

Zigzag persistence for image processing: New software and applications

Jose Divasón^a, Ana Romero^{a,*}, Pilar Santolaria^b, Jesús L. Yániz^b

^a Department of Mathematics and Computer Science, University of La Rioja, Logroño, 26006, Spain

^b BIOFITER research group, Environmental Sciences Institute (IUCA), Department of Animal Production and Food Sciences, University of Zaragoza, Huesca, 22071, Spain

ARTICLE INFO

Editor: Jiwen Lu

Keywords:

Zigzag persistence
Image processing
Topological data analysis
Mathematical morphology

ABSTRACT

Topological image analysis is a powerful tool for understanding the structure and topology of images, being persistent homology one of its most popular methods. However, persistent homology requires a chain of inclusions of topological spaces, which can be challenging for digital images. In this article, we explore the use of zigzag persistence, a recent variant of traditional persistence, for digital image processing. To this end, new algorithms are developed to build a simplicial complex associated to a digital image and to compute the relationships between homology classes of a sequence of binary images via zigzag persistence. Additionally, we provide a simple software to use them. We demonstrate its effectiveness by applying it to a real-world problem of analyzing honey bee sperm videos.

1. Introduction

Topological image analysis has emerged as a powerful tool for understanding the structure and topology of images. This approach relies on the use of algebraic topology to analyze images and extract topological features that are invariant under various image transformations. One of the most popular methods of topological image analysis is the use of persistent homology [1], which provides a powerful tool for detecting and characterizing topological features of images, such as connected components (0-dimensional homological features), holes (1-dimensional homological features), voids (2-dimensional homological features) and so on, which do not depend on specific measurements. These topological features are graphically represented in an easy and intuitive way by means of a barcode, a complete invariant which provides explainable information on the shape of the data. In the barcode, longer intervals correspond to more robust features, whereas shorter intervals are more likely to be noise in the data. The field of TDA has proven useful in many applications such as medical biology [2], physics [3], and atmospheric science [4]. Recently, persistent homology has also been used in combination with convolutional neural networks (CNNs); given a dataset, persistent homology is applied to produce some topological features that can then be fed into machine learning models to improve their results and provide explainable outputs [5]. Persistent homology has been already applied to a wide range of data and image analysis tasks, such as edge detection [6], skin lesions [7], cell trajectory inference [8], and many more.

Persistent homology requires a chain of inclusions of topological spaces. The idea is to construct a sequence of nested subspaces of the original space, where each subspace includes the previous one. The sequence of nested subspaces is defined by an increasing filtration. By computing the homology groups of each subspace, one can track the topological features, i.e., when each feature appears and disappears. The need for each subspace to include the previous one is an important restriction. In particular, if one has to work somehow with a sequence of images, the foreground pixels of each previous image must be maintained in each step (new ones may appear, though).

Zigzag persistence is a recent variant of the traditional persistence algorithm in algebraic topology, which is also used to extract topological features from datasets. It was introduced by Carlsson and Silva [9] as a generalization of the original persistence algorithm for the case of a sequence of topological spaces that are not related by means of an increasing filtration. In zigzag persistence, the chain of inclusions is defined by a zigzag filtration, which allows homology classes to be transported between different subspaces in the filtration by zigzag paths. Zigzag paths are sequences of inclusion maps that can be used to track the evolution of topological features, even when the features undergo non-linear transformations or noise. Compared to traditional persistence, zigzag filtration is also defined by a sequence of subspaces, but the crucial difference is that inclusion can go in any direction in each step. In practice, it is not necessary to define explicitly the inclusion (and surjection) maps, but simply providing the topological

* Corresponding author.

E-mail addresses: jose.divason@unirioja.es (J. Divasón), ana.romero@unirioja.es (A. Romero), psantola@unizar.es (P. Santolaria), jyaniz@unizar.es (J.L. Yániz).

<https://doi.org/10.1016/j.patrec.2024.06.010>

Received 3 April 2023; Received in revised form 11 June 2024; Accepted 12 June 2024

Available online 15 June 2024

0167-8655/© 2024 The Author(s). Published by Elsevier B.V. This is an open access article under the CC BY-NC license (<http://creativecommons.org/licenses/by-nc/4.0/>).

subspaces (the maps can be constructed automatically through unions or intersections). In other words, one important advantage is that there is no need to relate the images by means of an increasing filtration when working with digital images. This way, zigzag persistence is more flexible than persistent homology.

Zigzag persistence has been applied to topological data analysis [10], but there are barely any applications of zigzag persistence to digital images. To the best of our knowledge, the only application of zigzag persistence on digital images is a work about analyzing stacks of neuronal images [11].

There are some reasons why zigzag persistence is not so widely used for images: It is a relatively new and advanced topic in topological data analysis and requires a deeper understanding of algebraic topology and computational geometry than traditional persistence, which make it more challenging for its use. Indeed, there are currently fewer software packages available for computing zigzag persistence compared to traditional persistence, and none of them is specialized in digital images. The contributions of this work are the following:

1. An algorithm to build a simplicial complex associated to a binary digital image smaller than the ones used in the literature, but with the same topological information. This makes the zigzag computation more efficient.
2. An algorithm to compute the relationship between the n -homology classes of a sequence of binary images (of the same size, without any relationship among them) via zigzag persistence, using the algorithm for zigzag persistence of topological spaces introduced in [9].
3. An easy-to-use and open-source software (with GUI) to compute the previous algorithm for a sequence of binary images and plot the corresponding barcodes.
4. A dataset for experiments and also an example of zigzag application to a real-world problem (that cannot be tackled with persistent homology): analysis of videos of sperm of honey bee drones, where we can use our software to track motile spermatozoa and as a preprocessing step to detect static spermatozoa in noisy images.

