Learning Cross-Spectral Point Features for Visual Navigation with Task-Oriented Training

by

Mia Thomas

A thesis submitted in conformity with the requirements
for the degree of Master of Applied Science
Graduate Department of Aerospace Science and Engineering
University of Toronto

Abstract

Learning Cross-Spectral Point Features for Visual Navigation with Task-Oriented Training

Mia Thomas
Master of Applied Science
Graduate Department of Aerospace Science and Engineering
University of Toronto
2025

Unmanned aerial vehicles (UAVs) enable operations in remote and hazardous environments, yet the visible-spectrum, camera-based navigation systems relied upon by UAVs struggle in low-visibility conditions. Thermal cameras, which capture long-wave infrared radiation, are able to function effectively in darkness and smoke, where visible-light cameras fail. This thesis explores learned cross-spectral (thermal-visible) point features as a means to integrate thermal imagery into established camera-based navigation systems. While existing approaches focus training on image regions with limited appearance variation across spectra, we take a task-oriented approach to supervision. Our feature network feeds into a differentiable pipeline that performs homography estimation between thermal and visible-spectrum images. We compare different task-based loss formulations using the matching and registration outputs of this pipeline. Our selected feature model, trained with a matching-based loss, outperforms baseline methods on cross-spectral image matching and registration, both with a specialized homography estimation pipeline and its classical counterpart.

Acknowledgements

Throughout this journey, I have had the privilege of being supported by a number of people who truly believe in me, and I would like to express my heartfelt thanks to all of them. First and foremost, I would like to thank Prof. Jonathan Kelly for the opportunity to work on such an interesting project in an emerging field and for facilitating my internship at NASA's Jet Propulsion Laboratory, which proved to be the highlight of my degree. Under your guidance, I have grown both academically and personally. Thank you for your encouragement and insight. I would like to extend my thanks to all STARS Lab members for fostering a fun and supportive environment. In particular, I owe a debt of gratitude to Trevor Ablett, whose mentorship helped me navigate the most challenging aspects of graduate school. Your kindness, encouragement, empathy, and technical input set me on the right path. I am immensely grateful to Katie Allison for pushing me over the finish line with her thoughtful and constructive feedback. Thank you also for your friendship—it has been a pleasure to work alongside you for the past few years. The Robotics Institute has been central to my experience at the University of Toronto. This community significantly enriched my academic experience, but more importantly, it introduced me to a number of talented, knowledgeable, and kind individuals. Thank you all for your positivity and friendship. Finally, my deepest gratitude goes out to my family, whose confidence in me was a source of strength when I needed it. In particular, I would like to thank my parents for their support and understanding, my aunt Shauna for her mentorship and unshakeable optimism, and my sister Katie for always being there to listen and cheer me on.

Contents

List of Tables

List of Figures

Notation

$\mathbf{a}$	Column vector
$\mathbf{A}$	Matrix
$\mathbf{A}^i$	Instance i of quantity $\mathbf{A}$
a, A	Scalar quantities
a_i, a_{ij}	Elements i and (i, j) of vector $\mathbf{a}$ and matrix $\mathbf{A}$, respectively
a_{K}, A_{K}	Subtype K of quantities a, A, respectively
${}^{b}\mathbf{H}_{a}$	Homography that maps point $\mathbf{q}$ from image $\mathcal{A}$ to $\mathcal{B}$, such that ${}^{b}\mathbf{q} = {}^{b}\mathbf{H}_{a}\,{}^{a}\mathbf{q}$
$\hat{(\cdot)}$	Estimated quantity
$\bar{(\cdot)}$	Normalized or averaged quantity

Chapter 1

Introduction

Unmanned aerial vehicles (UAVs) play a vital role in a wide range of applications such as search and rescue, wildlife monitoring, firefighting, agriculture, building inspection, and mapping [1]. As UAVs explore increasingly extreme environments, robot navigation systems must adapt to operate in more perceptually challenging conditions. A standard tool for robot navigation in GPS-denied environments is camera-based or *visual* localization, in which a robot's position and orientation (pose) are estimated within an image-based map. At present, these systems primarily use RGB or *visible-spectrum* cameras.[1] Despite significant research efforts, current visual navigation systems remain sensitive to lighting variations and face performance limitations in low-visibility conditions. Dependence on favorable lighting impacts UAVs in particular, as weight and power constraints often prevent the addition of on-board lighting. Thermal cameras, however, measure long-wave infrared (LWIR) radiation *emitted from* the environment and therefore do not depend on external illumination, enabling operation even in complete darkness. In addition, LWIR radiation penetrates atmospheric obscurants such as smoke, fog, and dust. Visual navigation systems could therefore benefit from the use of thermal imaging when facing poor visibility, potentially enabling operation in environments that are currently untenable for visible-only systems.

1.1 Thermal Cameras in Visual Navigation Systems

Thermal imagery on its own can be useful for motion estimation, but additional benefits can be realized from using these sensors in concert with existing visible-spectrum systems. Thermal-only algorithms, often combined with inertial measurement unit (IMU)

[1] We distinguish *visual* as relating to cameras of any type, whereas *visible-spectrum*—or *visible* for short—refers to light that is perceptible to the human eye.

information, have been successfully deployed in incremental or *relative* localization systems, such as odometry [2–7] and simultaneous localization and mapping (SLAM) [8–13]. These implementations have demonstrated relative motion estimation in complete darkness and in the presence of smoke. Combining information from both spectra could therefore be particularly impactful for *absolute* localization, in which a map is provided *a priori*. There exists an abundance of visible-spectrum maps from publicly available datasets and georeferenced images from resources such as Google Earth [14]. Absolute localization may also be crucial for recovering from non-uniformity corrections (NUCs), a periodic online recalibration process that reduces accumulated noise in the thermal camera feed.

Current work towards a cross-spectral visual localization stack has explored place recognition [15, 16] and image registration [17]. Place recognition can serve as a precursor to localization; it matches the current sensor input to a previously-observed map scene without requiring a prior location estimate. Image registration is the process of aligning two images by estimating the parameters of a predefined 2D geometric transformation. In approximately planar scenes (i.e., with negligible depth variation from the observer's perspective), image registration can be used for map building or pose estimation. For example, a full 6-degree-of-freedom (DOF) pose can be derived from a homography model if scale is known through another source, such as map metadata or IMU measurements.

In an effort to advance the integration of thermal imagery into established visual navigation systems, we consider a cross-spectral extension of a general-purpose computer vision tool: the point feature. A point feature consists of a 2D interest point and an associated descriptor vector. By comparing point features across different viewpoints and scenes, one can perform a number of visual navigation tasks such as odometry, SLAM, place recognition, scene reconstruction, and more. As a use case, we will focus on improving feature-based methods for registering thermal and visible-spectrum images.

Within robotics, learned methods have emerged as the favored approach to handle cross-spectral data. Thermal and visible-spectrum cameras capture fundamentally different information about the environment. This creates a large appearance gap or *domain gap* between the two sensor outputs, making the development of cross-spectral computer vision algorithms very challenging. Compared to other fields that handle cross-spectral image data, such as medical imaging and remote sensing, robotics applications impose additional design objectives onto this problem. Robotic systems typically handle viewpoint changes algorithmically and prioritize computational efficiency, often with the goal of closed-loop operation. Handcrafted feature algorithms for cross-spectral image pairs, although meant to provide a lightweight solution, are too slow to enable real-time oper-

ation [18]. Learned features, on the other hand, offer a considerable speedup and have been shown to significantly outperform handcrafted features in the face of large appearance and viewpoint variations in the visible-spectrum domain [19–21]. Therefore, we opt to explore learned features as a means to bridge the domain gap for thermal-visible image registration.

1.2 Contributions

Given that point features lack a strict definition, generating a supervisory signal for feature learning can be challenging, even if ground-truth correspondences are known. This problem is exacerbated by the photometric differences between thermal and visible-spectrum imagery. Existing approaches focus training on image regions where the thermal-visible appearance gap is relatively small, which excludes potentially useful features in image regions with a large domain gap. Therefore, we propose a task-based supervision method, aiming to guide the network in identifying useful cross-spectral features regardless of their photometric differences.

In our framework, we first run our feature network on a thermal image and visible-spectrum image that are geometrically related by a random but known homography. The network response is then fed into a differentiable homography estimation pipeline. Using the ground-truth transform, we apply losses to the intermediate and final outputs of this pipeline, which are backpropagated to the feature network. This homography estimation pipeline was adapted from work on differentiable pose estimation [22]. It contains no learned parameters, which constrains updates to the feature model and promotes the reusability of the final network in other visual navigation pipelines and tasks.

We train multiple model variations with different task-based losses and evaluate our method alongside a set of baseline approaches. Our final model outperforms the baseline methods on cross-spectral image registration as well as feature detection and matching. Notably, we also observe improved performance over the baseline methods on a classical, nondifferentiable image registration pipeline despite its structural differences from the training pipeline. This performance boost on a classical framework carries the added advantage of more seamless integration into existing visual navigation systems. Finally, we attempt to filter ill-conditioned homographies and find that the majority of high-error registration estimates can be removed with a determinant-based heuristic.

The contributions of this thesis are summarized as follows:

- a task-oriented approach to learning cross-spectral point features;

- a feature model that outperforms a set of baseline methods on cross-spectral image matching and registration, both with a specialized homography estimation pipeline and its classical counterpart; and
- a comparison of different task-based loss formulations.

Overall, this work on learned features makes strides towards cross-spectral visual navigation systems for UAVs to enable operation in dark and obscured environments.

Chapter 2

Background

This chapter reviews key concepts and algorithms used in this thesis. We first cover the fundamentals of thermal imaging, followed by a primer on deep learning with a focus on convolutional neural networks. Next, we discuss the projective geometry of the camera model and homography transformation. Finally, we outline the steps of feature-based image registration, including feature extraction, matching, outlier rejection, and model estimation.

2.1 Thermal Imaging

This section provides an overview of thermal imaging. Below, we discuss the nature of thermal radiation, the differences between thermal and visible-spectrum images, and the practical use of thermal cameras in robotics.

2.1.1 Thermal Radiation

All objects with a temperature above absolute zero emit electromagnetic radiation (EMR). This is known as thermal radiation. The wavelength distribution of thermal radiation is largely determined by an object's temperature, and the peak wavelength typically falls in the infrared band of the electromagnetic spectrum, specifically within the mid-wave infrared (MWIR) (3–8 µm), long-wave infrared (LWIR) (8–15 µm), or far infrared (FIR) (15–1000 µm) subbands. Together, these subbands are known as *thermal infrared*. It follows that thermal infrared cameras measure EMR within one of these subbands. There are two other infrared subbands, near infrared (NIR) (0.7–1.4 µm) and short-wave infrared (SWIR) (1.4–3 µm), that are grouped together as *reflected* infrared because, as the name suggests, EMR in this range typically reflects off of objects instead of being emitted

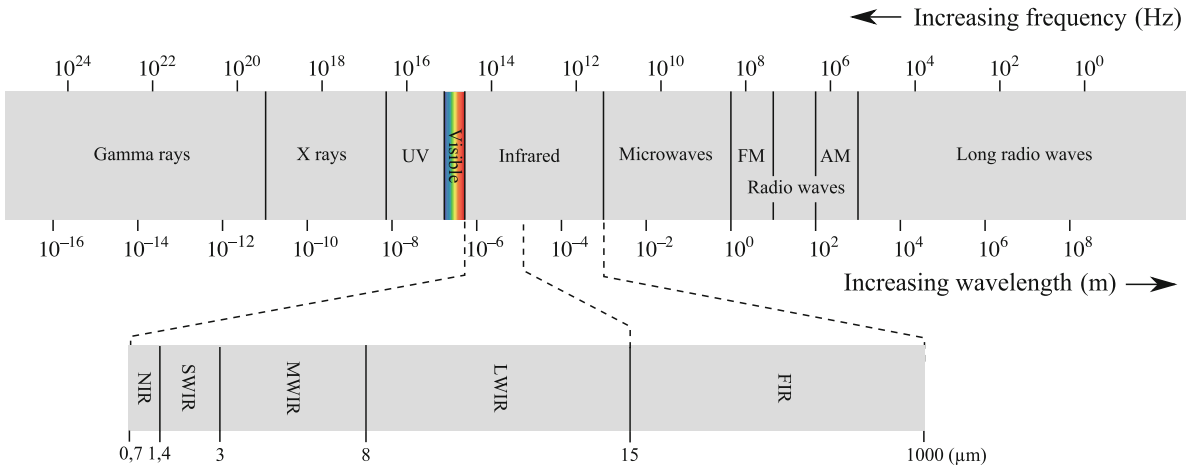


Figure 2.1: Electromagnetic spectrum with subdivisions of the infrared band shown. Figure from Gade and Moeslund [23].

from them. Figure 2.1 shows the infrared band and subbands within the electromagnetic spectrum. In this thesis, we discuss thermal infrared and use *thermal* as a shorthand.

2.1.2 Domain Gap: Thermal Versus Visible-Spectrum

Given that thermal radiation is emitted *from* the environment, the use of thermal cameras does not depend on external illumination or the use of active sensing, unlike visible-spectrum or reflected infrared devices. While this represents a significant advantage of thermal cameras, it comes with the trade-off that thermal images often exhibit soft edges, weak gradients, and a lack of texture, especially in environments with relatively uniform temperatures. Visible-spectrum images, in comparison, exhibit sharp edges and strong gradients. This fundamental difference in image formation creates a large appearance difference, also known as a domain gap, between thermal and visible-spectrum images. Although image contours tend to be preserved across spectra [24], the individual intensities of corresponding pixels between the two image types do not have a direct relationship. Figure 2.2 demonstrates this appearance gap.

2.1.3 Practical Considerations for Robotics

Thermal cameras can be divided into two categories: cooled and uncooled. Cooled thermal cameras employ MWIR photon detectors that require cryogenic cooling to maintain extremely low operational temperatures [23]. Although this technology offers superior sensitivity and frame rates, it incurs significant size, weight, and cost penalties. In con-

Figure 2.2: Thermal (left) and visible-spectrum (right) image of the same scene. Source: Mouats et al. [25].

trast, uncooled thermal cameras measure LWIR radiation with microbolometer sensors that impose fewer hardware requirements, making these devices the predominant choice for use in mobile robotics applications.

Uncooled thermal cameras, however, introduce their own implementation challenges. These devices accumulate fixed pattern noise that necessitates frequent online recalibration with a non-uniformity correction (NUC) [26]. Depending on the camera, NUCs are triggered 1–3 times per minute and last for approximately 0.3–2 s, during which time the camera feed is suspended [3]. Additionally, thermal cameras typically produce single-channel images with 8–16 bits per pixel [23] and often require dynamic scaling to achieve sufficient image contrast, which can vary dramatically as objects move in or out of view. These challenges should be addressed algorithmically for optimal robot operation. For additional details on thermal cameras, we refer the reader to Gade and Moeslund [23].

2.2 Deep Learning Primer

When a phenomenon is too complex or impossible to model analytically, deep learning—particularly supervised learning—offers a viable alternative. In its simplest form, a neural network is a large composite function, and supervised learning is the process of optimizing that function to replicate the input-output relationship of provided data. In this section, we present a brief introduction to neural networks, including an extension to convolutional neural networks and other common network types. We follow the outline of Szeliski [27] and refer the reader to this resource for further information.

2.2.1 Neural Networks

Below, we present a simple form of neural network: the multilayer perceptron (MLP). We start with the network structure followed by the optimization process, a process also referred to as *training* or *learning*.

The building block of a neural network is the artificial neuron, $f_{\text{neuron}} : \mathbb{R}^N \to \mathbb{R}$, defined as

$$f_{\text{neuron}}\left(\mathbf{x}; \mathbf{w}, b\right) = h\left(\mathbf{w}^T\mathbf{x} + b\right), \tag{2.1}$$

where $\mathbf{x}$ is an input vector with N elements, $\mathbf{w}$ is the weighting vector, b is the bias term, and $h : \mathbb{R} \to \mathbb{R}$ is a nonlinear activation function. The weight and bias terms are tunable parameters that are updated during optimization. Many different activation functions exist, but the most popular is the rectified linear unit (ReLU) [28], defined as

$$h(x) = \max(0, x). \tag{2.2}$$

Passing the input into a set of neurons and stacking the outputs into a vector creates a network *layer*. Layer l composed of N_l neurons is expressed as

$$\mathbf{f}_{\text{layer}}\left(\mathbf{x}_l; \boldsymbol{\theta}_l\right) = \mathbf{h}\left({\mathbf{W}_l}^T\mathbf{x}_l + \mathbf{b}_l\right), \tag{2.3}$$

where $\mathbf{x}_l$ is the layer input, $\mathbf{h} : \mathbb{R}^{N_l} \to \mathbb{R}^{N_l}$ is the element-wise application of h, and the stacked weights and biases are grouped together into the layer parameters, $\boldsymbol{\theta}_l = \{\mathbf{W}_l, \mathbf{b}_l\}$. Finally, by composing the functions of L layers, we obtain an equation for the full network,

$$\hat{\mathbf{y}} = \mathbf{f}_{\text{net}}\left(\mathbf{x}; \boldsymbol{\theta}\right) = \left(\mathbf{f}_{\boldsymbol{\theta}_L} \circ \ldots \circ \mathbf{f}_{\boldsymbol{\theta}_l} \circ \ldots \circ \mathbf{f}_{\boldsymbol{\theta}_1}\right)(\mathbf{x}), \tag{2.4}$$

where $\mathbf{f}_{\boldsymbol{\theta}_l} = \mathbf{f}_{\text{layer}}\left(\mathbf{x}_l; \boldsymbol{\theta}_l\right)$ denotes the application of layer l, $\mathbf{x}$ is the network input, $\boldsymbol{\theta} = \{\boldsymbol{\theta}_l\}_{l=1}^{L}$ is the full set of network parameters, and $\hat{\mathbf{y}} \in \mathbb{R}^{N_L}$ is the network output.

The difference between the network output or *estimate*, $\hat{\mathbf{y}}$, and its correct value or *ground truth*, $\mathbf{y}$, is quantified by a loss, $\mathcal{L}$. In the case of estimating continuous numbers, (i.e., regression) the ground truth and estimate can be directly compared using, for example, the squared-error loss, defined as

$$\mathcal{L}_{\text{SE}} = \|\mathbf{y} - \hat{\mathbf{y}}\|_2^2. \tag{2.5}$$

Alternatively, if the output represents a discrete set of classes (i.e., classification), the ground truth takes the form of a *one-hot* vector. A one-hot vector contains a value of one

at the index corresponding to the correct class and zeros elsewhere. For comparison with ground truth, the raw network output (i.e., in *logit* form) is transformed into class-wise probabilities using the *softmax* function, $\boldsymbol{\sigma} : \mathbb{R}^{N_L} \to \mathbb{R}^{N_L}$, defined as

$$\sigma(\hat{\mathbf{y}})_j = \frac{\exp(\hat{y}_j)}{\sum_{k=1}^{N_L} \exp(\hat{y}_k)}. \tag{2.6}$$

The weighted categorical cross-entropy loss is a common loss function for classification tasks and is defined as

$$\mathcal{L}_{\text{CE}} = -\sum_{j=1}^{N_L} \rho_j y_j \log\left(\sigma(\hat{\mathbf{y}})_j\right), \tag{2.7}$$

where $\boldsymbol{\rho}$ contains the weights for each class j.

