DOI: 10.1111/cgf.14304
Eurographics Conference on Visualization (EuroVis) 2021
R. Borgo, G. E. Marai, and T. von Landesberger
(Guest Editors)

Volume 40 (2021), Number 3

Leveraging Topological Events in Tracking Graphs for Understanding Particle Diffusion

T. McDonald[1], R. Shrestha[2], X. Yi[3], H. Bhatia[3], D. Chen[2], D. Goswami[2], V. Pascucci[1], T. Turbyville[2], and P.-T. Bremer[3]

[1] Scientific Computing & Imaging Institute, University of Utah, Salt Lake City, UT, USA
[2] The Frederick National Laboratory for Cancer Research, Frederick, MD, USA
[3] Lawrence Livermore National Laboratory, Livermore, CA USA

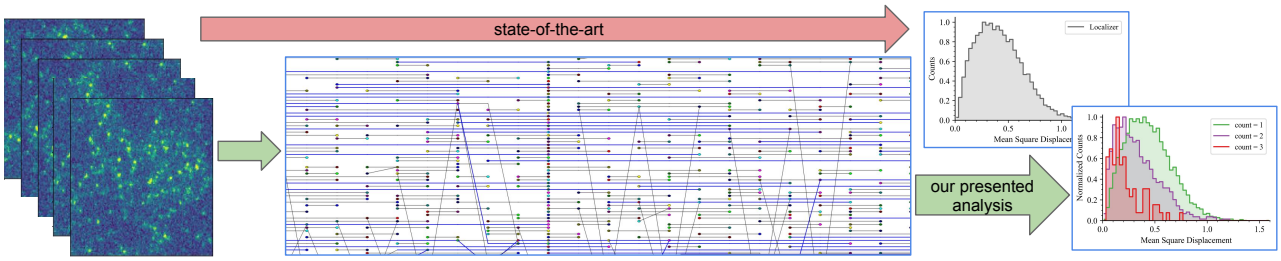


Figure 1: *Single particle tracking (SPT) of fluorescently labeled proteins (bright spots in the left image) is traditionally used to derive distribution of Mean Square Displacement (MSD) for all observed features (gray) as a measure of their diffusion. However, due to experimental limitations, it is impossible to distinguish single particles from clusters in the image leading to broad distributions of MSD and no direct link between cluster size and diffusion. Analyzing the merge and split events in the corresponding tracking graph (middle) determines, for the first time, estimates of the number of particles in each cluster leading to conditional MSD distributions (colored). These results confirm the prior hypothesis that observed changes in MSD are due to clustering, and that smaller clusters diffuse faster than bigger clusters.*

Abstract

Single particle tracking (SPT) of fluorescent molecules provides significant insights into the diffusion and relative motion of tagged proteins and other structures of interest in biology. However, despite the latest advances in high-resolution microscopy, individual particles are typically not distinguished from clusters of particles. This lack of resolution obscures potential evidence for how merging and splitting of particles affect their diffusion and any implications on the biological environment. The particle tracks are typically decomposed into individual segments at observed merge and split events, and analysis is performed without knowing the true count of particles in the resulting segments. Here, we address the challenges in analyzing particle tracks in the context of cancer biology. In particular, we study the tracks of KRAS protein, which is implicated in nearly 20% of all human cancers, and whose clustering and aggregation have been linked to the signaling pathway leading to uncontrolled cell growth. We present a new analysis approach for particle tracks by representing them as tracking graphs and using topological events – merging and splitting, to disambiguate the tracks. Using this analysis, we infer a lower bound on the count of particles as they cluster and create conditional distributions of diffusion speeds before and after merge and split events. Using thousands of time-steps of simulated and in-vitro SPT data, we demonstrate the efficacy of our method, as it offers the biologists a new, detailed look into the relationship between KRAS clustering and diffusion speeds.

CCS Concepts

• ***Human-centered computing*** → ***Scientific visualization;*** • ***Applied computing*** → ***Computational biology;***

1 Introduction

The development of fluorescence microscopes, coupled with the ability to tag individual proteins with fluorescence molecules, gives rise to *single particle tracking* (SPT), which is one of the most crucial tools in a wide range of biological applications [SJ97, Kra15]. With specific fluorescent labels, experimental biologists are able to tag proteins of interest (*e.g.,* a particular drug or a messenger compound) and observe its motion both *in-vitro* and *in-vivo* [AZG06, HF13, MGP15, MRS*18]. However, the state-of-the-art SPT tools [DDNZ12] are still limited in resolution, especially under noisy conditions, lacking the ability to distinguish individual particles from clusters of particles. This bottleneck implies that particle motion is widely analyzed without knowing the true count of particles in the observed tracks, which poses significant challenges in the scientific interpretation of the data and limits the insights that can be delivered through analysis.

Here, we present a new solution to the aforementioned challenges using techniques from topological analysis and from the vi-

sualization community, in the application context of cancer biology. Our collaborators at the Frederick National Laboratory for Cancer Research are interested in understanding the clustering behavior of the KRAS4b (KRAS) protein, which modulates the signaling pathway for cell growth. Approximately 19% of patients with cancer harbor RAS mutations, with KRAS responsible for 75% of that number [PHH20]. Consequently, there is significant interest in understanding the underlying biological mechanisms involved, in hopes of developing effective treatments. However, despite decades of efforts, such as the RAS Initiative [NCI] of the National Cancer Institute in the US, it has proven challenging to find effective inhibitors of KRAS to the extent where, until recently, it had often been labeled *undruggable* [KGM*19]. The community is spending significant attention into better understand the entire signaling cascade in hopes of finding ways to effect KRAS indirectly. Moreover, it is known that signaling only occurs with KRAS bound to a cell membrane and that may require *nanoclustering* (hereafter, *clustering*) of multiple KRAS to activate the next link in the chain (the RAF protein). The clustering of KRAS is modulated by the local lipid membrane composition. This paper presents new analysis techniques to help quantify the effects of different lipid environments on the clustering and diffusion of bound KRAS.

As discussed in more detail in Section 2.2, our collaborators create various lipid environments by varying the relative concentrations of different lipid types and observe the behavior of KRAS proteins tagged with fluorescent molecules under a total internal reflection fluorescence (TIRF) microscope. This setup acquires time-sequences of tens of thousands of frames, such as those shown in Figure 1, in which, labeled KRAS appear as fluorescent spots. Each of these frames is then segmented to locate individual particles, which are subsequently linked through time to create the so called *tracks* representing the paths of molecules over time. However, the current state-of-the-art SPT analysis tools [JLM*08, DDNZ12] do not distinguish single KRAS from clusters of multiple KRAS. Furthermore, proteins can attach and/or detach themselves to the membrane and, thus, leave or reappear in the observed frame, and there exist a substantial number of unlabeled and thus invisible KRAS (as much as 95%). Therefore, it is often ambiguous to determine from any single frame whether a given feature represents one or multiple KRAS, especially with molecules that are moving. To compensate for these challenges, the current state-of-the-art considers only standalone tracks, *i.e.,* those that contain neither merges nor splits, and instead treats each segment between such events as an individual track. Analyzing the diffusion of each standalone track provides an indirect measurement on how many KRAS might be present. More specifically, it is expected that clusters of multiple KRAS will diffuse slower than individual molecules, which would make the average diffusion an indirect measurement of the clustering dynamics. Indeed, scientists have observed that different membrane compositions expected to have promoted or inhibited clustering of KRAS show markedly different diffusion [INC*20]. Nevertheless, as tracks cannot be independently labeled according to the number of KRAS that are present, it remains unclear whether this difference is due to clustering.

Here, we introduce a new analysis approach that considers the entire graph of all tracks, including merge and split events, in order to provide more-direct evidence that clustering is indeed correlated with slower diffusion. In particular, by carefully following tracks through merge and split events, we provide estimates of the number of labeled KRAS within each track, *i.e.,* by recognizing that after observing a merge or before observing a split, the combined track is likely to contain at least two (labeled) KRAS. This analysis enables a direct comparison of the distribution of diffusion for tracks before and after merge/split events, which are highly likely to represent smaller/larger clusters. Our results indicate that not only do there exist populations with distinct diffusion, but, on average, the diffusion decreases after merges and increases after splits, further supporting the hypothesis that such changes are, in fact, directly correlated with KRAS clustering. We have integrated this analysis technique into an interactive linked-view tool that enables quick exploration of various types of segmentation and linking options as well as assembly of the corresponding statistics on-the-fly. Finally, we provide an in-depth verification study using simulated data of the same biological system [DNBC*19,INC*20] as well as an analysis of two different experiments demonstrating how our approach is providing novel insights to our collaborators. The specific contributions of this paper are:

- A novel analysis method for SPT tracks that takes advantage of topological events in the tracking graph to provide enriched statistics;
- An interactive, linked-view system that enables the intuitive exploration of experimental and simulated data, including parameter and sensitivity studies;
- Verification of the analysis approach using matching simulation data; and
- A case study of experiments that link observed diffusion to KRAS clustering and, thus, indicate how different lipid environments affect KRAS clustering.

