Method of aligning a charged particle beam apparatus

1. Introduction

Alignment quality determines the performance of a charged particle beam apparatus. Sequential decision-making algorithm (an agent) may be trained via reinforcement learning in order to perform the apparatus alignment procedure. Provided method of apparatus alignment comprises two phases: pre-deployment phase, where the two populations of agents are trained in an adversarial manner, and post-deployment phase, where the trained agents are utilized in order to align a charged particle beam apparatus. Further paragraphs describe the procedure of training, provide the detailed analysis of the training performance, and report the empirical results of the agent evaluation. Trained agents are able to outperform the human expert in the number of alignment tasks including the aperture, focus and stigmator alignments of the dual-beam microscope.

2. Related work

Real-world robotic tasks require the sample efficiency of the training procedure. Variety of techniques, including the offline [*1*], off-policy [*2*], model-based [*3*] reinforcement learning, imitation learning [*4*], pre-training in simulation [*5*], have been successfully applied in order to minimize the number of physical interactions required by the training procedure. Alignment of a charged particle beam apparatus involves the time-consuming physical processes that may cause the significant deterioration of the aligned mechanisms. Mentioned techniques, thus, are essential for the procedure of alignment.

An adversarial, population-based reinforcement learning has recently allowed the artificial agents to outperform the (teams of) humans in the challenging virtual environments, such as Dota [*6*] and Starcraft [*7*].

In order to train the high quality alignment agents, the provided method of aligning a charged particle beam apparatus relies on an adversarial (zero-sum) formulation of the training procedure and achieves the required degree of sample efficiency by incorporating the techniques that are widely used in real-world robotics.

3. Method of aligning a charged particle beam apparatus

Training procedure maintains two populations (teams) of agents: the *alignment* team and the *misalignment* team of agents. The iteration of the training procedure comprises the steps of:

- Selection of the agent from the *misalignment* team of agents.
- Selection of the agent from the *alignment* team of agents.
- Providing the charged particle beam apparatus to the misaligned (not optimally aligned) state by the *misalignment* team agent.
- Executing the alignment procedure of the charged particle beam apparatus by the *alignment* team agent.
- Computing the quality parameters related to the performed alignment procedure.
- Modification of at least one of the *alignment* team agents.
- Modification of at least one of the *misalignment* team agents.

Neural networks underlying the inference mechanism of the agent are updated according to the selected reinforcement learning algorithm such as Soft Actor Critic [*8*] or Proximal Policy Optimization [*9*]. An *alignment* team agent receives the reward proportional to the computed quality parameters related to the trajectory of the performed alignment procedure. The reward provided to the *misalignment* team agent is the negative reward provided to the *alignment* team agent. Training procedure is, thus, organized as a zero-sum game. Agents within the team (population) may have different architectures of the neural networks. Trainable parameters of each agent are unique. The best of the trained agents are selected at the end of the training procedure. Selected agents undergo the deployment procedure and are used in order to perform the alignment of the user charged particle beam apparatuses.

4. Defocus lens alignment of the dual-beam microscope

4.1. Defocus lens alignment

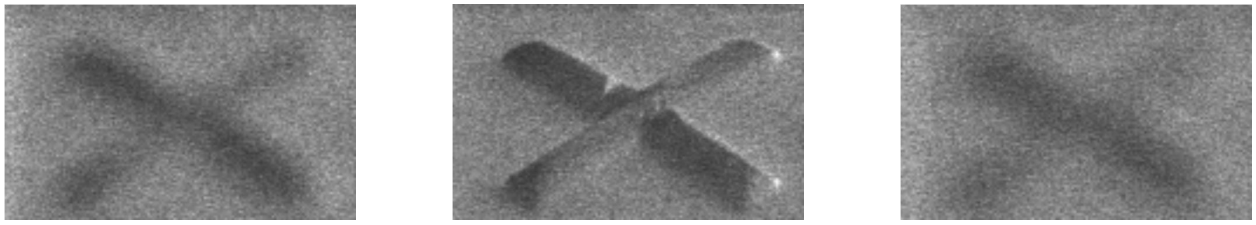

Figure 4.1.1. Voltage on the defocus lens affects the quality of the image acquired using the electron beam of the dual-beam microscope. Left to right: image acquired with the underfocused, focused and the overfocused beam.

