

SAMJ: Fast Image Annotation on ImageJ/Fiji via Segment Anything Model

Carlos García-López-de-Haro^{1,2,3}, Caterina Fuster-Barceló^{1,2},
Curtis T. Rueden⁴, Jónathan Heras⁵, Vladimír Ulman^{6,7},
Daniel Franco-Barranco^{8,9,10}, Adrián Inés⁵, Kevin W. Eliceiri⁴,
Jean-Christophe Olivo-Marin^{1, 3, 11}, Jean-Yves Tinevez¹²,
Daniel Sage^{13*}, Arrate Muñoz-Barrutia^{1,2*}

¹Bioengineering Department, Universidad Carlos III de Madrid,
Leganes, Spain.

²Instituto de Investigación Sanitaria Gregorio Marañón, Madrid, Spain.

³Bioimage Analysis Unit, Institut Pasteur, Université Paris Cité, Paris,
France.

⁴Center for Quantitative Cell Imaging, University of Wisconsin,
Madison, USA.

⁵Department of Mathematics and Computer Science, University of La
Rioja, Logroño, Spain.

⁶IT4Innovations, VSB – Technical University of Ostrava, Ostrava, Czech
Republic.

⁷Central European Institute of Technology (CEITEC), Masaryk
University, Brno, Czech Republic.

⁸MRC Laboratory of Molecular Biology, Cambridge, UK.

⁹Department of Physiology, Development and Neuroscience, University
of Cambridge, Cambridge, UK.

¹⁰Donostia International Physics Center (DIPC), San Sebastian, Spain.

¹¹CNRS UMR 3691, Institut Pasteur, Paris, France.

¹²Image Analysis Hub, Institut Pasteur, Université Paris Cité, Paris,
France.

¹³Biomedical Imaging Group and Center for Imaging, Ecole
Polytechnique Fédérale de Lausanne (EPFL), Lausanne, Switzerland.

Abstract

Mask annotation remains a significant bottleneck in AI-driven biomedical image analysis due to its labor-intensive nature. To address this challenge, we introduce SAMJ, a user-friendly ImageJ/Fiji plugin leveraging the Segment Anything Model (SAM). SAMJ enables seamless, interactive annotations with one-click installation on standard computers. Designed for real-time object delineation in large scientific images, SAMJ is an easy-to-use solution that simplifies and accelerates the creation of labeled image datasets.

Keywords: Image Annotation, Foundation Model, Biomedical Images, Java-based platform

Access to high-quality annotated images is key to training and developing new methods for biomedical image analysis and object segmentation. However, creating large datasets of annotated images is a time-consuming, labor-intensive, and subjective process [1]. The demand becomes even more pronounced when training supervised Deep Learning (DL) models, typically requiring hundreds to thousands of annotated images [2]. Therefore, there is a critical need for efficient annotation tools capable of running on commonly available computers. Such tools must reduce IT-related barriers, ensuring broader accessibility for domain-specific experts such as biologists.

These solutions should also be compatible with widely used bioimage software platforms, such as ImageJ/Fiji [3], and must be fast enough to accelerate the annotation of dozens of objects in large images, which are typical in biological research and essential for training machine learning models.

Recently, Meta Research released the Segment Anything Model (SAM) [4], a foundational DL framework designed for one-shot and promptable image segmentation. SAM represents a significant breakthrough, capable of delineating complex objects and enabling semi-automated, interactive annotation across diverse image types. In its standard mode of operation, SAM requires user-defined prompts, such as points, sets of points, or bounding boxes, to efficiently isolate target objects.

SAM has been made available in commonly used bioimage software packages, such as QuPath [5] and Napari [6], providing interactive annotation, but has not yet been incorporated into the ImageJ ecosystem [7]. Other proprietary software, such as Matlab, ArcGIS, Kili Technology, or Unitlab Magic Touch have incorporated SAM as an add-on tool to enhance their segmentation capabilities. Similarly, Segment Anything Annotator¹, an open-source Python-based UI application, uses SAM for pixel-level annotation.

Although many annotation tools are both user-friendly and widely accessible, they often face challenges when dealing with uneven illumination, overlapping structures, or subtle intensity gradients. Several plugins have introduced more advanced methods: AnnotatorJ [8], a popular Fiji plugin, supports semi-automatic cellular annotation by combining manual input with deep learning; Labkit [9] leverages machine learning for pixel-wise classification; and Thomas et al. [10] propose a streamlined approach to

¹<https://github.com/haochenhededa/segment-anything-annotator>

systematic manual annotation by assigning predefined categories to images or regions. However, none of these plugins yet benefit from the advantages that SAM provides.