2 Application

As mentioned above, this work is motivated by the needs of the experimental biology community, to use particle trajectories from SPT to explore the link between clustering and diffusion. In the context of KRAS and our specific application, the hypothesis is that the local membrane environment, *i.e.,* the types and relative concentrations of lipids making up the membrane, can promote or inhibit KRAS clustering. If proven correct, then manipulating the membrane environment could provide an indirect means to modulate the clustering and hence the downstream signaling pathway.

2.1 Role of KRAS Clustering in Cancer Signaling

Oligomerization, clustering, and the assembly and disassembly of macro-molecular complexes is a major component of regulating the timing, location, and function of molecules in cells. The cell membrane is a highly dynamic 2D organelle that functions, in part, to regulate cell signaling. Here, we are interested specifically in the KRAS4b protein (and other RAS isoforms) which are small G-proteins that tether to the plasma membrane and are frequently mutated in cancer. This signaling molecule only functions when it is tethered to the membrane, and acts as a molecular switch to turn on and off signaling that leads to cell growth and other cell fate functions. In cancer, KRAS is essentially stuck in the on position with uncontrolled growth of the cells harboring the mutation. There is speculation that some sort of KRAS clustering is part of

the activation process. However, the exact mechanism, ordering of events, number of KRAS involved, *etc.* remain unclear. While stable dimers of KRAS have not been observed in biochemical experiments, clusters of KRAS proteins have been seen in electron micrographs in nonrandom distributions [PMPH03]. This observation has lead to the intriguing hypothesis that KRAS nanoclusters, or perhaps dynamic oligomerization events, may be driving activation of the downstream signaling events. Since the biological significance of nanoclustering is not clear, it is important to clarify the dynamics of the behavior, and understand the precise mechanism of assembly. Questions that remain are: What is the order of events? What are the kinetics? Is it random, or nonrandom? And, finally and most importantly, is the KRAS clustering a drug target? One initial direction of inquiry to potentially answer the latter question is how to quantify changes in KRAS dynamics given the fact that direct experimental observation of nanoclusters are challenging.

2.2 Data Acquisition and Processing

To explore the aforementioned hypothesis, our collaborators created different plasma membranes and exposed them to fluorescently labeled KRAS. However, to reduce the observed particle density and allow SPT, only ∼5% of KRAS proteins were labeled. Subsequently, the KRAS molecules on the membrane were imaged with a Nikon N-STORM system built on an eclipse Ti microscope (Nikon, Japan) equipped with an APO 100×, 1.49 NA objective under TIRF illumination mode. Well-separated single molecules in each frame were captured by a thermoelectric-cooled EMCCD camera (iXon Ultra DU-897, Andor Technologies, USA) as diffraction limited patches at high speed (100 fps). Up to 5000 frames were acquired each run, but each experiment took around 20 runs. The raw image frames were organized into an image stack in TIFF format for further processing. The resolution of each image is 0.16 µm/pixel. The experiment used the Localizer package, embedded in the Igor Pro software (Wave Metrics, Inc. USA) [DDNZ12] to segment each single molecule as a diffraction limited spots from each frame with the eight-way adjacency particle detection algorithm with 30 GLRT [SBRM08] sensitivity and a point spread function (PSF) of 1.3 pixels. High resolution spot position localization was obtained through the 2D Gaussian fit of the PSFs for each frame.

3 Related Work

Traditional SPT is performed with two steps: (1) the fitting of the particle positions (*segmentation* or *feature detection* step), which provides a list of localization fittings in the form of time-tagged location coordinates of particles; and (2) the linking of these fitted particle positions into trajectories of moving particles (*tracking* or *feature correlation* step), which produces the spatial diffusion trajectories of each particle over time. Various tools exist to perform these tasks as either integrated packages or as separate steps. Here, we discuss prior work both in the context of the SPT data specifically and the analysis of time-dependent features more generally.

3.1 Techniques for SPT Data Analysis

Segmentation and tracking are critical steps in the SPT analysis pipeline. A number of standard algorithms exist for both steps, and even a cursory overview of all techniques is beyond the scope of this paper [DDNZ12, YPW19, TPS*17]. Since our analysis is independent of the choice of pre-processing algorithm we instead focus on existing techniques to analyze the resulting graphs. There are several approaches and tools for analyzing trajectories, ranging from data specific methods [MBP*15, ROBFG18] to software tools designed for general SPT analysis [QSE91, PLUE13, TPS*17, VVOW*17, LP18, HWG*18, BPS*06, RBZ06]. Broadly, these tools integrate segmentation and tracking methods into a complete SPT analysis package. Some of these tools also allow for manual or semi-automatic tracking for users who choose to not rely on a fully automatic tracking approach [TPS*17, VVOW*17], whereas others rely on an input of pre-computed trajectories [PLUE13, HWG*18, LP18]. Some packages (*e.g.,* SpotOn [HWG*18]) are used to correct trajectories for factors such as motion blurring in the data. Others, like vbSPT [PLUE13] uses Hidden Markov Models to extract distinct diffusion states. Other SPT analysis tools also include components for visualizing data, trajectories, and the resulting diffusion analysis. SMTracker [ROBFG18] uses a combination of panels to view multiple diffusion calculation methods. It also includes a panel that shows the spatial distribution of proteins throughout an entire data set. InferenceMAP [EBDM15] uses a Bayesian inference mapping algorithm to give a three-dimensional landscape view that visualizes spacial cellular dynamics. Diatrack [VVOW*17] is a comprehensive SPT analysis software package that performs segmentation, tracking, and analysis. It provides multiple methods for filtering out trajectories of interest and also gives a 3D view of how trajectories evolve over time. TrackMate [TPS*17] has a view that displays the evolution of trajectories as a time oriented hierarchical graph that has a tree like structure. While these tools are widely used, none of them infer useful properties like cluster size as proposed in this paper.

3.2 Tracking Graphs in Visualization

There also exist a wide variety of more general tools to analyze time-dependent features. In many simulation based applications, features are often defined through isosurfaces [LC87] or interval volumes [FMS95], and more recently through more sophisticated hierarchical equivalents such as Morse-Smale complex [GBPH08, BWP*10], contour trees [CSA03], or merge trees [BKL*11, WKK*15, LWM*17]. In these cases, correlating features across time is done via spatial overlaps [SW98, WCBP12], by interpolating their spatio-temporal evolution [EHMP04, BSS02, JSW03, WBP12], or by matching neighboring features according to their distance and attributes [TPS*17]. The resulting output from this tracking can be quite complex. A third stage of analysis is often added to visualize the correlations between features across time in the form of a *tracking graph* in order to support an intuitive exploration of the entire time series, *e.g.,* the one shown in Figure 1. These visualizations can be static [RPS01, LBM*06, BWT*11] or interactive [WCBP12, LGW*20], and a number of sophisticated approaches have been used to highlight the hierarchical structure [LWM*17, LGW*20] or to allow adaptive thresholds [WKK*15]. A comprehensive review of the corresponding literature is beyond the scope of this paper; we refer the reader to the large collection of papers following the 2016 Visualization Contest [Sci16] on analyzing time dependent particle simulations [GEG*18, LAS*17, SPD*19] as a starting point.

In this paper, we discuss yet another, much-less explored as-

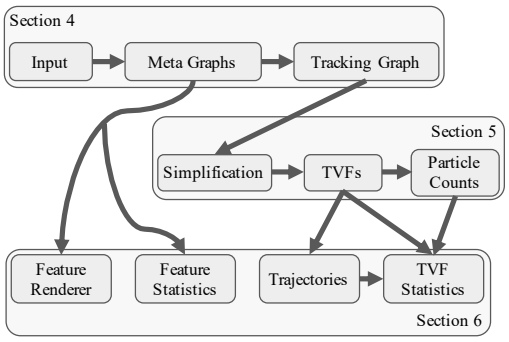


Figure 2: *The dataflow of our analysis pipeline consists of three stages: constructing tracking graphs from user input, analyzing the structure of the graph to produce time-varying features (TVFs) and particle count bounds, and processing these results to produce statistical analysis. The output of each step can be used for other modules in our tool, including a statistical overview, a rendering of the segmented features in a specified timestep, and a trajectory view.*

pect of the problem — analyzing the structure of a tracking graph not only as a visualization challenge, but to provide additional scientific insights. Our approach is independent of both the exact feature detection and feature correlation techniques. Here, we use the current state-of-the-art in microscopy as the most-accepted technique in the relevant application area. Despite the popularity of tracking graphs for visualization and data exploration, we are not aware of techniques that analyze its structure beyond simple time-dependent metrics. For example, many approaches extract the number, size, or location of features over time from the graph [LBM*06, BKL*11, LAS*17], but this information could also be computed independently from a tracking graph. Widanagamaachchi *et al.* [WKK*15] use the structure of the graph to adapt the segmentation to simplify the graph over time, but do not exploit the results beyond providing a less cluttered visualization. Instead, we present the first approach that analyzes the graph globally to infer new information — the expected number of KRAS in each feature, which cannot be obtained otherwise.

