Positron emission particle tracking using machine learning

Nicusan, Andrei; Windows-Yule, Christopher

DOI:
10.1063/1.5129251

Document Version
Peer reviewed version

Citation for published version (Harvard):
Nicusan, A & Windows-Yule, C 2020, 'Positron emission particle tracking using machine learning', *Review of Scientific Instruments*, vol. 91, 013329 . https://doi.org/10.1063/1.5129251

Positron Emission Particle Tracking using Machine Learning

A. L. Nicuşan[1] and C. R. K. Windows-Yule[1, a)]
School of Chemical Engineering, the University of Birmingham, Edgbaston, Birmingham, UK, B15 2TT

(Dated: 16 January 2020)

We introduce a new approach to positron emission particle tracking (PEPT) based on machine learning algorithms, demonstrating novel methods for particle location, tracking and trajectory separation. The method allows radioactively-labelled particles to be located, in three-dimensional space, with high temporal and spatial resolution, requiring no prior knowledge of the number of tracers within the system, and can successfully distinguish multiple particles separated by distances as small as 2 mm. The technique's spatial resolution is observed to be invariant with the number of tracers used, allowing large numbers of particles to be tracked simultaneously, with no loss of data quality.

I. INTRODUCTION

A. Positron Emission Particle Tracking

Positron emission particle tracking (PEPT) is a technique, developed at the University of Birmingham, with which one may non-invasively track the fully three-dimensional motion of a particle, with high temporal and spatial resolution, even in the interior of dense, optically-opaque systems[1–3]. Due to its great versatility both in terms of the scales and materials of particles which can be tracked[4,5], and the sizes and geometries of the systems which can be imaged[6], the technique has wide-ranging applicability in diverse scientific, industrial and biomedical applications[7].

PEPT is performed by radioactively labelling[4,5] a particle with a positron-emitting radioisotope such as fluorine-18 (^{18}F) or gallium-66 (^{66}Ga), and using the back-to-back gamma rays produced by electron-positron annihilation events in and around the tracer to triangulate its spatial position. Figure 1 provides a simplified representation of this process: when two gamma photons produced by an annihilation event are detected, their trajectory can be reconstructed[8] to give a single *line of response* (LoR). Due to the short path lengths of β^+ particles, it can be assumed that all pairs of gamma photons are emitted either within or near the tracer particle – i.e. all LoRs should pass through or close to said tracer. Thus, with a suitably large number of LoRs, one can algorithmically determine the tracer's spatial position.

Of course, the picture presented in Fig. 1 is over-simplified. In reality, one will also typically encounter significant numbers of false LoRs due to 'false coincidence' events, where random secondary or background photons happen to interact with opposing detectors within the resolution time of the camera[3]. As such, any algorithm used to locate and track a particle must also be able to remove or ignore this noise.

There exist several distinct algorithms developed for the above-described process. In the following section, we provide a brief overview of each.

B. Current Methods

1. The Birmingham Method

The Birmingham method is the original, and still most widely-used, PEPT algorithm. In order to track a single particle, the Birmingham algorithm works as follows[1]: for a chosen number, N, of events, the algorithm first determines the minimum distance point (MDP) which minimises the distance to all LoRs, before discarding those LoRs lying furthest from this point. This process is then repeated until a pre-determined fraction, f, of the original events remains. The MDP of the remaining (fN) LoRs is then taken as the tracer's position.

In 2006, the algorithm was adapted to track multiple particles[9]. The multiple-particle case begins in the same manner as the single-particle case, with the determination of an MDP, followed by the elimination of a number of outlier particles and so on until a single location is determined. The process is then repeated, this time using only the events that were discarded from the first location process. The process is then repeated again until all tracers are found. While the original multiple-particle tracking method developed by Yang *et al.*[9] required a fixed separation between particles – allowing, for example, the rotation of a solid body to be tracked by applying 3 tracers to its surface – this constraint was later relaxed[10], allowing the tracking of multiple *freely-moving* tracers.

In the above process, the first particle to be detected will be that with the highest data logging rate, thus potentially allowing particles with differing activities to be distinguished from one another[11], and hence individually tracked. However, as the data logging rate is affected by both particle activity *and* the position of particles relative to the system's detectors, in order for the trajectories of two particles to be separated based on their activities alone, a significant difference between the two is necessary. In practice, an activity ratio $\gtrsim 2$ is typically used. An issue with this method of trajectory separation, therefore, is that as the number of tracers increases, the total activity required increases exponentially, meaning that the detector used may rapidly become saturated. An additional limitation of this method is that, as data is discarded for each iteration of the particle-finding routine, the accuracy of location decreases with each additional

a)Electronic mail: c.r.windows-yule@bham.ac.uk

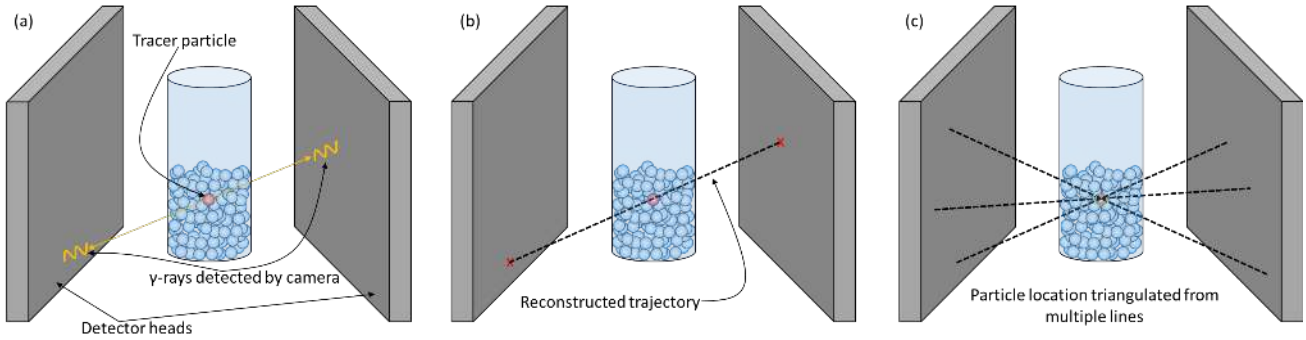

FIG. 1. Simple schematic diagram illustrating the manner in which PEPT can be used to locate a single tracer particle in a granular bed. If an active tracer particle (highlighted red for clarity, though in reality identical to all others of its species) emits a pair of back-to-back γ photons (panel (a)) that are both simultaneously recorded by the detectors of the positron camera, their trajectory can be reconstructed algorithmically (panel (b)). By finding the point of intersection of several such lines of response (panel (c)), the position of the tracer's centre may be determined algorithmically.

particle present.

Despite the above issues, the Birmingham multiple particle method has been shown to successfully locate and track up to 4 particles separated by twice the spatial resolution of the camera (approximately 12 mm).

Further details regarding the Birmingham method may be found in Parker *et al.*[1], Parker[3].

2. The Line-Density Method

The line-density method subdivides the experimental volume into a mesh of voxels recording, for each voxel, the number of LoRs passing through its volume. It is then possible to locate voxel(s) corresponding to the maximum (or, in the case of multiple particles, local maxima) in line density. The particle location is determined by spatially averaging the voxellated data to produce one-dimensional histograms in the neighbourhoods of these maxima – one each for the x, y and z directions – and fitting Gaussians to determine the centroid position. This approach has been successfully used to track up to 8 particles simultaneously. More details regarding the line-density method may be found in Bickell *et al.*[12].

3. Multiple Location-Allocation Algorithm (MLAA)

The multiple location-allocation algorithm (MLAA)[13] was traditionally used in social sciences to find the location of m sources of products that minimise the cost of supplying these goods to n destinations, taking into account both the distance and demand for the product. This is achieved by iteratively minimising the weighted sum of distances from the sources to the destinations, where each weight is a representation of the demand.

A variant of the multiple location-allocation algorithm was used by Gundogdu and Tarcan[14] in conjunction with the K-Medoids[15] algorithm to track between two and four particles. The method subdivides the experimental volume into voxels, assigning each of them a value corresponding to the number of LoRs passing through them. The voxel values are then smoothed using a thresholding method. The K-Medoids algorithm is used to calculate the starting guess for the multiple location-allocation algorithm, which uses the voxel values as the weights associated with the distances from the sources (the tracers locations) to the destinations (the non-zero voxels). The authors successfully tracked two particles separated by 30 mm in the z-direction using MWPC PET cameras.

4. K-Medoids

An algorithm developed by Gundogdu[16] computes the "midpoints", the points that minimise the distances between every pair of LoRs, removing the ones which have the nearest neighbour further than 2.479 mm. A variant of the K-Medoids algorithm[15] is used along with a method for validating the number of clusters based on the Silhouette Coefficient (SC). The author used 200 LoRs to compute one frame and noted that the clustering degrades with increasing number of LoRs. It successfully located two tracers separated by 30 mm in the z-direction using an MWPC PET camera, with the author mentioning the possibility to track up to four particles in some applications.