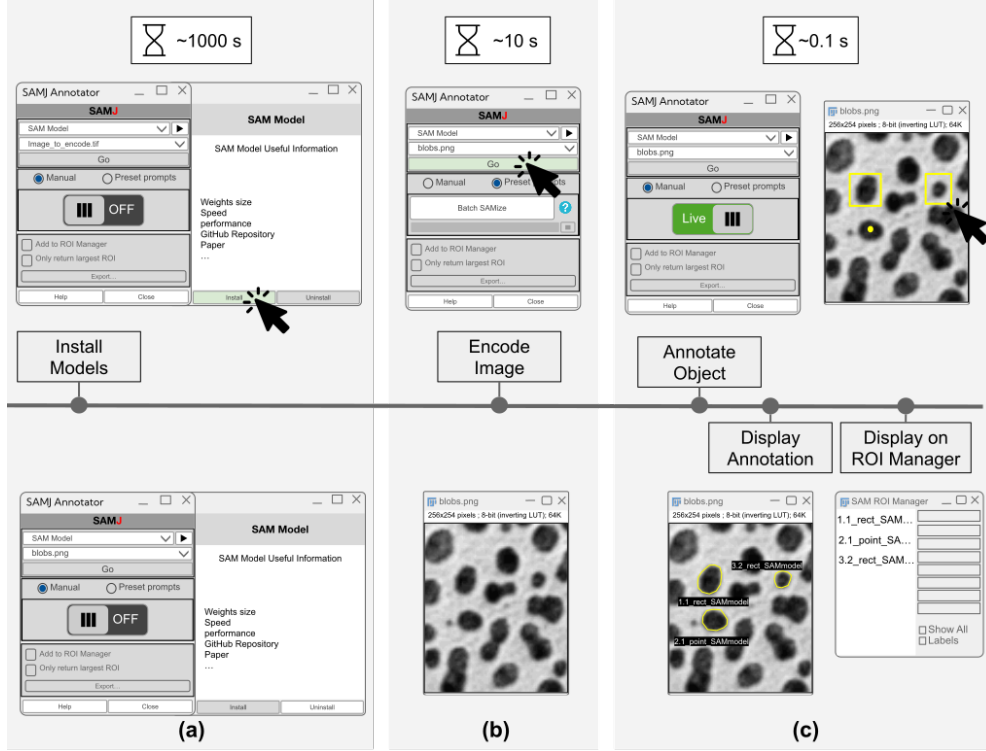


Figure 1 SAMJ User Interface and Workflow with estimated time. This figure illustrates the typical workflow of SAMJ for annotating objects in an image, including the time required for each step on standard workstations. (a) Model Installation: The most time-consuming step, where SAMJ installs the selected model and, if necessary, sets up the environment, taking approximately 1000 seconds. (b) Image Encoding: When the user clicks “Go” in the plugin, the image is processed to generate an embedding, which takes around 10 seconds. (c) Object Annotation: Since no re-embedding is required, this step is immediate, with each annotation generated in approximately 0.1 seconds per user click. The annotation process can be repeated multiple times as desired for different objects, offering rapid and interactive segmentation.

In this manuscript, we present SAMJ, a plugin that brings the capabilities of SAM into the Fiji ecosystem, both as a standalone Fiji plugin and as an extension for Labkit[9]. We designed SAMJ for one-click installation. It features a no-code interface, and runs on standard computers without requiring specialized hardware. By lowering technical barriers, it broadens access to cutting-edge AI tools within the life sciences community. A key contribution of our work is the tailored adaptation of SAMJ to efficiently handle scientific images. By harnessing the powerful segmentation capabilities of SAM, this tool has the potential to significantly accelerate the annotation process,

enabling users to instantly generate regions of interest around target objects with a single point click or bounding box.

SAMJ leverages the core architecture of SAM, which is built upon three key components: an image encoder, a prompt encoder, and a mask decoder. The image encoder is based on a Visual Transformer (ViT) [11] that encodes the input image, offering high representational power at the expense of significant computational resources. The prompt encoder is another transformer to encode the prompt given by the user to point or “hint” to the location of the object of interest. Finally, the encoded image and prompt are used by the mask decoder, which combines elements of both a transformer and a Convolutional Neural Network, to generate the mask of interest.

The SAMJ plugin is installed as any standard Fiji extension, eliminating the technical IT barriers often associated with Python packages and making advanced image segmentation accessible to a wider audience. Unlike most Python-based libraries that require command-line interfaces for environment configuration and dependency management, SAMJ simplifies the process with a seamless, one-click installation, enabling effortless deployment.

In addition, five SAM variants are available through SAMJ. Each of these models can run on standard workstations or mid-range laptops, eliminating the need for high-end computational resources that the original SAM typically requires. These variants are SAM-2 [12], EfficientSAM [13], and EfficientViTSAM [14].

The annotation process in the SAMJ plugin mirrors SAM’s workflow (Figure 1). Each image is encoded once by the image encoder, the most computationally intensive step. Afterwards, prompts are processed and masks generated almost instantly, allowing users to interactively produce multiple annotations with real-time responsiveness. SAMJ integrates seamlessly with Fiji’s existing tools, allowing users to create prompts such as points and rectangles directly through Fiji’s familiar toolbar, thereby minimizing the learning curve for new users.

The SAMJ plugin offers two annotation modes: Live and Batch mode (called BatchSAMize). In Live mode, prompts are drawn directly on the encoded image, and annotations are generated interactively, one at a time. In Batch mode, users can define multiple seed points to serve as prompts for segmenting multiple objects in a raw image. These seeds can be generated automatically using traditional Fiji commands (e.g., Find Maxima, Watershed) within an ImageJ macro, followed by a call to SAMJ’s Batch mode to segment the whole image. Additionally, users can enhance annotations by providing seeds from other segmentation methods as prompts, enabling iterative refinement.

Beyond its ease of use, SAMJ is designed for adaptability and seamless integration. Powered by SAM, it can enhance annotation workflows across a range of Java-based platforms, for example, in addition to Labkit, we have also integrated it into BigDataViewer. [15]. Moreover, to support broader adoption, SAMJ provides a well-documented API, enabling developers to easily incorporate SAM functionality into their preferred Java software effortlessly. Furthermore, it includes a software-agnostic Java GUI, facilitating straightforward integration into any Java-based environment.

