



BRNO UNIVERSITY OF TECHNOLOGY

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

FACULTY OF INFORMATION TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

DEEP LEARNING FOR IMAGE STITCHING

HLUBOKÉ NEURONOVÉ SÍTĚ PRO SEŠÍVÁNÍ OBRÁZKŮ

MASTER'S THESIS

DIPLOMOVÁ PRÁCE

AUTHOR

AUTOR PRÁCE

Bc. PETR ŠILLING

SUPERVISOR

VEDOUCÍ PRÁCE

Ing. MICHAL ŠPANĚL, Ph.D.

BRNO 2023

Master's Thesis Assignment



146325

Institut: Department of Computer Graphics and Multimedia (UPGM)
Student: **Šilling Petr, Bc.**
Programme: Information Technology and Artificial Intelligence
Specialization: Machine Learning
Title: **Deep Learning for Image Stitching**
Category: Image Processing
Academic year: 2022/23

Assignment:

1. Get familiar with deep neural networks and their learning.
2. Get acquainted with the problem of image stitching and existing methods based on deep neural networks.
3. Prepare a dataset for your own experiments.
4. Implement chosen image stitching method to combine several overlapping images into one overall image and experiment with it.
5. Evaluate and compare your results using appropriate metrics. Discuss possible future work.
6. Create a short poster or video presenting your work, its goals and results.

Literature:

- Sarlin et al., "SuperGlue: Learning Feature Matching with Graph Neural Networks", *CVPR 2020* (<https://arxiv.org/abs/1911.11763>).
- Nie et al., "Deep Rectangling for Image Stitching: A Learning Baseline", *CVPR 2022* (<https://arxiv.org/pdf/2203.03831v4.pdf>).

Requirements for the semestral defence:

- Items 1 to 3 of the assignment.

Detailed formal requirements can be found at <https://www.fit.vut.cz/study/theses/>

Supervisor: **Španěl Michal, Ing., Ph.D.**
Consultant: Ing. Oldřich Kodým, Ph.D.
Head of Department: Černocký Jan, prof. Dr. Ing.
Beginning of work: 1.11.2022
Submission deadline: 17.5.2023
Approval date: 3.11.2022

Abstract

Image stitching is an essential technique for reconstructing volumes of biological samples from overlapping tiles of electron microscopy (EM) images. Current volume EM stitching methods generally rely on handcrafted features, such as those produced by SIFT. However, recent developments indicate that convolutional neural networks (CNNs) can improve stitching accuracy by learning discriminative features directly from training images. Taking into account the potential of CNNs, this thesis proposes DEMIS, a novel EM image stitching tool based on LoFTR, an attention-based feature matching network. The thesis also proposes a novel dataset generated by splitting high-resolution EM images into grids of overlapping image tiles. The dataset is used to fine-tune LoFTR and to evaluate the DEMIS tool. Experiments on the synthetic dataset reveal higher feature matching accuracy compared to SIFT. Moreover, experiments on challenging images with small overlap regions and high resolution demonstrate significantly higher stitching robustness than SIFT. Overall, the results suggest that deep learning methods could be beneficial for EM imaging, for example, by allowing the use of smaller tile overlaps.

Abstrakt

Sešívání obrázků je klíčovou technikou pro rekonstrukci objemů biologických vzorků z překrývajících se snímků z elektronové mikroskopie (EM). Současné metody zpracování snímků z EM k sešívání zpravidla využívají ručně definované příznaky, produkované například technikou SIFT. Nedávný vývoj však ukazuje, že konvoluční neuronové sítě dokáží zlepšit přesnost sešívání tím, že se naučí diskriminativní příznaky přímo z trénovacích obrázků. S ohledem na potenciál konvolučních neuronových sítí tato práce navrhuje sešívací nástroj DEMIS, který staví na pozornostní síti LoFTR pro hledání shodných příznaků mezi páry obrázků. Dále práce navrhuje novou datovou sadu generovanou dělením obrázků z EM s vysokým rozlišením na pole překrývajících se dlaždic. Výsledná datová sada je použita pro dotrénování sítě LoFTR a k vyhodnocení nástroje DEMIS. Experimenty na dané datové sadě ukazují, že nástroj je schopen nalézt přesnější shody mezi příznaky než SIFT. Navazující experimenty na obrázcích s vysokým rozlišením a malými překryvy mezi dlaždicemi dále poukazují na výrazně vyšší robustnost oproti metodě SIFT. Dosažené výsledky celkově naznačují, že hluboké učení může vést k prospěšným změnám v oblasti EM, například k umožnění menších překryvů mezi snímanými obrázky.

Keywords

image stitching, volume electron microscopy, deep learning, convolutional neural networks, feature matching, SLAM optimisation, dataset of electron microscopy images, dataset synthesis, Python, DEMIS, LoFTR, SIFT

Klíčová slova

sešívání obrázků, objemová elektronová mikroskopie, hluboké učení, konvoluční neuronové sítě, hledání shod mezi příznaky, SLAM optimalizace, datová sada obrázků z elektronové mikroskopie, syntéza datové sady, Python, DEMIS, LoFTR, SIFT

Reference

ŠILLING, Petr. *Deep Learning for Image Stitching*. Brno, 2023. Master's thesis. Brno University of Technology, Faculty of Information Technology. Supervisor Ing. Michal Španěl, Ph.D.

Rozšířený abstrakt

Sešívání obrázků je metoda pro spojování několika překrývajících se obrázků do jednoho většího obrázku s širším zorným polem (FOV). Jedná se o klíčovou techniku pro rekonstrukci objemů biologických vzorků v elektronové mikroskopii (EM). V EM je totiž běžné, že se snímané vzorky nevejdou do FOV jediného mikroskopu. Často jsou proto vytvářena velká pole překrývajících se obrázků, která je následně nutné sešít dohromady. Současné metody pro zpracování snímků z EM k sešívání nadále využívají čistě tradičních postupů, jako jsou korelace obrázků nebo hledání ručně definovaných příznaků, produkovaných například technikou SIFT. V důsledku vazby na tradiční přístupy se však aktuální metody mohou potýkat s problémy při zpracování obrázků s opakujícími se vzory, špatnou texturou či vysokým rozlišením. Takové obrázky jsou ovšem v EM zcela běžné. Přesné sešívání tedy zpravidla vyžaduje velké překryvy mezi sousedními obrázky, což snižuje efektivní rychlost snímání a zvyšuje objem nasnímaných dat. Nedávný vývoj však ukazuje, že konvoluční neuronové sítě dokáží zlepšit přesnost sešívání tím, že se naučí diskriminativní příznaky přímo z trénovacích obrázků.

S ohledem na potenciál konvolučních neuronových sítí tato práce navrhuje DEMIS, nový nástroj pro sešívání obrázků z EM založený na konvoluční neuronové síti LoFTR. LoFTR je neuronová síť, která využívá konvolučních vrstev k extrakci příznaků z dvojic překrývajících se obrázků. Mezi extrahovanými příznaky pak LoFTR detekuje shody pomocí pozornosti. Na klasických fotografiích přitom shody detekované sítí LoFTR zpravidla dosahují vyšší kvality než shody generované tradičními přístupy, hlavně pak v případě obrázků s opakujícími se vzory či špatnou texturou. Využitím sítě LoFTR se tedy nástroj DEMIS snaží zvýšit přesnost detekovaných shod a celkovou robustnost sešívání v porovnání s aktuálními metodami používanými v EM.

Nástroj DEMIS na vstupu očekává pole překrývajících se snímků z EM a postupuje následovně. Nejprve znormalizuje jas a kontrast všech snímků, aby zjednodušil následné zpracování a co nejlépe zamaskoval přechody mezi dílčími snímky ve finálním sešitém obrázku. Následně nástroj mezi každou dvojicí překrývajících se obrázků detekuje shodující se příznaky pomocí sítě LoFTR. Podle nalezených shod jsou poté odhadnuty transformace, které popisují vztahy mezi souřadnými systémy příslušných obrázků. Dále je sestaven graf pro optimalizační metodu SLAM, který reprezentuje snímky sešívaného pole (vrcholy) a odhadnuté transformace mezi nimi (hrany). Počáteční pozice vrcholů jsou nastaveny s ohledem na očekávanou strukturu sešívaného pole. Transformace v grafu jsou následně globálně optimalizovány. Nakonec je celé vstupní pole sešito postupným aplikováním optimalizovaných transformací na odpovídající dílčí snímky.

Dále práce navrhuje novou datovou sadu DEMIS, vytvořenou z ručně vybraných snímků z EM s vysokou kvalitou a vysokým rozlišením. Sada je generována dělením vybraných obrázků do polí překrývajících se dlaždic o velikosti 1024×1024 pixelů. V rámci generování jsou rovněž na každou dlaždici aplikovány následující transformace: náhodné změny jasu a kontrastu, náhodné posunutí vůči předcházející dlaždici, náhodná rotace kolem středu dlaždice, přidání Gaussova šumu. Výsledná syntetická datová sada obsahuje celkem 10 883 jednotlivých dlaždic, přičemž 8339 z nich bylo generováno z obrázků získaných z veřejně dostupných zdrojů.

Nástroj DEMIS byl experimentálně vyhodnocen na 1306 evaluačních obrázcích z datové sady DEMIS a na dvou datových sadách reálných obrázků z EM s vysokým rozlišením a malými překryvy poskytnutých firmou TESCAN 3DIM. V experimentech byla porovnávána tři různá řešení pro sešívání snímků z EM: (1) varianta nástroje DEMIS, která místo sítě LoFTR využívá tradiční přístupy založené na metodě SIFT, (2) nástroj DEMIS,

jenž využívá síť LoFTR s oficiálními předtrénovanými vahami, a (3) nástroj DEMIS, který používá síť LoFTR dotrénovanou na datové sadě DEMIS. Experimenty se sadou DEMIS ukazují, že dotrénovaná varianta nástroje DEMIS generuje přesnější shody mezi příznaky než varianta používající SIFT. Experimenty na datech poskytnutých firmou TESCAN 3DIM pak poukazují na výrazně vyšší robustnost řešení využívajících k detekci shod mezi obrázky síť LoFTR. Dosažené výsledky celkově naznačují, že hluboké učení a neuronové sítě mohou vést k prospěšným změnám v oblasti EM, například k umožnění menších překryvů mezi snímanými obrázky a odpovídajícímu zvýšení rychlosti snímání.

Práce byla vypracována ve spolupráci s firmou TESCAN 3DIM a prezentována na studentské konferenci Excel@FIT 2023, pořádané Fakultou informačních technologií Vysokého učení technického v Brně. Výsledné zdrojové kódy jsou veřejně dostupné online prostřednictvím repozitáře [PSilling/demis](https://github.com/PSilling/demis) na platformě GitHub.

Deep Learning for Image Stitching

Declaration

I hereby declare that this Master's thesis was prepared as an original work by the author under the supervision of Ing. Michal Španěl, Ph.D. The supplementary information was provided by Ing. Oldřich Kodým, Ph.D. I have listed all the literary sources, publications and other sources, which were used during the preparation of this thesis.

.....
Petr Šilling
May 15, 2023

Acknowledgements

I want to thank my supervisor Ing. Michal Španěl, Ph.D., for his professional guidance, especially regarding the theoretical aspects of the thesis. I also thank my consultant, Ing. Oldřich Kodým, Ph.D., and TESCAN 3DIM as a whole for providing several data samples and additional support. Furthermore, I want to thank my family for the support I received during my studies. Finally, I thank the e-INFRA CZ project (ID:90254), supported by the Ministry of Education, Youth and Sports of the Czech Republic, for providing the necessary computational resources to finish my thesis.

Contents

1	Introduction	4
2	Volume Electron Microscopy	6
2.1	Image Stitching in Volume Electron Microscopy	8
3	Traditional Approaches to Image Stitching	11
3.1	Fundamental Steps of Traditional Image Stitching	11
3.2	Current State of Feature-Based Image Stitching	17
3.3	Traditional Stitching of Electron Microscopy Images	19
4	Deep Learning Methods for Image Stitching	20
4.1	Learning-Based Feature Detection and Matching	21
4.2	Methods Using Deep Homography Estimation	25
4.3	End-to-End Deep Image Stitching Networks	26
5	Proposed Synthetic Electron Microscopy Stitching Dataset	28
5.1	Selecting Images for the Synthetic Dataset	28
5.2	Generating the Synthetic Dataset	29
6	Proposed Electron Microscopy Image Stitching Tool	32
6.1	Top-Level Grid Stitching Algorithm	33
6.2	Estimating Transformations Between Pairs of Images	35
6.3	Global Optimisation Based on SLAM	37
6.4	Constructing the Final Composite Image	40
6.5	Fine-Tuning LoFTR on Electron Microscopy Images	41
7	Implementation	43
7.1	Primary Modules of the DEMIS Tool	43
7.2	Secondary Modules of the DEMIS Tool	45
8	Experimental Evaluation	47
8.1	Evaluating Feature and Transformation Processing	48
8.2	Evaluation of Image Stitching Quality	49
8.3	Analysing Stitching Robustness	51
9	Conclusion	53
	Bibliography	54

A Contents of the Attached Medium	62
B Depiction of the Created Poster	63

List of Figures

2.1	Visualisation of the five main volume electron microscopy techniques.	7
2.2	Standard volume electron microscopy workflow.	8
2.3	Examples of challenging images for current EM image stitching methods. . .	10
3.1	Two overlapping EM images being stitched together.	12
3.2	A visual representation of SIFT feature description.	13
3.3	SIFT features detected in a pair of stitched images.	14
3.4	Matches established between SIFT features detected in a pair of stitched images.	15
3.5	Filtering feature matches by RANSAC.	16
3.6	Stitching without any form of seam removal compared to stitching with adaptive pixel weighting.	17
4.1	Comparison of feature matching between SIFT, SuperPoint with SuperGlue, and LoFTR.	21
4.2	The architecture of LoFTR.	22
4.3	Computational graphs of a Transformer encoder layer, dot product attention, and linear attention.	23
4.4	The architecture of HomographyNet.	25
4.5	The architecture of UDIS.	26
4.6	Comparison of stitching using the traditional stitching pipeline and UDIS. .	27
5.1	Sample images from EMPIAR and CIL selected for the DEMIS dataset. . .	29
5.2	Splitting an image into a grid of overlapping image tiles.	30
5.3	The splitting of a single image tile.	31
6.1	The stitching pipeline of the proposed DEMIS tool.	33
6.2	A SLAM graph corresponding to an image grid of size 3×3	38
6.3	Comparison of different methods for processing pixels from overlapping regions of adjacent tiles.	41
7.1	The primary modules of the DEMIS tool.	44
8.1	Comparison of feature matching between a SIFT baseline and the DEMIS tool on a relatively challenging dataset.	50
8.2	Comparison of feature matching between a SIFT baseline and the DEMIS tool on a highly challenging dataset.	51
B.1	Depiction of the created poster.	63

Chapter 1

Introduction

Image stitching is the process of combining multiple overlapping images to create a composite image with a wider field of view (FOV). It is an essential technique for 3D reconstruction in volume electron microscopy (EM), where large arrays of overlapping images are produced to capture samples that do not fit under the FOV of a single microscope. Current volume EM stitching methods are based purely on traditional approaches and handcrafted features, such as those produced by SIFT. Consequently, they may struggle with repetitive patterns, poor texture, and high-resolution images—all of which are common in volume EM. Hence, accurate stitching often requires large image overlaps, which slows down the speed of imaging and increases data storage requirements.

Motivated by the above issues and recent advances in feature detection and matching using convolutional neural networks (CNNs), this thesis proposes DEMIS, a novel EM image stitching tool based on LoFTR, an attention-based feature detection and matching network. Using LoFTR, the DEMIS tool aims to increase the accuracy of detected feature matches and the overall robustness of EM image stitching.

The DEMIS tool works as follows. First, the brightness and contrast of raw image tiles in the stitched image grids are normalised. Second, for each pair of adjacent images, features are detected and matched by LoFTR. Subsequently, a SLAM graph is constructed based on the expected grid structure and transformations estimated from the detected feature matches. The transformations in the graph are then optimised globally. Finally, the grid is stitched by gradually applying the optimised transformations to individual image tiles.

Furthermore, this thesis proposes a novel synthetic dataset created from manually selected high-quality EM images with high resolution. The dataset is generated by splitting the selected images into grids of overlapping image tiles 1024×1024 pixels in size. Additionally, random brightness and contrast adjustments, random rotation and translation, and Gaussian noise are applied to each tile. The synthetic dataset is used to fine-tune LoFTR on EM images and to evaluate the DEMIS tool.

The experiments on the synthetic dataset show that the DEMIS tool generates more accurate feature matches than a comparable method based on SIFT. Moreover, experiments with the DEMIS tool on challenging images with small overlap regions and high resolution demonstrate significantly higher stitching robustness than SIFT. Overall, the results suggest that deep learning methods could be beneficial for EM imaging, for example, by allowing the use of smaller tile overlaps and, consequently, increasing imaging speeds.

This thesis was developed in collaboration with TESCAN 3DIM and presented at Excel@FIT 2023, a student conference held by the Faculty of Information Technology, Brno

University of Technology. The resulting source codes are publicly available through the DEMIS GitHub repository¹.

The rest of this thesis is organised as follows. First, volume electron microscopy and its relationship to image stitching are described in Chapter 2. Second, traditional image stitching methods are introduced in Chapter 3. Then, existing deep learning approaches to image stitching are discussed in Chapter 4. Chapter 5 follows by presenting the novel synthetic dataset of EM images. Next, the DEMIS image stitching tool is proposed in Chapter 6. Furthermore, the implementation of the DEMIS tool is described in Chapter 7. The results of the experimental evaluation are discussed in Chapter 8. Finally, Chapter 9 summarises the thesis and suggests potential future work.

¹DEMIS GitHub repository – <https://github.com/PSilling/demis>.

Chapter 2

Volume Electron Microscopy

Visualisation and a deep understanding of the fine three-dimensional *ultrastructure* of biological samples, ranging from cells and their organelles to organ tissue, have long been desired by many scientists. Due to the extremely small (nanometre) scale of biological samples, achieving this objective became feasible only in the twentieth century with the advent of *electron microscopy* (EM) [59]. However, early EM techniques required extensive manual labour to prepare samples and reconstruct 3D structures after imaging. Consequently, complex volumetric imaging could not have been realised until the development of highly automated computer-assisted processes in the twenty-first century, such as the *stitching* of partial EM images into composite ones, which is the main topic of this thesis. Collectively, these novel methods established *volume electron microscopy* (volume EM), a research field focused on high-resolution 3D tissue reconstruction [29].

Today, there are five main volume EM techniques, each with its own advantages and disadvantages.¹ For example, some destroy analysed samples during image acquisition, while others do not. A brief overview of these methods, based on [29, 59], is provided below. Furthermore, a visualisation of the underlying principles is presented in Figure 2.1.

- Serial section transmission electron microscopy (ssTEM) – sections of examined samples are placed on support grids and imaged with transmission electron microscopy.
- Electron tomography (ET) – reconstructs the volume of comparatively thick sample sections by recording at multiple tilt angles.
- Serial block-face scanning electron microscopy (SBF-SEM) – thin layers of the studied sample are repeatedly cut with a diamond knife and scanned in a highly automated manner.
- Focused ion beam scanning electron microscopy (FIB-SEM) – functions similarly to SBF-SEM, iteratively removing and imaging thin sample layers. However, FIB-SEM removes material using a focused ion beam instead of cutting it with a diamond knife.
- Array tomography (AT) – images ribbons of sample sections positioned on a solid surface with scanning electron microscopy.

Regardless of the chosen procedure, the general volume EM workflow remains the same [29]. First, the target sample has to be prepared for imaging. Usually, this involves

¹As a result of increasing demands for imaging speed, more techniques and enhancements, such as parallel multi-beam scanning, are emerging as well [29]. However, these are not yet fully mature.

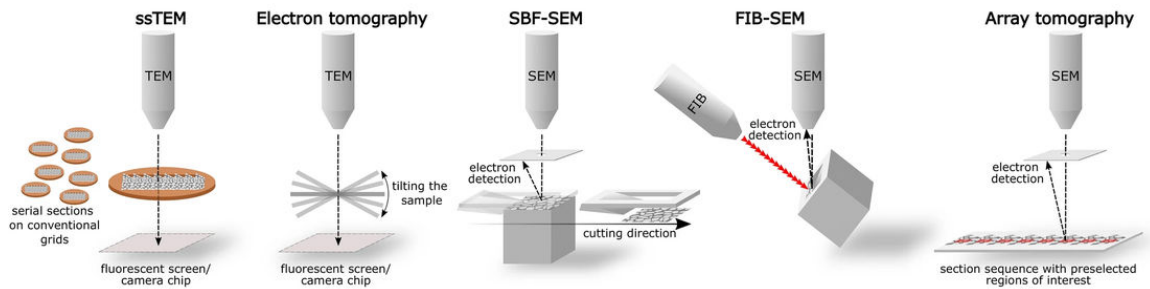


Figure 2.1: Visualisation of the five main volume electron microscopy techniques: serial section transmission electron microscopy (ssTEM), electron tomography, serial block-face scanning electron microscopy (SBF-SEM), focused ion beam scanning electron microscopy (FIB-SEM), and array tomography. The image was retrieved from [59].

fixation in aldehydes, staining with heavy metals (e.g., osmium), dehydration, and embedding in resin. Depending on the microscopy technique, sectioning may also be required. Second, the imaging itself is performed. Since samples often cannot fit in the field of view (FOV) of a single microscope, this step generally produces a grid of slightly overlapping² image tiles. An example set of overlapping image tiles can be seen in Figure 2.2. Moreover, to capture volume information, imaging must be repeated at each position in the depth dimension. Finally, the resulting stack of image grids needs to be combined to form a 3D reconstruction of the studied sample.

To reconstruct the 3D structure, multiple steps are necessary [29]. The steps are illustrated in Figure 2.2. Because sample processing may frequently take several months, there might be considerable differences in brightness and contrast between the retrieved images. Consequently, the first step is typically some form of intensity normalisation. Second, unless the entire region of interest (ROI) of the sample fits in the microscope FOV, image tiles at the same depth position have to be connected together. A common way to *stitch* the image tiles is by identifying point correspondences between adjacent tiles. Subsequently, alignment in the depth dimension is performed using similar methods. At this point, the 3D structure is fully reconstructed. Finally, image segmentation is applied to the recovered volume to allow efficient extraction of biological information from the data. Segmentation is often at least semi-automated (for example, with the help of machine learning). Therefore, it may be followed by manual inspection and correction.

Overall, recent advances in microscopy technology have brought more automation to the field, enabling more sophisticated volumetric imaging. However, while these developments allowed large volumes to be analysed, they also introduced several new challenges to data acquisition and processing [29]. For example, they increased the demand for high-capacity data storage options since EM datasets regularly measure hundreds of terabytes in size. Several such challenges are also related to image stitching, the third step of the volume reconstruction workflow outlined in Figure 2.2 and the main focus of this thesis. These, along with a more detailed description of image stitching itself, are presented in Section 2.1.

²While theoretically redundant under ideal conditions, overlaps are essential to guide further 3D reconstruction. This is because current microscopes cannot yet guarantee perfect image alignment [59].

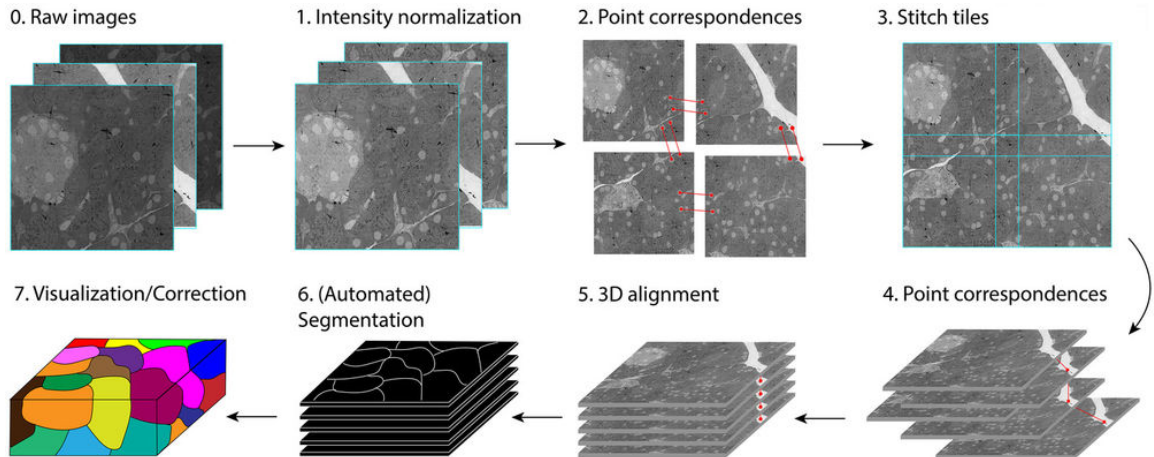


Figure 2.2: Standard volume electron microscopy workflow. First, raw overlapping image tiles are collected by an electron microscope. Second, the tiles are normalised to minimise brightness and contrast differences. Next, tiles at the same position in the depth dimension are stitched into single images, generally using point-to-point correspondences. Similarly, the resulting images are then aligned in the depth dimension to form a valid volume. Finally, data segmentation is performed. Further correction might be necessary as segmentation is often automated. The image was acquired from [29].

