Random walks in directed hypergraphs and application to semi-supervised image segmentation

Aurélien Ducournau, Alain Bretto

Random walks in directed hypergraphs and application to semi-supervised image segmentation ☆

Aurélien Ducournau [a], Alain Bretto [b,*]

[a] ENISE-DIPI, 58 rue Jean Parot, 42023 Saint-Etienne Cedex 2, France
[b] GREYC-CNRS UMR6072, Campus Côte de Nacre, Boulevard du Maréchal Juin, BP5186, 14032 Caen Cedex, France

Keywords:
Directed hypergraph
Random walks
Image segmentation
Semi-supervised learning

A B S T R A C T

In this paper, we introduce for the first time the notion of directed hypergraphs in image processing and particularly image segmentation. We give a formulation of a random walk in a directed hypergraph that serves as a basis to a semi-supervised image segmentation procedure that is configured as a machine learning problem, where a few sample pixels are used to estimate the labels of the unlabeled ones. A directed hypergraph model is proposed to represent the image content, and the directed random walk formulation allows to compute a transition matrix that can be exploited in a simple iterative semi-supervised segmentation process. Experiments over the Microsoft GrabCut dataset have achieved results that demonstrated the relevance of introducing directionality in hypergraphs for computer vision problems.

1. Introduction

Graph-based methods have played an important role in Computer Vision and Pattern Recognition (CVPR) due to their ability to represent relational patterns [1,2]. However, in many situations, a graph-based representation is incomplete, as only binary relations between nodes can be represented through graph edges. An extension is provided by hypergraphs [3,4], where each edge is a subset of the set of nodes. Hence higher-order relations between nodes can be directly modeled in a hypergraph, by the means of hyperedges. Since such a mode of representation is closer to the human visual grouping system, hypergraphs have been shown as more effective than graphs to solve many problems in applications of practical interest that includes VLSI design and partitioning [5], parallel scientific computing [6], database design [7] or categorical data clustering [8]. The introduction of hypergraphs in the image processing domain dates back to Bretto et al. [9] and has recently received an increasing attention [10–15]. In particular, hypergraph representations based on the INH (Image Neighborhood Hypergraph) model have achieved competitive results in unsupervised image segmentation [15–17] that outperform graph-based methods. Recently, Ding et al. [14,18] introduced new image hypergraph models and exploited them in an semi-supervised segmentation framework that relies on a transductive setting given by Zhou's random walk formulation on a hypergraph [8] that generalizes the graph random walk formulation.

All the above methods are based on undirected hypergraph models. While the notion of directionality (by the concept of a directed graph) has been developed for a large number of years in the graph theory, a generalization to the hypergraph case has been slow to appear and a standard mathematically formal definition is yet to be defined. First attempts for defining directed hypergraphs [19,20] considered a *hyperarc* (a generalization of a directed graph edge) as a single vertex connecting a set of vertices. In this paper we use a more complete generalization by defining a hyperarc as a connection between two sets of vertices [4]. In addition, the use of directed hypergraphs for practical applications remains marginal and only few papers report using directed hypergraphs for applications such as relational databases [20], natural language parsing [21], or for modeling high-level processes such as biochemical [22], wireless [23,24] or social networks [25]. The main goal of this paper is to introduce the notion of directed hypergraphs in the computer vision domain, and judge its relevance especially compared to simple undirected hypergraphs. For this purpose, we followed the idea of directed graphs that were recently introduced in image segmentation [26–28]. Few previous works that involve some image analysis report using directed hypergraphs, but only to model high-level relationships between visual elements, such as video events [29] or pre-segmented cells motion [30]. None of these representations are directly linked to the image analysis and processing methods or make use of the low-level image data. To the best of our knowledge, and unlike undirected hypergraphs, no image processing technique that involve directed hypergraphs and no directed hypergraph-based

☆ This paper has been recommended for acceptance by Nikos Paragios.
* Corresponding author.
E-mail addresses: aurelien.ducournau@enise.fr (A. Ducournau), alain.bretto@unicaen.fr (A. Bretto).

representation of image data has been investigated. This paper has the objective to investigate these two aspects and show that directed hypergraphs can be useful at image data representation.

Image segmentation is a low-level image analysis process that aims at partitioning an image into a number of disjoint regions, such that the visual features are coherent among the pixels of a single region. Although humans can easily extract meaningful segments from an image, this task remains difficult at computer level, where unsupervised segmentation algorithms are still unable to produce satisfactory results. In fact, fully automated segmentation is known to be an ill-posed problem due to the absence of a clear definition of a semantically meaningful segmentation, and the difficulty to judge its objective quality. Prior information about the image to segment should then be provided to make the segmentation problem well-posed. Such information can be supplied by the user through a set of representative pixels that labels the existing regions in the image. This issue has been addressed as interactive segmentation, and has been successfully used for foreground extraction in intelligent scissors [31], GrabCut [32], or interactive graph cuts [27].

Recent directions in this field rely on graph-based machine learning [33–36]. The main idea is to model the image data by the means of an affinity graph where each edge encodes the similarity of two neighboring pixels in the image. The segmentation problem is then formulated as an energy function minimization, where the target function to estimate is smooth with respect to the underlying graph structure. The graph is represented by a symmetric affinity matrix where each entry models the penalty if the two corresponding pixels belong to separated regions in the segmentation. Segmentation becomes then a labeling problem than can be solved by graph-based learning methods. Such methods include harmonic energy minimization [37], graph min-cuts [27,38], random walks [33], transduction by Laplacian regularization [35], geodesics [39], or watersheds [36].

Traditional variational methods for interactive (or semi-supervised) segmentation such as active contours [40] approach the segmentation problem by minimizing energy functions that favor alignment of the object boundaries with regions of high intensity gradient. Later, Vasilevskiy et al. [26] noted that the direction of the gradient contains valuable information that can improve the segmentation. This concept has been translated to discrete optimization by using graphs with directed edges (by opposition to the above methods that relies on undirected symmetric graphs), resulting in asymmetric affinity matrices. Directed graphs for semi-supervised segmentation have been used in min-cut [27] and random walk [28] algorithms. In practice, asymmetric penalties can help at segmenting thin elongated structures and can further improve the segmentation results in regions of low contrast.

