2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 25-29, 2020, Las Vegas, NV, USA (Virtual)

Deep Learning-Based Autonomous Scanning Electron Microscope

Jonggyu Jang[1], Hyeonsu Lyu[1], Hyun Jong Yang[2,3], Moohyun Oh[3], and Junhee Lee[4]

***Abstract*— By virtue of their ultra high resolution, scanning electron microscopes (SEMs) are essential to study topography, morphology, composition, and crystallography of materials, and thus are widely used for advanced researches in physics, chemistry, pharmacy, geology, etc. The major hindrance of using SEMs is that obtaining high quality images from SEMs requires a professional control of many control parameters. Therefore, it is not an easy task even for an experienced researcher to get high quality sample images without any help from SEM experts. In this paper, we propose and implement a deep learning-based autonomous SEM machine, which assesses image quality and controls parameters autonomously to get high quality sample images just as if human experts do. This world's first autonomous SEM machine may be the first step to bring SEMs, previously used only for advanced researches due to its difficulty in use, into much broader applications such as education, manufacture, and mechanical diagnosis, which are previously meant for optical microscopes.**

***Index Terms*— Scanning electron microscope (SEM), deep reinforcement learning (DRL), deep deterministic policy gradient (DDPG), advanced robotics**

I. Introduction

A scanning electron microscope (SEM) produces three-dimensional surface images of a sample from secondary electrons generated by projecting focused electron beams on the sample surface. The output quality of sample images is highly susceptible to many interrelated SEM parameters such as working distance, brightness, magnification, contrast, etc. Furthermore, an optimal set of these parameters is dependent on the direction and height of the sample stage, intensity of illumination, types of samples, etc. Hence, only well-trained SEM experts can find optimal parameters within feasible time. As a consequence, despite of their ultra higher resolution, SEMs have been used only for limited use cases such as advanced researches in physics, chemistry, geology, etc. In this paper, we propose and implement a fully autonomous machine which can evaluate the quality of sample images and control SEM parameters just as highly-skilled human experts do.

This work was supported in part by the MSIT(Ministry of Science and ICT), Korea, under the ITRC(Information Technology Research Center) support program(IITP-2020-2017-0-01635) supervised by the IITP(Institute for Information & communications Technology Promotion), and the Technology Innovation Program (or Industrial Strategic Technology Development Program (20005526, Development of Artificial Intelligence Scanning Electron Microscope) funded By the Ministry of Trade, Industry & Energy(MOTIE, Korea).

[1]Jonggyu Jang and Hyeonsu Lyu are with the School of Electrical and Computer Engineering, Ulsan National Institute of Science and Technology (UNIST), Ulsan 44919, Republic of Korea, (e-mail: {jonggyu,hslyu}@unist.ac.kr)

[2]Hyun Jong Yang (corresponding author) is with department of electrical engineering, Pohang University of Science and Technology (POSTECH), Pohang 37673, Republic of Korea, (email: hyunyang@postech.ac.kr)

[3]Hyun Jong Yang and Moohyun Oh are with Egovid Inc., Ulsan 44919, Republic of Korea, (e-mail: {hjyang,mhoh}@egovid.com)

[4]Junhee Lee is with Coxem Co. Ltd, Daejeon 34025, Republic of Korea, (e-mail: junhee@coxem.com)

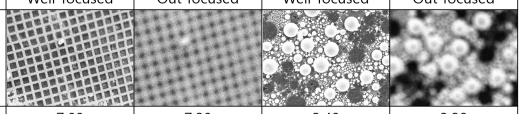
Sample	Grid		Tinball	
Focus	Well-focused	Out-focused	Well-focused	Out-focused
SEM Images				
Gradient	7.09	7.30	3.40	3.38
Variance	4.48	4.22	6.62	6.49

Fig. 1: Examples of the gradient and variance for Grid and Tinball samples, which are normalized into the range of $[0, 10]$.

A. Related Works

*1) **Existing Autofocus Schemes**:* Recently developed SEMs often have an 'autofocus' (AF) functionality that automatically controls the SEM parameters to improve image quality. In the literature [1], [2], AF schemes have been proposed to control one or multiple SEM parameters, with the aim of maximizing various image quality metrics defined based on mathematical functions such as the variance or spatial gradient of pixel values. However, there are fatal theoretical limitations in the existing image quality metrics in some cases, such as recognizing the picture of white noise as a high quality image because of the high variance and spatial gradients of its pixel values. For instance, Fig. 1 depicts the variance and average gradient of the pixel values for two different sample types, *Grid* and *Tinball*, which are normalized into the range of $[0, 10]$. As seen, the gradient and variance of the out-focused images are still high and not distinguishable from the well-focused cases. Consequently, no existing AF scheme can measure image quality while considering both the whole and partial parts of the sample, and consistently perform well for various sample types, sample stage heights, scales, etc. In reality, the existing AF functions are used merely to set initial values of the SEM parameters, and it still requires fine manual control of SEM experts to obtain a high-quality image.

*2) **Deep Learning for Microscope**:* Deep learning powered by deep neural networks (DNNs) and abundant training data has shown its miraculous capability in overdetermined, stationary, and deterministic problems such as classifying images, optimizing digital filters, solving an NP-hard stationary problem, etc. Recently, deep learning has expanded its application to SEM image classification and analysis [3], [4]. While the majority of the literature have studied deep learning as a SEM image classifier or analyzer, our previous study [5] is the only work showing that deep learning can be potentially used as a SEM parameter controller to get high quality images. However, only the image quality prediction problem is addressed in [5], and no parameter

control scheme is proposed. Though our scope is restricted to SEMs, there also have been few works studying deep learning as a controller even for optical microscopes [6].

B. Challenges

To implement a deep learning-based autonomous SEM machine, there are three major challenges:

- A new image quality evaluation metric should be defined, which overcomes the known critical limitations of the existing metrics and shows robust accuracy for a variety of environments such as sample types, height of the sample stage, magnifications, etc.
- Instead of simple supervised learning, a well-designed reinforcement learning-based machine should be designed, which controls the SEM parameters adapting itself to varying environments.
- A full integrated system should be implemented including the following parts: i) softwares and hardwares interconnecting a SEM and a computing node; ii) deep learning-based software machines at the computing node, which evaluates image quality and controls SEM parameters autonomously.

