JOURNAL OF APPLIED CRYSTALLOGRAPHY

ISSN 1600-5767

Received 25 July 2023
Accepted 19 September 2023

Edited by F. Meilleur, Oak Ridge National Laboratory, USA, and North Carolina State University, USA

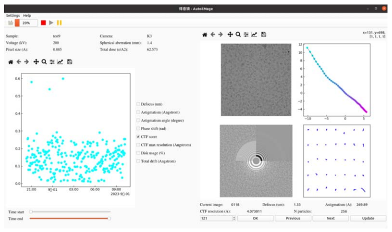
Keywords: cryo-EM; single-particle analysis; automation; real time.

AutoEMage: automatic data transfer, preprocessing, real-time display and monitoring in cryo-EM

Yuanhao Cheng,[a,b] Xiaojun Huang,[c] Bin Xu[d] and Wei Ding[a,b]*

[a]Laboratory of Soft Matter Physics, Institute of Physics, Chinese Academy of Sciences, Beijing 100190, People's Republic of China, [b]University of Chinese Academy of Sciences, Beijing 100049, People's Republic of China, [c]Center for Biological Imaging, Core Facilities for Protein Science, Institute of Biophysics, Chinese Academy of Sciences, Beijing 100101, People's Republic of China, and [d]Peking University Institute of Advanced Agricultural Sciences, Shandong Laboratory of Advanced Agricultural Sciences in Weifang, Weifang 261325, Shandong, People's Republic of China. *Correspondence e-mail: dingwei@iphy.ac.cn

Cryo-electron microscopy (cryo-EM), especially single-particle analysis, has become a powerful technique for visualizing the structure of biological macromolecules at high resolution. However, data acquisition in cryo-EM is time consuming because it requires the collection of thousands of images to achieve a high-quality reconstruction. Real-time preprocessing and display of the images can greatly enhance the efficiency and quality of data collection. This study presents *AutoEMage*, a new open-source software package that automates data transfer, preprocessing and real-time display in cryo-EM experiments. *AutoEMage* also includes a real-time data monitoring system that alerts users to issues with their data, allowing them to take corrective actions accordingly. The software is equipped with an easy-to-use graphical user interface that provides seamless data screening and real-time feedback on data quality and microscope status.



1. Introduction

Originating from a new generation of detectors, called direct electron detectors, and advances in image processing algorithms (Kühlbrandt, 2014), cryo-electron microscopy (cryo-EM), especially single-particle analysis, has become a commonly used method to determine the 3D structure of macromolecular proteins at near-atomic resolution (Bai *et al.*, 2015). In the past decade, there has been an increase in the deposition of cryo-EM structures in the Protein Data Bank (PDB; https://www.rcsb.org/). As of August 2023, the EMDataResource comprises a collection of over 30 000 EM maps and nearly 17 000 PDB EM models. Notably, more than 5000 entries have been submitted reporting resolutions better than 4 Å, with a majority of these structures obtained through single-particle analysis. (These and additional statistics can be found at https://www.emdataresource.org/.)

Direct electron detectors, such as the Gatan K3, Thermo Fisher Scientific Falcon 4i and Direct Electron Apollo, have revolutionized cryo-EM by enabling single-electron counting and high-frame-rate movie recording (Yang *et al.*, 2021; Zhang *et al.*, 2023; Peng *et al.*, 2023). These detectors can capture thousands of movie stacks per day, automatically and without the need for human intervention (Caesar *et al.*, 2020). However, numerous factors can potentially generate irrelevant images during data collection. These include significant mechanical drifts of the sample stage, inaccurate autofocus by the microscope, nonuniform ice thickness distribution, damaged carbon film and high astigmatism. Moreover, vitrified samples that are susceptible to aggregation, preferential orientation or denaturation on specific grids may also result in invalid data. Therefore, the need for an efficient monitoring tool has become paramount. With such a tool, users can readily identify and discard unnecessary images during data collection, selectively adjust microscope settings to exclude suboptimal regions from acquisitions or re-evaluate their samples. By adopting this proactive approach, researchers can effectively prevent wasting time and resources on collecting irrelevant data.

To increase data throughput and processing efficiency, the cryo-EM community has developed a range of automatic software tools. These tools streamline the data acquisition and processing pipeline, enabling users to efficiently collect and analyze high-quality cryo-EM data. For instance, the *Smart EPU* software (Thermo Fisher Scientific Inc.), *SmartScope* (Bouvette *et al.*, 2022) and *Smart Leginon* (Cheng *et al.*, 2023) automate specimen screening and exclusion of problematic grids and holes during data collection, optimizing data acquisition settings accordingly. During data collection, software tools like *Scipion* (Rosa-Trevín *et al.*, 2016; Gómez-Blanco *et al.*, 2018; Sharov *et al.*, 2021), *cryoSPARC Live* (Punjani *et al.*, 2017), *Warp* (Tegunov & Cramer, 2019), *RELION-4.0* (Kimanius *et al.*, 2021; Fernandez-Leiro & Scheres, 2017), *TranSPHIRE* (Stabrin *et al.*, 2020) and *Focus* (Biyani *et al.*, 2017) offer real-time data processing capabilities. They are very powerful and versatile, as they either have their own preprocessing tools or integrate widely used tools. As a result, they provide users with sophisticated functions and pipelines to effectively handle various types of samples, requiring minimal intervention. These software tools have significantly improved the efficiency and throughput of cryo-EM experiments by empowering users to make informed decisions, optimize data acquisition settings and quickly assess data quality.

Here we introduce *AutoEMage*, user-friendly Linux software designed for cryo-EM data acquisition that offers automated data transfer, preprocessing and real-time display capabilities. *AutoEMage* is built on Python3 and seamlessly integrates popular software tools like *MotionCor2* (Zheng *et al.*, 2017), *CTFFIND4* (Rohou & Grigorieff, 2015), *RELION-4.0* (Kimanius *et al.*, 2021) and *IMOD* (Mastronarde & Held, 2017). These tools enable tasks such as motion correction, contrast transfer function (CTF) estimation, particle picking, 2D classification, class ranking and 3D model generation. The processed results are promptly visualized through *AutoEMage*'s graphical user interface (GUI) and *UCSF ChimeraX* (Pettersen *et al.*, 2021), providing users with real-time data screening and feedback on sample quality and electron microscope status.