As noted above, and following the idea in [27,28], the objective of this paper is to investigate the reliability of adding directional information in hypergraph-based models and to judge the relevance of using directed hypergraphs as a means for image data representation. Consequently, we introduce the notion of directed hypergraphs in computer vision problems by proposing a directed version of the INH (Image Neighborhood Hypergraph) model. It is important to note that this article is, to the best of our knowledge, the first attempt to use directed hypergraphs in image analysis as a low-level image data representation tool. A first application of this DINH (Directed Image Neighborhood Hypergraph) representation is developed in terms of semi-supervised image segmentation, where the labels of all pixels are predicted according to the labels of pre-defined seed pixels. The problem is expressed in terms of Markov random walks, and a solution is found by estimating the probability that a random walk, starting at an unlabeled pixel, will first hit a given seed pixel. The random walk should be designed with respect to the underlying directed hypergraph structure, and by analogy with the undirected case, we give a formulation of the transition matrix of a random walk in a directed hypergraph. A theoretical solution to the hypergraph-learning process is given by considering the random walk as an *absorbing Markov chain* [41]. The transition matrix also served as a basis to design a simple iterative algorithm for semi-supervised segmentation based on label propagation, which solution converges to the theoretical one. Experiments over the GrabCut [32] dataset achieved interesting results, particularly compared to the same algorithm with transition matrices obtained from graphs and undirected hypergraphs. These results essentially demonstrate that the introduction of directed hypergraphs is relevant in the computer vision domain, and that they provide a richer model of representation than undirected hypergraphs and simple graphs.

The remainder of this paper is organized as follows: we first present some preliminary definitions for undirected and directed hypergraphs (Section 2). We then give our formulation of a random walk in a directed hypergraph as well as its transition matrix (Section 3), and describe our proposed directed hypergraph model to represent the content of an image (Section 4). We then present the theoretical solution to the learning process on directed hypergraphs in terms of random walks, and the iterative label propagation procedure used to approach this solution (Section 5). Finally, some experimental results are given and discussed (Section 6).

2. Preliminary definitions

The general undirected hypergraph theory [3] is well-known in the field of mathematics, and we refer the reader to [3,4] for more details and definitions. We will here focus on the definition of directed hypergraphs.

A *directed hypergraph* (*dirhypergraph*) is an ordered pair:

$$\overrightarrow{H} = (V; \overrightarrow{E} = \{\overrightarrow{e_i} : i \in I\}),$$

where V is a finite *set of vertices* and $\overrightarrow{E}$ is a set of *hyperarcs* with index set $I = \{1, 2, \ldots, M\}$, $M = |\overrightarrow{E}|$. Each hyperarc $\overrightarrow{e_i}$ is written

$$\overrightarrow{e_i} = \left(\overrightarrow{e_i^+} = (e_i^+, i); \overrightarrow{e_i^-} = (i, e_i^-)\right).$$

The set e_i^+ is the set of vertices of $\overrightarrow{e_i^+}$ and the set e_i^- is the set of vertices of $\overrightarrow{e_i^-}$. The vertices of $\overrightarrow{e_i}$ are denoted by $e_i = e_i^+ \cup e_i^-$ and $E = \{e_i : i \in I\}$. The hypergraph $H = (V; E)$ is the *underlying hypergraph* of the dirhypergraph $\overrightarrow{H} = (V; \overrightarrow{E})$. The element $\overrightarrow{e_i^+}$ is called the *tail* of the hyperarc $\overrightarrow{e_i}$, whereas $\overrightarrow{e_i^-}$ is its *head*. A *limb* is either a head or a tail. We do not allow $e_i^+ = \emptyset$ or $e_i^- = \emptyset$ and $e_i^+ \cap e_i^- \neq \emptyset$ for all $i \in I$. The index i stored in each limb of a hyperarc allows to distinguish from two hyperarcs that have the same set of vertices either in its head or in its tail.

A dirhypergraph $\overrightarrow{H} = (V; \overrightarrow{E})$ can be represented by two incidence matrices, the positive (or outer) incidence matrix $\mathcal{H}_+$ and the negative (or inner) incidence matrix $\mathcal{H}_-$, representing respectively the tails and the heads of the hyperarcs. Entries of $\mathcal{H}_+$ are given by $(\mathcal{H}_+)_{ij} = h^+(v_i, \overrightarrow{e_j}) = 1$ if $v_i \in e_j^+$ and 0 otherwise. Entries of $\mathcal{H}_-$ are given by $(\mathcal{H}_-)_{ij} = h^-(v_i, \overrightarrow{e_j}) = 1$ if $v_i \in e_j^-$ and 0 otherwise. We will consider that any hyperarc e can be weighted by a positive function $w(\overrightarrow{e})$ (called the *weight* of hyperarc $\overrightarrow{e}$) representing the importance of the hyperarc in the dirhypergraph structure, and we will note W the diagonal matrix containing the hyperarc weights. The positive (or outer) degree of a vertex $v_i \in V$ is given by $d^+(v_i) = \sum_{\overrightarrow{e_j} \in \overrightarrow{E}} w(\overrightarrow{e_j}) h^+(v_i, \overrightarrow{e_j})$. The negative (or inner) degree of v_i is defined as $d^-(v_i) = \sum_{\overrightarrow{e_j} \in \overrightarrow{E}} w(\overrightarrow{e_j}) h^-(v_i, \overrightarrow{e_j})$. Let D_{v^+} and D_{v^-} be the diagonal matrices containing respectively the positive and negative degrees of the vertices. We define the positive and negative degrees of a hyperarc $\vec{e_j}$ by $\delta^+(\vec{e_j}) = |e_j^+|$ and $\delta^-(\vec{e_j}) = |e_j^-|$. Let D_{e^+} and D_{e^-} be the diagonal matrices containing the positive and negative degrees of the hyperarcs. The transpose of a matrix (or vector) A will be noted A^T.

A *directed path or hyperpath* from x to y in $\vec{H} = (V; \vec{E})$ is a sequence $P_{x,y} = (x = v_1, e_1, v_2, e_2, v_3, \ldots, v_t, e_t, v_{t+1} = y)$ such that $x = v_1 \in e_1^+$, $y = v_{t+1} \in e_t^-$ and $v_i \in e_{i-1}^- \cap e_i^+$ for $i \geqslant 2$.

3. Random walks in directed hypergraphs

3.1. Introduction to Markov random walks

A random walk [33,42,43] is a particular case of a Markov random chain, a random process that consists on visiting a certain number of locations (or states) by taking random steps. Consider a starting location u. Then the next location visited by the random walk is taken randomly (following a given probability law) among all the neighbors of u. A random walk of length t is then a sequence of $t+1$ locations $\{v_0, v_1, \ldots, v_t\}$ with $v_0 = u$ the starting location and $v_t = v$ the ending location. The random walks are useful in a machine learning manner, where the different locations are data points to classify using a few sample seed points. The learning process can be interpreted by the following assumption: given a random walk starting at an unlabeled location, what is the probability that it first reaches each of the seed points? The final label of the starting location is then taken from the ending seed point of the random walk with the highest probability. In a random walk each step is taken independently from the previous steps, and consequently its behavior is completely determined by a *transition* probability matrix P where entry P_{ij} is the probability for a random walk at location v_i to "jump" to the location v_j. The "t-step" probabilities (representing the probabilities that a random walk starting at location v_i will reach the location v_j after exactly t steps) are simply given by the tth power of the transition matrix P^t.

