

I. INTRODUCTION

The scanning electron microscope (SEM) is a type of microscope that produces images using signals generated from the interaction between electrons and the surface under observation. Higher resolution can be achieved compared to the traditional optical microscope, since electrons have much lower wavelength than light. An SEM can have resolution lower than one nanometre, whereas that of an optical microscope is often limited to a few hundred nanometres. This has benefited a variety of fields. For example, scientists have been using the SEM to analyse the doping density in semiconductors [1] and to view changes in bacterial cells [2].

Fig. 1 shows the basic construction of an SEM. The electron gun generates an electron beam, which is transformed into an electron probe after passing through the condenser lens and objective lens. It is scanned across the specimen under the effect of the scanning coil. As a result of the interaction between the incident electrons and the specimen, some electrons are emitted from the specimen. These are called secondary electrons and are collected by the detector, which generates signals whose magnitude depend on the strength of the secondary electrons. The display unit produces one frame

of image after each complete scan of the specimen.

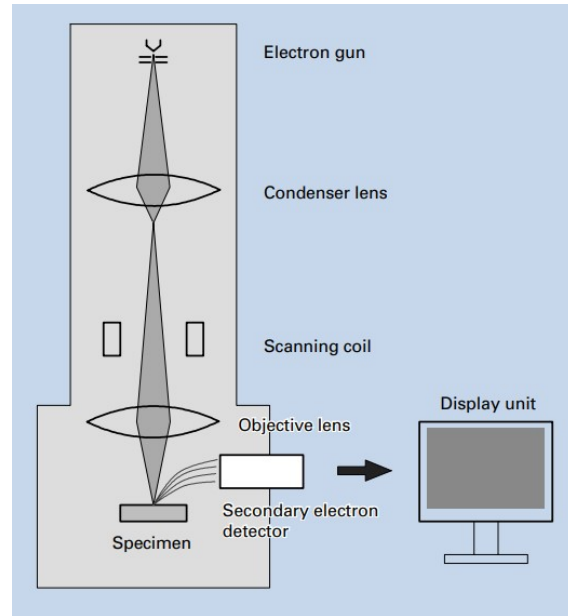


Fig. 1: Basic construction of an SEM [3].

Digital image analysis methods have been widely used in the field of SEM. For example, the fibre orientation distribution of non-woven fabrics can be determined using fast Fourier transform (FFT) and Hough transform (HT) [4]. The FFT is an especially popular algorithm and many papers have been published on the use of it. However, due to its complexity and a lack of fast hardware, real-time analysis had largely been impossible or impractical in the past, i.e. it was only feasible to perform FFT on images off-line. In 1997, with an advanced central processing unit (CPU) — the Pentium Pro, it was only possible to achieve a refresh rate of about 0.6 frames per second for an 8-bit $1024 \times$

1024 image [5]. The enhancement of CPUs has enabled faster frame rates and more recently, the development of graphics processing unit (GPU) has taken image analysis to a different level.

The GPU was originally designed to efficiently manipulate data in order to accelerate the rendering of 3D images on displays. Depending on the position, pixels in a 3D image may require different processing to achieve effects such as lighting, blurring and fogging. This is achieved by breaking down the image into a massive number of fragments and processing each fragment individually. The processes happen independently but may share the same logical sequence of control, and this pattern is named single instruction multiple data (SIMD). A GPU consists of a large array of processing cores, with each of them using SIMD to process a block of fragments.

The characteristic of the GPU allows it to outperform central processing units (CPUs) when performing algorithms that manipulate data in parallel. Take the dot product between two vectors of size 1000 as an example, the processing cores on the GPU may be 100 times slower than the CPU, but if 1000 of them work in parallel and each computes the multiplication of one element from each vector, the overall speed will be 10 times faster. This has allowed computer vision programs that were previously impossible or computationally too expensive

to build. For example, a wearable mediated reality device that make computer-generated information appear to the user as though it was anchored in the real world [6].

The goal of the project is to develop software tools based on fast computation provided by the GPU, to support interactive real-time diagnosis of SEM images for the purpose of assisting the operator or automating procedures, with a focus on the use of FFT.

II. SOFTWARE ASPECTS OF THE TOOLS

A. Design principle

Considering that the work of the project may serve as the foundation stone for many further developments, the design of the software has a strong emphasis on readability and maintainability.

B. The selection of programming language

The SEM used for the project is made by Carl Zeiss AG, which provides an application programming interface (API) that can be used for controlling the SEM and grabbing images from it. The API supports C++ and thus makes it a possible programming language for the project. Being a relatively low-level compiled language makes C++ extremely fast and useful for speed-critical applications. However, it also means that C++ has a complex syntax, which could significantly slow down development if the user does not have enough experience with it.

Considering the time scale of the project and to make development easier for people who will continue the work, Python is selected instead as the programming language, which has a much simpler syntax and is widely supported. With careful design, it has proved to be able to produce useful results despite the slower speed (see later sections).

C. Modules of the software

The software is highly modularised, which maximises readability and maintainability. This also makes it easier to translate the programs into C++ later if better performance is needed, since C++ is mostly used in an object-oriented manner. Fig. 2 shows the six main classes of the software.