C. Contributions

Our main contributions are summarized as follows:

- To overcome the known limitations of the existing AF metrics, we develop a new metric to evaluate SEM images based on highly-qualified SEM experts' assessment on the quality of images.
- We develop a reinforcement learning-based control machine which controls SEM parameters in order to obtain high quality images autonomously. Our control machine employs the newly developed metric as its reward, and adapts itself to the change in environments including sample types, heights of the sample stage, magnifications, etc., yielding robust control accuracy. Since human experts cannot give the control machine their assessment results on a real-time basis to compute the reward online, we also develop a supervised learning-based scoring machine that mimics human experts to evaluate image quality.
- To train our scoring and control machines, we propose a method to construct a dataset composed of experts' assessment scores, sample images, and control parameters. To make our machines perform robust for varying environments with limited amount of dataset, we design the reward by combining experts' scores and the existing AF metrics.
- We renovate an off-the-shelf SEM, EM-30, and implement the world's first autonomous SEM machine. Specifically, we develop a software/hardware to enable the SEM and the computing node to communicate with each other. We implement a deep learning-based software/hardware system at the computing node, which analyzes input images from the SEM and generates commands to the SEM for adjusting parameters. Our autonomous SEM produces a best-quality sample image within 25 frames in most of the experiments, and shows 93% accuracy in finding the experts' best picks on optimal parameters.

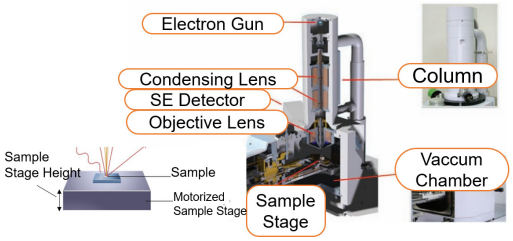


Fig. 2: Illustration of schematic diagram of SEM (model: CX-100S, manufacturer: Coxem).

D. Organization

Section II and III develop our SEM image quality scoring machine and parameter control machine, respectively. In Section IV, details on the implementation are provided, and experimental results are presented in Section V. Section VI concludes the paper.

II. SEM Image Quality Scoring Machine

We start with a brief introduction on SEMs to understand the SEM parameters. Based on this understanding, we develop our scoring machine.

A. Operation Principles and Properties of SEM

Fig. 2 shows a cross-sectional picture of a SEM (model: CX-100S, manufacturer: Coxem). Compared to optical microscopes, a SEM uses an electron gun and condensing lens instead of visible rays and glass lenses, respectively. SEMs can perform at much wider magnification range, up to x10^6, than the magnification range of optical microscopes, which is typically x1 to x1000.

We aim to control three SEM parameters, Working Distance (WD), Brightness (BRT), and Contrast (CNT), which are the most critical parameters to get high quality sample images. WD tunes the focal point of the SEM by changing the current on the coil of the condensing lens. BRT and CNT control brightness and contrast of sample images.

B. Supervised Learning-Based Scoring Machine

A supervised learning-based scoring machine is developed which mimics human SEM experts to assess the quality of images. In our previous work [5], the same concept is introduced. However, the score prediction algorithm proposed in [5] suffers from significant performance degradation in high magnification. To resolve this issue, we propose new methods taking the magnification factor into consideration in building datasets and designing a DNN. The output of the scoring machine shall be used to compute the reward of our parameter control machine which will be described in Section III.

*1) **Dataset**:* While the range of BRT and CNT values in which SEM images are clearly visible is relatively wide, the range of such WD values is very narrow. Hence, many different WD values should be considered in the dataset construction. SEM images are collected in the dataset for various (WD, BRT, CNT) values so that both out-focused and focused images are sufficiently included in the dataset.

Several highly-qualified SEM experts labeled perfectly focused images by 10 points, and totally unrecognizable images such as a noise image by 0 point. Though it is clear for 10-point and 0-point cases, it is not an easy task to evaluate images which are focused to some extent but not perfectly. For each sample type and magnification, the SEM experts selected several reference images representing 1-point to 9-point cases. Then, the other images were labeled in comparison to these representative reference images. Note that how accurately images are labeled in the range of 1 point to 9 points is not critical in constructing the dataset, since the goal is to get 10-point images after all avoiding 0-point cases.

Though finer granularity in the scoring range [0, 10] may help our parameter control machine adjust the parameters more carefully, it may be impossible for human experts to recognize very small difference in the image quality. To compromise, we employed integer scores of 0, 1, ..., 10. In Section III, we will introduce a new metric to complement the weakness of this experts' scoring, such as human mistakes, discrete score values, theoretically not perfect definition of the 1 to 9 points, by combining experts' scores with the existing AF metrics.

Finally, an equal number of images were collected for each magnification of x(500, 1000, 2000, 5000, 10000), because the DNN used in our parameter control machine could be biased if images at a particular magnification are dominating the other magnification cases.

*2) **Deep Neural Network Design**:* Our deep learning-based scoring machine architecture is illustrated in Fig. 3, which mimics human experts to evaluate the image quality score. In fact, if a DNN is excessively deep, i.e., it includes many hidden layers, then it may show robust prediction accuracy even without the magnification value as an input for varying magnification. However, unlike the DNN structure in [5], we propose to use a magnification value as an input as well as a sample image in order to keep the computational complexity feasible while enhancing the prediction accuracy for varying magnification. To standardize the input format, an input image is cropped to be 240×320-sized. In pursuit of enriching the training dataset, an image in the dataset is cropped randomly and flipped vertically and horizontally to generate multiple training images. In the testing, the 240×320-sized center block is cropped from an input image.

As seen from Fig. 3, we used ResNet18 [7][1] as an image processing network, which extracts a 512×1-sized feature vector from an input image.

[1]Although some other existing models such as ResNet50, ResNet101, VGG11, and VGG19 [8] perform also well, ResNet18 shows the best compromise between computation time and score prediction performance.

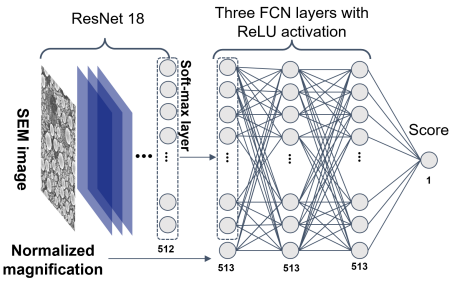


Fig. 3: Illustration of the proposed supervised learning-based scoring machine.