When the random walk is defined on a graph [33,43] the transition matrix is computed with respect to the underlying graph structure. In practice the random walk matrix is given by $P = D^{-1}A$, where D denotes the diagonal matrix containing the vertex degrees and A is the graph adjacency matrix. It has been shown in many occasions that random walks on graphs have a strong connection with usual electric networks problems [33,42,44]. In this paper we are interested in designing such a learning process on a directed hypergraph. The next sections will give a formulation of the transition matrix associated to a natural random walk in such a directed hypergraph.

3.2. Random walks in undirected hypergraphs

Following the concepts introduced by Zhou [8], we can associate a natural random walk to an undirected hypergraph with the following transition rule: given the current position $u \in V$, we first choose a hyperedge $e \in E$ over all the hyperedges incident to u with a probability proportional to $w(e)$. We then choose a vertex $v \in V,\ v \neq u$ uniformly at random. With the definitions given by Zhou in [8], the probability $p(u, v)$ associated to that transition rule is

$$p(u,v) = \sum_{e \in E} w(e) \frac{h(u,e)}{d(u)} \frac{h(v,e)}{\delta(e)}. \tag{1}$$

Obviously, a nonzero transition probability between u and v exists only if the two vertices are linked by at least one hyperedge (otherwise $h(u,e)$ or $h(v,e)$ equals 0). The normalization by the degrees of u and e is motivated by, respectively, that a vertex with a large degree has a smaller chance to choose a distinct hyperedge for the transition, and that the transition between u and an incident vertex v is chosen uniformly at random. The transition matrix P of the random walk is then defined by $P = D_v^{-1}\mathcal{H}WD_e^{-1}\mathcal{H}^T$.

3.3. Generalization to the directed case

At that time, very little attention has been dedicated to the study of random walks on hypergraphs, and especially on directed hypergraphs. A first tentative to define a random walk on a directed hypergraph was reported in [45], but this definition comes from the interpretation of a directed hypergraph as a bipartite graph, as we propose a direct formulation. More importantly, only the undirected case is further considered in [45] and consequently the application of a random walk on a directed hypergraph is not investigated. In [46], the authors propose the extension of a specific random walk named loop-erased random walk, but only consider directed hypergraphs where the tail is reduced as a single vertex. Consequently, we propose here a formal definition of a random walk on a directed hypergraph, as well as its associated transition matrix.

By analogy with the random walks defined in [8] for undirected hypergraphs, we associate a random walk to a dirhypergraph $\vec{H} = (V; \vec{E})$ that has the following natural transition rule. Given the current position $u \in V$, we choose a hyperarc $\vec{e} \in \vec{E}$ such that $u \in e^+$ with a probability proportional to $w(\vec{e})$. If we consider that a transition can only be made from a tail to a head of a hyperarc (following the definition a directed hyperpath given above), then we choose a vertex $v \in e^-$ uniformly at random. The probability $p(u, v)$ associated to that transition rule can be formulated as

$$p(u,v) = \sum_{\vec{e} \in \vec{E}} w(\vec{e}) \frac{h^+(u,\vec{e})}{d^+(u)} \frac{h^-(v,\vec{e})}{\delta^-(\vec{e})}. \tag{2}$$

With this formulation, a nonzero transition probability between u and v exists only if $\exists \vec{e}$ such that $u \in e^+$ and $v \in e^-$, which follows the intuitive idea that a transition can only be done in the direction given by a hyperarc. The probability is normalized by the outer degree of u to represent the probability of choosing a distinct hyperarc between those which contain u in their tails. The inner degree of the hyperarc $\vec{e}$ is also introduced to take into account that a vertex is chosen uniformly at random in the head of $\vec{e}$. In matrix notation, the transition matrix P of the directed random walk is defined by

$$P = D_{v^+}^{-1}\mathcal{H}_+WD_{e^-}^{-1}\mathcal{H}_-^T. \tag{3}$$

Proposition 1. *The matrix P given by Eq. (3) is stochastic, i.e. $\forall i,j, p_{ij} \geqslant 0$ and $\forall i, \sum_j p_{ij} = 1$, and is then suitable for the definition of a random walk.*

Proof. It is easy to verify that : for all $u, v \in V,\ p(u,v) \geqslant 0$. Now:

$$\begin{aligned}
\sum_{v \in V} p(u,v) &= \sum_{v \in V}\sum_{\vec{e} \in \vec{E}} w(\vec{e}) \frac{h^+(u,\vec{e})}{d^+(u)} \frac{h^-(v,\vec{e})}{\delta^-(\vec{e})} \\
&= \sum_{\vec{e} \in \vec{E}} w(\vec{e}) \frac{h^+(u,\vec{e})}{d^+(u)} \sum_{v \in V} \frac{h^-(v,\vec{e})}{\delta^-(\vec{e})} \\
&= \sum_{\vec{e} \in \vec{E}} w(\vec{e}) \frac{h^+(u,\vec{e})}{d^+(u)} \sum_{v \in V} \frac{h^-(v,\vec{e})}{|e^-|} = \sum_{\vec{e} \in \vec{E}} w(\vec{e}) \frac{h^+(u,\vec{e})}{d^+(u)} \\
&= \sum_{\vec{e} \in \vec{E}} w(\vec{e}) \frac{h^+(u,\vec{e})}{\sum_{\vec{e} \in \vec{E}} w(\vec{e}) h^+(u,\vec{e})} = 1. \qquad \square
\end{aligned}$$

4. A directed hypergraph image model

The INH (Image Neighborhood Hypergraph) model has been found very effective at representing the image content [11,15]. In this section we present an improvement of the INH that takes into account directed relationships between pixels.

4.1. The INH (Image Neighborhood Hypergraph) representation

Let $I : V \subseteq \mathbb{Z}^2 \longrightarrow F \subseteq \mathbb{Z}^n$ be an image. Elements of V are the collection of the image pixels, and elements of F are the visual features associated to each pixel. A distance d on V defines a grid (a connected, regular graph, without both loop and multi-edge, associated with a regular lattice $\mathbf{L}$ of $\mathbb{R}^n$). In this contribution, we will be concerned only with 8-connected grids defined by the distance $d(v, v') = \max\{|x - x'|, |y - y'|\}$, where (x,y) and (x',y') denote respectively the spatial coordinates of v and v' on the grid. Thus, we define the β-neighborhood of a pixel $v \in V$ by:

$$\Gamma_\beta(v) = \{v' \in V | d(v, v') \leqslant \beta\}. \tag{4}$$

Let d' be a distance measure on F, we have a neighborhood relation on an image defined for each pixel v by:

$$\Gamma_{\lambda,\beta}(v) = \{v' \in \Gamma_\beta(v) | d'(F(v), F(v')) \leqslant \lambda\}. \tag{5}$$

Here β and λ are real values, called respectively *spatial threshold* and *feature threshold*. Thanks to this neighborhood relation, to each image we can associate a hypergraph called *Image Neighborhood Hypergraph* (INH) [47]:

$$H_{\Gamma_{\lambda,\beta}} = (V; (\{v\} \cup \Gamma_{\lambda,\beta}(v))_{v \in V}). \tag{6}$$

Each pixel v in the image generates a distinct hyperedge $e(v)$, and v is called the *center* of $e(v)$. In this work, the feature distance between two pixels will be computed as $d'(F(v), F(v')) = |I(v) - I(v')|$, where $I(v)$ denotes the gray level of pixel v. The choice of the thresholds β and λ will be discussed in Section 6.