Electron beam is focused by the electrostatic lens *L1* (defocus lens). The agent is trained to adjust the voltage on the defocus lens in order to focus the beam to the surface of the specimen. Focused beam provides a sharp image.

4.2 Defocus lens simulator

Defocus simulator approximates the dynamics of the defocus lens of the electron microscope. Simulator allows to pre-train the reinforcement learning agents without the physical interaction with a microscope. The agent acquires the initial knowledge in a simulated environment, hyperparameters are tuned. Pre-trained agent continues the training procedure on the physical apparatus, where the agent adapts to the dynamics of the real-world tool.

4.3. Pre-training in a simulated defocus lens environment

Simulator engine comprises *2000* of simulated alignment scenarios. Alignment scenario is recorded according to the procedure, where a single iteration comprises the steps of:

- Setting the required voltage value on the defocus lens.
- Using the external image processing algorithm in order to assess the defocus (smoothness) of the image acquired by a charged particle beam apparatus.
- Storing the pair of values *(lens voltage, image defocus)* that corresponds to the actual iteration of the recording procedure.
- Increasing the voltage on the defocus lens by the constant or by an adaptive-size margin.

Record of a single alignment scenario corresponds to a row in a *csv* file that is used by the simulator engine in order to provide the values of the image defocus in response to the alignment actions performed by the agent.

An architecture of the agent is determined by the reinforcement learning algorithm. Actor-critic algorithms assume the particular architecture, where the agent comprises at least two (*policy* and the *value function*) neural networks. The defocus lens alignment agent is trained with Soft Actor Critic - sample efficient off-policy algorithm.

Agent operates in a partially observable environment, where the inference of the correct alignment action based on a single observation is not possible. The agent maintains the history of observations. *Policy* and the *critic* neural networks are implemented as *Gaited Transformers* [*10*]. *Policy* network receives the history of observations as the input and provides the alignment action - real value in range (*-1; 1*). Scaled value corresponds to the change of the voltage on the defocus lens. *Q-critics* receive the history of timesteps, where each timestep is a pair (*observation, alignment action*), and provide the estimation of future rewards. *V-critic* estimates the future rewards based on the provided history of observations.

Training procedure in a simulated defocus lens environment comprises the steps of:

- Adjusting the alignment complexity level by the *trainer* algorithm.
- Providing the simulated apparatus to the misaligned state according to the alignment complexity level.
- Executing the alignment procedure by the defocus lens alignment agent.
- Computing the quality parameters (rewards), by the *trainer* algorithm, related to the alignment trajectory.
- Storing the data related to the performed alignment procedure to the replay buffer.
- Sampling the training data from the replay buffer.
- Performing the off-policy update of the agent neural networks.

Quality parameters (rewards) are computed by the *trainer* algorithm according to the following rules:

- Zero reward corresponds to any action that sets the voltage on the defocus lens that belongs to the interval of allowed voltage values.
- Negative reward corresponds to the action that sets the voltage on the defocus lens that exceeds the boundaries of the interval of allowed voltage values.
- At the final alignment timestep, the agent receives the reward for quality of the performed alignment procedure. Reward is proportional to the difference between the image defocus values before the alignment and after the alignment procedure performed by the agent.

In most alignment scenarios the agent receives the reward only once - at the final timestep. Negative rewards are provided at the initial phase of training in order to encourage the agent to operate within the interval of meaningful voltage values.

Critics (*V* and *Q* neural networks) are used only at the training time, the alignment procedure performed by the trained agent involves only the *policy* network. Thus, at the training time the *critic* neural networks are allowed to receive the *secret* information that is not available to the *policy* network. The *secret* information provided to the *critics* during the training is the difference between the optimal voltage value and the voltage value that is on the defocus lens at the actual alignment timestep.

