Feature Analysis, Tracking, and Data Reduction: An Application to Multiphase Reactor Simulation MFiX-Exa for *In-Situ* Use Case

Ayan Biswas, *Los Alamos National Laboratory, Los Alamos, NM, 87544, USA*
James P. Ahrens, *Los Alamos National Laboratory, Los Alamos, NM, 87544, USA*
Soumya Dutta, *Los Alamos National Laboratory, Los Alamos, NM, 87544, USA*
Jordan M. Musser, *National Energy Technology Laboratory, Morgantown, WV, 26507, USA*
Ann S. Almgren, *Lawrence Berkeley National Laboratory, Berkeley, CA, 94720, USA*
Terece L. Turton, *Los Alamos National Laboratory, Los Alamos, NM, 87544, USA*

As we enter the exascale computing regime, powerful supercomputers continue to produce much higher amounts of data than what can be stored for offline data processing. To utilize such high compute capabilities on these machines, much of the data processing needs to happen in situ, ***when the full high-resolution data is available at the supercomputer memory. In this article, we discuss our MFiX-Exa simulation, which models multiphase flow by tracking a very large number of particles through the simulation domain. In one of the use cases, the carbon particles interact with air to produce carbon dioxide bubbles from the reactor. These bubbles are of primary interest to the domain experts for these simulations. For this particle-based simulation, we propose a streaming technique that can be deployed*** in situ ***to efficiently identify the bubbles, track them over time, and use them to down-sample the data with minimal loss in these features.***

Every year, supercomputers continue to become more powerful in terms of their computing capabilities. Compared to such high compute power, the I/O capabilities of the machines have not scaled proportionately. This creates a mismatch between the amount of data that can be computed by these supercomputers and how much data can be moved over to persistent storage for future data analysis and exploration tasks. Therefore, most of the important data analysis tasks for large-scale scientific simulations must occur while the data is still available at the supercomputer memory. Compared to the posthoc analysis pipeline, this new streaming data analysis regime is referred to as *in-situ* or *online* processing.

For scientific simulations, data can be divided primarily into two groups: 1) grid-based and 2) particle-based. For grid-based data, generally, the values of the variables (e.g., pressure, temperature) are defined on-grid locations over a computational domain. For particle-based data, a collection of particles is simulated by the physics codes that move through space and time. Depending on the simulation, individual particles with some local properties (e.g., energy transport via eddies in flow simulations) or a collection of particles with some global properties (e.g., particles forming halos in cosmological simulations) can be of interest to the domain experts.

In this article, we focus on a particle-based simulation named MFiX-Exa. Starting from a set of particles, the domain experts for this simulation want to

1521-9615 © 2020 IEEE
Digital Object Identifier 10.1109/MCSE.2020.3016927
Date of publication 14 August 2020; date of current version 26 February 2021.

formulate and extract bubble statistics in each time step. Given that these bubbles get created, move over time, and get released into the air, domain experts need to have efficient tracking of these features over time. But lots of particles are tracked over these time steps, and experts want to reduce the number of particles without losing the bubbles from the data. Due to the scale of the simulation, all these data analyses and reductions need to happen in batches, as the data becomes available *in-situ*. In this article, we propose a lightweight streaming bubble detection and tracking method and demonstrate a bubble-preserving particle-sampling algorithm. This algorithm is designed to handle the *in-situ* use case where data is distributed across multiple processors and the time steps are generated and analyzed in a streaming setting.

The manuscript is organized as follows: In the next section, we provide a brief overview of the existing works that relate to this article. Next, we discuss the simulation and its data output. Then, we provide the details of our *in-situ* feature (bubbles, in this case) detection method. Next, we discuss our *in-situ* feature tracking algorithm. After that, we provide details on our feature-driven particle down-sampling approach. Finally, we conclude this manuscript with a summary of the proposed *in-situ* approaches.