4 Computation of Dynamic Tracking Graphs

Our SPT analysis method is driven by a *dynamic tracking graph*, which is a tracking graph that interactively updates at user-defined thresholds for tracking. Once constructed, the structure of this graph facilitates the analysis of features' evolution over time. To construct tracking graphs, we use a procedure similar to that of the state-of-the-art [WCBP12] in our preproccesing and tracking graph construction steps. Our method is tailored to SPT analysis, and differs in the type of input that can be used. Figure 2 outlines our data flow and analysis pipeline. It consists of four main stages: data preprocessing (Section 4.1), construction of the tracking graph (Section 4.2), feature representation across time and particle counting (Section 5), and interactive exploration of the resulting time-varying features (Section 6).

4.1 Preprocessing of Input Data

SPT data is represented as images with "bright spots" at the locations where the equipment detects the presence of labeled particles (see Section 2.2 and Figure 1). Our analysis approach is independent of the exact details of the segmentation and tracking steps, and can take as input any user-defined segmentation and tracking: segmentation input requires IDs and coordinates of features for each frame, along with any associated attributes (*i.e.,* intensity); and tracking information must specify features associated with each track as well as a *correlation weight* assigned to each correlation.

If a tracking input isn't provided, our interactive tool utilizes the built-in distance-based tracking method from TALASS [WCBP12] due to its advanced capabilities for topological analysis. In particular, a user-defined *correlation threshold* is used to define correlations between features in adjacent frames (*i.e.,* the radius or overlap two features must be within to be considered correlated).

Once the segmentation and tracking have been assembled for a data set, the next step is to construct *meta graphs*. A meta graph encapsulates all possible tracking graphs for any user specified feature-correlation threshold, along with attributes for all features. The nodes of the meta graph encode a feature and its associated attributes, and its edges encode the correlations between features and their associated weights. Internally, meta graphs are represented as 3 sets of arrays, which store correlation types, correlation weights, and attributes. Each feature contains a pointer to a location in each of these arrays. The meta graph is output as a file for every time step. The correlation and correlation weight arrays correspond to features in the subsequent time step. Using this representation, the dynamic tracking graph can be quickly updated by checking if the correlation weight for a specific edge is above the user-specified threshold and, thus, preventing any recomputation for tracking.

4.2 Construction of Tracking Graphs

Tracking graphs are constructed given the meta graphs, a focus time step t and a time range r, and user-defined parameters correlation type and correlation threshold p. To provide interactivity, computational optimizations are made. The tracking graph construction process is started by loading all meta graphs into a global least-recently-used (LRU) cache, which is accessible to all processes in the application. Using an LRU cache reduces data I/O cost by storing data in memory for when a user is analyzing specific portions of the tracking graph. Upon request, the computation of the tracking graph is assigned a worker thread. Edges with correlation weights greater than p are added to the graph, starting with t, expanding outward in both directions at step k by alternating the computation from $t+k$ to $t+k+1$ and $t-k$ to $t-k-1$. After every $k=50$ iterations the tracking graph is served to the tracking graph view (Section 6.1) to facilitate a quick visualization of the tracking graph without having to wait for an entire tracking graph to be processed. The user can use this visualization as an initial indicator of whether the feature-correlation threshold is appropriate.

5 SPT Analysis using Tracking Graphs

In this section, we describe the method of using tracking graphs in context of SPT analysis. There are three critical components to this method: simplifying the graph structure to remove spurious correlations (Section 5.1), constructing a representation of individual features as they exist over time (Section 5.2), and creating an inference on the lower bound of number of particles in a cluster (Section 5.3).

5.1 Simplification of Tracking Graphs

Computationally, merge and split events are observed through the tracking graph structure. If a feature has more than one incoming or outgoing edges, it is marked as a merge or split event, respectively. However, this does not necessarily indicate that an actual merge or split event is occurring. This is because the edges in the tracking graph represent *correlations* between features. Depending on how these correlations are defined, certain edges may represent a *glancing* event. We define a *glancing* event as an occurrence where a feature, $\mathbf{x}$, may have more than one incoming/outgoing edge in the tracking graph at time t, but one or more of these edges are created due to another feature, $\mathbf{y}$, at $t-1$ or $t+1$ being within the feature-correlation threshold of the $\mathbf{x}$ at t and not a true link to the current feature. We observe that these glancing events have a specific structure in the tracking graph, where the correlated feature, $\mathbf{y}$, also has multiple edges going in the opposite direction, forming a *"Z-like"* structure. Figure 3 illustrates glancing merge and split events in the tracking graph structure and a corresponding intuition in the physical space.

We note that this structure may also represent a cluster of particles passing one particle to another cluster of particles. From a biological perspective, it is unclear whether this dynamic happens. In our analysis of 5000 frames of simulated data (Section 7.1), we found only 0.2% of edges removed from the graph exhibited this behavior, which leads us to believe that removing these edges is far more advantageous in terms of inferring accurate bounds.

The tracking graph is simplified by removing these glancing edges. If both edges in a merge or split event exhibit a glancing structure, the most correlated edge is kept as dictated by a local optimization function, which is defined as

$$C_{ij}(w,I) = \begin{cases} W_{ij}^2 \cdot \rho_{ij} & \text{for } \rho_{ij} > 1 \\ W_{ij}^2 \,/\, \rho_{ij} & \text{for } \rho_{ij} < 1 \end{cases}, \qquad (1)$$

where W_{ij} is the distance and $\rho_{ij} = \frac{I_i}{I_j}$ is the ratio of intensities between the two features i and j. This function is inspired by the local optimization function for tracking isotropic random motion used by Jaqaman *et al.* [JLM*08] — a tracking method that is considered to be state-of-the-art in the application domain. The original formulation defines ρ_{ij} as the ratio of I_j to the sum of intensity values for *all* incoming features in order to *classify* merge and split events. Instead, our modified formulation compares the ratios of each individual incoming feature to determine the most likely correlation.

5.2 Computation of Time-Varying Features

Once the entire tracking graph has been computed for a user-defined feature-correlation threshold and time range, the next step in the computational pipeline is to compute *time-varying features* (TVFs), which represent individual features as they exist over time. We refer to *features* as the representation of the segmentation at each individual time step, and TVFs as a set of features that are linked together across time. TVFs are defined by a set of properties. These include a birth and death time, which represent the time steps at which the TVF appears and disappears; correlated features, which represent all associated individual features for the TVF for every frame in the TVF's lifetime; and links to other TVFs that are

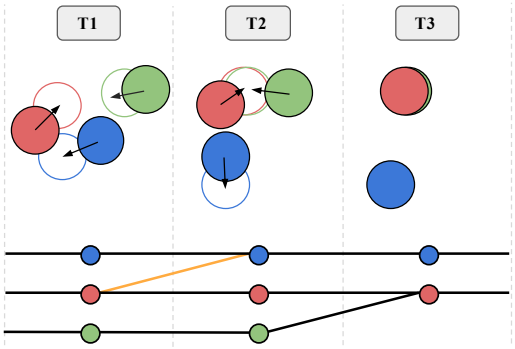


Figure 3: *Edges that show merging and splitting in the tracking graph may not represent true merge and split events, but rather glancing events, illustrated in this figure. Each filled circle represents a feature and its correlation radius. The unfilled circles represent the feature's position in the subsequent frame, and the overlap between these are represented by the edges in the graph. The orange edge represents the glancing event and will be removed by our simplification process.*

connected to the current TVF via merge and split events. TVFs are related to *trajectories*, however we separate the two with the notion that trajectories only contain the set of physical coordinates of each feature in a TVF. TVFs are a critical component in our analysis approach. After TVFs have been computed, they can be used for a variety of downstream analysis tasks, *e.g.,* creating trajectories and for creating statistical properties that describe how TVFs evolve over time.

The TVF computation starts with an input from the tracking graph, effectively a set of linked lists that represents the correlations between features at consecutive time steps. To construct TVFs from the tracking graph, we iterate through all features in consecutive time steps. There exist two scenarios where a new TVF is created: (1) if the current feature of interest has no backward edges or (2) exists directly after a split or merge event. Similarly, a TVF dies when either the current feature has no forward edges or is the node at which a merge or split event occurs. For each TVF that dies at a merge or split event, a link is maintained from each of the TVFs before the event to the TVFs following the event. For each of these events, the *most correlated* link is marked. These links are used for two purposes: the most correlated link is used for propagating counts (Section 5.3), and all links enable analyzing the change in diffusion before and after merge and split events.