The rest of the paper is organized as follows: Section 2 presents the proposed algorithms to compute zigzag persistence of digital images. The developed software is discussed in Section 3. Section 4 shows a potential real-world application of the new algorithms and a dataset for experiments. Finally, the conclusions and further work are detailed in Section 5. The following repository contains our programs:

<https://github.com/jodivaso/ImageZigZag>

2. New general algorithms for computing zigzag persistence of digital images

The theory of zigzag persistence [9] is defined for diagrams of topological spaces of the form:

$$X_1 \leftrightarrow X_2 \leftrightarrow \dots \leftrightarrow X_m$$

where the arrows can point either left or right.

Considering the induced morphisms on the homology groups of each topological space, for each $n \in \mathbb{N}$ one obtains a sequence of vector spaces and linear maps:

$$V_1 \equiv H_n(X_1) \leftrightarrow V_2 \equiv H_n(X_2) \leftrightarrow \dots \leftrightarrow V_m \equiv H_n(X_m)$$

which is called a *zigzag module*. Zigzag modules can be decomposed as a direct sum of submodules W^i of the form

$$0 \leftrightarrow \dots \leftrightarrow 0 \leftrightarrow W_{a_i}^i \equiv \mathbb{F} \leftrightarrow \dots \leftrightarrow W_{b_i}^i \equiv \mathbb{F} \leftrightarrow 0 \leftrightarrow \dots \leftrightarrow 0$$

for some $1 \leq a_i \leq b_i \leq m$, where \mathbb{F} is the base field and all arrows are the identity map, see [9] for further details. In this way, zigzag modules can be classified up to isomorphism by a multi-set of intervals $\{[a_i, b_i]\}$

with $1 \leq a_i \leq b_i \leq m$, which leads to the graphical representation of zigzag modules by means of *barcode diagrams* (see [9]).

As said in the introduction, although persistent homology has been used in many different problems in image processing, up to our knowledge there is only one application of zigzag persistence to digital images [11], being implemented for a particular situation. In this work, we propose a general algorithm for applying zigzag persistence to any sequence of binary images (with the same size).

Zigzag persistence is implemented in the Dionysus 2 software [12]. To use this program, we need to construct a simplicial complex associated with a digital image. A *simplicial complex* is a particular case of topological space defined by means of points (called vertices), line segments (edges), triangles, and their n -dimensional counterparts (see [13] for details). A binary (2D) image will be given by a matrix of pixels having two possible values: black and white (usually coded as 0 and 255, respectively). In this paper, we consider white as foreground and black as background (one can also choose the opposite convention), and we use 8-adjacency between pixels, because it is the most suitable one for our problems (see Section 4).

In a first step of our work we implemented directly the construction of the simplicial complex corresponding to the triangulation of the cubical complex of a binary image, but many simplices were obtained and the zigzag persistence computation was slow. To design our new [Algorithm 1](#), we have made use of the Vietoris–Rips simplicial complex associated to a binary image, considering each white pixel as a point in a two-dimensional Euclidean space (that is also a way to represent the topological properties of the image, see [14]) and discrete Morse theory [15] (in particular, the notion of discrete vector field, which allows us to *remove* unnecessary edges and triangles). Given a simplicial complex, a *discrete vector field* V is a list of pairs of simplices $V = \{(\sigma_i, \tau_i)\}_{i \in J}$ such that each σ_i is a face of τ_i (with some additional hypotheses). The simplices that do not appear in the vector field are called *critical*. If the vector field is admissible (see [15] for details), then the initial simplicial complex can be reduced to a smaller one, with the same homology groups, where only the critical simplices appear. In our case, for example, when three white pixels appear in the image at positions (i, j) , $(i + 1, j)$ and $(i, j + 1)$, the Vietoris–Rips complex includes a triangle T with vertices at these three pixels and an edge E between the vertices $(i, j + 1)$ and $(i + 1, j)$. However, if the pixel at $(i + 1, j + 1)$ is black, then a vector given by the pair $(\sigma = E, \tau = T)$ can be defined and then both simplices are *removed* from the simplicial complex maintaining the homology groups. With our new [Algorithm 1](#), the number of vertices, edges and triangles in the complex is smaller than in the triangulation of the cubical complex of a binary image and in the Vietoris–Rips complex and the computation of zigzag persistence is faster, see the repository for the detailed comparison. [Fig. 1](#) shows an example of application of [Algorithm 1](#).

Algorithm 1. Input: a binary image I , with r rows and c columns. Output: a simplicial complex K .

1. Start with $K = \emptyset$.
2. For each white pixel in I at position (i, j) , add to K the vertex $i * c + j$.
3. For each pair of white pixels in I at positions (i, j) and $(i + 1, j)$ or (i, j) and $(i, j + 1)$, add to K an edge between the corresponding vertices.
4. For each pair of white pixels at (i, j) and $(i + 1, j + 1)$ such that the pixels at $(i + 1, j)$ and $(i, j + 1)$ are both black, add to K an edge between the corresponding vertices. For each pair of white pixels at (i, j) and $(i + 1, j - 1)$ such that the pixels at $(i + 1, j)$ and $(i, j - 1)$ are both black, add to K an edge between the corresponding vertices.
5. For each pair of white pixels at (i, j) and $(i + 1, j + 1)$ such that the pixels at $(i + 1, j)$ and $(i, j + 1)$ are both white, add to K an edge between the corresponding vertices to (i, j) and $(i + 1, j + 1)$ and add two triangles with the vertices corresponding to the pixels at (i, j) , $(i + 1, j)$ and $(i + 1, j + 1)$, and (i, j) , $(i, j + 1)$ and $(i + 1, j + 1)$.

