CMInject: Python framework for the numerical simulation of nanoparticle injection pipelines

Simon Welker,[1] Muhamed Amin,[1,2,3] and Jochen Küpper[1,4,5,*]

[1] *Center for Free-Electron Laser Science, Deutsches Elektronen-Synchrotron DESY, Notkestraße 85, 22607 Hamburg, Germany*
[2] *Department of Sciences, University College Groningen, University of Groningen, Hoendiepskade 23/24, 9718 BG Groningen, Netherlands*
[3] *Groningen Biomolecular Sciences and Biotechnology Institute, University of Groningen, Nijenborgh 4, 9747 AG Groningen, Netherlands*
[4] *Center for Ultrafast Imaging, Universität Hamburg, Luruper Chaussee 149, 22761 Hamburg, Germany*
[5] *Department of Physics, Universität Hamburg, Luruper Chaussee 149, 22761 Hamburg, Germany*

(Dated: 2021-07-09)

CMInject simulates nanoparticle injection experiments of particles with diameters in the micrometer to nanometer-regime, e. g., for single-particle-imaging experiments. Particle-particle interactions and particle-induced changes in the surrounding fields are disregarded, due to low nanoparticle concentration in these experiments. *CMInject*'s focus lies on the correct modeling of different forces on such particles, such as fluid-dynamics or light-induced interactions, to allow for simulations that further the scientific development of nanoparticle injection pipelines. To provide a usable basis for this framework and allow for a variety of experiments to be simulated, we implemented first specific force models: fluid drag forces, Brownian motion, and photophoretic forces. For verification, we benchmarked a drag-force-based simulation against a nanoparticle focusing experiment. We envision its use and further development by experimentalists, theorists, and software developers.

Keywords: Nanoparticles, Injection, Numerical simulation, Single-particle imaging, X-ray imaging, Framework

PROGRAM SUMMARY

Program Title: CMInject
CPC Library link to program files: (to be added by Technical Editor)
Developer's repository link: `https://github.com/cfel-cmi/cminject`
Code Ocean capsule: (to be added by Technical Editor)
Licensing provisions: GPLv3
Programming language: Python 3
Supplementary material: Code to reproduce and analyze simulation results, example input and output data, video files of trajectory movies
Nature of problem: Well-defined, reproducible, and interchangeable simulation setups of experimental injection pipelines for biological and artificial nanoparticles, in particular such pipelines that aim to advance the field of single-particle imaging.
Solution method: The definition and implementation of an extensible *Python 3* framework to model and execute such simulation setups based on object-oriented software design, making use of parallelization facilities and modern numerical integration routines.
Additional comments including restrictions and unusual features: Supplementary executable scripts for quantitative and visual analyses of result data are also part of the framework.

1. INTRODUCTION

Single-particle imaging (SPI) with x-ray beams is a relatively new technique [1, 2] for the imaging of small particles down to the size of single macromolecules, promising to image nanometer-sized particles without the need for crystallization. In this context a "particle" can be anything from a small molecule to an entire protein or an artificial nanoparticle. In SPI, a beam of x-rays illuminates single particles in flight, with each particle hit by the x-ray pulse producing a diffraction pattern. From a collection of such patterns gathered from many identical particles, the particle 3D structure can be approximated. Substantial advances were made on the capabilities of x-ray free-electron lasers (XFELs) in recent years [3, 4], offering brilliant and collimated ultra-short pulsed x-ray beams that can outrun radiation damage to the sample [1, 5] and allow for time-resolved imaging on femtosecond timescales [6, 7].

There are multiple factors to consider for collecting and reconstructing electron densities and molecular structures with high resolution: Incident x-ray intensity, experimental repetition rate, and particle density in the interaction region. They all affect the quality of the reconstructed structure: increasing the incident intensity results in more signal in each diffraction pattern, and increasing the repetition rate or particle density results in more diffraction

* Email: jochen.kuepper@cfel.de; website: https://www.controlled-molecule-imaging.org

patterns being collected in the same timespan. It was suggested that incident laser intensity is not the limiting factor [8], which was corroborated by showing that the level of signal contained in collected patterns can be reduced drastically while maintaining good reconstruction quality [9]. However, a large number of good hits, i. e., diffraction patterns of single particles inside the focus of the x-ray pulse, need to be collected in any case. It was previously noted in the literature that "different injection strategies to extend XFEL imaging to smaller targets, such as single proteins" are needed [10], and that "improvements could be made through optimized focusing for the targeted size distribution or cryogenic injection systems that additionally allow conformational selection" [11]. Therefore, there is an urgent need for novel optimized particle injection systems.

To recover the 3D structure of the imaged particles from their 2D diffraction patterns, sophisticated computer algorithms are used [9, 11–13]. These algorithms use diffraction patterns from structurally identical particles. Thus, it is important to understand how the variation in particles' sizes/shapes and structural conformations will affect their trajectories in the injection system. These trajectories are also dependent on several experimental parameters, e. g., the geometry of the injection system, the temperature and pressure of the guiding aerosols, and the initial phase space distribution of the injected particles [14, 15]. Accordingly, selecting specific particle species, e. g., through the use of inhomogeneous electric fields [16], are an advanced topic for creating a high-quality particle beam.

A simulation framework provides a quick and efficient tool for searching the experimental parameters' space and to produce optimized molecular/nanoparticle beams. Furthermore, the feedback loop between simulation and experiment offers a road to progress in both theoretical and experimental physics. Simulations are repeatedly used as a basis, supplementary, guiding, or verification method in SPI research. Examples for this are (1) optimization of experimentally verified aerodynamical injector designs for a variety of specific particle sizes and materials [15, 17–21], (2) exploration of the effects of experimental injection parameters [22] and types of injectors [23] on diffraction patterns, and (3) control of shock frozen isolated particles of both biological and artificial origin [14]. Progress in all of these areas was the foundation of recent significant improvements of the amount of data that can be collected in a given timeframe in SPI experiments [11]. We therefore propose that providing a problem-tailored yet extensible simulation framework, as previously done by our research group and collaborators [16, 24, 25], will further help progress in the field of nanoparticle injection.

Here, we introduce and describe *CMInject*, a computational framework that aims to be an extensible basis for such simulations.Figure 1 depicts examples of simulation results, indicating that recent developments, as well as future ideas, are supported by our framework.

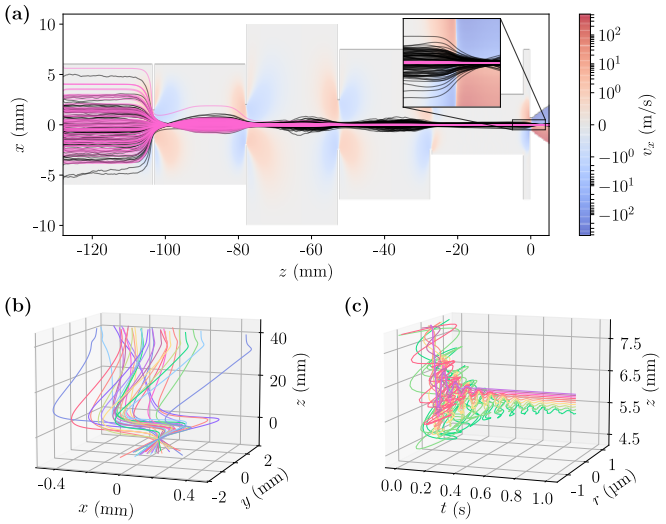

FIG. 1. Example trajectory plots of experiments simulated with *CMInject*: **(a)** 2D trajectories from a simulated focusing experiment using an axisymmetric aerodynamic lens stack (ALS) to focus $d = 27\,\mathrm{nm}$ gold nanoparticles [21]. The simulations include (black) or disregard (pink) Brownian motion. The background shows the fluid's velocity in x direction. **(b)** 3D trajectories of a focusing experiment [14], where a cooled buffer gas cell (BGC) focuses $d = 490\,\mathrm{nm}$ polystyrene nanoparticles by a flowing carrier gas at a cryogenic temperature (4 K). The used force model is a new theoretical development for particles moving in the molecular flow regime and at low temperatures [26], see Section 3.3.4. Trajectory colors indicate different starting positions for otherwise identical particles. **(c)** Qualitative reproduction of an optical trapping and levitation experiment [27], showing the interplay of photophoretic forces [28], gravity, and air resistance. Particles with different masses — indicated by the trajectory colors: heaviest in green, lightest in purple — settle into different equilibrium positions over time.