The objective of neural network optimization is to find the set of parameters, $\boldsymbol{\theta}^*$, that minimize the average loss, $\bar{\mathcal{L}}$, across all input-ground truth pairs in a dataset. Given that the overall network is a composition of each layer function, chain rule can easily be applied to calculate the loss gradient with respect to each weight and bias. At every iteration, each model parameter θ is updated in the direction opposite its gradient according to the following equation:

$$\theta \leftarrow \theta - \alpha \frac{\partial \bar{\mathcal{L}}}{\partial \theta}, \tag{2.8}$$

where α is the step size or *learning rate*. In practice, it would be computationally intractable to compute $\bar{\mathcal{L}}$ over the entire dataset at every iteration. Instead, the loss for a single update step is averaged over a random batch of N_{B} inputs, where N_{B} is referred to as the batch size. This is known as *stochastic gradient descent*. The batch size and learning rate are *hyperparameters*, meaning that they are chosen prior to network training (i.e., not learned).

Another practical element of training neural networks is *batch normalization*, in which the inputs to a given layer are normalized to have zero mean and unit variance across the batch. This improves the network's stability by mitigating large discrepancies between the parameter magnitudes of different layers and is also thought to increase the speed at which the optimization converges.

2.2.2 Convolutional Neural Networks

The rise of a different type of neural network, the convolutional neural network (CNN), has transformed the field of computer vision. CNNs typically operate on images. In contrast to the MLP, which densely connects all of the neurons of consecutive layers, CNNs learn a small window or *kernel* of weights that "slides" along the layer input,

computing a weighted sum at every step. The benefit of this approach is two-fold: it requires significantly fewer parameters than a densely connected network, reducing the computational requirements of optimization, and it enables equivariance along the spatial (i.e., height and width) dimensions. As with the original neuron equation (Equation 2.1), the output is offset with a bias before passing through a nonlinear activation function. Further, applying different sets of weights and biases creates multiple output *channels*. Adding commas between indices for clarity, the output at spatial location (j, k) and channel β is defined as

$$\hat{y}_{j,k,\beta} = \sum_{(m,n)\in\mathcal{D}} \sum_{\forall\alpha} \left(w_{m,\,n,\,\alpha,\,\beta} \cdot x_{j+m,\,k+n,\,\alpha} \right) + b_\beta, \tag{2.9}$$

where α is the input channel, $\mathcal{D}$ is the set of spatial displacements (m, n) relative to (j, k) and, as before, w, b, and x denote the layer weight, bias, and input, respectively. The spatial displacements are symmetric about (j, k), so the input is *padded* such that the kernel can be applied at the edges of the input. The step size between successive applications of the kernel is called the *stride*. For an input of spatial dimension N_{l-1}, the corresponding output length of the convolutional layer is given by

$$N_l = \frac{N_{l-1} - K + 2P}{S} + 1, \tag{2.10}$$

where P is the padding per side, S is the stride, and K is the kernel size in that spatial dimension.

In addition to convolutional layers, CNNs frequently incorporate *max pooling* layers. Instead of a linear combination, the output of a max pooling layer is simply taken as the largest value in the input window.

2.2.3 Other Network Types

Although this thesis primarily focuses on CNNs, we provide the reader with a brief overview of some additional network types utilized in related works: vision transformers (ViTs) and generative adversarial networks (GANs).

A transformer is a type of neural network that is designed to estimate an output sequence given an input sequence. The crux of a transformer network is the self-attention mechanism that determines dependencies between components of the input. Transformer networks were originally developed for natural language processing [29]. In 2021, Dosovitskiy et al. [30] created ViTs, which extend transformers to computer vision tasks by

treating an image as a sequence of patches.

The GAN is a more general network architecture containing two subnetworks: a generator and a discriminator, each of which can use nearly any network type, such as a CNN or transformer. The goal of the generator is to create outputs that are indistinguishable from samples of a dataset, and the goal of the discriminator is to determine whether samples are real (i.e., from the dataset) or generated. The two networks are trained simultaneously with these competing objectives, but after training, only the generator is retained. Given that no explicit ground truth is used, the application of a GAN is considered to be an *unsupervised* learning technique. For further details, we refer the reader to the seminal paper on GANs [31].

2.3 Projective Geometry

In this section, we describe the formation of a two-dimensional image from an observation of the three-dimensional world. We further introduce the homography model and describe how it can be used to transform points from one image plane to another. For more details, we refer the reader to Szeliski [32].

2.3.1 Camera Model

The most common camera model is the ideal pinhole model or *perspective camera model*, which assumes that all light rays pass through an infinitely-small aperture at the camera frame origin, known as the optical center. In this model, a 3D point, $\mathbf{p} = \begin{bmatrix} x & y & z \end{bmatrix}^T$, is projected onto the *normalized image plane* by dividing the x and y coordinates with the z coordinate (i.e., the depth in the camera's reference frame). These points are then transformed into sensor coordinates using the intrinsic parameter matrix, $\mathbf{K}$, and the depth component is removed. The process of obtaining a sensor coordinate, $\mathbf{q} = \begin{bmatrix} u & v \end{bmatrix}^T$, from a 3D point, $\mathbf{p}$, is summarized by

$$\mathbf{q} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \underbrace{\begin{bmatrix} f_U & 0 & c_U \\ 0 & f_V & c_V \\ 0 & 0 & 1 \end{bmatrix}}_{\mathbf{K}} \frac{1}{z} \underbrace{\begin{bmatrix} x \\ y \\ z \end{bmatrix}}_{\mathbf{p}}, \tag{2.11}$$

where f_U and f_V are the camera focal lengths in the u (horizontal) and v (vertical) direction, respectively, and $\begin{bmatrix} c_U & c_V \end{bmatrix}^T$ is the *principal point*, the sensor coordinates where

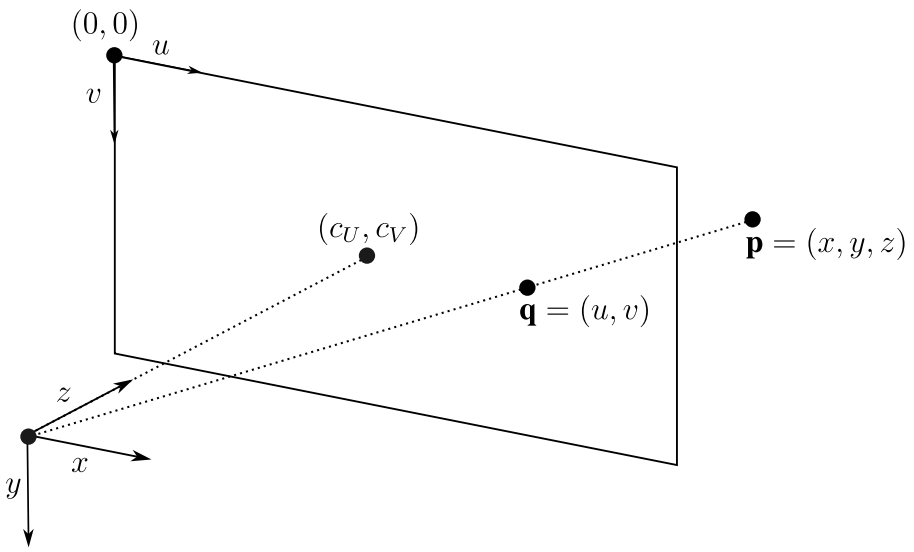


Figure 2.3: Diagram of the projective camera model, shown mapping a 3D real-world point, $\mathbf{p}$, from the camera frame into a 2D image coordinate pair, $\mathbf{q}$, in the sensor frame.

the z axis of the camera frame intersects with the sensor plane. Figure 2.3 shows a diagram of the geometry described by the projective camera model.

2.3.2 Homography Model

A *homography*, also known as a projective transformation or perspective transform, $\mathbf{H} \in \mathbb{R}^{3\times3}$, is a geometric transformation that relates two planes. Applying a homography to an image will preserve straight lines but not necessarily parallel lines. A homography is only defined up to scale. By convention and without loss of generality, the homography matrix is scaled such that the bottom-right element, h_{22}, is equal to 1. Therefore, despite having nine parameters, a homography matrix has eight degrees of freedom. Further, the successive application of multiple homographies can be composed into a single transformation through matrix multiplication.

A homography provides a valid mapping between two images if the camera motion between their viewpoints is limited to rotation about the camera's optical center or if the scene is truly planar. Further, a homography can approximately model a viewpoint change when the distance to the scene is much larger than the displacement between viewpoints. In applications where scale is known, a homography matrix can be used to recover the 3D pose relating two camera viewpoints.

Transforming a 2D source point, ${}^{s}\mathbf{q}$, to its location in the target image, ${}^{t}\mathbf{q}$, using the ground-truth source-to-target homography, ${}^{t}\mathbf{H}_{s}$, involves augmenting the point vec-

tor with unit depth, multiplying it with the homography matrix, and normalizing or *reprojecting* by dividing out the resultant depth:

$$ {}^{t}\mathbf{H}_{s} \begin{bmatrix} {}^{s}\mathbf{q} \\ 1 \end{bmatrix} = \begin{bmatrix} h_{00} & h_{01} & h_{02} \\ h_{10} & h_{11} & h_{12} \\ h_{20} & h_{21} & 1 \end{bmatrix} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} h_{00}u + h_{01}v + h_{02} \\ h_{10}u + h_{11}v + h_{12} \\ h_{20}u + h_{21}v + 1 \end{bmatrix}, \tag{2.12a} $$

$$ {}^{t}\mathbf{q} = \begin{bmatrix} \dfrac{h_{00}u + h_{01}v + h_{02}}{h_{20}u + h_{21}v + 1} \\ \dfrac{h_{10}u + h_{11}v + h_{12}}{h_{20}u + h_{21}v + 1} \end{bmatrix}. \tag{2.12b} $$

Moving forward, the following (shorthand) notation will be used to denote the transformation of point ${}^{s}\mathbf{q}$ by homography ${}^{t}\mathbf{H}_{s}$, including the augmentation, multiplication, and normalization steps: ${}^{t}\mathbf{q} = {}^{t}\mathbf{H}_{s}\,{}^{s}\mathbf{q}$. This notation is also used in [19] and [33].

2.4 Feature-Based Image Registration

Parametric image registration is the process of estimating the parameters of a pre-defined model that describes the geometric transformation from a source image to a target image, and, optionally, aligning the images using this transformation. Techniques for parametric image registration can be broadly divided into area-based and feature-based methods. Area-based methods, also known as pixel-based or intensity-based methods, determine the model parameters by optimizing a similarity measure between the two images, such as mutual information (MI) or sum of squared differences (SSD). This thesis focuses on feature-based parametric image registration methods. In this section, we will describe the steps of using point features (as opposed to line or blob features) to estimate a homography model, including feature extraction, matching, outlier rejection, and model estimation.

2.4.1 Feature Detection and Description

A *point feature*, or simply a feature, consists of a 2D image coordinate pair—known as an interest point, keypoint, or detection—and a unique associated vector called a descriptor. The algorithms used to identify these interest points and assign their descriptor vectors are known as detectors and descriptors, respectively. Although there exists a wide variety of feature extraction algorithms, a detector often takes the following steps: computing a 'corner-like' response over the whole image, applying rejection criteria to

discard low-quality interest point candidates, and finding the local extrema such that only the strongest keypoints in each neighbourhood are retained. A descriptor algorithm then encodes the information in a local patch about each detection.

Feature algorithms are designed to be stable under (limited) appearance and geometric variations, such as changes in illumination or viewpoint, respectively. For detectors, this means detecting the same physical points in different images of the same scene. Descriptor algorithms, however, are designed to both generate similar outputs for the same physical point under image variation and produce highly distinct outputs for non-corresponding points. This ability to generate distinct outputs for different points is known as being *discriminative*. Achieving the aforementioned qualities comes at the cost of algorithmic complexity. The exact tradeoff will ultimately depend on the application, including the expected image variations, error tolerance, and computational constraints.

As an example, we will outline the high-level details of two seminal point feature methods: the Scale-Invariant Feature Transform (SIFT) [34] and Oriented FAST and Rotated BRIEF (ORB) [35] algorithms. The SIFT detector first applies a difference of Gaussians (DoG) filter over the image to extract edges and corners. It then computes the Hessian at each point as a measure of local curvature and rejects points using a Hessian-based heuristic. Finally, the local extrema are taken as the keypoint locations. The SIFT descriptor divides the patch about each keypoint into a set of subpatches and constructs a histogram of the gradient orientations in each subpatch. The descriptor vector is shifted to standardize the direction of the dominant gradient orientation and normalized, yielding a rotation-invariant descriptor. Additionally, SIFT achieves scale invariance by running the detection algorithm across multiple image "scales" (downsampled and blurred versions of the input image) and computing the descriptor at the scale where the detection response is strongest.

In contrast to the SIFT algorithm, which operates primarily on image gradients, ORB relies on binary comparisons. The ORB algorithm uses the Features from Accelerated Segment Test (FAST) detector [36]. FAST identifies interest points by comparing a pixel's intensity to those in a circle around it. The central pixel is marked as an interest point if N contiguous circle pixels are all either brighter or darker than it by a threshold amount. This step is followed by non-maximum suppression (NMS), which iteratively discards keypoints that have a weaker detection response than other keypoints in their local neighborhood. ORB performs detection at multiple scales. Additionally, it assigns a principal orientation to each keypoint. The ORB algorithm then uses the Binary Robust Independent Elementary Features (BRIEF) descriptor [37], which stores the result of a set of binary comparisons within the descriptor patch. The sampling pattern of BRIEF

is rotated to the principal orientation to achieve rotation invariance. For additional examples of point feature algorithms, we refer the reader to the following works: Speeded Up Robust Features (SURF) [38], Binary Robust Invariant Scalable Keypoints (BRISK) [39], Fast Retina Keypoint (FREAK) [40], and KAZE [41].

Learned Features

It is also possible to apply CNNs to the task of feature extraction. Learned features can be divided into two categories: local and global. Local or *sparse* learned features train and operate on image patches and learn separate modules for the tasks of detection and description. During execution, like classical features, the patch surrounding each detection is fed to the descriptor network. This sequential operation is called a 'detect-then-describe' approach. Prominent examples of patch-based learned features include Learned Invariant Feature Transform (LIFT) [42] and Local Feature Network (LF-Net) [43].

In contrast, global or *dense* learned features train and operate on entire images. These types of feature networks output dense or semi-dense detection and description responses. Additional steps, such as the heuristic rejection and NMS described earlier when discussing SIFT and ORB, are typically required to convert the detection response into discrete 2D image coordinates. Once these are extracted, the descriptor output is sampled at the keypoint locations to obtain discrete descriptors. Further, the detection and description processes are typically coupled into a single network with separate *heads* for each task, which run in parallel. This is known as a 'detect-*and*-describe' approach. For prominent examples of dense learned features, we refer the reader to Detect-and-Describe Network (D2-Net) [20], SuperPoint [19], and Reliable and Repeatable Detector and Descriptor (R2D2) [21].

2.4.2 Matching

After keypoint extraction, a matcher identifies correspondences between source and target keypoints based on their descriptors' similarity. Nearest neighbor (NN) matching is standard in classical computer vision pipelines. For each source feature, the NN matcher identifies the closest target feature in the descriptor space as a potential match. Potential matches exceeding a predefined descriptor distance threshold may be discarded. The handling of ambiguous matches, in which multiple target descriptors lie close to a source descriptor, motivates two variants of the NN matcher: the ratio test and mutual nearest neighbor (MNN) matching. The ratio test, first introduced by Lowe [34], accepts matches

only if the ratio of distances to the nearest and second-nearest neighbors falls below a predefined threshold, while MNN matching requires source and target features to be each other's nearest neighbor.

2.4.3 Outlier Rejection

Even robust matching algorithms produce incorrect correspondences that must be filtered. The most widely adopted technique for outlier rejection is random sample consensus (RANSAC) [44]. RANSAC aims to identify a geometric model that maximizes agreement with the proposed matches.

The RANSAC algorithm consists of the following steps. First, the type of geometric model is specified *a priori*—in this case, a homography. A minimal set of matches is randomly sampled to compute a candidate model (e.g., four correspondences for a homography). Reprojection error then quantifies the agreement between the candidate model and individual correspondences; this error is computed as the distance between matched points after using the model to transform them into the same image.[1] Matches whose reprojection error falls below a threshold (typically 3–4 pixels) are labeled as inliers—that is, they *agree* with the model. This process repeats until a sufficient inlier-to-match ratio is achieved or a maximum number of iterations is reached, at which point the model with the highest number of inliers is selected.

2.4.4 Homography Estimation

Given a set of inlier matches, homography estimation can be performed using either a linear or nonlinear approach. In practice, the linear solution provides an initial estimate of the homography, which is then refined through nonlinear optimization. Below, we discuss the linear approach, its weighted extension, and the nonlinear optimization process. For additional details, we refer the reader to [45] and [46].

Linear solution

Given a set of N_{M} source keypoints and their estimated corresponding target keypoints, $\left\{{}^{s}\mathbf{q}^{i}, {}^{t}\hat{\mathbf{q}}^{i}\right\}_{i=1}^{N_{\mathrm{M}}}$, homography estimation can be formulated as a linear least squares problem

[1] The provided definition describes geometric error rather than reprojection error as the latter implies the computation of a 3D point. For 2D image registration, the terms are often used interchangeably.

by rearranging the projective transformation equation (Equation 2.12b) as follows:

$$ {}^{t}\hat{\mathbf{q}}^{i} = \begin{bmatrix} \hat{u} \\ \hat{v} \end{bmatrix} = \begin{bmatrix} \dfrac{h_{00}u + h_{01}v + h_{02}}{h_{20}u + h_{21}v + 1} \\ \dfrac{h_{10}u + h_{11}v + h_{12}}{h_{20}u + h_{21}v + 1} \end{bmatrix}, \tag{2.13a} $$

$$ \begin{bmatrix} \hat{u}(h_{20}u + h_{21}v + 1) \\ \hat{v}(h_{20}u + h_{21}v + 1) \end{bmatrix} = \begin{bmatrix} h_{00}u + h_{01}v + h_{02} \\ h_{10}u + h_{11}v + h_{12} \end{bmatrix}, \tag{2.13b} $$

$$ \begin{bmatrix} -h_{00}u - h_{01}v - h_{02} + h_{20}\hat{u}u + h_{21}\hat{u}v + \hat{u} \\ -h_{10}u - h_{11}v - h_{12} + h_{20}\hat{v}u + h_{21}\hat{v}v + \hat{v} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \tag{2.13c} $$

$$ \underbrace{\begin{bmatrix} -u & -v & -1 & 0 & 0 & 0 & \hat{u}u & \hat{u}v & \hat{u} \\ 0 & 0 & 0 & -u & -v & -1 & \hat{v}u & \hat{v}v & \hat{v} \end{bmatrix}}_{\mathbf{A}^i} \underbrace{\begin{bmatrix} h_{00} \\ h_{01} \\ h_{02} \\ h_{10} \\ h_{11} \\ h_{12} \\ h_{20} \\ h_{21} \\ 1 \end{bmatrix}}_{\mathbf{h}} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \tag{2.13d} $$

where $\mathbf{A}^i$ is the correspondence matrix for match i. The correspondence matrices of all matches are then stacked together,

$$ \mathbf{A}\mathbf{h} = \begin{bmatrix} \mathbf{A}^1 \\ \mathbf{A}^2 \\ \vdots \\ \mathbf{A}^{N_\mathrm{M}} \end{bmatrix} \mathbf{h} = \mathbf{0}, \tag{2.14} $$

where $\mathbf{h}$ is the homography matrix in vector form and $\mathbf{0}$ is a zero vector. Given the eight unknowns of $\mathbf{h}$, at least $N_\mathrm{M} = 4$ sets of non-collinear correspondences are needed to obtain a valid solution. The aggregate correspondence matrix, $\mathbf{A}$, is then decomposed using singular value decomposition (SVD):

$$ \mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T. \tag{2.15} $$

The last column of $\mathbf{V}$, which corresponds to the smallest eigenvalue of $\mathbf{A}$, is the solution that minimizes the algebraic error across all matches. Note that with $N_{\mathrm{M}} = 4$ correspondences (such as during RANSAC sampling), the resultant homography will exactly fit the provided matches. After solving, $\mathbf{h}$ is reshaped into the homography matrix, $\mathbf{H}$. This solution is known as the direct linear transform (DLT) algorithm.