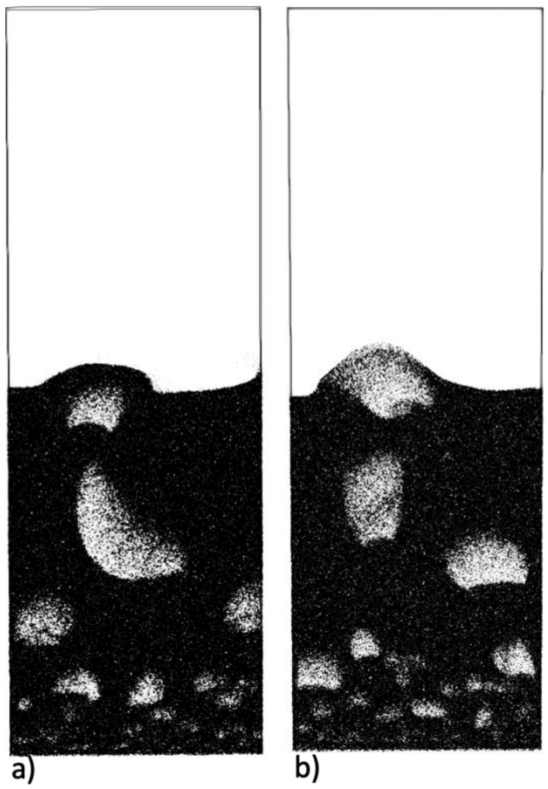

FIGURE 1. Example outputs from MFiX-Exa simulation. All the particles are shown for (a) time step 150 and (b) time step 408.

RELATED WORK

The need for data reduction techniques for an *in-situ* environment has grown significantly in recent years, and various approaches to *in-situ* data reduction techniques have been proposed. While several scientists have explored data subsampling as one of the potential techniques,[7] others have used distribution-based data representations as a suitable approach for *in-situ* data reduction.[10] Also, data reduction techniques such as wavelet-based data modeling techniques and data compression techniques[2,8] have been shown effective. For a comprehensive list of data reduction techniques, please refer to the state-of-the-art report by Li *et al.*[3]

Feature tracking is an important data analysis problem for scientific datasets. One of the earliest works in feature tracking was by Samataney *et al.*[5] where feature correspondence was used to track the features over time. Volume overlapping-based feature tracking was proposed by Silver and Wang.[6] Feature tracking in particle datasets was investigated by Sauer *et al.*, and the tracking was done in joint particle/volume datasets.[9] A predictor–corrector-based feature tracking algorithm was proposed by Muelder *et al.*,[11] and an *in-situ* application of a similar method was demonstrated in the article by Duque *et al.*[4] A broad survey of feature tracking techniques can be found by Post *et al.*[12] Compared to the above-mentioned works, here we focus on feature tracking techniques that can be deployed in an *in-situ* setting.

DESCRIPTION OF THE DATASET

The simulation represents a small-scale, pseudo 2-D ($0.15\,\mathrm{m} \times 0.0032\,\mathrm{m} \times 0.0508\,\mathrm{m}$) system where a constant density ($1.205\,\mathrm{kg/m^3}$), constant viscosity ($1.8 \times 10^{-5}\,\mathrm{Pa{\cdot}s}$) gas is used to fluidize spherical particles of uniform size ($148 \times 10^{-6}\,\mathrm{m}$ diameter) and density ($1300\,\mathrm{kg/m^3}$). A pressure outflow is prescribed at the top of the domain, while a constant velocity gas inlet ($0.0342\,\mathrm{m/s}$) is specified along the bottom. The remaining boundaries are modeled as solid walls. The domain is decomposed into $672 \times 10 \times 228$ computational cells and initialized with solids by randomly distributing approximately 3.6 million particles. As the simulation progresses, particles settle in the direction of gravity, forming a bubbling fluidized bed (see Figure 1).

Time-averaged particle statistics (e.g., averaged particle velocities) and transient field properties (e.g.,

bubble size and velocity) are commonly used to compare experimental data and model results.[1] A similar analysis could be employed in assessing reactor designs whereby bubble (and/or cluster) statistics would provide engineers insight into flow phenomena that adversely impact multiphase reactor performance (e.g., gas by-passing via bubble formation).

