Implementation of Artificial Intelligence at PK-4 for the Analysis of Complex Plasmas

Max Klein

Giessen 2026

DISSERTATION

Implementation of Artificial Intelligence at PK-4 for the Analysis of Complex Plasmas

Implementierung von künstlicher Intelligenz an PK-4 zur Analyse komplexer Plasmen

Max Klein

A cumulative dissertation submitted
in fulfillment of the requirements for
the degree of Dr. rer. nat.

Research conducted at
I. Institute of Physics, Justus-Liebig-University, Giessen
and
Department of Electrical and Information Engineering, University of Applied Sciences, Giessen

JUSTUS-LIEBIG-
UNIVERSITÄT
GIESSEN

Giessen, 20.03.2026

Implementation of Artificial Intelligence at PK-4 for the Analysis of Complex Plasmas
Implementierung von künstlicher Intelligenz an PK-4 zur Analyse komplexer Plasmen

Genehmigte kumulative Dissertation von Max Klein, M.Sc.
Fachbereich 07: Mathematik und Informatik, Physik, Geographie

Justus-Liebig-Universität Gießen
I. Physikalisches Institut
Heinrich-Buff-Ring 16
35392 Gießen

Eingereicht am 20.03.2026

Erstgutachter: Prof. Dr. Markus H. Thoma
Zweitgutachter: Prof. Dr. Mike Schwarz

Abstract

Complex plasmas serve as a distinctive experimental framework for particle-resolved investigations of strongly coupled many-particle systems under controlled laboratory and microgravity conditions. However, experiments such as the Plasmakristallexperiment-4 generate large image datasets whose analysis is complicated by noise, limited observation times, evolving experimental conditions and inherent ambiguities between ordered and disordered states. Furthermore, the necessity of human intervention during the experiments limits repeatability and is a source of error. This dissertation examines how machine learning methods can be systematically integrated into both the analysis and operation of complex plasma experiments in a physically consistent and computationally efficient manner. The core of this cumulative dissertation comprises three publications that address complementary aspects of particle-resolved data analysis. A convolutional encoder–decoder network is employed to identify string-like particle structures in two-dimensional camera images, with network architecture and training data generation guided by the underlying electrorheological physics. An unsupervised self-organizing map is applied and optimized for robust particle tracking between consecutive frames, enabling reliable trajectory reconstruction in dense and high-velocity particle flows without manual threshold tuning. To further improve data quality, an outlier detection procedure is developed. Based on the particle trajectories, turbulence in complex plasmas is formulated as a time-resolved classification problem and investigated using a long short-term memory network operating on physically motivated, invariant trajectory features. Beyond the individual publications, this work demonstrates the practical deployment of machine learning methods directly at the experiment. The developed models are adapted to embedded hardware through compact network architectures and mixed-precision inference, enabling data analysis with minimal latency implemented into the experiment. In addition, molecular dynamics simulations are analyzed using the machine learning framework, revealing systematic dependencies of turbulent particle motion on simulation parameters. Overall, this dissertation establishes a physically motivated and extensible machine learning framework that advances automated data analysis and diagnostics and supports future adaptive control strategies in complex plasma experiments.

Zusammenfassung

Komplexe Plasmen bieten einen einzigartigen experimentellen Rahmen für partikelaufgelöste Untersuchungen stark gekoppelter Vielteilchensysteme unter kontrollierten Labor- und Mikrogravitationsbedingungen. Experimente wie das Plasmakristallexperiment-4 erzeugen jedoch große Bilddatensätze, deren Analyse durch Rauschen, begrenzte Beobachtungszeiträume, sich verändernde Versuchsbedingungen und inhärente Unklarheiten zwischen geordneten und ungeordneten Zuständen erschwert wird. Darüber hinaus schränkt die Notwendigkeit menschlicher Eingriffe während der Experimente die Wiederholbarkeit ein und ist eine Fehlerquelle. Diese Dissertation untersucht, wie Methoden des maschinellen Lernens systematisch in die Analyse und Durchführung komplexer Plasmaexperimente integriert werden können und zwar auf physikalisch konsistente und rechnerisch effiziente Weise. Der Kern dieser kumulativen Dissertation umfasst drei Publikationen, die sich mit komplementären Aspekten der teilchenaufgelösten Datenanalyse befassen. Ein Convolutional Encoder-Decoder Netzwerk wird eingesetzt, um stringartige Teilchenstrukturen in zweidimensionalen Kamerabildern zu identifizieren, wobei die Netzwerkarchitektur und die Generierung der Trainingsdaten von der zugrunde liegenden elektrorheologischen Physik geleitet werden. Eine unsupervised Self-Organizing Map wird für eine robuste Partikelverfolgung zwischen aufeinanderfolgenden Bildern verwendet und optimiert, wodurch eine zuverlässige Trajektorienrekonstruktion in dichten und schnellen Partikelströmen ohne manuelle Schwellenwertanpassung ermöglicht wird. Um die Datenqualität weiter zu verbessern, wird ein Verfahren zur Erkennung von Ausreißern entwickelt. Basierend auf den Partikeltrajektorien wird Turbulenz in komplexen Plasmen als zeitaufgelöstes Klassifizierungsproblem formuliert und unter Verwendung eines Long Short-Term Memory Netzwerks untersucht, das auf physikalisch motivierten, invarianten Trajektorieneigenschaften basiert. Über die einzelnen Veröffentlichungen hinaus demonstriert diese Arbeit die praktische Implementierung von Methoden des maschinellen Lernens direkt im Experiment. Die entwickelten Modelle werden durch kompakte Netzwerkarchitekturen und Inferenz mit gemischter numerischer Genauigkeit an eingebettete Hardware angepasst, was eine Datenanalyse mit minimaler Latenz direkt im Experiment ermöglicht. Darüber hinaus werden Molekulardynamik-Simulationen mit dem Machine Learning Ansatz analysiert, wodurch systematische Abhängigkeiten der turbulenten Teilchenbewegung von Simulationsparametern aufgezeigt werden. Insgesamt etabliert diese Dissertation einen physikalisch motivierten und erweiterbaren Rahmen für maschinelles Lernen, der die automatisierte Datenanalyse und -diagnostik vorantreibt und zukünftige adaptive Regelungsstrategien in komplexen Plasmaexperimenten unterstützt.

Contents

List of Figures

List of Abbreviations

AC Alternating Current. 6, 21, 27

AI Artificial Intelligence. 1, 11, 27, 87

AIPEX Artificial Intelligence for Plasma Experiments. 21

BMU Best-Matching Unit. 16, 17

CAN Controller Area Network. 23, 24

CCD Charge-Coupled Device. 22, 23

CMOS Complementary Metal-Oxide-Semiconductor. 22–24

CNN Convolutional Neural Network. xi, 12, 14, 16, 19, 24, 27

DAW Dust Acoustic Wave. 1, 7, 10, 28, 29, 88

DC Direct Current. 5, 6, 21

DLR Deutsches Zentrum für Luft- und Raumfahrt. 21, 25, 103

EM Engineering Model. 21–23

ER Electrorheological. ix, xi, 1, 2, 5–7, 11, 27–29, 87, 88

FM Flight Model. 21–23

FPGA Field-Programmable Gate Array. 24

FPS Frames per Second. 22–24

HCI Human–Computer Interaction. 23, 24

ISS International Space Station. 1, 21

LAMMPS Large-scale Atomic/Molecular Massively Parallel Simulator. x, 85, 86, 88–92

LSTM Long Short-Term Memory. xi, 2, 10, 11, 18, 19, 29, 85–87

MF Melamine Formaldehyde. 5

ML Machine Learning. 1, 2, 7, 11, 13, 16, 19–21, 24, 27–30, 85–88

MLP Multilayer Perceptron. 7, 12, 14, 18

MSE Mean Squared Error. 12

OML Orbital Motion Limited. 4

PID Proportional–Integral–Derivative. 25

PK-4 Plasmakristallexperiment-4. xi, 1, 10, 11, 15, 17, 21–24, 27, 29, 30, 87, 88

PMMA Polymethyl Methacrylate. 5

PTV Particle Tracking Velocimetry. 16, 87

RNN Recurrent Neural Network. 18

SOM Self-Organizing Map. xi, 2, 11, 16–18, 20, 24, 28, 29, 87

TKE Turbulent Kinetic Energy. 10, 29

TOPS Trillions of Operations per Second. 24

WBCE Weighted Binary Cross Entropy. 12

Chapter 1

Introduction

Complex plasmas have long served as a foundation for fundamental research. Small monodisperse particles distributed in a partially ionized noble gas gather a negative electrical charge due to the fast electrons in the plasma and thus interact with each other as a consequence of the arising long-range Coulomb forces. Once illuminated, the micrometer-sized particles are visible to cameras. This strongly coupled system is used for simulating many-body systems, which would otherwise not be observable at the particle level, such as liquids and solids. Investigations of the formation, melting and recrystallization of crystals [1–4], convective flows [5], turbulence [6–11], Electrorheological (ER) properties [12–14], Dust Acoustic Waves (DAWs) [15–17] and other phenomena are feasible in complex plasmas. Therefore, various experimental setups were constructed to generate complex plasmas and enable such fundamental research. This work is focused on the experimental setup Plasmakristallexperiment-4 (PK-4) [18], which consists of an elongated Π-shaped glass plasma chamber. Unlike its predecessors, the experimental focus here is more on the analysis of plasma particle flows, which are made possible by the elongated plasma tube. Since gravitational forces act strongly on the comparatively massive micrometer-sized particles under laboratory conditions, most complex plasma experiments are conducted under microgravity conditions either on sounding rockets, during parabolic flights or on the International Space Station (ISS). During these experiments large datasets in the form of particle videos and other housekeeping data are generated. This results in challenges not only for an effective data evaluation but also for the data retrieval from the ISS, due to long waiting times and complex transfer of hard drives back to Earth. Furthermore, costly human resources on the ISS are needed for the experiment execution and the error-prone reliance on human intervention in fast experiments such as on parabolic flights are a significant source of error. Therefore, new methods of data analysis, which manage large datasets in a fast and accurate way, are needed to improve data evaluation, to enable on-device analysis and thus also an effective way of data reduction for a faster retrieval of results and to open up the possibility of experiment automation for a more reliable and repeatable experiment procedure.

With the current extensive progress in Artificial Intelligence (AI) and Machine Learning (ML), new opportunities arise for complex plasma data evaluation. An ML model comprises an interconnected structure of neurons, which all consist of inputs and outputs. Based on

the current inputs, internally saved weights and an activation function the neuron output is calculated and passed on to the successive neurons. A neural network usually requires initial training prior to its application. During training, labeled data is fed through the neural network. By applying a specific loss function and optimization method, the error between the network's output and the data label is backpropagated into the neural network and the neuron weights are updated. This is done for multiple epochs until the model reaches a stable maximum in accuracy. This enables the model to gradually form internal representations that capture the relevant patterns in the data and generate useful outputs. Using various neural network architectures and different training techniques ML models are able to generate more accurate results on large datasets and enable previously unfeasible evaluations with possibly lower latency times and on-device if the correct hardware basis is chosen.

The objective of this dissertation is to set the cumulative work into a scientific context and to underline the conceptual and thematic connections of the three individual publications. The focus of the publications lies on the development of novel ML approaches for the analysis of different complex plasma phenomena. At first a convolutional encoder-decoder network is constructed for the analysis of arising ER structures in complex plasmas in publication I. Then the movement of individual particles in the particle cloud is traced under different fast flowing conditions using a Self-Organizing Map (SOM) in publication II. Finally, a Long Short-Term Memory (LSTM) neural network for turbulence detection is developed and trained to enable the detection of particle-resolved turbulence in publication III. First, the current status of complex plasma research is reviewed with respect to the discussed phenomena and the corresponding ML approaches are examined and discussed. Then, the hardware integration of the ML algorithms and automation procedures is presented and lastly molecular dynamics simulations of the onset of turbulence are evaluated to gain insights on the dependence on different simulation parameters.

1.1 Complex Plasmas

The ionized state of a gas is often referred to as the fourth state of aggregation and is commonly known as a plasma. Plasmas occur naturally in a wide range of environments, from lightning and fire to stars and interstellar space. With only an ionizable medium and a source of energy for a sufficient degree of ionization required, almost the entire universe is in a plasma state [19]. The plasmas used as the basis for complex plasmas are weakly ionized low-pressure noble gas plasmas with pressures in the order of a few pascal. In the laboratory, these plasmas are usually generated by electric fields. The electric fields accelerate free electrons which gather kinetic energy until they hold enough energy for the ionization process which releases further electrons that then further ionize the noble gas. Such low-pressure plasmas find a wide range of application in industry, for instance in surface treatment, thin-film deposition, etching in semiconductor fabrication and sterilization processes [20]. Due to the large mass imbalance between electrons and ions, the energy transfer between the two species is inefficient. As a result electrons reach significantly higher thermal velocities and temperatures than ions, which usually remain close to the surrounding room temperature [19]. This disparity in so-called low-temperature plasmas is essential for the plasma shielding and charging effects of microparticles in the plasma.

1.1.1 Shielding and Charging Effects

The plasma bulk itself is said to be quasineutral on a macroscopic level when the charge densities of electrons and ions nearly cancel out everywhere with $n_e \approx n_i$ [21]. When a local charge perturbation is introduced into the bulk it is compensated by a redistribution of electrons and ions, resulting in a screened Coulomb interaction. Using Poisson's equation to calculate the screening effects on a charge potential in Boltzmann distributed electron and ion number densities the screening lengths, also known as the Debye lengths, for electrons and ions can be derived as

$$\lambda_{\mathrm{D}e,i} = \sqrt{\frac{\varepsilon_0 k_\mathrm{B} T_{e,i}}{n_{e,i} e^2}} \tag{1.1}$$

with the Boltzmann constant k_B, the vacuum permittivity ε_0, the electron and ion temperatures $T_{e,i}$, the electron and ion densities $n_{e,i}$ and the elementary charge e. These are then combined to their linearized form [22] by

$$\frac{1}{\lambda_\mathrm{D}^2} = \frac{1}{\lambda_{\mathrm{D}e}^2} + \frac{1}{\lambda_{\mathrm{D}i}^2} \tag{1.2}$$

to

$$\lambda_\mathrm{D} = \sqrt{\frac{\varepsilon_0 k_\mathrm{B} T_e}{e^2 n_e} + \frac{\varepsilon_0 k_\mathrm{B} T_i}{e^2 n_i}} \tag{1.3}$$

which is then usually simplified to

$$\lambda_\mathrm{D} \approx \sqrt{\frac{\varepsilon_0 k_\mathrm{B} T_i}{e^2 n_i}} \tag{1.4}$$

in low-temperature plasmas with $T_e \gg T_i$. The Debye length sets the scale of the screened potential of an introduced charge in the plasma bulk. This Yukawa potential (or Debye-Hückel potential) is described as

$$\Phi_{\mathrm{Y}}(r) = \frac{Q}{4\pi\varepsilon_0}\frac{\mathrm{e}^{-r/\lambda_{\mathrm{D}}}}{r} \tag{1.5}$$

with the charge Q and distance r. Therefore, the screening of charges in low-temperature plasmas is mainly determined by the ions and the quasineutrality of the plasma bulk only applies to scales much larger than the Debye length.

When micrometer-sized particles are introduced into the plasma, they interact with the surrounding electrons and ions. Due to the higher thermal velocity of electrons the flux of electrons on walls and objects in the plasma is much faster and higher than the flux of ions. As a result, the particles acquire a negative charge. An estimation of the charging effects in a low-temperature plasma is given by the Orbital Motion Limited (OML) theory [23, 24]. Hereby, the charging process is described as an interplay of both the electron and ion flux onto the particle. The fluxes are determined by the integral of the corresponding cross sections with each species Maxwellian velocity distribution. Since the electron and ion fluxes are dependent on the electrostatic potential of the particle, they finally result in an equilibrium particle charge, when the electron and ion fluxes are balanced with $I_e + I_i = 0$. The OML theory is based on the assumptions that the particle is isolated with no other particles interfering with the electron and ion motion in its vicinity, that the plasma species motion is collisionless up to the particle and that no effective potential barriers exist. The resulting particle charge generally lies within a range of thousands up to tens of thousands of elementary charges, depending on particle size and plasma parameters [25].

Consequently, the particles in a complex plasma are charged negatively and interact with each other via a screened Yukawa potential. In many experimental conditions, the ratio of the Coulomb energy to the thermal energy, which defines the plasma coupling parameter [26]

$$\Gamma = \frac{Q^2}{4\pi\varepsilon_0 r_{\mathrm{p}} k_{\mathrm{B}} T_{\mathrm{p}}} \tag{1.6}$$

with the interparticle distance r_{p} and particle temperature T_{p}, exceeds one. The particles in complex plasmas can therefore behave as strongly coupled systems. Under such conditions, complex plasmas exhibit properties analogous to classical condensed matter systems, such as liquid-like structures, crystallization, phase transitions and collective waves. Simultaneously, the micrometer-sized particles are sufficiently large to be observed individually in real-time using cameras. Accordingly, complex plasmas provide the perfect opportunity to simulate strongly coupled many-body systems that are observable at the particle level.

Complex plasmas are not only relevant for many-body system simulations but also have their background in nature and industry. First observations of dust plasma interactions were made in radial spokes, which form in Saturn's rings and consist of charged micrometer-sized dust particles in a dense plasma [27]. The interstellar medium and parts of the tail of comets are further natural environments in which complex plasmas occur. Next to that,

complex plasmas play a non-negligible role as particle contamination in semiconductor manufacturing and plasma processing. The main difference to laboratory complex plasmas is that here, the micrometer-sized particles are monodisperse spheres of defined size. Typical particle sizes are in the order of 0.5 to 10 μm and the particles are made of Melamine Formaldehyde (MF), Polymethyl Methacrylate (PMMA) or silicon dioxide.

1.1.2 Electrorheology (ER)

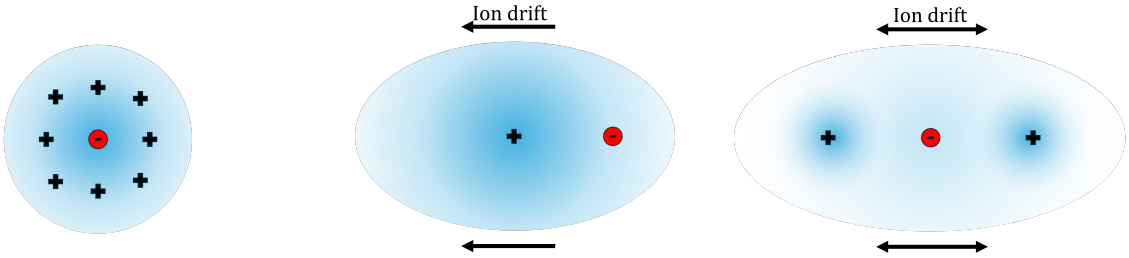
A fluid or suspension is referred to as ER if its flow behavior or viscosity is rapidly and reversibly dependent on an external electric field. Such a fluid consists of some kind of dielectric polarizable particles dispersed in an isolating fluid. The external electric field polarizes the particles and subsequently aligns them along the electric field direction into particle strings and thereby increasing the effective viscosity [28, 29]. This adjustable fluid behavior opens up various possibilities for industrial applications such as in clutches, dampers and actuators.

An analogous phenomenon is observable in complex plasmas [12]. As described in section 1.1.1, the negatively charged micrometer-sized particles are effectively shielded by the positive Debye sphere. In consequence of the electrons and ions moving according to the external electric field, in a Direct Current (DC) electric field the Debye sphere is deformed in the direction of ion drift. Due to the ions streaming around the negatively charged particle, it attracts the flowing ions and focuses their trajectories in an area downstream of the particle. A sketch of the undisturbed Debye sphere and the resulting DC ion wake is shown in figures 1.1a and 1.1b. This displacement of charge carriers results in a modified potential around the particle. The resulting potential is obtained by solving the Poisson equation together with a steady-state kinetic equation for the drifting ions, instead of assuming a Boltzmann distribution [30]. The solution of this coupled problem is approximated using a multipole expansion, yielding the anisotropic wake potential [31]

$$\Phi(r,\theta) \simeq Q\left[\frac{\mathrm{e}^{-r/\lambda_{\mathrm{D}i}}}{r} - 2\sqrt{\frac{2}{\pi}}\frac{M_{\mathrm{T}}\lambda_{\mathrm{D}i}^2}{r^3}\cos\theta - \left(2-\frac{\pi}{2}\right)\frac{M_{\mathrm{T}}^2\lambda_{\mathrm{D}i}^2}{r^3}(3\cos^2\theta - 1) + ...\right] \tag{1.7}$$

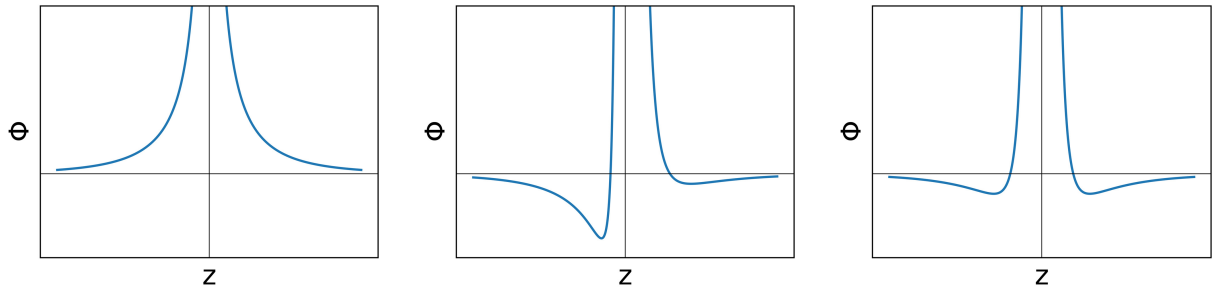
with Q denoting the effective dust particle charge, the thermal Mach number $M_{\mathrm{T}} = u_i/v_{\mathrm{T}}$, the ion drift velocity u_i and the ion thermal speed $v_{\mathrm{T}} = \sqrt{k_{\mathrm{B}}T_i/m_i}$ and the angle θ with respect to the electric field. In a DC electric field the dipole term dominates and an attractive potential well is formed in the ion wake as depicted in figure 1.2b. This leads to a non-reciprocal and non-Hamiltonian attraction of particles into the ion wake downstream of the particle, which causes particle strings to form. When the electric field polarization is changed in a frequency lower than the time scale of the ion response and high enough so that the particles are not able to react due to their high inertness, which is determined by their oscillation frequencies [21]

$$\omega_x = \sqrt{\frac{n_x q_x^2}{\varepsilon_0 m_x}} \tag{1.8}$$



(a) Isotropic Debye sphere (b) Anisotropic DC ion wake (c) Time-averaged AC wake

Figure 1.1: Spatial structures of the particle potential in ER plasmas. Sketch (a) shows the isotropic Debye sphere around a negatively charged plasma particle. Sketch (b) depicts the anisotropic potential, which is deformed by the ion stream (ion wake) and sketch (c) shows the time-averaged wake potential for ions streaming in alternating directions.



(a) Isotropic Yukawa potential (b) DC wake potential (c) AC time-averaged wake potential

Figure 1.2: One-dimensional profiles of the screened interaction potentials. Plot (a) shows the isotropic Yukawa potential as defined in equation (1.5). Plot (b) displays the anisotropic wake potential for ions streaming along the z axis and plot (c) shows the time-averaged wake potential for ions with alternating streaming direction, both corresponding to equation (1.7).

for electrons, ions and dust particles ($x \in \{e, i, d\}$) and which typically lies between 100 and 1000 Hz, the dipole term cancels out and the particle string formation is caused by the quadrupole term. In a so-called Alternating Current (AC) electric field the attraction is typically lower than in a DC field, but the interaction is reciprocal as shown in figure 1.2c. The particle strings form due to self-organization of particles into the ground state of this particle ensemble. A sketch of the AC ion wake is shown in figure 1.1c.

Simulations [32] showed, that for self-organization into strings no attractive potential wells are necessary, but even a less repulsive potential in the ion wakes is sufficient. The recrystallization process was studied using an approximation of the interaction potential

using the Yukawa potential and two adjustable wake potentials next to the particles $\Phi = \Phi_{\mathrm{Y}} + \Phi_{\mathrm{W}+} + \Phi_{\mathrm{W}-}$. The individual wake potentials are given by [32]

$$\Phi_{\mathrm{W}\pm} = \frac{-w_{\mathrm{c}} q}{4\pi\varepsilon_0 |\mathbf{r} \mp \mathbf{r}_{\mathrm{W}}|} \mathrm{e}^{-\kappa_{\mathrm{eff}}|\mathbf{r}\mp\mathbf{r}_{\mathrm{W}}|} \tag{1.9}$$

with the effective screening parameter defined as

$$\kappa_{\mathrm{eff}} = \sqrt{\frac{\kappa^2}{(1 + M_{\mathrm{T}}^2 \cos^2\zeta)} + \kappa_{\mathrm{e}}^2}\,, \tag{1.10}$$

$\kappa = \frac{1}{\lambda_{\mathrm{D}}}$ denoting the inverse Debye length and the electron screening parameter $\kappa_{\mathrm{e}} = \frac{1}{\lambda_{\mathrm{D}e}}$. Here, w_{c} denotes a dimensionless parameter controlling the relative strength of the wake potential, $\mathbf{r}_{\mathrm{W}}$ specifies the position of the wake charge which is given by q and ζ is the angle between ion flow an the displacement vector $\mathbf{r} \mp \mathbf{r}_{\mathrm{W}}$.

To analyze the formation of strings in ER complex plasmas a measurement of whether a particle belongs to a string or not is necessary. Due to the weak one-dimensional symmetry of the string-like structures and the lack of a conclusive definition of particle strings the determination of each particle's configuration is not trivial. To achieve this, several different algorithmic approaches exist. Traditional methods rely on structural indices, derived from the particle positions. The anisotropic weighted scaling index quantifies the local anisotropy by comparing the spatial distribution of neighboring particles within a defined radius [12,13,33]. Using this approach the anisotropy of larger structures and particle clouds is assessable, but determinations for individual particles are not possible. The modified bond orientational order parameter on the other hand evaluates the angular symmetry of particle arrangements through spherical harmonics [34]. This local measure is highly dependent on a definition of thresholds and the nearest-neighbors, which limits for example the detection of the two ends of the strings. Moreover, pair correlation functions have been used to capture global average interparticle correlations and providing estimations for ordered interparticle distances and angles [32, 35, 36]. Using these values as improved thresholds particle-wise string configuration determinations are possible, with the same dependence on neighbors and limitations for particles at the string ends. More recently a first ML approach to string detection was made using a network consisting of convolutional layers and a fully connected Multilayer Perceptron (MLP) for classification on reconstructed particle structures [14], facing the same limitation of strong locality and limitation on string ends.

In the scope of publication I, it is shown, that the traditional approaches not only face theoretical disadvantages but also deliver inferior performance on labeled particle string data. This work presents a new ML approach based on a specially adapted convolutional encoder-decoder network to resolve those errors. The presented network determines all present particle strings globally simultaneously on the image itself while not being dependent on the particle positions, enabling investigations of the influence of ER effects on DAW propagation [17].

1.1.3 Plasma Turbulence

Another collective many-body behavior that is widespread in nature as well as observable in complex plasmas is turbulence. As an ubiquitous phenomenon turbulence occurs in various fluid dynamic systems, such as flows in water [37, 38], atmospheric weather patterns [39], biophysical systems [40, 41] and astrophysical systems such as the sun [42]. Turbulence is essential for several engineering applications in optimizing designs for a higher efficiency and even ensuring safety. Therefore, turbulent flows have been studied extensively over several decades. Despite extensive research efforts, there is no single, universally accepted definition of turbulence that applies unambiguously across all flow configurations and levels of description [43]. Intuitively, a turbulent flow constitutes irregular and chaotic patterns, which contain vortices. Similarly, five key aspects were defined that attempt to distinguish turbulent flows from laminar flows as their considered counterpart [44].

First, turbulent flows are chaotic, exhibiting strong spatiotemporal fluctuations in velocity, density, pressure or temperature, which makes their evolution intrinsically unpredictable. Such fluctuations in flow parameters are referred to as intermittencies [45]. Second, a turbulent flow consists of rotational motion. The flow contains angular momentum and its vorticity is expressed in vortex structures and eddies across a range of scales. Third, turbulence incorporates a diffusive property. The presence of vortical motion enhances mixing processes in between flow layers and leads to an effective transport of momentum far exceeding molecular diffusion [46]. Fourth, turbulent flows are resistive, generating a significantly higher drag compared to laminar flow due to the increased momentum exchange between flow layers [47]. It was shown very early on by Reynolds that turbulence increases the resistive forces in a flow to scale quadratically with velocity $F_{\text{drag}} \propto v^2$ compared to linearly for laminar flows $F_{\text{drag}} \propto v$ [48]. Lastly, turbulence is self-similar across scales. The energy in turbulent systems is successively transferred from larger to smaller scales by vortices. This energy cascade is scale-invariant and constitutes the multi-scale structure of turbulence. The energy spectrum $E(k)$ exhibits a power-law dependence on the wavenumber k, which is inversely proportional to the spatial scale, with the typical relation $E(k) \propto k^{-5/3}$ for fully developed, isotropic turbulence [49, 50]. The underlying energy cascade from large to small spatial scales is illustrated schematically in figure 1.3. At the smallest scales, also referred to as the Kolmogorov microscale η, which is defined by the viscosity ν and the mean rate of kinetic energy dissipation per mass unit ε in [51]

$$\eta = \left(\frac{\nu^3}{\varepsilon}\right)^{1/4}, \tag{1.11}$$

the energy transfer ends in heat dissipation. Moreover, the flow is characterized by its Reynolds number $\mathrm{Re} = vL/\nu$, as a ratio of viscous and inertial forces, defined through the flow speed v, kinematic viscosity ν and the characteristic length scale L. Usually, a high Reynolds number of $\mathrm{Re} = \mathcal{O}(10^3)$ is an indication for turbulence in typical fluid dynamics [52]. Nevertheless, turbulence at lower Reynolds numbers is common as well in biological systems [41] and viscous polymers [53]. Turbulence in complex plasmas can be observed at Reynolds numbers of $\mathrm{Re} = \mathcal{O}(10)$ [54].

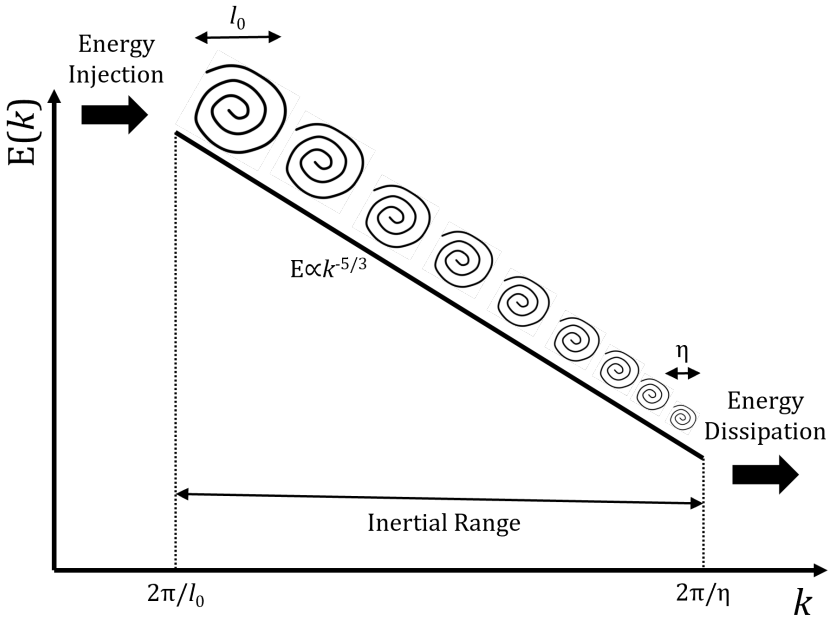


Figure 1.3: Schematic representation of the turbulent energy cascade in three-dimensional isotropic turbulence. Large-scale vortical structures transfer kinetic energy to progressively smaller scales, resulting in a continuous reduction of their characteristic size. The energy transfer proceeds without significant dissipation throughout the inertial range until the smallest scales, characterized by the Kolmogorov microscale η, are reached. At this point, viscous effects dominate and the remaining kinetic energy is converted into heat.

Complex plasmas provide a unique medium for turbulence research. In a complex plasma not the continuous background plasma is considered, but the turbulent behavior of the individual micrometer-sized particles. This enables the direct observation of kinetic properties of the particle motion. On the other hand the background plasma and its effect on the particle motion needs to be taken into account. In contrast to simple water currents, the particles experience a significant damping in the plasma environment. Turbulence can still develop in such systems with background damping [55] but it needs to be evaluated whether the damping is low enough compared to the rate of cascades. In a complex plasma the background damping is governed by the Epstein damping coefficient [56]

$$\gamma_{\mathrm{Ep}} = \delta_{\mathrm{Ep}} \sqrt{\frac{8}{\pi}} \frac{p}{a \rho_{\mathrm{d}} v_{\mathrm{th},n}} \tag{1.12}$$

using the gas pressure p, the particle radius a and mass density ρ_{d} of the dust particles, the neutral gas thermal velocity $v_{\mathrm{th},n}$ and an interaction parameter δ_{Ep}, which is approximated to be 1.44 on micrometer-sized particles in a plasma. The rate of cascades is given by $\omega_{\mathrm{cas}} = k\sqrt{2kE}$, where k denotes the wavenumber and $E = E(k)$ is the corresponding energy spectrum. If the rate of cascades exceeds the Epstein damping $\omega_{\mathrm{cas}} > \gamma_{\mathrm{Ep}}$ turbulent

vortices can emerge and an energy cascade can occur. The presence of turbulent structures in complex plasmas is therefore highly dependent on the background gas pressure and the used particle sizes.

In complex plasmas turbulences have primarily been observed and analyzed in combination with DAWs [6–8] and in auto oscillating plasmas [9, 10]. Further two-dimensional simulations have been done on vortices [57] and particle monolayers [58]. Additionally, the onset of turbulence was simulated in three dimensions for a complex plasma flowing past a fixed particle charge, which highlights the possibility of flowing particle turbulence in an experimental setup that closely resembles PK-4 [54].

One major challenge of turbulence research is the definite determination of whether turbulence occurs in the system. Due to the broad definition of turbulence a classification, especially in particle-resolved systems, is difficult. To confirm the onset of turbulence in the simulations [54], a mixture of the Turbulent Kinetic Energy (TKE) and vorticity is chosen. TKE is the kinetic energy of a particle calculated by the turbulent velocity $\mathbf{v}'$, which separates the background flow velocity from the turbulent interactions between particles with $\mathbf{v}' = \mathbf{v} - \overline{\mathbf{v}}$ and defines the TKE as $\mathrm{TKE} = \frac{1}{2} m \mathbf{v}'^2$, with $\overline{\mathbf{v}}$ being the locally averaged flow velocity computed in bins of predefined size. Pronounced local deviations of these measured values suggest that turbulence is present in the system, while a clear classification especially on an individual particle level is not possible.

Publication III addresses the challenge of turbulence classification in particle-resolved three-dimensional systems. In the context of this work an LSTM neural network is constructed and trained on synthetic turbulent and non-turbulent particle flow data to enable a turbulence classification based on a particle's trajectory. This trained network opens up further possibilities in complex plasma turbulence research but is also applicable to turbulence detection in virtually any particle-resolved three-dimensional system.

1.2 Machine Learning

AI is the general term for a system performing tasks in a way that normally requires human intelligence, such as perception, decision-making, reasoning and solving of previously unknown problems. ML refers to a subfield of AI that more specifically focuses on algorithms that acquire the same capabilities by learning patterns and relationships directly from provided data. The nature of the acquired capabilities, the kind and quantity of input and output data and the fundamental design of the ML model vary substantially depending on the application. This section is intended to provide an overview and the theoretical foundations of the developed and implemented ML algorithms within the three appended publications. In experimental complex plasma research the application of ML is particularly valuable since modern diagnostics such as used in the PK-4 experimental setup produce large image sequences with substantial particle-resolved data embedded within these images. Especially during time-critical experiments for example when capturing particles in the field of view of the cameras or during short periods of microgravity on parabolic flights, where large datasets need to be evaluated fast and reliably, ML algorithms offer many advantages over manual, rule-based and deterministic algorithms. Furthermore, recording conditions change continuously due to variations in illumination, particle density and particle motion. This makes it difficult to define fixed thresholds or deterministic rules that remain valid across different experimental runs. Additionally, many physically relevant states are not as sharply defined as they would allow for simple rule-based analysis methods.

The focus of this work is therefore to use physical insight as a guiding principle for model design, training strategy and evaluation to take advantage of the strong methodological advantages of ML algorithms in complex plasma research. Within this framework, three main classes of problems are addressed. The first class concerns the detection of structural patterns directly in camera images, with particular focus on the identification of particle strings formed under ER conditions. The second class addresses the reconstruction of particle motion, including particle matching between successive images using a SOM and the plausibility assessment of trajectories using sequence models. The third class focuses on the classification of dynamical states, where an LSTM network is used to distinguish turbulent from non-turbulent motion based solely on particle trajectories.