To restrict the cardinality of the variable domain, the magnification value is normalized into [0,1]. Similarly, the 512×1-dimensional feature vector passes through a nonlinear soft-max layer so that each element is normalized into [0,1]. Then, the soft-max layer output is augmented with the normalized magnification value to form a 513×1-sized vector. This augmented vector passes through three fully connected network (FCN) layers, where the final output is the predicted score.

Let us define our scoring machine as SCORE : $\mathbb{R}^{240\times320} \times \mathbb{R} \to \mathbb{R}$ and the dataset as $\mathcal{S}$. Formally, the inputs are defined as $\mathbf{X} \in \mathbb{R}^{240\times320}$ and MAG, where $\mathbf{X}$ denotes the pixel value matrix of an input image, and MAG is the normalized magnification. The elements of dataset $\mathcal{S}$ are tuples of ($\mathbf{X}$, MAG, ES), where ES denotes the experts' score. In the training, the DNN is updated such that the following root-mean-square-error (RMSE) loss function is minimized:

$$\sqrt{\frac{1}{|\mathcal{S}|}\sum_{(\mathbf{X},\mathrm{MAG},\mathrm{ES})\in\mathcal{S}}(\mathrm{ES}-\mathrm{SCORE}(\mathbf{X},\mathrm{MAG}))^2}. \quad (1)$$

III. Parameter Control Machine for Adjusting SEM Parameters

In this section, our reinforcement learning-based parameter control machine is presented, which controls the three parameters of (WD, BRT, CNT) to get high quality image autonomously. For continuous control of the parameters, the deep deterministic policy gradient (DDPG) algorithm [9] is adapted with careful design of states, actions, and rewards. Since in each reinforcement step, it is not possible for human experts to give the machine their score for computing reward online, we utilize the output of our scoring machine described in Section II, which predicts experts' scores. Furthermore, we also exploit the existing AF metrics in designing the reward to cope with cases unexplored in the training.

A. Notation of Variables

Let us denote WD, BRT, CNT, and the sample image updated at the t-th reinforced control step by $p_{1,t}$, $p_{2,t}$, $p_{3,t}$, and $\mathbf{X}_t \in \mathbb{R}^{240\times320}$, respectively. Note that the sample image also changes as a consequence of the change in the parameters. For notional convenience, the following notations are used: $\mathbf{p}_t = [p_{1,t}, p_{2,t}, p_{3,t}]^{\mathrm{T}} \in \mathbb{R}^3$; the upper

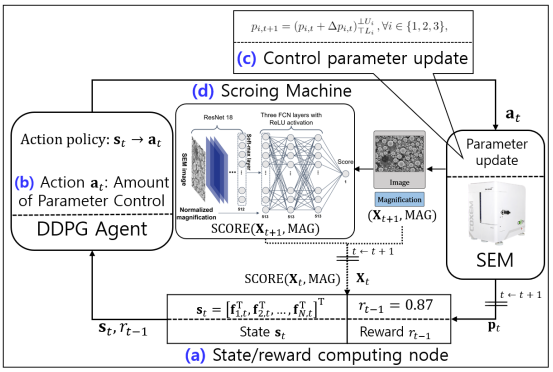


Fig. 4: Block diagram of the proposed parameter control machine.

and lower bounds of (WD, BRT, CNT) values are defined by (U_1, U_2, U_3) and (L_1, L_2, L_3), respectively; the amounts of updates in WD, BRT, CNT values in the t-th step are denoted by $\Delta p_{1,t}$, $\Delta p_{2,t}$, and $\Delta p_{3,t}$, respectively.

B. Deep Reinforcement Learning-Based Parameter Control

Fig. 4 depicts overall procedure of the proposed deep reinforcement learning (DRL)-based parameter control machine, each block of which is explained as follows:

(a) For $\mathbf{p}_t$, $\mathbf{X}_t$, and SCORE($\mathbf{X}_t$, MAG), the state/reward computing node computes the updated state $\mathbf{s}_t$, composed of the SCORE value, averaged pixel value, averaged gradient and variance of pixel values, etc., and the reward at the $(t-1)$-th step, r_{t-1}. Then, the computing node hands over $\mathbf{s}_t$ to the DDPG agent.

(b) For given state $\mathbf{s}_t$, the DDPG agent calculates the action $\mathbf{a}_t$, and then commands the SEM to update the parameters according to $\mathbf{a}_t$.

(c) In the SEM, a software agent updates the parameters complying with given $\mathbf{a}_t$. The SEM reprints the sample image with the updated parameters to get $\mathbf{X}_{t+1}$ and delivers it to the scoring machine along with the current parameter settings.

(d) The scoring machine calculates the score for the input image to get SCORE($\mathbf{X}_{t+1}$, MAG) and then passes it to the state/reward computing node.

* The machine repeats the above procedure until the score exceeds a certain threshold.

The details of each block are provided in what follows.

*1) **State**:* The state $\mathbf{s}_t$ should represent all the information on the status or environment at the t-th step. If we draw a graph that plots the score of the image against WD, the graph is not a monotonically increasing or decreasing function, but rather concave at least around the peak corresponding to the optimal WD. Thus, if only a single shot of an out-focused image is given, it is not possible to determine whether to increase or decrease the current WD in pursuing the optimal WD. Therefore, if $\mathbf{s}_t$ is composed of the sample image and parameters only at the t-th step, the control machine cannot properly find the optimal parameters.