BUBBLE DETECTION IN *IN-SITU* SCENARIO

For this dataset, the bubble features are vaguely defined as *low-density regions* or *voids* in the data. To detect such features, we first need to compute the density field from the point dataset. Since the application mandates the algorithms be suitable for an *in-situ* environment where the data is distributed across multiple processors, we propose a histogram-based density estimate.

Histograms are essentially *counts* of values falling into each bin, and these counts are efficiently parallelizable because they can be added up across different processors. In our case, we use a histogram over the spatial domain. Each processor is responsible for the advection of a subset of particles, and it has the knowledge of the complete spatial domain of the simulation. For each processor, we divide the domain into the same 3-D regular grid [e.g., as shown in Figure 2(a)]. Now, each processor can compute a local histogram by computing how many particles are falling into each bin. These local histograms can be summed together to derive the final density histogram. Since the size of the histogram is very small compared to the size of the original data, we now perform all the operations on this histogram for bubble identification and tracking. Further, because each bin is spread across the domain, simply visualizing the density field will give us information regarding where the bubbles are located. We can think of this spatial histogram as a regular discretization of space, and it can be used as a regular grid dataset for analysis and visualization purposes.

For identification of the bubbles, we apply a thresholding (K) on the density field to produce our feature field (F_{feature}) which is essentially a binary version of the density field of the same size:

$$F_{\text{feature}} = \begin{array}{l} 1 = \text{if } F_{\text{density}} \geq K \\ 0 = \text{otherwise} . \end{array} \quad (1)$$

Essentially, for all locations where the density field value $F_{\text{density}} \geq K$, we set the locations to zero. Otherwise, the value is set to 1. This step assigns all the locations that are part of a bubble as 1, signifying that this is the feature of interest.

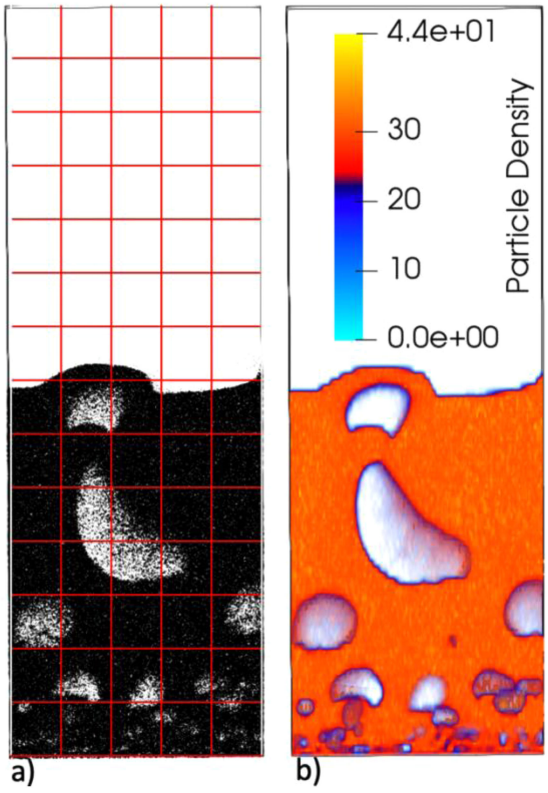

FIGURE 2. Conversion from particle dataset to density field. (a) For timestep 150, an example of spatial histogram creation is shown. (b) Visualization of the same data after conversion to its density field using a higher number of histogram bins. The high-density regions are shown in red.

At this stage, although the bubbles can be visualized [see Figure 2(b)], they cannot be distinguished from each other because all the bubbles are assigned a value of 1. To prepare each bubble for tracking and collecting statistics individually, we mark each bubble region with a unique identifier. To achieve this, we perform a *connected components* algorithm on this binary field [see Figure 3(a)]. Primarily used for images, this algorithm finds the total number of components in the field. We apply a flood-fill-based connected component algorithm for identifying all the bubbles. This method is performed on only one processor because we are not applying it to the particles, rather it is applied to the binary version of the density histogram.