2. PROBLEM DESCRIPTION

Creating high-quality nanoparticle beams poses diverse technical and scientific challenges [14, 15, 23, 29]. The development of improved or sample-adjusted injection pipelines [15] needs to be supported by a flexible and extensible simulation package, which enables quantitative predictions of arbitrary nanoparticle injection pipelines. Possibilities to easily implement additional virtual detectors, particle types, and force fields are crucial for the usability in a wider scientific context. Capabilities for the subsequent visualization and analysis of simulation results, on their own or in comparison with experimental data, are also important.

Reasonable assumptions within the set of possible simulated experiments were made when designing the initial computational framework presented here: (1) particles do not interact with each other nor do they affect the surrounding environment. (2) the particles' overall motion is predominantly in the direction of a designated spatial axis, which we refer to as z. Assumption (1) is appropriate

for our experiments of interest and makes implementation easier. If necessary, it could be relaxed by writing a parallelized implementation of particle-particle interactions, e. g., using established approaches from molecular-dynamics software [30].

Further points of interest for nanoparticle injection are the separation of species, e. g., by quantum state, conformation, or enantiomer [16, 31, 32], the alignment or orientation in space [33–37], or the preservation of native biological structures [14, 38, 39]. These are not yet implemented in *CMInject* and will not be discussed further in this paper, but we designed our framework foreseeing corresponding as well as unforeseen extensions.

Furthermore, the framework must be usable by theorists and experimentalists alike in order to evaluate and exchange ideas and experiments for nanoparticle injection. The framework should strike a balance between expressiveness and processing requirements: a long procedural script, written with optimized functions, might run simulations very quickly, but is likely incomprehensible to most potential developers and users. A very general framework, while intuitively usable by users and developers, might in turn require so much dynamicism in its implementation that simulations become unsuitably slow.

3. FRAMEWORK DESCRIPTION

CMInject enables theorists and experimentalists to work together toward inventing or optimizing nanoparticle injection pipelines [40]. *CMInject* is written in Python 3 and its design follows an object-oriented paradigm. Most objects in this framework represent real-world counterparts that are present in actual experiments. For example, a user might create a `Setup` instance, passing along one or more `Source` instances that generate particles, and one or more `Device` instances that affect particles throughout their simulated trajectories by simulating physical forces. The user can `run()` a concrete `Experiment` constructed by the `Setup` and observe the returned results: a list of `Particle` instances that have been updated and, if desired, tracked along each particle's trajectory.

CMInject does not impose many explicit constraints on how specific objects need to behave, it only requires that all parts of an `Experiment` work with each other in a well-defined way. For example, while all currently implemented sub-types of `Particle` are spherical objects, *CMInject* is in principle agnostic to the particle shape. If someone wished to, for example, simulate elliptical articles in a fluid flow, they could do so by (i) defining a `Particle` subclass `EllipticalParticle` with additional shape parameters, e. g., r_x, r_y, r_z for an ellipsoid and (ii) deriving an implementation of the `FluidFlowField` class to be able to handle these new particles by an appropriate force model.

Going further, one could even implement the manipulation of molecules by electric fields using the quantum-mechanical Stark effect [16, 25], something we are foreseeing for the near future.

3.1. Framework structure

CMInject's framework structure consists of:

1. a set of abstract definitions corresponding to real-world experimental objects, with a prescribed way of constructing a *virtual experiment* out of these objects.
2. a parallelized routine that uses numerical integration to generate particle trajectories through a virtual experiment.
3. supplementary executable scripts, mostly for the analysis of result data.
4. implementations of the abstract definitions for the concrete physical models listed in Section 3.3.

3.1.1. Base class definitions

The following list provides the base classes [41] of *CMInject* implemented in the `cminject.base` and `cminject.experiment` modules, including brief versions of their documentation. The full documentation is attached in the supplementary materials and updated versions are available at `https://cminject.readthedocs.org`.

`cminject.base`:

- `Particle`: A particle whose trajectory we want to simulate. First and foremost a simple data container.
- `Field`: An acceleration field acting on `Particle`s.
- `Action`: Updates the properties of a `Particle` after each integration step. Useful for changes over time that are not described by the ordinary differential equations in Section 3.3.1.
- `Boundary`: A spatial boundary, evaluates if a `Particle` is inside its spatial extent.
- `Device`: A combination of `Field`s, `Action`s, and a `Boundary`, modeling real-world experimental devices. Applies the effects of its `Field`s and `Action`s only if a particle is inside of its `Boundary`; otherwise does not affect the particle in any way.
- `Detector`: Evaluates if and where a `Particle` interacted with it.
- `ResultStorage`: Stores experiment results to disk, and offers convenience methods to read them back into memory later.

- `Setup`: Akin to a laboratory experimental setup with changeable pieces and parameters. Exposes a set of parameters that can be changed by the user, and constructs an `Experiment` instance from them that can then be simulated.

`cminject.experiment`:

- `Experiment`: Akin to a real-world experiment which has a fixed set of sources, devices, detectors, and experimental parameters. Contains one or more instances of all of the classes from `cminject.base` listed above (except for `Setup`, which constructs `Experiment` instances). Constitutes the entry point for simulation, and returns the results.

3.1.2. Numerical and technical implementation

To numerically solve the particle trajectories for any virtual experiment, we used the numerical integration routine LSODA [42] as offered by the `scipy.ode` module [43]. This routine was chosen for its automatic method switching for stiff and non-stiff problems [42], which is very useful in our generic multiphysics framework where various forces make up an ODE system that can exhibit different degrees of stiffness at different positions in space of the same experiment.

For storing trajectories the integrator is instructed to piecewise integrate from the current time t up until $t + \mathrm{d}t$ with a chosen macro-timestep $\mathrm{d}t$. Note that the integrator may automatically choose to calculate many smaller timesteps in each macro-timestep, which does happen by default and thus $\mathrm{d}t$ does not negatively affect the accuracy of the integration. However, the size of $\mathrm{d}t$ determines how often additional actions, e. g., Brownian motion updates, detectors, or termination checks, are executed as these actions occur for every trajectory point, see Section 3.2 for further details. We picked a default macro-timestep $\mathrm{d}t = 10$ µs, which we found appropriate for our simulations. The user can adjust this value in a tradeoff between the required resolution of the trajectories and the duration of calculations.

These integration calculations are, as well as most other calculations in *CMInject*, heavily based on *NumPy* arrays [44]. We wrote a parallel implementation based on the `multiprocessing` module offered by the Python 3 core library, letting simulated particles be processed in parallel by a pool of $\omega \in \mathbb{N}$ worker processes, where by default ω is the number of available CPU cores. We use the automatic optimization library *Numba* [45] as well as the compiled language *Cython* [46], for automatic and manual optimization of the calculation functions, respectively.

3.1.3. Executable scripts

CMInject is supported by a collection of executable scripts. The main program, `cminject`, simulates a specified setup, passing along mandatory and optional parameters and providing documentation for them if needed, and writes the results to a specified output (HDF5) file. `cminject_visualize` and `cminject_analyze-asymmetry` support the user's analysis of the result files: They provide visualization and metrics of beam profile asymmetry, respectively. Documentation for all utility programs is provided with the software.

3.2. Program flow of simulation runs

To provide a foundation for further discussion of the generality and possible improvements, we provide a description of the general program flow of a *CMInject* simulation. A listing of the steps involved in the current implementation is given in Algorithm 1. To clarify the short descriptions given there, we note the following: A particle is considered "done" if it is outside of all `Boundary` objects, or if its current time is outside of the simulation time-window. Whether integration is successful is determined by the integrator. When a `Detector` detects a given particle, it stores a detection event on the `Particle`, so this event is stored in the result list. `Action`s can change a particle's phase space position, and if this happens, it is taken into account for the integration routine by resetting the integrator accordingly.