Weighted Linear Solution

The linear solution can be extended to include a scalar confidence weight, w_i, for each correspondence i—not to be confused with the network weights from Section 2.2. This requires substituting each correspondence matrix with

$$\mathbf{A}^i \leftarrow \mathbf{W}^i \mathbf{A}^i, \tag{2.16a}$$

$$\mathbf{W}^i = w^i \, \mathbf{I}_2, \tag{2.16b}$$

where $\mathbf{I}_2$ is a 2×2 identity matrix.

Nonlinear Solution

Nonlinear least squares (NLS) is an optimization method that refines the parameters of a nonlinear model by minimizing the sum of squared errors, or *residuals*. Starting from an initial estimate, NLS computes incremental updates using a linearized model about the current parameters and iterates this process until convergence. Below, we show the standard NLS equations adapted for homography estimation. For this explanation, we use $\mathbf{f}(\mathbf{q}; \mathbf{h})$ to denote the transformation of point $\mathbf{q}$ by homography $\mathbf{H}$. Additionally, we shorten the notation of ${}^s\mathbf{q}$ to $\mathbf{q}$ and ${}^t\hat{\mathbf{q}}$ to $\hat{\mathbf{q}}$ for simplicity. The residuals are defined as the reprojection error for each match; for an estimated correspondence $\{\mathbf{q}^i, \hat{\mathbf{q}}^i\}$, the residual is expressed as

$$\mathbf{r}^i = \mathbf{f}(\mathbf{q}^i; \mathbf{h}) - \hat{\mathbf{q}}^i, \tag{2.17}$$

and the overall objective function for NLS optimization is

$$E_{NLS}(\mathbf{h}) = \sum_{i=1}^{N_{\mathrm{M}}} \left\| \mathbf{r}^i \right\|_2^2 = \sum_{i=1}^{N_{\mathrm{M}}} \left\| \mathbf{f}(\mathbf{q}^i; \mathbf{h}) - \hat{\mathbf{q}}^i \right\|_2^2 . \tag{2.18}$$

The objective function is then re-expressed with the linearized projective transformation function as

$$\begin{aligned} E_{NLS}(\boldsymbol{\Delta h}) &= \sum_{i=1}^{N_{\mathrm{M}}} \left\| \mathbf{f}(\mathbf{q}^i;\, \mathbf{h}+\boldsymbol{\Delta h}) - \hat{\mathbf{q}}^i \right\|_2^2, \\ &\approx \sum_{i=1}^{N_{\mathrm{M}}} \left\| \mathbf{J}(\mathbf{q}^i;\, \mathbf{h})\boldsymbol{\Delta h} - \mathbf{r}^i \right\|_2^2, \\ &= \boldsymbol{\Delta h}^T \underbrace{\left[\sum_{i=1}^{N_{\mathrm{M}}} \mathbf{J}^{iT}\mathbf{J}^i \right]}_{\mathbf{A}} \boldsymbol{\Delta h} - 2\boldsymbol{\Delta h}^T \underbrace{\left[\sum_{i=1}^{N_{\mathrm{M}}} \mathbf{J}^{iT}\mathbf{r}^i \right]}_{\mathbf{b}} + \underbrace{\sum_{i=1}^{N_{\mathrm{M}}} \left\| \mathbf{r}^i \right\|_2^2}_{c}, \\ &= \boldsymbol{\Delta h}^T \mathbf{A} \boldsymbol{\Delta h} - 2\boldsymbol{\Delta h}^T \mathbf{b} + c, \end{aligned} \tag{2.19}$$

where $\boldsymbol{\Delta h}$ is a change in the homography parameters, and $\mathbf{J}^i = \mathbf{J}(\mathbf{q}^i;\, \mathbf{h})$ is the Jacobian of $\mathbf{f}(\mathbf{q}^i;\, \mathbf{h})$ with respect to $\mathbf{h}$. For a projective transformation, the Jacobian is expressed as

$$\mathbf{J}(\mathbf{q}^i; \mathbf{h}) = \frac{\partial\, \mathbf{f}(\mathbf{q}^i; \mathbf{h})}{\partial\, \mathbf{h}} = \frac{1}{h_{20}u + h_{21}v + 1} \begin{bmatrix} u & v & 1 & 0 & 0 & 0 & -u'u & -u'v \\ 0 & 0 & 0 & u & v & 1 & -v'u & -v'v \end{bmatrix}. \tag{2.20}$$

To solve for the optimal parameter update direction, $\boldsymbol{\Delta h}^*$, the derivative of $E_{NLS}(\boldsymbol{\Delta h})$ is set to zero, which yields a linear system of equations,

$$\mathbf{A}\boldsymbol{\Delta h}^* = \mathbf{b}, \tag{2.21}$$

that can be solved with an off-the-shelf linear solver such as Cholesky decomposition. The homography parameters are then updated according to

$$\mathbf{h} \leftarrow \mathbf{h} + \alpha \boldsymbol{\Delta h}^*, \tag{2.22}$$

where α is the update rate. The residuals, Jacobian, and parameter update direction are recomputed at every iteration with the refined parameters until convergence. A common modification to this method involves modifying Equation 2.21 to

$$(\mathbf{A} + \lambda\, \mathrm{diag}(\mathbf{A}))\boldsymbol{\Delta h}^* = \mathbf{b}, \tag{2.23}$$

where $\mathrm{diag}(\mathbf{A})$ extracts the diagonal elements of matrix $\mathbf{A}$ and λ is a damping parameter that improves convergence. This is known as the Levenberg–Marquardt algorithm (LMA) or damped least squares.

Chapter 3

Related Work

This chapter presents an overview of existing work on cross-spectral image registration. As mentioned in Section 2.4, registration methods are broadly categorized into feature-based and area-based approaches. While our primary focus is on point features, we cover both categories in this chapter. We first review handcrafted methods followed by their learned counterparts.

3.1 Handcrafted Registration Methods

Early works on cross-spectral registration stem from fields outside of robotics, namely medical imaging and remote sensing. In robotics, these handcrafted methods have fallen out of favor as they often demonstrate too strong a tradeoff between computation time and robustness to multi-modal appearance change. Nevertheless, these classical approaches provide valuable insights for handling cross-spectral image pairs and ground our discussion of more current works.

3.1.1 Area-Based Approaches

Area-based methods optimize the parameters of a predefined geometric transformation to maximize a similarity measure between two overlapping images. These methods are often categorized according to their similarity measure. Ideally, similarity measures depend only on the quality of image alignment, responding to geometric shifts while remaining invariant to appearance change. The simplest and most common similarity measures—such as sum of squared differences (SSD), sum of absolute differences (SAD), and normalized cross-correlation (NCC)—are designed for relatively small lighting changes and are therefore insufficient to overcome the nonlinear intensity variations between images

of different modalities [47]. Mutual information (MI) is an information theory-based similarity measure that quantifies the statistical dependence between two random variables (e.g., images). This similarity measure has been successfully deployed in the medical imaging community to align images of different modalities [48]. Unfortunately, the computational burden of MI renders these methods infeasible for real-time applications. In remote sensing, a common area-based approach involves densely computing descriptors over the source and target images (i.e., transforming the images into a *descriptor space*) and comparing the descriptor maps with a simple similarity measure. Two prominent examples include histogram of oriented phase congruency (HOPC) [49] and channel features of oriented gradients (CFOG) [50]. In [49], the authors compute an HOPC descriptor at uniform grid points on each image and compare them with NCC; and in [50], the authors apply a modified histogram of gradients descriptor algorithm at each pixel for comparison with SSD. Although these approaches demonstrate some robustness to nonlinear intensity variations, they offer only modest computational speedups compared to MI and lack robustness to geometric transformations such as scale and rotation [51].

3.1.2 Feature-Based Approaches

Point features offer a sparse alternative to dense, area-based approaches, but these methods similarly struggle to identify robust and efficient ways of overcoming the domain gap between thermal and visible-spectrum images. Traditional, visible-spectrum feature algorithms typically form descriptors from local image gradients or a set of binary comparisons in a patch centered around a keypoint, such as the Scale-Invariant Feature Transform (SIFT) [34] and Oriented FAST and Rotated BRIEF (ORB) [35] algorithms, respectively, as described in Section 2.4.1. Unsurprisingly, both of these description approaches often fail when faced with nonlinear intensity variations in cross-spectral data [52]. Alternative handcrafted methods developed specifically for cross-spectral data rely on the fact that contours are generally preserved between visible-spectrum and thermal images [24]. Aguilera et al. [53] first extract contours using a Canny edge detector [54], then apply Sobel filters to the edge response to compute a histogram of gradient orientations around each interest point. A handful of works propose descriptors that, like edge detection, focus on high-frequency image components, describing the patch around each keypoint with a histogram of responses from multi-scale and multi-oriented log-Gabor filters [18, 55, 56]. While an improvement on the naive use of visible-spectrum methods, the robustness of these feature algorithms to the domain gap and viewpoint variations could be improved. Further, although feature-based approaches are intended

to be lightweight, the descriptor extaction processes for the described methods are slow, typically taking well over one second per image [18].

3.2 Learned Registration Methods

Learning has emerged as the dominant paradigm for bridging the cross-spectral domain gap. Not only are these methods faster than traditional approaches, but data-driven methods have proven more accurate under dramatic appearance and viewpoint changes. In this section, we explore learning-based approaches for registering thermal and visible-spectrum images with a focus on homography estimation.

3.2.1 Regression Models

Learned extensions of area-based registration generally use regression techniques to estimate a transformation directly from two images. Although area-based techniques fall outside the scope of our specific approach, the findings of these works can inform our choices on, for example, error formulation. A representative work of this type is VisIR-Net [57]. The authors use a convolutional encoder and regression network to directly estimate the homography transform between two images of different spectra. This network is trained in a supervised manner: the authors apply a similarity loss between the encoder outputs and formulate two losses between the estimated and ground-truth homographies. The first homography loss penalizes the squared error between the parameters of the homography matrices, while the second calculates the average error of the estimated image corner locations. Further, the authors consider two network types: one that returns homography parameters and another that outputs transformed image corner coordinates. Although these representations are mathematically equivalent, the latter achieves superior accuracy. This work and its findings serve as a reference point to discuss other advances in homography regression for cross-spectral registration.

A number of works build upon the framework of [57]. Xiao et al. [17] introduce an iterative, coarse-to-fine method. Multiple works incorporate domain translation modules into their frameworks: in [58] and [59], the authors use generative adversarial networks (GANs) to perform image translation as a means of reducing the cross-spectral appearance gap between images, while [17] extends their training set by converting visible-spectrum images into "pseudo-thermal" images. In [59] and [60], the authors train a separate convolutional neural network (CNN) to act as an inlier mask. A number of unsupervised homography estimation techniques have also been proposed. Debaque et al.

[60] train both a supervised and an unsupervised model, the latter of which simultaneously estimates 'forward' (source-to-target) and 'reverse' (target-to-source) homographies and enforces consistency between them. The authors also apply a contrastive loss to ensure that the encoder outputs remain discriminative. The unsupervised model proposed by Shin et al. [59] employs a phase congruency loss to evaluate the alignment quality. Two works, [61] and [62], use an adversarial framework in which a base model estimates the homography and a discriminator network evaluates the quality. Further, in [62], the authors substitute the typical CNN encoder for a transformer network to capture the relationship between image patches. Despite these advances, research in camera pose estimation has demonstrated that replacing entire estimation pipelines with a neural network may result in lower generalization capability compared to learning only select modules (e.g., feature extraction or matching) and leaving the rest of the pipeline in its standard form [63].

3.2.2 Learned Point Features

Learning point features offers a targeted incorporation of deep learning into the cross-spectral image registration pipeline. The earliest works of this nature developed local learned features; as noted in Section 2.4.1, these methods replace handcrafted descriptor algorithms with CNNs that, like traditional descriptors, compute a unique vector from the patch about a given interest point. Aguilera et al. [64] train a quadruplet network, Q-Net, on sets of aligned thermal and visible-spectrum image patches. The authors use a *contrastive loss*, which simultaneously encourages cross-spectral similarity for the descriptors of a given patch and distinctiveness between descriptors for non-corresponding patches. Jeong et al. [65] jointly train an image translation network and descriptor network. The image translation network creates pseudo-paired thermal-visible patches, which are then used to train the descriptor network with a contrastive loss. One shortfall of these local learned methods is their reliance on classical detectors. Further, since the learned descriptors train on small patches, they operate on low-level image information, which is less reliable for large domain gaps.

More recent work, which focuses on cross-spectral extensions of *dense* learned features, follows a few key themes. As described in Section 2.4.1, dense methods exhibit large overlap in their network structure, often using a single image encoder followed by separate detector and descriptor decoders. This coupling of detection and description has been shown to provide superior robustness to appearance and viewpoint change in the visible-spectrum domain [19–21]. In addition, training on whole images leverages the

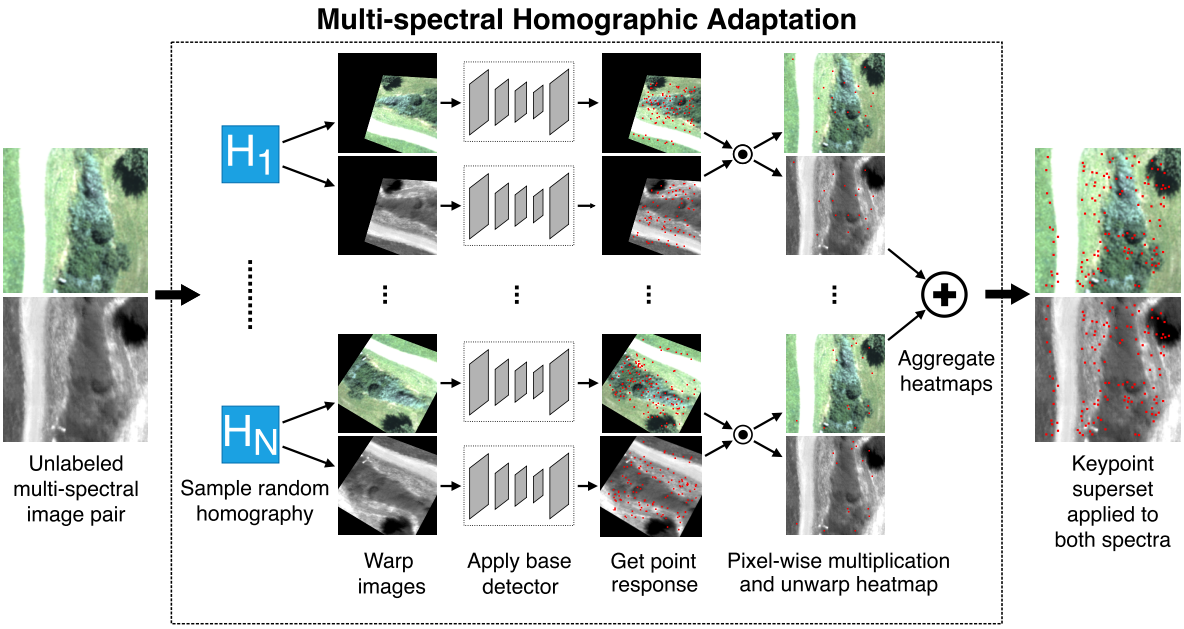


Figure 3.1: Multi-spectral homographic adaptation process from MultiPoint. The base detector response on warped versions of the thermal-visible image pair is aggregated into a pseudo-ground truth detection map. Figure from Achermann et al. [33].

ability of CNNs to encode high-level information, which may help overcome the lack of a relationship between individual pixel values across spectra. To obtain ground-truth correspondences, works on cross-spectral learned features typically use aligned thermal-visible image pairs for training and apply homographies to simulate viewpoint change.

The primary distinction among these methods is their approach to generating a supervisory signal given the ground-truth geometric relationship between a thermal image and a visible-spectrum image. The earliest example is MultiPoint [33], a multi-spectral extension of the prominent SuperPoint [19]. Similar to the local learned features in [64] and [65], this work trains the descriptor with a contrastive loss. Detection, however, is treated as a classification problem. The authors generate pseudo-ground truth for the keypoints of each aligned thermal-visible image pair using a process called multi-spectral homographic adaptation. In this process, depicted in Figure 3.1, a base detector's responses on warped versions of the thermal and visible-spectrum images are aggregated into a single heatmap. The authors use Speeded Up Robust Features (SURF) [38], a classical visible-spectrum detector, as their base detector. This method of keypoint supervision, however, is limited by the robustness—or lack thereof—of the base detector. Multiple works on cross-spectral features build upon MultiPoint's method. Elsaeidy et al. [66] first train a GAN to translate thermal images into pseudo-visible images and then

retrain MultiPoint with the real and synthetic visible-spectrum image pairs. In [67], the authors replace the CNN encoder with a Swin Transformer, which is a specialized type of vision transformer (ViT). A different cross-spectral method, ReDFeat [68], is modeled after the Reliable and Repeatable Detector and Descriptor (R2D2) network [21], a popular work on learned features in the visible-spectrum domain. As with the other learned methods, the authors use a contrastive loss for the descriptor. For detection, this method sheds the reliance on a handcrafted algorithm. The authors instead drive detection with an edge-based prior and a peaking loss that encourages a singular maximum in each of a set of local windows. Then, a similarity loss is applied between the responses of the two spectra. However, in the absence of an external signal, the network risks collapsing to a degenerate output (e.g., a uniform response), requiring the authors to couple the detection and description losses in a way that masks areas of low matching confidence.

Overall, while these approaches may find a *reliable* set of features, they concentrate training on image regions that already exhibit a degree of cross-spectral similarity. However, since thermal and visible-spectrum images are formed by fundamentally different underlying phenomena, a significant amount of photometric variation between images of different spectra is to be expected and, ideally, accounted for in the training process. Therefore, an alternative approach to supervision may be required to allow the network to identify useful features across large domain gaps.

Chapter 4

Methodology

This chapter presents our task-oriented approach to learning cross-spectral point features. As shown in Figure 4.1, our framework operates on an aligned thermal and visible-spectrum image pair. A randomly sampled homography, ${}^{t}\mathbf{H}_{s}$, transforms one of the images, simulating a viewpoint change from the *source* (i.e., rectified) image to the *target* (i.e., transformed) image, denoted by s and t, respectively. The source and target images are fed to the feature network. From the network response, we extract and match features, perform outlier rejection, and compute a homography estimate, ${}^{t}\hat{\mathbf{H}}_{s}$. Losses are applied to the intermediate outputs and final homography estimate of our pipeline, which backpropagate through the differentiable registration pipeline and feature network. In addition, we apply losses directly to the detection and description outputs of the feature network. Below, we describe the details of our method, including the network architecture, the differentiable registration pipeline, and the three types of losses: detector, descriptor, and task-based.

4.1 Feature Network Architecture

Our feature network follows the convolutional encoder-multi-decoder architecture of MultiPoint [33]: an encoder generates a latent representation of the input image, which is passed to separate decoders for detection and description. The same network is used for both spectra. Figure 4.2a shows the full network architecture.

