Internship report
06.12.2021 – 25.02.2022

Electron Beam Parameter Estimation using Neural Networks and First Steps toward State Estimation for a Particle Accelerator

Submitted by

Jannik Jorge Grothusen

52576
General Engineering Science
Hamburg University of Technology

Under the guidance of

Dr. Annika Eichler
Project Leader

Dr. Oliver Stein
Postdoctoral Researcher

Deutsches Elektronen-Synchrotron DESY

Group MSK (Maschine Beam Control)

Table of contents

1 Introduction

1.1 Institution Portrait

The Deutsches Elektronen-Synchrotron commonly referred to by the abbreviation DESY, is a national research center in Germany that operates particle accelerators used to investigate the structure of matter. It conducts a broad spectrum of inter-disciplinary scientific research in three main areas: particle and high energy physics; photon science; and the development, construction, and operation of particle accelerators. Its name refers to its first project, an electron synchrotron. DESY is publicly financed by the Federal Republic of Germany, the States of Germany, and the German Research Foundation (DFG). DESY is a member of the Helmholtz Association and operates at sites in Hamburg and Zeuthen [1].

DESY is organized in different groups. This internship was done on a project of the MSK group. The group is responsible for the development and operation of systems for the damping of beam vibrations, for the control of parameters such as magnetic current and high frequency, as well as for the synchronization and timing of pulsed components. In addition, systems for beam diagnostics are maintained.

The Helmholtz AI project *Machine Learning toward Autonomous Accelerators* is a collaboration between DESY and Kalsruhe Institute of Technology (KIT) that works on investigating and developing reinforcement learning (RL) applications for the automatic start-up of electron linear accelerators. The work is carried out in parallel at two similar research accelerators: ARES at DESY and FLUTE at KIT, giving the unique opportunity of transfer learning between facilities.

ARES (Accelerator Research Experiment at SINBAD) is an S-band radio frequency linear accelerator (linac) at the DESY Hamburg site equipped with a photoinjector and two independently driven traveling wave accelerating structures [2]. The main research focus is the generation and characterization of sub-femtosecond electron bunches at relativistic particle energy. The generation of short electron bunches is of high interest for radiation generation, i.e., by free electron lasers. The limits of particle beam diagnostic technologies and novel electron acceleration methods will also be investigated at the site.

1.2 Project Overview

Modern particle accelerators provide beams for new discoveries in science. The required flexibility, number of operation modes, and better performance in simultaneously more compact and more energy-efficient accelerators demand advanced control methods. One major challenge is the start-up of such accelerators, which requires frequent manual intervention. Low repetition rates, often only one acceleration event per second, lead to slow optimization rates, thus demanding expert knowledge. Although a complete autonomous accelerator seems far from being reachable, the project takes the first steps by bringing reinforcement learning to accelerator operation. Reinforcement learning yields a policy for every initial state taking the impact of the current action on the future into account, eventually replacing the need for manual intervention [3].

Focusing and centering the electron beam on diagnostic screens along the beam line are frequently performed and time-consuming tasks during start-up and working point changes. The project considers focusing and centering the beam using a quadrupole triplet and corrector magnets in the ARES experimental area (EA) as a particular instance of this type of task.

The current goal of the project is to enable an RL agent to focus and center the beam in just a few iterations and thereby significantly reducing the time required for optimizing the beam parameters. Such an RL agent observes the beam parameters on the screen and current magnet settings in order to propose a set of actions for changing the magnets' settings, improving the beam's focusing and centering. A flow chart of this setup is shown in Fig 1.

For the RL Agent to perform a useful action on the machine, the observation of the beam parameters must be reliable and accurate. The development and analysis of a new method for this was a part of this internship. Another part was the design of a state estimator for unknown system states, such as quadrupole magnet misalignments and parameters of the incoming beam.