To resolve this problem, we propose to construct the state by the information gathered from the previous N control steps. Formally, the feature vector of the $(t-N+n)$-th control step is defined by

$$\mathbf{f}_{n,t} = \big[\mathrm{SCORE}(\mathbf{X}_{t-N+n}, \mathrm{MAG}), \mathrm{AVG}(\mathbf{X}_{t-N+n}), \mathrm{GRD}(\mathbf{X}_{t-N+n}), \mathrm{VAR}(\mathbf{X}_{t-N+n}), \hat{\mathbf{p}}_{n,t}^{\mathrm{T}}, \mathbf{b}_{t-N+n}^{\mathrm{T}}\big]^{\mathrm{T}} \in \mathbb{R}^{10}, \tag{2}$$

where the functions (AVG, GRD, VAR) and the variables ($\hat{\mathbf{p}}_{n,t}$, $\mathbf{b}_t$) are defined as follows:

- Average pixel value: $\mathrm{AVG}(\mathbf{X}_t) = \frac{\sum_{m=1}^{240}\sum_{l=1}^{320} x_t^{(m,l)}}{240\cdot 320}$, where $x_t^{(m,l)}$ is the (m,l)-th element of $\mathbf{X}_t$.
- Gradient-based sharpness measuring function [1], [2]:

$$\mathrm{GRD}(\mathbf{X}_t) = \sum_{m=1}^{239}\sum_{l=1}^{319} |x_t^{(m,l+1)} - x_t^{(m,l)}| + |x_t^{(m+1,l)} - x_t^{(m,l)}|.$$

- Variance-based sharpness measuring function [1], [2]:

$$\mathrm{VAR}(\mathbf{X}_t) = \frac{\sum_{m=1}^{240}\sum_{l=1}^{320}\Big(x_t^{(m,l)} - \mathrm{AVG}(\mathbf{X}_t)\Big)^2}{240\cdot 320}.$$

- $\hat{\mathbf{p}}_{n,t} = \mathbf{p}_{t-N+n} - \mathbf{p}_{t-N+1}$ for all $n = 1, 2, ..., N$.
- $\mathbf{b}_t = [\mathbb{I}_{U_1}(p_{1,t}) - \mathbb{I}_{L_1}(p_{1,t}), \mathbb{I}_{U_2}(p_{2,t}) - \mathbb{I}_{L_2}(p_{2,t}), \mathbb{I}_{U_3}(p_{3,t}) - \mathbb{I}_{L_3}(p_{3,t})]^{\mathrm{T}} \in \mathbb{R}^3$, where $\mathbb{I}_a(b) = 1$ if $a = b$, and $\mathbb{I}_a(b) = 0$ otherwise.

To consider more directly how the parameters have been updated, we define $\hat{\mathbf{p}}_{i,t}$ in the state, which is the difference of the control parameters at the $(t-N+1)$-th and $(t-N+i)$-th control steps. Each element of $\mathbf{b}_t$ indicates whether a parameter reaches an upper or lower bound. For instance, the first element $\mathbb{I}_{U_1}(p_{1,t}) - \mathbb{I}_{L_1}(p_{1,t})$ becomes 1 if WD reaches its upper-bound, i.e., $p_1 = U_1$, and -1 if WD reaches its lower-bound, i.e., $p_1 = L_1$.

Formally, the state $\mathbf{s}_t$ is defined as follows

$$\mathbf{s}_t = \big[\mathbf{f}_{1,t}^{\mathrm{T}}, \mathbf{f}_{2,t}^{\mathrm{T}}, \cdots, \mathbf{f}_{N,t}^{\mathrm{T}}\big]^{\mathrm{T}}. \tag{3}$$

*2) **Action**:* By observing the state $\mathbf{s}_t$ defined in (3), the DDPG agent takes an action to control the following parameter values: $p_{1,t}$, $p_{2,t}$, and $p_{3,t}$. The action $\mathbf{a}_t$ is defined as $\mathbf{a}_t = [\Delta p_{1,t}, \Delta p_{2,t}, \Delta p_{3,t}]^{\mathrm{T}} \in \mathbb{R}^3$. For given $\mathbf{a}_t$, the parameters are updated from

$$p_{i,t+1} = (p_{i,t} + \Delta p_{i,t})_{\top L_i}^{\perp U_i}, \forall i \in \{1,2,3\}, \tag{4}$$

where $x_{\top a}^{\perp b} = \max\{\min\{x, b\}, a\}$ denotes the clamping function. Note that if $\Delta p_{i,t}$ is unbounded, the DDPG agent may lead us directly to optimal parameter values. Most importantly, however, even for an identical sample, optimal parameters are different for different sample stage heights. Thus, if a sample with a certain stage height, which is not included in the training dataset, is tested, then the agent may end up with an output over-fitted to a wrong height case in the training dataset. Thus, we propose to bound $\Delta p_{i,t}$ by

$$|\Delta p_{i,t}| \le \rho_i \cdot (U_i - L_i), \quad i \in \{1,2,3\}, \tag{5}$$

where $\rho_i \in (0,1)$ is a design parameter. Note that we do not fix the value $\Delta p_{i,t}$ such that $\Delta p_{i,t}$ can be naturally decreased as the reward converges.

*3) **Reward**:* By taking action $\mathbf{a}_t$, the environment (SEM) updates the parameters. Subsequently, the DDPG agent

receives the reward for that action from the state/reward computing node as in Fig. 4. The reward is generally defined as a goal to maximize in DRL, such as winning rate, incomes, and game scores. Our goal is obviously to get an image with a high experts' score. Thus, the reward should be designed such that the output of our scoring machine is maximized.

In reality, however, we have limited data in the training dataset. Hence, if some cases with stage heights and sample types, which are not included in the training dataset, are encountered, then there is a chance that the control performance is degraded or the DDPG agent gives an overfitted output. Furthermore, score labeling cannot be perfectly consistent for many data, since it is done by *human* experts.

Thus, we design the reward also with supplementary deterministic functions such as averaged gradient and variance of the input pixel values, which played a role of measuring image quality in the conventional studies. Supposedly, one may think of the reward $\text{SCORE}(\mathbf{X}_t, \text{MAG}) + \text{GRD}(\mathbf{X}_t) + \text{VAR}(\mathbf{X}_t)$. Recall that our control machine terminates the procedure if SCORE exceeds a certain threshold, say ω. Suppose that the DDPG has reached $\omega - \varepsilon$ in reward with an arbitrarily small number ε. If the DDPG tries to exceed ω, then it is obvious that the future reward becomes 0. Since the ultimate goal of DRL is to maximize the future reward, our DDPG will be trained with the aforementioned reward design such that it stays forever at a set of parameters producing the reward of $\omega - \varepsilon$.