The VGG-style encoder [69] consists of four convolutional blocks interleaved with three max pooling layers. Each convolutional block is composed of two convolutional layers, and each convolutional layer is followed by a rectified linear unit (ReLU) activation [28] and batch normalization. The max pooling layers each reduce the spatial dimensions by a factor of two, resulting in a latent representation of shape $H/8 \times W/8 \times 128$, where

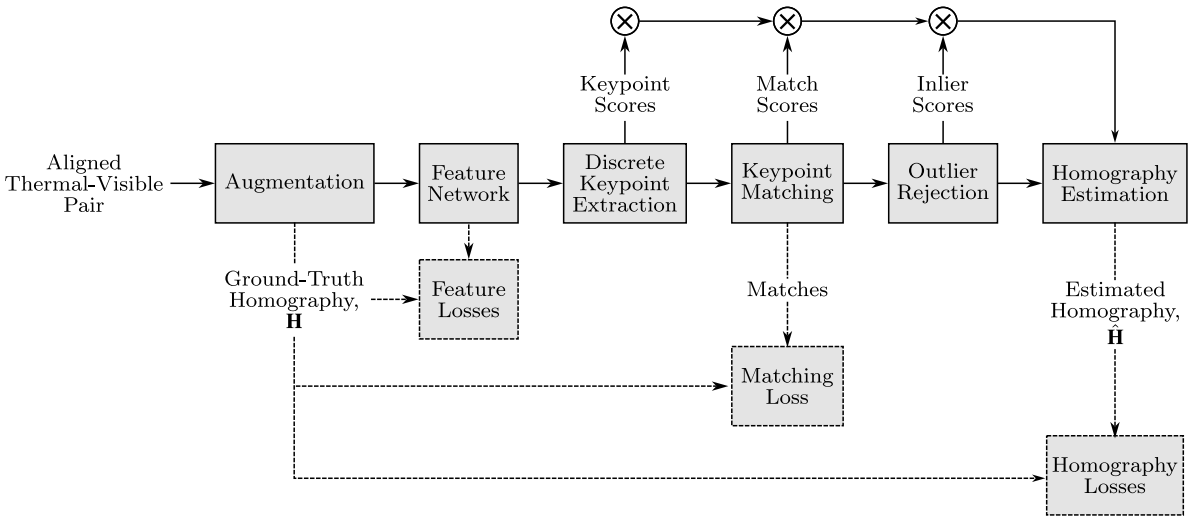


Figure 4.1: Method overview flowchart. One image of a thermal-visible image pair is warped by a randomly-sampled homography (i.e., augmented). The feature network response to the augmented image pair is fed to a differentiable homography estimation pipeline, and losses are applied to the matching and registration estimates. Solid lines denote the pipeline steps and dashed lines denote the loss computation steps.

H and W are the height and width of the input images, respectively.

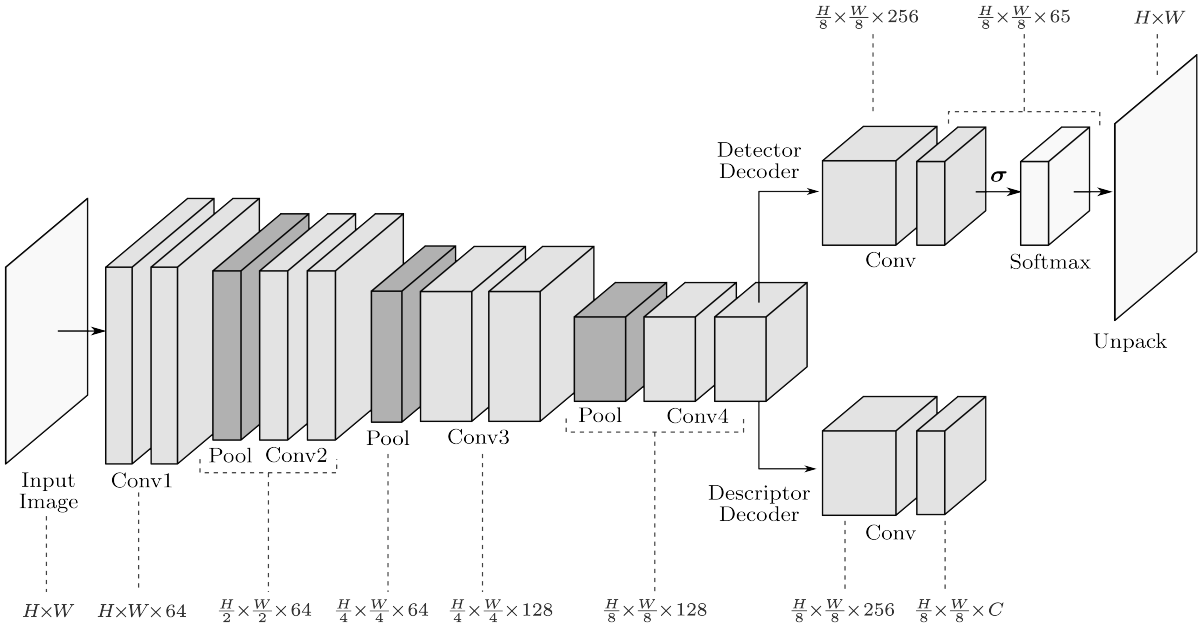
The descriptor decoder applies an additional convolutional block without a final activation, followed by L2 normalization along the channel dimension. This process generates a semi-dense descriptor map with C channels, $\mathbf{D} \in \mathbb{R}^{H/8 \times W/8 \times C}$. In our implementation, C is chosen to be 64.

The detector decoder similarly uses a convolutional block without a final activation to produce an unnormalized detection map, $\mathbf{K} \in \mathbb{R}^{H/8 \times W/8 \times 65}$.[1] As illustrated in Figure 4.2b, this detection map is a collection of *cells*, $\mathbf{k} \in \mathbb{R}^{65}$, each of which contains the keypoint response for an 8×8 pixel patch of the input image plus a *dustbin* response for the absence of a keypoint in the patch. Depending on the method of discrete keypoint extraction, the detection response can be used either in this unnormalized form or first converted into a probabilistic format. Applying a cell-wise softmax, $\boldsymbol{\sigma}(\cdot)$, to an unnormalized cell, $\mathbf{k}$, yields a discrete probability distribution of a keypoint's position within the patch, $\mathbf{p}$, expressed as

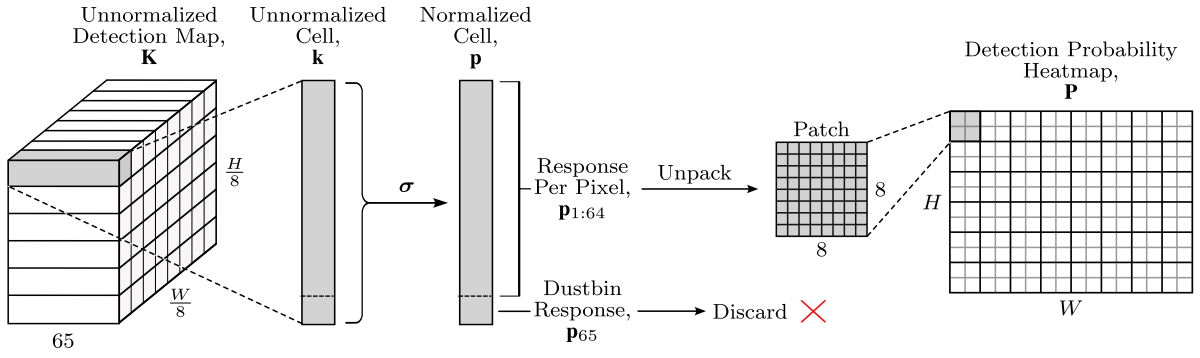
$$\mathbf{p} = \boldsymbol{\sigma}(\mathbf{k}). \tag{4.1}$$

The softmax function, as shown in Equation 2.6, considers the relative magnitude of the unnormalized detection responses, meaning that the detection probabilities of the image

[1] Not to be confused with the camera intrinsic matrix.



(a) Overview of the convolutional neural network architecture. The VGG-type encoder [69] consists of four convolutional blocks (light gray) alternating with three max pooling layers (dark gray) that compress the spatial dimensions. The latent representation branches into the descriptor decoder and detector decoder (for which the normalization steps are shown in (b)). Dimensions at each stage shown at the top or bottom of the figure.



(b) Visualization of the (optional) conversion from the unnormalized detection response into the detection probability heatmap. A softmax operation is applied to each cell, after which the dustbin probability is discarded and the probabilities are unpacked into the image-sized heatmap.

Figure 4.2: Network architecture diagram (top) and conversion from unnormalized detection response into a detection probability heatmap (bottom).

patch account for the dustbin (i.e., the probability of *no* keypoint). After this operation, the now-redundant dustbin is discarded and the remainder of the detection probabilities are unpacked into a heatmap, $\mathbf{P}$, of shape $H \times W$.

4.2 Differentiable Registration Pipeline

In this section, we describe the components of the differentiable pipeline for estimating a homography from the network responses to the source and target images. This homography estimation pipeline builds on the differentiable pose estimator from Gridseth and Barfoot [22]. This pipeline contains no learned parameters, which, as noted by [22, 63], is intended to promote the generalization of the feature network. We first cover the discrete keypoint extraction and matching (illustrated in Figure 4.3), followed by the outlier rejection and model estimation.

4.2.1 Discrete Keypoint Extraction

The network provides a detection response, but an additional step is required to extract discrete, 2D keypoints. The differentiable keypoint estimator used in our training pipeline operates on the unnormalized detection response, $\mathbf{K}$. First, the dustbin responses are discarded (without normalization), and the remainder of the detection map is unpacked into an array $\tilde{\mathbf{K}} \in \mathbb{R}^{H \times W}$. This array is divided into a set of non-overlapping windows, and a subpixel keypoint coordinate, $\mathbf{q}^i = \begin{bmatrix} u^i & v^i \end{bmatrix}^T$, is estimated in each window. The coordinates are obtained with a spatial soft argmax$(\cdot)$, expressed as

$$u^i = \sum_{(m,n)\in\mathcal{W}^i} \left(n \frac{\exp(\tilde{k}_{mn})}{\sum_{(q,r)\in\mathcal{W}^i} \exp(\tilde{k}_{qr})} \right), \quad v^i = \sum_{(m,n)\in\mathcal{W}^i} \left(m \frac{\exp(\tilde{k}_{mn})}{\sum_{(q,r)\in\mathcal{W}^i} \exp(\tilde{k}_{qr})} \right), \tag{4.2}$$

where $\mathcal{W}^i$ denotes the set of coordinate pairs within window i. In our implementation, the windows are chosen to be 8×8 pixels in size.

Not every window will contain a useful keypoint, so we mitigate the impact of low-quality detections by assigning each keypoint a score, s^i_{K}, that is sampled from the keypoint probability heatmap, $\mathbf{P}$, at the subpixel keypoint coordinate using bilinear interpolation. We also interpolate and renormalize the keypoint's associated descriptor, $\mathbf{d}^i \in \mathbb{R}^C$, from the descriptor map, $\mathbf{D}$.

Perspective transformations may create invalid image regions. We filter out keypoints that fall outside of the valid region with a binary mask determined by the ground-truth homography. We define the final, filtered sets of detected points in the source and target

images as ${}^{s}\mathbf{Q} = \left[{}^{s}\mathbf{q}^1 \dots {}^{s}\mathbf{q}^i \dots {}^{s}\mathbf{q}^{N_{\mathrm{KS}}}\right]$ and ${}^{t}\mathbf{Q} = \left[{}^{t}\mathbf{q}^1 \dots {}^{t}\mathbf{q}^i \dots {}^{t}\mathbf{q}^{N_{\mathrm{KT}}}\right]$, where N_{KS} and N_{KT} represent the number of valid source and target keypoints, respectively.

4.2.2 Matching

In contrast to classical matchers that determine one-to-one correspondences between keypoints, the matcher in our training pipeline uses soft matching scores to enable differentiability. For a given source keypoint, ${}^{s}\mathbf{q}^i$, an implicit match is defined with a *pseudo*-target keypoint, ${}^{t}\hat{\mathbf{q}}^i$, that is computed as a weighted sum of all extracted target keypoints. First, for source point i, a vector of zero-normalized cross-correlation (ZNCC) similarity scores, $\boldsymbol{\phi}^i$, to all target points ($j = 1, \dots, N_{\mathrm{KT}}$) is computed as

$$\phi^i_j = f_{\mathrm{zncc}}({}^{s}\mathbf{d}^i, {}^{t}\mathbf{d}^j) + 1, \tag{4.3}$$

where ${}^{s}\mathbf{d}^i$ and ${}^{t}\mathbf{d}^j$ are the source and target descriptors, respectively, and $f_{\mathrm{zncc}}(\cdot)$ denotes the ZNCC of two vectors. These scores are then adjusted with a temperature-scaled softmax to compute the pseudo-target keypoint location, expressed as

$${}^{t}\hat{\mathbf{q}}^i = \sum_{j=1}^{N_{\mathrm{KT}}} \sigma\left(\frac{\boldsymbol{\phi}^i}{\tau}\right)_j {}^{t}\mathbf{q}^j, \tag{4.4}$$

where τ is the softmax temperature. A temperature value of $\tau = 0.01$ is used, which assigns a strong weighting to the nearest-neighbour target descriptor. Figure 4.3 illustrates the windowed keypoint extraction, filtering, and weighted matching steps.

As with the original keypoint extraction, the associated descriptor, ${}^{t}\hat{\mathbf{d}}^i$, and keypoint score, ${}^{t}\hat{s}^i_{\mathrm{K}}$, of the pseudo-target keypoint are interpolated from the descriptor map and detection probability heatmap, respectively. Further, a matching score, s^i_{M}, is given by

$$s^i_{\mathrm{M}} = \frac{1}{2}\left(f_{\mathrm{zncc}}({}^{s}\mathbf{d}^i, {}^{t}\hat{\mathbf{d}}^i) + 1\right). \tag{4.5}$$

4.2.3 Outlier Rejection

To ensure training convergence, the impact of highly erroneous matches should be mitigated prior to model estimation. Therefore, we assign an inlier score, s^i_{I}, to each match based on its reprojection error when transformed with the ground-truth homography. Using the homography notation defined in Section 2.3.2, the error for match i, in pixels, is computed as

$$x^i = \left\| {}^{t}\mathbf{H}_s \, {}^{s}\mathbf{q}^i - {}^{t}\hat{\mathbf{q}}^i \right\|_2, \tag{4.6}$$

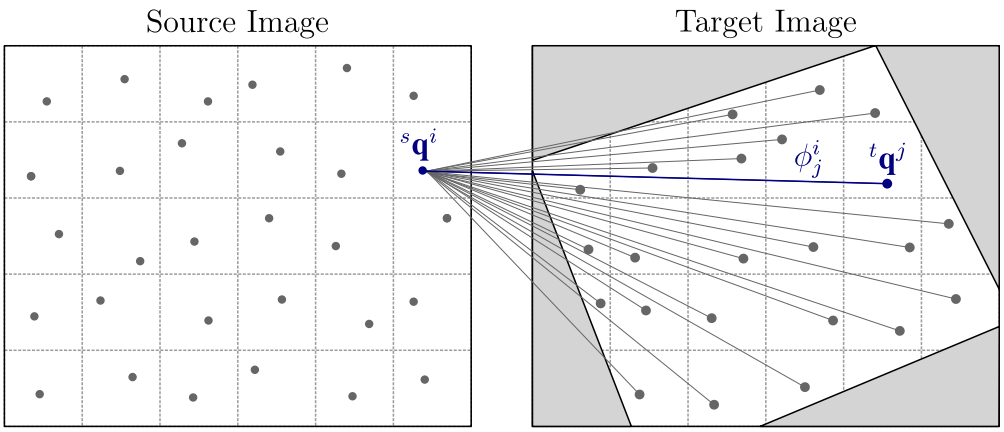


Figure 4.3: Visualization of the discrete keypoint extraction, filtering, and differentiable matching steps. The images are divided into non-overlapping windows (gray dashed lines), and a subpixel keypoint coordinate (gray dot) is extracted from each window. Keypoints detected in the invalid region created by the perspective warping (shaded region) are removed. Finally, similarity scores, ϕ^i_j, are computed from a source point, $^s\mathbf{q}^i$, to every target point, $^t\mathbf{q}^j$, and used to estimate a matching pseudo-target keypoint.

and the associated inlier score is given by

$$s^i_{\mathrm{I}} = \frac{1}{1 + \exp\left(b\left(\frac{x^i}{a} - 1\right)\right)}, \tag{4.7}$$

where a is the outlier threshold and b controls the score's rate of decline or *sharpness* as the reprojection error increases. This is an alternative parametrization of a sigmoid function. Figure 4.4 shows a plot of the inlier score as a function of reprojection error.

Downweighting poor-quality matches reduces their training impact. Therefore, a large outlier threshold value is chosen to maximize data utilization while ensuring convergence. For a training image size of 240 × 320, we choose a rejection threshold of $a = 50$ pixels and a sharpness parameter of $b = 5$.

4.2.4 Model Estimation

The estimated source-to-target homography, $^t\hat{\mathbf{H}}_s$, is computed with the weighted direct linear transform (DLT) algorithm as described in Section 2.4.4. The overall weight, s^i, for each match is computed as the product of the source keypoint score, the pseudo-target keypoint score, the matching score, and the inlier score:

$$s^i = {}^s s^i_{\mathrm{K}}\ {}^t\hat{s}^i_{\mathrm{K}}\ s^i_{\mathrm{M}}\ s^i_{\mathrm{I}}. \tag{4.8}$$

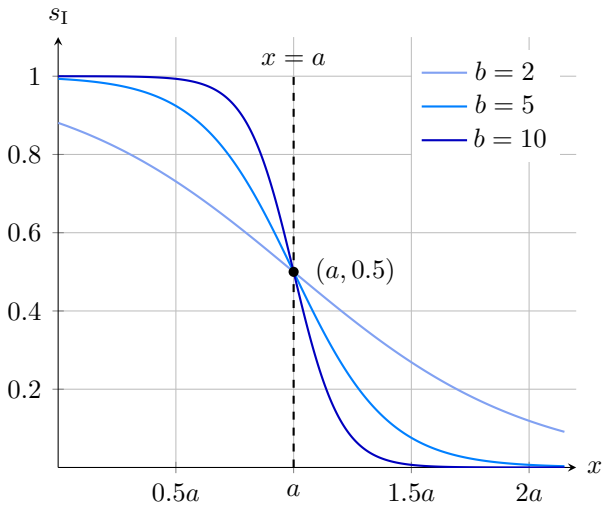


Figure 4.4: Inlier score as a function of the reprojection error, x, at multiples of the outlier threshold, a, for different values of sharpness, b.

4.3 Losses

In this section, we describe the losses used to train our feature network. We first cover our task-based losses followed by the detector and descriptor losses.

4.3.1 Task-Based Loss

We explore three losses to quantify the error between the ground-truth homography and the outputs of the differentiable registration pipeline: a corner loss, a Frobenius norm loss, and a transfer loss. We first describe the computation of error matrices for each loss, followed by the application of a robust loss function and the combination of the individual loss values. The errors are computed *symmetrically*, meaning that we calculate an error matrix in both the forward (source-to-target) and inverse (target-to-source) directions and combine their individual loss contributions to create a balanced loss.

