University of Groningen

An efficient attribute-space connected filter on graphs to reconstruct paths in point-clouds

Babai, M.; Kalantar-Nayestanaki, N.; Messchendorp, J. G.; Wilkinson, M. H. F.

Published in:
Pattern recognition

DOI:
[10.1016/j.patcog.2020.107467](https://doi.org/10.1016/j.patcog.2020.107467)

IMPORTANT NOTE: You are advised to consult the publisher's version (publisher's PDF) if you wish to cite from it. Please check the document version below.

Document Version
Publisher's PDF, also known as Version of record

Publication date:
2020

[Link to publication in University of Groningen/UMCG research database](#)

Citation for published version (APA):

Babai, M., Kalantar-Nayestanaki, N., Messchendorp, J. G., & Wilkinson, M. H. F. (2020). An efficient attribute-space connected filter on graphs to reconstruct paths in point-clouds. *Pattern recognition*, 106, [107467]. <https://doi.org/10.1016/j.patcog.2020.107467>

Copyright

Other than for strictly personal use, it is not permitted to download or to forward/distribute the text or part of it without the consent of the author(s) and/or copyright holder(s), unless the work is under an open content license (like Creative Commons).

The publication may also be distributed here under the terms of Article 25fa of the Dutch Copyright Act, indicated by the "Taverne" license. More information can be found on the University of Groningen website: <https://www.rug.nl/library/open-access/self-archiving-pure/taverne-amendment>.

Take-down policy

If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

Downloaded from the University of Groningen/UMCG research database (Pure): <http://www.rug.nl/research/portal>. For technical reasons the number of authors shown on this cover page is limited to 10 maximum.

An efficient attribute-space connected filter on graphs to reconstruct paths in point-clouds

M. Babai^{a,*}, N. Kalantar-Nayestanaki^b, J.G. Messchendorp^b, M.H.F. Wilkinson^{c,*}

^a Centre for Information Technology, University of Groningen, P.O. Box 11044, 9700 CA Groningen the Netherlands

^b KVI-Center for Advanced Radiation Technology, University of Groningen, Zernikelaan 25, 9747 AA Groningen the Netherlands

^c Bernoulli Institute for Mathematics, Computer Science and Artificial Intelligence, University of Groningen, PO Box 07, 9700 AK Groningen, the Netherlands

ARTICLE INFO

Article history:

Received 13 August 2018

Revised 25 April 2020

Accepted 21 May 2020

Available online 23 May 2020

Keywords:

Graph-based image analysis

Attribute-space connectivity

Orientation-based segmentation

Irregular graph morphology

Graph morphology

Orientation-based path merging

Sub-atomic particle tracking

Straw tube tracker (STT)

ABSTRACT

Measurements by many multi-sensor systems can be considered as point-clouds. One such system is the tracker for the PANDA experiment. Charged particles passing through the tracker produce patterns representing their paths. We present a new, graph-based, attribute-space morphological connected filter for reconstructing particle paths through such a detector. We introduce the concept of attribute-spaces and attribute-space connected filters on graphs, rather than binary images and show a new processing scheme to reduce the size of the memory required to store the attribute-space representations of binary images and graphs. The result is an $O(N \log(N))$ algorithm with a total recognition error of approximately 0.10, a significant improvement compared to our previous state-of-the-art $O(N^2)$ algorithm with a total error of 0.17.

© 2020 Elsevier Ltd. All rights reserved.

1. Introduction

Many scientific applications require processing huge data sets produced by complex multi-sensor machines. The growth of data complexity, production rate and size makes the design of new algorithms inevitable as, in many cases, the conventional methods fail to meet the performance requirements. PANDA (anti-Proton ANnihilations at DARMstadt) is one of the experiments at the future Facility for Antiproton and Ion Research (FAIR), which is currently under construction in Darmstadt, Germany. It is designed to study the structure of the matter by the annihilation of antiprotons with protons at high interaction rates, up to 20 MHz. To cope with the high interaction rate, the data must be processed at run-time and reduced by a factor of about 10^3 . To meet the performance requirements, efficient algorithms need to be developed which are capable of reconstructing the particle tracks in-situ. The trajectories of the charged-particles are reconstructed using tracking detectors placed inside a magnetic field. The curvatures of the reconstructed tracks are used to obtain the physical parameters of the corresponding particles. Different methods are used to perform the path recognition task, among them the high precision

transformation-based (Riemann, Hough, conformal, etc.) methods mentioned in [1]. Many of these methods have high computational complexity, however. Therefore, there is still a need to introduce new and efficient algorithms to reduce the amount of required hardware and power consumption; leading to a greener computing architecture for PANDA. There are also other approaches [2] to meet the PANDA requirements, which are usually based on massive parallelization running on GPU hardware architectures.

Reconstructing the charged-particles tracks passing through the Straw Tube Tracker (STT) detector of PANDA involves recognition of thin, curvi-linear structures in a point-cloud obtained by projecting the firing STT nodes into the xy -plane. For segmentation of such structures in images, there are different approaches, based on path operators, proposed in previous studies [3,4]. Other solutions have been suggested in which connected components of an image are accepted or rejected based on the value of an attribute [5,6]. Morphological operators on graphs were introduced and studied in [7,8], where dilations and erosions were constructed and in combination with probes, one could define opening and closing operators. The majority of the proposed methods are not directly capable of grouping points in the image space into disjoint paths. Furthermore, they fail to segment overlapping structures into disjoint segments. The concept of connectivity in higher dimensional spaces, *attribute spaces*, was proposed by Wilkinson in [9] to solve these problems. It was shown that transforming the image into a

* Corresponding author.

E-mail addresses: m.babai@rug.nl (M. Babai), m.h.f.wilkinson@rug.com (M.H.F. Wilkinson).

higher dimensional space could theoretically be a solution to segment overlapping structures into disjoint segments.

In our previous work [10], we have applied the concepts of connectivity in attribute-spaces to process the images produced by the STT graph. We have introduced a first $O(N^2)$ morphological connected filter to reconstruct particle tracks through the STT graph with a total recognition error of approximately 0.17, which was shown to be better than the state of the art approach on single track images [10]. In this method, an orientation attribute-space was computed for a number of pre-defined orientations. Conceptually, the original graph was copied to each slice of the attribute-space, each slice corresponding to a given orientation. Every pixels was marked as active or inactive based on their contribution to that particular slice. This way, connectivity was defined by the nodes' neighbourhood relations and it was extended by the neighbourhood relations in the orientation-based attribute-spaces. This allowed splitting crossing paths into separate connected structures in attribute space. Furthermore, the STT graph was extended by "virtual nodes" to have a smoother transition through the layers with a different slope and also to estimate the z-coordinates of the space points.