After this stage, we need to make one correction to the detected bubbles. The top layer of air will be detected as one bubble, which in fact, should not be considered a bubble at all. To fix this, we determine

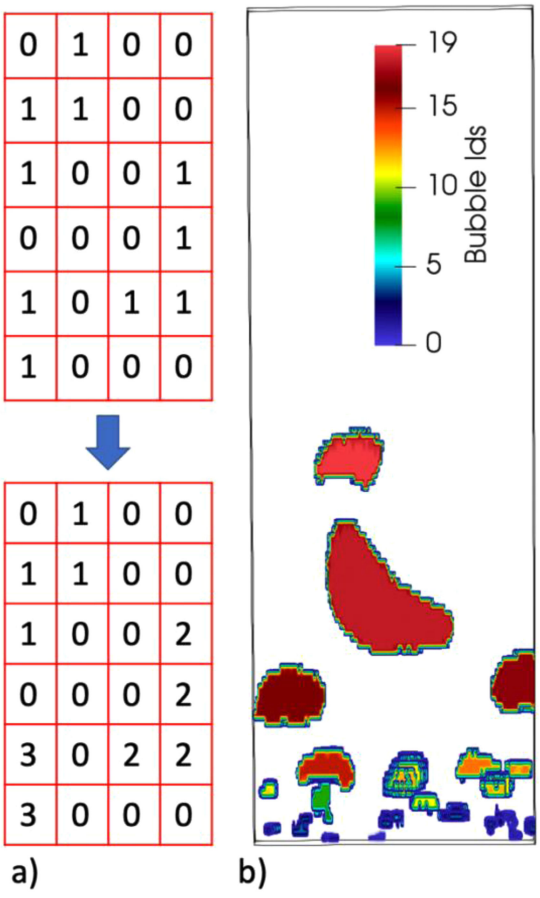

FIGURE 3. Individual bubble identification from the binary labeled field. (a) Schematic example showing how a connected-components algorithm changes the binary field to assign individual bubbles a unique identifier at each time step. (b) Visualization of time step 150 after identifying individual bubbles.

which bubble has a top layer that is the same as the data domain. We remove that bubble from our list of bubbles [see Figure 3(b)] and then move on to track them over time.

BUBBLE TRACKING IN *IN-SITU* SCENARIO

Understanding how bubbles evolve is an important aspect of the exploration process for this dataset. Bubbles get created; move over time; and merge, split, or die. Domain experts want to explore such events associated with the bubbles for the course of the simulation time. In the previous section, we described how the individual bubbles can be assigned a unique identifier. But when such a method is applied independently for each time step, the same bubble might be assigned a different identifier when it moves from time step t to $t+1$. Tracking ensures that the same bubbles always get the same tag or id irrespective of which time step they belong to.

Tracking consists of the following scenarios.

1. Born: A new bubble is born.
2. Move: A bubble has moved to a new location with or without changing shape.
3. Split: An existing bubble has split into two or more smaller bubbles.
4. Merge: Two (or more) smaller bubbles have merged to form a larger bubble.
5. Die: A bubble moves out of the particles and gets mixed into the air.

Identification of these events *in-situ* can be challenging due to the streaming nature of the problem—the time steps come one at a time and as we move to the next time step, the previous time step no longer exists in the memory. To alleviate these constraints, we retain the bubble information from the previous time step to match with the next one. Because the bubble information generated from the spatial histograms is much smaller in size compared to the full-scale particle data, we can carry it from a given time step to the next.