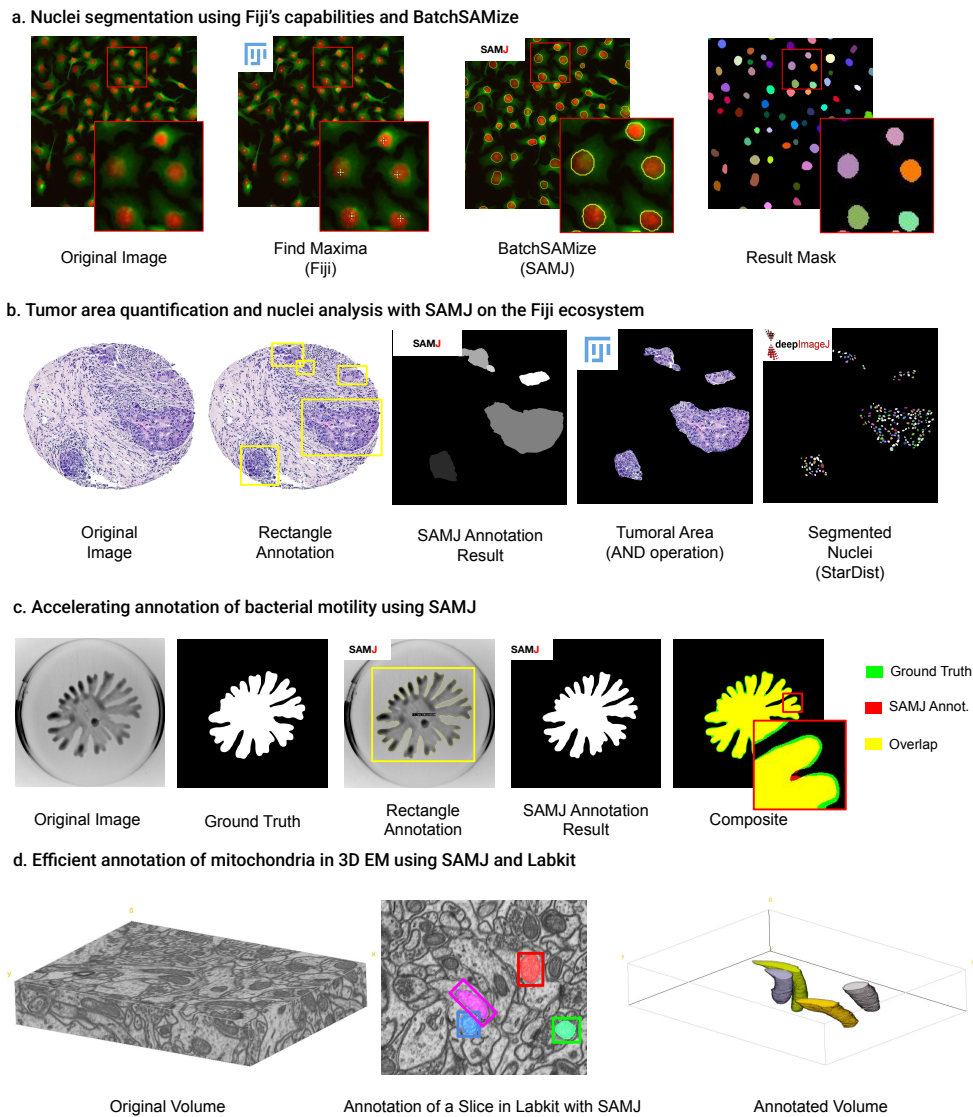


Figure 2 Use cases demonstrating the versatility of SAMJ for bioimage analysis. (a) Nuclei Segmentation using Fiji's capabilities and BatchSAMize: An image from the CellPose dataset [16] is used to demonstrate the segmentation of nuclei in the red channel. Pixel intensity maxima are identified in Fiji to generate single-point prompts for each nucleus. These prompts are processed in batch mode using SAMJ, resulting in semantic segmentation of individual nuclei. (b) Tumor Area Quantification and Nuclei Analysis with SAMJ and StarDist plugins for Fiji: Breast cancer TMA images, stained with H&E provided by the British Columbia Cancer Agency (BCCA) [17], are used to quantify tumoral regions and their nuclei. Tumoral areas are annotated with SAMJ's rectangle prompt, generating masks that are combined with the original image in Fiji using an AND operation to obtain the intersection. StarDist is then applied through deepImageJ to segment individual nuclei in the tumoral areas. (c) Accelerating annotation of Bacterial Motility: SAMJ is applied to annotate motile bacteria using a single rectangle annotation. When compared to Ground Truth, SAMJ's annotations achieve comparable or superior precision, showcasing its efficiency in handling complex shapes and streamlining high-throughput workflows. Each use case highlights SAMJ's integration with Fiji, combining SAM's advanced annotation capabilities with Fiji's extensive image processing tools. (d) Efficient annotation of 3D Electron Microscopy images of mitochondria [18] using SAMJ and Labkit: SAMJ is integrated into Labkit to support 3D and multi-label annotation. Users can annotate structures on arbitrarily oriented and scaled slices, improving visibility and accuracy of objects with complex spatial orientation. The multi-class labeling capability of Labkit allows the annotation of several distinct structures within the same volume. This workflow reduces annotation effort and ensures spatial consistency across slices.

A key strength of SAMJ is its integration of Python methods into Java environments. This is achieved through Appose², a Java package that allows Java and Python to run as separate processes yet communicate in real time. Leveraging Micromamba³, Appose also automates Python environment setup, totally eliminating manual configuration via the command line. As a result, SAMJ is remarkably user-friendly, granting effortless access to advanced methods.