1.2.1 Theoretical Background

Fundamentally, ML can be formulated as the approximation of an unknown mapping function $f_\theta(x) : X \to Y$, where input data $x \in X$ is mapped to corresponding targets $y \in Y$. The functional parameters θ are adjusted during a process called training. The objective of the training is to determine a set of parameters that minimizes a defined loss function that measures the discrepancy between model output and the desired target values. Therefore, ML corresponds to an optimization problem in a high-dimensional parameter space. The usual learning procedure in ML is supervised learning, in which training data consists of precise input and output pairs. In contrast, unsupervised learning occurs when no targets are included in the training data and the mapping function parameters are

optimized using other quantities and specifications. After training the mapping function should recognize patterns and have them embedded in their function parameters so that it can generalize to further data, which is not part of the training dataset. The mapping function is practically represented by a neural network model. This usually consists of a concatenation and interlinking of single neurons. The neurons receive a number of input signals upon inference and calculate a number of output signals using arithmetic operations and activation functions, which are then further propagated to the following neurons. The model size, architecture and mathematical operations vary strongly based on the application and training data. This starts with simple connections of individual neuron layers in an MLP, which can be used for simpler classification tasks and for example handwritten digit recognition [59], over to more complex neuronal filter kernels in Convolutional Neural Networks (CNNs) that move across higher-dimensional data for image segmentation [60] or three-dimensional reconstruction of particle positions [61].

The model parameters, which are also referred to as neuron weights since they affect the neuron's output value calculation, are optimized using gradient-based methods. The gradients of the loss function are computed with respect to the model parameters and are updated iteratively in the direction that reduces loss. This backpropagation algorithm applies to the layered structure of the neural network model. The optimization is usually performed on small subsets of the training data referred to as batches. This approach provides a computationally efficient approximation of the true gradient and allows the training process to scale to large datasets and modern hardware architectures. The choice of loss functions depends on the specific task, training data and network architecture. For simple classification processes the Mean Squared Error (MSE) function is defined as the difference between model outputs $\hat{y}$ and the target labels y as

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2. \tag{1.13}$$

Due to the squared error weight strong deviations have a strong impact on the backpropagation. The MSE loss function performs well on classification problems, where no strong class imbalances are present. For image-based segmentation problems, pixel-wise classification losses are employed, which can be adapted to more strongly imbalanced class distributions, when for example a large amount of pixels display a background and only few pixels deliver important information. This is usually done using cross entropy loss functions. Additionally, class imbalances are compensated by class weights and model outputs and targets are regarded in binary as in the Weighted Binary Cross Entropy (WBCE) function [62]

$$\mathrm{WBCE} = -\frac{1}{n}\sum_{i=1}^{n}\frac{w_1 \cdot y_i \cdot \log(\hat{y}_i + \epsilon) + w_0 \cdot (1 - y_i) \cdot \log(1 - \hat{y}_i + \epsilon)}{w_0 + w_1} \tag{1.14}$$

with class weights w_0 and w_1 and a small ϵ to avoid numerical instabilities due to $\log(0)$. The loss function is chosen to reflect the physical interpretation of the task and to guide the optimization toward physically meaningful solutions.

The optimization itself is typically performed using the Adam optimizer [63]. The Adam optimizer combines the advantages of momentum-based methods and adaptive learning rate approaches by maintaining exponentially decaying estimates of both first and second moments of the gradients. This allows each parameter to be updated with an individually adapted step size, which leads to a fast convergence and an increased robustness with respect to noisy or sparse gradients.

1.2.2 Training Process

The loss optimization process strongly depends on the training data. Training data is fed through the untrained neural network model in batches and deviations of outputs and targets are backpropagated into the model weights based on the chosen loss function. Therefore, when using supervised learning methods, corresponding labels or targets are required. Additionally, depending on the generalization task and model size large training datasets are necessary. Since such amounts of prepared data are rarely available and the quality of pattern recognition depends on it, training data is often generated synthetically. The training data is divided into three groups. The largest group is reserved for the training process, while two smaller groups are used for validation during training and for testing after training. The validation and test datasets should be different in content to ensure that generalization and pattern recognition are successful. The training data is normalized to values in the interval of [0,1] to ensure consistent feature scaling and stable training behavior. The training data should be adjusted to the expected experimental data as well as possible regarding noise levels and other physical characteristics while preventing sharp unwanted boundaries that could themselves affect the pattern recognition.

To further enhance robustness and generalization, several regularization strategies are employed during training. A dropout procedure is often applied, where several weights are instantaneously cut down towards zero to reduce coadaptation of network components. The training is monitored over all epochs, which are training sections in which all training data is passed through once in batches, using the validation dataset. Large deviations in accuracy and loss between training and validation indicate errors in training and uneven distributions in the datasets. An early stopping of training is employed upon plateauing loss values to prevent overfitting on the training data. In addition, adaptive learning rate schedules are used to stabilize convergence and improve optimization efficiency across different network architectures.

Model performance is then evaluated using task specific metrics that reflect the physical objectives of each application. For most ML methods the performance on an unknown synthetic test dataset or experimental datasets are a good measurement. Metrics such as accuracy, as the percentage of correct classifications, precision, as the fraction of predicted positives that are correct, recall, as the fraction of actual positives that are correctly identified, and the F1 score, defined as the harmonic mean of precision and recall, are used to quantify the quality of classifications.

1.2.3 Convolutional Neural Networks

CNNs are commonly used for image segmentation and analysis tasks. In contrast to fully connected neurons in structured layers in an MLP a convolutional layer consists of weight kernels that are applied across the input as a filter and provides an output mapping that includes spatial correlations of the input. The mathematical formulation of the two-dimensional convolution is given by

$$y(i,j) = \sum_{m=0}^{k_h-1} \sum_{n=0}^{k_w-1} x(i+m, j+n) \cdot w(m,n) + b, \tag{1.15}$$

where x is the input feature map, w denotes the trainable kernel of size $k_h \times k_w$, a bias term b and the resulting feature map y. This formulation corresponds to the sliding window dot product used in standard convolutional layers [64]. Each convolutional layer increases the effective receptive field of the network, meaning that deeper layers integrate information from progressively larger spatial regions. This hierarchical structure allows CNNs to capture both fine local details and more global spatial patterns. Pooling operations further reduce spatial resolution while retaining the most relevant information, thereby increasing robustness against noise and small positional variations. A pooling layer employs a filter kernel to compute either the maximum or mean of local input regions, effectively reducing the spatial resolution of the input image. Such image downsampling and inverse upsampling methods enable an image to be encoded and decoded. For image segmentation and structure detection tasks, encoder-decoder architectures are particularly well suited. In the encoder, successive convolution and pooling operations extract increasingly abstract features while compressing the spatial representation. The decoder then reconstructs a dense output by progressively upsampling the encoded features back to the original resolution. Skip connections between corresponding encoder and decoder layers preserve high-resolution spatial information and enable precise localization, which is essential when individual particles or fine structures must be resolved. Such an encoder-decoder CNN with skip connections is referred to as a U-Net [65].

U-Net

The U-Net architecture with its dimensional bottleneck allows for high-resolution spatial information to be recognized, retained throughout the network and combined with semantically rich features extracted at deeper levels. The U-Net output is to be interpreted as a probability map, where each output pixel value represents the likelihood of belonging to a specific class. Using an activation function at the network output this probability map is essentially converted to a binarized result map, that marks all recognized structures. Usually, a Sigmoid activation function as in

$$\sigma(x) = \frac{1}{1+\mathrm{e}^{-x}} \tag{1.16}$$

is used to map the output values to the interval [0,1], thus essentially binarizing the output.

Such a U-Net is trained and applied directly on camera images of the PK-4 experimental setup, where it reliably recognized particles in the image and marks them in the output probability map of the same resolution [60]. This way, a binary particle mask is generated that is based on the learned spatial context, rather than local intensity thresholds. Therefore, the network remains reliable in the presence of strong noise, non-uniform illumination and particles of different sizes. The weighted centroid position of the marked particles is then identified and the exact particle positions are determined. Next to its strong reliability, the simultaneous global particle localization accelerates the analysis process.

Asymmetric Kernels for String Detection

A similar U-Net architecture is constructed in publication I to identify string formation patterns in the complex plasma image data. The network is constructed based on the same encoder-decoder principle. The image resolution is decreased in three convolution and pooling steps. An image with a resolution of 2048×2048 pixels is encoded to 256×256 pixels. At the same time the filter dimensions increase. While the first layer consists of 4 filtering dimensions, the dimensions double in size until they reach 32 in the bottleneck of the neural network. Afterwards, the image is decoded back to its original size using three reversed transposing deconvolutions and skip connections to earlier points of the network to retain the positional information. Lastly, four resolution retaining convolutions are completed with a Sigmoid activation to create a binarized output image. Hereby, the convolution and deconvolution trainable filter kernels are elongated in size, with increasing elongations for the higher resolution convolutions. This is done to facilitate neuron activation in the horizontal direction to increase the string detection accuracy, since the particle strings present in PK-4 only appear in horizontal direction. This way, the model's receptive field is purposefully anisotropic, reflecting the anisotropy of the particle strings. A schematic of the network architecture is shown in publication I.

Using this network architecture alone does not enable particle strings to be detected. Customized synthetic training data is necessary, since no usable labeled particle string data exists. For training input and target image pairs are created, displaying individual particles, which are partly ordered in string-like arrangements and partly isotropically distributed. The target images only display a trace of the string-like arranged particles. This way, the model learns to trace particles in a string arrangement and ignores isotropically distributed particles. It was shown that the original PK-4 experimental images feature an insufficient signal-to-noise ratio for efficient neuron activation. Therefore, binarized input images are chosen. For model inference, the binarized output of a preceding simple U-Net for particle detection [60] is used. An example of an original PK-4 image, a binarized U-Net output, the string detection network output and the final particle-wise classification is shown in figure 1.4.

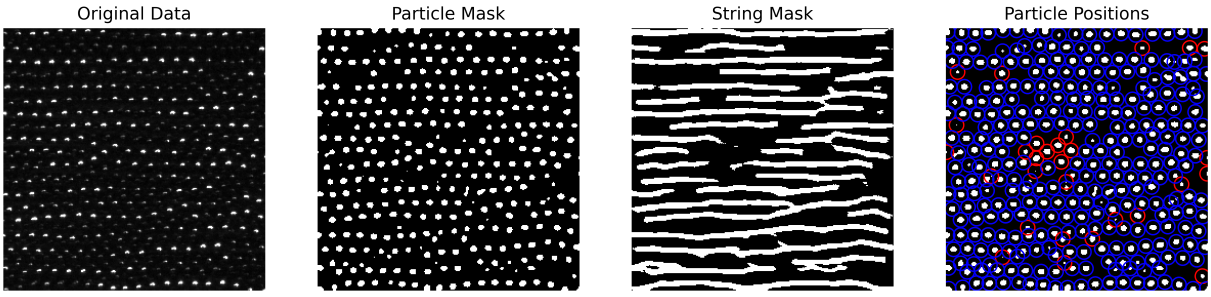


Figure 1.4: Schematic of the input and output data of the CNNs. The *original image* is processed by the initial U-Net, producing a binarized *particle mask* that is used as input for the CNN designed to detect string formations. The output of this network is a *string mask* highlighting particles within string structures (blue) and particles outside of strings (red).

1.2.4 Self-Organizing Maps

In contrast to the above methods, the SOM is an unsupervised ML algorithm. Therefore, the SOM does not feature any training data using labels. In fact, no training is necessary at all. The SOM is based on a Kohonen network [66], whose optimization process consists of competitive learning. Consequently, the input neurons constantly compete for activation based on preset rules and each activation reorganizes the neurons in the network. Therefore, using nothing other than the given input data the network organizes itself, hence the name. The usage of a SOM for Particle Tracking Velocimetry (PTV) and the underlying algorithm idea was first presented by Labonté [67]. The two-dimensional particle coordinates of two successive images represent two separate neural networks with neurons, each consisting of two weights representing the x and y coordinates of each particle. The weights of the neurons of the two neural networks a_i and b_j are therefore defined by the particle coordinates as

$$\begin{aligned} a_i &= (x_i, y_i), \text{ with } i = 1, 2, ... N \\ b_j &= (x_j, y_j), \text{ with } j = 1, 2, ... M \end{aligned} \tag{1.17}$$

with N particles in the first and M particles in the second image. On inference, every neuron of one network is used as input for the other network. The competitive learning approach is now defined through an activation of the Best-Matching Unit (BMU). Using the Euclidean distance the nearest-neighbor based on the two neuron weights in the other network to the input neuron is determined. When the BMU is closer than a predefined threshold d_{th} an activation takes place. The activation process not only affects the BMU neuron but also its neighbors in a radius R. The activation features a weight adjustment Δb_j, which in other words describes a small displacement of the neuron in the network. The weight adjustment is defined by a parameter α_j and the displacement between the weights of the BMU b_w and the input neuron weights a_i as in

$$\Delta b_j = \alpha_j * (a_i - b_w), \text{ with } j = 1, 2, ... M. \tag{1.18}$$

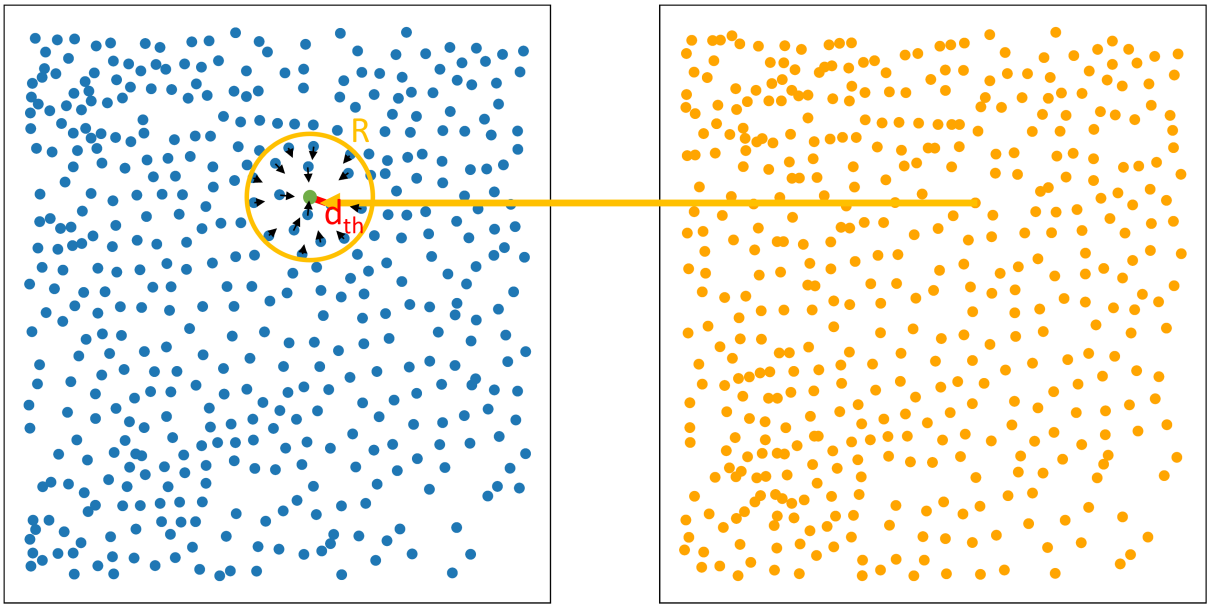


Figure 1.5: Schematic illustration of the SOM operating principle. The two networks represent particle configurations from consecutive images. Neurons are successively mapped onto the opposing network and when a BMU is found within a threshold distance d_{th}, all neurons inside a radius R are activated, which is depicted by the arrows and their weights are adjusted accordingly.

The parameter α_j is defined by a radial-dependent Gaussian function as in

$$\alpha_j = \begin{cases} \alpha & \text{for } |b_w - b_j| \leq R \\ \alpha \cdot \exp\left(-\dfrac{(|b_w - b_j| - R)^2}{2R^2}\right) & \text{for } |b_w - b_j| > R. \end{cases} \tag{1.19}$$

This way, the BMU and its surrounding neighbors are displaced in the direction of the input neuron. A schematic of this activation process is shown in figure 1.5. This activation process is repeated for every neuron of one network. Then all neurons of the other network are used as input vice versa. The entire process is then repeated for a defined amount of iterations, while the BMU neighbor radius R is further decreased, to only include the BMU in the last iterations, and while α increases. Finally, the neurons of the two successive images have converged. The particles that are represented by two neurons that are closer than a small ϵ after all iterations are considered matching particles.

The SOM is particularly well suited for particle tracking in PK-4 and similar experiments as it does not rely on fixed heuristics or manually tuned thresholds. In contrast to classical nearest-neighbor approaches or conventional preceding tracking methods such as Trackpy [68], the SOM establishes particle connections through a global, competitive optimization that adapts to the instantaneous particle configuration. This is advantageous in dense scenes with strongly varying velocities, where particles can undergo large and heterogeneous

displacements between frames. Furthermore, particles can leave or enter the illuminated plane visible to the camera causing inconsistent particle numbers and trajectories. In such cases, fixed maximum displacement or velocity thresholds are often violated, leading to mismatches. The SOM overcomes this limitation by iteratively minimizing the overall mismatch between two particle sets rather than enforcing local, frame to frame constraints.

1.2.5 Long Short-Term Memory Networks

Many physically relevant processes in complex plasmas are inherently time-dependent and cannot be characterized reliably from single snapshots. Particle motion, collective rearrangements and dynamical transitions evolve over multiple time frames and their identification requires access to temporal context. Time series models are therefore a natural choice when the goal is to analyze local dynamics rather than instantaneous configurations. Recurrent Neural Networks (RNNs) extend the concept of an MLP to sequential data by introducing feedback connections that allow information from previous timesteps to influence the current output. In a standard RNN, a hidden state is updated recursively and acts as a memory of past inputs. However, classical RNNs suffer from the vanishing gradient problem, which limits their ability to learn long-range temporal dependencies. Gradients propagated through many timesteps tend to decay, making it difficult to retain information over extended sequences [69]. LSTM networks address this limitation by introducing an explicit memory cell and a set of gating mechanisms that control information flow. In addition to the input and output transformations known from feedforward networks, LSTM cells employ input i_t, forget f_t and output o_t gates that regulate which information is stored, retained or discarded. This gated structure enables stable gradient propagation over long sequences and allows the network to capture both short-term fluctuations and long-term trends in time series data. A schematic of an LSTM cell is depicted in figure 1.6. As a result, LSTMs are well suited for the classification of dynamical processes in which relevant information may be distributed over many frames.

In this work, LSTMs are applied to time series derived from particle trajectories. Rather than using raw positions, the input is constructed from trajectory bound features such as velocity, acceleration, jerk, curvature and angle change. These quantities are directly motivated by classical mechanics, invariant under translation and rotation and normalized with respect to characteristic spatial and temporal scales. This feature design ensures that the network focuses on intrinsic dynamical behavior rather than on experimental geometry or absolute motion. Two main application scenarios are addressed. In the first, an LSTM is used to assess the plausibility of reconstructed particle trajectories and to detect tracking errors or outliers that arise from ambiguous particle matching. In the second, a more complex architecture is employed for the classification of turbulent versus non-turbulent motion. In this case, the network learns characteristic temporal patterns associated with complex, multi-scale dynamics directly from individual trajectories, without requiring field-based quantities or spatial averaging. Together, these applications demonstrate the suitability of LSTM networks for particle-resolved, time-dependent analysis in complex plasma experiments.

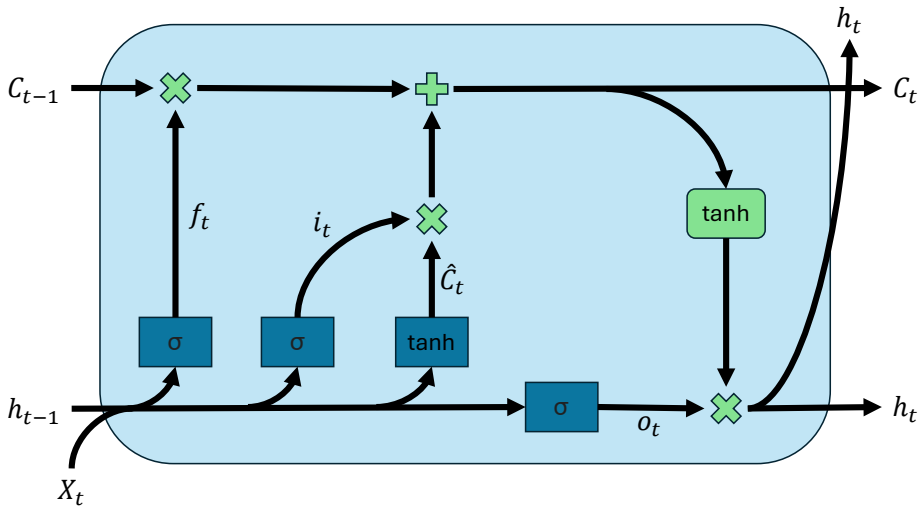


Figure 1.6: Illustration of an LSTM cell. The incoming signals consist of the previous cell state C_{t-1}, the previous hidden state h_{t-1} and the current input X_t. These are processed by the forget, input and output gates (f_t, i_t, o_t) as indicated by the directed connections. Sigmoid (σ) and hyperbolic tangent ($tanh$) activation functions are highlighted in blue boxes, while the arithmetic operations are indicated in green. On the right hand side, the updated cell state C_t and the output of the LSTM cell are depicted.

1.2.6 Implementation

The ML algorithms presented in this work are to be implemented to operate directly at the experiment and are therefore subject to the constraints of edge computing, where computational resources, memory and latency times are inherently limited. In this context, implementation aspects become crucial, as inference must be performed reliably and with low latency in order to enable real-time or near-real-time feedback during the experiment, for example when evaluating short temporal windows or reacting to rapidly evolving dynamics. To deploy the models efficiently on embedded hardware, model compression and acceleration techniques are employed. Quantization refers to the reduction of numerical precision used to represent model parameters, weights, activation functions, input and output data, thereby decreasing memory usage and improving computational performance. In the present setup, the CNNs are executed on the GPU of an NVIDIA Jetson platform using mixed-precision floating-point arithmetic (Float16/32). In this approach, model parameters are stored in high-precision format (Float32), while the majority of arithmetic operations are carried out in half-precision (Float16). This significantly reduces memory bandwidth requirements and increases computational throughput, as modern embedded GPUs provide specialized hardware support for Float16 operations. This choice was found to offer an optimal trade-off between inference speed and accuracy, as it aligns well with the hardware capabilities of the target platform, while more aggressive integer quantization schemes did not yield additional benefits in this specific configuration.

During the particle matching process using the SOM the Euclidean distance to all other present particles needs to be calculated repeatedly. This is highly inefficient for high numbers of particles on the image, as the computational effort scales quadratically with the number of particles. To address this, on high particle numbers the image is subdivided into separate sections in which particles are matched separately. Overlaps and a careful merging of sections after matching ensure that no boundary effects occur. Using these image sectioning and multiprocessing techniques the inference time for particle matching is drastically reduced.

In addition, the network architectures are deliberately kept compact. The number of convolutional layers and the size of the convolutional filters are minimized while preserving task performance, effectively reducing model complexity and inference time. This form of architectural pruning lowers the overall model size and improves runtime efficiency without compromising the robustness of the analysis. All remaining CPU-bound processing steps, such as preprocessing and postprocessing operations, are parallelized using multiprocessing to fully utilize available system resources. Together, these measures enable fast, reliable and reproducible deployment of ML-based analysis directly at the experiment under realistic hardware constraints.

1.3 Plasmakristallexperiment-4

The novel ML algorithms and neural networks presented here can be applied to various particle-resolved systems but are specifically designed for an application in complex plasma systems. In the scope of this work and the underlying project Artificial Intelligence for Plasma Experiments (AIPEX), funded by Deutsches Zentrum für Luft- und Raumfahrt (DLR) and the German Federal Ministry of Economic Affairs and Climate Action under grant number 50WK2270B, the developed ML algorithms are to be implemented into the experimental setup PK-4 with the aim of enabling on-device image evaluation for a faster retrieval of experimental results and a partial automation of the experiment procedure to reduce the demand for human involvement and minimize the potential for errors. In this section the PK-4 setup is to be briefly introduced and the practical implementation of ML algorithms is discussed.

PK-4 is part of a long history of predecessor Plasmakristallexperiment setups. As a successor to the experiments PKE-Nefedov [70] and PK-3 Plus [71], PK-4 [18] is stationed on the ISS since 2014. This work is based on an Engineering Model (EM) of PK-4 that is present at Justus-Liebig-University in Giessen, which represents a slightly altered and modified version of the original Flight Model (FM) of PK-4 which is installed on the ISS. This EM allows for laboratory use, improvements of the experiment diagnostics, additional hardware implementations for data evaluation and experiments under microgravity during parabolic flights. The main differences of the EM to the FM are highlighted in the following. The novelty of the PK-4 experiment is the background plasma generation using a DC electric field discharge and its elongated Π-shaped glass plasma chamber. This geometry is significantly well suited for studying dynamical processes and particle flows. A schematic of the PK-4 experimental setup is shown in figure 1.7. Nevertheless, stationary processes such as crystallization and string formation are accessible as well using a fast switching of the electric field polarity, as discussed in section 1.1.2.

As further described in [18], the plasma chamber glass tube with a diameter of 30 mm is closed off by two high-voltage electrodes for plasma generation on each side. Using a high-voltage generator voltages of up to 2.7 kV are achieved which enable a DC discharge and generate the main plasma. The plasma discharge is then current controlled, with currents ranging from -3.1 mA to 3.1 mA. In addition to the stationary DC discharge the high-voltage electric field can be altered by a pulse generator, as shown in figure 1.8. By setting the discharge currents I_0–I_3 and times t_0–t_3 an adjustable AC discharge is generated and the electric field duty cycle $\delta = t_3/t_1$ is defined and modifiable. This way, AC discharges with frequencies of 100 to 1000 Hz can be generated to keep the micrometer-sized particles in the working area and to control their velocity along the tube axis by adjusting the duty cycle δ.

The high-voltage electrodes additionally serve as inlets for the gas supply and vacuum system. On the one side a turbomolecular vacuum pump and a backing pump provide the base vacuum for the system and on the other side either Argon or Neon gas bottles can be connected to supply the gas flow controllers and produce a gas flow of 0.1 to 10 sccm in a working pressure environment of 10 to 250 Pa. Near the two ends of the plasma glass

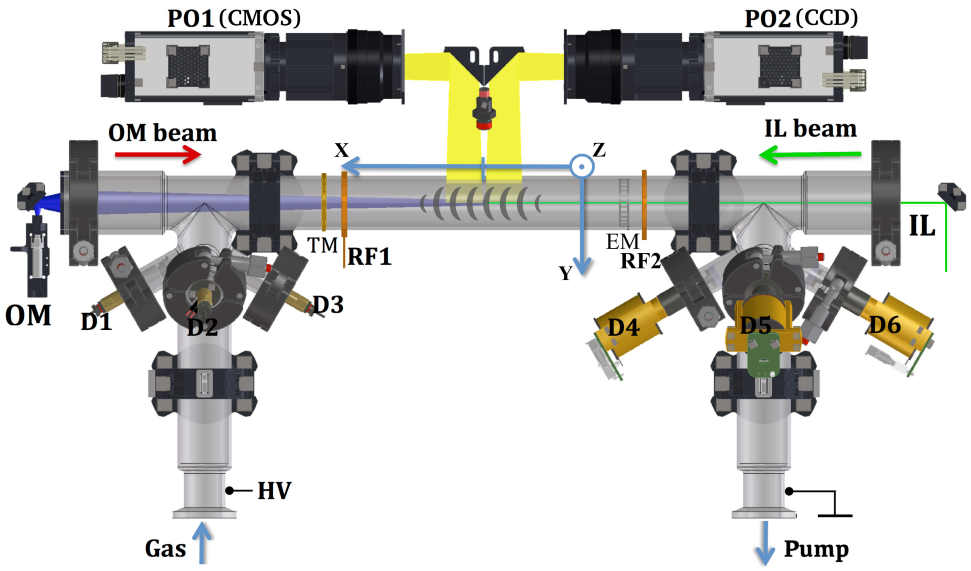


Figure 1.7: Schematic of the PK-4 experimental setup. The high-voltage, vacuum and gas systems as well as particle dispenser and manipulation systems are all attached to the plasma glass tube. The inserted micrometer-sized particles are visible to cameras installed on a movable mount. Abbreviations used in the figure include IL (illumination laser), OM (optical manipulation), RF (radio-frequency), EM (electromagnetic manipulator), TM (thermal manipulator), PO1/PO2 (particle observation), D1-D6 (dispenser) and HV (high-voltage). Image adapted from [72].

tube five particle dispensers are located. These work by shaking a reservoir filled with micrometer-sized particles covered with a sieve by an electronic actuator. The rapidity and number of shakes is adjustable to have an impact on the amount of emitted particles. The type and size of particles and according sieve sizes are interchangeable. In contrast, the FM is additionally equipped with gas jet dispensers that work by using short gas pulses to inject the particles, which have been replaced in the EM. To manipulate the particles in the plasma PK-4 is equipped with a ring electrode, two radio-frequency coils, a thermal element and a manipulation laser that enable the disturbance of particles in according experiments.

The transparent glass plasma chamber allows for a direct observation of particles in the plasma. To make the particles visible to cameras an illumination laser with a wavelength of 532 nm is expanded into a thin laser sheet, which illuminates a two-dimensional plane of particles in the particle cloud. The focus of the laser sheet is movable to adjust the thickness of the illuminated plane. The illuminated particles are then recorded by three cameras. Two of those cameras are used for particle observation. Mounted next to each other on a movable translation stage are a Complementary Metal-Oxide-Semiconductor (CMOS) camera with a resolution of 2048×2048 pixels operating at 90 Frames per Second (FPS), resulting in a spatial resolution of 11.4 μm/px, and a Charge-Coupled Device (CCD)

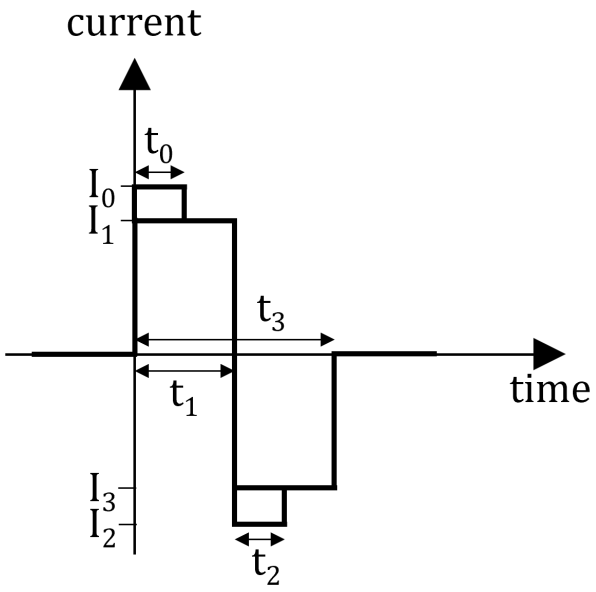


Figure 1.8: Illustration of the high-voltage generator signal. The polarity of the generated voltage can be adjusted precisely based on the configured currents I_0–I_3 and times t_0–t_3. This way, an alternating electric field can be generated.

camera with a resolution of 1600×1200 pixels at 35 FPS, resulting in a spatial resolution of $14.2\,\mu$m/px. The CMOS camera was added to the EM to enable a faster and more accurate particle observation compared to the FM, which only includes CCD cameras. Additionally, a particle glow observation camera is mounted to observe the plasma glow of the entire chamber using different wavelength filters to generate an overview of the plasma glow and particle bulk. Lastly, a spectrometer is mounted to analyze the wavelength spectrum of the plasma glow.

1.3.1 Artificial Intelligence Hardware Implementation

The experimental controls are subdivided into eight according modules: Power Distribution, High-Voltage Generator, Radio-Frequency Generator, Particle Observation, Optical Manipulation, Dispenser Control and Gas Control and lastly Vacuum and Gas Control. Those modules are connected via two separate Controller Area Network (CAN) buses. The PK-4 experimental setup is controlled by a Windows PC that is connected to both of the two CAN bus systems. Using a Human–Computer Interaction (HCI) program the user is able to address every individual experimental module and execute predefined commands and settings to the experiment. To simplify and automate the experiment startup and experimental procedures the HCI is able to execute scripts using a provided library similar to the C programming language. Since some processes, such as capturing the particles within the field of view, cannot be preprogrammed, human intervention during the experiment

is required, which also means that errors cannot be completely avoided. Within such a closed system no implementation of ML algorithms and according automation is possible. Therefore, additional dedicated hardware is integrated.

For this purpose an NVIDIA Jetson AGX Orin developer board was chosen. With a 12 core ARM Cortex A78AE CPU, 64GB of LPDDR5 RAM, 2048 CUDA cores and 64 Tensor cores it is able to deliver up to 275 Trillions of Operations per Second (TOPS) of processing capability. This delivers much more graphical processing capabilities than other edge devices such as Raspberry Pis and provides much more development flexibility than for example using a Field-Programmable Gate Array (FPGA). Additionally, the Jetson board is equipped with CAN bus integration. The aim of this NVIDIA Jetson board is to display and save all data delivered by the updated CMOS camera, which operates independently to the CAN bus systems of PK-4, to enable an on-device analysis using the presented ML algorithms and to further automate the experimental procedure so that less human intervention is needed during experiments.

The Jetson board is physically mounted inside the experiment rack that also houses the rest of the computing and control units. It is directly wired into the CAN bus 2, which enables a direct connection to the High-Voltage module. The updated CMOS camera is also connected to the Jetson board. The NVIDIA Jetson board operates with an Ubuntu 22.04 LTS-based Linux root filesystem hosting a Python application responsible for experiment control. As a basis for the application and its graphical user interface, Tkinter [73] is used. The developed application allows for a comprehensive camera system control including a downscaled and contrast-enhanced camera live feed and real-time asynchronous image storage in a separate thread to ensure that no experimental data is lost. In addition, a frame counting mechanism recognizes lost frames and prevents the camera image buffer from overflowing. The metadata of each image in the form of the image number and its time-stamp is saved within the filename following previous conventions and ensuring backward compatibility. For communication to the HCI the Jetson board uses the unused CAN ID 0x020059. This way, the camera recording can also be controlled using the conventional experiment scripts and synchronized to the experiment procedure, making the Jetson board compatible with the previous scripts. For on-device analysis the following ML algorithms are implemented: particle detection using a U-Net [60], particle tracking using a SOM [74], particle string detection using an asymmetrical kernel CNN [62] and the detection and classification of local crystalline structures [75]. The algorithms are implemented in a runtime-optimized manner using multiprocessing techniques and detached in separate threads to prevent interferences with the experiment control. The algorithms are based on TensorFlow [76] and the CNNs are implemented and executed using a Float16 mixed-precision quantization, as described in section 1.2.6 to further accelerate the inference. This way, only the matrix multiplications and tensor operations are processed in reduced Float16 precision while all model weights, activations, optimizations, losses as well as input and output data are kept in full-precision (Float32). The overall model accuracy therefore does not undergo any measurable decline, while the inference time of the first particle detection U-Net and the string detection network combined, when implemented on the GPU, decreases by 41.4 % with FPS increasing by 70.5 %. Further quantization modes,

such as a more aggressive 8-bit integer quantization, including input and output data start delivering measurable decreased model accuracy, for example with the particle detection U-Net detecting particles, which are not present in the original data. Since such more aggressive quantization modes do not imply any further acceleration of inference, because they are not optimized for this GPU hardware, they are not considered further. This way, comprehensive analyses of complete experiment runs are possible directly on-device enabling for example fast evaluations in between two succeeding parabolas on parabolic flights.

Additionally, a direct control of the electric field duty cycle δ is implemented to enable manual and automated control over the particle movement during the experiment. The particle detection and tracking algorithms are implemented to work in real-time on image sections to automate the process of capturing the particles in the field of view of the cameras, as well as stabilizing their movement to come to a complete stop or to exhibit a constant velocity along the tube axis using the calculated velocities and a Proportional–Integral–Derivative (PID) controller to adjust the duty cycle δ. This way, a complete experiment control, process automation and data analysis is possible without directly affecting the intensity and frequency of the high-voltage electric field. Furthermore, a two step activation process ensures that operations on the high-voltage module are only executed, when authorized by the user. The implemented hardware and software integrations were successfully tested in two DLR parabolic flight campaigns in campaign 42 [77] and 44 [78].