4.2. The Directed INH (DINH) model

In directed graphs (digraphs) for image representation, two neighboring pixels x and y in the image are linked by two distinct directed edges (arcs) (x,y) and (y,x). If the same weight is assigned to both arcs than the digraph reduces to an undirected graph. It was then suggested [27,28] to use two different weighting functions w_1 and w_2 such that the penalty assigned to an arc (x,y) should be equal to $w_1((x,y))$ if $I(x) < I(y)$, and to $w_2((x,y))$ if $I(x) \geqslant I(y)$, leading to an asymmetric adjacency matrix. We will follow that concept by building two different sets of hyperarcs in our following Directed Image Neighborhood Hypergraph (DINH) model. For an image I with set of pixels V, each pixel $v_i \in V$ will generate two distinct hyperarcs $\vec{e_1}$ and $\vec{e_2}$:

$$\begin{aligned}
&\vec{e_1}(v_i) = (e_1^+(v_i); e_1^-(v_i)),\\
&e_1^+(v_i) = (\{v_i\} \cup \{v' \in \Gamma_{\lambda,\beta}(v_i) | I(v') < I(v_i)\}),\\
&e_1^-(v_i) = (\{v' \in \Gamma_{\lambda,\beta}(v_i) | I(v') \geqslant I(v_i)\}),
\end{aligned} \tag{7}$$

$$\begin{aligned}
&\vec{e_2}(v_i) = (e_2^+(v_i); e_2^-(v_i)),\\
&e_2^+(v_i) = (\{v_i\} \cup \{v' \in \Gamma_{\lambda,\beta}(v_i) | I(v') > I(v_i)\}),\\
&e_2^-(v_i) = (\{v' \in \Gamma_{\lambda,\beta}(v_i) | I(v') \leqslant I(v_i)\}).
\end{aligned} \tag{8}$$

Since the center pixel v_i is in each tail of both hyperarcs, it will exist a nonzero transition probability between v_i and every pixel in its $\Gamma_{\lambda,\beta}$ neighborhood. The presence of the two types of hyperarc is necessary to allow a transition from regions of low intensity to regions of high intensity (this transition is represented by the hyperarc $\vec{e_1}$), and also a transition from regions of high intensity to regions of low intensity (represented by the hyperarc $\vec{e_2}$). Finally, the Directed Image Neighborhood Hypergraph (DINH) associated to an image can be computed as:

$$\vec{H}(V) = \left(V; \vec{E} = (\vec{e_1}(v))_{v \in V} \cup (\vec{e_2}(v))_{v \in V}\right). \tag{9}$$

4.3. Choice of hyperarc weights

With the definition of the hyperarcs of the DINH model given by Eqs. (7) and (8), assigning the same weights to $\vec{e_1}$ and $\vec{e_2}$ will lead to a symmetric transition matrix associated to the random walk formulation (2), equivalent to a transition matrix obtained from an INH model, up to a factor 2. Consequently, a hyperarc $\vec{e}(v)$ centered at a pixel v will be weighted by the following weighting function w:

$$\begin{aligned}
w : \vec{E} \quad &\longrightarrow \mathbb{R}^+,\\
\vec{e}(v) = (e^+(v); e^-(v)) \quad &\mapsto \exp(\overline{d'(v, v')}_{v' \in e^-(v)}).
\end{aligned} \tag{10}$$

In other words, each hyperarc will be weighted by the exponential of the average value of the feature distance (defined in Section 4.1) between the center pixel v (in the tail) and every pixel in the head of the hyperarc. We should note that this definition favors transitions between pixels that are close according to the given feature distance. By definition, the tails of e_1 and e_2 centered at a pixel v are different, and so are the weights associated to them. Consequently, the transition matrix of the directed random walk will be asymmetric.

5. Segmentation by semi-supervised learning

We consider the image segmentation problem in a transductive setting, in which a set of known labels of some pixels (called seeds) are used to predict the labels for all other pixels. More mathematically, let us reorganize the image pixels as $V = \{v_1, v_2, \ldots, v_L, \ldots, v_N\}$, such that $V_L = \{v_i\}_{i=1}^{L}$ is the labeled pixels set and $V_U = \{v_i\}_{i=L+1}^{N}$ is the unlabeled pixels set. Note $\mathcal{L} = \{1, \ldots, l\}$ (with $l \leqslant L$) the set containing the labels, and $y : V \to \mathcal{L}$ the function associating to each pixel $v_i \in V$ its label $y(v_i)$ in the final segmentation. Our goal is to estimate $y(v_i)$ for each unlabeled pixel v_i.

In our framework, the pixels set is taken as the vertex set of the directed hypergraph defined by Eq. (9), and let P be the transition matrix of the random walk defined on this dirhypergraph by Eq. (3). Let $f^{(k)} : V \to [0, 1]$ be the function associating to each pixel v_i its membership $f^{(k)}(v_i)$ to the label k, i.e. the probability for the pixel v_i to belong to label k in the final segmentation. If $v_j \in V_L$ is a labeled pixel, we set $f^{(k)}(v_j) = 1$ if the (known) label of v_j is k, and 0 otherwise. We want to estimate the value of $f^{(k)}$ for each unlabeled pixel v_i, so its label $y(v_i)$ will be taken as the k which maximizes $f^{(k)}(v_i)$.

With our random walk interpretation of this learning process, $f^{(k)}(v_i)$ will be calculated as the probability that a random walk, given by the transition matrix P and starting at location v_i, will first reach a labeled location belonging to the label k, *before* hitting a seed location with a different label. It is not difficult to see that simulating such a random walk and see what seeded point it reaches first may be impractical, because the number of unlabeled pixels is in practice considerably higher than the number of seed pixels. In addition, it is subject to random biases, since it is possible for a random walk to take a direction it has a small probability to take, and then be trapped in a sub-optimal solution. This section presents two ways of resolving this problem by estimating the probabilities $f^{(k)}$, one theoretical and one more practical, and the summary of the segmentation algorithm.