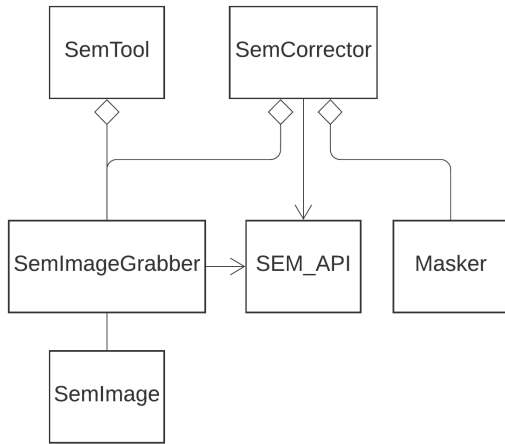


Fig. 2: Class diagram of the software in unified modelling language (UML)

SEM_API is a Python wrapper for the native SEM API written by Luyang Han, which allows

the user to directly control the SEM in Python.

SemImage encapsulates variables and methods that are directly relevant to an image taken by the SEM, and *SemImageGrabber* is a helper class that helps obtain image data from the SEM and create instances of *SemImage* from them. Images can be obtained in two ways:

- From the SEM.
- From a local folder, which is helpful when the SEM is not available.

SemImageGrabber detects if the SEM is available and decides on which way to use.

SemTool handles the creation and rendering of the graphical user interface (GUI). Fig. 3 shows a screenshot of the control panel. The push buttons open a window for the corresponding plot and the radio buttons select algorithms (see section III and IV for detailed descriptions) to be performed on the image. The plots are shown on different windows, and can be opened and closed individually. This improves frame rate by allowing the user to close unneeded windows, and also makes it easier to add other plots later. Table I shows that the improvement is rather significant, as rendering windows is a major time-consuming process of the software. Most of the time, the only window needed is the FFT plot, which means a refresh rate of about 16 frames per second is possible. Note that the algorithms do not slow down the rendering very much, since the computations are high optimised by

the GPU.

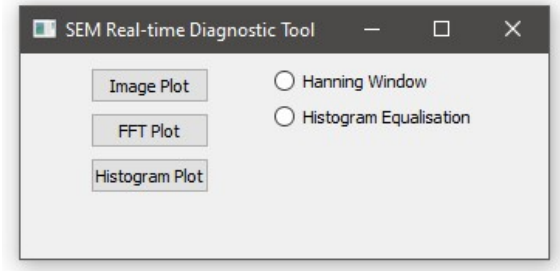


Fig. 3: Screenshot of the GUI.

TABLE I: Results of software framerate test

Windows and algorithms selected	Framerate
no windows, no algorithms	10 ms
one window, no algorithms	60 ms
three windows, no algorithms	180 ms
three windows, two algorithms	200 ms

^aThe numbers are approximate.

^bWith 8-bit grey-scale 1024×768 input images.

^cWith Intel Core i7 and NVIDIA GeForce GTX 1060.

SemCorrector implements the automatic focusing and astigmatism correction algorithm and uses a helper class *Masker* to achieve fast computation (see section V for detailed descriptions). Due to the impact of the COVID-19 pandemic, the algorithm has not been fully tested, and is therefore not integrated to the GUI yet.

III. REAL-TIME HISTOGRAM EQUALISATION FOR IMPROVING IMAGE CONTRAST

IV. REAL-TIME FAST FOURIER TRANSFORM FOR EVALUATING IMAGE FOCUSING AND ASTIGMATISM

V. AUTOMATIC FOCUSING AND ASTIGMATISM CORRECTION ALGORITHM

VI. CONCLUSIONS

REFERENCES

- [1] T. Agemura and T. Sekiguchi, "Secondary electron spectroscopy for imaging semiconductor materials," 2018 International Symposium on Semiconductor Manufacturing (ISSM), Tokyo, Japan, 2018, pp. 1-3, doi: 10.1109/ISSM.2018.8651171.
- [2] T. Cushnie, N. O'Driscoll and A. Lamb, "Morphological and ultrastructural changes in bacterial cells as an indicator of antibacterial mechanism of action," Cellular and Molecular Life Sciences, 2016, vol. 73, no. 23, pp. 4471-4492, doi: 10.1007/s00018-016-2302-2.
- [3] "SEM A to Z", Jeol.co.jp, [Online], available: https://www.jeol.co.jp/en/applications/pdf/sm/sem_atoz_all.pdf. [Accessed: 18 May 2020].
- [4] E. Ghassemieh, M. Acar and H. Versteeg, "Microstructural analysis of non-woven fabrics using scanning electron microscopy and image processing. Part 1: Development and verification of the methods," Proceedings of the Institution of Mechanical Engineers, Part L: Journal of Materials: Design and Applications, 2002, vol. 216, no. 3, pp. 199-207, doi: 10.1177/146442070221600305.
- [5] A. Vladár, M. Postek and M. Davidson, "Image sharpness measurement in scanning electron microscopy-part II," Scanning, 2006, vol. 20, no. 1, pp. 24-34, doi: 10.1002/sca.1998.4950200104.
- [6] J. Fung, F. Tang and S. Mann, "Mediated reality using computer graphics hardware for computer vision," Proceedings. Sixth International Symposium on Wearable Computers, Seattle, WA, USA, 2002, pp. 83-89, doi: 10.1109/ISWC.2002.1167222.