1.4 Research Context of Publications

With its current popularity, AI and ML algorithms in general are increasingly used for applications in which their added value is almost negligible or even serves only for marketing purposes. One goal of this cumulative dissertation is to show the real value and benefit of ML algorithms, when applied appropriately. With complex plasmas offering a platform that allows a direct observation of a wide range of physical phenomena and resolves individual particle dynamics, substantial opportunities as well as challenges arise for data analysis. In this work, selected challenges in the data analysis of various physical phenomena are identified and addressed using ML algorithms that are adapted and optimized based on the underlying physical principles. This includes a detection and investigation of string formations due to ER effects in complex plasmas, a particle tracking algorithm that is optimized for analyzing fast particle flows with an outlier detection mechanism for improved results and a neural network trained on recognizing turbulent particle flows in individual particle trajectories. During algorithm development and network training the physical processes were taken into account to further optimize the algorithm's performance for complex plasma analysis. This is achieved, for example, by reshaping the convolutional kernels of the encoder-decoder network to account for the directed formation of particle strings or by using training datasets generated from turbulence simulations. As a result new types of analyses are enabled or significantly improved, allowing large image datasets to be evaluated even under poor and noisy imaging conditions. Upon implementation, these methods also provide substantial inference time advantages and can be deployed directly at the experiment. The main goal of this work is therefore to develop and optimize ML algorithms for an accurate and fast evaluation of complex plasma experimental data and to implement them directly into the PK-4 experiment. The key results of this dissertation are presented in the following three publications, which constitute the central body of this thesis and focus on physics-informed algorithm development considering particle string detection, particle tracking and turbulence detection.

1.4.1 Publication I

The first publication focuses on string formation in ER complex plasmas. When using an AC electric field to retain the particles in the cameras field of view, particles no longer arrange isotropically but form string-like structures aligned with the ion flow. These strings represent a distinct phase with reduced symmetry and play a key role in understanding ER behavior in plasmas. From a physical point of view, the challenge lies in the fact that strings do not have a strong local signature. A particle in a string is typically defined by only one or two neighbors and short accidental alignments can also appear in otherwise isotropic configurations. This weak symmetry makes rule-based classification unreliable and highly sensitive to thresholds and definitions.

In this work, a CNN with an encoder-decoder structure based on a U-Net is applied to detect strings on a global image level. The model does not decide particle-wise based only on local neighborhoods but processes the entire particle configuration at once and

learns the macroscopic appearance of a string. An important design choice was to include the known physical anisotropy of the system directly in the network by using deformed anisotropic convolution kernels aligned with the electric field direction. This way, physical knowledge about ion flow directions is embedded into the architecture itself. Similarly, the training process is physics-guided as well. Since no labeled particle string data is present for training synthetic training images are generated that represent the geometric and statistical properties of experimental particle strings. This is done by distributing string-like particle formations in binary training images. These display random isotropic particle distributions in one case and a mixture of isotropic particle distributions with string-like formations in the other case. The image labels are generated by only reproducing the particle strings in binary lines on the labels. Therefore, the neural network learns to recognize particle strings and ignore isotropic particles. For performance evaluation an experimental test dataset is manually labeled and the accuracies of the present and preceding string detection algorithms are determined.

Next to the immediate evaluation of all global particle strings, this approach delivers further information over the formed particle strings, such as their length and vertical expansion. With an accuracy of over 95 % this publication establishes a new ML algorithm that drastically outperforms conventional methods and preceding ML approaches by implementing the physical structures into network design and training data generation. The presented string detection method further serves as a foundational analysis tool for subsequent investigations of ER effects and DAW dynamics, where reliable identification and characterization of particle strings is essential for correlating structural ordering with plasma parameters and wave excitation [17].

1.4.2 Publication II

Publication II addresses a more fundamental problem of complex plasma analysis: a reliable particle tracking algorithm. Next to the localization of particles on the image the tracking of individual particles over a series of images is the basis for almost all further analysis, since structural and dynamical properties all rely on correct particle trajectories. Here, a SOM is used to match particles between successive images. Unlike preceding conventional particle tracking methods, the SOM does not depend on predefined search radii and flow assumptions. The SOM adapts itself to the data and is able to handle large particle displacements per timestep and complex particle flow patterns. However, the SOM algorithm does depend on predefined hyperparameters and its performance drops for high-velocity analyses, as with all particle tracking approaches.

In this publication, the influence on SOM hyperparameter settings is studied systematically using synthetically generated particle flows that cover a wide range of physically relevant flowing scenarios. This way, a physics-informed calibration of parameters is provided and an automatic hyperparameter adjustment is implemented into the particle tracking algorithm. As a result, this SOM implementation enables automated particle tracking that is optimized for high-velocity particle flows and delivers increased performance compared to conventional methods.

In addition, a second ML model based on an LSTM network is introduced to identify incorrect particle trajectories after tracking, since faulty particle matches are unavoidable but might have large impacts on the outcome of the physical evaluations. The idea is that physically plausible trajectories have characteristic temporal structures, while wrong matches often show unphysical jumps and irregularities. The LSTM is trained on synthetic particle flow data with inserted faulty particle matches and known ground truth. This way, the network learns to distinguish physically consistent and inconsistent particle motion without depending on fixed thresholds.

This publication presents a comprehensive and powerful framework for particle tracking and outlier detection. Additionally, this approach is expandable to for example three-dimensional data and applicable to further experiments and particle flow setups, where a robust particle tracking algorithm is needed. Beyond its use for trajectory reconstruction and outlier detection, the SOM-based matching approach forms the basis for a wide range of subsequent analyses. In particular, it enables reliable velocity and acceleration measurements in high-velocity particle motion and thus provides the essential input for quantitative studies of ion drag forces [79], DAW dynamics with ER effects [17], electric field determination in the plasma sheath region [80], particle tracking for turbulence detection [81] and supports three-dimensional particle position reconstruction from two-dimensional image scans for the classification of crystalline structures [75, 82].

1.4.3 Publication III

The third publication investigates the highly dynamic and multi-scale phenomenon of turbulence. In complex plasmas, turbulence does not necessarily follow classical criteria such as high Reynolds numbers. At the particle level it is not obvious how to define a sharp boundary between turbulent and non-turbulent motion. Instead of relying on global field-based quantities, turbulence is defined as a classification problem on individual particle trajectories in this work. An LSTM network is used to analyze a time series of particle bound features including velocity, acceleration, jerk, curvature and angle change. These features are chosen to be invariant under translation and rotation and to be directly computable from experimental trajectories without requiring spatial averaging or field reconstruction.

The training data is generated synthetically both from turbulent and non-turbulent flow fields. The trajectory labels are defined at the level of the underlying flow field. The training, validation and test data include a wide variety of different turbulent and non-turbulent flow fields, which are excluded separately to either training or test data. This way, the network is able to classify the particle trajectories as turbulent or non-turbulent solely based on their trajectory bound features and generalize to previously unseen flow configurations. An ablation study of input features demonstrates the importance of the feature selection and especially the importance of the angle change and jerk for turbulence classification.

The high performance on synthetically generated test data is confirmed by application to simulation data of the obstacle-driven turbulence in the PK-4 setup based on the simulations in [54]. The network is able to classify particle trajectories that are of high TKE and vorticity as turbulent without access to such data. This further confirms the

ability of the neural network to generalize to new and unseen flows. This way, a novel approach to trajectory-based turbulence detection for complex plasmas and comparable systems is opened up. In addition to the results presented in the associated publication, further simulations based on [54] are performed for varying parameters such as pressure, Langevin thermostat and external driving force to represent the dependence of onset of turbulence on these parameters. The analysis results are shown in table 5.0.1 in chapter 5.

1.4.4 Extension within Dissertation

Taken together, the three publications illustrate a progression from spatial pattern recognition over adaptive tracking to time-resolved classification of complex dynamics. Across all three, the integration of physical reasoning into the design of the ML methods and training data is essential. The dissertation extends this work into another main direction. Beyond the conceptualization and development of ML algorithms their practical integration into the PK-4 experiment is addressed. This includes the selection of suitable embedded hardware, the adaptation of network architectures to limited computational resources and the coupling of real-time analysis with experimental control. Altogether, this dissertation advances both complex plasma research and ML in physics by showing how physical insight can be embedded directly into algorithm design and training. The developed methods have already led to new scientific results and enable novel ways of analyzing particle-resolved data. In addition, the PK-4 experiment has been extended by a flexible ML framework that was successfully implemented and tested on embedded hardware. This makes ML an integral part of the experimental setup, enabling faster evaluation, automation and new experimental possibilities.

Chapter 2

Publication I: Enhancing particle string detection in electrorheological plasmas using asymmetrical kernel convolutional networks

Published as:
Klein M., Dormagen N., Dietz C., Thoma M. H. & Schwarz M. (2024). Enhancing particle string detection in electrorheological plasmas using asymmetrical kernel convolutional networks. Machine Learning: Science and Technology, 5(2), 025050.

Author Contributions:

M. Klein:
Conceptualization, Methodology, Software, Validation, Formal Analysis, Investigation, Resources, Data Curation, Writing — Draft Preparation, Review and Editing, Visualization

N. Dormagen:
Conceptualization (supporting), Data Curation (supporting), Writing — Review and Editing (supporting)

C. Dietz:
Methodology (supporting), Writing — Review and Editing (supporting)

M. H. Thoma and M. Schwarz:
Funding Acquisition, Project Administration, Writing — Review and Editing (supporting)

IOP Publishing Mach. Learn.: Sci. Technol. 5 (2024) 025050 https://doi.org/10.1088/2632-2153/ad4d3e

MACHINE LEARNING
Science and Technology

OPEN ACCESS

RECEIVED
6 February 2024

REVISED
8 April 2024

ACCEPTED FOR PUBLICATION
17 May 2024

PUBLISHED
29 May 2024

PAPER

Enhancing particle string detection in electrorheological plasmas using asymmetrical kernel convolutional networks

M Klein[1,2,*], **N Dormagen**[1,2], **C Dietz**[1], **M H Thoma**[1] **and M Schwarz**[2]

[1] I. Institute of Physics, Justus Liebig University, Giessen 35392, Germany
[2] NanoP, THM University of Applied Sciences, Giessen 35390, Germany
[*] Author to whom any correspondence should be addressed.

E-mail: max.klein@ei.thm.de

Keywords: complex plasma, electrorheology, convolutional neural network, encoder-decoder network

Abstract

Under different plasma conditions and electric fields in a complex plasma the plasma particles organize themselves in a string-like or chain-like manner. A phase transition from string-like to an isotropic particle distribution is observed at different electrical conditions. The streaming of charged ions around plasma particles with the surrounding electric field gives the plasma its electrorheological properties. The visibility of individual particles in a complex plasma opens up the opportunity to examine properties and phase transitions of such electrorheological fluids in detail. Because of the limited one-dimensional symmetry, determining the configuration of a particle and recognizing strings in particle distributions is not always straightforward. Several approaches have already been used to analyse particle clouds while either considering each particle locally or considering the particle cloud as a whole without providing information about single particle configurations. This paper presents a new machine learning approach that takes advantage of particle distributions over the entire particle cloud and detects all string-like particles at once, using a convolutional neural network in form of an encoder-decoder network with asymmetric kernel convolutions. This not only enhances the result quality but also accelerates the evaluation process, possibly enabling real-time analyses on electrorheological phase transitions, while achieving an accuracy of over 95% on manually labelled data.

1. Introduction

When micron size monodisperse particles are introduced to a plasma, a complex plasma is formed and the particles interact with the electrons, ions and neutral gas atoms. With electrons being much hotter and therefore faster than ions and neutral gas atoms in low-temperature plasmas, that normally forms the background for complex plasmas, the high amount of electrons impacting on the injected particles charge them up negatively. Due to this charging effect long range Coulomb forces arise leading to strong interactions among the particles [1]. When illuminated with a laser particles in a thin two dimensional plane of the cloud are made visible and can be recorded with a camera. With particle distances of typically more than 100 μm single particles and their movements are resolvable. This strongly coupled system, therefore, is the basis for simulating many body systems for studying various physical fundamentals, such as convection flows [2], turbulence [3] and phase transitions between the liquid and crystalline state of matter [4].

The Plasmakristall-4 (PK-4) experiment is a setup, where such complex plasmas can be generated and investigated. It consists of a Π-shaped glass tube, which is evacuated and filled with a noble gas, such as Argon or Neon. Electrodes on either side of the glass tube exert a high-voltage electric field onto the low pressure noble gas, ionizing it partially and generating a low temperature plasma. Multiple particle injectors provide monodisperse particles to the plasma, creating the complex plasma. From the sides of the glass tube a planar focused laser illuminates a plane of particles in the plasma, which are then recorded by cameras. The electric field for plasma generation in this case is a DC field. The negatively charged particles, therefore, move according to the direction and strength of this field. To stop the movement of particles in the field of view of

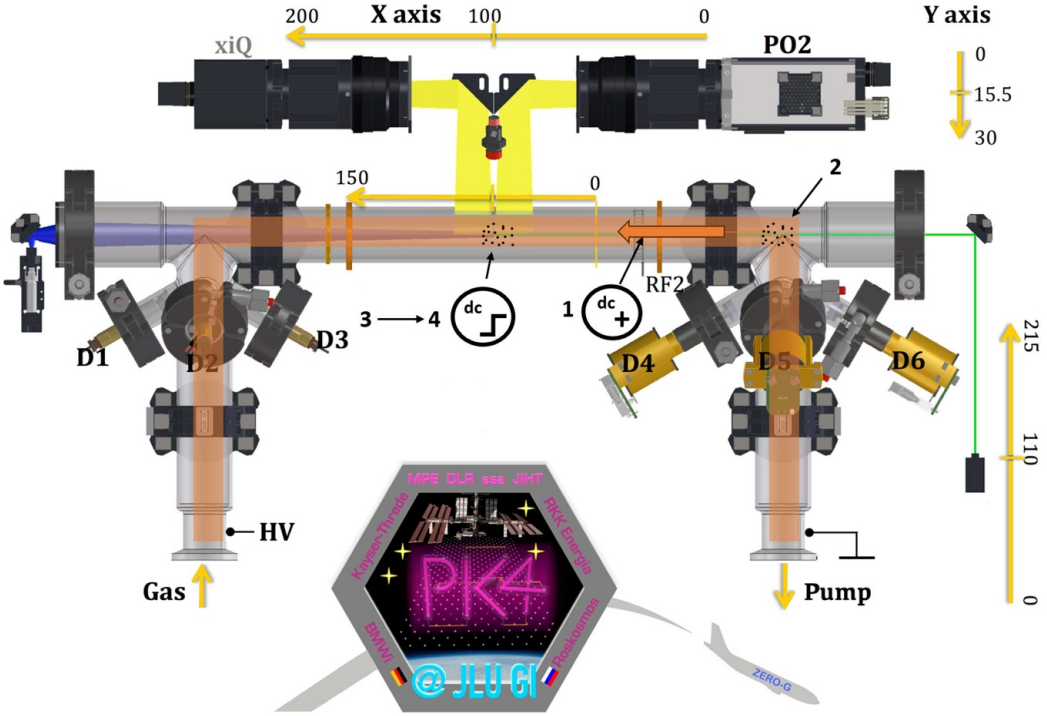
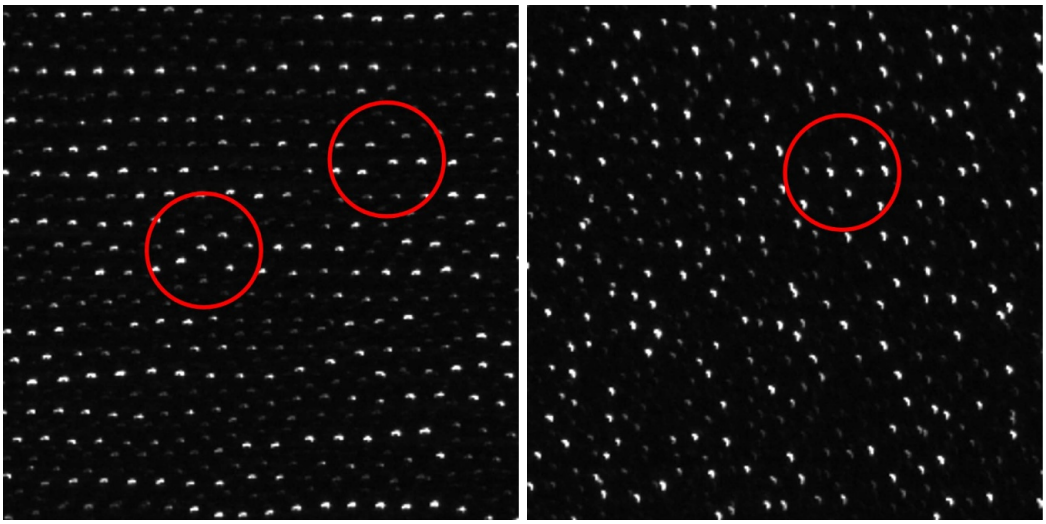


Figure 1. This is a sketch of the PK-4 experimental setup provided by the Thoma research group of the JLU Giessen. The Π-shaped plasma tube, the particle observation cameras (xiQ/PO2) and the devices for manipulating the plasma particles are shown. D1 to D6 denote the different particle injectors to insert plasma particles.

the cameras, the DC field is turned into a high frequency AC field of around 1 kHz. This frequency is chosen to be high enough so that the heavy and inert particles cannot react to the fast changing fields and thus only experience the time averaged electric field, which is zero, while the smaller electrons are accelerated back and forth to further ionize neutral gas atoms and maintain stable plasma conditions. With particles in the field of view, several manipulation options, such as a ring electrode, radio frequency coils, a thermal element and a manipulation laser are available to exert disturbances onto the particles [5]. A sketch of the PK-4 experimental setup is shown in figure 1. Under laboratory conditions the particles tend to gather at the bottom of the plasma tube, since the gravitational force pulls them down. For an undisturbed and uniform distribution of particles these experiments are usually conducted under micro gravity at drop towers, parabolic flights or on the International Space Station.

Electrorheological properties have been observed in complex plasmas [6]. An electrorheological fluid is a suspension of non-conducting particles in an isolating fluid. Applied external electric fields have a drastic influence of the fluids properties [7], as for instance the fluids viscosity can be increased by many orders of magnitude by increasing the external electric field. This is due to the arising attractive forces in the direction of the electric field. This process is generally very fast and reversible. Similar properties were observed in complex plasmas. In a DC electric field, electrons and ions move in accordance to the field in opposite directions. When particles in the plasma are levitated against gravity they create a wake potential in the ion flow with several minima and maxima [8]. This results in non-Hamiltonian attractive forces onto other particles located in this potential, since the force acts only onto the subsequent particle and not also in the other direction. Therefore, particles have a tendency to align in the direction of ion flow. When an AC electric field is applied in PK-4 to keep the plasma particles in the field of view of the cameras, an analogue alignment of particles takes place. With ion flows changing rapidly in direction, the time averaged wake potential of the now stationary particles is symmetric in the direction of electric field. A phase transition between isotropic and string-like configurations in these electrorheological plasmas has been observed when continuously changing the electric field from DC to AC [9].

To investigate phase transitions in electrorheological plasmas a determination method of the particles string-like configuration is needed. This determination is not trivial task. Several different approaches have already been made in this regard, using global measures like the anisotropic scaling index [6, 10, 11], or using local measures, such as the bond orientational order [12] or irreducible minkowski tensors [13], which all deliver insufficient results. A newer approach using machine learning [9] delivers much better results on



(a) Image with strings (b) Image with isotropic distribution

Figure 2. Exemplary images of real string-like and isotropic particle distributions. On the left a string-like particle configuration is visible. The red circles mark areas, where strings come together that make the string detection difficult for certain particles. On the right an isotropic particle configuration is visible with a string-like formation marked in red. Distinguishing a string in an isotropic configuration from randomly evenly spaced particles is challenging.

local string-like detection. This paper presents a new and improved machine learning approach, providing even better string-like detection, while considering all particles on a global scale simultaneously and connecting particles that are forming a string.

2. Machine learning

Strings forming in a two dimensional image of particles are easy to identify with the bare eye. But when it comes to evaluating single particles, it is noticeable that it is very hard to decide on many particles. Especially, when not only clear strings or isotropy are present but a mixture of both. This is due to strings only showing a comparably weak symmetry. With only two neighbors that account for a particle being in a string compared to for example six neighbors in a hexagonal structure. The last particle at the end of a string even only has one neighbor that is defining it to be a part of the string. Other undefined fundamentals such as the minimum number of particles that are needed for a structure to be a string and the maximum connection angle for particles to still be part of a preceding string make decisions even harder. In some cases strings come together at the end which raises additional challenges for a correct differentiation. In addition, isotropic particle configurations partly consist of randomly arranged particles in short strings, which makes it hard to distinguish string formations from other formations. Exemplary images of such particle configurations are shown in figure 2 for visualisation. Therefore, analogue approaches in general face many difficulties. As already shown, machine learning is a promising way of overcoming these problems [9]. Additionally, strings in complex plasmas can often become lengthy, which is why local investigations focusing only on proximate surroundings of a particle ignore helpful information that could potentially be incorporated into the decision-making process. A global machine learning approach therefore shows great potential.

To realize this, an analogical method to [14] is chosen, where the original image is fed through a convolutional U-Net [15], that is trained to binarize the image. The neural network outputs a binary image of original size representing a mask, that shows, where the network found particles. This way, only the positions of these masks need to be read out to localize the particles. In a similar way, the neural network presented here uses an image of the entire particle cloud as input and outputs a binary mask. Hereby, the neural network masks all present strings and gives out an empty image, when no strings are present. Comparing the generated mask to all particle positions, the string-like or isotropic configuration of each particle can be easily read out. Furthermore, length and vertical elongation of each string can be analyzed.

2.1. Machine learning model

To achieve this binarization we make use of a fully convolutional network with an encoder-decoder architecture. The neural network is split up into two parts: an encoder and a decoder. The encoder

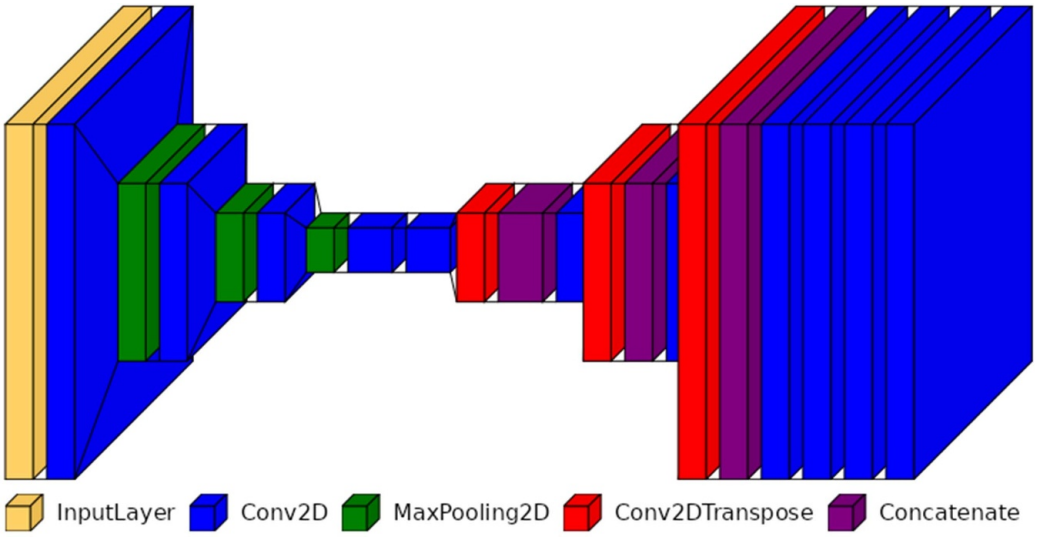


Figure 3. Illustration of the neural network used for string detection created with [16]. The concatenation layers connect the preceding transpose layers with the foregoing convolutional layers of the same size.

constitutes the first part of the network and is designed to sample down the input to a certain degree. Afterwards, the input image is sampled back up to original size as seen in figure 3. This is meant to reduce the original image to just the necessary information that is needed. Using this information, the neural network is then able to reproduce a mask of the image highlighting all strings. To regain the positional information of the input image after downsampling, several forward concatenations are built into the model after each sampling step. The downsampling steps are realized by two dimensional convolutions, followed by activation functions, and finalized by two dimensional pooling layers. The upsampling is then realized by two dimensional convolutional layers, activation functions and followed by transposed two dimensional convolutional layers. These layers form the basis of a U-Net model [15].

The disadvantage of this U-Net is its immense size. With all downsampling steps the number of channels is increased, reaching a value of 1024. This is not just memory intensive but also computationally intensive and increases latency times of the neural network. Especially, when large images are fed through the network, large memory and computing resources are needed. The high amount of channels is not needed in this case and is therefore reduced starting with 4 channels for the first convolution, doubling in size every convolution until 32 channels are reached in the bottleneck of the network and then halving in size each convolution until 4 channels are reached for the last convolutions. This largely decreases the size of the network from 389 521 to 25 581 trainable parameters, while simultaneously decreasing latency times. Furthermore, all particle strings form in the direction of electric field, which in the case of PK-4 means, that all particle strings form horizontally. With these directed phenomena to be detected, asymmetric kernel sizes for convolution, deconvolution and pooling layers are deployed to account for this asymmetry and to get better results. Thus all kernel sizes are also elongated horizontally with decreasing kernel elongation towards the network constriction. This enables a broader activation of neurons to increase the recognition of strings. An illustration of the used neural network is shown in figure 3.

2.2. Training

The finalized convolutional network needs an appropriate training before being applicable. Sufficient training data is needed for training. In the case of this binarizing encoder-decoder network suitable training images with corresponding labels, in form of binary masks, are required. While enough real image data of complex plasmas with isotropic and string-like particle distributions is present, no fitting string masks to operate as training labels are available. With several thousands of training images and masks needed for training, labelling images by hand would not just be too inaccurate but also too elaborate. Therefore, artificially created data is used for training.

The process of generating training images in a way that the output string masks reliably detect string formations is a trial and error process. To display a string on the input image, randomized coordinates for a single particle in the field of view are generated. Next to that, six additional particle coordinates are generated

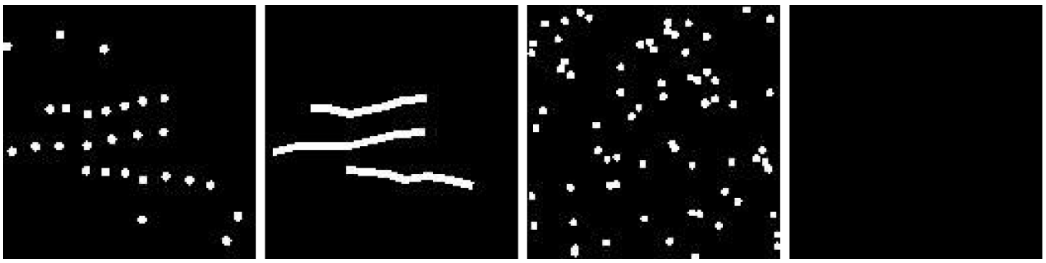


Figure 4. Example training images and labels. The according labels only mask the generated strings, ignoring the randomly placed particles and leaving the labels for isotropic distributions empty. Both input image and label are binary images. Image (a) shows an input image containing 3 strings, image (b) shows the according label containing only the strings, image (c) shows an isotropic input image and (d) shows the according empty label without any strings.

to be situated on the left and on the right side of the initial particle in a string formation. These coordinates are generated with a fixed distance in horizontal direction.

A limited random offset in vertical direction is also added for each particle on the side to make the generated string not entirely straight. The x and y distances to neighboring particles are additionally superimposed with a certain degree of random noise in x and y direction to create more realistic string formations. In addition to that, the whole string is rotated by a limited random angle, to make the string not entirely horizontal. This way an artificial string is created. With a single string visible on the input image, the network would take large amounts of images for sufficient training. Therefore, and to make the input images even more realistic, randomly chosen either two or three strings are generated per image in a limited random vertical distance to each other, preventing overlapping. Looking at real string formation data, some strings show a sort of kink, where two string ends come together. To further improve the realness of artificial training images, this kink is also introduced with a small vertical offset in the string. All limits for random values and noises are chosen to resemble real data as much as possible.

Next to training the network on recognizing string formations it also needs to be trained to recognize isotropic particle formations and not to mask these in the output mask. To address this, randomly distributed particles are added into the areas, where no strings are generated. Furthermore, additional images with randomly distributed particles containing no strings at all are also generated. All particles are displayed on images of 128 to 128 pixels in size. This resolution has shown to be high enough for successful training, while being small enough to be handled in large numbers by a conventional GPU.

The input images are generated by displaying particles at the position of generated particle coordinates on the image. When particle sizes, form and brightness are chosen to resemble real image data from preceding PK-4 campaigns, the low input values are not decisive enough for recognising particle strings. Therefore, binarized particle images are used as input for the neural network. The original image data is fed through a convolutional network, as described in [14], outputting a binarized mask of the original image, where all pixels containing a particle are set to 1, while all others are set to 0. This process is used for particle localization on PK-4 data and is therefore already available at no further computational costs. In a similar manner, the artificially created training images consist of a binary mask of all particles at the position of the generated coordinates in varying sizes.

The according labels are then generated, by simply displaying four pixel wide connection lines between the coordinates of all particles, that are situated in a string. The thickness is chosen so that the labels are distinct enough to be reconstructed by the network and to make sure that the marked particles are reliably covered by the output mask. The string masks are also displayed binary with value 1 on a background with value 0. This way, only strings are visible on the labels and all labels for training images with only an isotropic particle distribution are completely empty. An example of the artificial training images is shown in figure 4.

Best results are then achieved by training with 3000 training images containing strings and 3000 images containing no strings and only isotropic distributions. These 6000 images are shuffled and used in training for 20 epochs with a batch size of 50 images. The Adam optimizer with a constant learning rate of 0.001 was used. The loss was calculated with a weighted binary cross entropy function as in

$$\text{Loss} = -\frac{1}{N}\sum_{i=1}^{N}\frac{w_1 \cdot T_i \cdot \log\left(P_i+\epsilon\right) + w_0 \cdot \left(1-T_i\right) \cdot \log\left(1-P_i+\epsilon\right)}{w_0 + w_1} \tag{1}$$

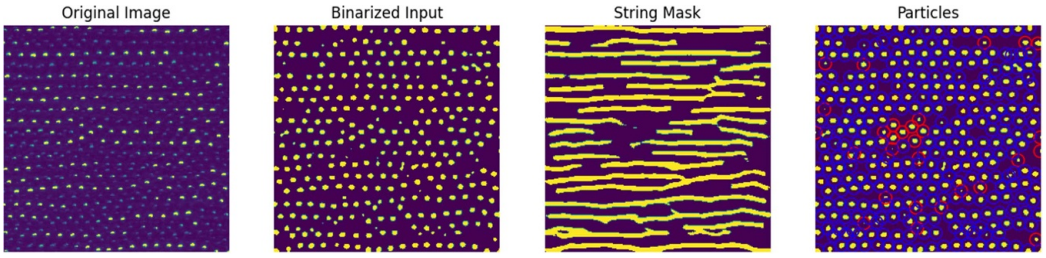


Figure 5. Illustration of the *original image*, the *binarized input*, the generated *string mask* and finally the analyzed *particles*. The *original image* is used as input for the first convolutional network. This networks binarized output is used as *binarized input* for the convolutional network that detects string formations. The output of this string formation network is the *string mask* to mark all *particles* that are inside a string formation, with all particles inside strings marked in blue and all particles outside of strings marked in red.

with T_i and P_i being the truth and predicted value of each pixel, N being the number of pixels and ϵ being a small value of 10^{-4}, to prevent taking the logarithm of zero. The weight of the value 1 w_1 was chosen to be 10, while the weight of the value 0 w_0 was chosen to be 1, since the amount of pixels with the value 1 is significantly lower for the used images. With this loss function the cross entropy loss for each pixel is calculated, while giving the fewer but more meaningful pixels with value 1 more weight.

3. Application and results

After training the network is ready for application. To detect strings on image data, the image first needs to be binarized and all particles need to be located, following [14]. The binarized image is then input into the neural network, that outputs the string mask. This mask can be displayed to gain a visual impression of all formed strings in the image. Next to that, the value of the string mask is read out on all coordinates, where particles are present. If the pixel containing the particle coordinate has the value 1, the particle is found to be part of a particle string and vice versa. This way all visible particles are evaluated simultaneously. To gain an impression of how the image data, input and output of the network looks it is shown in figure 5 . Furthermore, x and y elongation of coherent string mask areas are read out.

3.1. Evaluation

To evaluate the trained model it is first tested on artificial data. For this, additional artificial validation data is generated in the same way as for the training process, to gather data that is unknown to the trained network. Evaluated on 240 images the accuracy of the network is 99.25% considering the prediction of pixels in the output image, which normally tends to be high, since most pixels are background pixels. When it comes to recognizing particle formations, the model detects 96.06% of particles in the correct formation. Of the incorrectly recognized particles roughly 98% are particles, that do not belong to a string, but have been detected in one. The model therefore tends to detect more particles in strings than are present.

To confirm the high accuracy of the model on artificial data, the model needs to be verified on real data to assure the the neural network has generalized to problem and has not just adapted to the training data. Since the artificial data is of the same kind as the training data, these high accuracies were foreseeable. The strings in the training data are randomly generated to only resemble strings in real data. Real images are much more particle filled and tend to form many more and longer strings, than present in training data. Furthermore, the resolution of real images with 2048 to 2048 is much higher than on the training images. It needs to be shown, that the network has learned to recognize particle strings, by recognizing real particle strings. Because no labelled string data exists for this case, manually labelled data is used. Particles in real images of preceding PK-4 experiments are marked considering their string-likeness. This is done on 20 images with 10 images predominantly showing particle strings and 10 images predominantly showing isotropic distributions. In some cases, deciding on a particles configuration is not possible by hand, since no clear rules exist. Therefore, clearly string-like and isotropic particles are marked as such, while undecidable particles were left out of consideration. These particles are used as an estimation, whether the network is more likely to mask particles as string-like or not.

The trained model predicts 95.3% of all marked particles in real data correctly. This proves that the model has generalized the problem and is able to recognize string-like formation also on new and unknown data, that looks entirely different than the training data. Of the incorrectly evaluated particles 53% have been falsely predicted as string-like and 47% have been falsely marked as isotropic. This demonstrates, that the

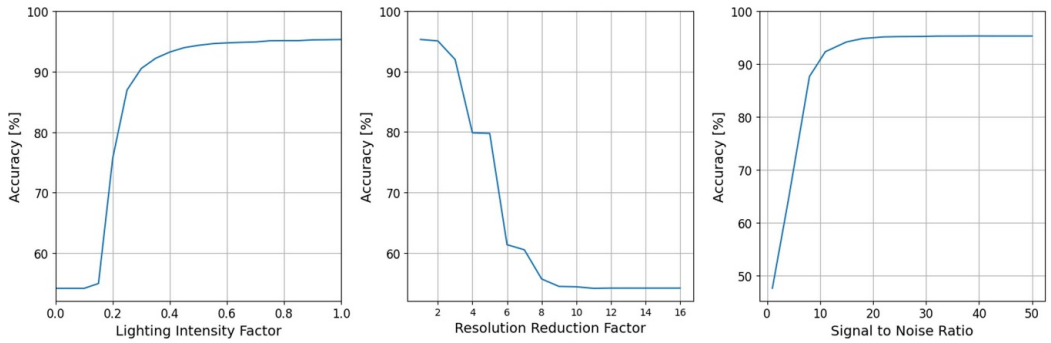


Figure 6. Accuracy test results on manually labelled data with various types of degradation, such as lower lighting conditions, image resolution reduction and Gaussian noise.

network does not prefer a configuration over the other on real data. In the manually labelled images 4.26% of particles have been marked as undecidable by hand. Of these particles 22% have been marked as string-like and 78% as isotropic. So it seems as the network rather marks particles as isotropic, when the distinction is not possible by hand.

3.2. Model limits

To verify the applicability of the model to real data, it is helpful to determine the limits to which the string detection accuracy holds up. Since the neural network uses camera image data to generate string masks, its sensitivity to input image quality, such as noise, lighting conditions, and image resolution, should be assessed. As mentioned in section 2.2, the original camera images must be fed through a convolutional neural network in a preprocessing step to binarize them, otherwise they would be too indecisive for analysis. This ensures a consistent, high signal intensity and low noise. In the following, these binarized input images are modified in different ways to assess the influence of image quality. Figure 6 shows the accuracy results of the same labelled data used for testing as above, degraded for lighting conditions by scaling the input intensity from 1 to 0, reduced image resolution by scaling the image resolution by factors of 1 (original) to 16, and image noise by adding Gaussian noise with signal to noise ratios of 50 to 1. It can be seen that the accuracy of the model drops significantly when the input intensity is more than halved, the image resolution is more than halved and the signal to noise ratio is less than 20. These results show that the sensitivity of neural networks to image quality degradation is not high. Typically, the signal to noise ratio of real PK-4 image data is between 100 and 150, which is far from the point where accuracy starts to degrade. In addition, the preprocessing step of image binarization prevents such drops in accuracy due to image quality entirely.