5.1. Random walks as absorbing Markov random chains

The solution of the learning process in terms of random walk is to consider the latter as an *absorbing* Markov chain. More details about the absorbing Markov chains theory can be found in [41]. In an absorbing Markov chain (given by its transition matrix P'), some of the locations are *absorbing* states that can be considered as traps, where the random walk ends when it reaches one of those absorbing states. More formally, $P'_{ii} = 1$ and $P'_{ij} = 0 \forall j$ when v_i is an absorbing state. Then if we pose all the seed labeled locations as absorbing states, and if we assume that every random walk starting at a given unlabeled location (which is called a *transient* state in terms of Markov random chains) can reach at least one absorbing state (not necessary after just one step), then the random walk is an absorbing Markov chain, with some interesting properties [42]. Thanks to the reorganization of the pixels set by grouping seed and unlabeled pixels together, we can split the transition matrix P of the Markov chain in the following canonical form:

$$P = \begin{bmatrix} P_{LL} & P_{LU} \\ P_{UL} & P_{UU} \end{bmatrix},$$

where P_{UL} and P_{UU} represent the submatrices of P restricted to the probabilities between respectively the unlabeled and seed locations, and between all the unlabeled locations. P_{LL} and P_{LU} are the submatrices corresponding to the transition probabilities between, respectively, all the seed locations, and the seed and unlabeled locations. Since the seed pixels are considered as absorbing states, P_{LL} must reduce to the $L \times L$ identity matrix I_L and P_{LU} to $\mathbf{0}$, a matrix with all entries equal to 0. It is not difficult to see that this is not the case with our formulation of the transition matrix P of a random walk in a directed hypergraph (see Eq. (3)). In fact, this formulation does not depend on the seeds, which will be placed *after* the construction of P. It is not a problem, because the submatrices P_{LL} and P_{LU} are not useful for the learning process (since we do not have to start a random walk from already labeled locations) and do not intervene in the following. Therefore, P_{LL} and P_{LU} can easily be fixed to, respectively, I_L and $\mathbf{0}$ after placing the seeds.

The matrix $N = (I_U - P_{UU})^{-1}$ is called the *fundamental matrix* for the absorbing chain given by P (in this case I_U denotes the identity matrix of size $N - L \times N - L$). It can be found in [41] that the matrix $(I_U - P_{UU})$ is invertible and thus the matrix N exists. The entry N_{ij} can be interpreted as the expected number of times that a random walk starting at v_i will pass through v_j before being absorbed by a seed location. Then the vector $\mathbf{t} = N\mathbf{1}$ (where $\mathbf{1}$ is a column vector of all 1's) gives the expected number of steps before absorption for each starting state. Consequently, let B be the $N \times L$ matrix which entry B_{ij} gives the probability that a random walk starting at v_i will be absorbed by the absorbing state v_j. Then B is calculated by:

$$B = NP_{UL} = (I_U - P_{UU})^{-1} P_{UL}. \tag{11}$$

This can be interpreted as follows: to get the probability of starting at v_i and ending up a given absorbing state v_j, we add up the probabilities of going to v_j from all the transient states, weighted by the number of times we expect to be in those transient states starting from v_i.

Now that we have the absorption probabilities from each starting state, we have to estimate the label probabilities $f^{(k)}$. Let $\mathbf{f}^{(k)} = (f^{(k)}(v_1), \ldots, f^{(k)}(v_L), \ldots, f^{(k)}(v_N))^T$ be the vector containing the label memberships of all pixels for the label k. As well as the matrix P, we can split $\mathbf{f}^{(k)}$ as:

$$\mathbf{f}^{(k)} = \left((\mathbf{f}_L^{(k)})^T, (\mathbf{f}_U^{(k)})^T \right)^T.$$

In full generality, the probability U_{im} that a random walk P starting at v_i will reach the absorbing state v_m is given by the solution to the system of equations [41]

$$U_{im} = \sum_{v_j \in V} P_{ij} U_{jm}. \tag{12}$$

Since the value $f^{(k)}(v_i)$ is interpreted as the probability that a random walk starting at v_i will first reach the location belonging to the label k, it can be calculated as the sum of the probabilities of reaching each absorbing state labeled with k. If we note V_k the set of all labeled pixels of label k, thus we have:

$$\begin{aligned} f^{(k)}(v_i) &= \sum_{v_m \in V_k} U_{im}, = \sum_{v_m \in V_k} \sum_{v_j \in V} P_{ij} U_{jm}, = \sum_{v_j \in V} P_{ij} \sum_{v_m \in V_k} U_{jm}, \\ &= \sum_{v_j \in V} P_{ij} f^{(k)}(v_j). \end{aligned} \tag{13}$$

We can clearly see that $f^{(k)}(v_i)$ is the harmonic average between all the values of $f^{(k)}$ among the neighborhood of v_i, because $\sum_j P_{ij} = 1$. Consequently, $f^{(k)}$ is a *harmonic function* with domain the state space of the absorbing Markov chain P. If we consider the absorbing states (i.e. the seed locations) of P as the boundaries of the harmonic function $f^{(k)}$, then the solution to the learning process is the same as the one to the *combinatorial Dirichlet problem* [42,44], which is to find a harmonic function subject to its boundary values. Those values are known, because obviously we set $f^{(k)}(v_j) = 1$ if the label of the seed pixel v_j is k, and 0 otherwise. The solution to the Dirichlet problem can be expressed in terms of absorbing Markov chains [42], since $f^{(k)}$ is harmonic for P means that $P\mathbf{f}^{(k)} = \mathbf{f}^{(k)}$, which implies that

$$\forall n, \quad P^n \mathbf{f}^{(k)} = \mathbf{f}^{(k)}, \tag{14}$$

where P^n is the nth power of matrix P. It has been shown [41] that the nth power of P will approach a matrix of the form

$$P^\infty = \begin{bmatrix} I & \mathbf{0} \\ B & \mathbf{0} \end{bmatrix},$$

so we can rewrite Eq. (14) as follows:

$$\mathbf{f}^{(k)} = P^\infty \mathbf{f}^{(k)},$$

$$\begin{bmatrix} \mathbf{f}_L^{(k)} \\ \mathbf{f}_U^{(k)} \end{bmatrix} = \begin{bmatrix} I & \mathbf{0} \\ B & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{f}_L^{(k)} \\ \mathbf{f}_U^{(k)} \end{bmatrix},$$

$$\begin{aligned} \mathbf{f}_U^{(k)} &= B\mathbf{f}_L^{(k)}, \\ \mathbf{f}_U^{(k)} &= (I - P_{UU})^{-1} P_{UL} \mathbf{f}_L^{(k)}. \end{aligned} \tag{15}$$