As a remedy, we define the reward of the t-th step as

$$r_t = \text{SCORE}(\mathbf{X}_{t+1}, \text{MAG}) - \text{SCORE}(\mathbf{X}_t, \text{MAG}) + c_1(\text{GRD}(\mathbf{X}_{t+1}) - \text{GRD}(\mathbf{X}_t)) + c_2(\text{VAR}(\mathbf{X}_{t+1}) - \text{VAR}(\mathbf{X}_t)), \quad (6)$$

where c_1 and c_2 are weight coefficients. Note that since the reward is defined by the change in the image quality from the t-th to $(t+1)$-th step, the termination policy looks hazy to the DDPG agent during the training. Hence, the DDPG is simply trained such that the reward increases at any instance. Furthermore, this reward design results in robust deep learning performance in different SEM hardwares, where the scale of the measurements on $\mathbf{X}$ can be different under different hardware settings.

*4) **State-Action Function (Actor DNN)**:* In the conventional DDPG algorithm [9], an actor DNN is a function of state $\mathbf{s}_t$, and designed to obtain the value of the action $\mathbf{a}_t$ in a continuous space. Specifically, if we denote the optimal action as $\mathbf{a}_t^* = A^*(\mathbf{s}_t)$, then the actor DNN aims to estimate $A^*(\mathbf{s}_t)$. For the DNN parameter $\boldsymbol{\theta}$, the output of the actor DNN is denoted by as $\hat{A}(\mathbf{s}_t; \boldsymbol{\theta})$.

*5) **Action-state-value Function (Critic DNN)**:* By the Bellman equation, the action-state-value function with policy π and forgetting factor γ is defined as

$$Q(\mathbf{s}_t, \mathbf{a}_t) = \mathop{\mathbb{E}}_{\mathbf{s}_{t+1}, \mathbf{a}_{t+1}, \mathbf{s}_{t+2}, \ldots} \left[\sum_{n=t}^{\infty} \gamma^{n-t} r_n \right] \overset{(a)}{=} \mathop{\mathbb{E}}_{\mathbf{s}_{t+1}, \mathbf{a}_{t+1} \sim \pi} \left[r_t + \gamma Q_\pi(\mathbf{s}_{t+1}, \mathbf{a}_{t+1}) \right]. \quad (7)$$

With the DNN parameter $\boldsymbol{\phi}$, the critic DNN outputs the estimate of $Q(\mathbf{s}_{t+1}, \mathbf{a}_{t+1})$, denoted by $\hat{Q}(\mathbf{s}_{t+1}, \hat{A}(\mathbf{s}_{t+1}; \boldsymbol{\theta}); \boldsymbol{\phi})$.

*6) **Training Objectives and Target DNN Parameters**:* From (7), the target value in update of the critic DNN is $r_t + \gamma \hat{Q}(\mathbf{s}_{t+1}, \hat{A}(\mathbf{s}_{t+1}; \boldsymbol{\theta}); \boldsymbol{\phi})$. Note that $\boldsymbol{\phi}$ and $\boldsymbol{\theta}$ are recursively included in this target value. Hence, if $\boldsymbol{\phi}$ and $\boldsymbol{\theta}$ are trained for the DNNs to pursuit this target value, they often diverge during the training. Therefore, the concept of target parameters is considered as in [10]. The target DNN parameters for $\boldsymbol{\phi}$ and $\boldsymbol{\theta}$ are denoted by $\boldsymbol{\phi}^-$ and $\boldsymbol{\theta}^-$, respectively. Then, the target value for the critic DNN update is defined as

$$z_t = r_t + \gamma \hat{Q}(\mathbf{s}_{t+1}, \hat{A}(\mathbf{s}_{t+1}; \boldsymbol{\theta}^-); \boldsymbol{\phi}^-). \quad (8)$$

Therefore, the critic DNN parameter $\boldsymbol{\phi}$ is updated to reduce the critic loss function, which is defined as

$$L(\boldsymbol{\phi}) = \mathbb{E}\ \ \delta_t^2, \quad (9)$$

where $\delta_t = z_t - \hat{Q}(\mathbf{s}_t, \mathbf{a}_t; \boldsymbol{\phi})$ represents the temporal difference error.

On the other hand, the aim of the actor DNN $\hat{A}(\mathbf{s}_t; \boldsymbol{\theta})$ is to obtain an estimate of the optimal policy. Hence, the actor DNN parameter is updated to maximize the output of the critic DNN, $\hat{Q}(\mathbf{s}_t, \hat{A}(\mathbf{s}_t; \boldsymbol{\theta}); \boldsymbol{\phi})$, with policy $\hat{A}(\mathbf{s}_t; \boldsymbol{\theta})$.

*7) **Exploration and Exploitation**:* The DDPG agent either randomly explores the environment to get more information or exploits its best actions during training. We employ the ϵ-greedy policy for exploration and exploitation. Specifically, the action $\mathbf{a}_t$ is taken from

$$\mathbf{a}_t = \begin{cases} \hat{A}(\mathbf{s}_t; \boldsymbol{\theta}), & \text{with probability } 1 - \epsilon \\ \hat{A}(\mathbf{s}_t; \boldsymbol{\theta}) + \mathbf{n}_t, & \text{otherwise,} \end{cases} \quad (10)$$

where $\mathbf{n}_t$ denotes additive Gaussian random noise, i.e., $\mathbf{n}_t \sim \mathcal{N}(\mathbf{0}, \sigma^2 \mathbf{I}_3)$.

*8) **Prioritized Experience Replay**:* In the training, an experience tuple $\mathbf{e}_t = (\mathbf{s}_t, \mathbf{a}_t, r_t, \mathbf{s}_{t+1})$ is obtained after each control step. Because experience tuples of consecutive control steps are highly correlated, the DNN parameters converge slow during training. Therefore, we resolve this issue by employing the prioritized experience replay (PER) scheme [11]. Let us denote the set of indices of experience tuples stored in the replay buffer by $\mathcal{D}$. Then, each of experience tuples in the replay buffer is selected by probability P_t to form a mini-batch $\mathcal{R}$, where P_t is defined by

$$P_t = \frac{o_t^\kappa}{\sum_{j \in \mathcal{D}} o_j^\kappa}. \quad (11)$$

Here, $o_t = |\delta_t|$ and κ denote the priority of experience $\mathbf{e}_t$ and a hyper-parameter, respectively.