To resolve the tracking scenarios [see Figure 4(a)], we first try to detect the *move* event. For this event, we use the spatial overlap criteria. For a given bubble, we attempt to search for another bubble in the next time step that has maximum overlap. Because all the bubbles in this dataset are primarily moving upwards, we use the predictor–corrector method.[11] We first predict that the bubble will be moving upwards given the average speed of the surrounding particles of that location and then find the bubble with maximum spatial overlap with this predicted bubble location. When a *move* event is found, the bubble id from the previous time step is assigned to the *moved bubble* of the new time step. This way we can be consistent with the bubble ids over time.

Next, we check for split/merge events. For detecting split events, we check the bubble data of the current time step against the previous one. For each bubble of the current time step, we perform a location-based overlap comparison with the bubbles from the previous time step. If we observe there were overlaps with more than one bubble, we conclude a merge event occurred. Merge and split can be thought of as similar events except reversed in time direction; that is, a merge event in the forward comparison of time t with $t+1$ can be thought of as a split event when

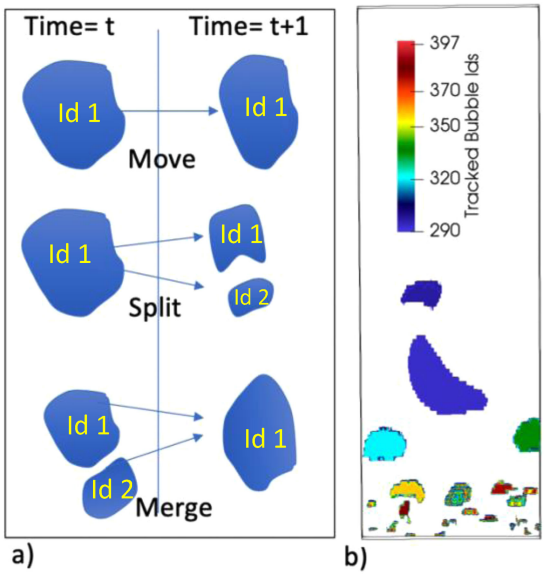

FIGURE 4. Illustration of proposed *in-situ* bubble tracking method. (a) From one time-step to the next, existing bubbles can move (change shape), split, or merge with other bubbles. A new bubble can also be born with a new id. (b) After applying bubble tracking, visualization from time step 150. Compared to Figure 3(b), we see the bubble ids are now more numerous in Figure 4(b) because they are uniquely assigned to each bubble moving over time.

comparing time $t+1$ with time t. We can use this idea to detect merge events similar to detecting split events *in situ*, except we compare time t with time $t+1$ for bubble overlaps, as mentioned in the split detection earlier.

After the application of this tracking algorithm, bubbles can be efficiently assigned corresponding identifiers that are consistent over time. In Figure 4(b), we show the outcome of applying this method on the MFiX-Exa time step 150. Compared to Figure 3(b), where our method is not applied, we observe that the bubble ids are more numerous when using our tracking method. An increased number is intuitive given that after tracking bubbles for approximately 150 time steps, many bubbles have appeared or disappeared. Since unique ids are assigned over time steps, we see that the minimum bubble id in Figure 4(b) starts at 292, whereas in Figure 3(b), all the bubbles have local ids, resulting in a range of numbers from 0 to 19.

As we track the bubbles over time, we can further collect their statistics and life-span summary reports. For each bubble, we can track its time history (see Figure 5) and compute properties—such as volume—as it moves through time.

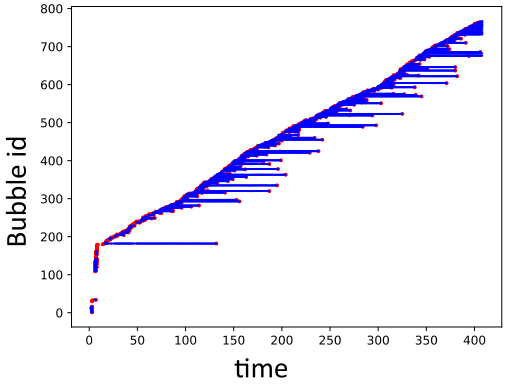

FIGURE 5. Bubble statistics: visualization of evolution of bubbles over time.