2. Implementation

2.1. Graphical user interface

Based on *PyQt6* (Riverbank Computing, 2023), the primary GUI comprises two panels displaying data and experiment information. Above the two panels, a toolbar allows users to input parameters and execute the automatic pipeline. Beside the toolbar, a progress bar is included to provide users with a clear indication of task progress. All tasks are performed by Python and Perl scripts, with which *AutoEMage* interacts through parameter files and log files.

2.2. Project management

To utilize the complete range of functions offered by *AutoEMage*, including its monitoring and reminder features, users must first log in to their account. Upon successful authentication, a directory is automatically generated with the username as its name, where all relevant files are stored. Whenever users initiate a new task of file transfer, *AutoEMage* starts a new project. For each project, *AutoEMage* creates a new directory within the user directory to manage all relevant files. Within this project directory, several sub-directories are also created to store images and metadata for other processing tasks such as motion correction, CTF estimation, particle picking, 2D classification, class ranking and 3D model generation. As *AutoEMage* utilizes some programs of *RELION*, *AutoEMage* organizes these sub-directories in the same way as *RELION* does.

2.3. Scripts

AutoEMage's processing functions are implemented in task-specific scripts. File transfer and motion correction are carried out by Perl scripts titled `Auto_mv`, while CTF estimation is performed by Perl scripts named `Auto_ctf`. Automatic particle picking, 2D classification, class ranking and 3D model generation are controlled by a Python script named `autoemage_threads`. These scripts call upon a few executables (such as *MotionCor2* and *CTFFIND4*) and transmit parameters through text files. To use the external software tools, they have to be installed on the system and their paths have to be added to environment variables. All scripts are readable and editable, giving users the freedom to modify any code as they see fit.

3. Data processing workflow

Upon the commencement of data acquisition, users are able to initiate a data transfer task in *AutoEMage*. As the data are transferred to the users' external disk drive, a series of automatic preprocessing tasks are executed sequentially, including motion correction, CTF estimation, particle picking, 2D classification, class ranking and 3D model generation, as illustrated in Fig. 1. Throughout the preprocessing stage, data are monitored and any outliers are promptly reported to users via emails.

3.1. Data transfer

To begin a data transfer task, users need to click on the 'file transfer' icon on the toolbar. They will then be prompted to input various parameters into the corresponding interface. These parameters include the detection mode, pixel size, total

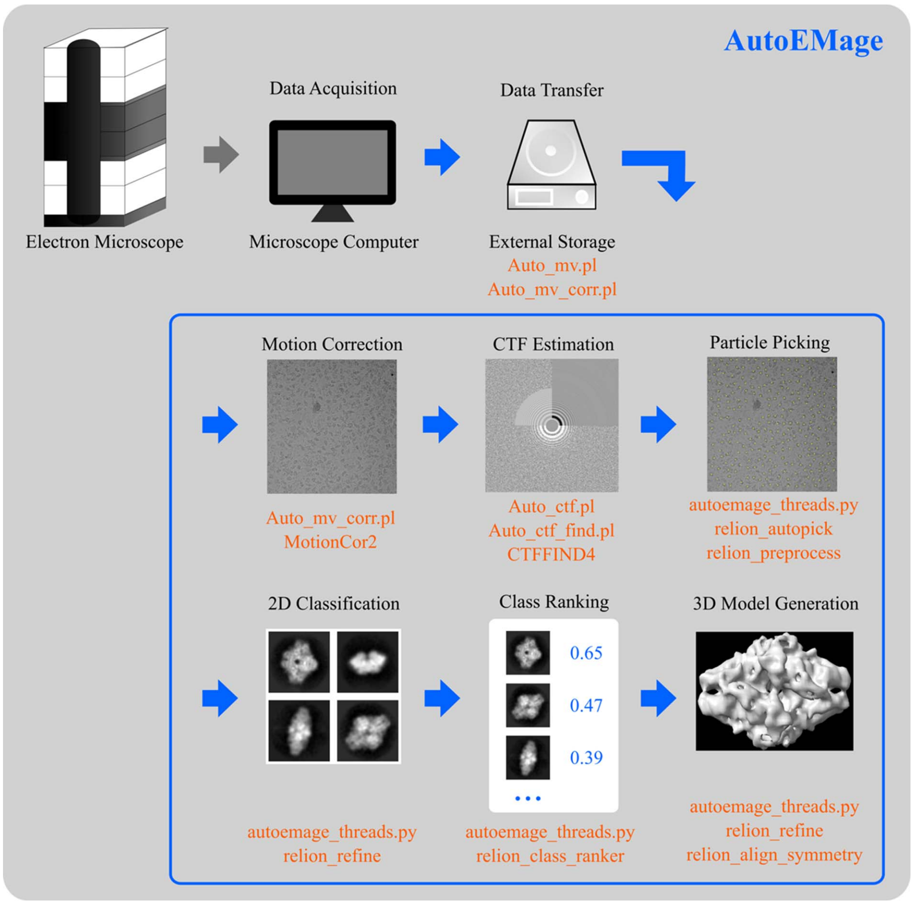


Figure 1
The data processing pipeline of *AutoEMage*. When data are continuously collected by automation software such as *EPU* (Thermo Fisher Scientific Inc.) or *SerialEM* (Mastronarde, 2005), *AutoEMage* begins its pipeline according to user specifications. The pipeline includes file management tasks, *i.e.* data transfer and file backup, and pre-processing tasks, *i.e.* motion correction, CTF estimation, particle picking, 2D classification, class ranking and 3D model generation.

dose, particle diameter, image directory, disk directory, project name and so on. These parameters can be saved to a text file so that they can be loaded for future experimental sessions that require similar settings. During data acquisition, the electron microscope automatically saves thousands of images on the computer. *AutoEMage* can detect these files by their formatted names. For example, movie stacks stored by *EPU* are named `FoilHole_...Data..._fractions`. Then, *AutoEMage* automatically transfers these files to an external disk drive (a mobile hard disk drive or a disk array). At the destination, the files are renamed in chronological order, which determines the order in which they are processed and displayed by *AutoEMage*. The transfer speed for a single image is less than 2 s, which matches the pace of data acquisition. This eliminates any delays in data transfer after data acquisition. *AutoEMage* now supports data collected by *EPU/SerialEM* software.

