UNIVERSITY OF TURKU

Developing a Python-Based Image Analysis Pipeline for 3D Segmentation and Quantification of Plasmodium falciparum Ookinetes from Volume Electron Microscopy Datasets

Master’s Degree Program in Biomedical Imaging

Master's thesis

University of Turku

Faculty of Medicine

Institute of Biomedicine

Author(s):

Ramish Bibi

25.07.2025

Turku

Master's thesis

Subject: Master's Degree Programme in Biomedical Imaging
Author(s): Ramish Bibi
Title: Developing Python-Based Image Analysis Pipeline for 3D Segmentation and Quantification of *Plasmodium falciparum* Ookinetes from Volume Electron Microscopy Datasets
Supervisor(s): Dr. Pasi Kankaanpää; Junel Solis
Number of pages: 74 pages
Date: 25.07.2025

Abstract

Understanding the complex 3D organization of Plasmodium falciparum ookinetes is essential for uncovering how these malaria-causing parasites develop and invade their mosquito hosts. This thesis presents a practical and semi-automated image analysis pipeline designed to extract meaningful biological information from large-scale volumetric electron microscopy (EM) datasets of ookinetes. The pipeline was developed in Python and orchestrated using the workflow management tool Nextflow. It was designed to handle the challenges posed by large-sized raw tiled images, and image artifacts commonly found in volumetric EM data.

The analysis pipeline begins with automated tile stitching, producing complete 3D image stacks from raw TIFF mosaics. These stitched volumes are converted to the efficient next-generation file format OME-Zarr, which enables reduced storage, faster visualization, and accessibility. To facilitate downstream processing, parasite-containing regions are isolated through automatic volume cropping, followed by semi-automated image registration to align z-slices.

A key focus of this study is on 3D segmentation of ookinetes and their ultrastructural organelles including micronemes, hemozoin crystals, nuclei, and crystalloids. Initial attempts to use Meta AI's Segment Anything Model (SAM) directly on EM images were unsuccessful due to its limited performance in low-contrast biological images. Instead, segmentation was successfully performed using Napari-based tools (napari-SAM and napari-SAMv2), which combine SAM's strengths with user-guided annotation to produce accurate 3D masks.

From these segmented volumes, a series of organelle-level quantifications were carried out, including measurements of count, volume, and spatial distribution of micronemes, crystalloids and hemozoin crystals. The final output of the pipeline is a set of structured CSV files that biologists can use to explore developmental stage specific differences and gain insight into parasite biology. While this thesis does not attempt to draw statistical or biological conclusions, it establishes a robust computational framework for future expert-led biological interpretation and discovery.

In summary, this work demonstrates a scalable, semi-automated solution to process challenging 3D EM datasets from raw image tiles to quantifiable biological features providing a strong foundation for future studies into the cell biology of Plasmodium parasites.

Key words: volume EM, Python, image analysis pipeline, OME-Zarr, Nextflow, image stitching, image registration, 3D segmentation, big image data.

Table of contents

List of Figures:

List of Tables:

List of Abbreviations:

AT-SEM	Array Tomography Scanning Electron Microscopy
CNN	Convolutional Neural Networks
CSV	Comma Separated Values
EM	Electron Microscopy
FOV	Field of View
hpi	Hours Post Infection
JSON	JavaScript Object Notation
NGFF	Next-Generation File Format
OME	Open Microscopy Environment
P.falciparum	Plasmodium falciparum
RBCs	Red Blood Cells
ROI	Regions of Interest
SAM	Segment Anything Model
TIFF	Tagged Image File Format
vEM	Volume Electron Microscopy

1 Introduction

Malaria continues to be one of the world's most pressing public health concerns, especially in tropical and subtropical regions. Among the parasites responsible, *Plasmodium falciparum* (*P. falciparum*) is the most lethal and widespread, particularly in sub-Saharan Africa [1]. While many studies have focused on the parasite's blood-stage biology, much less is known about the early mosquito stages, especially the ookinete, a highly motile form that plays a key role in malaria transmission [2].

Understanding how ookinetes develop and migrate through the mosquito midgut is essential, as this step represents a bottleneck in the parasite's life cycle and a potential target for new transmission-blocking strategies [3], [4]. Studying these is extremely challenging due to their sparse distribution, dynamic intracellular changes, and the complexity of the surrounding mosquito tissues [2]. Thanks to decades of microscopy work, especially electron microscopy (EM), researchers have uncovered many of its key features. However, important questions still remain, particularly regarding its organellar dynamics and developmental changes.

To gain meaningful insight into the structural complexity of this critical stage, this project was designed as a collaborative effort between three specialized institutions, funded by ISIDORe (Integrated Services for Infectious Disease Outbreak Research). A European infrastructure initiative that supports research preparedness and response to infectious disease threats. Sample preparation was carried out at the Max Planck Institute for Infection Biology in Berlin, Germany, in the lab of Elena Levashina by Pablo Suárez-Cortés. These samples were then processed and imaged at the Euro-BioImaging Prague ALM & EM Node using array tomography scanning electron microscopy (AT-SEM). Finally, the resulting 3D volumetric image datasets were sent to the Euro-BioImaging Finnish Advanced Microscopy (FiAM) Node, where this thesis project was carried out, focusing on developing a scalable and automated image analysis pipeline to make full use of these high-resolution volumetric data.

The imaging component of this work generates extremely detailed and informative data, the analysis of such large 3D volumes presents a major technical challenge. For instance, a single AT-SEM dataset can contain hundreds or even thousands of overlapping 2D image tiles per section and many such sections stacked into a full volume, resulting in tens to hundreds of gigabytes of raw data per sample [5]. These datasets must be stitched, aligned, segmented, and quantified, an enormous task if done manually.

This thesis focuses on solving the AT-SEM analysis challenge by developing a Python-based image analysis pipeline tailored to the large-scale, high-resolution data generated in this project. The goal is to create a reproducible, modular workflow that can process raw volume electron microscopy (vEM) data of ookinetes, starting from stitching and registration of image tiles, continuing through segmentation of parasites and surrounding structures, and ending in detailed morphological quantification.

In modern biological imaging, there is a growing need for pipelines that are not only accurate but also reproducible, scalable and interoperable with new data formats. Therefore, this thesis also integrates next-generation bioimaging tools such as OME-Zarr, a scale-friendly data format designed for large multidimensional datasets, and Nextflow, a workflow manager that allows complex analysis steps to be linked together and executed in a controlled, reproducible manner.

By combining these technologies, this project aims to streamline the entire data processing workflow, enabling biologists to analyze 3D parasite morphology across entire volumes with minimal manual intervention. In doing so, the pipeline will help unlock new insights into how *P. falciparum* ookinetes develop, migrate, and interact with mosquito tissues during infection.

2 Aims of This Thesis

The overall aim of this thesis is to develop a complete, Python-based pipeline that takes raw vEM image datasets of *P. falciparum* ookinetes and produces 3D segmentation and quantitative morphological analyses in an automated and scalable fashion.

2.1.1 Specific Objectives

The specific objectives are:

1. Data Preprocessing and Organization
   Assemble and manage large volume EM datasets by reading and organizing unstitched image tiles based on their original acquisition layout. Store data efficiently using the OME-Zarr format, enabling scalable access and visualization.
2. Image Stitching
   Implement methods to stitch 2D image tiles into seamless slices, handling overlapping fields and typical image artifacts such as drift, folds, or illumination differences.
3. 3D Image Registration
   Align the stitched slices into coherent 3D volumes, taking into account biological variability and distortions between sections. Explore both automated and semi-automated registration approaches.
4. Segmentation of Ookinetes and Organelles
   Develop semi-automated or automated segmentation tools to identify ookinetes and their key structures e.g., micronemes, nucleus, hemozoin, and crystalloids across different developmental stages and spatial environments.
5. Quantitative Morphological Analysis
   Extract and compute biological metrics such as volume, number, and spatial distribution and arrangements of parasites and their internal structures. This will allow detailed characterization of parasite development and behavior.
6. Pipeline Automation with Nextflow
   Combine all processing steps into a reproducible and portable workflow using Nextflow. This will allow the pipeline to run consistently across systems, improve transparency, and make it easier for collaborators to adopt and reuse the workflow.
7. Scalability for Big Datasets
   Ensure that the pipeline can handle large 3D image volumes (tens to hundreds of gigabytes) without overloading memory, using multi-core processing and effective

memory management to maintain performance. This includes chunk-based processing and on-demand data loading, supported by the OME-Zarr format.

The diagram in Figure 2.1 illustrates the steps of the image analysis pipeline graphically.

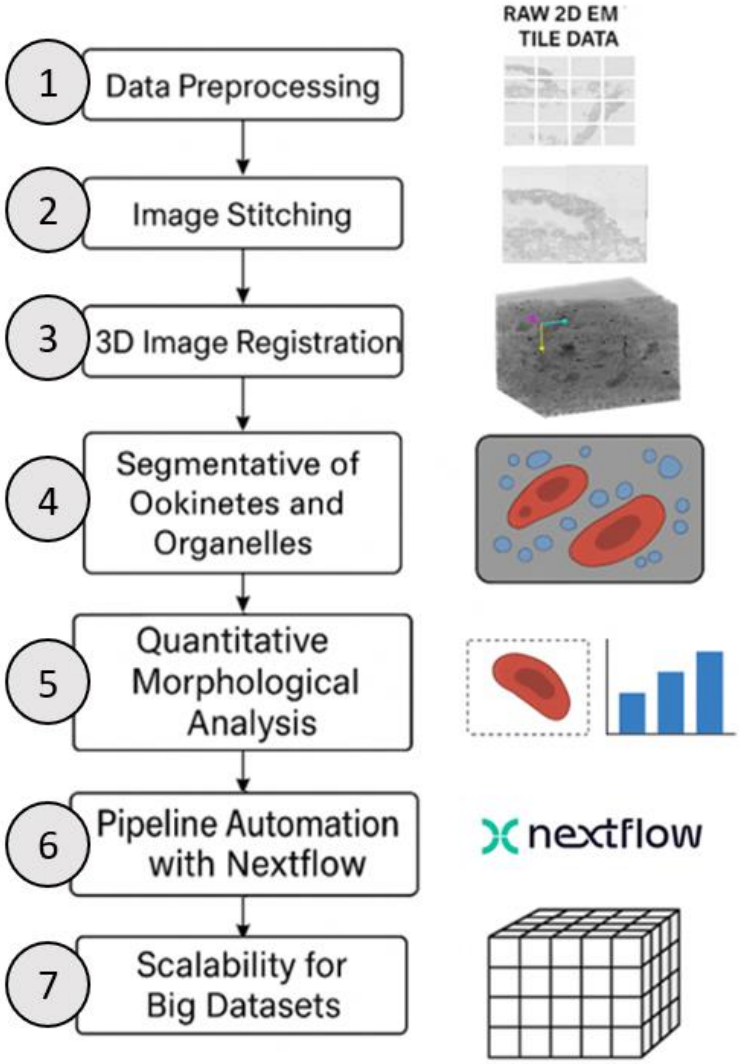


Figure 2.1. Overview of the image analysis pipeline developed for 3D segmentation and quantification of P. falciparum ookinetes from vEM datasets.

This work represents an essential step in bridging cutting-edge vEM with reproduceable, scalable, automated computational analysis.

3 Background and Literature Review

Understanding the biology of *P. falciparum* ookinetes and the technical challenges of processing volumetric EM data is essential for building an effective 3D image analysis pipeline. This chapter outlines the theoretical basis and tools that support the pipeline developed in this thesis.

3.1 Malaria as a Global Health Challenge

Malaria remains one of the world's most pressing infectious diseases, imposing a huge health and economic burden, especially in tropical and subtropical regions. Nearly half of the global population lives at risk of malaria across roughly 85 countries, with an estimated 249 million cases and 608,000 deaths in 2022 alone [6]. The vast majority of fatalities occur in sub-Saharan Africa (around 95% of malaria deaths), disproportionately affecting young children and pregnant women [7]. Malaria is caused by protozoan parasites of the genus *Plasmodium*, of which five species infect humans are *P. falciparum*, *P. vivax*, *P. ovale*, *P. malariae*, and the zoonotic *P. knowlesi*. Among these, *P. falciparum* is the most virulent and predominant, responsible for the most severe cases and over 90% of malaria mortality worldwide [8]. In fact, *P. falciparum* accounts for virtually all malaria infections in many African regions (e.g. ~99% of cases in sub-Saharan Africa) and is the species that most often causes life-threatening complications [9]. This high burden and deadly potential make *P. falciparum* a central focus of malaria research and control efforts. Overall, despite concerted global efforts and some progress in reducing malaria incidence over past decades, the disease remains entrenched as a leading cause of illness and death in endemic countries, underscoring the need for continued research into its biology and new tools for intervention [6].

3.2 Plasmodium Life Cycle in Human and Mosquito Hosts

P. falciparum has a complex life cycle, involving both asexual and sexual stages in two hosts: humans and mosquitoes [10]. In humans, infection begins when an infected female *Anopheles* mosquito takes a blood meal and inoculates sporozoites into the person's bloodstream. These sporozoites rapidly home to the liver and invade hepatocytes (liver cells), each developing into a liver-stage schizont (a large, multinucleated cell filled with newly formed parasites). After about 1–2 weeks, the schizonts rupture to release thousands of merozoites (invasive daughter cells) into the blood. Notably, in *P. vivax* and *P. ovale*, a subset of sporozoites can become dormant hypnozoites (non-replicating liver-stage forms that can reactivate later) in the liver,

causing relapses much later [11]. The released merozoites then initiate the erythrocytic cycle by invading red blood cells (RBCs). Inside RBCs, the parasites multiply asexually through three distinct stages (ring, trophozoite and schizont), eventually bursting the host cell to release a new wave of merozoites that infect fresh RBCs. This repeated blood-stage replication (known as erythrocytic schizogony) is what produces the clinical symptoms of malaria such as periodic fever and anemia. After several asexual cycles, some of the parasites differentiate into sexual forms called gametocytes (male microgametocytes and female macrogametocytes) [11]. These gametocytes circulate in the human blood and represent the transmissible stage that can continue the life cycle if ingested by a mosquito.

When a mosquito vector feeds on an infected person, it ingests blood containing gametocytes. Within the mosquito midgut, the gametocytes rapidly undergo sexual reproduction. The male microgametocyte undergoes ex-flagellation to produce motile flagellated microgametes that swim and fertilize the female macrogamete, forming a zygote [1]. The zygote then develops into an elongate motile form called the ookinete, which is an invasive stage adapted to traverse the mosquito's midgut epithelium. The ookinete burrows through the midgut lining and lodges on the exterior of the stomach wall, where it rounds up to form an oocyst [1][9][12]. Inside the oocyst, the parasite undergoes sporogonic development: over 1–2 weeks, the oocyst grows and divides internally to generate thousands of new sporozoites. Eventually, the oocyst ruptures, releasing these sporozoites into the mosquito's body cavity, from which they migrate to the salivary glands. Now the mosquito is infectious, during a subsequent bite on a human host, the sporozoites in its saliva are injected and can initiate a new infection, thus completing the cycle [2]. The entire life cycle of *Plasmodium* is illustrated in Figure 3.1, highlighting the alternation between human liver and blood stages and mosquito gut stage. Understanding each stage is crucial for developing interventions, and notably the sexual and mosquito-stage parasites are essential for transmission of malaria to new hosts [9].

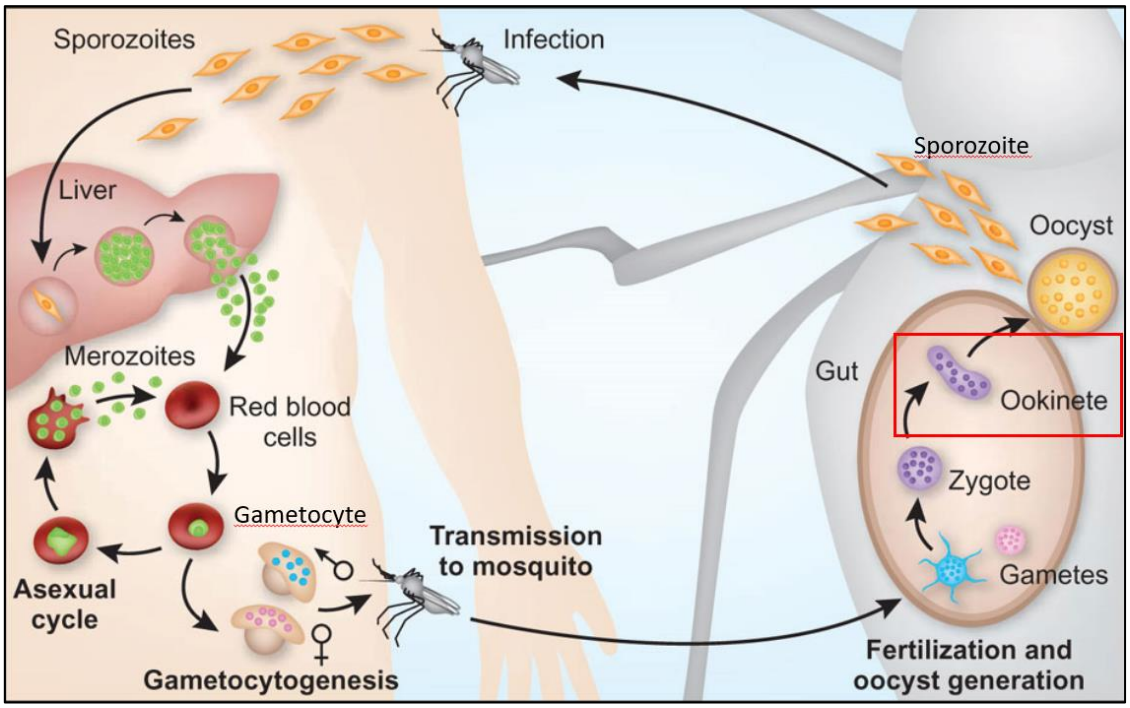


Figure 3.1. The P. falciparum life cycle spans two hosts - humans and mosquitoes - and entails multiple developmental stages in each. Image adapted from Pasvol et al. 2010. [12]

3.2.1 The Mosquito Stages and the Ookinete

Once inside the mosquito, *Plasmodium* must successfully navigate the mosquito's midgut environment to continue its development. This bottleneck phase in the parasite's journey is critical: for *P. falciparum*, it is estimated that over 99.9% of parasites do not survive past the mosquito midgut – there is roughly a 300-fold drop in numbers from ingested gametocytes to ookinetes, and another ~100-fold loss from ookinetes to oocysts [13]. Only a tiny fraction of the original parasites manages to form oocysts, which makes the ookinete stage a key point of vulnerability in the life cycle. The ookinete's role is to bridge the parasite from the mosquito's lumen (where the blood meal resides) to a secure niche on the midgut wall, enabling subsequent oocyst and sporozoite development [14]. In other words, the ookinete is the vehicle by which the fertilized parasite escapes the hostile gut contents and establishes infection in the mosquito, without a successful ookinete, the parasite's development in the vector would halt, and no sporozoites would be produced to infect new human hosts.

3.2.2 Ookinete Development and Invasion of the Midgut

After fertilization in the mosquito's stomach, the resulting zygote transforms into an ookinete, typically within 12–24 hours, see red box in Figure 2.1. The ookinete is a slender, elongated

cell (often banana-shaped, a few tens of microns long) equipped for active migration [15]. It is an *Apicomplexan* parasite stage, and like other invasive forms such as sporozoites and merozoites, it possesses an apical complex of secretory organelles and a polarized cytoskeleton that facilitates cell invasion. The ookinete is covered by a triple-layered pellicle formed by the parasite's plasma membrane and an underlying inner membrane complex which provides rigidity and gliding motility [15]. This motility allows the ookinete to bore through the peritrophic matrix and traverse the mosquito midgut epithelium. Figure 3.2 illustrates this invasion process: the ookinete (green) first attaches to and invades an epithelial cell from the midgut lumen, then passes through the cell and emerges out the basal side of the midgut wall [16][17]. Upon reaching the exterior of the midgut, the ookinete comes to rest underneath the basal lamina (a thin membrane surrounding the gut) [18]. There, it rounds up into an oocyst, encased just beneath the mosquito midgut surface. In this protected site, the parasite undergoes sporogony, multiplying within the oocyst to create sporozoites. Eventually, dozens to thousands of sporozoites emerge from each oocyst and migrate to the mosquito's salivary glands, completing the parasite's development in the vector [19][20]

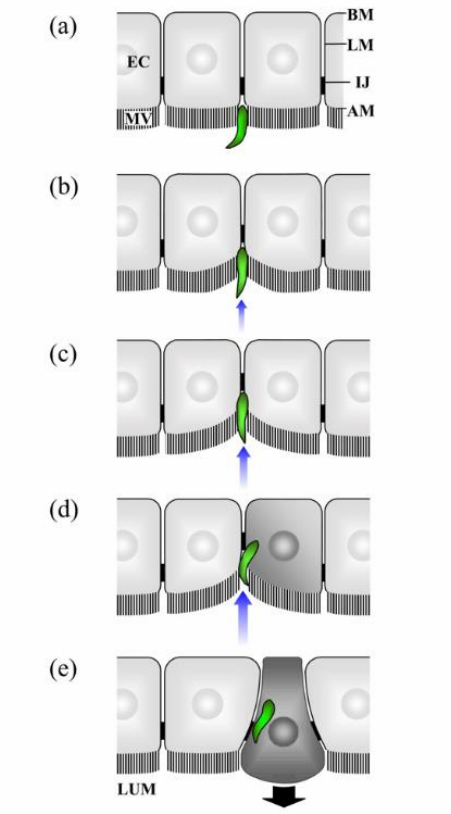


Figure 3.2. Schematic of an ookinete (green) invading the mosquito midgut epithelium. Panels (a)–(e) illustrate the sequence of midgut invasion: (a) the ookinete attaches to the apical brush border (microvilli) of an epithelial cell (EC) at the cell junction (IJ, intercellular junction); (b–

c) the parasite pushes into the epithelial layer (blue arrows indicate the invagination of the cell layer); (d) the ookinete penetrates the cell's lateral border and enters the cell interior (the invaded cell is shaded grey); (e) the ookinete exits at the basal side of the epithelium, while the heavily damaged host cell undergoes extrusion into the lumen (black arrow indicates the dying cell being sloughed off). Figure adapted from [21]

The traversal of the midgut by ookinetes is pivotal for the parasite's life cycle. However, it is also fraught with challenges the mosquito's immune defenses often target and kill ookinetes, and only the most robust manage to establish oocysts. From a transmission standpoint, the ookinete represents a choke point where the malaria parasite population is drastically filtered [22]. This makes the ookinete an attractive target for transmission-blocking interventions: for example, vaccines that elicit antibodies against ookinete surface proteins (like P25 and P28) aim to immobilize or kill the ookinete in the mosquito midgut, thereby preventing the parasite from developing further [3][4]. Studying the biology of ookinetes their morphology, motility, and interactions with the mosquito gut environment is therefore extremely important for understanding transmission and for devising new strategies to block it.

3.3 Microscopy Approaches for Studying *P. falciparum* Ookinetes

Initial observations of *P. falciparum* ookinetes were made using classical light microscopy, particularly through Giemsa-stained mosquito gut smears. These studies enabled basic morphological descriptions—for example, the identification of the motile ookinete stage within the mosquito midgut—but the internal architecture of the parasite remained largely unresolved due to the limited resolution of light microscopy [23]. To gain deeper insight into subcellular structures, researchers turned to EM. From the mid-20th century through 2000, conventional EM provided pivotal insights into the parasite's ultrastructure [24], revealing organelles such as the inner membrane complex (IMC), and micronemes, especially in blood-stage and sporozoite forms [25] [26]. These EM studies relied on single thin sections and offered only static ultrastructural snapshots; researchers had to infer 3D relationships from isolated slices, often missing contextual detail.

In parallel, advances in light microscopy—such as fluorescence, confocal, and super-resolution imaging—have been applied to study *Plasmodium* ookinetes, enabling researchers to visualize the localization of specific proteins, track parasite motility, and examine interactions with mosquito midgut tissues [27]. These techniques have provided important insights into the dynamics of cytoskeletal components, surface proteins, and secretory organelles like

micronemes. However, despite these advances, light microscopy remains constrained by diffraction-limited resolution and limiting its ability to fully resolve the complex 3D ultrastructure of the ookinete.