6. Return K .

Now, given a sequence of binary images, in Algorithm 2 we study the relationship between the homology classes of each image by combining Algorithm 1 with the computation of zigzag persistence (of a sequence of simplicial complexes). To this aim, we follow the same idea as in [9, Example 1.1]: given a sequence of simplicial sets K_1, \dots, K_m , we consider the union sequence:

$$K_1 \hookrightarrow K_1 \cup K_2 \hookrightarrow K_2 \hookrightarrow K_2 \cup K_3 \hookrightarrow \dots \hookrightarrow K_{m-1} \cup K_m \hookrightarrow K_m$$

This way, when computing the homology groups and the corresponding barcode of the zigzag filtration, the relation between the homology classes of the initial simplicial complexes K_i is obtained (each bar corresponds to a homology class that appears in different simplicial complexes; different bars correspond to different homology classes in the complexes). Given two consecutive simplicial complexes K_i and K_{i+1} (corresponding, respectively, to the images I_i and I_{i+1}), the union is computed without considering the vectors of the discrete vector field which have been only applied in one of the images (these vectors may not be compatible with the zigzag filtration).

Algorithm 2. Input: a sequence of binary images I_1, \dots, I_m of the same size (that is, the same numbers of rows and columns) and an integer $n \geq 0$.

Output: The zigzag barcode describing the continuity of the n -homology classes between the different images and a list of generators associated with each of the bars in the barcode.

1. For each $0 \leq i \leq m$, apply Algorithm 1 to the binary image I_i and construct the associated simplicial complex K_i .
2. For each $0 \leq i < m$, compute the union of the simplicial complexes K_i and K_{i+1} and add the simplices corresponding to vectors which appear only in one of the images. We denote this union by K'_i .
3. Consider the following sequence of maps:

$$K_1 \hookrightarrow K'_1 \hookrightarrow K_2 \hookrightarrow K'_2 \hookrightarrow \dots \hookrightarrow K'_{m-1} \hookrightarrow K_m$$

4. Compute the zigzag persistence of the sequence of vector spaces: $H_n(K_1) \rightarrow H_n(K'_1) \leftarrow H_n(K_2) \rightarrow \dots \leftarrow H_n(K_m)$
5. Draw the barcode and return the generators.

Algorithm 2 can be used to study the relationship between the n -homology classes of a sequence of binary images (of the same size) without any relationship between them. For instance, let us consider as a didactic example the three binary images of Fig. 2. In the first two images, three 1-dimensional classes appear, whereas the last one only contains two. One of the holes *lives* in the three images (top left corner; some pixels are different, but the hole is essentially the same); another 1-dimensional class appears in the first image (top right) and disappears in the third one; there is a hole that is only shown in the first image (bottom) and finally, another homology class appears in the second image (bottom right corner) and is still alive in the third image. We can obtain this information by computing the zigzag barcode following our Algorithm 2, as shown in Fig. 3.

As a remark, we would like to observe that, to apply Algorithm 2, it is not necessary to relate the binary images by means of a filtration (which is a necessary condition to apply persistent homology). This allows us to apply our algorithm to problems that have not been tackled before with topological data analysis, as the applications presented in Section 4.

We have enhanced Algorithm 2 by allowing three optional parameters: *interval-length*, an integer representing the minimum length necessary for a zigzag interval to be considered in the barcode (to discard very short bars, which correspond to homology classes that persist a very short period of time), and *generator-min-length* and *generator-max-length*, the minimum and maximum number

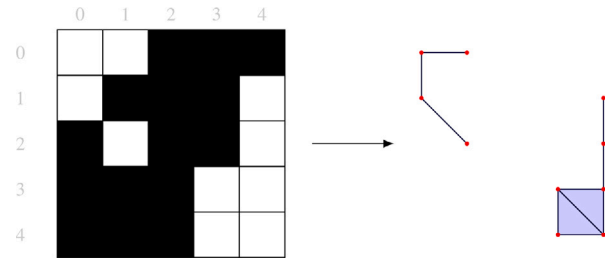


Fig. 1. On the left, a digital image; on the right, its simplicial complex representation.



Fig. 2. Sequence of three binary digital images.

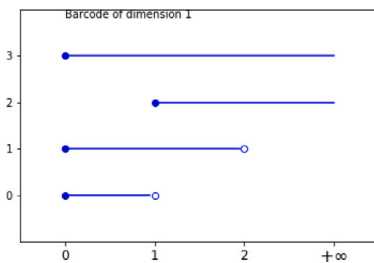


Fig. 3. Zigzag barcode of images in Fig. 2.

respectively of simplices in a generator that are necessary for a bar to be considered in the barcode (to discard, for instance, small holes). This lead to the following algorithm.

Algorithm 3. Input: a sequence of binary images I_1, \dots, I_m of the same size (that is, the same numbers of rows and columns) and integers n , *interval-length*, *generator-min-length*, *generator-max-length* ≥ 0 .

Output: A subset of the zigzag barcode of the images of dimension n containing the bars with length greater than or equal to *interval-length* and such that the corresponding generators are made of l simplices, with *generator-min-length* $\leq l \leq$ *generator-max-length*.

- 1–4. Same steps as in Algorithm 2.
5. Select those intervals in the barcode with length greater than or equal to *interval-length* and such that the corresponding generators are made of l simplices, with *generator-min-length* $\leq l \leq$ *generator-max-length*.
6. Draw the barcode and return the generators of the selected intervals.

