High-Performance Outlier Detection Algorithm for Finding Blob-Filaments in Plasma

Lingfei Wu*, Kesheng Wu†, Alex Sim†, Michael Churchill‡,
Jong Y. Choi§, Andreas Stathopoulos*, CS Chang‡, Scott Klasky§

*College of William and Mary, Williamsburg, Virginia, USA
†Lawrence Berkeley National laboratory, Berkeley, California, USA
‡Princeton Plasma Physics Laboratory, Princeton, New Jersey, USA
§Oak Ridge National Laboratory, Oak Ridge, Tennessee, USA

***Abstract*—Magnetic fusion could provide an inexhaustible, clean, and safe solution to the global energy needs. The success of magnetically-confined fusion reactors demands steady-state plasma confinement which is challenged by the edge turbulence such as the blob-filaments. Real-time analysis can be used to monitor the progress of fusion experiments and prevent catastrophic events. We present a real-time outlier detection algorithm to efficiently find blobs in fusion experiments and numerical simulations. We have implemented this algorithm with hybrid MPI/OpenMP and demonstrated the accuracy and efficiency with a set of data from the XGC1 fusion simulation code. Our tests show that we can complete blob detection in two or three milliseconds using Edison, a Cray XC30 system at NERSC and achieve linear time speedup. We plan to apply the detection algorithm to experimental measurement data from operating fusion devices. We also plan to develop a blob tracking algorithm based on the proposed method.**

I. Introduction

To extract knowledge from the massive amounts of data available, data mining techniques are frequently used. Many traditional data mining techniques attempt to find patterns occurring frequently in the data, but in this work, we explore outlier detection approaches to discover patterns happening infrequently. Outlier detection is employed in a variety of applications such as fraud detection, time-series monitoring, medical care and public safety and security [1] [2]. Conceptually, an outlier is a data object that deviates significantly from the rest of the objects, as if it were generated by a different mechanism [2]. In some cases, outliers are treated as errors or noise to be eliminated; while in many other cases, outliers can lead to the discovery of important information in the data.

Outlier detection is an important task in many safety critical environments since the outlier indicates abnormal running conditions from which significant performance degradation may well result. An outlier in these applications demands to be detected in real-time and a suitable feedback is provided to alarm the control system. Moreover, the size of ever increasing amounts of data sets dictates the needs for fast and scalable outlier detection methods. In this research, we apply the outlier detection techniques to effectively tackle the fusion blob detection problem on extremely large parallel machines. The blob-filaments are detected as outliers by constantly monitoring specific features of the experimental or simulation data and comparing the real-time data with these features.

With increased global energy needs, magnetic fusion could be a viable future energy which is inexhaustible, clean, and safe. The success of magnetically-confined fusion reactors, like the International Thermonuclear Experimental Reactor (ITER) [3], demand steady-state plasma confinement which is challenged by the edge turbulence such as the blob-filaments. A blob-filament (or blob) is a magnetic-field-aligned plasma structure that appears near the edge of the confined plasma, and has significantly higher density and temperature than the surrounding plasma [4]. Blobs are particularly important to study since they convect filaments of plasma outwards towards the containment wall, which results in substantial heat loss from plasma, degradation of the magnetic confinement, and erosion of the containment wall.

Edge turbulence in magnetic fusion plasmas is an intricate and complex topic, which is not currently well understood. In particular, the mechanisms which blobs are generated by edge turbulence and the complicated feedback systems affecting the turbulence are active areas of research. Previous works have suggested that the drift holes (blobs) and turbulence driven flows are dynamically coupled and regulate each other[5][6]. By identifying and characterizing these blob-filaments over time, physicists can improve their understanding of the dynamics and interactions of such coherent structures (blobs) with edge turbulence and its role in the transport of heat and particles to the edge of the plasmas.

This work is motivated by several considerations. Fusion experiments and numerical simulations can easily generate massive amounts of data per run. During a magnetic fusion device experiment (or "shot"), terabytes of data are generated over short time periods (on the order of hundreds of seconds). In the XGC1 fusion simulation [7][8], a few tens of terabytes can be generated per second. Timely access to this amount of data can already be a challenge, but analyzing all this data in real time is impractical. Currently, there are three types of analysis in most of fusion experiments: in-shot-analysis, between-shot-analysis, and post-run-analysis. All existing blob detection methods address post-run-analysis challenges, but in this work, we focus on the more challenging first two cases to provide a real-time analysis so that scientists can monitor the progress of fusion experiments.

To this end, this work has been integrated into International Collaboration Framework for Extreme Scale Experiments

(ICEE), a wide-area in transit data analysis framework for near real-time scientific applications [9]. ICEE is taking advantage of an efficient IO solution ADIOS [10] and a cutting-edge indexing solution FastBit [11] to design and construct a real-time remote data processing framework over wide-area networks for oceans apart international collaborations such as ITER. In this system, a blob detection algorithm is served to monitor the health of the fusion experiments at Korea Superconducting Tokamak Advanced Research (KSTAR). However, existing data analysis approaches are often single-threaded, only for post-run analysis, and take a long time to produce results. Also, compared to the simulation data, the resolution of the raw camera data may be coarse, but interesting features can still be identified after normalizing process. In order to meet the real-time feedback requirement, we address the challenges by developing a real-time blob detection method, which can leverage in situ raw data in the ICEE server and find blob-filaments efficiently during the fusion experiments. Our blob detection algorithm is not limited to KSTAR only, and can be applied to other real fusion experiments and numerical simulations.