Recent developments in three-dimensional imaging, particularly vEM, now allow for nanometer-scale reconstruction of entire parasite cells within their tissue context. Techniques such as focused ion beam scanning electron microscopy (FIB-SEM) have been applied to blood-stage schizonts to reveal 3D architectures of cell division [28], and to oocysts to study nuclear organization during sporogony [29]. More recently, Evers et al. (2025) reconstructed *P. falciparum* gametocytes in 3D using high-resolution vEM, uncovering unexpected structural features [24]. While such studies provide valuable insights, precise quantification of organelle counts, volumes and their 3D spatial arrangement in *P. falciparum* ookinetes across different developmental stages are still lacking in the literature. Quantitative vEM holds great promise for addressing these gaps, especially in relation to ookinete morphology and midgut traversal.

Although modern vEM techniques, such as AT-SEM offer unprecedented views into parasite ultrastructure, they also present significant image analysis challenges. These techniques generate large, tiled 3D datasets that are rich in detail but difficult to interpret without computational support [5]. Transforming raw image volumes into meaningful biological insights requires robust image analysis pipelines. A typical vEM analysis workflow involves stitching, volume alignment, segmentation and quantification—serving as a critical bridge between raw image data and scientific discovery. Developing a Python-based image analysis pipeline (as undertaken in this thesis) can greatly accelerate the processing and quantification of these vEM datasets, enabling researchers to manage these complex datasets and extract biologically meaningful metrics in a high-throughput and reproducible manner.

3.4 Image Stitching: Overview and Relevance in Microscopy

Image stitching is a computational technique for merging multiple overlapping images into a single seamless composite. This approach is widely used to overcome the limited field of view (FoV) of high-magnification imaging systems by creating a larger panorama without sacrificing resolution [30]. In microscopy, stitching allows researchers to capture expansive areas of a specimen at cellular or sub-cellular resolution and then digitally “stitch” these tiles together into one large image. The resulting composite image provides both the breadth (large FoV) and detail (high resolution) needed to analyze complex biological structures. For example, in digital pathology and whole-slide imaging, dozens or hundreds of high-power field images are stitched

to reconstruct an entire tissue section for examination [31]. This principle equally applies to other domains like electron microscopy, fluorescence microscopy and even panoramic photography, underscoring that image stitching is a general solution for imaging beyond a single frame's limits.

3.4.1 Image Stitching in vEM

The need for image stitching is particularly acute in vEM, where ultrahigh resolution is required over relatively large sample volumes. A fundamental challenge in vEM is the trade-off between FoV and resolution: at the high magnifications needed for nanometer detail, the FoV of an electron microscope is extremely small (often only tens of micrometer) [32]. To map out a large region or an entire cell in 3D, one cannot avoid imaging many small tiles and then stitching them together. Thus, in vEM data acquisition, the sample is imaged as a mosaic of overlapping fields, which are later computationally merged [33]. In other words, stitching is a fundamental process in the overall vEM pipeline, allowing researchers to achieve both the needed scope and detail of ultrastructural information.

A prime example of vEM requiring extensive stitching is array AT-SEM. In AT-SEM, a biological sample such as a block of tissue is physically sectioned using an ultramicrotome, which slices it into hundreds of ultrathin slices (typically 50–100 nm thick) that are placed on a substrate in an ordered array [34]. Each section is then imaged, typically using an automated stage to acquire a grid of partially overlapping tile images that cover the entire section at high resolution, see Figure 3.3. This method yields a series of 2D mosaic images (one per section), which together represent a 3D volume when stacked. Stitching in this context occurs at the level of each section: the tile images within a single section must be accurately stitched to reconstruct the full field for that slice [35]. Each section must be assembled first, and then the slices can be registered to each other along the z-axis to reconstruct the 3D volume. By combining many high-res tiles per section, AT provides "ultrastructure with large scale and high resolution simultaneously" [36].

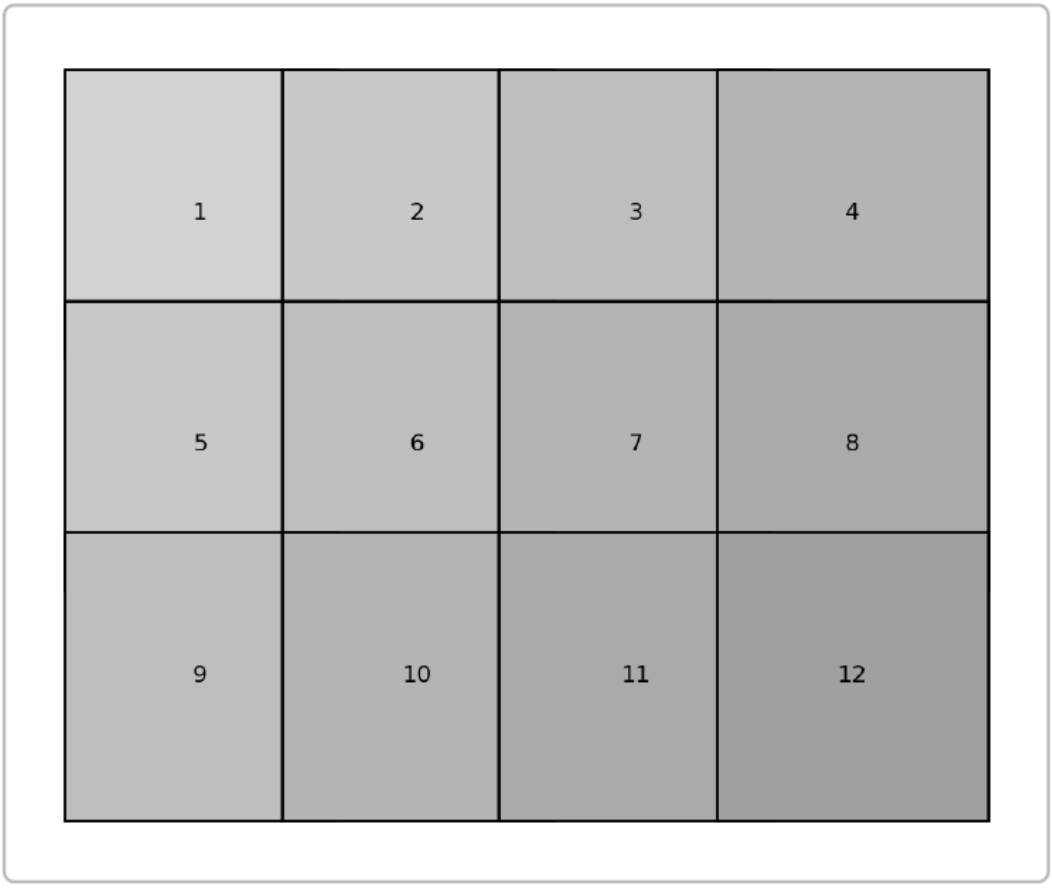


Figure 3.3 Tile layout in a single 2D section (array tomography slice) acquired by SEM. Multiple partially overlapping image tiles (here labeled 1–12) cover the entire tissue section in a grid. During stitching, these tiles are mosaicked together by aligning overlapping content, yielding one contiguous image of the section at high detail.

3.4.2 Importance for 3D Segmentation and Quantification

Stitching in vEM not only expands the FoV but also ensures that quantitative analyses can be performed on complete structures. It serves as the bridge between data acquisition and data understanding, any error in stitching at the 2D level would directly affect the quality of downstream segmentation and measurement accuracy [37]. If there are stitching errors such as a misalignment or a blurred seam running through an ookinete, a segmentation algorithm could mistakenly split one object into two or merge separate objects into one, potentially leading to over-segmentation or under-segmentation of cellular structures [38]. Furthermore, quantification of morphological or volumetric features relies on consistent data [39]. Consider measuring the volume of an ookinete or the surface area of its plasma membrane in 3D. Any small shift or rotation between tiles that is not corrected will propagate into the volume assembly, potentially distorting the shape of the parasite in the 3D model. This could lead to errors in volume calculations or in counting features like the number of micronemes (secretory

organelles) present. Therefore, accurate image stitching in vEM is not merely a technical convenience but a prerequisite for meaningful 3D analysis of the biological specimens.

3.4.3 Challenges in vEM Image Stitching

Stitching vEM datasets is complex and comes with unique challenges and artifacts that can hinder the creation of a seamless mosaic. Some common issues include:

- Stage Drift and Mechanical Inaccuracies: During the acquisition of a tile grid, slight drift or backlash in the microscope stage can cause the captured tiles to be misaligned relative to their nominal positions. Even if the stage is programmed to move in a precise grid, mechanical errors often accumulate, so the overlap between tiles might not match the expected coordinates. If uncorrected, this results in visible misalignments, see Figure 3.4. Furthermore, the process of manually repositioning the sample or vibrations in the instrument can introduce translational or rotational offsets between tiles [40]. As a result, the stitching algorithm must be robust to such shifts, often by detecting the true overlap via image content rather than relying solely on the stage position metadata.

- Non-Uniform Illumination and Intensity Variations: In an ideal scenario, all tiles would have the same brightness and contrast, but in practice illumination can vary across the field or between sequential images. In SEM-based volume imaging, factors like the electron beam profile or detector sensitivity can cause one tile to appear slightly brighter or darker than its neighbor [41].

- Overlapping Tiles and Ghosting Artifacts: By design, tiles overlap by ~5–20% to assist alignment. However, any mis-registration can produce "double images" or blur in the overlap region, commonly called ghosting. Ghosting occurs when features in the overlap do not perfectly superimpose, causing a duplicated or fuzzy appearance of structures in the fused image [42]. Overlaps can also suffer from parallax issues if the section is not perfectly flat – one tile might capture a feature at a slightly different Z height than another. Dealing with overlaps involves both precise alignment (to minimize ghosting) and choosing an optimal blending or seam-cutting strategy to ensure that transitions between tiles appear as smooth as possible.

- Sample Distortions and Physical Artifacts: Biological sample preparation for EM can introduce physical distortions that complicate stitching. Folds, wrinkles, or tears in

serial sections are fairly common in vEM. If such a fold crosses an overlap between two tiles, it disrupts the continuity of image features and can confuse the alignment algorithm [40] [43]. Likewise, knife marks from sectioning or cracks in the section ("breakage") result in missing or displaced content.

- Lack of Reliable Metadata: Modern microscopes record the stage coordinates for each tile and imaging parameters, which can be used for stitching. However, in some vEM situations, especially with older datasets or certain SEM setups, the coordinates of tiles may not be accurately recorded or provided. This absence of positional metadata forces the stitching to rely entirely on image content for alignment. Developing a pipeline for such cases requires strategies to organize tiles e.g., knowing the expected grid layout or using filename conventions and then perform blind stitching using feature matching or correlation across all prospective overlaps [44].

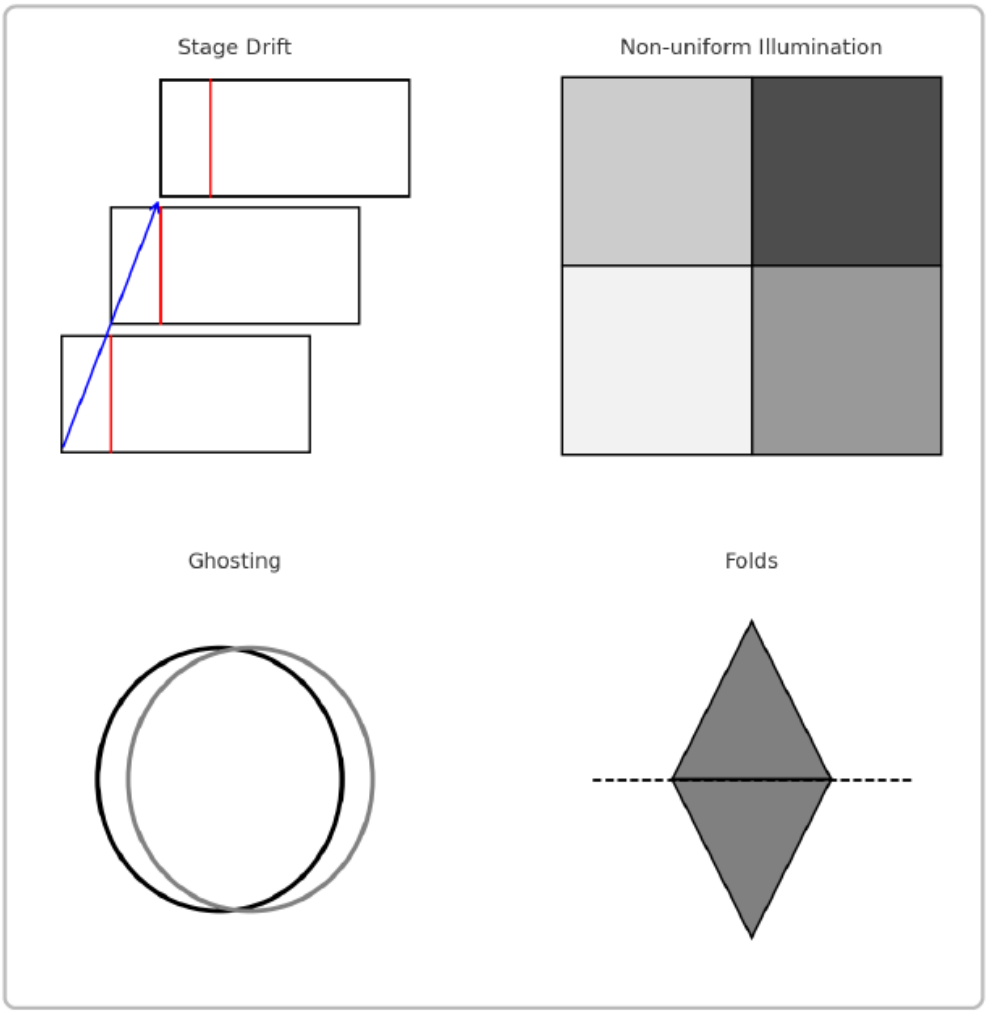


Figure 3.4 Common image stitching artifacts encountered in vEM mosaics. (Top-left: Stage drift) – mechanical drift of the microscope stage between tiles causes misalignment (a vertical

feature, red line, appears progressively shifted in successive tiles). (Top-right: inconsistent lighting or detector sensitivity across the field results in uneven brightness between tiles. (Bottom-left: slight misregistration in overlapping area can duplicate features (same object appear as a "ghost" offset in gray). (Bottom-right: physical folds or tears in serial sections (dashed line indicates a fold) lead to discontinuities or repeated structures in the stitched image.

These challenges make vEM stitching a non-trivial task. In practice, an effective stitching solution for vEM must combine high-accuracy alignment with the ability to compensate for both photometric differences and geometric distortions.

3.4.4 Developing a Stitching Pipeline

Building an automatic stitching pipeline for vEM data offers flexibility to tailor solutions to the specific challenges described above. Python has become increasingly popular for automating stitching workflows, due to its rich ecosystem of scientific libraries and its ability to integrate external tools through scripting. This enables not only the implementation of custom alignment strategies but also the automation of third-party stitching tools. While core stitching algorithms are often implemented in standalone software or platforms like Fiji [45] Python is often used to wrap, control, and scale these tools within automated pipelines. This allows for parameterized execution, batch processing, and easy integration with other image analysis steps. A robust stitching pipeline must accurately align and merge tiles into section mosaics while coping with the real-world imperfections of the data. Leveraging Python in this context enhances control, reproducibility, and workflow scalability— all of which are essential for reliable downstream 3D reconstruction of vEM samples.

3.5 Image Registration

Image registration is the process of aligning two or more images (often of the same scene or object) into a common coordinate system [46]. Typically, one image is chosen as a fixed reference, and the other image(s) are geometrically transformed to overlay and match the reference image. This alignment corrects for any differences in viewpoint, scale, or imaging conditions between the images [47]. Registration is a crucial precursor in many imaging applications, including remote sensing (e.g., aligning satellite images), medical diagnostics (e.g., combining CT and MRI scans), and biological imaging (e.g., reconstructing 3D volumes from serial sections) [48].

In practice, image registration is not a one-size-fits-all process. The specific goal and challenge of registration depend on the imaging context. Below are some of the common use cases where image registration plays a role:

- Across modalities or sensors: aligning images from different imaging modalities (e.g. MRI vs. CT scans) or different sensors so that they share the same anatomical or scene coordinates [49].
- Across time points: aligning images of the same subject or scene taken at different times, for example, before-and-after images or time-lapse sequences to observe changes over time.
- Across viewpoints: aligning images captured from different angles or viewpoints, ensuring that the scene lines up as if viewed from a common perspective.
- Across depth slices: aligning consecutive slices of a 3D image stack (along with the Z-axis) so that the stack forms a coherent volumetric image [50].

In the context of vEM, this last use case Z-axis registration is particularly important which involves aligning a z-stack into a coherent 3D volume. The vEM techniques like AT-SEM produce many 2D slices that must be registered to reconstruct the 3D ultrastructure of a specimen [51]. For example, an EM dataset of an entire cell or tissue volume consists of dozens or hundreds of ultrathin serial sections. Accurate alignment of these sections is crucial for any 3D analysis; misalignments can lead to blurred or doubled structures and are often a dominant source of error in subsequent image reconstructions [52]. It is important to clarify that image registration (as discussed here) is related to but distinct from image stitching. Image stitching typically refers to combining overlapping image tiles side-by-side in the XY plane to create a larger, seamless panorama. Stitching uses registration techniques to align the overlapping areas of those tiles, but its goal is to expand the field of view horizontally. In contrast, the term *image registration* in our context refers specifically to aligning images along the Z-axis (depth). After tiles within a single section are stitched together in the XY plane, the resulting mosaics must be registered across sections along the Z-axis. This Z-axis registration ensures that features line up correctly between consecutive slices.

3.5.1 Image Registration Methods

Registration methods typically aim to bring the sections into correct spatial alignment so that biological structures are preserved across the depth of the volume. The choice of registration method depends on the extent of transformation required and is often categorized based on the transformation model used to align one image to another [53]. Common transformation types include [49]:

- Rigid transformations: preserve distances and angles, allowing only translation and rotation, and sometimes global rotation-scale, if a uniform scale is considered [54].
- Affine transformations: incorporate translation, rotation, scaling, and sharing. Affine mapping can correct for stretching or squeezing of the image and other linear distortions [54].
- Non-rigid (deformable) transformations: allow local distortions and warping to accommodate more complex differences between images. Nonlinear deformation is especially useful when images exhibit local stretches, bends, or other intricate deformations that cannot be captured by a single global affine transform [55].

Early methods to serial vEM image registration were often semi-automated and relied on combinations of global rigid alignment and local warping. A typical image registration pipeline might start by doing a coarse alignment of each section based on maximizing cross-correlation with its neighbors or by manually picking a few corresponding landmarks on adjacent sections. Coarse alignment corrects large shifts or rotations but does not remove stretching or nonlinear bends. To tackle local deformations, elastic or affine transformations can be applied that model slight scaling, shearing, or bending of images to best match adjacent sections [50]. For example, one influential approach introduced by Saalfeld et al. [38] finds a sparse set of corresponding feature points between neighboring sections and then computes an elastic warp that smoothly interpolates those correspondences across the image. This elastic volume reconstruction strategy, implemented in tools like TrakEM2, uses an optimization solver to minimize misalignments while imposing a smoothness constraint to avoid unrealistic distortions [38]. A smooth elastic model may fail in the presence of discontinuities like cracks and other image artifacts. In response, domain-specific knowledge or constraints to guide alignment has been incorporated. For example, methods have been proposed to enforce the continuity of morphological structures across sections so that the algorithm resists warping that would break

a neurite or blood vessel trajectory [51]. Another major wave of innovation leverages machine learning and deep learning for image registration. Self-supervised deep learning has proven especially useful: in one early example, a spatial transformer network was trained to align serial EM images by optimizing a loss function that quantifies image match, effectively learning the features and deformation needed for registration [56]. Recently, Popovych et al. (2024) introduced a comprehensive alignment pipeline that addresses many of the aforementioned challenges with modern techniques [50] In their approach, neural networks are trained to produce matching feature encodings for corresponding points on adjacent sections. In practice, image registration for vEM often involves a combination of tools: initial automated alignment, user inspection and manual tweaking where necessary, and then advanced algorithms for fine alignment.

3.5.2 Challenges in vEM Image Registration

The registration step must be as precise and robust as possible to preserve the true morphology of the sample. However, achieving robust alignment of EM images is challenging due to various artifacts and biological factors. A number of issues can lead to imperfect alignments or "broken" image stacks; examples of some common imaging artifacts are shown in Figure 3.5.

- Physical distortions in the sections themselves are common: ultrathin sections may suffer from folds, tears, or wrinkles when placed on supports, resulting in discontinuities or lost tissue that complicate alignment [57]. Further, each section may undergo slight stretching or shrinking (nonlinear deformation) during processing.

- Intensity and imaging artifacts: there can be intensity and imaging artifacts that confuse alignment algorithms. For instance, one section might have uneven staining or brightness, or contain imaging noise such as streaks from knife chatter, grid lines, or dust particles. These artifacts can produce false matching cues for example; a strong stripe pattern repeated in adjacent sections might trick a naive alignment method into aligning the stripes instead of actual biological features [58].

- Biological variability between sections poses a challenge: the content of consecutive EM slices is not identical, since each 2D section captures a slightly different plane through 3D structures. Some small structures like organelles, or membranes may appear or disappear from one section to the next or change shape. With few obvious landmarks, finding correspondence between sections becomes difficult. [59].

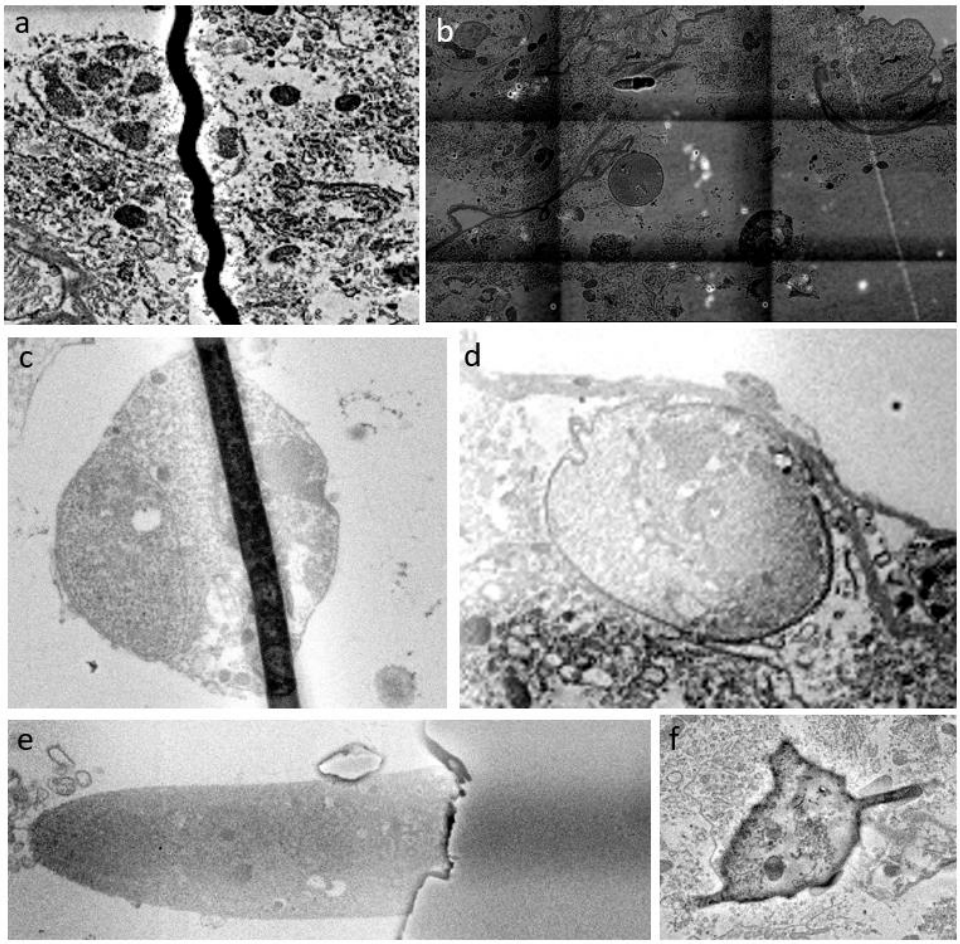


Figure 3.5 Examples of common AT-SEM imaging artifacts. (a) Knife mark: crack (b) Tiling artifacts: grid pattern (c, e) Section fold and tear (d) Uneven brightness (e) Surface contamination: foreign material or dust.

Designing an automated registration pipeline for vEM datasets requires addressing the above challenges. These defects cause tissue loss, making neighboring regions specifically challenging to register. Various strategies can be used, ranging from simple rigid alignment to more flexible non-rigid approaches that are correct for local distortions.