To illustrate the power of SAMJ and its seamless integration with the Fiji ecosystem, we present four representative use cases highlighting its capabilities (Figure 2). These examples illustrate SAMJ’s ability to adapt to diverse image annotation tasks while taking advantage of Fiji’s powerful image processing tools. In all cases, the targets are compact and well-defined—conditions under which SAMJ performs particularly well. Elongated or branched structures may pose more of a challenge; However, users can refine prompts, adjust scale settings, or combine SAMJ with complementary methods. Experimentation is encouraged, as SAMJ often proves effective even under suboptimal imaging conditions.

In the first use case, we illustrate nuclei segmentation in a fluorescence image using the BatchSAMize mode of SAMJ. Classical Fiji commands are used to detect nuclei, which serve as single-point prompts for batch processing in SAMJ. This workflow highlights the synergy between the preprocessing capabilities of Fiji and the segmentation power of SAMJ, streamlining large-scale annotation tasks.

The second use case intended for tumor area quantification focuses on breast cancer TMA images stained with H&E. The task involved quantifying tumoral areas and the number of nuclei within these regions. Using SAMJ’s rectangle annotation feature, tumoral regions were outlined, generating masks for these areas. Subsequently, individual nuclei were segmented with StarDist [19] using the deepImageJ [20] [7] plugin. This workflow demonstrates SAMJ’s role as a flexible and efficient annotation tool that seamlessly integrates with Fiji, enabling efficient annotation and precise quantification for tumor analysis.

The third use case highlights SAMJ’s efficiency in annotating complex shapes [21]. Bacteria from motility studies were annotated using a single rectangle prompt per image, significantly reducing manual effort. When compared to manually generated ground truth, SAMJ’s annotations are visually indistinguishable. This use case demonstrates how SAMJ accelerates annotation tasks for intricate biological structures, making it a powerful tool for high-throughput studies.

The final use case demonstrates SAMJ’s capability for efficient annotation of mitochondria instances in 3D electron microscopy volumes through its integration with Labkit. Annotating objects in 3D is inherently challenging due to the difficulty of maintaining spatial coherence across slices and the substantial effort required for manual annotation. SAMJ streamlines this process, and its integration into Labkit—designed for interactive 3D visualization—enables users to annotate structures on arbitrarily oriented slices, ensuring optimal views of the object of interest. Moreover, Labkit support for multi-class labeling allows users to assign distinct labels to multiple structures within the same volume. This combination significantly reduces annotation effort while

²<https://github.com/apposed/appose>

³https://mamba.readthedocs.io/en/latest/user_guide/micromamba.html

improving both accuracy and consistency, highlighting the power of integrating SAMJ with complementary Fiji tools for complex 3D bioimage annotation tasks.

Together, these use cases highlight the versatility and strength of SAMJ integration within the Fiji ecosystem. By combining the advanced segmentation capabilities of SAM with the rich set of image processing tools of Fiji, SAMJ enables more efficient, precise, and scalable annotation workflows across a wide range of biological imaging tasks. This integration lowers technical barriers and supports diverse use cases, from 2D fluorescence images to complex 3D volumes. Consequently, it empowers the broader life science community—particularly biologists who primarily rely on GUI software platforms—with powerful deep-learning tools that would otherwise remain accessible only to a few accelerating and improving their bioimage analysis.

Accurate annotation is fundamental to deep learning-based image analysis, playing a critical role in model training, validation, and fine-tuning. To address the limitations posed by the high embedding latency of SAM, particularly in large microscopy images, we developed SAMJ, an intuitive and interactive annotation tool tailored for the life sciences. SAMJ integrates efficient SAM2 models optimized to run on mid-range computers and includes a Java-Python bridge to eliminate complex installation steps. As an ImageJ/Fiji plugin, it remains fully compatible with standard Fiji tools such as ROI selection, the ROI manager, and macros. Through this seamless integration, SAMJ enables efficient, high-quality annotations and supports the creation of curated datasets, thus contributing to the development of more accurate and robust models.

In conclusion, SAMJ provides a user-friendly and accessible way for biologists and bioimage analysts to adopt cutting-edge AI methods, facilitating faster and more precise image annotation and ultimately accelerating scientific discovery.

Acknowledgements. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Council Executive Agency. Neither the European Union nor the granting authority can be held responsible for them.

Author contributions. Conceptualization: D.S., C.G-L-H. and A.M-B. Software: C.G-L-H., V.U., D.S and C.T.R. Interface design: D.S., C.G-L-H., C.T.R. and C.F-B. Validation and testing: C.G-L-H., C.F-B., V.U., D.S and D.F-B. Writing: C.G-L-H., C.F-B., V.U., A.M-B. and D.S. Writing—review and editing: all authors. Supervision: A.M-B. and D.S.

Funding statement. This work was partially funded by the project CNRS France 2030 Recherche à Risque (RI2) ProteoVir (J-C.O-M. and C.G-L-H). This work was partially supported by the European Union’s Horizon Europe research and program under grant agreement number 101057970 (AI4Life project) awarded to A.M-B. and the Ministerio de Ciencia, Innovación y Universidades, Agencia Estatal de Investigación, MCIN/AEI/10.13039/501100011033/, under grants PID2019-109820RB and PID2023-152631OB-I00, co-financed by European Regional Development Fund (ERDF), “A way of making Europe”. This work has been funded in part by the Agence Nationale de la Recherche through the programs PR[AI]RIE-PSAI