String detection performance has been tested and verified in a range of complex plasma conditions. Our real experimental data includes pressures from 0.4 to 1 mbar, plasma currents from 0.5 to 1 mA, AC electric field frequencies of 250 and 1000 Hz and particle sizes of 2.05, 3.4 and 6.8 μm. Changing the background plasma conditions does not noticeably change the visual appearance of the plasma particles and therefore does not have a major effect on the string detection performance. Particle sizes and densities, however, do change the visual appearance of the plasma strings. In order to cover all the different configurations, the artificial training data has been matched as closely as possible to the real data. As a result, there is no significant loss of performance during the experiments. Strings with very different particle sizes and densities could potentially lead to losses in the accuracy of string detection. In this case, the artificial training data could be extended with such examples and the model retrained to cover such cases. Other disturbances in complex plasmas, such as dust acoustic waves, can possibly distort the strings to such an extent that the model is no longer able to detect them. Plasma waves should therefore be avoided or kept to a minimum in string formation experiments.

4. Comparison to preceding methods

While the accuracy of the presented network is very high, it still needs to be compared to preceding conventional methods to verify its usefulness and improvement. To gain an overview of the accuracy of other methods, they are implemented accordingly and evaluated on the same data.

4.1. Anisotropic weighted scaling index

At first the anisotropic weighted scaling index is compared to the machine learning approach. According to [6, 10, 11] the weighted scaling index α is implemented by calculating the logarithmic derivative of $\rho(\mathbf{p}_i, r)$ with respect to r, where ρ is the local weighted cumulative point distribution of each point particle $\mathbf{p}_i$ and r being the scale parameter. With

$$\rho\left(\mathbf{p}_i, r\right) = \sum_{j=1}^{N} e^{-\left(d_{ij}/r\right)^q} \tag{2}$$

and

$$d_{ij} = \left\|\mathbf{p}_i - \mathbf{p}_j\right\| \tag{3}$$

the scaling index simplifies to

$$\alpha\left(\mathbf{p}_i, r\right) = \frac{\sum_{j=1}^{N} q\left(\frac{d_{ij}}{r}\right)^q e^{-\left(d_{ij}/r\right)^q}}{\sum_{j=1}^{N} e^{-\left(d_{ij}/r\right)^q}}. \tag{4}$$

The exponent q controls the weight of the points contribution to α based on their distance to $\mathbf{p}_i$. With increasing q points with $d_{ij} > r$ are more and more neglected, leading to a more local measure. In this case $q = 2$ has been used. This way a scaling index for every particle is calculated. The value of α is a measurement of the particles surrounding structure with $\alpha \approx 0$ suggesting a cluster-like structure, $\alpha \approx 1$ suggesting a filamentary structure, $\alpha \approx 2$ suggesting a sheet-like structure, $\alpha \approx 3$ suggesting a uniform distribution and $\alpha > 3$ indicating points in underdense regions but in the vicinity of structures [11].

Furthermore, longitudinal and transverse scaling indices $\alpha_{\parallel}$ and $\alpha_{\perp}$ can be calculated by adding an eigenvector $\mathbf{q}$ to d_{ij} as in

$$d_{ij} = \sqrt{q_x^2\left(x_i - x_j\right)^2 + q_y^2\left(y_i - y_j\right)^2}. \tag{5}$$

Since in this case all strings lay parallel in x-direction the eigenvectors $\mathbf{q}_x = (1, 0)$ and $\mathbf{q}_y = (0, 1)$ are used to calculate $\alpha_{\parallel}$ and $\alpha_{\perp}$.

The anisotropic weighted scaling index could be an indication of string-like structure in particle clouds and the differences of longitudinal and transverse scaling index distribution a classification of if the present image is filled with string-like structures or not.

4.2. Modified bond orientational order parameter

As shown in [12] the modified bond orientational order parameter

$$p_n(i) = \frac{2\pi}{(2n+1)N(i)} \sum_{j=1}^{N(i)} P_n\left(\cos\theta_{ij}\right) \tag{6}$$

can be calculated for each particle i, using a number of nearest neighbors $N(i)$ and Legendre Polynomial of the order n, with θ_{ij} being the angle between the two particles i and j. This parameter is used to illustrate structural transitions from weakly anisotropic fluids to string fluids and gives out a value of string-likeness for each particle. Since this approach is heavily dependent on the nearest neighbors, its accuracy for two-dimensional string structures is limited. In particular, the two ends of each string cannot be detected. By choosing a threshold value for $p_n(i)$ a distinction between string-like and isotropic particles can be made. In this evaluation six nearest neighbors $N(i)$ and the second order Legendre polynomial $n = 2$ was used.

4.3. Pair correlation function

Previous studies evaluating string formations in complex plasmas in three [17, 18] and two [19] dimensions make use of pair correlation functions to determine the structural properties of particles. The pair correlation function in two dimensions, expressed in polar coordinates, is given by [19]

$$g_{2D}(\rho, \alpha) = \frac{1}{\sigma_d N} \sum_{\substack{i,j=1 \\ i \neq j}}^{N} \frac{\delta\left(\rho_{i,j} - \rho\right)\delta\left(\alpha_{i,j} - \alpha\right)}{\rho} \tag{7}$$

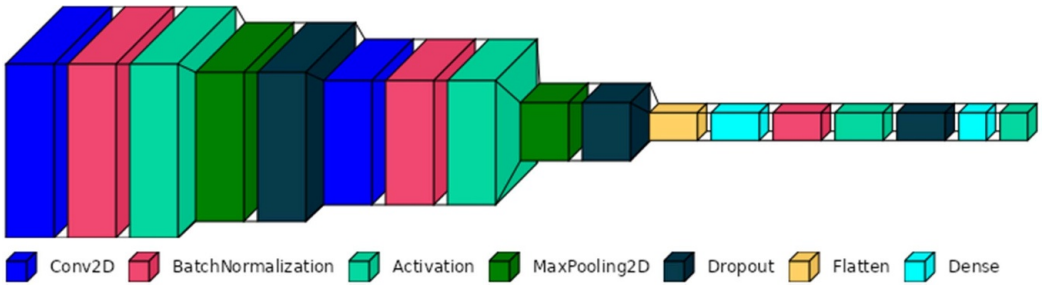


Figure 7. Illustration of the neural network used for string detection in [9] created with [16]. The input size of the network is a 24 to 24 pixel image and the output is 2 different classes.

with N being the number of particles, σ_d being the particle density, $\rho_{i,j}$ and $\alpha_{i,i}$ being the distance and angle of the vector connecting particle i and j and ρ and α being the current particles polar coordinates.

Visualizing the pair correlation function with respect to ρ and α a distinction between an isotropic and string-like ordered system is visible.

For an isotropic distribution the pair correlation function is distance and angle independent and for ordered structures maxima in the pair correlation function are visible at certain distances and angles. After estimating the ordered distance and angle regions, all particles with neighbors that have an according distance and angle can be viewed as string-like. In this case it needs to be decided, whether one string-like neighbor is enough for defining a particle as in string, while enabling possible wrong classifications, or if two string-like neighbors are needed, while then not recognizing string ends as part of the string.

4.4. Preceding machine learning approach

To analyze phase transitions in electrorheological complex plasmas a machine learning approach based on a small convolutional neural network is presented in [9]. Here, the particle positions are extracted out of the original image at first. A 24 to 24 pixel image is then generated for each particle displaying an area of 5 times the median particle distance in total. The image is binary with every pixel containing a particle being 1. This image is fed through two convolutional layers, each followed by a batch normalisation, activation, pooling and dropout layer, and finally flattened by several fully connected layers to a two dimensional output. An illustration of this neural network is shown in figure 7. This way every particle is classified into either string-like or isotropic. The network is trained on partly real and artificial data. After classifying all particles falsely categorized strings are filtered out by reviewing the neighborhood of each particle.

After training this convolutional neural network reaches an accuracy of 96.6% [9]. This evaluation is, at least partly, done on artificially created data. While this approach is very promising, it will be applied to the same manually labelled data to verify its performance.

4.5. Comparison results

The results of applying each implementation of the different methods on the same set of manually labelled data is shown in table 1. Applying the anisotropic weighted scaling index method to the real data it is quickly visible that no clear distinction between string-like and isotropic particles can be made. Even modifying the parameters of index calculation does not give results that could be decisive by some kind of threshold particle-wise. Therefore, no result is shown in table 1. Calculating the longitudinal and transverse scaling indices for the whole particle cloud image-wise, the delta between both indices is a good indication of if the particle cloud is filled with strings or not. But in this case no information about individual particles is given. Using the modified bond orientational order parameter to recognize strings is equally prone to errors. While in this case a connection between the order parameter and a particles string-likeness is perceptible, the accuracy is just slightly higher than 50% which can be compared to just guessing the particles configuration without seeing it. Even adjusting the decision threshold and the number of neighbors used in the calculations does not improve the results.

The pair correlation function method delivers much more useful results in comparison. Table 1 shows both outcomes for implementations considering one and two nearest neighbors. With over 80% accuracy the method using only one neighbor performs best. This way the last particle of each string is also detected as string-like. On the other hand this way also some number of false classifications are made with particles that do not form a string but coincidentally have a neighbor in the fitting distance and angle. The smaller convolutional neural network approach from [9] performs comparably well with over 77% accuracy, while not reaching the 96.6% accuracy that was evaluated after training.

Table 1. Comparison of string detection accuracy of different approaches on manually labelled data. The values correspond to the methods using the encoder-decoder network with asymmetric kernels presented in this work, the anisotropic weighted scaling index, the modified bond orientational order parameter, the pair correlation function using one neighbor and two neighbors and the preceding machine learning approach with a smaller convolutional neural network [9].

Method	Accuracy
Encoder–Decoder with asymmetric Kernels	95.24%
Anisotropic Weighted Scaling Index	—
Modified Bond Orientational Order Parameter	52.66%
Pair Correlation Function (1 Neighbor)	81.83%
Pair Correlation Function (2 Neighbors)	77.21%
Convolutional Neural Network [9]	77.11%

The machine learning approach using a convolutional encoder-decoder network with asymmetrical kernels presented here performs best in comparison. While the improvement in accuracy is very apparent and distinct, it is crucial to consider, that these yielded lower values could be influenced by differences in implementation details and the adaptation on the new data.

4.6. Computational efficiency

Comparing computational efficiency is a difficult task, since it depends very much on the code implementation and also on the hardware used, since in this case the conventional analysis methods are executed on a CPU and the neural networks on a GPU. Besides these differences, the main advantage of the presented approach is its independence from the number of particles. While the presented neural network has a fixed interference time of only about 0.03s on a NVIDIA RTX3060M graphics card, including the preprocessing convolutional network, for fullsize images (2048 $\times$ 2048 pixels), the previous methods analyse the particles one by one, which has scaling disadvantages. On smaller images (512 $\times$ 512 pixels) with about 2000 particles, these methods already have latencies of 2 to 6s per image. When scaled up to fullsize images with easily over 20 000 particles, the latency times rise to over a minute on a CPU core of an Intel i7-12 700 H. Of course, these processes could be accelerated using different implementations and multiprocessing techniques, but these approaches would always suffer from scaling effects. Therefore, the new approach shows a significant acceleration compared to previous methods.

5. Conclusion

In conclusion this paper presents a new machine learning approach to recognize string-like particle formations in complex plasmas using a convolutional encoder-decoder network with asymmetric kernels. This new approach improves the accuracy of string detection immensely compared to preceding methods up to above 95% accuracy on manually labelled data, while also enabling a global evaluation, considering all particles of the cloud at once and not just particle-wise, simultaneously accelerating the whole evaluation process and possibly enabling real-time video analyses. A convolutional U-Net was adapted, reduced and improved to suit the anisotropy of the present string formation phenomenon. Adequate training data and labels were artificially created to ensure good network training. Real complex plasma image data were manually labelled to validate, that the trained network has generalized the problem and is applicable to real data and to assess the real performance of the network.

In direct comparison to preceding methods on similar data the networks performance is superior. This improvement opens up new possibilities in particle string evaluation and could possibly improve previous findings in this field of science. Simultaneously, the new approach makes further analyses of the present strings more simple, because length and vertical elongation of the strings can be easily read out of the string mask, without any further complex calculations necessary.

The current method is limited to two dimensional image data, but could also be applied to three dimensional scans of particle clouds, providing further information about plasma crystals. Since the PK-4 setup is limited to two dimensional images of particle planes within the particle cloud, a scan of the particle cloud can be performed to obtain three dimensional information about the particles. The resulting two dimensional string masks could be stacked, taking into account the scanning speed and camera frame rate, to obtain a three dimensional string mask. Similar experiments have already been carried out using preceding analysis methods [17, 18] to obtain more information about the boundary regions of particle strings and their density distribution. A long-term evolution of the string-like order is observed, showing a redistribution of particles that significantly modifies the plasma conditions. Furthermore, crystalline structures and phase transitions such as melting and recrystallisation [19] could be observed in three dimensions, providing further information on electrorheological fluids.

The newer complex plasma setup COMPACT [20] could potentially improve insights into such dynamic processes because it can provide time-resolved three dimensional particle positions using four stereoscopic cameras rather than slow two dimensional scans [21]. These three dimensional particle coordinates could be analyzed in a completely different way, possibly using a three dimensional convolutional voxel based neural network.

In future work this neural network could be extended. As of now, binarized input images that are generated by another U-Net are used is input for the network, making two neural networks necessary for application. These networks could be conjoined to further simplify and accelerate the whole process.

Data availability statement

All data that support the findings of this study are included within the article (and any supplementary files).

Acknowledgments

This work is supported by the German Aerospace Agency (DLR). We thank the research group around Markus H Thoma and M. Kretschmer at Justus Liebig University Giessen for providing the data used in this study and the German Aerospace Society for providing powerful PCs in accordance with the funding decision, which facilitated the studies. This project was supported by the German Federal Ministry of Economic Affairs and Climate Action under Contract No. 50WK2270B.

ORCID iD

M Klein https://orcid.org/0009-0009-9596-7671

References

[1] Bonitz M, Henning C and Block D 2010 *Rep. Prog. Phys.* **73** 066501
[2] Schmitz A S, Schulz I, Kretschmer M and Thoma M H 2023 *Microgravity Sci. Technol.* **35** 13
[3] Schwabe M, Zhdanov S and Räth C 2018 *IEEE Trans. Plasma Sci.* **46** 684–7
[4] Thomas H M and Morfill G E 1996 *Nature* **379** 806–9
[5] Pustylnik M *et al* 2016 *Rev. Sci. Instrum.* **87** 093505
[6] Ivlev A *et al* 2008 *Phys. Rev. Lett.* **100** 095003
[7] Winslow W M 1949 *J. Appl. Phys.* **20** 1137–40
[8] Ishihara O and Vladimirov S V 1997 *Phys. Plasmas* **4** 69–74
[9] Dietz C, Budak J, Kamprich T, Kretschmer M and Thoma M H 2021 *Contrib. Plasma Phys.* **61** e202100079
[10] Ivlev A V, Brandt P C, Morfill G E, Rath C, Thomas H M, Joyce G, Fortov V E, Lipaev A M, Molotkov V I and Petrov O F 2010 *IEEE Trans. Plasma Sci.* **38** 733–40
[11] Räth C, Bunk W, Huber M B, Morfill G E, Retzlaff J and Schuecker P 2002 *Mon. Not. R. Astron. Soc.* **337** 413–21
[12] Brandt P, Ivlev A V and Morfill G E 2011 *J. Magn. Magn. Mater.* **323** 1368–71
[13] Schröder-Turk G E, Mickel W, Kapfer S C, Schaller F M, Breidenbach B, Hug D and Mecke K 2013 *New J. Phys.* **15** 083028
[14] Dormagen N, Klein M, Schmitz A S, Thoma M H and Schwarz M 2024 *J. Imaging* **10** 40

[15] Ronneberger O, Fischer P and Brox T 2015 U-net: Convolutional networks for biomedical image segmentation *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th Int. Conf. (Munich, Germany, 5–9 October 2015)* (*Proc. Part III*) vol 18 (Springer) pp 234–41
[16] Gavrikov P 2020 visualkeras (available at: https://github.com/paulgavrikov/visualkeras)
[17] Pustylnik M *et al* 2020 *Phys. Rev. Res.* **2** 033314
[18] Mitic S *et al* 2021 *Phys. Rev.* E **103** 063212
[19] Joshi E, Pustylnik M, Thoma M H, Thomas H M and Schwabe M 2023 *Phys. Rev. Res.* **5** L012030
[20] Knapek C A, Couedel L, Dove A, Goree J, Konopka U, Melzer A, Ratynskaia S, Thoma M H and Thomas H M 2022 *Plasma Phys. Control. Fusion* **64** 124006
[21] Himpel M and Melzer A 2021 *Mach. Learn.: Sci. Technol.* **2** 045019

Chapter 3

Publication II: Advancing Particle Tracking: Self-Organizing Map Hyperparameter Study and Long Short-Term Memory-Based Outlier Detection

Published as:
Klein M., Dormagen N., Wimmer L., Thoma M. H. & Schwarz M. (2025). Advancing Particle Tracking: Self-Organizing Map Hyperparameter Study and Long Short-Term Memory-Based Outlier Detection. Machine Learning and Knowledge Extraction, 7(2), 37.

Author Contributions:

M. Klein:
Conceptualization, Methodology, Software, Validation, Formal Analysis, Investigation, Resources, Data Curation, Writing — Draft Preparation, Review and Editing, Visualization

N. Dormagen:
Conceptualization (supporting), Investigation (supporting), Data Curation (supporting), Writing — Review and Editing (supporting)

L. Wimmer:
Data Curation (supporting), Writing — Review and Editing (supporting)

M. H. Thoma and M. Schwarz:
Funding Acquisition, Project Administration, Writing — Review and Editing (supporting)

machine learning & knowledge extraction

MDPI

Article

Advancing Particle Tracking: Self-Organizing Map Hyperparameter Study and Long Short-Term Memory-Based Outlier Detection

Max Klein [1,2,*], Niklas Dormagen [1,2], Lukas Wimmer [2], Markus H. Thoma [2] and Mike Schwarz [1]

[1] NanoP, THM University of Applied Sciences, 35390 Giessen, Germany; niklas.dormagen@nanop.thm.de (N.D.)
[2] I. Institute of Physics, Justus Liebig University, 35392 Giessen, Germany
* Correspondence: max.klein@ei.thm.de

Abstract: Particle tracking velocimetry (PTV) forms the basis for many fluid dynamic experiments, in which individual particles are tracked across multiple successive images. However, when the experimental setup involves high-speed, high-density particles that are indistinguishable and follow complex or unknown flow fields, matching particles between images becomes significantly more challenging. Reliable PTV algorithms are crucial in such scenarios. Previous work has demonstrated that the Self-Organizing Map (SOM) machine learning approach offers superior outcomes on complex-plasma data compared with traditional methods, though its performance is sensitive to hyperparameter calibration, which requires optimization for specific flow scenarios. In this article, we describe how the dependence of the various hyperparameters on different flow scenarios was studied and the optimal settings for diverse flow conditions were identified. Based on these results, automatic hyperparameter calibration was implemented in the PTV framework. Furthermore, the SOM's performance was directly compared with that of the preceding conventional PTV method, Trackpy, for complex plasmas using synthetic data. Finally, as a new approach to identifying incorrectly matched particle traces, a Long Short-Term Memory (LSTM) neural network was developed to sort out all inaccuracies to further improve the outcome. Combined with automatic hyperparameter calibration, outlier detection and additional computational speed optimization, this work delivers a robust, versatile and efficient framework for PTV analysis.

Keywords: self-organizing map; neural network; particle matching; particle tracking velocimetry; outlier detection

Academic Editor: Andreas Holzinger

Received: 7 February 2025
Revised: 24 March 2025
Accepted: 2 April 2025
Published: 17 April 2025

Citation: Klein, M.; Dormagen, N.; Wimmer, L.; Thoma, M.H.; Schwarz, M. Advancing Particle Tracking: Self-Organizing Map Hyperparameter Study and Long Short-Term Memory-Based Outlier Detection. *Mach. Learn. Knowl. Extr.* **2025**, *7*, 37. https://doi.org/10.3390/make7020037

1. Introduction

When it comes to analyzing particle flows and other fluid dynamic experiments, knowledge about the position and movement of the particles to be observed is essential. Usually, particle movements are recorded in a stationary state at various points in time, for example, in recorded video files. This poses a challenge, since not only is the exact localization of particles at every time step crucial, but the correct matching of particles over the series of images is also of great importance, since this is needed to determine the particles' velocity. The process of particle tracking velocimetry (PTV), therefore, has already been addressed before in various different ways. A good overview and comparison of PTV methodologies was presented by Dabiri and Pecora in [1]. One way to assess the suitability of a PTV technique for a certain experiment, where PTV is defined as the ratio of the mean particle displacement per image pair to the mean particle spacing (ξ), is the achievable ratio

of incorrectly matched particles to the total number of matchable particles (E_{track}). Higher ξ values mostly result in a larger number of tracking errors, E_{track}, up to a point where the results are unusable.

Recent studies in complex plasmas investigating dust acoustic wave speed [2] and ion drag have introduced the challenge of high ξ values. A complex plasma consists of a low-pressure noble gas which is ionized through the application of a high voltage and into which micron-sized monodisperse particles are injected. Due to the fast free electrons in the plasma, the particles are negatively charged, and long-range Coulomb forces arise in between the particles. When these microparticles are illuminated by a laser, they are visible to a camera that records the particles. Complex plasmas are often used to study physical fundamentals such as turbulent [3] or convective [4] flows and phase transitions between the liquid and crystalline states of matter [5] and to simulate many body systems due to the particle-wise visibility of this strongly coupled system. The studies discussed here were conducted in Plasmakristall-4 (PK-4), which is an experimental setup that provides a low-temperature plasma in an elongated glass tube with electrodes on each side [6], which allows the microparticles to be moved through the field of view of three cameras at different speeds and under various plasma conditions. The recorded particle image data are used for analysis. Current studies make use of microparticles at up to 30 pixels per image at densities of up to 0.002 particles per pixel (N_{ppp}), resulting in an average minimum particle spacing of 22.36 px, which translates into a ξ of roughly 1.34. At this point, the commonly used PTV framework Trackpy [7] reaches its limits and is not able to deliver sufficiently low error rates, E_{track}, or even find correct particle matches.

The use of a Self-Organizing Map (SOM) has already proven to be a great alternative for PTV in complex-plasma applications [8], while its matching performance relies on correctly set hyperparameters. The SOM was, therefore, optimized to work in scenarios with such fast-moving particles. This was achieved by optimizing the SOM hyperparameters to different ξ regions and additionally enlarging the possible application range by using artificial data. Also, the superiority of the SOM compared with Trackpy was quantified by using various artificially created flow fields. Lastly, a new approach to identifying outliers in particle traces is presented by using a Long Short-Term Memory (LSTM) neural network that reached 93.5% accuracy on artificial data.

Particle Tracking Velocimetry

To provide context, an initial comparison and overview of PTV methods is presented, offering insights into their capabilities and limitations. One very common technique is the multi-frame approach, where between two and four successive images are taken into account for analysis. The particles are matched by minimizing values such as the variance of the particle's trajectory and velocity [9], change in acceleration [10] or the distance to a projected particle position [11] by using the information from preceding images. Therefore, this approach needs some degree of initialization to be used and is also prone to errors if the ratio of the mean particle displacement to the mean particle spacing (ξ) is larger than 0.2, since then the ratio of incorrectly matched particles to the total number of matchable particles (E_{track}) grows larger than 0.01 and rises quickly [11].

Another common PTV method that requires only two successive images is the use of cross-correlation. This approach is based on the nearly rigid movement of a particle and its nearest neighbors in a fluid. By using the positions and movement of the nearest neighbors, a sub-region is defined around the particle in which a matching particle is searched for. This is either performed by using binarized images [12,13] or directly using grayscale images [14]. This method works best for irrotational flows. However, if the velocity gradients become increasingly higher and the flow is rotational, the interrogation

windows of the cross-correlation process need to be deformed according to the underlying flow [15].

The relaxation method, in contrast [16–18], is based on a match probability matrix that is initially populated by using particle displacement and subsequently updated iteratively by considering the motion and motion consistency of neighboring particles. This iterative process increases computational time, which can be reduced by incorporating cross-correlation techniques during initialization [19]. However, if the match probability updates fail to converge, tracking errors occur.

Trackpy, a widely used tool in complex-plasma research, combines particle displacement and trajectory consistency with predictive linking based on velocity [7]. The algorithm searches for the best particle match in the subsequent image within a user-defined search radius. If an estimated underlying velocity is available, it is also incorporated. As particle displacements increase, larger search radii are required, which increases the number of potential matches—a problem that is further exacerbated by high particle densities. Consequently, computational cost increases significantly, and matching accuracy deteriorates once the number of possible matches becomes too large. To mitigate this, Trackpy employs an adaptive linking feature that dynamically reduces the search radius when the candidate match set becomes excessively large. However, this also risks excluding valid matches in cases of high displacement or density. As a result, in the absence of known flow velocities, particle flows with large ξ values cannot be reliably matched.

Other approaches using neural networks have been explored, starting with feedforward networks [20], which require pre-training before application, and later with the SOM, which does not require training and demonstrated improved matching performance [21,22]. The SOM approach also proved beneficial in the analysis of wind-blown sand flows [23] and was extended to include orientation characteristics for non-spherical particles [24]. Due to its promising matching performance, minimal initialization requirements, and customizability, the SOM has been successfully applied to complex-plasma data [8].

However, when applied to current high-ξ plasma particle experiments, a decline in matching performance is observed due to uncalibrated hyperparameters. Earlier SOM implementations relied on manually selected hyperparameters, limiting their adaptability to varying flow conditions. In contrast, our work introduces an automated hyperparameter optimization framework that systematically adjusts key parameters based on flow characteristics by using synthetic, data-driven evaluation to maximize matching accuracy. By reducing the need for manual tuning and dynamically adapting to different ξ values, our approach significantly improves the robustness and versatility of SOM-based tracking across diverse experimental settings. A detailed calibration of these hyperparameters is presented in this study. Furthermore, we introduce an LSTM-based outlier detection system—an innovation not previously applied in SOM-based PTV—which substantially enhances tracking accuracy in complex-plasma environments.

2. Methodology

2.1. Network Design

The SOM network architecture is based on the design proposed by Labonté [21]. It consists of two separate sub-networks, each comprising a single layer of neurons. The first network corresponds to the first of the two consecutive images, where each neuron represents a particle in that image. Similarly, the second network corresponds to the second image, with each neuron representing a particle therein.

Accordingly, the number of neurons in each network is determined by the number of particles in the respective images: the first network contains N neurons, and the second network contains M neurons, where N and M denote the number of particles in the first

and second images, respectively. Each neuron is defined by a two-dimensional weight vector. Initially, the weight vector of each neuron is set to the x and y coordinates of the particle it represents. Therefore, the weight vectors of the first network a_i and the second network b_j are defined as

$$\begin{aligned} a_i &= (x_i, y_i), \text{ with } i = 1, 2, \dots N \\ b_j &= (x_j, y_j), \text{ with } j = 1, 2, \dots M. \end{aligned} \tag{1}$$

After initializing the two neural networks, particle matching is performed iteratively. The neurons' weight vectors are repeatedly updated based on the corresponding neurons in the other network and their surrounding neighborhood. After a predefined number of iterations, the neurons self-organize such that corresponding neurons from the two sub-networks move into close proximity.

The process begins by identifying the best-matching unit (BMU) for each neuron in network 1 within network 2. The BMU is defined as the nearest neighbor in the other network, determined by the Euclidean distance:

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}. \tag{2}$$

This BMU neuron best represents the neuron from the first network within the second network and is therefore activated. To ensure that only neurons with sufficient similarity are activated, activation occurs only if the distance between the input neuron and the BMU is less than a predefined threshold distance d_{th}. If this condition is not met, no activation is performed for the corresponding input neuron.

The activation of the BMU also affects its neighboring neurons in network 2. Activation involves adjusting the weight vectors of the affected neurons. The amount of adjustment Δb_j is defined by the value of α_j and the displacement between the best-matching weight vector b_w and the input weight vector a_i as follows:

$$\Delta b_j = \alpha_j \cdot (a_i - b_w), \text{ with } j = 1, 2, \dots, M. \tag{3}$$

To preserve the topological relationships within the data, activation should only influence neurons near the BMU. Therefore, α_j depends on the distance between neuron b_j and the BMU b_w. It remains constant within a radius r and decays with the increase in distance beyond that, following a Gaussian function as proposed by [22]:

$$\alpha_j = \begin{cases} \alpha & \text{for } |b_w - b_j| \leq r \\ \alpha \cdot \exp\left(-\dfrac{(|b_w - b_j| - r)^2}{2r^2} \right) & \text{for } |b_w - b_j| > r. \end{cases} \tag{4}$$

The activation process is repeated until every neuron in network 1 has been used as input for network 2. An illustration of the SOM activation process is shown in Figure 1. During this step, all weight vector adjustments are accumulated and applied only after all neurons have been processed. In the subsequent step, the process is reversed: all neurons in network 2 are used as input for network 1 in a similar manner as described above.

This entire procedure—alternating input from both networks—is repeated for a predefined number of iterations, after which the matching process is considered complete. With each iteration, the initially large radius r decreases linearly, until in the final iterations, only the weights of the BMU are adjusted, while neighboring neurons remain unaffected. This approach enables large-scale structural adaptations during early iterations and the

fine-tuning of local matching accuracy in the final stages. Conversely, the adjustment factor increases over iterations by a factor β, as described by

$$\alpha_{iter+1} = \frac{\alpha_{iter}}{\beta} \tag{5}$$

with $0 < \beta \leq 1$.

At the end of the iterative process, the neurons in both networks have self-organized such that corresponding neurons are positioned topologically close to one another. The final matching is evaluated by comparing the distances between neurons in both networks. If two neurons are separated by less than a predefined threshold ϵ—measured by their Euclidean distance as defined in Equation (2)—they are considered a match, and the corresponding particles they represent are paired. If multiple candidate matches are identified, the closest one is selected.

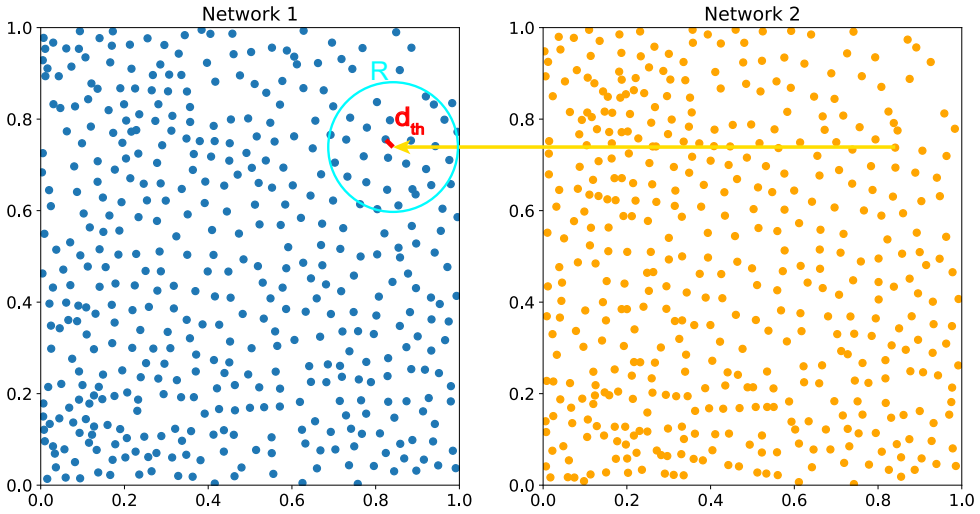


Figure 1. An illustration of the operating principle of the SOM. The two networks represent the particles of two consecutive images. Each neuron is input into the opposite network, and if a BMU is found within a threshold distance d_{th}, all neurons within a radius R are activated, and their weights are updated accordingly. The blue dots represent the particles in network 1, while the orange dots correspond to particles in network 2. The activation radius R is shown in turquoise, and the threshold distance d_{th} is visualized in red.

Computational Optimization

The SOM matches particles by arranging representative neurons based on their mutual distances. Consequently, during the matching process, the Euclidean distance between all particle pairs must be computed. Since the SOM operates iteratively, this calculation is performed multiple times. As a result, the computational complexity increases exponentially with the number of particles and linearly with the number of iterations. This can lead to substantial computation times, which may limit the applicability of the SOM for real-time analysis.

To enhance performance, particle distances are computed only once at the beginning of each iteration. Neuron weight updates are then calculated and accumulated, with the final adjustments applied at the end of the iteration.

To address the challenge of high particle counts in large images, the image is divided into smaller sub-images with overlapping edges. The overlap should significantly exceed the typical particle displacement to ensure proper matching. Particles are matched within these smaller segments with considerably reduced computational effort, and the segments

are merged upon completion. Special attention is required during merging to eliminate duplicate particles resulting from overlapping regions. If two duplicate particles are matched to different counterparts, the match with the smallest distance is retained.

With these improvements, the computation time for matching two consecutive 2048 × 2048 images containing over 17,000 particles was reduced from over four minutes to approximately five seconds by dividing the image into 16 smaller segments—without introducing matching errors at the segment boundaries.

2.2. Hyperparameter Optimization

The particle matching process described above operates without the need for prior initialization or training. The algorithm requires only the input data in the form of x and y coordinates of particles from two consecutive images, along with a set of predefined SOM hyperparameters: d_{th}, α, β, ϵ, the number of iterations, and the initial and final radii r_0 and r_f for the linear decrease. These hyperparameters are typically initialized with values that yield good overall performance. Based on previous SOM implementations [8,22–24], the typical parameter ranges are as follows: α between 0.002 and 0.011, β from 0.7 to 1, ϵ between 0.03 and 0.05, number of iterations between 5 and 100, r_0 from 60 to 40 and r_f from 0.1 to 0.0001. The threshold distance d_{th} should be larger than the average particle displacement between images. These hyperparameters directly affect the performance of the matching process. While these settings provide good results for simple and slow particle flows, careful initialization is essential to maintaining robust matching performance in high-ξ flows. The influence of each parameter on the matching process is discussed below.

The initial radius r_0 may need to be adjusted when particle density varies. However, as long as it is large enough to encompass several neighboring neurons, its impact on performance remains limited. In contrast, the final radius r_f and the minimum matching distance ϵ should remain small, regardless of particle density or displacement. This is because r_f is intended to include only the BMU neuron and ϵ, as the matching criterion, strongly influences the risk of incorrect particle matches.

At higher particle displacements, other hyperparameters exert a more significant influence. For effective self-organization, the network must accommodate the displacement range, which depends on the values of d_{th}, α and β, and the number of iterations. If any of these parameters are set too low, neurons may be unable to overcome large displacements, resulting in missed or incorrect matches. In cases where $\xi > 1$, this can lead to particle matches that falsely indicate movement in the opposite direction of the actual flow.

This phenomenon is observed when analyzing increasingly fast complex-plasma particle clouds using the SOM. As particle velocity increases while maintaining constant density, the matching performance deteriorates significantly, rendering reliable analysis infeasible. Trackpy [7], commonly used in complex-plasma experiments, also fails to accurately identify particle traces under these conditions, even when employing its adaptive linking feature. Therefore, the objective is to identify optimized hyperparameter regions that enable accurate particle tracking at high velocities using the SOM.

To achieve this, isotropic particle clouds are randomly generated at a fixed particle density of 0.002 N_{ppp}, closely matching the actual complex-plasma densities observed in current experiments. An adjustable horizontal displacement is then applied to simulate particle motion. The cloud is constructed by continuously adding particles at random positions while maintaining a minimum distance between them until the target density is reached. To further align the synthetic data with experimental conditions, random noise is added in both the x and y directions.

Particles are displayed within a 512 × 512 pixel window, and 200 time steps of movement are generated. The SOM is initialized with various combinations of α, β and d_{th}

values, along with different iteration counts, before being applied to the synthetic dataset. After each run, the matching results are evaluated, and the matching performance is calculated as the percentage of correctly matched particles out of all possible matches. This procedure enables the systematic analysis of hyperparameter effects and the identification of optimal configurations.

In our tests, the SOM consistently performed best with a β value of 0.95 across all scenarios; therefore, this value was fixed for the sake of presentation. The threshold distance d_{th} was automatically set to 1.3 times the particle displacement per time step in the artificial dataset. Consequently, the remaining parameters—α and the number of iterations—were calibrated relative to the corresponding ξ value of the synthetic data. Representative results are illustrated in Figure 2, and a comprehensive summary is provided in Table 1.

It is expected that for flows with the same ξ value but different particle densities—and thus different particle displacements over time steps—the SOM will yield similar overall matching performance. However, the optimal α value and number of iterations may vary depending on the specific displacement characteristics. This behavior is analyzed for flows with a ξ value of 1.0 at different particle densities.

Table 1. Optimal α and iteration values for flows with different ξ values that maximize particle matching performance. The corresponding best performance is reported as the percentage of correctly matched particles relative to all possible matches.