In this work, we present the results of our improved algorithm with a computational complexity of $O(N \log(N))$ and a more compact and memory efficient representation of the attribute-spaces, by extension of the theory to graphs. This results in an improved track-recognition performance with a total recognition error of approximately 0.10; a significant improvement, compared to our previous work.

In this paper, in Section 2 the geometry of the STT subsystem is introduced. Section 3 will give a brief summary of the basic concepts of filtering and connectivity. In Section 4 we describe the extension of the attribute space and attribute space filters to graphs and we present our core method. Finally, in Section 5 we discuss the results and future works.

2. Geometry and structure of STT graph

One of the tracking detectors for the detection of charged-particles in the PANDA system is the Straw Tube Tracker (STT). It is mounted in a cylindrical volume around the beam-target interaction point and it contains in its present design 4224 straw tubes. The STT consists of two different types of tubes; the axial tubes, which are mounted parallel to the z-axis (antiproton-beam direction) and the skewed tubes which are skewed by a few degrees with respect to the z-axis ($+2.9^\circ$ or -2.9°). The latter are meant to estimate the z coordinates of the paths. All tubes have the same inner diameter of 10 mm and a length of 1400 mm except for a few shorter tubes mounted in the boundary regions of each skewed layer. Each tube could be considered as a spatially variant Point Spread Function (PSF). The axials provide an accurate position in the xy-plane and almost no information along the z-axis while the skewed provide a more accurate position along the z-axis and have a large position uncertainty in the xy-plane. The detector readout-system reads the tubes from one end. Fig. 1 shows a schematic view of the STT in the xy-plane and a 3D view of a part of the detector [1].

In order to have a realistic model of the particle interactions with matter and the hardware readout of the detector, a software package has been developed which provides functionality to generate simulated detector outputs [11] using a Monte Carlo (MC) method. For further details, we refer to the STT technical design report [1]. The STT detector is described by its geometry which contains the three-dimensional coordinates of the centre of each tube, the list of their direct neighbours and a direction-vector which indicates the slope of the tube in space. Because of the skewed straws, the tubes do not have equal number of neighbours, it varies

between 2 and 22. The detector structure could be considered as an irregular undirected 3D graph. The centre points of the tubes represent the vertices and the neighbour relations describe the edges. The structure of the graph provides accurate positions in the xy-plane, in the regions where axial (z-parallel) tubes are mounted. One can estimate a maximum displacement of ± 3 cm for the skewed tubes. The magnitude of the displacement depends on the length and direction of these tubes and the displacement is due to the fact that the position of hits, along the individual tubes, is not defined.

When a charged-particle travels through space in a magnetic field, it may hit a number of nodes and produce a signal while the remaining nodes remain off. The combined signals could be considered as a binary image of a number of curved, thin paths through the graph. The curvature and multiplicity of a path depend on the particle type, its momentum and the strength of the magnetic field. Fig. 2 shows an example of such paths. One can observe that there are gaps of one or more missing pixels; this effect is a consequence of detector inefficiencies.

To reduce the hit displacements, in the case of skewed tubes, and also to have an estimate of the z-coordinates of nodes in a path, we have introduced 10194 virtual nodes (virtuals). The virtuals are placed between the layers with different slopes, at the centre of the "virtual" intersection volumes between tubes with different slopes. Each of these nodes has exactly two neighbours. These nodes are turned on iff both neighbours are on. Fig. 3 illustrates the positioning of virtuals and Fig. 4 shows the grid after its extension by the virtuals.

3. Basic concepts of connectivity and filtering

Given a graph $G = (V, E)$, with V the set of vertices, and E the set of edges with $E \subseteq (V \times V)$. The set of edges E defines a neighbour relation on V . In line with binary morphology, in which any binary image X is represented as a subset of the image domain M , we can define a binary pattern X on graph G as a subset of the vertices V . Every such set X induces a subgraph $g = (v, e)$ of G , with $v = X$ and $e = (X \times X) \cap E$. The lattice $\mathcal{L}(G)$ associated with G in this way is simply the powerset of V denoted $\mathcal{P}(V)$, with set inclusion as the partial order. The least element is the empty set $\emptyset$, and the largest element is V itself. Note that this approach to binary patterns on graphs is less general than the morphology on graphs discussed in [8], but the restrictions make definition of attribute-space connectivity on these graphs easier.

One can define connectivity using the connectivity classes described in [9,12,13]. Every connectivity class $\mathcal{C} \subseteq \mathcal{L}(G)$ can be seen as the family of all connected elements of a lattice, or in this case, the family of all $X \subseteq V$ which induce a connected subgraphs of G . Usually, a graph (v, e) is considered connected if for all $a, b \in v$ there exists a sequence $x_1, x_2, \dots, x_n \subseteq v$ such that $x_1 = a$ and $x_n = b$ and all edges $e_i = (x_i, x_{i+1}) \in e$. In other words, there is a path from each vertex in the graph to each other. We can however use a more general notion of connectivity by defining different types of connectivity classes, e.g. by permitting gaps of a pre-defined number of edges in paths. For any connectivity class $\mathcal{C}$ on $\mathcal{L}(G)$ we have

- The empty graph $\emptyset \in \mathcal{C}$.
- Singletons $\{x\} \in \mathcal{C}$ for all $x \in V$.
- For any family $\{X_i\} \subseteq \mathcal{C}$, $\bigcap_i X_i \neq \emptyset$ implies $\bigcup_i X_i \in \mathcal{C}$.

as shown in [13]. Each set $X \in \mathcal{L}(G)$ consists of a set of connected components C_i , with i from an index set I . Connected components are connected subgraphs of maximal extent, i.e. $\{\forall C_i, \nexists (C \supset C_i) : C \subseteq X \wedge C \in \mathcal{C}\}$. At the node x , a binary connected opening Γ_x on $X \in \mathcal{L}(G)$ can be defined as

$$\Gamma_x(g) = \bigcup \{C \in \mathcal{C} \mid x \in C \wedge C \subseteq X\}. \quad (1)$$

Fig. 1. (left) a view of the STT in the xy -plane. The grey dots represent the axial tubes and the skewed tubes are shown in red and blue. (right) a 3D view of an assembled part of the STT detector. It was originally published on <http://www.fz-juelich.de/ikp/>. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Fig. 2. A front view of the STT detector in xy -plane; the nodes are shown in grey. (left) red points show the true hit positions of MC-generated particles (MC). (right) red points show the position of the reconstructed hits after digitisation. The event is composed of two generated μ^- particles, each with a momentum of 0.5 GeV/c. The magnetic field strength is set to 2 T. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

This returns the connected component to which $x \in V$ belongs, if $x \in X$, and $\emptyset$ otherwise.