3.2. Preprocessing

During the course of a typical half-day data collection process, the availability of real-time feedback regarding sample quality and system status is advantageous. However, the raw movie stacks obtained through EM are often dose-fractionated and visually uninformative. In order to overcome this limitation, *AutoEMage* has been developed to automatically carry out a series of preprocessing steps. These steps include motion correction, CTF estimation, particle picking, 2D classification, class ranking and 3D model generation. By performing these tasks, *AutoEMage* facilitates the identification and assessment of Thon rings and particles in each image, as well as the qualitative evaluation of 2D classes and 3D models across a batch of images. Additionally, the preprocessing tasks yield valuable physical variables such as defocus and astigmatism, which are graphed to provide insights into the system status. *AutoEMage* is also capable of processing three common image formats: TIFF, MRC and EER.

3.2.1. Motion correction. First, correcting for mechanical movement of the sample stage and beam-induced motion is crucial in order to obtain high-quality micrographs with a high signal-to-noise ratio. *AutoEMage* addresses this issue by utilizing *MotionCor2*, a software tool that performs global and local alignment of movie stacks. The relevant parameter settings for *MotionCor2*, such as the number of patches, the number of graphical processing units (GPU) and *B* factor, are pre-programmed in the Perl scripts `Auto_mv` and `Auto_mv_corr`. These scripts efficiently allocate GPU resources and automate the alignment process for each raw movie stack.

3.2.2. CTF estimation. Second, it is necessary to estimate the CTF in order to correct the contrast of the micrographs. This estimation process involves fitting a calculated CTF to the power spectrum of a registered image, allowing for the determination of lens defocus and astigmatism (Mindell & Grigorieff, 2003). These parameters are valuable for assessing the performance of the electron microscope, as discussed further below. In *AutoEMage*, the CTF for each aligned micrograph is calculated using the tool *CTFFIND4*. The parameter settings for *CTFFIND4* are encoded within the Perl script named `Auto_ctf_find`. This script automates the CTF estimation process, ensures consistent and accurate results for each micrograph, and prepares necessary text files for subsequent processing and display.

3.2.3. Particle picking. Third, the process of particle picking involves identifying and extracting particles of various orientations from the micrographs. This task typically requires a significant amount of time and expertise when performed manually, as a single micrograph can contain hundreds of particles. Fortunately, modern algorithms such as template matching (Tang *et al.*, 2007) and convolutional neural networks (Bepler *et al.*, 2019) have made particle picking fully automated, saving scientists from the laborious task of manual selection. In *AutoEMage*, particle picking and extraction are automated using the `relion_autopick` and `relion_preprocess` programs from *RELION-4.0*. Two strategies are available for users. Firstly, for computers with limited GPU memory, the Laplacian-of-Gaussian (LoG) filter (Zivanov *et al.*, 2018) can be used for initial autopicking, treating the resulting 2D classes as references. Subsequently, reference-based autopicking can be employed for more accurate particle picking in the remaining data. Alternatively, if the GPU memory is sufficient (larger than 4 GB), LoG autopicking is first performed to generate a set of 2D classes, which are then used to train a *Topaz* model (Bepler *et al.*, 2019). This trained model is utilized to automatically pick particles in the remaining data. In *AutoEMage*, users only need to input the minimum and maximum particle diameters to enable non-interactive particle picking. This streamlined process eliminates the need for manual intervention and simplifies the particle picking workflow.

3.2.4. 2D alignment and classification. Fourth, the particle images obtained in the previous step can be organized into distinct 2D classes based on their orientations. This grouping allows for the calculation of class averages, which improves the signal-to-noise ratio of the images and is beneficial for subsequent 3D reconstruction (Cheng *et al.*, 2015). In *AutoEMage*, the number of particles in each image is stored and presented in the GUI. When the cumulative number of particles exceeds a predefined threshold, such as 20 000, the particles from these cumulative images are automatically classified and averaged using the `relion_refine` program in *RELION-4.0*. This program incorporates two types of algorithms: the regularized likelihood optimization algorithm (Kimanius *et al.*, 2016) and variable-metric gradient descent with adaptive moments estimation (Kimanius *et al.*, 2021). In *AutoEMage*, regularized likelihood optimization is utilized for particles obtained through LoG autopicking, while variable-metric gradient descent is employed for particles obtained