ξ	α	Iterations	Performance [%]
0.1	0.002	20	98.15
0.2	0.002	20	98.41
0.3	0.009	5	98.24
0.4	0.0055	10	97.81
0.5	0.009	10	96.64
0.6	0.0125	5	95.46
0.7	0.009	10	93.82
0.8	0.002	75	92.69
0.9	0.009	45	90.2
1.0	0.009	50	85.98
1.1	0.0055	65	75.06
1.2	0.009	55	71.81
1.3	0.009	55	59.29
1.4	0.0125	50	49.35
1.0	0.009	55	84.96 @ 0.001 N_{ppp}
1.0	0.002	75	85.88 @ 0.004 N_{ppp}

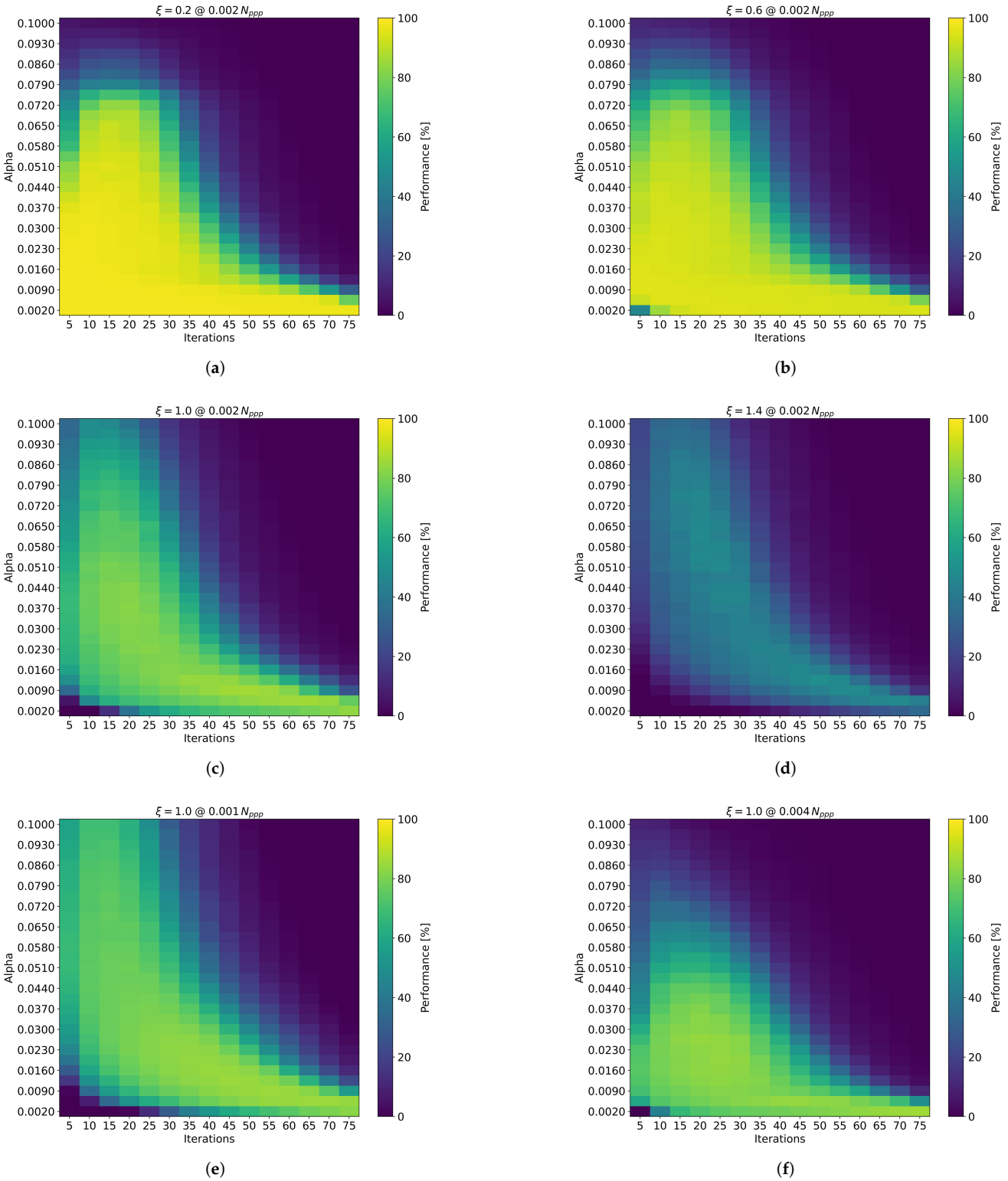


Figure 2. Exemplary displays of particle matching results for varying α and iteration settings across different ξ values. (**a**) Results for $\xi = 0.2$, (**b**) $\xi = 0.6$, (**c**) $\xi = 1.0$ and (**d**) $\xi = 1.4$. Subfigures (**e**) and (**f**) both depict $\xi = 1.0$ at particle densities of 0.001 N_{ppp} and 0.004 N_{ppp}, respectively. The color scale represents the percentage of correctly matched particles relative to all possible matches.

3. Results

The results clearly demonstrate that the α and iteration settings play a critical role in particle matching performance. As expected, higher ξ values require a combination of increased α and/or a greater number of iterations to achieve accurate particle matching. It is evident that for higher ξ values, increasing the number of iterations is generally more effective than increasing α. Figure 2 further illustrates that the range of hyperparameter values yielding good matching performance narrows as ξ increases. Therefore, for high flow speeds, the careful tuning of SOM hyperparameters is essential. The optimal α and iteration values are summarized in Table 1.

Notably, the matching performance drops significantly at ξ values of 1.0 and above, and even more so beyond $\xi = 1.3$, where the SOM identifies only about half of the possible particle matches. Remarkably, these results are achieved solely by adjusting the hyperparameters to the flow conditions, without requiring prior knowledge of the flow direction. Nevertheless, to ensure optimal accuracy for subsequent data analysis, robust outlier detection and error assessment remain crucial—particularly in this challenging regime.

Direct comparisons with other SOM implementations are limited due to differences in particle flow characteristics and the scarcity of detailed documentation in published studies. However, our results qualitatively align with the hyperparameter configurations reported by Ji et al. [23], who used 100 iterations to achieve convergence in wind-blown sand flow analysis, emphasizing the necessity of extended training cycles in highly dynamic environments. In contrast, Hoseini et al. [24] observed convergence in just 12 iterations when tracking rod-like particles in dispersed two-phase flows, even under high-ξ conditions ($\xi \geq 1$). Our empirical observations suggest that increasing the iteration count while lowering α can yield measurable improvements in performance, supporting the idea of extended optimization even after apparent convergence. These findings underscore the importance of context-specific hyperparameter tuning in SOM-based PTV, especially when balancing computational efficiency and matching accuracy in complex flow regimes.

The hyperparameter optimization was conducted for a fixed particle density of 0.002 N_{ppp}. Particle flows with the same ξ value but varying particle densities and corresponding displacements are illustrated in Figure 2c,e,f. As a result, the SOM exhibits comparable performance for flows with higher densities and lower displacements, as well as for those with lower densities and higher displacements, provided that the ξ value remains constant. Consequently, SOM matching performance primarily depends on the flow's ξ value. However, variations in particle density influence the optimal hyperparameter settings—particularly the α value. In this study, the hyperparameters were fully calibrated for the specific case of 0.002 N_{ppp}. For applications involving different particle densities, either the α value must be scaled accordingly, or the particle coordinates should be rescaled to maintain consistent matching behavior.

To support broader applicability, an automated hyperparameter calibration feature has been integrated into the SOM implementation. This functionality streamlines the tuning process for key parameters, particularly α and the number of iterations, by adapting them to the characteristics of the particle flow. The calibration involves matching particles between the first two images in the dataset while systematically varying the hyperparameters within predefined ranges. For each configuration, the algorithm evaluates performance by calculating the percentage of correctly matched particles relative to the total number of matchable particles. This quantitative assessment enables the identification of the hyperparameter combination that yields maximum matching accuracy. By automating this calibration step, the SOM becomes more adaptable to diverse flow conditions and significantly reduces the need for manual fine-tuning, thereby enhancing its robustness and practical utility across a wide range of applications.

In conclusion, the particle matching performance of the SOM in low-ξ flows is very good, requiring only minor adjustments to the hyperparameters. In high-ξ flows, characterized by elevated particle densities and velocities, careful hyperparameter tuning becomes essential to achieving accurate matching and reliable results. However, due to the increased likelihood of errors under such conditions, robust performance additionally depends on the availability of sufficient data and the implementation of effective outlier filtering techniques.

3.1. Validation on Experimental Complex-Plasma Data

To further validate the effectiveness of our hyperparameter-optimized SOM, we applied it to experimental complex-plasma data obtained from recent PK-4 experiments conducted on the International Space Station and during parabolic flights. The experimental data featured fast-moving plasma particle clouds with ξ values reaching up to 1.34. The optimized SOM successfully tracked particles in these flows, demonstrating both robustness and accuracy under real-world conditions.

Due to the fact that the experimental data are unlabeled, it is not possible to provide an exact quantitative assessment of accuracy or matching performance. However, detailed results and a comprehensive analysis of this application have been published by Wimmer et al. [2]. These findings confirm that the proposed hyperparameter tuning framework enables reliable PTV even in highly challenging experimental scenarios.

3.2. Comparison Overview

With optimal hyperparameter settings for the SOM, a direct comparison to the conventional method Trackpy becomes feasible. This comparison is carried out using artificially generated complex particle flow data under various conditions to evaluate and verify matching performance. Artificial data are used to enable a controlled, qualitative comparison with definitive ground truth while maintaining similarity to real complex-plasma datasets.

In addition, a widely accepted method for validating particle tracking performance involves the use of VSJ PIV standard images [25], which is also employed here to provide a comprehensive assessment of particle matching performance.

3.2.1. Evaluation on Synthetic Data

Similar to Section 2.2, isotropic particle distributions are generated and various flow fields are applied. These flow fields range from simple laminar flows to more complex scenarios, including multiple vortex flows, radial distortion, shear flows, divergent and convergent flows and random turbulent motion—designed to represent a wide range of possible flow situations. The schematics of the synthetic flow types are presented in Figure 3.

The laminar flow field is used to evaluate the impact of increasing particle displacement over time steps. The two vortex flow fields introduce the challenge of directional changes, one in a uniform and the other in a more irregular manner. The radial distortion flow tests the ability to detect non-coherent, omnidirectional motion. Shear, divergent and convergent flows introduce particle motion with strongly varying speeds and local densities. Finally, the random flow field simulates fully inconsistent and chaotic motion.

These synthetic flows are applied at varying intensities, and the matching performance of both algorithms is assessed. The corresponding ξ value is calculated based on the mean particle displacement per time step and the mean particle density, which is fixed at $0.002\ N_{ppp}$ to closely replicate real complex-plasma data. For particle matching, the Trackpy algorithm is used with adaptive linking enabled. A comprehensive comparison of results is provided in Table 2, where matching performance is reported as the ratio of correctly matched particle pairs to all possible pairs.

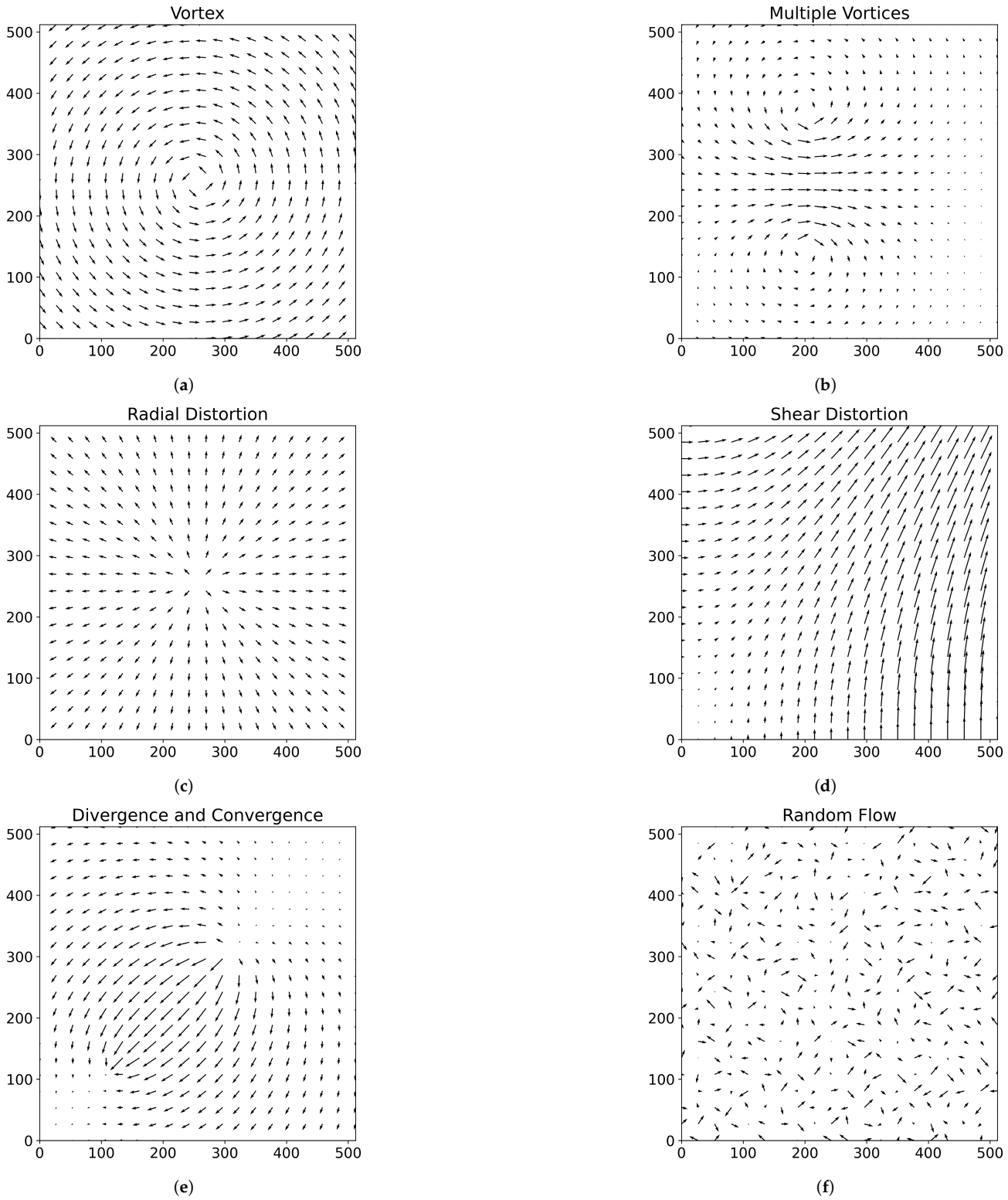


Figure 3. Vector flow fields used to generate synthetic particle flow data for comparing the matching performance of the SOM and Trackpy algorithms. The length of each arrow indicates the flow speed of the corresponding particles. Subfigures depict the following flow types: (**a**) single vortex flow, (**b**) multiple vortex flow, (**c**) radial distortion flow, (**d**) shear distortion flow, (**e**) divergent and convergent flow and (**f**) random flow.

Table 2. Matching performance of SOM and Trackpy algorithms for different flow types and ξ values. Performance is defined as the percentage of correctly matched particles relative to all possible particle matches.

Flow Type	**Method**	**Performance [%]**				
		ξ				
		0.2	**0.4**	**0.6**	**0.8**	**1.0**
Laminar						
	SOM	99.90	99.65	98.34	96.13	87.79
	Trackpy	99.90	86.36	12.1	0.02	0
Vortex						
	SOM	100	99.7	99.33	98.43	96.46
	Trackpy	100	87.84	18.74	1.24	0.04
Multiple vortices						
	SOM	99.41	98.17	88.28	84.48	61.67
	Trackpy	84.06	46.81	9.14	7.01	3.03
Radial distortion						
	SOM	100	99.47	96.92	85.06	73.28
	Trackpy	100	94.92	31.74	9.16	4.37
Shear distortion						
	SOM	100	99.84	99.38	98.03	97.14
	Trackpy	100	88.62	60.70	42.92	30.33
Divergent and convergent						
	SOM	96.81	90.7	80.51	58.11	37.68
	Trackpy	61.49	19.45	10.2	4.17	2.18
Random						
	SOM	81.68	35.28	17.07	10.26	6.1
	Trackpy	96.10	63.63	24.44	14.06	8.71

It is clearly evident that the SOM consistently outperforms Trackpy under high-ξ flow conditions. Beginning with the laminar flow case, the SOM maintains strong performance even at high particle displacements over successive time steps, whereas Trackpy's performance rapidly declines below 50% for ξ values exceeding 0.6. Similar trends are observed in the vortex flow scenarios: while Trackpy delivers acceptable results only at very low displacements, the SOM maintains high accuracy. In the single vortex case, the SOM's performance remains nearly constant up to $\xi = 1$, with only a slight drop being observed for multiple vortices. In contrast, Trackpy becomes ineffective as particle velocities increase.

Comparable results are found for the radial distortion and divergent/convergent flows, where the SOM significantly outperforms Trackpy across the tested ξ values. An exception is observed in the shear flow, where Trackpy shows marginally better performance relative to other cases. This can be attributed to the presence of very slow-moving particles in the lower region of the image, which remain easily matchable. Since the ξ value is calculated based on the mean particle speed, the impact of these nearly stationary particles skews the apparent difficulty of the scenario.

The only case where Trackpy consistently outperforms the SOM is the random flow field. In this highly chaotic regime, both algorithms experience a marked decline in performance, but Trackpy maintains a slight advantage. This is likely due to the rapidly and unpredictably changing flow directions, which present a particular challenge for the SOM. While both methods struggle under these conditions, the SOM exhibits a more pronounced drop in accuracy.

3.2.2. Evaluation on VSJ PIV Images

To further quantify the particle matching performance of the SOM and Trackpy algorithms, both methods were applied to VSJ PIV standard image #301 [25]. A similar evaluation has previously been conducted for a SOM implementation [22], although in that case, particle positions were identified by using dynamic binary thresholding prior to matching. This approach detected only a subset of the particles, thereby significantly reducing the effective particle density in the image. In contrast, our analysis considers all known particle positions.

Among the 4042 matchable particles in the #301 image, the SOM successfully identified 3959 matches, of which 3829 were correct. This corresponds to a matching yield of 97.95% and a reliability of 96.72%. The test was performed by using an α value of 0.002 and 15 iterations, which yielded the best results for this dataset. The hyperparameters were optimized by using the same method described in Section 2.2, though lower α values were optimal in this case due to the higher particle density of the image, measured at 0.061 N_{ppp}. The matching performance across different hyperparameter settings is visualized in the heatmap in Figure 4, and the matched particle pairs are illustrated in Figure 5.

In comparison, Trackpy produced only 3337 matches, of which just 1160 were correct. This corresponds to a matching yield of 28.7% and a match reliability of 34.76%.

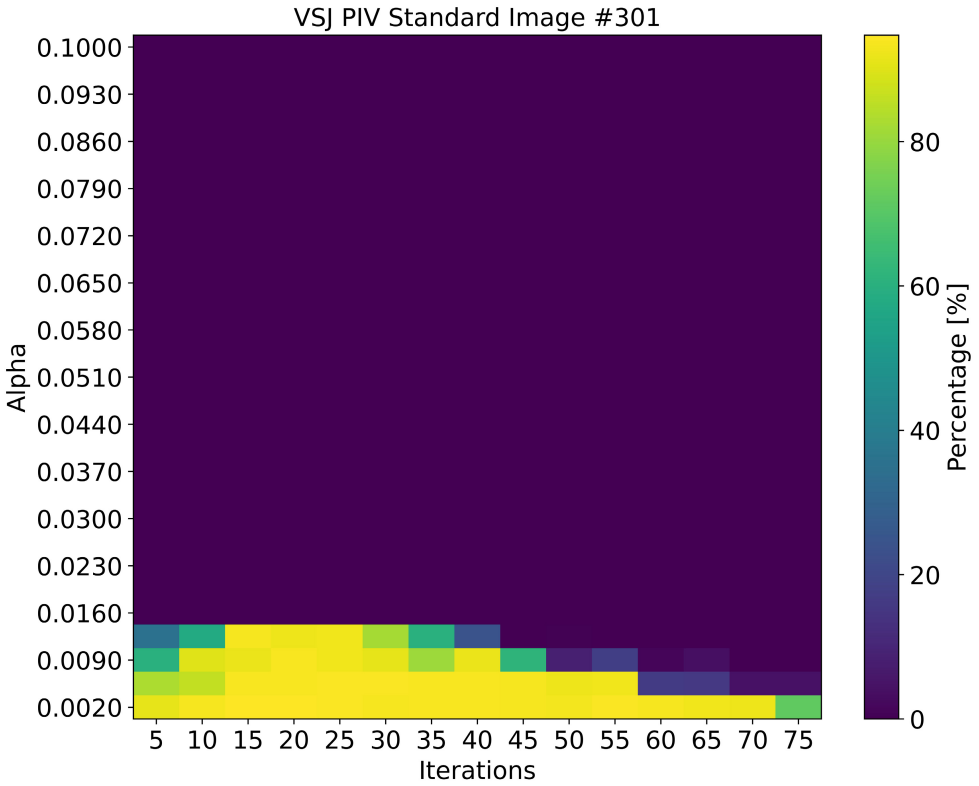


Figure 4. Particle matching results for varying α and iteration settings on the VSJ PIV standard image #301 [25]. The color scale represents the percentage of correctly matched particles relative to all possible particle matches.

The SOM demonstrates impressive performance across a wide range of flow speeds, particle densities and ξ values, underscoring its versatility in particle matching tasks. These results highlight the SOM's potential not only for complex-plasma experiments but also for a broad range of other applications. Furthermore, the SOM significantly outperforms

Trackpy, as evidenced by its substantially higher matching yield and reliability in the evaluation using the VSJ PIV standard image.

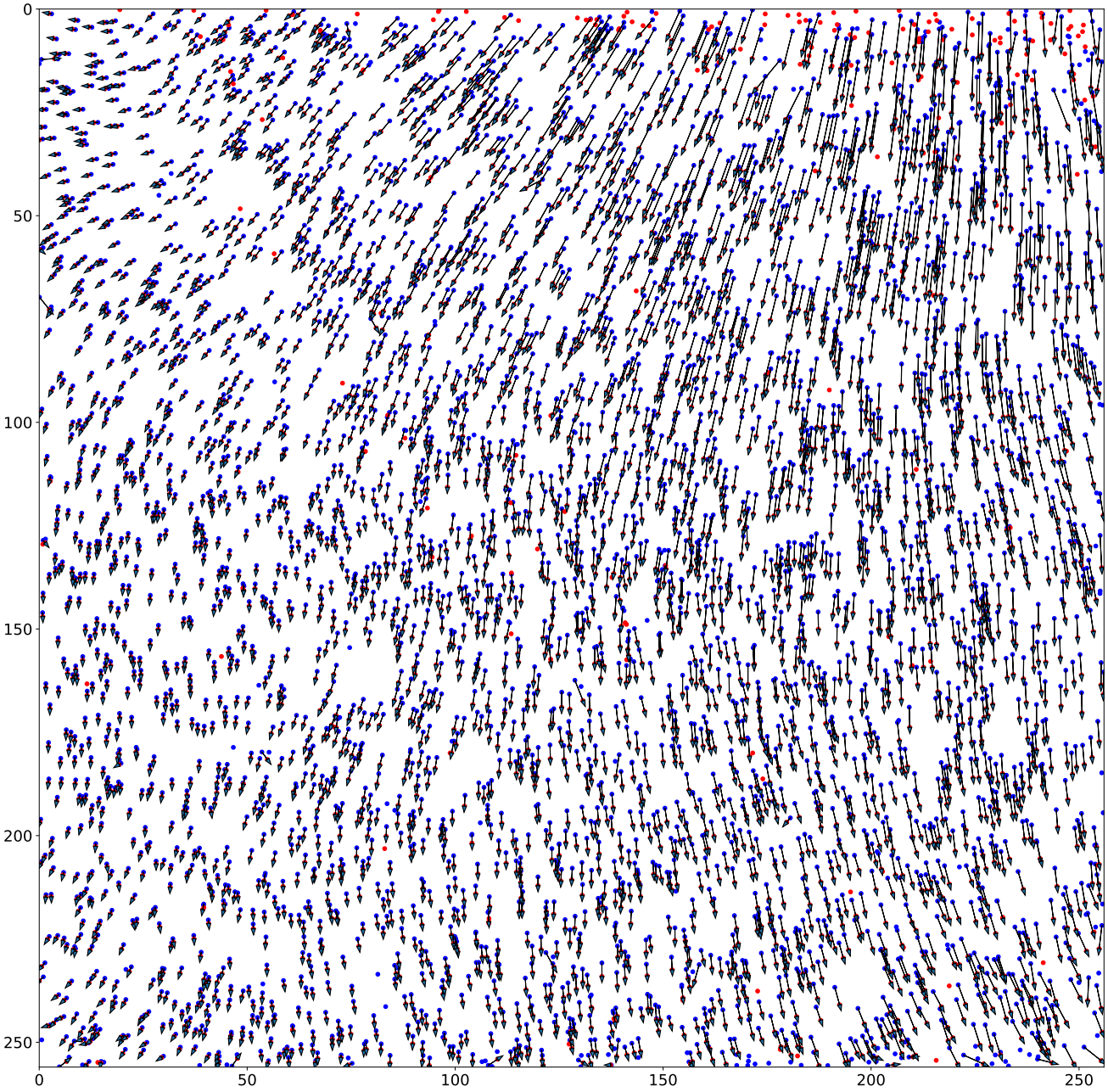


Figure 5. Visualization of all particles in the first two VSJ PIV standard images, #301 [25]. Particles from the first image are shown in blue, and particles from the second image are shown in red. Arrows indicate the particle matches identified by the SOM.

3.3. Outlier Detection Using LSTM Network

When it comes to detecting outliers, multiple approaches are available, ranging from simple velocity and/or directional filters [26] to continuity-based evaluations [27] and more advanced methods such as fuzzy logic, which enables more nuanced decision making for individual particles [28]. While these algorithms can yield excellent results, they generally require calibration to a specific dataset. When filter ranges and fuzzy logic rules are

appropriately configured, outlier filtering is straightforward. However, in chaotic or poorly characterized flows, these approaches may fall short.

To address this limitation, we introduce a novel outlier detection method based on a Long Short-Term Memory (LSTM) neural network, which analyzes particle trajectories to identify and remove defective traces.

The LSTM network is a type of recurrent neural network (RNN) characterized by a memory cell that retains past information, making them particularly effective for sequential data analysis. In contrast to traditional neural networks, where the output depends solely on the current input and fixed weights, LSTM maintains both a cell state C_t and a hidden state h_t, which—together with the current input X_t—are used to compute the next states and output.

An LSTM cell comprises three gates: the forget gate f_t, the input gate i_t and the output gate o_t. The forget gate determines which information from the previous cell state should be discarded; the input gate controls how much new information is stored; and the output gate governs the contribution of the cell state to the final output. These operations utilize nonlinear activation functions, primarily the sigmoid function σ, which produces gating values between 0 and 1, and the *tanh* function, which maps values to the range $[-1, 1]$ for smooth, bounded transformations.

A schematic of the LSTM cell is presented in Figure 6. The ability of LSTMs to model temporal dependencies is particularly advantageous for analyzing sequential datasets such as particle trajectories. Their memory structure allows them to extract meaningful patterns across sequences of varying length and complexity, making them well suited for robust outlier detection in particle tracking applications.

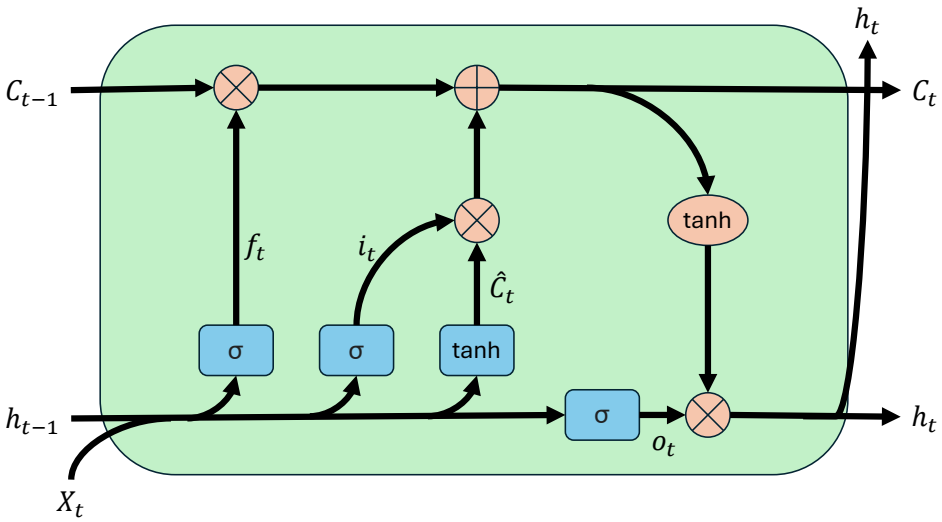


Figure 6. Schematic of LSTM cell. The previous cell state C_{t-1}, hidden state h_{t-1} and current input X_t are processed through the forget gate f_t, input gate i_t and output gate o_t, as indicated by the arrows. The blue boxes represent the sigmoid (σ) and *tanh* activation functions, while the red circles denote the corresponding arithmetic operations. The updated cell state and the output of the LSTM cell are shown on the right.

LSTM networks are particularly well suited for outlier detection in sequential data due to their ability to model long-range temporal dependencies. Previous studies have demonstrated their effectiveness in identifying anomalies across various domains. For example, Chen et al. [29] employed LSTM autoencoders to detect anomalies in particle accelerator systems, achieving high accuracy in identifying orbit lock anomalies. Similarly, Ergen and Kozat [30] proposed an LSTM-based framework that integrates One-Class Support Vector

Machines (OC-SVMs) and Support Vector Data Description (SVDD) for unsupervised anomaly detection in variable-length sequences. Saha et al. [31] introduced a Quantile LSTM for detecting anomalies in industrial time-series data, while Park et al. [32] utilized an ensemble of LSTM autoencoders to identify outliers in indoor air quality monitoring.

Beyond anomaly detection, LSTMs have also been applied to predict and filter trajectory-based motion patterns. Hug et al. [33] developed a Long Short-Term Memory with Minimum Description Length (LSTM-MDL) model for pedestrian path prediction, demonstrating the LSTM's capability to filter anomalous movement patterns and optimize trajectory representations in sequential data.

Inspired by these developments, we propose an LSTM-based method to detect incorrect particle traces in PTV. Our approach analyzes sequential particle displacements and directional changes to identify and filter trajectories that significantly deviate from expected motion patterns. Unlike prior applications focused on accelerator monitoring, industrial fault detection or pedestrian tracking, our method is specifically designed for tracking particle motion in complex-plasma flows, representing a novel adaptation of LSTM networks within this context.

An LSTM neural network is constructed, beginning with a ragged input layer that accepts sequences of variable lengths, each containing two features. This is followed by an LSTM layer with 64 units and L2 regularization (0.01) to reduce the risk of overfitting. A batch normalization layer is applied after the LSTM layer to stabilize and accelerate training. The output layer consists of a dense layer with a single neuron, also employing L2 regularization (0.001), followed by another batch normalization layer and a sigmoid activation function to produce the final output. A schematic of the network architecture is shown in Figure 7. The sigmoid output determines whether the input particle trace is classified as correct (output = 1) or faulty (output = 0).

Synthetic particle trace data are used for training. Following the approach described in Section 2.2, isotropic particle clouds are randomly generated at various densities, and flow fields are applied to produce labeled particle positions over successive time steps. A combination of laminar and vortex flow fields with varying directions and intensities is used to simulate diverse motion patterns. These flows are also superimposed with random fluctuations to enhance realism. Correct particle traces are extracted from the labeled dataset. To simulate faulty traces, half of the correct trajectories are modified by replacing one or two matched particles with incorrect neighbors at random positions, thereby imitating common tracking errors. The resulting traces are labeled accordingly, with correct traces assigned a label of 1 and incorrect traces assigned a label of 0.

Prior to training, each particle trace—comprising a sequence of particle coordinates—is transformed into a sequence of features representing the distance and angle between successive matched particles. These features are then converted into relative values by expressing each distance and angle in relation to those of the preceding step. This results in a two-dimensional vector sequence representing relative distances and angles.

A total of 170,000 traces are generated for training, equally divided between correct and incorrect cases. The dataset is split, with 20% used for validation during training. Approximately 67% of the traces are derived from laminar flow data and 33% from vortex flow data. The model is trained by using the Adam optimizer [34] with an initial learning rate of 0.0001. The learning rate is reduced by a factor of 0.5 after three consecutive epochs without improvement, with a final minimum learning rate of 0.0000125. Binary cross-entropy (BCE) is used as the loss function to quantify the discrepancy between predicted probabilities and true labels. Early stopping is employed to prevent overfitting, terminating the training process after five consecutive epochs without improvement. Training concludes after 48 epochs, achieving a final accuracy of 93.5%.

Figure 8 shows the training history, including loss and accuracy over all epochs, to illustrate model performance. After training, the model achieves a recall of 97.6% and a precision of 90.4%, indicating that it successfully detects nearly all outliers, though it may occasionally misclassify correct traces as outliers.

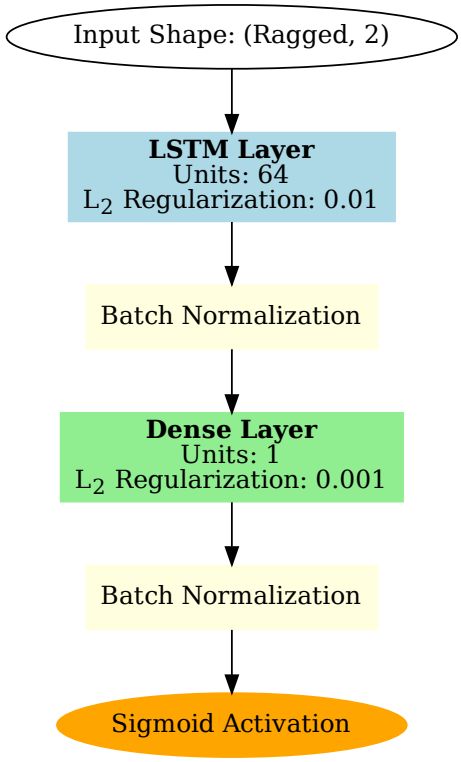


Figure 7. A schematic of the LSTM network used for filtering defective particle traces. The input layer accepts sequences of variable length, enabling the analysis of particle traces with differing durations. The output is passed through a sigmoid activation function to classify each trace as valid or faulty.

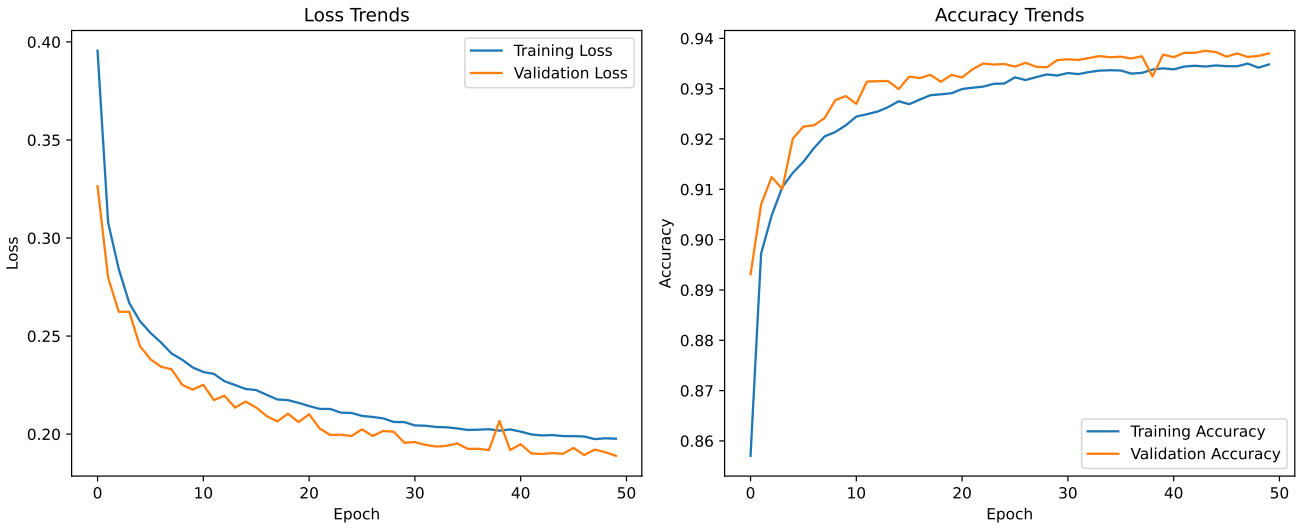


Figure 8. Training and validation loss and accuracy trends during the training of the LSTM-based outlier detection model.