Algorithm 1 Program flow of a *CMInject* simulation.

1. Get particles from all `Source`s, merge into one list
2. Initialize an empty result list
3. For each particle, parallelized via `multiprocessing.Pool`:
   (a) Initialize integrator: $t = t_0, x = x_0$
   (b) If particle "not done" and integration successful:
   - Update particle phase space position from integrator
   - Update done-ness of particle using every `Boundary`
   - Let each `Action` update the particle
   - If particle position changed, reset the integrator
   - Let each `Detector` try to detect the particle
   - Update t, by incrementing it by the time step $\mathrm{d}t$
   - Run the integrator until t. At each evaluated point:
     – Consult each `Device`'s applicable `Field`s
     – Sum all accelerations
   - Go to (b)

   (c) Store fully simulated particle in the result list
4. Store the result list as an HDF5 file

Figure 2 simplifies the description given in the step-by-step listing, Algorithm 1 to a higher-level form and omits implementation details in favor of general concept. We anticipate that the community will discuss and optimize, or even replace, the concrete implementation further, while keeping the conceptual program flow as illustrated

in Figure 2 fairly consistent across future versions of *CMInject*.

3.3. Physics models

This first release of *CMInject* provides several physical models that are briefly described in the following.

3.3.1. Newton's equations of motion

We treat particle trajectories as a collection of incremental numerical solutions to the initial value problem:

$$\begin{aligned} \phi'(t) &= f(t, \phi(t)) \\ \phi(t) &\coloneqq (x, y, z, v_x, v_y, v_z)^T(t) \\ \phi(0) &\coloneqq \phi_0 = (x_0, y_0, z_0, v_{x,0}, v_{y,0}, v_{z,0})^T \\ f(t, \phi(t)) &\coloneqq (v_x, v_y, v_z, a_x, a_y, a_z)^T(t) \end{aligned} \tag{1}$$

ϕ is a time-dependent vector in $(2n)$-dimensional phase-space, with $n = 3$ in the general case or $n = 2$ for axially symmetrical simulations. v_i are the velocities and a_i the accelerations corresponding to spatial dimension i. $\vec{a} = \vec{F}/m_p$, with the total force $\vec{F}$ exerted on a particle having mass m_p.

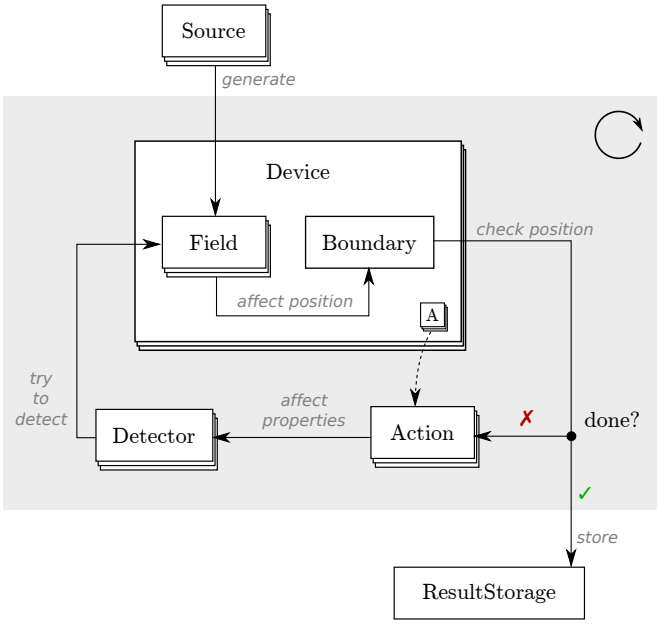

FIG. 2. Conceptual program flow of a particle simulation with *CMInject*, following a single particle through a collection of objects instantiated from the classes provided by the framework. Solid arrows follow the particle's path; their grey annotations show the effect each object can have on the particle. The shaded background indicates the integration loop, which is repeated until the particle's simulation is considered "done". Classes displayed as a stack of layered rectangles, like `Source`, imply that a simulation can contain more than one object of such a class. The stack simply labeled "A" and the dashed arrow to the `Action` stack indicate that each `Device` can contain `Actions`, which only affect particles inside that `Device`.

3.3.2. Stokes' drag force

We use Stokes' model for the drag force of an isolated spherical particle embedded in a flowing medium [47] for very small Reynolds numbers $\mathrm{Re} \ll 1$, which is essential to our simulations of aerodynamical focusing. It is formulated in terms of the fluid dynamic viscosity μ, particle radius r, particle mass m, difference in velocity between fluid and particle Δv, and a Cunningham slip-correction factor C_c [48]. For room-temperature ALS simulations we used an empirical slip-correction-factor model valid for high Knudsen numbers [49]. For cryogenic temperatures, e. g., 4 K, we used a temperature-dependent model [50] scaled by an experimentally determined factor of 4, detailed further in previous work [14]:

$$\vec{F}_{\mathrm{Stokes}} = \frac{6\pi\mu r\vec{\Delta}v}{C_c} \tag{2}$$

3.3.3. Brownian motion

Since we model nanoparticle injection, Brownian motion becomes non-negligible, especially for smaller nanoparticles. The model for Brownian motion used is that of a Gaussian white-noise random process with a spectral intensity S_0 taken from Li and Ahmadi [51].

$$\begin{aligned} \vec{a}_{\mathrm{Brown}} &= \vec{\mathcal{N}}(0, 1, k)\sqrt{\frac{\pi S_0}{\Delta t}} \\ S_0 &= \frac{216\mu k_B T}{\pi^2(2r)^5\rho^2 C_c} \end{aligned} \tag{3}$$

$\vec{\mathcal{N}}(0, 1, k)$ denotes a vector of k entries, each being independently and randomly drawn from a zero mean unit variance normal distribution. Δt is the duration of the time-interval over the force should be calculated, which is the time increment of each integration step. r, m, μ and C_c are the same quantities as defined in Section 3.3.2. k_B is the Boltzmann constant, T is the temperature of the fluid, and ρ is the density of the particle material.

3.3.4. Microscopic drag force

For the simulation of nanoparticles moving through fluids with a wide range of pressures, velocities, and temperatures, Stokes' drag force is often not well applicable. Thus, a new drag force model based on the kinetic theory of gases was developed [26]. The original formulation [52] of this model was extended to broad sets of conditions encountered in novel nanoparticle injection experiments, for instance, temperatures as low as 4 K [14]. This force

is defined as a combination of 10 % specular reflections and 90 % diffuse reflections and takes into account the time-dependent temperature difference between the injected particles and the fluid. An accompanying model for Brownian motion was also provided [26].

3.3.5. *Photophoretic force*

Furthermore, a model of the photophoretic force, i. e., the force of the surrounding gas exerted on an anisotropically radiatively-heated particle. This has found various applications in the physical and biological sciences [53] and has also been exploited for controlling and focusing particle beams [27, 54–57]. A full theoretical description is not available [28] and we have implemented an approximate force model described and benchmarked before [28]. It assumes a Laguerre-Gaussian laser beam of order 1, and uses a phenomenological constant κ to model the axial and transverse components separately. A description of how we implemented this model, which closely follows the publication by Desyatnikov, is given in the supplementary information.

4. SIMULATION RESULTS AND COMPARISON WITH EXPERIMENT

To verify baseline correctness of our framework, we benchmarked it against particle distributions from experiment [21]. There, $d = 27$ nm gold spheres were injected into vacuum in an electrospray-ionization setup, passed through a differential pumping stage to remove background gas, and then guided into an optimized aerodynamic lens stack (ALS) [15, 21]. The 1D position distributions, an arbitrary cross-section of the true 2D distribution assuming axisymmetry around the z axis, was measured at various distances from the exit of the ALS along the propagation axis z.