5.3 Determination of Lower Bound on Particle Count

Although the ability to observe and label merge and split events of particles is a critical component of SPT analysis, knowing *how many* particles are involved in these events is important to biologists studying SPT data. Using the methods for data acquisition described in Section 2.2, it is impossible to determine the exact number of particles in a cluster of proteins due to the presence of a "dark" (unlabeled) population. Recent work [PMPH05] estimates the upper bound on the number of particles that may exist in a cluster to be 6 to 8. However, it is possible to determine a *lower bound* on the particle count of a feature by analyzing the tracking graph, in conjunction with the computation of TVFs.

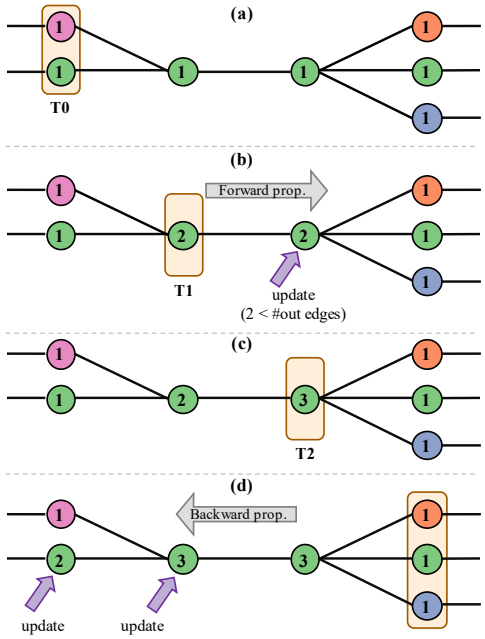


Figure 4: *Our counting method utilizes edges in the tracking graph to infer a lower bound on the count of particles within a cluster. From top to bottom: (a) Every feature initially receives a count of 1. (b) For merge events, we add the counts of features with incoming edges, and propagate the count forward. (c) For split events, the count is distributed among the outgoing edges. (d) If there are too many outgoing edges to account for the current feature's count, the count is propagated backward along the TVF and adjust accordingly. The most-correlated features for merge/split events are marked by an edge between nodes of the same color.*

Our approach for computing these bounds is straightforward; Figure 4 gives a visual representation of the process. Each feature at the first time step receives a count of one. If a merge event is encountered, the counts of the incoming features are added together (*e.g.,* in Figure 4(b), the green feature gets a count of two by adding in the count from the pink feature). These counts are then passed along to the subsequent features (highlighted by the arrow). If a split event is encountered, the count of the feature before the split is distributed among the features after the split (*e.g.,* in Figure 4(c) the red, green, and blue features receive a count of one each, adding up to the observed count of three of the green feature at T3). This distribution of counts is done according to the local optimization function (Eqn. 1). In general, if a split event has k outgoing edges, with an inferred count j before the event, and $k < j$, then we assign the the highest-correlated feature after the event a count of $j-k+1$ and the rest of the features a count of one (*e.g.,* if the green feature at time step T2 in Figure 4(d) had a count of four, the green feature would receive a count of two, and the red and blue features would receive a count of one each).

After the construction of TVFs and computation of the lower bound of the number of particles in each feature, we have an enhanced representation of a feature's existence over time. TVFs can be broken into sets corresponding to the number of particles the feature contains, and they can also be processed using their links to other TVFs to analyze how particle diffusion changes before and after merge and split events. In this manner, our approach enables analyzing how specific protein aggregation and splitting events affect the motion of these proteins.

5.4 Filtering of TVFs

The ability to filter the tracking graph is a vital component in our analysis pipeline. It serves two purposes: creating statistically meaningful analysis, and visualizing targeted data and TVFs. The user can filter the TVFs in three ways: by the lifetime of the TVF, by the derived particle count, or by a statistical aggregation of individual feature attribute values within the TVF. For visualizing the filtered data, the user can select either removing all TVFs from the tracking graph if they are below the user specified filtering threshold or alternatively highlighting the features that exist above the specified threshold.

6 Interactive Exploration of Time-Varying Features

Analysis of SPT data can often be an iterative process. Due to the low resolution and sometimes ambiguous nature of such data, meaningful statistical analysis often requires manual adjustment of certain parameters, *e.g.,* correlation criteria, filtering out trajectories of a certain length, and identifying trajectories of interest. To facilitate this process, it is helpful to have an interactive visualization of the results and a quick access to statistics describing the results. In this section, we present an interactive exploration tool for time-varying features that was developed with the specific focus to the needs of SPT analysis (see Figure 5 for an overview). Our tool serves three purposes: (1) it enables visualization and statistical analysis of a given data set — a task critical to users for validating experiments and gaining an understanding of the properties of the features in the data set (*e.g.,* intensity variations); (2) it facilitates filtering of TVFs and the associated visualization and statistical analysis; and (3) it incorporates the analysis techniques presented in Section 5. Furthermore, it is immensely helpful to visualize the features and trajectories in a spatial context for exploring the data set; therefore, our tool additionally includes a feature rendering view and trajectory view, which are both also included in standard tools for SPT analysis.

To facilitate this exploratory analysis, our visualization framework comprises several inter-linked visualizations. Different views can be enabled/disabled on demand as separate windows that are constructed using the Qt framework [Qt] and built with the OpenVisus [PLF*03, SCI] API using C++. At the center of the framework lies the primary view, the tracking graph visualization, which spawns new worker threads for computations and additional views when needed, as well as interacts with the primary UI to control options such as filtering.

6.1 Tracking Graph Visualization

Visual exploration of tracking graphs demands the visualization of all features and their correlations between time steps. Visualization of the structure of the tracking graph can provide immediate insights into the data, exposing the complexity and frequency of particle interactions. Figure 5 and Figure 6 give a comparison between data sets with lipid bilayer formulation that either induce merging and splitting versus one that does not.

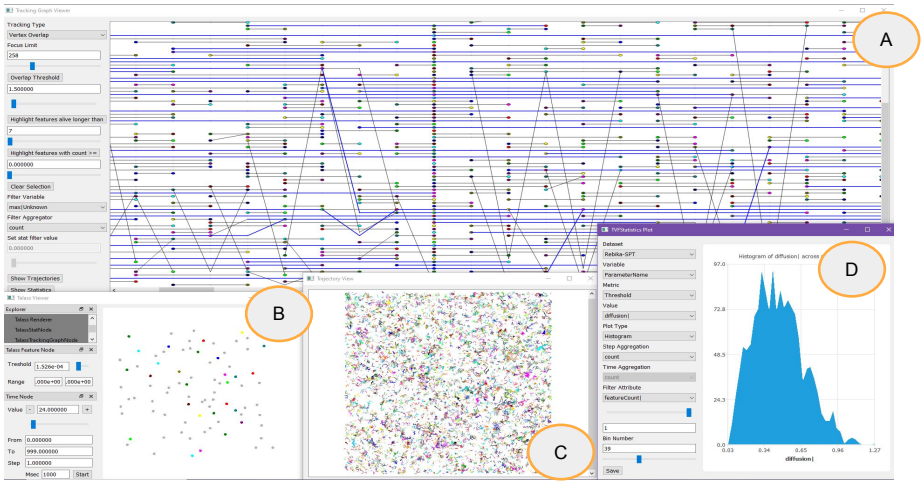


Figure 5: *Our interactive analysis tool contains multiple inter-linked windows for visualizing, interacting with, and analyzing the data. (A) shows the tracking graph, which contains a panel for adjusting the feature-correlation threshold and filtering parameters. The feature view (B) and the trajectory view (C) shows the rendering of all the features for the specified time step and the trajectories for each TVF, in physical space. The statistics view (D) can be used for producing statistics for feature and TVF attributes, including diffusion distributions.*

The main window in our tool focuses on an interactive visualization of the tracking graph with associated user interface panel for adjusting correlation parameters and filtering. The two correlation parameters (as mentioned in Section 4.2) include the tracking type and correlation weight. The use of these parameters is largely dependent on the type of tracking input to the tool. The tracking type parameter is used to define whether the built in tracking method or user defined tracking is used. The correlation weight parameter is used to define the degree of correlation (*i.e.,* distance that is considered for linking features together) — a higher threshold will decrease the distance that is considered to define correlations. It is expected that users have prior knowledge of an estimation for this correlation from an expected range of particle diffusion. For the distance based tracking method used in this work, the results are sensitive to using a high correlation threshold since correlations between fast moving particles will be lost. Decreasing this correlation threshold will have a low impact on the lower bound results, since the spurious correlations that result from a lower threshold will be removed through the graph simplification process described in Section 5.1. The tracking graph view is linked to several other view panels (discussed ahead) so that any user interactions with filtering options cascade through the system, and associated visualizations are updated correspondingly. The colors of the nodes in the tracking graph correspond to the color of the individual features in the feature renderer to aid data exploration.