To validate the performance of the LSTM-based outlier detection method, we conducted a comparative evaluation against three additional outlier detection algorithms

applied to various particle flow scenarios. The first algorithm was a simple threshold-based approach, which filters particles exhibiting abrupt changes in velocity magnitude or angular displacement. The second method employed a continuity filter that evaluates the temporal consistency of individual particle trajectories. The third was a fuzzy logic-based algorithm that assigns an outlier probability to each particle match based on predefined rules and parameter ranges; this probability determines whether a match is classified as correct or incorrect. All four algorithms, including the LSTM method, were tested on three distinct particle flow datasets to assess their performance under different conditions.

The first two datasets consisted of synthetic particle flows designed to simulate laminar and vortex flow patterns. In both cases, particles were randomly distributed across the domain with a minimum inter-particle distance enforced to ensure isotropic spacing. A laminar flow field was applied to the first dataset, while the second incorporated a vortex velocity field. To introduce additional complexity and realism, random noise was added to the particle motion in both datasets, simulating deviations from idealized flow and increasing the difficulty of outlier detection.

The third dataset consisted of VSJ PIV standard image set #301 [25], a widely used benchmark for validating particle image velocimetry (PIV) methods. To evaluate algorithm robustness, artificial outliers were introduced into all three datasets at contamination levels of 5%, 10% and 15%. These outliers were created by randomly selecting particle trajectories and replacing segments with sections from neighboring trajectories—effectively mimicking realistic tracking errors without relying on overly simplified anomalies such as large, easily detectable jumps.

Prior to outlier detection, all datasets were preprocessed by using the SOM for particle trajectory matching. To simulate challenging conditions, SOM parameters were left unoptimized, resulting in an initial outlier density of approximately 50%. This setup provides a rigorous testing environment for the algorithms.

In the evaluation, a particle trace is classified as incorrect if it contains at least one outlier, regardless of whether the algorithm analyzes complete trajectories or individual particles. For algorithms that flag individual particle-level outliers, the corresponding full trace is marked as incorrect. This standardized evaluation protocol ensures a consistent and fair comparison across all algorithmic approaches.

The performance of each outlier detection algorithm depends heavily on both the characteristics of the data and the specific parameter settings of the algorithms. In this study, algorithm parameters were coarsely tuned to each particle flow scenario but were not subjected to detailed optimization for maximum performance. This approach was chosen to ensure a fair comparison with the LSTM-based method, which benefits from end-to-end learning and does not require explicit parameter tuning. While this lack of optimization leaves room for potential improvements in the results of the traditional algorithms, the chosen configurations still provide meaningful benchmarks for evaluating the LSTM's performance.

The datasets used in this study pose substantial challenges for outlier detection, given their high noise levels, realistic particle densities and carefully constructed outliers that are not trivially identifiable. Despite these complexities, the LSTM-based algorithm demonstrated strong and consistent performance without the need for manual parameter adjustment. In contrast, traditional methods relied on heuristic thresholds and rules that required manual tuning for each dataset.

The full results of the comparative analysis are presented in Table 3. Two key performance metrics are reported: the total number of detected outliers and the percentage of correctly identified outliers among those detected. These metrics offer insights into the sensitivity of each method and the reliability (precision) of outlier detection.

Table 3. Detection rates and reliability of outlier detection algorithms in laminar, vortex and VSJ PIV flows across varying outlier contamination levels.

Flow	Outlier [%]	Detected Outliers [%]/Reliability [%]			
		LSTM	Threshold	Continuity	Fuzzy
Laminar	5	3.10/82.46	35.76/4.13	3.07/1.85	3.03/82.15
	10	9.39/74.07	20.29/38.22	7.99/71.53	6.53/83.93
	15	12.14/77.39	25.78/52.48	12.07/80.3	11.4/88.24
	48	0.58/42.86	7.53/47.54	0.58/42.86	9.54/45.84
Vortex	5	2.24/80	2.24/80	4.04/33.33	38.57/8.14
	10	4.95/100	5.86/100	5.86/61.54	38.74/12.79
	15	7.94/100	9.35/95	9.81/71.43	39.25/20.24
	47	31.74/91.91	22.20/99.17	5.50/66.67	53.76/67.24
VSJ PIV [25]	5	11.06/15.03	19.85/22.79	5.91/6.51	12.01/6.8
	10	12.13/26.57	23.74/38.75	5.84/10.86	12.18/13.83
	15	13.32/36.87	27.12/49.51	5.94/17.49	12.68/19.96
	50	9.88/80.02	46.58/87.73	8.84/59.83	11.7/51.76

The performance comparison of the four outlier detection methods across different flow regimes and outlier density levels reveals several key insights. The LSTM-based approach consistently demonstrated high reliability—exceeding 74%—in detecting outliers without requiring parameter tuning, particularly excelling in complex scenarios such as vortex flows. In contrast, the threshold-based method exhibited variable performance, with a high rate of false positives at low contamination levels but improved reliability under extreme outlier densities. Notably, the LSTM network's ability to recognize temporal patterns enabled it to outperform other methods in vortex flows, achieving perfect reliability at a 15% outlier contamination level. However, at very high outlier levels (around 50%), the LSTM network's detection rate declined significantly, indicating limitations in handling extreme contamination. This performance crossover suggests that LSTM is particularly well suited for moderate contamination levels in complex flows, while traditional methods may be more effective in severely corrupted datasets.

Although traditional outlier detection algorithms may achieve improved results when carefully calibrated to specific particle flow conditions, the LSTM network's performance is remarkable—especially considering its applicability to previously unseen flow fields. Furthermore, when the underlying flow characteristics are known, the training data can be augmented with synthetic traces tailored to that flow type, enhancing model performance in targeted applications. In this study, the first 30 frames of the VSJ PIV standard image sequence were isolated and used as training data, while the remaining images were reserved for evaluation. After 20 additional epochs of retraining on these data, the LSTM detected 16.34% of outliers with a reliability of 83.57%, demonstrating substantial performance gains through targeted fine-tuning.

It is important to note that due to the sequential nature of the LSTM model, outlier detection is not feasible when only two consecutive images are available. A minimum of three or more time steps is required to construct analyzable particle traces. Beyond outlier detection, this type of neural network holds potential for broader applications. For instance, it could be trained to classify particle trace characteristics, such as distinguishing between turbulent and laminar flows, provided that appropriate labeled training data are available. In this way, particle traces could be analyzed for dynamic properties previously inaccessible through traditional methods.

In combination with the SOM, the LSTM outlier detection model expands the particle tracking framework, offering robust and adaptive analysis capabilities for complex and unknown flow conditions.

4. Discussion

In conclusion, this study presents the comprehensive hyperparameter optimization of the SOM, enabling its effective application to high-ξ flow scenarios—particularly relevant for fast complex-plasma experiments. While traditional methods such as Trackpy perform well under lower-density and simpler-flow conditions, they encounter substantial limitations in high-density, high-velocity and rotational flow environments typical of complex plasmas. This work demonstrates that the SOM algorithm, with its adaptability and minimal initialization requirements, provides a more reliable and robust approach to accurately tracking particles across a wide range of flow conditions.

A central factor contributing to the SOM's success is the fine-tuning of hyperparameters to optimize particle matching performance across various ξ values, representing different flow intensities and particle displacements. By evaluating SOM performance on artificially generated particle flows with controlled velocity profiles, this study identified optimal ranges for parameters such as α and the number of iterations. Such optimization significantly improved matching accuracy, including in regimes with $\xi > 1.0$. The enhanced SOM configuration facilitates its application in high-speed complex-plasma experiments, where traditional algorithms often suffer from high error rates. An integrated automated hyperparameter optimization function further supports efficient deployment in unknown flow scenarios.

Beyond SOM calibration, this study introduces a novel outlier detection method using an LSTM neural network to refine particle trajectory accuracy. The LSTM's ability to model temporal dependencies allowed it to achieve an accuracy of 93.5% in identifying and filtering incorrect particle matches. This significantly reduced false positives and improved the overall reliability of the particle tracking process. Particularly in flows with unpredictable directional changes, this data-driven approach enables a more automated and less manually intensive analysis pipeline. The combination of SOM for particle matching and LSTM for outlier detection thus provides a unified and powerful framework for PTV in complex-plasma research.

Comparative analyses against Trackpy further confirmed the superior performance of the SOM-based framework. The SOM consistently outperformed Trackpy across a diverse set of synthetic flow patterns—including vortices, radial distortions, shear flows and divergent/convergent flows—maintaining high reliability even in regimes with significant rotational, velocity and density gradients. While Trackpy remained effective for simpler laminar flows, its accuracy declined rapidly with the increase in ξ and trajectory complexity. The SOM's adaptability, combined with computational enhancements such as image segmentation and parallelized weight updates, not only preserved accuracy but also reduced processing time substantially, supporting the feasibility of real-time applications and automated workflows in plasma diagnostics.

Overall, this research study establishes the SOM, augmented by LSTM-based error detection, as a versatile and high-performance solution for PTV in dynamic and complex environments. The insights into optimal hyperparameter ranges and computational efficiency render this approach highly suitable for complex-plasma studies and transferable to other fields involving dense particle flows, such as atmospheric physics and fluid dynamics. Future work may further extend this framework by integrating additional machine learning models to capture specific flow features. Moreover, the LSTM-based network could be adapted for advanced particle trace analysis, such as turbulence classification or flow regime identification, unlocking new capabilities for understanding complex fluid behaviors.

Author Contributions: Conceptualization, M.K. and N.D.; methodology, M.K.; software, M.K.; validation, M.K.; formal analysis, M.K.; investigation, M.K. and N.D; resources, M.K.; data curation, M.K., N.D. and L.W.; writing—original draft preparation, M.K.; writing—review and editing, M.K., N.D., L.W., M.H.T. and M.S.; visualization, M.K.; supervision, M.K.; project administration, M.H.T. and M.S.; funding acquisition, M.H.T. and M.S. All authors have read and agreed to the published version of the manuscript.

Funding: This research study was funded by the German Federal Ministry of Economic Affairs and Climate Action under grant number 50WK2270B.

Institutional Review Board Statement: Not applicable.

Data Availability Statement: The original contributions presented in this study are included in the article. Further inquiries can be directed to the corresponding author.

Acknowledgments: This work was supported by the German Aerospace Agency (DLR). We thank the research group of Markus H. Thoma at Justus Liebig University Giessen for providing the data used in this study and the German Aerospace Society for providing powerful PCs in accordance with the funding decision, which facilitated the studies.

Conflicts of Interest: The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

Abbreviations

The following abbreviations are used in this manuscript:

PTV	particle tracking velocimetry
SOM	Self-Organizing Map
LSTM	Long Short-Term Memory
PK-4	Plasmakristallexperiment 4
BMU	best-matching unit
BCE	binary cross-entropy
OC-SVM	One-Class Support Vector Machine
SVDD	Support Vector Data Description
LSTM-MDL	Long Short-Term Memory with Minimum Description Length

References

1. Dabiri, D.; Pecora, C. Particle tracking techniques. In *Particle Tracking Velocimetry*; IOP Publishing: Bristol, UK, 2019; pp. 5-1–5-60. [CrossRef]
2. Wimmer, L.; Dormagen, N.; Klein, M.; Kretschmer, M.; Lipaev, A.M.; Schwarz, M.; Usachev, A.D.; Petrov, O.F.; Zobnin, A.; Thoma, M. Impact of particle charge and electrorheology-effects on dust-acoustic waves in low pressure complex plasma under microgravity. *New J. Phys.* **2025**, *27*, 033001. [CrossRef]
3. Schwabe, M.; Zhdanov, S.; Räth, C. Turbulence in an auto-oscillating complex plasma. *IEEE Trans. Plasma Sci.* **2017**, *46*, 684–687. [CrossRef]
4. Schmitz, A.S.; Schulz, I.; Kretschmer, M.; Thoma, M.H. Dust cloud convections in inhomogeneously heated plasmas in microgravity. *Microgravity Sci. Technol.* **2023**, *35*, 13. [CrossRef]
5. Thomas, H.M.; Morfill, G.E. Melting dynamics of a plasma crystal. *Nature* **1996**, *379*, 806–809. [CrossRef]
6. Pustylnik, M.; Fink, M.; Nosenko, V.; Antonova, T.; Hagl, T.; Thomas, H.; Zobnin, A.; Lipaev, A.; Usachev, A.; Molotkov, V.; et al. Plasmakristall-4: New complex (dusty) plasma laboratory on board the International Space Station. *Rev. Sci. Instrum.* **2016**, *87*, 093505. [CrossRef]
7. Allan, D.B.; Caswell, T.; Keim, N.C.; van der Wel, C.M.; Verweij, R.W. *Soft-Matter/Trackpy*, v0.6.1; Zenodo: Geneva, Switzerland, 2023. [CrossRef]
8. Klein, M.; Dormagen, N.; Schmitz, A.S.; Thoma, M.H.; Schwarz, M. Machine Learning Approach for Particle Matching, Tracing and Velocimetry with Self-Organizing Map: Application to Complex Plasmas. In Proceedings of the 2023 International Conference on Machine Learning and Applications (ICMLA), Jacksonville, FL, USA, 15–17 December 2023; pp. 839–844.
9. Hassan, Y.; Canaan, R. Full-field bubbly flow velocity measurements using a multiframe particle tracking technique. *Exp. Fluids* **1991**, *12*, 49–60. [CrossRef]

10. Malik, N.; Dracos, T.; Papantoniou, D. Particle tracking velocimetry in three-dimensional flows: Part II: Particle tracking. *Exp. Fluids* **1993**, *15*, 279–294. [CrossRef]
11. Ouellette, N.T.; Xu, H.; Bodenschatz, E. A quantitative study of three-dimensional Lagrangian particle tracking algorithms. *Exp. Fluids* **2006**, *40*, 301–313. [CrossRef]
12. Yamamoto, F.; Uemura, T.; Tian, Z.H.; Ohmi, K. Three-Dimensional PTV Based on Binary Cross-Correlation Method: Algorithm of Particle Identification. *JSME Int. J. Ser. Fluids Therm. Eng.* **1993**, *36*, 279–284. [CrossRef]
13. Hassan, Y.; Blanchat, T.; Seeley, C., Jr. PIV flow visualisation using particle tracking techniques. *Meas. Sci. Technol.* **1992**, *3*, 633. [CrossRef]
14. Saga, T.; Kobayashi, T.; Segawa, S.; Hu, H. Development and evaluation of an improved correlation based PTV method. *J. Vis.* **2001**, *4*, 29–37. [CrossRef]
15. Jambunathan, K.; Ju, X.; Dobbins, B.; Ashforth-Frost, S. An improved cross correlation technique for particle image velocimetry. *Meas. Sci. Technol.* **1995**, *6*, 507. [CrossRef]
16. Wu, Q.X.; Pairman, D. A relaxation labeling technique for computing sea surface velocities from sea surface temperature. *IEEE Trans. Geosci. Remote Sens.* **1995**, *33*, 216–220. [CrossRef]
17. Baek, S.; Lee, S. A new two-frame particle tracking algorithm using match probability. *Exp. Fluids* **1996**, *22*, 23–32. [CrossRef]
18. Ohmi, K.; Li, H.Y. Particle-tracking velocimetry with new algorithms. *Meas. Sci. Technol.* **2000**, *11*, 603. [CrossRef]
19. Brevis, W.; Niño, Y.; Jirka, G. Integrating cross-correlation and relaxation algorithms for particle tracking velocimetry. *Exp. Fluids* **2011**, *50*, 135–147. [CrossRef]
20. Grant, I.; Pan, X. An investigation of the performance of multi layer, neural networks applied to the analysis of PIV images. *Exp. Fluids* **1995**, *19*, 159–166. [CrossRef]
21. Labonté, G. A new neural network for particle-tracking velocimetry. *Exp. Fluids* **1999**, *26*, 340–346. [CrossRef]
22. Ohmi, K. SOM-based particle matching algorithm for 3D particle tracking velocimetry. *Appl. Math. Comput.* **2008**, *205*, 890–898. [CrossRef]
23. Ji, L.; Yang, F.; Guan, M. The Application of SOM Network to Particle Tracking Velocimetry in a Wind-Blown Sand Flow. In Proceedings of the 2015 2nd International Conference on Information Science and Control Engineering, Shanghai, China, 24–26 April 2015; pp. 493–496.
24. Abbasi Hoseini, A.; Zavareh, Z.; Lundell, F.; Anderson, H.I. Rod-like particles matching algorithm based on SOM neural network in dispersed two-phase flow measurements. *Exp. Fluids* **2014**, *55*, 1705. [CrossRef]
25. Okamoto, K.; Nishio, S.; Saga, T.; Kobayashi, T. Standard images for particle-image velocimetry. *Meas. Sci. Technol.* **2000**, *11*, 685. [CrossRef]
26. Sun, J.; Yates, D.; Winterbone, D. Measurement of the flow field in a diesel engine combustion chamber after combustion by cross-correlation of high-speed photographs. *Exp. Fluids* **1996**, *20*, 335–345. [CrossRef]
27. Song, X.; Yamamoto, F.; Iguchi, M.; Murai, Y. A new tracking algorithm of PIV and removal of spurious vectors using Delaunay tessellation. *Exp. Fluids* **1999**, *26*, 371–380. [CrossRef]
28. Sapkota, A.; Ohmi, K. Error detection and performance analysis scheme for particle tracking velocimetry results using fuzzy logic. *Int. J. Innov. Comput. Inf. Control* **2009**, *5*, 4927–4934.
29. Chen, Z.; Lu, W.; Bhong, R.; Hu, Y.; Freeman, B.; Carpenter, A. Anomaly Detection of Particle Orbit in Accelerator using LSTM Deep Learning Technology. *arXiv* **2024**, arXiv:2401.15543.
30. Ergen, T.; Kozat, S.S. Unsupervised anomaly detection with LSTM neural networks. *IEEE Trans. Neural Netw. Learn. Syst.* **2019**, *31*, 3127–3141. [CrossRef]
31. Saha, S.; Sarkar, J.; Dhavala, S.; Sarkar, S.; Mota, P. Quantile LSTM: A Robust LSTM for Anomaly Detection In Time Series Data. *arXiv* **2023**, arXiv:2302.08712.
32. Park, J.; Seo, Y.; Cho, J. Unsupervised outlier detection for time-series data of indoor air quality using LSTM autoencoder with ensemble method. *J. Big Data* **2023**, *10*, 66. [CrossRef]
33. Hug, R.; Becker, S.; Hübner, W.; Arens, M. Particle-based pedestrian path prediction using LSTM-MDL models. In Proceedings of the 2018 21st International Conference on Intelligent Transportation Systems (ITSC), Gold Coast, QLD, Australia, 18–21 November 2018; pp. 2684–2691.
34. Kingma, D.P. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.

Chapter 4

Publication III: Particle-resolved turbulence detection in complex plasmas using LSTM neural network

Published as:
Klein M., Dormagen N., Thoma M. & Schwarz M. (2026). Particle-resolved turbulence detection in complex plasmas using LSTM neural network. Machine Learning: Science and Technology, 7(1), 015032.

Author Contributions:

M. Klein:
Conceptualization, Methodology, Software, Validation, Formal Analysis, Investigation, Resources, Data Curation, Writing — Draft Preparation, Review and Editing, Visualization

N. Dormagen:
Conceptualization (supporting), Data Curation (supporting), Writing — Review and Editing (supporting)

M. H. Thoma and M. Schwarz:
Funding Acquisition, Project Administration, Writing — Review and Editing (supporting)

IOP Publishing Mach. Learn.: Sci. Technol. 7 (2026) 015032 https://doi.org/10.1088/2632-2153/ae3cf9

MACHINE LEARNING
Science and Technology

OPEN ACCESS

RECEIVED
30 September 2025

REVISED
29 December 2025

ACCEPTED FOR PUBLICATION
23 January 2026

PUBLISHED
6 February 2026

PAPER

Particle-resolved turbulence detection in complex plasmas using LSTM neural network

Max Klein[1,2,*], **Niklas Dormagen**[1,2], **Markus H Thoma**[1] **and Mike Schwarz**[2]

[1] I. Institute of Physics, Justus Liebig University, Giessen, Germany
[2] NanoP, THM University of Applied Sciences, Giessen, Germany
[*] Author to whom any correspondence should be addressed.

E-mail: max.klein@ei.thm.de

Keywords: turbulence detection, LSTM neural networks, machine learning, complex plasmas

Abstract

Turbulence plays an important role in fluids and plasmas, but its detection at the level of individual particles remains challenging. In this work, we introduce a long short-term memory (LSTM) neural network for the classification of turbulence in complex plasmas based on particle-resolved trajectory data. The model is trained on a large synthetic dataset of particle trajectories generated from multiple turbulent and non-turbulent flow fields, including homogeneous isotropic turbulence, channel-flow turbulence and structured non-turbulent reference fields. From basic trajectory features such as velocity, acceleration, jerk, angle change and curvature, the network learns to identify motion patterns characteristic of turbulence. The trained model achieves a classification accuracy of 99.35%, with a precision of 99.89% and a recall of 98.83% on a strictly separated test dataset, demonstrating its ability to distinguish turbulent from non-turbulent trajectories across previously unseen flow configurations. The approach is further evaluated using molecular dynamics simulations of obstacle-driven complex plasma flows. In these simulations, trajectories classified as turbulent are predominantly associated with elevated turbulent kinetic energy and vorticity, even though these quantities are not provided as explicit input features. The absence of a sharp threshold in either quantity indicates that the classification is not based on a trivial energy criterion but instead exploits additional temporal and geometric information encoded in the particle motion. Overall, this study demonstrates that an LSTM-based analysis can be used to classify turbulent and non-turbulent particle trajectories in complex plasmas using particle-bound kinematic features alone and that the resulting classifier generalizes across diverse simulated flow regimes.

1. Introduction

Although turbulence is a widespread and extensively studied phenomenon, it remains an important subject in theoretical physics and fluid dynamics. While classical flow configurations admit well-defined criteria for laminar-turbulent transitions, a universally accepted definition of turbulence at the level of individual particle trajectories is lacking. This ambiguity becomes particularly relevant for particle-resolved measurements, where turbulence must be inferred from the dynamics of individual trajectories rather than from bulk flow properties. A turbulent flow is often described as the counterpart of a laminar flow, characterized by a set of key features that distinguish these limiting regimes, while transitional or mixed states may occur in between [1]. In particle-resolved systems, such transitional behavior does not necessarily correspond to a sharp global transition but can manifest locally at the level of individual particle trajectories. A turbulent flow involves a *rotational* motion accompanied by vortices and angular momentum. In addition, turbulence is a *chaotic* phenomenon comprising abrupt fluctuations in velocity, density, pressure or temperature. Such irregularities make a turbulent flow difficult to predict in time,

as small differences in initial conditions can lead to rapidly diverging flow evolutions. Due to its rotational and chaotic behavior, a turbulent flow is *diffusive*, with an increased rate of mixing and *resistive*, introducing an amplified amount of drag compared to its laminar counterpart [2]. Lastly, a turbulent flow is characterized by its *self-similarity*. According to Kolmogorov's theory [3, 4], energy introduced at large scales is transferred successively into smaller scales by an energy cascade of vortices until it is dissipated by viscosity at the smallest scales. This process results in a scale-invariant, multi-scale structure of turbulence, which is expressed in its typical power-law distributions. In particular, the energy spectrum $E(k)$ follows a power law with respect to the wavenumber k, which is inversely proportional to the spatial scale. For fully developed, isotropic turbulence, this results in the characteristic relation $E(k) \propto k^{-5/3}$ [5]. Furthermore, a flow is characterized by its Reynolds number $Re = vL/\nu$, which is defined by the characteristic flow speed v, length scale L and kinematic viscosity ν. While high Reynolds numbers are a defining characteristic of many classical hydrodynamic turbulent flows [6, 7], turbulence-like behavior can also arise at relatively low Reynolds numbers in specific systems, such as plasmas, active fluids or flows driven by additional instabilities or external forcing [8–10]. In such systems, conventional Reynolds-number-based criteria may not be sufficient to characterize turbulence, particularly at the level of individual particle motion.

As a naturally occurring phenomenon, turbulence arises in diverse physical systems, such as water [11, 12], air [13], blood in vessels [14], active swimmers such as bacteria [15] and also plasmas [16–20]. Plasmas are ionized gases, which exhibit unique properties due to the presence of free charge carriers such as electrons and ions. When micron-sized monodisperse particles are introduced into a low-temperature plasma they quickly acquire a negative charge due to the much faster electrons. This enables long-range Coulomb interactions between the particles [21]. Since the micron-sized dust particles can be observed by camera, complex plasmas provide a suitable medium for studying particle-resolved many-body systems such as plasma crystals [22], convective flows [23] and turbulence [24–29]. The ability to resolve motion at the level of individual particles provides additional insight into the underlying dynamics of these systems. Consequently, turbulence investigations in complex plasmas rely not only on experimental studies but also on numerical simulations [30, 31].

With particle-resolved turbulent flow data, the task of turbulence detection and classification becomes increasingly challenging. The required turbulence detection algorithm differs from methods commonly used in atmospheric [32–34], astrophysical and fusion plasma studies [35, 36], which typically rely on field-based diagnostics or ensemble-averaged quantities. In contrast, the present work focuses on experimentally observable, micron-sized dust particles in complex plasmas and on the analysis of their Lagrangian trajectories. Here, it is not sufficient to verify turbulence conditions at the bulk level. Instead, each particle must be examined individually to determine whether it participates in turbulence-related motion. Previous studies have analyzed particle trajectories and accelerations in turbulent flows [37, 38], primarily in regimes where the presence of fully developed turbulence is assumed *a priori*. In such cases, an explicit particle-wise distinction between turbulent and non-turbulent motion is not required. In contrast, the present work addresses the problem of identifying turbulence-related particle motion in situations where the turbulent state is not known in advance. In complex plasma simulations, quantities such as the vorticity and the turbulent kinetic energy $\mathrm{TKE} = \frac{1}{2} m \mathbf{v}'^2$, based on the fluctuating velocity $\mathbf{v}' = \mathbf{v} - \overline{\mathbf{v}}$, where $\overline{\mathbf{v}}$ denotes the locally averaged flow velocity, are commonly used as indicators of turbulence [31]. While these measures are physically meaningful, they do not yield a clear particle-wise classification, particularly in regimes where turbulent and non-turbulent dynamics may coexist.

In particle-resolved flows, turbulence is therefore not known *a priori* and cannot be identified solely based on global flow properties. Instead, it must be inferred from the temporal and geometric structure of individual particle trajectories. This motivates the development of data-driven approaches capable of capturing such structure without explicitly prescribing turbulence-specific thresholds. To address this challenge, a particle-wise turbulence detection algorithm based on machine learning is introduced in this work. As demonstrated previously [39], long short-term memory (LSTM) based recurrent neural networks are well suited for the analysis of particle trajectories. Due to their internal memory structure, LSTM networks are able to capture temporal correlations in sequential data, making them suitable for variable-length trajectory analysis. LSTM-based models have been applied to anomaly detection in time series [40, 41], trajectory prediction [42] and plasma particle trajectory analysis [39]. In this work, an LSTM-based classifier is trained on synthetic turbulent and non-turbulent particle trajectory data generated from a diverse set of flow fields. The model operates on individual, variable-length trajectories and uses only particle-bound kinematic and geometric features derived from the particle motion. The performance of the trained network is systematically evaluated on previously unseen flow fields, providing a quantitative assessment of its ability to distinguish turbulence-related particle motion beyond the specific conditions encountered during training.

2. Methods

2.1. LSTM model

To create a turbulent particle trajectory classifier we resort to an LSTM neural network due to its successful implementation in preceding related applications [39–42]. The operating principle of LSTM networks is well suited for sequential data such as particle trajectories, as temporal dependencies can be captured through recurrent processing and internal memory states. The present task is formulated as a supervised time-series classification problem rather than as an unsupervised outlier detection task. Each particle trajectory is represented as a variable-length sequence of physically motivated features and the goal is to classify entire trajectories as turbulent or non-turbulent based on their temporal dynamics.

Here, the initialization of the LSTM network begins with a ragged input layer. This way, the network accepts particle trajectories of individual lengths, which is practical to use and avoids unnecessary padding. This is particularly relevant for particle tracking data, where trajectories can terminate early due to boundary effects, tracking loss or finite observation windows. Following the input layer, three LSTM layers with 64 units and L2 regularization of 0.01 are added. The purpose of the L2 regularization is to prevent the risk of overfitting during training. For training stabilization, a batch normalization layer is also applied afterwards. The output layer comprises a single neuron dense layer with L2 regularization of 0.001 followed by a sigmoid activation which forms a clear binary output for the final particle trajectory classification. By definition, a particle trajectory is classified as turbulent on an output of 1 and as non-turbulent on an output of 0. A schematic of the presented LSTM network architecture is shown in figure A1. A fixed decision threshold of 0.5 is used throughout this work, such that outputs greater than or equal to 0.5 are classified as turbulent and values below 0.5 as non-turbulent.

In order to assess whether the proposed LSTM-based approach provides an advantage over simpler or alternative time-series classifiers, several baseline models were implemented and evaluated. In addition to a gated recurrent unit (GRU) model with a comparable ragged-input setup, both convolutional and classical time-series baselines were tested, namely InceptionTime [43], ROCKET [44], a Time Series Forest classifier [45] and a dynamic time warping (DTW) based k-nearest-neighbor approach [46]. All baseline models were trained and evaluated using the same datasets, feature sets and train/validation/test splits as the LSTM model to ensure a fair comparison. No model-specific tuning based on test performance was performed.

2.2. Training data

The data used for training the LSTM network is generated synthetically, as there is not nearly enough experimental turbulent particle data, let alone labeled turbulent particle data. The training, validation and test datasets are generated synthetically by integrating passive tracer particles in prescribed time-dependent velocity fields. This approach allows full control over the underlying flow properties while providing particle-resolved trajectory data suitable for supervised learning. The basis for the synthetic particle trajectory generation is a randomly constructed particle cloud. In three dimensions a set number of particles is randomly placed over a set volume. These particles are then exposed to a specific time-dependent vector field. The particle movement in this vector field is then simulated in defined time steps over a fixed duration. At each time step the coordinates and velocities of every particle are recorded and serve as the basis for the training data generation. The process of particle cloud initialization and movement simulation are the same for turbulent and non-turbulent particle trajectories, while the underlying vector fields differ significantly. The distinction between turbulent and non-turbulent trajectories is therefore defined at the level of the underlying vector fields rather than at the level of individual particles and class labels are not assigned using trajectory-level thresholds or post hoc criteria.

While the generation of non-turbulent particle trajectories follows a well-defined and straightforward procedure, the generation of turbulent particle trajectories requires additional considerations to ensure physically meaningful and statistically representative motion. Since the turbulence classifier is trained entirely on synthetically generated trajectory data, it is essential that these trajectories faithfully reflect real turbulent dynamics rather than numerical artifacts. In the turbulent case, particle trajectories are generated by integrating passive tracer particles in prescribed, time-dependent velocity fields obtained from high-resolution numerical flow simulations. These velocity fields provide an Eulerian description of the flow, whereas the particle trajectories represent the corresponding Lagrangian dynamics. The tracer particles are assumed to be massless and non-inertial and do not interact with each other or with the flow, such that their motion directly reflects the underlying fluid velocity field.

As one source of turbulent dynamics, the publicly available Johns Hopkins Turbulence Database (JHTDB) [47] is used, specifically the ‘isotropic1024coarse’ dataset. This dataset is derived from a direct

numerical simulation of forced homogeneous isotropic turbulence obtained by solving the incompressible three-dimensional Navier–Stokes equations on a periodic cubic domain. The flow reaches a statistically stationary state characterized by a Taylor-scale Reynolds number of approximately $Re_\lambda \approx 433$. Temporally resolved velocity fields are provided at intervals of $\Delta t = 0.002$ over 1024 time steps, making the dataset well suited for Lagrangian particle tracking and turbulence research. In addition to homogeneous isotropic turbulence, further turbulent flow classes are included to increase the diversity of turbulent dynamics represented in the dataset. These comprise channel-flow turbulence and buoyancy-driven mixing flows, which introduce shear, boundary effects, and externally driven instabilities. Moreover, an independent test dataset based on magnetohydrodynamic turbulence is used exclusively for evaluation and is not included during training or validation.

The velocity field is queried using sixth-order Lagrangian interpolation in space (lag6) and piecewise cubic Hermite interpolation in time (pchip). Particle trajectories are computed using a second-order predictor-corrector scheme, formally equivalent to a Runge–Kutta method of second order,

$$\mathbf{x}_{n+1} = \mathbf{x}_n + \frac{\Delta t}{2}\left[\mathbf{u}\left(\mathbf{x}_n, t_n\right) + \mathbf{u}\left(\mathbf{x}_n + \Delta t \cdot \mathbf{u}\left(\mathbf{x}_n, t_n\right), t_n + \Delta t\right)\right] \tag{1}$$

where $\mathbf{x}(t)$ denotes the particle position and $\mathbf{u}(\mathbf{x}, t)$ the local fluid velocity obtained from the underlying Eulerian field.

This approach is well established in the literature. Yu *et al* [48] demonstrated its accuracy for tracer tracking using JHTDB data and similar methodologies are employed in recent studies of Lagrangian particle statistics in turbulent flows, such as those by Shnapp *et al* [49, 50]. By using varying random seeds and different initial conditions, multiple statistically independent turbulent particle trajectories are generated.

The same numerical integration framework is employed for non-turbulent particle trajectories, using analytically defined or structured, non-turbulent velocity fields. The non-turbulent dataset includes a uniform field, a circular field, a sinusoidal field, a double gyre configuration, a shear field and a vortex tube field, which are included during training and validation, as well as a helical field, a constricted field and a random-wave field, which are reserved exclusively for testing as unseen non-turbulent flow cases. An overview of some of the implemented non-turbulent flow fields is shown in figure 1. This setup provides a direct test of whether the classifier relies on specific hand-crafted flow templates rather than on more general trajectory structure.

To assess the ability of the model to generalize beyond the specific flow realizations used during training, a strict split strategy based on entire vector fields is employed. Complete flow configurations are assigned exclusively to either the training and validation or test sets and all particle trajectories originating from a given vector field are confined to a single split. As a consequence, no trajectories from the same underlying flow, run or temporal realization appear simultaneously in training and validation or testing. This applies equally to turbulent and non-turbulent data. All data preprocessing steps, including feature normalization, are performed using statistics computed solely from the training data and are applied unchanged to validation and test data. This prevents any form of data leakage arising from shared flow structures, temporal overlap or normalization across dataset splits.

Using this data generation pipeline, a total of 62 500 turbulent and 84 000 non-turbulent particle trajectories were generated. Each trajectory represents the three-dimensional motion of a passive tracer particle and consists of a time-ordered sequence of coordinates and three-dimensional velocities. The underlying simulations are performed with a fixed temporal resolution of $\Delta t = 0.08$ and all trajectories are initially generated with an identical length of 120 time steps, corresponding to the full simulation duration. The dataset is split into training, validation and test subsets based on complete velocity fields, as described above. The training set comprises 45 000 turbulent and 66 000 non-turbulent trajectories. The validation set consists of 5000 turbulent and 6000 non-turbulent trajectories, while the test set contains 12 500 turbulent and 12 000 non-turbulent trajectories generated from entirely different flow configurations that are not used during training or validation. Although all trajectories are generated with equal length of 120, a preprocessing step is applied prior to training in which each trajectory is randomly truncated to a length uniformly distributed between 40 and 90 time steps. This truncation is performed independently for each trajectory and for each training epoch. The purpose of this procedure is twofold. First, it prevents the network from relying on a fixed temporal extent as a cue for classification. Second, it mimics realistic experimental conditions, where particle trajectories are typically of variable length due to limited observation windows, particles leaving the field of view or temporary occlusions. This design choice improves the robustness of the model and facilitates its transferability to particle tracking data obtained from physical experiments.