3.6 3D Segmentation

Image segmentation is the process of partitioning a digital image into multiple meaningful regions (segments) to simplify the image representation and facilitate analysis [60]. In practice, segmentation assigns a label to every pixel (or voxel, in 3D) such that pixels of the same label correspond to the same object or region of interest. The result is typically a set of delineated components, for example, distinguishing foreground objects from background, or separating different anatomical structures in a medical scan [61]. This is an indispensable step in many microscopy and imaging workflows because it allows researchers to isolate specific structures

like cells, organelles, tissues and extract quantitative information about them such as volume, surface area, count, and spatial arrangements, etc. It makes the data more meaningful to analyze by breaking a complex image into components, one can focus on each component's properties without distraction from irrelevant features.

In vEM of biological samples, segmentation plays a critical role in understanding ultrastructure. Modern vEM techniques can produce stacks of images where cell organelles and compartments are visible at nanometer resolution throughout a cell. Segmenting these structures allows scientists to create a 3D model of the cell's architecture, identify the location and morphology of each organelle, and quantitatively analyze cellular organization [62]. This yields data on morphology and development of specimen that would be otherwise difficult to obtain from 2D images alone.

3.6.1 Applications and Importance of 3D Segmentation of Ookinetes

Accurate 3D segmentation of the ookinete and its organelles is important for several reasons. First, it enables measurement of the ookinete's shape and size. The ookinete's elongated shape (often ~10–15 μm long) and its curvature can be quantified once the cell is segmented as a 3D object [1]. Any variations in shape, for instance, due to genetic mutations or environmental conditions can be assessed by comparing segmented models. Second, internal structures can be counted and measured. For example, micronemes are secretory organelles clustered at the ookinete's apical end. These structures are crucial for invasion of the mosquito midgut epithelium, they secrete proteins that help the ookinete move and penetrate tissues [63]. Micronemes are among the most abundant organelles in the cell [48][65], and being able to segment and count them in 3D can inform us about the parasite's invasive apparatus. Third, the crystalloid organelle is a unique, transient structure specific to the ookinete and young oocyst stages. It appears in EM as an electron-dense crystalline aggregate of vesicles; segmenting the crystalloid allows to measure its size and position. This is biologically important because the crystalloid is known to be required for the parasite's development into oocysts and eventually sporozoites [66]. Fourth, hemozoin is the malaria pigment, formed by the parasite as it detoxifies heme derived from digested host blood [67]. Segmenting hemozoin in the EM volume would enable measurement of how much hemozoin the ookinete carries and how it is spatially distributed. Finally, the nucleus of the ookinete which contains the parasite's genetic material can be segmented to assess its morphology and volume. Segmenting all these components can yield a comprehensive 3D map of the ookinete's anatomy. Such a map is

invaluable for comparing normal versus experimentally manipulated parasites. For instance, if a gene knockout is suspected to reduce microneme formation, 3D segmentation can provide hard evidence by directly counting micronemes in mutant vs. wild-type cells.

3.6.2 Challenges in vEM Segmentation

Despite its importance, 3D segmentation of vEM data is notoriously difficult. EM images are grayscale with often low intrinsic contrast between different organelles, many cellular structures have similar electron density, making their boundaries hard to distinguish. As one study notes, “the inherent low contrast of electron microscopy (EM) datasets presents a significant challenge for rapid segmentation of cellular ultrastructures” [68].

Unlike fluorescence microscopy where specific labels can make a structure bright against a dark background, EM shows all cellular material superimposed, differing mostly in texture or slight density variations. For example, an ookinete’s cytoplasm is filled with membranes and proteins that create a busy background in which micronemes, or crystalloids must be discerned by subtle differences in texture or pattern. Although specific labeling is possible in EM using techniques like immuno-gold, such methods are typically limited by lower sensitivity, lower spatial coverage, and more complex sample preparation making them less versatile than fluorescent labeling.

The signal-to-noise ratio can also be low; EM images may contain grainy noise that obscures fine details. Densely packed and overlapping structures further complicate the task [69]. In a single 2D EM slice, multiple organelles can overlap in projection, consider a cluster of micronemes: they might appear as a crowded collection of vesicles, some touching or occluding others in the slice. Determining boundaries between such touching objects can be very challenging. A concrete example is given by a recent segmentation effort where a dataset of tightly packed, low-contrast mitochondria in a cell required special tuning to segment properly. The authors reported that the small size and tight clustering of those organelles led to initial under-segmentation (multiple mitochondria merged as one) or over-segmentation (single mitochondrion split into pieces), until their algorithm was adjusted [70]. This illustrates how crowded organelles and weak contrast demand more sophisticated segmentation approaches.

Another issue in serial-section EM is consistency across slices: even after registration, minor misalignments or section loss can cause an organelle to appear discontinuous, which can confuse automated segmentation algorithms that rely on continuity. Likewise, variation in

staining or imaging conditions between slices can cause intensity shifts that throw off simple threshold-based segmentation. If one slice is slightly darker, a static intensity threshold might segment a structure on that slice differently (or not at all) compared to adjacent slices. Thus, segmentation methods must be robust to intensity variation and ideally incorporate some 3D continuity logic so that an object is traced through the volume despite slice-to-slice changes.

Manual segmentation of EM volumes involved tracing each organelle in each slice by hand which is extremely labor-intensive and time-consuming, to the point of being impractical for large datasets [71]. In recent years, advances in machine learning, particularly deep convolutional neural networks (CNNs), have greatly improved the prospects of automated segmentation in vEM. Researchers have started training CNN models to recognize and segment organelles in EM volumes. For example, one pipeline used a CNN to segment seven different subcellular structures (mitochondria, ER, Golgi, etc.) in FIB-SEM (abbreviation) images [72]. The result was a dramatic acceleration in obtaining segmentations, alleviating the manual bottleneck. However, deep learning methods also face challenges: they require annotated training data which itself is a scarce resource because manual annotation is slow, and their performance can degrade if the target dataset differs from the training data in staining or noise patterns [68]. Another challenge is that a network might have difficulty with extremely crowded regions or novel organelles it wasn't trained on [73].

3.6.3 Python-based 3D Segmentation

In developing a segmentation solution in Python, one has access to both traditional image processing algorithms and modern AI techniques. A classical approach might involve filtering the EM images to enhance contrast (for instance, using edge detectors or denoising filters), then applying thresholding or region-growing to segment structures of a certain intensity or texture. Many image processing libraries [74] provide functions for building rule-based segmentation pipelines using operations such as thresholding, morphological filtering, and region labeling. Such a pipeline might, for example, threshold the image to get a preliminary mask of the parasite cell vs. background, then within the cell mask use size filters to pick out large organelles (nucleus, crystalloid) versus small vesicles (micronemes). The advantage of a classical approach is that it is interpretable and does not require training data, but the downside is limited adaptability for example, fixed thresholds might not work for all slices, and hand-tuned filters might fail on slight image changes.

On the other hand, a machine learning-based approach in Python could leverage libraries like PyTorch or TensorFlow [75], [76] to train a 3D model for organelle segmentation. With a trained model, inference on the volume could be very fast and more accurate in capturing organelle shapes, since the network can learn complex features beyond simple intensity differences. Python's ecosystem has high-level tools like Keras [77] which simplify building such models. There are also pre-trained networks available for EM segmentation (for generic structures like mitochondria), which one could fine-tune on the ookinete data. A recent benchmarking platform highlighted that no single segmentation model is best for all EM data, and performance depends on image properties [78] underlining that some experimentation may be needed to find the optimal model and parameters for this specific dataset.

Regardless of the method, accurate segmentation is crucial for downstream quantitative analysis. Often, a combination of approaches yields the best outcome: one might use an automatic segmentation model to get an initial segmentation and then apply some post-processing using traditional image processing. The end result should be a set of labeled 3D objects from which one can compute statistics.

3.7 Zarr: Chunked Multi-Dimensional Array Storage for Large Datasets

Modern bioimaging produces extremely large datasets that challenge traditional file formats. Zarr has emerged in recent years as an open-source format to store chunked, compressed, N-dimensional arrays, specifically designed to handle big data efficiently [79]. Using Zarr, a large 3D image volume can be divided into many smaller *chunks*, each stored separately often as its own file or object along with lightweight metadata in JSON (JavaScript Object Notation) format [80]. This chunking strategy enables flexible, scalable storage: users or programs can read or write only the needed chunks of the dataset without loading the entire file into memory. These chunks can be compressed to reduce storage size and I/O overhead, using a compression codec and configuration defined at the array level. The result is that even petabyte-scale arrays can be accessed and analyzed with interactive speeds, as only a subset of the data is accessed at any one time [81].

Traditional bioimage formats such as Tagged Image File Format (TIFF) and more advanced formats like Hierarchical Data Format (HDF5) provide mechanisms for storing large datasets. In TIFF files, chunking is typically implemented as 2D tiles, allowing access to individual frames in time-lapse sequences or slices within a 3D volume. HDF5 offers greater flexibility, enabling users to define chunks across multiple dimensions [82]. However, both formats

generally store data in single, monolithic files that assume fast and reliable random access on a local file system. This assumption becomes a limitation when working with cloud storage or distributed computing environments, where even small read operations can incur significant overhead due to the need to retrieve portions of a large file [82]. In contrast, Zarr's design (many small chunk files) is naturally suited for remote and parallel access. By splitting a dataset into many pieces, it avoids the need to download or parse a single gigantic file for every operation. As [83] note in the context of volume electron microscopy, chunking "enables streaming-based access" to large datasets, which is ideal for visualization and processing on demand [84]. For example, if a single slice or a specific sub-volume is needed to examine, Zarr will fetch just the relevant chunk files from storage, see Figure 3.6. This avoids loading the full volumetric image into memory, saving time and resources. Moreover, the chunk size and layout are configurable, allowing researchers to optimize the balance between read/write throughput and memory usage for their specific access patterns. These features make Zarr a scalable solution for large image data. Researchers in fields from genomics to geospatial and astrophysics have adopted Zarr for managing big arrays [82].

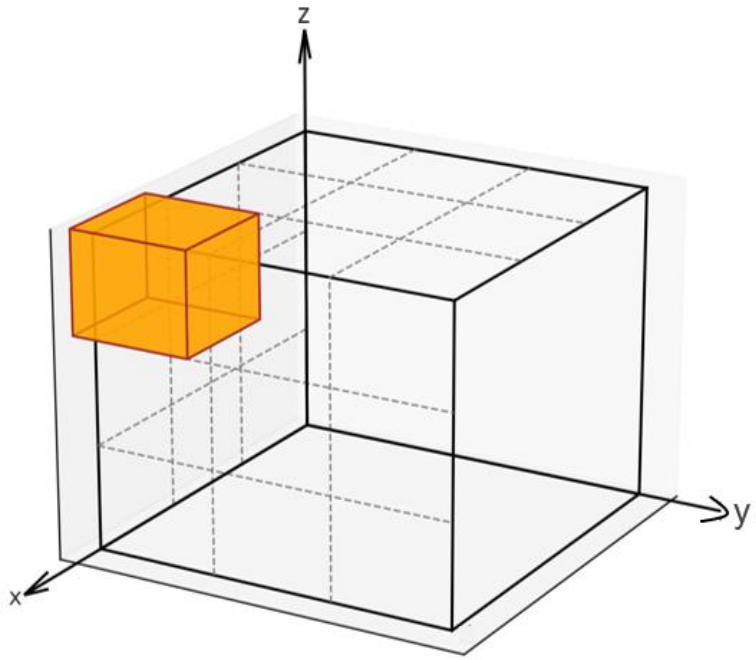


Figure 3.6 Cartoon illustration of chunking a 3D image volume with Zarr. The full volume (black outline) is subdivided into many smaller blocks or "chunks." Here, one chunk is highlighted (orange) and can be accessed independently.

3.7.1 OME-Zarr: A Next-Generation Bioimage Format

Leveraging the Zarr format as a foundation, the Open Microscopy Environment (OME) 0introduced OME-Zarr in 2021 as part of a "next-generation file format" (OME-NGFF) initiative for bioimaging data [85]. The goal of OME-Zarr is to address the scalability, interoperability, and metadata challenges inherent in modern microscopy datasets [86]. It builds upon Zarr's chunked storage to support the complex needs of biological imaging, following FAIR data principles (findable, accessible, interoperable, reusable) [87]. In practice, OME-Zarr acts as a container format tailored for microscopy: it can hold not only the raw multi-dimensional pixel data but also multiple related images (e.g., multichannel acquisitions or label masks), organized metadata, and even annotations, all under one hierarchy [86]. This design was driven by a broad community effort to create a common format that works for diverse imaging modalities from light microscopy to volume EM and analysis workflows. By adopting open standards and a community-driven specification, OME-Zarr aims to ensure that bioimaging data can be easily shared between different tools and institutions without conversion [87].

Key features of OME-Zarr extend the metadata capabilities of Zarr to meet bioimaging needs. First, it supports *multiscale image pyramids*: one can store multiple resolutions of an image (full resolution and down-sampled levels) within the same dataset [88]. This is conceptually similar to how Google Maps allows smooth zooming: image-viewers-tool like Napari [89] can first load a low-resolution overview and then retrieve only the necessary high-resolution chunks when the user zooms in. Since each resolution level is itself stored as independently chunked data, this enables efficient, on-demand access to the parts of the image, improving interactivity when working with big datasets [89]. Second, OME-Zarr uses a hierarchical directory structure to organize data [88]. Images are stored in *groups* (directories in the Zarr store) that can contain nested subgroups or arrays; for example, a top-level group might represent an entire experiment or multi-well plate, with subgroups for each sample or image position, see Figure 3.7. This hierarchy not only keeps related data together but also allows storing metadata at each level of the hierarchy in human-readable JSON files. Crucially, this means all relevant information such as imaging settings, sample identifiers, spatial scales, etc. travels with the dataset in a standardized way [89]. OME-Zarr's metadata model is built for flexibility and growth: unlike OME-TIFF which has a single XML schema, OME-Zarr can incorporate multiple metadata schemas side by side, for instance, microscope hardware info, analysis results, and quality control metrics can all be included. This ensures that as experiments become more complex,

the format can still capture the necessary context without forcing everything into one rigid schema.

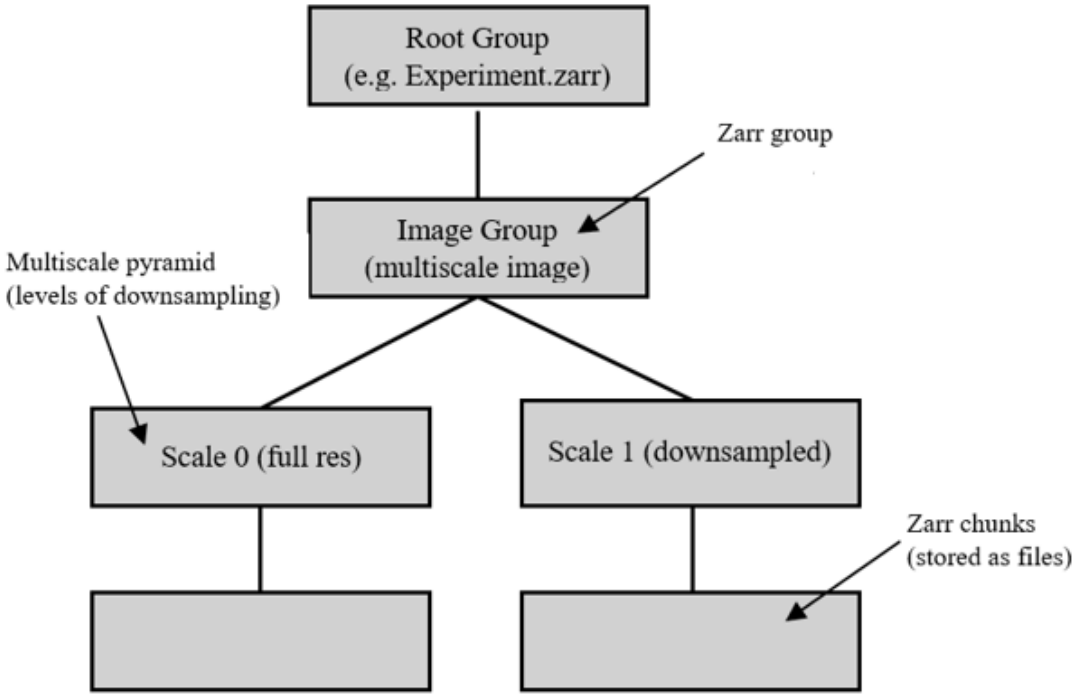


Figure 3.7 Illustration of OME-Zarr data organization. In this schematic, a root group contains an Image Group (a multiscale image), which in turn has multiple resolution levels (Scale 0 = full resolution, Scale 1 = down-sampled). At the lowest level, each scale consists of many chunk files storing the pixel data. JSON metadata (not depicted) can be stored at any level of the hierarchy (root, image group, or scale).

Another important capability is the inclusion of label images and regions of interest (ROIs) directly within the OME-Zarr dataset [88]. Scientists often generate segmentation masks, annotations, or derived quantitative images as part of analysis. Traditional formats often require separate files for these e.g., a TIFF for the raw image and another file for the mask, which can lead to data management headaches. OME-Zarr allows storing these derived data layers in the same container as the raw images, maintaining a clear link between them. For example, a segmented image volume can be saved as an additional multiscale image in the OME-Zarr hierarchy next to the original image, and ROI shapes or tables of measurements can be attached as well. This capability fills a gap left by formats like TIFF or HDF5 which lack standardized support for such rich metadata and annotations. By supporting these use cases, OME-Zarr fosters interoperability — imaging results and their annotations remain packaged together and reusability, since anyone who opens the dataset has immediate access to all relevant data layers.

3.8 Nextflow (a Workflow Management Tool)

Nextflow, a modern workflow management tool is designed to organize and automate computational pipelines, allowing researchers to easily write data-intensive workflows by chaining multiple analysis steps together [90]. It was originally developed to tackle common problems in big-data science, for example, the need for analyses to be reproducible and portable across different computing environments. By using Nextflow, one can focus on the analysis logic while the software handles the execution of each step in the correct order, which greatly simplifies the management of a multi-step pipeline [91].

A Nextflow pipeline has two fundamental building blocks: processes and channels. It enables orchestrating complex image processing workflows by queueing data in channels that are then fed to process blocks, see Figure 3.8. Each process block represents a single step of the workflow, for example, running a Python script or a command-line tool, and it declares what inputs it needs such as files or parameters and what output it will produce. These processes are isolated from each other (they do not share memory or state), and the only way they pass data along is through channels [92]. In practical terms, this means that one process's output is sent via a channel to become the input of the next process. Nextflow uses these input/output relationships to determine the execution order implicitly: a given step will run only when all its declared inputs are available [93]. This data-driven approach frees the user from manually scheduling tasks, ensuring that each step runs in the proper sequence. Independent steps can also run in parallel, as illustrated by task 1, task 2, and task 3 in Figure 2.8. This parallelism can speed up the analysis without requiring additional effort from the user.

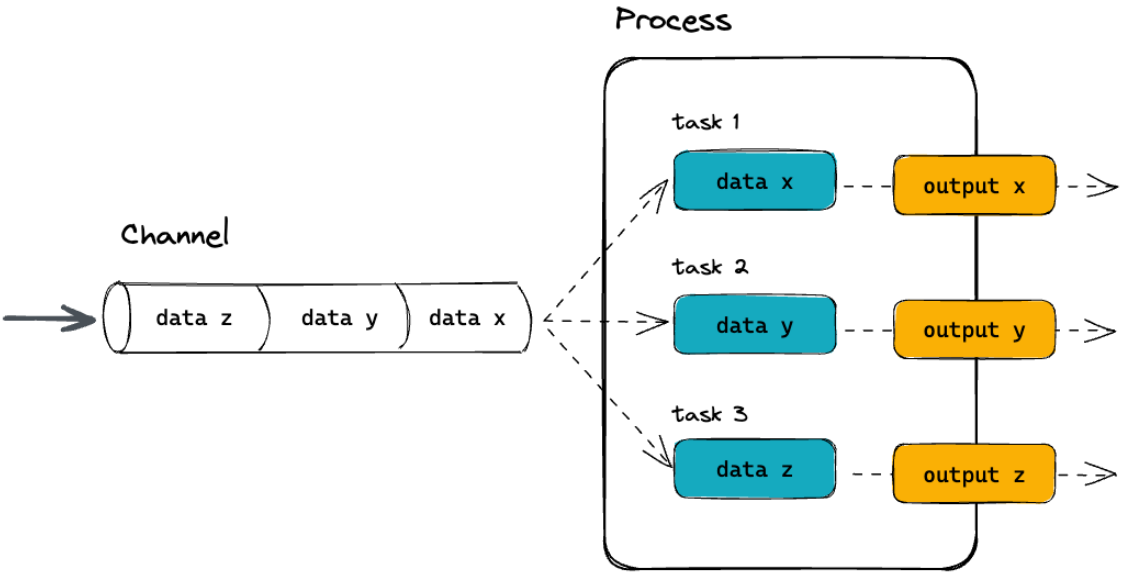


Figure 3.8 This schematic of how data flows through a Nextflow pipeline. A channel delivers a stream of input data elements (e.g., data x, y, z), which are consumed by a process. Each input triggers a separate task execution, running independently and producing corresponding outputs. Figure adopted from *Genomics Aotearoa Introduction to Nextflow Workshop* website. [94]

Using Nextflow makes the pipeline easier to run, reproduce, and maintain overtime. All the steps and their connections are explicitly documented in the Nextflow script, which means anyone with the pipeline file and the required software environment can rerun the entire analysis and obtain the same results [93]. This built-in clarity contributes to more reproducible research, the analysis can be repeated or adapted reliably, since the workflow will always execute the steps in a consistent manner. Maintaining the pipeline is also more straightforward: if a script is updated or a new processing step is added, the Nextflow workflow definition and configuration can be modified without disrupting the overall pipeline structure [93]. These features make Nextflow a powerful tool for streamlining analysis pipelines and ensuring they are easy to reuse as the project evolves.

4 Materials and Methods

The analysis pipeline is designed to handle the full workflow, from raw image data to structured, quantitative outputs including stitching, 3D reconstruction, registration, segmentation, and final measurements. It was implemented in Python, making use of open-source libraries for image processing, data handling, and workflow automation, as well as publicly available tools. Development and testing were carried out on a local workstation, with fallback options available through CSC, the Finnish IT Center for Scientific Computing. The focus was on identifying and integrating the most suitable open-source solutions into a single, efficient, and reproducible workflow, rather than developing new algorithms from scratch. The following sections of this chapter describe the technical framework of each stage of the pipeline in detail.

4.1 About Datasets

The volumetric datasets used in this study were acquired through AT-SEM and represent different developmental stages and tissue contexts of *P. falciparum* ookinetes. Sample preparation and image acquisition were performed by project collaborators, as outlined in the introduction chapter. Each dataset was received as a separate folder labelled by its acquisition date (e.g., data-2024-07-26), as shown in Table 4.1. These datasets include two sample types: blood bolus and midgut tissue and span three parasite developmental stages: early (20 hours post-infection, hpi), intermediate (24 hpi), and mature (30 hpi). Detail image volumes were acquired at a 2.5 nm XY pixel size with 90 nm Z-spacing for both sample types. However, for gut samples, additional image volumes were acquired at a 10 nm XY pixel size (with the same 90 nm Z-spacing) to provide a broader tissue-level view. The five datasets, together, totaled approximately 594 GB (see Table 4.1 for a summary of the raw datasets).

Table 4.1. Represents the summary of datasets of P. falciparum ookinetes acquired through AT-SEM

Dataset	Developmental Stage	Type	Size (GB)
data-2024-03-20	intermediate	Gut	132
data-2024-07-26	intermediate	Blood	106
data-2024-09-10	early	Blood	109

Dataset	Developmental Stage	Type	Size (GB)
data-2025-03-17	mature	Gut	163
data-2025-03-25	mature	Gut	84.1

Each dataset contained a large number of raw 2D grayscale TIFF images, representing tiled fields of view from serial ultrathin sections. These tile images were named systematically to encode their spatial positions in the imaging grid, with file names indicating the tile's row, column, and z-slice index, see Figure 4.1 (a). Tiles were acquired with an intentional overlap of ~15-20% in X and Y to ensure coverage of the sample; an example raw tile mosaic is illustrated in Figure 3(b). Adjacent tiles thus shared overlapping content, which aids alignment but requires handling of redundant regions. The data was not accompanied by metadata files such as stage coordinates, meaning that all positional information had to be parsed directly from the filenames. Additionally, some tiles were missing due to acquisition gaps or out-of-focus regions, resulting in empty areas in the grid, Figure 3(c) schematically highlights the missing tiles region. The full dataset spanned tens to hundreds of z-slices, forming deep stacks that required assembly into coherent 3D volumes, see Figure 3(d).