One of the classes of the connected filters is the group of attribute filters. An attribute filter can be defined as:

$$\Gamma^T(X) = \bigcup_{x \in X} \Gamma_T(\Gamma_x(X)). \quad (2)$$

In which Γ_T is a trivial filter, which returns the input connected component if it meets a criterion T and returns an empty graph otherwise. Usually the criterion T is of the form $T(C) = \text{Attr}(C) \geq \lambda$; λ is a threshold value [14] and $\text{Attr}(C)$ is a real-valued attribute of C . In combination with our filter, one can use $\Gamma^T(X)$ to define filters for selecting or excluding paths based on certain values of attributes. For example, it would be easy to remove single-pixel connected components as noise, by using length as attribute.

4. Attribute-space connectivity

In [9] attribute-space connectivity and filters were introduced in the case of binary images. In this framework, a image $X \subseteq M$, with M the (typically 2D or 3D) image domain, is transformed into a higher dimensional space “attribute space” $(M \times A)$, with A the attribute dimension, which represents some local property of the image, such as width, or orientation. This is done by mapping every pixel $x \in X$ to one or more tuples of the form (x, a) with $a \in A$, by some operator Ω . It was shown that this approach allows splitting up overlapping structures in X into separate components in $\Omega(X)$. This is because multiple copies of a pixel at different attribute values in $M \times A$ can be far enough apart that they are not considered adjacent. These separated structures can then be filtered in the $M \times A$, after which the filtered results can be projected back onto M by the inverse operator Ω^{-1} . A drawback of this approach

Fig. 3. A schematic view of axial and skewed tubes. Axial nodes are represented in green and skewed tubes in blue; red dots represent the virtual nodes. The z and y coordinates of the virtual nodes are calculated from the intersection points of the skewed, projected onto the axial plane; their x coordinate is set to the middle of both planes. (top) shows a virtual point viewed from the xy -plane; (bottom) schematic view of virtuals in yz -plane. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

is the potentially huge extra memory requirement due to the mapping of a 2D binary image to 3D, or a 3D volume to 4D.

By representing binary images as patterns on a graph G , we can reduce this burden considerably. We now need to map G to a graph $G^A = (V^A, E^A)$ with $V^A = V \times A$, and $E^A \subseteq V^A \times V^A$. For convenience, each element of V^A is denoted $v_{ij}^A = (v_i, a_j)$, with i and j from index sets over V and A respectively. For every v_{ij}^A a_j is referred to as the node's *attribute value*. The edges in G^A are defined

as

$$E^A = \{(v_{ij}^A, v_{kl}^A) \in V^A \times V^A \mid (v_i, v_k) \in E \wedge \|a_j - a_l\| < D\}, \quad (3)$$

with D representing some threshold. This simply means that nodes in the new graph G^A are connected by an edge, if their counterparts in graph G are connected by an edge, and their attribute values are not too different. This defines a neighbour relation on V^A . The operator $\Omega : \mathcal{L}(G) \rightarrow \mathcal{L}(G^A)$ creates a set X^A by mapping every node v_i of binary pattern X to one or more nodes v_{ij}^A . This induces a new graph $g^A = (X^A, e^A)$, where

$$e^A = (X^A \times X^A) \cap E^A, \quad (4)$$

i.e., e^A contains all the edges in the neighbour relation between any pair of nodes in X^A . In this way we can split overlapping structures in X into separate connected components in $\Omega(X)$. The inverse operator Ω^{-1} projects $\Omega(X)$ back to X ($\forall X \in \mathcal{L}(G) : \Omega^{-1}(\Omega(X)) = X$). Now, the attribute-space connectivity class $\mathcal{A}$ on a graph G by the transformation pair (Ω, Ω^{-1}) and connectivity class $\mathcal{C}_{G^A}$ on the transformed graph can be defined as

$$\mathcal{A} = \{g \in \mathcal{L}(G) \mid \Omega(g) \in \mathcal{C}_{G^A}\}. \quad (5)$$

Using this and the transformation pair, an attribute-space connected filter $\Psi^A : \mathcal{L}(G) \rightarrow \mathcal{L}(G)$ can be defined as

$$\Psi^A(X) = \Omega^{-1}(\Psi(\Omega(X))), \quad (6)$$

with $g \in \mathcal{L}(G)$ and $\Psi : \mathcal{L}(G^A) \rightarrow \mathcal{L}(G^A)$ a connected filter on $\mathcal{L}(G^A)$. Using this framework, one can define various operator-pairs to transform a pattern X into the attribute-space where connected filters could be defined for segmentation.

4.1. Layer-based orientation attributes

Using the binary-image based attribute-space representation, an operator $\Omega_\alpha : \mathcal{P}(E) \rightarrow \mathcal{P}(E \times A)$, with $A = [0, 1, \dots, 180]$, representing 180 possible orientations, could be defined which assigns one or more orientations α_i to each active node (pixel) x in the binary image X [9]. The result is a binary image $f : (E \times A) \rightarrow \{0, 1\}$, which would be the indicator function of the set $\Omega_\alpha(X) \subseteq E \times A$. One can observe that the computed attributes define a new neighbourhood relation between the nodes, namely, an

Fig. 4. The structure of the graph after extension by virtuals. (left) STT graph in the xy -plane; (right) STT graph in the yz -plane. Virtuals are shown in magenta and the grey points represent the centre points of the tubes; the detector readout provides only the position of the grey dots in the z -direction, as they are shown in the right panel.

Fig. 5. An example of assigned orientation attribute to pixels of an image. For each attribute-value a new copy of the entire image is created. (left) original image; (middle) pixels with $\alpha = 90^\circ$; (right) pixels with $\alpha = 0^\circ$. One can observe that the central pixel is active in both orientations while the other pixels have only one orientation attribute.

orientation-based relation. For orientation-based attribute-space, we can rewrite Eq. (3) as follows:

$$E^A = \{(v_{ij}^A, v_{kl}^A) \in V^A \times V^A \mid (v_i, v_k) \in E \wedge 180 - |a_j - a_l| - 180| < D\}. \quad (7)$$

For example, the nodes receiving orientation attribute-values 0, 1 or 179 are considered to be neighbours in the orientation attribute-space.

A naive implementation would simply create $\#A = 180$ copies of the 2D array storing the original image X , and set the pixels $x \in \Omega_\alpha(X)$ to one. An example of a few slices through such a volume is shown in Fig. 5.

In principle this same approach can be made in the graph-based case by simply making 180 copies of the graph G , add all allowable edges between the slices and set all appropriate nodes to active. This is a waste of memory, because many nodes receive just one orientation, and only a handful get more. Worse, the naive approach would copy all inactive nodes as well. This has a huge impact on computational complexity.