(ANR-23-IACL-0008) and France-BioImaging infrastructure (ANR-10-INBS-04) (J-C.O-M. and J-Y.T.). This project has received funding from the Innovative Medicines Initiative 2 Joint Undertaking under grant agreement No 945358 (BIGPICTURE). This Joint Undertaking receives support from the European Union’s Horizon 2020 research and innovation program and EFPIA (J-C.O-M.). J-C.O-M and J-Y.T were also supported by the French National Research Agency (France BioImaging, ANR-24-INBS-0005 FBI BIOGEN). V.U. was supported by grant number 2024-342803 from the Chan Zuckerberg Initiative DAF, an advised fund of Silicon Valley Community Foundation. C.T.R. and K.W.E were funded by the National Institutes of Health, grant number NIH P41 GM135019. J. H. and A. I. were supported by Agencia de Desarrollo Económico de La Rioja ADER 2022-I-IDI-00015 and by project AFIANZA 2024/01 granted by the Autonomous Community of La Rioja. J. H. was also supported by the Government of La Rioja through Proyecto Inicia 2023/01.

Appendix A Online Methods

A.1 Review of Annotation Tools for Image Segmentation

We provide a comprehensive overview of the existing annotation tools for bioimage analysis, summarizing key features, such as their release dates, supported annotation types, and integration with the Segment Anything Model (SAM) or any other AI assistant (Table [Table A.1](#)). This table highlights the diversity of tools available across platforms like Fiji, QuPath, and Napari, as well as their varying levels of SAM integration. Notably, while tools such as QuPath and Napari have embraced SAM, Fiji-based plugins like AnnotatorJ lack direct SAM support. SAMJ addresses this gap by bringing SAM’s advanced annotation capabilities to the Fiji environment, combining flexibility with ease of use. This comparative analysis underscores SAMJ’s unique contribution to enabling SAM-powered annotation within the Fiji ecosystem.

A.2 Available SAM Variants

The original SAM model is computationally intensive and was initially introduced in three versions, differentiated by the size of the Vision Transformer (ViT): base, large, and huge. However, even the smallest model, SAM-base, is too heavy to run on standard CPUs due to its high resource demands.

To address these limitations, newer and more lightweight variants have been developed. These models (efficientSAM [\[13\]](#); efficientViTSAM [\[14\]](#)) achieve a significant improvement in computational efficiency with only a minimal trade-off in annotation quality, making SAM-like models accessible for lower-end CPUs.

Moreover, the release of the second generation of the original SAM, known as SAM-2 [\[12\]](#), marked a breakthrough. This new family of models not only enhances annotation quality but also significantly reduces computational costs.

For these reasons, the list of models available in SAMJ does not include any of the original SAM versions. This list is expected to evolve as new models achieve improvements in performance and efficiency. Currently, the supported models in SAMJ include: **SAM-2 (tiny, small, and large)**, **EfficientSAM**, and **EfficientViTSAM**.

Tool/ Plugin	Platform	Release	Last Update	AI Assistance	Annotation	Programming Language
MicroSAM	Napari	2023	03/2025	Yes (SAM fine-tune)	Polygons BBox Points	Python
QuPath Ext. SAM	QuPath	2023	09/2024	Yes (SAM2)	AutoMask BBox Points	Python
Napari Plugin of SAM	Napari	2023	04/2023	Yes (SAM1)	BBox Points	Python
Napari Plugin of SAM2	Napari	2024	09/2024	Yes (SAM2)	BBox Points	Python
AnnotatorJ	Fiji	2020	10/2020	Yes (CNN)	Instance Semantic BBox	Java
Labkit	Fiji	2022	10/2024	Yes (Random Forest SAM1,2)	Pixel Classification	Java
Qualitative Annotations	Fiji	2020	04/2021	Yes (CNN)	One or Multiple ROI	Python
SAMJ	Fiji	2025	04/2025	Yes (SAM1,2)	BBox Points	Java

Table A.1 Summary of tools integrating annotation capabilities in bioimage analysis platforms.

The table highlights the main tools and plugins used for annotation in various bioimage analysis platforms, detailing their release (YYYY) and last update dates (MM/YYYY, Last accessed: 20MAY2025), annotation features, integration with the SAM, and the programming languages used for their development. SAM integration is specified with its supported annotation types, including bounding boxes (BBox), points, polygons, and more, showcasing the variety of annotation modes available.

For the most up-to-date list, please refer to the official SAMJ GitHub repository: <https://github.com/segment-anything-models-java/SAMJ/blob/main/README.md>.

A.3 SAMJ Embedding Strategy

To generate annotations, SAM must first embed the image, or at least a portion of it. This step creates an embedding of the image, which is then used to generate subsequent annotations. The region for which the embedding has been computed determines the area SAMJ can work with to create annotations. If the user navigates to a region of the image that has not yet been processed, the embedding will need to be recalculated for that specific region before new annotations can be generated.

Having a clear strategy to define the portion of the image for which the embedding is created is therefore critical. In SAMJ, this is achieved by leveraging the interactive features provided by Fiji, such as the region of the image currently being viewed and user input. Depending on the user’s input, the method for determining the portion of the image to be embedded varies (see [Figure A.1](#)). This strategy aims to enhance the quality of annotations produced by SAMJ and enables annotations to be created for a wider range of object shapes and sizes.

Table A.2 Encoding time comparison for SAM-2, EfficientSAM, and original SAM models. The table shows model sizes (in MB) and the time required (in seconds) to perform encoding. Lightweight models such as SAM-2 Tiny and EfficientViTSAM-l2 significantly reduce both size and encoding time compared to the original SAM models. All results in this table were obtained on a workstation equipped with a 13th Gen Intel® Core™ i7-13700H processor and 32 GB of RAM. Although running the model on different hardware may produce varying absolute timings, the relative performance differences are expected to remain consistent.