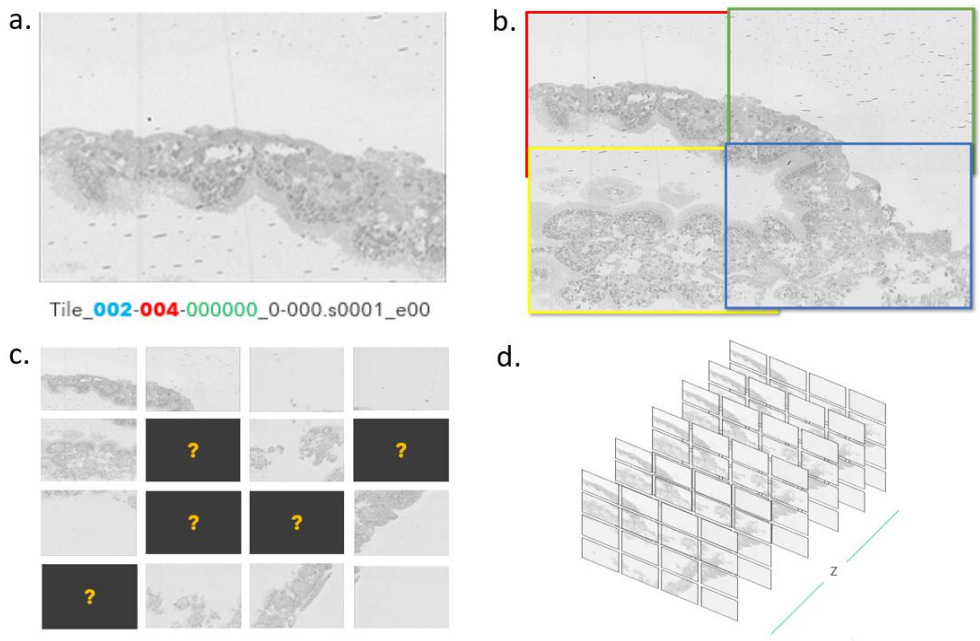


Figure 4.1 Overview of the raw tile structure in the vEM datasets. (a) Example of a single tile with its filename encoding the row (blue), column (red), and slice (green) position. (b) A raw tile mosaic showing variable overlaps in X and Y directions. (c) A tile grid with missing tiles (indicated by black boxes with "?"). (d) A conceptual illustration of the full 3D dataset as a stack of 2D unstitched tile mosaics arranged along the Z-axis (Z-slices).

These raw image tiles served as the input to our automated image-stitching and 3D reconstruction pipeline, described in the following sections.

4.2 Image Stitching Implementation

To reconstruct 3D volumes from tiled images of *P. falciparum* ookinetes, we developed an automated image-stitching pipeline. The pipeline automatically takes the above-described raw tile images and produces stitched 3D image stack in OME-Zarr format, as detailed below.

4.2.1 Data Parsing and Preparation

The first processing step was to parse the raw tile files and reorganize them into a consistent directory structure. To achieve this, a custom Python script was executed by a Nextflow process. This script reads each TIFF file's name to extract its slice number and tile coordinates. It then created a subdirectory for each slice and placed symbolic links to the original TIFF tiles into the appropriate slice folder. This way, all tiles belonging to same slice were collected in their corresponding subdirectory. Using symbolic links avoids duplicating the large image files

while still organizing them for easy access. The directory name or link naming often included the row/column indices so that the spatial layout could be inferred from the file system. As a result, after this step each slice directory contained a grid of tile images (with overlapping content) ready for stitching.

4.2.2 Handling Missing Tiles

Some tiles were intentionally omitted from the raw data grid during acquisition, either to reduce acquisition time or because they fell outside the region of interest, resulting in gaps in the dataset. To address this, the pipeline included a Fill-Missing-Tiles step in the stitching pipeline. For each slice directory, the script determined the expected grid extent by finding the maximum row and column indices from the present files. It then scanned the grid and identified any missing tile positions. For each missing (row, column) location, the script generated a blank image (all zeros) as a placeholder, ensuring that the grid of tiles became a complete rectangle. This filling step prevents the stitching software from failing due to gaps. The script also automatically grouped tiles into contiguous regions: it treated all tiles that are edge-adjacent, for instance, touching at least one side as belonging to the same connected block. In practice, if missing tiles break a slice's mosaic into separate "islands" of tiles, each island is detected and processed independently. Grouping contiguous tiles is useful because stitching is more reliable when applied to a connected block of images. The output of this process is, for each slice, tiles are grouped into one or more subfolders called "regions", each containing a complete grid of tile images, including filled-in blanks where necessary. These regions represent contiguous blocks of tiles ready for stitching in the next step.

4.2.3 Tile Stitching

Each contiguous tile region was then stitched into a single image. We automated Fiji's Grid/Collection stitching plugin via a Python wrapper script, the plugin was supplied with tile coordinates extracted from the file names, along with the specified overlap percentage. The 'Linear blending' fusion method was selected to ensure smooth merging of overlapping regions. As the Fiji documentation notes, "Linear blending will obscure the seams between tiles", which helps produce a seamless mosaic [45]. The stitching plugin computed the optimal translations between overlapping tiles and blended their intensities. The result for each region is one high-resolution mosaic image per slice.

4.2.4 Volume Assembly and OME-Zarr

After stitching, the 2D slice mosaics were assembled into a 3D volume. A Python script was used to collect all stitched slice images in the correct order and stack them into a single volumetric dataset, preserving the spatial relationships along the Z-axis. The final volume was written and saved as OME-Zarr file, suitable for scalable visualization and downstream processing.

4.2.5 Workflow Orchestration (Nextflow)

All steps above described were combined into a unified Nextflow workflow. We defined each stage, which includes StoreTileData, FillMissingTiles, StitchTiles, and AssembleStack, as a separate Nextflow process and chained them in the main workflow. This ensures proper execution order, for instance, stitching is triggered only after the missing-tile filling process is completed. It automatically parallelized independent tasks: for instance, stitching of different slices or regions ran in parallel to speed up processing. The Nextflow pipeline orchestrates the entire image-stitching procedure from raw files to the final OME-Zarr volume in a reproducible way. Figure 4.2 provides a pictorial summary of all steps involved in the image stitching pipeline.

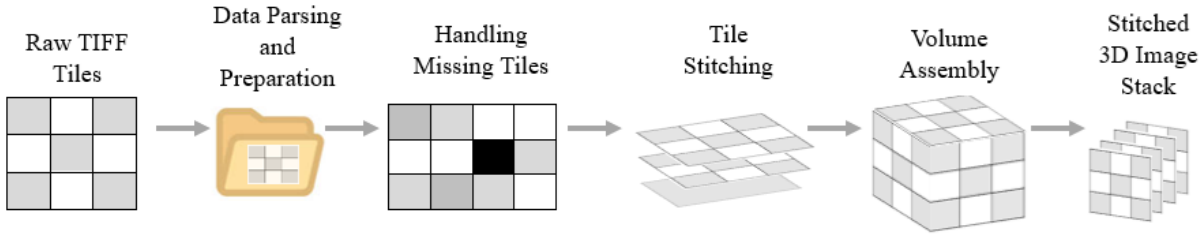


Figure 4.2 Schematic workflow of the automated image stitching pipeline.

4.3 Volume Cropping

Volume cropping served as a crucial intermediate step, aiming to reorganize and isolate specific regions within the large stitched stacks. The full stitched 3D EM volume stored in OME-Zarr format was first opened in the Napari viewer using napari-ome-zarr plugin for interactive inspection of each 3D stack. For each parasite, a single 2D bounding box was manually drawn using Napari's Shapes layer and adjusted as needed to ensure it covered the parasite across all Z-slices in the stack. This approach also accounted for the translational and rotational variations between slices, as parasites could shift in position or orientation along the Z-axis. The bounding

boxes were drawn larger enough to compensate for misalignment between slices, ensuring that the region of interest (ROI) remained fully enclosed throughout the stack. This is illustrated in Figure 4.3 (a and b), which shows the same stitched gut dataset slices at different depths, the size of the bounding box reflects the need to compensate for translational and rotational misalignments across slices. Figure 4.3 (c) shows another example from a blood sample stack where four parasites (ookinetes) were identified. For automated volume cropping, the coordinates of these bounding boxes were saved in a tabular format — CSV (Comma-Separated Values) files for each stack.

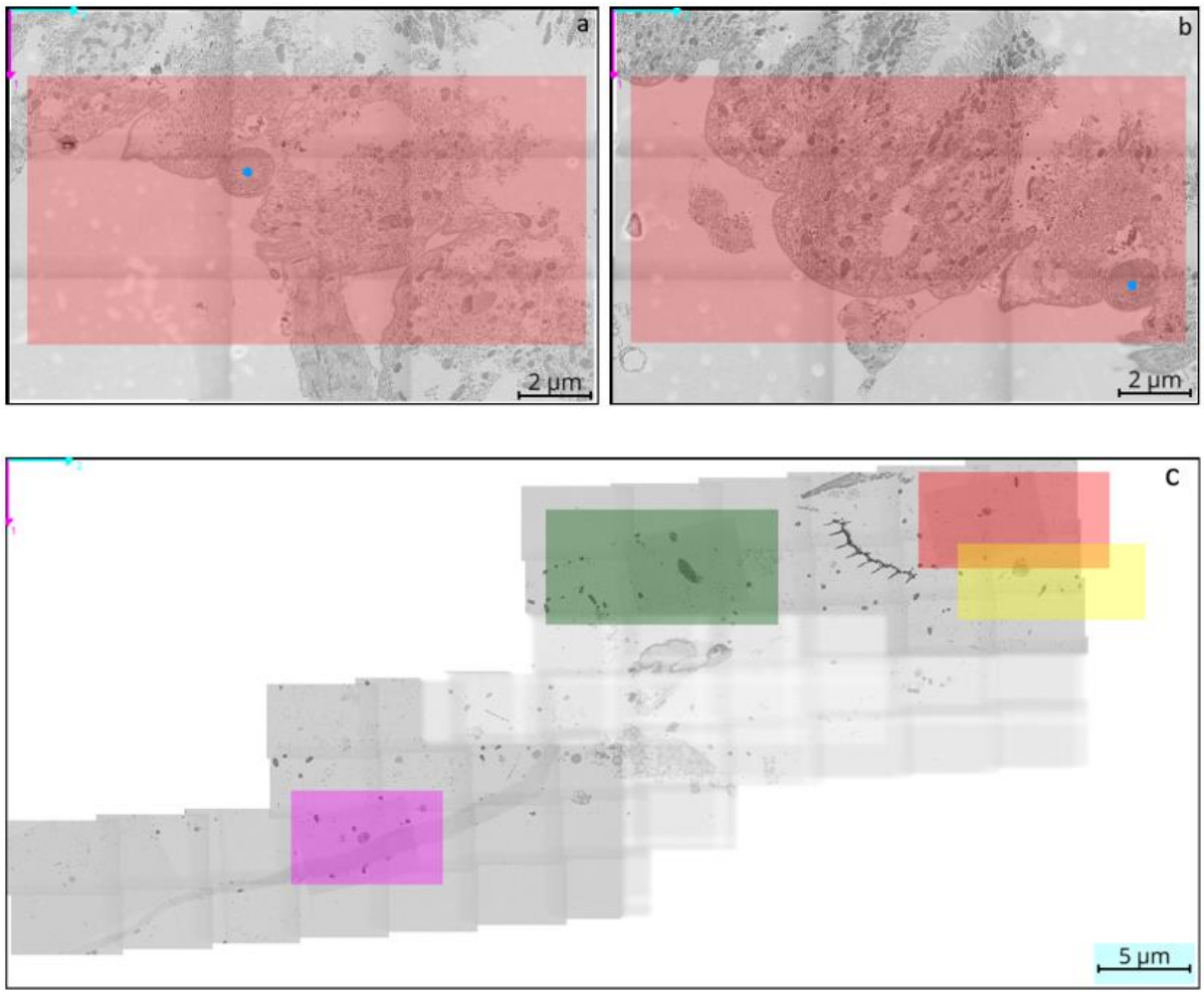


Figure 4.3 Manual annotation of parasite ROIs in stitched EM stacks using Napari. (a–b) Two different slices from the same stack showing the red bounding box enclosing the same parasite (marked with a blue dot) at different Z-positions. (c) A stitched blood dataset showing four distinct ookinetes annotated using separate-colored bounding boxes (red, yellow, green, and magenta) to mark their individual ROIs.

A custom Python script was used to crop out each ROI from the full volume according to the saved bounding boxes co-ordinates. In practice, the script iterated over all z-slices in the box,

extracting the 2D image for each slice and stacking them into a 3D array for that ROI. Each cropped ROI volume was then saved in two forms:

a) as a new OME-Zarr sub-dataset (using the same chunking scheme as the original).

b) as a series of 2D TIFF images (one file per slice) written with the Python tifffile library [95] for maximum compatibility.

Writing to TIFF prepares the data for semi-automatic tools; for example, the Fiji TrakEM2 [96] plugin used later for slice alignment natively accepts TIFF stacks. In parallel, the cropped OME-Zarr datasets were used to explore and test other image registration approaches.

The volume cropping step implemented in Python with open-source tools efficiently extracted each parasite ROI and converted it into formats optimized for downstream analysis and image registration.

4.4 Image Registration Implementation

To correct Z-axis misalignments in the volumes, we initially attempted a fully automated registration pipeline. Each dataset consisted of a stitched 3D stack (a series of 2D sections) cropped around an individual parasite. Initially, we explored fully automated Python-based approaches using multiple existing standard open-source libraries. These included methods based on phase cross-correlation, SimpleITK registration framework [97] and feature matching with ORB (Oriented FAST and Rotated BRIEF), implemented with tools such as scikit-image, SimpleITK, Dask, and Zarr. The custom Python scripts attempted to estimate shifts or transformations between adjacent slices using intensity-based or feature-based similarity. In addition, various preprocessing strategies like Gaussian filtering, or contrast inversion were tested to improve performance.

In principle, these methods can register slices by matching image intensities or sparse features. In practice, however, our ookinete data proved challenging. The serial EM sections often contained large, irregular displacements between slices, locally varying rotations, and artifacts such as wrinkles or folding of the tissue. Contrast differences and biological changes from one section to the next further confounded the matching. In effect, the assumptions of simple cross-correlation or feature-based registration were violated by the complex distortions in the data. As noted in the literature, physical damage and deformation in long serial sections make alignment by standard algorithms very difficult [98]. Our automated routines frequently

converged on incorrect alignments or failed entirely, yielding obviously mis-registered slices that could not be fixed by parameter tuning alone.

Because the fully automated methods could not reliably align our stacks, we switched to a semi-automated strategy using the TrakEM2 plugin in Fiji (ImageJ). TrakEM2 was explicitly designed for large-scale volume EM reconstruction [99]. It provides an interactive interface for aligning serial sections, combining automatic alignment tools with options for manual adjustment using landmarks. In our workflow, each individual parasite stack (26 stacks in total) was imported into TrakEM2. We used the default alignment settings, which provided good initial results across most slices. TrakEM2's algorithms completed most of the alignment automatically, but we carefully examined each volume and intervened wherever misalignments remained. In slices affected by severe artifacts, for example, folds, tears, or staining gradients, we manually correct misalignments. These manual corrections allowed us to "nudge" problem sections into proper register. The interactive TrakEM2 approach let us combine algorithmic alignment with human oversight, directly addressing the slice-specific artifacts that the Python methods could not handle.

For each stack, we repeated the automatic fitting and manual refinement until no visible misregistration remained. We then exported the aligned volume as a single registered TIFF stack. The output TIFF volumes preserved the original image intensities and geometry, and these well-aligned stacks were then used for all downstream segmentation and analysis. However, a more detailed comparison of alignment quality (before vs. after registration) is shown in the Results chapter.

4.5 3D Segmentation and Quantification

Following the image registration process, the 3D EM stacks exhibited improved alignment, which facilitated the delineation of more precise bounding boxes around each parasite (ookinete). Initially, due to significant translational and rotational variations, larger bounding boxes were necessary to encompass the entire parasite. However, post-registration, the enhanced alignment allowed for the use of tighter bounding boxes without compromising the inclusion of relevant structural details. Napari was used again to draw tight rectangular ROIs around each parasite and the coordinates of these ROIs were saved as CSV files using Napari's Shapes layer. A custom Python script then applied these coordinates to crop the registered TIFF volumes, resulting in significantly smaller sub-volumes, typically under 6 GB per parasite. This post-registration cropping step mirrored the earlier volume cropping procedure, further

reducing data size and focusing on the parasite regions of interest as a preprocessing step before 3D segmentation.

For the segmentation step, the goal was to extract each parasite's full body and its ultrastructural organelles from the volume stacks. Initially, we attempted automated segmentation of the full parasite body using Meta AI's Segment Anything Model (SAM) [100] by applying a direct Python implementation adapted from the official SAM repository to each 2D slice of the registered cropped stacks. SAM successfully identified parasite outlines in many of the central slices; however, its performance diminished near the top and bottom of the volume and on slices affected by artifacts. This decline was attributed to artifacts present in the slices of the stack and genuine changes in parasite morphology along the Z-axis. These limitations of SAM are evident in our preliminary outputs and will be illustrated in the Results chapter.

Subsequently, we explored Napari-based semi-automatic tools for the 3D segmentation. First, we employed the napari-SAMV2 plugin [101], which integrates SAM v2.1 into Napari and is tailored specifically for 3D volumetric data. By placing a few positive "point" clicks on each parasite in Napari, the plugin propagates the labels through the stack and generates a full 3D mask of the ookinete. It requires minimal user input for adjustments particularly for damaged slices and generates acceptable masks for most slices. Napari-SAMV2 was effective at capturing the overall parasite shape, generating complete binary masks of each ookinete. However, its segmentations were relatively coarse and did not reliably capture fine organelle structures within the parasite.

For parasite ultrastructure, we adopted another semi-supervised approach using the napari-sam plugin which supports interactive 3D annotation [102]. This interactive plugin also leverages SAM but supports semi-automated annotation of ultrastructural features. Together with this plugin and manual annotations, label masks for five classes that are micronemes, hemozoins, nucleus, crystalloid and whole parasite were generated. Each of the 26 ookinetes had two sets of masks: a whole-parasite mask from napari-SAMV2 and organelle masks from napari-sam with manual annotation and corrections where needed. All segmentation results were saved as TIFF volumes (parasite and organelles separate) for downstream quantitative analysis. Pictorial examples of these segmentations including the earlier failures of the direct SAM approach and the improved masks from the Napari plugins will be presented in the Results chapter.

A Nextflow pipeline was implemented to standardize and automate the quantification of ultrastructural features from each segmented 3D parasite volume. The pipeline first converted

each TIFF image stack and its segmentation masks into the OME-Zarr format. This ensures that all data (raw images and labels) have consistent organization and metadata.

After conversion, custom analysis scripts were executed on each sample's labelled structures. The scripts counted individual micronemes and calculated their volumes by summing voxel counts, then converting to physical units in nm^3 and μm^3 using the known voxel dimensions. A 3D Euclidean distance transform [103] from the segmented parasite surface was applied to compute the distance of each microneme from the cell membrane. Similarly, the volumes of the segmented hemozoin crystals were also computed (in nm^3 and μm^3). For each sample, these computed measurements including microneme count, microneme volumes, microneme–membrane distances, distance of micronemes to its nearest micronemes and hemozoin count and volumes were saved to a CSV file. Measurement routines for other structures such as crystalloid body, nucleus and overall volume of ookinetes also follow the same volumetric and spatial analysis.

In practice, this automated pipeline processes all 26 parasite volumes in the same standardized way, yielding organized CSVs output files for each sample. The preliminary quantitative results and data visualizations such as tables and plots of counts and sizes will be described in the Results chapter.

5 Results

This chapter presents the outcomes of the image processing pipeline developed during this study, following the key stages from initial image stitching to the final quantification of *P. falciparum* ookinetes and their ultrastructural features. Each section highlights how the pipeline handled and processed large vEM datasets and turned them into organized biological data.

5.1 Stitching

In this step of the analysis pipeline, five large volumetric microscopy datasets totaling approximately 594 GB of raw TIFF tile images were processed to generate stitched 3D image stacks for analysis. An automated pipeline, fully described in Chapter Three, was built to perform tile stitching and 3D volume assembly.

Table 5.1 presents a summary of the stitched dataset outputs, including the number of 3D stacks generated per dataset, the number of parasites (ookinetes) present per stack, the number of z-slices per stack, and the dataset sizes before and after stitching. Each experimental dataset comprises multiple stitched 3D stacks ranging from 3 up to 6 stacks per dataset, see Table 5.1. Most stacks typically contained 1-2 ookinetes per volume, whereas the stacks of the blood dataset *data-2024-07-26* contained up to four ookinetes in a single stack. Likewise, the depth of each z-stack varied from as few as 9 slices in the smallest volumes to as many as 94 slices in the largest. In terms of data size, raw input TIFF tile images ranged from ~84 GB to ~163 GB. After stitching and conversion, the corresponding OME-Zarr output volumes were between ~23 GB and ~141 GB, demonstrating a noticeable reduction in size due to OME-Zarr compression feature while retaining all structural information.

Table 5.1. Summary of the five microscopy datasets after performing image stitching.

Dataset	No. of 3D stacks	No. of Ookinetes per Stack	No. of 2D Slices per Stack	TIFF Input Size (GB)	OME-Zarr Output Size (GB)
data-2024-03-20	4	1, 1, 1,1	41, 94, 38, 52	132	93.4

Dataset	No. of 3D stacks	No. of Ookinetes per Stack	No. of 2D Slices per Stack	TIFF Input Size (GB)	OME-Zarr Output Size (GB)
data-2024-07-26	3	4, 2, 2	35, 35, 35	106	62
data-2024-09-10	6	1, 0, 2, 2, 2, 1	9, 9, 9, 9, 9, 9	109	23.2
data-2025-03-17	3	2, 1,1	49, 49, 49	163	140.6
data-2025-03-25	3	1, 1, 1	49, 49, 49	84.1	69.7

To illustrate the results of the image stitching process, Figures 5.1, 5.2, and 5.3 present a couple of outputs from both types of datasets: blood and gut. Figure 5.1 shows a stitched 2D slice from a blood bolus sample, clearly demonstrating the assembly of the overlapping image tiles. Figure 5.2 displays a stitched 2D slice from a midgut dataset, showcasing a more densely packed region with greater cellular detail. Finally, Figure 5.3 provides a 3D rendering of one of the stitched volumes, offering a perspective view that highlights the Z-axis depth across the optical slices. These images collectively demonstrate the effectiveness of the automated stitching pipeline in producing high-quality volumetric reconstructions across both sample types.

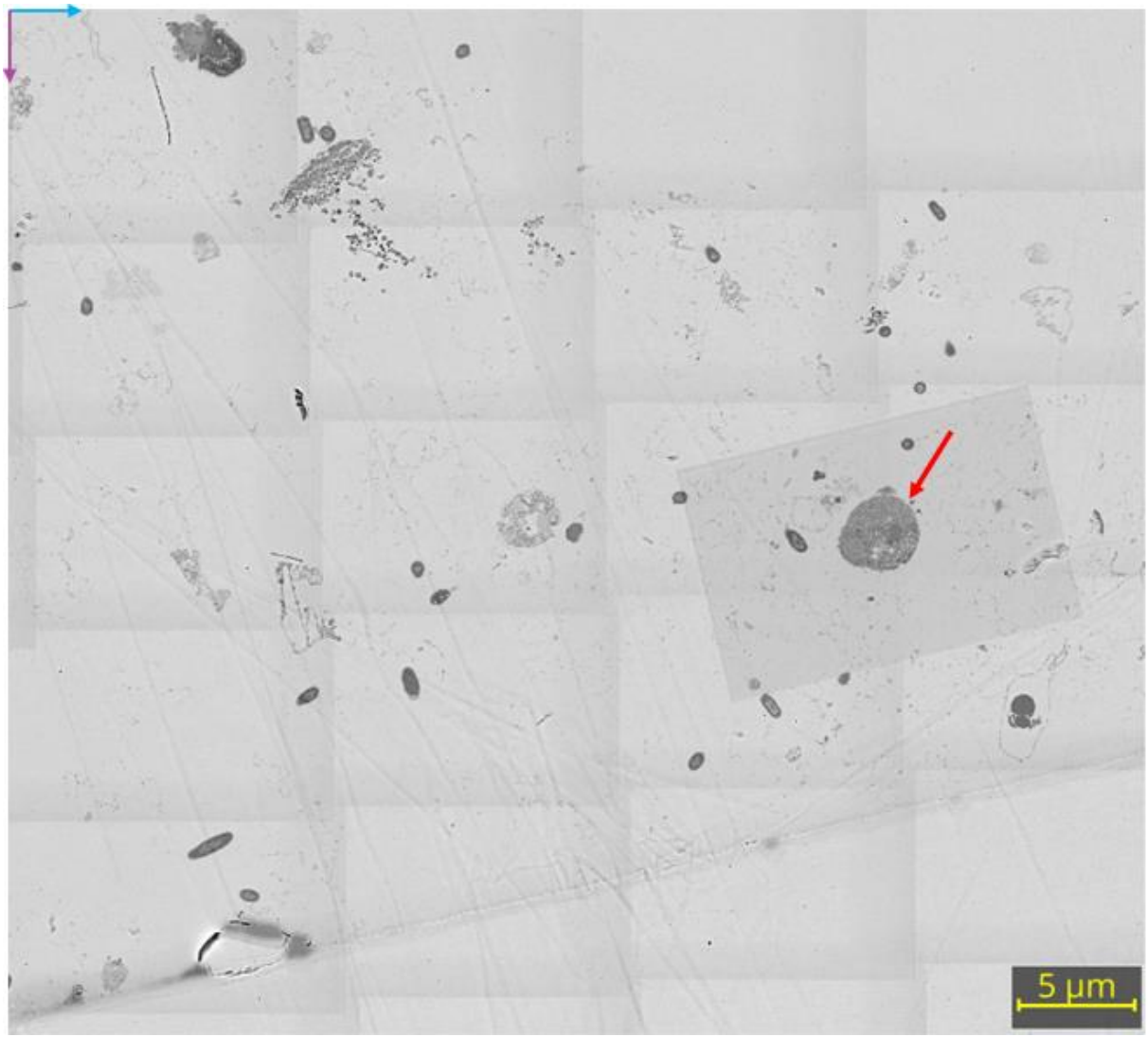


Figure 5.1 Example of a stitched 2D slice from a blood bolus dataset, showing smooth tile alignment and parasite presence (red arrow).