In this paper, we propose a real-time outlier detection algorithm to efficiently find blobs in fusion experiments or numerical simulations. To the best of our knowledge, this is the first research work to achieve real-time blob detection in a few milliseconds. The proposed algorithm is based on two step outlier detection with various criteria and a fast connected component labeling method to find blob components. In the first step, a relatively large electron density area is determined by using a desired confidence level based on the normalized density in the region of interests from all sixteen poloidal planes. In the second step, blob candidates are identified from the relatively large density area by applying an appropriately chosen confidence level in a single poloidal plane. Several blob criteria are applied in order to filter out unwanted plasma points. We also adopt a fast two-pass connected component labeling algorithm from [23] to apply on a refined triangular mesh to find different blob components. We have implemented our blob detection algorithm with hybrid MPI/OpenMP, and demonstrated the effectiveness and efficiency of our implementation with a set of data from the XGC1 fusion simulations. Our tests show that we can complete blob detection in two or three milliseconds using a cluster at NERSC, and achieve linear time speedup. We are currently integrating it into the ICEE system, and plan to test the algorithm in the KSTAR experiments. We also plan to develop a blob tracking algorithm based on the proposed blob detection method.

The rest of paper is organized as follows. In section II, we discuss related work and why we present our blob detection method. Then we describe the outlier detection algorithm for finding blobs, and present a real time blob detection approach to leverage MPI/OpenMP parallelization in section III. The blob detection results and its real time evaluation are shown in section IV. We conclude the paper, and give our future plans in section V.

II. Related Work

The definition of a blob is varied in the literature depending on the fusion experiment or numerical simulation as well as available diagnostic information for measurements. Blob-filaments are often difficult to detect with single-point or two-points correlation methods since they can move erratically and are not necessarily periodic in each cycle [12]. In order to better detect blobs and track their trajectories in 2D poloidal planes, a 2D probe array or an imaging technique like gas puffing imaging (GPI) are employed to directly measure plasma fluctuations at a fast frame rate. After the diagnostic data has been obtained and the blob-filaments have been identified, blob features such as birth rate, size, lifetime, radial speed, burstiness, self-similarity, and multiscale fluctuation components are usually studied to investigate the characteristics of blob structures and its relation with edge turbulent flows.

To study the impact of the size, movement and dynamics of blobs, various post-run blob detection methods have been proposed to identify and track these structures. A plasma blob is most commonly determined by some threshold computed statistically in the local plasma density signal [13][14][12][15]. However, the exact criterion has varied from one experiment to another, which reflects the intrinsic variability and complexity of the blob structures. In [13], a conditional averaging approach is applied to analyze spatiotemporal fluctuation data obtained from a two-dimensional probe array inside the last closed flux surface (LCFS) of the HL-2A tokamak. When the vorticity is larger than one standard deviation at some time frame, a blob is considered to be detected by the probe. In [14], the conditional averaging technique is also used to study the evolution of the blob-filaments using Langmuir probes and a fast camera. If a reference signal with certain sampling interval has large fluctuation amplitudes greater than a specified trigger condition, a blob structure is declared at that time frame.

Without using a conditional averaging technique, [12] searches for blob structures can be done using local measurements of the 2D density data obtained from a 2D probe array. Identification of a blob is based on the choices of several constraints such as the threshold intensity level, the minimum distance of blob movement, and the maximum allowed blob movement between successive frames. The trajectories of the different blobs can be computed with the blob centers based on identification results in each time frame. The seminal work by Zweben, et. al.[12] was the first attempt to take only individual time frame data into account to detect blobs and track their movements, although the process of identification of a blob was somewhat arbitrary and oversimplified. In [15], an analysis method was presented in terms of object-related observables to allow a sound probabilistic analysis. After preprocessing the signals from 2D imaging data to form the signal matrix, a threshold-segmentation approach is used to identify blob structures when the local density is greater than an appropriately chosen threshold. Bounding polygons of the blob structures are also employed to track blob movements and compute the trajectories.

Since the emergence of fast cameras and beam emissions spectroscopy in the last decade, the situations of insufficient diagnostic access and limited spatial and temporal resolution have been greatly improved. In [16], an image analysis for the identification of blobs has been presented based on a gas puff imaging (GPI) diagnostic images from an ultra-high speed, high resolution camera. The raw images are firstly processed to remove the noise spikes, followed by further smoothing

using a Gaussian filter. The blobs are identified by various image segmentation techniques after further processing which removes the background intensity from the images. However, due to noise and lack of a ground truth image, this approach can be sensitive to the setting of parameters and hard to use generic method for all images. Some sophisticated statistical analysis techniques have been exploited to characterize the blob structures and motions. In [17][18], various researchers have leveraged the eigenvalue or singular value decomposition technique to identify the basic components and properties of blob structures.

Recently, several researchers [19][20][21] have developed a blob-tracking algorithm that uses directly raw fast camera data using GPI technique. In [19][21], they leverage a contouring method, database techniques and image analysis software to track the blob motion and changes in the structure of blobs. After normalizing each frame by an average frame created from about a thousand frames around the target time frame, resulting images are contoured and the closed contours satisfying certain size constraints are determined as blobs. Then an ellipse is fitted to the contour midway between the smallest level contours and the peak. All information about blobs are added into a SQL database for more data analysis. This method is close to our approach but it can not be used for real time blob detection since they compute time-averaged intensity to normalize the local intensity. Additionally, only closed contours are treated as blobs, which may miss detection of blobs in the edges of the region of interests. Finally, these methods are still post-run-analysis which cannot provide real-time feedback in real fusion experiments.