To cope with the undesirable memory expansion, we determine these attributes in a layer-based manner, by starting from the outer-most layer and moving toward the inner-most layer; at each stage, we only consider the active nodes in the current or the next layer (look-ahead layer). The nodes are copied as needed for each actually computed attribute-value. Because each node participates in a few levels in the attribute dimension, the numbers of copies remain small and cause a negligible memory expansion. This method consists of the following steps:

- Fetch all active nodes in the current layer and the list of their neighbours. The compiled list L contains only active and not processed nodes in the current or the next layer.
- For all axial nodes (parallel to z axis) in L determine α_i . Create new attribute, if not existing or update otherwise. If α_i was not observed before, create one and add a copy of the node to the list of participants of that attribute-space. Otherwise, the list of participants is updated by adding the new nodes.
- For all virtual nodes, follow arcs via neighbours to find the next active virtual and determine α_i . Create new attribute, if not existing or update otherwise. Visiting a virtual indicates that we are passing layers with different slopes. In those regions the number of neighbours change and the position precision in the xy -plane decreases dramatically. We skip the layers between the virtual to reduce the number of possible orientations.
- Add skewed neighbours if they fit on the line through the virtual nodes.

The resulting method is shown in Algorithm 4.1. Because of the large position uncertainty in the skewed layers, as described in Section 2, computing the orientations using these nodes lead into local sharp turns, and consequently discontinuity in the reconstructed paths in the attribute-space. Therefore, there are no

Algorithm 4.1 Compute attribute space layer-based.

Input: ($ListOfAllNodes$, tol)

```

1: Init a FIFO  $Q$  and insert all active nodes; determine min and max layer.
2: while  $\neg$  empty( $Q$ ) do
3:   Compile the list of all active nodes in current layer,  $LA$ . Delete from  $Q$ .
4:   while  $\neg$  empty( $LA$ ) do
5:     Fetch first element  $CurrentNode$  and the list of neighbours  $ListOfNeighbors$ .
6:     if  $CurrentNode$  is parallel node then
7:       for all  $neighborNd \in ListOfNeighbors$  do
8:         if Active and  $\neg$  Skewed( $neighborNd$ ) then
9:           Compute slope if neighbour in current or next layer.
10:          if Orientation exists for  $CurrentNode$  then
11:            Add neighbour node to member list.
12:          else Create new attribute, add to attribute list of  $CurrentNode$ .
13:          end if
14:          if Orientation exists for  $neighborNd$  then
15:            Add  $CurrentNode$  to the members list.
16:          else Create new attribute and add to neighbour node.
17:          end if
18:        end if
19:      end for
20:    end if
21:    if  $CurrentNode$  is Virtual then
22:      Remove from  $Q$ 
23:      Follow neighbourhood arcs to next virtual and compile list of skewed  $SL$ .
24:      Determine  $\alpha_i$  for current and next virtual node.
25:      if Attribute exists then
26:        update attribute.
27:      else Create new one.
28:      end if
29:      for all  $skNode \in SL$  do
30:        if  $skNode$  fits on the line with slope  $\alpha$  then
31:          Add  $skNode$  to attribute  $\alpha_i$ 
32:        end if
33:      end for
34:    end if
35:  end while
36:  Go to next layer.
37: end while

```

attributes computed for these nodes and they are added to the list of nodes of the attribute list of other neighbouring nodes; one can consider this as a local smoothing operation. This way, all skewed nodes are considered to be singletons; hence, they have no directional preference. This processing scheme for the skewed tubes makes it unnecessary to set a value for the threshold D as it was introduced in Eq. (3). In the regions where the parallel tubes are mounted, the threshold is determined by the geometry of the detector.

4.2. Connected components analysis

Computing the orientation attribute for a binary image X using $\Omega_\alpha : \mathcal{L}(G) \rightarrow \mathcal{L}(G^A)$, yields an image with an additional dimension, namely the orientation-based dimension. The value of this parameter determines the orientation(s) in the space in which the current pixel participates and its magnitude defines the strength of the preferred orientation for a given pixel. Using these properties, one can extend the neighbourhood relations between the graph vertices by a weighted orientation-based relation. Using the neighbourhood relations and the newly computed parameters, one can define connected operators which follow the neighbouring arcs and step through the orientation spaces to collect participating points in order to compile a list of nodes for every connected component (particle path candidate). Note that in this case, a connected component is not necessarily a set of path-connected, neighbouring pixels. Instead, it is any set of pixels which is member of the connectivity class in use. This might, e.g., allow tracks with gaps to be considered as connected components, as in the case of mask-based connectivity [15]. Here, we have applied the following method:

1. Initialise a priority queue Q (FIFO) and insert all active nodes. The nodes are inserted from outermost to innermost layer.

2. Initialise an empty list to hold path candidates.
3. For all nodes $n \in Q$:
 - (a) If visited for the first time and non-virtual (virtual nodes cannot start a new path), Create a candidate for each orientation attribute. Note that singletons have an empty attribute list, hence they have no directional preference and the generated path has length one.
 - (b) If the node was visited before, find path candidates in which it was added before. Add members of the attributes with equal value to their member list. For all other attributes, create a new path candidate.

Note that skewed nodes have no orientation attributes and they can only appear in member lists of attributes of other nodes. This procedure is shown in [Algorithm 4.2](#).

Algorithm 4.2 Attribute space connected component.

Input: The graph of the tubes, minimum required response.

Output: List of sets “AllComplList” containing all connected components.

```

1: Init an empty queue Q and insert all activenodes. ▷ Nodes are added in
  a layer based ordering
2: while ¬ empty(Q) do
3:   Fetch node CurrentNode and the list of its attributes orientList
4:   if Not visited(CurrentNode) & (not virtual) then
5:     Increment times visited.
6:     if Singleton(CurrentNode) then
7:       Create Singleton candidate. Add to AllComplList.
8:     else
9:       for all (p ∈ orientList) do
10:        Create a new tracklet(cnd) and set its orientation to value
11:        of p
12:        Add CurrentNode to cnd members set.
13:        Add all members of p to cnd members set and mark them
14:        as visited.
15:        Add cnd to AllComplList.
16:      end for
17:    end if
18:  else if visited(CurrentNode) & (not virtual) then
19:    Find the list of path candidates containing CurrentNode, CandList.
20:    for all p ∈ orientList do
21:      Find cand ∈ CandList, with (p.α ≈ cand.α)
22:      if cand exists then
23:        Update Cand and mark members of p as visited.
24:      else
25:        Create new path candidate.
26:      end if
27:    end for
28:  end if
29: end while
30: return AllComplList
  
```

Application of [Algorithm 4.2](#) to images produced on the STT graph, lead into a number of (disjoint) sub-paths. The broken paths occur mainly when a track passes through the skewed detectors. [Fig. 6](#) shows a number of reconstructed tracklets for two different events.