*9) **Training and Target DNN Update**:* For the update of the target DNNs for given mini-batch $\mathcal{R}$, we employ the important-sampling (IS) method such that the DNN updates can be not over-fitted to the experiences which are frequently selected in $\mathcal{R}$. Specifically, the IS weight w_t for the t-th

experience is defined by

$$w_t = \left(\frac{1}{|\mathcal{D}|}\frac{1}{P_t}\right)^{\zeta}, \tag{12}$$

where ζ is a hyper-parameter. Then, the critic DNN parameter $\boldsymbol{\phi}$ is updated to minimize the loss function (9) by the stochastic gradient method with the IS weight (12) as

$$\boldsymbol{\phi} \leftarrow \boldsymbol{\phi} + \alpha_{\boldsymbol{\phi}} \sum_{t\in\mathcal{R}} w_t \delta_t \nabla_{\boldsymbol{\phi}} \hat{Q}(\mathbf{s}_t, \mathbf{a}_t, \boldsymbol{\phi}), \tag{13}$$

where $\alpha_{\boldsymbol{\phi}}$ represents learning rate of the critic DNN.

The actor DNN parameter $\boldsymbol{\theta}$ is updated to obtain the action policy $\hat{A}(\mathbf{s}_t;\boldsymbol{\theta})$ that maximizes the critic DNN output $\hat{Q}(\mathbf{s}_t, \hat{A}(\mathbf{s}_t;\boldsymbol{\theta});\boldsymbol{\phi})$ as follows:

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \alpha_{\boldsymbol{\theta}} \sum_{t\in\mathcal{R}} \nabla_{\boldsymbol{\theta}} \hat{A}(\mathbf{s}_t;\boldsymbol{\theta}) \left(\nabla_{\mathbf{a}} \hat{Q}(\mathbf{s}_t, \mathbf{a};\boldsymbol{\phi})|_{\mathbf{a}=\hat{A}(\mathbf{s}_t;\boldsymbol{\theta})}\right), \tag{14}$$

where $\alpha_{\boldsymbol{\theta}}$ denotes learning rate of the actor DNN.

On the other hand, the target DNN parameters are updated by the soft target DNN update scheme [9] as follows:

$$\boldsymbol{\theta}^- \leftarrow \tau\boldsymbol{\theta} + (1-\tau)\boldsymbol{\theta}^-, \quad \boldsymbol{\phi}^- \leftarrow \tau\boldsymbol{\phi} + (1-\tau)\boldsymbol{\phi}^-, \tag{15}$$

where $\tau \ll 1$ is a hyper-parameter.

*10) **DNN Structure**:* The actor and critic DNNs have (input size, output size) of $(10N,\ 3)$ and $(10N+3,\ 1)$, respectively. Since the input of the actor/critic network is a simple feature vector with no spatial correlation unlike images, the control machine can be operated effectively enough without complicated network design. The critic and actor DNNs are designed to have the same hidden layers of size of 256, 512, 512, 512, 512 and 256 neurons, and have the same ReLU activation function.

The overall proposed DRL algorithm is summarized in Algorithm 1.

Algorithm 1 Training Algorithm of the Proposed Parameter Control Machine.

Initialize:
  DNN parameters: $\boldsymbol{\theta}$, $\boldsymbol{\theta}^-$, $\boldsymbol{\phi}$, and $\boldsymbol{\phi}^-$.
  Hyper-parameter s: κ, ζ, γ, τ, and ϵ.
  Initial state $\mathbf{s}_0$ and termination threshold ω.
for episode $= 1$ to E **do**
  for $t = 1$ to T **do**
    Action $\mathbf{a}_t$ is taken by ϵ-greedy policy in (10).
    Observe reward r_t and state $\mathbf{s}_{t+1}$.
    Store experience $(\mathbf{s}_t, \mathbf{a}_t, r_t, \mathbf{s}_{t+1})$ in $\mathcal{D}$.
    Sample mini-batch $\mathcal{R}$ from replay $\mathcal{D}$ with probability (11).
    Compute IS weights by (12).
    Update priority $o_i \leftarrow |\delta_i|, \forall i \in \mathcal{R}$.
    Update DNN parameters $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$ by (13) and (14), respectively.
    Update the target DNN parameters $\boldsymbol{\theta}^-$ and $\boldsymbol{\phi}^-$ by (15).
    if SCORE$(\mathbf{X}_t, \text{MAG}) > \omega$ **then**
      break
    end if
  end for
end for

IV. Implementation of the Autonomous SEM

Fig. 5 shows the overall implementation of our autonomous SEM. The SEM software named 'NanoStation' saves the SEM image and the current SEM parameters in image and text files, respectively in the parameter control machine (' (a)' in Fig. 5). After reading the saved files, the scoring machine ('(b)' in Fig. 5) and state/reward computing node ('(c) in Fig. 5) calculate the score of the image and state/reward, respectively. The parameter control machine reads the calculated score, state, and reward to calculate the action value and write the calculated action value in a text file ('(d)' in Fig. 5). A software named 'Middle Client' transmits the contents of that txt file to NanoStation ('(e)' in Fig. 5). The SEM hardware controls the parameter complying with the message from Middle Client ('(f)' in Fig. 5), and prints a new image to deliver it to NanoStation ('(g)' in Fig. 5).

A. SEM Hardware Specification

An 'EM-30' SEM machine is used, the specification of which is listed as follows:

Element	Specification
Software	NanoStation 3.0™
Magnification Range	x20∼x150,000
Accelerating Voltage	1∼30kV
Resolution	5nm at 30kV
Electron Source	Tungsten Filament
Detector	Secondary electron detector
Observation Area	40mm
Maximum sample size	60mm in diameter

B. Data Set Collection

We developed a Microsoft Foundation Class Library (MFC)-based software to construct a dataset. Our dataset consists of 30,272 images of grid and tinball samples in total, amongst which 27,235 images are used for training and 3,037 images are used for the scoring machine evaluation in test.

C. Implementation Settings for the Scoring Machine

In the training of the DNN of the scoring machine, the maximum epoch of 1000, batch size of 1, and learning rate of $2 \cdot 10^{-4}$ were used. In addition, NVIDIA RTX 2080 Ti, CUDA 10.0, Python 3.6.8, Pytorch 1.1.0, and OpenCV 4.1.0 were used to implement the scoring machine.