5. Clustering Methods

The clustering method of Wiggins, Santos, and Ruggles[17] uses as its starting point a three-dimensional mesh such as that used in the line-density method described above.

The method then uses Gaussian-means (G-means) clustering[18] – an adaptation of the well-known k-means algorithm[19] – to identify the centroid of the particle(s) to be located. The approach utilises principal component analysis and goodness-of-fit testing to determine, without *a priori* knowledge, the number (k) of clusters in a data set and determine their positions.

The algorithm begins by assuming a value $k = 1$ to calculate a global centroid. The algorithm then determines whether to 'accept' or 'split' the corresponding cluster dependent on its adherence to a Gaussian fit: if the data is well described by a Gaussian, it is accepted as a single

cluster; if not, it is split into two separate clusters and the process repeated for $k = 2$. The splitting process continues until all clusters are found[20], and the centroids of these clusters are taken as the tracer locations.

Further details regarding this method – including full details of the method used to determine the quality of the Gaussian fit used and hence choose whether to accept/split particles – may be found in Wiggins, Santos, and Ruggles[17].

6. The Feature Point Identification (FPI) Method

The feature point identification method again begins with the consideration of a 3D array of line-densities[21]. Using this method, the line density values belonging to the various voxels are treated as grey values, and conventional (optical) image analysis techniques, namely feature point identification[22,23] are applied to the PEPT data. This technique has been shown to be capable of locating up to 100 particles, though the accuracy of location is shown to decrease with the increasing number of tracers used. Specifically, the location error is found to scale approximately as $\sqrt{N_{tracer}}$.

7. The Odo Triangulation Method

The method detailed in the work of Odo *et al.*[24] operates, at a fundamental level, like the original Birmingham method, iteratively discarding erroneous LoRs until, theoretically, only 'true' (i.e. non-corrupted) LoRs remain, from which an accurate particle position may be determined. The method of Odo *et al.*[24] differs significantly from the original work of Parker *et al.*[1] in that, rather than simply discarding those LoRs falling furthest from the current minimum distance point, a series of heuristic conditions are instead applied to determine, for a given sample, which LoRs are corrupted and hence should be discarded. Notably, the algorithm utilises known information regarding the size of the tracer – and hence the likely volume from which 'true' LoRs are likely to emanate – to refine the location of the tracer's centre autonomously, without requiring the user to stipulate a specific fraction of LoRs to discard. While this method is yet to be directly tested for the case of multiple particle tracking, the authors suggest that – unlike the Birmingham algorithm – it should be possible to use the *a priori* information provided to the algorithm to separate the trajectories of particles without the necessity of labelling them with differing activity levels.

8. Voronoi-based Multiple Particle Tracking (VMPT)

The VMPT method of Blakemore *et al.*[25] represents the most recently-published PEPT algorithm to date (though the algorithm itself was originally described in the Master's thesis of the lead author[26]). The method begins by discretising the LoRs emitted by a tracer, typically into a set of points separated in space by a distance equivalent to the particle diameter. These discrete points are then used as seed points[27] to produce a Voronoi tesselation. Naturally, as we approach the location of the tracer, the density of seed points will increase, and the size of the local Voronoi regions will correspondingly decrease. The data produced through the above process are then filtered so as to remove outliers, and any remaining points corresponding to Voronoi cells with areas significantly larger than the mean are discarded (full details of this filtering process may be found in Blakemore *et al.*[25]). The remaining data points are then clustered, and the geometric centre of these clusters taken as the tracer location. The VMPT method can track at least 20 particles, requires no *a priori* knowledge of the number of tracers in the system, and allows for tracers leaving and entering the field of view.

C. PEPT-ML

Though all reliable and effective, the methods described in the previous section all carry certain weaknesses. For example the Birmingham algorithm, while highly effective and computationally efficient in the single-particle case, is only proven in the multiple-particle case for only a small number of tracers, requires the use of tracers with differing activities to perform trajectory separation, and necessitates an *a priori* knowledge of the number of particles in the system (an issue shared by the line-density method) for particle location. This may be a significant issue, for instance in systems where tracers leave and re-enter the field of view.

The MLAA-based method is very computationally expensive due to the complexity of the algorithms involved. The K-Medoids approach, while more robust than the MLAA algorithm, also proved to have difficulty distinguishing more than two particles. Both algorithms were able to track particles separated by more than 30 mm in the z-direction, albeit using a lower-resolution camera than in this present study.

The line-density, clustering, MLAA, FPI and VMPT methods, meanwhile, are all mesh-based, meaning their results are inherently grid-size dependent and, with the exception of the VMPT method, carry potential issues in terms of pixelisation[17].

The Odo triangulation method shows great promise, but is yet to be tested for the multiple-particle case, and at present relies on the assumption of a spherically-symmetric tracer, and requires a known distribution of positron-emitting-radioisotope on or in the tracer.

In this article, we introduce a new, machine-learning-based method which is mesh-free, requires no prior knowledge of particle number, and whose accuracy and acquisition rate are scalable with particle number.

II. ALGORITHM DETAILS

PEPT-ML is novel, machine-learning-based algorithm for tracking the trajectories of multiple particles in Euclidian space based on their lines of response. As discussed in section I A, for any given set of LoRs, the main problem is finding the regions in space (representing the

tracers) from which the lines stem. The trajectory of each tracer can then be constructed based on the individual positions found. The main steps underlying the PEPT-ML algorithm are as follows:

1. Split the data into a series of individual '*samples*', each containing a given number of LoRs[28].
2. For every sample of LoRs, compute the points ('*cutpoints*') in space that minimise the distance between every pair of lines.
3. Cluster every sample of cutpoints using a Hierarchical Density-Based Spatial Clustering of Applications with Noise (HDBSCAN) algorithm[29] and extract the centres of the clusters ('*1-pass clustering*').
4. Split the centres into samples of a given size.
5. Cluster every sample of centres using the HDBSCAN algorithm and extract the centres of the clusters ('*2-pass clustering*').
6. Construct the trajectory of every particle using the centres from the previous step.

Each LoR is essentially a line in a three-dimensional Cartesian space, defined by two points. In this sense, the algorithm is agnostic to the arrangement of the detectors, as virtually any geometry can be used if the 3D coordinates of the points are supplied. For any 'real' data set, the time at which the LoR was recorded is also included.

In order to capture the movement of particles in space, the dataset is split into samples of a given number of LoRs. A given sample will be used to compute one pseudo-instantaneous position for each tracer present in the system. As such, the sample size (ie. the number of LoRs per sample) should be large enough that these positions may be accurately determined, but small enough that the tracer does not move significantly during the sample's duration. The use of a smaller sample size also increases the temporal resolution of the PEPT data produced. In reality, this temporal resolution is a function of the velocity and activity (i.e. number of gamma-ray pairs emitted per unit time) of the tracer, and the properties of the detector system used.

A. Space transformation

Most clustering algorithms, including that employed in our present work, operate using points, rather than lines. As such, in order to apply the algorithm to PEPT data, we must find a suitable means through which to convert the line of response (LoR) data provided by our detector system into point data. In order to capture the position of multiple tracers, every pair of lines must be considered. For every pair of lines (L_A, L_B), each defined by two points A_1, A_2 and B_1, B_2 respectively, the *cutpoint* M is defined as the point which minimises the distances to the two lines. Assuming the two lines are not parallel, this point corresponds to the middle of the unique segment that is perpendicular to both lines. Writing the lines in terms of position and direction vectors yields:

$$\begin{cases} L_A : A(s) = P + sU, \text{ where } P, U \in \mathbb{R}^3, s \in \mathbb{R} \\ L_B : B(t) = Q + tR, \text{ where } Q, R \in \mathbb{R}^3, t \in \mathbb{R} \end{cases} \tag{1}$$

with the additional condition that $U \neq R$. Based on the definition of the two lines, the position and direction vectors can be defined as $P = A_1$, $U = A_2 - A_1$ and $Q = B_1$, $R = B_2 - B_1$. The unique segment that is perpendicular to both lines is defined by two endpoints $A(s_0)$, $B(t_0)$, with the property:

$$\begin{gathered} (A(s_0) - B(t_0)) \perp U, R \iff \\ \begin{cases} (P + s_0 U - Q - t_0 R) \cdot U = 0 \\ (P + s_0 U - Q - t_0 R) \cdot R = 0 \end{cases} \end{gathered} \tag{2}$$

This system of equations can be solved to find s_0 and t_0 and hence the cutpoint M:

$$M = \frac{A(s_0) + B(t_0)}{2} \tag{3}$$

In order to minimise crosstalk between lines that stem from different regions in space, only the cutpoints of segments smaller than a certain maximum length should be considered. The cutpoints corresponding to 400 LoRs from two real tracers and a maximum length of 0.05 mm are depicted in panel *a*) of Fig. 2. The image depicts the inherent characteristics of PEPT data: high-density clouds of cutpoints around the two tracers, a significant amount of noise and crosstalk between the particles.