	Model Size (MB)	Time to encode (seconds)	In SAMJ
SAM-2 Tiny	148.7	1.14	Yes
SAM-2 Small	175.8	1.39	Yes
SAM-2 Large	856.4	5.59	Yes
EfficientSAM	105.7	3.19	Yes
EfficientViTSAM-l2	245.7	0.74	Yes
SAM Base	375.0	15	No
SAM Large	1191.64	15	No
SAM Huge	2445.74	15	No

Consequently, there may be slight differences between annotations generated by SAMJ and those produced using the original SAM implementation. These differences reflect the flexibility of SAMJ’s embedding strategy in accommodating diverse annotation requirements.

A.4 Bounding Box Size vs. Image Size

SAM and its variants were primarily trained on natural images rescaled to a 1024×1024 pixel size. Under these training conditions, objects generally maintain a consistent size ratio relative to the image, which may bias the model towards favoring certain object-to-image size ratios at inference time.

As shown in Table A.3, images with sizes closer to 1024×1024 pixels are better understood by SAM. Since SAM only accepts images at this fixed size, the more the input image must be rescaled, the more its quality and interpretability may degrade.

Additionally, the table suggests that objects should not be extremely large or very small to be correctly annotated. Finding a balanced object-to-image size ratio appears crucial for achieving optimal performance.

A.5 Enabling Seamless Integration of SAM with Fiji: The Role of Appose and SAMJ

The integration of SAM into the Fiji ecosystem via SAMJ is facilitated by Appose⁴, a software framework designed to streamline inter-process communication. Within SAMJ, Appose orchestrates communication between the Java environment—where Fiji and SAMJ reside—and the Python environment, which runs SAM’s inference via PyTorch.

Appose employs process pipes to transmit variables between Java and Python and shared memory to exchange images and tensors efficiently, thus avoiding unnecessary data duplication. This architecture neatly segregates computational tasks, mitigating

⁴<https://github.com/apposed/appose>

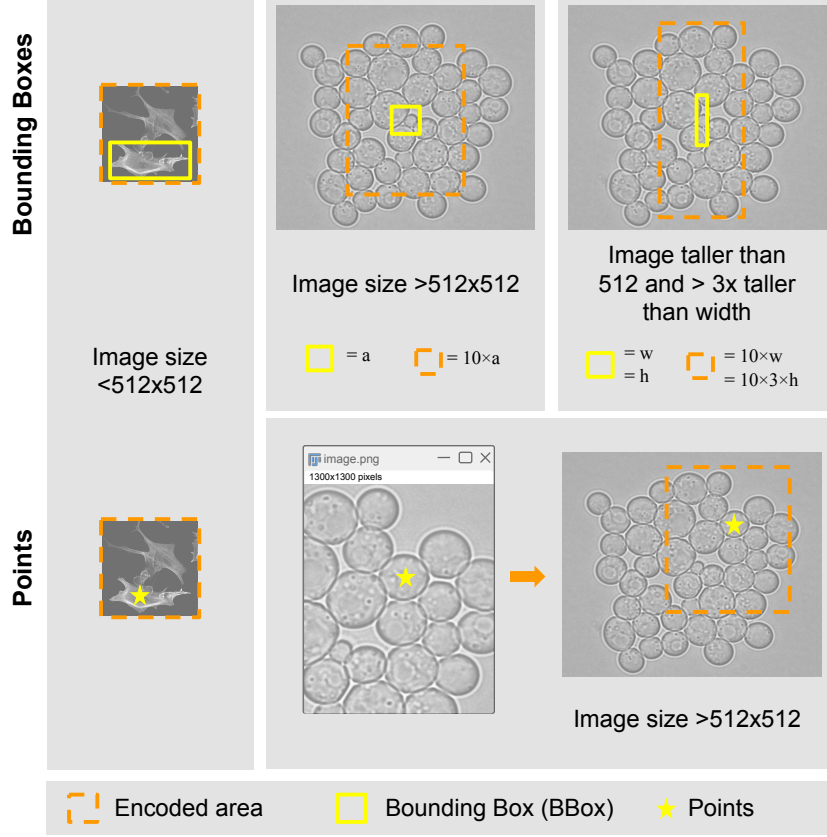


Figure A.1 Embedding strategies. SAMJ generates embeddings for regions of the image based on the user-defined input and the size of the image. For smaller images (smaller than 512×512 pixels), the entire image is embedded regardless of the prompt. For larger images (bigger than 512×512 pixels), different strategies are followed depending on the prompt and size of the bounding box. If the bounding box is relatively regular, the embedded area will be ten times the size of the bounding box. If one side of the bounding box is three times smaller than the other side of the bounding box, the embedded area will extend ten times the length of the smaller side and ten-thirds (approximately 3.33 times) the length of the larger side. For larger images and point prompts, the embedded area will include the visible region in the Fiji window plus an additional margin.

the risk of memory conflicts and enabling multiple models to be run concurrently without interference. Once a model is deselected, the associated processes are automatically terminated, ensuring a clean and self-contained execution environment.

In addition to managing communication, Appose automatically handles the installation and setup of required software environments. Using Micromamba, it can create and maintain isolated environments with all necessary dependencies without requiring user intervention. This fully automated process ensures a seamless user experience, simplifying the deployment and use of SAM within Fiji.