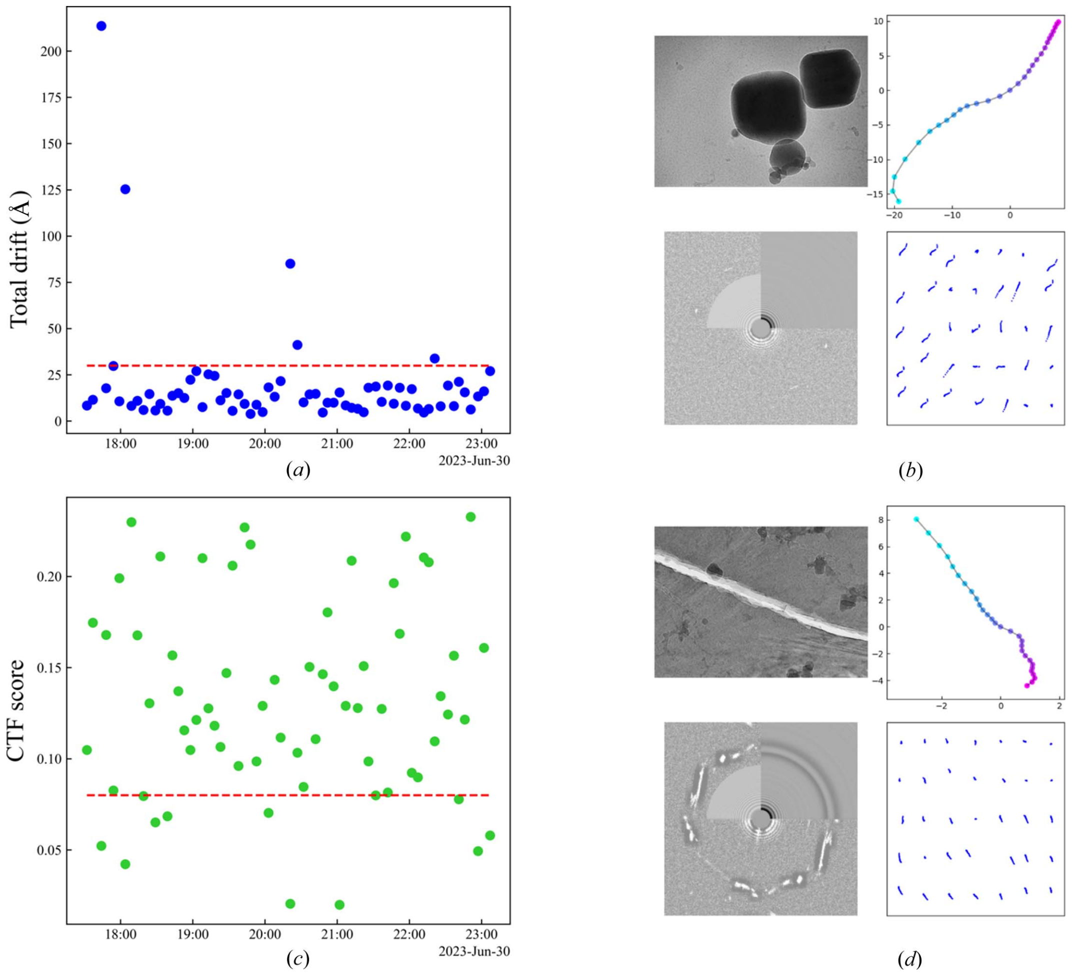


Figure 2
Two kinds of outliers and the respective data. (*a*) Total drift during data acquisition. Data above the red dashed line are outliers with large drift. (*b*) One of the outliers with large drift. Contamination takes up a large area of the image and total drift exceeds 30 Å. (*c*) CTF fitting score during data collection. Data below the red dashed line are outliers with low CTF scores. (*d*) An image with a low CTF fitting score. An ice crack appears across the image and the corresponding CTF score is below 0.08. Data are from our data set SPARTA.

through reference-based autopicking or *Topaz* neural network autopicking. By employing these algorithms, *AutoEMage* ensures the generation of high-quality class averages, setting the stage for accurate and reliable 3D reconstruction.

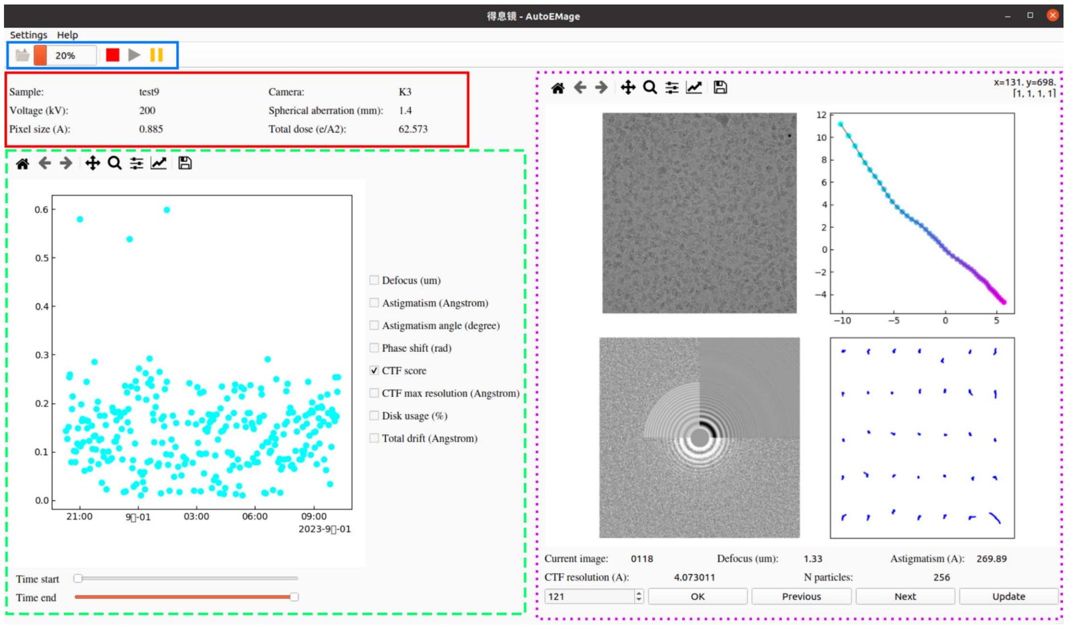
3.2.5. Class ranking and 3D model generation. Fifth, utilizing the generated 2D class averages, we can identify and select 'good' classes to create a 3D initial model. Traditionally, this process requires manual effort and expertise, as it is done in *cryoSPARC Live* (Punjani *et al.*, 2017). However, with the introduction of the `relion_class_ranker` program in *RELION-4.0*, this task is now automated using a convolutional neural network. The resulting 'good' classes are then saved in `.star` files, which can be easily utilized in *RELION* for subsequent processing steps, including 3D model generation and refinement. The generation of a 3D initial model is accomplished through the utilization of the `relion_refine` and `relion_align_symmetry` programs. This automated approach streamlines the selection of high-quality classes and facilitates the creation of a relatively accurate 3D initial model.