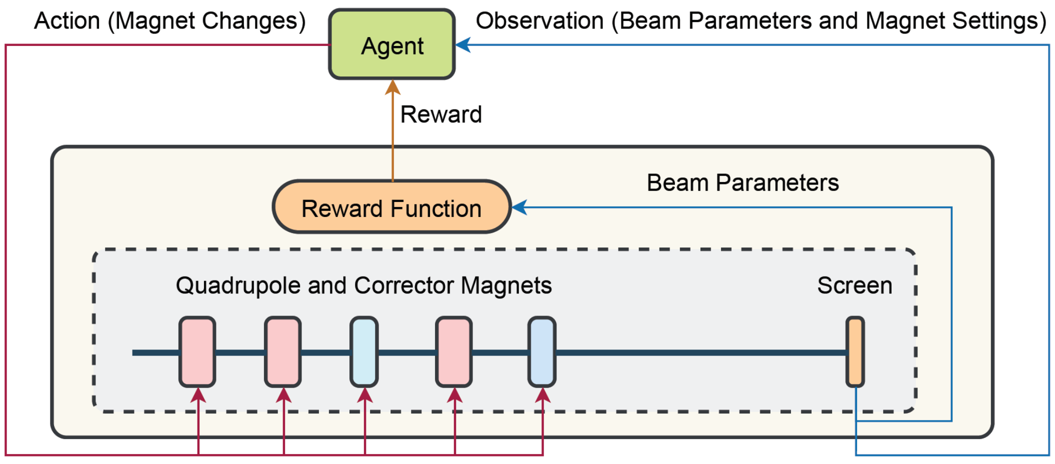
Action (Magnet Changes)
Agent
Observation (Beam Parameters and Magnet Settings)
Reward
Beam Parameters
Reward Function
Quadrupole and Corrector Magnets
Screen

Figure 1: RL environment for beam optimization in the ARES EA [3].

2 Beam Parameter Estimation

The main problem currently addressed in the project is the optimization of the beam parameters. Optimization methods such as the RL agent require accurate and reliable observations of the environment they are trying to optimize. During the internship, a method for detecting the beam parameters using the supervised machine learning approach was developed and tested. With supervised learning the computer is presented with example inputs and their desired outputs and the goal is to learn a general rule that maps inputs to outputs.

2.1 Fundamentals: Particle Beam Dimensions

Particle beams are characterized by a set of quantifying parameters being either constants of motion or functions varying from point to point along a beam transport line [4]. Such parameters are for example the beam's energy, time structure, current, and dimensions. In this report, only the beam's dimensions are considered and the terms parameter and dimensions are used interchangeable.

Electron bunch dimensions can be represented as Gaussian charge distributions. Four parameters are required to determine the beam size in each plane. In the plane that is perpendicular to the beam's direction these transverse dimensions are μ_x, μ_y, σ_x and σ_y. The parameter μ is the mean of the distribution and the parameter σ is its standard deviation.

These beam dimensions are relevant when the beam hits the experiment. To identify the parameters, the beam is made visible on a fluorescent screen which then gets captured by a camera. An exemplary image is shown in Fig. 2.1. The images are taken with a resolution of 2464 by 2056 pixels, while the absolute screen size is 8.18 by 5.03 mm. The images are processed with a binning of four to reduce memory size and increase processing speed. It can be observed that the real beam does not correspond to an ideal Gaussian distribution but can be well approximated by it.

2.2 Approach: Neural Networks

To identify the four transversal beam parameters from the camera images, artificial neural networks (ANNs) were chosen as the novel solution approach. ANNs are comprised of node layers containing an input layer, one or more hidden layers, and an output layer. Each node, or artificial neuron, connects to another and has an associated weight and threshold. If the output of any individual node is above the specified threshold value, that node is activated, sending data to the next layer of the network. Otherwise, no data is passed along to the next layer of the network.