2.1 Image Stitching in Volume Electron Microscopy

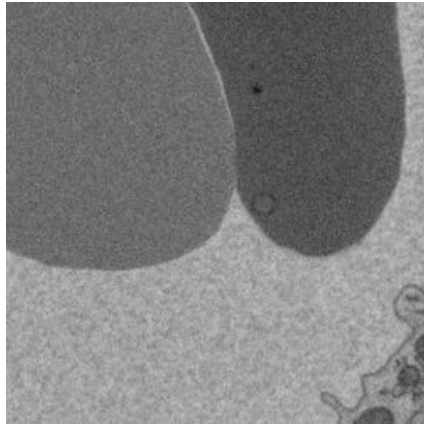
Image stitching is the process of combining multiple overlapping images to generate a new image with a larger field of view (and higher resolution) [6]. Although its most well-known applications are panorama imaging, digital mapping, and satellite imaging, image stitching is also crucial for volume EM, as illustrated in Figure 2.2. This is because the size of the EM image arrays created when imaging large biological samples can become rather extensive. Hence, it would be extremely impractical to combine the produced image tiles manually. Instead, image stitching is required to provide an ideally fully automated way to merge the overlapping tiles together. However, even though image stitching is essential for volume EM and many stitching tools are already available, such as ImageJ [58] and TrackEM2 [10], several challenges remain. Since this thesis aims to overcome these challenges with the help of novel deep learning methods, the remainder of this section briefly describes the most common challenges in EM image stitching. The challenges are listed below. The descriptions are inspired by the information provided by TESCAN 3DIM.

- The first challenge in EM image stitching arises directly from the nature of EM imagery since, in volume EM, it is relatively common for images to have low-quality texture. In more extreme cases, where the imaged samples cover only a tiny fraction of the imaged area, the images may also be filled almost entirely with noise. Since current image stitching methods generally rely on detecting key points of interest in the stitched images [10, 43], both low-quality texture and noise can have a detrimental effect on stitching quality. Examples of EM images with low-quality texture and high amounts of noise are presented in Figures 2.3a and 2.3b, respectively.
- Second, differences in brightness and contrast between individual image tiles (depicted, for example, in Figure 2.3c) can also reduce the accuracy of current stitching

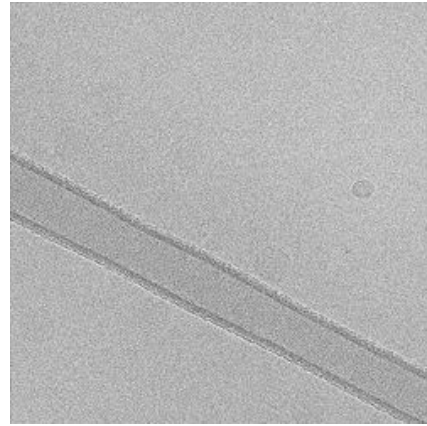
methods. While, as shown in Figure 2.2, this issue is largely mitigated by prior intensity normalisation, it is improbable for brightness and contrast differences to be completely eliminated. Therefore, EM stitching methods must be robust enough to deal with such differences effectively.

- Finally, current image stitching tools require considerable image tile overlaps to ensure high-quality stitching results. Unfortunately, this further increases the generally already large size of volume EM datasets and slows down the speed of image acquisition. Therefore, a compromise between stitching quality and imaging speed is usually necessary. An example of an image pair with overlaps that are too small for current stitching techniques to provide satisfactory results is displayed in Figure 2.3d.

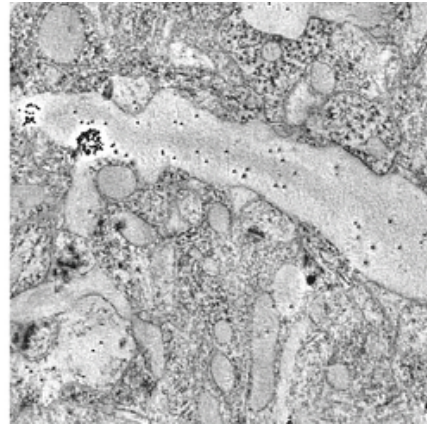
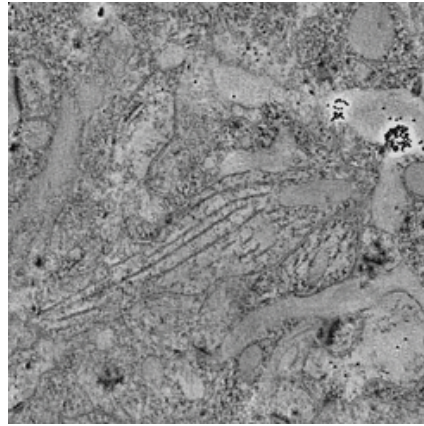
In summary, current EM image stitching methods face challenges caused by low-quality texture, noise, brightness and contrast differences, and small image tile overlaps. More details on image stitching and how to potentially overcome these challenges are presented in Chapters 3 and 4.



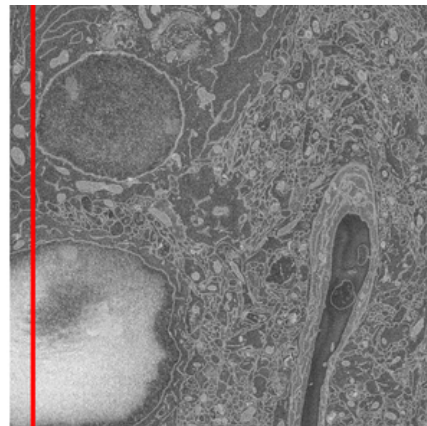
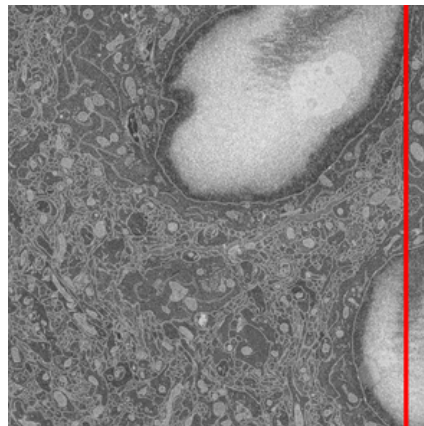
(a) EM image depicting a poorly textured sample. The image was generated from [33].



(b) Sample EM image filled mostly with noise. The image was obtained from [11].



(c) A pair of overlapping EM image tiles with substantial brightness and contrast differences. The tiles were generated from [8].



(d) A pair of EM image tiles with an extremely small 5% overlap. The overlapping regions are separated by red lines. The images were provided by TESCAN 3DIM.

Figure 2.3: Examples of challenging images that current EM image stitching methods may struggle with.

Chapter 3

Traditional Approaches to Image Stitching

As described in Chapter 2, image stitching is a crucial technique for processing large arrays of electron microscopy (EM) images. However, regardless of the application, traditional image stitching follows the same general steps for processing individual pairs of overlapping images: feature detection and description, feature matching, estimation of homography (or of other less generic transformations), and image alignment and stitching. This chapter explains these steps in detail and defines the current state-of-the-art of traditional image stitching. In particular, the image stitching steps, along with various enhancement techniques, are described in Section 3.1. Section 3.2 then reviews the state-of-the-art of traditional feature-based image stitching. Finally, Section 3.3 analyses traditional image stitching methods designed for EM imagery.

3.1 Fundamental Steps of Traditional Image Stitching

Image stitching can be defined as the process of constructing a new image I_{12} by *registering*¹ a *source* image I_1 and a *template* image I_2 into a shared coordinate system, aligning their overlapping regions [63]. Doing so is only possible when a one-to-one mapping exists between the coordinates of I_1 and the coordinates of I_2 . In particular, this happens when the capturing camera was (a) simply rotating around its optical centre, or (b) viewing a planar surface from different positions [6]. Conventional image stitching methods try to find the parameters of this mapping and use the solution to register the stitched images.

Since image stitching algorithms are among the oldest in computer vision [45], several models of the relationship between stitched images exist nowadays. These include, but are not limited to, direct *pixel-to-pixel* estimation methods based on *gradient descent*², approaches based on *feature-matching*, complete 3D motion modelling, and techniques utilising cylindrical or spherical coordinates [64]. Because this thesis focuses primarily on image stitching aided by deep learning, this section does not try to provide an exhaustive review of these diverse methods. Instead, the remainder of this section only briefly explains the widely adopted (even in the context of deep learning) feature-based image stitching,

¹**Image registration** – a technique for combining different images of the same area from different times, views and camera sensors [68].

²**Gradient descent** – a method for minimising an objective function by moving in the direction opposite of its gradient [5].

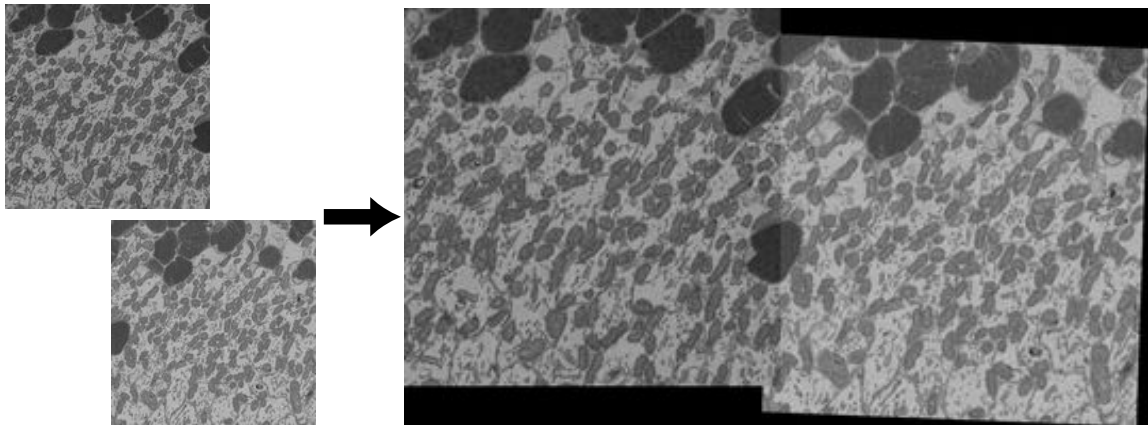


Figure 3.1: Two overlapping EM images being stitched together. Since no post-processing was applied, the resulting image has a non-rectangular boundary and visible brightness and contrast differences, which form a seam between the two images. Both of these attributes are characteristic features of stitched images. The image tiles were generated from [30].

which, generally, consists of four steps: feature detection and description, feature matching, homography estimation, and image alignment and stitching [64]. Optionally, several enhancement techniques, such as seam removal and image blending, might be applied as well. A more complete overview of image stitching algorithms can be found, e.g., in [64] or [68]. An example of the stitching of two EM images without any form of post-processing or image enhancement is presented in Figure 3.1.

Feature Detection and Description

Local *features* are localised geometric entities that represent the containing image [1]. In particular, individual points, corners, lines, and edges are all typical features. Since features directly describe image content, they can be used to establish correspondences between images, making them incredibly valuable in the context of image stitching. As a result, the first step of most current image stitching algorithms is *feature detection*, which tries to identify distinctive features of stitched images [64]. However, detecting features might not be sufficiently robust by itself due to view variances between the stitched images. Therefore, the image regions around the detected features need to be converted into *feature descriptors* – a more stable format designed for *matching* against descriptors from different images (i.e., for finding correspondences between them) [64].

One of the most well-established local feature detection and description methods is the scale-invariant feature transform (SIFT) [41]. SIFT begins by identifying candidate features of the analysed image (also referred to as *keypoints*). It convolves the image with Gaussian functions with gradually increasing scales and down-sampling rates, creating a pyramid of blurred images. Adjacent Gaussian images are then subtracted to form a *difference-of-Gaussian* (DoG) pyramid. Local extrema in the DoG pyramid (scale-space) are selected as potential features. Next, location and scale are assigned to each candidate feature according to its neighbourhood. Furthermore, extrema along edges or with low contrast are rejected to ensure stability against noise. Subsequently, orientations are calculated for each feature by analysing Gaussian-weighted gradients in their vicinity. Finally, the features are described

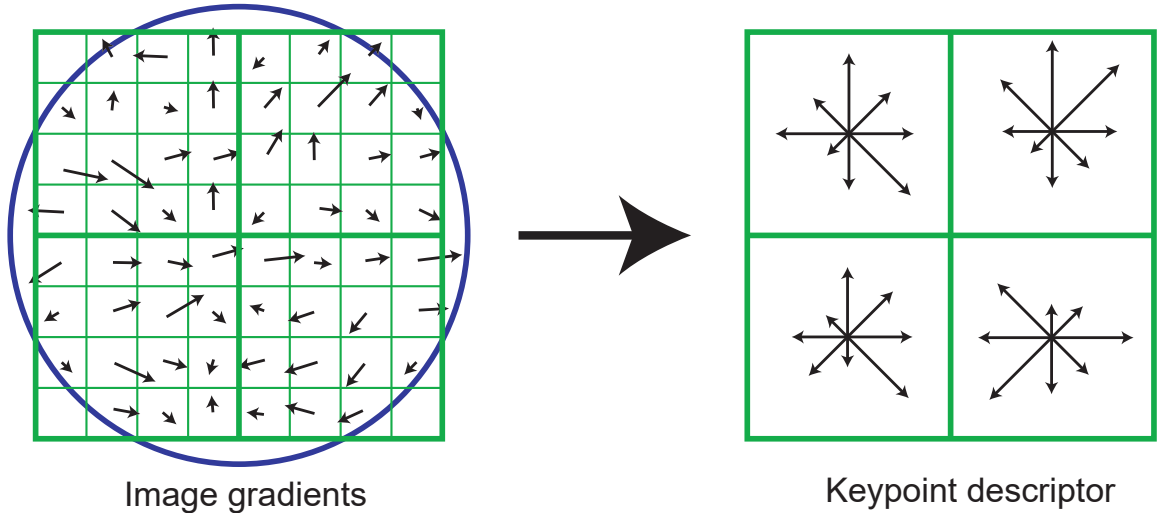


Figure 3.2: A visual representation of SIFT feature description. First, gradient magnitudes and orientations are computed around the feature location (left) and weighted by a Gaussian window (blue circle). Afterwards, the gradient samples are accumulated into eight-bin orientation histograms summarising the contents of 4×4 subregions (right). The image was obtained from [41].

by an eight-bin orientation histogram which accumulates gradients in 4×4 subregions around their locations, as shown in Figure 3.2.

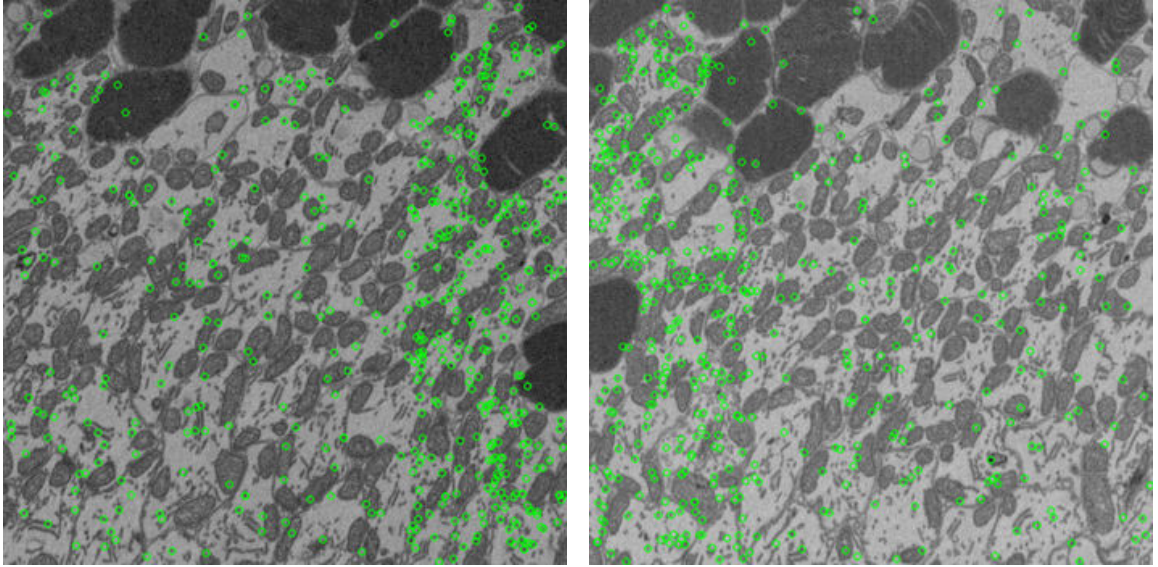
Overall, the above process makes SIFT features quite robust as they end up being translation, scale, and rotation invariant [41]. However, detecting SIFT features is computationally expensive compared to other methods, which are summarised in Section 3.2 [68]. Moreover, as a traditional feature detection method, SIFT may fail to detect a sufficient number of features when used under the challenging conditions described in Section 2.1. The potential ways to overcome this limitation with the help of deep learning are presented in Chapter 4. SIFT features of the two EM images from Figure 3.1 are highlighted in Figure 3.3.

Feature Matching

After identifying features of the stitched images, it is necessary to *match* them, i.e., determine which features of I_1 are also located in I_2 and where. The most straightforward strategy would be to compare the feature descriptors of I_1 against all feature descriptors of I_2 using a suitable distance metric³ and obtain the nearest neighbours in feature space. However, while this method is easy to implement, it scales quadratically with the expected number of features, making it too computationally expensive for some applications [63]. Therefore, more efficient matching algorithms utilising indexing structures, such as multi-dimensional search trees and hash tables, can be devised to decrease the number of required feature comparisons [64].

Moreover, since many feature matches are expected to be *false positive*, i.e., incorrectly established (generally due to background noise), it might be essential to discard as many invalid matches as possible [41]. A common heuristic for false positive detection is the *nearest neighbour distance ratio* (NNDR) [44], which compares the distance between the

³For SIFT features, Euclidean distance measurement should be used [41].



(a) Left image features

(b) Right image features

Figure 3.3: SIFT features [41] (represented by green circles) detected in the two stitched images from Figure 3.1. To preserve visual clarity, only 500 features are displayed in each image (features in the overlapping region were preferred). Most features were discovered near corners and edges.

nearest and second nearest neighbours of each feature. Formally, the NNDR can be defined as

$$\text{NNDR} = \frac{\|D_A - D_B\|}{\|D_A - D_C\|}, \quad (3.1)$$

where D_B is the first and D_C the second nearest neighbour of descriptor D_A . If the NNDR for D_A is below a specified threshold, the corresponding feature match can be discarded. This process is commonly known as *ratio testing*. With adequate threshold levels (for SIFT generally around 0.8), the majority of false positives are removed, while valid matches remain mostly unaffected [41].

The result can be seen in Figure 3.4. While several invalid feature matches are still present even after applying the ratio test (some are not even in the overlapping regions), the number of false positives decreased substantially. In fact, without ratio testing, the number of incorrect matches would be several times higher than currently visible.

Homography Estimation

After constructing a set of feature matches, the mathematical relationship between the stitched images needs to be modelled. Assuming the capturing cameras are *rectilinear*⁴, the coordinates of I_1 and I_2 can be related by a *perspective transform*, which can be described by a *homography matrix* $\mathbf{H} \in \mathbb{R}^{3 \times 3}$ (often shortened as *homography*) [6, 63].

Let us consider a pair of matching points (e.g., features) in homogeneous coordinates $\hat{\mathbf{p}}_1 = [x_1 \ y_1 \ 1]^T$, $\hat{\mathbf{p}}_2 = [x_2 \ y_2 \ 1]^T$, from I_1 and I_2 , respectively. The homographic relation-

⁴**Rectilinear camera** – a camera with a lens that renders straight lines as straight without any apparent curvature and distortion. [52].

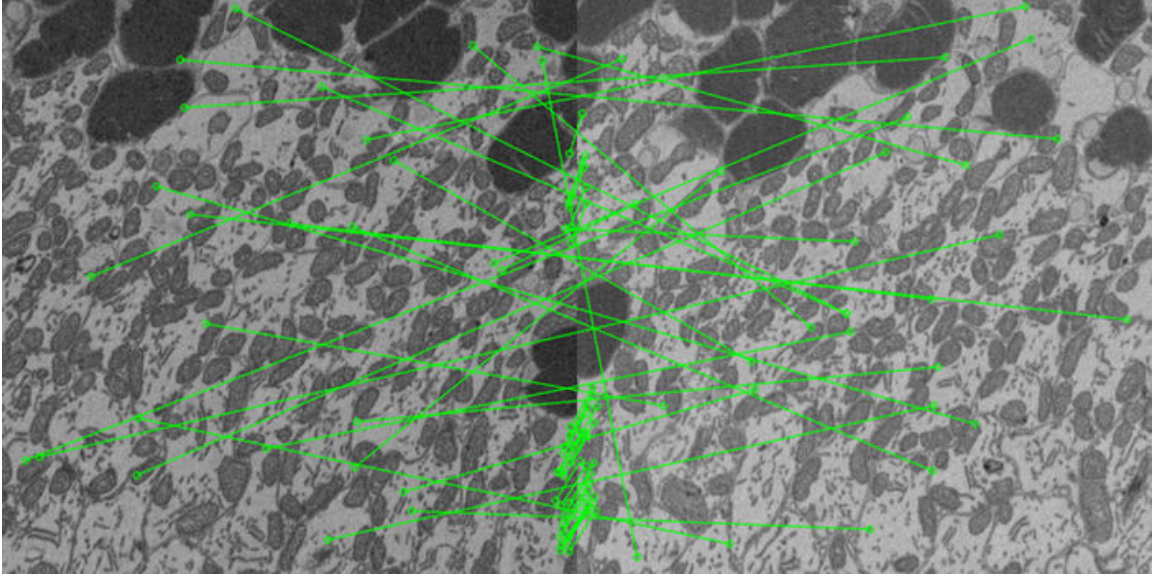


Figure 3.4: Matches established between SIFT [41] features of the stitched images from Figure 3.1. Matching features (green circles) are connected by green lines. The nearest neighbour distance ratio [44] heuristic was applied with the threshold set to 0.75 to eliminate as many incorrect matches as possible. Despite that, several false positives remain.

ship between $\hat{\mathbf{p}}_1$ and $\hat{\mathbf{p}}_2$ can be represented as

$$\hat{\mathbf{p}}_1 \sim \mathbf{H}\hat{\mathbf{p}}_2 = \begin{bmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & 1 \end{bmatrix} \hat{\mathbf{p}}_2, \quad (3.2)$$

where \sim denotes equality up to scale [37, 68]. The inhomogeneous coordinates of $\hat{\mathbf{p}}_1$, i.e., x_1 and y_1 , can then be obtained using the following equations [68]:

$$x_1 = \frac{h_1x_2 + h_2y_2 + h_3}{h_7x_2 + h_8y_2 + 1}, \quad (3.3a)$$

$$y_1 = \frac{h_4x_2 + h_5y_2 + h_6}{h_7x_2 + h_8y_2 + 1}. \quad (3.3b)$$

Provided at least 4 match pairs are known (the homography matrix has 8 degrees of freedom), the parameters of \mathbf{H} can be estimated by the *least squares method*, that is, by minimising the sum of squared residual errors, or by its more robust iterative variants [63]. However, while this approach is the most straightforward, issues might arise with excessive amounts of incorrect matches [64]. Because false positives are generally quite common during image stitching (as demonstrated in Figure 3.4), it is often better to first further reduce the number of mismatches.