3. Description of the new software

Algorithms 1, 2 and 3 have been implemented as Python functions using the library Dionysus 2 [12] for computing the zigzag modules. These algorithms are also implemented in a Jupyter notebook, available in the repository. To allow users to apply our programs in an easier way, we have also developed a graphical interface. The new software inputs a set of binary images and it allows the user to decide the order

of the images (to compute the zigzag persistence). As parameters, as specified in Algorithm 3, the user can choose the dimensions for which the barcode will be determined (0 or/and 1), the minimum length of a bar to be showed and the minimum and maximum number of simplices (vertices or edges) in a generator that are needed to be showed in the barcode. It is also possible to decide whether the interval generators are shown in the barcode. The program then applies Algorithm 3 to compute the associated zigzag barcode. Fig. 4 shows the interface of our application. The result is a graphical representation of the barcode diagrams shown within the application, see Fig. 5.

4. Applications

In this section, we present two applications of our algorithms for computing zigzag persistence of a sequence of binary images. Both applications are illustrated with images obtained from videos of sperm of honey bee drones, which have been acquired using an Olympus BX40 microscope (Olympus Optical Co., Tokyo, Japan). The aim of the analysis of these images is to evaluate the sperm motility and concentration by identifying two types of spermatozoa: motile ones, which show a circular shape in most videos, and static ones, which show a linear shape. In a previous work [16], an open-source software called CASABee was developed in Python using libraries such as OpenCV (for image processing). The program inputs one or several videos and analyzes them in order to study sperm motility and concentration by identifying both the motile and static sperm that appear in the videos. Fig. 6 shows two examples of CASABee analysis, showing the classification of spermatozoa in motile (circles) and static (red lines). Once the spermatozoa are identified in the videos, CASABee produces the following numerical results: total number of sperms, number of static spermatozoa, number of motile spermatozoa, motile percentage, and concentration. Now, we consider our new software for computing zigzag persistence of a sequence of binary images and apply it for the computation of both static (in Section 4.1) and motile spermatozoa (in Section 4.2). In addition, Section 4.3 presents more experiments with an artificial dataset of different basic shapes moving in the space.

4.1. Combining mathematical morphology operators

Mathematical morphology or simply *morphology* is a field that provides several techniques for processing digital images. It is based on the study of spatial structures on an image and it is applied to a large number of imaging problems and applications (see, for instance, the recent papers [17,18]). The main mathematical morphology operators are *dilation* and *erosion*. In a binary image, dilation enlarges the number of foreground pixels (in our case, white pixels) by turning into white the black pixels that are neighbors of white pixels (the notion of neighborhood is defined by means of a *structuring element*, see [19] for details), while erosion erases (turns into black) white pixels that are neighbors of a black one.

Let I be a binary image and K its associated simplicial complex (obtained, for instance, by applying Algorithm 1). Given $j \geq 0$, let us denote by $D^j I$ (resp. $E^j I$) the binary images obtained by applying j dilations (resp. j erosions) to I , and by $D^j K$ (resp. $E^j K$) the corresponding simplicial complexes. It is easy to observe that the following relations are satisfied:

$$\dots \subseteq E^j K \subseteq E^{j-1} K \subseteq \dots \subseteq EK \subseteq K$$

$$K \subseteq DK \subseteq \dots \subseteq D^{j-1} K \subseteq D^j K \subseteq \dots$$

In other words, a filtration is obtained, and this makes it possible to apply persistent homology. This approach has been considered, for instance, in [20].

However, in other problems dilation and erosion are combined in a way such that they do not provide a filtration. For instance, one could be interested in applying i dilations followed by j erosions, and one wants to know the *most adequate* numbers i and j . In that situation,

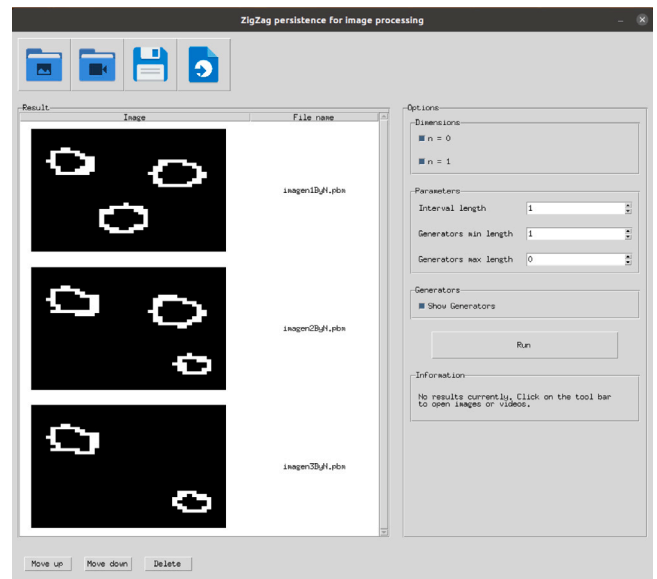


Fig. 4. Interface of the new application.

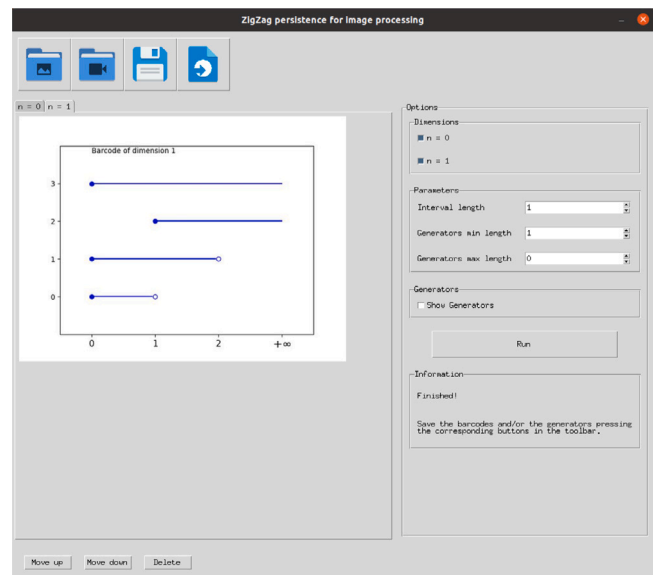


Fig. 5. Results of the new application.