Corner Error

We first normalize the ground-truth and estimated homographies to a pixel range of $[-1, 1]$ to obtain ${}^{t}\bar{\mathbf{H}}_{s}$ and ${}^{t}\hat{\bar{\mathbf{H}}}_{s}$, respectively. This is done to ensure independence from image size. We then compute a residual homography, which is equivalent to the successive application of the estimated and inverse ground-truth (normalized) homographies:

$$\tilde{\mathbf{H}} = {}^{t}\bar{\mathbf{H}}_{s}^{-1}\, {}^{t}\hat{\bar{\mathbf{H}}}_{s}. \tag{4.9}$$

Recall from Section 2.3.2 that we follow the notation of [19] and [33], using $\mathbf{Hq}$ to denote the transformation of point $\mathbf{q}$ by homography $\mathbf{H}$. We extend this notation such that $\mathbf{HQ}$ denotes the transformation of point set $\mathbf{Q}$ by homography $\mathbf{H}$, including the augmentation, matrix multiplication, division, and reprojection steps. The corner error matrix, $\mathbf{R}_{\mathrm{C}}$, is computed as the difference between the normalized image corner coordinates, $\bar{\mathbf{Q}}_{\mathrm{C}}$, and those transformed by the residual homography:

$$\mathbf{R}_{\mathrm{C}} = \bar{\mathbf{Q}}_{\mathrm{C}} - \tilde{\mathbf{H}}\bar{\mathbf{Q}}_{\mathrm{C}}, \quad \bar{\mathbf{Q}}_{\mathrm{C}} = \begin{bmatrix} -1 & 1 & -1 & 1 \\ -1 & -1 & 1 & 1 \end{bmatrix}, \quad \mathbf{R}_{\mathrm{C}} \in \mathbb{R}^{2\times 4}. \tag{4.10}$$

Similarly, we use the inverse of the residual homography,

$$\tilde{\mathbf{H}}^{-1} = {}^{t}\hat{\mathbf{H}}_{s}^{-1}\,{}^{t}\bar{\mathbf{H}}_{s}, \tag{4.11}$$

to compute the corner error for the inverse direction, which we will denote as $\mathbf{R}_{\mathrm{C}}^{'}$:

$$\mathbf{R}_{\mathrm{C}}^{'} = \bar{\mathbf{Q}}_{\mathrm{C}} - \tilde{\mathbf{H}}^{-1}\bar{\mathbf{Q}}_{\mathrm{C}}, \quad \mathbf{R}_{\mathrm{C}}^{'} \in \mathbb{R}^{2\times 4}. \tag{4.12}$$

In both cases, if the estimated and ground-truth homographies are identical, the residual homography will become identity and the errors will reduce to zero matrices.

Frobenius Norm Error

The Frobenius norm error, or Frobenius error for short, draws a more direct comparison between the estimated and ground-truth homography parameters. Boittiaux et al. [70] showed that the reprojection error between two planes offset by a homography $\mathbf{H}$ can be expressed as the squared Frobenius norm of the difference between identity and the homography: $\|\mathbf{I} - \mathbf{H}\|_F^2$. In our case, we interpret the previously defined residual homography, $\tilde{\mathbf{H}}$, as the offset between the estimated and ground-truth planes. Therefore, the Frobenius error matrices in the forward and inverse directions are, respectively, defined as

$$\mathbf{R}_{\mathrm{F}} = \tilde{\mathbf{H}} - \mathbf{I}, \quad \mathbf{R}_{\mathrm{F}}^{'} = \tilde{\mathbf{H}}^{-1} - \mathbf{I}, \quad \mathbf{R}_{\mathrm{F}},\ \mathbf{R}_{\mathrm{F}}^{'} \in \mathbb{R}^{3\times 3}. \tag{4.13}$$

As with the corner error, if the estimated and ground-truth homographies are equal, the errors will reduce to zero matrices.

Transfer Error

In contrast to the homography-based losses, transfer error operates on the matches of the differentiable pipeline. Transfer error is defined as reprojection error between source and pseudo-target keypoints when transformed into the same image with the ground-truth homography. The forward and inverse transfer error matrices are defined as

$$\mathbf{R}_{\mathrm{T}} = {}^{t}\mathbf{H}_{s}\,{}^{s}\mathbf{Q} - {}^{t}\hat{\mathbf{Q}}, \quad \mathbf{R}'_{\mathrm{T}} = {}^{t}\mathbf{H}_{s}^{-1}\,{}^{t}\hat{\mathbf{Q}} - {}^{s}\mathbf{Q}, \quad \mathbf{R}_{\mathrm{T}},\ \mathbf{R}'_{\mathrm{T}} \in \mathbb{R}^{N_{\mathrm{KS}} \times 2}, \tag{4.14}$$

respectively, where ${}^{s}\mathbf{Q}$ is the set of source keypoints and ${}^{t}\hat{\mathbf{Q}}$ is the set of corresponding pseudo-target keypoints.

Robust Loss Function

To improve training convergence in the presence of large estimation errors, the error matrices are passed element-wise through a robust loss function. We use Welsch loss [71, 72], which can be expressed as

$$f_{\mathrm{robust}}(r) = 1 - \exp\left(-\frac{1}{2}\left(\frac{r}{c}\right)^{2}\right) \tag{4.15}$$

for scalar error r, where c is a scaling hyperparameter. Figure 4.5 shows that for errors $|r| < c$, Welsch loss behaves similarly to squared-error loss, but the gradient diminishes past this point. Empirically, we chose a threshold value of $c = 0.1$ for all task-based losses.

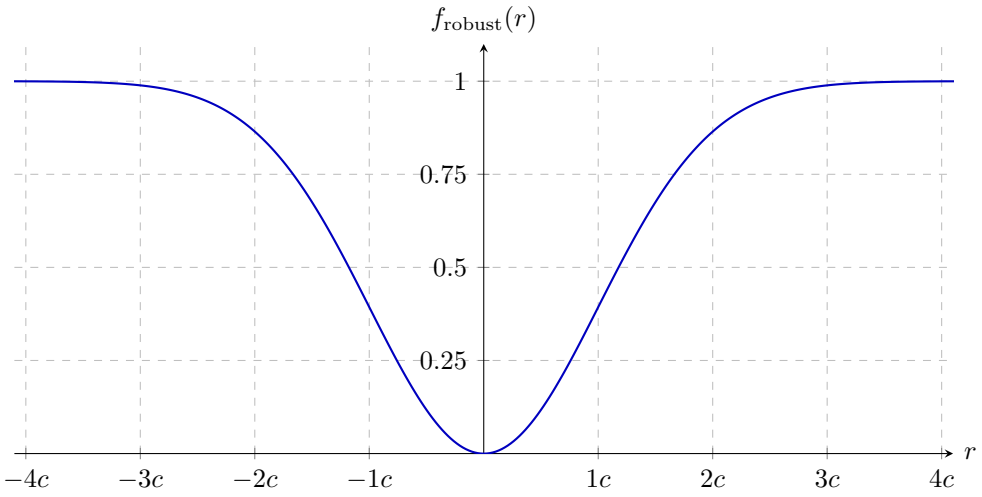


Figure 4.5: Welsch loss output for error r as multiples of threshold c.

For error matrix $\mathbf{R} \in \mathbb{R}^{N \times M}$, the corresponding loss is taken as the average of the

robust loss outputs:

$$\mathcal{L}(\mathbf{R}) = \frac{1}{NM} \sum_{i=1}^{N} \sum_{j=1}^{M} f_{\text{robust}}\left(r_{ij}\right). \tag{4.16}$$

The final scalar loss is averaged between the forward and inverse directions.

4.3.2 Descriptor Loss

In addition to task-oriented training losses, we compute the contrastive descriptor loss from MultiPoint [33], $\mathcal{L}_{\text{D}}$. It is a hinge loss that contains two components: the positive distance loss, $\mathcal{L}_{\text{P}}$, encourages similarity between corresponding descriptors, and conversely, the negative distance loss, $\mathcal{L}_{\text{N}}$, promotes the divergence of non-corresponding descriptors. Recall that the descriptor map is semi-dense, defining one descriptor per cell. First, the ground-truth homography is used to determine binary correspondence values, g^{ij}, between the centers of the source and target cells. The positive distance loss for corresponding cells (i.e., $g^{ij} = 1$) is computed as

$$\mathcal{L}_{\text{P}}^{ij} = \max(0, m_{\text{P}} - {}^{s}\mathbf{d}^{i^T}\, {}^{t}\mathbf{d}^{j}), \tag{4.17}$$

where ${}^{s}\mathbf{d}^{i}$ and ${}^{t}\mathbf{d}^{j}$ are the descriptors for source cell i and target cell j, respectively, and m_{P} is the positive margin of the hinge loss. Similarly, the negative distance loss for non-corresponding cells (i.e., $g^{ij} = 0$) is computed as

$$\mathcal{L}_{\text{N}}^{ij} = \max(0, {}^{s}\mathbf{d}^{i^T}\, {}^{t}\mathbf{d}^{j} - m_{\text{N}}), \tag{4.18}$$

where m_{N} is the negative margin of the hinge loss. Given the high ratio of non-corresponding to corresponding descriptor pairs, the positive distance loss is weighted by λ_{P}. For a single source-target cell pair, the descriptor loss is expressed as

$$\mathcal{L}_{\text{D}}^{ij} = \lambda_{\text{P}} g^{ij} \mathcal{L}_{\text{P}}^{ij} + (1 - g^{ij}) \mathcal{L}_{\text{N}}^{ij}. \tag{4.19}$$

As in [33], we use a positive margin of $m_{\text{P}} = 1.0$, a negative margin of $m_{\text{N}} = 0.2$, and a positive distance loss weighting of $\lambda_{\text{P}} = 250$. The overall descriptor loss is averaged over all source-target cell pairs.

4.3.3 Detector Loss

We also use a modified version of the detector loss from MultiPoint [33]. The detector loss is formulated as a per-cell classification problem that uses keypoint pseudo-ground truth

generated from homographic adaptation (as described in Section 3.2.2). We find, however, that the provided ground truth is imbalanced, as the "no keypoint" case is significantly overrepresented, leading to low detection probabilities that are more susceptible to noise. To counteract this issue, we opt to use a *weighted* categorical cross entropy loss. For cell i, the loss is expressed as

$$\mathcal{L}_{\mathrm{K}}^{i} = -\sum_{j=1}^{65} \rho_j y_j^i \log\left(\frac{\exp(k_j^i)}{\sum_{l=1}^{65}\exp(k_l^i)}\right), \tag{4.20}$$

where $\mathbf{k}^i$ is the unnormalized cell response, $\mathbf{y}^i$ is the one-hot encoded label, and $\boldsymbol{\rho}$ contains the weights of each cell element. Empirically, we choose $\rho_j = 64/65$ for $j = 1, \ldots, 64$ and $\rho_{65} = 1/65$ for the dustbin. The final detector loss, $\mathcal{L}_{\mathrm{K}}$, is averaged over all cells in the source and target images.

4.3.4 Overall Loss

Finally, we arrive at an expression for the overall loss, a weighted sum of the corner loss ($\mathcal{L}_{\mathrm{C}}$), Frobenius loss ($\mathcal{L}_{\mathrm{F}}$), transfer loss ($\mathcal{L}_{\mathrm{T}}$), descriptor loss ($\mathcal{L}_{\mathrm{D}}$), and detector loss ($\mathcal{L}_{\mathrm{K}}$), with respective weights λ_{C}, λ_{F}, λ_{T}, λ_{D}, and λ_{K}:

$$\mathcal{L} = \lambda_{\mathrm{C}}\mathcal{L}_{\mathrm{C}} + \lambda_{\mathrm{F}}\mathcal{L}_{\mathrm{F}} + \lambda_{\mathrm{T}}\mathcal{L}_{\mathrm{T}} + \lambda_{\mathrm{D}}\mathcal{L}_{\mathrm{D}} + \lambda_{\mathrm{K}}\mathcal{L}_{\mathrm{K}}. \tag{4.21}$$

The relative loss weightings are set empirically from a coarse tuning stage, the results of which are discussed in Sections 5.4 and 5.7.

Chapter 5

Results

In this chapter, we present results from the performance evaluation of our feature network against a set of baseline methods. We first cover the experimental setup, including the dataset, baselines, homography estimation pipeline, performance metrics, and training details. Then, we present the registration and standalone feature performance for each method. We further compare the effectiveness of the task-based losses. Finally, we examine the impact of heuristic-based homography rejection on the distribution of registration error.

5.1 Dataset

For training and testing, we use the MultiPoint dataset [33], which consists of aligned pairs of thermal and visible-spectrum images. These images were captured by a unmanned aerial vehicle (UAV) with two downward-facing cameras over an agricultural area. To account for the offset in shutter times and different fields of view between the two cameras, the thermal and visible-spectrum images were aligned offline using an (expensive) mutual information-based registration procedure and cropped to a matching resolution of 512×640 pixels. The data collection was performed at a high altitude such that the depth variation within the environment can be considered negligible, approximating a planar scene. Therefore, after the images have been aligned, they are considered to have dense, pixel-to-pixel correspondence. Additionally, this dataset provides keypoint pseudo-ground truth from the homographic adaptation process described in Section 3.2.2. Data was collected over ten flights on two separate days between 09:00 a.m. and 03:00 p.m. under varying lighting conditions and large temperature changes. From these potentially overlapping flight paths, seven flights were used for training data and three for testing data. In total, the dataset contains 13,731 thermal-visible image

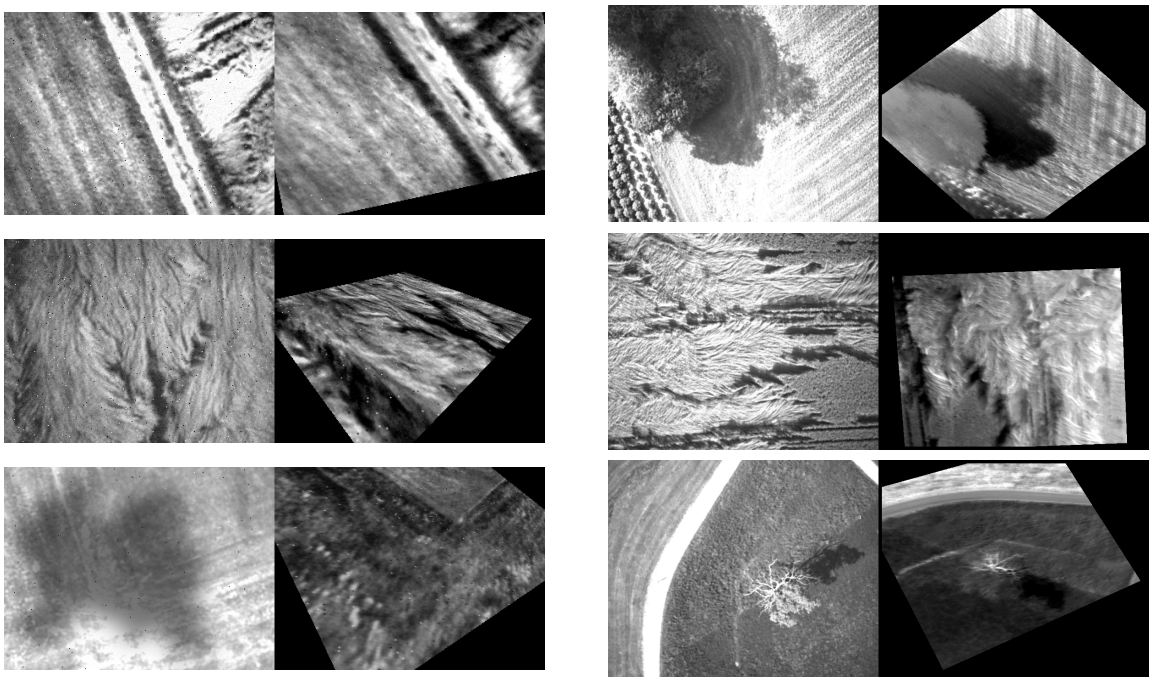
pairs: 9,340 training pairs and 4,391 testing pairs. We further split the training data into training and validation subsets of 7,472 (80%) and 1,868 (20%) image pairs, respectively.

During training, images are randomly cropped to a size of 240 × 320 pixels and undergo both homographic and photometric augmentation. Homographic augmentation is the process of randomly sampling a homography and applying it to one of the aligned images. After an additional central crop of the input image, the homography is composed by applying the following elementary transformations in random order: translation, scaling about the image's center, in-plane rotation, and symmetric perspective distortion. In practice, these transformations are applied to the image corner coordinates, and the direct linear transform (DLT) algorithm described in Section 2.4 returns the final homography matrix, which constitutes ground truth. Photometric augmentation takes place for *both* images in a pair and includes the following operations: brightness offset; contrast scaling about the mean intensity; additive gaussian noise; additive speckle noise; the superposition of translucent, blurred ellipses to simulate shade; and the application of Gaussian-weighted, directional averaging filters to simulate motion blur. The parameters for each image transformation, both homographic and photometric, are uniformly sampled from a predefined range. Finally, during training only, in addition to thermal-visible image pairs, same-spectrum pairs are shown to the model (i.e., a single image and a warped version, both with photometric augmentation applied).

At test time, full-sized images (512 × 640 pixels) are used. Only homographic augmentation is applied to the test images, and the parameters for the central crop, scaling, and perspective distortion transformations are sampled from a more conservative range than during training. Figure 5.1 shows examples of image pairs from the MultiPoint dataset with training and testing augmentation applied.

5.2 Baselines

We compare our model to a range of different methods. First, to gain an understanding of how visible-spectrum-based features perform on a cross-spectral task, we evaluate the Scale-Invariant Feature Transform (SIFT) [34], Oriented FAST and Rotated BRIEF (ORB) [35], and SuperPoint [19] methods. SIFT and ORB are the de facto standard handcrafted feature extractors for a variety of computer vision tasks. As discussed in Section 2.4.1, SIFT primarily operates on local image gradients while ORB relies on binary detection and description algorithms. SuperPoint is a seminal work on dense learned features. We use its publicly available pretrained weights for evaluation. In addition, we compare our model to well-known cross-spectral methods: Log-Gabor His-



(a) Training augmentation

(b) Testing augmentation

Figure 5.1: Augmented thermal-visible image pairs from the MultiPoint dataset during training (left) and testing (right). Starting with the aligned pair, one of the images is warped with a randomly sampled homography (homographic augmentation). Additionally, during training only, photometric augmentation is applied to both images, including brightness changes, contrast changes, additive speckle and Gaussian noise, simulated motion blur, and simulated shade.

togram Descriptor (LGHD) [55] and MultiPoint [33]. LGHD is a handcrafted descriptor that applies a bank of Log-Gabor filters to combine frequency and spatial information in an attempt to overcome nonlinear intensity variations. We use the Python implementation of LGHD provided by [33], which pairs the LGHD descriptor with the Features from Accelerated Segment Test (FAST) detector. Finally, MultiPoint is a cross-spectral extension of SuperPoint as described in Section 3.2.2. Since MultiPoint is trained on the same dataset as our method, we retrained the MultiPoint network after splitting the original training set into training and validation subsets. We also trained a modified version of MultiPoint that uses our weighted version of the detector loss (in place of the original, unweighted loss), which we refer to as MultiPoint-W.

5.3 Experimental Procedure

In this section, we describe our procedure to evaluate each method's performance. At a high level, we extract features from augmented thermal-visible image pairs and feed the responses through a homography estimation pipeline. We evaluate performance using standard feature detection and matching statistics as well as a registration error between the estimated and ground-truth homographies. These results are aggregated over the test set. Below, we describe the homography estimation pipelines, followed by the performance metrics.

5.3.1 Estimation Pipelines

We use two different pipelines for evaluation: a "classical" pipeline that follows a standard, feature-based registration procedure as outlined in Section 2.4 and a "weighted" pipeline that is similar to the homography estimation pipeline used during training.

Classical: In the classical pipeline, the feature extraction step varies by method. Handcrafted methods perform keypoint extraction without modification while feature networks require an additional step to convert the detection heatmap into a set of discrete interest points. For this step, we adopt the non-maximum suppression (NMS) process used by [33]. We first threshold the detection heatmap and, starting with the highest-probability pixel, iteratively prune lower-scoring neighbors. We use a probability threshold of 0.05 by default, but for MultiPoint and SuperPoint, we use their original provided threshold of 0.015. Further, we use a neighborhood radius of four pixels for pruning. After keypoint extraction, the pipeline follows the standard procedure outlined in Section 2.4:

mutual nearest neighbor (MNN) matching, random sample consensus (RANSAC) for outlier rejection, and model estimation via DLT, which is then refined by the Levenberg–Marquardt algorithm (LMA).