III. A REAL-TIME BLOB DETECTION APPROACH

A. *Outlier detection algorithm for efficiently finding blobs*

In this section, we illustrate our outlier detection algorithm to determine blobs to study their characteristics. The main idea is to apply a two-step outlier detection with various criteria and a fast connected component labeling method to separate out the selection of outlier point from regions in space formed by the points. Values for various criteria are determined subjectively by examining the resulting images and adjusting them until satisfied. A sequence of sample processed frames can be either obtained from in situ raw fast camera data from real fusion experiments or numerical simulations. Our data sets are simulated electron density from the code XGC1 [7][8]. In the present data sets, simulation data is captured every 2.5 microseconds for a total time window of 2.5 milliseconds, including all important plasma quantities and a triangulated measurement grid as well as connectivity information.

The first step of the analysis of the turbulence structure is to preprocess the sample frame to compute needed quantities in the users desired region of interests, as shown in figure 1. Then it is analyzed by normalizing the total electron density $n_e(r, z, t)$ (which includes fluctuations) to the initial background electron density, $n_e(r, z, 1)$ (if using real diagnostic data from, e.g. GPI, actual emission intensity $I(r, z, t)$ would be used instead of electron density). Note that using the initial time frame as the benchmark is an important factor to achieve real time blob detection since the normalized electron density in the subsequent time frames can be easily computed,

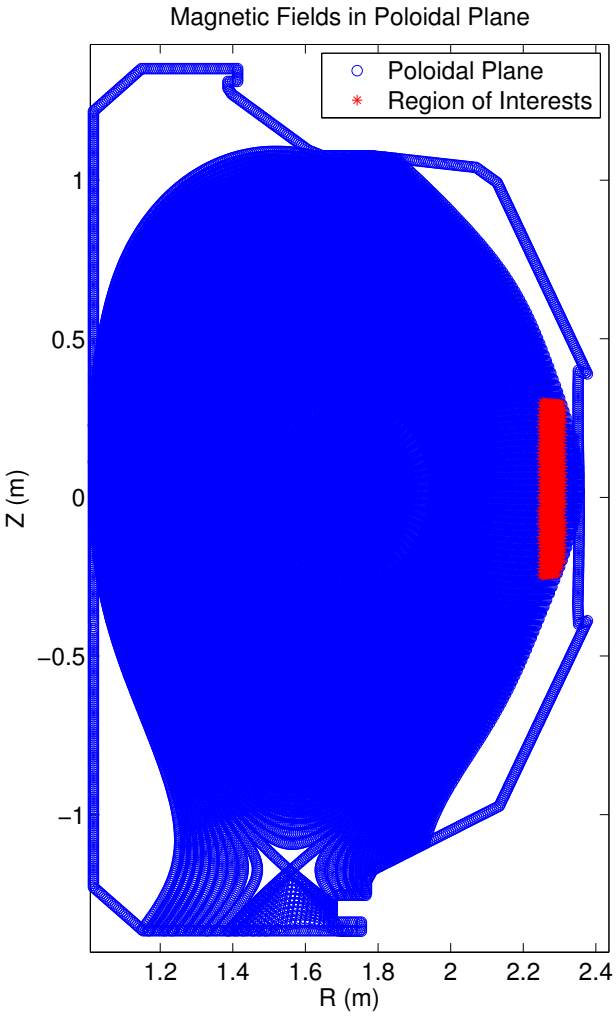


Fig. 1: Region of interests

especially compared to the time-average electron density with long time interval (like 10 milliseconds in [21]).

To obtain a meaningful blob components using connected component labeling method, it is necessary to have a fine grained connectivity information. This particular simulation mesh had coarse vertical resolution, so resolution enhancement techniques were applied to generate higher resolution triangular mesh based on the original triangulated mesh. Although there are different variation of the Delaunay refinement algorithm generating unstructured meshes of triangles [22], we are using a simple triangular mesh refinement algorithm since the original triangular mesh has been created well without small angles. The resulting triangular mesh is refined to a higher resolution one with 4 times more triangles by creating new vertexes with the three middle points of original mesh edges in each triangle. The corresponding density of generated vertexes can be obtained by a linear interpolation in the original triangular mesh. This step can be applied recursively until the satisfactory resolution of the triangular mesh is computed.

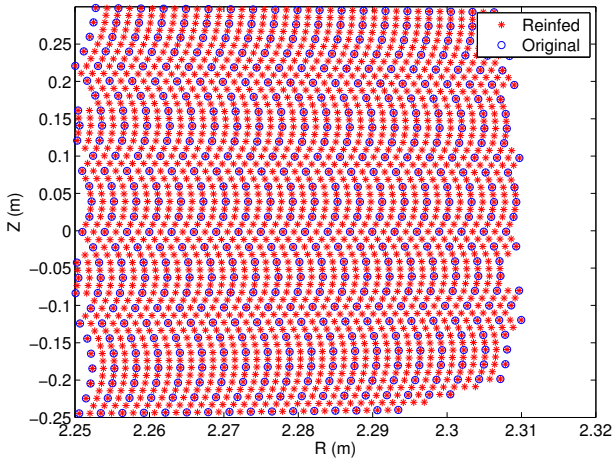


Fig. 2: Refined and original vertexes

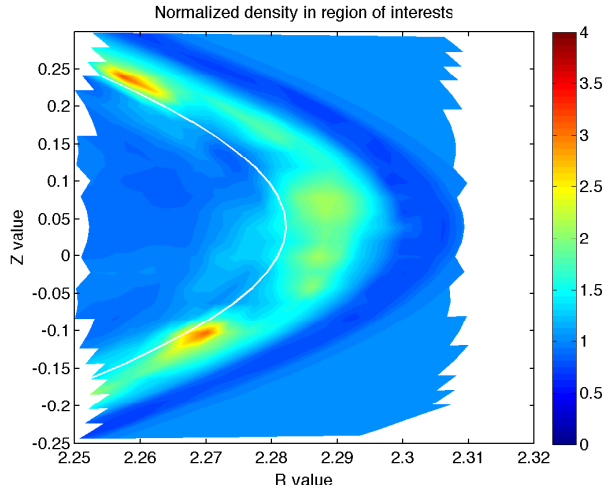


Fig. 4: A contour plot of the local normalized density in the region of interests in one time frame

Figure 2 shows the resulting triangular mesh vertexes after applying the triangular mesh refinement algorithm once.