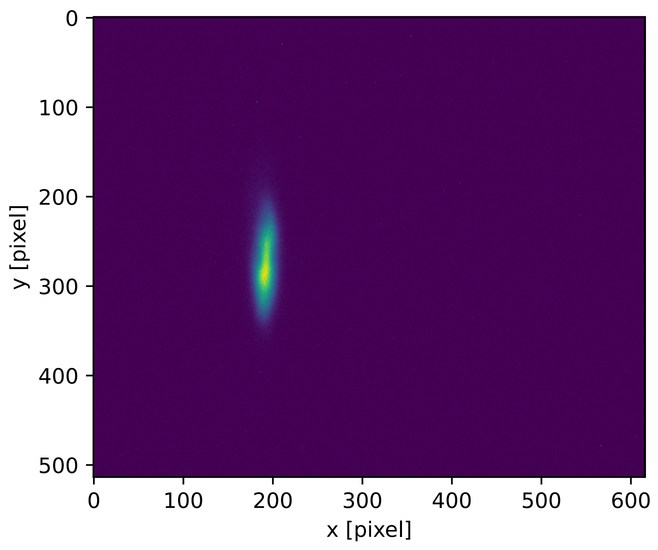

Figure 2.1: Sample image of the transverse dimensions of an electron beam.

Neural networks can be classified into different types, which are used for different purposes: feedforward neural networks, convolutional neural networks (CNNs) and recurrent neural networks (RNNs) [5]. The first two were used in this internship. Feedforward neural networks are comprised of an input layer, a hidden layer or layers, and an output layer. CNNs are similar to feedforward networks, but they're usually utilized for image recognition, pattern recognition, and/or computer vision. Due to their suitability for image recognition, the focus was placed on CNNs. For further information please refer to [6].

2.3 Implementation

Neural networks rely on training data to learn and improve their accuracy over time. For this purpose, large data sets with up to 100.000 beam images and their associated parameters were generated. The beam was assumed to have an ideal Gaussian distribution. During the training 80% of the images were used for model training and 20% for model evaluation.

In the developed pipeline, pre-processing steps such as resizing and normalization were included. Different optimizers were tested during training. The Adam optimization algorithm [7] gave the best results. For loss functions mean squared error (MSE) and mean absolute error (MAE) were tested. In the end the MAE function was chosen due to its intuitive representation of the error.

All implementations were written in the high-level general-purpose programming language Python. The library Keras [8] was used to implement, train, and evaluate the neural networks. It is the high-level API for the open-source machine learning platform TensorFlow 2 [9]. Keras offers an approachable, highly productive interface for solving machine learning problems, with a focus on modern deep learning.

CNNs are commonly made up of three layer types: convolutional layers, pooling layers, and fully connected layers. All of these layers must be defined by multiple parameters. A convolutional layer for example has a specified number of filters, their spatial extend, a stride and a zero-padding value. These values are called hyperparameters. Again, for more information please refer to [6]. Various known and self-designed network architectures were tested during the internship.

Since there are a lot of values used to control the learning process of neural networks there is a need to attempt them in a structured manner. This so called hyperparameter optimization was be done using Weights & Biases (W&B) - a platform for experiment tracking, dataset versioning and model management [10]. An exemplary hyperparameter analysis can be seen in Fig. 2.2. It shows the achieved MAE of the neural network in connection to its hyperparameters.

Training complex neural networks, especially CNNs, with large datasets is very computationally expensive and time-consuming. Therefore, the Maxwell High Performance Computing (HPC) Cluster of DESY was used for training. Its components are fast dedicated storage, a fast, low-latency network, and 789 high-memory compute nodes with substantial number of GPUs. Its theoretical peak performance is 2790 TFlops [11].

2.4 Results

The metric currently used to detect the beam parameters is called full width at half maximum (FWHM). It is the difference between the two values of the independent variable at which the dependent variable is equal to half of its maximum value. This value is determined for the pixel values in both dimensions of an image and from it the mean values and standard deviations of the charge distribution are calculated. This works very accurately and reliably.