B. Clustering Cutpoints

As noted above, PEPT-ML locates particles by clustering the cutpoints computed as described in the preceding section. This method works on the assumption that the cutpoints will be centred around the tracers from which the lines originate. However, tracers which are more active than others will yield more LoRs, and hence there will exist a higher density of cutpoints around them. Therefore, a density-based clustering method which can handle varying levels of density and noise in the data should be employed. The HDBSCAN[29] (Hierarchical Density-Based Spatial Clustering of Applications with Noise) algorithm was chosen due to its versatility, and the availability of a high-performance implementation[30].

The HDBSCAN algorithm in essence leverages a series of machine-learning techniques: defining a new metric that "pushes" low-density points away from higher-density regions, constructing the minimum spanning tree using this metric, performing hierarchical clustering using the single-linkage method and extracting optimal clusters from this hierarchy. These steps are illustrated on real PEPT data in Fig. 2.

As single-linkage hierarchical clustering is very sensitive to sparse points connecting higher-density regions,

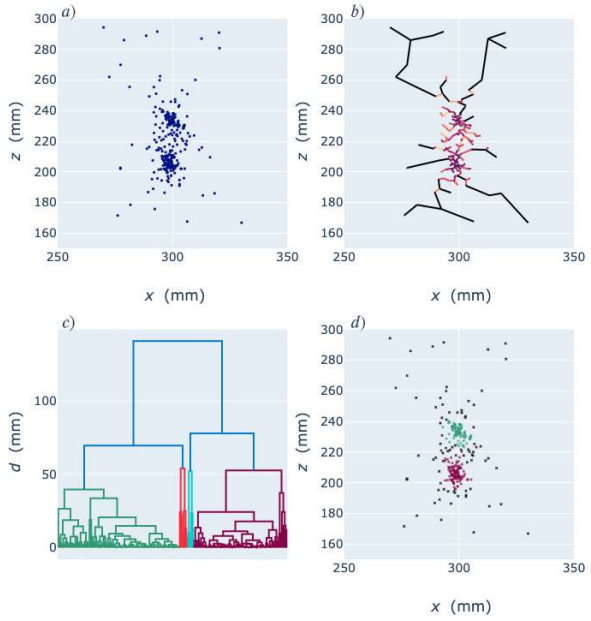

FIG. 2. Visualisation of the HDBSCAN algorithm's steps using real PEPT data. The experiment used two static 0.6 mm diameter particles, separated by 32 mm in the z-direction. Panel *a*) depicts cutpoints of a 400-LoR sample from a top-down view of the PEPT scanner. Panel *b*) shows the minimum spanning tree based on the mutual reachability distance of the cutpoints, colour-coded for increasing d_{mreach} from purple to yellow, and finally black for d_{mreach} larger than twice the mean of the distances. Panel *c*) illustrates a dendrogram built from the hierarchy of connected components of the minimum spanning tree. Panel *d*) shows the two clusters found (green and purple circles), along with the cutpoints classified as noise (black crosses). Note that there are two additional steps (omitted for brevity) between *c*) and *d*): condensing the dendrogram and extracting clusters of high stability and possibly differing density levels.

an initial step is required to prepare the data for robust clustering. For a given point a and a set parameter k, the *core distance*, denoted $core_k(a)$ is defined as the Euclidian distance between the point a and its k-th nearest neighbour. Using this concept, the *mutual reachability distance* between any two points a and b is defined as:

$$d_{mreach-k}(a,b) = max\left(core_k(a), core_k(b), d(a,b)\right) \quad (4)$$

If both points a and b are within high-density regions in space, then the Euclidian distance $d(a,b)$ between them is going to be approximately the same as their core distances. However, if, for example, a is an isolated point (being part of a sparse region in space), which does not have many close neighbours, the k-th nearest neighbour is going to be much farther away than point b. Therefore, the mutual reachability distance is a metric which has effectively the same value as the Euclidian distance for points in high-density regions, while being much larger for sparse points. This effectively 'pushes' sparse points away from higher-density regions, ensuring that neighbouring clusters are not inaccurately treated as a single particle due to the presence of sparse points in between them.

Using the mutual reachability distance as the metric for the points in the dataset, the *minimum spanning tree* can be constructed. This is a weighted graph containing a set of *vertices* (in this case, 3D points) and a set of *edges* connecting them, where the weight of each edge is the mutual reachability distance between the points that the edge connects. The most important property of a minimum spanning tree is that it has the minimum sum of weights of any graph that can be constructed from the given data. Therefore, the minimum spanning tree is essentially the shortest "path" which connects all the points. For a set of cutpoints corresponding to 400 LoRs of two real tracers, the minimum spanning tree is illustrated in panel *b*) of Fig. 2, where the edges are colour-coded according to their mutual reachability distances. Note that the edges connecting sparse points around the tracers have very large values due to the mutual reachability distance being much larger than the Euclidian distance.

A dendrogram can be constructed, depicting the distance between any two connected subgraphs (corresponding to any two clusters) in the minimum spanning tree. Sorting all the edges in the minimum spanning tree and starting from the smallest edge, a hierarchical tree structure can be constructed where the height of any split corresponds to the weight (mutual reachability distance) of the edge connecting the two subgraphs. Panel *c*) in Fig. 2 depicts the dendrogram constructed for the cutpoints of real PEPT data.

At this point, the dendrogram can be 'cut' at a certain height, and the remaining subgraphs would be classified as the persistent clusters. However, this approach would only yield clusters of the same density. Therefore, the dendrogram will be condensed to include a notion of "cluster stability". The tree can be traversed from the bottom up, condensing any split that has fewer vertices than a defined *minimum cluster size*, keeping track of the height at which the splits occurred. A stable cluster will have points gradually 'falling out' of it, rather than having large differences between the heights at which the splits occur. Hence a measure of the stability of a cluster is simply how small the differences are between the heights at which the splits happen. The final step of extracting the clusters from the condensed dendrogram is a comparison between the stability of the nodes. The extracted clusters are shown in panel *d*) of Fig. 2.

C. Particle Location

In the previous section, we have discussed how the PEPT-ML algorithm determines which cutpoints within a given sample of data belong to a given tracer. The algorithm then assigns a position to this tracer, for the current sample, by determining the geometric centre of this cluster. This is done simply by averaging the x-, y- and z-positions belonging to all the cutpoints currently assigned to said tracer by the HDBSCAN algorithm. This '*centre of cutpoints*' can then either be simply taken as the current particle's location ('*one-pass clustering*') or used to calculate a still more accurate position ('*two-pass*

clustering').

Two-pass clustering is performed by taking the 'centres of cutpoints' from several samples of data (either overlapping or contiguous), and then re-applying the HDBSCAN algorithm – following the same methodology as described in section II B – to cluster the centres themselves and, as above, calculating the geometric midpoint of these clustered points (the '*centre of centres*'). For a suitably-chosen number of samples, this centre of centres will typically represent a significantly more accurate estimation of the tracer's position. In section IV A we show examples in which this 'reclustering' method achieves a five-fold increase in accuracy.

The application of the re-clustering algorithm will – due to its use of multiple individual samples, and hence a larger number of LoRs than for single-pass clustering – reduce the temporal resolution with which data are acquired. However, this effect may be reduced by utilising a 'windowing' technique – i.e. applying two-pass clustering to overlapping samples. In fact, with a large enough overlap, the decrease in temporal resolution can be effectively eliminated. The use of overlapping samples will, however, act to smooth the data obtained, a factor which – dependent on the specific application for which PEPT is applied, be undesirable.

As is clear from the above, the balance of temporal and spatial resolution, and the degree of smoothing achieved on data, can be altered by the choice of overlap, and indeed the decision of whether or not to apply two-pass clustering at all, as well as the sizes of samples used for both clustering and reclustering, affording the user a significant degree of control over the optimisation of the data output. For example, in applications tracking fast-moving particles and requiring extremely high temporal resolution, one may apply only first-pass clustering, utilising a minimal sample size. Conversely, if one wishes to track slow-moving particles with high precision, one may apply two-pass clustering, using both large samples of cutpoints and large samples of centres.