In order to apply an appropriate predefined quantile in the two-step outlier detection, it is advised to perform exploratory data analysis to exploit main characteristics of the data sets. As shown in Figure 3, extreme value distribution and log normal distribution are fitted best with one of our sample data sets over sixteen different common distributions. After analyzing the underlying distribution, a two-step outlier detection is performed to determine blob candidates in the region of interests. As shown in Figure 4, the basic idea of the proposed two-step outlier detection is motivated from the observations that there are a relatively high density region (a half banded ellipse area with cyan color) in the edge and several significantly high density small regions (a few small areas with reddish yellow color) in this relatively high density region. It extends the previous approach that applies one step outlier detection with conditional averaging intensity value, and applies more intelligent two-step outlier detection with only considering individual time frame data. Compared to traditional single threshold segmentation approach, our approach is more generic, flexible and easier to tune a satisfactory result.

In the first stage of the two-step outlier detection, the standard deviation σ and the expected value μ are computed over all poloidal planes in one time frame. Using the best fitted distribution, we apply first step outlier detection to identify the relative high density regions with a specified predefined quantile:

$$N(r_i, z_i, t) - \mu > \alpha * \sigma, \forall (r_i, z_i) \in \Gamma \quad (1)$$

where N is the normalized electron density, α is the multiple of σ associated to the specified predefined quantile and Γ is the domain in the region of interests. Once the relative high density regions are determined, we compute another standard deviation σ_2 and the expected value μ_2 in these regions. Then we employ second step outlier detection to identify the blob candidates in the relative high density regions with an appropriately chosen predefined quantile:

$$N(r_i, z_i, t) - \mu_2 > \beta * \sigma_2, \forall (r_i, z_i) \in \Gamma_2 \quad (2)$$

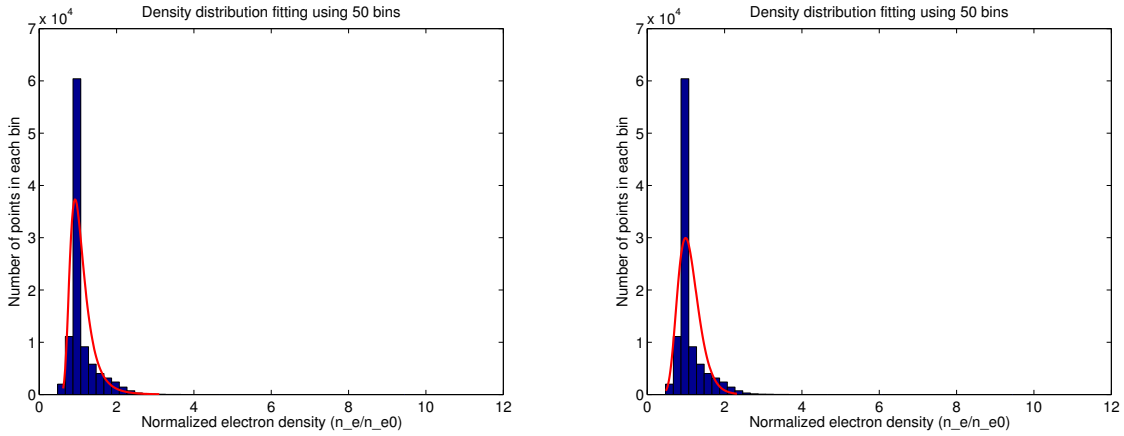
where β is the multiple of σ_2 associated to the judiciously chosen confidence level and Γ_2 is the domain of blob candidates. In practice, α and β could be chosen same or different, depending on the characteristics of blob-filaments in the fusion experiments or numerical simulations. In our experience, α value is generally greater than β since the standard deviation σ over the region of interests is much smaller than the standard deviation σ_2 from the relative high density regions.

However, the two-step outlier detection cannot be used alone to distinguish the blob candidates since identified blob candidates may have actual small density, which does not satisfy traditional definition of blobs. Therefore, the density of the mesh points in the blob candidates smaller than certain minimum absolute value criterion need to be filtered out. On the other hand, it is also possible that the middle areas between surrounding plasmas and blob components have density value higher than the given minimum absolute value criterion. Thus, we also apply a minimum relative value criterion to remove these unwanted points. To combine these two rules together, we have a more robust and flexible criterion:

$$N(r_i, z_i, t) > \mathbf{max}(d_{ma}, (d_{mr} * \mu_2)), \forall (r_i, z_i) \in \Gamma_3 \quad (3)$$

where d_{ma} and d_{mr} are minimum absolute value and minimum relative value respectively, and Γ_3 is the domain of good blob candidates.