Weighted: The weighted pipeline, on the other hand, is tailored to learned features. It is a modified version of the training pipeline with a different outlier rejection method. As described in Section 4.2, the pipeline uses windowed differentiable keypoint extraction, a differentiable zero-normalized cross-correlation (ZNCC) matcher, and the weighted DLT algorithm for model estimation. Instead of performing outlier rejection with the ground-truth homography as in training, this evaluation pipeline uses RANSAC with weighted random sampling. The sampling weight for each correspondence i is given by

$$\omega^i = \frac{{}^s s_{\mathrm{K}}^i \; {}^t \hat{s}_{\mathrm{K}}^i \; s_{\mathrm{M}}^i}{\sum_{j=1}^{N_{\mathrm{M}}} \left({}^s s_{\mathrm{K}}^j \; {}^t \hat{s}_{\mathrm{K}}^j \; s_{\mathrm{M}}^j \right)}, \tag{5.1}$$

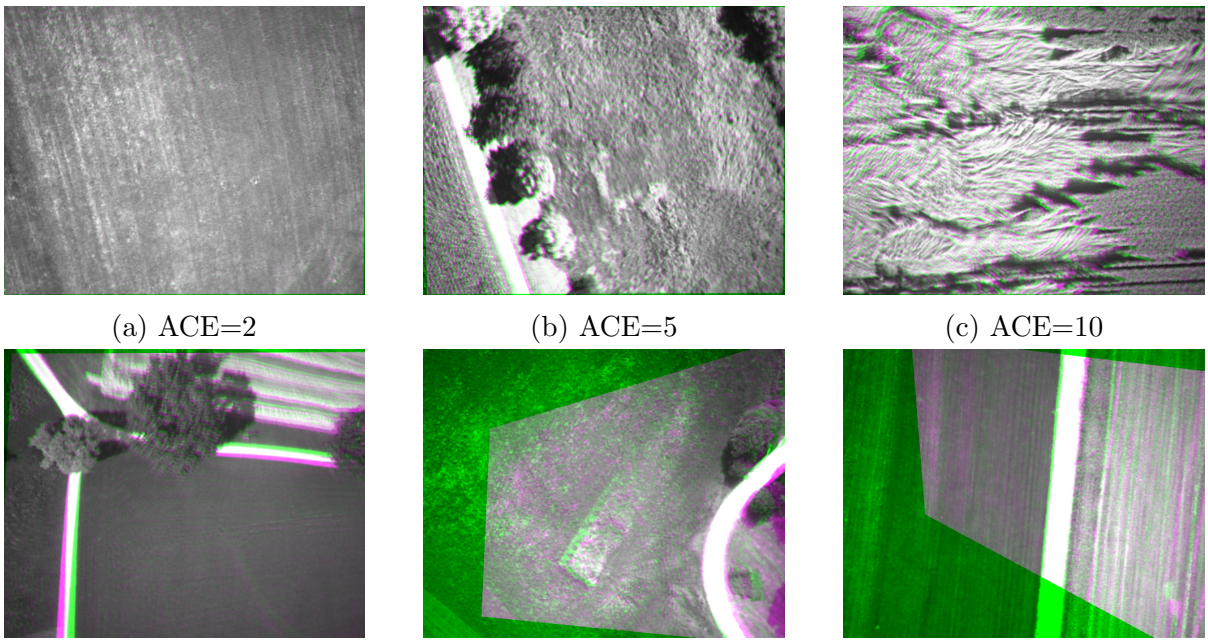
where N_{M} is the total number of correspondences. Inlier matches identified by the weighted RANSAC algorithm are assigned an inlier score of one ($s_{\mathrm{I}}^i = 1$) and all other matches receive an inlier score of zero. With this change, model estimation proceeds with the weighted DLT algorithm as outlined in Section 4.2.4.

5.3.2 Performance Metrics

Given the ground-truth homography and pipeline outputs, we compute the registration and feature matching performance metrics as follows.

Registration: There is no single, standard metric to quantify the similarity between two homographies. Some works on homography regression networks directly compare the parameters of the ground-truth and estimated homographies [57], similar to the Frobenius loss presented in Section 4.3.1. However, this error metric lacks interpretability. Therefore, we elect to use average corner error (ACE) as the registration error measure for a single image pair. Similar to the corner error presented in Section 4.3.1, ACE is defined as the average reprojection error of the image corners, computed as the L2 distance between the original corner locations and those obtained by applying the ground-truth homography followed by the inverse estimated homography. For a given image pair, ACE is expressed (in pixels) as

$$e_{\mathrm{C}} = \frac{1}{4} \left\| \mathbf{Q}_{\mathrm{C}} - \left({}^t\hat{\mathbf{H}}_s{}^{-1} \, {}^t\mathbf{H}_s \right) \mathbf{Q}_{\mathrm{C}} \right\|_2, \quad \mathbf{Q}_{\mathrm{C}} = \begin{bmatrix} 0 & 0 & W-1 & W-1 \\ 0 & H-1 & 0 & H-1 \end{bmatrix}, \tag{5.2}$$



(a) ACE=2 (b) ACE=5 (c) ACE=10

(d) ACE=20 (e) ACE=100 (f) ACE=135

Figure 5.2: Visualization of alignment error for different average corner error (ACE) values (expressed in pixels) on test images of shape 512 × 640. Each panel shows a rectified image in green overlaid with a "recovered" version of the image (i.e., warped by the ground-truth homography followed by the estimated inverse homography) in purple.

where $\mathbf{Q}_C$ is the set of image corner coordinates.[1] Figure 5.2 shows examples of alignment attempts at varying ACE levels.

For each method, we evaluate the overall registration performance using the *distribution* of errors over the test dataset. We compute the first quartile (Q_{25}), median, third quartile (Q_{75}), and median absolute deviation (MAD) of ACE. Note that if a homography cannot be estimated (i.e., fewer than four matches are identified), the ACE is set to 999.

Features: Although features are intermediate quantities, there exist metrics that are representative of a feature extractor's success on a variety of downstream tasks. For all image pairs in the dataset, we extract features and perform matching in both directions (i.e., source-to-target and target-to-source). We then compute the following metrics: repeatability, matching score (M. Score), and mean matching accuracy (MMA). For a given image pair, repeatability is the ratio of keypoints detected at the same physical locations in both images to the average number of detections in the overlapping image region. This metric quantifies the detector's ability to reidentify points under geometric and photo-

[1]Recall that point set $\mathbf{Q} \in \mathbb{R}^{2\times N}$ transformed by homography $\mathbf{H}$ is denoted as $\mathbf{HQ} \in \mathbb{R}^{2\times N}$.

metric image variations. M. Score is a combined measure of detector and descriptor performance [19], calculated as the ratio of correct matches to the average number of detections in the overlapping image region. MMA measures a descriptor's discriminating power by computing the proportion of reported matches that are correct. For each of these three metrics, we take the per-image average over the dataset, following the same procedure as [19, 33], and use a reprojection threshold of four pixels. Additionally, we report the average number of keypoints per image, N_{K}. While this is an important metric on its own for ensuring a sufficient number of detected keypoints, it also provides context for other metrics, as N_{K} is positively correlated with repeatability and often negatively correlated with descriptor-based metrics such as M. Score and MMA.

5.4 Training Details

Of the three proposed task-based losses, we opt to use *only* the transfer loss with a weighting of $\lambda_{\mathrm{T}} = 1$ in our final model. During a coarse tuning stage, models trained with transfer (i.e., matching-based) loss significantly outperformed those trained with the corner or Frobenius (i.e., homography-based) losses. A quantitative comparison and discussion of the models' relative performance is presented in Section 5.7. In addition, we use the descriptor and weighted detector losses with an equal weighting of $\lambda_{\mathrm{D}} = \lambda_{\mathrm{K}} = 1$ as in the MultiPoint-W network, which we use to initialize our network weights. We train our model with the PyTorch package [73] and use the Adam optimizer [74] with learning rate $\alpha = 10^{-5}$ and batch size $N_{\mathrm{B}} = 32$.

To select a final model, we track our feature network's performance with the weighted pipeline by computing the registration metrics on the validation subset every 10 epochs. During this evaluation, we apply test-level augmentation as described in Section 5.1 (full-sized images, no photometric augmentation, milder homographic augmentation than training, and exclusively cross-spectral image pairs). Although we impose an early stopping condition to select the model that minimizes the third quartile of ACE on the validation subset (Q_{75}), we observe continued improvement (decrease) of Q_{75} for the duration of training. Our model therefore trains for the maximum number of epochs, 1000. We also extend the training of MultiPoint and MultiPoint-W for the same number of epochs at the same learning rate in order to achieve a fair comparison.

Our approach to early stopping differs from the standard procedure of minimizing the validation *loss* because we train and evaluate on separate pipelines. It is worth noting, however, that we also compute the validation loss every 10 epochs, which consistently decreases throughout training, indicating no signs of overfitting to the training data.

Pipeline	Method	Q_{25}	Median	Q_{75}	MAD
Classical	SIFT [34]	7.40	416	539	351
	ORB [35]	357	440	549	100
	SuperPoint [19]	328	465	673	188
	LGHD [55]	427	471	694	67.2
	MultiPoint [33]	2.81	5.06	50.5	3.09
	MultiPoint-W	2.63	4.52	24.8	2.58
	Ours	2.65	4.23	14.9	2.13
Weighted	SuperPoint	309	442	556	117
	MultiPoint	2.81	4.79	91.5	2.67
	MultiPoint-W	2.51	4.15	13.5	2.25
	Ours	**2.26**	**3.72**	**9.13**	**1.90**

Table 5.1: Registration performance for all pipeline-method combinations as measured by the distribution of average corner error (ACE) (expressed in pixels) for test images of shape 512×640. The best result for each metric is bolded. Our model with the weighted pipeline outperforms all other pipeline-method combinations.

5.5 Registration Performance

In this section, we evaluate the registration performance of our learned features against the baseline methods by comparing their respective average corner error (ACE) distributions on the test set. Homography estimation is performed with the classical pipeline for all methods and the weighted pipeline for the learned features. Figure 5.3 presents a box plot of ACE on a logarithmic scale for all methods with their original proposed pipelines. In addition, Table 5.1 summarizes the key error distribution statistics. It is clear that the visible-spectrum-based features, both handcrafted and learned, demonstrate insufficient performance on the task of cross-spectral registration; SIFT, ORB, and SuperPoint all achieve a median error around 400 pixels. Surprisingly, the handcrafted cross-spectral method, LGHD, underperforms even the visible-spectrum features. As noted by [33], LGHD struggles in part due to its lack of viewpoint invariance. Learned methods may be better equipped than handcrafted descriptors to achieve simultaneous invariance to large appearance and geometric changes provided that the data sufficiently captures these variations. Indeed, all the cross-spectral learned methods achieve a reasonable median error in the neighborhood of 5 pixels. Our method, however, outperforms all baseline methods on the task of cross-spectral registration, achieving the lowest Q_{25}, median, and Q_{75} values, as well as the tightest spread of errors. Notably, over 75% of registration estimates from our method produced an ACE of 10 pixels or less.

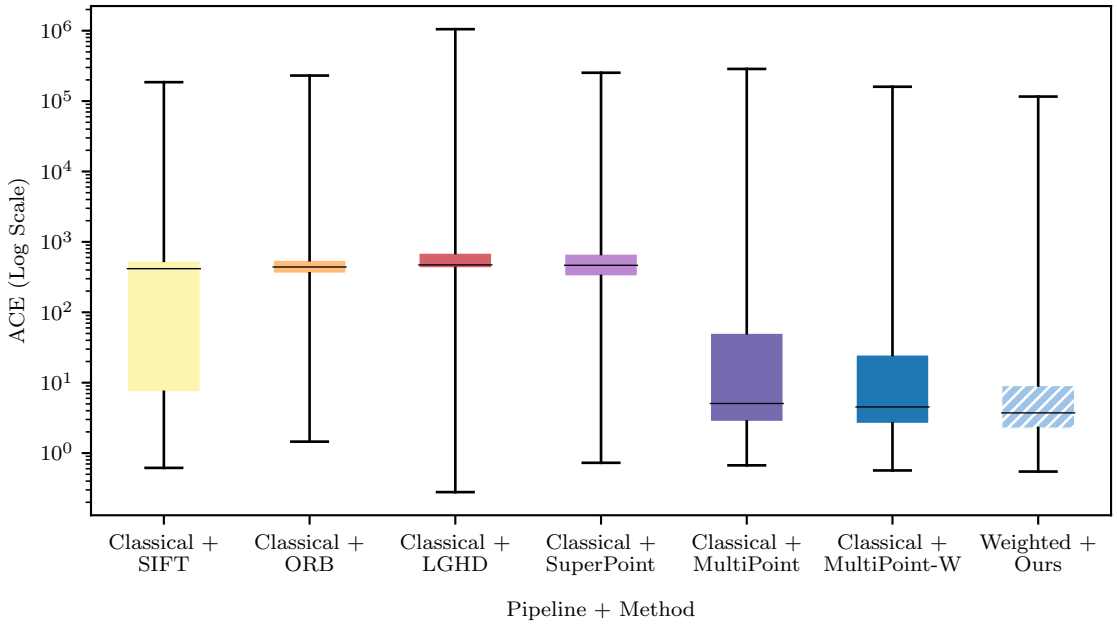


Figure 5.3: Logarithmic scale box plot of average corner error (ACE) expressed in pixels for all methods with their original pipelines. The plot whiskers extend to all data points. Solid colors denote the use of the classical pipeline and stripes denote the weighted pipeline. Our method with the weighted pipeline outperforms all baselines with their original (classical) pipelines.

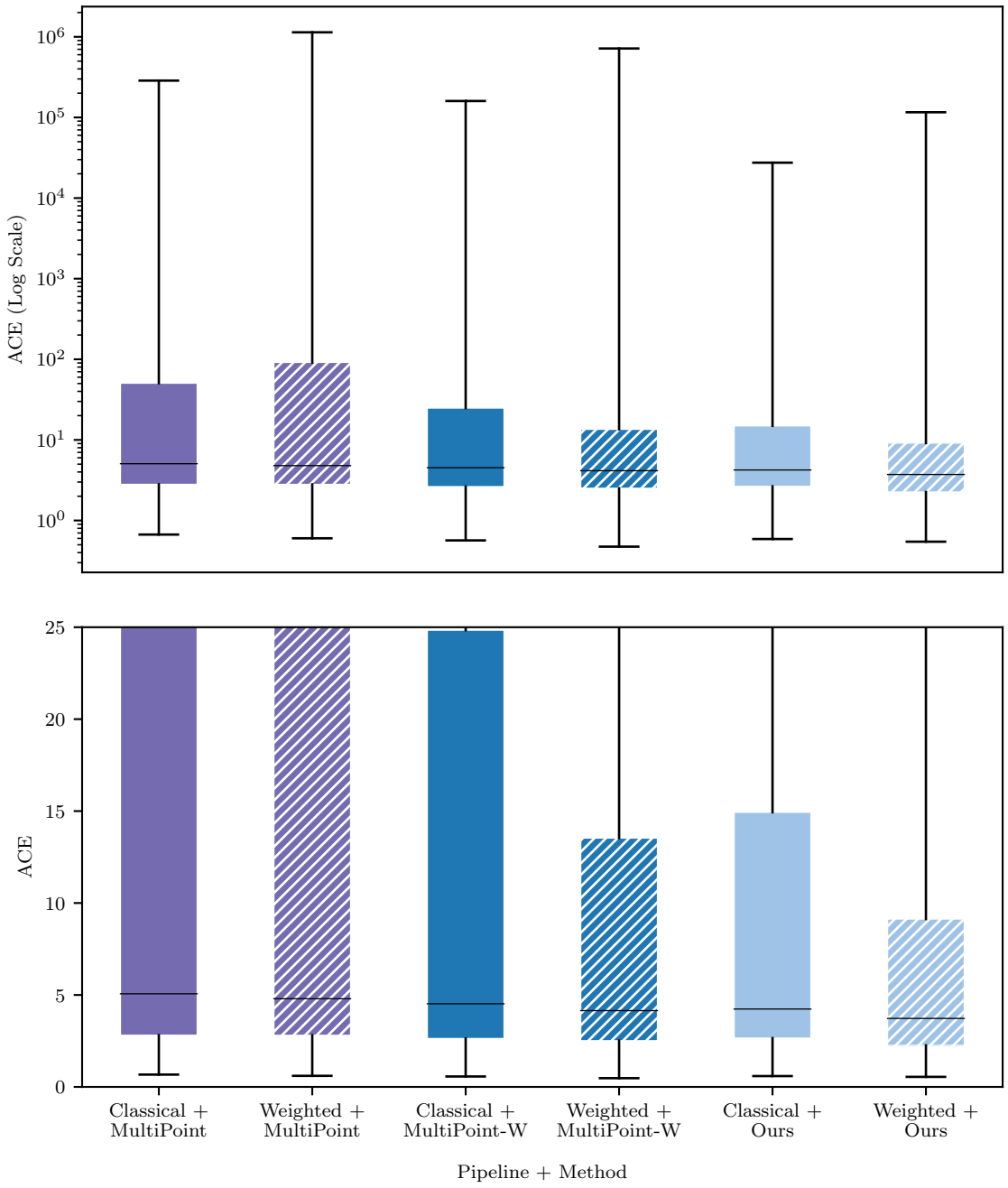


Figure 5.4: Box plots of average corner error (ACE) expressed in pixels in logarithmic (top) and linear (bottom) scale for the cross-spectral learned feature methods on both pipelines. The plot whiskers extend to all data points. Solid colors denote the use of the classical pipeline and stripes denote the weighted pipeline. Our method outperforms the baselines on both pipelines.

Next, we more closely examine the relative performance of the cross-spectral learned features on both pipelines, focusing on a smaller range of errors. A homography is a high-degree-of-freedom transform. In contrast to other global transforms, the flexibility afforded by these additional degrees of freedom can make homography estimation more sensitive to estimation inaccuracies or insufficient match coverage, which can lead to extremely high registration errors. During operation, however, the difference between an ACE of 1,000 and 10,000 pixels is functionally irrelevant. Although it is difficult to draw a binary distinction between a success and a failure, we chose to examine the error distribution below 25 pixels, which is approximately 5% of the test image height.

Figure 5.4 shows two box plots of the average corner error for MultiPoint, MultiPoint-W, and our model on the classical and weighted pipelines; one plot shows the entire distribution on a logarithmic scale and the other shows errors below 25 pixels on a linear scale. The linear scale highlights the advantage of our method, showing more clearly that our method achieves consistently lower error than the baselines on both pipelines.

This comparison leads to two unexpected observations. First, despite not training with our differentiable registration pipeline, MultiPoint-W achieves lower registration error on the closely-related weighted pipeline than the classical pipeline. More notably, training on the differentiable registration pipeline improves the performance of our model on the nondifferentiable, *classical* pipeline. In addition, the weighted detector loss significantly improves the registration performance of the MultiPoint network. The biggest impact can be seen on the third quartile of error, bringing Q_{75} down from 50.5 to 24.8 pixels with the classical pipeline and 91.5 to 13.5 pixels with the weighted pipeline. We explore these points further in Section 5.6.