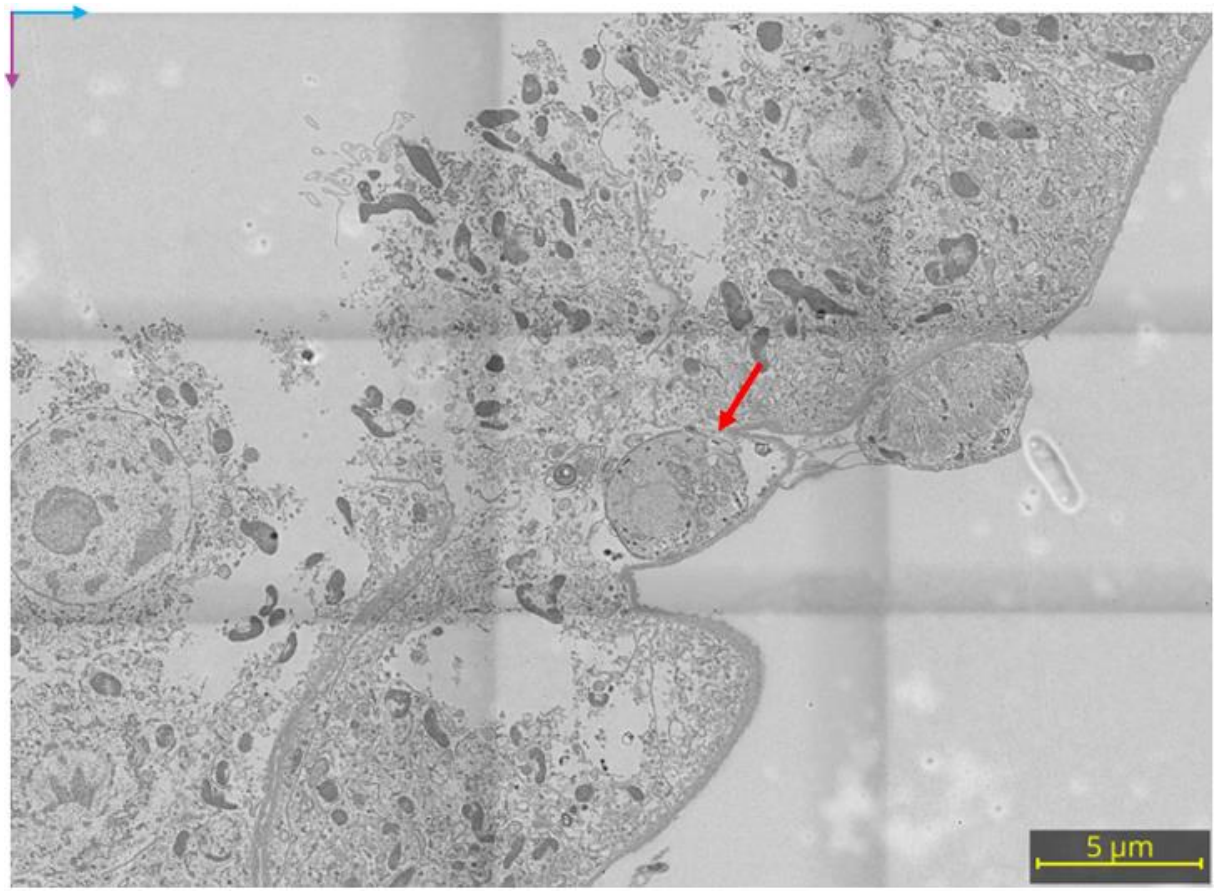


Figure 5.2. Stitched 2D slice from a midgut dataset, arrow (red) indicating the parasite position.

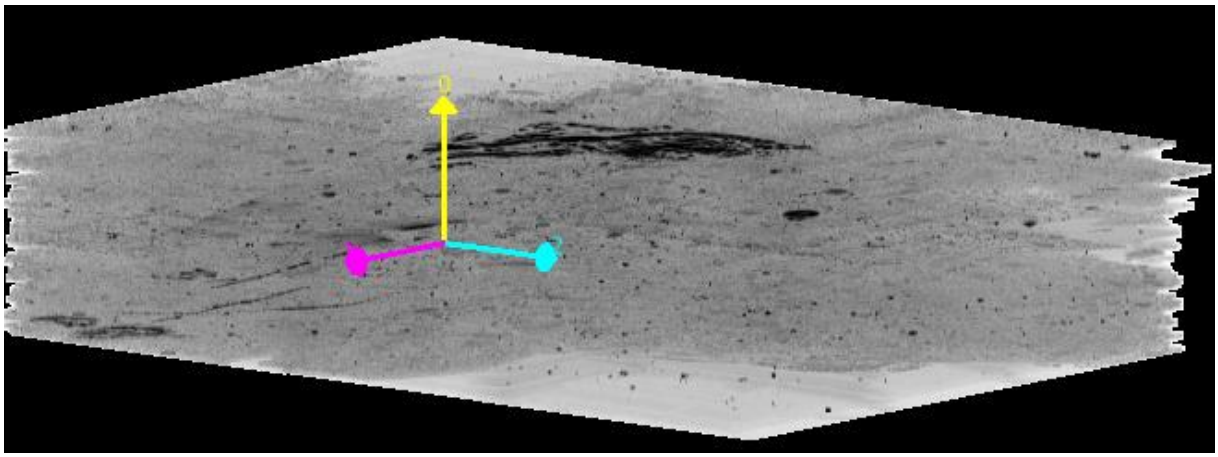


Figure 5.3. 3D rendering of a stitched image stack, visualized along the Z-axis to illustrate volume depth and stack integrity.

Converting the stitched image data from raw TIFF into the OME-Zarr format resulted in a considerable reduction in storage size. In total, the five stitched datasets occupy approximately 389 GB in OME-Zarr format, compared to around 594 GB in the original TIFF format, presenting roughly a one-third reduction in data volume. This reduction reflects the use of Blosc

compression with the Zstandard (zstd) codec, (compression level 5, shuffle enabled), which was consistently applied across all multiscale pyramids. Figures 5.4 and 5.5 illustrate the storage differences and compression gains achieved by incorporating the next generation file format (OME-Zarr) for each dataset.

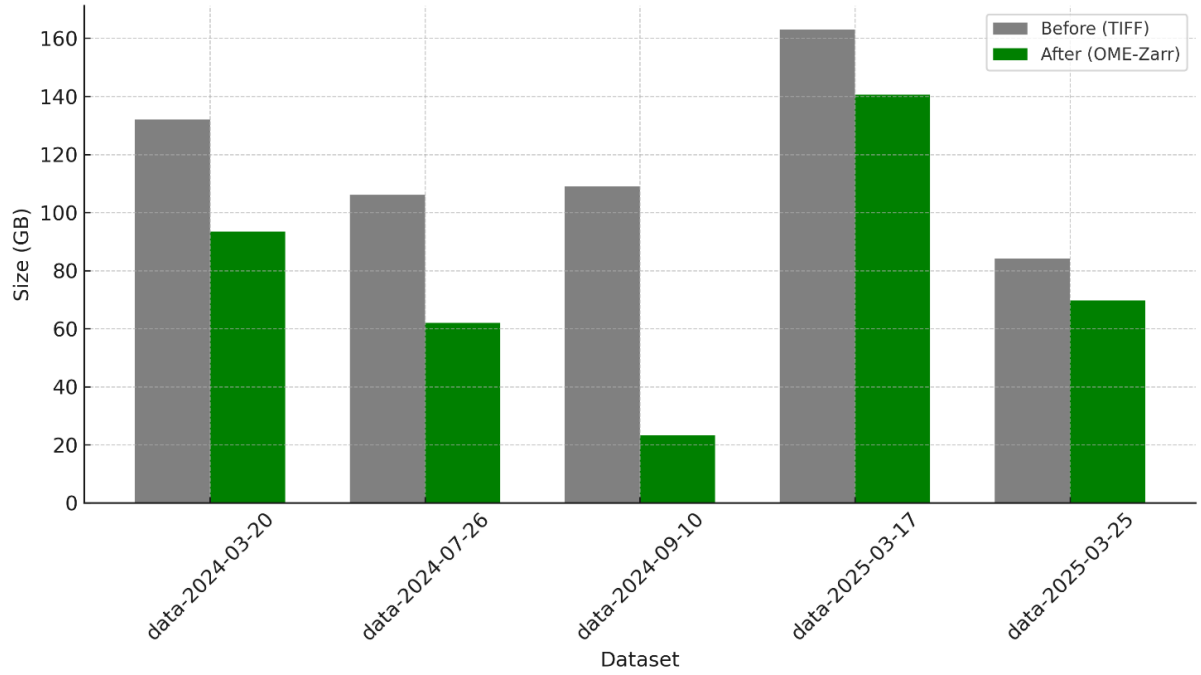


Figure 5.4. Comparison of input vs. output data sizes for each dataset. This bar chart shows the TIFF input size (gray) and OME-Zarr output size (green) for the five datasets listed in Table 5.1.

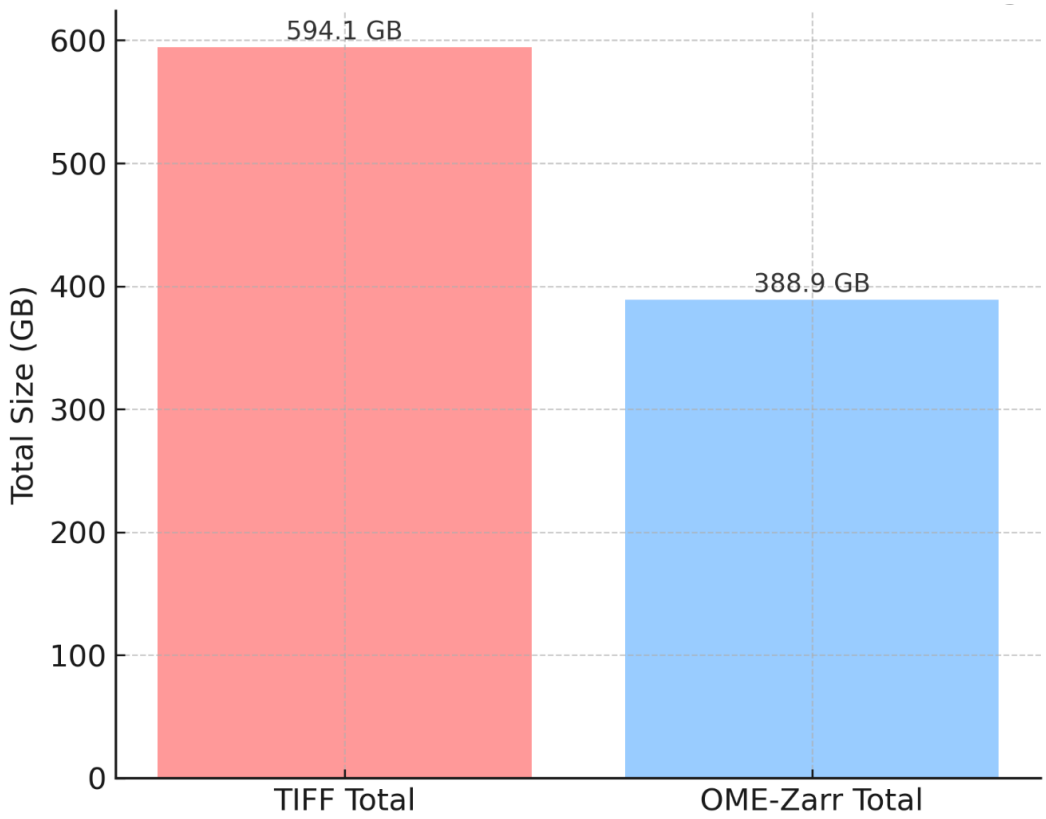


Figure 5.5. Compares the cumulative size of raw TIFF tiles with the final stitched OME-Zarr volumes across all datasets.

The overall percentage reduction in dataset size after converting to OME-Zarr format is approximately 34.54%. This reduction reflects the efficiency gained by transitioning from raw TIFF tiles to the optimized OME-Zarr structure, contributing to faster downstream processing and reduced storage demands.

5.2 Volume Cropping and Image Registration

Volume cropping served as a crucial intermediate step, aiming to reorganize and isolate specific regions within the large stitched stacks. The high-resolution "detail" stacks acquired at 2.5 nm XY resolution were used as the primary datasets for fine-scale ultrastructural analysis, while lower-resolution 10 nm "tissue-view" images were used primarily for identifying regions of interest (ROIs) in gut samples. As shown earlier in Table 5.1, some stacks, especially from the blood datasets (e.g., *data-2024-07-26*, *data-2024-09-10*), contained multiple ookinetes in a single volume. It was also observed that each parasite occupied only a small fraction of the total stack volume in both blood and mid-gut samples.

To address this, an automatic volume cropping was performed, as explained in Section 3.3 (Materials and Methods). Using manually drawn bounding boxes around each ookinete in Napari, we extracted smaller, focused sub-volumes each centered around a single parasite.

For example, one stitched stack from *data-2025-03-17* (gut dataset) originally measured ~34.5 GB and contained two ookinetes. After cropping, the data was split into two lower-sized separate stacks of 18.6 GB and 13.2 GB, respectively. Similarly, in a blood sample from *data-2024-07-26*, a 25 GB stack containing four ookinetes was split into four smaller volumes, each under 5 GB, see Figure 5.6. This approach reduced stack complexity through parasite-focused cropping, thus enabled more efficient processing, faster data access, and easier storage handling in the subsequent steps of the pipeline.

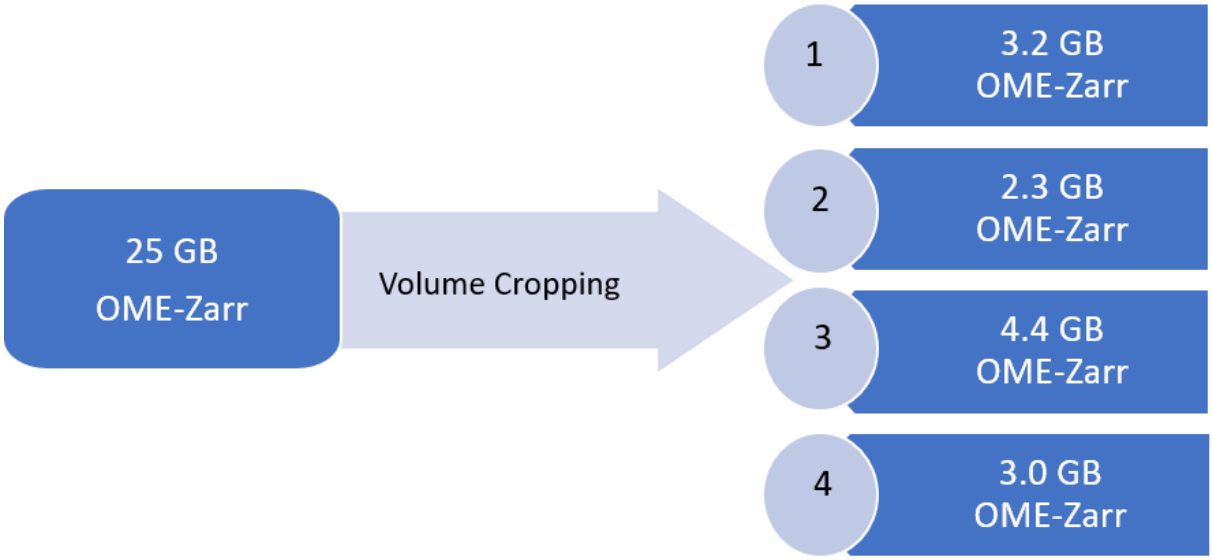


Figure 5.6. Volume optimization by splitting a large Stack into individual parasite crops.

Table 5.2 summarizes the file size of each 3D stack before and after cropping across all datasets. The data were already converted into OME-Zarr format while performing image stitching before cropping. For blood datasets, each cropped stack (containing a single ookinete) was typically under 5 GB, while for gut datasets, the cropped volumes were generally under 20 GB. In addition to the table, Figure 5.7 visualizes the same information using a bar chart for easier comparison. It can be seen in the bar chart; each dataset experienced a substantial reduction in file size up to nearly 50% through volume cropping. The reduction in size reflects the isolation of individual ookinetes into smaller, more manageable sub-volumes. This step was especially

beneficial for datasets with multiple parasites per stack, as the cropped volumes were more focused and individually tailored for further image registration and segmentation.

Table 5.2 Size comparison of stitched 3D stacks before and after automatic volume cropping. Each row represents a dataset grouped by type (blood or gut), with the number of stitched stacks, and their corresponding OME-Zarr file sizes (in GB) before and after cropping.

Dataset	Type	No. of 3D stacks	Size before Volume Cropping (GB)	Size after Volume Cropping (GB)
data-2024-03-20	Gut	4	1.92	0.634
			8.21	1.5
			1.78	0.64
			11.1	5.19
data-2024-07-26	Blood	3	25.1	3.2
				2.31
				4.35
				3.02
			17.9	1.14
				0.7
			19	2.9
				1.31
data-2024-09-10	Blood	5	5.27	0.7
			6.51	0.39
				0.25
			3.91	0.66
				0.66
			3.29	0.36
				0.25
			2.49	0.49
data-2025-03-17	Gut	3	34.5	18.6
				13.2
			19.2	12.5
			16.1	12.4
data-2025-03-25	Gut	3	19.1	12.8
			19.5	14.1
			19	5.07

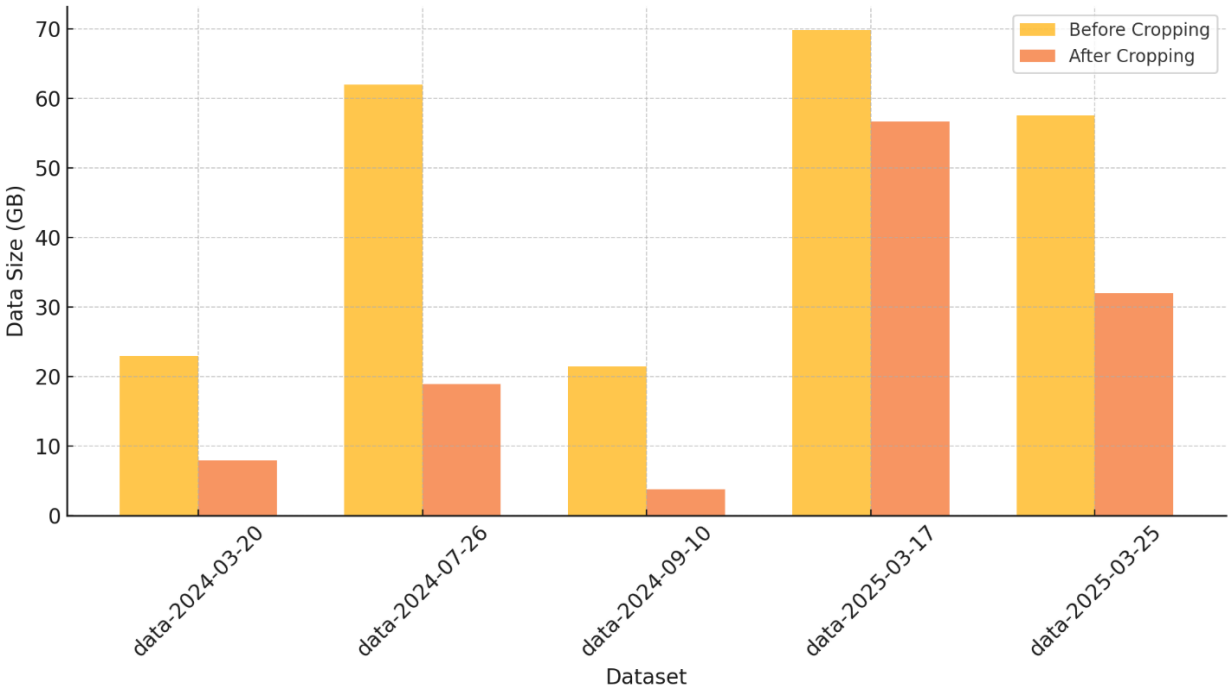


Figure 5.7 Comparison of total data size before and after volume cropping across the five experimental datasets. The left bar (Yellow) in each pair represents the combined size of stitched 3D stacks before cropping, and the right bar (Orange) shows the total size after cropping individual ookinetes into smaller volumes.

To enhance data manageability of large-scale image data, two key strategies were applied: converting the original TIFF files to the next-generation OME-Zarr format and performing automatic volume cropping to isolate single parasites per stack. Together, these approaches streamlined better data handling for downstream image processing steps, while preserving essential biological information. The bar chart below (Figure 5.8) illustrates the cumulative size reduction achieved by these approaches across all datasets.

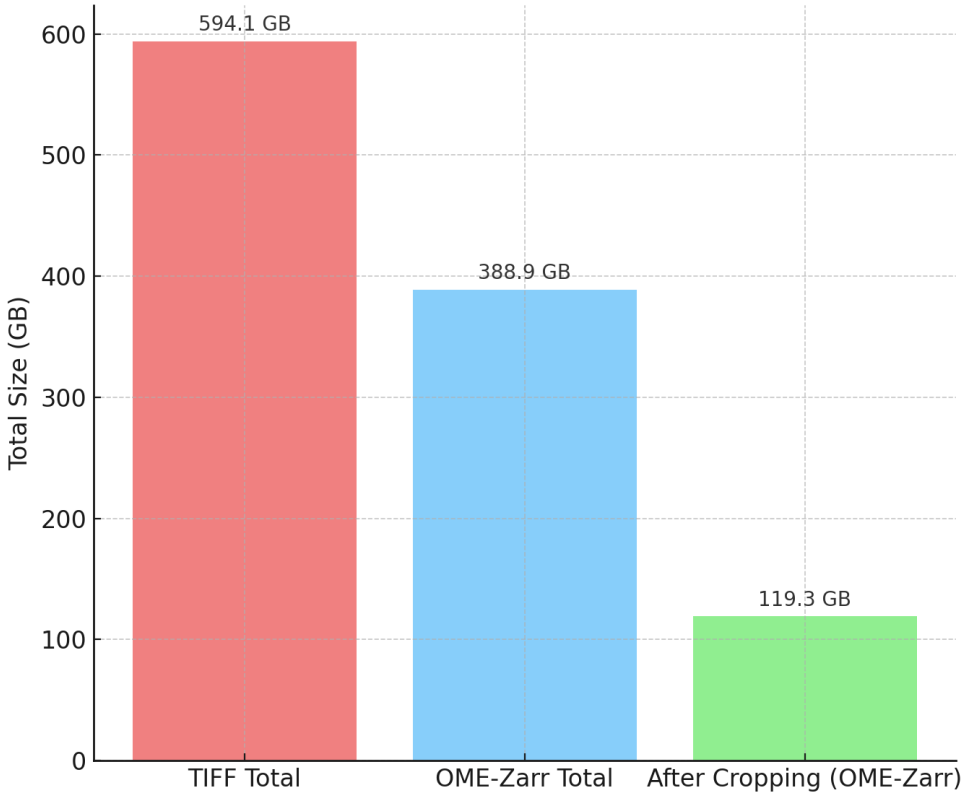


Figure 5.8. Bar chart showing total data size at three key stages of the image analysis workflow: original raw TIFF data (~594 GB), converted OME-Zarr volumes (~389 GB), and the final cropped OME-Zarr volumes (~144 GB).