With good blob candidates, we apply an efficient connected component labeling algorithm adopted from [23] on a refined triangular mesh to find different blob components. A connected component labeling algorithm generally considers the problem of labeling binary 2D images with either 4-connectedness or 8-connectedness. It performs efficient scanning technique, and fills the label array labels so that the neighboring object pixels have the same label. In our problem, we are working on a refined triangular mesh thereby each triangle are scanned firstly. Since we know the three vertexes in a triangle are connected, we can reduce unnecessary memory accesses once any vertex in a triangle is found to be connected with another vertex in a different triangle. Then we compute the current



(a) Extreme Value Distribution

(b) Log Normal Distribution

Fig. 3: An example of exploratory data analysis to analyze the underlying distribution of the local normalized density over all poloidal planes and time frames.

minimum parent label in this triangle, and assign each vertex a parent label if its label has already filled or a label if its label has not initialized yet. If all three vertexes in a triangle are first time scanned, then a new label number is issued and assigned to their labels and associated parent label. After the label array is filled full, we need flatten the union and find tree. Finally, second pass is performed to correct labels in the label array, and all blob candidates components are found.

After all blob candidates components are determined, a blob is claimed to be found if the median of a blob candidate component satisfies certain minimum absolute median value criterion. The reason we are setting this constraint to select the blobs is that the minimum value criterion has to be a reasonably small value to produce more blob candidate components. It is possible that the minimum absolute median value criterion is too large that it may also remove the blobs, while it is also possible that this value is too small that it does not have effect on filtering out unwanted components. Therefore, with the same philosophy of measurement, a minimum relative median value criterion is also applied to determine the blobs. However, in order to protect the blobs from being removed due to the extremely large mean value μ_2, we also set the maximum absolute median value criterion to limit the power of minimum relative median value criterion. We unify these three rules to be one:

$$N(r_i, z_i, t) > \mathbf{max}(\hat{d_{ma}}, min((\hat{d_{mr}} * \mu_2), \hat{d_{xa}})), \quad \forall (r_i, z_i) \in \Gamma_4 \quad (4)$$

where $\hat{d_{ma}}$, $\hat{d_{mr}}$ and $\hat{d_{xa}}$ are minimum absolute median value, minimum relative median value and maximum absolute median value respectively and Γ_4 is the domain of blobs.

B. A hybrid MPI/OpenMP parallelization

Existing blob detection approaches cannot tackle the two challenges of the large amount of data produced in a shot and the real time requirement. In addition, existing data analysis approaches are often operated in a single thread, only for post-run analysis and take often a few hours to generate the all results [15]. In order to meet the real-time feedback requirement, we address the challenges by developing a real time blob detection method, which can leverage in situ raw data and find blob-filaments efficiently in the fusion experiments or numerical simulations. In our approach, we can complete our blob detection in a few milliseconds using a hybrid MPI/OpenMP parallelization with in situ evaluation. The key idea is to exploit many cores in a large cluster system by running MPI to allocate n processes to process each time frame and by leveraging OpenMP to accelerate the computations using m threads. Our hybrid MPI/OpenMP parallelization for blob detection is shown in Figure 5.

In order to achieve blob detection in real time, the goal is to minimize the data movements and speed up the computation in each process. Ideally, the performance is optimal without any communication if we can perform the job correctly. The proposed blob detection algorithm in the previous section supports embarrassed parallel since we only need the initial time frame and the target time frame to do the computation. This is an important difference of our blob detection method compared to recently developed methods [19][21] in term of the real time requirement. Furthermore, we can explore many-core processor architecture to speed up the computation by taking advantage of multithreading in the shared memory. Therefore, our real time blob detection method based on hybrid MPI/OpenMP parallelization is a natural choice and expected to provide the optimal performance.

C. Outline of the implementation

We implement our blob detection algorithm in C with a hybrid MPI/OpenMP parallelization. The algorithm 1 summarizes the proposed blob detection algorithm without considering OpenMP. Users can specify the region of interests by (Rmin, Rmax, Zmin, Zmax), target files range by (t_start, t_end), and directory containing data files by FileDir. However, with in situ evaluation, there is no need to specify the file

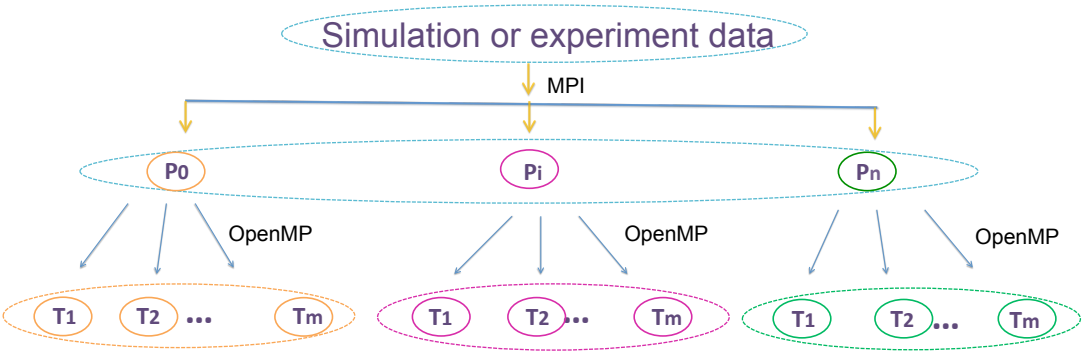


Fig. 5: Hybrid MPI/OpenMP parallelization

Algorithm 1 Real time blob detection algorithm

1: Initial inputs: Rmin, Rmax, Zmin, Zmax, t_start, t_end, FileDir.
2: Process i loads raw data in each time frame and computes position and normalized electron density (r,z,ne)
3: Refine the triangular mesh in the region of interests
4: Apply two-step outlier detection to identify blob candidates with judiciously chosen confidence level
5: Compare the normalized density of blob candidates with the minimum density criterion to filter out unwanted ones
6: Apply fast connected component labeling algorithm on a refined triangular mesh and compute blob components
7: A blob is found if its median satisfies minimum median density criterion

directory since all data are already in the memory. The n MPI processes are allocated to process one or several time frames and m OpenMP threads are launched to accelerate the computation in one time frame. Note that MPI process is also the master thread in the runtime environment.