The most widely adopted technique for false positive elimination after ratio testing is the random sample consensus (RANSAC) [19]. RANSAC is an iterative non-deterministic algorithm that starts by selecting a random subset of n feature matches (for homography estimation, $n \geq 4$), which is used to instantiate the initial estimate of \mathbf{H} . Using this estimate, RANSAC determines the expected locations of the features of I_2 in the coordinate space of I_1 (with the help of Equation 3.2) and calculates the number of feature correspondences within a specified error margin (so-called *inliers* [64]). Subsequently, the random

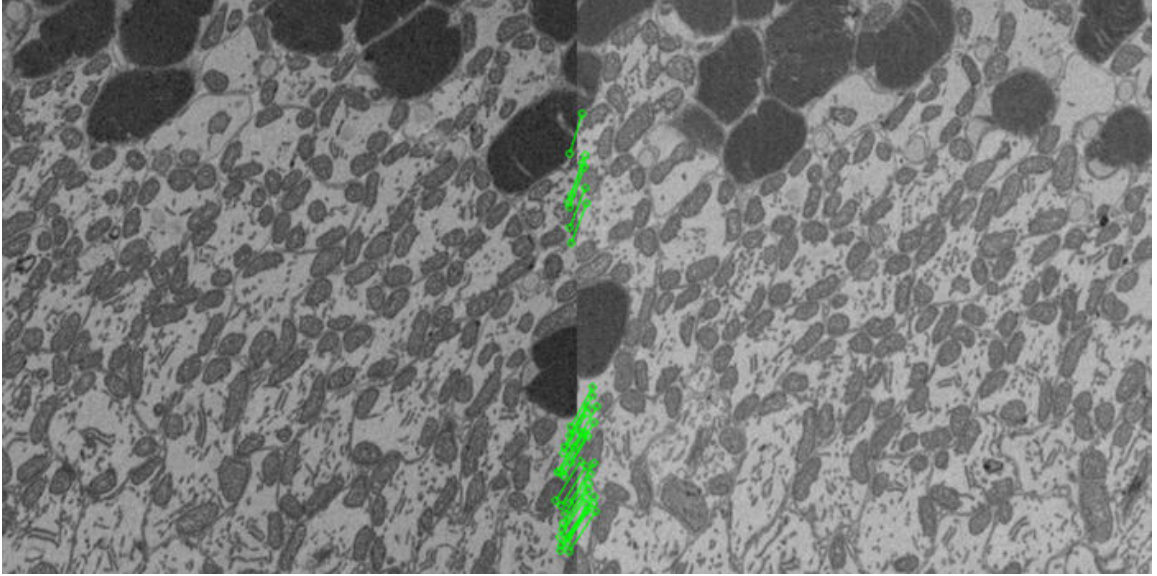


Figure 3.5: The feature matches from Figure 3.4 filtered by RANSAC [19]. All false positive matches from Figure 3.4 are removed, while valid (inlier) matches remain present.

selection process is repeated N times, and the result is chosen as the subset of matches with the largest number of inliers. The result is then used to generate the final estimate of \mathbf{H} (e.g., with the help of least squares) [64].

The effectiveness of RANSAC is illustrated in Figure 3.5, which shows the inliers detected by RANSAC when used on the matches from Figure 3.4. As opposed to Figure 3.4, no false positive matches are visible. Consequently, the estimation of the parameters of \mathbf{H} should remain stable.

Image Alignment and Stitching

After computing the homography matrix \mathbf{H} , the coordinate systems of images I_1 and I_2 can be aligned with each other by applying (3.2) on all points in I_2 . The stitching itself can then be performed simply by positioning the registered images on a new image with a wider view⁵, and by giving precedence to either I_1 or I_2 for shared pixels from the overlapping area. Assuming the images from Figure 3.5 and render priority given to the left image (i.e., image I_1), the stitching result would be identical to the one presented in Figure 3.1. However, as demonstrated in Figure 3.1, this approach may lead to visible seams between the stitched images due to illumination differences (even if the stitching was otherwise correct). Considering that humans are highly sensitive to discernible image seams, it might be preferable to also apply *seam removal methods* [68].

The simplest seam removal techniques are based on *pixel weighting*, which calculates the pixel values $I_{12}(x, y)$ from the overlapping area of I_{12} as

$$I_{12}(x, y) = \alpha\Omega_1(x, y) + (1 - \alpha)\Omega_2(x, y), \quad (3.4)$$

where $\alpha \in \langle 0, 1 \rangle$ is a weight coefficient, and $\Omega_1(x, y)$, $\Omega_2(x, y)$ are the pixel values of registered images I_1 , I_2 at coordinates $[x, y]^T$ in the overlapping region Ω , respectively [68].

⁵In the worst case scenario, the dimensions of the final image, i.e. image I_{12} , are constrained by the sums of the dimensions of I_1 and I_2 (assuming the transformation induced by \mathbf{H} is correct).

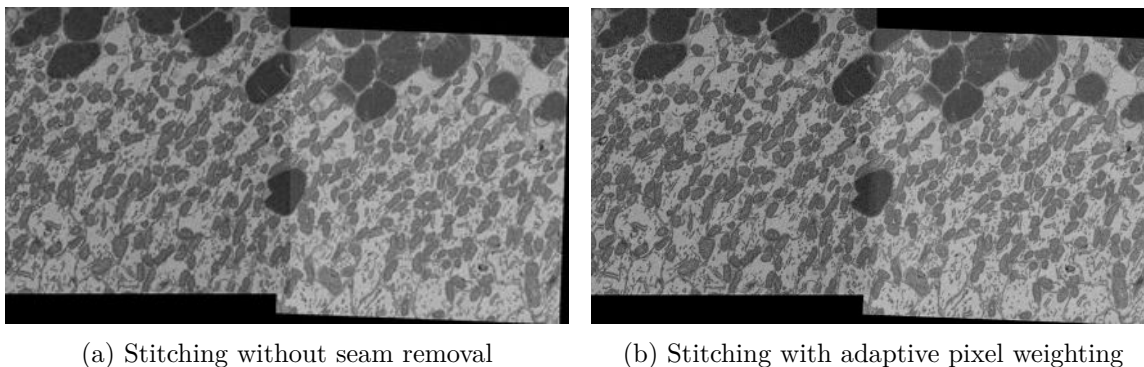


Figure 3.6: Comparison of the stitching of the images from Figure 3.1 without any form of seam removal and with adaptive pixel weighting. While brightness and contrast differences can be observed in both images, the stitching seam is much less noticeable when pixel weighting is applied.

Since fixed values of α , such as 0.5 (i.e., averaging), are not very effective, α is generally adaptive, with pixels being weighted more heavily the closer to image centres they are [64, 68]. Figure 3.6 presents a comparison between the original stitching result and the result with adaptive pixel weighting.

Additionally, it might be necessary to stitch more than just two images. However, the general steps needed for multi-image stitching remain the same. The only difference is the application of global optimisation methods, which minimise misalignments between all pairs of stitched images [64]. Doing so is necessary to reduce the accumulation of image registration errors during the stitching process, which could otherwise become too large. For conventional images, *bundle adjustment*, a global optimisation technique that relies on 3D geometry, is generally used [64]. However, bundle adjustment is unnecessarily complex for the stitching of standard arrays of EM images. Therefore, EM image stitching tools tend to use simpler methods. A common approach is to construct a minimum spanning tree (for example, using the Prim–Jarník algorithm [54]) from a graph in which nodes correspond to image tiles and edges represent tile adjacency [12, 47]. The edges need to be weighted by the quality of the estimated transformations that relate the corresponding nodes (i.e., image tiles) together. However, other techniques, which once again optimise all transformations, can be employed in EM image stitching as well [43]. With that in mind, this thesis proposes to optimise the transformation parameters using graph-based simultaneous localisation and mapping (SLAM) [21]. The proposed solution is described in Chapter 6.

3.2 Current State of Feature-Based Image Stitching

This section provides a brief overview of the current state-of-the-art in conventional feature-based image stitching. The review was inspired by [25, 42, 68].

Feature-based image stitching methods generate a sparse set of features to establish correspondences between the stitched images. Lowe et al. [41] introduced the most widely used feature detection and description method to date, the Scale-invariant feature transform (SIFT), described in detail in Section 3.1. PCA-SIFT [28] tries to improve the efficiency of SIFT at a cost to robustness to scale by PCA dimensionality reduction [26]. Alternatively, Speeded-up robust features (SURF) [3] presents a Hessian matrix-based technique to obtain

faster evaluation time at the expense of accuracy. Additionally, KAZE [2] detects two-dimensional multi-scale features in nonlinear scale spaces. Corner detectors, such as the Harris operator [22] or the Features from accelerated segment test (FAST) detector [55], can also be employed.

With these feature descriptors, image stitching can be performed by estimating homography-based transformation models. The simplest but least robust approach is to evaluate a single global homography [7]. To better handle more complicated scenes, it might be beneficial to split the processed images into multiple separate planes. For example, Gao et al. [20] proposed a dual-homography method, which divided input images into two dominant planes (the ground plane and the distant plane). Furthermore, Lin et al. [40] utilised a pre-calculated affine transform to generate a smooth affine stitching field. Unfortunately, these methods may still fail in particularly complex scenes.

To further improve image stitching precision, local transformation algorithms have been developed. In particular, As-projective-as-possible (APAP) [70] divided the images into uniform meshes and estimated homographies for each mesh separately. Chang et al. [13] then attempted to align the stitched images using a combination of projective and similarity transformations. Building on this approach, Lin et al. [37] created a smooth stitching field to linearise the underlying homography and minimise perspective distortion. A quasi-homography warping technique, which balanced the perspective distortion against projective distortion in non-overlapping regions, was proposed in [35].

Furthermore, some methods optimise seams or mesh-based alignment between stitched images. Zhang et al. [71] presented a seam-finding technique that considered both geometric alignment and image content to better address parallax⁶ distortion. Lin et al. [38] enhanced the previous method by weighting features according to their distance from the predicted seam. To reduce distortion in the non-overlapping area, Chen et al. [14] formulated image alignment as a constrained optimisation problem with a global similarity prior. Zhang et al. [72] improved on [14] by adopting additional prior constraints. Hermann et al. [23] attempted stitching with multiple registrations, each for a different image segment. Lee et al. [32] then partitioned the input images into superpixels and warped each superpixel adaptively using an optimal homography, alleviating parallax artefacts.

Additionally, other methods try to detect higher-level features, such as whole lines. For example, Xiang et al. [69] guided warping using line features and a global similarity constraint. Liao et al. [36] employed point and line features and emphasised alignment, distortion, and saliency of single-perspective warps. Finally, Jia et al. [25] divided the stitched images into coplanar regions, resulting in more consistent line and point correspondences in wide parallax images.

In general, traditional feature-based image stitching methods continue to suffer from inconsistencies in more challenging situations, often caused by large amounts of stitched images, parallax, or low-quality texture. Consequently, their performance in some practical applications remains inadequate. Recent methods seek to alleviate this issue with deep learning techniques, such as feature extraction using convolutional neural networks. Deep learning approaches to image stitching are discussed in Chapter 4.

⁶**Parallax** – the displacement between the projected positions of a point on an image plane viewed from different perspectives [57].

3.3 Traditional Stitching of Electron Microscopy Images

This section reviews image stitching methods designed for EM data. Since the vast majority of image stitching algorithms focus on generic panorama stitching, this is a comparatively much less explored area of research [61, 68]. Nevertheless, several tools for stitching tiles of EM images, and often also for constructing 3D volumes from the stitched images, exist.

One of the first and most commonly used such tools is ImageJ [58], a powerful image processing tool tailored to scientific images. Specifically, for EM images, various ImageJ plugins provide comprehensive stitching, registration, and visualisation utilities. The most notable of the plugins is TrakEM2 [10], which incorporates a SIFT-based image stitching algorithm. In contrast to that, the Microscopy Image Stitching Tool (MIST) [12], which is also available as an ImageJ plugin, employed phase correlation [31] to compute image registrations. Additionally, MIST estimated the mechanical stage model parameters (e.g., actuator backlash and camera angle) to minimise stitching errors.

More recently, standalone methods, often completely separate from ImageJ, were developed as well. For example, Ding et al. [34] proposed a novel image stitching technique based on SURF feature detection [3] and PCA dimensionality reduction [26]. Evaluation on ceramic EM images displayed slightly better performance than traditional SIFT stitching. Moreover, Singla et al. [61] presented NanoStitcher, an EM image stitching tool combining the approaches from ImageJ and MIST. In doing so, Singla et al. minimised feature detection issues in low-resolution images and phase correlation inconsistencies caused by image intensity variations. However, these enhancements resulted in longer execution times. Vescovi et al. [67] then introduced an end-to-end pipeline for stitching (performed with TrackEM2), volume assembly, and segmentation of EM images. Furthermore, Mahalingam et al. [43] proposed ASAP, the Assembly Stitching and Alignment Pipeline. ASAP targeted high-resolution petascale datasets, processing them at rates that match microscope imaging speeds. For stitching, ASAP employed SIFT feature detection aided by lens distortion estimation. Finally, Alignment by Simultaneous Harmonization of Layer/Adjacency Registration (ASHLAR), a tool for stitching and registration of highly multiplexed images, was presented in [47]. ASHLAR represented image tiles with an adjacency graph and utilised phase correlation and minimum spanning tree construction to stitch the tiles together.

Overall, all of the above methods continue to rely on handcrafted features or similar approaches. Consequently, they may have difficulty adequately addressing the challenges of volume EM stitching described in Section 2.1. With that in mind, this thesis proposes a novel EM image stitching tool, which attempts to improve stitching performance using deep learning. The proposed solution is presented in Chapter 6.

Chapter 4

Deep Learning Methods for Image Stitching

Chapter 3 presented several conventional techniques for combining overlapping images into a single composite image. However, these traditional image stitching methods have a substantial number of shortcomings. For example, stitching of images captured at significantly different camera angles might result in noticeable parallax-induced blurring and ghosting [42, 63]. Furthermore, feature-based approaches may fail to identify sufficient amounts of interest points in areas with low-quality texture, repetitive patterns, or high intensity variance [42, 62]. This could prove detrimental to applications where noisy and poorly textured images are common, such as volume electron microscopy (volume EM), introduced in Chapter 2.

As outlined in Section 3.3, volume EM stitching methods may try to overcome feature detection issues by using other stitching strategies, e.g., variants of phase correlation [31]. Unfortunately, while correlation approaches are generally faster and do not rely on features, they are limited in terms of supported transformations [29, 47]. In particular, phase correlation can adequately align only images that are translated relative to each other. Rotation, scaling, and general affine transformations are much more difficult to process appropriately. Moreover, intensity variations, another common phenomenon in volume EM, might influence the stitching result to even greater extents [61].

As a consequence of the above issues, researchers have recently started turning towards deep learning and convolutional neural networks (CNNs) in an effort to develop more robust image stitching techniques [49]. Specifically, some works replace parts of the traditional stitching pipeline from Section 3.1, namely feature detection and matching or the entire homography estimation stage (including the initial feature analysis). These are described in Sections 4.1 and 4.2, respectively. Section 4.3 then reviews a different class of deep learning stitching approaches: complete end-to-end networks that directly generate the final combined image. It should be emphasised that, compared to Chapter 3, we do not discuss deep learning-based methods designed specifically for EM images. This is because, to the best of our knowledge, no other published works have yet attempted to use deep learning for EM image stitching.

4.1 Learning-Based Feature Detection and Matching

The simplest approach to deep image stitching is to replace handcrafted feature detection and matching with deep learning-based alternatives. In this scenario, the last two stages of the traditional stitching pipeline from Section 3.1, i.e., homography estimation and image alignment and stitching, remain unchanged. As proven by recent developments in image processing, CNNs are prime candidates for such a task since they allow highly robust feature extraction [17]. For example, Shi et al. [60] propose a classical CNN-based architecture for image stitching and achieve slightly better results than traditional approaches, such as SIFT [41]. However, this architecture employs deep learning for feature detection only and leaves feature matching to ordinary methods, which could significantly limit its performance [56]. Additionally, it appears to be unnecessary to design a specialised CNN for image stitching, since, in principle, any network capable of detecting features in pairs of images can be used. Hoang et al. [24] reinforce this idea by successfully stitching images while relying purely on a pre-trained feature extraction model.

In light of the above, it might be preferable to utilise well-established deep learning frameworks for both feature extraction and matching. The pioneering works in this area are SuperPoint [17] and SuperGlue [56] for feature detection and matching, respectively. SuperPoint presents a fully convolutional feature extraction network that significantly outperforms conventional techniques, especially on noisy images and under large illumination changes. SuperGlue then builds on the output of SuperPoint, calculating feature matches using an attentional graph neural network. Attention (introduced later in this section) was chosen because of its ability to focus on both local and global relationships. Again, this results in significant improvements over traditional methods (nearest neighbour distance ratio testing [44] and RANSAC [19]), further demonstrating the advantages of learning-based approaches. However, the fact that SuperPoint and SuperGlue are two completely separate networks can have a negative impact on overall accuracy, especially in areas with less texture or repetitive patterns [62]. To mitigate this issue, Sun et al. [62] propose LoFTR, a unified network for the direct detection of feature matches between pairs of images. To

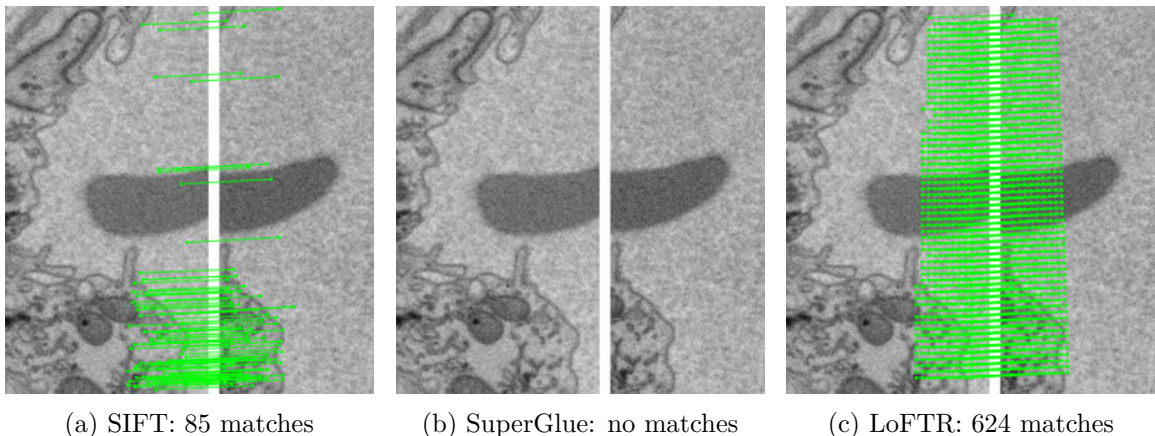


Figure 4.1: Comparison of feature matching results between SIFT [41], SuperPoint [17] with SuperGlue [56], and LoFTR [62] on an EM image with low-quality texture (sourced from [33]). Matches are shown after outlier elimination using RANSAC [19]. LoFTR managed to detect over seven times more inlier matches than SIFT, while SuperPoint with SuperGlue found no matches at all.

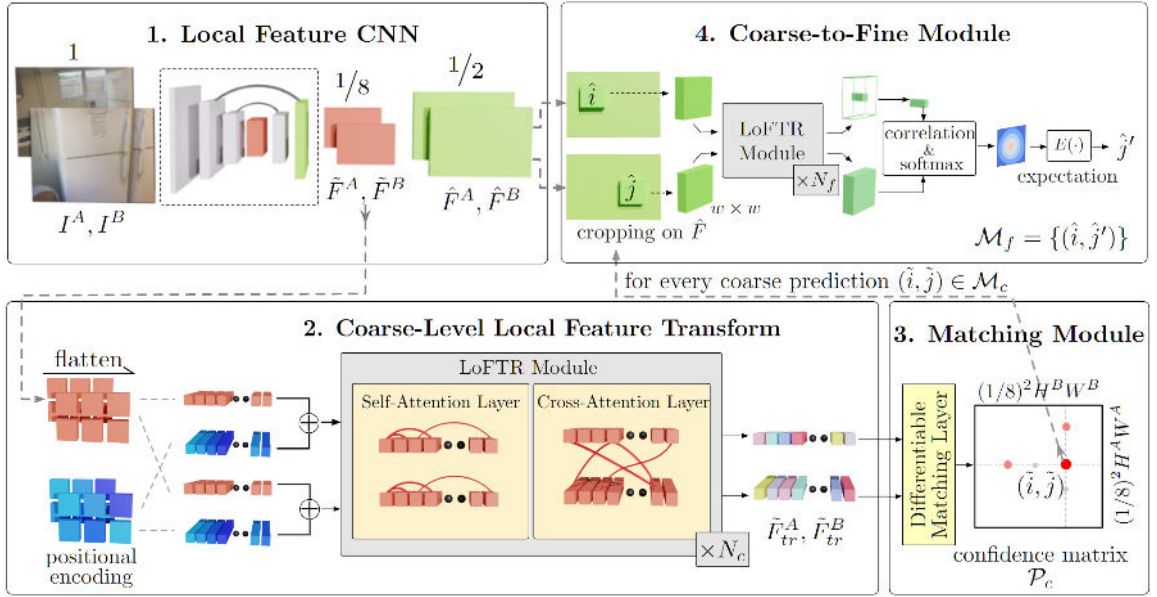


Figure 4.2: The architecture of LoFTR. LoFTR has four sequential components. First, a local feature CNN extracts coarse-level feature maps \tilde{F}^A , \tilde{F}^B and fine-level feature maps \hat{F}^A , \hat{F}^B from the input images I_A , I_B , respectively. Second, the coarse-level feature maps are flattened to 1D vectors and positionally encoded. The encoded features are processed by the main Local Feature Transformer module (LoFTR module), which contains several self-attention and cross-attention layers. Next, a differentiable matching layer matches the transformed features, generating the confidence matrix \mathcal{P}_c . Coarse-level matches \mathcal{M}_c are selected using a confidence threshold and the mutual nearest neighbour criterion. Finally, a local window is cropped from fine-level feature maps for each coarse-level match prediction, and the final prediction set \mathcal{M}_f is calculated. The image was retrieved from [62].

illustrate its effectiveness on images with poor texture, a comparison between SIFT [41], SuperPoint and SuperGlue, and LoFTR is presented in Figure 4.1.

The remainder of this section provides a more detailed overview of LoFTR since LoFTR is the backbone of the stitching solution proposed in Chapter 6. A complete description of LoFTR can be found in [62].

Local Feature Transformer (LoFTR)

The *Local Feature Transformer* (LoFTR) [62] operates directly on pairs of input images, extracting feature matches without the need for a separate feature detector. To this end, LoFTR employs a convolutional backbone, a Transformer [66] with a combination of *self-attention* and *cross-attention* layers, a differentiable matching layer, and a coarse-to-fine refinement module. The architecture, illustrated in Figure 4.2, is described in detail in the following.

Local feature extractor LoFTR follows the notion of using an established feature extraction network as the backbone. In particular, given a pair of images I^A and I^B , the images first pass through a standard feature pyramid network (FPN) [39], denoted as the local feature CNN in Figure 4.2. The FPN extracts two types of features from the respective

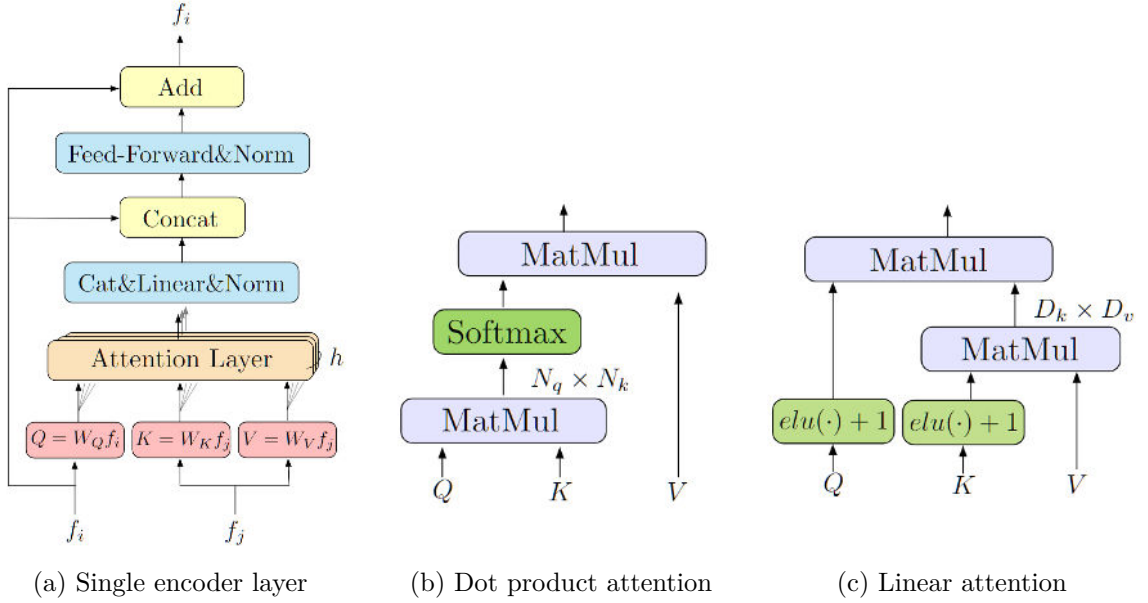


Figure 4.3: Computational graphs of a Transformer encoder layer, dot product attention, and linear attention. The encoder layer features multi-head attention with h heads. Due to kernel function alterations, the computational complexity is quadratic for dot product attention, while being linear for linear attention. The images were obtained from [62].

input images: *coarse-level features* \tilde{F}^A and \tilde{F}^B , and *fine-level features* \hat{F}^A and \hat{F}^B . The coarse-level features are extracted at $1/8$ of the original image resolution, while the fine-level features are extracted at $1/2$ of the original resolution.