D. Particle Identification and Tracking

After clustering all samples of cutpoints (one-pass clustering) and then, if desired, all samples of centres (two-pass clustering), the particle trajectories have to be reconstructed based on the located clusters' centres. However, there are cases when tracers' trajectories may intersect or collide, briefly losing each particle's identity as they are too close in space to be differentiated (as illustrated in Fig. 3). Therefore, besides tracers' locations, a new discriminating factor must be introduced such that, even following an intersection or collision, individual particles can be 'recognised' and their trajectories distinguished.

There is a strong correlation between a tracer's activity level and the number of LoRs originating from it. Given n LoRs in a sample, there are a maximum of $\frac{1}{2}(n^2-n)$ cutpoints to be considered. Even though the cutpoints associated with lines farther than a set maximum distance are ignored, the number of cutpoints can be expected to scale quadratically with the number of LoRs. Therefore, differences in the activity level of different tracers will be amplified when computing the cutpoints (i.e. if one tracer is twice as active as another, the former will have four times more cutpoints around it than the latter).

When computing the centres of every cluster of cutpoints, it is possible to keep track of the number of cutpoints included in the aforementioned cluster. This way, we exploit the stored number of cutpoints to provide our particles with distinct '*signatures*'.

Using the cluster size as the signature of the tracers, their trajectories can be correctly reconstructed even after an intersection or collision (see Fig. 3).

E. Implementation Notes

The algorithms described in this section were implemented as a series of modules in the Python programming language, using C extensions for compute-intensive tasks such as cutpoint-finding. The modules are included as part of `pept`, a Python library that integrates all the tools necessary to perform research using PEPT, including tracking, simulation, data analysis and visualisation tools. The open-source package, which also includes functionality to run the traditional Birmingham algorithm, can be found **here**. Using a PC with 8 GB RAM and a 2.6 GHz Intel Core i7 quad-core processor, a 4-particle data set with 1,000,000 LoRs requires approximately 13 seconds (real elapsed time) to initialise, 9 seconds to find determine the cutpoints, 10 seconds for the first pass of clustering and one second for the second pass of clustering.

The PEPT-ML algorithm is written such that it may be operated in two distinct 'modes' by setting the `allow_single_particle` flag to `true` or `false`. Setting the flag to `false` allows multiple particles to be distinguished at a smaller separation, but will not work for systems containing only a single particle. Conversely, if set to `true`, the algorithm can detect a single particle in isolation, but may falsely interpret two or more nearby particles as a single cluster. While the choice of a suitable mode is typically obvious, for systems in which tracers may move in and out of the field of view, users must choose either to ensure that all particles are tracked at all times but increase the minimum separation above which two or more particles may be distinguished (`allow_single_particle = true`), or risk occasionally 'missing' particles when only one is present in the system, but being able to better resolve individual particles in close proximity (`allow_single_particle = false`).

III. EQUIPMENT AND SIMULATION METHOD

A. ADAC Forté Dual-Headed Positron Camera

Experimental tests of the PEPT-ML algorithm are conducted using the University of Birmingham's ADAC *Forté* dual-headed positron camera. The camera consists of two parallel, planar detector heads, each com-

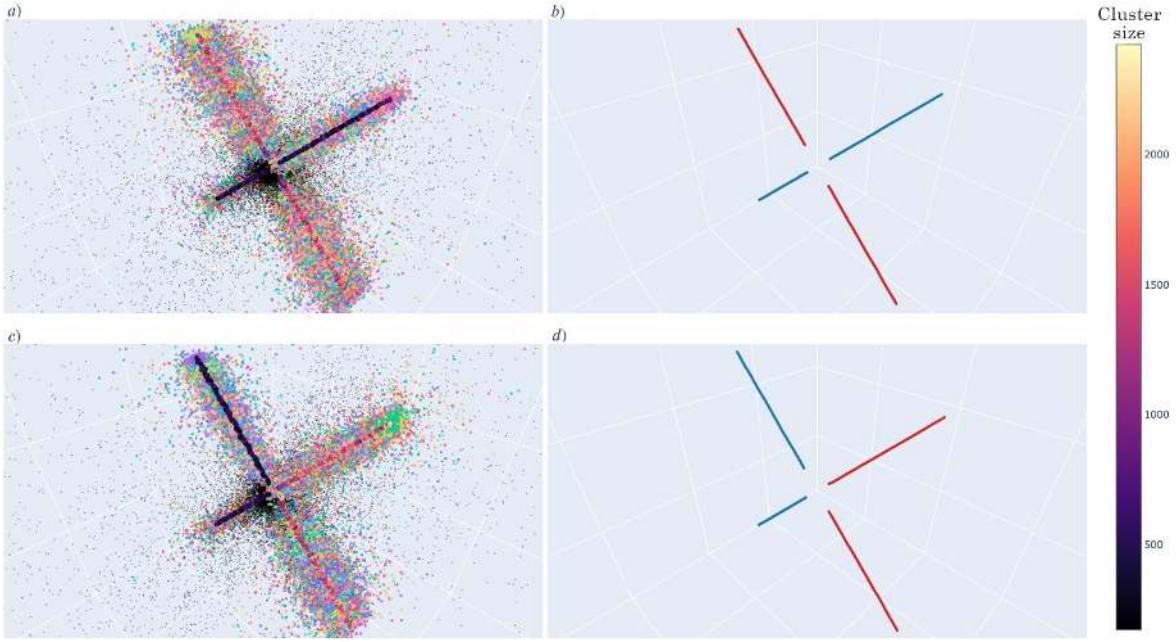

FIG. 3. Visualising the intersection [panels a), b)] and collision [panels c), d)] of two simulated tracers (see Section III B), in which one tracer is twice as active as the other. The left-hand panels [a), c)] show the clustered cutpoints (coloured small circles) and noise (black small circles) after one pass of clustering, along with the centres of the clusters (large circles). The cluster size of every centre has been colour-coded, with colours toward the orange (purple) end of the spectrum representing clusters containing more (fewer) cutpoints. The right-hand panels [panels b), d)] show the separated trajectories after the second pass of clustering. Note that as particles come very close, all cutpoints are classified as noise, as the HDBSCAN algorithm has, in this instance, been implemented so as to specifically reject single clusters, which acts to increase the spatial resolution achieved.

prising a single, $590 \times 470\text{mm}^2$ sodium iodide scintillator crystal coupled to an array of 55 photomultiplier tubes. The camera has an intrinsic spatial resolution of approximately 6mm and facilitates a maximum data rate of the order of 100kHz. Further details of the system may be found in Parker *et al.*[2].

The camera described is the same as that used by Yang *et al.*[9,10] in their work testing the Birmingham multiple-particle tracking algorithm, allowing us to draw direct, quantitative comparison between this method and PEPT-ML.

The work of Bickell *et al.*[12] and Wiggins, Santos, and Ruggles[17], however, used cameras with significantly differing resolutions, data rates and geometries. While these differences preclude a full, quantitative comparison between the methods, it is nonetheless possible to compare certain key features and trends relating to the various methods.

B. Simulations

In order to extend the analysis of the tracking algorithm beyond what is reasonably achievable in a lab setting, a '*PEPT simulator*' was developed. This creates the possibility to track hundreds of particles at the same time, far exceeding the maximum event rate of the PEPT scanner available to the authors. In addition, tracer positions can be known analytically before creating and analysing the corresponding PEPT data, allowing the true accuracy of the tracked positions to be computed.

In order to accurately simulate PEPT data, one must recreate the various noise- and error-sources inherent to 'real', experimental detector systems. To this end, the PEPT simulator developed takes into account the following crucial features:

- Tracers activity (including the random nature of β^+ decay).
- Positron movement from the point of generation to the point of annihilation and γ-ray emission, including the relevant dependence on the tracer material and surrounding environment.
- Area, depth of the detector crystals, and inherent resolution of the detector system.
- Random noise.

The PEPT simulator receives a trajectory composed of n_p discrete points in space, each representing the centre of a simulated object. Because particle decay is a random event, the time values associated with every position along the trajectory are taken from a sorted random uniform distribution. A second parameter, '*sampling times*' – another sorted random uniform distribution – represents the times at which one positron is emitted. The location from which this happens is taken as the particle position in the trajectory that has the closest time value to the given sampling time. Note that multiple particles' paths can be included in the trajectory received by the PEPT simulator, as long as the times of the positions

in space are sorted. This creates the possibility of having multiple particles of different relative activities in the same trajectory. For example, let the path of particle A be defined by 1000 points in space, and let the path of particle B be defined by 2000 points in space. The paths of A and B both span the same time interval. Because the sampling times (the times representing positron emission) are taken from a random uniform distribution, the chance of the decay event having the location of particle B is twice as large as for particle A. Hence, particle B will have twice as many LoRs as particle A, accurately simulating their differing activities.