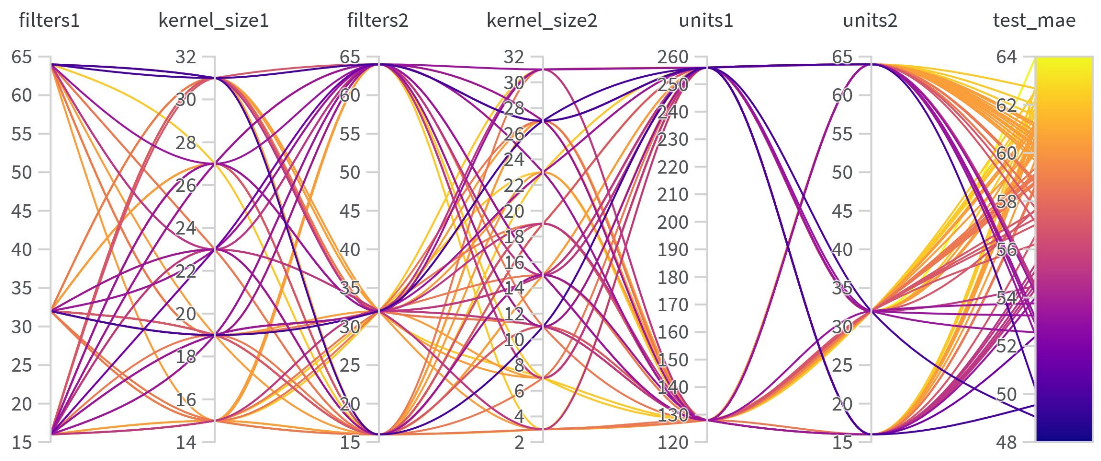

Figure 2.2: W&B hyperparameter sweep visualized with a parallel coordinates plot.

Figure 2.3 shows the most accurate models of each type after training. The dashed line represents the desired accuracy. FWHM is the currently used algorithm. It is the most accurate method and achieves the desired precision. NN-3 is a simple feedforward neural network which surprisingly performs better than the more complex CNN-8 which was the most accurate CNN. The VGG-11 architecture is part of the VGGNet architecture family. VGGNet is one of the most popular image recognition architectures and surpasses baselines on many tasks and datasets like ImageNet. However, these architectures obviously work poorly for the problem at hand.

Figure 2.4 shows the performance of neural networks on real beam images. The results of the FWHM method were used here as reference values. Again, it can be observed that the models do not achieve the desired accuracy.

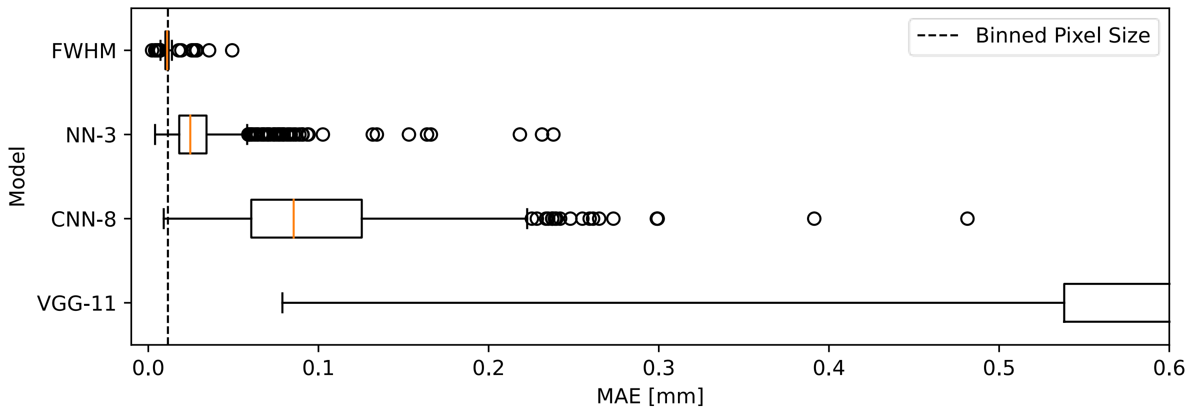

Figure 2.3: Comparison of different methods on generated images by MAE.