These cropped volumes were then used as input for the next step: image registration, where misalignments across the Z-axis were corrected to accurately reconstruct the 3D shape and continuity of the parasites. Misalignments in the original stacks were common and varied between samples. These included both variable translational shifts that is the parasite appearing in different X–Y positions across slices and rotational drift meaning tilting or warping through the stack.

To correct these issues, we applied a registration method as described in Section 4.4. The registration was performed on the cropped volumes using TrakEM2 in Fiji, allowing both automated and manual alignment refinements. This step ensured that parasite structures such as the membrane, nucleus, and organelles remained consistently positioned across consecutive slices.

To illustrate the improvements, Figure 5.9 presents two examples of volume rendering before and after registration.

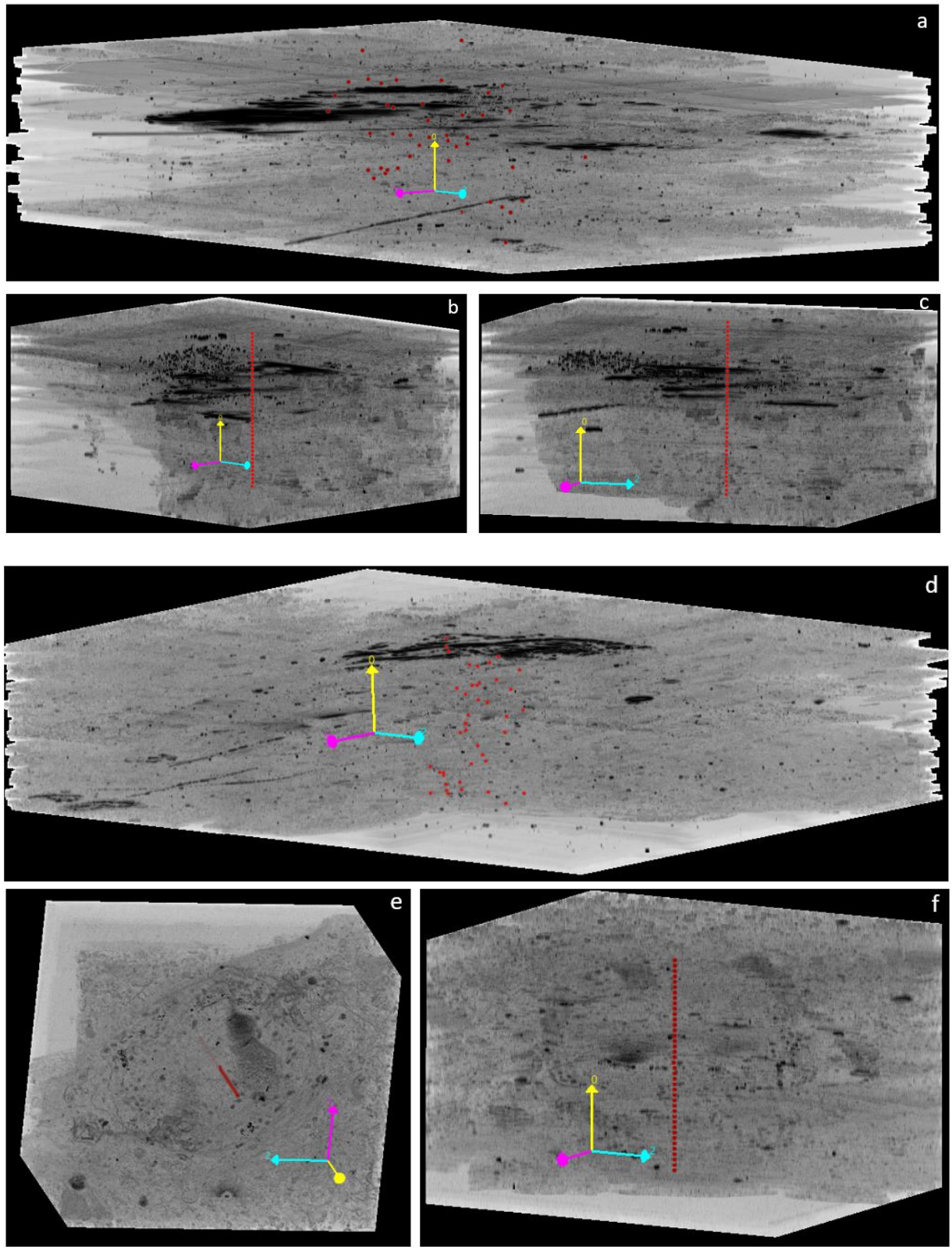


Figure 5.9 Volume rendering of cropped parasite stacks before and after image registration. (a) and (d) show the unregistered volumes with scattered red dot annotations indicating misalignment. (b), (c), (e), and (f) show the same volumes after registration at two different orientations, red dots are now aligned, indicating successful correction along the Z-axis. The yellow axis represents depth (Z-direction), and red dots correspond to the parasite center across slices.

5.3 3D Segmentation of Ookinetes and Ultrastructural Organelles

To begin the segmentation stage, we tested whether the SAM, a general-purpose image segmentation framework developed by Meta AI could be directly applied to segment *Plasmodium* ookinetes from vEM image stacks.

As a test case, few volumes stack of ookinetes from the data were selected and binary masks for the parasite on each slice were automatically generated using the direct SAM Python implementation. The segmentation masks produced by SAM varied significantly in quality across the stack. Mid-volume slices occasionally showed reasonably accurate segmentation of the parasite body, see slice number 11, 15 and 24 in Figure 4.10. However, top and bottom slices (e.g., slice number 3, 8 and 35 in Figure 4.10), which typically suffer from low contrast boundaries due to the genuine morphological variations, resulted in fragmented or incorrect masks. Poor segmentation is observed in the presence of imaging artifacts, highlighting the model's limited reliability, see slice number 6 and 31 in Figure 4.10. In many slices, SAM either ignored the parasite entirely or labeled irrelevant regions such as background textures or boundaries, full set of results is provided in Appendix I. These examples emphasize the need for tailored semi-automatic methods when analyzing such complex biological volumes.

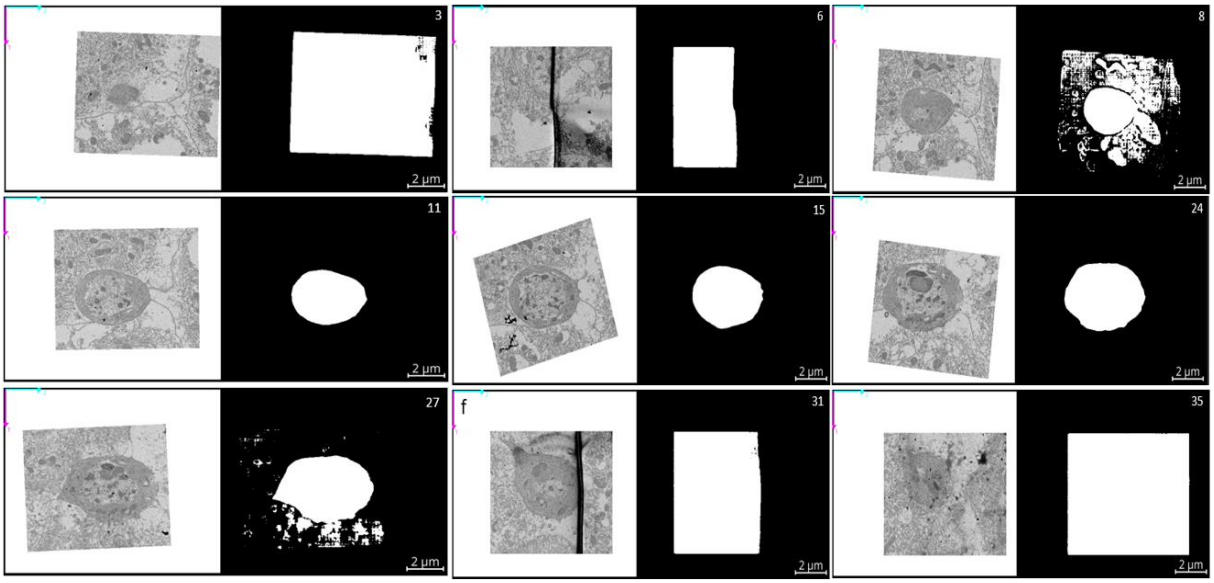


Figure 5.10 Represent segmentation results of SAM on 9 selected slices from a 38-slice vEM stack of a P. falciparum ookinete. Each pair shows an original EM slice (left) and the corresponding binary mask predicted by SAM (right).

Due to the shortcomings of direct application of the SAM to vEM data, a more practical approach was adopted for segmentation workflow, as detailed in the Materials and Methods

chapter. This method combines SAM's segmentation strengths with interactive controls, enabling focused and accurate annotation of ookinete and its subcellular structures. With some manual correction and annotations, high-quality 3D labels for five classes were generated. Each parasite (ookinete) was annotated with two sets of binary masks:

- A complete mask of the whole parasite body, primarily generated using Napari-SAMv2.
- Label masks for four ultrastructural classes: micronemes, hemozoin, nucleus and crystalloids, annotated using napari-sam with manual corrections.

In total, segmentation masks were produced for 26 individual ookinetes, covering different developmental stages:

- 7 mature-stage ookinetes
- 10 intermediate-stage ookinetes
- 8 early-stage ookinetes
- 1 knock-out ookinete from the mid-gut sample.

Figures 5.11 through 5.13 showcase examples from each developmental stage, illustrating the segmentation quality.

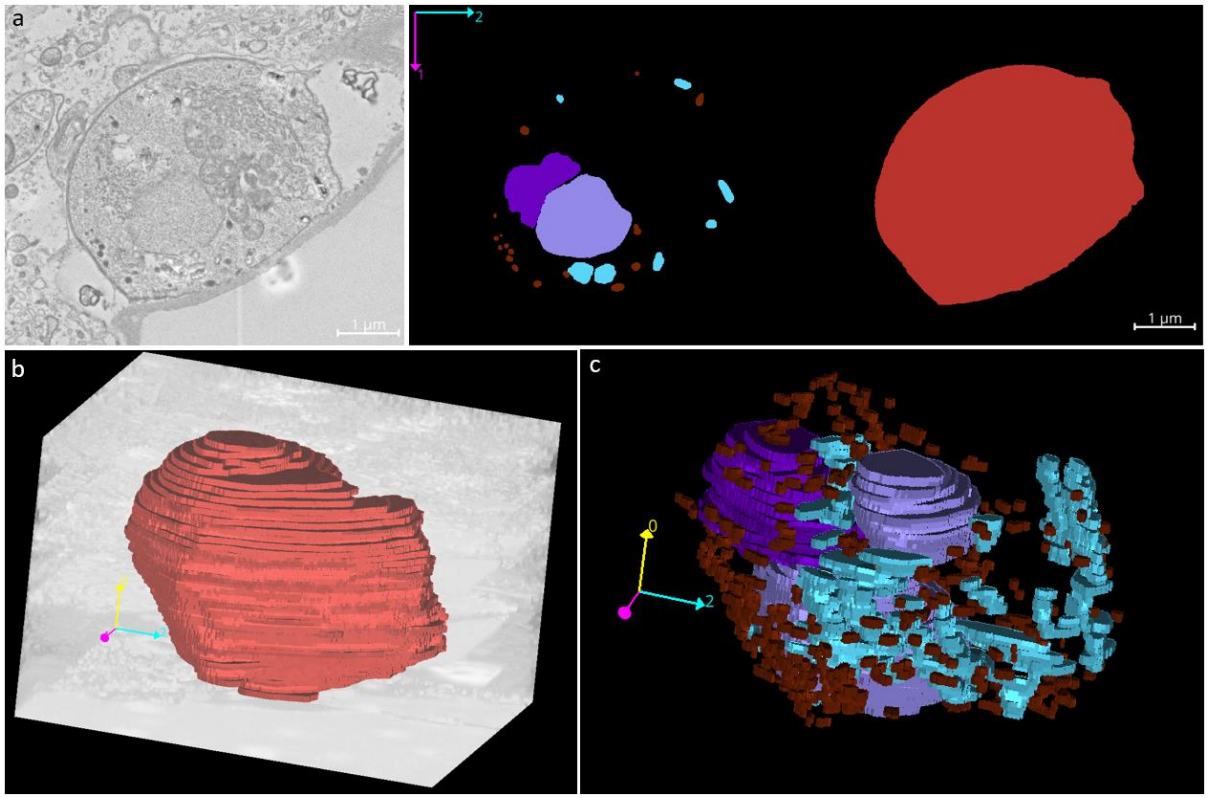


Figure 5.11 Representative segmentation results for a mature ookinete from a mid-gut sample. (a) Left: original EM slice; middle: organelle segmentation showing nucleus (light-purple),

micronemes (brown), crystalloids (purple), and hemozoin (cyan); right: full-body parasite mask (red). (b) Volume rendering of the whole parasite mask. (c) Volume rendering of segmented organelles. Stack contains 49 slices at 90 nm Z-spacing.

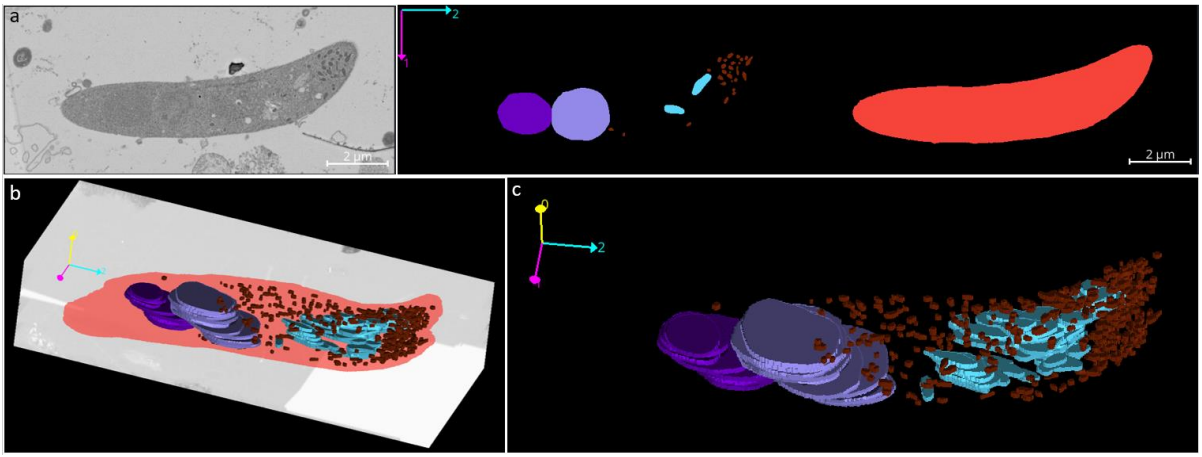


Figure 5.12. Segmentation of an intermediate-stage ookinete from a blood sample. (a) Middle slice and corresponding organelle and full-body segmentation masks. (b) 3D rendering of the full parasite mask over the EM volume. (c) Organelles displayed in 3D. Stack comprises 35 slices.

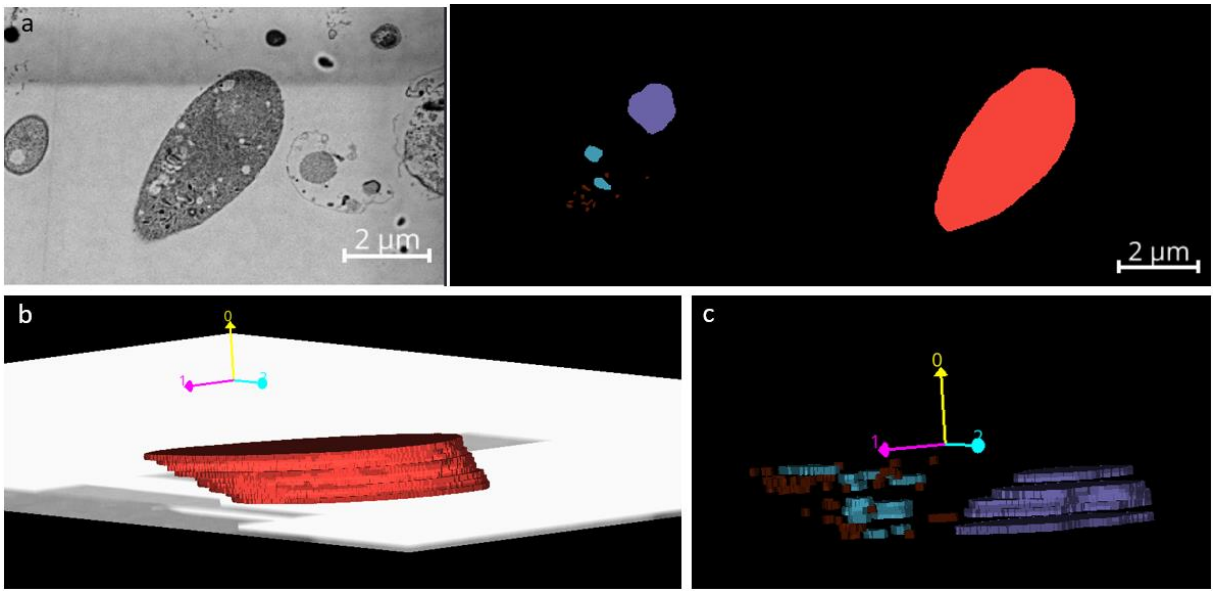


Figure 5.13. Example of an early-stage ookinete segmentation from a blood sample. (a) Middle slice with organelle and whole-parasite segmentation masks. (b) Volume rendering with the full parasite mask overlaid on the image data. (c) 3D rendering of ultrastructural organelles. This dataset includes 9 slices.

This tiered approach, starting from fully automatic, then refining with semi-automated methods helped balance accuracy for generating segmentation masks. All masks were saved in TIFF format for use in following quantitative analyses.

5.4 Quantification

To systematically assess and compare ultrastructural features across developmental stages of *P. falciparum* ookinetes, we applied an automated Nextflow quantification pipeline to all segmented 3D volumes. For each segmented parasite, the following quantitative features were extracted:

- Number and volume of micronemes
- Microneme distances to the parasite membrane
- Inter-microneme distances
- Number and volume of hemozoin crystals
- Crystalloid size (if present)
- Whole parasite volume

Each organelle's mask was converted into voxel counts and mapped into physical units using voxel dimensions, providing nanometer-scale measurements. Additionally, 3D Euclidean distance transforms and k-dimensional tree [104] were applied to calculate spatial proximities e.g., microneme-to-surface, and micronemes to nearest micronemes.

All outputs were saved as CSV files, allowing for quantitative insights. In below, I presented some key observations and limitations across all the developmental stages:

- Micronemes were consistently detected in most parasites across early, intermediate, and mature stages.

- Hemozoin crystals appeared primarily in intermediate and mature parasites.

- Crystalloids were only confidently identified in a few intermediate and mature ookinetes. In many cases mostly for intermediate stage, suspected crystalloids could not be conclusively labeled due to low image quality or ambiguous boundaries, requiring expert validation or future post-processing using some denoising.

- In mid-gut samples, a key biological question "whether parasites traverse or squeeze between host cells" remains unanswered in this study. Addressing this would have required segmenting both the parasite and surrounding host tissue in the mid-gut datasets. Due to time constraints, this complex analysis was not feasible within the current thesis, but is marked as a high-priority direction for future work.

To summarize the preliminary quantification results, Table 5.3 presents a selection of ookinetes from the dataset for which the full 3D structure was well captured. Out of the total 26 segmented parasites, only those with complete volume stacks (where both ends of the parasite were entirely visible and intact) were considered for these measurements. Parasites with incomplete 3D stacks, such as those cut off at the top or bottom, were excluded from volumetric and count-based analysis, since key metrics like parasite volume or microneme count would not be reliable.

For each included parasite, total volume, number of micronemes, number of hemozoin crystals, presence or absence of a crystalloid body, and the crystalloid volume (if applicable) were reported.

Table 5.3 Summary of volumetric and structural features of fully segmented parasites.

	Samples	Ookinete Volume (μm^3)	Number of Micronemes	Number of Hemozoins	Number of Crystalloids	Crystalloid Volume (μm^3)
Early	20hpi-para1	0.593	10	0	0	0
	20hpi-para1	2.764	18	0	0	0
	20hpi-para3	2.052	23	3	0	0
	20hpi-para4	6.744	35	7	0	0
	20hpi-para5	1.670	24	0	0	0
	20hpi-para6	1.721	31	0	0	0
Intermediate	24hpi-para1	16.332	135	21	0	0
	24hpi-para2	26.315	224	12	1	1.081
	24hpi-para3	20.834	81	8	0	0
	24hpi-para4	27.353	0	8	0	0
	24hpi-para5	16.704	118	6	0	0
	24hpi-para6	27.600	255	10	0	0
Mature	30hpi-para1	19.109	65	11	1	1.226
	30hpi-para2	20.605	2	31	0	0
	30hpi-para3	28.320	147	21	1	1.023

	30hpi-para4	33.908	231	25	1	2.403

Figure 5.14 represents the scatter plots showing parasite volume versus subcellular structures (micronemes and hemozoins) by developmental stage. Each point is colored according to developmental stage: blue for early, orange for intermediate, and green for mature. A clear trend is observed: early-stage parasites tend to have smaller volumes and fewer micronemes, while intermediate and mature parasites show larger volumes and higher organelle counts. Notably, one sample from the 30hpi group had only 2 micronemes, and an intermediate sample showed 0 micronemes**.** Hemozoin presence was more consistent in intermediate and mature developmental stages, while early-stage parasites (20hpi) often had none.

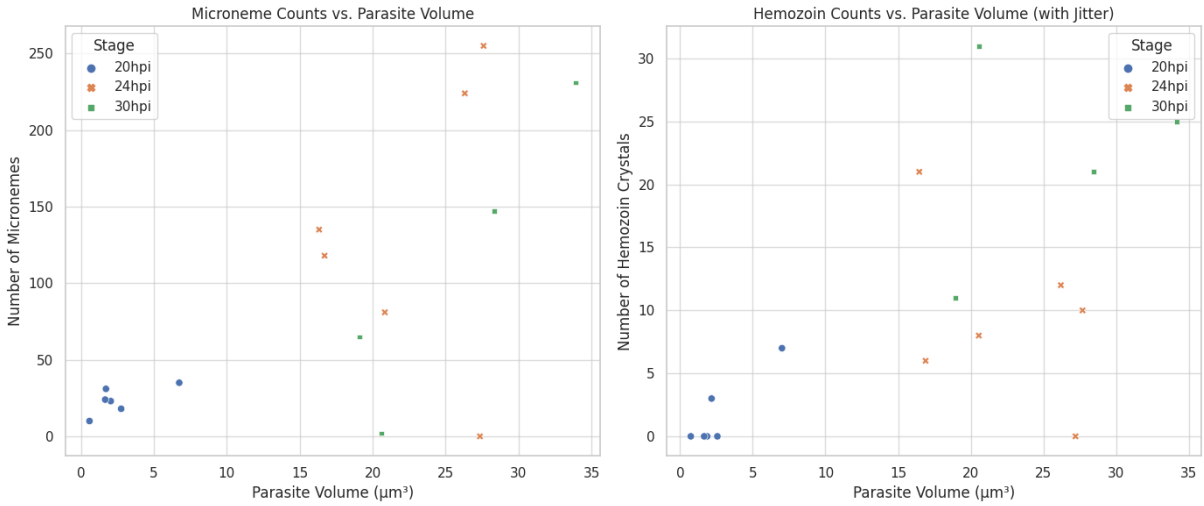


Figure 5.14. Scatter plots of microneme and hemozoin crystal count as a function of parasite volume across developmental stages. (Left) Number of micronemes plotted against parasite volume. (Right) Number of hemozoin crystals plotted against parasite volume, with slight vertical jitter added to better display overlapping points, particularly where hemozoin count is zero.

6 Discussion

This study set out to build an end-to-end Python-based computational pipeline capable of handling and processing large-scale vEM datasets of *P. falciparum* ookinetes and their ultrastructural organelles. From raw, tiled 2D TIFF mosaics to detailed 3D segmentations and organelle-level quantifications, the pipeline was developed with the aim of transforming fragmented image data into biologically meaningful insights. The study presented a complete analysis chain integrating a variety of open-source tools including, Nextflow workflows, next-generation file format (OME-Zarr), ImageJ and Napari plugins, and custom Python scripts. While the primary emphasis was on computational reproducibility and data management, several important observations also emerged along the way, which are discussed here.