	500x500	1000x1000	2000x2000	3000x3000	4000x4000
Object: 26x26	0.808	0.828	0.828	0.828	0.828
Object: 52x52	0.927	0.930	0.930	0.930	0.930
Object: 78x78	0.956	0.944	0.933	0.933	0.933
Object: 104x104	0.953	0.959	0.929	0.929	0.929
Object: 130x130	0.968	0.958	0.932	0.920	0.920
Object: 156x156	0.967	0.969	0.927	0.903	0.903
Object: 182x182	0.969	0.974	0.936	0.912	0.873
Object: 208x208	0.969	0.977	0.929	0.892	0.871
Object: 234x234	0.975	0.970	0.957	0.921	0.868
Object: 260x260	0.611	0.976	0.946	0.898	0.879
Object: 286x286	0.612	0.978	0.940	0.911	0.898
Object: 312x312	0.542	0.972	0.954	0.907	0.888
Object: 338x338	0.429	0.976	0.951	0.922	0.910
Object: 364x364	0.566	0.969	0.965	0.979	0.899
Object: 390x390	0.974	0.979	0.976	0.919	0.913
Object: 416x416	0.000	0.976	0.974	0.903	0.891
Object: 442x442	0.459	0.983	0.976	0.931	0.907

Table A.3 Impact of object size and image resolution on SAM detection scores. The table shows detection scores for objects of varying sizes across different image resolutions (from 500×500 to 4000×4000 pixels). Bold values indicate the maximum score in a row or scores ≥ 0.95 . Results demonstrate that SAM performs optimally when objects maintain a balanced size relative to the image resolution, with sizes closer to 1024×1024 yielding higher and more consistent scores.

Appose creates a robust foundation for SAMJ to function as a powerful Java-to-Python bridge. This allows users to integrate SAM’s advanced segmentation capabilities, enabling the annotation of images in the Fiji ecosystem, benefiting from all tools, plugins, and capabilities of Java. Appose and SAMJ mark a significant check-point in the integration of novel, cutting-edge technologies developed in Python into the user-friendly world of Java.

A.6 Extending SAMJ for Large-Scale and Multi-dimensional Image Annotation with BigDataViewer

Building on the SAMJ plugin, we developed an extension for BigDataViewer, a core Fiji tool for visualizing large and multi-dimensional images [15]. This extension enables users to annotate these types of images by drawing rectangular prompts directly on the displayed image, which are immediately processed by the selected SAM model from the SAMJ Annotation control panel. The resulting polygons are rendered over the image and stored with their spatio-temporal coordinates. Polygons are displayed only when the user navigates back to the corresponding view angle and time point, but to simplify navigation, the extension memorizes view configurations, allowing users to cycle through and revisit them seamlessly. New view configurations trigger automatic image embedding, and additional features such as an undo/redo mechanism and mask export to Fiji further enhance usability.

The software is a lightweight library designed to integrate with an existing BigDataViewer instance, enabling users to add prompts directly. It allows client programs to define custom routines triggered when prompts are entered or new polygons are created. Our implementation facilitates communication between BigDataViewer and SAM models using this mechanism, while also enabling client programs to override default behaviors—for instance, converting prompts into polygons or customizing how polygons are rendered. This flexibility has been applied in Mastodon (a large-scale tracking framework for large, multi-view images), where polygons are replaced with

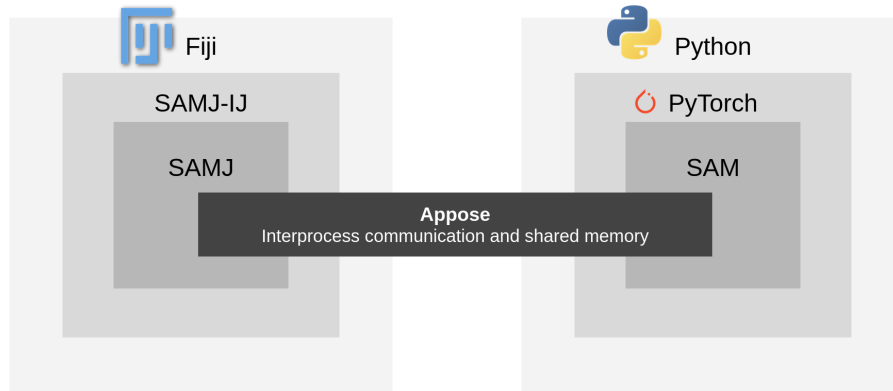


Figure A.2 Overview of the integration between SAMJ and SAM via Appose. On the left, the Fiji/ImageJ environment hosts the SAMJ plugin, which acts as the interface for users within Java-based image processing tools. On the right, the Python with Pytorch runs the SAM model for segmentation tasks. Appose bridges these two environments, enabling seamless inter-process communication between Java and Python. It facilitates variable exchange via pipes and shared memory, ensuring efficient handling of images/tensors while avoiding memory duplication. This architecture encapsulates processes, preventing conflicts, and allows for the concurrent execution of multiple models.

spots for tracking, and in Labkit, where SAM-generated polygons are used for pixel classification labels.

This new library for BigDataViewer is central to enabling efficient annotation of large and multi-dimensional images. By connecting SAMJ with BigDataViewer, users can seamlessly annotate much larger datasets, including those in 3D, significantly expanding the scope of bioimage analysis.