Finding an optimal layout for minimizing edge crossings is an NP-complete problem [GJ83]. To reduce edge crossings in the graph, we use a *median heuristic* approach, similar to the one used in [WCBP12] to optimize the layout. For the first time step, each node corresponding to a living feature is assigned a position according to its threshold hierarchy. For each subsequent time step, nodes are assigned to bins according to the median of all of the positions of its connected nodes in the previous time step. In practice, we find this approach to be sufficient for avoiding significant edge crossings. In addition, we minimize clutter by removing any nodes from the graph that do not have any incoming or outgoing edges.

6.2 Statistical Visualization of Feature Attributes

To facilitate instantaneous insights from the tracking graph, we provide a statistical visualization window that focuses on the the evolution of feature attributes with time. A statistical overview of feature attributes is important for biologists to gain a broad understanding of a data set, like the number of features, variations in intensity, *etc.* Understanding how these attributes change over time can help the user determine filtering thresholds. For example, a segmentation method may pick up features with low intensity values. The user can use the statistics viewer to identify these values and, if deemed to be an outlier, can filter TVFs below this threshold.

Currently, there are six types of statistics that can be computed instantly on demand: histograms, probability- and cumulative-density functions (PDFs and CDFs), weighted PDFs and CDFs, and time plots. Various parameters of these statistics can be controlled through the UI — examples of these controllable parameters are *the attribute type*, which corresponds to every feature (*e.g.,* intensity) and *aggregation mode* (*e.g.,* maximum, minimum, sum, and mean), which corresponds to how the statistics are accumulated within a feature (*i.e.,* across pixels).

Whereas the distribution plots consider all features across all time steps, time plots require additional specifications for how values are further aggregated for each time step. Since the time plots abstract the distribution of feature attributes per frame, rather than a distribution of all of the feature attributes present over all time steps, a *time aggregation mode* is also provided to the user, which controls how the attributes of all of the features in a single frame are aggregated (*e.g.,* maximum, minimum, sum, and mean). Such aggregated values present a high-level descriptor of how feature attributes evolve over time, and are useful to identify parts of the temporal data that may exhibit irregularities.

6.3 Statistical Visualization of Time-Varying Features

Focusing specifically on SPT analysis, we provide additional statistical visualization that capture the properties of time-varying features, such as particle counts and diffusion. Similar to Section 6.2, several interactive options are provided via the UI, including the additional *diffusion distribution* and *particle counts plot*.

The input to this functionality is a set of TVFs and their corresponding *trajectories*. For every TVF, its *trajectory* represents a set of the physical coordinates of each feature per frame in its lifetime and can be used to compute the *diffusion* using the Mean Square Displacement (MSD) as

$$\mathrm{MSD}(n\,\Delta t) = \frac{\sum_{i=1}^{N-n}(x_{i+n}-x_i)^2+(y_{i+n}-y_i)^2}{(N-n)}$$

where n is the step size used for the computation and $(N-n)$ is the number of frames that the MSD can be computed for.

There are multiple ways for analyzing the diffusion. The first is a distribution for the diffusion of all TVFs — this uses the trajectories and corresponding diffusion values for the entire lifetime of a TVF. The second way is to perform a similar analysis, but after filtering the trajectories by the number of particles within a trajectory segment. This filtering is performed by iterating over all TVFs and extracting the *segments* that contain the number of particles specified by the user. Furthermore, the user may also see how this distribution changes with respect to the particle counts by aggregating the diffusion for each count present in the data and displaying the aggregated values in a particle count plot, which shows these aggregated values against the particle counts.

Finally, the user may create distributions to analyze how diffusion change before/after merge or split events. Our data structures make this computation straightforward: we use the linked trajectories and the associated diffusion before the merge/split events and compute the differences. After aggregating these values, the distributional shifts indicate slower or faster (left or right shifts, respectively) with respect to the event.

6.4 Computational Cost and Scalability

The construction of TVFs and the computation of the lower bound counts is very scalable. We performed a scaling study for the Localizer and PRIS segmentations of the data sets (Section 7.3) on a laptop computer with an i7–7700HQ CPU. For the PRIS segmentation (average of 83 features per frame), the computation took 0.17, 0.34, 0.52, 0.71, and 0.83 seconds for 1000, 2000, 3000, 4000, and 5000 time steps, respectively. For the Localizer segmentation (average of 60 features per frame), the computation took 0.14, 0.25, 0.45, 0.63, and 0.74 seconds for the same time step progression. The computation time for the tracking graph construction and layout was 1.14, 2.94, 5.18, 10.18, and 13.76 seconds (PRIS) and 1.12, 2.85, 8.62, 11.28, 13.8 seconds (Localizer).

7 Results

To demonstrate our analysis technique, we have applied it to two different experimental conditions aimed at understanding the dependency of KRAS diffusion on the lipid composition of the supporting membrane. The key contribution of our approach is the ability to estimate the number of labeled KRAS within each observed cluster, which provides more-direct evidence of the link between observed average diffusion and the size of clusters.

7.1 Simulation of KRAS-Membrane Interactions

The experiments that are the focus of this paper are part of a larger collaboration between computational and experimental biologists as well as computer scientists and others, aimed at building a computational framework able to simulate KRAS on a lipid membrane for experimentally relevant time- and length-scales. A complete description is beyond the scope of this paper, and we refer the reader to [DNBC*19] and [INC*20] for more information. In particular, the computational campaign [DNBC*19] includes a continuum scale simulation of 300 KRAS proteins interacting with a $1\times1\ \mu m^2$ membrane of eight different lipid types designed to approximate an average cell membrane. Based on this simulation, the team identified a membrane composition, *i.e.,* specific proportions between lipid types, conducive to KRAS clustering [INC*20], which has subsequently been replicated *in-vitro*. In the following discussion, this membrane and the corresponding experimental data is referred to as the *8-lipid membrane*.

This collaborative computational-experimental campaign makes the continuum simulation a good candidate for a validation study as it provides ground truth tracking information for all 300 constituent KRAS proteins. However, it is important to recognize that although the simulation and experiment were matched as best as possible, one would still expect significant discrepancies. For example, the time-scales remain incomparable with the experiments covering a significantly longer time span than what can be simulated. Furthermore, the membrane composition cannot be replicated perfectly *in-vitro*, and only about 5% of KRAS are labeled. Finally, in the experiments, KRAS can attach and detach from the membrane — an effect not part of the simulation. Nevertheless, the overall motion of particles is expected to be similar.

To validate our approach in estimating cluster sizes, we assume a perfect segmentation (*i.e.,* we use the true spatial coordinates for each simulated KRAS) and apply the algorithm described in Section 5 to 5000 frames of simulated data. To account for effects of unlabeled KRAS, we track only 100 randomly-selected (of the 300 simulated) proteins. Although this is still a significantly larger fraction of labeled KRAS than in the experimental conditions, tracking 5% KRAS would result in too few events to be statistically meaningful, given the overall low number of KRAS and the comparatively short time period of the simulation. The estimated counts are very accurate. For clusters with cluster sizes 1, 2, 3, and 4, the percentage of incorrect counts were 0.001%, 0.8%, 0.01%, and 0.09%, respectively. A more detailed analysis shows that the majority of errors are due to glancing clusters exchanging particles (see Section 5) as the corresponding edges are erroneously removed during the simplification step. However, this has little impact as in all 5000 frames only 0.2% of all removed edges are due to glancing events and should be kept, whereas the other 99.8% are spurious edges caused by tracking artifacts that should be removed. Consequently, since one cannot distinguish (except in simulated data) these two cases, we remove all such edges from the tracking graphs of the experimental data. In summary, our validation demonstrates that the algorithm is accurate and correctly estimates the number of labeled KRAS in a cluster.