The first major milestone in the workflow was the successful implementation of an automated tile-stitching system using the ImageJ Grid/Collection plugin within a Nextflow pipeline. Given the volume and heterogeneity of the five microscopy datasets spanning both blood and midgut samples, and three developmental stages of the parasite this step played a foundational role. Despite challenges such as missing tiles and variable tile overlap across datasets, the pipeline reliably produced high-resolution, stitched 3D image volumes stored in OME-Zarr format. Over 590 GB of tiled TIFF images were successfully assembled into 19 stitched 3D volumes, laying the groundwork for downstream analysis.

A major improvement in data handling came from converting stitched datasets into the OME-Zarr format. This next-generation bioimaging format not only provided chunked and compressed storage but also enabled rapid data access. By converting from TIFF to OME-Zarr, a one-third reduction in data size was achieved without compromising resolution or content, demonstrating the benefits of adopting scalable formats in modern imaging workflows. These results reinforce the importance of format-aware infrastructure in large-scale biological imaging projects.

Besides efficient storage and accessibility, the OME-Zarr format also remained invaluable for visual inspection of the large 3D datasets. In this project, the ability to explore the volumetric data interactively such as slice by slice, and at varying zoom levels was essential for both quality control and development of the analysis pipeline. Using the napari-ome-zarr plugin within the Napari viewer, navigating through these multi-gigabyte volumes became noticeably smoother compared to working with traditional TIFF stacks, thanks to OME-Zarr's multiscale resolution

levels. Load times were much faster, zooming and scrolling were more responsive as lower-resolution data was loaded at broader zoom levels and higher-resolution data as zoomed-in, making the overall experience more practical for iterative testing and refinement of analysis routines during various stages of pipeline development.

Following stitching, many of the resulting stitched image stacks contained multiple ookinetes within a single image stack and it was also observed that each ookinete occupied only a small fraction of the total volume. However, the focus of this study was on single-parasite analysis, therefore, an automated volume cropping approach was applied to isolate individual parasites, minimizing irrelevant background while maintaining biological context. For example, a 34.5 GB gut stack was reduced to two smaller and more manageable volumes of ~18 GB and ~13 GB each, by isolating two parasites. The effectiveness of this approach is reflected not only in reduced file sizes, but also in its impact on processing speed and clarity in later image registration and segmentation tasks.

A major challenge in working with serial EM images is misalignment between slices introduced during image acquisition. Both translational and rotational displacements were observed along the Z-axis, across all datasets. By applying a semi-automated registration method using TrakEM2 an ImageJ plugin, the parasite structures were aligned along depth to produce anatomically coherent 3D stacks. The improvement was evident both visually (when rendering the red-dot annotations of parasite centers) and in the stability of segmentation outputs. This step, though not fully automated, was essential in enabling volumetric analyses and highlights the trade-off between automation and accuracy in biomedical image processing.

For segmentation, an initial attempt to apply Meta AI's Segment Anything Model (SAM) directly to vEM images in a fully automated fashion proved insufficient. While SAM showed reasonable results in the central slices of the 3D stack, its performance dropped near the edges and in slices with imaging artifacts. At this stage, the possibility of exploring other automated solutions, such as deep learning (DL) models for segmentation, was considered. However, the limited number of available parasites from each developmental stage resulted in insufficient training data for a DL-based solution. Further, the complexity of vEM datasets due to the presence of imaging artifacts led us to explore and test traditional segmentation solutions available for vEM.

In response, the study adopted a semi-automatic segmentation approach using two Napari plugins: napari-SAMv2 for full parasite body segmentation, and napari-sam for organelle

annotation, along with manual corrections and annotations. Using this approach, 3D masks for parasite and its ultrastructure (organelles) including micronemes, hemozoin crystals, and crystalloids were generated. In total, 26 ookinetes with 4 ultrastructural classes were segmented across early, intermediate, and mature stages. An important insight from this experience is that while full automation for vEM segmentation remains elusive, combining tailored segmentation tools developed specifically for 3D EM data can yield more dependable outcomes. It is important to note that in vEM studies overall, even today, semi-manual approaches are still necessary due to the complexity of the data. Although time-consuming, this approach proved to be a practical and effective solution for generating accurate, high-quality masks in this study.

Once segmented, both the original 3D EM volumes and the corresponding segmentation masks were stored in the OME-Zarr format. This ensured that all data layers including original EM images, parasite body masks, and organelle segmentations remained spatially aligned and accessible within a unified scalable structure. By consolidating raw and derived data in the same scalable format, this pipeline streamlined data handling, enhanced reproducibility and simplified following quantitative analyses. Also, the use of OME-Zarr enabled seamless data sharing with field-expert collaborators abroad by providing them access to both the raw and segmented volumes in an open, standardized format. This facilitated the collection of valuable feedback and validation remotely. Moreover, this format choice aligns with the FAIR data principles, ensuring the datasets remain findable, accessible, interoperable, and reusable for future research efforts.

For quantification, the segmented parasite structures were passed through a Nextflow measurement workflow that computed key quantitative descriptors from each 3D mask. For each sample, the measurements were saved as structured CSV outputs which include counts and volumes of whole ookinetes, micronemes, crystalloids and hemozoin crystals, distances of micronemes from the parasite membrane, inter-microneme distances and crystalloid-hemozoin distances. These metrics addressed important biological questions about the size, number and distribution of secretory organelles involved in host invasion. Since the quantitative vEM studies of *P. falciparum* ookinetes are still underexplored, the tabular information stored in these CSV outputs have great significance could provide valuable insights into the biology of this deadly parasite.

This thesis mainly focused on the development and validation of the computational pipeline, rather than on deriving biological conclusions, therefore, a selection of basic preliminary

measurements was presented in the Results chapter of this study. However, more detailed statistical summaries and biological interpretation were intentionally left for future collaborative work with domain-expert researchers. Notably, all measurements were saved in an accessible, standardized format, laying the foundation for future statistical analyses by *plasmodium* cell biologists.

6.1 Limitations

Despite the promising outcomes of this study, several limitations emerged during the development and application of the pipeline. One of the most significant challenges was the reliance on semi-automated methods for image registration and segmentation. Although the pipeline successfully transformed raw 3D EM data into quantitative organelle-level outputs, key steps like segmentation and registration still required manual input and adjustments. Tools such as TrakEM2, Napari-SAMv2 and napari-sam enabled a degree of automation and offered interactive labeling and correction features, however substantial manual effort was still needed, especially for volumetric stacks affected by damage, low contrast, or imaging artifacts. This partial automation and continued dependence on human refinement limit the scalability of the pipeline, and a fully automated workflow remains a future goal. In addition, several parasite volumes were incomplete (either truncated at the top or bottom of the Z-stack) making them unsuitable for quantitative analysis and thereby reducing the sample size available for developmental stage comparisons. Another limitation arose from the biological scope of the study: while the computational framework was successfully established, more detailed biological analyses such as spatial mapping of parasite invasion within midgut tissue or comparative assessments across stages remained unexplored. This was largely due to time constraints and the need for collaboration with domain experts, including the team in Berlin responsible for sample preparation. Lastly, although the pipeline generated rich quantitative data in structured formats, statistical analysis and hypothesis testing were beyond the scope of this thesis and are left for future work. In summary, while this project lays a strong technical foundation, future progress will rely on advancing automation and engaging expert collaborators to fully uncover the biological insights contained in these complex 3D EM datasets.

7 Conclusions and Future Work

This thesis successfully established a scalable and semi-automated pipeline for analyzing large vEM datasets of *P. falciparum* ookinetes. The pipeline handled all stages from stitching and registration to segmentation and quantification, while leveraging open-source tools and standardized file formats such as OME-Zarr. The work demonstrates that by combining open-source tools with modular workflow engines like Nextflow, it is possible to build robust image processing systems capable of extracting meaningful biological information from challenging volumetric datasets.

By producing outputs such as stitched stacks, segmented organelles, and standardized CSV measurements, the pipeline enables researchers to move from raw tiled EM data to biologically interpretable results. While the pipeline does not yet achieve full automation, it provides a strong framework for future extensions and has already delivered results suitable for expert review and hypothesis generation.

The key contributions of this work include:

- A reproducible and fully automatic image stitching pipeline capable of handling large, complex volumetric EM datasets and assembling them into coherent 3D volume stacks.
- Efficient data organization and volume reduction using volume cropping and OME-Zarr file format.
- 3D parasite (ookinete) and organelle segmentation models for Plasmodium falciparum at three different developmental stages.
- Quantitative extraction of organelle-level features, ready for biological interpretation.

By laying this foundation, the current study ensures that future biological research can build on a reproducible and extensible computational base.

7.1 Future Work

Looking ahead, this study opens several meaningful directions for future research. A key priority is the further automation of segmentation, particularly for small structures like micronemes, which are quite many in numbers and remain a challenge in this work. With sufficient annotated data, custom-trained models for micronemes segmentation could be

developed, which will potentially enhance both the speed and consistency of their detection. Another promising avenue involves improving the detection of crystalloids, which were often difficult to identify reliably due to poor contrast or imaging noise. Incorporating denoising algorithms or advanced filtering techniques may improve their visibility and segmentation. In terms of biological exploration, a more in-depth analysis of the midgut datasets focusing on how ookinetes interact with their surrounding tissue could shed light on key aspects of parasite invasion, a critical yet underexplored stage in the malaria life cycle. Achieving this would require segmenting both the parasite and the host tissue in 3D, a technically demanding and challenging but highly valuable task. Finally, while this thesis generated a rich set of quantitative data stored in structured CSV files, the next step involves statistical analysis and biological interpretation, ideally carried out in collaboration with parasitology experts. These future directions build upon the computational foundation established in this work and point toward deeper biological insights and expanded interdisciplinary collaboration.

Acknowledgement

I would like to express my deepest gratitude to all the individuals and institutions whose support has been instrumental in the completion of this thesis.

First and foremost, I am sincerely thankful to my supervisors, Dr. Pasi Kankaanpää and Junel Solis, for their invaluable guidance, feedback, and encouragement throughout this journey. Pasi, thank you for giving me the opportunity to join Turku BioImaging—first as a summer intern and later for my master's thesis project. Junel, your constructive mentorship has greatly contributed to the direction and quality of this research.

Special thanks to the Turku BioImaging team—Irina, Linda, Chris, and Dado—for your continuous support and encouragement, which helped me refine and improve my work. I am especially grateful to Irina for her thoughtful feedback on my thesis.

I gratefully acknowledge the financial support provided by the University of Turku International Scholarship, which enabled me to fully dedicate myself to my studies and research.

I also wish to acknowledge the ISIDORe for funding this project, and to thank our collaborator Pablo Cortés from Germany, who designed the project and prepared the biological samples. Special thanks to the Euro-BioImaging Prague Node and Jiří Týč for facilitating the image acquisition.

Finally, I would like to thank my family for their unwavering support throughout my academic journey. Their belief in me has been a constant source of strength, especially as I pursued my studies abroad.

Some diagrams in this thesis were created with the assistance of the AI-Napkin tool, used for conceptual design and idea formatting in partial fulfillment of the graphical requirements.

Turku, 25.07.2025
Ramish Bibi

References

[1] V. Bounkeua, F. Li, and J. M. Vinetz, "In vitro generation of Plasmodium falciparum ookinetes," *American Journal of Tropical Medicine and Hygiene*, vol. 83, no. 6, pp. 1187–1194, Dec. 2010, doi: 10.4269/ajtmh.2010.10-0433.

[2] G. Siciliano, G. Costa, P. Suárez-Cortés, A. Valleriani, P. Alano, and E. A. Levashina, "Critical Steps of Plasmodium falciparum Ookinete Maturation," *Front Microbiol*, vol. 11, Mar. 2020, doi: 10.3389/fmicb.2020.00269.

[3] Syafruddin, R. Arakawa, K. Kamimura, and F. Kawamoto, "Penetration of the mosquito midgut wall by the ookinetes of Plasmodium yoelii nigeriensis," *Parasitol Res*, vol. 77, no. 3, pp. 230–236, Mar. 1991, doi: 10.1007/BF00930863/METRICS.

[4] K. Kadota, T. Ishino, T. Matsuyama, Y. Chinzei, and M. Yuda, "Essential role of membrane-attack protein in malarial transmission to mosquito host," *Proceedings of the National Academy of Sciences*, vol. 101, no. 46, pp. 16310–16315, Nov. 2004, doi: 10.1073/PNAS.0406187101.

[5] K. J. Czymmek, I. Belevich, J. Bischof, A. Mathur, L. Collinson, and E. Jokitalo, "Accelerating data sharing and reuse in volume electron microscopy," *Nat Cell Biol*, vol. 26, no. 4, pp. 498–503, Apr. 2024, doi: 10.1038/S41556-024-01381-3.

[6] "Malaria's Impact Worldwide | Malaria | CDC." Accessed: Jun. 09, 2025. [Online]. Available: https://www.cdc.gov/malaria/php/impact/index.html

[7] M. Desai *et al.*, "Review Epidemiology and burden of malaria in pregnancy," 2007. [Online]. Available: http://infection.thelancet.comVol

[8] R. Varo, C. Chaccour, and Q. Bassat, "Update on malaria," *Medicina Clínica (English Edition)*, vol. 155, no. 9, pp. 395–402, Nov. 2020, doi: 10.1016/j.medcle.2020.05.024.

[9] L. Zekar and T. Sharman, "Plasmodium falciparum Malaria," *StatPearls*, Aug. 2023, Accessed: Jun. 09, 2025. [Online]. Available: https://www.ncbi.nlm.nih.gov/books/NBK555962/

[10] R. Varo, C. Chaccour, and Q. Bassat, "Update on malaria," Nov. 13, 2020, *Ediciones Doyma, S.L.* doi: 10.1016/j.medcli.2020.05.010.

[11] "CDC - DPDx - Malaria." Accessed: Jun. 09, 2025. [Online]. Available: https://www.cdc.gov/dpdx/malaria/index.html

[12] G. Pasvol, "Protective hemoglobinopathies and Plasmodium falciparum transmission," Apr. 2010. doi: 10.1038/ng0410-284.

[13] J. A. Vaughan, B. H. Noden, and A. C. Beier, “SPOROGONIC DEVELOPMENT OF CULTURED PLASMODJUM FALCIPARUM IN SIX SPECIES OF LABORATORY-REARED ANOPHELES MOSQUITOES,” 1994.

[14] R. E. Sinden and R. Strong, “An ultrastructural study of the sporogonic development of Plasmodium falciparum in Anopheles gambiae,” 1978. [Online]. Available: http://trstmh.oxfordjournals.org/

[15] D. T. Ouologuem, A. Dara, A. Kone, A. Ouattara, and A. A. Djimde, “Plasmodium falciparum Development from Gametocyte to Oocyst: Insight from Functional Studies,” *Microorganisms 2023, Vol. 11, Page 1966*, vol. 11, no. 8, p. 1966, Jul. 2023, doi: 10.3390/MICROORGANISMS11081966.

[16] L. A. Baton and L. C. Ranford-Cartwright, “Plasmodium falciparum ookinete invasion of the midgut epithelium of Anopheles stephensi is consistent with the Time Bomb model,” *Parasitology*, vol. 129, no. 6, pp. 663–676, Dec. 2004, doi: 10.1017/S0031182004005979.

[17] S. M. Paskewitz, M. R. Brown, A. O. Lea, and F. H. Collins, “Ultrastructure of the encapsulation of Plasmodium cynomolgi (B strain) on the midgut of a refractory strain of Anopheles gambiae,” *Journal of Parasitology*, vol. 74, no. 3, pp. 432–439, 1988, doi: 10.2307/3282053.

[18] Y. S. Han, J. Thompson, F. C. Kafatos, and C. Barillas-Mury, “Molecular interactions between Anopheles stephensi midgut cells and Plasmodium berghei: the time bomb theory of ookinete invasion of mosquitoes,” *EMBO J*, vol. 19, no. 22, pp. 6030–6040, Nov. 2000, doi: 10.1093/EMBOJ/19.22.6030.

[19] A. Danielli, C. Barillas-Mury, S. Kumar, F. C. Kafatos, and T. G. Loukeris, “Overexpression and altered nucleocytoplasmic distribution of Anopheles ovalbumin-like SRPN10 serpins in Plasmodium-infected midgut cells,” *Cell Microbiol*, vol. 7, no. 2, pp. 181–190, Feb. 2005, doi: 10.1111/J.1462-5822.2004.00445.X.

[20] J. F. G. M. Meis, G. Pool, G. J. van Gemert, A. H. W. Lensen, T. Ponnudurai, and J. H. E. T. Meuwissen, “Plasmodium falciparum ookinetes migrate intercellularly through Anopheles stephensi midgut epithelium,” *Parasitol Res*, vol. 76, no. 1, pp. 13–19, Jan. 1989, doi: 10.1007/BF00931065/METRICS.

[21] L. A. Baton and L. C. Ranford-Cartwright, “Do malaria ookinete surface proteins P25 and P28 mediate parasite entry into mosquito midgut epithelial cells?,” *Malar J*, vol. 4, no. 1, pp. 1–8, Feb. 2005, doi: 10.1186/1475-2875-4-15.

[22] S. Yahiya *et al.*, “Live-cell fluorescence imaging of microgametogenesis in the human malaria parasite Plasmodium falciparum,” *PLoS Pathog*, vol. 18, no. 2, p. e1010276, Feb. 2022, doi: 10.1371/JOURNAL.PPAT.1010276.

[23] J. Guizetti, “Imaging malaria parasites across scales and time,” 2025, *John Wiley and Sons Inc*. doi: 10.1111/jmi.13384.

[24] F. Evers *et al.*, “Comparative 3D ultrastructure of Plasmodium falciparum gametocytes,” *Nature Communications*, vol. 16, no. 1, Dec. 2025, doi: 10.1038/s41467-024-55413-5.

[25] M. Aikawa, R. T. Cook, J. J. Sakoda, and H. Spi~tnz, “Fine Structure of the Erythrocytic Stages of Plasmodium knowlesi A Comparison between Intracellular and Free Forms *,” 1969.

[26] R. E. Sinden, “Gametocytogenesis of Plasmodium falciparum in vitro: An electron microscopic study,” *Parasitology*, vol. 84, no. 1, pp. 1–11, 1982, doi: 10.1017/S003118200005160X.

[27] G. Volohonsky, P. Paul-Gilloteaux, J. Štáfková, J. Soichot, J. Salamero, and E. A. Levashina, “Kinetics of Plasmodium midgut invasion in Anopheles mosquitoes,” *PLoS Pathog*, vol. 16, no. 9, Sep. 2020, doi: 10.1371/journal.ppat.1008739.

[28] R. M. Rudlaff, S. Kraemer, J. Marshman, and J. D. Dvorin, “Three-dimensional ultrastructure of Plasmodium falciparum throughout cytokinesis,” *PLoS Pathog*, vol. 16, no. 6, Jun. 2020, doi: 10.1371/journal.ppat.1008587.

[29] T. Araki *et al.*, “Three-dimensional electron microscopy analysis reveals endopolygeny-like nuclear architecture segregation in Plasmodium oocyst development,” *Parasitol Int*, vol. 76, Jun. 2020, doi: 10.1016/J.PARINT.2019.102034.

[30] A. Dutta, G. Suseela, and A. Sood, “Robust Multi-Modal Image Stitching for Improved Scene Understanding.”

[31] F. S. Mohammadi, S. E. Mohammadi, P. Mojarad Adi, S. M. A. Mirkarimi, and H. Shabani, “A comparative analysis of pairwise image stitching techniques for microscopy images,” *Sci Rep*, vol. 14, no. 1, pp. 1–10, Dec. 2024, doi: 10.1038/S41598-024-59626-Y.

[32] C. J. Peddie *et al.*, “Volume electron microscopy,” *Nature Reviews Methods Primers 2022 2:1*, vol. 2, no. 1, pp. 1–23, Jul. 2022, doi: 10.1038/s43586-022-00131-9.

[33] C. J. Peddie and L. M. Collinson, “Exploring the third dimension: Volume electron microscopy comes of age,” *Micron*, vol. 61, pp. 9–19, Jun. 2014, doi: 10.1016/J.MICRON.2014.01.009.

[34] I. Kolotuev, “Work smart, not hard: How array tomography can help increase the ultrastructure data output,” *J Microsc*, vol. 295, no. 1, pp. 42–60, Jul. 2024, doi: 10.1111/jmi.13217.

[35] A. Bria and G. Iannello, “TeraStitcher-A tool for fast automatic 3D-stitching of teravoxel-sized microscopy images,” 2012. [Online]. Available: http://www.biomedcentral.com/1471-2105/13/316SOFTWARE

[36] S. J. Smith, “Q&A: Array tomography,” Sep. 06, 2018, *BioMed Central Ltd.* doi: 10.1186/s12915-018-0560-1.

[37] S. Preibisch, S. Saalfeld, and P. Tomancak, “Globally optimal stitching of tiled 3D microscopic image acquisitions,” *Bioinformatics*, vol. 25, no. 11, pp. 1463–1465, Jun. 2009, doi: 10.1093/bioinformatics/btp184.

[38] S. Saalfeld, R. Fetter, A. Cardona, and P. Tomancak, “Elastic volume reconstruction from series of ultra-thin microscopy sections,” *Nat Methods*, vol. 9, no. 7, pp. 717–720, Jul. 2012, doi: 10.1038/nmeth.2072.

[39] N. Nešić *et al.*, “Automated segmentation of cell organelles in volume electron microscopy using deep learning,” *Microsc Res Tech*, vol. 87, no. 8, pp. 1718–1732, Aug. 2024, doi: 10.1002/jemt.24548.

[40] B. He, Y. Zhang, F. Zhang, and R. Han, “Correction of image distortion in large-field ssEM stitching by an unsupervised intermediate-space solving network,” *Bioinformatics*, vol. 38, no. 20, pp. 4797–4805, Oct. 2022, doi: 10.1093/BIOINFORMATICS/BTAC566.

[41] E. Y. D. Chua, L. M. Alink, M. Kopylov, J. D. Johnston, F. Eisenstein, and A. de Marco, “Square beams for optimal tiling in transmission electron microscopy,” *Nat Methods*, vol. 21, no. 4, pp. 562–565, Apr. 2024, doi: 10.1038/S41592-023-02161-X.

[42] B. He, Y. Zhang, F. Zhang, and R. Han, “Correction of image distortion in large-field ssEM stitching by an unsupervised intermediate-space solving network,” *Bioinformatics*, vol. 38, no. 20, pp. 4797–4805, Oct. 2022, doi: 10.1093/BIOINFORMATICS/BTAC566.

[43] B. He *et al.*, “vEMstitch: an algorithm for fully automatic image stitching of volume electron microscopy,” *Gigascience*, vol. 13, p. giae076, Jan. 2024, doi: 10.1093/GIGASCIENCE/GIAE076.

[44] N. Lindow *et al.*, “Semi-automatic stitching of filamentous structures in image stacks from serial-section electron tomography,” *J Microsc*, vol. 284, no. 1, pp. 25–44, Oct. 2021, doi: 10.1111/jmi.13039.

[45] S. Preibisch, S. Saalfeld, and P. Tomancak, “Globally optimal stitching of tiled 3D microscopic image acquisitions,” *Bioinformatics*, vol. 25, no. 11, pp. 1463–1465, Jun. 2009, doi: 10.1093/bioinformatics/btp184.

[46] H. Peng *et al.*, “BrainAligner: 3D registration atlases of Drosophila brains,” *Nature Methods 2011 8:6*, vol. 8, no. 6, pp. 493–498, May 2011, doi: 10.1038/nmeth.1602.

[47] “Approaches to Registering Images.” Accessed: Jun. 09, 2025. [Online]. Available: https://www.mathworks.com/help/releases/R2021a/images/approaches-to-registering-images.html

[48] J. B. A. Maintz and M. A. Viergever, “A survey of medical image registration,” 1998.

[49] A. A. Tomaka, D. Pojda, M. Tarnawski, and L. Luchowski, “Transformation trees — Documentation of multimodal image registration,” *Comput Biol Med*, vol. 193, p. 110311, Jul. 2025, doi: 10.1016/j.compbiomed.2025.110311.

[50] S. Popovych *et al.*, “Petascale pipeline for precise alignment of images from serial section electron microscopy,” *Nat Commun*, vol. 15, no. 1, Dec. 2024, doi: 10.1038/s41467-023-44354-0.

[51] K. D. Micheva, J. J. Burden, and M. Schifferer, “Array tomography: trails to discovery,” *Methods in Microscopy*, vol. 1, no. 1, pp. 9–17, Aug. 2024, doi: 10.1515/mim-2024-0001.