BUBBLE-AWARE PARTICLE SAMPLING IN *IN-SITU* SCENARIO

For exascale simulations such as MFiX-Exa, one of the primary needs is to reduce the datasets such that they can also be explored in a posthoc analysis pipeline. For *in-situ* data reduction, there are a few popular choices available to domain experts, such as random and regular sampling. For a particle dataset such as this one, regular sampling often generates less useful results because it can produce visualization artifacts. Random sampling, on the other hand, is generally a safe choice given that the selection of particles is random, thereby avoiding sampling artifacts.

One drawback of using such generic sampling methods for a dataset where the experts are interested in preserving the features (in this case, bubbles) stems from the fact that random methods cannot ensure the features of the dataset will be preserved with high fidelity. In this manuscript, we propose feature-based sampling for this dataset so that the features of the data are better preserved.

In our approach, we first perform the bubble detection as mentioned in the previous section. Since bubble information is contained in a histogram-like derived product, when performing sampling, we use this density histogram. To ensure the bubbles of the data are preserved as best as possible given the storage limitation, our primary goal is to preserve the density histogram. For example, if the original time step consisted of 1 million particles and we need to select 100,000 particles from that time step, the selections are made such that the resulting density histogram from the sampled data looks very similar to the histogram from the original data. To ensure this, we devise a two-pass algorithm over the data where first the density histogram is created. Then, the bubble regions are left empty and particles from the nonbubble regions are selected

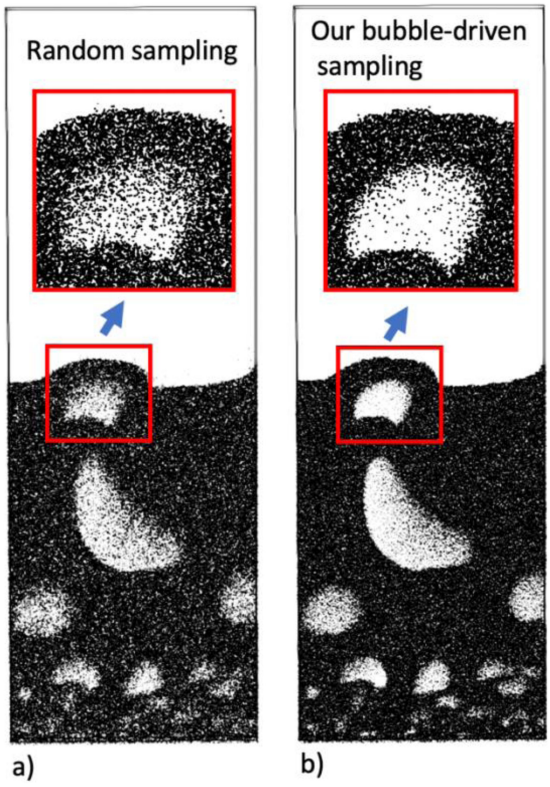

FIGURE 6. Sampling results for the particles from time step 150. Five percent of samples have been taken using (a) random sampling and (b) our proposed bubble-preserving sampling. As seen from the results, the bubbles are more prominent and better preserved (refer to the zoomed-in view) with our method.

proportionate to the original density. Again, since we are applying the analysis primarily to a histogram that is much smaller in size compared to the data, this histogram is shared across all the processors. This enables the distributed sampling of the particles.

An example of such sampling is shown in Figure 6. In this case, we take 5% of samples from the original data using both the random method [see Figure 6(a)] and our proposed method [see Figure 6(b)]. Looking closely into the zoomed-in views, it can be observed that the bubbles are better preserved with our data-sampling method.

DISCUSSION

In this section, we briefly discuss a few of the design choices and reproducibility aspects that will facilitate porting this workflow to the parallel and distributed setting.

Histogram Creation: The creation of a density histogram from the particles is an important step. For scalability, histograms can be created efficiently by the local processors and then MPI_Reduce like operation can be applied to collect all the local histograms into a global histogram. This keeps the data communication quite low and scale to a large-scale data simulation.