To simulate this experiment we used the models for the drag force and Brownian motion described in Section 3.3. We modeled the ALS using its known geometry and experimentally recorded pressures at fixed points in the system. Details about the setup and these measurements were provided elsewhere [21]. We then solved for a laminar flow through this geometry using a finite-element solver [58] and exported a regular grid of the quantities flow velocity $\vec{v}$ and gas pressure p throughout the ALS. We defined one `FluidFlowDevice` and nine `SimpleZDetectors` at the distances from the ALS exit where the experimental measurements were made. Then we let *CMInject* read in the flow field and run a simulation for 10^5 gold spheres with $d = 27$ nm. We chose a macro-timestep of 10 µs. The code to reproduce these results is provided in the supplementary materials.

To get a comparable measure for the quality of the particle beam's focus that does not depend on fitting any particular beam shape, we calculated the distance from the origin in X to which 70 % of the particles were detected, both for the simulated and the measured data. The results are shown in Figure 3. One can see that there is good agreement regarding the minimum focus size, ∼35 µm at $z = 3$ mm and the defocusing behavior after $z = 3$ mm. The focusing at $z < 3$ mm is in fair agreement, but deviations are clearly visible and tentatively ascribed to the neglect of gas-particle interactions in the initial space outside the ALS. Nevertheless, the position and size of the minimum focus are the most important results for an injection pipeline used for single-particle X-ray imaging, which our simulations model very well.

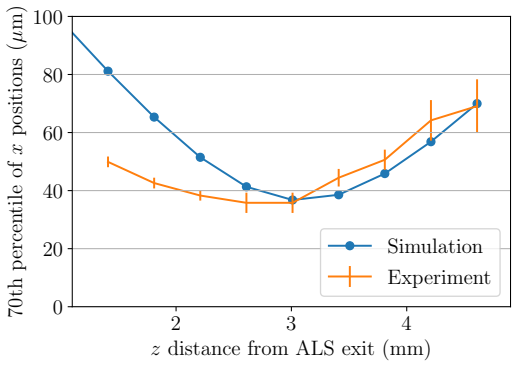

FIG. 3. Focus curves determined by experiment (orange) and simulation (blue) for 27nm gold particles, moving through an ALS [21]. We measure the x positions of all particles relative to the origin, and take the 70% quantile of these positions as our measure of focus size. The results agree well on the minimum focus size and position, i. e., a 38 µm focus at $z = 3$ mm after the ALS exit, and also agree on the defocusing behavior after this minimum.

To better understand the focusing and defocusing behavior, we visually examined the results. For instance, Figure 4 shows 2D histograms of useful quantity pairs at different z positions in the experiment described above. This allows for a visual, somewhat intuitive, disentangling of the evolving ensemble of particles. Such plots can be generated with the provided `cminject_visualize` tool using the `-H` option. Alternatively, a qualitative visual analysis can be obtained by plotting and inspecting particle trajectories as lines, using the `-T` option, as shown for this and other experiments in Figure 1. Less congested visualizations are obtained by animated trajectory evolutions, using the `-M` option, where time-dependent snapshots of the trajectories provide a visualization of the particles positions and velocities. Examples are provided as video files in the supplementary materials.

5. PROGRAM PERFORMANCE

The achievable simulation performance was benchmarked on modern multicore computers, specifically nodes of the Maxwell compute cluster at DESY. The nodes we used are equipped with "Intel(R) Xeon(R) CPU E5-2698

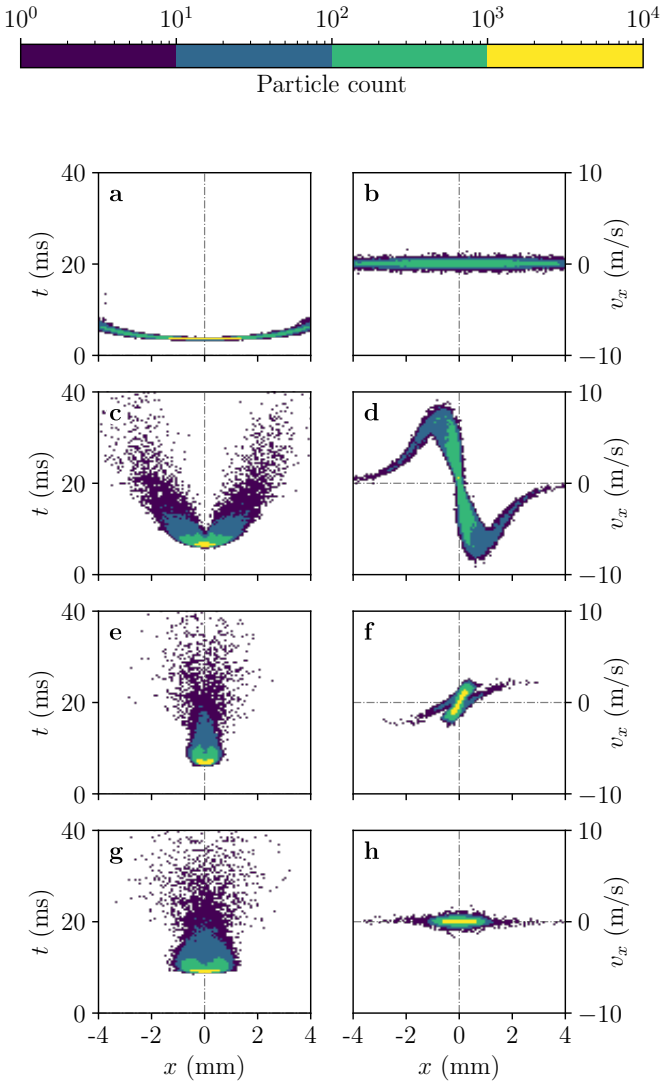

FIG. 4. Phase-space histograms of 10^5 simulated particles at four detectors in an ALS. Two detectors are positioned in the first chamber at the beginning and just before the first aerodynamic lens (**a**–**d**). Two further detectors are positioned after the last lens (**e**–**h**). The detectors' z positions increase downward. The left column shows the evolving t/x distribution and the right one the v_x/x distribution. From the t/x distributions one sees that particles with a larger initial x deviation take longer to arrive at the lens, with slowest particles traveling more than 30 ms longer than the fastest ones (**c**). The v_x/x distribution is initially Gaussian with a large deviation in x (**b**). One can see strong focusing just before the first lens (**d**) and slight defocusing just after the last lens (**f**). The distribution finally turns into a more focused, collimated particle beam (**h**).

v3" or "Intel(R) Xeon(R) E5-2640" CPUs, offering 32/64 and 16/32 cores/hyperthreads, respectively.

We note that performance may improve or degrade substantially compared to what is shown here when different force models, experiment sizes, or time steps are used. Here, we benchmarked the fluid dynamics simulation described in Section 4, involving only Stokes' drag force and Brownian motion at a macro-timestep of 10 µs and an experiment length of ∼13 cm.

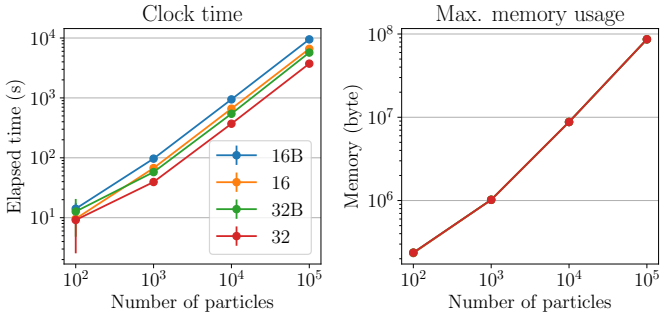

FIG. 5. Scaling behavior of clock time and memory usage for the simulation described in Section 5. "32" and "16" refer to a Intel Xeon E5-2698 (32 cores) and a Intel Xeon E5-2640 CPU (16 cores), respectively. "B" indicates that Brownian motion was enabled, whereas it wasn't otherwise. Note that the variance in memory usage is very low for a fixed number of particles and all curves look like one. Besides initial setup overhead, linear scaling of both clock time and memory is clearly visible.