5.6 Feature Metrics

Although we achieve the best registration results with the weighted pipeline, feature performance on the classical pipeline is important from the standpoint of robot integration and versatility. The classical pipeline contains standard components of a visual navigation system. We compute the standalone feature performance metrics (N_{K}, repeatability, M. Score, and MMA) for each method on the classical pipeline as an indication of effectiveness on downstream tasks beyond matching and registration, such as 3D reconstruction, mapping, odometry, and localization. Note that we do not consider these metrics on the weighted pipeline because this pipeline does not assign equal importance to individual keypoints and matches. Table 5.2 summarizes the results. We again observe that our method achieves the best performance, reporting the highest repeatability, M. Score, and

Method	N_K	Repeatability	M. Score	MMA
SIFT [34]	643	0.269	0.0294	0.103
ORB [35]	359	0.273	0.0188	0.0572
SuperPoint [19]	248	0.183	0.0274	0.0876
LGHD [55]	1243	0.234	0.00448	0.0371
MultiPoint [33]	438	0.291	0.111	0.295
MultiPoint-W	1609	0.446	0.107	0.278
Ours	1708	**0.454**	**0.124**	**0.317**

Table 5.2: Feature performance metrics for all methods on the classical pipeline. The best result for each applicable metric is bolded. Our method achieves the best performance in all categories.

MMA despite being trained on a pipeline with different discrete keypoint extraction and matching modules than the classical pipeline. This result highlights the inherent flexibility of feature-based algorithms and supports the choice to constrain training to the feature network through the use of a non-learned differentiable pipeline.

We also revisit the impact of the weighted detector loss. Our model and MultiPoint-W exhibit higher repeatability scores than the original MultiPoint by a wide margin, improving from 29% to 45%. This is likely due to an increased number of keypoints. In fact, MultiPoint's repeatability can be artificially inflated by lowering the detection threshold used during discrete keypoint extraction. This is demonstrated in Table 5.3, which shows MultiPoint's feature metrics recomputed at different detection thresholds. As noted by [19], however, high repeatability is not necessarily an indicator of strong downstream performance. In fact, we observe that for a given feature, an increased number of keypoints is often negatively correlated with descriptor-based metrics, likely due to an increased risk of false positive matches. Table 5.3 illustrates this effect with MultiPoint: for a comparable number of keypoints, MultiPoint's repeatability matches that of our method, but this repeatability increase comes at the expense of M. Score and MMA. In contrast, our method (and to an extent, MultiPoint-W), achieves high levels of repeatability without sacrificing matching accuracy, leading to superior downstream performance.

5.7 Task-Based Loss Comparison

In this section, we compare the effectiveness of the transfer, corner, and Frobenius losses. For brevity, we will refer to the descriptor and weighted detector losses as the "base"

Threshold	N_K	Repeatability	M. Score	MMA
1.5×10^{-2}	438	0.291	**0.111**	**0.295**
5.0×10^{-3}	910	0.369	0.0995	0.271
1.5×10^{-3}	1731	**0.460**	0.0854	0.254

Table 5.3: MultiPoint network's feature performance metrics with the classical pipeline at varying detection thresholds. The best result for each applicable metric is bolded. A lower detection threshold results in more keypoints, which increases the repeatability but reduces the matching performance.

losses. Recall that these were used to train MultiPoint-W, while the original MultiPoint model used the unweighted detector loss. As mentioned in Section 5.4, out of the task-based losses, our selected model included only the transfer loss with a weighting of $\lambda_T = 1$. We trained two additional models that substitute the transfer loss for the corner loss or Frobenius loss, each with a weighting of 0.1. All other training parameters, such as the base losses, number of epochs, and initial weights, did not change. We tested the registration performance of these model variants on the weighted pipeline and present their ACE distributions alongside our selected model and MultiPoint-W in Table 5.4.

We observe that models trained with the corner or Frobenius losses performed worse than the transfer loss-based model and even the base losses alone. These homography-based models exhibit higher Q_{25}, median, and Q_{75} registration error values, as well as a higher variance of errors. This performance decrease is most pronounced for the third quartile of ACE.

While this result may initially be surprising, upon further examination, we see that minimizing a homography-based loss function on our specific training pipeline creates an ill-posed optimization problem. The homography errors are backpropagated through the

Losses	Q_{25}	Median	Q_{75}	MAD
Base (MultiPoint-W)	2.51	4.15	13.5	2.25
Base+Transfer (Ours)	**2.26**	**3.72**	**9.13**	**1.90**
Base+Corner	2.67	4.79	26.7	2.86
Base+Frobenius	2.58	4.50	20.3	2.58

Table 5.4: Distribution of average corner error (ACE) expressed in pixels for models trained with different task-based losses. The best result for each applicable metric is bolded. Models trained with homography-based losses (corner or Frobenius) performed worse than those trained with the match-based loss (transfer) or even the base losses alone.

match locations without considering additional constraints needed to obtain a unique solution, such as the reprojection error between the matches and the model. As a result, we observe an "averaging" effect in which matched keypoint locations are shifted to offset each other's errors in the homography parameters. This lack of constraints is due to our use of ground-truth-based outlier rejection, which accepts all matches within a large range of reprojection error. Further, our findings agree with Gridseth and Barfoot [22], who use this same outlier rejection method in their differentiable pose estimation pipeline. The authors found that their model required a match-based loss in addition to training on localization error.

Our results are also similar, in some sense, to observations from end-to-end learned homography estimation for thermal-visible image pair registration. As mentioned in Section 3.2.1, Özer and Ndigande [57] trained two model types that estimated different homography representations: the four corners of the transformed image versus the homography parameters themselves. Although mathematically equivalent, the model that estimated the corner locations outperformed the parameter-based model. In general, optimizing homography parameters in the absence of a cohesive geometric framework appears to yield substandard results.

Homography-based losses could potentially be effective if coupled with a mechanism that enforces geometric consistency prior to model estimation, such as RANSAC. However, the use of differentiable RANSAC in the training pipeline demands its own set of considerations, such as sufficient gradient flow, convergence, and stability.

5.8 Determinant-Based Filtering

In this section, we explore the feasibility of incorporating our learned feature and homography estimation pipeline into a broader visual navigation application, such as pose estimation for localization. To do so, we consider the nature of high-error estimates and attempt to filter these estimates out based on their matrix properties alone. We then compare the original error distribution to the filtered subset.

Figure 5.5 shows a histogram of the registration errors *above* 25 pixels for cross-spectral learned features on both the classical and weighted pipelines. All methods exhibit a bimodal distribution, with one peak near zero and a second peak close to 450 pixels. We suspect that the failed estimates of the second mode are a result of ill-formed homography matrices. Degenerate homographies compress the entire input image into a point or line. The heatmap in Figure 5.6 shows the ACE values for each internal image coordinate—that is, the resultant ACE value if the input image were compressed

Pipeline	Method	Q_{25}	Median	Q_{75}	Q_{90}	Q_{95}	% Removed
Classical	MultiPoint	2.57	4.16	9.08	68.0	305	15.4%
	MultiPoint-W	2.48	3.99	9.73	80.3	307	10.5%
	Ours	2.55	3.88	8.02	62.4	207	**8.97%**
Weighted	MultiPoint	2.48	3.69	5.99	**11.7**	**23.5**	23.6%
	MultiPoint-W	2.31	3.52	6.43	17.4	50.4	14.9%
	Ours	**2.11**	**3.30**	**5.74**	14.9	38.9	11.7%

Table 5.5: Updated average corner error (ACE) distributions expressed in pixels for all cross-spectral learned feature methods after determinant-based filtering of homography estimates, including the percentage of estimates that were removed. The best result for each metric is bolded. After filtering, our method achieves an ACE of less than 15 pixels for 90% of all estimates.

to that point. The values of this heatmap coincide with the second mode of the ACE distributions.

We chose to use the determinant of the estimated homography matrix, $\left|{}^{t}\hat{\mathbf{H}}_{s}\right|$, to filter out failed registration estimates. Given that near-singular transformation matrices have small determinants, we imposed a minimum determinant threshold on the estimated homographies. We also apply this rejection criterion to the inverse estimated homography to avoid excessively dilative transformations. Therefore, we filter out homographies that fail the following criterion: $1/\eta < \left|{}^{t}\hat{\mathbf{H}}_{s}\right| < \eta$, $\eta = 10$. The determinant is a measure of scale, which means that we are choosing to reject homographies that scale the source image by more than a factor of 10. For a given application, the appropriate threshold value will ultimately depend on the expected amount of scale variation.

Figure 5.7 shows the filtered error histograms superimposed on the originals. It is clear that the majority of the estimates from the second peak were removed with the rejection heuristic. Some high-error estimates remain, particularly on the classical pipeline, leaving a heavy-tailed, unimodal error distribution. In addition, we compute the filtered distribution statistics, including the 90th and 95th error quantiles. These updated results are presented in Table 5.5. Our model with the weighted pipeline achieves errors below 15 pixels for 90% of all image pairs after the removal of just 12% of estimates. The MultiPoint network's 90th error quantile is 11.7 pixels, but this requires the removal of almost 24% of estimates.

Ideally, these failed estimates would not occur, but even the ability to *detect* such failures makes our method of cross-spectral registration much more plausible for use in visual navigation tasks. The optimal thresholds and types of heuristic rejection would

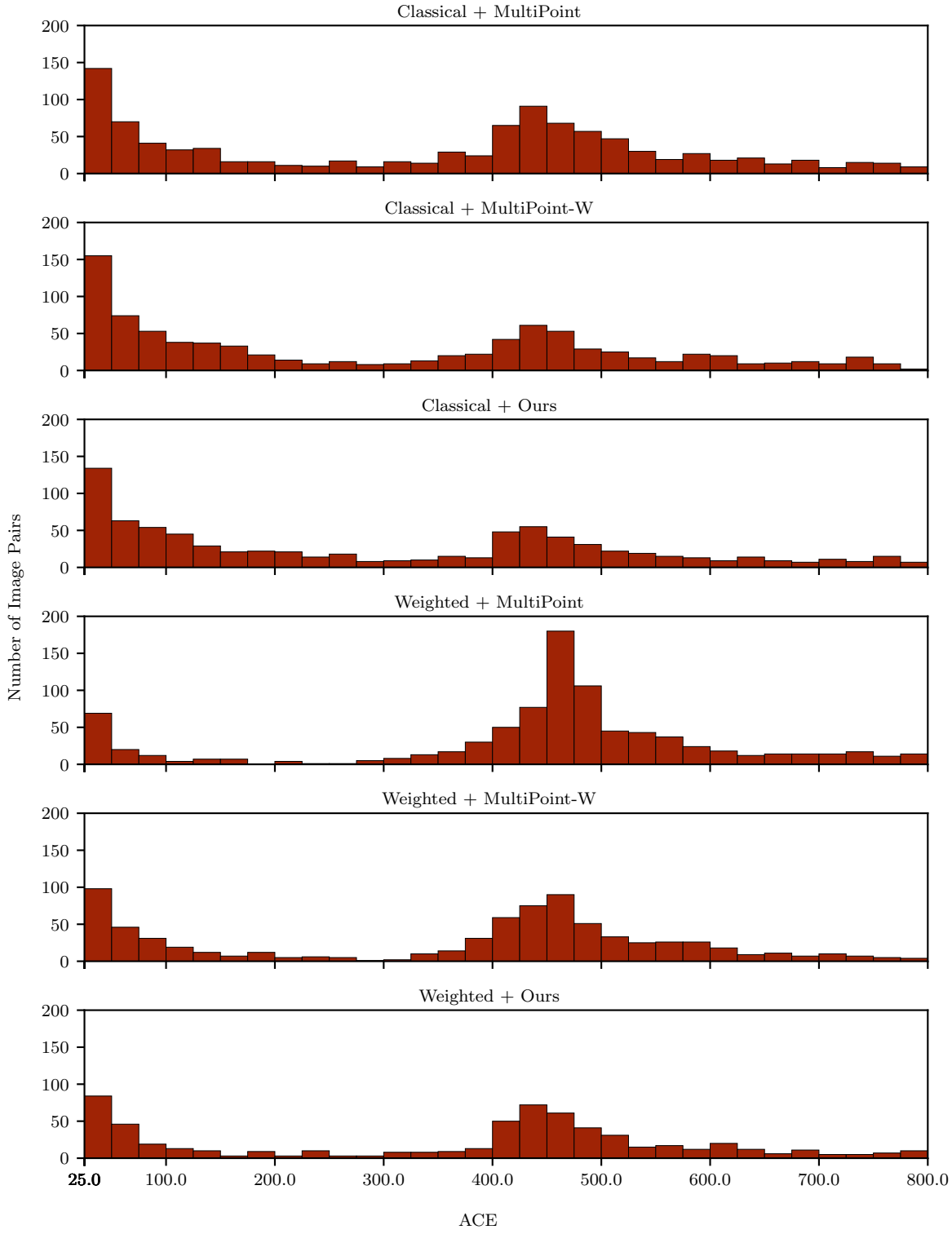


Figure 5.5: Histogram of average corner error (ACE) values above 25 pixels for different pipeline-method combinations. For all methods, there exists a peak of error between 400 and 500 pixels on test images of resolution 512 × 640 pixels.

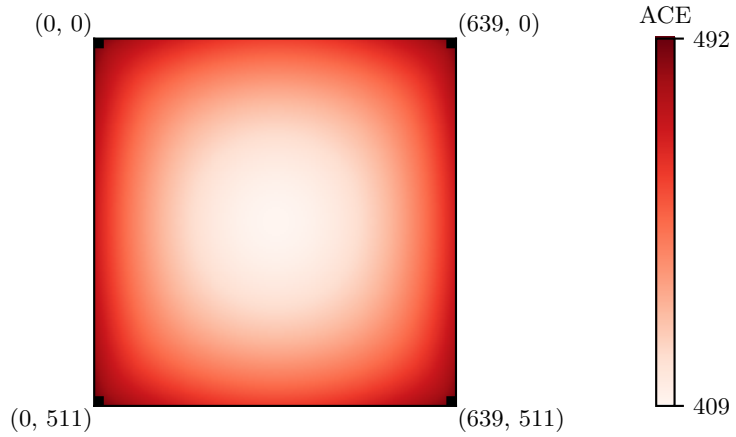


Figure 5.6: Heatmap of the average corner error (ACE) values for the internal coordinates of a 512 × 640 image.

likely require tailoring to the use case and its available domain knowledge. Further, a real robot implementation would include additional error monitoring and handling systems. A UAV localization system, for example, may filter position and orientation (pose) estimates using data from other sensors, such as an inertial measurement unit (IMU). As one component of a larger system, our model could facilitate a range of cross-spectral registration applications.

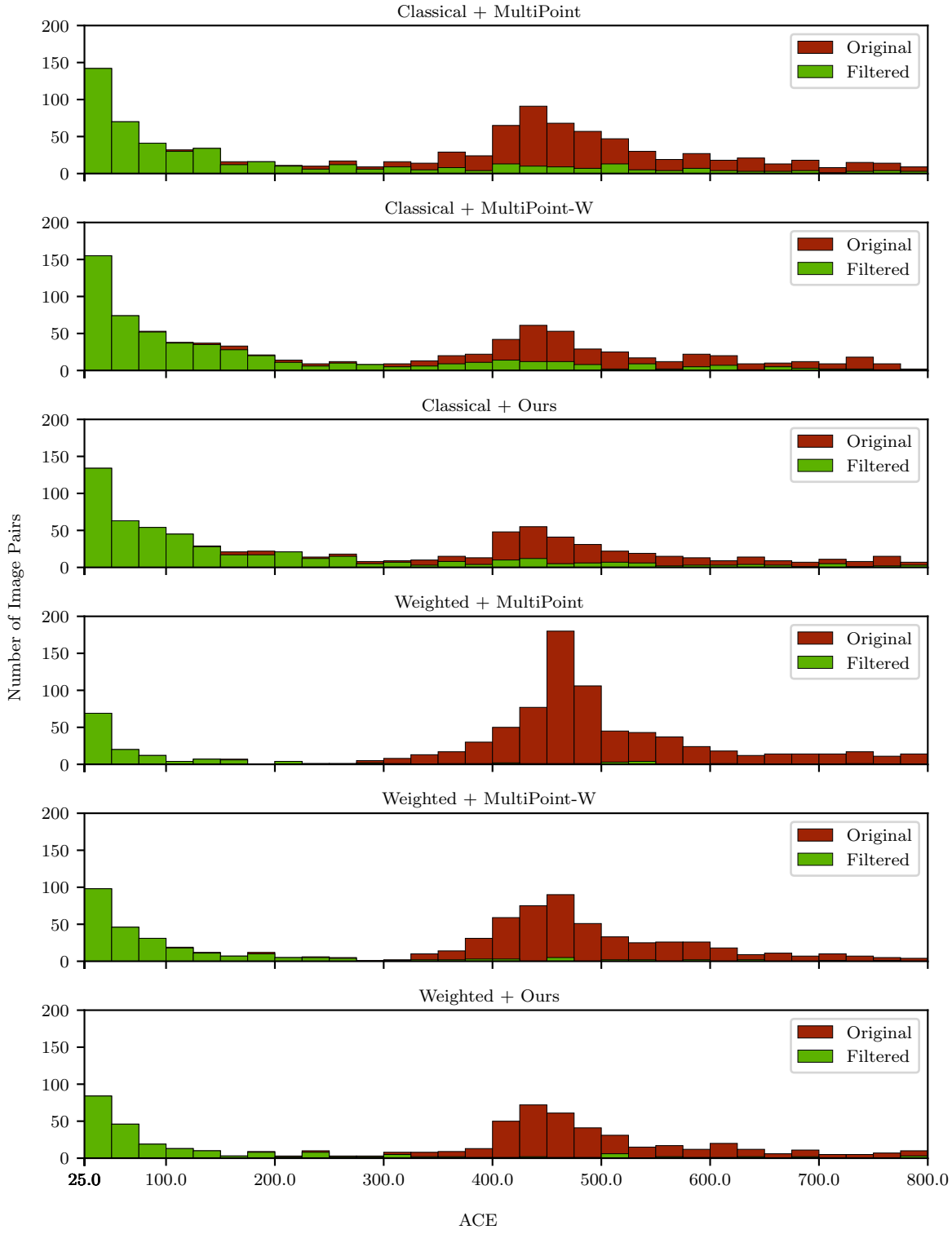


Figure 5.7: Original and filtered average corner error (ACE) histograms above 25 pixels for different pipeline-method combinations. The second mode of the registration error is not present in the filtered histograms.

Chapter 6

Conclusions

In this thesis, we investigated how thermal imagery can be incorporated into existing unmanned aerial vehicle (UAV) perception systems as a means to improve their performance under poor visibility. Visual localization is a crucial capability for UAVs, but despite their maturity, visible-only systems continue to struggle in dark and obscured conditions. Thermal cameras, on the other hand, measure long-wave infrared (LWIR) radiation emitted from the environment, which does not depend on external illumination and penetrates obscurants such as smoke or dust. This advantage of thermal cameras motivated our development of thermal-visible point features as building blocks for cross-spectral visual navigation algorithms.

6.1 Contributions

Our core contribution is a method that guides the process of learning cross-spectral features with the task of image matching. Point features are intermediate quantities with an ambiguous definition. As a result, a core challenge of learning features lies in the choice—or lack thereof—of a supervisory signal. A key limitation of existing supervision approaches is their focus on areas with smaller photometric differences. This strategy does not fully utilize the known geometric relationship between images. We aim to enable the network to learn features in regions with larger domain gaps by taking a task-based approach to supervision. In our framework, we apply our feature network to a pair of thermal and visible-spectrum images that are geometrically related by a known homography. From the network response, we perform differentiable homography estimation and apply our task-based losses to the outputs of this pipeline.