4.3. Merging tracklets

Hierarchical segmentation methods have been an important tool for partitioning sets and are discussed in previous studies on partition tree based methods [16–20]. Here, we are interested in merging compatible partitions of the produced images to reconstruct a full particle path in 2D (segmentation reduction) and afterwards estimating the z-coordinates for each connected component. One possibility to reduce the over-segmentation is to merge the tracklets in a post-processing step. In this case, the orientation compatibility of the tracklets is one of the important criteria to select candidate pairs to merge. Determining the curvature and orientation of each path by fitting a curve through its set of points, is an obvious choice. The curve fitting procedures are computationally expensive and complex; using those methods, one suffers from numerical instability and convergence issues [21,22]. Those methods are of interest when the aim is to reconstruct momentum with the best possible precision. To avoid fitting, we propose

using a distance measure that takes into account the direction of the spread of the point-cloud in the space. One such a distance is the Mahalanobis distance of a point to a cloud of points.

One can use this measure to define the quality of a possible merge of two or more tracklets. Denoting Mahalanobis distance by D_M , define $D_M(X_i, X_j)$ to be D_M of the end-node of X_i with the shortest Euclidean distance to X_j ; we evaluate $D_M(X_i, X_j) = \max(D_M(X_i, X_j), D_M(X_j, X_i))$ for all selected pairs of sub-tracks with $i \neq j$; pairs of sub-tracks are pre-selected based on the Euclidean distances of their head-nodes. The pair with the global minimum of D_M is selected to be merged. The details of this method are shown in [Algorithm 4.3](#).

Algorithm 4.3 Mahalanobis merge.

Input: (ListOfAllCandidates, distThreshold)

```

1: Initialise an empty list to hold the merged candidates, mergedList.
2: for all cand ∈ ListOfAllCandidates do
3:   mpairst = candidate merge pairs(cand) ▷ Pre-selected on Euclidean
4:   distance(alg. 4.4).
5:   Compute Mahalanobis distance for all pairs.
6:   sort mpairst ▷ To find minimum.
7:   Select the best match ▷ Pair with minimum D_M.
8:   Create new path candidate mergedCand containing the selected track-
9:   lets.
10:  Add mergedCand to mergedList
11: end for
12: Extend ListOfAllCandidates by mergedList
  
```

In [Algorithm 4.4](#), the method “numEndStopsIfMerged” determines the number of tails of the resulting tracklet, if the selected sub-paths are merged and “selectCandidatesMatchNodes” determines which combination of end-nodes of the selected tracklets must be used to evaluate the quality of a possible merge. Furthermore, the list “AllSelectedTracklets” contains the list of all tracklets with their head or tail node in the vicinity of the head or tail node of the input tracklet; we compile the list using the following scheme:

- Select both end-nodes of the input node.
- Select all direct neighbours and their neighbours of the both end-nodes. The selected nodes must lie within a given Euclidean distance and their layer difference with the current node is smaller than a pre-specified number of layers.
- Select all active nodes in the compiled list if they are one of the end-nodes of a valid tracklet.
- Add tracklet to “AllSelectedTracklets”.

Algorithm 4.4 candidate merge pairs.

Input: (inputCand, AllSelectedTracklets, distThreshold)

Output: List of possible merge pairs, OutputPairs

```

1: Fetch ipH = inputCand.head and ipT = inputCand.tail
2: for all c ∈ AllSelectedTracklets do
3:   if ¬(inputCand = c) then
4:     Fetch cH = c.head and cT = c.tail
5:     minDist = min(dist(ipH, cH), dist(ipH, cT),
6:       dist(ipT, cH), dist(ipT, cT))
7:     endsCount = numEndStopsIfMerged(inputCand, c)
8:     selectCandidatesMatchNodes(inputCand, c)
9:     if (minDist ≤ distThreshold) then
10:      Add pair (inputCand, c) to OutputPairs
11:    end if
12:  end for
  
```

This scheme limits the search area for the possible merge candidate tracklets. The allowed layer difference and Euclidean distance are a measure for the accepted gaps between the selected tracklets.

Considering the reconstructed tracklets as shown in [Fig. 6](#), we have applied the method described in this section in a step wise

Fig. 6. Constructed tracklets after application of Algorithm 4.2 on μ^- events with a momentum of 0.5 GeV/c. Each colour and marker combination represent a different connected component. One can observe that the tracklets often break, when passing through the layers with skewed nodes.

Fig. 7. A schematics view of merging process. At each level two or more tracklets could be joined to create a single larger track. A merge operation is performed if the tracklet compatibility criterion is fulfilled.

and hierarchical manner with an increasing value for the pair-selection distance and gap-size thresholds. This way, the set of all tracklets could be considered as a dendrogram; at each merging step one or more internal nodes are generated (if acceptance criterion is met) and receive the label of that particular merging level. Fig. 7 shows a schematic view of merging steps.

In this case, we have empirically determined that merging tracklets in a number of steps with an increasing distance-threshold (≤ 6) and gap-size (≤ 3) leads to results with the smallest error. Fig. 8 shows the result of the application of this method to events shown in Fig. 6.

In the left panel of Fig. 8, we can observe that the track in the lower left corner is not merged correctly; this is due to the large Mahalanobis distance of the isolated points to the nearest tracklet.

4.4. Computational complexity

In this section, we argue the estimated computational complexity of our proposed algorithm, albeit we do not provide a formal proof.

Given an instance of STT read-out with a total number of nodes equal to $T = N + B$, one can divide this in two subsets of nodes, namely the foreground (N) and the background (B) nodes. The attribute space is computed in a layer-based order. This way, active nodes in each layer are visited at most equal to the number of their active neighbours ($O(N)$). In the worst case, construction of the tracklets leads into N sub-tracks, singletons, ($O(N)$). After each merge, the total number of available sub-tracks decreases; during each merging step, only the head nodes of each tracklet are visited. Each tracklet has exactly two end-points. Hence, as the length of

tracklets grows by merging, the number of possible nodes to visit for the next merging step can only decrease. Accounting for all the steps gives an algorithm with an average computational complexity $O(N)$. At each merge step the distances between all pre-selected tracklet-pairs are determined. The pairs are selected based on the neighbourhood relations of the end-nodes limited by a distance and a gap-size threshold. The tracklet pairs are sorted according to their distances to find the best merge candidates; therefore Algorithm 4.3 has a computational complexity of $O(N \log(N))$, as in the worst case the first merging requires sorting N tracklets (singletons). In summary, we can evaluate our method to be of $O(N \log(N))$ complexity.