3.3. Data screening

During data collection, irrelevant images are simultaneously screened so that they will not be included in subsequent processing. *AutoEMage* screens three types of irrelevant images: images with significant drift, those with low CTF fitting scores and images whose CTF resolution is worse than 6 Å. The first type of bad data is detected by calculating the drift between consecutive frames and the overall drift of the movie stack. If the drift between two consecutive frames exceeds 5 Å or the total drift of the movie stack exceeds 30 Å [as shown in Figs. 2(*a*) and 2(*b*)], the movie stack is considered an outlier. Such outliers are unlikely to contribute significantly to 'good' classes or high-resolution 3D reconstruction. The second type of bad data is identified using the CTF fitting score, which reflects the visibility of Thon rings. If an image's power spectrum is heavily blurred and the Thon rings are barely visible, its CTF score is typically below 0.08 [as shown in Figs. 2(*c*) and 2(*d*)]. A low CTF score indicates that the image may not be suitable for high-resolution reconstruction. The third type of bad data is also filtered by the result of CTF estimation through setting a threshold of 6 Å for the CTF resolution. *AutoEMage* detects all types of outliers and moves them to a designated folder named 'Outliers' for further inspection by users. If users want to process the collected data by themselves or share with others, they can directly use the remaining 'good' data.

3.4. Demonstration

Upon the completion of each preprocessing step, *AutoEMage* provides immediate feedback to users by automatically displaying the corresponding results in the main GUI, different from *Scipion* (Rosa-Trevín *et al.*, 2016) and *RELION-4.0* (Kimanius *et al.*, 2021) that need user interaction. *AutoEMage*'s performance was tested on two data sets: TIR-APAZ proteins (SPARTA) from *Crenotalea thermophila* (Crt) and *β*-galactosidase particles from the EMPIAR-10204 set (Electron Microscopy Public Image Archive; Iudin *et al.*, 2016), as shown in Fig. 3. The main GUI of *AutoEMage* consists of two panels. The right panel primarily displays a two-by-two grid of images and diagrams. In the top-left corner,



Sample:	test9	Camera:	K3
Voltage (kV):	200	Spherical aberration (mm):	1.4
Pixel size (A):	0.885	Total dose (e/A2):	62.573

☐ Defocus (um)
☐ Astigmatism (Angstrom)
☐ Astigmatism angle (degree)
☐ Phase shift (rad)
☑ CTF score
☐ CTF max resolution (Angstrom)
☐ Disk usage (%)
☐ Total drift (Angstrom)

Current image:	0118	Defocus (um):	1.33	Astigmatism (A):	269.89
CTF resolution (A):	4.073011	N particles:	256		

121

Figure 3
The main GUI of *AutoEMage* during a 12 h data acquisition session. The display of images and diagrams is situated on the right panel enclosed by a purple dotted rectangle. The left panel features a graph plotting multiple variables changing over time, which is enclosed by a green dashed rectangle. The current experiment information is displayed above the graph and is encompassed by a red rectangle, while the toolbar and progress bar are shown overhead encompassed by a blue rectangle.

the aligned micrograph currently being processed is shown, allowing users to clearly visualize the target particles. The bottom-left panel displays the result of the CTF calculation, where Thon rings can be inspected to assess the quality of the micrograph. The top-right panel presents the trajectory of global frame movement in the current movie stack, providing insights into the overall motion of the sample. Finally, the bottom-right panel shows the trajectories of patch frame movement in the movie stack, offering a detailed view of the local motion patterns. These images and diagrams collectively provide users with an overview of the sample's quality and allow for quick assessment of the preprocessing results.

The left panel of the main GUI in *AutoEMage* provides information about the current experiment and includes a graph that plots various physical variables that change over time. These variables include lens defocus, astigmatism, astigmatism angle, maximum resolution of the CTF fitting and disk drive usage, among others. Below the graph, users can specify a time slot to view the data for each variable at any given time period. This feature allows users to track and analyze trends in the plotted variables. Some of these variables, such as lens defocus and astigmatism angle, can reflect the status of the electron microscope. Over time, as the sample stage in the electron microscope undergoes mechanical movements, system errors may gradually accumulate, resulting in observable shifts in these variables. For example, if the lens defocus continuously deviates from the mean value of previous data points, *AutoEMage* will automatically detect this trend and send a reminder email to users. This email serves as a prompt for users to check the settings of the electron microscope and take action, such as stopping data acquisition, if required. This proactive approach helps ensure that any potential issues with the microscope are identified in a timely manner, allowing users to maintain the quality and integrity of their data.

The average of multiple 2D classes is shown in Fig. 4, each providing a unique perspective on the 3D target particle. As data processing continues, users will observe different 2D class averages for every 20 000 particles (this number can be modified accordingly). The quality of the data can be assessed by the presence of a substantial number of well defined and detailed 2D class averages. Ideally, there should be more than 15 such classes. In contrast, images of poor quality tend to contain only a few 'good' classes, usually below ten. With access to these processed results 'on-the-fly', users can evaluate the quality of their samples with more depth, which enables them to make informed decisions and adjust their experimental settings accordingly. For instance, if the samples are of poor quality, users can choose to halt the current data collection rather than wait for an additional 10 h. This proactive approach allows users to optimize their data acquisition strategy and avoid wasting time and resources on low-quality samples.

Fig. 5 displays the 3D model obtained through *AutoEMage*'s automated processing pipeline, utilizing 20 000 particles. The figure includes the postprocessed map as well as the corresponding Fourier shell correlation (FSC) curve for β-galactosidase from the EMPIAR-10204 data set. Notably, the resolution at 0.154 FSC is determined to be 3.2 Å, which aligns with the value reported in the related Electron Microscopy Data Bank (Lawson *et al.*, 2016) entry EMD-6840 (T. Kato, N. Terahara & K. Namba).