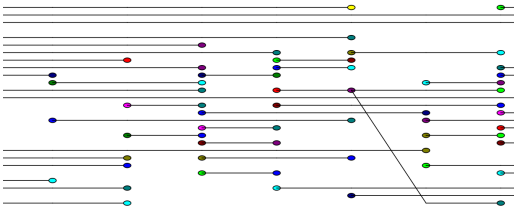


Figure 6: *The tracking graph for the 2-lipid membrane, using the Localizer segmentation. Very few merge and split events exist, in comparison to the 8-lipid membrane's tracking graph (Figure 5).*

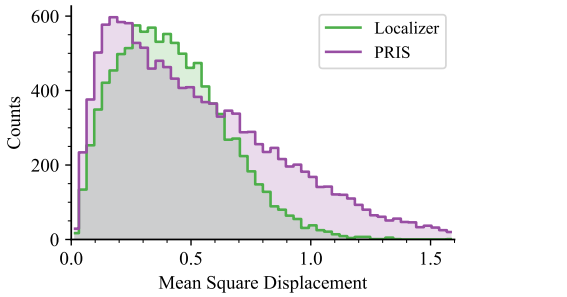


Figure 7: *Histograms of MSD for the trajectories computed with the Localizer and the PRIS segmentation techniques. Such distributions are normally extracted by current SPT analysis techniques to be further analysed to extract diffusion states.*

7.2 Experiment with 2-Lipid Membrane

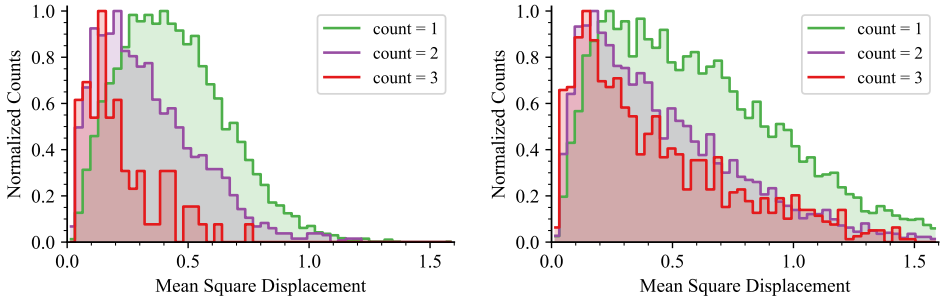
Similar to the simulation data used for validating the algorithms, our collaborators use a much-simpler membrane consisting of only two lipid types as a baseline and comparison. This composition is lacking some of the highly charged lipids of the 8-lipid membrane that assemble beneath the (oppositely charged) KRAS and are known to promote clustering. Consequently, we expect much fewer merge and split events, which is confirmed by the resulting tracking graph in Figure 6. Although the figure shows only a comparatively small number of frames, even the entire 5000 frame sequence contains only 72 topological events compared to the 19612 merge and split events in the equivalent 8-lipid membrane data discussed in Section 7.3. This data was processed using Localizer [DDNZ12] segmentation, using the parameters discussed in Section 2.2. We note that for both this experiment and the one in Section 7.3, each data set had previously been processed and analyzed by our collaborators, which gives insights into how the parameters and filtering should be performed. We used the TALASS distance based tracking approach using a correlation threshold of 1.5 pixels. The threshold is determined by previously estimated diffusion coefficients and is the maximal distance any KRAS is expected to travel in between two frames. Furthermore, as discussed above, the graph is simplified by removing edges that represent glancing events or the exchange of KRAS between clusters. Finally, we follow the established experimental protocols of our collaborators and remove tracks shorter than six frames as they do not provide a reliable estimate of the MSD. Although the lack of merge and split events makes the 2-lipid membrane uninteresting for the cluster estimation presented here, the resulting plain tracking graph has provided a quick and intuitive way for our collaborators to validate their intuition.

7.3 Experiment with 8-Lipid Membrane

The primary dataset of interest is an 8-lipid membrane with focus on understanding how much clustering it induces. The current state-of-the-art is to compute the tracking graph as we are doing here, but subsequently splitting all tracks at merges and splits to generate a set of independent, simple trajectories. Given these trajectories, one then measures the MSD of each track as a measure of diffusion of the corresponding protein(s). Slower diffusion is attributed to clustering, based on the expectation that a cluster of multiple proteins has a larger total mass and a larger footprint of charged lipids underneath, both leading to greater confinement. However, as one cannot reliably determine the number of labeled proteins within a cluster and there exist a large fraction of unlabeled KRAS the connection between diffusion and clustering remains a conjecture, albeit a well-accepted and logical one.

Here, we use 5000 frames of the 8-lipid membrane data to demonstrate how the tracking graph analysis introduced in Section 5 provides a more-direct link between MSD and clustering than previously reported. In particular, we present results from two different segmentation approaches: Localizer [DDNZ12] (the technique used in the original publication [INC*20]) and PRIS [YPW19] (a recently developed method based on progressively refining compressive sensing reconstructions). We maintain the same parameters for the Localizer segmentation, tracking method, and filtering operations as the 2-lipid membrane. For the PRIS segmentation, a Gaussian PSF model is used in the reconstruction with an estimated full width at half maximum (FWHM) of 320.25 nm ($\sigma = 0.85$ pixel width) based on empirical inspection of the PSF image compared to the imaging results of fluorescently labeled single KRAS molecules under the same microscope. In future applications of the proposed tool, PSF calibrations acquired alongside with the data acquisition could be used to ensure minimum model error introduced by instrumentation variability, and provide a more-precise estimation of the PSF model to achieve optimum performance [LMH*18]. PRIS is optimized for higher particle densities and is better able to separate nearby particles. Given the limited resolution, changes in brightness, particles moving into and out of the focal plane of microscope, *etc.*, segmentation is typically considered the main source of uncertainty in SPT approaches and, thus, using two very different segmentation techniques provides another chance for validating the results. The Localizer segmentation typically identifies around 100 clusters per frame, while the PRIS segmentation picks up a significantly higher number of particles, with roughly 175 particles per frame. Note that both techniques rely on a number of internal parameters adjusted by the relevant experts, but neither should be considered ground truth.

Figure 7 shows the average MSD for individual tracks, *i.e.,* the tracks separated at merge and split events, which represents the information that traditionally would have been extracted by our collaborators. Both segmentation techniques show a similar distribution of MSD, with PRIS resulting in a slightly tighter main mode but a heavier tail of fast particles. In particular, we do not see any reliable indication of multiple modes in this distribution that could indicate distinct speeds for single KRAS, KRAS dimers, trimers, *etc.* Instead, lipid compositions would be compared in their entirety with overall lower speeds attributed to an increased number



Mean MSD

#particles	Localizer	PRIS
1	0.441612	0.451615
2	0.332365	0.341706
3	0.218165	0.330200

Figure 8: *Normalized distributions (max = 1) of MSD partitioned by the lower-bound count of a TVF, using the Localizer (left) and PRIS (middle) segmentation, and the mean MSD values for the corresponding data (right) are shown. As the inferred lower bound of the count increases, the distributions skew towards a slower MSD.*

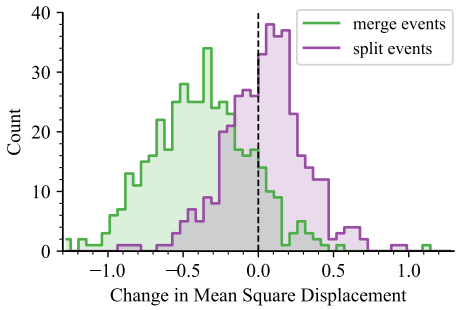


Figure 9: *The distributions of the change (after − before) in MSD for merge and split events identified for PRIS segmentation. Left and right skew correspond to slowing down and speeding up after the event, respectively. Our analysis confirms that larger clusters (after merge and before split) correlates with slower diffusion.*

of clusters. Instead, by processing the merge and split events, our technique provides an estimate for the lower bound of the number of (labeled) KRAS in each cluster, which allows us to separate the global distribution of tracks according to their size. Figure 8 shows the average MSD per estimated cluster size for the two segmentation techniques with both showing a clear trend of slower speeds for larger clusters. Since the distributions are noticeably skewed, we also provide the respective conditional histograms for clusters of sizes 1 through 3 (see Figure 8). In particular, the Localizer-based approach shows a clear separation between distributions with clusters estimated to contain more KRAS being slower than those with fewer KRAS. Note that all distributions are normalized as there are significantly more individual particles (count = 1) than clusters (count = 2, 3). Alternatively, one can aggregate the data by comparing the MSD before to the MSD after a merge/split. Figure 9 show the respective changes in MSD before and after and event. As expected, merge events are strongly correlated with a decrease in speed and split events with marked increases in speed.

8 Discussion

Our analysis provides the first direct evidence that MSD is correlated to the forming or breaking up of clusters. Going forward, the modes of the MSD distribution may provide crucial information on the actual differences in MSD between clusters of different size which would not only provide important insight into KRAS-membrane interactions in general, but would also inform the simulation model. Despite the success in demonstrating the link between diffusion and clustering, it is important to be cognizant of potential shortfalls. First, as presented here our approach is heavily dependent on the quality of both the segmentation and tracking graph construction and, thus, failures in either step can skew the results. Our method is valid for the distance based tracking method, and can be extended for other tracking methods that consider the correlations between atomic features. Our method may be invalid for segmentation techniques that distinguish individual particles within a cluster, however it could potentially be used to validate these methods. One potential future direction is to better understand the sensitivity to changes in preprocessing and, in particular, the impact of noise on the segmentation step. Furthermore, given the limited temporal resolution, unlabeled proteins, and unobservable events, such as a KRAS leaving the membrane, there will always exist uncertainties in the counts. For example, it is conceivable – even if unlikely – that each labeled KRAS is always paired with a second (or multiple) unlabeled KRAS. Although this would make our lower bound incorrect, note that the analysis of diffusion before/after an event would still point to the same conclusion. Finally, the current graph simplification is valid since our observed system, as represented in the simulation, appears to have almost no clusters exchanging particles. In a different application, these events might be more common, which might lead to more errors in removing all such edges from the tracking graph. As part of ongoing work, we are investigating the potential to use the global particle count to disambiguate tracking artifacts from a particle exchange. As counts get propagated through the entire graph, assuming a particle exchange and, thus, retaining the corresponding edge, increases the total particle count that is inferred. If bounds on the expected counts are known, these could provide an indication on how many such edges should be removed. We are currently working with our collaborators on applications to new experiments.