IV. Experiments and Results

In this section we present an evaluation of our blob detection algorithm, and report its performance with real time detection. Before showing experimental results in the next, we briefly introduce our experimental environment, data sets and parameters setting in our blob detection algorithm. We have tested our implementation on the NERSC's newest supercomputer Edison, where each compute node has two Intel "Ivy Bridge" processors (2.4GHz with 12 cores) and 64 GB of memory. Our data sets are small simulation data sets (30GB) with 1024 time frames based on the XGC1 simulation [7][8] from the Princeton Plasma Physics Laboratory, which last around 2.5 milliseconds. One of our main goals is to complete the blob detection on the entire data set in a time close to 2.5 milliseconds, which would indicate that our algorithm could monitor fusion experiments in real time (neglecting data transfer latency).

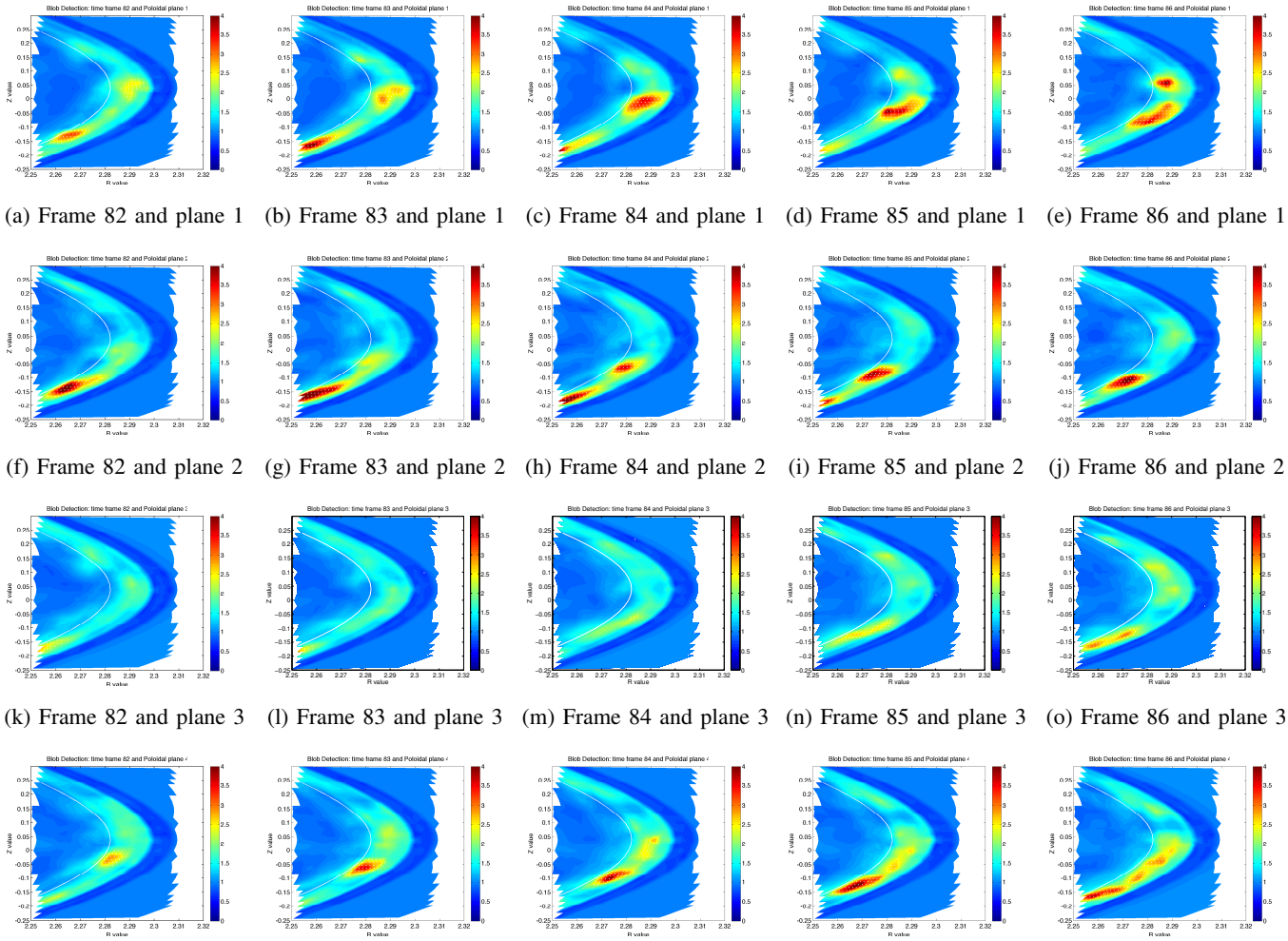
Another goal is to validate the effectiveness of the proposed algorithm. In the algorithm 1, we apply various criteria in order to identify the blobs. The parameters for blob detection in our experiments are given in Table I. One more criterion we have not mentioned in the previous section is parameter "minArea". This parameter is used to decide how many points a blob should have, which can mainly prevent the very small blobs. In our experiment, this parameter is set to 3 since there are at least three vertexes connected as a 2D component in a triangular mesh. We have to mention that these parameters need to be tuned in order to achieve optimal performance in different fusion experiments or numerical simulations. The reasons for this uncertainty in the context of blob detection are from the intrinsic variability and complexity of the blob structures observed in different experiments [4].