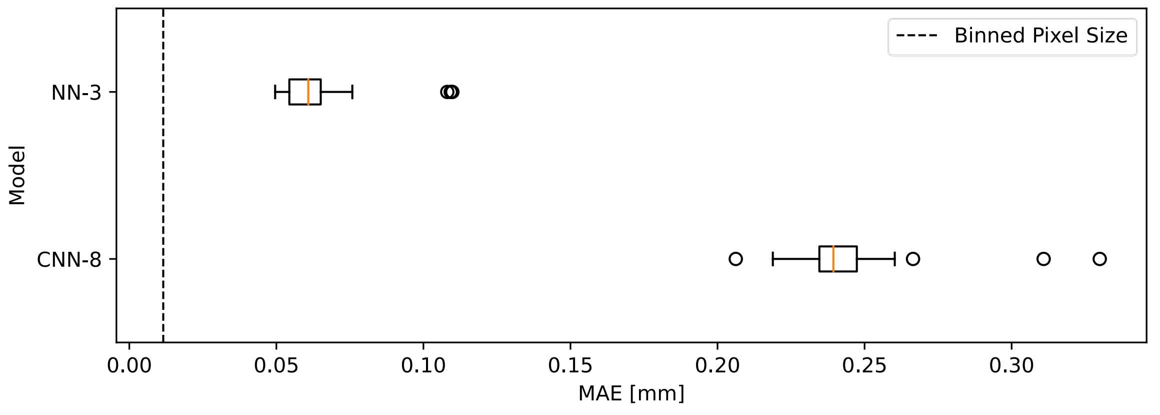

Figure 2.4: Comparison of different methods on real images by MAE.

3 Machine State Estimation

Other optimization methods project besides the RL approach are tested in the DESY project. Advanced control technologies like adaptive dynamic programming (ADP) and model predictive control (MPC) rely on mathematical models to find optimal control actions for the system. For these models to work, information about the internal system state must be available. With the discussed control problem this state includes information like additional beam parameters for the incoming and outgoing beam and misalignments of the control magnets of the ARES EA. Not all of these system parameters can be determined by direct observation and for that reason a state estimator was developed during the internship.

3.1 Fundamentals: State Estimation

In this subchapter state estimation is explained using the Kalman filter theory. The main goal of state estimators is to estimate a state based on a system model and measurements of the system process. The Kalman filter is based on a linear dynamic system model discretized in the time domain. The algorithm works by a two-phase process. For the prediction phase, the Kalman filter produces estimates of the current state variables, along with their uncertainties. Once the outcome of the next measurement is observed, these estimates are updated using a weighted average, with more weight being given to estimates with greater certainty. The algorithm is recursive. It can operate in real time, using only the present input measurements and the state calculated previously and its uncertainty matrix; no additional past information is required. For further information please refer to [12].

3.2 Approach: Unscented Kalman Filter

Simple Kalman filters work for linear systems. Looking at the treated system model, one sees that it is nonlinear. The model of a beam transformation through quadrupole and corrector magnets is not particularly complex: the beam is described by a six-dimensional vector with three expected values of the beam distribution and its derivative with respect to the beam direction in the three spatial directions. In addition, a six by six covariance matrix describing the beam distribution along the three spatial directions is defined. The effects that the control magnets have on the beam parameters are described by matrix multiplication of the lattice components. All transfer maps are linear except for the ones of the quadrupole magnets.

Due to the nonlinearity, a simple Kalman filter is not sufficient as a state estimator. An extended Kalman filter (EKF) or an unscented Kalman filter (UKF) can be used. An EKF linearizes the process model at the point of observation and with that a linearization error occurs. This linearization error is reduced in a UKF by using carefully chosen sample points – so called sigma points. Compared to an EKF, a UKF does not need a linearized model in the form of a Jacobi matrix. This would be difficult to derive for the given system anyway. For these reasons the usage of an UKF is chosen.

3.3 Implementation

First, a dedicated simulation was written for validation of the used model and for generation of the observations. A typical process as performed by an RL agent serves as the basis of the analysis. Its actions (the control inputs for the magnets) and the four measurable transverse beam parameters on the screen are passed to the state estimator.