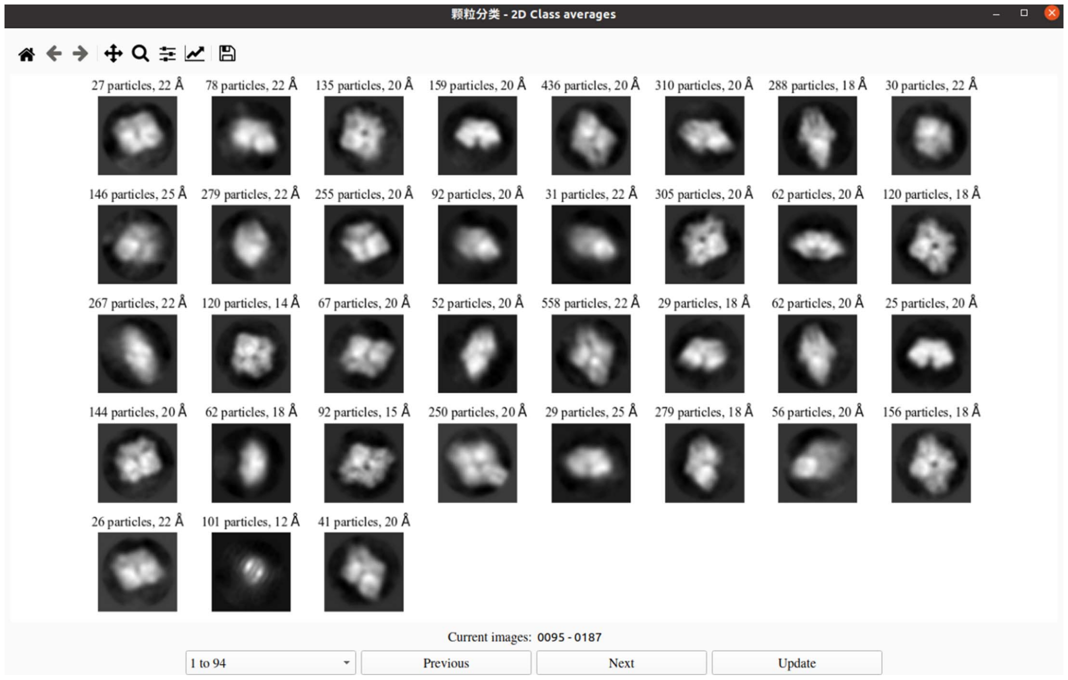


Figure 4
The GUI of 2D classes. For every 20 000 picked particles, 2D classification and class ranking is automatically performed and classes whose score is above a pre-defined threshold are displayed. Above each class average, the number of classified particles and the estimated resolution are also displayed. Data are from the EMPIAR-10204 set.

3.5. Monitoring and reminding

AutoEMage offers a solution to address the challenge of users having to remain near the electron microscope for extended periods during data collection. It incorporates an automated outlier detection system that identifies problematic data and sends email notifications to users, allowing them to come back and check the microscope settings accordingly. During a typical data acquisition session, a large number of images are recorded, and it is common to encounter some bad data. However, if a significant number of bad images occur with a certain frequency (*e.g.* ten outliers out of 50 consecutive movie stacks), they are considered outliers. The presence of these outliers may indicate potential errors with the microscope status or specimen quality. In such cases, *AutoEMage* automatically sends an email notification to the respective user as a reminder. *AutoEMage* can currently identify three types of outliers: images with significant drift, those with low CTF fitting scores and images with deviated defocus. The first two types of outliers are described in Section 3.3. The third type refers to images whose defocus values fall outside the range set by the microscope. In addition to outlier detection, *AutoEMage* also sends email notifications when data collection is complete or when the external disk drive reaches its storage capacity. These email notifications enable users to stay informed about the progress of their data collection and remotely monitor the process, eliminating the need for constant physical presence.

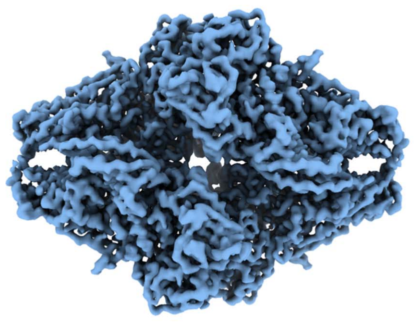
4. Software speed performance

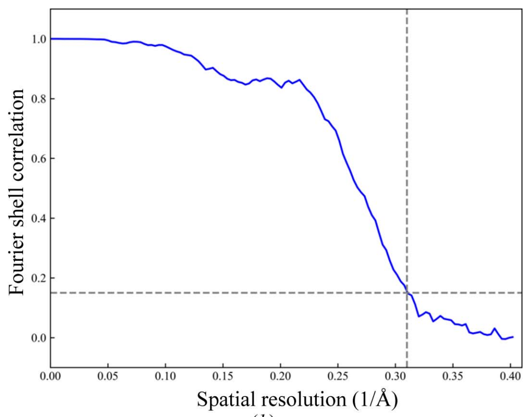
AutoEMage was installed on two different machines for testing and performance assessment. The first machine is an Ubuntu 20.04.6 computer with 24 cores and 48 threads in total. It has 32 GB of RAM and a 3 TB SSD for data storage. This machine also has an NVIDIA Quadro P620 GPU card installed, which has 2 GB of video memory. The second machine is a Rocky Linux 8.7 server located at the Cryo-EM Center, School of Advanced Agricultural Sciences, Peking University. It is equipped with 32 cores and 64 threads in total, along with 128 GB of RAM and a 20 TB SSD for data storage. The server also has an NVIDIA GeForce RTX 4090 GPU card installed, which has 24 GB of video memory.