Figure 5 shows runtime and memory requirements for this simulation when varying the number of particles, and demonstrates that both scale linearly with the number of particles — as expected from a Monte Carlo simulation with no particle-particle interactions, which is trivially parallel.

In Figure 6, we analyze multiple performance metrics as functions of the number of parallel computation processes. The optimal runtime is reached when we use exactly as many processes as there are physical CPU cores. When we use more processes, runtime performance degrades significantly, together with several other performance metrics. This is observed even though the CPU

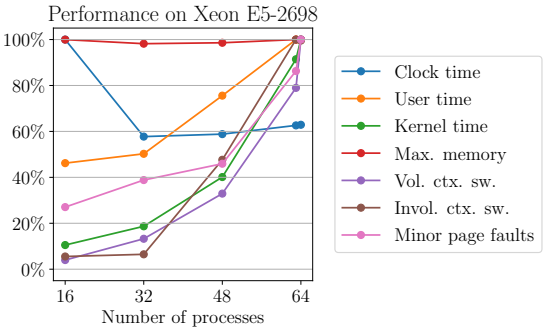

FIG. 6. Performance measurements made on an Intel Xeon E5-2698 CPU with 32 physical cores for the same simulation with different numbers of threads. The maximum value for each measurement is set to be 100 %, and the other values displayed in relation to it. "Vol."/"Invol." are shorthand for "voluntary"/"involuntary", and "ctx. sw." is shorthand for "context switches".

offers up to 64 available hyperthreads, which points to our current implementation not being well-suited to gain performance from hyperthreading. In line with previous literature on this topic, we assume the reason to be that hyperthreading increases competition for resources in the memory hierarchy [59]. If this is indeed the reason, it could mean that our current implementation performs a significant number of memory accesses that under-utilize caches.

Graphics-processing units (GPUs) are particularly well-suited to trivially parallelizable calculations that largely consist of repeated, similar floating-point operations. They offer much higher internal bandwidths in their memory hierarchy than the bandwidths between CPU and main memory [60], and as such should have less trouble maintaining reasonable performance even when faced with many cache misses. Therefore, they should exhibit significantly better runtime-performance scaling at a much larger number of parallel threads. We had also discussed other reasons why GPUs could offer significant speedups for our calculations [40, ch. 7]. With recent developments in the automatic optimization library *Numba* [45] making GPU calculations in Python more accessible, GPUs could be effectively utilized in future versions of our object-oriented framework.

6. SUMMARY AND OUTLOOK

We introduced, described, and benchmarked a new Python framework for the simulation of nanoparticle-injection pipelines. We hope that it will not only improve the sample delivery in single-particle x-ray imaging [11], but also other isolated-nanoparticle experiments [61, 62].

The force models already implemented in *CMInject* enable simulation-based development and exchange of improved and novel injector designs, help to understand the effects of Brownian motion and how to control it better, and facilitate scientific development for injecting single, noncrystalline proteins, e. g., for single particle/molecule imaging experiments. Improvements directly relevant to scientific applications could be made through the systematic derivation and implementation as well as experimental comparison of models for novel techniques, e. g., acoustic [63] or photophoretic [27, 57] focusing. This would open up possibilities to explore these new and exciting pathways toward higher-quality particle beams with *CMInject*, pushing the limits of the imaging of chemical and biochemical processes with atomic resolution.

From a software perspective, development effort should be well-invested to make MPI bindings and GPUs available for users of *CMInject*, e. g., by use of the mpi4py library [64] or the CUDA bindings in the automatic optimization library *Numba* [45], which should significantly improve simulation runtimes [40] and is foreseen for future versions of *CMInject*.

Besides such efforts, a direct integration with computational-fluid-dynamics software packages such as COMSOL [58] or OpenFOAM [65] would allow to run *CMInject* simulations without the need to manually calculate the flow fields beforehand. Users could then provide the description of an experimental device simply as a set of numerical parameters. This could greatly speed up iterations of geometric optimization in the simulations and offer options for automatic optimization of experimental parameters, e. g., using learning-loop approaches.

To facilitate fast availability of improvements as well as community contributions to the development, the framework has been published at `https://github.com/cfel-cmi/cminject` under a modified GPLv3 license requiring attribution, e. g., through referencing of this publication. Up to date documentation is available at `https://cminject.readthedocs.org`. Additional forces and experiments will be modeled and open problems that were discussed here and elsewhere [40] will be resolved, in close exchange with the user community.

DECLARATION OF INTERESTS

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

CREDIT AUTHORSHIP CONTRIBUTION STATEMENT

Simon Welker: Methodology, Software, Validation, Formal analysis, Investigation, Visualization, Writing - Original draft, Writing - Review & Editing; **Muhamed Amin:** Conceptualization, Writing - Review & Editing, Supervision; **Jochen Küpper:** Conceptualization, Resources, Writing - Review & Editing, Supervision, Project administration

ACKNOWLEDGMENTS

We gratefully acknowledge useful discussions with Andrei Rode on photophoretic forces. We thank Lena Worbs for many useful comments and the CMI COMOTION team for tests and feedback of the software package.

We acknowledge support by Deutsches Elektronen-Synchrotron DESY, a member of the Helmholtz Association (HGF), and the use of the Maxwell computational resources operated at Deutsches Elektronen-Synchrotron DESY. This work has been supported by the European Research Council under the European Union's Seventh Framework Programme (FP7/2007-2013) through the Consolidator Grant COMOTION (614507) and by the Deutsche Forschungsgemeinschaft through the Cluster of Excellence "Advanced Imaging of Matter" (AIM, EXC 2056, ID 390715994).

[1] R. Neutze, R. Wouts, D. van der Spoel, E. Weckert, and J. Hajdu, Potential for biomolecular imaging with femtosecond x-ray pulses, Nature **406**, 752 (2000).

[2] J. C. H. Spence and H. N. Chapman, The birth of a new field, Phil. Trans. R. Soc. B **369**, 20130309 (2014).

[3] P. Emma, R. Akre, J. Arthur, R. Bionta, C. Bostedt, J. Bozek, A. Brachmann, P. Bucksbaum, R. Coffee, F. J. Decker, Y. Ding, D. Dowell, S. Edstrom, A. Fisher, J. Frisch, S. Gilevich, J. Hastings, G. Hays, P. Hering, Z. Huang, R. Iverson, H. Loos, M. Messerschmidt, A. Miahnahri, S. Moeller, H. D. Nuhn, G. Pile, D. Ratner, J. Rzepiela, D. Schultz, T. Smith, P. Stefan, H. Tompkins, J. Turner, J. Welch, W. White, J. Wu, G. Yocky, and J. Galayda, First lasing and operation of an angstrom-wavelength free-electron laser, Nat. Photon. **4**, 641 (2010).