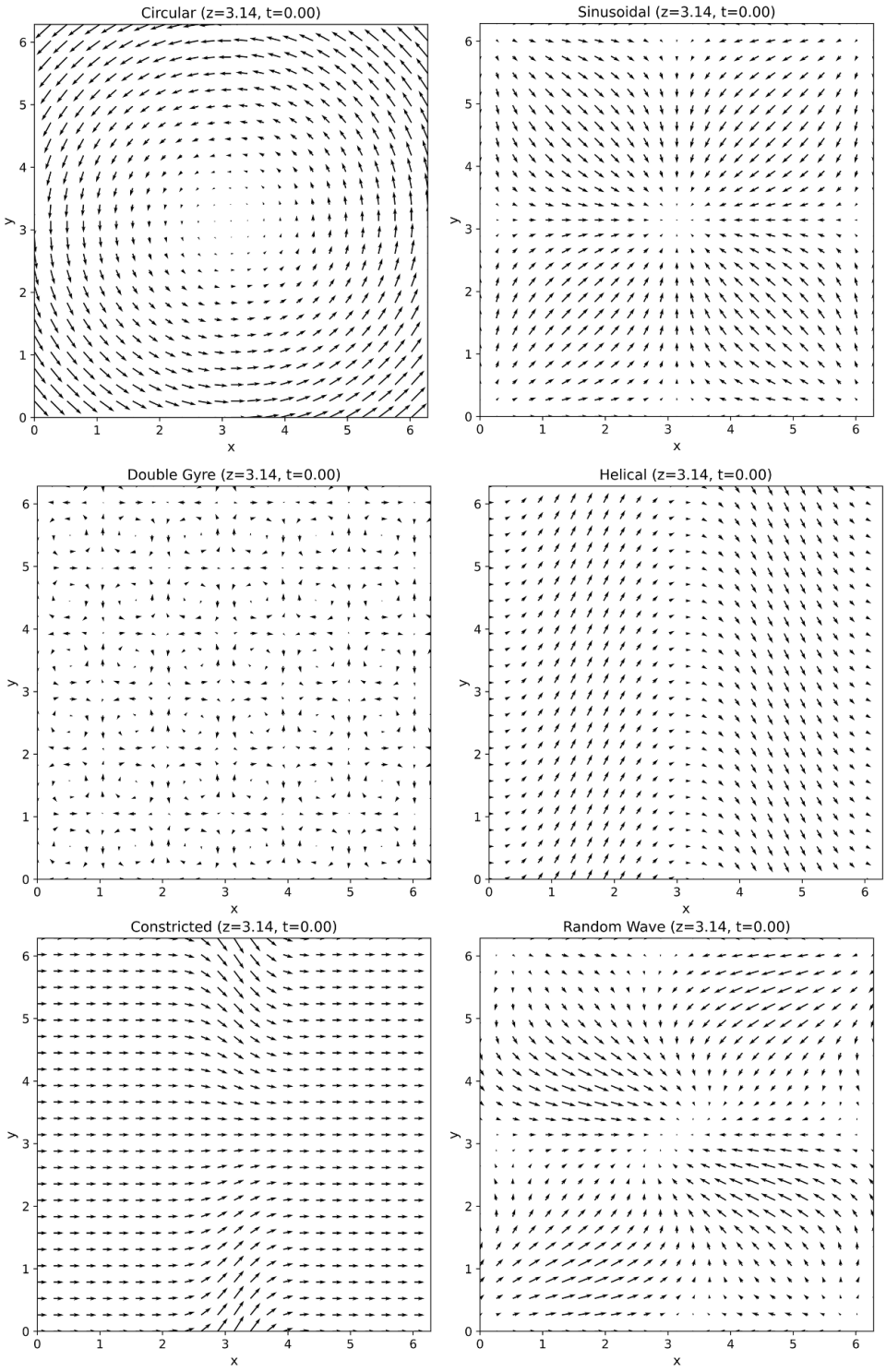


Figure 1. Vector fields used to generate synthetic non-turbulent particle trajectories. Panels (a)–(c) show the circular, sinusoidal and double gyre fields used during training and validation, while panels (d)–(f) show the helical, constricted and random-wave fields reserved exclusively for testing as unseen non-turbulent flow cases. Next to the shown vector fields the training dataset consists of a uniform field, a shear field and a vortex tube field. The arrow plots show simplified flow fields without added noise. All three-dimensional fields are shown at $z = 3.14$.

For each time step, the particle coordinates and three-dimensional velocities are computed, resulting in variable-length input sequences. The generated feature vectors are labeled as turbulent (1) or non-turbulent (0) based solely on the physical nature of the velocity field in which the trajectories were generated. No trajectory-level criteria or post hoc thresholds are applied for labeling. The LSTM and GRU

models naturally operate on these variable-length sequences through the use of ragged inputs. In contrast, baseline models that require fixed-length inputs are trained and evaluated on the same truncated trajectories after zero-padding to a fixed length of $L = 90$. This ensures that all models are exposed to identical trajectory information, while preserving a fair comparison between sequence-based and fixed-length time-series classification approaches.

However, prior to training these particle trajectories require some preparation before being input into the LSTM neural network. Since the particle coordinates and velocities are not normalized, scale-variant and presumably not directly suitable for turbulence classification, five features are computed out of these values per trajectory time step. Using the coordinates and velocities the total velocity $v(t)$, total acceleration $a(t)$, jerk $j(t)$, angle change $\theta(t)$ and curvature $\kappa(t)$ are calculated using

$$v(t) = \|\mathbf{v}(t)\| = \sqrt{v_x^2(t) + v_y^2(t) + v_z^2(t)}, \tag{2}$$

$$\mathbf{a}(t) = \frac{\mathbf{v}(t+\Delta t) - \mathbf{v}(t-\Delta t)}{2\Delta t}, \quad a(t) = \|\mathbf{a}(t)\|, \tag{3}$$

$$\mathbf{j}(t) = \frac{\mathbf{a}(t+\Delta t) - \mathbf{a}(t-\Delta t)}{2\Delta t}, \quad j(t) = \|\mathbf{j}(t)\|, \tag{4}$$

$$\theta(t) = \arccos\left(\frac{\mathbf{v}(t)\cdot\mathbf{v}(t+\Delta t)}{\|\mathbf{v}(t)\|\cdot\|\mathbf{v}(t+\Delta t)\|}\right), \tag{5}$$

and

$$\kappa(t) = \frac{\|\mathbf{v}(t)\times\mathbf{a}(t)\|}{\|\mathbf{v}(t)\|^3}. \tag{6}$$

All input features are normalized globally using statistics computed over the entire training dataset. No per-trajectory normalization is applied. This prevents the network from implicitly encoding relative energy or scale information on a per-trajectory basis and ensures that absolute feature magnitudes remain physically comparable across different trajectories. The normalization parameters derived from the training set are applied unchanged to the validation and test data.

The input features are deliberately restricted to particle-bound kinematic quantities that can be directly computed from individual trajectories, namely the magnitudes of velocity, acceleration, and jerk, as well as the local angle change and curvature of the trajectory. These features are invariant under rotations and translations and are well defined even in the absence of a spatially resolved velocity field. Using the full velocity vectors as input would implicitly introduce directional information tied to the global flow geometry, which is not generally available in experimental particle tracking data and would reduce the transferability of the approach to other systems. In contrast, the selected scalar features characterize the temporal and geometric structure of particle motion independently of a specific coordinate system. Quantities such as kinetic energy or vorticity are not included as input features by design. Both quantities require additional assumptions or spatial averaging and are commonly used as reference measures for turbulence identification. Including them directly as inputs would therefore bias the classifier toward reproducing predefined turbulence indicators rather than learning motion patterns intrinsic to the particle trajectories themselves. Instead, these quantities are used exclusively for validation and physical interpretation of the model output. The relevance of the chosen features is further examined by a systematic ablation study.

3. Results

3.1. Training results

During training, the dataset is split into training and validation subsets. To account for the strong class imbalance between turbulent and non-turbulent trajectories, the split is performed in a stratified manner, preserving the relative class proportions in both subsets. In addition, the separation is applied at the level of complete flow realizations and simulation runs, such that no trajectories originating from the same underlying flow appear simultaneously in the training and validation sets. This ensures that validation performance reflects generalization across independent trajectory sets rather than interpolation within a single flow realization. A batch size of 32 is used throughout training for the shuffled training data, resulting in 3469 update steps per epoch for the final training set.

The neural network is compiled using the *binary crossentropy* loss function to measure the difference between predicted probabilities and true labels and the *Adam* optimizer [51] with an initial learning rate of 0.0001. An early stopping mechanism is implemented that concludes the training after five epochs with no loss improvements on the validation data. Additionally, the learning rate is scheduled to decrease by half after three epochs of no loss improvement on the validation data. This adaptive scheme allowed the optimizer to gradually converge, resulting in a final learning rate of 1.0×10^{-6} at the end of training. This way the LSTM neural network training concluded after 48 epochs with a final accuracy of 99.35% on the test dataset.

After training the model reaches a recall value of 98.83% and a precision of 99.89% on the test data. The high recall demonstrates that the classifier correctly identifies almost all truly turbulent trajectories, meaning that only very few turbulent particles are missed. The equally high precision indicates that nearly all trajectories labeled as turbulent by the model are in fact turbulent, which means that false positives are rare. Together, these metrics show that the LSTM network not only achieves excellent overall accuracy, but also provides a reliable distinction between turbulent and non-turbulent motion at the particle level. The high classification accuracy observed on the test data therefore reflects the consistency of trajectory-level differences across independent flow classes rather than data leakage, as the train/validation/test split is performed flow-wise and all preprocessing is based on training data only.

To assess the contribution of individual input variables to the classification performance, we conducted an ablation study. In this analysis, the LSTM model is trained and validated on systematically reduced subsets of the available trajectory features. The influence of individual input features is analyzed using a drop-one ablation study, where the model is trained on the full feature set and on variants in which exactly one feature is removed. This enables a direct comparison of the contribution of each input channel. Results for all 31 feature subsets are reported in appendix B (figure A2).

The drop-one results show that removing the speed feature $v(t)$ has only a negligible effect on the test performance, indicating that the classifier does not rely primarily on an energy-related proxy. In contrast, removing the angular change $\theta(t)$ or the jerk magnitude $j(t)$ leads to a pronounced degradation of performance, which highlights the relevance of geometric and higher-order temporal information in the particle motion for separating turbulent from non-turbulent trajectories. Some feature combinations yield near-perfect precision or recall on the held-out test set. This does not indicate trivial separability, but reflects that certain geometric and temporal features encode highly discriminative information for this classification task, while strict flow-wise data separation and training-only normalization prevent information leakage.

In addition to the proposed LSTM model, we evaluated several established time-series classification baselines to provide a point of reference for the difficulty of the task and for the role of temporal modeling. The comparison includes a GRU with the same ragged (variable-length) input representation as the LSTM, as well as three classical baselines: InceptionTime [43], ROCKET [44], Time Series Forest [45] and DTW+kNN [46]. Table 1 summarizes the results.

The recurrent models (LSTM/GRU) operate directly on variable-length trajectories using ragged tensors and therefore do not require any padding or masking. In contrast, ROCKET, Time Series Forest and the InceptionTime baseline in our implementation operate on fixed-length inputs. For these baselines, each trajectory was converted to a fixed length of $L = 90$ time steps by zero-padding shorter sequences. The same padded representation was used for all fixed-length baselines. Time Series Forest was implemented in a channel-wise manner by training one univariate classifier per feature channel and averaging the resulting probabilities, reflecting the univariate design of the standard TSF implementation. Time Series Forest was used with 300 trees and 20 intervals per tree, extracting the standard TSF interval features (mean, standard deviation, and slope). This configuration follows the original TSF design, with minor adjustments to accommodate dataset size and computational cost. For ROCKET, 1000 random convolutional kernels were used. In this configuration, the method is computationally very efficient, with training and inference times that are orders of magnitude shorter than those of the recurrent models. Although this represents a comparatively lightweight and unoptimized setup operating on padded inputs, ROCKET nevertheless achieves strong performance, highlighting its robustness and effectiveness beyond purely speed-focused applications. In addition, we evaluated InceptionTime [43] as a strong convolutional baseline for multivariate time-series classification. We used a padded InceptionTime variant operating on the same zero-padded $L = 90$ representations. The model uses a depth-4 InceptionTime backbone with residual connections, multi-scale 1D convolutions (kernel sizes 9, 19 and 39) and global average pooling before the final sigmoid output. DTW+kNN was evaluated on the original variable-length sequences without padding. Because DTW scales quadratically in sequence length and linearly with the number of reference trajectories, we computed DTW distances using a randomly selected subset of 5000 training trajectories and 2000 test trajectories. We used $k = 3$ nearest neighbors and defined the predicted

Table 1. Comparison of the proposed LSTM model with selected deep learning and classical time-series baselines. For each method, the best-performing run on the held-out test set is reported. All models are evaluated using the same decision threshold of 0.5.

Model	Accuracy	Precision	Recall
LSTM (ragged)	0.9935	0.9989	0.9883
GRU (ragged)	0.9911	0.9868	0.9958
InceptionTime (padded) [43]	0.9914	0.9843	0.9990
ROCKET [44]	0.9660	0.9903	0.9426
Time Series Forest [45]	0.9426	0.9004	0.9978
DTW + kNN [46]	0.7655	0.7885	0.7578

turbulence score as the fraction of turbulent labels among the nearest trajectories. As DTW+kNN is evaluated on subsets of the full dataset for computational reasons, its results are not directly comparable in absolute terms to those of the other models and should be interpreted as indicative rather than exhaustive. For all models in table 1, we report performance at a fixed decision threshold of 0.5 without threshold optimization or extensive hyperparameter tuning. Consequently, the baseline results should be interpreted as conservative reference points under a unified preprocessing protocol rather than as fully optimized benchmark numbers. Within this controlled setting, recurrent sequence models yield the highest scores, while classical baselines remain competitive but clearly lower, indicating that temporal and geometric information beyond static criteria is informative for the classification task.

3.2. Complex plasma turbulence simulation

To verify the robustness and applicability of the LSTM turbulence classification it is applied to simulated complex plasma turbulence data. Simulations indicate the possibility of an onset of turbulence in the complex plasmas created by the experimental setup Plasmakristallexperiment-4 (PK-4) [31, 52]. Similar simulations are performed here to generate test data using the molecular dynamics simulator LAMMPS [53]. In these simulations, only the complex plasma particle movements are modeled, not the surrounding plasma. The particle dynamics are modeled using screened Coulomb (Yukawa) interactions, together with Langevin forcing to represent thermal fluctuations and Epstein damping to account for friction with the background gas. This setup follows the same general approach as in [31], where turbulence-like behavior is investigated in obstacle-driven complex plasma flows.

First, a simulation box with the dimensions of $10 \times 3 \times 3\,\mathrm{mm}^3$ and periodic boundaries is set up. Point charges with a particle charge of $q = -3\,481\,e$ and a Debye screening length of $\lambda_\mathrm{D} = 100\,\mu\mathrm{m}$ are initialized. To imitate real complex plasma particles with a diameter of $d = 3.38\,\mu\mathrm{m}$ and a density of $\rho = 1\,510\,\mathrm{kg\,m}^{-3}$ the point charge mass is set to $3.053 \times 10^{-14}\,\mathrm{kg}$. The number of simulated particles is chosen to reach an equilibrium average interparticle distance of $150\,\mu\mathrm{m}$. An obstacle is then introduced in the form of a fixed point charge at coordinates $(x, y, z) = (2, 1.5, 1.5)\,\mathrm{mm}$ with a significantly larger electrical charge of $3 \times 10^7\,e$. An external force is applied in the x direction to drive particle motion past the obstacle. Due to the periodic boundaries of the simulation cell this corresponds to a periodic array of obstacles. The simulation cell size is chosen such that obstacle-induced perturbations do not interact across the boundaries. The simulation is executed considering various pressure ranges, with more turbulence being observable at low pressures of around 10 Pa.

To determine whether a simulated particle is turbulent, its maximum turbulent kinetic energy TKE is calculated based on the discrepancy to the local averaged velocity in bins of size $0.2 \times 0.03 \times 0.03\,\mathrm{mm}^3$. High values of TKE suggest turbulence-related fluctuations, but do not provide a unique particle-wise label. Furthermore, the maximum vorticity of each trajectory is calculated. These two values provide physically motivated reference measures to interpret the classification results, but they are not used as network inputs.

3.2.1. Results

For our analysis a simulation at a gas pressure of 10 Pa is chosen as this provides the largest amount of seemingly turbulent particle trajectories. Of the total 116 630 trajectories the LSTM model classified 447 trajectories as turbulent and 116 183 as non-turbulent. These simulations are not used during training or validation and serve exclusively as an external test case. The network is applied to the LAMMPS-generated trajectories without any retraining or fine-tuning and all model parameters are kept fixed. The LAMMPS simulations differ substantially from the training data in terms of interaction models, forcing mechanisms and flow characteristics, such that good performance on these data cannot be attributed to similarity with the training set. This test therefore provides an independent assessment of whether the

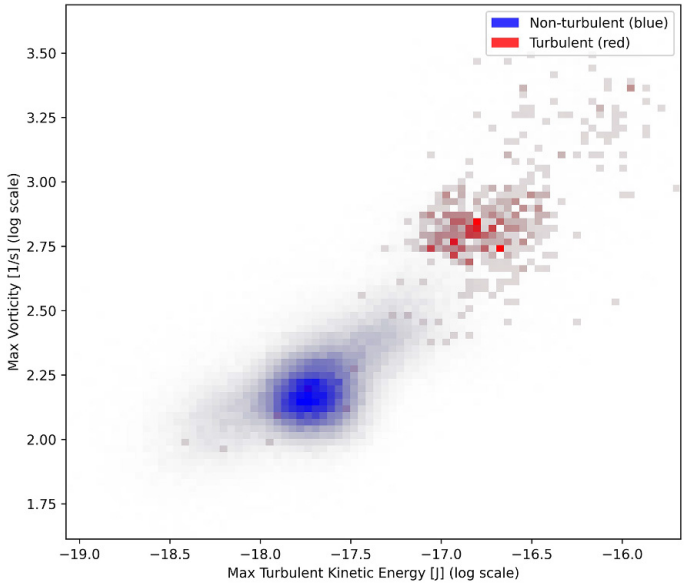


Figure 2. Joint 2D histogram of the maximum turbulent kinetic energy TKE and the maximum vorticity of individual particle trajectories from the complex plasma turbulence simulation. Trajectories classified as non-turbulent are shown in blue and trajectories classified as turbulent are shown in red. Both classes are binned using the same bin edges on a logarithmic scale and are normalized separately such that the sum over all bins equals one, yielding a probability per bin and reducing the effect of the strong class imbalance. Regions where both classes occur overlap and appear in mixed color. The distribution shows that turbulent classifications are concentrated at elevated maximum turbulent kinetic energy and vorticity, while a noticeable overlap remains.

model identifies turbulence-related particle motion patterns rather than reproducing characteristics of the training data.

Due to the nature of the data and the occurrence of turbulence in these simulations, a pronounced class imbalance is present. The number of non-turbulent particle trajectories substantially exceeds the number of trajectories classified as turbulent. Due to the pronounced class imbalance in the simulation data, the joint histogram in figure 2 is normalized separately for each predicted class. In addition, figure 3 reports the fraction predicted turbulent as a function of max TKE, together with the number of trajectories per bin. Figure 2 shows that trajectories classified as turbulent concentrate at elevated turbulent kinetic energy and vorticity, but that no strict separation exists. Importantly, the model does not reduce to a simple threshold in either quantity, as overlap persists even at comparable max TKE and max vorticity.

Since the LSTM model does not receive turbulent kinetic energy as an explicit input feature, it is necessary to assess whether the observed classification behavior could nevertheless be explained by a trivial separation in energy-related input quantities. To this end, figure 3 shows the fraction of trajectories classified as turbulent as a function of the maximum turbulent kinetic energy, binned over the full dataset. The results demonstrate that no sharp threshold in turbulent kinetic energy exists that separates turbulent from non-turbulent trajectories. In particular, a wide range of trajectories with elevated kinetic energy are consistently classified as non-turbulent and even in the highest energy bins only a small fraction of trajectories, on the order of 15%–17%, are predicted as turbulent. This behavior is incompatible with a classification based solely on energy-related input features and is consistent with the drop-one ablation results in table 2, where removing $\nu(t)$ does not remove the model's ability to distinguish turbulent trajectories.

Although the complex plasma simulation data are entirely new to the trained LSTM model, the turbulence classification appears to perform consistently, indicating strong generalization capabilities. While it is difficult to compute precise performance metrics due to the lack of a clear ground truth distinguishing turbulent from non-turbulent trajectories, the analyses based on turbulent kinetic energy and vorticity provide evidence that the model output is physically plausible and not dominated by trivial energy thresholds. The qualitative trajectory examples in figure 4 further illustrate that trajectories classified as turbulent tend to exhibit irregular direction changes and increased geometric complexity compared to non-turbulent trajectories, consistent with the geometric importance indicated by the ablation study.

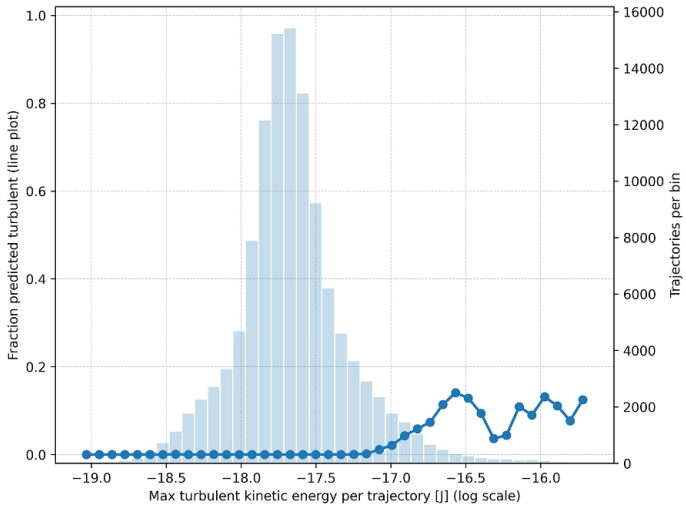


Figure 3. Fraction of particle trajectories classified as turbulent as a function of the maximum turbulent kinetic energy. Trajectories are grouped into logarithmically spaced bins and the fraction classified as turbulent within each bin is shown as a solid line (left axis). The bar plot indicates the number of trajectories per bin (right axis). No sharp energy threshold separating turbulent from non-turbulent trajectories is observed.

Table 2. Drop-one feature ablation. The same LSTM architecture is trained under identical conditions using all five features and using all but one feature. Reported are test accuracy, precision and recall at a fixed decision threshold of 0.5.

Feature set	Accuracy	Precision	Recall
$v(t), a(t), j(t), \theta(t), \kappa(t)$	0.9935	0.9989	0.9883
$a(t), j(t), \theta(t), \kappa(t)$	0.9916	0.9862	0.9974
$v(t), j(t), \theta(t), \kappa(t)$	0.9907	0.9851	0.9968
$v(t), a(t), \theta(t), \kappa(t)$	0.8054	0.7249	0.9970
$v(t), a(t), j(t), \kappa(t)$	0.6682	0.8240	0.4446
$v(t), a(t), j(t), \theta(t)$	0.9627	0.9322	0.9995

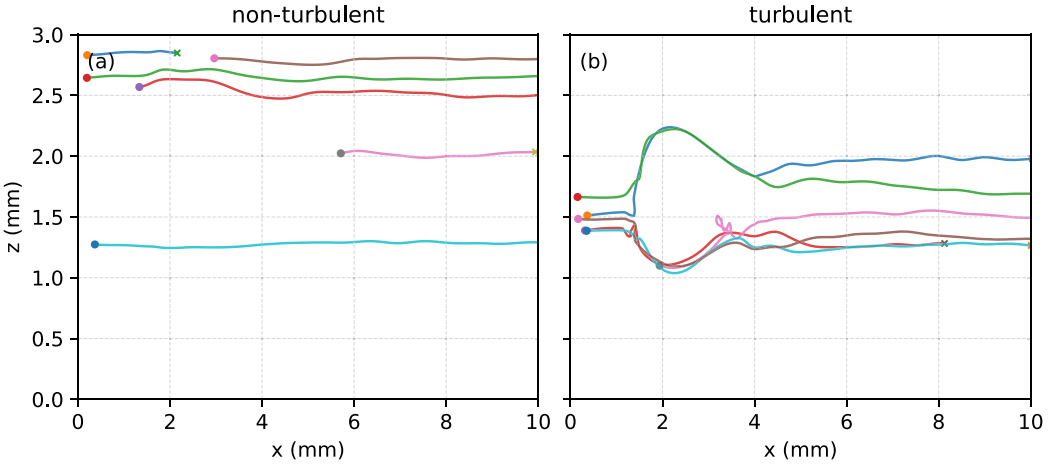


Figure 4. Sample particle trajectories shown as x–z projections from the complex plasma turbulence simulation. Panel (a) shows trajectories classified as non-turbulent and panel (b) shows trajectories classified as turbulent by the trained model. Markers indicate the start and end points of each trajectory.

4. Discussion

In this work, we have demonstrated that LSTM neural networks provide a useful framework for particle-resolved turbulence classification in complex plasmas. Trained on a large and diverse set of synthetic turbulent and non-turbulent particle trajectories, the model achieved a final accuracy of 99.35% with a precision of 99.89% and a recall of 98.83% on the held-out test dataset under strict flow-wise separation,

indicating that the classifier generalizes beyond the specific flow realizations used during training. The comparison with multiple baseline models further supports that the observed performance is not the result of a trivial classification criterion, but relies on capturing temporal dependencies in the particle motion. The high classification accuracy observed in several test cases should therefore be interpreted in the context of training-only normalization, strict split strategies and evaluation against classical time-series baselines.

Importantly, the classifier output is consistent with physically motivated reference measures when applied to independent complex plasma turbulence simulations. In the LAMMPS-based obstacle-driven simulations, trajectories classified as turbulent tend to concentrate at elevated turbulent kinetic energy and vorticity, while a pronounced overlap remains and no sharp threshold in either quantity is observed. Together with the feature ablation results, in particular the drop-one study showing that removing the velocity magnitude does not eliminate classification performance, this indicates that the model does not reduce to an energy threshold and instead exploits additional geometric and temporal information encoded in the trajectories. At the same time, these simulation-based analyses cannot provide an absolute particle-wise ground truth for turbulence, since turbulent kinetic energy and vorticity themselves do not define a unique trajectory label in transitional or mixed regimes. The results should therefore be understood as evidence that the model identifies systematic dynamical differences between trajectories originating from physically distinct flow classes and that its predictions align with established turbulence indicators without being explicitly given those indicators as inputs.

The present work does not aim to resolve the general problem of defining turbulence across all possible flow configurations. Instead, it demonstrates that particle trajectories generated in physically distinct turbulent and non-turbulent flow classes exhibit systematic dynamical differences that can be identified using data-driven methods, even across diverse and previously unseen flow types. This directly addresses the practical situation in particle-resolved systems where turbulence cannot be inferred solely from global flow properties and must be assessed from trajectory dynamics.

Several limitations remain. First, the training labels are defined at the level of the generating velocity fields rather than by a universal particle-wise turbulence criterion. This choice avoids circular definitions based on trajectory thresholds, but it also implies that transitional cases are not assigned a separate class and may naturally populate the overlap regions observed in the simulation-based reference plots. Second, the approach is currently based on a restricted set of kinematic trajectory features. While this was a deliberate design choice to ensure transferability and to avoid injecting turbulence indicators directly as inputs, it also limits the interpretability of the internal decision process to those kinematic observables. Finally, the model performance on any given application will depend on the similarity of its trajectory statistics to those represented in the training data, which motivates continued expansion of the training dataset to cover additional turbulent and structured non-turbulent regimes.

Looking forward, several directions for future research emerge. First, extending the training and test suite to further turbulent flow classes and controlled transitional regimes would allow a more fine-grained assessment of model behavior near the onset of turbulence. Second, additional interpretability analyses, for example focusing on model confidence and feature distributions in overlap regions at comparable turbulent kinetic energy, could further clarify which aspects of trajectory geometry and temporal structure drive the classification. Third, the framework can be adapted to incorporate additional trajectory-derived observables or multi-view measurements where available, while keeping the central constraint that inputs remain particle-bound and do not require a fully resolved velocity field.

Beyond the immediate context of complex plasmas, the presented method contributes to a broader class of data-driven diagnostics for particle-resolved flows. By relying solely on trajectory information, the framework is transferable to other particle tracking experiments in fluids, soft matter or active matter systems, provided that appropriate non-turbulent and turbulent reference flow classes are available for training and evaluation. In this sense, the results demonstrate that supervised sequence models can serve as practical tools to distinguish particle-level dynamical regimes in situations where a trajectory-wise turbulence label is not directly accessible from classical bulk measures.

Data availability statement

All data that support the findings of this study are included within the article (and any supplementary files).

Acknowledgments

We thank the research group around Markus H Thoma at Justus Liebig University Giessen for providing and assisting with the Plasmakristallexperiment-4 setup and the German Aerospace Society for providing powerful PCs in accordance with the funding decision, which facilitated the studies.

Funding

This project was supported by the German Federal Ministry of Economic Affairs and Climate Action under Contract No. 50WK2270B.

Author contributions

Max Klein: Conceptualization, Methodology, Software, Formal Analysis, Investigation, Visualization, Writing—Original Draft, Review & Editing. Niklas Dormagen: Validation, Investigation, Writing—Review & Editing. Markus H. Thoma: Supervision, Writing—Review & Editing. Mike Schwarz: Supervision, Writing—Review & Editing.

Appendix A. Network architecture

Figure A1 illustrates the LSTM architecture used in this work.

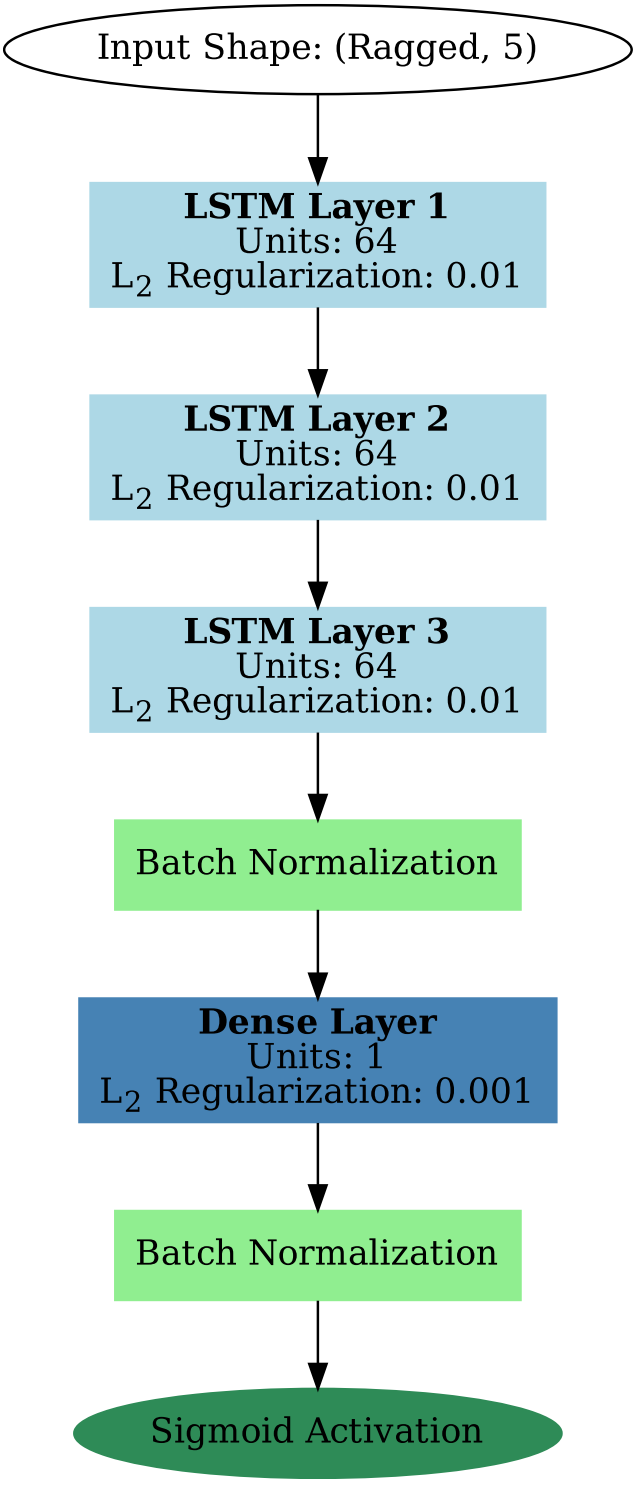


Figure A1. Architecture of the LSTM network used for turbulence classification.

Appendix B. Complete feature ablation study

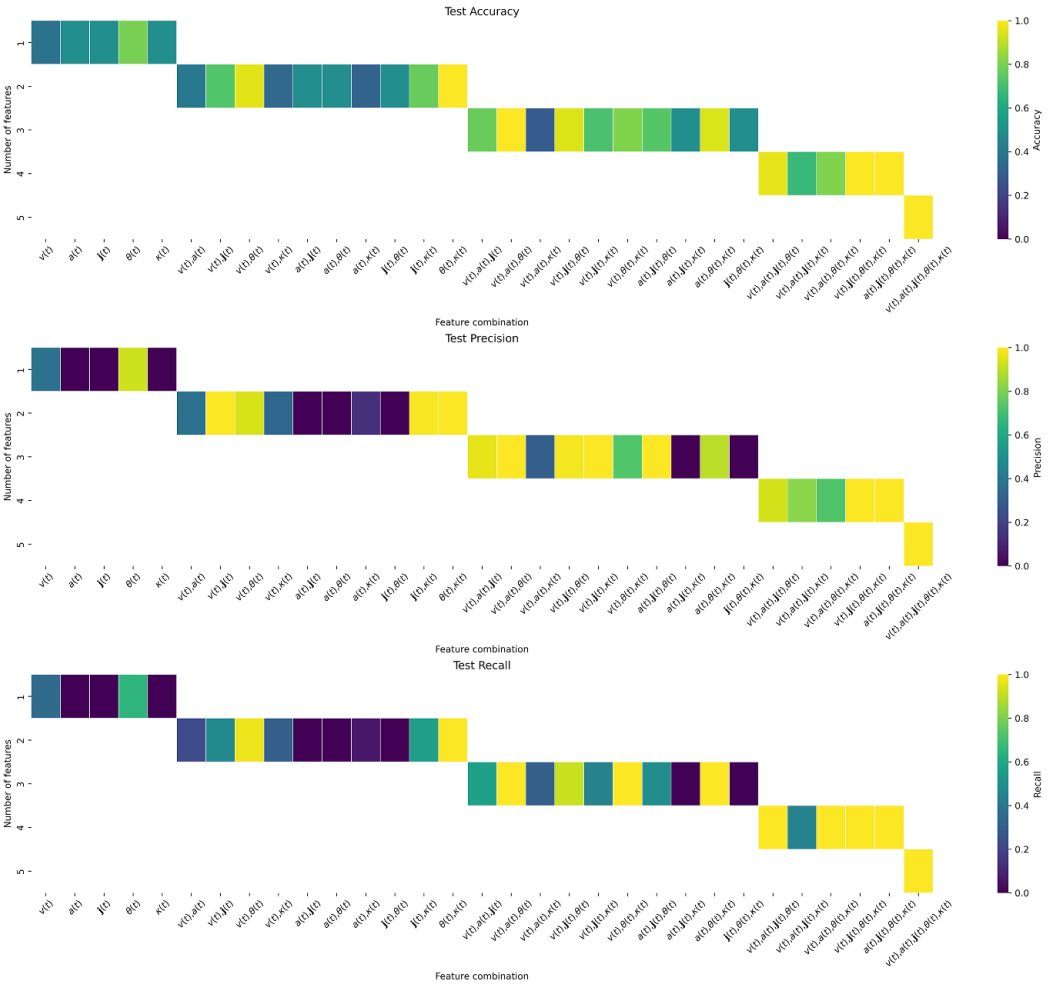


Figure A2. Complete feature ablation over all 31 subsets of the trajectory features.