5. Results and discussion

In this section, we present the results of the application of our method to data sets generated using the PANDA simulator¹. We have generated 10^4 negatively charged muon (μ^-) events with a range of different plausible momenta up to 2 GeV/c and a multiplicity of six tracks per event. Although there are various particle types that are produced in antiproton-proton annihilations, we have chosen to take muons since these particles are electrically charged leading to clear signatures in the STT. Moreover, the lifetime of muons is large enough such that the production of secondary tracks inside the detector is very unlikely. The interaction point is set to (0,0,0) and all tracks are generated isotropically around the origin.

To measure the segmentation error, we have used a modified version of the measures described in [23]. It is the same method as we applied in [10]. It is defined as: given an image X with a total area of A , let a reference segmentation divide the image into N regions $\{R_1, \dots, R_N\}$ with corresponding areas $\{A_1, \dots, A_N\}$. An example segmentation finds M regions $\{T_1, \dots, T_M\}$ with areas $\{a_1, \dots, a_M\}$. For all regions T_j , we find R_{k_j} such that $(T_j \cap R_{k_j})$ is maximal. One can define the “under-merging” (E_{um}), “over-

¹ The software package “pandaroot” is available from: subversion.gsi.de (rev: 26379). To make this analysis reproducible, we have fixed the starting seed.

Fig. 8. Constructed tracklets after application of merging to the events shown in Fig. 6. Units along both axis are in centimetres. Each colour and marker combination represents a different connected component.

Fig. 9. Distribution of the total error for 10^4 events with a multiplicity of six tracks per event. Particles are generated with a momentum of 0.8 GeV/c. (left) linear scale; (right) log scale. Log scale plots are included to show more detailed structures in these plots.

merging" (E_{om}) and normalised total error (E_{tot}) as:

$$E_{um} = \frac{1}{A} \sum_{j=1}^M \frac{\{A_{k_j} - \#(T_j \cap R_{k_j})\} \#(T_j \cap R_{k_j})}{A_{k_j}}, \quad (8)$$

$$E_{om} = \frac{1}{A} \sum_{j=1}^M \{(a_j - \#(T_j \cap R_{k_j}))\}, \quad (9)$$

$$E_{tot} = \sqrt{E_{um}^2 + E_{om}^2}, \quad (10)$$

in which $\#C$, denotes the cardinality of C . Here, we consider the true Monte Carlo (MC) paths as the reference segmentation to determine the two error-types. An example of the distribution of total error for particles with a momentum of 0.8 GeV/c is shown in Fig. 9 and the mean values of different error types as function of different momenta is shown in Fig. 10.

One can observe that the value of the errors show the same trend for both methods and it is significantly improved with our

new algorithm. This improvement is mainly achieved by track-merging steps. The tracklets are merged hierarchically based on their geometrical compatibility, followed by merging based on Mahalanobis distance as a measure for the quality of the merge. Furthermore, we allow gaps between the tracklets that are candidates for merging 4.3. This allows merging broken paths as a consequence of readout imperfections and contributes to the reduction of the error. The error is the largest for slower particles (lower momentum). Particles with a lower momentum have a stronger interaction with the magnetic field and hence their paths are more curved while particles with a higher momentum (moving faster) will travel on a straighter path. The high curvature tends to break the paths and together with gaps between the tracklets, they are the major contributor to the under-merging error. In practice gaps can occur when neighbouring tubes are missing as a consequence of readout imperfections, but certainly when the paths cross the space between the different sets of tubes; the latter spaces are visible in Fig. 1. There is a large gap between the left and right halves of the detector, and there are smaller gaps between the

Fig. 10. Error estimates for $10^4 \mu^-$ events with a multiplicity of six tracks per event. The results of our previous method are marked as “old”.

Fig. 11. (left) the detector read-out. (right) reconstructed paths in xy -plane. Each colour and marker combination represents a different connected component. Under-merging is visible in the top left corner and the lower left tracks show an example of over-merging.

skewed tubes from different sectors. Over-merging is observed mainly when two or more paths run through direct neighbouring nodes or when passing through near neighbours in skewed layers. Fig. 11 shows an event where both error types occur. During the reconstruction of the tracklets, we accept no gaps between the tubes (missing pixels); allowing gaps can lead to a smaller under-merging error, although it may cause a more frequent over-merging problem. A portion of under-merging problem is solved by our tracklet-merging method, where gaps are allowed if tracklets are compatible. An example was shown in Fig. 8.

Considering the available STT geometry data and readout parameters, the results of our method in recognising particle paths in 2D are within the 10% error margin. Note that in STT-TDR [1], the results are based on full reconstruction of the particle paths for single particle events. This means that the paths are fully reconstructed in three dimensions and fitted to determine momenta with the best possible resolution. Here, we reconstruct paths in two dimensions for events with a multiplicity of six particles per event and currently are not determining momentum. These differences make a one-to-one comparison of the results impossible.

Fig. 12. Distribution of the total error for 10^4 single track μ^- events. The particles are generated with a momentum of 0.8 GeV/c. (left) linear scale; (right) log scale. Log scale plots are included to show more detailed structures in these plots.

Table 1

Error estimates for 10^4 single track μ^- events. The momentum is in units of GeV/c.

Momentum	0.4	0.5	0.8	1.0	1.5	2.0
E_{tot}	0.048	0.041	0.036	0.036	0.037	0.036

In order to compare our results to the STT-TDR, we have generated 10^4 single track muon events. Since each event contains only a single track, the total error and the under-merging error are equal, consequently the over-merging error is equal to zero. The mean error-values for single track events are listed in Table 1 and an example of the distribution of the total error is shown in Fig. 12.

One can observe that the same trends holds for the single track events; the larger error values appear for the lower momentum tracks.

When reconstructing paths, it is important to estimate the effect of missing pixels, when paths are partially recovered due to over- or under-segmentation. We have used the curvature of the reconstructed paths as an indicator to qualify the goodness of reconstructed tracks. We define the curvature as $1/r$, where r is the radius of a circle fitted to the points participating in a path with the smallest least square error [22]. The error is evaluated using the following scheme:

- For each path R_k , find T_j such that $(T_j \cap R_k)$ is maximal.
- Determine curvature for both R_k and T_j .
- Define $D_c = (|\frac{1}{r_{MC}} - \frac{1}{r_{trk}}|)$; here, r_{MC} is the radius of the circle fitted to R_k and r_{trk} belongs to the corresponding T_j .
- For all error thresholds t , determine the fraction of track pairs with $D_c < t$; t is in units of cm^{-1} .

We have evaluated D_c for 10^4 single track events for a set of thresholds $t \in [0, 2]$. The results are shown in Fig. 13.

One can observe that for fairly small threshold values ($t < 0.25 cm^{-1}$), our method reaches $> 74\%$ correct recognition of the input tracks; although we cannot directly use this value to improve the results of our method, it can be considered as a measure for reliability of our proposed algorithm.