5. Setup of the training procedure

Training in a defocus lens alignment simulator requires the installation of the following packages:

- tensorflow-gpu==1.15 [*11*]
- gym [*12*]
- stable-baselines [*13*]

6. Evaluation

6.1. Training performance

Following charts report the results of two subsequent training procedures, where the agent performs the defocus lens alignment in a simulated environment.

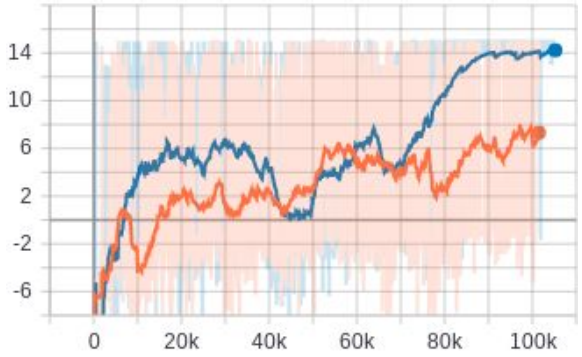

Figure 6.1.1. Training performance of the agent. Single point corresponds to the average reward received by the agent over the last *100* training episodes.

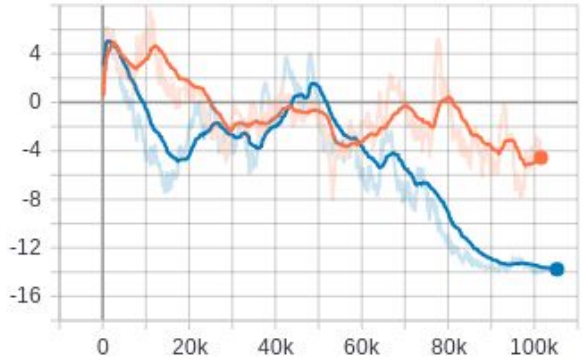

Figure 6.1.2. Training loss of *policy* neural network.

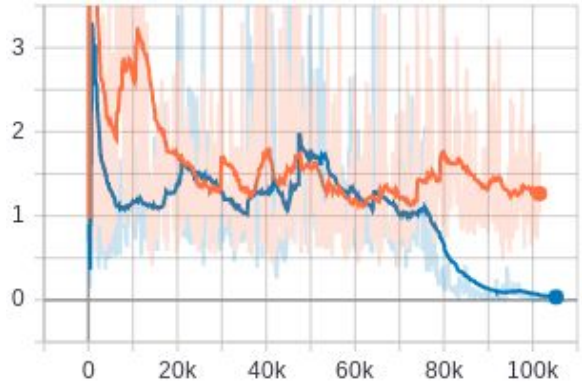

Figure 6.1.3. Training loss of *Q1-critic* network.

According to *figure 6.1.1,* the second training procedure (*blue curve*) leads to significantly better final performance of the agent than that achieved in the first training procedure (*orange curve*). Setup of two procedures is identical - the same architecture of the agent, same learning algorithm and the same amount of training iterations. Subsequent analysis aims to establish the factors underlying the difference in the agent's performance along the course of training.

Figures 6.1.2 - 6.1.5 report the training loss of the *policy*, *Q1-critic, Q2-critic* and *V-critic* neural networks.

Interval *0 - 2.5K* corresponds to the period when the *policy* loss is positive and increases. The reason is the following: at the initial phase of training, the agent receives mostly the negative rewards, *critic* neural networks are trained to predict the negative values. Training of the *critic* networks, however, requires several thousands of network updates - the increase of the *policy* loss, thus, is not immediate. Negative values predicted by the *critic* networks correspond to positive loss of the *policy* network.

Loss of *Q1*, *Q2* and *V-critic* networks decreases rapidly on interval *0 - 7K*. At timestep *2.5K*, critics are able to identify the deficient actions that lead to immediate negative response - i.g. repeated attempts to exceed the boundaries that determine the range of meaningful voltage values. Avoidance (minimization of probability) of such deficient actions leads to the decrease of the *policy* loss (*figure 6.1.2, interval 2.5 - 7K*) and to increase of the performance of the agent (*figure 6.1.1*).