Local Feature Transformer module After feature extraction, the coarse-level features \tilde{F}^A and \tilde{F}^B are flattened and processed by the Local Feature Transformer module (LoFTR module), which makes the features dependent on position and context. The transformed features are denoted as \tilde{F}_{tr}^A and \tilde{F}_{tr}^B in Figure 4.2. To achieve the transformation, each element of \tilde{F}^A and \tilde{F}^B is first positionally encoded, receiving unique position information in the sinusoidal domain. The acquired positional dependency allows LoFTR to operate on mutually indistinguishable regions (e.g., walls) of the input images with higher accuracy. The positionally encoded features are passed to a Transformer [66] encoder, which consists of several sequential encoder layers. The architecture of a single encoder layer is shown in Figure 4.3a.

Transformer encoder and attention The fundamental part of an encoder layer is the attention layer. Its input vectors, generally known as the query, key, and value vectors, function similarly to information retrieval from a database. In particular, the query vector Q retrieves information from the value vector V based on attributes given by the key vector K . In its most well-known form, illustrated in Figure 4.3b, this process can be formally described as the dot product attention [62]:

$$\text{attention}(Q, K, V) = \text{softmax}(QK^T)V. \quad (4.1)$$

Linear Transformer Let us denote the length of Q and K as N . Then, the computational complexity of the original dot product attention is $O(N^2)$. This is impractical since even at the coarse-level resolution the length of the input features can be rather large. Therefore, LoFTR employs linear attention [27], which replaces the original softmax kernel with the similarity function $\text{sim}(Q, K) = \phi(Q) \cdot \phi(K)^\top$, where $\phi(\cdot) = \text{elu}(\cdot) + 1$ [62]. As a result of this change, linear attention reduces the computational complexity of attention to $O(N)$, achieving up to several thousand times faster speeds on long input sequences at negligible costs to network performance [27]. Figure 4.3c displays the computational graph of linear attention.

Self-attention and cross-attention To complete the transformation module, N_c pairs of so-called *self-attention* and *cross-attention* layers are sequentially stacked. These linear attention layers differ only in the type of input features f_i and f_j they process: self-attention layers use features from the same image (either I_A or I_B) for both f_i and f_j , while cross-attention layers use features from both images.

Coarse-level matching After obtaining the transformed features \tilde{F}_{tr}^A and \tilde{F}_{tr}^B , LoFTR attempts to extract high-confidence matches. First, a score matrix \mathcal{S} , representing all possible feature matches, is calculated between the transformed features. If we denote the number of features in \tilde{F}_{tr}^A and \tilde{F}_{tr}^B as N^A and N^B , respectively, then the score matrix $\mathcal{S} \in \mathbb{R}^{N^A \times N^B}$ can be obtained as

$$\mathcal{S}_{i,j} = \frac{1}{\tau} \cdot \langle \tilde{F}_{tr_i}^A, \tilde{F}_{tr_j}^B \rangle, \quad \forall 1 \leq i \leq N^A, \quad \forall 1 \leq j \leq N^B, \quad (4.2)$$

where $\langle \cdot, \cdot \rangle$ is the inner product and τ is a coefficient.

Second, LoFTR uses a dual-softmax operator¹ to calculate the matching probability matrix $\mathcal{P}_c \in \mathbb{R}^{N^A \times N^B}$:

$$\mathcal{P}_{c_{i,j}} = \text{softmax}(\mathcal{S}_{i,\cdot})_j \cdot \text{softmax}(\mathcal{S}_{\cdot,j})_i, \quad \forall 1 \leq i \leq N^A, \quad \forall 1 \leq j \leq N^B. \quad (4.3)$$

Intuitively, LoFTR obtains the probability of a pair of features being mutual nearest neighbours in the score space by applying softmax in both dimensions of \mathcal{S} separately.

Finally, coarse-level matches \mathcal{M}_c are selected as matches that satisfy the mutual nearest neighbour criterion and have a confidence score higher than a specified threshold.

Coarse-to-fine refinement After identifying matches at the coarse level, they need to be refined to the original input image resolution to achieve higher precision. To this end, LoFTR crops windows of size $w \times w$ from the fine-level feature maps \hat{F}^A and \hat{F}^B for each coarse match. In particular, windows around the estimated fine-level position (\hat{i}, \hat{j}) of each coarse match $(\tilde{i}, \tilde{j}) \in \mathcal{M}_c$ are cropped. The feature windows are then transformed by a secondary LoFTR module (smaller, with only $N_f < N_c$ pairs of self-attention and cross-attention layers). For each (\hat{i}, \hat{j}) , this yields two transformed feature maps: $\hat{F}_{tr_i}^A$ and $\hat{F}_{tr_j}^B$, centred around pixels \hat{i} and \hat{j} , respectively. Subsequently, the central vector of $\hat{F}_{tr_i}^A$ is correlated with the vectors in $\hat{F}_{tr_j}^B$, producing a distribution of match probability between

¹Additionally to dual-softmax, LoFTR supports matching with an optimal transport layer as in [56]. However, the performance differences between the two are generally negligible [62].

the pixels in the neighbourhood of \hat{j} and the pixel \hat{i} . The final fine-level matching pixel \hat{j}' from I^B can be obtained by computing the expectation over the resulting distribution. Finally, all refined matches are combined to form the fine-level set of matches \mathcal{M}_f , which represents the final output of LoFTR.

Supervision LoFTR is trained in a fully supervised manner, with losses for both the coarse and fine resolutions. The coarse-level loss calculates the negative log-likelihood over the matching probability matrix \mathcal{P}_c . The ground truth labels for \mathcal{P}_c are generated automatically from camera poses and depth maps attached to the training images. The fine-level loss is based on the standard ℓ_2 loss.

In general, LoFTR and other learning-based feature detection and matching techniques provide a robust, yet relatively simple approach to deep image stitching. Hence, compared to the methods introduced in Sections 4.2 and 4.3, methods that only replace the traditional feature detection and matching stages are extremely versatile. Consequently, they can also be used for many other computer vision tasks in addition to image stitching, such as pose estimation [56, 62].

4.2 Methods Using Deep Homography Estimation

The second approach to deep image stitching is *deep homography estimation*. In principle, deep homography estimation is rather similar to the learning-based feature detection and matching methods introduced in Section 4.1. However, instead of replacing only the initial two feature processing stages of the traditional stitching pipeline, deep homography estimation techniques go one step further and try to directly compute a homography relating a pair of input images together. As described in Section 3.1, the resulting homography can then be used to perform image alignment and stitching of the input images.

The idea of deep homography estimation was first proposed by DeTone et al. [16] and their HomographyNet (shown in Figure 4.4). HomographyNet starts by stacking two input images channel by channel. Then, the stacked images pass through eight convolutional layers. Finally, two fully connected layers regress the eight parameters of the estimated homography.

Although the above architecture is relatively straightforward and suitable only for small images, experiments in [16] already suggest slightly better performance than traditional

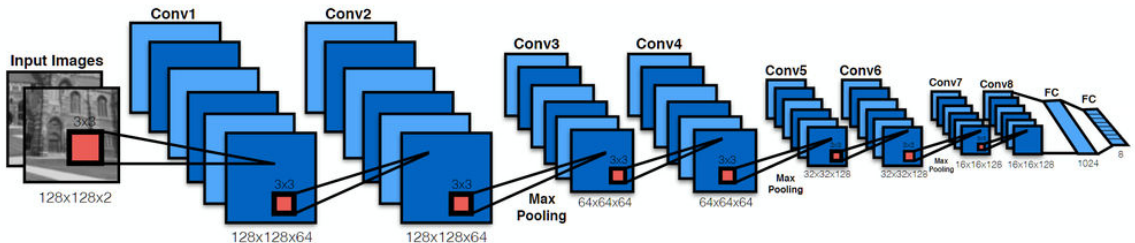


Figure 4.4: HomographyNet [16], a CNN for direct estimation of a homography relating two input images together. The estimation is done by standard regression of the eight parameters of the homography. The image was acquired from [16].

methods. However, these performance improvements do not necessarily translate to image stitching, because simple channel-wise image stacking might be insufficient for images with small overlaps [74]. Therefore, Zhao et al. [74] instead propose correlating the features extracted from the input images. Moreover, they suggest a coarse-to-fine homography estimation strategy, in which they first retrieve a rough estimate of the required homography. The rough estimate is then iteratively refined at progressively higher resolutions. These modifications enable the network to handle most image stitching tasks better than SIFT [41] with RANSAC [19]. This is especially true for images with poor texture or repetitive patterns [74]. The effectiveness of the coarse-to-fine correlation approach is further evidenced by Nie et al. [50], who adopt a similar strategy and also obtain higher accuracy compared to SIFT with RANSAC.

Overall, deep homography methods can achieve superior performance with increased robustness compared to traditional approaches. However, since these networks directly estimate homography parameters, they may learn to expect more complex image transformations than those generally found in applications such as volume EM. Therefore, these methods could be unnecessarily difficult to adapt to EM images, as even rigid transformations tend to be more than sufficient for EM images [47].

4.3 End-to-End Deep Image Stitching Networks

The last and currently least explored deep image stitching approach is stitching via end-to-end neural networks. This is a particularly challenging task since it integrates feature detection and matching, homography estimation, and image alignment and stitching directly into CNNs [48]. Therefore, multistage networks are adopted to perform end-to-end stitching in an attempt to decompose the problem into smaller parts.

The first completely end-to-end stitching network was proposed in [48]. However, this network contains fully connected layers. Consequently, it cannot handle images with arbitrary resolutions. To mitigate this issue and improve the overall performance of the original network, Nie et al. presented UDIS [49]. The architecture of UDIS is shown in Figure 4.5.

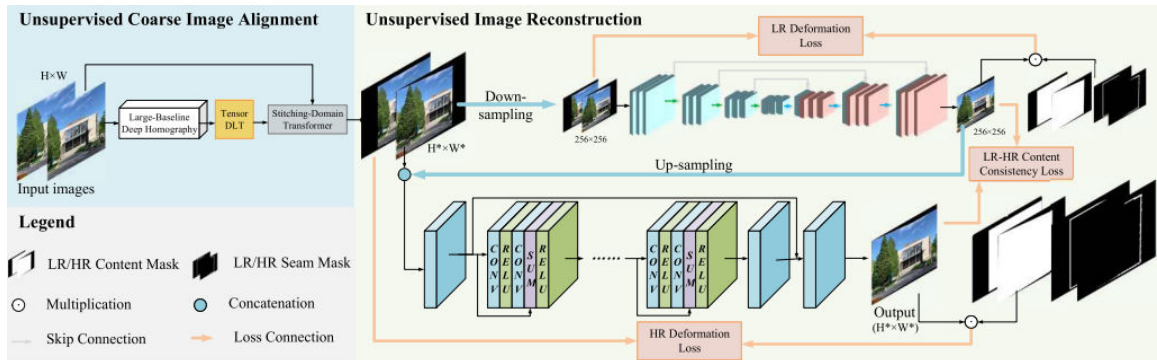


Figure 4.5: The architecture of UDIS [49]. First, a homography that relates two input images together is estimated using the homography estimation network from [50]. Second, the images are separately aligned according to the estimated homography. Then, a low-resolution hourglass network calculates the deformation needed to complete the stitching at a coarse level (top right). Finally, the coarse deformation is refined to the original resolution by a fully convolutional network (bottom right). The image was retrieved from [49].

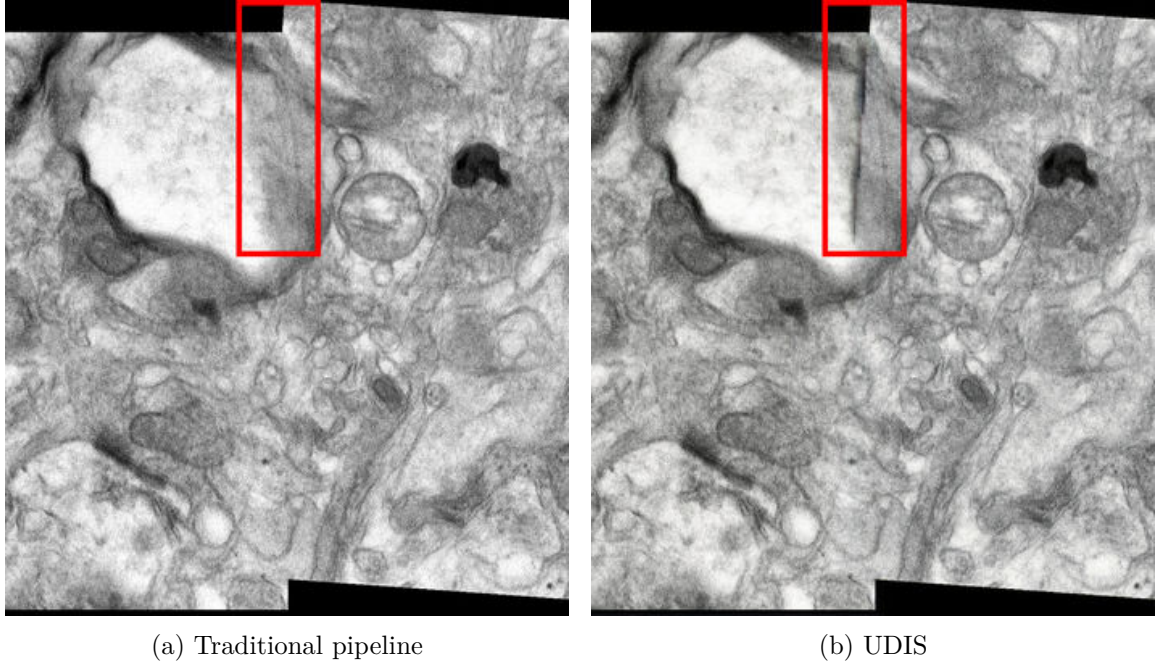


Figure 4.6: Comparison of stitching using the traditional pipeline from Section 3.1 and the end-to-end UDIS [49] network. Black spot artefacts (highlighted by the red rectangle) can be seen in the image stitched by UDIS. The stitched images were generated from [18].

UDIS is divided into two primary stages. The first stage, unsupervised coarse image alignment, utilises the coarse-to-fine correlation network from [50] to compute a homography that relates a pair of input images together. The homography is then used to coarsely align the input images. This produces two images with a shared coordinate system, which are fed into the unsupervised image reconstruction stage. The reconstruction stage generates the final stitched image in a coarse-to-fine manner. In particular, an hourglass network first learns the deformation required to stitch the images at a lower resolution. The coarse stitching estimate is then refined to the original resolution by a high-resolution refinement branch composed entirely of convolutional layers. In doing so, the reconstruction stage attempts to correct any misalignments caused by the initial coarse alignment stage. Such corrections are crucial for many real-life applications and are the main potential benefit of end-to-end stitching networks. This is because a single homography can only accurately describe a transformation between images at the same depth [49].

Unfortunately, while UDIS demonstrates promising results compared to both traditional and other deep learning methods, it has one major downside that limits its applicability. Due to its generative nature, it may produce visible artefacts, such as black spots, in the stitched images (especially at higher resolutions). An example of such artefacts can be seen in Figure 4.6. This shows that more research is needed before end-to-end stitching networks can be employed in areas where image precision is essential, such as volume EM. Hence, other learning-based approaches should be preferred in such scenarios.

Chapter 5

Proposed Synthetic Electron Microscopy Stitching Dataset

Data are a crucial part of any machine learning task. Unfortunately, when it comes to electron microscopy (EM), it is difficult to find a suitable and publicly available dataset tailored towards image stitching (described in detail in Chapter 3). In general, EM datasets are intended for biological research [65] or other image processing tasks, such as pre-training for image segmentation models [15]. These datasets are composed of already stitched images or individual image tiles that did not require stitching in the first place. Hence, they lack the training metadata required for image stitching. To the best of our knowledge, the only stitching-focused EM dataset is proposed in [12] for evaluating MIST. However, this dataset is not intended for machine learning tasks and only contains images related to stem colony growth, making it rather domain-specific.

With respect to the above, this chapter proposes a novel synthetic dataset, the Deep Electron Microscopy Image Stitching (DEMIS) dataset, designed specifically with EM image stitching in mind. The dataset is generated programmatically from a manually selected set of publicly¹ available high-resolution EM images. The image selection process is described in Section 5.1. Each of the selected images is divided into a grid of overlapping tiles, and randomised image transformations are applied to each tile. The tile generation procedure is presented in Section 5.2.

5.1 Selecting Images for the Synthetic Dataset

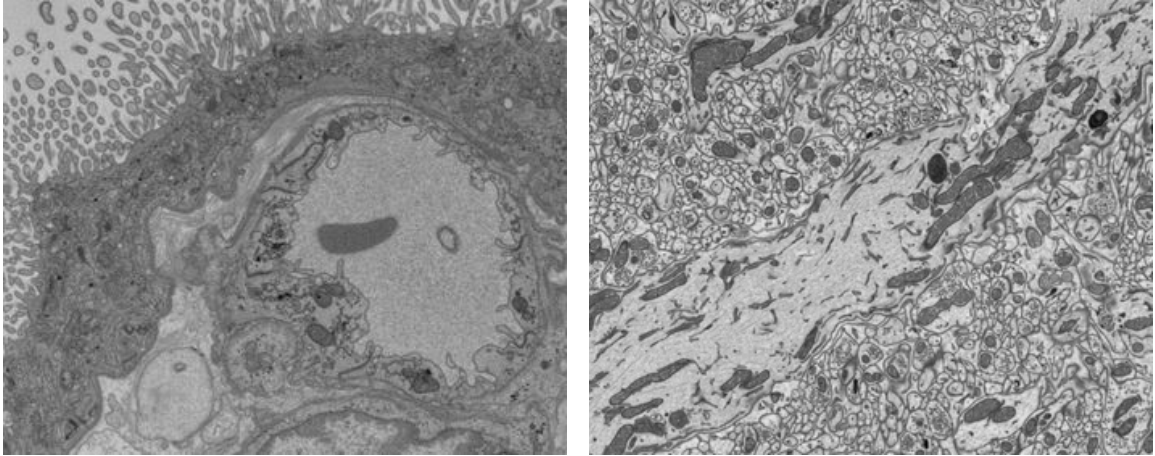
To support further research in the area of EM image stitching, images for DEMIS were selected primarily from public sources of EM images. In particular, two online databases of EM images were used: the Electron Microscopy Public Image Archive (EMPIAR)² and The Cell Image Library (CIL)³. Examples of selected images are shown in Figure 5.1.

Both EMPIAR and CIL provide a place for researchers to share their volume EM images with other researchers and the general public (EMPIAR is focused specifically on volume EM imagery, while CIL allows images captured by other methods as well). However, while EMPIAR and CIL both provide an extensive collection of EM images, it is important

¹Some images were provided by TESCAN 3DIM. However, these were used only for training purposes and will not be a direct part of this thesis.

²EMPIAR – <https://www.ebi.ac.uk/empiar/>.

³CIL – <http://cellimagelibrary.org/>.



(a) Example image selected from EMPIAR [33]. (b) Example image selected from CIL [9].

Figure 5.1: Sample images from EMPIAR and CIL selected for the DEMIS dataset. These images were further divided into grids of overlapping images as explained in Section 5.2.

to respect the variety of different acquisition methods used in volume EM (summarised in Chapter 2). Additionally, imaging defects and noise can also complicate the selection process. Consequently, all images for the DEMIS dataset were selected manually. During the selection, several rules were followed:

- Only images with a resolution of at least 2048×2048 pixels were considered. This guarantees that high-resolution images can be generated during the subsequent splitting into grids of overlapping image tiles, which is crucial to mimic the high-resolution imagery typical of volume EM.
- For an image to be selected, it had to contain its imaged sample in the vast majority of its area (ideally all of its area, as demonstrated, for example, in Figure 5.1b). This ensures that the DEMIS dataset will be rich in relevant EM image content.
- Although similar images were allowed to a certain degree (especially when the overall image quality was high), variety in types of imaged samples and imaging techniques was strongly preferred. Some differences in image content are illustrated in Figure 5.1.

Using the rules presented above, a total of 424 individual EM images (259 from EMPIAR, 165 from CIL) from 36 different public EM imaging projects were selected as the basis for the DEMIS dataset. Additionally, 172 high-quality and high-resolution EM images were kindly provided by TESCAN 3DIM, making the total number of source images available for the DEMIS dataset 596. However, the images provided by TESCAN 3DIM will not be publicly available as part of this thesis. Hence, they were not used for evaluation purposes to facilitate better reproducibility of the solution proposed in Chapter 6.

Each of the selected images discussed above was then further split into overlapping image tiles. The image splitting process is outlined in Section 5.2.

5.2 Generating the Synthetic Dataset

This section provides details about the splitting of images selected for the DEMIS dataset according to Section 5.1. Each of the 596 source images was split into as many overlapping

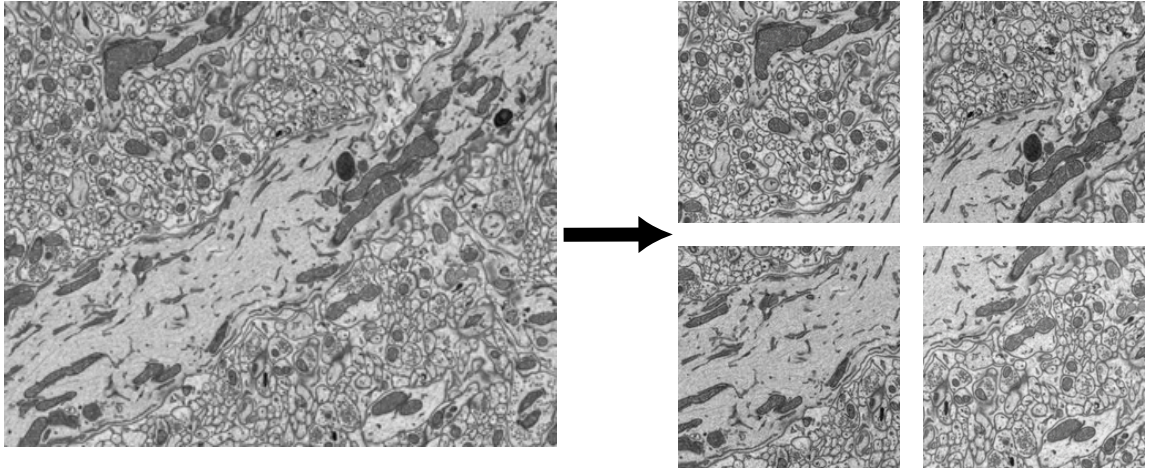


Figure 5.2: The image from Figure 5.1b split into a grid of four overlapping image tiles of size 1024×1024 pixels. The number of generated tiles depends on the resolution of the source image.

image tiles of size 1024×1024 pixels as the original image resolution allowed. An example of the splitting can be seen in Figure 5.2. Each image tile was generated as follows. The process is illustrated in Figure 5.3.

1. First, the initial pixel position of the tile in the original image is determined. The position is decided primarily based on the resolution of the tile, the position of adjacent tiles, and the base overlap between neighbouring tiles (set to 20% of the tile resolution). Additionally, the starting position is shifted slightly in a deterministic way to avoid the boundaries of the original image and to give space for further transformations (described below). This shifting prevents the creation of black bars in the generated tiles.
2. Second, the starting position is shifted on both axes by a uniformly generated random translation. The maximum translation shift was set to 3% of the tile resolution.
3. Furthermore, the original image is rotated around the translated tile position in either direction. To achieve this, a random rotation angle is generated using a uniform distribution. The largest obtainable angle and the largest possible rotation relative to the original orientation were both set to 5 degrees. The rotation angle of the first tile is always 0.
4. Then, the image tile is cropped around the translated tile position from the rotated source image.
5. Finally, the tile has Gaussian noise and randomised brightness and contrast changes applied to it. The values corresponding to these transformations were generated using Gaussian distributions with zero means and variances of 25, 75, and 0.0033, respectively.