TABLE I: Parameters setting for the proposed blob detection algorithm on XGC1 simulation data sets in this paper.

detection criteria	
minArea	3
minRden	1.2
minAbsRden	2.05
maxAbsRden	2.75
minMden	1.3
minAbsMden	2.15

The blob detection results in five continuous time frames and four different poloidal planes are shown in Figure 6. Compared to the recent developed methods in [19][21], our method does not miss blobs in the edge of regions of interests as shown in subfigures 6b, 6g, 6c and 6h. It is interesting to see that large-scale blob structures are often generated, which could cause substantial plasma transport [12]. As pointed out in [17], these large-scale structures are mainly contributed by the low-frequency and long-wavelength fluctuating components, which may be responsible for the observations of long-range correlations. We also noticed that different poloidal planes may display significant diversity of the edge turbulence even in the same time frame. We have shown that we are able to effectively detect the blobs and reveal some interesting results to help physicists improve their understanding on the characteristic of blobs and its correlation with other plasma properties.

Our most encouraging results are that we can complete blob detection on the simulation data set described above in around 2 milliseconds with MPI/OpenMP using 4096 cores and in 3 milliseconds with MPI using 1024 cores. In Figure 7, we can see that the hybrid MPI/OpenMP implementation is about two



(a) Frame 82 and plane 1 (b) Frame 83 and plane 1 (c) Frame 84 and plane 1 (d) Frame 85 and plane 1 (e) Frame 86 and plane 1

(f) Frame 82 and plane 2 (g) Frame 83 and plane 2 (h) Frame 84 and plane 2 (i) Frame 85 and plane 2 (j) Frame 86 and plane 2

(k) Frame 82 and plane 3 (l) Frame 83 and plane 3 (m) Frame 84 and plane 3 (n) Frame 85 and plane 3 (o) Frame 86 and plane 3

(p) Frame 82 and plane 4 (q) Frame 83 and plane 4 (r) Frame 84 and plane 4 (s) Frame 85 and plane 4 (t) Frame 86 and plane 4

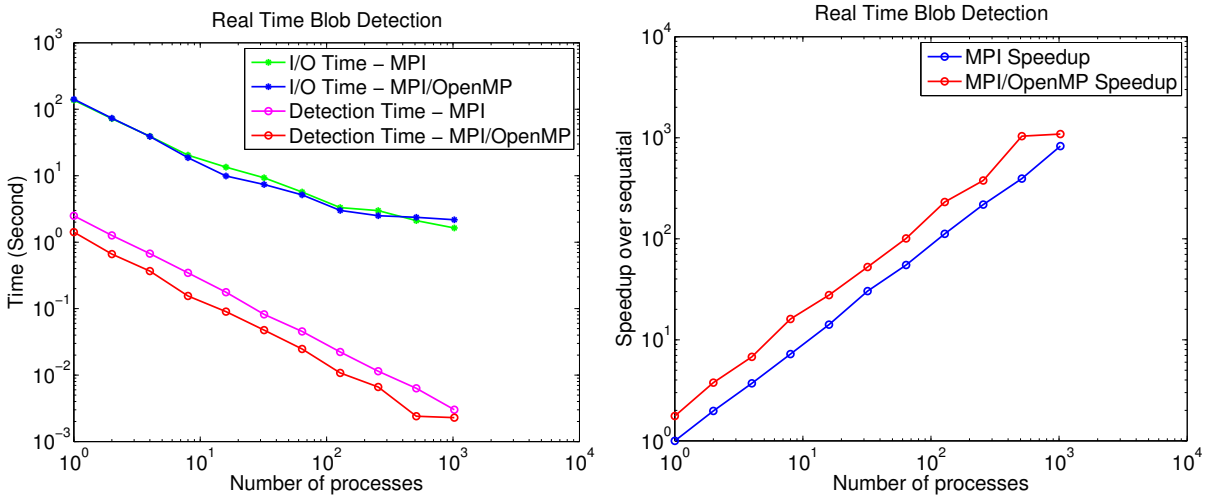
Fig. 6: An example of the blob detection in five continuous time frames and four different poloidal planes in the R (radial) direction and the Z (poloidal) direction. The separatrix position is shown by a white line and the different color stars denote blob components.

times faster than the MPI implementation when varying the number of processes from 1 to 512. With 1024 processes, both of them achieve close performance, but the MPI/OpenMP one is slightly better. Also, we can achieve linear time scalability in blob detection time and slightly superlinear in I/O time. MPI and MPI/OpenMP implementations accomplish 800 and 1200 times speedup when the number of processes scale to 1024, respectively. We have been able to control analysis speed by varying the number of processes. One of future plans is to exploit a different number of threads to tune the best performance.

V. Conclusion and Future Work

Large blob structures elongated along the magnetic field lines significantly contribute to the energy and plasma transport in the scrape-off layer, which degrades the plasma confinement and causes deleterious effects due to wall interactions. In this paper, we present for the first time a real time blob detection method for finding blob-filaments in real fusion experiments or numerical simulations. The proposed algorithm is based on two-step outlier detection with various criteria and a fast connected component labeling method to find blob components. We have implemented our blob detection algorithm with hybrid MPI/OpenMP and demonstrated the accuracy and efficiency of our implementation with a set of data from the fusion plasma simulation code XGC1. Our tests show that we can complete blob detection in two or three milliseconds using a cluster at NERSC and achieve linear time speedup.

We are currently working on integrating our blob detection algorithm into the ICEE system where the blob detection function serves a central data analysis component and the resulting detection results are monitored and controlled from portable devices like an iPad. We plan to test the proposed method very soon in both numerical simulation and real fusion experiments. An interesting future work is to develop a blob tracking algorithm based on the proposed blob detection method. Furthermore, it would be also interesting to explore biorthogonal decomposition technique for the determination of



(a) Time (b) Speedup

Fig. 7: Blob detection time, I/O time and speedup with MPI and MPI/OpenMP varying number of processes

blob-filaments.