Memory Footprint: Global histograms are used by each processor for sampling and bubble detection. Generally, given 3.6 million particles that need *x,y,z* locations to be stored, a density histogram bin resolution of $128 \times 8 \times 64$ produces almost two orders of magnitude smaller memory overhead.

Data Storage: While sampling the particles, each processor operates individually, resulting in good scaling. When writing out the reduced set of particles, parallel I/O modules such as HDF5 can be incorporated in the future.

Reproducibility: The software package MFiX is available at https://mfix.netl.doe.gov (password-protected repository); however, to date no public releases of MFiX-Exa have been made available. MFiX-Exa will be made publicly available in the future, but no release schedule has been established. The particle-tracking software was developed in Los Alamos National Laboratory and is currently in the process of being released as an open source code. Due to national laboratory rules, open sourcing of code is a time-consuming process. We would like to make the code available to the readers as soon as the open-source release process is completed.

FUTURE WORK

In the future, we plan to deploy these in-situ-ready algorithms in a real *in-situ* setting with MFiX-Exa simulations and conduct a detailed performance analysis. We would like to explore generic feature-detection approaches that can apply to a variety of particle/grid-based simulations. Additionally, we would like to apply both unsupervised (e.g., clustering) and semi-supervised (e.g., collecting some expert labeled data and training a classifier) machine learning techniques for more robust and automated feature extraction and tracking. Finally, we would like to explore the possibilities of new data reduction schemes that can be incorporated into the *in-situ* scenario.

CONCLUSION

In this manuscript, we discussed a feature-based analysis of MFiX-Exa particle simulations for various *in-situ* use cases. We described a lightweight bubble-detection algorithm based on spatial histograms for particle density computation. As the time steps arrive one by one in the *in-situ* scenario, we discussed a

bubble-tracking approach from these individually detected bubbles. With a reliable bubble extraction method, we discussed how that can be used for collecting efficient samples from the original set of particles. This will enable *in-situ* data reduction with high fidelity feature preservation. Our proposed in-situ-ready approaches are primarily geared towards MFiX-Exa simulations, but it can still be useful to other particle-based simulations with similar feature-driven analysis needs.

ACKNOWLEDGMENTS

The authors would like to thank the Department of Energy and Los Alamos National Laboratory for the funding and support in carrying out this research. This research was supported by the Exascale Computing Project (17-SC-20- SC), a collaborative effort of the U.S. Department of Energy Office of Science and the National Nuclear Security Administration.