In total, the above method resulted in a total of 10 883 generated image tiles (8339 without images from TESCAN 3DIM), all of which were labelled by the respective tile positions in the original image to allow use in deep learning-based image stitching. Of the

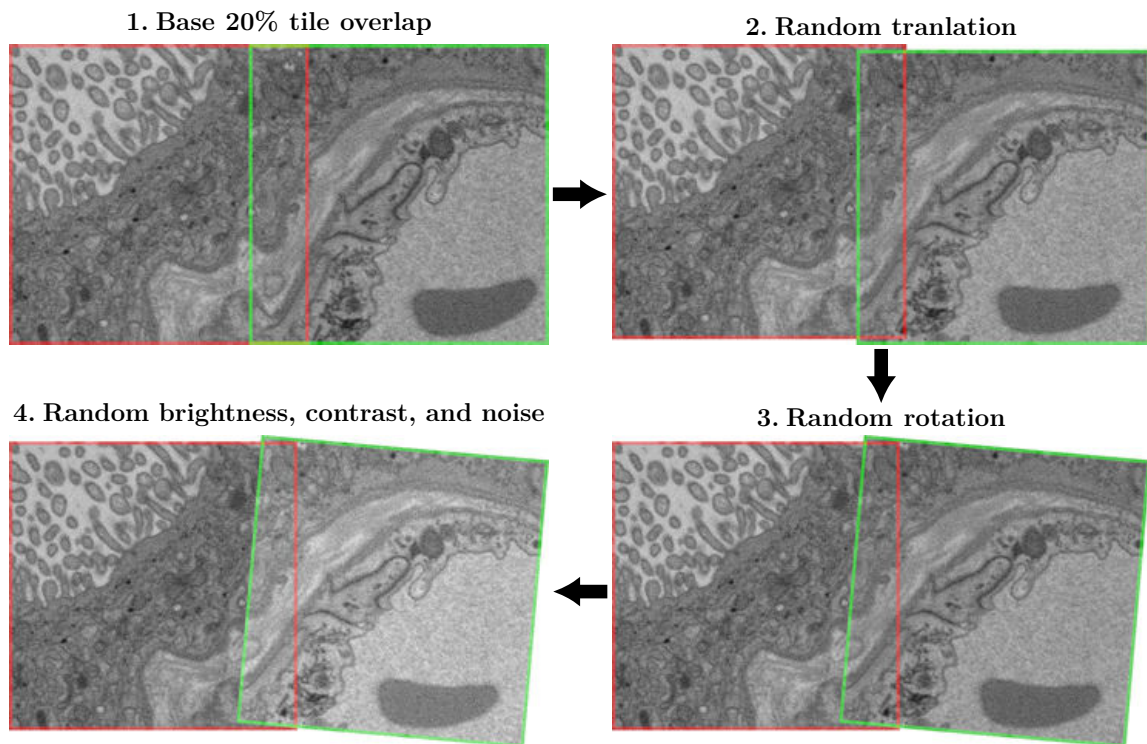


Figure 5.3: The splitting of a new tile (green) taken from the image presented in Figure 5.1a. First, the starting position of the new tile is determined according to the base 20% overlap. Then, random translation and random rotation are applied. Finally, the brightness and contrast of the tile are randomly adjusted, and a small amount of Gaussian noise is added.

10 883 image tiles, corresponding to the complete set of 595 source images from Section 5.1, 8826 were selected for training (6282 without TESCAN 3DIM images), 751 for validation, and 1306 for evaluation purposes.

Chapter 6

Proposed Electron Microscopy Image Stitching Tool

As discussed in Section 3.3, existing stitching tools for volume electron microscopy (EM) images, such as ImageJ [58] and TrackEM2 [10], MIST [12], and ASHLAR [47], continue to rely purely on variants of the traditional image stitching pipeline, presented in Section 3.1. However, as described in Chapter 4, current deep learning methods promise improvements in stitching performance and robustness compared to traditional approaches, which often employ SIFT [41] features. Therefore, this chapter proposes a novel tool—the Deep Electron Microscopy Image Stitching (DEMIS) tool—for stitching EM images using deep neural networks that have shown satisfactory results on conventional images. To the best of our knowledge, there have not yet been any published research works that try to apply deep learning to stitch EM images. The stitching pipeline used by the DEMIS tool is displayed in Figure 6.1, and the top-level algorithm that formalises the pipeline is briefly introduced below and described in detail in Section 6.1.

First, DEMIS loads a set of raw EM images that form a grid of overlapping tiles. As outlined in Section 2.1, the raw EM image tiles in the grid may suffer from brightness and contrast inconsistencies, which could affect both the stitching accuracy and the quality of the final result. Hence, DEMIS attempts to normalise both the brightness and contrast of the loaded images to a predefined range.

Subsequently, DEMIS goes over all pairs of adjacent tiles (which should have a large enough overlap to allow stitching) and detects and matches features between them. DEMIS utilises these feature matches to estimate transformations that relate each of the image pairs together. In this step, explained in Section 6.2, feature detection and matching are performed by LoFTR [62] (described in Section 4.1), and transformation estimation is done using OpenCV¹. LoFTR is proposed as the deep learning technique of choice since, as discussed in Sections 4.2 and 4.3, other deep learning stitching methods might introduce unnecessarily complex image transformations or produce visible artefacts in the final image. Since LoFTR was originally trained only on conventional image pairs, it is also suggested to fine-tune LoFTR directly on EM images (specifically, on images from the DEMIS dataset proposed in Chapter 5). The fine-tuning process is presented in Section 6.5.

Finally, the pairwise transformations are optimised globally using graph-based simultaneous localisation and mapping (SLAM) [21]. In particular, a graph of the stitched grid is constructed, in which vertices represent images in the grid and edges the estimated pairwise

¹OpenCV – <https://opencv.org/>.

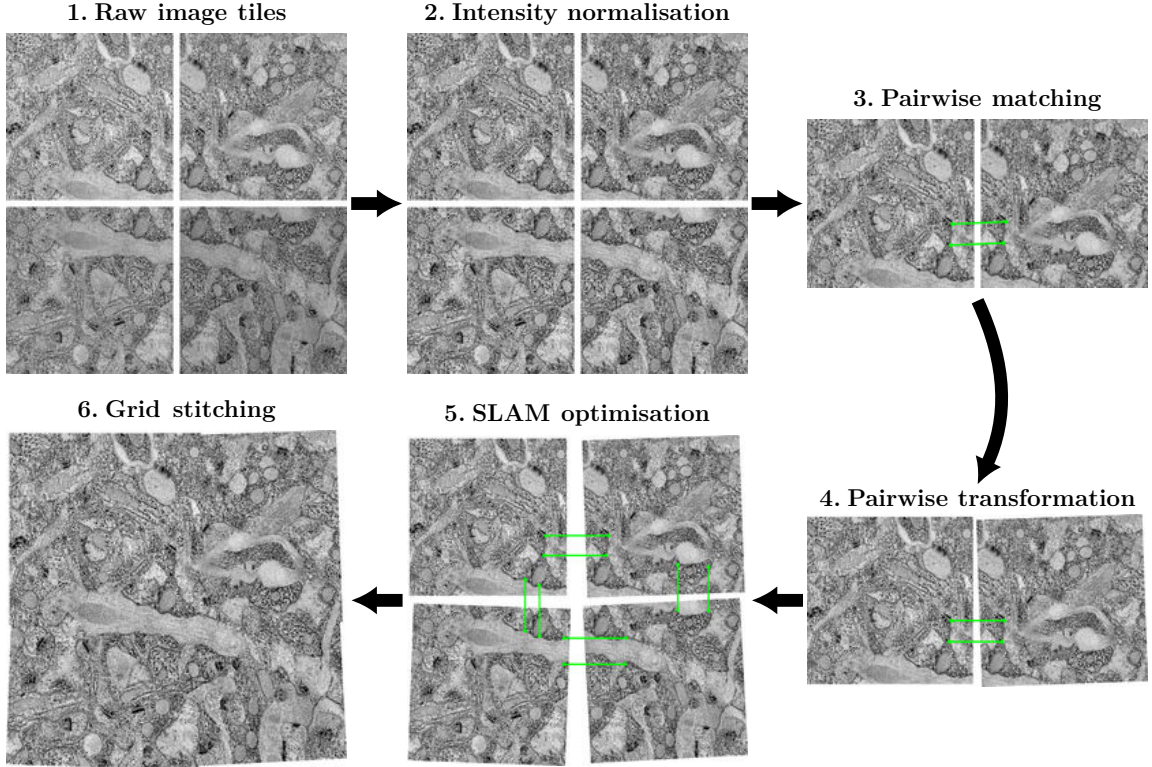


Figure 6.1: The stitching pipeline of the proposed DEMIS tool. First, the brightness and contrast of raw input images are normalised. Second, for each pair of adjacent images, features are detected and matched by LoFTR [62]. The matches are then used to estimate the initial transformations relating the adjacent images together. Subsequently, a SLAM graph is constructed based on the expected grid structure and the estimated transformations. The transformations in the graph are then optimised globally. Finally, the grid is stitched by gradually applying the optimised transformations to individual image tiles. The image tiles were generated from [8].

transformations between them. The transformations in the graph are then optimised, and the final composite image is created by gradually applying the transformations to images in the grid. The SLAM optimisation and the construction of the composite image are explained in Sections 6.3 and 6.4, respectively.

6.1 Top-Level Grid Stitching Algorithm

This section describes the top-level grid stitching algorithm used by the DEMIS tool. The top-level algorithm is the entry point of the grid stitching process and, as such, directs the execution of all the steps outlined by the pipeline in Figure 6.1. The top-level algorithm, presented in Algorithm 6.1, expects to receive information about the grid of EM images to stitch. This is represented by the mapping G , which maps 2D grid positions (that is, pairs of row and column indices) to the corresponding images in the grid. The grid is assumed to have g_r rows and g_c columns and to contain images $w \times h$ pixels in size. The complete algorithm is explained below.

Algorithm 6.1: Top-level EM image grid stitching

Input: Mapping G of 2D grid positions to stitched EM image tiles forming the grid
Width w and height h of images in G
Number of rows g_r and columns g_c in G
Expected tile overlap ratio o
Scaling factor s for feature detection and matching
Result: Images in G stitched into a single composite image

```
// Normalise brightness and contrast of all tiles
1: foreach  $(p, I) \in G$  do
2:    $I := \text{normalise}(I)$ 

// Estimate transformations between adjacent tiles
3:  $T := \text{map}()$ 
4: foreach  $(p_1, I_1) \in G$  do
   // Let  $p_1 = (r_1, c_1)$ 
   // Find positions of adjacent tiles in top-to-bottom, left-to-right order
5:    $A := \emptyset$ 
6:   if  $r_1 < g_r$  then
7:      $A := A \cup \{(r_1 + 1, c_1)\}$ 
8:   if  $c_1 < g_c$  then
9:      $A := A \cup \{(r_1, c_1 + 1)\}$ 

   // Estimate transformations to adjacent tiles
10:  foreach  $p_2 \in A$  do
11:     $I_2 := G(p_2)$ 
12:     $T(p_1, p_2) := \text{get\_tile\_transform}(I_1, I_2, p_1, p_2, o, s)$ 

// Optimise transformation estimates using SLAM
13:  $T_{\text{opt}} := \text{optimise\_transforms}(T, w, h, o)$ 

// Stitch the grid using the optimised transformations
14: return  $\text{stitch\_grid}(G, T_{\text{opt}})$ 
```

First, Algorithm 6.1 iterates over all images in G and normalises their brightness and contrast to a predefined range (Lines 1–2). Doing so ensures that brightness and contrast inconsistencies, which are typical for EM images, do not negatively affect feature detection and matching or the visual aspects of the final stitched image (e.g., by creating evident image tile boundaries). To perform normalisation, contrast-limited adaptive histogram equalisation (CLAHE) [53], implemented, e.g., by OpenCV, is proposed. Adaptive histogram normalisation is important, since, as shown in Figure 5.1, the content in different parts of individual EM images can vary considerably.

Second, Algorithm 6.1 creates the mapping T of pairs (p_s, p_t) to pairwise tile transformations, where p_s and p_t represent the 2D grid positions of the *source* and *target* image tiles of the transformations, respectively (Lines 3–12). In other words, $T(p_s, p_t)$ is the transformation that relates the coordinate space of the image at position p_s to the coordinate space of the image at position p_t . Initially, the mapping T is empty, which is symbolised by $\text{map}()$ on Line 3. T is then progressively built by iterating over each grid position $p_1 = (r_1, c_1)$

and its corresponding image I_1 in G . For each $(p_1, I_1) \in G$, the following operations are performed.

1. First, the positions of adjacent tiles in top-to-bottom, left-to-right order are found and placed in the set A (Lines 5–9). For the top-to-bottom and left-to-right directions, these are tiles at positions $(r_1 + 1, c_1)$ and $(r_1, c_1 + 1)$, respectively. Positions outside the dimensions of the grid (given by g_r and g_c) are ignored.
2. For each position $p_2 \in A$ of an adjacent tile, the corresponding image I_2 is then located in the grid G . The image at position p_2 is denoted by $G(p_2)$ on Line 11.
3. Finally, the transformation from I_1 to I_2 is estimated by `get_tile_transform`, a function for estimating translational and rotational transformations using feature matches produced by LoFTR [62] (Line 12). `get_tile_transform` is explained in Section 6.2 and, apart from I_1 and I_2 , requires two additional arguments: the expected overlap between neighbouring tiles $o \in (0, 1)$, and the scaling factor $s \in (0, 1)$ for feature detection and matching. These are configurable parameters of the grid stitching process as a whole.

Subsequently, Algorithm 6.1 optimises the initial pairwise transformations from T globally to minimise transformation errors across the entire grid G , producing the optimised mapping T_{opt} (Line 13). It achieves this by constructing and optimising a SLAM graph using the function `optimise_transforms`, introduced in Section 6.3.

Finally, with the help of the optimised transformations in T_{opt} , Algorithm 6.1 stitches all images in the grid G into a single composite image (Line 14). This is done by the function `stitch_grid`, described in Section 6.4, which gradually applies the transformations from T_{opt} to the corresponding images from G . The composite image is the final result of Algorithm 6.1 and is the output of the DEMIS stitching tool.

6.2 Estimating Transformations Between Pairs of Images

This section follows the explanation of the top-level stitching algorithm discussed in Section 6.1. In particular, it describes the function `get_tile_transform` (used on Line 12 of Algorithm 6.1), which estimates the translational and rotational transformation from the source image I_1 to the target image I_2 . By applying the transformation, the coordinate space of I_1 can be shifted to that of I_2 using the approach presented in Section 3.1. This is because a transformation composed of translation and rotation is a special case of homography. Translation and rotation are proposed as sufficient since, based on methods used by other EM stitching tools [12, 47] and our discussions with TESCAN 3DIM, the general perspective transformation is not necessary due to the nature and precision of EM imaging. In fact, estimating the general homography could potentially introduce undesirable perspective deformations instead.

The implementation of the function `get_tile_transform` is defined in Algorithm 6.2. Since feature detection and matching – which are the core of the transformation estimation itself – can require significant amounts of computation time and memory, the algorithm first modifies the input images according to their expected overlap ratio $o \in (0, 1)$, and the resolution scaling factor $s \in (0, 1)$. In particular, it first crops the input images according to o , producing cropped images I'_1 and I'_2 that contain the expected overlapping regions of I_1 and I_2 , respectively (Lines 1–2). Then, Algorithm 6.2 rescales the resolutions of both

Algorithm 6.2: Estimate transformation relating two overlapping image tiles

Input: Source image to I_1 and target image I_2 of the transformation
Positions p_1 and p_2 of I_1 and I_2 in the encompassing grid, respectively
Expected tile overlap ratio o
Scaling factor s for feature detection and matching

Result: Transformation from I_1 to I_2 composed of translation and rotation

`get_tile_transform(I_1, I_2, p_1, p_2, o, s):`

- `// Crop and rescale the overlapping regions`
- 1: $I'_1 := \text{crop}(I_1, o, p_1, p_2)$
- 2: $I'_2 := \text{crop}(I_2, o, p_1, p_2)$
- 3: $I'_1 := \text{resize}(I'_1, s)$
- 4: $I'_2 := \text{resize}(I'_2, s)$
- `// Compute feature matches using the scaled overlapping regions`
- 5: $M := \text{LoFTR}(I'_1, I'_2)$
- 6: correct matches in M to fit the original images
- 7: use M to estimate the translational and rotational transformation \mathbf{T}_{12}
- 8: **return** \mathbf{T}_{12}

I'_1 and I'_2 based on s (Lines 3–4). The cropping and scaling operations are denoted by the functions `crop` and `resize`, respectively. The direction from which to crop the input images can be determined from the relative positions p_1 and p_2 of I_1 and I_2 in the stitched grid. For example, let us assume that I_1 and I_2 have a resolution of $w \times h$ pixels and that I_1 is on the left of I_2 . Then, I_1 would be cropped from the right and I_2 from the left, and the final resolution of I'_1 and I'_2 would be $(w \cdot o \cdot s) \times (h \cdot s)$ pixels.

Subsequently, Algorithm 6.2 employs LoFTR [62] to compute a set of matches M between I'_1 and I'_2 (Line 5). It also corrects the positions of the detected matches to fit the original images I_1 and I_2 by reversing the cropping and resizing operations (Line 6). The feature detection and matching technique used by LoFTR is described in Section 4.1. Since the original LoFTR is trained on conventional images only, we propose to utilise a version of LoFTR that is fine-tuned on EM images. The fine-tuning process is presented in Section 6.5.

Finally, the corrected matches can be used to directly estimate the translational and rotational transformation matrix $\mathbf{T}_{12} \in \mathbb{R}^{3 \times 3}$ from I_1 to I_2 (Line 7). To handle this step, we propose to utilise the RANSAC [19]-based affine transformation estimation implemented by OpenCV (with the scaling factor removed since only translation and rotation are required). Denoting the estimated translation shift on the horizontal and vertical axes as t_x and t_y , respectively, and the estimated rotation angle by α , \mathbf{T}_{12} can be constructed as

$$\mathbf{T}_{12} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos \alpha & -\sin \alpha & 0 \\ \sin \alpha & \cos \alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \cos \alpha & -\sin \alpha & t_x \\ \sin \alpha & \cos \alpha & t_y \\ 0 & 0 & 1 \end{bmatrix}. \quad (6.1)$$

The matrix \mathbf{T}_{12} provides the initial estimate of the transformation between I_1 and I_2 and serves as the final result of Algorithm 6.2. As outlined in Section 6.1, these estimates are

Algorithm 6.3: Optimise pairwise transformations between grid tiles

Input: Mapping T of pairs of tile positions to transformations between them
Width w and height h of images in the grid corresponding to T
Expected tile overlap ratio o

Result: Mapping T_{opt} of tile positions to globally optimised transformations

`optimise_transforms(T, w, h, o):`

1: $w' = \frac{w}{2}, h' = \frac{h}{2}$

2: $o' = (1 - o)$

// Use the transformations in T to construct a SLAM graph

3: $V := \emptyset, E := \emptyset$ *// Vertices and edges of graph (V, E)*

4: **foreach** $((p_1, p_2), \mathbf{T}_{12}) \in T$ **do**

// Let $p_1 = (r_1, c_1), p_2 = (r_2, c_2)$

// Vertices represent tile position and angle (2D pose); initial angle is 0

5: $v_{p_1} := (w' + (c_1 - 1) \cdot w \cdot o', h' + (r_1 - 1) \cdot h \cdot o', 0)$

6: $v_{p_2} := (w' + (c_2 - 1) \cdot w \cdot o', h' + (r_2 - 1) \cdot h \cdot o', 0)$

7: $V := V \cup \{v_{p_1}, v_{p_2}\}$

// Edges represent changes in pose between vertices

8: transform \mathbf{T}_{12} to the equivalent 2D pose e_{12}

9: $E := E \cup \{e_{12}\}$

// Optimise the SLAM graph and compute the global transformations

10: optimise poses of vertices in (V, E)

11: $p_{ref} := (1, 1)$

// Reference position

12: $T_{opt} := \text{map}()$

13: **foreach** $v_p \in V$ **do**

14: $(t_x, t_y, \alpha)_p := v_p - v_{p_{ref}}$ *// Element-wise pose subtraction*

15: convert $(t_x, t_y, \alpha)_p$ to the equivalent transformation matrix \mathbf{T}_p

16: $T_{opt}(p) := \mathbf{T}_p$

17: **return** T_{opt}

further optimised globally by graph-based SLAM using the entire grid of stitched images. The SLAM optimisation is presented in Section 6.3.

6.3 Global Optimisation Based on SLAM

Section 6.2 introduced the algorithm for estimating translational and rotational transformations between pairs of EM images in a two-dimensional grid. In principle, these transformations could be used to stitch all images in the grid directly, e.g., by constructing a minimum spanning tree (MST), as described in Section 3.1. However, while doing so might be the most straightforward approach, grids of volume EM images may become rather large (as explained in Chapter 2). Since the errors caused by the pairwise transformations can accumulate during stitching (which could negatively impact the final image), it is crucial to minimise the global error caused by all pairwise transformations combined. Therefore, this section presents the function `optimise_transforms` (used on Line 13 of Algorithm 6.1),

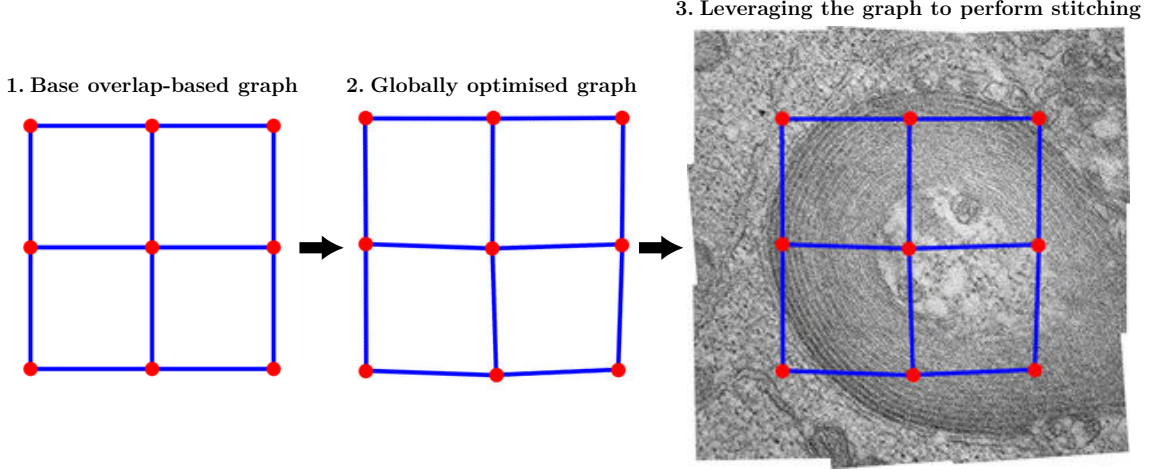


Figure 6.2: A SLAM graph corresponding to an image grid of size 3×3 . In the graph, vertices denote individual image tiles, and edges represent the estimated transformations between them. Initially, the vertices are placed at positions determined based on tile resolution and the expected overlaps. This is indicated by the regular shape of the base graph (left). Afterwards, the vertex poses are optimised globally according to the estimated pairwise tile transformations. The image grid corresponding to the transformations is shown on the right. The image tiles were sourced from [4].

which optimises the pairwise transformations using graph-based SLAM. The transformations are represented by the mapping T of pairs of grid positions to the corresponding transformations. Additionally, the function expects to receive the width w , height h , and expected overlap ratio o of image tiles in the stitched grid. These variables are introduced in Section 6.1. The function is described by Algorithm 6.3 and the SLAM optimisation is illustrated in Figure 6.2.

Algorithm 6.3 starts by constructing a SLAM graph (V, E) that mirrors the expected structure of the entire stitched grid (Lines 3–9). V and E denote the sets of vertices and edges in the graph, respectively. Both V and E are built gradually by iterating over all pairs of grid positions (p_1, p_2) and the corresponding transformation matrices \mathbf{T}_{12} in T . In each iteration, the following steps are conducted.

1. Vertices v_{p_1} and v_{p_2} , which represent the image tiles at positions $p_1 = (r_1, c_1)$ and $p_2 = (r_2, c_2)$, respectively, are constructed and added to V (Lines 5–7). Each vertex is characterised by its *2D pose* – the estimated 2D position of the centre pixel and the rotation angle of the corresponding image tile. For example, the vertex $v = (100, 200, 3)$ describes an image tile that (1) has its centre located at 100 pixels horizontally and 200 pixels vertically, and (2) is rotated 3 degrees to the right along its centre. Initially, the centre positions for v_{p_1} and v_{p_2} are determined based on the tile resolution ($w \times h$ pixels) and the expected overlap ratio o . The angles are assumed to be 0. In essence, the initial poses correspond to the ideal case scenario, in which no misalignments between tiles exist and the expected overlaps are precise. Formally, v_{p_1} and v_{p_2} can be constructed as

$$v_{p_1} = (w' + (c_1 - 1) \cdot w \cdot o', h' + (r_1 - 1) \cdot h \cdot o', 0), \quad (6.2a)$$

$$v_{p_2} = (w' + (c_2 - 1) \cdot w \cdot o', h' + (r_2 - 1) \cdot h \cdot o', 0), \quad (6.2b)$$

where $w' = \frac{w}{2}$, $h' = \frac{h}{2}$, and $o' = (1 - o)$.