Appendix B Supplementary Materials

B.1 Links to Videos

- SAMJ: Nuclei Segmentation using Fiji’s capabilities and BatchSAMize: <https://www.youtube.com/watch?v=vpOzqLyxzk>
- SAMJ: Tumor Area Quantification and Nuclei Analysis with SAMJ and StarDist plugins for Fiji: <https://www.youtube.com/watch?v=4JhSEtDxY9g>
- SAMJ: Accelerating Annotation of Bacterial Motility: <https://www.youtube.com/watch?v=10PTr5DkgBc>
- SAMJ: Integration with Labkit: <https://www.youtube.com/watch?v=2l0cyjNm80o>

References

- [1] Pelt, D.M.: Tackling the challenges of bioimage analysis. *Elife* **9**, 64384 (2020)
- [2] Zulueta-Coarasa, T., Jug, F., Mathur, A., Moore, J., Muñoz-Barrutia, A., Anita, L., Babalola, K., Bankhead, P., Gilloteaux, P., Gogoberidze, N., et al.: Mifa: Metadata, incentives, formats, and accessibility guidelines to improve the reuse of ai datasets for bioimage analysis. *arXiv preprint arXiv:2311.10443* (2023)
- [3] Schindelin, J., Arganda-Carreras, I., Frise, E., Kaynig, V., Longair, M., Pietzsch, T., Preibisch, S., Rueden, C., Saalfeld, S., Schmid, B., et al.: Fiji: an open-source platform for biological-image analysis. *Nature methods* **9**(7), 676–682 (2012)
- [4] Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., Xiao, T., Whitehead, S., Berg, A.C., Lo, W.-Y., et al.: Segment anything. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4015–4026 (2023)
- [5] Sugawara, K.: Training deep learning models for cell image segmentation with sparse annotations. *BioRxiv*, 2023–06 (2023)
- [6] Archit, A., Freckmann, L., Nair, S., Khalid, N., Hilt, P., Rajashekar, V., Freitag, M., Teuber, C., Buckley, G., Haaren, S., et al.: Segment anything for microscopy. *Nature Methods*, 1–13 (2025)
- [7] Schindelin, J., Rueden, C.T., Hiner, M.C., Eliceiri, K.W.: The imagej ecosystem: An open platform for biomedical image analysis. *Molecular reproduction and development* **82**(7-8), 518–529 (2015)
- [8] Hollandi, R., Diósi, Á., Hollandi, G., Moshkov, N., Horváth, P.: Annotatorj: an imagej plugin to ease hand annotation of cellular compartments. *Molecular biology of the cell* **31**(20), 2179–2186 (2020)
- [9] Arzt, M., Deschamps, J., Schmied, C., Pietzsch, T., Schmidt, D., Tomancak, P., Haase, R., Jug, F.: Labkit: labeling and segmentation toolkit for big image data. *Frontiers in computer science* **4**, 777728 (2022)
- [10] Thomas, L.S., Schaefer, F., Gehrig, J.: Fiji plugins for qualitative image annotations: routine analysis and application to image classification. *F1000Research* **9**, 1248 (2021)
- [11] Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., Houlsby, N.: An image is worth 16x16 words: Transformers for image recognition at scale. In: *International Conference on Learning Representations* (2021). <https://openreview.net/forum?id=YicbFdNTTy>

- [12] Ravi, N., Gabeur, V., Hu, Y.-T., Hu, R., Ryali, C., Ma, T., Khedr, H., Rädle, R., Rolland, C., Gustafson, L., et al.: Sam 2: Segment anything in images and videos. arXiv preprint arXiv:2408.00714 (2024)
- [13] Xiong, Y., Varadarajan, B., Wu, L., Xiang, X., Xiao, F., Zhu, C., Dai, X., Wang, D., Sun, F., Iandola, F., et al.: EfficientSAM: Leveraged masked image pretraining for efficient segment anything. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 16111–16121 (2024)
- [14] Zhang, Z., Cai, H., Han, S.: Efficientvit-sam: Accelerated segment anything model without performance loss. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 7859–7863 (2024)
- [15] Pietzsch, T., Saalfeld, S., Preibisch, S., Tomancak, P.: Bigdataviewer: visualization and processing for large image data sets. *Nature methods* **12**(6), 481–483 (2015)
- [16] Stringer, C., Wang, T., Michaelos, M., Pachitariu, M.: Cellpose: a generalist algorithm for cellular segmentation. *Nature methods* **18**(1), 100–106 (2021)
- [17] Shamaï, G., Livne, A., Polónia, A., Sabo, E., Cretu, A., Bar-Sela, G., Kimmel, R.: Deep learning-based image analysis predicts pd-l1 status from h&e-stained histopathology images in breast cancer. *Nature Communications* **13**(1), 6753 (2022)
- [18] Casser, V., Kang, K., Pfister, H., Haehn, D.: Fast mitochondria detection for connectomics. In: Medical Imaging with Deep Learning, pp. 111–120 (2020). PMLR
- [19] Schmidt, U., Weigert, M., Broaddus, C., Myers, G.: Cell detection with star-convex polygons. In: Medical Image Computing and Computer Assisted intervention–MICCAI 2018: 21st International Conference, Granada, Spain, September 16–20, 2018, Proceedings, Part II 11, pp. 265–273 (2018). Springer
- [20] Gómez-de-Mariscal, E., García-López-de-Haro, C., Ouyang, W., Donati, L., Lundberg, E., Unser, M., Muñoz-Barrutia, A., Sage, D.: Deepimagej: A user-friendly environment to run deep learning models in imagej. *Nature methods* **18**(10), 1192–1195 (2021)
- [21] Casado-Garcia, A., Chichón, G., Dominguez, C., Garcia-Dominguez, M., Heras, J., Ines, A., Lopez, M., Mata, E., Pascual, V., Saenz, Y.: Motilityj: An open-source tool for the classification and segmentation of bacteria on motility images. *Computers in biology and medicine* **136**, 104673 (2021)