This representation can also be used for any transformation, not only affine as we are using here.

A smooth L1 loss function was used to be more robust to outliers. The model was trained using the Adam optimiser and the OneCycle learning rate scheduler, with 64 examples per batch and for 20 epochs. The learning rate finder approach corresponding to the OneCycle policy was used to find the optimal max learning rate to be 0.001. This was trained on 1060 GPU (laptop version) and 16GB of RAM. It took around 25 minutes to complete.

Evaluation and Inference

We use the data generated from the validation survey images as the validation set. We report the performance on the validation set using the Mean Average Corner Error (MACE) which is the average euclidean distance between the corners of two images. This metric provides an intuitive description of how far off the alignment is.

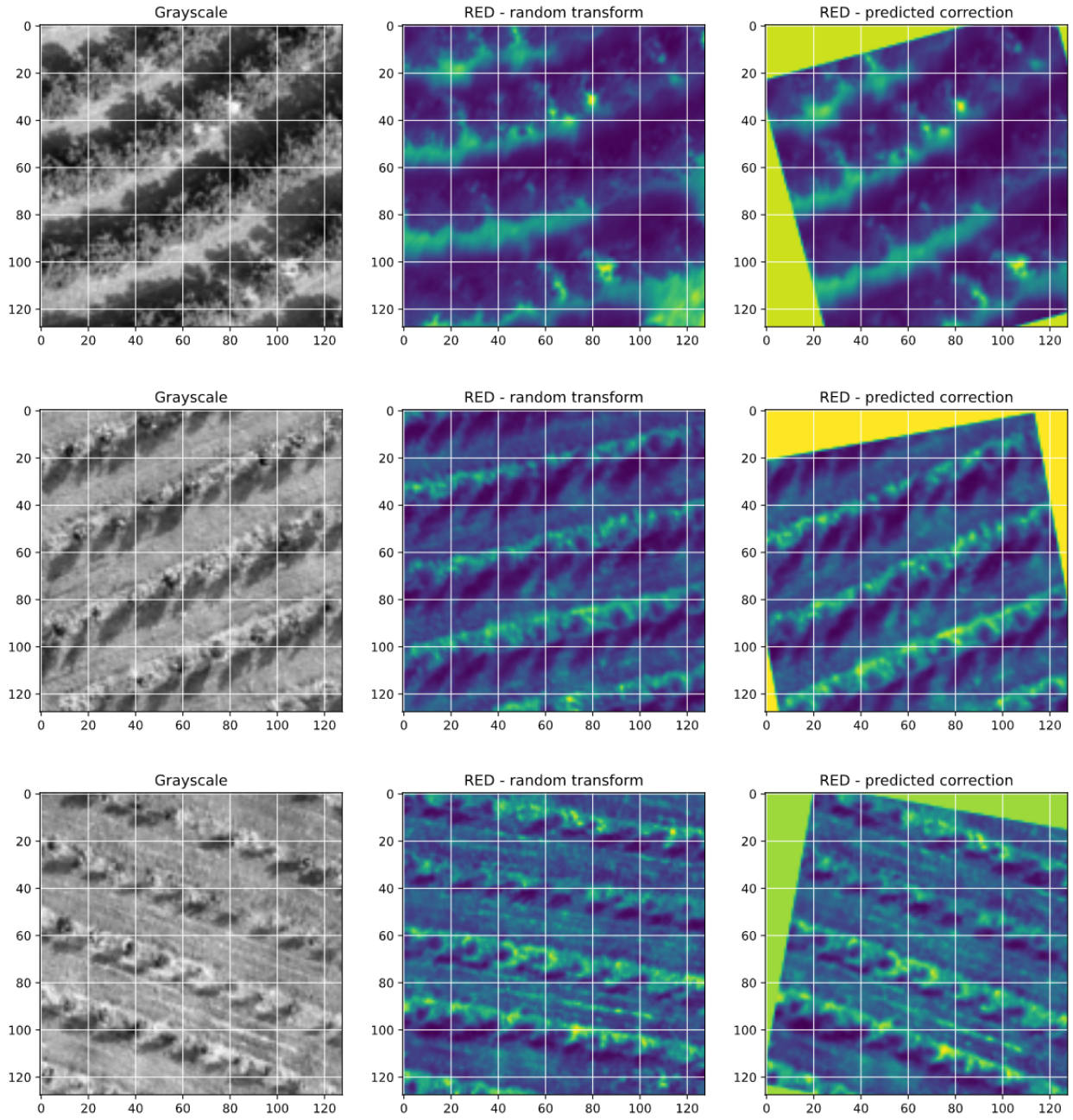
From the offsets produced by the network we can choose any of the three corner points and find the affine transformation that will map the corners of the RED image to the points with its predicted offset. If one of the corner offsets is wrong, the whole transformation will be wrong. To be less sensitive to this scenario, we can find the transformation for all combinations of 3 corners and calculate an average transformation. If one wants to align the full survey image, we can either extract a grid of or random crop patches from the full image and predict corners offsets on those. We can use the corners of all the crops and the transformed locations as the source and destination points to solve the transformation using the RANSAC algorithm.