ORCID iDs

Max Klein 0009-0009-9596-7671
Niklas Dormagen 0009-0001-9158-4034

References

[1] Mathieu J and Scott J 2000 *An Introduction to Turbulent Flow* (Cambridge University Press)
[2] Groisman A and Steinberg V 2000 *Nature* **405** 53–55
[3] Kolmogorov A N 1941 Equations of turbulent motion in an incompressible fluid *Dokl. Akad. Nauk SSSR* **30** 299–303
[4] Kolmogorov A N 1991 *Proc. R. Soc. A* **434** 15–17
[5] Kolmogorov A N 1995 *Turbulence: The Legacy of AN Kolmogorov* (Cambridge University Press)
[6] Hof B, Van Doorne C W, Westerweel J and Nieuwstadt F T 2005 *Phys. Rev. Lett.* **95** 214502
[7] Smits A J, McKeon B J and Marusic I 2011 *Annu. Rev. Fluid Mech.* **43** 353–75
[8] Malm A and Waigh T 2017 *Sci. Rep.* **7** 1186
[9] Wang G, Yang F and Zhao W 2014 *Lab Chip* **14** 1452–8
[10] Linkmann M, Boffetta G, Marchetti M C and Eckhardt B 2019 *Phys. Rev. Lett.* **122** 214503
[11] Nazarenko S and Lukaschuk S 2016 *Annu. Rev. Condens. Matter Phys.* **7** 61–88
[12] Falcon E and Mordant N 2022 *Annu. Rev. Fluid Mech.* **54** 1–25
[13] Busch N E and Panofsky H A 1968 *Q. J. R. Meteorol. Soc.* **94** 132–48
[14] Ghalichi F, Deng X, De Champlain A, Douville Y, King M and Guidoin R 1998 *Biorheology* **35** 281–94

[15] Peng Y, Liu Z and Cheng X 2021 *Sci. Adv.* **7** eabd1240
[16] Vedenov A A 1967 Theory of a weakly turbulent plasma *Reviews of Plasma Physics: Vol 3* (Springer) pp 229–76
[17] Hasegawa A and Wakatani M 1983 *Phys. Rev. Lett.* **50** 682
[18] Goldman M V 1984 *Rev. Mod. Phys.* **56** 709
[19] Comisso L and Sironi L 2022 *Astrophys. J. Lett.* **936** L27
[20] Marino R and Sorriso-Valvo L 2023 *Phys. Rep.* **1006** 1–144
[21] Morfill G E and Ivlev A V 2009 *Rev. Mod. Phys.* **81** 1353–404
[22] Thomas H M and Morfill G E 1996 *Nature* **379** 806–9
[23] Schmitz A S, Schulz I, Kretschmer M and Thoma M H 2023 *Microgravity Sci. Technol.* **35** 13
[24] Pramanik J, Veeresha B, Prasad G, Sen A and Kaw P 2003 *Phys. Lett. A* **312** 84–90
[25] Tsai Y Y and Chang M C I L 2012 *Phys. Rev. E* **86** 045402
[26] Tsai J Y, Lin P C and Lin I 2020 *Phys. Rev. E* **101** 023210
[27] Zhdanov S, Schwabe M, Räth C, Thomas H and Morfill G E 2015 *Europhys. Lett.* **110** 35001
[28] Schwabe M, Zhdanov S and Räth C 2017 *IEEE Trans. Plasma Sci.* **46** 684–7
[29] Hu H W, Zhao Y C and Lin I 2022 *Phys. Rev. Res.* **4** 023116
[30] Schwabe M, Zhdanov S, Räth C, Graves D B, Thomas H M and Morfill G E 2014 *Phys. Rev. Lett.* **112** 115002
[31] Joshi E, Thoma M H and Schwabe M 2024 *Phys. Rev. Res.* **6** L012013
[32] Goodwin M, Jenkins C and Lambert A 2007 *Opt. Express* **15** 14844–60
[33] Li J, Zhang M, Wang D, Wu S and Zhan Y 2018 *Opt. Express* **26** 10494–508
[34] Jiang P, Yuan J, Wu K, Wang L and Xia H 2022 *Remote Sens.* **14** 2951
[35] Conway G 2008 *Plasma Phys. Control. Fusion* **50** 124026
[36] Vlahos L and Isliker H 2016 *Eur. Phys. J. Spec. Top.* **225** 977–99
[37] Voth G A, La Porta A, Crawford A M, Alexander J and Bodenschatz E 2002 *J. Fluid Mech.* **469** 121–60
[38] La Porta A, Voth G A, Crawford A M, Alexander J and Bodenschatz E 2001 *Nature* **409** 1017–9
[39] Klein M, Dormagen N, Wimmer L, Thoma M H and Schwarz M 2025 *Mach. Learn. Knowl. Extr.* **7** 37
[40] Ergen T and Kozat S S 2019 *IEEE Trans. Neural Netw. Learn. Syst.* **31** 3127–41
[41] Saha S, Sarkar J, Dhavala S, Sarkar S and Mota P 2023 Quantile LSTM: a robust LSTM for anomaly detection in time series data (arXiv:2302.08712)
[42] Hug R, Becker S, Hübner W and Arens M 2018 Particle-based pedestrian path prediction using LSTM-MDL models *2018 21st Int. Conf. on Intelligent Transportation Systems (ITSC)* (*Maui, HI, USA, 04–07 November 2018*) (IEEE) pp 2684–91
[43] Ismail Fawaz H, Lucas B, Forestier G, Pelletier C, Schmidt D F, Weber J, Webb G I, Idoumghar L, Muller P A and Petitjean F 2020 *Data Min. Knowl. Discovery* **34** 1936–62
[44] Dempster A, Petitjean F and Webb G I 2020 *Data Min. Knowl. Discovery* **34** 1454–95
[45] Deng H, Runger G, Tuv E and Vladimir M 2013 *Inf. Sci.* **239** 142–53
[46] Petitjean F, Forestier G, Webb G I, Nicholson A E, Chen Y and Keogh E 2014 Dynamic time warping averaging of time series allows faster and more accurate classification *2014 IEEE Int. Conf. on Data Mining* (*Shenzhen, China, 14–17 December 2014*) (IEEE) pp 470–9
[47] Li Y, Perlman E, Wan M, Yang Y, Meneveau C, Burns R, Chen S, Szalay A and Eyink G 2008 *J. Turbul.* **9** N31
[48] Yu H, Kanov K, Perlman E, Graham J, Frederix E, Burns R, Szalay A, Eyink G and Meneveau C 2012 *J. Turbul.* **13** N12
[49] Brizzolara S, Neamtu-Halic M, Gambino A and Holzner M 2023 *Nat. Commun.* **14** 4195
[50] Shnapp R 2025 Velocity alignment explains Lagrangian irreversibility in turbulence (arXiv:2503.12090)
[51] Kingma D P and Ba J 2014 Adam: a method for stochastic optimization (arXiv:1412.6980)
[52] Pustylnik M *et al* 2016 *Rev. Sci. Instrum.* **87** 093505
[53] Thompson A P *et al* 2022 *Comput. Phys. Commun.* **271** 108171

Chapter 5

Complementary LAMMPS Simulation Study

As demonstrated above in publication III, trajectory-based ML methods provide a robust framework for identifying turbulent motion in particle-resolved complex plasmas. To further investigate the dependence of turbulence on external control parameters under reproducible conditions, additional numerical simulations were performed using the Large-scale Atomic/-Molecular Massively Parallel Simulator (LAMMPS) molecular dynamics framework. These simulations complement the experimental analysis by allowing systematic parameter scans. The simulation setup follows a standardized protocol introduced in [54]. Initially, a particle ensemble interacting via a screened Coulomb potential is initialized in a homogeneous configuration. The system is then allowed to relax until transient effects decay and a stationary background state is reached. Subsequently, the target Langevin thermostat and pressure are imposed using appropriate thermostating and barostatting procedures. In the Langevin thermostat, the damping parameter controls the strength of friction, while the temperature sets the amplitude of thermal fluctuations. Increasing the temperature therefore enhances stochastic forcing but does not modify the damping rate itself. After equilibration at the desired thermodynamic conditions, an external driving force is applied to the system to induce collective motion onto the implemented obstacle. Following an additional transient phase, particle trajectories are recorded over a fixed time window for further analysis. A more detailed description of the simulation process is given above in publication III and in [54]. The LAMMPS input script used in this work is provided in appendix A.

The recorded particle trajectories are analyzed using the same LSTM-based turbulence classifier introduced above. For each trajectory, the classifier assigns a turbulent or non-turbulent label based solely on trajectory-bound features. From this classification, the fraction of turbulent trajectories is computed as a percentage of all trajectories in the dataset. The resulting turbulence fractions for different parameter scans are summarized in table 5.0.1. Overall, the absolute fraction of trajectories classified as turbulent remains small and typically lies well below one percent due to the nature of the simulation. Nevertheless, clear and systematic trends can be identified. Over the investigated Langevin thermostat range, the turbulence fraction varies only weakly, indicating that the thermostat plays a

Table 5.0.1: Turbulence fraction obtained from LAMMPS trajectory data using the trained LSTM turbulence detection algorithm. The table reports the fraction of trajectories classified as turbulent, grouped by simulation parameters.

p [Pa]	T_L [K]	F [N]	Turbulent [%]
Pressure variation (fixed $T_L = 2000\,\mathrm{K}$, $F = 1 \times 10^{-13}\,\mathrm{N}$)			
5	2000	1×10^{-13}	0.0340
10	2000	1×10^{-13}	0.3841
20	2000	1×10^{-13}	0.1201
30	2000	1×10^{-13}	0.2036
50	2000	1×10^{-13}	0.0096
Thermostat variation (fixed $p = 10\,\mathrm{Pa}$, $F = 1 \times 10^{-13}\,\mathrm{N}$)			
10	1000	1×10^{-13}	0.3748
10	1500	1×10^{-13}	0.3885
10	2000	1×10^{-13}	0.3841
10	2500	1×10^{-13}	0.3859
10	3000	1×10^{-13}	0.3565
Force variation (fixed $p = 10\,\mathrm{Pa}$, $T_L = 2000\,\mathrm{K}$)			
10	2000	5×10^{-14}	0.1585
10	2000	1×10^{-13}	0.3841
10	2000	2×10^{-13}	0.0172

secondary role in the generation of turbulent particle motion under the chosen conditions. In contrast, as expected the system exhibits a pronounced dependence on pressure. Turbulent trajectories occur most frequently at very low pressures, while both even lower and higher pressure regimes show significantly reduced turbulence levels. In addition to pressure, the magnitude of the applied driving force strongly influences the occurrence of turbulent motion. For fixed pressure and temperature, moderate forcing leads to an increased fraction of turbulent trajectories, whereas both weaker and stronger forcing result in a marked reduction of turbulence. These observations indicate that turbulence in the simulated system emerges from a balance between driving strength and dissipative effects, rather than from a monotonic increase with external forcing. These simulation results demonstrate that the trajectory-based ML approach is sensitive enough to detect subtle changes in dynamical behavior across a wide parameter space. At the same time, they provide independent numerical evidence for a strong pressure dependence of turbulence in particle-resolved plasma systems, consistent with the theoretical expectations on the simulation.

Chapter 6

Conclusion and Outlook

In summary, this work demonstrates how ML methods can be systematically integrated into the analysis and operation of complex plasma experiments. Instead of treating AI as a purely auxiliary data analysis tool, this work establishes ML as a physics-guided instrument for observing, classifying and interpreting complex many-particle dynamics under realistic experimental constraints. A methodological progression is presented within the three publications forming the core of this thesis. The research commences with the identification of spatially ordered structures in two-dimensional camera images. The work then advances toward ML-based PTV and finally to the classification of complex dynamical behavior based on particle trajectories. While the individual studies address different physical phenomena, they are unified by the common design principle that physical insight is explicitly incorporated into model architecture, feature selection, training data generation and evaluation, ensuring robustness, interpretability and experimental relevance.

The first publication introduces a convolutional encoder–decoder network for the detection of string-like particle arrangements. By employing anisotropic convolutional kernels aligned with the externally applied electric field, the model directly reflects the physical origin of the observed structures. Trained exclusively on synthetic data, the network enables reliable segmentation under varying illumination, particle density and noise conditions, thereby resolving ambiguities that are difficult to address with classical threshold approaches. The presented neural network reaches an accuracy of over 95 % on manually labeled data, thereby outperforming all preceding methods and enabling further insight into ER complex plasmas. Building on this, the second publication focuses on the connection of particle trajectories over successive images using an unsupervised SOM. In contrast to threshold-based nearest-neighbor methods, the SOM provides a globally consistent matching strategy that remains robust in dense scenes and under strongly varying and high particle flow velocities. This enables reliable trajectory reconstruction without extensive parameter tuning and establishes a stable basis for subsequent dynamical analysis. The third publication extends the framework to the classification of turbulent motion in complex plasmas. Turbulence is formulated as a trajectory-level classification problem and addressed using an LSTM network operating on physically motivated, invariant trajectory features. The successful application to both simulated PK-4 data and synthetic flow fields demonstrates strong generalization capabilities and shows that subtle dynamical regimes can be identified directly

from particle-resolved motion, without relying on field reconstruction or global averaging. The trained network establishes a foundation for ML-based, particle-resolved turbulence detection in complex plasma experiments and is equally applicable to other comparable particle-resolved systems.

Beyond the published results, this dissertation demonstrates the practical feasibility of deploying ML directly at the experiment. The developed models were adapted for execution on embedded hardware in the form of an NVIDIA Jetson board under constraints on space and power consumption, employing compact architectures and mixed-precision, multiprocessing inference to achieve low-latency analysis. This establishes a sustainable and extensible ML framework that offers a real-time automation and near-real-time comprehensive data analysis. Additional numerical simulations using LAMMPS complement the experimental investigations and further validate the trajectory-based turbulence analysis. Systematic variations of pressure, Langevin thermostat and external force reveal a pronounced dependence of turbulent particle motion on pressure and external forcing, while thermostat effects remain comparatively weak within the investigated range. The observed trends demonstrate that the ML-based classifier is sensitive to subtle dynamical changes and can be applied consistently across experimental and simulated data.

Looking forward, the methods developed in this work are directly transferable to future experimental platforms. In particular, the PK-4 successor experiment COMPACT [83], with its improved three-dimensional camera diagnostics [61, 84] and higher spatial and temporal resolution, offers an ideal environment for the continued application and extension of the presented ML framework. The models introduced here are designed to be platform independent and can be deployed on the embedded systems foreseen for COMPACT, enabling automated analysis pipelines and opening the door to adaptive, feedback-controlled experiments. Moreover, the modular structure of the framework allows straightforward extension to additional physical phenomena. In this perspective, the present dissertation contributes a central building block toward a comprehensive, experiment-wide ML ecosystem for complex plasma research. Furthermore, the developed algorithms are already actively employed in ongoing research, enabling quantitative analyses of charge and ion drag force dynamics [79], investigations of ER effects on DAWs [17], the determination of the electric field in the plasma sheath [80] and supporting the classification of crystalline structures [75, 82].

Beyond complex plasmas, the concepts developed here are broadly applicable to other particle-resolved systems characterized by strong interactions and complex dynamics, including granular matter, colloidal suspensions and active particle systems. In such contexts, the combination of physics-guided model design, synthetic data generation and deployment on embedded hardware provides a powerful strategy for extracting physically meaningful information from large and noisy datasets. In conclusion, this work shows that the close integration of physical modeling and modern ML enables new ways of observing, analyzing and ultimately controlling complex many-particle systems. By embedding physical principles directly into data-driven methods and deploying them sustainably at the experiment, this dissertation contributes to the development of autonomous, adaptive and physics-informed experiments of the next generation.

Appendix A

LAMMPS Simulation Input Files

This appendix provides the LAMMPS input script used to simulate the onset of turbulence in an obstacle-driven complex plasma (see publication III and [54] for details). The listing is provided for reproducibility.

```
# Initialize variables and simulation domain
units si                          # Use SI units throughout
dimension 3                       # Three-dimensional simulation
   domain
atom_style charge                 # Charged particles (stores per-
   atom charge)
variable debye equal 100e-6       # Debye length: 100 μm
lattice fcc 50e-6                 # FCC lattice constant: 50 μm
boundary p p p                    # Periodic boundaries in all
   directions
region box1 block 0 10e-3 0 3e-3 0 3e-3 units box
                                  # Main domain: 10 mm × 3 mm × 3 mm
region box2 block 4.5e-3 8.5e-3 1.25e-3 1.75e-3 1.25e-3 1.75e-3
   units box
                                  # Subregion for initial dust
                                     placement
region simbox union 2 box1 box2   # Union of the two regions
create_box 2 simbox               # Create simulation box with 2
   atom types
create_atoms 1 region box2         # Create dust particles in
   subregion
group dust type 1                  # Group dust particles
set type 1 charge -5.57e-16        # Dust charge: -3481 e
mass 1 3.053e-14                   # Dust mass in kg
create_atoms 2 single 40 30 30      # Obstacle particle
set type 2 charge -5e-12            # Obstacle charge (~10,000×
   dust charge), ~ +3×10^7 e
```

```
mass 2 10                          # Obstacle mass 16 orders of magnitude higher than dust mass
group obstacle type 2              # Group obstacle particle

# Interactions: Debye-screened Coulomb (Yukawa) potential
variable kappa equal 1.0/${debye}  # Screening parameter kappa = 1 / Debye length
variable cutoff equal 10*${debye}  # Potential cutoff = 10 × Debye length
neigh_modify delay 0 every 1 check yes page 5000000 one 500000
pair_style coul/debye ${kappa} ${cutoff}  # Yukawa potential with cutoff
pair_coeff * * ${cutoff}                  # Pair coefficients (as specified)

# Langevin damping parameters (Epstein-like friction model, pressure-dependent)
variable dust_radius equal 1.69e-6         # Dust radius in m
variable pressure equal 150                # Neutral gas pressure in Pa
variable coeff equal 6.0899e-6             # Coefficient for thermal friction (Tn=300 K, rho=1510 kg/m^3)
variable gammaEp equal (${coeff}*${pressure})/(${dust_radius})  # Epstein damping rate
variable gamma equal 1.0/${gammaEp}        # Damping time scale
fix damping dust langevin 5000 5000 ${gamma} 38533 tally yes   # Langevin thermostat on dust group

# Step 1: Initial equilibration / distribution
fix moveatoms dust nve
fix myprint all print 100 "Step: $(step)" screen yes
thermo_style custom step temp pe ke epair etotal
thermo 100
timestep 1e-4
run 20000                          # 2.0 s

# Step 2: Melting of the lattice
fix melting dust temp/berendsen 5000 10000 1e-2
timestep 1e-3
run 5000                           # 5.0 s

# Step 3: Adjustment and relaxation at reduced pressure
unfix melting
unfix damping
variable pressure equal 30
```

```
fix damping_2 dust langevin 3000 3000 ${gamma} 298 tally yes
run 10000                        # 10.0 s

# Step 4: Adjustment and relaxation at even lower pressure
unfix damping_2
variable pressure equal 5
fix damping_3 dust langevin 1000 1000 ${gamma} 298 tally yes
run 10000                        # 10.0 s

# Step 5: Simulation settings (pressure, thermostat, forcing)
#Simulation pressure here
variable pressure equal 10
#Langevin thermostat here
fix damping_3 dust langevin 2000 2000 ${gamma} 298 tally yes
# External force applied here
fix force dust addforce 10e-14 0.0 0.0
# Start of the electric forcing
thermo 10
fix myprint all print 10 "Step: $(step)" screen yes
timestep 1e-4
run 1000                         # 0.1 s
# Start of trajectory output
dump mydump all custom 10 simulationen/press_10_temp_2000_force_10
   -14.lammpstrj id ix x y z vx vy vz
timestep 1e-4
run 10000                         # 1 s
```

Listing A.1: LAMMPS input script for obstacle-driven complex plasma turbulence simulations.

Bibliography

[1] M. Rubin-Zuzic, G. Morfill, A. Ivlev, R. Pompl, B. Klumov, W. Bunk, H. Thomas, H. Rothermel, O. Havnes and A. Fouquet, *Nature Physics* **2** (2006), 181.

[2] C. Knapek, A. Ivlev, B. Klumov, G. Morfill and D. Samsonov, *Physical review letters* **98** (2007), 015001.

[3] C. Knapek, D. Samsonov, S. Zhdanov, U. Konopka and G. Morfill, *Physical review letters* **98** (2007), 015004.

[4] H.M. Thomas and G.E. Morfill, *Nature* **379** (1996), 806.

[5] A.S. Schmitz, I. Schulz, M. Kretschmer and M.H. Thoma, *Microgravity Science and Technology* **35** (2023), 13.

[6] J. Pramanik, B. Veeresha, G. Prasad, A. Sen and P. Kaw, *Physics Letters A* **312** (2003), 84.

[7] Y.Y. Tsai, M.C. Chang and L. I, *Physical Review E—Statistical, Nonlinear, and Soft Matter Physics* **86** (2012), 045402.

[8] J.Y. Tsai, P.C. Lin and L. I, *Physical Review E* **101** (2020), 023210.

[9] S. Zhdanov, M. Schwabe, C. Räth, H. Thomas and G.E. Morfill, *Europhysics Letters* **110** (2015), 35001.

[10] M. Schwabe, S. Zhdanov and C. Räth, *IEEE Transactions on Plasma Science* **46** (2017), 684.

[11] H.W. Hu, Y.C. Zhao and L. I, *Physical Review Research* **4** (2022), 023116.

[12] A. Ivlev, G. Morfill, H. Thomas, C. Räth, G. Joyce, P. Huber, R. Kompaneets, V. Fortov, A. Lipaev, V. Molotkov et al. , *Physical review letters* **100** (2008), 095003.

[13] A.V. Ivlev, P.C. Brandt, G.E. Morfill, C. Rath, H.M. Thomas, G. Joyce, V.E. Fortov, A.M. Lipaev, V.I. Molotkov and O.F. Petrov, *IEEE transactions on plasma science* **38** (2010), 733.

[14] C. Dietz, J. Budak, T. Kamprich, M. Kretschmer and M.H. Thoma, *Contributions to Plasma Physics* **61** (2021), e202100079.

[15] R. Merlino, A. Barkan, C. Thompson and N. D'angelo, *Physics of Plasmas* **5** (1998), 1607.

[16] S. Nunomura, S. Zhdanov, D. Samsonov and G. Morfill, *Physical review letters* **94** (2005), 045001.

[17] L. Wimmer, N. Dormagen, M. Klein, M. Kretschmer, A. Lipaev, M. Schwarz, A. Usachev, O. Petrov, A. Zobnin and M. Thoma, *New Journal of Physics* **27** (2025), 033001.

[18] M. Pustylnik, M. Fink, V. Nosenko, T. Antonova, T. Hagl, H. Thomas, A. Zobnin, A. Lipaev, A. Usachev, V. Molotkov et al. , *Review of scientific instruments* **87** (2016), 093505.

[19] U. Stroth: *Plasmaphysik*. Springer, 2011.

[20] M.A. Lieberman and A.J. Lichtenberg, *MRS Bulletin* **30** (1994), 899.

[21] F.F. Chen et al. : *Introduction to plasma physics and controlled fusion*, volume 1. Springer, 1984.

[22] P.K. Shukla and A. Mamun: *Introduction to dusty plasma physics*. CRC press, 2015.

[23] J. Allen, B. Annaratone and U. de ANGELIS, *Journal of plasma physics* **63** (2000), 299.

[24] R. Kennedy and J. Allen, *Journal of Plasma Physics* **69** (2003), 485.

[25] V. Fortov, A. Ivlev, S. Khrapak, A. Khrapak and G. Morfill, *Physics reports* **421** (2005), 1.

[26] P.K. Shukla and B. Eliasson, *Reviews of Modern Physics* **81** (2009), 25.

[27] C. Goertz and G. Morfill, *Icarus* **53** (1983), 219.

[28] W.M. Winslow, *Journal of applied physics* **20** (1949), 1137.

[29] T. Hao: *Electrorheological fluids: the non-aqueous suspensions*, volume 22. Elsevier, 2011.

[30] R. Kompaneets, U. Konopka, A. Ivlev, V. Tsytovich and G. Morfill, *Physics of plasmas* **14** (2007).

[31] A. Ivlev, M. Thoma, C. Räth, G. Joyce and G. Morfill, *Physical Review Letters* **106** (2011), 155001.

[32] E. Joshi, M. Pustylnik, M.H. Thoma, H.M. Thomas and M. Schwabe, *Physical Review Research* **5** (2023), L012030.

[33] C. Räth, W. Bunk, M.B. Huber, G.E. Morfill, J. Retzlaff and P. Schuecker, *Monthly Notices of the Royal Astronomical Society* **337** (2002), 413.

[34] P. Brandt, A.V. Ivlev and G.E. Morfill, *Journal of magnetism and magnetic materials* **323** (2011), 1368.

[35] M. Pustylnik, B. Klumov, M. Rubin-Zuzic, A. Lipaev, V. Nosenko, D. Erdle, A. Usachev, A. Zobnin, V. Molotkov, G. Joyce et al. , *Physical Review Research* **2** (2020), 033314.

[36] S. Mitic, M. Pustylnik, D. Erdle, A. Lipaev, A. Usachev, A. Zobnin, M.H. Thoma, H.M. Thomas, O. Petrov, V. Fortov et al. , *Physical Review E* **103** (2021), 063212.

[37] S. Nazarenko and S. Lukaschuk, *Annual Review of Condensed Matter Physics* **7** (2016), 61.

[38] E. Falcon and N. Mordant, *Annual Review of Fluid Mechanics* **54** (2022), 1.

[39] N.E. Busch and H.A. Panofsky, *Quarterly Journal of the Royal Meteorological Society* **94** (1968), 132.

[40] F. Ghalichi, X. Deng, A. De Champlain, Y. Douville, M. King and R. Guidoin, *Biorheology* **35** (1998), 281.

[41] Y. Peng, Z. Liu and X. Cheng, *Science advances* **7** (2021), eabd1240.

[42] M.L. Goldstein, D.A. Roberts and W. Matthaeus, *Annual review of astronomy and astrophysics* **33** (1995), 283.

[43] D. Tritton. *Physical Fluid Dynamics, 519 pp., Clarendon*, 1988.

[44] J. Mathieu and J. Scott: *An introduction to turbulent flow*. Cambridge University Press, 2000.

[45] T. Carter, *Physics of plasmas* **13** (2006).

[46] U. Frisch and A.N. Kolmogorov: *Turbulence: the legacy of AN Kolmogorov*. Cambridge university press, 1995.

[47] A. Groisman and V. Steinberg, *Nature* **405** (2000), 53.

[48] O. Reynolds, *Proceedings of the royal society of London* **35** (1883), 84.

[49] A.N. Kolmogorov: *Equations of turbulent motion in an incompressible fluid. Equations of turbulent motion in an incompressible fluid*, In *Dokl. Akad. Nauk SSSR*, volume 30. (1941) pages 299–303.

[50] A.N. Kolmogorov, *Proceedings of the Royal Society of London. Series A: Mathematical and Physical Sciences* **434** (1991), 15.

[51] M. Landahl and E. Mollo-Christensen: *Turbulence and random processes in fluid mechanics*. Cambridge University Press, 1992.

[52] A.J. Smits, B.J. McKeon and I. Marusic, *Annual Review of Fluid Mechanics* **43** (2011), 353.

[53] V. Steinberg, *Annual Review of Fluid Mechanics* **53** (2021), 27.

[54] E. Joshi, M.H. Thoma and M. Schwabe, *Physical Review Research* **6** (2024), L012013.

[55] P. Bajaj, A. Ivlev, C. Räth and M. Schwabe, *Physical Review E* **107** (2023), 064603.

[56] A. Melzer et al. : *Physics of dusty plasmas*, volume 962. Springer, 2019.

[57] M. Schwabe, S. Zhdanov, C. Räth, D.B. Graves, H.M. Thomas and G.E. Morfill, *Physical review letters* **112** (2014), 115002.

[58] E.G. Kostadinova, R. Banka, J.L. Padgett, C.D. Liaw, L.S. Matthews and T.W. Hyde, *Physics of Plasmas* **28** (2021).

[59] S. Pashine, R. Dixit and R. Kushwah, *arXiv preprint arXiv:2106.12614* (2021).

[60] N. Dormagen, M. Klein, A.S. Schmitz, M.H. Thoma and M. Schwarz, *Journal of Imaging* **10** (2024), 40.

[61] M. Himpel and A. Melzer, *Machine Learning: Science and Technology* **2** (2021), 045019.

[62] M. Klein, N. Dormagen, C. Dietz, M. Thoma and M. Schwarz, *Machine Learning: Science and Technology* **5** (2024), 025050.

[63] D.P. Kingma, *arXiv preprint arXiv:1412.6980* (2014).

[64] M. Krichen, *Computers* **12** (2023), 151.

[65] O. Ronneberger, P. Fischer and T. Brox: *U-net: Convolutional networks for biomedical image segmentation. U-net: Convolutional networks for biomedical image segmentation*, In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*. Springer (2015) pages 234–241.

[66] T. Kohonen, *Proceedings of the IEEE* **78** (1990), 1464.

[67] G. Labonté, *Experiments in Fluids* **26** (1999), 340.

[68] D.B. Allan, T. Caswell, N.C. Keim, C.M. van der Wel and R.W. Verweij, (2023).

[69] Y. Bengio, P. Simard and P. Frasconi, *IEEE transactions on neural networks* **5** (1994), 157.

[70] A.P. Nefedov, G.E. Morfill, V.E. Fortov, H.M. Thomas, H. Rothermel, T. Hagl, A.V. Ivlev, M. Zuzic, B.A. Klumov, A.M. Lipaev et al. , *New journal of physics* **5** (2003), 33.

[71] H. Thomas, G. Morfill, V. Fortov, A. Ivlev, V. Molotkov, A. Lipaev, T. Hagl, H. Rothermel, S. Khrapak, R. Suetterlin et al. , *New journal of physics* **10** (2008), 033036.

[72] M. Kretschmer, T. Antonova, S. Zhdanov and M. Thoma, *IEEE Transactions on Plasma Science* **44** (2015), 458.

[73] P.S. Foundation. *Tkinter — Python interface to Tcl/Tk*. `https://docs.python.org/3/library/tkinter.html`, 2024. Accessed: 2026-03-16.

[74] M. Klein, N. Dormagen, L. Wimmer, M.H. Thoma and M. Schwarz, *Machine Learning and Knowledge Extraction* **7** (2025), 37.

[75] N. Dormagen, M. Klein, A. Schmitz, L. Wimmer, M. Thoma and M. Schwarz, *Machine Learning: Science and Technology* **5** (2024), 045006.

[76] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G.S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu and X. Zheng. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*, 2015. Software available from tensorflow.org.

[77] Deutsches Zentrum für Luft- und Raumfahrt (DLR). *42. DLR-Parabelflug*. `https://www.dlr.de/de/ar/themen-missionen/weltraumforschung/forschung-unter-weltraumbedingungen/forschungsplattformen/parabelflug/parabelflug-42`, 2024. Accessed: 2026-03-16.

[78] Deutsches Zentrum für Luft- und Raumfahrt (DLR). *44. DLR-Parabelflug*. `https://www.dlr.de/de/ar/themen-missionen/weltraumforschung/forschung-unter-weltraumbedingungen/forschungsplattformen/parabelflug/parabelflug-44`, 2025. Accessed: 2026-03-16.

[79] L. Wimmer, S. Beppler, N. Dormagen, M. Klein, M. Kretschmer, A.M. Lipaev, M. Schwarz, A.D. Usachev, O.F. Petrov, T.A. Zeller et al. , *New Journal of Physics* (2025).

[80] A.S. Schmitz, L. Hanstein, M. Klein, M. Kretschmer, C. Lotz, A. Shemakhin and M.H. Thoma, *Microgravity Science and Technology* **37** (2025), 7.

[81] M. Klein, N. Dormagen, M.H. Thoma and M. Schwarz, *Machine Learning: Science and Technology* **7** (2026), 015032.

[82] N. Dormagen, M. Klein, A. Schmitz, L. Wimmer, M. Thoma and M. Schwarz, *Machine Learning: Science and Technology* **6** (2025), 045016.

[83] C.A. Knapek, L. Couedel, A. Dove, J. Goree, U. Konopka, A. Melzer, S. Ratynskaia, M.H. Thoma and H.M. Thomas, *Plasma Physics and Controlled Fusion* **64** (2022), 124006.

[84] A. Melzer, C. Knapek, D. Maier, D. Mohr and S. Schütt, *Physical Review E* **111** (2025), 045214.

List of Publications

This section provides an overview of all scientific publications over the course of the doctoral research. Both first-author and collaborative publications are included and arranged in order of publication.

1. Dormagen N., Klein M., Thoma M. H. & Schwarz M. (2023, June). Machine learning approach for multi particle tracking in complex plasmas. In *2023 30th International Conference on Mixed Design of Integrated Circuits and System (MIXDES)* **(pp. 232-237)**. IEEE.

2. Klein M., Dormagen N., Schmitz A. S., Thoma M. H. & Schwarz M. (2023, December). Machine learning approach for particle matching, tracing and velocimetry with self-organizing map: application to complex plasmas. In *2023 International Conference on Machine Learning and Applications (ICMLA)* **(pp. 839-844)**. IEEE.

3. Dormagen N., Klein M., Schmitz A. S., Thoma M. H. & Schwarz M. (2024). Multi-particle tracking in complex plasmas using a simplified and compact u-net. *Journal of imaging*, **10**(2), 40.

4. Klein M., Dormagen N., Dietz C., Thoma M. H. & Schwarz M. (2024). Enhancing particle string detection in electrorheological plasmas using asymmetrical kernel convolutional networks. *Machine Learning: Science and Technology*, **5**(2), 025050.

5. Schwarz M., Dormagen N., Klein M., Pappert S., Schmitz A. S. & Thoma M. H. (2024, June). AI Algorithms for Plasma Particle Tracking. In *2024 31st International Conference on Mixed Design of Integrated Circuits and System (MIXDES)* **(pp. 31-36)**. IEEE.

6. Dormagen N., Klein M., Schmitz A. S., Wimmer L., Thoma M. H. & Schwarz M. (2024). Local classification of crystalline structures in complex plasmas using a PointNet. *Machine Learning: Science and Technology*, **5**(4), 045006.

7. Schmitz A. S., Hanstein L., Klein M., Kretschmer M., Lotz C., Shemakhin A. & Thoma M. H. (2025). Determination of the Electric Field by Particle Tracking in a Plasma Sheath Region during Free Fall. *Microgravity Science and Technology*, **37**(1), 7.

8. Wimmer L., Dormagen N., Klein M., Kretschmer M., Lipaev A. M., Schwarz M., ... & Thoma M. H. (2025). Impact of particle charge and electrorheology-effects on dust-acoustic waves in low pressure complex plasma under microgravity. *New Journal of Physics*, **27**(3), 033001.

9. Klein M., Dormagen N., Wimmer L., Thoma M. H. & Schwarz M. (2025). Advancing Particle Tracking: Self-Organizing Map Hyperparameter Study and Long Short-Term Memory-Based Outlier Detection. *Machine Learning and Knowledge Extraction*, **7**(2), 37.

10. Dormagen N., Klein M., Schmitz A. S., Wimmer L., Thoma M. & Schwarz M. (2025). Time-resolved classification of plasma crystals in a direct current discharge using an advanced graph neural network. *Machine Learning: Science and Technology*, **6**(4), 045016.

11. Wimmer L., Beppler S., Dormagen N., Klein M., Kretschmer M., Lipaev A. M., ... & Thoma M. H. (2025). Investigation of charge and ion drag force dynamics in complex plasma experiments with neon and argon under microgravity. *New Journal of Physics*, **27**(11), 114301.

12. Klein M., Dormagen N., Thoma M. H. & Schwarz M. (2026). Particle-resolved turbulence detection in complex plasmas using LSTM neural network. *Machine Learning: Science and Technology*, **7**(1), 015032.

Declaration of Authorship

I declare that I have completed this dissertation single-handedly without the unauthorized help of a second party and only with the assistance acknowledged therein. I have appropriately acknowledged and cited all text passages that are derived verbatim from or are based on the content of published work of others, and all information relating to verbal communications. I consent to the use of an anti-plagiarism software to check my thesis. I have abided by the principles of good scientific conduct laid down in the charter of the Justus Liebig University Giessen *„Satzung der Justus-Liebig-Universität Gießen zur Sicherung guter wissenschaftlicher Praxis“* in carrying out the investigations described in the dissertation.

Information on the use of artificial intelligence (AI)-based tools such as ChatGPT or SchulKI by OpenAI or Gemini by Google in the preparation of this dissertation (please tick as appropriate):

☐ I did not use any AI tool in the preparation of this text.

☒ I used an AI tool in the following areas (multiple selections possible):

- ☐ Generating ideas and stimulating creativity
- ☐ Understanding concepts, researching facts and definitions
- ☒ Optimizing text written by myself
- ☐ Generating entire text passages based on my instructions

The following AI tools were used, and the respective parts of my text benefited from them as follows:

- I used DeepL as a translation aid for individual formulations and, to a limited extent, for the linguistic refinement of text passages that I had written myself.

Date: _____ Signature: _____

Acknowledgments

First, I would like to thank both of my supervisors Prof. Dr. Markus H. Thoma and Prof. Dr. Mike Schwarz for their continuous support, guidance, trust and the many opportunities they provided throughout my doctoral studies. Their expertise and encouragement were invaluable for both my professional and personal development. I also thank the other members of my thesis committee Prof. Dr. Simone Sanna and Prof. Dr. Stefan Schippers for their time and valuable feedback.

I gratefully acknowledge the funding of this work by the DLR and the German Federal Ministry of Economic Affairs and Climate Action under grant no. 50WK2270B.

I thank Dr. Michael Kretschmer and Thomas Nimmerfroh for their assistance during both the experimental work and the parabolic flight campaigns. I am also grateful to Andreas S. Schmitz and Dr. Lukas Wimmer for the excellent collaboration and many fruitful discussions within and beyond the university, as well as for their friendship.

I would especially like to thank Niklas Dormagen, who, as a colleague, went through the same challenges as I did and, as a close friend, accompanied me throughout this entire journey. I am deeply grateful for his critical perspective, his constructive feedback and for consistently being a reliable discussion partner. Beyond that, I truly value his friendship and ongoing support, which made this journey significantly easier. His input played a crucial role in both the direction and the quality of this work. I will always value the many experiences we shared throughout this time.

I am deeply grateful to my parents Birgit and Timo for their love and support throughout my life as well as to my brother Kevin, who always inspired me and encouraged my interest in physics.

Above all, I owe my deepest gratitude to my wife Larissa for her constant support, strength and unwavering belief in me. Without you by my side, I would not have been able to complete this work. Thank you for your love, your positivity, for always listening and most importantly for the greatest gift of my life: our daughter Isabella.