Since the radius of the circle fitted through the space points of each particle path is inversely proportional to the transverse momentum of the particle, one can use this property to estimate the transverse momentum resolution achieved by our method. We de-

Table 2

The error in the estimated momentum (ΔR) for 10^4 single track μ^- events. The momentum is in units of GeV/c and ΔR and R_{MC} in cm.

Momentum	0.4	0.5	0.8	1.0	1.5	2.0
$\Delta R/R_{MC}$	0.07	0.08	0.09	0.09	0.11	0.11
RMS	0.24	0.24	0.24	0.23	0.21	0.21

fine the error in the estimated radius as $\Delta R = R_{MC} - R_{Tr}$, where R_{MC} and R_{Tr} denote the radius of the circle passing through the Monte Carlo and the reconstructed points, respectively. We have evaluated $\Delta R/R_{MC}$ for a number of different momenta. The mean values are summarised in Table 2. The distribution of $\Delta R/R_{MC}$ shows the same pattern for all selected momenta; as an illustration, Fig. 14 shows the distribution of $\Delta R/R_{MC}$ for particles with a momentum of 0.4 and 2.0 GeV/c.

We observe that the mean value of the estimated differences is positive for all momenta. This means that our method systematically underestimates the transverse momentum. It is the smallest for low-momentum particles and increases for particles with a higher momentum. This effect could be partially explained by the fact that faster moving particles create straighter paths. This leads into a less accurate estimation of the radius of the fitted circles. Furthermore, we need to mention that we have excluded the extreme outliers during this analysis; those are most probably produced because we use a primitive and fast curve-fitting method which obviously does not produce accurate values for R ; using a more sophisticated fitting method will improve the results; this is maybe a task for future work, but it is beyond the scope of this manuscript.

Although we still cannot perform a one-to-one comparison of our results to the ones shown in the STT-TDR, one can observe that the method used in the STT-TDR estimates the momentum with a higher precision and is more stable in all momentum regions. These differences were expected as the method used in the STT-TDR is designed to achieve the highest precision when reconstructing the particle momenta. While our method is designed for low computational complexity and suit online data-reduction and event pre-selections. Furthermore, our method uses the geometry of the detector exclusively, and we are neglecting other parameters such as the drift-time. Hence, our method has the advantage of not depending on exact time calibrations.

Fig. 13. Completely matched tracks for 10^4 single track μ^- events as a function of error threshold value (along x-axis) for events with a momentum of 0.4, 0.5, 0.8, 1.0, 1.5 and 2.0 GeV/c.

Fig. 14. The distribution of $\Delta R/R_{MC}$ for 10^4 single track muon (μ^-) events with a momentum of 0.4 and 2.0 GeV/c. (left) linear scale; (right) log scale. Log scale plots are included to show more detailed structures in these plots.

5.1. Conclusion

We have presented an $O(N \log(N))$ algorithm with a total error of approximately 0.10. The results are significantly improved compared to our previous work [10], where we presented an $O(N^2)$ algorithm with an error of 0.17. Furthermore, we have introduced the concept of attribute-spaces and attribute-space connected filters on graphs, rather than binary images. In addition, we have shown a new processing scheme to reduce the size of the memory required to store the attribute-spaces representations of binary images and graphs. This has been achieved by copying the nodes as needed for every actual attribute-value observed. This property makes our method suited for design and application of more complex operators using attribute-vectors to process higher dimensional and larger images. Furthermore, the on-demand copying of the nodes makes it unnecessary to use predefined and discrete attribute-values; this is a powerful feature at the cost of slightly more complex code.

However, our algorithm will need further optimisation to meet the actual requirements for online data-processing for PANDA. Fu-

ture work will consist of exploration of advanced data structures in combination with enabling non-uniform merge or segmentation criteria per path-construction level and in addition lowering the execution time and the size of required memory to suit online data processing applications. Furthermore, extending the existing theoretical base of graph-based attribute-spaces and attribute-space connected filters to cover general cases and hyper-graphs is an important future task. One can assign a single attribute to each edges or allow different attribute-value per edge end-point. This extension will allow processing data from generic graphs of an arbitrary dimension and neighbourhood relation between the nodes. Investigating the effects of assigning attribute values to the edges is another important topic for future work. Combining attributes computed in a transformed domain with the attributes determined in the original image domain is another important future work to expand capabilities of our method. A measure for the curvature can be introduced in the distance using discrete approximations; investigating the effect of this introduction is another step for our future work. Designing methods to deal with the secondary paths produced by decaying particle is another important exten-

sion of our method that needs further investigation. Improving the z-coordinate estimation for paths is another important issue that needs an in-depth investigation. Modifying our method to measure the same entities as the ones presented in the STT-TDR is another important step to take; this will allow for a one-to-one comparison between our algorithm and the ones presented in the STT-TDR. Finally, testing our algorithm using a more realistic event generator is an important step to be taken.

Declaration of Competing Interest

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

Acknowledgements

We thank the PANDA collaboration for their support and special thank to the STT developers group. We thank the anonymous reviewers for many suggestions to improve the paper.