Again, the implementation was written in Python. The UKF was implemented using the FilterPy library presented in [12]. The state was chosen as a vector with 22 entries. Eight parameters for the incoming and outgoing beam, respectively, and two-dimensional misalignments of the three quadrupole magnets. The longitudinal beam parameters are not of particular interest and misalignments of the corrector magnets can be neglected, since they have no effect on the beam due to their homogeneous magnetic field.

3.4 Results

Most parameters of the UKF are selected heuristically. The initial state and its covariance matrix can include whether information about the state is already present and how certain this information is. The result in Fig. 3.1 was generated by assuming that the eight parameters of the input beam are known with high certainty. It can be observed that the magnet misalignments have been initialized to zero and are successfully estimated by the filter after only three steps.

With less accurate initial states, the filter estimate also converges but not to the correct values. It seems that these values are shifted by an offset. This has to be proven. On real measured data, the filter estimate also converges to values. This can be observed in Fig. 3.2. However, these values were not tested against any reference, since the true values can not be measured.

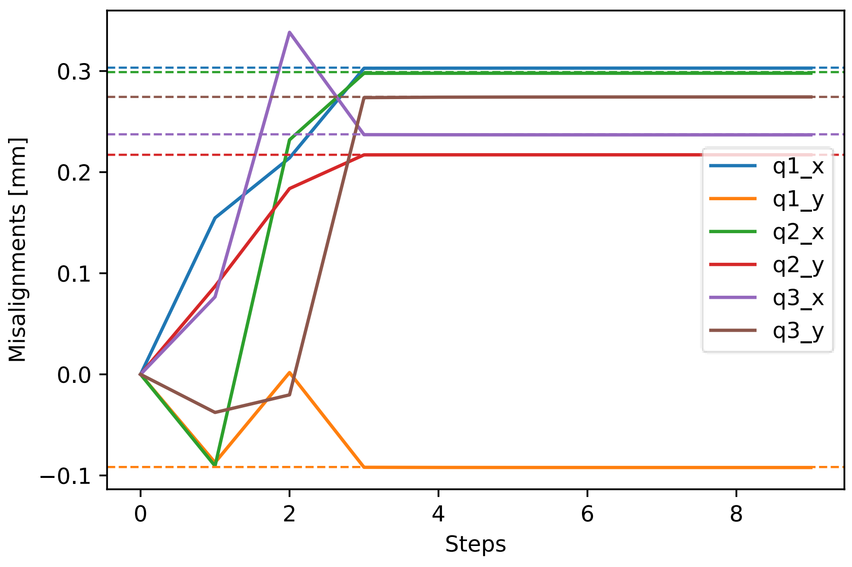

Figure 3.1: Estimated magnet misalignments (solid) using simulated measurements.

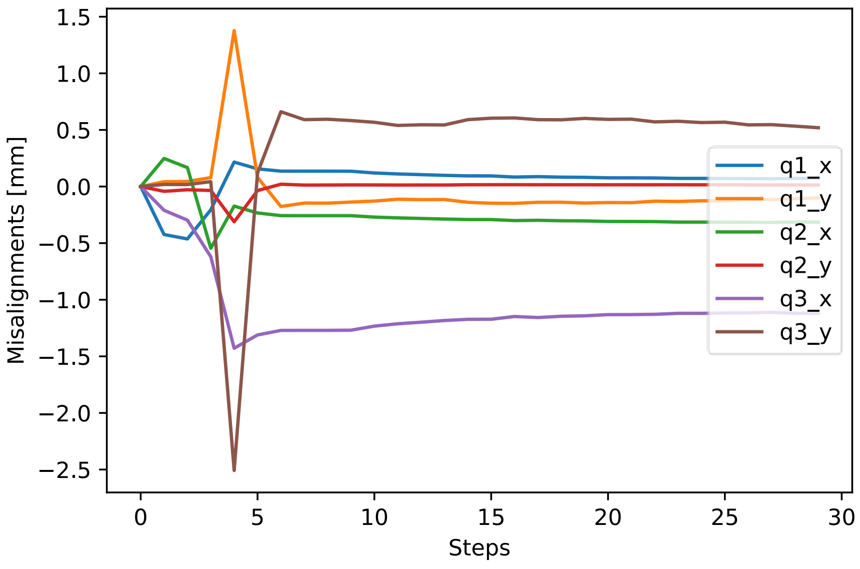

Figure 3.2: Estimated magnet misalignments using real measurements.