Acknowledgements

This work has been supported in part by the Joint Design of Advanced Computing Solutions for Cancer (JDACS4C) program established by the U.S. Department of Energy (DOE) and the National Cancer Institute (NCI) of the National Institutes of Health (NIH). For computing time, we thank Livermore Computing (LC) and Livermore Institutional Grand Challenge. This work was performed under the auspices of the U.S. DOE by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344 and The Frederick National Laboratary for Cancer Research under Contract HHSN261200800001E. Release LLNL-JRNL-817310.

References

[AZG06] ANTHONY S., ZHANG L., GRANICK S.: Methods to track single-molecule trajectories. *Langmuir 22*, 12 (06 2006), 5266–5272. 1

[BKL*11] BENNETT J., KRISHNAMURTHY V., LIU S., PASCUCCI V., GROUT R., CHEN J., BREMER P.-T.: Feature-based statistical analysis of combustion simulation data. *IEEE Trans. Vis. Comp. Graph. 17*, 12 (2011), 1822–1831. 3, 4

[BPS*06] BETZIG E., PATTERSON G. H., SOUGRAT R., LINDWASSER O. W., OLENYCH S., BONIFACINO J. S., DAVIDSON M. W., LIPPINCOTT-SCHWARTZ J., HESS H. F.: Imaging intracellular fluorescent proteins at nanometer resolution. *Science 313*, 5793 (2006), 1642–1645. 3

[BSS02] BAJAJ C., SHAMIR A., SOHN B.-S.: Progressive tracking of isosurfaces in time-varying scalar fields, 2002. 3

[BWP*10] BREMER P.-T., WEBER G., PASCUCCI V., DAY M., BELL J.: Analyzing and tracking burning structures in lean premixed hydrogen flames. *IEEE Trans. Vis. Comp. Graph. 16*, 2 (2010), 248–260. 3

[BWT*11] BREMER P.-T., WEBER G., TIERNY J., PASCUCCI V., DAY M., BELL J. B.: Interactive exploration and analysis of large scale simulations using topology-based data segmentation. *IEEE Trans. Vis. Comp. Graph. 17*, 9 (2011), 1307–1324. 3

[CSA03] CARR H., SNOEYINK J., AXEN U.: Computing contour trees in all dimensions. *Comput. Geom. Theory Appl. 24*, 3 (2003), 75–94. 3

[DDNZ12] DEDECKER P., DUWÉ S., NEELY R. K., ZHANG J.: Localizer: fast, accurate, open-source, and modular software package for superresolution microscopy. *Journal of biomedical optics 17*, 12 (2012), 126008. 1, 2, 3, 9

[DNBC*19] DI NATALE F., BHATIA H., CARPENTER T., NEALE C., SCHUMACHER S., OPPELSTRUP T., STANTON L., ZHANG X., SUNDRAM S., SCOGLAND T., DHARUMAN G., SURH M., YANG Y., MISALE C., SCHNEIDENBACH L., COSTA C., KIM C., D'AMORA B., GNANAKARAN S., NISSLEY D., STREITZ F., LIGHTSTONE F., BREMER P.-T., GLOSLI J., INGÓLFSSON H.: A massively parallel infrastructure for adaptive multiscale simulations: Modeling RAS initiation pathway for cancer. In *Proc. ACM/IEEE Int. Conf. for High Performance Computing, Networking, Storage, and Analysis on Supercomputing (SC)* (2019), no. 57. 2, 8

[EBDM15] EL BEHEIRY M., DAHAN M., MASSON J.-B.: Inferencemap: mapping of single-molecule dynamics with bayesian inference. *Nature methods 12*, 7 (2015), 594–595. 3

[EHMP04] EDELSBRUNNER H., HARER J., MASCARENHAS A., PASCUCCI V.: Time-varying Reeb graphs for continuous space-time data. In *20th Symp. on Computational Geometry* (New York, NY, USA, 2004), ACM Press, pp. 366–372. 3

[FMS95] FUJISHIRO I., MAEDA Y., SATO H.: Interval volume: A solid fitting technique for volumetric data display and analysis. In *Proc. of the 6th Conf. on Visualization* (1995), pp. 151–158. 3

[GBPH08] GYULASSY A., BREMER P.-T., PASCUCCI V., HAMANN B.: A practical approach to Morse-Smale complex computation: Scalability and generality. *IEEE Trans. Vis. Comp. Graph. 14*, 6 (2008), 1619–1626. 3

[GEG*18] GRALKA P., ERTL T., GROTTEL S., STAIB J., SCHATZ K., KARCH G. K., HIRSCHLER M., KRONE M., REINA G., GUMHOLD S.: 2016 ieee scientific visualization contest winner: Visual and structural analysis of point-based simulation ensembles. *IEEE Comp. Graph. Apps. 38*, 03 (May 2018), 106–117. 3

[GJ83] GAREY M. R., JOHNSON D. S.: Crossing number is np-complete. *SIAM Journal on Algebraic Discrete Methods 4*, 3 (1983), 312–316. 7

[HF13] HÖFLING F., FRANOSCH T.: Anomalous transport in the crowded world of biological cells. *Reports on Progress in Physics 76*, 4 (mar 2013), 046602. 1

[HWG*18] HANSEN A. S., WORINGER M., GRIMM J. B., LAVIS L. D., TJIAN R., DARZACQ X.: Robust model-based analysis of single-particle tracking experiments with spot-on. *Elife 7* (2018), e33125. 3

[INC*20] INGÓLFSSON H. I., NEALE C., CARPENTER T. S., SHRESTHA R., LÓPEZ C. A., TRAN T. H., OPPELSTRUP T., BHATIA H., STANTON L. G., ZHANG X., SUNDRAM S., NATALE F. D., AGARWAL A., DHARUMAN G., SCHUMACHER S. I. L. K., TURBYVILLE T., GULTEN G., VAN Q. N., GOSWAMI D., JEAN-FRANCIOS F., AGAMASU C., CHEN D., HETTIGE J. J., TRAVERS T., SARKAR S., SURH M. P., YANG Y., MOODY A., LIU S., GARCÍA A. E., ESSEN B. C. V., VOTER A. F., RAMANATHAN A., HENGARTNER N. W., SIMANSHU D. K., STEPHEN A. G., BREMER P.-T., GNANAKARAN S., GLOSLI J. N., LIGHTSTONE F. C., MCCORMICK F., NISSLEY D. V., STREITZ F. H.: Machine Learning-driven Multiscale Modeling Reveals Lipid-Dependent Dynamics of RAS Signaling Proteins. *Nature Biotechnology* (2020). Under Review. doi:10.21203/rs.3.rs-50842/v1. 2, 8, 9

[JLM*08] JAQAMAN K., LOERKE D., METTLEN M., KUWATA H., GRINSTEIN S., SCHMID S. L., DANUSER G.: Robust single-particle tracking in live-cell time-lapse sequences. *Nature Methods 5*, 8 (2008), 695–702. 2, 5

[JSW03] JI G., SHEN H.-W., WEGNER R.: Volume tracking using higher dimensional isocontouring. In *Proc. IEEE Visualization* (2003), IEEE Computer Society, pp. 209–216. 3

[KGM*19] KESSLER D., GMACHL M., MANTOULIDIS A., MARTIN L. J., ZOEPHEL A., MAYER M., GOLLNER A., COVINI D., FISCHER S., GERSTBERGER T., GMASCHITZ T., GOODWIN C., GREB P., HÄRING D., HELA W., HOFFMANN J., KAROLYI-OEZGUER J., KNESL P., KORNIGG S., KOEGL M., KOUSEK R., LAMARRE L., MOSER F., MUNICO-MARTINEZ S., PEINSIPP C., PHAN J., RINNENTHAL J., SAI J., SALAMON C., SCHERBANTIN Y., SCHIPANY K., SCHNITZER R., SCHRENK A., SHARPS B., SISZLER G., SUN Q., WATERSON A., WOLKERSTORFER B., ZEEB M., PEARSON M., FESIK S. W., MCCONNELL D. B.: Drugging an undruggable pocket on kras. *Proceedings of the National Academy of Sciences 116*, 32 (2019), 15823–15829. 2