ACKNOWLEDGMENT

This work was supported by the Office of Advanced Scientific Computing Research, Office of Science, of the U.S. Department of Energy under Contract No. DE-AC02-05CH11231 and partially supported by NSF under a grant No. CCF 1218349, and by DOE under a grant No. DE-FC02-12ER41890. The authors would like to thank Scientific Data Management Group at LBNL, and our collaborators in PPPL and ORNL for their contributions to this work.

REFERENCES

[1] Hodge, Victoria J., and Jim Austin, "A survey of outlier detection methodologies." *Artificial Intelligence Review*, 22.2 (2004): 85-126.

[2] Jiawei Han and Micheline Kamber, *Data Mining, Southeast Asia Edition: Concepts and Techniques*, Morgan kaufmann, 2006.

[3] ITER web page, "The International Thermonuclear Experimental Reactor (ITER) project", http://www.iter.org/, November 2004.

[4] D. A. D'Ippolito, J. R. Myra, and S. J. Zweden, "Convective transport by intermittent blob-filaments: Comparison of theory and experiment", *PHYSICS OF PLASMAS* 18, 060501 (2011).

[5] Y. Kosuga and P. H. Diamond, "On relaxation and transport in gyrokinetic drift wave turbulence with zonal flow", *PHYSICS OF PLASMAS* 18, 122305 (2011).

[6] Y. Kosuga and P. H. Diamond, "Drift hole structure and dynamics with turbulence driven flows", *PHYSICS OF PLASMAS* 19, 072307 (2012).

[7] S. Ku, C.S. Chang, and P.H. Diamond, "Full-f gyrokinetic particle simulation of centrally heated global ITG turbulence from magnetic axis to edge pedestal top in a realistic tokamak geometry", *Nuclear Fusion* 49, 115021 (2009).

[8] C.S. Chang, S. Ku, P. H. Diamond, Z. Lin, S. Parker, T. S. Hahm and N. Samatova, "Compressed ion temperature gradient turbulence in diverted tokamak edge," *Phys. Plasmas* 16, 056108 (2009).

[9] Jong Y. Choi, Kesheng Wu, Jacky C. Wu, et al, "ICEE: Wide-area In Transit Data Processing Framework For Near Real-Time Scientific Applications", *4th SC Workshop on Petascale (Big) Data Analytics: Challenges and Opportunities* in conjunction with SC13, 2013.

[10] J. Lofstead, S. Klasky, K. Schwan, et al, "Flexible IO and integration for scientific codes through the adaptable IO system (ADIOS)", in *Proceedings of the 6th international workshop on Challenges of large applications in distributed environments* ACM, 2008, pp. 15-24.

[11] K. Wu, S. Ahern, E. W. Bethel, et al, "Fastbit: Interactively searching massive data", in *SciDAC 2009*, 2009, IBNL-2164E.

[12] K. J. Zweben, "Search for coherent structure within tokamak plasma turbulence", *Phys. Fluids* 28(3), March 1985.

[13] M. Xu, G.R.Tynan, P.H.Diamond, et al, "Turbulent eddy-mediated particle, momentum, and vorticity transport in the edge of HL-2A tokamak plasma", in *24th IAEA Fusion Energy Conference*, San Diego, 2012, IAEA CN-197/EX/72Rb (APS, 2012).

[14] G Fuchert, G Birkenmeier, B Nold, et al, "The influences of plasma edge dynamics on blob properties in the stellarator TJ-K", *Plasma Phys. Control. Fusion* 55(2013), 125002(8pp).

[15] S. H. Muller, A. Diallo, A. Fasoli, et al, "Probabilistic analysis of turbulent structures from two-dimensional plasma imaging", *Phys. Plasmas* 13, 100701 (2006).

[16] Nicole S. Love and Chandrika Kamath, "Image analysis for the identification of coherent structures in plasma", *Applications of Digital Image Processing XXX*, SPIE Conference 6696, San Diego, August 2007

[17] G.S. Xu, B.N. Wan, W. Zhang, et al, "Multiscale coherent structures in tokamak plasma turbulence", *Phys. Plasmas* 13, 102529 (2006).

[18] H. Tanaka, N. Ohno, Y. Tsuji, et al, "2D statistical analysis of non-diffusive transport under attached and detached plasma conditions of the linear divertor simulator", *Contrib. Plasmas Phys.* 50, No.3-5, 256-266 (2010).

[19] W.M Davis, M. K. Ko, et al, "Fast 2-D camera control, data acquisition, and database techniques for edge studies on NSTX", *Fusion Engineering and Design*, Volume 89, Issue 5, May 2014, Pages 717720.

[20] R. Kube, O. E. Garcia, et al, "Blob sizes and velocities in the Alcator C-Mod scrape-off layer", *Journal of Nuclear Materials*, Volume 438, Supplement, July 2013, Pages S505S508.

[21] J.R. Myra, W.M Davis, D. A. D'Ippolito, et al, "Edge sheared flows and the dynamics of blob-filaments", *Nuclear Fusion*, Volume 53 (2013) 073013 (15pp).

[22] Jonathan Richard Shewchuk, "Delaunay refinement algorithms for triangular mesh generation", *Computational Geometry*, Volume 47, Issue 7, August 2014, Pages 741-778.

[23] Kesheng Wu, Ekow Otoo, and Kenji Suzuki, "Optimizing two-pass connected-component labeling algorithms", *Pattern Analysis and Application*, (2009) 12:117-135.