Results

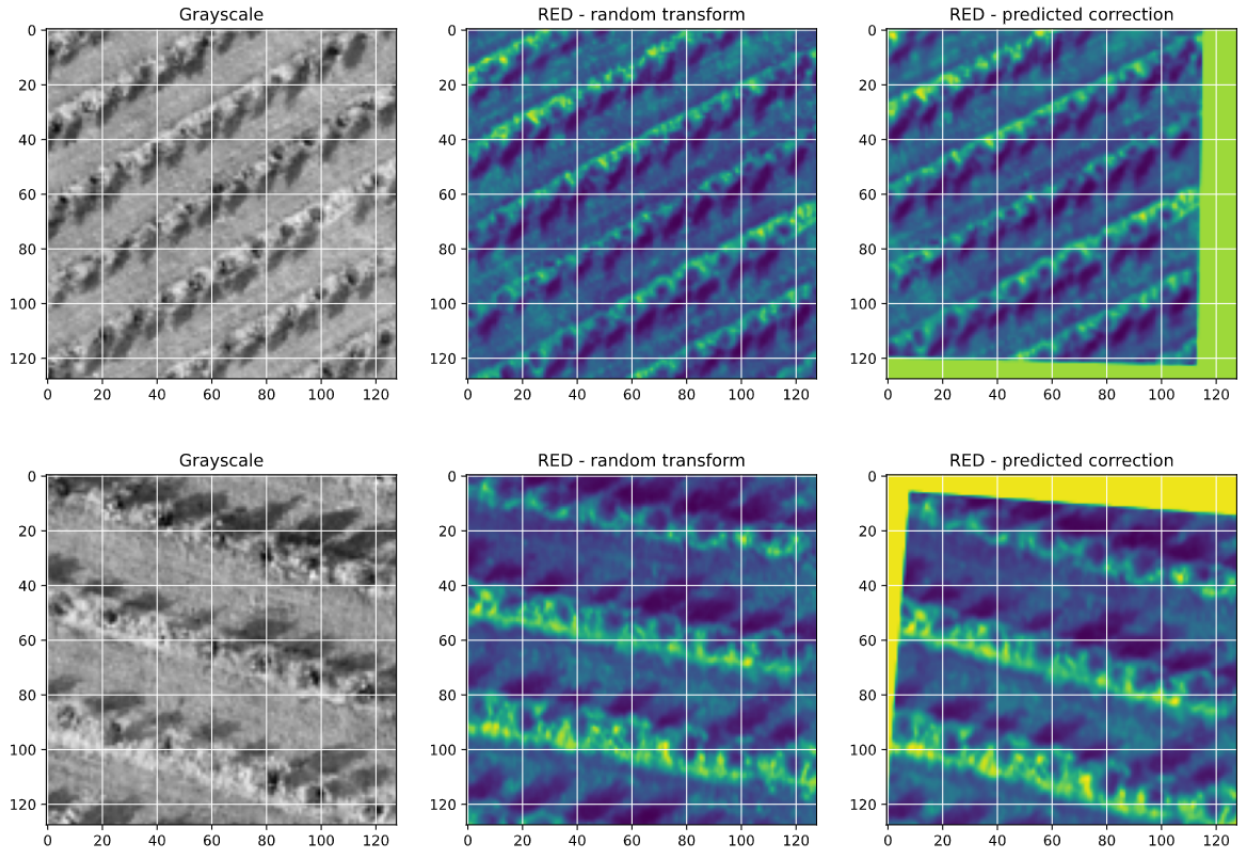
The MACE of the misaligned pair in the validation set is 61.62. After the pairs were aligned using the model described above, the MACE became 11.1.

Below are some examples of successful transformations, as well as failure cases:

Good predictions:



Bad predictions:



Just from looking at a sample of the results, it seems that the model struggles when there is a lot of uniformity in the image. But it does quite well besides that. This needs further investigation. For this approach to really be useful we will need near perfect results. In the next section we list some ways this can be achieved.

Next steps

The align from multiple image patches pipeline (described in *Evaluation and Inference* still needs to be developed.

Below is a list of avenues that can be explored from a modelling perspective which are most likely to improve the results.

- Add augmentation to transform pipeline
- Use RGB as input
- Larger model

- Modern model layers and training tricks.
- Hyperparemeter optimisation.
- More data (from more surveys)
- Larger image sizes
- Output perspective fields instead of corner offsets.

With a more thorough investigation of the results we will be able to better understand the limitations and how to overcome them.