For the provided hardware configurations, the processing times for different steps of the *AutoEMage* workflow are as follows. File transfer, CTF estimation, LoG autopicking and class ranking each take around 3 s for a single 4k $\times$ 4k raw movie stack with 50 frames. Motion correction, which is done sequentially because of the single GPU, takes around 3 min for a raw movie stack on the Ubuntu computer. However, on the server with the better GPU, it takes less than 8 s for the same task. 2D classification takes less than 30 min for around 20 000 particles from more than 90 images, which barely catches up with the speed of data acquisition. 3D model generation takes around 30 min for 'good' classes selected from more than 14 000 particles, but the same task takes around 5 min on the server. On the basis of these processing times, it can be observed that CTF estimation, autopicking, 2D classification and class ranking are all ahead of the speed of data acquisition. However, motion correction and 3D model generation are slower and fall slightly behind the speed of data collection on the Ubuntu computer. On the server with the better GPU, the processing speed of motion correction and 3D model generation can catch up with the speed of data acquisition. Additionally, computers equipped with more GPUs can further accelerate the processing speed by parallelization and match the speed of data collection. The processing times for each step of preprocessing with multiple data sets are shown in Table 1.

5. Discussion

In this paper, we introduce *AutoEMage*, open-source user-friendly software specifically developed for real-time data transfer, preprocessing and evaluation during the cryo-EM data acquisition process. The main objective of *AutoEMage* is to enhance data collection strategies by enabling users to monitor image quality and the status of the electron microscope. Additionally, *AutoEMage* includes a scientific data screening method, which offers valuable insights for subsequent data processing.


(*a*)



(*b*)

Figure 5
(*a*) The postprocessed map and (*b*) the corresponding FSC curve of β-galactosidase from the EMPIAR-10204 set. The resolution is around 3.2 Å.

Table 1
Processing time for each step of preprocessing with multiple data sets.

	Detector/image size (MB)	Symmetry	Pixel size (Å)/box size (pixel)	Data transfer for a raw movie stack (s)	Motion correction for a raw movie stack† (s)	CTF estimation for one micrograph (s)	Particle picking and extraction for one micrograph† (s)	2D classification for ∼20 000 particles† (min)	3D model generation for ∼14 000 particles† (min)
EMD-22025 (β-galactosidase)‡	Gatan K2/260	*C*2	0.885/256	<1	∼180/8	<2	<4/1	∼28/3	∼27/5
EMD-35787 (LHCII-nanodisc)§	Gatan K3/380	*C*1	1.04/200	<1	∼240/10	<3	<5/1	∼30/3	∼30/5
Ferritin (Unpublished)	Falcon 4i/256	*O*	0.96/180	<1	∼180/8	<2	<4/1	∼28/3	∼20/5

† The two sets of values are for two types of GPU: NVIDIA Quadro P620/NVIDIA GeForce RTX 4090. ‡ McSweeney *et al.* (2020). § Ruan *et al.* (2023).

AutoEMage is an open-source program, meaning that its source code is freely available for users to access, modify and contribute to its development. This allows academic or authorized users to customize and extend the functionality of *AutoEMage* according to their specific requirements. The software is designed to be easy to install and use, and interested users can download *AutoEMage* from the GitHub repository at https://github.com/FanLabIOP/AutoEMage.

In the future, we plan to enhance *AutoEMage* with additional functionality to further improve its utility and convenience for users. We aim to develop an image segmentation tool within *AutoEMage*. This tool will focus on extracting particles from images that were previously considered unusable due to defects like cracks or contamination. By extracting valuable particle information from these images, the data throughput of the imaging process can be improved. We also plan to explore the development of web applications that allow users to remotely control *AutoEMage*. This eliminates the need for users to download the software or be physically present in front of their computers to use it. Moreover, instead of relying solely on emails, we will investigate the use of mobile messages or social media for more timely reminders. This approach ensures that users receive important notifications promptly, reducing potential delays. Furthermore, *AutoEMage* aims to provide real-time feedback to users during data acquisition. This includes the possibility of automatic parameter adjustment based on previously processed results. As sample characteristics may vary, default parameter settings may not be optimal for all circumstances. By analyzing preliminary results and allowing users to adjust parameters on this basis, *AutoEMage* can help users obtain better results and potentially achieve higher-resolution 3D structures. These proposed additions will further enhance the functionality, usability and efficiency of *AutoEMage*, making it a more powerful and user-friendly tool for cryo-EM data acquisition and analysis.

Acknowledgements

We express our gratitude to the Cryo-EM Center at the School of Advanced Agricultural Sciences, Peking University, for their invaluable support in conducting tests and utilizing our software. We thank Linzhu Bi and Zengru Li for critically evaluating the manuscript. 3D display in *AutoEMage* is performed with *UCSF ChimeraX*, developed by the Resource for Biocomputing, Visualization and Informatics at the University of California, San Francisco, with support from the National Institutes of Health R01-GM129325 and the Office of Cyber Infrastructure and Computational Biology, National Institute of Allergy and Infectious Diseases.