We evaluated our model's cross-spectral matching and registration performance against a set of baseline methods. Registration performance was evaluated with two differ-

ent pipelines: one for learned features (similar to our training pipeline) that performs weighted homography estimation, and a classical image registration pipeline. For matching evaluation, we used the classical pipeline only. We found that our model outperformed all baselines on both matching and registration. We achieved the best registration performance on the specialized weighted pipeline, but our model's improvement over the baselines on the classical pipeline is notable for two reasons. First, performance with the classical pipeline may be more important for integration into an existing visual navigation system. In addition, given the large differences between the training and classical pipelines, this transferability speaks to the versatility of point features and validates our choice to use a differentiable registration pipeline without any learnable parameters.

We trained three separate models that each augmented the standard detection and description losses with a different task-based loss—two homography losses and one matching loss—and compared the results. The matching-based model achieved superior performance, so we opted to use only this loss out of the task-oriented options in our final network. In contrast, augmenting the standard losses with either formulation of homography error yielded worse performance than training with the standard losses alone. Upon further inspection, we determined that homography-based losses were incompatible with our differentiable registration pipeline due to the lack of geometric constraints needed to optimize the match locations from the homography error.

Finally, we investigated the nature of failed registration attempts and experimented with determinant-based filtering to remove ill-conditioned homography estimates. We observed that this eliminated the majority of high-error estimates. This demonstrated the potential viability of incorporating our model into a robotic perception system.

6.2 Potential Improvements

Although our method outperformed the baselines on cross-spectral image matching and registration, a number of training and architectural changes could enhance our results. First, our system involves a large number of hyperparameters. We performed only a coarse hyperparameter search, so maximizing performance would require a more thorough investigation into the most suitable loss weightings, robust loss parameters, number of descriptor channels, learning rate, and batch size. Our network could also benefit from increased training time: our early stopping mechanism was not triggered during training, indicating that the registration performance had not yet plateaued. Finally, the introduction of curriculum learning could improve our training process. This could be accomplished by gradually tightening the robust loss parameters or increasing the mag-

nitude of homographic warps throughout training.

The architectural improvements to our method involve substituting the matching and outlier rejection modules of our differentiable pipeline to reflect the current state of the art and address challenges that we encountered during training, respectively. Learned matchers, such as SuperGlue [75], now rival and even outperform classical matchers. Incorporating SuperGlue or one of its variants into our training and testing pipelines could boost performance while maintaining differentiability. As noted earlier, homography-based losses were ineffective with our training pipeline due to an absence of geometric constraints on the matches. Substituting our current ground truth-based outlier rejection for differentiable random sample consensus (RANSAC) could enforce geometric consistency while retaining differentiability, potentially enabling the use of homography-based registration losses. It should be noted, however, that the weighted RANSAC sampling and homography estimation were crucial to the success of our specialized evaluation pipeline. Therefore, the use of RANSAC in the training pipeline may require the careful incorporation of weights in the sampling, scoring, and estimation of models.

6.3 Future Work

The strong performance of our model in this preliminary investigation opens up several avenues for further exploration and extension of our work. The first follow-up question centers around transferability. We chose to research point features because of their role in multiple distinct components of a visual navigation pipeline. As a next step, our model's performance should be evaluated on tasks beyond registration (e.g., place recognition and pose estimation) and different types of datasets (e.g., non-planar scenes, urban environments, and indoor settings).

Our work focuses on bridging the domain gap between thermal and visible-spectrum images, but we do not consider the temporal changes *within* each spectra over a 24-hour period. Thermal images, although not *reliant* on external illumination, still experience dramatic appearance changes throughout the day as the relative temperatures of objects change [13]. We propose addressing this challenge in a similar manner to existing strategies for visible-spectrum image variation, which involves capturing these changes in the training data. Gridseth and Barfoot [22] learned day-night visible-spectrum features using temporally separated image pairs generated from repeated traverses of an unmanned ground vehicle (UGV) over a 30-hour period. Extending this idea to a cross-spectral framework presents a pathway to tackling intraspectrum variation.

Finally, two large challenges arise when learning point features, both of which are ex-

acerbated in the cross-spectral case: obtaining ground truth correspondences and leveraging these correspondences effectively during training. In this work, we have focused on tackling the latter problem via task-oriented training. However, a limitation of our method, and of other works in cross-spectral learned features, is the reliance on a planar scene approximation to obtain dense pixel-to-pixel correspondences. As mentioned in Section 5.1, the original thermal and visible-spectrum training images were captured with slight asynchronicity and aligned offline with a 2D transformation. This procedure relies on the assumption of a planar scene. Otherwise, when a scenes's depth variation is non-negligible, additional information is required to triangulate 3D correspondences from images with offset viewpoints. This parallax is unavoidable for cross-spectral image pairs without highly specialized equipment such as a beam splitter. Further, most visible-spectrum methods for obtaining ground-truth 3D correspondences are not applicable to cross-spectral data. For example, some works such as SuperPoint [19] use simulated data, but this approach would require extensive modeling to extend to the thermal domain. Another popular method, dense 3D reconstruction [20, 21], relies on the existence of robust handcrafted features and photometric consistency, both of which are not available for cross-spectral image pairs.

Instead of identifying individual 3D correspondences, we could extend our method to use the ground-truth pose difference between image viewpoints for supervision, similar to [22]. The matching loss presented in Section 4.3.1 measures the agreement between estimated correspondences and the ground-truth homography model. Extending this from a planar to a depth-varying scene, the agreement between individual correspondences and the fundamental matrix (derived from the pose difference and camera calibration parameters) could be quantified with Sampson distance. Sampson distance is a first-order approximation of reprojection error, similar to the distance between an estimated 2D point and its "correct" epipolar line [76]. This approach could simplify the identification of correspondences and enable training on non-planar scenes. Overall, with these suggested improvements, task-oriented training could potentially alleviate the challenges of identifying and leveraging correspondences in cross-spectral data and extend the limits of camera-based autonomy for UAVs.

Bibliography

[1] A. Couturier and M. A. Akhloufi, "A review on absolute visual localization for UAV," *Robotics and Autonomous Systems*, vol. 135, Jan. 2021.

[2] T. Mouats, N. Aouf, L. Chermak, and M. A. Richardson, "Thermal Stereo Odometry for UAVs," *IEEE Sensors Journal*, vol. 15, no. 11, pp. 6335–6347, Nov. 2015.

[3] P. V. K. Borges and S. Vidas, "Practical Infrared Visual Odometry," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 8, pp. 2205–2213, Aug. 2016.

[4] J. Delaune, R. Hewitt, L. Lytle, C. Sorice, R. Thakker, and L. Matthies, "Thermal-Inertial Odometry for Autonomous Flight Throughout the Night," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Macau, China, Nov. 2019, pp. 1122–1128.

[5] S. Zhao, P. Wang, H. Zhang, Z. Fang, and S. Scherer, "TP-TIO: A Robust Thermal-Inertial Odometry with Deep ThermalPoint," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Las Vegas, NV, USA, Oct. 2020, pp. 4505–4512.

[6] S. Khattak, C. Papachristos, and K. Alexis, "Keyframe-based thermal–inertial odometry," *Journal of Field Robotics*, vol. 37, no. 4, pp. 552–579, Jun. 2020.

[7] V. Polizzi, R. Hewitt, J. Hidalgo-Carrió, J. Delaune, and D. Scaramuzza, "Data-Efficient Collaborative Decentralized Thermal-Inertial Odometry," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 10 681–10 688, Oct. 2022.

[8] S. Vidas and S. Sridharan, "Hand-held monocular SLAM in thermal-infrared," in *International Conference on Control Automation Robotics & Vision (ICARCV)*, 2012, pp. 859–864.

[9] M. R. U. Saputra, C. X. Lu, P. P. B. de Gusmao, B. Wang, A. Markham, and N. Trigoni, "Graph-Based Thermal–Inertial SLAM With Probabilistic Neural Networks," *IEEE Transactions on Robotics*, vol. 38, no. 3, pp. 1875–1893, 2022.

[10] J. Jiang, X. Chen, W. Dai, Z. Gao, and Y. Zhang, "Thermal-Inertial SLAM for the Environments With Challenging Illumination," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 8767–8774, 2022.

[11] Y. Wu et al. "Improving autonomous detection in dynamic environments with robust monocular thermal SLAM system," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 203, pp. 265–284, 2023.

[12] B. R. van Manen, V. Sluiter, and A. Y. Mersha, "FirebotSLAM: Thermal SLAM to Increase Situational Awareness in Smoke-Filled Environments," *Sensors*, vol. 23, no. 17, 2023.

[13] C. Keil, A. Gupta, P. Kaveti, and H. Singh, *Towards Long Term SLAM on Thermal Imagery*, Mar. 2024. [Online]. Available: http://arxiv.org/abs/2403.19885.

[14] *Google Earth*. [Online]. Available: https://earth.google.com/.

[15] D. Han, Y. Hwang, N. Kim, and Y. Choi, "Multispectral Domain Invariant Image for Retrieval-based Place Recognition," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2020, pp. 9271–9277.

[16] J. Xiao, D. Tortei, E. Roura, and G. Loianno, "Long-Range UAV Thermal Geo-Localization with Satellite Imagery," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct. 2023, pp. 5820–5827.

[17] J. Xiao, N. Zhang, D. Tortei, and G. Loianno, *STHN: Deep Homography Estimation for UAV Thermal Geo-localization with Satellite Imagery*, May 2024. [Online]. Available: http://arxiv.org/abs/2405.20470.

[18] C. F. G. Nunes and F. L. C. Pádua, "A Local Feature Descriptor Based on Log-Gabor Filters for Keypoint Matching in Multispectral Images," *IEEE Geoscience and Remote Sensing Letters*, vol. 14, no. 10, pp. 1850–1854, Oct. 2017.

[19] D. DeTone, T. Malisiewicz, and A. Rabinovich, "SuperPoint: Self-Supervised Interest Point Detection and Description," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, Jun. 2018, pp. 337–349.

[20] M. Dusmanu et al. "D2-Net: A Trainable CNN for Joint Description and Detection of Local Features," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2019, pp. 8084–8093.

[21] J. Revaud, C. De Souza, M. Humenberger, and P. Weinzaepfel, "R2D2: Reliable and Repeatable Detector and Descriptor," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

[22] M. Gridseth and T. D. Barfoot, "Keeping an Eye on Things: Deep Learned Features for Long-Term Visual Localization," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 1016–1023, Apr. 2022.

[23] R. Gade and T. B. Moeslund, "Thermal cameras and applications: A survey," *Machine Vision and Applications*, vol. 25, pp. 245–262, 2014.

[24] N. J. W. Morris, S. Avidan, W. Matusik, and H. Pfister, "Statistics of Infrared Images," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2007.

[25] T. Mouats, N. Aouf, A. D. Sappa, C. Aguilera, and R. Toledo, "Multispectral Stereo Odometry," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 3, pp. 1210–1224, Jun. 2015.

[26] O. Riou, S. Berrebi, and P. Bremond, "Nonuniformity correction and thermal drift compensation of thermal infrared camera," in *Thermosense*, Apr. 2004, pp. 294–302.

[27] R. Szeliski, "Deep Learning," in *Computer Vision: Algorithms and Applications*, R. Szeliski, Ed., Cham, 2022, pp. 187–271.

[28] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2012.

[29] A. Vaswani et al. "Attention is All you Need," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

[30] A. Dosovitskiy et al. *An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale*, Jun. 2021. [Online]. Available: http://arxiv.org/abs/2010.11929.

[31] I. J. Goodfellow et al. "Generative adversarial nets," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2014.

[32] R. Szeliski, "Image Formation," in *Computer Vision: Algorithms and Applications*, R. Szeliski, Ed., Cham, 2022, pp. 27–83.

[33] F. Achermann et al. "MultiPoint: Cross-spectral registration of thermal and optical aerial imagery," in *Conference on Robot Learning (CoRL)*, 2020.

[34] D. G. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," *International Journal of Computer Vision*, vol. 60, pp. 91–110, Nov. 2004.

[35] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: An efficient alternative to SIFT or SURF," in *International Conference on Computer Vision (ICCV)*, Nov. 2011, pp. 2564–2571.

[36] E. Rosten, R. Porter, and T. Drummond, "Faster and Better: A Machine Learning Approach to Corner Detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 1, pp. 105–119, Jan. 2010.

[37] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, "BRIEF: Binary Robust Independent Elementary Features," in *European Conference on Computer Vision (ECCV)*, 2010, pp. 778–792.

[38] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, "Speeded-Up Robust Features (SURF)," *Computer Vision and Image Understanding*, vol. 110, no. 3, pp. 346–359, Jun. 2008.

[39] S. Leutenegger, M. Chli, and R. Y. Siegwart, "BRISK: Binary Robust Invariant Scalable Keypoints," in *International Conference on Computer Vision (ICCV)*, Nov. 2011, pp. 2548–2555.

[40] A. Alahi, R. Ortiz, and P. Vandergheynst, "FREAK: Fast Retina Keypoint," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2012, pp. 510–517.

[41] P. F. Alcantarilla, A. Bartoli, and A. J. Davison, "KAZE Features," in *European Conference on Computer Vision (ECCV)*, 2012, pp. 214–227.

[42] K. M. Yi, E. Trulls, V. Lepetit, and P. Fua, "LIFT: Learned Invariant Feature Transform," in *European Conference on Computer Vision (ECCV)*, 2016, pp. 467–483.

[43] Y. Ono, E. Trulls, P. Fua, and K. M. Yi, "LF-Net: Learning Local Features from Images," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

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

[45] R. Hartley and A. Zisserman, "Estimation – 2D Projective Transformations," in *Multiple View Geometry in Computer Vision*, 2004, pp. 87–131.

[46] R. Szeliski, "Image Alignment and Stitching," in *Computer Vision: Algorithms and Applications*, R. Szeliski, Ed., Cham, 2022, pp. 401–441.

[47] X. Jiang, J. Ma, G. Xiao, Z. Shao, and X. Guo, "A review of multimodal image matching: Methods and applications," *Information Fusion*, vol. 73, pp. 22–71, 2021.

[48] F. Maes, A. Collignon, D. Vandermeulen, G. Marchal, and P. Suetens, "Multimodality image registration by maximization of mutual information," *IEEE Transactions on Medical Imaging*, vol. 16, no. 2, pp. 187–198, Apr. 1997.

[49] Y. Ye, J. Shan, L. Bruzzone, and L. Shen, "Robust Registration of Multimodal Remote Sensing Images Based on Structural Similarity," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 55, no. 5, pp. 2941–2958, 2017.

[50] Y. Ye, L. Bruzzone, J. Shan, F. Bovolo, and Q. Zhu, "Fast and Robust Matching for Multimodal Remote Sensing Image Registration," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 57, no. 11, pp. 9059–9070, 2019.

[51] B. Zhu, L. Zhou, S. Pu, J. Fan, and Y. Ye, "Advances and Challenges in Multimodal Remote Sensing Image Registration," *IEEE Journal on Miniaturization for Air and Space Systems*, vol. 4, no. 2, pp. 165–174, 2023.

[52] J. Cronje and J. De Villiers, "A comparison of image features for registering LWIR and visual images," in *Annual Symposium of the Pattern Recognition Association of South Africa (PRASA)*, Pretoria, South Africa, Nov. 2012, pp. 1–8.

[53] C. Aguilera, F. Barrera, F. Lumbreras, A. D. Sappa, and R. Toledo, "Multispectral Image Feature Points," *Sensors*, vol. 12, no. 9, pp. 12 661–12 672, Sep. 2012.

[54] J. Canny, "A Computational Approach to Edge Detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 8, no. 6, pp. 679–698, Nov. 1986.

[55] C. A. Aguilera, A. D. Sappa, and R. Toledo, "LGHD: A feature descriptor for matching across non-linear intensity variations," in *IEEE International Conference on Image Processing (ICIP)*, Sep. 2015, pp. 178–181.

[56] J. Li, Q. Hu, and M. Ai, "RIFT: Multi-Modal Image Matching Based on Radiation-Variation Insensitive Feature Transform," *IEEE Transactions on Image Processing*, vol. 29, pp. 3296–3310, 2020.

[57] S. Özer and A. P. Ndigande, "VisIRNet: Deep Image Alignment for UAV-Taken Visible and Infrared Image Pairs," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 62, 2024.

[58] T. Pouplin, H. Perreault, B. Debaque, M.-A. Drouin, N. Duclos-Hindie, and S. Roy, "Multimodal Deep Homography Estimation Using a Domain Adaptation Generative Adversarial Network," in *IEEE International Conference on Big Data (Big Data)*, Osaka, Japan, Dec. 2022, pp. 3635–3641.

[59] J. Shin, J. Kim, S. Kwon, N. Kim, S. Hwang, and Y. Choi, "UMHE: Unsupervised Multispectral Homography Estimation," *IEEE Sensors Journal*, vol. 24, no. 10, pp. 17 259–17 268, 2024.

[60] B. Debaque et al. "Thermal and Visible Image Registration Using Deep Homography," in *International Conference on Information Fusion (FUSION)*, Jul. 2022.

[61] 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, 2023.

[62] X. Wang et al. "Infrared and Visible Image Homography Estimation Based on Feature Correlation Transformers for Enhanced 6G Space–Air–Ground Integrated Network Perception," *Remote Sensing*, vol. 15, no. 14, 2023.

[63] T. Sattler, Q. Zhou, M. Pollefeys, and L. Leal-Taixé, "Understanding the Limitations of CNN-Based Absolute Camera Pose Regression," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 3297–3307.

[64] C. A. Aguilera, A. D. Sappa, C. Aguilera, and R. Toledo, "Cross-Spectral Local Descriptors via Quadruplet Network," *Sensors*, vol. 17, no. 4, p. 873, Apr. 2017.

[65] S. Jeong, S. Kim, K. Park, and K. Sohn, "Learning to Find Unpaired Cross-Spectral Correspondences," *IEEE Transactions on Image Processing*, vol. 28, no. 11, pp. 5394–5406, Nov. 2019.

[66] M. Elsaeidy, M. E. Erkol, B. K. Güntürk, and H. F. Ateş, "Infrared-to-Optical Image Translation for Keypoint-Based Image Registration," in *Signal Processing and Communications Applications Conference (SIU)*, May 2022.

[67] M. Elsaeidy, İ. C. Yağmur, H. F. Ateş, and B. K. Güntürk, "Swin Transformer based Siamese Network for Thermal and Optical Image Registration," in *Signal Processing and Communications Applications Conference (SIU)*, Jul. 2023.

[68] Y. Deng and J. Ma, "ReDFeat: Recoupling Detection and Description for Multimodal Feature Learning," *IEEE Transactions on Image Processing*, vol. 32, pp. 591–602, 2022.

[69] K. Simonyan and A. Zisserman, *Very Deep Convolutional Networks for Large-Scale Image Recognition*, Apr. 2015. [Online]. Available: http://arxiv.org/abs/1409.1556.

[70] C. Boittiaux, R. Marxer, C. Dune, A. Arnaubec, and V. Hugel, "Homography-Based Loss Function for Camera Pose Regression," *IEEE Robotics and Automation Letters*, vol. 7, no. 3, pp. 6242–6249, Jul. 2022.

[71] J. E. Dennis Jr. and R. E. Welsch, "Techniques for nonlinear least squares and robust regression," *Communications in Statistics - Simulation and Computation*, vol. 7, no. 4, pp. 345–359, Jan. 1978.

[72] J. T. Barron, "A General and Adaptive Robust Loss Function," 2019, pp. 4331–4339.

[73] A. Paszke et al. "PyTorch: An Imperative Style, High-Performance Deep Learning Library," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

[74] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," in *International Conference for Learning Representations*, 2015.

[75] P.-E. Sarlin, D. DeTone, T. Malisiewicz, and A. Rabinovich, "SuperGlue: Learning Feature Matching With Graph Neural Networks," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Seattle, WA, USA, Jun. 2020, pp. 4937–4946.

[76] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed. Cambridge, 2004.