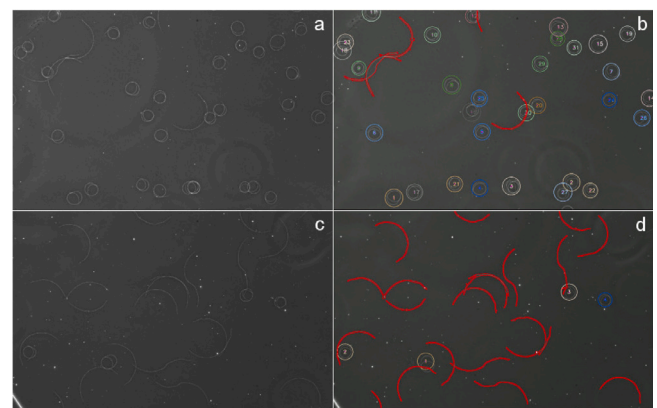


Fig. 6. Examples of CASABee analysis. Phase contrast images from two video sequences of different sperm motility (a, c), and the resulting CASABee output (b, d), showing the classification of spermatozoa in motile (circles) and static (red lines).

the corresponding images and simplicial complexes do not provide a filtration and, therefore, one cannot apply persistent homology. To solve this problem, we propose to apply zigzag persistence by using our Algorithms 2 and 3, which determine the relationship between the n -homology classes of the different images.

As a particular case of this situation, we consider images obtained from videos of sperm of honey bee drones. In order to detect the static spermatozoa, the CASABee software presented in [16] works as follows. First, the enhanced frames of the video (after smooth filter and normalization) are examined and binarized using an appropriate threshold. Then, dilation is applied and the intersection of all the binary images, corresponding to white pixels appearing in all of them, is computed. In the next step, the contours in the binary image are determined. Those with an area greater than 20% of the average size of the spermatozoa (given by a parameter chosen by the user) and those in which the proportion between the contour area and the bounding box area is less than 0.3 (to discard artifacts) are selected. Some fragments of circles appearing in the image are removed by applying the Hough transform [21], a feature extraction technique used in image analysis to detect arbitrary shapes, most commonly circles or lines. To achieve this, parameters for the minimum radius and maximum radius of the detected circles are required, together with two other parameters related with the threshold. Finally, some “broken” contours are joined by drawing lines between closed pixels in the direction of the contour. The program was evaluated and showed good results [16]. It is able to detect correctly static spermatozoa in most situations. However, there are a few videos in which it also detects as spermatozoa some areas that are not, see for instance Fig. 7. In that figure (which corresponds to a frame of a video), there are many static spermatozoa (red lines) that are detected incorrectly, mainly corresponding to noise in the image. Indeed, experts only count 4 static spermatozoa in that video, but CASABee detects (wrongly) many more.

In order to solve this problem, we propose to use zigzag persistence combined with morphological operations. The idea consists in considering the intersection of all frames (after binarization and enhancement, as explained before) and then applying a different number of erosions and dilations. This produces a sequence of binary images, and the barcode obtained by computing the zigzag persistence of degree 0 of this sequence allows one to identify the static spermatozoa as the components which survive at least 5 steps (a value obtained experimentally for our problem, and related with the thickness of the spermatozoa), and discard artifacts and noise. The proposed method consists in applying the following steps:

1. We consider the intersection of the enhanced binary images of all frames (as explained in the previous paragraph, before the step of computing the contours).
2. We dilate twice, in order to remove small holes in the different components.
3. For $i = 1, 2, \dots, 10$, we apply i erosions followed by $i - 1$ dilations. We obtain a list of binary images I_1, I_2, \dots, I_{10} .
4. We erode each image I_i twice. We obtain binary images $I'_1, I'_2, \dots, I'_{10}$.
5. We do the difference between the image obtained in Step 1 and each image I'_i of the previous step. We obtain different binary images $I''_1, I''_2, \dots, I''_{10}$, with the same size, which do not provide a filtration.
6. We apply to each image the same postprocessing as the CASABee software (draw the contours, select those with an adequate area, etc.). The result is a list of binary images, again with the same size and without defining a filtration.
7. We apply Algorithm 3 to the images obtained in the previous step, with $n = 0$, interval-length = 5, generator-min-length = 10 (these have been obtained experimentally for our problem, and they are considered to be adequate; the parameter generator-max-length has not any limitation in this case).

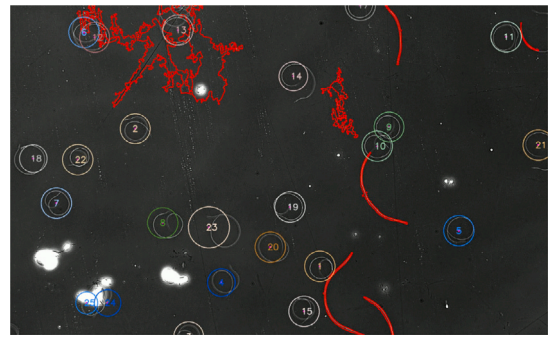


Fig. 7. Example of CASABee output with noise detected as static spermatozoa.