[52] S. Popovych *et al.*, “Petascale pipeline for precise alignment of images from serial section electron microscopy,” *Nat Commun*, vol. 15, no. 1, pp. 1–15, Dec. 2024, doi: 10.1038/S41467-023-44354-0.

[53] J. Chen *et al.*, “A survey on deep learning in medical image registration: new technologies, uncertainty, evaluation metrics, and beyond,” Nov. 2024, [Online]. Available: http://arxiv.org/abs/2307.15615

[54] X. Zhang, H. Dong, D. Gao, and X. Zhao, “A Comparative Study for Non-rigid Image Registration and Rigid Image Registration,” Jan. 2020, [Online]. Available: http://arxiv.org/abs/2001.03831

[55] B. Rigaud *et al.*, “Deformable image registration for radiation therapy: principle, methods, applications and evaluation,” *Acta Oncol (Madr)*, vol. 58, no. 9, pp. 1225–1237, Sep. 2019, doi: 10.1080/0284186X.2019.1620331.

[56] S. Zhou *et al.*, “Fast and Accurate Electron Microscopy Image Registration with 3D Convolution,” 2019, pp. 478–486. doi: 10.1007/978-3-030-32239-7_53.

[57] G. Knott, H. Marchman, D. Wall, and B. Lich, “Serial Section Scanning Electron Microscopy of Adult Brain Tissue Using Focused Ion Beam Milling,” *The Journal of*

Neuroscience, vol. 28, no. 12, p. 2959, Mar. 2008, doi: 10.1523/JNEUROSCI.3189-07.2008.

[58] D. G. Lowe, "Object recognition from local scale-invariant features," *Proceedings of the IEEE International Conference on Computer Vision*, vol. 2, pp. 1150–1157, 1999, doi: 10.1109/ICCV.1999.790410.

[59] M. Kazhdan, K. Lillaney, W. Roncal, D. Bock, J. Vogelstein, and R. Burns, "Gradient-Domain Fusion for Color Correction in Large EM Image Stacks," Jun. 2015, Accessed: Jun. 09, 2025. [Online]. Available: https://arxiv.org/pdf/1506.02079

[60] S. K. Abdulateef and M. D. Salman, "A Comprehensive Review of Image Segmentation Techniques," *Iraqi Journal for Electrical and Electronic Engineering*, vol. 17, no. 2, pp. 166–175, Dec. 2021, doi: 10.37917/ijeee.17.2.18.

[61] S. A. Taghanaki, K. Abhishek, J. P. Cohen, J. Cohen-Adad, and G. Hamarneh, "Deep Semantic Segmentation of Natural and Medical Images: A Review," Mar. 2024, [Online]. Available: http://arxiv.org/abs/1910.07655

[62] B. Gallusser *et al.*, "Deep neural network automated segmentation of cellular structures in volume electron microscopy," *Journal of Cell Biology*, vol. 222, no. 2, Feb. 2023, doi: 10.1083/JCB.202208005.

[63] K. P. Patra and J. M. Vinetz, "New Ultrastructural Analysis of the Invasive Apparatus of the Plasmodium Ookinete," *Am J Trop Med Hyg*, vol. 87, no. 3, p. 412, Sep. 2012, doi: 10.4269/AJTMH.2012.11-0609.

[64] F. Li, K. P. Patra, and J. M. Vinetz, "An anti-chitinase malaria transmission-blocking single-chain antibody as an effector molecule for creating a Plasmodium falciparum-refractory mosquito," *Journal of Infectious Diseases*, vol. 192, no. 5, pp. 878–887, Sep. 2005, doi: 10.1086/432552/2/M_192-5-878-FIG001.GIF.

[65] F. Li *et al.*, "Plasmodium ookinete-secreted proteins secreted through a common micronemal pathway are targets of blocking malaria transmission," *Journal of Biological Chemistry*, vol. 279, no. 25, pp. 26635–26644, Jun. 2004, doi: 10.1074/jbc.M401385200.

[66] J. M. Santos *et al.*, "Maternally supplied S-acyl-transferase is required for crystalloid organelle formation and transmission of the malaria parasite," *Proc Natl Acad Sci U S A*, vol. 113, no. 26, pp. 7183–7188, Jun. 2016, doi: 10.1073/PNAS.1522381113.

[67] T. Dalapati and J. M. Moore, "Hemozoin: a Complex Molecule with Complex Activities", doi: 10.1007/s40588-021-00166-8.

[68] A. Khadangi, T. Boudier, and V. Rajagopal, “EM-stellar: benchmarking deep learning for electron microscopy image segmentation,” *Bioinformatics*, vol. 37, no. 1, pp. 97–106, Jan. 2021, doi: 10.1093/BIOINFORMATICS/BTAA1094,.

[69] R. Conrad and K. Narayan, “Instance segmentation of mitochondria in electron microscopy images with a generalist deep learning model trained on a diverse dataset,” *Cell Syst*, vol. 14, no. 1, p. 58, Jan. 2023, doi: 10.1016/J.CELS.2022.12.006.

[70] “C. elegans II - PubMed.” Accessed: Jun. 09, 2025. [Online]. Available: https://pubmed.ncbi.nlm.nih.gov/21413221/

[71] B. Gallusser *et al.*, “Deep neural network automated segmentation of cellular structures in volume electron microscopy,” *Journal of Cell Biology*, vol. 222, no. 2, Feb. 2023, doi: 10.1083/JCB.202208005,.

[72] X. Liu, L. Qu, Z. Xie, J. Zhao, Y. Shi, and Z. Song, “Towards more precise automatic analysis: a systematic review of deep learning-based multi-organ segmentation,” Dec. 01, 2024, *BioMed Central Ltd.* doi: 10.1186/s12938-024-01238-8.

[73] B. Chai, C. Efstathiou, H. Yue, and V. M. Draviam, “Opportunities and challenges for deep learning in cell dynamics research,” *Trends Cell Biol*, vol. 34, no. 11, pp. 955–967, Nov. 2023, doi: 10.1016/J.TCB.2023.10.010.

[74] S. Van Der Walt *et al.*, “Scikit-image: Image processing in python,” *PeerJ*, vol. 2014, no. 1, 2014, doi: 10.7717/peerj.453.

[75] “Get Started.” Accessed: Jul. 24, 2025. [Online]. Available: https://pytorch.org/get-started/locally/

[76] “Tutorials | TensorFlow Core.” Accessed: Jul. 24, 2025. [Online]. Available: https://www.tensorflow.org/tutorials

[77] “Getting started with Keras.” Accessed: Jul. 24, 2025. [Online]. Available: https://keras.io/getting_started/

[78] A. Khadangi, T. Boudier, and V. Rajagopal, “EM-stellar: benchmarking deep learning for electron microscopy image segmentation,” *Bioinformatics*, vol. 37, no. 1, pp. 97–106, Jan. 2021, doi: 10.1093/BIOINFORMATICS/BTAA1094,.

[79] S. Ambatipudi and S. Byna, “A Comparison of HDF5, Zarr, and netCDF4 in Performing Common I/O Operations,” Feb. 2023, [Online]. Available: http://arxiv.org/abs/2207.09503

[80] A. Miles *et al.*, “zarr-developers/zarr-python: v2.5.0”, doi: 10.5281/ZENODO.4069231.

[81] D. Rotem, E. Otoo, S. Seshadri, and L. N. Stern, “Optimal Chunking of Large Multidimensional Arrays for Data Warehousing.”

[82] J. Moore *et al.*, “OME-Zarr: a cloud-optimized bioimaging file format with international community support,” *bioRxiv*, p. 2023.02.17.528834, Feb. 2023, doi: 10.1101/2023.02.17.528834.

[83] F. Rzepka, P. Hematty, M. Schmitz, and J. Kowal, “Neural Network Architecture for Determining the Aging of Stationary Storage Systems in Smart Grids,” *Energies (Basel)*, vol. 16, no. 17, Sep. 2023, doi: 10.3390/en16176103.

[84] N. Rzepka, J. A. Bogovic, and J. A. Moore, “Toward scalable reuse of vEM data: OME-Zarr to the rescue,” *Methods Cell Biol*, vol. 177, pp. 359–387, Jan. 2023, doi: 10.1016/bs.mcb.2023.01.016.

[85] P. Bajcsy *et al.*, “Enabling global image data sharing in the life sciences,” *Nat Methods*, vol. 22, no. 4, pp. 672–676, Apr. 2025, doi: 10.1038/s41592-024-02585-z.

[86] J. Moore *et al.*, “OME-NGFF: a next-generation file format for expanding bioimaging data-access strategies,” *Nat Methods*, vol. 18, no. 12, pp. 1496–1498, Dec. 2021, doi: 10.1038/S41592-021-01326-W.

[87] I. Kemmer *et al.*, “Building a FAIR image data ecosystem for microscopy communities,” Sep. 01, 2023, *Springer Science and Business Media Deutschland GmbH*. doi: 10.1007/s00418-023-02203-7.

[88] N. Rzepka, J. A. Bogovic, and J. A. Moore, “Toward scalable reuse of vEM data: OME-Zarr to the rescue,” in *Methods in Cell Biology*, vol. 177, Academic Press Inc., 2023, pp. 359–387. doi: 10.1016/bs.mcb.2023.01.016.

[89] napari contributors, “napari: a multi-dimensional image viewer for python,” 2019, doi: doi:10.5281/zenodo.3555620.

[90] “Nextflow — Nextflow documentation.” Accessed: Jul. 25, 2025. [Online]. Available: https://www.nextflow.io/docs/latest/index.html#citations

[91] S. Baichoo *et al.*, “Developing reproducible bioinformatics analysis workflows for heterogeneous computing environments to support African genomics,” *BMC Bioinformatics*, vol. 19, no. 1, Nov. 2018, doi: 10.1186/s12859-018-2446-1.

[92] “Overview — Nextflow documentation.” Accessed: Jun. 09, 2025. [Online]. Available: https://www.nextflow.io/docs/latest/overview.html

[93] C. Schmitt, B. Yu, and T. Kuhr, “A Workflow Management System Guide,” Sep. 2023, [Online]. Available: http://arxiv.org/abs/2212.01422

[94] “Introduction to Nextflow - Reproducible Bioinformatics Workflows with Nextflow and nf-core.” Accessed: Jul. 25, 2025. [Online]. Available: https://genomicsaotearoa.github.io/Nextflow_Workshop/session_1/1_introduction/

[95] “tifffile · PyPI.” Accessed: Jun. 09, 2025. [Online]. Available: https://pypi.org/project/tifffile/

[96] A. Cardona *et al.*, “TrakEM2 software for neural circuit reconstruction,” *PLoS One*, vol. 7, no. 6, Jun. 2012, doi: 10.1371/journal.pone.0038011.

[97] “Registration Overview — SimpleITK 2.4.0 documentation.” Accessed: Jul. 25, 2025. [Online]. Available: https://simpleitk.readthedocs.io/en/master/registrationOverview.html

[98] T. Xin, L. Shen, L. Li, X. Chen, and H. Han, “Expected affine: A registration method for damaged section in serial sections electron microscopy,” *Front Neuroinform*, vol. 16, Sep. 2022, doi: 10.3389/fninf.2022.944050.

[99] A. Cardona *et al.*, “TrakEM2 software for neural circuit reconstruction,” *PLoS One*, vol. 7, no. 6, Jun. 2012, doi: 10.1371/journal.pone.0038011.

[100] “GitHub - facebookresearch/segment-anything: The repository provides code for running inference with the SegmentAnything Model (SAM), links for downloading the trained model checkpoints, and example notebooks that show how to use the model.” Accessed: Jul. 25, 2025. [Online]. Available: https://github.com/facebookresearch/segment-anything?tab=readme-ov-file#citing-segment-anything

[101] “GitHub - Krishvraman/napari-SAMV2: Napari plugin to use SAM V2 in napari.” Accessed: Jun. 09, 2025. [Online]. Available: https://github.com/Krishvraman/napari-SAMV2

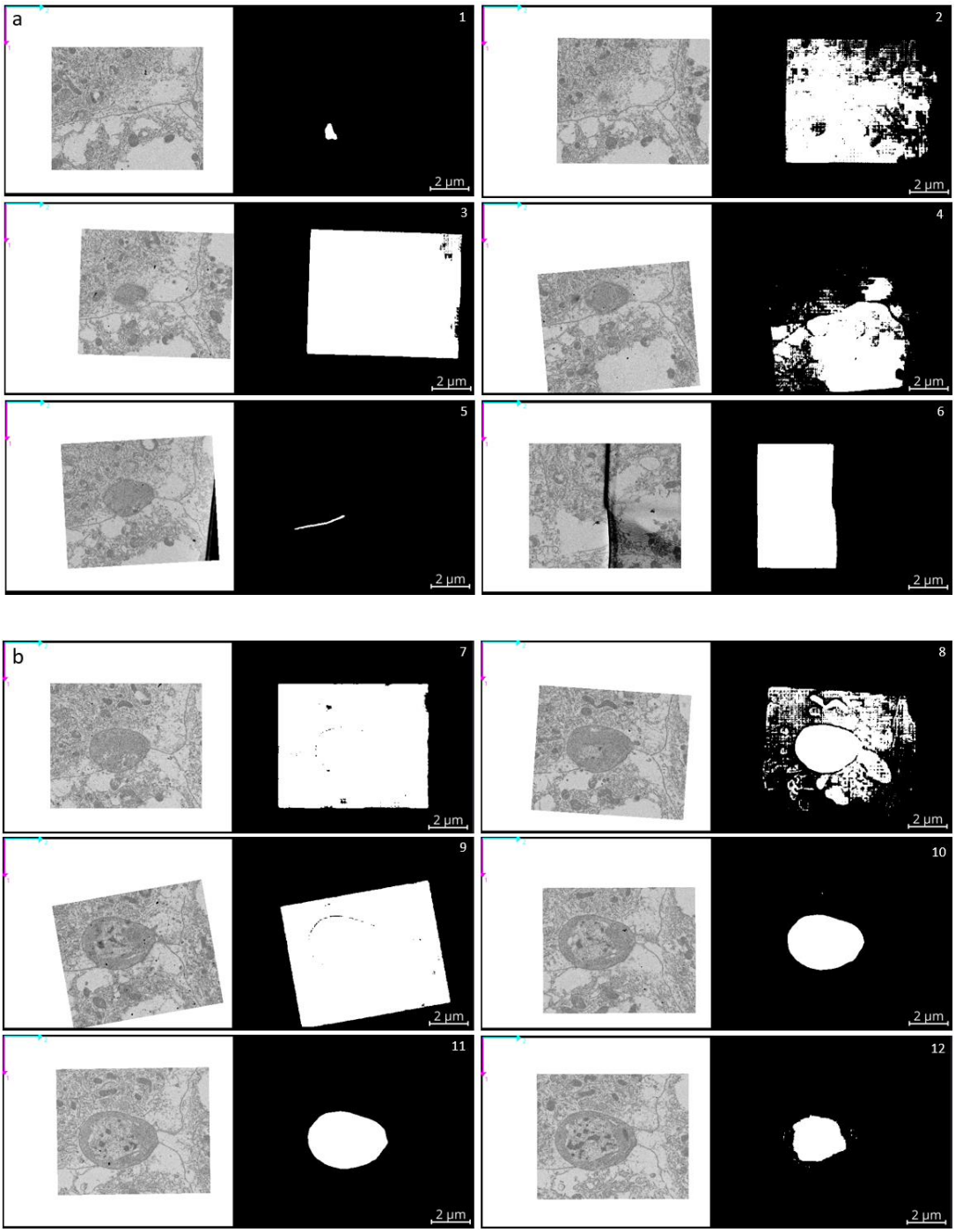
[102] “Helmholtz Imaging CONNECT - Napari - Segment Anything Model (SAM).” Accessed: Jun. 09, 2025. [Online]. Available: https://connect.helmholtz-imaging.de/solution/67

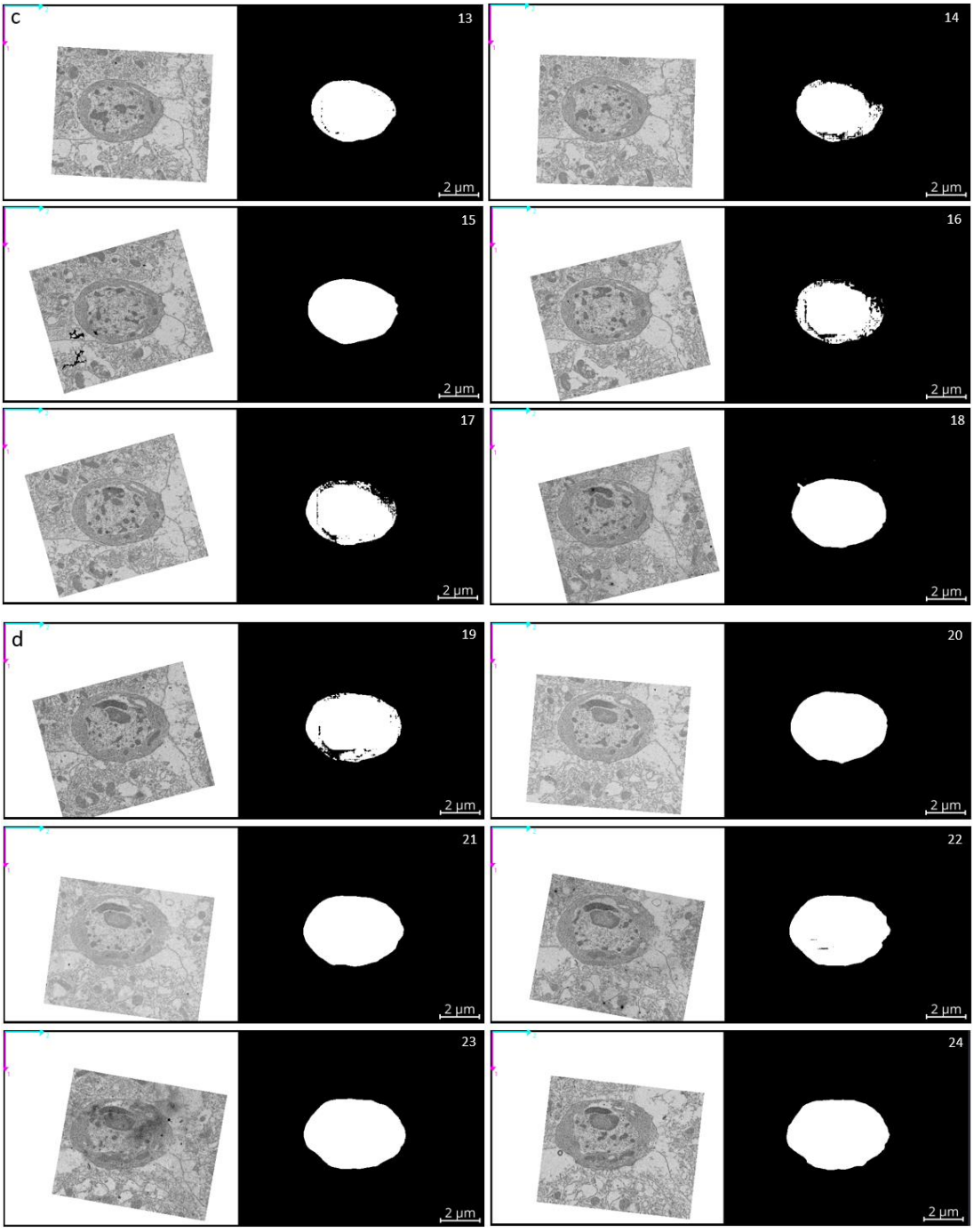
[103] “distance_transform_edt — SciPy v1.16.0 Manual.” Accessed: Jul. 25, 2025. [Online]. Available: https://docs.scipy.org/doc/scipy/reference/generated/scipy.ndimage.distance_transform_edt.html#scipy.ndimage.distance_transform_edt

[104] “KDTree — scikit-learn 1.7.1 documentation.” Accessed: Jul. 25, 2025. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KDTree.html

Appendix I

Together, the 38 slice results highlight the inconsistent and unreliable nature of direct SAM application on vEM data, especially for biological structures with subtle boundaries and varying morphology.




c
13
2 µm
14
2 µm
15
2 µm
16
2 µm
17
2 µm
18
2 µm
d
19
2 µm
20
2 µm
21
2 µm
22
2 µm
23
2 µm
24
2 µm

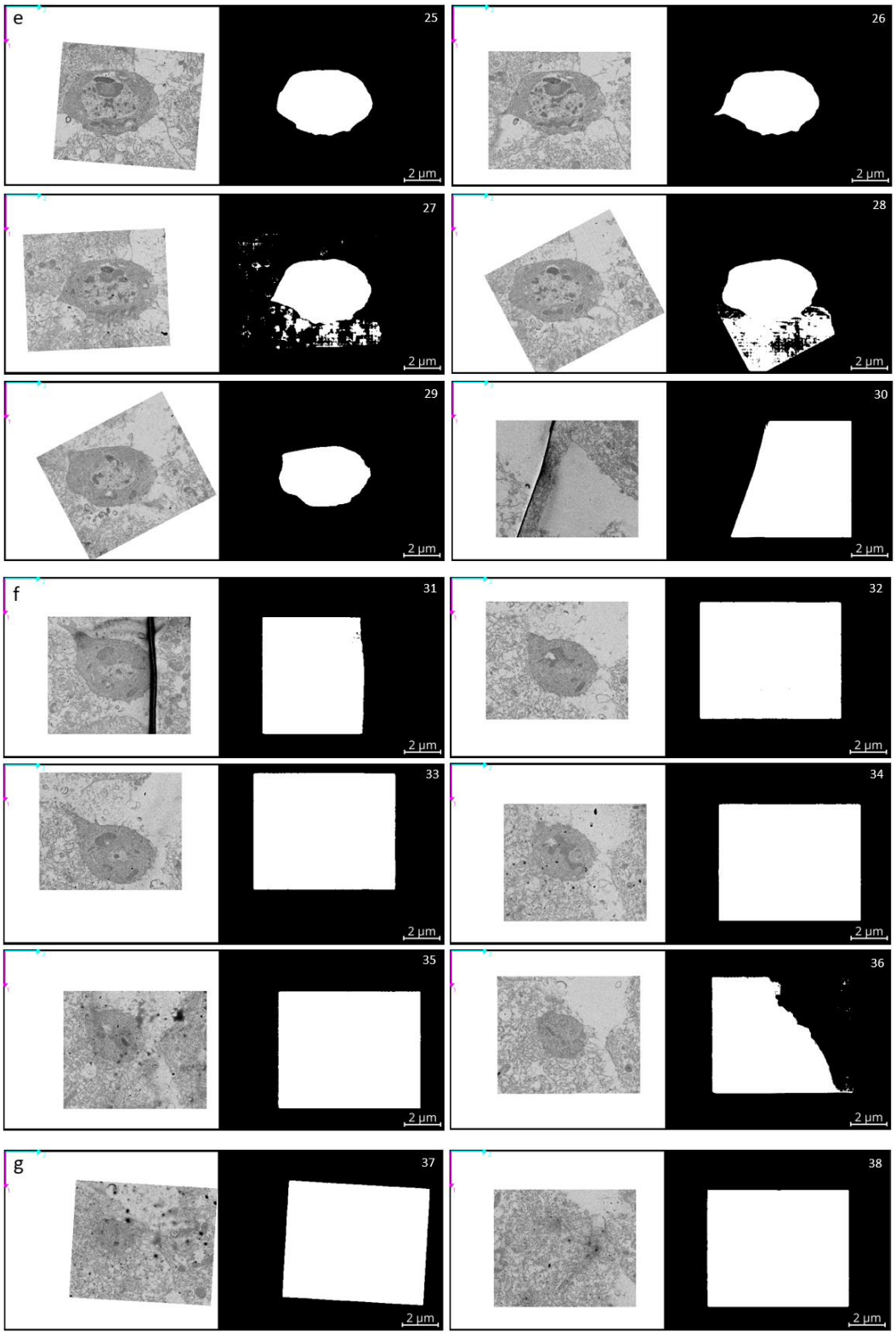


Figure (a-g). Segmentation results of a 38-slice 3D volume of a *Plasmodium* ookinete from a gut sample using the SAM. Each row presents pairs of original EM slices (left) and the corresponding binary masks predicted by SAM (right). While mid-slices captured the parasite boundaries with moderate accuracy, whereas slices near the top and bottom—where boundaries are often less distinct—frequently resulted in poor or fragmented masks. SAM also showed sensitivity to imaging artifacts, at times mislabeling background textures or ignoring the parasite entirely. These examples highlight SAM's limitations when applied directly to complex EM data.