D. Implementation Settings for the Parameter Control Machine

We implemented the parameter control machine using a desktop (model: DELL XPS 8930) and SEM (model: EM-30). In the training, the following hyper-parameters are used: maximum episode E of 500, maximum step T of 50, learning rate $(\alpha_{\boldsymbol{\theta}}, \alpha_{\boldsymbol{\phi}})$ of $(3 \cdot 10^{-6}, 1 \cdot 10^{-4})$, forgetting factor γ of 0.25, ϵ-greedy parameter ϵ of $0.2 \sim 0.99$, target network parameter τ of 0.001, PER parameter (κ, ζ) of $(0.7, 0.4)$, and noise parameter σ of 0.25. In addition, the control

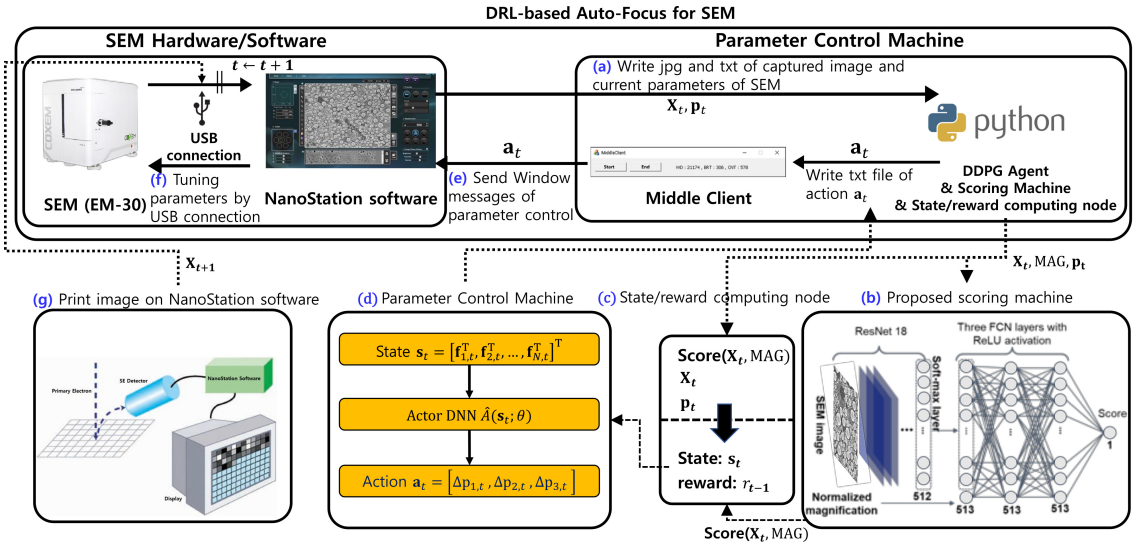


Fig. 5: Overall block diagram of the implemented autonomous SEM machine.

machine was implemeted under the following environments: NVIDIA GTX 1660 Ti, CUDA 10.0, Python 3.6.8, Pytorch 1.10, and Tensorflow 1.12.0. The range of the SEM control parameters was defined by $(L_1, L_2, L_3) = (4, 200, 400)$ and $(U_1, U_2, U_3) = (27, 400, 600)$. The action control parameters were set to be $(\rho_1, \rho_2, \rho_3) = (0.038, 0.013, 0.013)$. The image quality threshold for termination was set to be $\omega = 8.8$.

E. Conventional Autofocus Scheme

For comparison, one of the widely-used conventional AF schemes was implemented [1], [2]. The conventional scheme utilizes a 1-dimension variable search only for WD to maximize the absolute gradient score function. To lower the computational complexity, the conventional scheme observes 50×500 pixels of a SEM image for the computation of the score function. Note that the size of 50×500 pixels is enough for evaluating the quality of images, and there is no performance difference even if the score function is computed for bigger sized images. We manually found the optimal BRT and CNT values by exhaustive search.

V. Experimental Results

A. Experimental Results of the Scoring Machine

Fig. 6 depicts the RMSE loss during the training and testing for each epoch. In the figure, the loss is successfully decreased for both training and testing. In the test, we obtained the average RMSE of 0.1887. Therefore, our scoring machine is proven to mimic human experts in scoring the image quality as desired with only average error of 0.1887.

B. Experimental Results of the Parameter Control Machine

*1) **Rightness of the DNN Design**:* Fig. 7 depicts the sum of the rewards and the average RMSE of the critic DNN for each episode. In the figure, the average loss of the critic DNN decreases as designed, from which it is

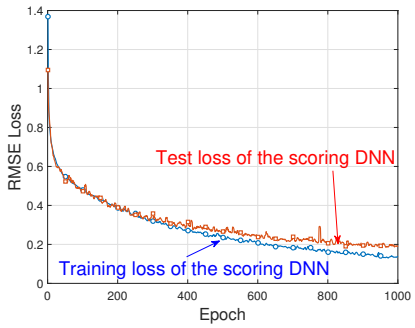


Fig. 6: RMSE loss of the proposed scoring machine in the training and testing for each epoch.

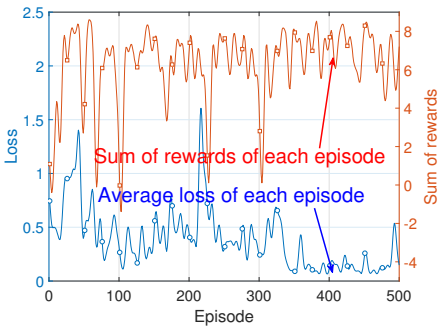


Fig. 7: Sum rewards and loss of the critic DNN for each episode.

seen that our critic DNN well-estimates the action-state-value function. Note that the sum reward is limited by the difference between the score at the initial step and 10. In most of the initial steps, the score was 1 point, and hence the maximum sum reward should be 9 points. As seen from the figure, as episode continues, the control machine reinforces itself and shows relatively high rewards of 5 to 8 points.