For every sampling time and particle position in space, the β^+ decay is simulated. The particle position represents the centre of the simulated object in space. This object can have any defined shape, as a "shape function" returns a random point inside the object relative to its centre, representing the point of decay (and positron generation). However, the positron travels a distance from its point of emission before hitting an electron, annihilation and so emitting the γ-ray. This *positron range* is a function of the kinetic energy of the positron, which is itself a function of the radioactive material.

For any given tracer material that undergoes β^+ decay, the decay energy (equal to the energy difference between the parent and daughter nuclei) is imparted as kinetic energy to the positron and neutrino formed. This kinetic energy is not divided equally, resulting in a *beta energy spectrum*, a distribution representing the fraction of the total decay energy that is carried by the positron. This distribution can be found analytically from the Fermi theory of beta decay, resulting in a relatively skewed normal distribution. Therefore, a Gaussian distribution is fitted to the beta energy spectrum as an approximation, centred at half of the particle decay energy. The standard deviation is set to one third of the mean, such that 99.7% of the data will lie between 0 and the particle decay energy. Once the kinetic energy of the positron is taken from the beta energy spectrum approximation, the positron range is calculated from the following correlation[31]:

$$\begin{aligned} R_{ex}(E_i) &\approx \frac{b_1 E_i^2}{b_2 + E_i} \\ b_1 &= \frac{4.569A}{Z^{1.209}} \\ b_2 &= \frac{1}{2.873 - 0.02309Z} \end{aligned} \tag{5}$$

where E_i is the initial energy of the positron, R_{ex} is the 'extrapolated range' and A and Z are the effective atomic weight and number of the material surrounding the tracer. The extrapolated range R_{ex} relates the kinetic energy of the positron and the distance travelled before annihilation by fitting a Gaussian distribution in each of the three dimensions, each centred at zero. The standard deviation given is:

$$\sigma(E_i) \approx \frac{R_{ex}(E_i)}{2} \tag{6}$$

Once a travelled distance has been computed from this distribution in the x-, y- and z-dimensions, this is added to the emission location, finally resulting in the point from which the LoR is emitted. An LoR angle is taken from a random uniform distribution, finally calculating the x and y positions on the detector screens, simulating the location error due to the positron range, tracer material and surrounding environment.

However, the detector crystals are discrete elements and hence introduce another set of errors by approximating the spot the LoR hit to the centre of the detector. First, the γ-ray enters the detector head at an angle, travelling a certain distance before being captured. This 'angular error' is modelled by taking a random distance between zero and the depth of the detector crystal (16 mm for the detector system modelled here), multiplying it by the tangent of the ray's incident angle and adding the result to the original position on the detector screen. This is done in both x- and y-dimensions, for both screens. However, the γ-ray can travel through the initial detector crystal, then pass into the one adjacent to it before being captured. This "positional error" is modelled by adding a random value between $[-w, w]$ to the LoR point on the screen, where w is the width of the active area of the detector (4 mm for our PEPT scanner). This way, the area and depth of the detector crystals are simulated.

The final step is to add random noise to the simulated PEPT data, representing the false coincidence events. This is done by adding 30% of the total number of LoRs extra random points on the detector screens, at random positions in the data.

Though, due to the simplifying assumptions described above, our model cannot be expected to reproduce experimental data with 100% accuracy it is nonetheless found to produce a pleasingly close representation thereof, as illustrated in Fig. 4. The standard deviation of the experimental data – i.e. the 'spread' of cutpoints – is 15.33 mm in the x-dimension and 41.41 mm in the z-dimension, while for the simulated data it is 16.36 mm in the x-dimension and 38.05 mm in the z-dimension. Note that while the z-dimension exhibits large scattering due to the 2 parallel screen configuration of the scanner (see Sect. IV A), the x and y directions show similar, smaller scattering. Therefore, the 2D histograms (panels a) and b) in Fig. 4) computed for the xz-plane projection of the cutpoints is also representative of the yz-plane projection, covering the full three-dimensional space.

IV. TESTING THE ALGORITHM

A. Spatial Resolution

Fig. 5 shows PEPT data corresponding to a pair of particles following a known, circular trajectory, processed using both PEPT-ML (panels b), c)) and the Birmingham algorithm (panel d)). From these data, it is possible to extract a quantitative measure of the spatial accuracy achieved by the two algorithms. Specifically, we quantify the positional error, $\overline{\Delta\chi}$ ($\chi = x, y, z$), as the mean devia-

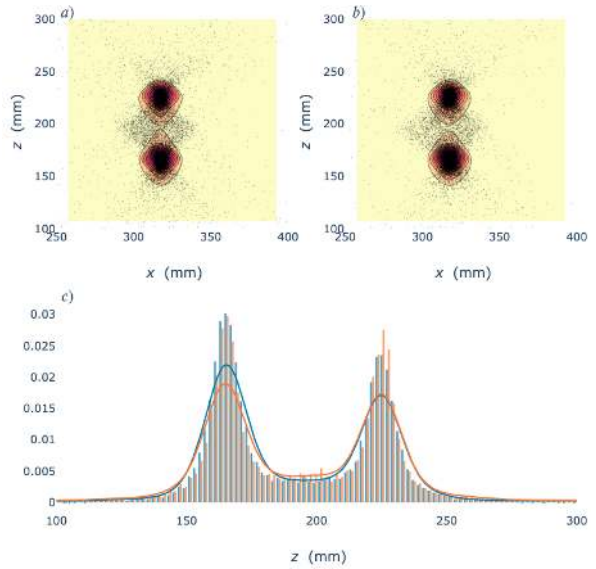

FIG. 4. Comparison between experimental and simulated PEPT data, based on two 0.6 mm diameter tracers separated in the z-dimension by 64 mm. The two top panels – with panel *a*) showing experimental data and *b*) simulated data – depict 19,000-cutpoint clouds as black circles from a top-down perspective (the xz-plane) along with 2D histograms colour-coding the density of the cutpoints, where the yellow (purple) end of the spectrum represents a lower (higher) density. The bottom panel shows the superimposed histograms of the cutpoints for the experimental (orange lines) and simulated (blue lines) PEPT data in the z-dimension. The bin sizes used are 15 mm for the 2D Histograms and 2 mm for the 1D Histogram.

tion of the measured points, χ_m from the corresponding 'real', analytically known values, χ_r, in each of the three Cartesian coordinates, i.e.

$$\overline{\Delta\chi} = \frac{1}{N_P}\sum_{i}^{N_P}|\chi_m - \chi_r|, \qquad \chi = x, y, z \tag{7}$$

where N_P is the number of data points used.

Using the Birmingham method, we obtain mean errors $\overline{\Delta x} = 0.50$ and $\overline{\Delta y} = 0.43$ in the horizontal and vertical directions parallel to the detector face, and $\overline{\Delta z} = 2.10$ for the perpendicular direction. As expected, data in the z-direction show significantly higher errors than the x- or y-directions due to the geometry of the camera.

For PEPT-ML, after a single pass, these values are $\overline{\Delta x} = 0.35$, $\overline{\Delta y} = 0.34$ and $\overline{\Delta z} = 2.07$. In all cases, PEPT-ML – even before reclustering – demonstrates an improved accuracy, though the difference is less pronounced in the z-dimension. It should be noted, however, that – as we will see in section IV D – the difference in accuracy between PEPT-ML and the Birmingham algorithm becomes more pronounced as the number of particles tracked increased. After the implementation of 2-pass clustering, the errors obtained using the new algorithm reduce dramatically to $\overline{\Delta x} = 0.18$, $\overline{\Delta y} = 0.17$ and $\overline{\Delta z} = 0.41$.

B. Temporal Resolution

The *dimensional* temporal resolution (i.e. the number of locations per second) of a PEPT measurement is highly dependent on a variety of factors, for example tracer activity and detector efficiency. As such, we explore here instead a more easily-generalisable dimensionless proxy measurement – the number of lines of response required to successfully locate a particle – from which one can easily estimate the temporal resolution for any given tracer activity and detector set-up.

In order to test the temporal resolution, we simulate a pair of particles separated by a distance of 170mm and use PEPT-ML to detect their positions using a varying number of lines of response, N_L (see, as an example, Fig. 6). For each value of N_L, the particle centroids are determined 1,000 times using 1,000 distinct simulated samples, and the mean deviation of the measured positions from the true positions computed as per equation 7.

Our analysis shows that PEPT-ML is capable of locating a tracer to within 3.5mm using as few as 25 lines of response; for Birmingham's ADAC camera operating at its maximal acquisition rate of 100kHz, this value corresponds to a temporal resolution of 0.25ms. For more modern systems, such as the ECAT EXACT3D scanner used in the PEPT Cape Town facility which operates at coincidence rates of up to $4 \times 10^6 \mathrm{s}^{-1}$, one may therefore anticipate temporal resolution of the order of 6.25μs using PEPT-ML.