[Kra15] KRAPF D.: Chapter five - mechanisms underlying anomalous diffusion in the plasma membrane. In *Lipid Domains*, Kenworthy A. K., (Ed.), vol. 75 of *Current Topics in Membranes*. Academic Press, 2015, pp. 167 – 207. 1

[LAS*17] LUKASCZYK J., ALDRICH G., STEPTOE M., FAVELIER G., GUEUNET C., TIERNY J., MACIEJEWSKI R., HAMANN B., LEITTE H.: Viscous fingering: A topological visual analytic approach. In *Physical Modeling for Virtual Manufacturing Systems and Processes* (2017), vol. 869 of *Applied Mechanics and Materials*, Trans Tech Publications Ltd, pp. 9–19. 3, 4

[LBM*06] LANEY D., BREMER P.-T., MASCARENHAS A., MILLER P., PASCUCCI V.: Understanding the structure of the turbulent mixing layer in hydrodynamic instabilities. *IEEE Trans. Vis. Comp. Graph. 12*, 5 (2006), 1052–1060. 3, 4

[LC87] LORENSEN W., CLINE H.: Marching cubes: A high resolution 3d surface construction algorithm. *Comp. Graph. 21*, 4 (July 1987), 163–169. 3

[LGW*20] LUKASCZYK J., GARTH C., WEBER G. H., BIEDERT T., MACIEJEWSKI R., LEITTE H.: Dynamic nested tracking graphs. *IEEE Trans. Vis. Comp. Graph. 26*, 1 (2020), 249–258. 3

[LMH*18] LI Y., MUND M., HOESS P., DESCHAMPS J., MATTI U., NIJMEIJER B., SABININA V. J., ELLENBERG J., SCHOEN I., RIES J.: Real-time 3d single-molecule localization using experimental point spread functions. *Nature methods 15*, 5 (2018), 367–369. 9

[LP18] LEE B. H., PARK H. Y.: Hybtrack: A hybrid single particle tracking software using manual and automatic detection of dim signals. *Scientific reports 8*, 1 (2018), 1–7. 3

[LWM*17] LUKASCZYK J., WEBER G., MACIEJEWSKI R., GARTH C., LEITTE H.: Nested tracking graphs. *Comp. Graph. Forum 36*, 3 (2017), 12–22. 3

[MBP*15] MONNIER N., BARRY Z., PARK H. Y., SU K.-C., KATZ Z., ENGLISH B. P., DEY A., PAN K., CHEESEMAN I. M., SINGER R. H., ET AL.: Inferring transient particle transport dynamics in live cells. *Nature Methods 12*, 9 (2015), 838–840. 3

[MGP15] MANZO C., GARCIA-PARAJO M. F.: A review of progress in single particle tracking: from methods to biophysical insights. *Rep Prog Phys 78*, 12 (Dec 2015), 124601. 1

[MRS*18] MIR M., REIMER A., STADLER M., TANGARA A., HANSEN A. S., HOCKEMEYER D., EISEN M. B., GARCIA H., DARZACQ X.: Single molecule imaging in live embryos using lattice light-sheet microscopy. *Methods Mol Biol 1814* (2018), 541–559. 1

[NCI] NCI: NATIONAL CANCER INSTITUTE:. Ras initiative [online]. URL: https://www.cancer.gov/research/key-initiatives/ras. 2

[PHH20] PRIOR I. A., HOOD F. E., HARTLEY J. L.: The frequency of ras mutations in cancer. *Cancer Research 80*, 14 (2020), 2969–2974. 2

[PLF*03] PASCUCCI V., LANEY D. E., FRANK R., SCORZELLI G., LINSEN L., HAMANN B., GYGI F.: Real-time monitoring of large scientific simulations. In *Proc. 18th Annual ACM Symp. on Applied Computing* (2003), pp. 194–198. 6

[PLUE13] PERSSON F., LINDÉN M., UNOSON C., ELF J.: Extracting intracellular diffusive states and transition rates from single-molecule tracking data. *Nature Methods 10*, 3 (2013), 265–269. 3

[PMPH03] PRIOR I. A., MUNCKE C., PARTON R. G., HANCOCK J. F.: Direct visualization of ras proteins in spatially distinct cell surface microdomains. *The Journal of cell biology 160*, 2 (2003), 165–170. 3

[PMPH05] PLOWMAN S. J., MUNCKE C., PARTON R. G., HANCOCK J. F.: H-ras, k-ras, and inner plasma membrane raft proteins operate in nanoclusters with differential dependence on the actin cytoskeleton. *Proceedings of the National Academy of Sciences 102*, 43 (2005), 15500–15505. 5

[QSE91] QIAN H., SHEETZ M. P., ELSON E. L.: Single particle tracking. analysis of diffusion and flow in two-dimensional systems. *Biophysical J. 60*, 4 (1991), 910–921. 3

[Qt] QT:. Qt [online]. URL: https://www.qt.io/. 6

[RBZ06] RUST M. J., BATES M., ZHUANG X.: Sub-diffraction-limit imaging by stochastic optical reconstruction microscopy (storm). *Nature Methods 3*, 10 (2006), 793–796. 3

[ROBFG18] RÖSCH T. C., OVIEDO-BOCANEGRA L. M., FRITZ G., GRAUMANN P. L.: SMTracker: A tool for quantitative analysis, exploration and visualization of single-molecule tracking data reveals highly dynamic binding of b. subtilis global repressor abrb throughout the genome. *Scientific Reports 8*, 1 (2018), 1–12. 3

[RPS01] REINDERS F., POST F. H., SPOELDER H. J. W.: Visualization of time-dependent data with feature tracking and event detection. *The Visual Computer 17*, 1 (2001), 55–71. 3

[SBRM08] SERGÉ A., BERTAUX N., RIGNEAULT H., MARGUET D.: Dynamic multiple-target tracing to probe spatiotemporal cartography of cell membranes. *Nature methods 5*, 8 (2008), 687–694. 3

[SCI] SCI INSTITUTE:. Openvisus [online]. URL: https://github.com/sci-visus/OpenVisus. 6

[Sci16] Scientific visualization contest [online]. 2016. URL: https://www.uni-kl.de/sciviscontest/. 3

[SJ97] SAXTON M. J., JACOBSON K.: Single-particle tracking: applications to membrane dynamics. *Annu Rev Biophys Biomol Struct 26* (1997), 373–399. 1

[SPD*19] SOLER M., PETITFRERE M., DARCHE G., PLAINCHAULT M., CONCHE B., TIERNY J.: Ranking viscous finger simulations to an acquired ground truth with topology-aware matchings. In *IEEE 9th Symp. on Large Data Analysis and Vis. (LDAV)* (2019), pp. 62–72. 3

[SW98] SILVER D., WANG X.: Tracking scalar features in unstructured datasets. In *Proc. IEEE Visualization* (1998), IEEE Computer Society Press, pp. 79–86. 3

[TPS*17] TINEVEZ J.-Y., PERRY N., SCHINDELIN J., HOOPES G. M., REYNOLDS G. D., LAPLANTINE E., BEDNAREK S. Y., SHORTE S. L., ELICEIRI K. W.: Trackmate: An open and extensible platform for single-particle tracking. *Methods 115* (2017), 80–90. 3

[VVOW*17] VALLOTTON P., VAN OIJEN A. M., WHITCHURCH C. B., GELFAND V., YEO L., TSIAVALIARIS G., HEINRICH S., DULTZ E., WEIS K., GRÜNWALD D.: Diatrack particle tracking software: Review of applications and performance evaluation. *Traffic 18*, 12 (2017), 840–852. 3

[WBP12] WEBER G., BREMER P.-T., PASCUCCI V.: Topological cacti: Visualizing contour-based statistics. In *Topological Methods in Data Analysis and Visualization II*. Springer Verlag, 2012, pp. 63–76. 3

[WCBP12] WIDANAGAMAACHCHI W., CHRISTENSEN C., BREMER P.-T., PASCUCCI V.: Interactive exploration of large-scale time-varying data using dynamic tracking graphs. In *Proc. IEEE Symposium Large-Scale Data Analysis and Vis. (LDAV)* (2012). 3, 4, 7

[WKK*15] WIDANAGAMAACHCHI W., KLACANSKY P., KOLLA H., CHEN J., BHAGATWALA A., PASCUCCI V., BREMER P.-T.: Tracking features in embedded surfaces: Understanding extinction in turbulent combustion. In *Proc. IEEE Symp. Large-Scale Data Analysis and Vis. (LDAV)* (2015). 3, 4

[YPW19] YI X., PIESTUN R., WEISS S.: 3d super-resolution imaging using a generalized and scalable progressive refinement method on sparse recovery (pris). In *Single Molecule Spectroscopy and Superresolution Imaging XII* (2019), vol. 10884, International Society for Optics and Photonics, p. 1088406. 3, 9