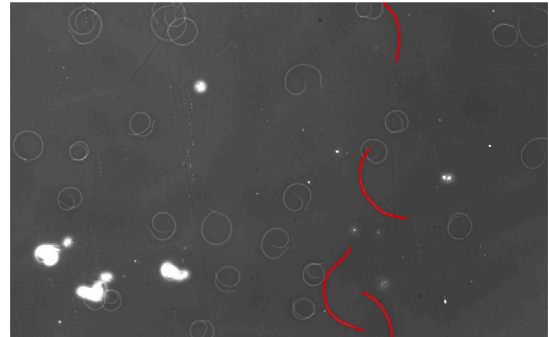


Fig. 8. Results of the zigzag approach to detect static spermatozoa (without noise).

8. The barcode and the generators provide us the connected components corresponding to static sperm, discarding artifacts and noise in this way.

This approach based on zigzag persistence improves the CASABee results since noise is not selected anymore; see for instance Fig. 8 (only static spermatozoa are displayed). This method can be applied for all videos and we obtain the graphical interpretation of the different connected components behavior by means of the barcode.

4.2. 1-homology tracking

Zigzag persistence can also be used to track the 1-homology components in a sequence of binary images. For instance, we consider again our images obtained from videos of sperm of honey bee drones, observing in this case the motile spermatozoa (circles in Fig. 6). In order to detect the motile spermatozoa, the CASABee program extracts all the video frames and enhances each image by means of a smooth filter and image normalization. Then, the Hough transform is applied again to select the circles. To detect spermatozoa that appear in the borders of the video and that are not closed circles, CASABee adds a border to each frame with a symmetry criterion. Furthermore, circles with a high density of white pixels, corresponding to artifacts (and not to motile spermatozoa), were discarded.

The circles selected in the first frame were then tracked in all the images in the following manner. Each circle in the first frame was labeled with an integer number. In the next frame, circles whose center was inside each circle detected in the first frame were looked for. If there was only one circle in this situation, then this circle was labeled with the same number as the previous one. If there were at least two circles satisfying the condition, then we chose the one whose center was closest to the center of the previous circle. We continued this process for all frames of the image. Once all the circles of the first frame had been tracked in all the frames, the circles which appeared in at least half of the frames were selected. Those that ended with the same label (which

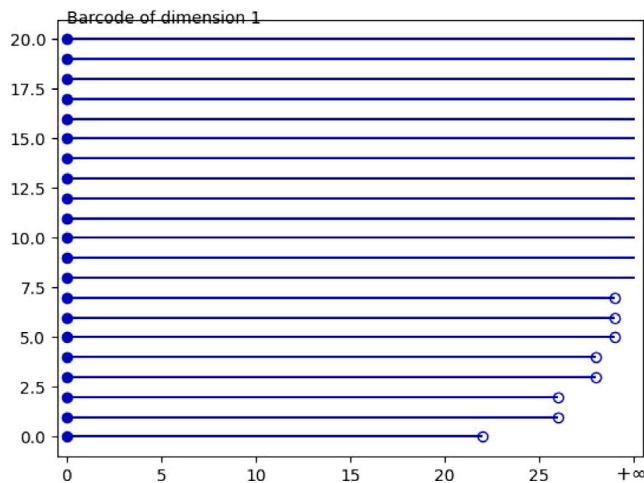


Fig. 9. Results of the zigzag approach to detect motile spermatozoa (1-homology tracking).

means that they corresponded to the same motile spermatozoon) were combined, and the circles with the correct numbers were relabeled. Although this method works well for most motile spermatozoa, we propose the following alternative method by using zigzag persistence:

1. For each frame in the video, we consider a binary image and we draw on it the circles detected by the Hough transform (as explained previously in this subsection).
2. We apply Algorithm 3, in this case with parameter $n = 1$, and interval-length computed as the half of the number of frames of the video. The size of generators depend on the values of some parameters of the CASABee software, concretely on the minimum and maximum radius, which restrict the size of the circles detected by the program using Hough transform.
3. The barcode and the generators provide us now the tracking of the motile spermatozoa in the videos.

Fig. 9 shows an example of the result of applying this method. The video is the one already presented in Fig. 7. For the analysis, interval-length is set to 15 (to capture holes that appear in, at least, half of the frames). In this case, generator-min-length is set to 8 (to remove minor holes that could appear if spermatozoa overlap).¹ We do not impose any limitation on generator-max-length. The video has a total of 23 spermatozoa and the program detects 21. Two of them are never detected, but for this video it is an expected result since they appear close to a border and only half of the sperm is seen (one above and one below in Fig. 7). CASABee is able to detect them because it does border symmetry. Some spermatozoa disappear in the zigzag tracking because their intensity is reduced in the last frames.

Our method is an alternative to track 1-homology classes that can be useful when other techniques (such as Hough transform) cannot be applied or when we are interested in having the graphical representation by means of the barcode.

4.3. More experiments with a new dataset

The repository also contains a new dataset (19 videos with 60 frames, 1 video with 160 frames) for testing and information on how to

¹ In this potential example of application of honey bee sperm analysis, this is usually avoidable, as scientists can dilute the sample to reduce the concentration. However, such a problematic video does contain overlapped spermatozoa.

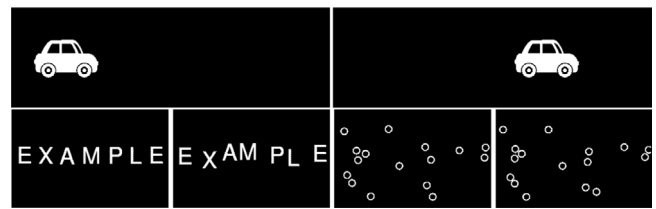


Fig. 10. Initial and final frame of three videos presented in the dataset.

Table 1

Accuracy and time comparison among the different approaches. For the baseline approaches we only show the best results. Parentheses indicate the detector and the tracking algorithm used for the best result.