2. The edge e_{12} , connecting v_{p_1} to v_{p_2} , is created from the transformation matrix \mathbf{T}_{12} and added to E (Lines 8–9). The edge describes the estimated change in pose between v_{p_1} and v_{p_2} . Hence, it can be constructed by extracting from \mathbf{T}_{12} the horizontal and vertical translation shifts t_x and t_y , respectively, and the rotation angle α (introduced in Section 6.2). In particular, e_{12} is defined by the tuple $(t_x, t_y, \alpha)^2$.

Then, after the graph (V, E) is constructed, the initial vertex poses are optimised with respect to the pose change estimates of the edges connecting them (Line 10). For optimisation, we propose to use the `graphslam`³ library. During the optimisation, the vertex corresponding to the tile at position $p_{ref} = (1, 1)$, that is, the reference tile, has its pose parameters locked to the initial values from Equation 6.2. This ensures that the final stitched image has a predictable structure (with the reference tile being parallel to both the horizontal and vertical axes). Further details of the optimisation process used by `graphslam` are beyond the scope of this thesis and are thus omitted for brevity. More details can be found, for example, in [21]. As can be seen in Figure 6.2, after the optimisation, the graph loses its idealistic regular structure (left). Instead, it begins to conform to the actual estimated structure of the stitched grid, which is generally slightly deformed (middle).

Subsequently, the mapping T_{opt} of the grid positions to optimised transformations is generated from the optimised vertex poses (Lines 11–16). It should be noted that this mapping has a different structure compared to the original mapping T since it is indexed by individual tile positions, not pairs of tile positions. This is because it describes the global transformations needed to directly convert the coordinate spaces of tiles in the stitched grid to the coordinate space of the reference tile at position p_{ref} (as opposed to T , which instead contains the local transformations between adjacent tiles). T_{opt} , which is initially empty, is constructed by iterating over each vertex v_p (representing the tile at position p) in V and performing the following steps.

1. The element-wise difference between the poses of v_p and $v_{p_{ref}}$ is calculated (Line 14). This yields the pose change estimate $(t_x, t_y, \alpha)_p = v_p - v_{p_{ref}}$, which defines the translation shifts t_x and t_y and the rotation angle α necessary to transform the coordinate space of the tile at position p to that of the reference tile.
2. $(t_x, t_y, \alpha)_p$ is then converted to the corresponding transformation matrix \mathbf{T}_p using Equation 6.1 (Line 14).
3. \mathbf{T}_p is saved to T_{opt} at position p , denoted as $T_{opt}(p)$ on Line 16.

Finally, the mapping T_{opt} is returned as the final result of Algorithm 6.3. Afterwards, the optimised transformations in T_{opt} can be used to stitch all images in the grid into one, as explained in Section 6.4.

²In addition to the estimated change in 2D pose, Gaussian prior distributions are associated with the edge parameters. The variances of these distributions are configurable hyperparameters and should be selected based on the quality and resolution of the stitched images. For the DEMIS dataset, the recommended variances are 4 pixels for t_x and t_y , and 1 degree for α .

³`graphslam` – <https://github.com/JeffLrion/python-graphslam/>.

Algorithm 6.4: Stitch images in a grid into a single composite image

Input: Mapping G of 2D grid positions to stitched EM image tiles forming the grid
Mapping T_{opt} of tile positions to globally optimised transformations

Result: Image I_G composed of all images in G stitched together

```
stitch_grid( $G, T_{opt}$ ):  
    // Create a blank target image  
1:  $C := \emptyset$   
2: foreach  $(p, \mathbf{T}_p) \in T_{opt}$  do  
    // Let  $C_p$  be the set of all corners of image  $G(p)$   
3:     transform all corners in  $C_p$  using  $\mathbf{T}_p$   
4:      $C := C \cup C_p$   
5: create a blank image  $I_G$  large enough to fit all corners in  $C$   
    // Stitch the tiles together  
6: foreach  $(p, \mathbf{T}_p) \in T_{opt}$  do  
7:      $I_p := G(p)$   
8:     transform  $I_p$  using  $\mathbf{T}_p$   
9:     draw  $I_p$  on  $I_G$   
10: return  $I_G$ 
```

6.4 Constructing the Final Composite Image

This section completes the description of the pipeline from Figure 6.1. It presents the function `stitch_grid` from Line 14 of Algorithm 6.1, which constructs the final stitched image. For that, the function requires the mappings G and T_{opt} of grid positions to images in the stitched grid and their globally optimised transformations, respectively. The function is defined by Algorithm 6.4, which is described below.

First, Algorithm 6.4 generates a blank image I_G that is large enough to fit all the images in G after the transformations from T_{opt} are applied (Lines 1–5). It starts by transforming the corners of each image tile. In particular, it builds the set C of warped corner pixels of all tiles. The set is initially empty. To fill the set, Algorithm 6.4 iterates over the tile positions p and the corresponding transformation matrices \mathbf{T}_p in T_{opt} . In each iteration, it creates the set C_p of all four corners of the image tile $G(p)$ at position p and transforms the coordinates of the corners in C_p using \mathbf{T}_p (Line 3). The transformed corners are then placed in C (Line 4). With the complete set of transformed corners, the required size of I_G can be determined by searching for the lowest and highest coordinate values in both the horizontal and vertical directions (Line 5).

Afterwards, the initially blank target image I_G is gradually filled with the stitched grid (Lines 6–9). This is done by once again going over all the positions p and the corresponding transformation matrices \mathbf{T}_p in T_{opt} . However, this time, the transformation matrix \mathbf{T}_p is applied to the entire image $I_p = G(p)$ at the grid position p , not just to its corners (Lines 7–8). Then—that is, when the coordinate space of I_p is converted to that of the reference tile, as described in Section 6.3—the transformed image I_p is drawn on I_G (Line 9). When drawing I_p on I_G , any pixels already filled by previously drawn image tiles are simply

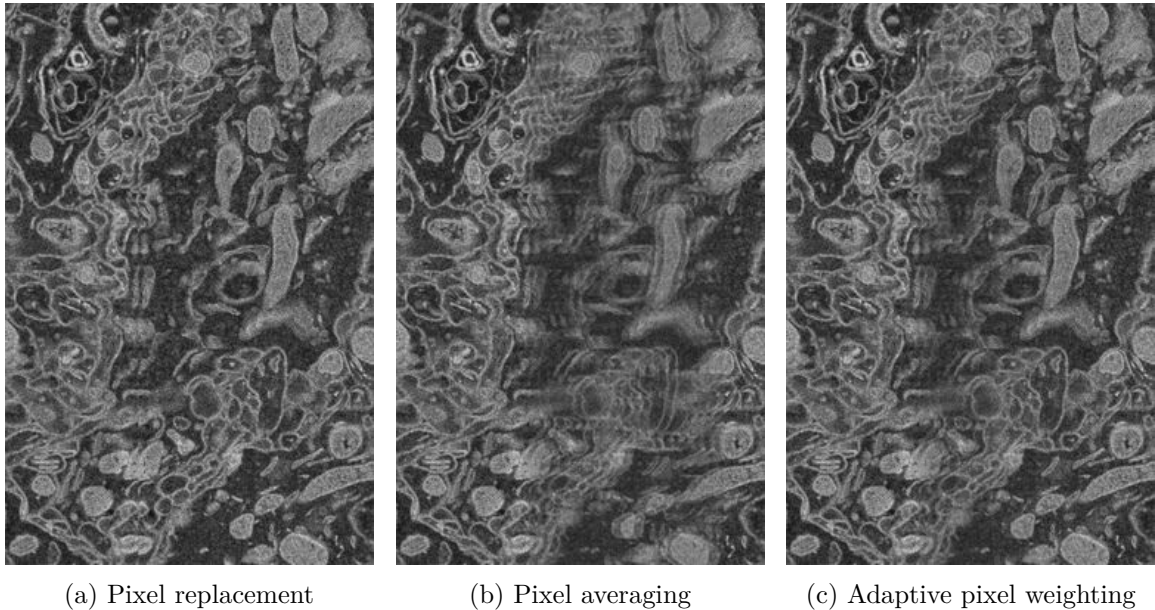


Figure 6.3: Comparison of different methods for processing pixels from overlapping regions of adjacent tiles. The methods are described in Section 3.1. Noticeable blurring can be seen when pixel averaging or adaptive pixel weighting is used. Hence, pixel replacement, where pixels are selected from one of the overlapping tiles with priority, is proposed for EM images. The source images were provided by TESCAN 3DIM.

replaced. This is because methods based on pixel weighting can cause significant blurring due to the structure of EM images, as illustrated in Figure 6.3.

Finally, the complete image I_G is returned (Line 10). This image, which contains the entire stitched grid, is the result of the stitching process proposed by the DEMIS tool.

6.5 Fine-Tuning LoFTR on Electron Microscopy Images

The previous sections of this chapter described the stitching technique used by the proposed DEMIS tool. As explained in Section 6.2, the core of the solution is the deep learning-based feature detection and matching performed by LoFTR [62], introduced in Section 4.1. However, the original LoFTR is trained only on conventional images (either indoor or outdoor photographs). Although it is possible to use the original model directly on EM images due to its high robustness, the performance might be limited as a result of differences in the type of processed data. Therefore, in an attempt to improve feature-matching accuracy, this section proposes to fine-tune LoFTR on EM images. Specifically, it proposes to leverage the training and validation images from the DEMIS dataset, presented in Chapter 5. The rest of this section describes the process of converting the labels of images from the DEMIS dataset to a format that is expected by LoFTR.

As outlined in Section 4.1, LoFTR requires the relative camera poses between pairs of overlapping images—defined by the intrinsic and extrinsic camera matrices⁴—and the

⁴The intrinsic matrix describes the transformation from the 3D camera space to the space of 2D pixel coordinates, while the extrinsic matrix defines the position and orientation of the camera in the original 3D space [64].

corresponding depth maps to construct its internal ground truth labels. This approach is not directly compatible with the DEMIS dataset, which contains only the positions and rotation angles of the tiles generated from each source EM image. Therefore, the following steps are performed to obtain the labels expected by LoFTR.

First, since no depth information is provided for the images in the DEMIS dataset, the training depth maps are filled with a constant value. In other words, all segments of the images are assumed to be imaged at exactly the same depth. This approach is acceptable because of the inherent nature of electron microscopy imaging.

Second, the extrinsic matrices are created based on differences in translation and rotation between all pairs of adjacent tiles in the DEMIS dataset. Since, as presented in Section 5.2, the first image tile in each grid has no rotation applied to it, it can be used as a reference point for all other tiles. In particular, let us assume that the first image tile in a grid, denoted as I_0 , is translated by t_{x_0} pixels horizontally and t_{y_0} vertically. Furthermore, let us assume that the tile I_1 from the same grid is translated similarly by t_{x_1} and t_{y_1} pixels, respectively, and rotated by α_1 degrees along its centre. Let us also denote the width and height of the tiles as w and h , respectively. Then, the transformation matrix \mathbf{T}_{10} from the coordinate space of I_1 to the coordinate space of I_0 can be calculated as

$$\mathbf{T}_{10} = \begin{bmatrix} a & b & (1-a) \cdot c_x - b \cdot c_y \\ -b & a & b \cdot c_x + (1-a) \cdot c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & \Delta x \\ 0 & 1 & \Delta y \\ 0 & 0 & 1 \end{bmatrix}, \quad (6.3)$$

where $\Delta x = t_{x_1} - t_{x_0}$, $\Delta y = t_{y_1} - t_{y_0}$, $a = \cos(-\alpha_1)$, $b = \sin(-\alpha_1)$, $c_x = \frac{w}{2} + \Delta x$, and $c_y = \frac{h}{2} + \Delta y$ [51]. In other words, I_1 is first translated by Δx and Δy to its position relative to I_0 . Afterwards, it is rotated by $-\alpha_1$ degrees along its centre (c_x, c_y) at that position.

Subsequently, the transformation matrix \mathbf{T}_{12} from I_1 to the image tile I_2 from the same grid can be created as follows. First, the transformation matrices \mathbf{T}_{10} and \mathbf{T}_{20} to the reference tile are computed for both I_1 and I_2 , respectively (using Equation 6.3). Then, \mathbf{T}_{12} is obtained by applying \mathbf{T}_{10} followed by the inverse of \mathbf{T}_{20} : $\mathbf{T}_{12} = \mathbf{T}_{20}^{-1} \mathbf{T}_{10}$. The desired extrinsic matrix \mathbf{E}_{12} relating I_1 to I_2 can be created by extending \mathbf{T}_{12} by the depth dimension. In particular, if we denote the elements of \mathbf{T}_{12} as

$$\mathbf{T}_{12} = \begin{bmatrix} t_{11} & t_{12} & t_{13} \\ t_{21} & t_{22} & t_{23} \\ 0 & 0 & 1 \end{bmatrix}, \quad (6.4)$$

then \mathbf{E}_{12} can be obtained as

$$\mathbf{E}_{12} = \begin{bmatrix} t_{11} & t_{12} & 0 & t_{13} \\ t_{21} & t_{22} & 0 & t_{23} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \quad (6.5)$$

Finally, since the transformations between pairs of overlapping image tiles can be inferred directly from the constructed extrinsic matrices, the intrinsic matrices are set to the identity matrix.

In summary, the translational and rotational parameters inside the DEMIS dataset are used to compute suitable extrinsic matrices that define the exact relationship between pairs of overlapping image tiles. The intrinsic camera parameters and the depth maps are kept in their most elementary forms because the specifics of capturing the source images of the DEMIS dataset are unknown. Nevertheless, the extrinsic matrices are sufficient to train LoFTR on pairs of overlapping image tiles from the DEMIS dataset.

Chapter 7

Implementation

This chapter describes the most important implementation details of the DEMIS electron microscopy (EM) stitching tool, proposed in Chapter 6. The DEMIS tool is written entirely in the Python programming language. As such, it requires no compilation to run. However, it does require access to a CUDA-enabled GPU and packages from its specific Anaconda¹ environment, the most important being the PyTorch framework², which is used as the backbone of the LoFTR [62] module. With the environment, the DEMIS tool can be utilised to stitch grids of overlapping EM images, as outlined in Chapter 6. The specification of the environment is included in the implementation. The complete source code is publicly available through the DEMIS GitHub repository³ under the standard MIT licence.

The rest of this chapter is organised as follows. First, Section 7.1 presents the essential components of the DEMIS tool, which directly implement the algorithms described in Chapter 6. Then, Section 7.2 introduces the remaining modules of the DEMIS tool, which generally provide more supportive functions, such as generating the DEMIS dataset proposed in Chapter 5.

7.1 Primary Modules of the DEMIS Tool

This section describes the *primary modules* of the DEMIS tool, which are implemented based on the algorithms proposed in Chapter 6. The descriptions (provided below) are directly inspired by the source files of the DEMIS tool, which are available publicly in its GitHub repository. The primary modules and the dependencies between them are shown in Figure 7.1.

Config Contains the default configuration of the DEMIS tool. This includes parameters such as the path to the processed dataset, expected tile overlaps, and the resolution scaling factor for feature detection and matching. Additionally, the methods used by the DEMIS tool can also be configured. For example, it is possible to select image compositing based on adaptive pixel weighting instead of the default pixel replacement recommended in Section 6.4. The default settings can be overridden by configuration files written in the YAML language.

¹**Anaconda** – a Python and R distribution tailored primarily towards data science and machine learning – <https://www.anaconda.com/>.

²**PyTorch** – an open source framework for tensor operations and deep learning – <https://pytorch.org/>.

³**DEMIS GitHub repository** – <https://github.com/PSilling/demis>.

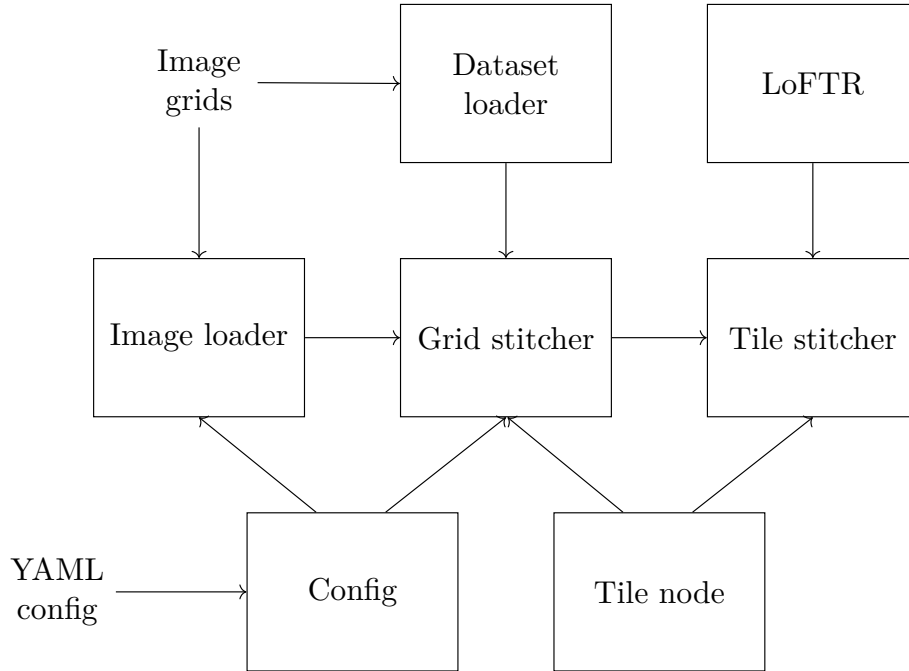


Figure 7.1: The primary modules of the DEMIS tool. The primary modules implement the EM image stitching algorithms proposed in Chapter 6.

Dataset loader Loads metadata about an input dataset, which should contain an arbitrary number of grids of overlapping EM images. The metadata are composed of the expected grid size and the paths to image tiles in each grid in the dataset. The expected grid size is derived from the name of the directory that contains the target dataset. The name of the containing directory should be formatted as `<rows>x<cols>`, where `<rows>` and `<cols>` represent the numbers of rows and columns, respectively. The image paths (and their association with the corresponding image grids) are parsed from image tile filenames. The filenames should be formatted as `<name>_g<grid_idx>_t<tile_idx>_s<slice_idx>`, where `<name>` is the name of the dataset, `<grid_idx>` is the index of the grid containing the image, `<tile_idx>` is the index of the image tile within the grid, and `<slice_idx>` is the index of the corresponding slice (i.e., of the imaging depth). The formats were inferred from the standard EM dataset structure used by TESCAN 3DIM.

Image loader Reads EM images based on the paths provided by the Dataset loader module. The input images are expected to be in the standard TIFF file format. Additionally, normalises the contrast and brightness of the loaded images using contrast-limited adaptive histogram equalisation (CLAHE) [53] provided by OpenCV.

LoFTR As described in Chapter 6, the DEMIS tool relies on LoFTR [62] to perform feature detection and matching between adjacent image tiles. Hence, LoFTR (using its default architecture) is included as a module of the DEMIS tool. However, compared to the original implementation of LoFTR, the LoFTR module used by the DEMIS tool contains additional data loading and data manipulation classes that allow training on the DEMIS dataset.

Grid stitcher The main component of the DEMIS tool. Controls the entire grid stitching process. In other words, it implements the top-level grid stitching algorithm presented in Section 6.1. In particular, it receives a grid of overlapping EM images that should be stitched into a single composite image. Then, it goes over the stitched images and estimates transformations between adjacent image tiles with the help of the Tile stitcher module. Subsequently, it builds a SLAM [21] graph to obtain globally optimised transformations. Finally, it stitches all tiles in the grid using the Tile node module.

Tile stitcher Performs pairwise feature detection and matching. In particular, it expects to receive a pair of adjacent image tiles, and (after cropping and resizing, as described in Section 6.2) detects feature matches between them using the LoFTR module. Moreover, the Tile stitcher module handles the drawing of individual tiles in the grid on the final composite image (using pixel replacement by default).

Tile node Represents a tree node that contains a single image tile and its estimated transformation. If connected to a parent node, the transformation from the parent node is automatically applied as well. This enables the DEMIS tool to represent grids of stitched images using a tree, which can then guide the stitching process (i.e., select which tile should be stitched in the next step). To implement the grid stitching presented in Algorithm 6.4, only a simple tree needs to be constructed, in which a single reference tile node serves as the root, and the nodes of other tiles are connected directly to the root. However, it is possible to use the Tile node module for other stitching methods, such as those that rely on building a minimum spanning tree (as discussed in Section 3.1).

Together, the primary modules of the DEMIS tool can be used to stitch grids of EM images based on the methods described in Chapter 6. However, since the modules function, in essence, as a standalone library, additional supportive modules are required for ease of use (as well as to handle the DEMIS dataset, proposed in Chapter 5). These *secondary modules* are introduced in Section 7.2.

7.2 Secondary Modules of the DEMIS Tool

This section follows the description of the primary modules of the DEMIS tool in Section 7.1 by providing an overview of its secondary modules, such as the modules for evaluation or generating the DEMIS dataset proposed in Chapter 5. The secondary modules are introduced below. The descriptions are once again directly inspired by the source files of the DEMIS tool.

DEMIS loader A specialised variant of the Dataset loader module, which is adjusted to work well with the DEMIS dataset. In addition to providing the same functionality as the standard Dataset loader module, the DEMIS loader module is also able to parse the labels of images from the DEMIS dataset.

DEMIS stitcher A specialised variant of the Grid stitcher module. Similarly to the DEMIS loader module, the DEMIS stitcher module is also adapted for use with the DEMIS dataset. Compared to the default Grid stitcher module, the DEMIS stitcher module has the additional capability to stitch grids of images from the DEMIS dataset according to their

ground truth labels. To perform label-based stitching, the module employs the principles presented in Section 6.5 to construct ground truth transformations between pairs of tiles in the stitched grids.

Stitch Provides a command-line interface (CLI) for interacting with the DEMIS tool. In particular, it handles configuration loading and the execution of the Grid stitcher module for each grid in the configured input dataset. Additionally, after each input grid is stitched, it saves the resulting image locally in the TIFF format. The arguments available through the CLI are listed below.

- `cfg_path` – Path to the YAML configuration file that specifies method parameters and data paths.
- `-d, --use-demis-labels` – Flag indicating that if the input dataset is the DEMIS dataset, it should be stitched using its labels.
- `-g <grid_idx>, --grid-index <grid_idx>` – Specifies that only grids with the index `<grid_idx>` should be stitched. Can be applied multiple times to select multiple grid indices.
- `-s <slice_idx>, --slice-index <slice_idx>` – Signals that only image tiles with their slice index equal to `<slice_idx>` should be stitched. Can be applied multiple times to select multiple slice indices.

DEMIS synthesiser Synthesises the DEMIS dataset from its source images using the method described in Section 5.2. In particular, it splits each source image into overlapping tiles 1024×1024 pixels in size. Then, it applies random translation and rotation, random brightness and contrast adjustments, and a limited amount of Gaussian noise to each generated tile. The images are saved in the TIFF format and named according to the filename format expected by the Dataset loader module.

DEMIS splits generator Divides the DEMIS dataset (generated by the DEMIS synthesiser module) into training, validation, and testing splits, as explained in Section 5.2. When creating the splits, the module also generates the corresponding LoFTR training labels using the technique presented in Section 6.5.

DEMIS evaluator Provides evaluation methods for the DEMIS tool. When evaluating, the default DEMIS tool is compared against its variant in which LoFTR is replaced by the traditional feature detection and matching based on SIFT [41] (introduced in Section 3.1). Further details on experimental evaluation are provided in Chapter 8.

In summary, the secondary modules extend the DEMIS tool with a command-line interface, evaluation methods, and modules for creating and manipulating the DEMIS dataset. Combined with the primary modules from Section 7.1, they create a complete set of tools prepared for stitching grids of EM images.

Chapter 8

Experimental Evaluation

The previous chapters described the fundamental aspects and components of the proposed DEMIS tool for stitching grids of electron microscopy (EM) images. This chapter follows by presenting a series of experiments, which were conducted to assess the quality of the images produced by the DEMIS tool. The experiments are summarised below.

- Experiments that evaluate the accuracy of feature detection, feature matching, and transformation estimation. These experiments are described in Section 8.1.
- Experiments quantifying the quality of stitched images using common image quality estimation metrics, such as PSNR. These are discussed in Section 8.2.
- Experiments focusing on the impact of deep learning (represented in the DEMIS tool by the LoFTR [62] module) on stitching robustness. The robustness analysis is presented in Section 8.3.

In all of the above experiments, feature detection and matching were performed at half the original image resolution, and the following three stitching solutions were compared against each other.