The ability to successfully track particles using so few lines of response carries considerable benefits not just in terms of the potential speed with which particles may be tracked, but also in term of the *sizes* of particles which can be successfully imaged. As the amount of β^+ activity a particle can potentially hold – i.e. the rate at which it can produce gamma-ray pairs – decreases as d^{-3}, the ability to track particles using fewer LoRs allows the possibility of tracking smaller particles. The miniaturisation of PEPT tracers is a significant aim in the field, with the ultimate goal of extending the PEPT technique into the biomedical field, where it may prove invaluable, for example, in tracking and mapping blood flow or gastrointestinal circulation.

Of course, by using more than 25 lines of response and/or by applying reclustering, the spatial resolution with which particles are located may be significantly increased, as illustrated in Fig. 7. In the x- and y-coordinates, as one may intuitively expect, the spatial resolution is observed to increase monotonically (i.e. the error of location to decrease monotonically) with increasing N_L, and to improve further when reclustering is applied. For the z-coordinate, the situation is somewhat more complex. For one-pass clustering, the results again vary monotonically with N_L. For the two-pass case, however, the accuracy of location is notably improved for $N_L \lesssim 100$, but offers no statistically-significant improvement over the one-pass case for larger values of N_L. The origin of this behaviour is thought to be related to the inherent asymmetry of the cut-point clouds generated by the two-detector geometry, which creates a 'tear-drop'-shaped cluster, with a higher density of points lying to

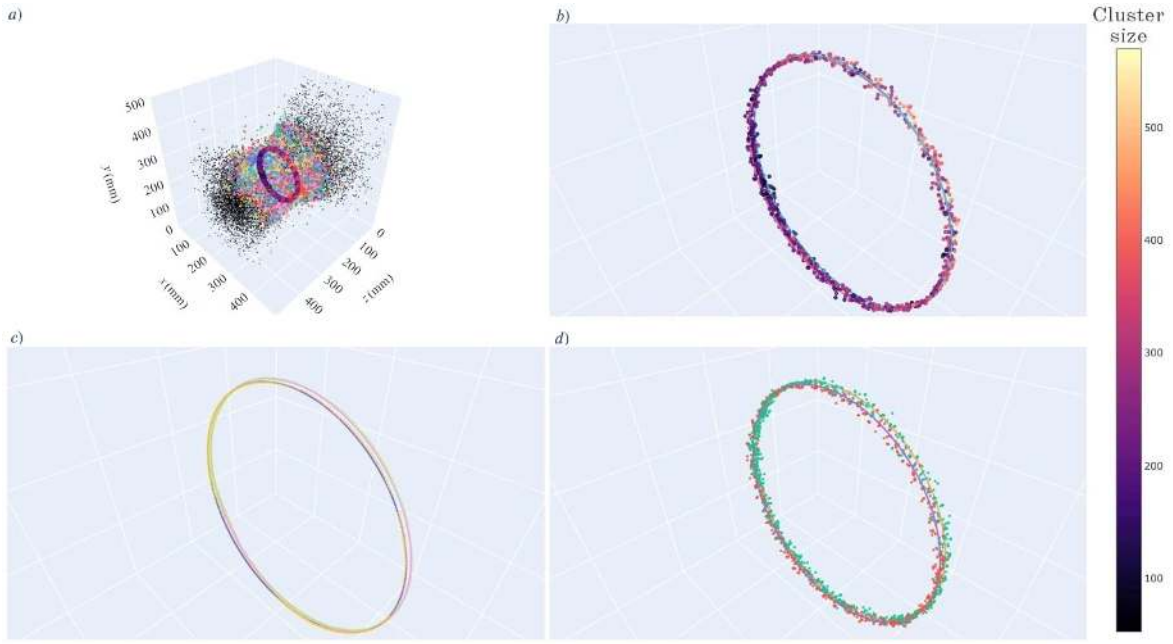

FIG. 5. Tracking two particles rotating at 42 RPM, each describing one full circle. Plot a) depicts all the cutpoints after one pass of clustering using PEPT-ML: in colour are cutpoints which were clustered; in black are the cutpoints deemed as noise. Plot b) shows the centres of the clusters from the previous plot: the number of points in the cluster ("cluster size") is colour-coded from purple to yellow. Here, purple represents a smaller cluster size, corresponding to less tracer activity, while yellow represents a larger cluster size, corresponding to more tracer activity. Plot c) depicts the cluster centres after the second pass of clustering using PEPT-ML (see section II C), after trajectory separation has been applied in the post-processing step (see section II D). Plot d) depicts the trajectories found using the Birmingham algorithm. In both images c) and d), the solid lines shown correspond to the 'real', analytically-determined circular path followed by the tracers.

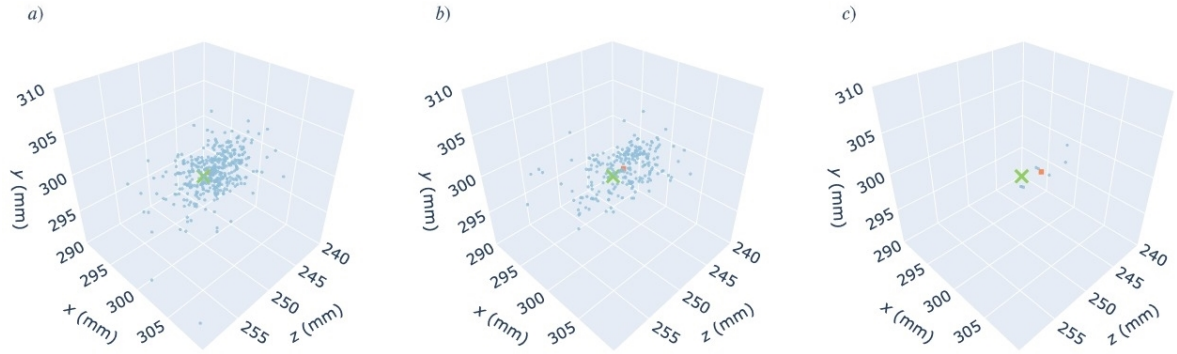

FIG. 6. Visualising the manner in which the clustered cutpoints (blue dots) vary with the number of lines in one sample, for a simulated data set showing two particles separated by a distance of 170mm (note that, for clarity, only one of the two particles is shown). The actual particle position is depicted with a green cross, while the found particle position (the mean of the cutpoints) is shown with an orange square. Plot a) depicts 250 lines per particle per sample, plot b) shows 140 lines per particle per sample, while plot c) illustrates 25 lines per particle per sample. Note that the number of cutpoints scales quadratically with the number of lines of response.

the side of the particle further from the nearest detector head, and a lower density on the side closer to it (see Fig. 6)[32].

For small N_L values, the sparsity of points increases the first-pass error of location, but also increases the likelihood that the first-pass clusters will include all points in the tear-drop, and hence may appear at any point within its volume. As such, when re-clustered, these data give a reasonable approximation of the particle's true centre.

For overly-large N_L values, however, first-pass clusters, despite individually possessing a reduced error compared to the low-N_L case, will tend to exclude points in the more sparsely-populated region of the cutpoint cloud, hence all falling nearer the 'bulb' of the tear-drop. This leads in essence to a systematic error, which is reflected in the relatively increased $\overline{\Delta z}$ for the two-pass case. In other words – unlike for prior PEPT algorithms where accuracy always correlates positively with sample size – the use of a smaller sample size may, in some cases, not only improve temporal but also spatial resolution.

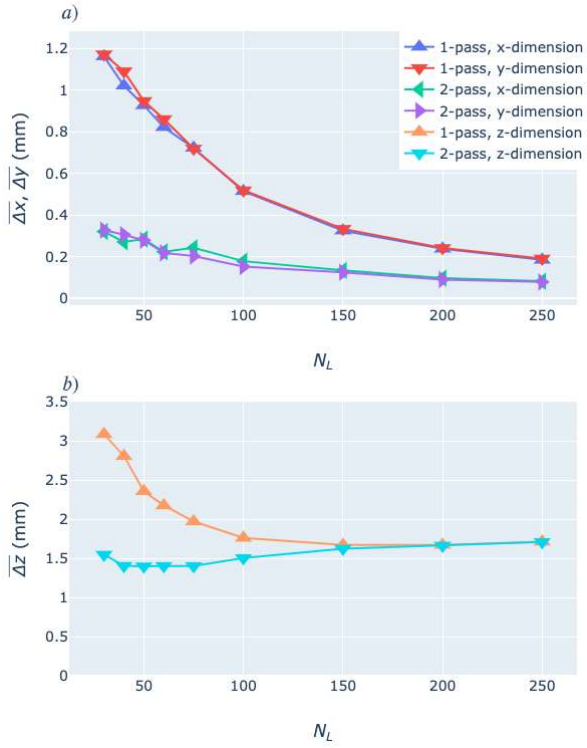

FIG. 7. Mean location error, $\overline{\Delta x}$, $\overline{\Delta y}$, $\overline{\Delta z}$, of the PEPT-ML algorithm as a function of the number of lines of response per tracer, N_L, used to determine the centroids of a simulated pair of particles separated by a distance of 170mm. For clarity, values for the x and y directions are shown in panel a), while those corresponding to the z direction are shown in panel b).