References

Bai, X.-C., McMullan, G. & Scheres, S. H. W. (2015). *Trends Biochem. Sci.* **40**, 49–57.
Bepler, T., Morin, A., Rapp, M., Brasch, J., Shapiro, L., Noble, A. J. & Berger, B. (2019). *Nat. Methods*, **16**, 1153–1160.
Biyani, N., Righetto, R. D., McLeod, R., Caujolle-Bert, D., Castano-Diez, D., Goldie, K. N. & Stahlberg, H. (2017). *J. Struct. Biol.* **198**, 124–133.
Bouvette, J., Huang, Q., Riccio, A. A., Copeland, W. C., Bartesaghi, A. & Borgnia, M. J. (2022). *eLife*, **11**, e80047.
Caesar, J., Reboul, C. F., Machello, C., Kiesewetter, S., Tang, M. L., Deme, J. C., Johnson, S., Elmlund, D., Lea, S. M. & Elmlund, H. (2020). *J. Struct. Biol. X*, **4**, 100040.
Cheng, A., Kim, P. T., Kuang, H., Mendez, J. H., Chua, E. Y. D., Maruthi, K., Wei, H., Sawh, A., Aragon, M. F., Serbynovskyi, V., Neselu, K., Eng, E. T., Potter, C. S., Carragher, B., Bepler, T. & Noble, A. J. (2023). *IUCrJ*, **10**, 77–89.
Cheng, Y., Grigorieff, N., Penczek, P. & Walz, T. (2015). *Cell*, **161**, 438–449.
Fernandez-Leiro, R. & Scheres, S. H. W. (2017). *Acta Cryst.* D**73**, 496–502.
Gómez-Blanco, J., de la Rosa-Trevín, J. M., Marabini, R., del Cano, L., Jiménez, A., Martínez, M., Melero, R., Majtner, T., Maluenda, D., Mota, J., Rancel, Y., Ramírez-Aportela, E., Vilas, J. L., Carroni, M., Fleischmann, S., Lindahl, E., Ashton, A. W., Basham, M., Clare, D. K., Savage, K., Siebert, C. A., Sharov, G. G., Sorzano, C. O. S., Conesa, P. & Carazo, J. M. (2018). *J. Struct. Biol.* **204**, 457–463.
Iudin, A., Korir, P. K., Salavert-Torres, J., Kleywegt, G. J. & Patwardhan, A. (2016). *Nat. Methods*, **13**, 387–388.
Kimanius, D., Dong, L., Sharov, G., Nakane, T. & Scheres, S. H. W. (2021). *Biochem. J.* **478**, 4169–4185.
Kimanius, D., Forsberg, B. O., Scheres, S. H. W. & Lindahl, E. (2016). *eLife*, **5**, e18722.
Kühlbrandt, W. (2014). *Science*, **343**, 1443–1444.
Lawson, C. L., Patwardhan, A., Baker, M. L., Hryc, C., Garcia, E. S., Hudson, B. P., Lagerstedt, I., Ludtke, S. J., Pintilie, G., Sala, R., Westbrook, J. D., Berman, H. M., Kleywegt, G. J. & Chiu, W. (2016). *Nucleic Acids Res.* **44**, D396–D403.
Mastronarde, D. N. (2005). *J. Struct. Biol.* **152**, 36–51.
Mastronarde, D. N. & Held, S. R. (2017). *J. Struct. Biol.* **197**, 102–113.

McSweeney, D. M., McSweeney, S. M. & Liu, Q. (2020). *IUCrJ*, **7**, 719–727.

Mindell, J. A. & Grigorieff, N. (2003). *J. Struct. Biol.* **142**, 334–347.

Peng, R., Fu, X., Mendez, J. H., Randolph, P. S., Bammes, B. E. & Stagg, S. M. (2023). *J. Struct. Biol. X*, **7**, 100080.

Pettersen, E. F., Goddard, T. D., Huang, C. C., Meng, E. C., Couch, G. S., Croll, T. I., Morris, J. H. & Ferrin, T. E. (2021). *Protein Sci.* **30**, 70–82.

Punjani, A., Rubinstein, J. L., Fleet, D. J. & Brubaker, M. A. (2017). *Nat. Methods*, **14**, 290–296.

Riverbank Computing (2023). *PyQt*, https://www.riverbankcomputing.com/software/pyqt/.

Rohou, A. & Grigorieff, N. (2015). *J. Struct. Biol.* **192**, 216–221.

Rosa-Trevín, J. M. de la, Quintana, A., del Cano, L., Zaldívar, A., Foche, I., Gutiérrez, J., Gómez-Blanco, J., Burguet-Castell, J., Cuenca-Alba, J., Abrishami, V., Vargas, J., Otón, J., Sharov, G., Vilas, J. L., Navas, J., Conesa, P., Kazemi, M., Marabini, R., Sorzano, C. O. S. & Carazo, J. M. (2016). *J. Struct. Biol.* **195**, 93–99.

Ruan, M., Li, H., Zhang, Y., Zhao, R., Zhang, J., Wang, Y., Gao, J., Wang, Z., Wang, Y., Sun, D., Ding, W. & Weng, Y. (2023). *Nat. Plants*, **9**, 1547–1557.

Sharov, G., Morado, D. R., Carroni, M. & de la Rosa-Trevín, J. M. (2021). *Acta Cryst.* D**77**, 403–410.

Stabrin, M., Schoenfeld, F., Wagner, T., Pospich, S., Gatsogiannis, C. & Raunser, S. (2020). *Nat. Commun.* **11**, 5716.

Tang, G., Peng, L., Baldwin, P. R., Mann, D. S., Jiang, W., Rees, I. & Ludtke, S. J. (2007). *J. Struct. Biol.* **157**, 38–46.

Tegunov, D. & Cramer, P. (2019). *Nat. Methods*, **16**, 1146–1152.

Yang, Y., Arseni, D., Zhang, W., Huang, M., Lövestam, S., Schweighauser, M., Kotecha, A., Murzin, A. G., Peak-Chew, S. Y., Macdonald, J., Lavenir, I., Garringer, H. J., Gelpi, E., Newell, K. L., Kovacs, G. G., Vidal, R., Ghetti, B., Falcon, B., Scheres, S. H. W. & Goedert, M. (2021). *bioRxiv*, https://doi.org/10.1101/2021.10.19.464936.

Zhang, B., Gorman, J., Kwon, Y. D., Pegu, A., Chao, C. W., Liu, T., Asokan, M., Bender, M. F., Bylund, T., Damron, L., Gollapudi, D., Lei, P., Li, Y., Liu, C., Louder, M. K., McKee, K., Olia, A. S., Rawi, R., Schön, A., Wang, S., Yang, E. S., Yang, Y., Carlton, K., Doria-Rose, N. A., Shapiro, L., Seaman, M. S., Mascola, J. R. & Kwong, P. D. (2023). *mAbs*, **15**, 2165390.

Zheng, S. Q., Palovcak, E., Armache, J. P., Verba, K. A., Cheng, Y. & Agard, D. A. (2017). *Nat. Methods*, **14**, 331–332.

Zivanov, J., Nakane, T., Forsberg, B. O., Kimanius, D., Hagen, W., Lindahl, E. & Scheres, S. H. (2018). *eLife*, **7**, e42166.