[4] W. Decking, S. Abeghyan, P. Abramian, A. Abramsky, A. Aguirre, C. Albrecht, P. Alou, M. Altarelli, P. Altmann, K. Amyan, V. Anashin, E. Apostolov, K. Appel, D. Auguste, V. Ayvazyan, S. Baark, F. Babies, N. Baboi, P. Bak, V. Balandin, R. Baldinger, B. Baranasic, S. Barbanotti, O. Belikov, V. Belokurov, L. Belova, V. Belyakov, S. Berry, M. Bertucci, B. Beutner, A. Block, M. Blöcher, T. Böckmann, C. Bohm, M. Böhnert, V. Bondar, E. Bondarchuk, M. Bonezzi, P. Borowiec, C. Bösch, U. Bösenberg, A. Bosotti, R. Böspflug, M. Bousonville, E. Boyd, Y. Bozhko, A. Brand, J. Branlard, S. Briechle, F. Brinker, S. Brinker, R. Brinkmann, S. Brockhauser, O. Brovko, H. Brück, A. Brüdgam, L. Butkowski, T. Büttner, J. Calero, E. Castro-Carballo, G. Cattalanotto, J. Charrier, J. Chen, A. Cherepenko, V. Cheskidov, M. Chiodini, A. Chong, S. Choroba, M. Chorowski, D. Churanov, W. Cichalewski, M. Clausen, W. Clement, C. Cloué, J. A. Cobos, N. Coppola, S. Cunis, K. Czuba, M. Czwalinna, B. D'Almagne, J. Dammann, H. Danared, A. de Zubiaurre Wagner, A. Delfs, T. Delfs, F. Dietrich, T. Dietrich, M. Dohlus, M. Dommach, A. Donat, X. Dong, N. Doynikov, M. Dressel, M. Duda, P. Duda, H. Eckoldt, W. Ehsan, J. Eidam, F. Eints, C. Engling, U. Englisch, A. Ermakov, K. Escherich, J. Eschke, E. Saldin, M. Faesing, A. Fallou, M. Felber, M. Fenner, B. Fernandes, J. M. Fernández, S. Feuker, K. Filippakopoulos, K. Floettmann, V. Fogel, M. Fontaine, A. Francés, I. F. Martin, W. Freund, T. Freyermuth, M. Friedland, L. Fröhlich, M. Fusetti, J. Fydrych, A. Gallas, O. García, L. Garcia-Tabares, G. Geloni, N. Gerasimova, C. Gerth, P. Geßler, V. Gharibyan, M. Gloor, J. Głowinkowski, A. Goessel, Z. Gołębiewski, N. Golubeva, W. Grabowski, W. Graeff, A. Grebentsov, M. Grecki, T. Grevsmuehl, M. Gross, U. Grosse-Wortmann, J. Grünert, S. Grunewald, P. Grzegory, G. Feng, H. Guler, G. Gusev, J. L. Gutierrez, L. Hagge, M. Hamberg, R. Hanneken, E. Harms, I. Hartl, A. Hauberg, S. Hauf, J. Hauschildt, J. Hauser, J. Havlicek, A. Hedqvist, N. Heidbrook, F. Hellberg, D. Henning, O. Hensler, T. Hermann, A. Hidvégi, M. Hierholzer, H. Hintz, F. Hoffmann, M. Hoffmann, M. Hoffmann, Y. Holler, M. Hüning, A. Ignatenko, M. Ilchen, A. Iluk, J. Iversen, M. Izquierdo, L. Jachmann, N. Jardon, U. Jastrow, K. Jensch, J. Jensen, M. Jeżabek, M. Jidda, H. Jin, N. Johansson, R. Jonas, W. Kaabi, D. Kaefer, R. Kammering, H. Kapitza, S. Karabekyan, S. Karstensen, K. Kasprzak, V. Katalev, D. Keese, B. Keil, M. Kholopov, M. Killenberger, B. Kitaev, Y. Klimchenko, R. Klos, L. Knebel, A. Koch, M. Koepke, S. Köhler, W. Köhler, N. Kohlstrunk, Z. Konopkova, A. Konstantinov, W. Kook, W. Koprek, M. Körfer, O. Korth, A. Kosarev, K. Kosiński, D. Kostin, Y. Kot, A. Kotarba, T. Kozak, V. Kozak, R. Kramert, M. Krasilnikov, A. Krasnov, B. Krause, L. Kravchuk, O. Krebs, R. Kretschmer, J. Kreutzkamp, O. Kröplin, K. Krzysik, G. Kube, H. Kuehn, N. Kujala, V. Kulikov, V. Kuzminych, D. La Civita, M. Lacroix, T. Lamb, A. Lancetov, M. Larsson, D. Le Pinvidic, S. Lederer, T. Lensch, D. Lenz, A. Leuschner, F. Levenhagen, Y. Li, J. Liebing, L. Lilje, T. Limberg, D. Lipka, B. List, J. Liu, S. Liu, B. Lorbeer, J. Lorkiewicz, H. H. Lu, F. Ludwig, K. Machau, W. Maciocha, C. Madec, C. Magueur, C. Maiano, I. Maksimova, K. Malcher, T. Maltezopoulos, E. Mamoshkina, B. Manschwetus, F. Marcellini, G. Marinkovic, T. Martinez, H. Martirosyan, W. Maschmann, M. Maslov, A. Matheisen, U. Mavric, J. Meißner, K. Meissner, M. Messerschmidt, N. Meyners, G. Michalski, P. Michelato, N. Mildner, M. Moe, F. Moglia, C. Mohr, S. Mohr, W. Möller, M. Mommerz, L. Monaco, C. Montiel, M. Moretti, I. Morozov, P. Morozov, and D. Mross, A MHz-repetition-rate hard x-ray free-electron laser driven by a superconducting linear accelerator, Nat. Photon. **14**, 391 (2020).

[5] U. Lorenz, N. M. Kabachnik, E. Weckert, and I. A. Vartanyants, Impact of ultrafast electronic damage in single-particle x-ray imaging experiments, Phys. Rev. E **86**, 051911 (2012), arXiv:1206.6960 [physics].

[6] A. Barty, J. Küpper, and H. N. Chapman, Molecular imaging using x-ray free-electron lasers, Annu. Rev. Phys. Chem. **64**, 415 (2013).

[7] K. Pande, C. D. M. Hutchison, G. Groenhof, A. Aquila, J. S. Robinson, J. Tenboer, S. Basu, S. Boutet, D. P. DePonte, M. Liang, T. A. White, N. A. Zatsepin, O. Yefanov, D. Morozov, D. Oberthuer, C. Gati, G. Subramanian, D. James, Y. Zhao, J. Koralek, J. Brayshaw, C. Kupitz, C. Conrad, S. Roy-Chowdhury, J. D. Coe, M. Metz, P. L. Xavier, T. D. Grant, J. E. Koglin, G. Ketawala, R. Fromme, V. Šrajer, R. Henning, J. C. H. Spence, A. Ourmazd, P. Schwander, U. Weierstall, M. Frank, P. Fromme, A. Barty, H. N. Chapman, K. Moffat, J. J. van Thor, and M. Schmidt, Femtosecond structural dynamics drives the trans/cis isomerization in photoactive yellow protein, Science **352**, 725 (2016).

[8] A. Ourmazd, Cryo-EM, XFELs and the structure conundrum in structural biology, Nat. Meth. **16**, 941 (2019).

[9] K. Ayyer, A. J. Morgan, A. Aquila, H. DeMirci, B. G. Hogue, R. A. Kirian, P. L. Xavier, C. H. Yoon, H. N. Chapman, and A. Barty, Low-signal limit of x-ray single particle diffractive imaging, Opt. Exp. **27**, 37816 (2019).

[10] M. F. Hantke, J. Bielecki, O. Kulyk, D. Westphal, D. S. D. Larsson, M. Svenda, H. K. N. Reddy, R. A. Kirian, J. Andreasson, J. Hajdu, and F. R. N. C. Maia, Rayleigh-scattering microscopy for tracking and sizing nanoparticles in focused aerosol beams, IUCrJ **5**, 673 (2018).

[11] K. Ayyer, P. L. Xavier, J. Bielecki, Z. Shen, B. J. Daurer, A. K. Samanta, S. Awel, R. Bean, A. Barty, M. Bergemann, T. Ekeberg, A. D. Estillore, H. Fangohr, K. Giewekemeyer, M. S. Hunter, M. Karnevskiy, R. A. Kirian, H. Kirkwood, Y. Kim, J. Koliyadu,

H. Lange, R. Letrun, J. Lübke, T. Michelat, A. J. Morgan, N. Roth, T. Sato, M. Sikorski, F. Schulz, J. C. H. Spence, P. Vagovic, T. Wollweber, L. Worbs, O. Yefanov, Y. Zhuang, F. R. N. C. Maia, D. A. Horke, J. Küpper, N. D. Loh, A. P. Mancuso, and H. N. Chapman, 3D diffractive imaging of nanoparticle ensembles using an x-ray laser, Optica **8**, 15 (2021), arXiv:2007.13597 [physics].