*2) **Convergence Speed and Probability of Control Success**:* In Fig. 8, the numbers of frames required for our control machine to converge and the probability of suc-

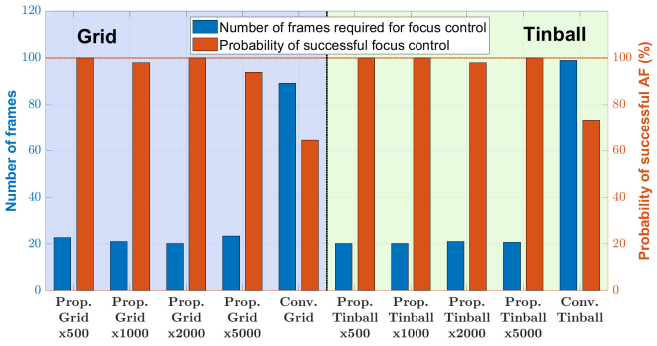


Fig. 8: Number of frames required for converge and probability of successful control under various environment settings.

sample	Scheme	Magnification	RMSE of WD	Prob. of successful BRT & CNT control	Avg. Score
Grid	Proposed	500	0.2267	1.0000	9.5353
		1000	0.1893	0.9796	9.2535
		2000	0.1324	1.0000	9.5559
		5000	0.0932	1.0000	9.0366
	Conventional		0.3681	-	7.0182
Tinball	Proposed	500	0.1292	1.0000	9.3748
		1000	0.1034	1.0000	9.2749
		2000	0.0813	1.0000	9.1504
		5000	0.0718	1.0000	9.3766
	Conventional		0.2153	-	7.5945

TABLE I: Parameter control accuracy and the average score after the control.

cessful controls are depicted for various magnifications of tinball and grid samples. It is seen that the proposed scheme requires much less frames than the conventional scheme until converge, while having higher probability of success. In addition, the proposed scheme performs consistently under various environment settings except for 'grid with magnification of 5000'. This is because images of the grid sample in high magnification sometimes capture hollow parts of the sample, i.e., there is no feature on the captured images.

*3) **Control Accuracy**:* Table I summarizes the control accuracy of the proposed control machine and the average score after the control. In general, it becomes more difficult to find the optimal WD value in high magnification. As seen from the table, our control machine even with the highest magnification shows higher accuracy in finding the optimal WD than the conventional scheme, even though the RMSE of the conventional scheme is averaged across the entire magnification range. Unlike WD, we can get high quality images for relatively wide ranges of BRT and CNT values. As seen from the table, our control machine successfully adjusts BRT and CNT to make them fall into the high quality ranges with almost 100% probability. Finally, as seen from the last column of the table, the average score after the control significantly surpasses that of the conventional scheme for both the sample types under all the magnification setups.

*4) **Exemplary Results**:* Fig. 9 shows examples of sample images before and after our parameter control for four different magnification setups and two different sample types. As seen from the figure, the images after the control looks almost like those of experts' best picks.

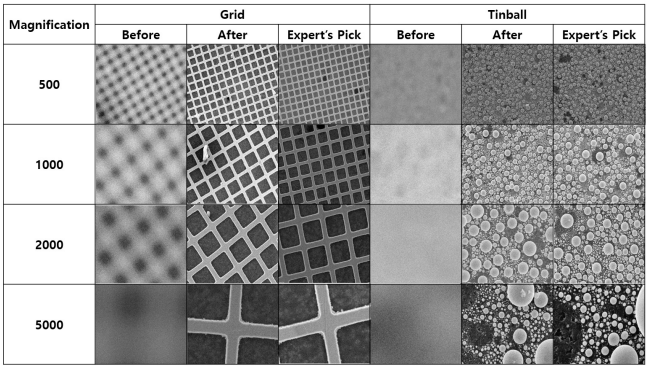


Fig. 9: Control result examples for Grid and Tinball sample types.

VI. Conclusion

An autonomous SEM has been implemented by proposing and implementing a image quality scoring and parameter control machines. Our autonomous SEM can produce high quality images, to which human experts would give 9 points out of 10 points. Therefore, it can be said that our machine can deal with the SEM just as if human experts do. Using our scoring and parameter control machines, SEMs become extremely easy to use and thereby can replace optical microscopes in a variety of applications such as education, mechanical diagnosis, and manufacture. Exemplary video results of our implementation experiments can be found in the following link: https://youtu.be/MvSaoPQvDdo.

References

[1] A. Santos, C. O. D. Solorzano, J. J. Vaquero, J. M. Pena, N. Malpica, and F. D. Pozo, "Evaluation of autofocus functions in molecular cytogenetic analysis," *Journal of Microscopy*, vol. 188, 1997.

[2] F. C. A. Groen, I. T. Young, and G. Ligthart, "A comparison of different focus functions for use in autofocus algorithms," *Cytometry*, vol. 6, no. 2, pp. 81–91, 1985.

[3] M. H. Modarres, R. Aversa, S. Cozzini, R. Ciancio, A. Leto, and G. P. Brandino, "Neural network for nanoscience scanning electron microscope image recognition," *Scientific Reports*, vol. 7, 2017.

[4] C. D. Nobili and S. Cozzini, "Deep learning for nanoscience scanning electron microscope image recognition," *Master in High Performance Computing (MHPC)*, 2017.

[5] H. Kim, M. Oh, H. Lee, J. Jang, M. U. Kim, H. J. Yang, M. Ryoo, and J. Lee, "Deep-learning based autofocus score prediction of scanning electron microscope," in *Microscopy and Microanalysis (M&M)*, 2019.

[6] H. Pinkard, Z. Phillips, A. Babakhani, D. A. Fletcher, and L. Waller, "Deep learning for single-shot autofocus microscopy," *Optica*, vol. 6, no. 6, pp. 794–797, Jun. 2019.

[7] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *CoRR*, vol. abs/1512.03385, 2015. [Online]. Available: http://arxiv.org/abs/1512.03385

[8] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *CoRR*, vol. abs/1409.1556, 2014. [Online]. Available: http://arxiv.org/abs/1409.1556

[9] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," in *Proc. International Conference on Learning Representations (ICLR)*, 2016.

[10] V. Mnih *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[11] T. Schaul, J. Quan, I. Antonoglou, and D. Silver, "Prioritized experience replay," *CoRR*, vol. abs/1511.05952, Nov. 2015. [Online]. Available: http://arxiv.org/abs/1511.05952