Method	H0	H1	Time (s)	Parameter tuning?
ZigZag	97.54	98.95	9296	No
Baseline method 1 (findContours+MIL)	98.95	99.01	1031	Yes
Baseline method 2 (findContours+MIL)	98.95	99.01	1027	Yes
Baseline method 3 (findContours)	96.34	96.57	2.58	Yes

reproduce the results. Some examples of frames contained in the dataset are shown in Fig. 10. Specifically, the dataset contains:

1. Processed frames of several sample videos contained in the CASABee software about detecting motile spermatozoa.
2. Artificially generated videos of objects of different shapes (letters, squares, cars, etc.) that move around the space.

Experiments with this dataset show that the zigzag method allows tracking the connected components (0-homology) and holes (1-homology) with good results (about 99% and 98% accuracy, respectively). The experiments also show that the total computation time (simplicial complex construction + zigzag persistence) is much better when our Algorithm 1 is used, compared to, for instance, cubical complexes (about 23% of reduction in that case). The repository also contains comparisons to three object tracking approaches. The first two are based on the OpenCV Tracker API [22,23] using six tracking algorithms (CSRT [24], MIL [25], KCF [26], Mosse [27], Median-Flow [28] and Boosting [29]), combined to Hough circle transform and findContours as detectors. The third is the well-known basic method consisting of contour detection and centroid matching based on the Euclidean distance (note that false positives could easily arise from incorrect matchings).

Table 1 presents a summary of the results. Baseline methods always performed better using findContours as detector (Hough transform achieves at most a 77.45% of accuracy in H1; some false positives could be detected using this approach) and MIL as the tracking algorithm (similar results also with Boosting and CSRT). The parameters of the detectors in the baseline approaches were manually fine-tuned to get this performance. In view of the results, the three baseline approaches work well and results are very similar to the zigzag ones in terms of accuracy (we refer to the repository for further details).

The main benefit of the zigzag method is that it allows one to perform the three steps automatically, i.e., it allows tracking objects in an automatic way (in the sense that the method indicates in which frames each object appears and disappears). In addition, as opposed to the Hough transform, this method also allows one to track holes other than circles. The drawback is the time: as expected, the zigzag persistence computation is much slower than the use of a specialized tracking algorithm or the contour and centroid matching computation.

5. Conclusions and further work

In this work, we propose general algorithms to compute zigzag persistence over digital images. Our algorithms have demonstrated their effectiveness by helping in a real-world problem that cannot be addressed with persistent homology. Specifically, we have shown how our approach can be used to analyze videos of sperm of honey bee drones, allowing us to track motile sperm using 1-dimensional homology. We have also developed a user-friendly graphical interface that makes it easy to use our algorithms. Our code is open-source and freely available, which means that it can be easily adapted for specific purposes. As further work, we intend to study and integrate the recent advances for faster zigzag persistence computation [30], since potential real-world applications related to digital images or videos (such as those ones presented in Section 4) usually require much execution time. In fact, the inclusion of our modifications based on zigzag persistence in CASABee causes the analysis time of a video to increase from a few seconds to a few minutes. Furthermore, although our algorithms and programs have been presented only for 2D-images, they can be easily extended for n -dimensional images. To this aim, it is only necessary to implement or use an algorithm computing the associated simplicial (or cubical) complex. A more difficult problem consists in working with grayscale images, where the sublevel sets filtration can be considered, and study how to combine this filtration with the zigzag modules.

CRedit authorship contribution statement

Jose Divasón: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Resources, Methodology, Investigation, Funding acquisition, Formal analysis, Conceptualization. **Ana Romero:** Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Resources, Project administration, Methodology, Investigation, Funding acquisition, Formal analysis, Conceptualization. **Pilar Santolaria:** Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Resources, Project administration, Methodology, Investigation, Funding acquisition, Formal analysis, Conceptualization. **Jesús L. Yániz:** Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Resources, Project administration, Methodology, Investigation, Funding acquisition, Formal analysis, Conceptualization.

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.

Data availability

Data will be made available on request.

Acknowledgments

This work is supported by the Spanish MICIU/AEI/10.13039/501100011033 (grants PID2020-112673RB-I00 and PID2020-116641 GB-I00), by project Inicia 2023/02 funded by La Rioja Government (Spain) and by the DGA-FSE (grant A07_23R).