[12] R. Fung, V. Shneerson, D. Saldin, and A. Ourmazd, Structure from fleeting illumination of faint spinning objects in flight, Nat. Phys. **5**, 64 (2009).

[13] G. Bortel and G. Faigel, Classification of continuous diffraction patterns: a numerical study, J. Struct. Biol. **158**, 10 (2007).

[14] A. K. Samanta, M. Amin, A. D. Estillore, N. Roth, L. Worbs, D. A. Horke, and J. Küpper, Controlled beams of shockfrozen, isolated, biological and artificial nanoparticles, Struct. Dyn. **7**, 024304 (2020), arXiv:1910.12606 [physics].

[15] N. Roth, S. Awel, D. A. Horke, and J. Küpper, Optimizing aerodynamic lenses for single-particle imaging, J. Aerosol. Sci. **124**, 17 (2018), arXiv:1712.01795 [physics].

[16] Y.-P. Chang, D. A. Horke, S. Trippel, and J. Küpper, Spatially-controlled complex molecules and their applications, Int. Rev. Phys. Chem. **34**, 557 (2015), arXiv:1505.05632 [physics].

[17] P. Liu, P. J. Ziemann, D. B. Kittelson, and P. H. McMurry, Generating particle beams of controlled dimensions and divergence: I. theory of particle motion in aerodynamic lenses and nozzle expansions, Aerosol Sci. Techn. **22**, 293 (1995).

[18] X. Wang and P. H. McMurry, An experimental study of nanoparticle focusing with aerodynamic lenses, Int. J. Mass Spectrom. **258**, 30 (2006).

[19] X. Wang, A. Gidwani, S. L. Girshick, and P. H. McMurry, Aerodynamic focusing of nanoparticles: II. numerical simulation of particle motion through aerodynamic lenses, Aerosol Sci. Techn. **39**, 624 (2005).

[20] X. Wang and P. H. McMurry, A design tool for aerodynamic lens systems, Aerosol Sci. Technol. **40**, 320 (2006).

[21] L. Worbs, N. Roth, J. Lübke, A. Estillore, P. L. Xavier, A. K. Samanta, and J. Küpper, Optimizing the geometry of aerodynamic lens injectors for single-particle coherent diffractive imaging of gold nanoparticles, arXiv:2105.15084v1 [physics] (2021), submitted.

[22] E. Sobolev, S. Zolotarev, K. Giewekemeyer, J. Bielecki, K. Okamoto, H. K. N. Reddy, J. Andreasson, K. Ayyer, I. Barak, S. Bari, A. Barty, R. Bean, S. Bobkov, H. N. Chapman, G. Chojnowski, B. J. Daurer, K. Dörner, T. Ekeberg, L. Flückiger, O. Galzitskaya, L. Gelisio, S. Hauf, B. G. Hogue, D. A. Horke, A. Hosseinizadeh, V. Ilyin, C. Jung, C. Kim, Y. Kim, R. A. Kirian, H. Kirkwood, O. Kulyk, R. Letrun, D. Loh, M. Messerschmidt, K. Mühlig, A. Ourmazd, N. Raab, A. V. Rode, M. Rose, A. Round, T. Sato, R. Schubert, P. Schwander, J. A. Sellberg, M. Sikorski, A. Silenzi, C. Song, J. C. H. Spence, S. Stern, J. Sztuk-Dambietz, A. Teslyuk, N. Timneanu, M. Trebbin, C. Uetrecht, B. Weinhausen, G. J. Williams, P. L. Xavier, C. Xu, I. Vartanyants, V. Lamzin, A. Mancuso, and F. R. N. C. Maia, Megahertz single-particle imaging at the European XFEL, Comm. Phys **3**, 97 (2020), arXiv:1912.10796 [physics].

[23] J. Bielecki, M. F. Hantke, B. J. Daurer, H. K. N. Reddy, D. Hasse, D. S. D. Larsson, L. H. Gunn, M. Svenda, A. Munke, J. A. Sellberg, L. Flueckiger, A. Pietrini, C. Nettelblad, I. Lundholm, G. Carlsson, K. Okamoto, N. Timneanu, D. Westphal, O. Kulyk, A. Higashiura, G. van der Schot, N.-T. D. Loh, T. E. Wysong, C. Bostedt, T. Gorkhover, B. Iwan, M. M. Seibert, T. Osipov, P. Walter, P. Hart, M. Bucher, A. Ulmer, D. Ray, G. Carini, K. R. Ferguson, I. Andersson, J. Andreasson, J. Hajdu, and F. R. N. C. Maia, Electrospray sample injection for single-particle imaging with x-ray lasers, Science Advances **5**, eaav8801 (2019).

[24] Y.-P. Chang, D. Horke, S. Trippel, and J. Küpper, CMIfly, `https://github.com/CFEL-CMI/cmifly` (2020), originally published in [16].

[25] F. Filsinger, J. Küpper, G. Meijer, L. Holmegaard, J. H. Nielsen, I. Nevo, J. L. Hansen, and H. Stapelfeldt, Quantum-state selection, alignment, and orientation of large molecules using static electric and laser fields, J. Chem. Phys. **131**, 064309 (2009), arXiv:0903.5413 [physics].

[26] N. Roth, M. Amin, A. K. Samanta, and J. Küpper, Microscopic force for aerosol transport, submitted (2020), arXiv:2006.10652 [physics].

[27] N. Eckerskorn, R. Bowman, R. A. Kirian, S. Awel, M. Wiedorn, J. Küpper, M. J. Padgett, H. N. Chapman, and A. V. Rode, Optically induced forces imposed in an optical funnel on a stream of particles in air or vacuum, Phys. Rev. Appl. **4**, 064001 (2015).

[28] A. S. Desyatnikov, V. G. Shvedov, A. V. Rode, W. Krolikowski, and Y. S. Kivshar, Photophoretic manipulation of absorbing aerosol particles with vortex beams: theory versus experiment, Opt. Exp. **17**, 8201 (2009).

[29] B. J. Daurer, K. Okamoto, J. Bielecki, F. R. N. C. Maia, K. Mühlig, M. M. Seibert, M. F. Hantke, C. Nettelblad, W. H. Benner, M. Svenda, N. Timneanu, T. Ekeberg, N. D. Loh, A. Pietrini, A. Zani, A. D. Rath, D. Westphal, R. A. Kirian, S. Awel, M. O. Wiedorn, G. van der Schot, G. H. Carlsson, D. Hasse, J. A. Sellberg, A. Barty, J. Andreasson, S. Boutet, G. Williams, J. Koglin, I. Andersson, J. Hajdu, and D. S. D. Larsson, Experimental strategies for imaging bioparticles with femtosecond hard x-ray pulses, IUCrJ **4**, 251 (2017).

[30] K. B. Tarmyshov and F. Müller-Plathe, Parallelizing a molecular dynamics algorithm on a multiprocessor workstation using openmp, Journal of Chemical Information and Modeling **45**, 1943 (2005).

[31] F. Filsinger, U. Erlekam, G. von Helden, J. Küpper, and G. Meijer, Selector for structural isomers of neutral molecules, Phys. Rev. Lett. **100**, 133003 (2008), arXiv:0802.2795 [physics].

[32] A. Yachmenev, J. Onvlee, E. Zak, A. Owens, and J. Küpper, Field-induced diastereomers for chiral separation, Phys. Rev. Lett. **123**, 243202 (2019), arXiv:1905.07166 [physics].

[33] H. Stapelfeldt and T. Seideman, Colloquium: Aligning molecules with strong laser pulses, Rev. Mod. Phys. **75**, 543 (2003).

[34] J. C. H. Spence and R. B. Doak, Single molecule diffraction, Phys. Rev. Lett. **92**, 198102 (2004).

[35] L. Holmegaard, J. H. Nielsen, I. Nevo, H. Stapelfeldt, F. Filsinger, J. Küpper, and G. Meijer, Laser-induced alignment and orientation of quantum-state-selected large molecules, Phys. Rev. Lett. **102**, 023001 (2009), arXiv:0810.2307 [physics].