4 Conclusion

4.1 Summary

In the course of the internship, two contributions were made to the research project *Machine Learning toward Autonomous Accelerators.* One was the examination of an alternative method for beam parameter detection using machine learning - more precisely neural networks. This involved the generation of large data sets for training and the investigation of different network architectures. The models examined did not achieve the desired accuracy and were therefore less accurate than the currently used method for parameter detection.

The second contribution was the development and partial analysis of a state estimator for unknown accelerator state information. This included the familiarization with a particle accelerator system model, the design of a suitable system state, and the implementation of an unscented Kalman filter. The results collected in this subproject looked promising and provide incentive to further investigate and pursue the development of the state estimator.

Besides dealing with the two topics mentioned above, I participated in group meetings of the MSK group and accompanied measurements on the ARES accelerator. On this occasion I spent time in the main control room of DESY and got an introduction on how to use the control interface of the ARES accelerator.

Besides the subject specific knowledge I gained in the field of state estimation and neural networks, I learned how to use version control systems like Git to integrate into the software structure of a research project, how to write useful documentation and how to use a high performance computing cluster for complex computational tasks.

4.2 Outlook

The developed neural networks for beam parameter estimation will likely not be applied in the research project due to their insufficient performance. The state estimator on the other hand provides a basis for advanced control and optimization algorithms and should therefore be further developed and analyzed.

Bibliography

[1] Wikipedia, “DESY,” Dec. 26, 2021. [Online] Available at: https://en.wikipedia.org/wiki/DESY [Accessed Feb. 24, 2022].

[2] E. Panofski et al., “Status Report of the SINBAD-ARES RF Photoinjector and LINAC Commissioning,” in Proceedings of the 10th International Particle Accelerator Conference (IPAC), p. 5, JACoW Publishing, 2019.

[3] A. Eichler et. al., “First Steps toward an Autonomous Accelerator, a Common Project Between DESY and KIT,” 2021.

[4] H. Wiedemann, “Particle Accelerator Physics,” Fourth Edition, Springer Nature, 2015.

[5] IBM Cloud Education, “Neural Networks,” Aug 17, 2020. [Online] Available at: https://www.ibm.com/cloud/learn/neural-networks [Accessed Feb. 27, 2022].

[6] F.-F. Li et. al., “CS231n: Convolutional Neural Networks for Visual Recognition,” Mar 30, 2021. [Online] Available at: https://cs231n.github.io/ [Accessed Feb. 27, 2022].

[7] D. P. Kingma and J. Ba, “Adam: A Method for Stochastic Optimization,” in Proceedings of the 3rd International Conference for Learning Representations, San Diego, 2015.

[8] F. Chollet et. al., “Keras,” 2015. [Online] Available at: https://keras.io.

[9] M. Abadi et. al., “TensorFlow: Large-scale machine learning on heterogeneous systems,” 2015. [Online] Available at: https://tensorflow.org.

[10] L. Biewald, “Experiment Tracking with Weights and Biases,” 2020. [Online] Available at: https://wandb.com.

[11] DESY, “Maxwell Cluster,” Jan 19, 2022. [Online] Available at: https://confluence.desy.de/display/MXW/Maxwell+Cluster [Accessed Feb. 27, 2022].

[12] R. R. Labbe, “Kalman and Bayesian Filters in Python,” Oct. 14, 2020. [Online] Available at: https://github.com/rlabbe/Kalman-and-Bayesian-Filters-in-Python [Accessed Feb. 27, 2022]

Report Approval

I hereby declare that I have written this internship report independently, using only the resources cited.

[illegible]

Jannik Jorge Grothusen

I hereby confirm the correctness of this internship report.

[illegible]

Dr. Annika Eichler

O. Stein

Dr. Oliver Stein