Interval *0 - 7K* reports almost identical dynamics of two training procedures. At timestep *7K*, the agent continues to improve the performance in the second training procedure (*figure 6.1.1, blue curve*). In the first training procedure (*figure 6.1.1, orange curve*), however, performance of the agent begins to degrade - the amount of rewards decreases and the *policy* loss begins to increase.

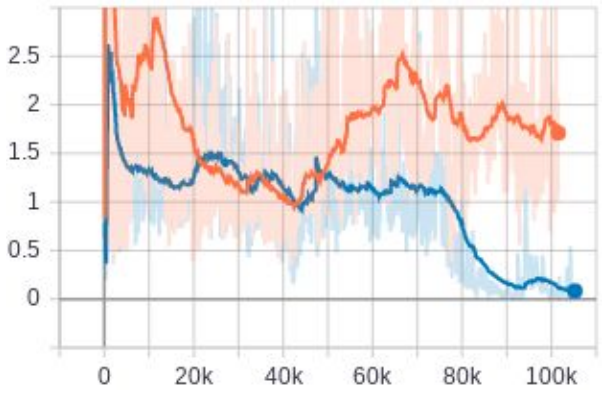

Figure 6.1.4. Training loss of *Q2-critic* network.

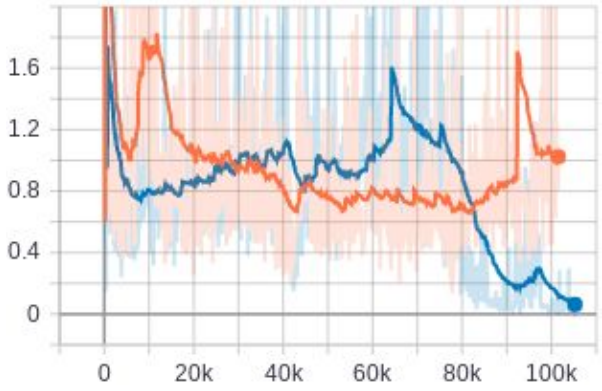

Figure 6.1.5. Training loss of *V critic* network.

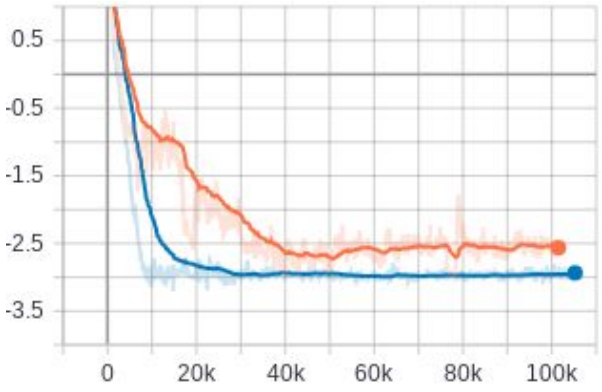

Figure 6.1.6. Policy entropy.

The main reason for such difference is the following: the order, in which the agent acquires the knowledge about the dynamics of the environment, is different in two training procedures.

Defocus lens is provided to the initial state randomly within the alignment complexity level that is gradually increased along the course of training. Interval *7 - 11K* (*figure 6.1.3 - 5, orange curve*) indicates that the agent experiences the states in which the *critics* are not able to correctly assess the alignment actions. *Critics* are retrained to provide the correct assessment of deficient *policy* actions. Quality of the action assessment gradually improves, the *policy* loss, therefore, increases, the *policy* network is updated in such a way, that the deficient actions become less probable. As a result, at timestep *11K*, the performance of the agent (*figure 6.1.1, orange curve*) begins to improve, the *Q1-critic* loss (*figure 6.1.3, orange curve*) begins to decrease.

Interval *11 - 29K* reports the increase of the agent's performance in the first training procedure (*figure 6.1.1, orange curve*), the *policy* loss, the *policy* entropy and the loss of the *critic* networks continuously decrease. Agent experiences the states that match the agent's understanding about the dynamics of the environment. Agent improves the learned alignment strategy. The amount of exploration decreases (*figure 6.1.6 - 7, orange curve*).