The last equation is fundamental, because it allows us to compute the value of each function $f^{(k)}$ at each unlabeled pixel. The final labeling of a given pixel v_i can be obtained by taking the highest probabilities among all the labels, i.e.

$$y(v_i) = \mathrm{argmax}_k f^{(k)}(v_i). \tag{16}$$

5.2. A label propagation approach

A practical means to estimate the theoretical solution of the learning process given by Eq. (15) (since its direct computation is difficult due to the size of the matrices in use) is to adopt an iterative label propagation behavior [34,48], that has been chosen in this paper for its computational feasibility and quickness of convergence. It is based on the assumption that at each iteration of the algorithm, each unlabeled pixel will update its label membership by "absorbing" the label information given by the pixels in its spatial neighborhood. Intuitively, the pixel v_i should learn more information from a pixel which is more likely to belong to the same semantically meaningful object. Note p_{ij} the likelihood that v_i and v_j share the same label in the final segmentation, then the k label membership of an unlabeled pixel v_i at iteration step t will be computed by

$$f^{(k)(t)}(v_i) = \sum_{v_j \in V} p_{ij} f^{(k)(t-1)}(v_j). \tag{17}$$

As that notation suggests, the likelihood p_{ij} will be taken from the transition matrix P defined by Eq. (3), with $p_{ij} = P_{ij}$. Since the labels of the seed pixels are user-defined labels, they should not be updated and can be clamped at each iteration. Consequently, we can split $\mathbf{f}^{(k)}$ and P as

$$\mathbf{f}^{(k)(t)} = \left(\left(\mathbf{f}_L^{(k)(t)}\right)^T, \left(\mathbf{f}_U^{(k)(t)}\right)^T\right)^T, \qquad P = \begin{bmatrix} P_{LL} & P_{LU} \\ P_{UL} & P_{UU} \end{bmatrix},$$

where $\mathbf{f}_L^{(k)(t)}$ and $\mathbf{f}_U^{(k)(t)}$ correspond to the predicted label memberships for label k of the labeled and unlabeled labels respectively. P_{LU} and P_{UU} represent the submatrices of P restricted to the transition probabilities between respectively the labeled and unlabeled locations, and between all the unlabeled locations. Then the update rule given by Eq. (17) can be rewritten as [34,48]:

$$\mathbf{f}_U^{(k)(t)} = P_{UL}\mathbf{f}_L^{(k)} + P_{UU}\mathbf{f}_U^{(k)(t-1)}. \tag{18}$$

Here the iteration step of $\mathbf{f}_L^{(k)}$ has been omitted since it remains the same in all iteration steps. Therefore, the labels of the unlabeled pixels can be predicted using Eq. (18) until a convergence is achieved. The final labeling after the learning process at a pixel v is then given by $y(v) = \text{argmax}_k \mathbf{f}_U^{(k)}(v)$. We will now show that the label memberships given by this iterative procedure converge to a solution that is equivalent to the theoretical solution found by considering the random walk as an absorbing Markov chain.

Proposition 2. *The iterative update rule from Eq. (18) converges to* $\mathbf{f}_U^{(k)} = (I - P_{UU})^{-1} P_{UL} \mathbf{f}_L^{(k)}$.

Proof. When the number of iterations approaches $+\infty$, the iterative update rule from Eq. (18) leads to

$$\mathbf{f}_U^{(k)} = \lim_{t\to\infty} \left(\sum_{i=1}^{t} P_{UU}^{i-1}\right) P_{UL}\mathbf{f}_L^{(k)} + P_{UU}^t \mathbf{f}_U^{(k)(0)},$$

where $\mathbf{f}_U^{(k)(0)}$ denotes the initial configuration of the label membership of the unlabeled pixels, and P_{UU}^t represents the tth power of the matrix P_{UU}. We now have to show that the second term $P_{UU}^t \mathbf{f}_U^{(k)(0)} \to 0$. If we assume that P is row-stochastic, i.e. $\forall i, j$, we have $0 \leqslant P_{ij} \leqslant 1$ and $\sum_j P_{ij} = 1$, since P_{UU} is a sub-matrix of P it follows that (with $(P_{UU})_{ij}$ denoting the (i,j)th entry of matrix P_{UU})

$$\exists i, \quad \sum_{j=1}^{N-l} (P_{UU})_{ij} < 1.$$

Since there is N pixels and L labeled points, note $n = N - L$ and P_{UU} is of size $n \times n$. Pose r_i as the sum of the ith row of P_{UU}, i.e. $r_i = \sum_j (P_{UU})_{ij}$. P_{UU} is nonnegative, so from Perron-Frobenius theorem [49] we know that the spectral radius $\rho(P_{UU})$ (i.e. the maximum eigenvalue of P_{UU}) satisfies $\rho(P_{UU}) \leqslant \max_i r_i$, because $\forall i \sum_j P_{ij} = 1$ we have $r_i \leqslant 1$, so $\rho(P_{UU}) \leqslant 1$. Minc theorem [49] also states that

$$\rho(P_{UU}) \leqslant \max_i \left(\sum_{m=1}^{n} (P_{UU})_{im} r_m\right) \Big/ r_i.$$

Recall that we have $r_i \leqslant 1 \forall i$ and that $\exists i, r_i < 1$. So it comes that

$$\left(\sum_{m=1}^{n} (P_{UU})_{im} r_m\right) \Big/ r_i < \left(\sum_{m=1}^{n} (P_{UU})_{im}\right) \Big/ r_i = 1,$$

and consequently

$$\rho(P_{UU}) \leqslant \max_i \left(\sum_{m=1}^{n} (P_{UU})_{im} r_m\right) \Big/ r_i < 1.$$

Because the spectral radius of P_{UU} is less than 1, thus

$$\lim_{t\to\infty} P_{UU}^t = 0, \qquad \lim_{t\to\infty} \sum_{i=1}^{t} P_{UU}^{t-1} = (I - P_{UU})^{-1}.$$

Consequently $P_{UU}^t \mathbf{f}_U^{(k)(0)} \to 0$, and therefore

$$\begin{aligned} \mathbf{f}_U^{(k)} &= \lim_{t\to\infty} \left(\sum_{i=1}^{t} P_{UU}^{i-1}\right) P_{UL}\mathbf{f}_L^{(k)} + P_{UU}^t \mathbf{f}_U^{(k)(0)}, \\ &= (I - P_{UU})^{-1} P_{UL} \mathbf{f}_L^{(k)}. \qquad \square \end{aligned}$$

Another version of the proof can be found in [34]. Since $\mathbf{f}_L^{(k)}$ is immutable, the iterative update rule converges to a unique fixed point and gives us a theoretical guarantee of the feasibility of the algorithm. In addition, this fixed point is equivalent as the theoretical solution given by Eq. (15), which shows that the iterative algorithm is able to solve the labeling problem in terms of random walks.