- *Baseline*—a modified variant of the DEMIS tool, in which the LoFTR module was replaced by a module that performs feature detection and matching using SIFT [41]. Serves as a representative of the traditional stitching pipeline presented in Section 3.1, which is the core of several current state-of-the-art EM stitching solutions [10, 43]. The SIFT processing module was implemented using the OpenCV library.
- *Pre-trained*—a variant of the DEMIS tool that uses the original LoFTR model (trained on conventional images only) for feature detection and matching.
- *Fine-tuned*—the DEMIS tool proposed in Chapter 6 (including the fine-tuning of LoFTR on the DEMIS dataset). The fine-tuning was carried out on a machine with eight 16 core, 2.30 GHz Intel Xeon Gold 5218 CPUs, two NVIDIA Tesla T4 GPUs with 16 GB of memory each, and 32 GB of RAM. The computational resources were provided by the e-INFRA CZ project (ID:90140), supported by the Ministry of Education, Youth and Sports of the Czech Republic. The training configuration was based on the original training configurations of LoFTR and is included in the official implementation of the DEMIS tool, described in Chapter 7.

Table 8.1: Comparison of feature processing and transformation estimation performance. The best results are highlighted in bold. The SIFT baseline achieved the fastest speed of feature matching, although the differences are negligible. The fine-tuned DEMIS tool demonstrated the best accuracy.

Metric	Baseline	Pre-trained	Fine-tuned
Feature matching time	80.44 ms	83.52 ms	83.18 ms
Feature matches found	694	770	778
Outlier feature matches	5	3	2
Inlier reprojection error	2.86 px	2.90 px	2.82 px
Corner error AUC at 3 px	7.22 %	6.20 %	11.51 %
Corner error AUC at 5 px	42.70 %	40.99 %	46.02 %
Corner error AUC at 10 px	71.35 %	70.48 %	73.01 %

The experiments can be reproduced using the DEMIS evaluator module (introduced in Section 7.2), which is available on the attached medium as part of the DEMIS tool. The contents of the attached medium are described in Appendix A.

8.1 Evaluating Feature and Transformation Processing

The feature detection, feature matching, and transformation estimation of the DEMIS tool were evaluated experimentally on the 1306 evaluation images of the DEMIS dataset. A machine with a 6 core, 2.80 GHz Intel Core i5-8400 CPU, an NVIDIA GeForce GTX 1070 GPU with 8 GB of memory, and 16 GB of RAM was used for the experiments.

The experiments were carried out on pairs of adjacent image tiles in the evaluation split of the DEMIS dataset. In particular, feature matches were detected between each pair of images. Then, the transformations that relate the images in each pair together were estimated. Using the feature matches and the estimated transformations, the following metrics were calculated.

- *Feature matching time* – the mean amount of time needed to compute feature matches between a pair of images. Measured as the average of five independent runs.
- *Feature matches found* – the mean number of unfiltered feature matches detected between a pair of images.
- *Outlier feature matches* – the mean number of feature matches between a pair of images that were identified as outliers by RANSAC [19].
- *Inlier reprojection error* – the mean distance between a pair of matching features after being projected to the same coordinate space by the corresponding ground truth transformation (calculated using DEMIS dataset labels). Only feature matches considered as inliers by RANSAC were evaluated.
- *Corner error AUC* – the area under the cumulative error curve (AUC) of the reprojection error of image tile corners at specified pixel (px) thresholds, as presented in [56]. Displayed as a percentage. Can be understood as the probability that a random image tile corner will be within the specified threshold distance of its ground truth position after getting transformed by the estimated transformation.

Table 8.2: Comparison of pairwise image stitching quality. The best results are displayed in bold. The fine-tuned DEMIS tool achieved the best image quality by a small margin.

Metric	Baseline	Pre-trained	Fine-tuned
PSNR	22.07	21.88	22.33
SSIM	0.68	0.67	0.69
FSIM	0.95	0.94	0.95
BRISQUE	32.07	32.09	32.09

Table 8.3: Comparison of grid stitching quality. The best results are highlighted in bold. The SIFT baseline and the pre-trained DEMIS tool both demonstrated similar performance, which was slightly superior to that of the fine-tuned DEMIS tool.

Metric	Baseline	Pre-trained	Fine-tuned
PSNR	15.22	15.23	15.04
SSIM	0.22	0.22	0.21
FSIM	0.87	0.87	0.86
BRISQUE	33.64	33.60	33.59

The results, displayed in Table 8.1, show that the SIFT baseline runs the fastest of the three stitching solutions. However, the differences in the speed of feature matching are relatively small. Therefore, they can be considered negligible and should not lead to any issues in practice.

Furthermore, the results reveal that the fine-tuned DEMIS tool finds, on average, a little over 12 % more feature matches than the SIFT baseline. The increase in match count suggests a greater overall robustness of the solution. The feature matches were also more accurate in terms of mean reprojection errors and corner error AUCs. This is especially apparent in the corner error AUC at the lowest threshold of 3 pixels. A decrease in the number of outliers can be observed as well, which further supports the claim that the fine-tuned DEMIS tool can, on average, generate feature matches of higher quality than the SIFT baseline.

Finally, the results suggest that the fine-tuning of LoFTR had a positive impact on the overall accuracy. This is because the pre-trained DEMIS tool demonstrated an overall performance that is slightly inferior even to that of the SIFT baseline. Combined with the findings presented above, it can be concluded that by properly training a deep learning model, feature detection and matching accuracy (and consequently, the quality of estimated transformations) can be improved compared to traditional approaches.

8.2 Evaluation of Image Stitching Quality

Although the metrics introduced in Section 8.1 are generally rather reliable due to their focus on fully quantifiable aspects of the stitching process (e.g., errors associated with feature matches), they do not directly assess the quality of the images stitched by the DEMIS tool. Therefore, it is important to also compare the produced images to the expected results stitched using ground truth transformations. This was done in the following set of experiments, which attempted to evaluate the quality of (1) stitched pairs of adjacent tiles,

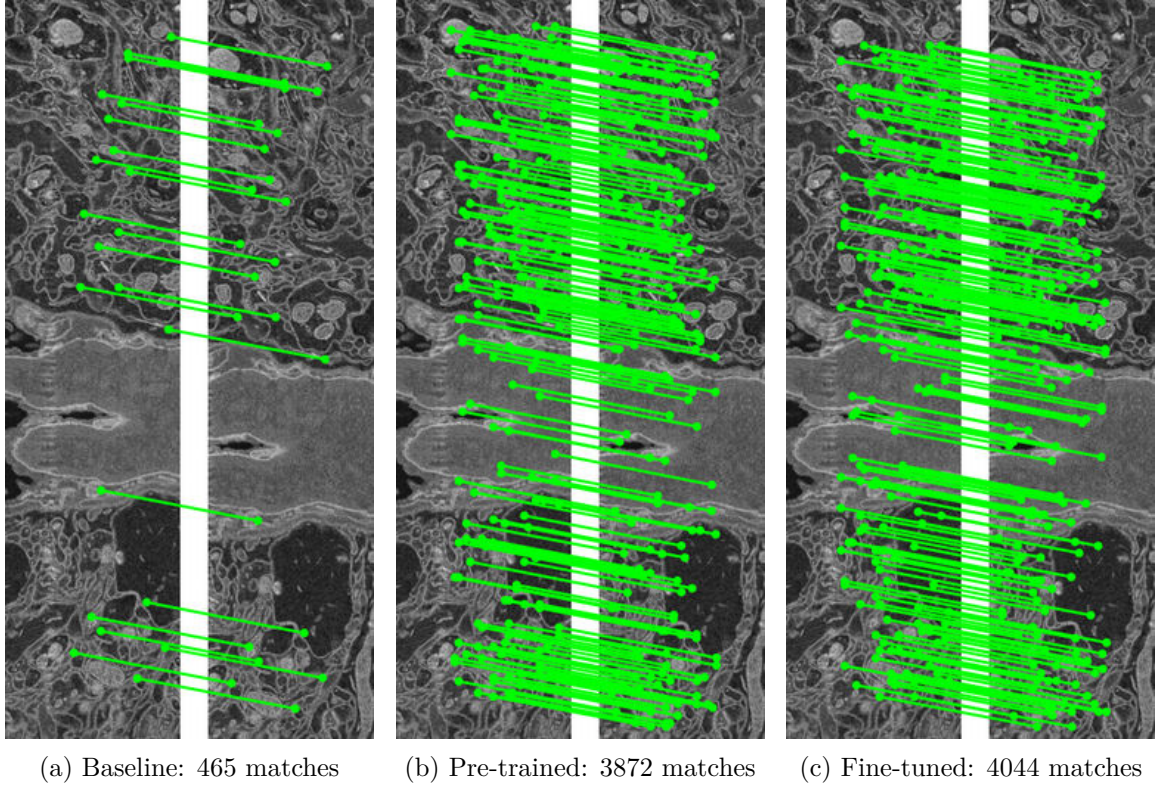


Figure 8.1: Comparison of feature matching results on an image pair from a relatively challenging dataset with tiles of size 4096×3072 pixels and a 15% tile overlap. The SIFT baseline is able to generate a sufficient amount of feature matches. However, the amount is much lower compared to the other solutions. The fine-tuned DEMIS tool achieves the best performance by a small margin. For clarity, only 5% of the detected matches are displayed. Outlying matches were removed by RANSAC. The dataset images were provided by TESCAN 3DIM.

and (2) entire stitched image grids. To evaluate the quality, three *full-reference*¹ metrics – PSNR, SSIM [75], and FSIM [73] – and a single *no-reference* metric, BRISQUE [46], were used. Higher values measured by these metrics correspond to higher perceived image quality. Similarly to Section 4.1, the evaluation images of the DEMIS dataset were selected for the experiments.

The results for pairwise stitching and complete grid stitching are shown in Tables 8.2 and 8.3, respectively. The results show that all three evaluated solutions achieve similar image quality for both pairwise and grid stitching. In particular, the fine-tuned DEMIS tool achieves slightly better image quality on pairwise stitching compared to the other two solutions. However, it also demonstrates the worst performance of the three solutions on complete grid stitching. This result is surprising, since it does not correspond to any of the previously observed behaviour. One possible cause could be brightness and contrast differences between image tiles, which might affect grid stitching results in possibly undesirable ways. Further work could focus on validating the outcome.

¹In contrast to no-reference metrics, full-reference metrics require information about a reference (ground truth) image [46].

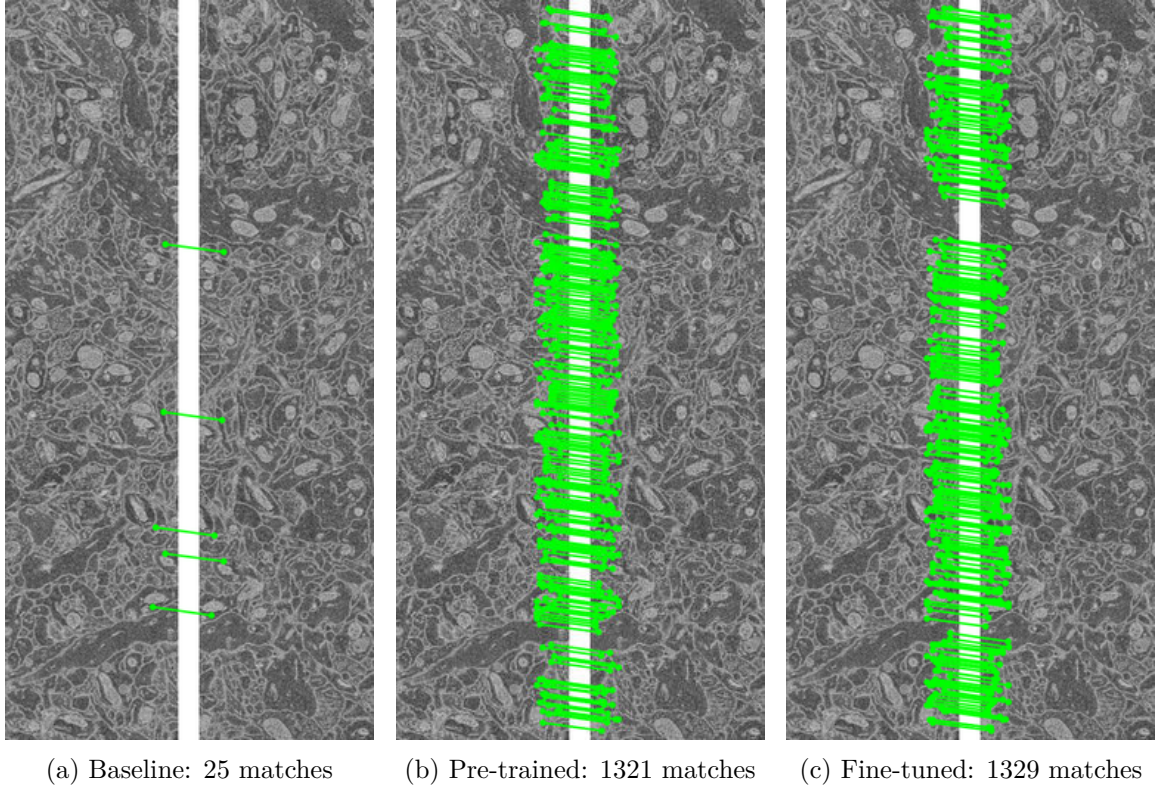


Figure 8.2: Comparison of feature matching results on an image pair from a highly challenging dataset with tiles of size 4096×4096 pixels and only a 5% tile overlap. The SIFT baseline nearly fails to detect a sufficient number of feature matches for the following transformation estimation. The fine-tuned DEMIS tool demonstrates performance that is similar to the pre-trained DEMIS tool, but significantly superior to that of the SIFT baseline. Only 20% of all matches are shown to increase visual clarity. The matches were filtered by RANSAC. The dataset images were provided by TESCAN 3DIM.

Despite the differences in results described above, the performance of all three solutions is extremely close to each other in both pairwise and grid stitching. Therefore, it is safe to assume that the increase in the accuracy of feature detection, feature matching, and transformation estimation discussed in Section 4.1 does not come at the cost of any substantial decrease in overall stitching quality.

8.3 Analysing Stitching Robustness

A common advantage of deep learning methods is their higher robustness compared to traditional approaches. For that reason, this section presents a series of experiments that analyse the impact of LoFTR-based feature detection and matching on image stitching robustness. In contrast to the previous experiments, discussed in Sections 8.1 and 8.2, the robustness experiments were conducted on two challenging EM datasets with high resolution and small tile overlaps. The datasets were provided by TESCAN 3DIM (for evaluation purposes only). The goal of the experiments was to compare the number of detected feature matches between the three evaluated stitching solutions. The resulting

matches (after RANSAC [19]) are displayed in Figures 8.1 and 8.2 on sample images that are representative of the overall performance of the evaluated solutions on the two datasets.

The result in Figure 8.1 corresponds to a dataset with high tile resolution and a 15% overlap between adjacent image tiles. This makes the dataset quite challenging for traditional approaches, as evidenced by the fine-tuned DEMIS tool finding over 8 times more matches than the SIFT baseline on the sample image pair from Figure 8.1. A similar (although slightly inferior) outcome was achieved by the pre-trained DEMIS tool. Nevertheless, despite the reduced number of feature matches, the SIFT baseline was able to detect a sufficient number of feature matches to perform the subsequent stitching accurately.

However, the results changed considerably on the second dataset, which featured an even higher tile resolution and only a 5% overlap between adjacent image tiles. The results on a sample image pair from the second dataset can be seen in Figure 8.2. In particular, it can be observed that the SIFT baseline detected a very small amount of feature matches (over 50 times less than the fine-tuned DEMIS tool), which may not be sufficient to create a robust estimate of the transformation that relates the two adjacent image tiles together. In fact, for some pairs of adjacent tiles from the dataset, the SIFT baseline was unable to produce any valid matches. This was not the case for the other two evaluated solutions, which displayed sufficient performance on all pairs of image tiles.

In summary, the above results demonstrate that the DEMIS tools have much higher robustness compared to the SIFT baseline on real-life datasets. This is especially apparent as the resolution increases and the overlap between adjacent image tiles decreases. In these scenarios, the SIFT baseline may even completely fail to generate an adequate number of feature matches.

Chapter 9

Conclusion

This thesis presented the problem of stitching grids of overlapping electron microscopy (EM) images into larger composite images. Furthermore, it described the issues of traditional stitching approaches, which may fail on images with low-quality texture, small image tile overlaps, or high resolution, all of which are common in EM imagery.

To address the issues of traditional methods, the thesis proposed the DEMIS tool, a novel EM image stitching tool based on SLAM-based global optimisation and LoFTR, an attentional neural network for detecting feature matches between pairs of overlapping images. Moreover, the thesis proposed the DEMIS dataset, a novel synthetic dataset generated by splitting high-quality and high-resolution EM images into grids of overlapping image tiles. Finally, the thesis proposed to use the DEMIS dataset to fine-tune LoFTR in an attempt to increase the accuracy of feature matches detected between EM images.

The proposed DEMIS tool was evaluated experimentally on the evaluation images of the DEMIS dataset and two challenging datasets provided by TESCAN 3DIM. The results showed that the fine-tuning of LoFTR helped the DEMIS tool achieve better performance in feature detection, feature matching, and transformation estimation when compared to a baseline solution based on SIFT. Compared to the SIFT baseline, the results also revealed significantly higher robustness of feature matching on EM images with high resolution and small overlaps. Finally, the results demonstrated that the increase in accuracy and robustness does not come with any significant decrease in the overall quality of the images produced by the tool.

Despite the promising results presented above, future work could improve the DEMIS tool and the DEMIS dataset in several ways. For instance, smaller-sized architectures of the LoFTR module could be examined, since the original architecture may be unnecessarily large for EM image stitching. Moreover, support for more input dataset formats and image file types could be added to the DEMIS tool. Furthermore, memory requirements of the tool could be lowered, for example, by constructing the final stitched image in chunks instead of all at once. Finally, the DEMIS dataset could be increased in size and enhanced by additional data augmentation (e.g., radial distortions) to enable more robust evaluation and fine-tuning of the DEMIS tool.

Bibliography

- [1] ADEL, E., ELMOGY, M. and ELBAKRY, H. Image Stitching based on Feature Extraction Techniques: A Survey. *International Journal of Computer Applications*. Bangalore, India: Foundation of Computer Science. August 2014, vol. 99, no. 6, p. 1–8. IJCA, no. 2014. DOI: 10.5120/17374-7818. ISSN 0975-8887.
- [2] ALCANTARILLA, P. F., BARTOLI, A. and DAVISON, A. J. KAZE Features. In: FITZGIBBON, A., LAZEBNIK, S., PERONA, P., SATO, Y. and SCHMID, C., ed. *Computer Vision – ECCV 2012*. 1st ed. Berlin, Germany: Springer-Verlag, October 2012, p. 214–227. Lecture Notes in Computer Science, no. 7577. DOI: 10.1007/978-3-642-33783-3_16. ISBN 978-3-642-33782-6.
- [3] BAY, H., TUYTELAARS, T. and GOOL, L. V. SURF: Speeded Up Robust Features. In: LEONARDIS, A., BISCHOF, H. and PINZ, A., ed. *Computer Vision – ECCV 2006*. 1st ed. Berlin, Germany: Springer-Verlag, May 2006, p. 404–417. Lecture Notes in Computer Science, no. 3951. DOI: 10.1007/11744023_32. ISBN 978-3-540-33832-1.
- [4] BEYER, E., SOSINSKY, G., CRUM, J., BERTHOUD, V., LICHTENSETIN, A. et al. Homo sapiens, Multi-lamellar structure, HeLa. *The Cell Image Library* [online]. [cit. 2023-05-02]. Dataset. CCDB:6348. Available at: <https://doi.org/doi:10.7295/W9CCDB6348>.
- [5] BOYD, S. and VANDENBERGHE, L. Unconstrained minimization. In: *Convex Optimization*. Cambridge, UK: Cambridge University Press, March 2004, chap. 9, p. 457–520. ISBN 978-0-521-83378-3.
- [6] BROWN, M. Image Stitching. In: IKEUCHI, K., ed. *Computer Vision: A Reference Guide*. 1st ed. Boston, MA, USA: Springer, May 2014, p. 385–387. ISBN 978-0-387-30771-8.
- [7] BROWN, M. and LOWE, D. G. Automatic Panoramic Image Stitching using Invariant Features. *International Journal of Computer Vision*. Berlin, Germany: Springer Nature. August 2007, vol. 74, p. 59–73. IJCV, no. 2007. DOI: 10.1007/s11263-006-0002-3. ISSN 0920-5691.
- [8] BURETTE, A. C., WEINBERG, R., CRUM, J., ELLISMAN, M. and MARTONE, M. *Rattus rattus*, neuropil. *The Cell Image Library* [online]. January 2000. Revised on 9 January 2000 [cit. 2023-05-02]. Dataset. CCDB:8641. Available at: <https://doi.org/doi:10.7295/W9CCDB8641>.
- [9] BUSHONG, E. and DEERINCK, T. *Mus*, neuropil. *The Cell Image Library* [online]. [cit. 2023-04-20]. Dataset. CCDB:8192. Available at: <https://doi.org/doi:10.7295/W9CCDB8192>.

- [10] CARDONA, A., SAALFELD, S., SCHINDELIN, J., ARGANDA CARRERAS, I., PREIBISCH, S. et al. TrakEM2 Software for Neural Circuit Reconstruction. *PLOS ONE*. San Francisco, CA, USA: Public Library of Science. June 2012, vol. 7, no. 6, p. e38011. DOI: 10.1371/journal.pone.0038011. ISSN 1932-6203.
- [11] CASTELLS GRAELLS, R., RIBEIRO, J. R. S., DOMITROVIC, T., HESKETH, E. L., SCARFF, C. A. et al. Nudaurelia capensis omega virus capsid: virus-like particles expressed in Nicotiana benthamiana. *EMPIAR* [online]. April 2022. Revised on 4 April 2022 [cit. 2023-05-15]. Dataset. EMPIAR-10560. Available at: <https://doi.org/10.6019/EMPIAR-10560>.
- [12] CHALFOUN, J., MAJURSKI, M., BLATTNER, T., BHADRIRAJU, K., KEYROUZ, W. et al. MIST: Accurate and Scalable Microscopy Image Stitching Tool with Stage Modeling and Error Minimization. *Scientific Reports*. London, UK: Nature Portfolio. July 2017, vol. 7, p. 4988. DOI: 10.1038/s41598-017-04567-y. ISSN 2045-2322.
- [13] CHANG, C.-H., SATO, Y. and CHUANG, Y.-Y. Shape-Preserving Half-Projective Warps for Image Stitching. In: *Conference on Computer Vision and Pattern Recognition*. Columbus, OH, USA: IEEE, June 2014, p. 3254–3261. CVPR, no. 2014. DOI: 10.1109/CVPR.2014.422. ISSN 1063-6919.
- [14] CHEN, Y.-S. and CHUANG, Y.-Y. Natural Image Stitching with the Global Similarity Prior. In: LEIBEAND, B., MATAS, J., SEBE, N. and WELLING, M., ed. *Computer Vision – ECCV 2016*. 1st ed. Cham, Switzerland: Springer, September 2016, p. 186–201. Lecture Notes in Computer Science, no. 9909. DOI: 10.1007/978-3-319-46454-1_12. ISBN 978-3-319-46453-4.
- [15] CONRAD, R. and NARAYAN, K. CEM500K, a large-scale heterogeneous unlabeled cellular electron microscopy image dataset for deep learning. *ELife*. Cambridge, UK: eLife Sciences Publications, Ltd. April 2021, vol. 10, p. e65894. DOI: 10.7554/eLife.65894. ISSN 2050-084X.
- [16] DETONE, D., MALISIEWICZ, T. and RABINOVICH, A. *Deep Image Homography Estimation* [online]. ArXiv, June 2016, Revised on 13 June 2016 [cit. 2023-04-08]. Preprint. Available at: <https://doi.org/10.48550/arXiv.1606.03798>.
- [17] DETONE, D., MALISIEWICZ, T. and RABINOVICH, A. SuperPoint: Self-Supervised Interest Point Detection and Description. In: *Conference on Computer Vision and Pattern Recognition Workshops*. Salt Lake City, UT, USA: IEEE, June 2018, p. 337–349. CVPRW, no. 2018. DOI: 10.1109/CVPRW.2018.00060. ISSN 2160-7508.
- [18] ELLISMAN, M., RANGANATHAN, R., DEERINCK, T. J., YOUNG, S. J., HESSLER, D. et al. Homo sapiens, Neocortex pyramidal cell. *The Cell Image Library* [online]. [cit. 2023-04-10]. Dataset. CCDB:6332. Available at: <https://doi.org/doi:10.7295/W9CCDB6332>.
- [19] FISCHLER, M. A. and BOLLES, R. C. Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. *Communications of the ACM*. New York, NY, USA: Association for Computing Machinery. June 1981, vol. 24, no. 6, p. 381–395. DOI: 10.1145/358669.358692. ISSN 0001-0782.