[36] J. Küpper, S. Stern, L. Holmegaard, F. Filsinger, A. Rouzée, A. Rudenko, P. Johnsson, A. V. Martin,

M. Adolph, A. Aquila, S. Bajt, A. Barty, C. Bostedt, J. Bozek, C. Caleman, R. Coffee, N. Coppola, T. Delmas, S. Epp, B. Erk, L. Foucar, T. Gorkhover, L. Gumprecht, A. Hartmann, R. Hartmann, G. Hauser, P. Holl, A. Hömke, N. Kimmel, F. Krasniqi, K.-U. Kühnel, J. Maurer, M. Messerschmidt, R. Moshammer, C. Reich, B. Rudek, R. Santra, I. Schlichting, C. Schmidt, S. Schorb, J. Schulz, H. Soltau, J. C. H. Spence, D. Starodub, L. Strüder, J. Thøgersen, M. J. J. Vrakking, G. Weidenspointner, T. A. White, C. Wunderer, G. Meijer, J. Ullrich, H. Stapelfeldt, D. Rolles, and H. N. Chapman, X-ray diffraction from isolated and strongly aligned gas-phase molecules with a free-electron laser, Phys. Rev. Lett. **112**, 083002 (2014), arXiv:1307.4577 [physics].

[37] E. T. Karamatskos, S. Raabe, T. Mullins, A. Trabattoni, P. Stammer, G. Goldsztejn, R. R. Johansen, K. Długołęcki, H. Stapelfeldt, M. J. J. Vrakking, S. Trippel, A. Rouzée, and J. Küpper, Molecular movie of ultrafast coherent rotational dynamics of OCS, Nat. Commun. **10**, 3364 (2019), arXiv:1807.01034 [physics].

[38] J. Miao, T. Ishikawa, Q. Shen, and T. Earnest, Extending x-ray crystallography to allow the imaging of noncrystalline materials, cells, and single protein complexes, Annu. Rev. Phys. Chem. **59**, 387 (2007).

[39] J. Miao, P. Charalambous, J. Kirz, and D. Sayre, Extending the methodology of x-ray crystallography to allow imaging of micrometre-sized non-crystalline specimens, Nature **400**, 342 (1999).

[40] S. Welker, *Simulation of artificial and biological nanoparticles' trajectories in hybrid force-fields*, Bachelor thesis, Universität Hamburg (2019).

[41] G. van Rossum and Talin, PEP 3119: Introducing abstract base classes, Website, URL: `https://www.python.org/dev/peps/pep-3119` (2007), accessed on 2019-10-04.

[42] L. Petzold, Automatic selection of methods for solving stiff and nonstiff systems of ordinary differential equations, SIAM J. Sci. & Stat. Comp. **4**, 136 (1983).

[43] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. Jarrod Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. Carey, İ. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and SciPy 1.0 Contributors, SciPy 1.0: fundamental algorithms for scientific computing in Python, Nat. Meth. **17**, 261 (2020).

[44] S. van der Walt, S. C. Colbert, and G. Varoquaux, The NumPy array: A structure for efficient numerical computation, Comp. in Sci. & Eng. **13**, 22 (2011).

[45] S. K. Lam, A. Pitrou, and S. Seibert, Numba: A LLVM-based Python JIT compiler, in *Proceedings of the Second Workshop on the LLVM Compiler Infrastructure in HPC*, LLVM '15 (Association for Computing Machinery, New York, NY, USA, 2015).

[46] S. Behnel, R. Bradshaw, C. Citro, L. Dalcin, D. S. Seljebotn, and K. Smith, Cython: The best of both worlds, Comp. Sci. & Eng. **13**, 31 (2011).

[47] G. G. Stokes, On the effect of the internal friction of fluids on the motion of pendulums, Trans. Cambridge Phil. Soc. **9**, 8 (1851).

[48] E. Cunningham and J. Larmor, On the velocity of steady fall of spherical particles through fluid medium, Proc. Royal Soc. London A **83**, 357 (1910).

[49] D. K. Hutchins, M. H. Harper, and R. L. Felder, Slip correction measurements for solid spherical particles by modulated dynamic light scattering, Aerosol Sci. Techn. **22**, 202 (1995).

[50] K. Willeke, Temperature dependence of particle slip in a gaseous medium, J. Aerosol. Sci. **7**, 381 (1976).

[51] A. Li and G. Ahmadi, Dispersion and deposition of spherical particles from point sources in a turbulent channel flow, Aerosol Sci. Techn. **16**, 209 (1992).

[52] P. S. Epstein, On the resistance experienced by spheres in their motion through gases, Phys. Rev. **23**, 710 (1924).

[53] R. W. Bowman and M. J. Padgett, Optical trapping and binding, Rep. Prog. Phys. **76**, 026401 (2013).

[54] N. O. Eckerskorn, *Trapping and guiding microscopic particles with light-induced forces*, Dissertation, College of Physical & Mathematical Sciences, Research School of Physics and Engineering, Laser Physics Centre, Australia (2016).

[55] V. G. Shvedov, A. S. Desyatnikov, A. V. Rode, W. Krolikowski, and Y. S. Kivshar, Optical guiding of absorbing nanoclusters in air, Opt. Exp. **17**, 5743 (2009).

[56] V. G. Shvedov, C. Hnatovsky, A. V. Rode, and W. Krolikowski, Robust trapping and manipulation of airborne particles with a bottle beam, Opt. Exp. **19**, 17350 (2011).

[57] S. Awel, S. Lavin-Varela, N. Roth, D. A. Horke, A. V. Rode, R. A. Kirian, J. Küpper, and H. N. Chapman, Optical funnel to guide and focus virus particles for x-ray laser imaging (2020), submitted.

[58] Comsol, Multiphysics 5.5 (2019), cOMSOL Multiphysics v. 5.5. `http://www.comsol.com`. COMSOL AB, Stockholm, Sweden.

[59] S. Saini, H. Jin, R. Hood, D. Barker, P. Mehrotra, and R. Biswas, The impact of hyper-threading on processor resource utilization in production applications, in *2011 18th International Conference on High Performance Computing* (2011) pp. 1–10.

[60] NVIDIA Corporation, CUDA C++ best practices guide (2020), accessed on 2020-12-09.

[61] L. Seiffert, Q. Liu, S. Zherebtsov, A. Trabattoni, P. Rupp, M. C. Castrovilli, M. Galli, F. Submann, K. Wintersperger, J. Stierle, G. Sansone, L. Poletto, F. Frassetto, I. Halfpap, V. Mondes, C. Graf, E. Rühl, F. Krausz, M. Nisoli, T. Fennel, F. Calegari, and M. F. Kling, Attosecond chronoscopy of electron scattering in dielectric nanoparticles, Nat. Phys. **13**, 766 (2017).

[62] A. Aquila, A. Barty, C. Bostedt, S. Boutet, G. Carini, D. DePonte, P. Drell, S. Doniach, K. H. Downing, T. Earnest, H. Elmlund, V. Elser, M. Gühr, J. Hajdu, J. Hastings, S. P. Hau-Riege, Z. Huang, E. E. Lattman, F. R. N. C. Maia, S. Marchesini, A. Ourmazd, C. Pellegrini, R. Santra, I. Schlichting, C. Schroer, J. C. H. Spence, I. A. Vartanyants, S. Wakatsuki, W. I. Weis, and G. J. Williams, The linac coherent light source single particle imaging road map, Struct. Dyn. **2**, 041701 (2015).

[63] Z. Li, L. Shi, L. Cao, Z. Liu, and J. Küpper, Acoustic funnel and buncher for nanoparticle injection, Phys. Rev. Appl. **11**, 064036 (2019), arXiv:1803.07472 [physics].

[64] L. Dalcín, R. Paz, and M. Storti, MPI for Python, J. Parallel Distr. Comp. **65**, 1108 (2005).

[65] The OpenFOAM Foundation, OpenFOAM | free CFD software, Website, URL: `https://openfoam.org/` (2020), accessed on 2020-05-28.