5.3. Segmentation algorithm

The proposed semi-supervised segmentation framework can be summarized as follows:

1. Obtain a set of labeled pixels, either automatically or interactively.
2. For each labeled pixel u with label k, set $\mathbf{f}_L^{(k)}(u) = 1$.
3. For each unlabeled pixel v and for each label k, set $\mathbf{f}_U^{(k)(0)}(v) = 0$.
4. Build $\overrightarrow{H}$ as the DINH representation of the image following Eq. (9).
5. Compute the directed transition matrix P following Eq. (3).
6. Set $t = 1$ and $\mathbf{f}_U^{(k)(1)} = \mathbf{f}_U^{(k)(0)}$.
7. While $t < t_{max}$ or $|\mathbf{f}_U^{(t-1)} - \mathbf{f}_U^{(t)}| > \delta$,
   (a) For each label k, update $\mathbf{f}_U^{(k)}$ by:
   $\mathbf{f}_U^{(k)(t)} = P_{UL}\mathbf{f}_L^{(k)} + P_{UU}\mathbf{f}_U^{(k)(t-1)}$.
   (b) Set $t = t + 1$ and $\mathbf{f}_U^{(k)(t)} = \mathbf{f}_U^{(k)(t-1)}$.
8. For each pixel v, give it label k such that

   $\text{argmax}_k \mathbf{f}_U^{(k)(t)}(v)$.

The stopping condition of our iterative algorithm is here controlled by the parameter $t_{max} > 0$ if one wants to limit the maximum number of iterations, or the parameter $\delta > 0$ if one wants to achieve a convergence over the label memberships.

6. Experiments

In order to evaluate the reliability of our approach and to judge the relevance of our directed hypergraph model, we conducted experiments on the commonly used Microsoft GrabCut dataset [32]. It includes a set of 50 real images, as well as ground-truth segmentations that can be used as a comparison basis. Seed regions are also provided in the form of trimaps, where white, dark gray and light gray areas denote respectively the labeled object, the labeled background and the unlabeled region. Some original images from the GrabCut dataset and their corresponding trimaps can be found in Fig. 1. One may note that this dataset (and particularly the seeds) is of a particular type, where the unlabeled data lies only within a narrow band around the real objects boundaries. Consequently, it is reasonable to think that an algorithm exploiting the particular shape of the seeds could provide a good segmentation. But nevertheless, none of the algorithms compared in this experimental section makes use of a priori information on the nature of the seeds, or involves a particular spatial distance to the labeled pixels in the energy function to minimize. For this reason, the GrabCut dataset remains of a considerable interest in the evaluation of an algorithm that can be used in an interactive framework [35], and some examples are difficult due to the lack of seed pixels in some regions (see for instance the last image of Fig. 1). Furthermore, this dataset (and its original seeds) remains very standard in the supervised segmentation field, as many state-of-the-art algorithms evaluate their performance on this dataset, and thus allows us to quantitatively compare our proposal to these algorithms. One may argue that the seeds can be modified to judge the stability of the algorithm (as in [36]), but we chose to use the original seeds for the sake of consistency, since the evaluation measures reported in many state-of-the-art articles that exploit this dataset are also computed with the original seeds, thus providing a benchmark for this dataset and allowing a direct comparison with our proposal.

For a quantitative comparison, the performance of each semi-supervised segmentation algorithm will be measured by the average error rate over all 50 images in the dataset. The error rate is computed as the ratio of the number of wrongly classified pixels to the total number of pixels to classify [32,50]. The wrongly classified pixels can be directly obtained from the ground truth labeling provided in the dataset (see Fig. 1). Please note that the foreground object is surrounded by a thin band of grey pixels corresponding to an uncertainty about the precise location of the boundaries of the object in the ground truth (since it has been annotated by multiple persons). Consequently, these pixels are not counted in the error rate to ensure its consistency. Again, other performance measures can be computed [51], but the error rate we use here is widely reported in many state-of-the-art papers, thus allowing a direct comparison with our algorithm. Results of our approach using the DINH model have been first compared to those obtained with the same iterative algorithm (see Section 5.3), but with different transition matrices exploiting different representation models (see Section 6.1), to judge the reliability of directed hypergraphs in this context. Second, the performance of our proposed segmentation approach is compared to some state-of-the-art algorithms (see Section 6.2) to judge its efficiency as a standalone segmentation method. And finally, some examples of multi-label segmentation are given (Section 6.3).

6.1. Comparison with different representation models

To judge the reliability of using directed hypergraphs as a means of image representation (and thus our proposed DINH model) independently from the learning method used, we first compared the results of our approach with those obtained with the same iterative algorithm (see Section 6.1) but with different transition matrices built from:

- A typical undirected graph model [34], which connects every pixel with its 8 neighbors in the image. The edge between two pixels x and y is weighted by a standard Gaussian kernel $w(x,y) = \exp(-\alpha|I(x) - I(y)|)$ exploiting the image intensity, as well as our approach. The standard value 1 has been used for the α parameter since it provided the best results. The transition matrix is computed as in [34] and is equivalent as a graph random walk formulation.
- A directed graph model obtained from the DINH, generated by converting every hyperarc of the DINH representation by a set of directed graph edges (arcs) starting from the center pixel of the hyperarc. The weights associated to each arc is the same as the hyperarcs weights defined in Section 4.3. The parameters used for the arcs construction are obviously the same as the ones used for the DINH. We can observe that with this definition a directed hypergraph as defined in the first works in this direction [19,20] reduced to the present directed graph model.
- An undirected INH model as described in Section 4.1. The INH model have been previously successfully used for image segmentation [11,16], and the transition matrix can be obtained by Zhou's random walk formulation [8] for undirected hypergraphs, as suggested in [18]. The β and λ parameters used for this representation are the same as in our DINH model.

The proof of convergence and the theoretical properties of the algorithm with these transition matrices can be found in [34] (for the graph version) and [14] (for the undirected hypergraph version). The remaining parameters used along these experiments have been set as follows:

- For the parameters that control the stopping criterion of the iterative algorithm (see Section 5.3), we chose not to limit the number of iterations and fixed the δ parameter to 0.1 which is in practice enough to achieve convergence.
- The standard value $\beta = 1$ has been used for the spatial threshold in INH and DINH representations, since it has been shown in many occasions [11,16] that increasing this parameter has a small influence over the segmentation output.
- Finally, the λ parameter for the feature threshold has been computed in an adaptive way [16] as $\lambda(v) = \left(\frac{\overline{I(v')}}{\sigma(I(v'))}\right)_{v' \in \Gamma_\beta(v)}$ at pixel v, where $\overline{I(v')}$ and $\sigma(I(v'))$ design respectively the mean and the standard deviation of the intensities of each pixel (including v) in the β-neighborhood of v.
- One should note that with the given INH and DINH definitions (see Section 4), two neighboring pixels in the image can be disconnected in the corresponding hypergraph, which will result in a zero transition probability in the propagation matrix. To overcome this limitation and to guarantee that the random walk will reach at least one seed pixel (see Section 5.1), we chose to compute such a probability as a very small value (typically $10e^{-4}$). This is equivalent to adding a hyperarc (or hyperedge) of weight $10e^{-4}$ between two pixels that are neighbors in the image but not in the DINH (or INH).