C. Minimum Separation of Multiple Particles

In addition to the ability to accurately locate a given particle in space, another key property of any multiple-particle tracking algorithm is its capacity to separately resolve two particles in close proximity.

Fig. 8 shows examples of (experimental) cutpoint clouds and located centres for a pair of particles separated by 64, 16 and 4 millimetres. The first, and most notable, observation to be drawn from this figure is that, unlike the Birmingham algorithm, which can only successfully locate pairs of particles for distances $\gtrsim$ 12mm, PEPT-ML can – using the same detector system – distinguish pairs of particles separated by as little as 2mm.

The above observation is particularly impressive as this minimum separation is in fact significantly smaller than the inherent resolution ($\sim$ 6mm) of the detector[3], yet the machine learning algorithms implemented remain capable of distinguishing two distinct point clouds – albeit with a high rejection rate.

Perhaps more remarkable still is that – as shown in Fig. 9 – the spatial resolution with which the particle centres may be determined remains almost constant with separation, in stark contrast to previous PEPT algorithms. In fact, for some smaller separations, the error of location is actually observed to *drop* slightly. This observation is thought to arise due to the fact that, for these smaller separations, the particles are both placed more centrally between the detectors, minimising the asymmetric 'skew' of the point clouds discussed in section IV B.

Considering the above, it is highly likely that location errors would be significantly reduced in other camera geometries (e.g. ring detectors) as the limiting factor in the current case is not the PEPT-ML algorithm itself, but rather the asymmetric point clouds inherent to the detectors used.

Despite the impressive consistency in the *spatial* resolution PEPT-ML with varying particle separation, as is clear from both Figs. 8 and 9, the number of cutpoints rejected by the clustering algorithm increases dramatically with reducing distance for separations $|z_1 - z_2| \lesssim 25$mm, thus acting to effectively reduce the maximum *temporal* resolution achievable. For separations $|z_1 - z_2| \gtrsim 10$mm, however, this effect can be compensated for through careful use of the reclustering method (see section II C).

D. Scalability

Having explored the various key properties of the PEPT-ML for the two-particle case, it is finally informative to assess the *scalability* of the algorithm – i.e. how well it maintains its accuracy as the number of particles within the system is increased.

Fig. 10 shows experimental data giving the accuracy (denoted by the values of the symbols) and precision (denoted by the size of their error bars) of location for 2-5 tracers placed, evenly spaced, along a line perpendicular to the detector faces. Firstly notable is the fact that both the location accuracy and – even more markedly – precision are significantly higher for PEPT-ML than for the Birmingham algorithm.

Secondly, and perhaps more surprisingly, we observe that the accuracy with which PEPT-ML locates the particles actually shows a statistically significant *increase* with the number of tracers used. A closer analysis of our results suggests that this arises due to the fact that additional noise created by the introduction of more particles acts to 'mask' the tear-drop shape of the particles' cutpoint clouds from the clustering agorithm, thus to an extent mitigating the apparent systematic errors discussed in section IV B.

In Figs. 11 and 12, we use simulations to extend our study to higher tracer numbers. These simulations show two striking results: firstly, well over 100 tracers can be successfully, simultaneously located. In fact, the upper limit of 128 particles was chosen based on limitations concerning available processing power to handle the large volumes of data involved, *not* the limitations of the PEPT-ML algorithm itself. Secondly, our results suggest a degree of scalability not achievable using conventional PEPT algorithms – across the full range ($N_T \in [2 - 128]$) of tracer numbers used, the location accuracy shows no statistically significant variation. In other words, using PEPT-ML, 128 particles can be located just as accurately as 2 particles.

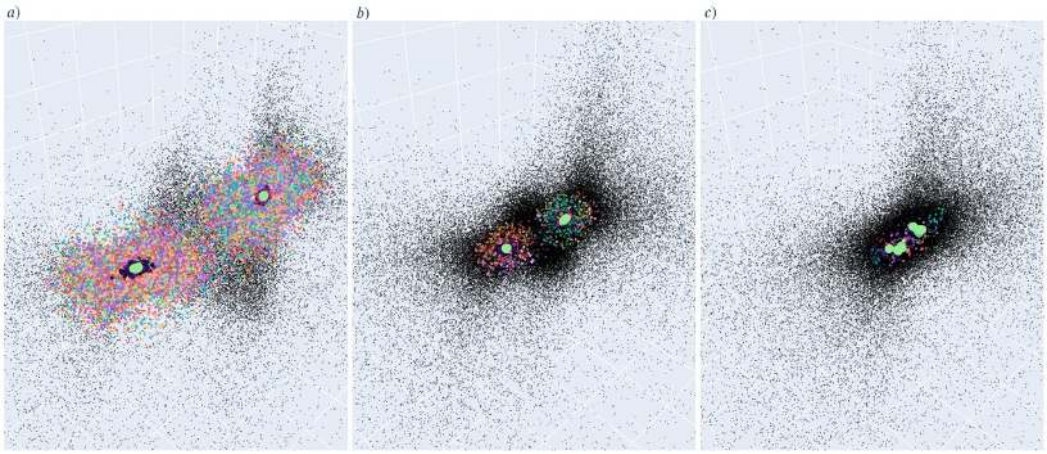

FIG. 8. Experimental data showing cutpoint clouds and located particle centres for pairs of 0.6mm alumina particles separated by distances of (a) 64 mm, (b) 16 mm and (c) 4 mm. In all cases, coloured cutpoints represent those included by the clustering algorithm, and black cutpoints those 'rejected'. The located centres are represented by larger, light green points. Note that for plot *c*), the same number of LoRs (100,000) were used as for *b*) and *c*), but with an increased overlap (380). This is to compensate for the very high rejection ratio in the 4 mm case.

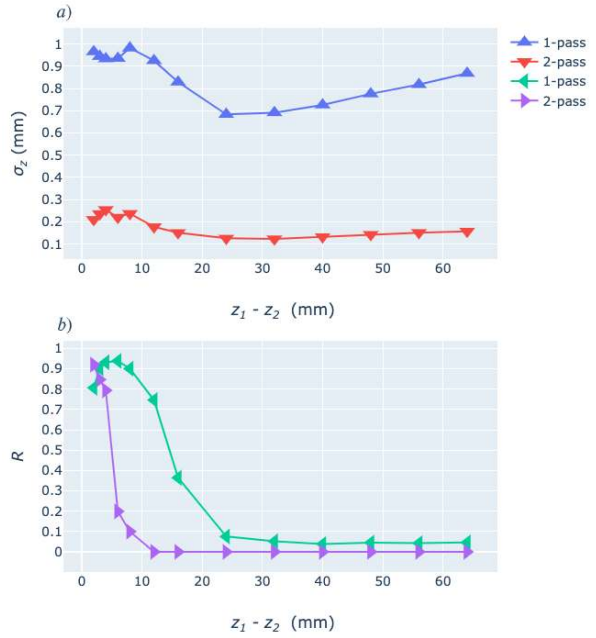

FIG. 9. a) Variation of the accuracy of location, $\overline{\Delta z}$, of a pair of 0.6 mm diameter, alumina particles as their separation, $|z_1 - z_2|$, is varied. Each data point represents the average of 1,000 individual particle locations, each using 100 lines of response, with the error bars shown representing the standard deviation of the measured locations, signifying the location *precision*. b) The variation with $|z_1 - z_2|$ of the rejection rate, R, i.e. the fraction of calculated midpoints not used by the clustering algorithm.

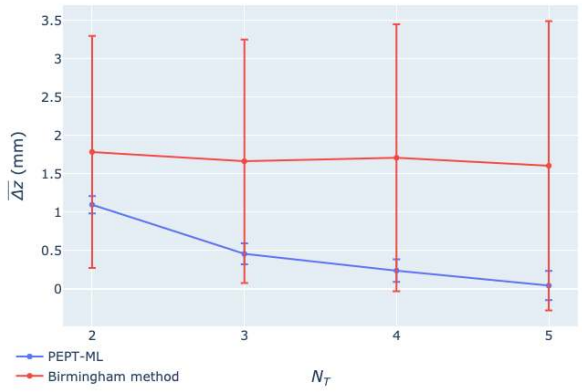

FIG. 10. Comparison between the accuracy $\overline{z}$ (mm) (denoted by colour-filled circles) and precision (denoted by the size of the error bars) of PEPT-ML with two-pass clustering and the Birmingham method for varying number of tracers N_T. Up to five 0.6 mm diameter alumina tracers were placed at 30 mm separation in the z-dimension. The accuracy represents the mean difference between the found and real separations. The precision represents the standard deviation of the centres found.