References

- [1] A. Zomorodian, G. Carlsson, Computing persistent homology, *Discrete Comput. Geom.* 33 (2) (2005) 249–274, <http://dx.doi.org/10.1007/s00454-004-1146-y>.
- [2] L. Li, W.-Y. Cheng, B.S. Glicksberg, O. Gottesman, R. Tamler, R. Chen, E.P. Bottinger, J.T. Dudley, Identification of type 2 diabetes subgroups through topological analysis of patient similarity, *Sci. Transl. Med.* 7 (311) (2015) <http://dx.doi.org/10.1126/scitranslmed.aaa9364>.
- [3] I. Donato, M. Gori, M. Pettini, G. Petri, S.D. Nigris, R. Franzosi, F. Vaccarino, Persistent homology analysis of phase transitions, *Phys. Rev. E* 93 (5) (2016) <http://dx.doi.org/10.1103/physreve.93.052138>.
- [4] S. Tymochko, E. Munch, J. Union, K. Corbosiero, R. Torn, Using persistent homology to quantify a diurnal cycle in hurricanes, *Pattern Recognit. Lett.* 133 (2020) 137–143, <http://dx.doi.org/10.1016/j.patrec.2020.02.022>.
- [5] J.R. Clough, N. Byrne, I. Oksuz, V.A. Zimmer, J.A. Schnabel, A.P. King, A topological loss function for deep-learning based image segmentation using persistent homology, *IEEE Trans. Pattern Anal. Mach. Intell.* 44 (12) (2020) 8766–8778.
- [6] V. Kurlin, G. Muszynski, Persistence-based resolution-independent meshes of superpixels, *Pattern Recognit. Lett.* 131 (2020) 300–306, <http://dx.doi.org/10.1016/j.patrec.2020.01.014>.
- [7] R. Vandaele, G.A. Nervo, O. Gevaert, Topological image modification for object detection and topological image processing of skin lesions, *Sci. Rep.* 10 (1) (2020) <http://dx.doi.org/10.1038/s41598-020-77933-y>.
- [8] R. Vandaele, B. Rieck, Y. Saeys, T. De Bie, Stable topological signatures for metric trees through graph approximations, *Pattern Recognit. Lett.* 147 (2021) 85–92, <http://dx.doi.org/10.1016/j.patrec.2021.03.035>.
- [9] G. Carlsson, V. de Silva, Zigzag persistence, *Found. Comput. Math.* 10 (4) (2010) 367–405, <http://dx.doi.org/10.1007/s10208-010-9066-0>.
- [10] A. Tausz, G. Carlsson, Applications of zigzag persistence to topological data analysis, 2011, [arXiv:1108.3545](https://arxiv.org/abs/1108.3545).
- [11] G. Mata, M. Morales, A. Romero, J. Rubio, Zigzag persistent homology for processing neuronal images, *Pattern Recognit. Lett.* 62 (2015) 55–60, <http://dx.doi.org/10.1016/j.patrec.2015.05.010>.
- [12] D. Morozov, *Dionysus 2*, 2012, mrzv.org/software/dionysus2.
- [13] A. Hatcher, *Algebraic Topology*, Cambridge Univ. Press, 2000.
- [14] A. Garin, G. Tauzin, A topological “reading” lesson: Classification of MNIST using TDA, in: 18th IEEE International Conference On Machine Learning And Applications, ICMLA, 2019, [arXiv:1910.08345](https://arxiv.org/abs/1910.08345).
- [15] R. Forman, Morse theory for cell complexes, *Adv. Math.* 134 (1998) 90–145, <http://dx.doi.org/10.1006/aima.1997.1650>.
- [16] J. Divasón, A. Romero, M.A. Silvestre, P. Santolaria, J.L. Yániz, *In vitro* maintenance of drones and development of a new software for sperm quality analysis facilitate the study of honey bee reproductive quality, *J. Apicult. Res.* (2023) <http://dx.doi.org/10.1080/00218839.2023.2231673>.
- [17] D. Kumar Jain, S. Neelakandan, A. Vidyarthi, D. Gupta, Deep learning-based intelligent system for fingerprint identification using decision-based median filter, *Pattern Recognit. Lett.* 174 (2023) 25–31, <http://dx.doi.org/10.1016/j.patrec.2023.08.006>.
- [18] E. Paulus, J.-C. Burie, F.J. Verbeek, Text line extraction strategy for palm leaf manuscripts, *Pattern Recognit. Lett.* 174 (2023) 10–16, <http://dx.doi.org/10.1016/j.patrec.2023.08.007>.
- [19] P. Soille, *Morphological Image Analysis: Principles and Applications*, second ed., Springer-Verlag, Berlin, Heidelberg, 2003.
- [20] Y. Chung, S. Day, C. Hu, A multi-parameter persistence framework for mathematical morphology, *Sci. Rep.* 12 (2022) 6427, <http://dx.doi.org/10.1038/s41598-022-09464-7>.
- [21] R.O. Duda, P.E. Hart, Use of the hough transformation to detect lines and curves in pictures, *Commun. ACM* 15 (1) (1972) 11–15.
- [22] X. Li, W. Hu, C. Shen, Z. Zhang, A. Dick, A.V.D. Hengel, A survey of appearance models in visual object tracking, *ACM Trans. Intell. Syst. Technol.* 4 (4) (2013).
- [23] S. Salti, A. Cavallaro, L. Di Stefano, Adaptive appearance modeling for video tracking: Survey and evaluation, *IEEE Trans. Image Process.* 21 (10) (2012) 4334–4348, <http://dx.doi.org/10.1109/TIP.2012.2206035>.
- [24] A. Lukežič, T. Vojří, L. Čehovin, J. Matas, M. Kristan, Discriminative correlation filter tracker with channel and spatial reliability, *Int. J. Comput. Vis.* 126 (7) (2018) 671–688, <http://dx.doi.org/10.1007/s11263-017-1061-3>.
- [25] B. Babenko, M.-H. Yang, S. Belongie, Visual tracking with online multiple instance learning, in: *IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 983–990.
- [26] J.F. Henriques, R. Caseiro, P. Martins, J. Batista, High-speed tracking with kernelized correlation filters, *IEEE Trans. Pattern Anal. Mach. Intell.* 37 (03) (2015) 583–596, <http://dx.doi.org/10.1109/TPAMI.2014.2345390>.
- [27] D.S. Bolme, J.R. Beveridge, B.A. Draper, Y.M. Lui, Visual object tracking using adaptive correlation filters, in: *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2010, pp. 2544–2550, <http://dx.doi.org/10.1109/CVPR.2010.5539960>.

- [28] Z. Kalal, K. Mikolajczyk, J. Matas, Forward-backward error: Automatic detection of tracking failures, in: 20th International Conference on Pattern Recognition, 2010, pp. 2756–2759, <http://dx.doi.org/10.1109/ICPR.2010.675>.
- [29] H. Grabner, M. Grabner, H. Bischof, Real-time tracking via on-line boosting, in: *Proceedings of the British Machine Vision Conference*, Vol. 1, 2006, pp. 47–56.
- [30] T.K. Dey, T. Hou, Fast computation of zigzag persistence, 2022, [arXiv:2204.11080](https://arxiv.org/abs/2204.11080).