- [20] GAO, J., KIM, S. J. and BROWN, M. S. Constructing image panoramas using dual-homography warping. In: *Conference on Computer Vision and Pattern Recognition*. Colorado Springs, CO, USA: IEEE, June 2011, p. 49–56. CVPR, no. 2011. DOI: 10.1109/CVPR.2011.5995433. ISSN 1063-6919.
- [21] GRISETTI, G., KÜMMERLE, R., STACHNISS, C. and BURGARD, W. A Tutorial on Graph-Based SLAM. *IEEE Intelligent Transportation Systems Magazine*. New York, NY, USA: IEEE. 2010, vol. 2, no. 4, p. 31–43. DOI: 10.1109/MITS.2010.939925. ISSN 1939-1390.
- [22] HARRIS, C. and STEPHENS, M. A Combined Corner and Edge Detector. In: TAYLOR, C. J., ed. *Proceedings of the Alvey Vision Conference*. Manchester, UK: Alvey Vision Club, September 1988, p. 23.1–23.6. AVC, no. 1988. DOI: 10.5244/C.2.23.
- [23] HERRMANN, C., WANG, C., BOWEN, R. S., KEYDER, E., KRAININ, M. et al. Robust Image Stitching with Multiple Registrations. In: FERRARI, V., HEBERT, M., SMINCHISESCU, C. and WEISS, Y., ed. *Computer Vision – ECCV 2018*. 1st ed. Cham, Switzerland: Springer, September 2018, p. 53–69. Lecture Notes in Computer Science, no. 11206. DOI: 10.1007/978-3-030-01216-8_4. ISBN 978-3-030-01215-1.
- [24] HOANG, V.-D., TRAN, D.-P., NHU, N. G., PHAM, T.-A. and PHAM, V.-H. Deep Feature Extraction for Panoramic Image Stitching. In: NGUYEN, N. T., JEANARAITANAKIJ, K., SELAMAT, A., TRAWIŃSKI, B. and CHITTAYASOTHORN, S., ed. *Intelligent Information and Database Systems (ACIIDS 2020)*. 1st ed. Cham, Switzerland: Springer, March 2020, p. 141–151. Lecture Notes in Computer Science, no. 12034. DOI: 10.1007/978-3-030-42058-1_12. ISBN 978-3-030-42057-4.
- [25] JIA, Q., LI, Z., FAN, X., ZHAO, H., TENG, S. et al. Leveraging Line-point Consistency to Preserve Structures for Wide Parallax Image Stitching. In: *Conference on Computer Vision and Pattern Recognition*. Nashville, TN, USA: IEEE, June 2021, p. 12181–12190. CVPR, no. 2021. DOI: 10.1109/CVPR46437.2021.01201. ISSN 1063-6919.
- [26] JOLLIFFE, I. T. *Principal Component Analysis*. 2nd ed. New York, NY, USA: Springer New York, October 2002. ISBN 978-0-387-95442-4.
- [27] KATHAROPOULOS, A., VYAS, A., PAPPAS, N. and FLEURET, F. Transformers are RNNs: fast autoregressive transformers with linear attention. In: DAUMÉ, H. and SINGH, A., ed. *Proceedings of the 37th International Conference on Machine Learning* [online]. JMLR.org, July 2020, p. 5156–5165 [cit. 2023-04-05]. PMLR, no. 119. Available at: <http://proceedings.mlr.press/v119/katharopoulos20a/katharopoulos20a.pdf>.
- [28] KE, Y. and SUKTHANKAR, R. PCA-SIFT: a more distinctive representation for local image descriptors. In: *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004*. Washington, DC, USA: IEEE, July 2004, vol. 2, p. II–II. CVPR, no. 2004. DOI: 10.1109/CVPR.2004.1315206. ISSN 1063-6919.
- [29] KIEVITS, A. J., LANE, R., CARROLL, E. C. and HOOGENBOOM, J. P. How innovations in methodology offer new prospects for volume electron microscopy.

Journal of Microscopy. New York, NY, USA: John Wiley & Sons Ltd. July 2022, vol. 287, no. 3, p. 114–137. DOI: 10.1111/jmi.13134. ISSN 0022-2720.

- [30] KIM, K.-Y. and ELLISMAN, M. Ultrastructural Characterization of the Mouse Optic nerve Head and Retina. *The Cell Image Library* [online]. August 2010. Revised on 2 August 2010 [cit. 2022-12-23]. Dataset. CCDB:7742. Available at: <https://doi.org/doi:10.7295/W9CCDB7742>.
- [31] KUGLIN, C. and HINES, D. A. The phase correlation image alignment method. In: *Proceedings of the 1975 IEEE International Conference on Cybernetics and Society*. New York, NY, USA: IEEE, 1975, p. 163–165.
- [32] LEE, K.-Y. and SIM, J.-Y. Warping Residual Based Image Stitching for Large Parallax. In: *Conference on Computer Vision and Pattern Recognition*. Seattle, WA, USA: IEEE, June 2020, p. 8195–8203. CVPR, no. 2020. DOI: 10.1109/CVPR42600.2020.00822. ISSN 1063-6919.
- [33] LEWIS, R. M., BASKARAN, H., GREEN, J., TASHEV, S., PALEOLOGOU, E. et al. SBF SEM of Human term placental villi. *EMPIAR* [online]. March 2022. Revised on 14 March 2022 [cit. 2023-04-10]. Dataset. EMPIAR-10967. Available at: <https://doi.org/10.6019/EMPIAR-10967>.
- [34] LI, K. and DING, G. A Novel Automatic Image Stitching Algorithm for Ceramic Microscopic Images. In: *International Conference on Audio, Language and Image Processing*. Shanghai, China: IEEE, July 2018, p. 17–21. ICALIP, no. 2018. DOI: 10.1109/ICALIP.2018.8455766. ISBN 978-1-5386-5196-4.
- [35] LI, N., XU, Y. and WANG, C. Quasi-Homography Warps in Image Stitching. *IEEE Transactions on Multimedia*. New York, NY, USA: IEEE. November 2018, vol. 20, no. 6, p. 1365–1375. DOI: 10.1109/TMM.2017.2771566. ISSN 1520-9210.
- [36] LIAO, T. and LI, N. Single-Perspective Warps in Natural Image Stitching. *IEEE Transactions on Image Processing*. New York, NY, USA: IEEE. August 2019, vol. 29, p. 724–735. DOI: 10.1109/TIP.2019.2934344. ISSN 1057-7149.
- [37] LIN, C.-C., PANKANTI, S. U., RAMAMURTHY, K. N. and ARAVKIN, A. Y. Adaptive as-natural-as-possible image stitching. In: *Conference on Computer Vision and Pattern Recognition*. Boston, MA, USA: IEEE, June 2015, p. 1155–1163. CVPR, no. 2015. DOI: 10.1109/CVPR.2015.7298719. ISSN 1063-6919.
- [38] LIN, K., JIANG, N., DO, L.-F. C. M. and LU, J. SEAGULL: Seam-Guided Local Alignment for Parallax-Tolerant Image Stitching. In: LEIBEAND, B., MATAS, J., SEBE, N. and WELLING, M., ed. *Computer Vision – ECCV 2016*. 1st ed. Cham, Switzerland: Springer, September 2016, p. 370–385. Lecture Notes in Computer Science, no. 9907. DOI: 10.1007/978-3-319-46487-9_23. ISBN 978-3-319-46486-2.
- [39] LIN, T.-Y., DOLLAR, P., GIRSHICK, R., HE, K., HARIHARAN, B. et al. Feature Pyramid Networks for Object Detection. In: *Conference on Computer Vision and Pattern Recognition*. Honolulu, HI, USA: IEEE, July 2017, p. 936–944. CVPR, no. 2017. DOI: 10.1109/CVPR.2017.106. ISSN 1063-6919.

- [40] LIN, W.-Y., LIU, S., MATSUSHITA, Y., NG, T.-T. and CHEONG, L.-F. Smoothly varying affine stitching. In: *Conference on Computer Vision and Pattern Recognition*. Colorado Springs, CO, USA: IEEE, June 2011, p. 345–352. CVPR, no. 2011. DOI: 10.1109/CVPR.2011.5995314. ISSN 1063-6919.
- [41] LOWE, D. G. Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision*. Berlin, Germany: Springer. November 2004, vol. 60, no. 2, p. 91–110. IJCV, no. 2004. DOI: 10.1023/B:VISI.0000029664.99615.94. ISSN 0920-5691.
- [42] LYU, W., ZHOU, Z., CHEN, L. and ZHOU, Y. A survey on image and video stitching. *Virtual Reality & Intelligent Hardware*. Amsterdam, Netherlands: Elsevier. February 2019, vol. 1, no. 1, p. 55–83. DOI: 10.3724/SP.J.2096-5796.2018.0008. ISSN 2096-5796.
- [43] MAHALINGAM, G., TORRES, R., KAPNER, D., TRAUTMAN, E. T., FLISS, T. et al. A scalable and modular automated pipeline for stitching of large electron microscopy datasets. *ELife*. Cambridge, UK: eLife Sciences Publications, Ltd. July 2022, vol. 11, p. e76534. DOI: 10.7554/eLife.76534. ISSN 2050-084X.
- [44] MIKOLAJCZYK, K. and SCHMID, C. A performance evaluation of local descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. New York, NY, USA: IEEE. August 2005, vol. 27, no. 10, p. 1615–1630. DOI: 10.1109/TPAMI.2005.188. ISSN 0162-8828.
- [45] MILGRAM, D. L. Computer Methods for Creating Photomosaics. *IEEE Transactions on Computers*. Washington, DC, USA: IEEE Computer Society. November 1975, C-24, no. 11, p. 1113–1119. DOI: 10.1109/T-C.1975.224142. ISSN 0018-9340.
- [46] MITTAL, A., MOORTHY, A. K. and BOVIK, A. C. No-Reference Image Quality Assessment in the Spatial Domain. *IEEE Transactions on Image Processing*. New York, NY, USA: IEEE. August 2012, vol. 21, no. 12, p. 4695–4708. DOI: 10.1109/TIP.2012.2214050. ISSN 1057-7149.
- [47] MUHLICH, J. L., CHEN, Y.-A., YAPP, C., RUSSELL, D., SANTAGATA, S. et al. Stitching and registering highly multiplexed whole-slide images of tissues and tumors using ASHLAR. *Bioinformatics*. Oxford, UK: Oxford University Press. October 2022, vol. 38, p. 4613–4621. DOI: 10.1093/bioinformatics/btac544. ISSN 1367-4803.
- [48] NIE, L., LIN, C., LIAO, K., LIU, M. and ZHAO, Y. A view-free image stitching network based on global homography. *Journal of Visual Communication and Image Representation*. Amsterdam, Netherlands: Elsevier. November 2020, vol. 73, p. 102950. DOI: 10.1016/j.jvcir.2020.102950. ISSN 1047-3203.
- [49] NIE, L., LIN, C., LIAO, K., LIU, S. and ZHAO, Y. Unsupervised Deep Image Stitching: Reconstructing Stitched Features to Images. *IEEE Transactions on Image Processing*. New York, NY, USA: IEEE. July 2021, vol. 30, p. 6184–6197. DOI: 10.1109/TIP.2021.3092828. ISSN 1057-7149.
- [50] NIE, L., LIN, C., LIAO, K. and ZHAO, Y. *Learning Edge-Preserved Image Stitching from Large-Baseline Deep Homography* [online]. ArXiv, December 2020, Revised on

11 December 2020 [cit. 2023-04-09]. Preprint. Available at:
<https://doi.org/10.48550/arXiv.2012.06194>.

- [51] OPENCV. Geometric Transformations of Images. *OpenCV Modules* [online]. 6 May 2023 [cit. 2023-05-06]. Available at:
https://docs.opencv.org/3.4/da/d6e/tutorial_py_geometric_transformations.html.
Path: Main Page; OpenCV-Python Tutorials; Image Processing in OpenCV.
- [52] PERES, M. R. *The Focal Encyclopedia of Photography*. 4th ed. Abingdon, UK: Taylor & Francis, May 2013. 176 p. ISBN 978-1-136-10614-9.
- [53] PIZER, S. M., AMBURN, E. P., AUSTIN, J. D., CROMARTIE, R., GESELOWITZ, A. et al. Adaptive histogram equalization and its variations. *Computer Vision, Graphics, and Image Processing*. Amsterdam, Netherlands: Elsevier. September 1987, vol. 39, no. 3, p. 355–368. DOI: 10.1016/S0734-189X(87)80186-X. ISSN 0734-189X.
- [54] PRIM, R. C. Shortest connection networks and some generalizations. *The Bell System Technical Journal*. Murray Hill, NJ, USA: Nokia Bell Labs. November 1957, vol. 36, no. 6, p. 1389–1401. DOI: 10.1002/j.1538-7305.1957.tb01515.x. ISSN 0005-8580.
- [55] ROSTEN, E. and DRUMMOND, T. Machine Learning for High-Speed Corner Detection. In: LEONARDIS, A., BISCHOF, H. and PINZ, A., ed. *Computer Vision – ECCV 2006*. 1st ed. Berlin, Germany: Springer-Verlag, May 2006, p. 430–443. Lecture Notes in Computer Science, no. 3951. DOI: 10.1007/11744023_34. ISBN 978-3-540-33832-1.
- [56] SARLIN, P.-E., DETONE, D., MALISIEWICZ, T. and RABINOVICH, A. SuperGlue: Learning Feature Matching With Graph Neural Networks. In: *Conference on Computer Vision and Pattern Recognition*. Seattle, WA, USA: IEEE, June 2020, p. 4937–4946. CVPR, no. 2020. DOI: 10.1109/CVPR42600.2020.00499. ISSN 1063-6919.
- [57] SAWHNEY, H. S. Simplifying motion and structure analysis using planar parallax and image warping. In: *Proceedings of 12th International Conference on Pattern Recognition*. Jerusalem, Israel: IEEE, October 1994, vol. 1, p. 403–408. ICPR, no. 1994. DOI: 10.1109/ICPR.1994.576308. ISBN 0-8186-6265-4.
- [58] SCHNEIDER, C. A., RASBAND, W. S. and ELICEIRI, K. W. NIH Image to ImageJ: 25 years of image analysis. *Nature Methods*. London, UK: Nature Portfolio. June 2012, vol. 9, p. 671–675. DOI: 10.1038/nmeth.2089. ISSN 1548-7091.
- [59] SCHNEIDER, J. P., HEGERMANN, J. and WREDE, C. Volume electron microscopy: analyzing the lung. *Histochemistry and Cell Biology*. Berlin, Germany: Springer-Verlag. February 2021, vol. 155, no. 2, p. 241–260. DOI: 10.1007/s00418-020-01916-3. ISSN 0948-6143.
- [60] SHI, Z., LI, H., CAO, Q., REN, H. and FAN, B. An Image Mosaic Method Based on Convolutional Neural Network Semantic Features Extraction. *Journal of Signal Processing Systems*. Berlin, Germany: Springer Nature. April 2020, vol. 92, no. 4, p. 435–444. DOI: 10.1007/s11265-019-01477-2. ISSN 1939-8018.
- [61] SINGLA, A., LIPPMANN, B. and GRAEB, H. Recovery of 2D and 3D Layout Information through an Advanced Image Stitching Algorithm using Scanning

- Electron Microscope Images. In: *International Conference on Pattern Recognition*. Milan, Italy: IEEE, January 2021, p. 3860–3867. ICPR, no. 2020. DOI: 10.1109/ICPR48806.2021.9412334. ISSN 1051-4651.
- [62] SUN, J., SHEN, Z., WANG, Y., BAO, H. and ZHOU, X. LoFTR: Detector-Free Local Feature Matching with Transformers. In: *Conference on Computer Vision and Pattern Recognition*. Nashville, TN, USA: IEEE, June 2021, p. 8918–8927. CVPR, no. 2021. DOI: 10.1109/CVPR46437.2021.00881. ISSN 1063-6919.
- [63] SZELISKI, R. Image Alignment and Stitching: A Tutorial. *Foundations and Trends® in Computer Graphics and Vision*. Delft, Netherlands: Now Publishers. January 2007, vol. 2, no. 1, p. 1–104. DOI: 10.1561/0600000009. ISSN 1572-2740.
- [64] SZELISKI, R. *Computer Vision: Algorithms and Applications*. 2nd ed. Cham, Switzerland: Springer, January 2022. Texts in Computer Science. ISBN 978-3-030-34372-9.
- [65] THE MICRONS CONSORTIUM, BAE, J. A., BAPTISTE, M., BISHOP, C. A., BODOR, A. L. et al. *Functional connectomics spanning multiple areas of mouse visual cortex* [online]. Cold Spring Harbor Laboratory, July 2021, Revised on 19 April 2023 [cit. 2023-04-25]. Preprint. Available at: <https://www.biorxiv.org/content/early/2023/04/19/2021.07.28.454025>.
- [66] VASWANI, A., SHAZEER, N., PARMAR, N., USZKOREIT, J., JONES, L. et al. Attention Is All You Need. In: LUXBURG, U. von, GUYON, I., BENGIO, S., WALLACH, H. and FERGUS, R., ed. *Proceedings of the 31st International Conference on Neural Information Processing Systems*. Red Hook, NY, USA: Curran Associates Inc., December 2017, p. 6000–6010. NIPS, no. 2017. DOI: 10.1109/CVPR.2015.7298719. ISBN 978-1-5108-6096-4.
- [67] VESCOVI, R., LI, H., KINNISON, J., KEÇELI, M., SALIM, M. et al. Toward an Automated HPC Pipeline for Processing Large Scale Electron Microscopy Data. In: *IEEE/ACM Annual Workshop on Large-scale Experiment-in-the-Loop Computing*. Georgia, USA: IEEE, November 2020, p. 16–22. XLOOP, no. 2020. DOI: 10.1109/XLOOP51963.2020.00008. ISBN 978-1-6654-2283-3.
- [68] WANG, Z. and YANG, Z. Review on image-stitching techniques. *Multimedia Systems*. Berlin, Germany: Springer-Verlag. August 2020, vol. 26, no. 4, p. 413–430. DOI: 10.1007/s00530-020-00651-y. ISSN 0942-4962.
- [69] XIANG, T.-Z., XIA, G.-S., BAI, X. and ZHANG, L. Image stitching by line-guided local warping with global similarity constraint. *Pattern Recognition*. Amsterdam, Netherlands: Elsevier. November 2018, vol. 83, p. 481–497. DOI: 10.1016/j.patcog.2018.06.013. ISSN 0031-3203.
- [70] ZARAGOZA, J., CHIN, T.-J., TRAN, Q.-H., BROWN, M. S. and SUTER, D. As-Projective-As-Possible Image Stitching with Moving DLT. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. New York, NY, USA: IEEE. December 2013, vol. 36, no. 7, p. 1285–1298. DOI: 10.1109/TPAMI.2013.247. ISSN 0162-8828.
- [71] ZHANG, F. and LIU, F. Parallax-Tolerant Image Stitching. In: *Conference on Computer Vision and Pattern Recognition*. Columbus, OH, USA: IEEE, June 2014, p. 3262–3269. CVPR, no. 2014. DOI: 10.1109/CVPR.2014.423. ISSN 1063-6919.

- [72] ZHANG, G., HE, Y., CHEN, W., JIA, J. and BAO, H. Multi-Viewpoint Panorama Construction With Wide-Baseline Images. *IEEE Transactions on Image Processing*. New York, NY, USA: IEEE. February 2016, vol. 25, no. 7, p. 3099–3111. DOI: 10.1109/TIP.2016.2535225. ISSN 1057-7149.
- [73] ZHANG, L., ZHANG, L., MOU, X. and ZHANG, D. FSIM: A Feature Similarity Index for Image Quality Assessment. *IEEE Transactions on Image Processing*. New York, NY, USA: IEEE. January 2011, vol. 20, no. 8, p. 2378–2386. DOI: 10.1109/TIP.2011.2109730. ISSN 1057-7149.
- [74] ZHAO, Q., MA, Y., ZHU, C., YAO, C., FENG, B. et al. Image stitching via deep homography estimation. *Neurocomputing*. Amsterdam, Netherlands: Elsevier. August 2021, vol. 450, p. 219–229. DOI: 10.1016/j.neucom.2021.03.099. ISSN 0925-2312.
- [75] ZHOU, W., BOVIK, A. C., SHEIKH, H. R. and SIMONCELLI, E. P. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*. New York, NY, USA: IEEE. April 2004, vol. 14, no. 4, p. 600–612. DOI: 10.1109/TIP.2003.819861. ISSN 1057-7149.

Appendix A


Contents of the Attached Medium

The most notable directories and files on the attached memory medium are as follows.


- `/demis/`
 - `/configs/` – Configuration files for executing the DEMIS tool.
 - `/datasets/DEMIS/` – Sample data from the DEMIS dataset.
 - `/datasets/DEMIS Source/` – Sample source images of the DEMIS dataset.
 - `/LoFTR/` – Source codes of the included LoFTR module.
 - `/scripts/` – Scripts for image stitching and DEMIS dataset generation.
 - `/src/` – Source codes of the DEMIS tool.
 - `/docs/demis_references.md` – References for images in the DEMIS dataset.
 - `/notebooks/stitch.ipynb` – Jupyter notebook for manual image stitching.
 - `/environment.yml` – Anaconda environment specification.
 - `/README.md` – Manual for setting up the environment and code execution.
- `/tex/` – \LaTeX source codes of this thesis.
- `/poster.pdf` – Poster presenting the goals and results of this thesis.
- `/README.md` – Manual for working with the attached medium.
- `/thesis.pdf` – This thesis in PDF.

Appendix B

Depiction of the Created Poster



**BRNO FACULTY
UNIVERSITY OF INFORMATION
OF TECHNOLOGY TECHNOLOGY**



TESCAN 3DIM
3D IMAGING & ANALYSIS JOINT VENTURE

Deep Learning for Image Stitching

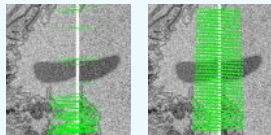
Bc. Petr Šilling, xsilli01@stud.fit.vutbr.cz

Supervisor: Ing. Michal Španěl, Ph.D. Consultant: Ing. Oldřich Kodým, Ph.D., TESCAN 3DIM s.r.o.

Motivation

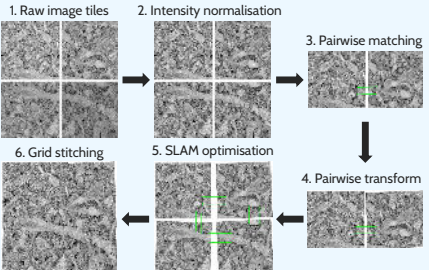
- Grids of overlapping EM images need to be stitched together
- Current methods tend to use SIFT
- Issues with low-texture regions
- Could CNNs increase robustness?

SIFT: 85 matches LoFTR: 624 matches



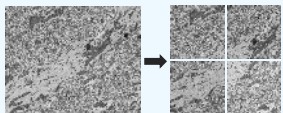
Proposed Solution – DEMIS

- Standard feature-based stitching pipeline with SLAM global optimisation and feature matching by LoFTR.



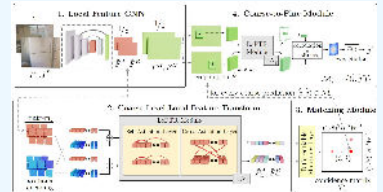
Synthetic Dataset

- 424 images from EMPIAR and CIL split into grids of image tiles
- Random brightness, contrast, translation, rotation, and Gaussian noise applied to each tile



Local Feature Transformer (LoFTR)

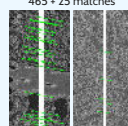
- Fine-tuned on EM images from the synthetic dataset.



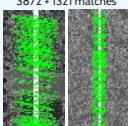
Results on Evaluation Images

Metric (average on 1306 images)	SIFT baseline	Pre-trained	Fine-tuned
Feature matching time	80.44 ms	83.52 ms	8318 ms
Feature matches found	694	770	778
Outlier feature matches	5	3	2
Inlier reprojection error	2.86 px	2.90 px	2.82 px
Corner error AUC at 5 px	42.70 %	40.99%	46.02 %
PSNR of image pairs	22.07	21.88	22.33
PSNR of complete grids	15.22	15.23	15.04
SSIM of image pairs	0.68	0.67	0.69
SSIM of complete grids	0.22	0.22	0.21

SIFT baseline
465 + 25 matches



Pre-trained DEMIS
3872 + 1321 matches



Fine-tuned DEMIS
4044 + 1329 matches

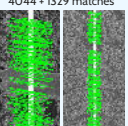


Figure B.1: Illustration of the poster that presents the goals and results of this thesis.