V. SUMMARY AND CONCLUSIONS

A novel algorithm, utilising advanced machine learning techniques, has been developed for the positron imaging of particulate and multiphase flows. The method is shown to facilitate high temporal and spatial resolution, the latter of which can be further improved through the use of a novel 'two-pass' clustering algorithm. The PEPT-ML algorithm can successfully distinguish particles separated by distances as small as 2 mm, a value smaller even than the inherent spatial resolution of the detector used in the present study.

The ability of the algorithm to successfully locate a particle using as few as 25 lines of response carries significant positive consequences for the application of PEPT not only to extremely rapid flows, but also for the miniaturisation of PEPT tracers, potentially opening the door to new biomedical applications of the technique, e.g. labelling and tracking blood cells.

The algorithm also includes novel methods for the identification of individual tracers and their tracks, which exploit the properties of the algorithms implemented to

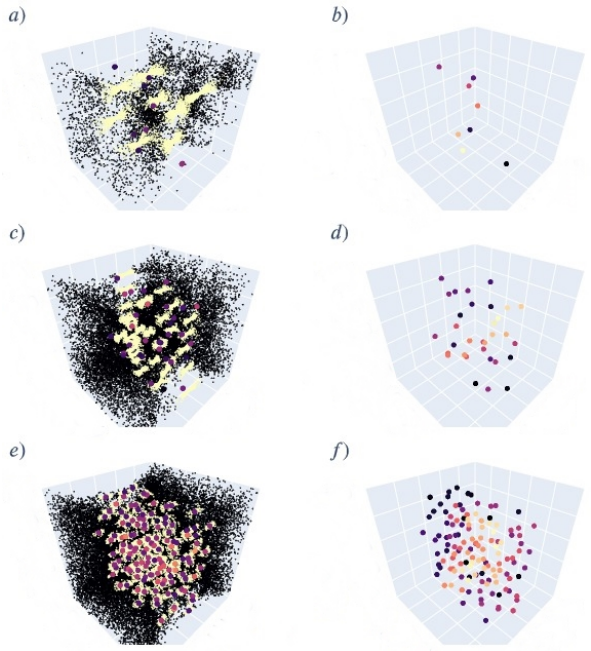

FIG. 11. Visualisation of the clustered cutpoints (left-hand column) and the centres found after the second pass of clustering (right-hand column) for 8, 32 and 128 simulated particles, respectively. On panels *a*), *c*) and *e*), the clustered cutpoints are shown as yellow small circles, the noise is depicted in black small circles, while the centres found after one pass of clustering are shown as purple large circles. Panels *b*), *d*) and *f*) show the centres of centres, where the cluster size has been colour-coded such that the purple (yellow) end of the spectrum represents smaller (larger) cluster sizes.

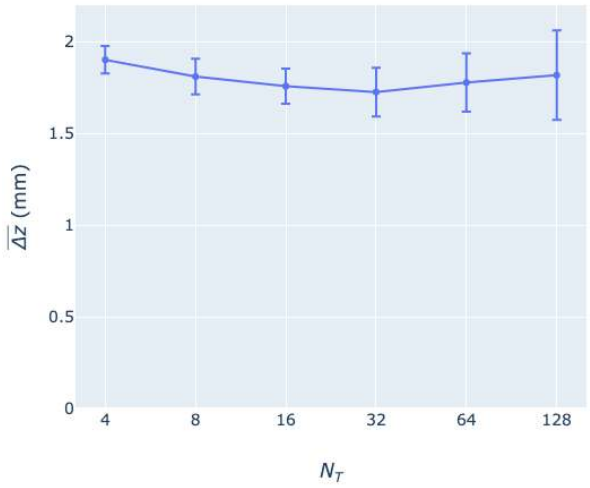

FIG. 12. Variation in accuracy (denoted by colour-filled circles) and precision (denoted by the size of the error bars) for increasing number of simulated tracers N_T. The accuracy represents the mean difference between the known, simulated positions and the found locations after two-pass clustering. The precision represents the standard deviation of the centres found.

distinguish particles based on their distinct 'signatures'.

Uniquely to this algorithm, the accuracy of the PEPT-ML algorithm is observed to remain effectively invariant with the number of tracers in the system, for tracer numbers of up to 128 – this number being limited not by the abilities of the algorithm itself, but by the computational power available to the authors. In the absence of such constraints, it is expected that this invariance will persist to still higher tracer numbers.

ACKNOWLEDGMENTS

This work was conducted thanks to support from the EPSRC Vacation Internship fund.

REFERENCES

[1] D. Parker, C. Broadbent, P. Fowles, M. Hawkesworth, and P. McNeil, Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment **326**, 592 (1993).
[2] D. Parker, R. Forster, P. Fowles, and P. Takhar, Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment **477**, 540 (2002).
[3] D. Parker, Review of Scientific Instruments **88**, 051803 (2017).
[4] D. J. Parker and X. Fan, Particuology **6**, 16 (2008).
[5] X. Fan, D. Parker, and M. Smith, Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment **562**, 345 (2006).
[6] D. Parker, T. Leadbeater, X. Fan, M. Hausard, A. Ingram, and Z. Yang, Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment **604**, 339 (2009).
[7] J. P. K. Seville, C. R. K. Windows-Yule, and D. J. Parker, To be published in the Annual Review of Chemical and Biomolecular Engineering (2019).
[8] Due to conservation of momentum, any pair of γ photons created by an annihilation event must follow approximately collinear, antiparallel trajectories – i.e. their paths always follow a straight line, which can be trivially reconstructed based on the locations of their intersections with the detectors of a positron camera.
[9] Z. Yang, D. Parker, P. Fryer, S. Bakalis, and X. Fan, Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment **564**, 332 (2006).
[10] Z. Yang, P. Fryer, S. Bakalis, X. Fan, D. Parker, and J. Seville, Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment **577**, 585 (2007).
[11] Note that particles with identical data logging rates may still both be *located* but will not be located in a known order, meaning that their *trajectories* cannot be distinguished based on this factor.
[12] M. Bickell, A. Buffler, I. Govender, and D. Parker, Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment **682**, 36 (2012).
[13] H. D. Vinod, Journal of the American Statistical association **64**, 506 (1969).
[14] O. Gundogdu and E. Tarcan, Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment **523**, 223 (2004).
[15] L. K. P. J. RDUSSEEUN, (1987).
[16] O. Gundogdu, Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment **534**, 562 (2004).
[17] C. Wiggins, R. Santos, and A. Ruggles, Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment **811**, 18 (2016).
[18] G. Hamerly and C. Elkan, in *Advances in neural information*

processing systems (2004) pp. 281–288.
[19]J. MacQueen *et al.*, in *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, Vol. 1 (Oakland, CA, USA, 1967) pp. 281–297.
[20]In the work of Wiggins, Santos, and Ruggles [17], a user-defined threshold for line density is applied in order to determine which voxels may and may not correspond to a true centroid.
[21]C. Wiggins, R. Santos, and A. Ruggles, Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment **843**, 22 (2017).
[22]J. C. Crocker and D. G. Grier, Journal of colloid and interface science **179**, 298 (1996).
[23]I. F. Sbalzarini and P. Koumoutsakos, Journal of structural biology **151**, 182 (2005).
[24]A. E. Odo, I. Govender, A. Buffler, and J.-P. Franzidis, Applied Radiation and Isotopes (2019).
[25]D. Blakemore, I. Govender, A. McBride, and A. Mainza, Chemical Engineering Science **207**, 780 (2019).
[26]D. Blakemore, *Multiple particle tracking in PEPT using Voronoi tessellations*, Master's thesis, School of Engineering, University of Cape Town (2016), available at https://open.uct.ac.za/handle/11427/22929.
[27]F. Aurenhammer, ACM Computing Surveys (CSUR) **23**, 345 (1991).
[28]Note that samples are allowed to overlap in time, as discussed in greater detail in section II C.
[29]R. J. Campello, D. Moulavi, and J. Sander, in *Pacific-Asia conference on knowledge discovery and data mining* (Springer, 2013) pp. 160–172.
[30]L. McInnes, J. Healy, and S. Astels, The Journal of Open Source Software **2**, 205 (2017).
[31]M. R. Palmer and G. L. Brownell, IEEE transactions on medical imaging **11**, 373 (1992).
[32]The asymmetric point clouds produced by the dual-headed camera arrangement, their origin, and their possible influence on the PEPT technique more broadly, will be discussed in detail in a future publication. For clarity and brevity, and to ensure a focused article, they are not discussed at length here.