Interval *29 - 31K* corresponds to the decrease of performance in the first training procedure (*figure 6.1.1, orange curve*), *figure 6.1.3* reports the rapid increase of the *Q1-critic* loss meaning that *Q1-critic* is not able to correctly estimate the decreasing amount of provided rewards. The increasing amount of rewards on the interval *27 - 29K* (*figure 6.1.1, orange curve*), however, is estimated precisely - *Q1-critic* loss is relatively low (*figure 6.1.3, orange curve*). At timestep *29K*, thus, assessment of the *policy* actions is optimistic. *Q1-critic* predicts a higher amount of rewards than that provided by the environment in response to the actions performed by the agent. In order to minimize the overestimation of provided rewards, the *minimum* of two values, predicted by *Q critics*, is used in computation of the loss of *policy* network.

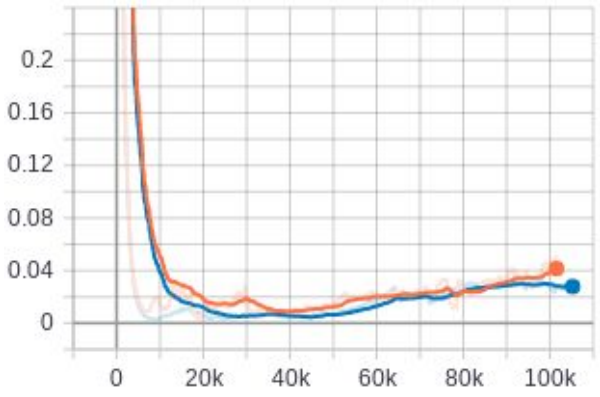

Figure 6.1.7. Entropy term coefficient.

In the second training procedure (*figure 6.1.1, interval 11 - 43K, blue curve*), the performance of the agent increases, remains constant and then decreases. Performance decrease (*figure1, interval 35 - 43K, blue curve*) corresponds to a period when the agent visits the novel states of environment. Loss of *V* and *Q-critic* networks (*figure 6.1.3 - 5, blue curve*) increases and then decreases - *critics* are trained to precisely estimate the amount of rewards provided to the agent in the novel states. *Critics* are able to identify the deficient actions performed by the agent, the *policy* loss (*figure 6.1.2, interval 35 - 43K, blue curve*), therefore, increases.

At timestep *43K*, the performance of the agent (*figure 6.1.1*) is equal in both training procedures. The agent has visited the different states of the environment, has acquired the different knowledge and has evolved the different alignment policies whose average performance, however, is equal.

The training procedure on the interval *43 - 70K* is stable (*figure 6.1.1*): the agent adjusts the *policy* and the *critic* networks. From the time step *70K*, however, the dynamics of two training procedures are significantly different.

Interval *70 - 90K* reports the rapid improvement of the agent's performance in the second training procedure (*figure 6.1.1, blue curve*). At timestep *90K*, the alignment *policy* is almost optimal. On the interval *90 - 100K*, the agent achieves the maximal possible amount of rewards per training episode, meaning that the alignment can be reliably performed from any initial state of the defocus lens.

Interval *70 - 100K*, however, corresponds to the noisy improvement of the agent's performance in the second training procedure (*figure 6.1.1, orange curve*). The *policy* entropy stays relatively constant (*figure 6.1.6, orange curve*), the *policy* entropy coefficient (*figure 6.1.7, orange curve*), however, increases, meaning that the agent increases the amount of exploration in order to acquire the more profitable alignment policies. At timestep *91K*, the agent explores novel states that are yet inconsistent with the *V-critic's* model of the environment. *Figure 6.1.5* reports the rapid increase in the *V-critic* loss (*orange curve, timestep 91K*). At timestep *100K*, the agent continues to explore.

The dynamics of both training procedures obeys the following rules:

- The loss of *V* and *Q-critics* increases as the agent experiences the novel states.
- The *policy* loss may remain constant for some time, since the *critics* do not provide the correct assessment of the actions performed by the agent in the novel states.
- Adaptation of the *critic* networks to the novel states requires the number of retraining iterations that depends on the size of the replay buffer.
- The *policy* loss increases gradually, according to the speed of adaptation of the *critic* networks.
- Order, in which the agent acquires the knowledge about the dynamics of the environment significantly influences the learning progress of the agent.