References

- [1] P.C.W. Erni, I. Keshelashvili, B. Krusche, et al., Technical design report for the panda straw tube tracker, *The European Physical Journal A* 49 (2013) 25. Special Article - Tools for Experiment and Theory
- [2] L. Bianchi, A. Herten, J. Ritman, T. Stockmanns, A. Adinets, J. Kraus, D. Pleiter, Online tracking algorithms on GPU for the panda experiment at fair, *J. Phys. Conf. Ser.* 664 (8) (2015) 82006.
- [3] M.B. Kurdy, D. Jeulin, Directional mathematical morphology operations, in: *Proceedings of the Fifth European Congress for Stereology*, 8/2, 1989, pp. 473á–480.
- [4] H. Talbot, B. Appleton, Efficient complete and incomplete path openings and closings, *Image Vis. Comput.* 25 (4) (2007) 416–425.
- [5] M.A. Westenberg, J.B.T.M. Roerdink, M.H.F. Wilkinson, Volumetric attribute filtering and interactive visualization using the max-tree representation, *IEEE Trans. Image Proc.* 16 (2007) 2943–2952.
- [6] M.H.F. Wilkinson, M.A. Westenberg, Shape preserving filament enhancement filtering, in: W.J. Niessen, M.A. Viergever (Eds.), *Proc. MICCAI'2001, Lecture Notes in Computer Science*, 2208, 2001, pp. 770–777.
- [7] H.J.A.M. Heijmans, P. Nacken, A. Toet, L. Vincent, Graph morphology, *J. Vis. Commun. Image Represent.* 3 (1992) 24–38.
- [8] J. Cousty, L. Najman, F. Dias, J. Serra, Morphological filtering on graphs, *Computer Vision and Image Understanding* 117 (4) (2013) 370–385, doi:10.1016/j.cviu.2012.08.016. Special issue on Discrete Geometry for Computer Imagery
- [9] M.H.F. Wilkinson, Attribute-space connectivity and connected filters, *Image Vis. Comput.* 25 (2007) 426–435.
- [10] M. Babai, N. Kalantar-Nayestanaki, J.G. Messchendorp, M.H.F. Wilkinson, Tracking sub-atomic particles through the attribute space, in: *Proc. Int. Symp. Math. Morphology (ISMM)* 2015, Springer, Cham, 2015, pp. 86–97, doi:10.1007/978-3-319-18720-4_8.
- [11] S. Spataro, the PANDA Collaboration, The pandaroot framework for simulation, reconstruction and analysis, *J. Phys. Conf. Ser.* 331 (3) (2011) 32031.
- [12] C. Ronse, Set-theoretical algebraic approaches to connectivity in continuous or digital spaces, *J. Math. Imag. Vis.* 8 (1998) 41–58.
- [13] J. Serra, Connectivity on complete lattices, *J. Math. Imag. Vis.* 9 (3) (1998) 231–251.
- [14] E.J. Breen, R. Jones, Attribute openings, thinnings and granulometries, *Comp. Vis. Image Understand.* 64 (3) (1996) 377–389.
- [15] G.K. Ouzounis, M.H.F. Wilkinson, Mask-based second generation connectivity and attribute filters, *IEEE Trans. Pattern Anal. Mach. Intell.* 29 (2007) 990–1004.
- [16] P. Soille, Constrained connectivity and connected filters, *IEEE Trans. Pattern Anal. Mach. Intell.* 30 (7) (2008) 1132–1145.
- [17] G.K. Ouzounis, P. Soille, The Alpha-Tree Algorithm, Publications Office of the European Union, 2012, doi:10.2788/48773.
- [18] M.Á. Vezanones, M. Mura, G. Tochon, J. Chanussot, Binary partition trees-based spectral-spatial permutation ordering, in: *Proc. Int. Symp. Math. Morphology (ISMM)* 2015, Springer, Cham, 2015, pp. 434–445, doi:10.1007/978-3-319-18720-4_37.
- [19] P. Salembier, Study of binary partition tree pruning techniques for polarimetric SAR images, in: *Proc. Int. Symp. Math. Morphology (ISMM)* 2015, Springer International Publishing, Cham, 2015, pp. 51–62, doi:10.1007/978-3-319-18720-4_5.
- [20] J. Serra, B.R. Kiran, Constrained optimization on hierarchies and braids of partitions, in: *ismm2015*, Springer, Cham, 2015, pp. 229–240, doi:10.1007/978-3-319-18720-4_20.
- [21] D. Umbach, K.N. Jones, A few methods for fitting circles to data, *IEEE Trans. Instrum. Meas.* 52 (6) (2003) 1881–1885, doi:10.1109/TIM.2003.820472.
- [22] D. Eberly, Least squares fitting of data, <http://www.geometrictools.com/Documentation/LeastSquaresFitting.pdf> [Online version] (2015).
- [23] M.D. Levine, A.M. Nazif, An experimental rule based system for testing low level segmentation strategy, in: K. Preston, L. Uhr (Eds.), *Multicomputers and Image Processing: Algorithms and Programs*, Academic Press, New York, 1982, pp. 149–160.

M. Babai obtained an M.Sc. in computing science from the Institute of Mathematics and computing science, University of Groningen, after which he worked on developing machine-learning based tools for subatomic particle identification at Kernfysisch Versneller Instituut (KVI), University of Groningen. After this he worked as a programmer at the KVI - Center for Advanced Radiation Technology (KVI-CART) on developing high-performance data-processing and analytic methods and tools. He is currently working at Center for Information Technology (CIT), University of Groningen, as a scientific-programmer/data-scientist. He is interested in morphological image analysis and graph morphology and also in application and development of data-science techniques for large scale data-processing.

N. Kalantar-Nayestanaki obtained his Ph.D. in Physics at MIT in Cambridge, Massachusetts in 1986 by studying the magnetic structure of 17O. Subsequently, he joined NIKHEF in Amsterdam as a post-doc. In 1989, he joined the staff at the Free University Amsterdam (VU) to continue his activities in nuclear structure and few-body systems. Since 1993, he has been at KVI(-CART) at the university of Groningen, first as assistant professor and later as associate and full professor. In this period, he has been involved in the design and construction of several detector setups for different experiments. His fields of activity include study of the dynamics of few-nucleon systems, and investigation of nuclear and hadron structure. He is a member of several collaborations. Currently, he is the spokesperson of the largest nuclear physics collaboration in the world. He is also a member of Academia Europaea since 2017.

J. G. Messchendorp obtained his Ph.D. cum laude in 1999 at the University of Groningen in the field of few-nucleon scattering, followed by a postdoc position at the University of Giessen, Germany. He played a leading role in the study of in-medium effects of light mesons using photo-induced reactions. In 2003 he joined the nuclear-accelerator facility KVI in Groningen, leading various few-body experiments with the aim to study (iso)spin degrees-of-freedom in few-nucleon scattering processes. His research expertise and interest lies in the field of subatomic physics, addressing the question on how matter is being formed on a subatomic scale. A crucial asset of his work is the development and application of data processing and analytic methods that allow a needle-in-a-haystack discovery. At present, he is involved in two international collaborations, BESIII and PANDA, both exploiting the annihilation of matter with antimatter to discover new forms of hadronic matter exploiting sensitive probes such as isospin-violating reactions. Since 2016, he is member of the executive board of PANDA, steering and coordinating its future physics program. Moreover, he is board member of the subatomic or nuclear division of the Dutch and European physical societies.

M. H. F. Wilkinson obtained an M.Sc. in astronomy from the Kapteyn Laboratory, University of Groningen in 1993, after which he worked on image analysis of intestinal bacteria at the Department of Medical Microbiology, University of Groningen, obtaining a PhD at the Institute of Mathematics and Computing Science, also in Groningen, in 1995. He was appointed as researcher at the Centre for High Performance Computing in Groningen working on simulating the intestinal microbial ecosystem on parallel computers. During that time he edited the book "Digital Image Analysis of Microbes" (John Wiley, UK, 1998) together with Frits Schut. After this he worked in various capacities at the Bernoulli Institute for Mathematics, Computer Science, and Artificial Intelligence on pattern recognition, image analysis and computer vision in a range of applications ranging from subatomic particles and single-celled organisms through 2-D and 3-D medical imaging to cosmic structures in astronomy. He is currently associate professor at the Bernoulli Institute, working on morphological image analysis and especially connected morphology.