REFERENCES

1. C. R. Müller, S. A. Scott, D. J. Holland, B. C. Clarke, A. J. Sederman, J. S. Dennis, and L. F. Gladden, "Validation of a discrete element model using magnetic resonance measurements," *Particuology*, vol. 7, no. 4, pp. 297–306, 2009.
2. P. Lindstrom, "Fixed-rate compressed floating-point arrays," *IEEE Trans. Vis. Comput. Graph.*, vol. 20, no. 12, pp. 2674–2683, Dec. 31, 2014.
3. S. Li, N. Marsaglia, C. Garth, J. Woodring, J. Clyne, and H. Childs, "Data reduction techniques for simulation," *Vis. Data Anal. Comput. Graph. Forum*, vol. 37, pp. 422–447, 2018.
4. P. N. Duque, D. E. Hiepler, S. M. Legensky, and C. P. Stone, "In-situ feature tracking and visualization of a temporal mixing layer, in *Proc. SC Companion, High Perform. Comput., Netw. Storage Anal.*, 2012, pp. 1593–1593.
5. R. Samtaney, D. Silver, N. Zabusky, and J. Cao, "Visualizing features and tracking their evolution," *Computer*, vol. 27, no. 7, pp. 20–27, Jul. 1994.
6. D. Silver and X. Wang, "Volume tracking," in *Proc. 7th Annu. IEEE Vis.*, 1996, pp. 157–164.
7. A. Biswas, S. Dutta, J. Pulido, and J. Ahrens, "In situ data-driven adaptive sampling for large-scale simulation data summarization," in *Proc. Workshop Situ Infrastruct. Enabling Extreme-Scale Anal. Vis.*, 2018, pp. 13–18.
8. W. Austin, G. Ballard, and T. G. Kolda, "Parallel tensor compression for large-scale scientific data," in *Proc. IEEE Int. Parallel Distrib. Process. Symp.*, 2016, pp. 912–922.
9. F. Sauer, H. Yu, and K. Ma, "Trajectory-based flow feature tracking in joint particle/volume datasets," *IEEE Trans. Vis. Comput. Graph.*, vol. 20, no. 12, pp. 2565–2574, Dec. 31, 2014.
10. S. Dutta, C. Chen, G. Heinlein, H. Shen, and J. Chen, "In situ distribution guided analysis and visualization of transonic jet engine simulations," *IEEE Trans. Vis. Comput. Graph.*, vol. 23, no. 1, pp. 811–820, Jan. 2017.
11. C. Muelder and K. Ma, "Interactive feature extraction and tracking by utilizing region coherency," in *Proc. IEEE Pacific Vis. Symp.*, 2009, pp. 17–24.
12. F. H. Post, B. Vrolijk, H. Hauser, R. S. Laramee, and H. Doleisch, "The state of the art in flow visualisation: Feature extraction and tracking," *Comput. Graph. Forum*, vol. 22, pp. 775–792, 2003.

AYAN BISWAS is a Scientist with the Data Science Group (CCS-7), Los Alamos National Laboratory, Los Alamos, NM, USA. His current research interests include large-scale data reduction, in-situ data analysis, uncertainty quantification, statistical analysis, and high-dimensional data visualization. He received the Ph.D. degree in data visualization from The Ohio State University, Columbus, OH, USA. He is a member of the IEEE and the IEEE Computer Society. Contact him at ayan@lanl.gov.

JAMES P. AHRENS is a Senior Scientist with Los Alamos National Laboratory, Los Alamos, NM, USA. His research interests include large-scale data analysis and visualizations. He received the Ph.D. degree from the University of Washington, Seattle, WA, USA. He is a member of the IEEE and the IEEE Computer Society. Contact him at ahrens@lanl.gov.

SOUMYA DUTTA is a Staff Scientist in the Data Science With Scale Team, Los Alamos National Laboratory, Los Alamos, NM, USA. He received the M.S. and Ph.D. degrees in computer science and engineering from The Ohio State University in 2017 and 2018, respectively. His research interests include statistical data modeling, summarization, and analysis; in-situ data analysis, reduction, and feature exploration; HPC; uncertainty quantification analysis; and time-varying, multivariate data exploration. Contact him at sdutta@lanl.gov.

ANN S. ALMGREN is a Senior Scientist with the Computational Research Division, Lawrence Berkeley National Laboratory, Berkeley, CA, USA, and the Group Lead of the Center for Computational Sciences and Engineering, Berkeley, CA, USA. Her primary research interest is in computational algorithms for solving PDEs in a variety of application areas. Her current projects include the development and implementation of new multiphysics algorithms in high-resolution adaptive mesh codes designed for the latest hybrid architectures. She is an SIAM Fellow and the Deputy Director of the ECP AMR Co-Design Center. Contact her at asalmgren@lbl.gov.

JORDAN M. MUSSER is a Physical Research Scientist with the National Energy Technology Laboratory. His research interests include the development and application of computational multiphase models. He received the Ph.D. degree from West Virginia University, Morgantown, WV, USA. Contact him at jordan.musser@netl.doe.gov.

TERECE L. TURTON is a Staff Scientist with Los Alamos National Laboratory, Los Alamos, NM, USA. Her current research interests include perceptual user evaluation and workflow analysis in scientific visualization. She received the Ph.D. degree from the University of Michigan, Ann Arbor, MI, USA. Contact her at tlturton@lanl.gov.