6.2. Assessment of the quality of the beam focusing procedure

Method of beam focusing quality assessment comprises the steps of:

- Providing of Zn-Au specimen to the chamber of the dual-beam microscope.
- Executing the alignment procedure of the defocus lens.
- Acquiring the image whose quality depends on the quality of the performed alignment procedure.
- Providing the acquired image to the *image processing algorithm* in order to assess the image quality.
- Comparing the result of image quality assessment with the threshold value of the image quality determined by the specification of the microscope.

Alignment procedure of the *defocus lens* may be performed by an automatic alignment algorithm that undergoes the testing procedure or may be performed by a human expert, in particular, to identify the possible defects of the microscope hardware.

Image quality assessment is performed by the *image processing algorithm* that detects the edges on the provided image and computes the average width of the detected edges. Low value of the average width of edges corresponds to the sharp image, high value corresponds to the smooth image acquired by the suboptimally focused beam. Threshold value of the image quality determines the binary outcome (success/failure) of the alignment procedure.

6.3. Evaluation of the defocus lens alignment agent

The agent, pre-trained in the defocus lens alignment simulator, continues the training procedure on the dual-beam microscope. Pre-training phase comprises *500K - 3M* of training iterations, adaptation phase comprises *10 - 250K* of interactions with a microscope.

Evaluation session comprises *100* alignment scenarios. The agent, experienced *1M* of training iterations in simulation and *50K* of adaptation steps on the microscope, is able to pass the evaluation session - all *100* alignment scenarios are completed successfully.

Since the dynamics of each microscope is unique, the training of the agent on a single microscope is not sufficient. In practice, the agent is trained on multiple microscopes in parallel according to the method described in section *3*.

7. References

1. Sergey Levine, Aviral Kumar, George Tucker, Justin Fu., Offline Reinforcement Learning: Tutorial, Review, and Perspectives on Open Problems., arXiv:2005.01643

2. Kate Rakelly, Aurick Zhou, Deirdre Quillen, Chelsea Finn, Sergey Levine., Efficient Off-Policy Meta-Reinforcement Learning via Probabilistic Context Variables., arXiv:1903.08254

3. Laura Smith, Nikita Dhawan, Marvin Zhang, Pieter Abbeel, Sergey Levine., AVID: Learning Multi-Stage Tasks via Pixel-Level Translation of Human Videos., arXiv:1912.04443

4. Xue Bin Peng, Erwin Coumans, Tingnan Zhang, Tsang-Wei Edward Lee, Jie Tan , Sergey Levine., Learning Agile Robotic Locomotion Skills by Imitating Animals., arXiv:2004.00784

5. OpenAI et al., Solving Rubik's Cube With a Robot Hand., arXiv:1910.07113

6. OpenAI et al., Dota 2 with Large Scale Deep Reinforcement Learning., arXiv:1912.06680

7. Oriol Vinyals et al., Grandmaster Level in StarCraft II Using Multi-Agent Reinforcement Learning., Nature:s41586-019-1724-z

8. Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, Sergey Levine., Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor., arXiv:1801.01290

9. John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, Oleg Klimov., Proximal Policy Optimization Algorithms., arXiv:1707.06347

10. Emilio Parisotto, H. Francis Song, Jack W. Rae, Razvan Pascanu, Caglar Gulcehre, Siddhant M. Jayakumar, Max Jaderberg, Raphael Lopez Kaufman, Aidan Clark, Seb Noury, Matthew M. Botvinick, Nicolas Heess, Raia Hadsell., Stabilizing Transformers For Reinforcement Learning, arXiv:1910.06764

11. Tensorflow., Installation steps., https://www.tensorflow.org/install/pip

12. OpenAI Gym., Installation steps., https://gym.openai.com/docs/

13. Stable Baselines., Installation steps., https://stable-baselines.readthedocs.io