Table 1 compiles the average error rates obtained on the 50 images of the GrabCut [32] dataset with our iterative algorithm with the four different transition matrices described above. When we observe these values we can clearly see that the best results are obtained with our proposed directed hypergraph model compared to the previously cited ones. Consequently, our label propagation behavior has been found more effective when the transition matrix of the associated random walk is built by the means of our proposed directed hypergraph model. Since this model is based on the existing INH model, we can conclude when looking at the results that simply adding a directional information and asymmetric weights to the underlying hypergraph structure provided by the INH can help at improving the segmentation. It is also interesting to observe that the average error rate is higher in the case of a graph structure, which confirms the fact that hypergraph-based model are more accurate than graph-based ones for image content representation. This can be explained by the ability of hypergraphs to take into account multiple relationships between elements, when graphs are only able to maintain pairwise affinities. Consequently, a hypergraph is able to handle relationships that are not only based on pixel-to-pixel comparison, but on the comparison of whole regions of neighboring pixels, and thus allows some pixels to be linked by multiple relationships from different points of view.

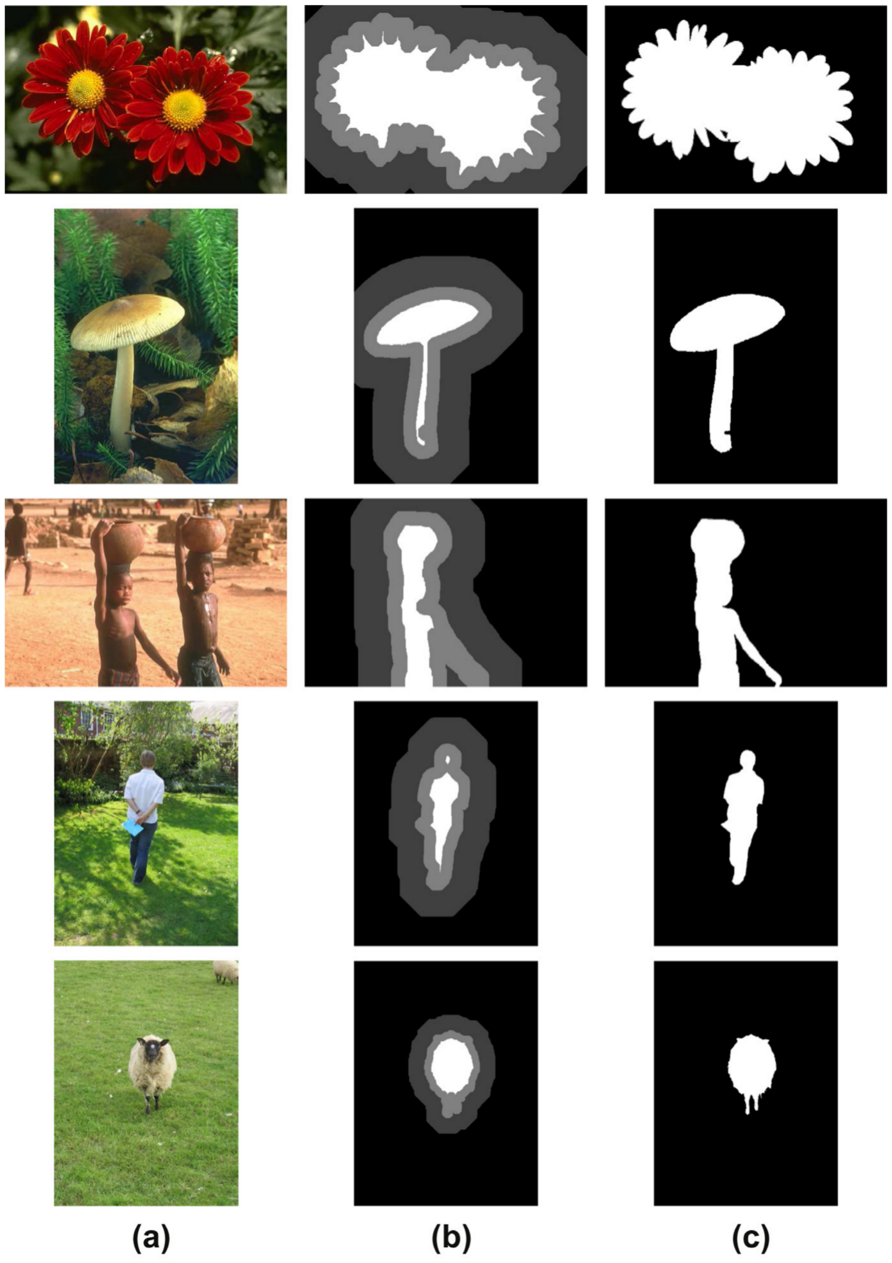

Fig. 1. (a) Original images from the GrabCut dataset, (b) the trimaps providing seed regions, and (c) the original ground truth.

On the contrary, graphs used in image representation (and particularly their matrix representations) do not allow vertices to be linked by multiple edges. For the same reasons, the results obtained with a DINH model are better than those obtained with a directed graph built from the DINH, showing the reliability of using a hypergraph formulation. However, the performance obtained from the directed graph are close to those from the INH case, which shows that using directional information and asymmetric weighting can improve segmentation results.

Table 1
Comparison of the average error rates obtained on the GrabCut dataset [32] between our iterative framework with different transition matrices exploiting different models of data representation. The value obtained with our proposed method is highlighted in bold.

Model	Average error rate (%)
Undirected graph [34]	6.92
Directed graph	6.57
Undirected hypergraph	6.53
Directed hypergraph (proposed)	**6.15**

Fig. 2 presents a comparison of qualitative results obtained with our proposed directed hypergraph approach, to those from a typical graph model, a directed graph model and an INH model. The results displayed are close-ups of the foregrounds obtained with the different algorithms corresponding to the images in the first three rows of Fig. 1. Despite the fact that all the segmentation results displayed in this figure can be considered as semantically meaningful, the computed error rates are different, in general due to the boundary placement. In the top row example, the boundaries around the flowers are more precise in the directed hypergraph case. In the middle row, the directed hypergraph is the only model that was able to correctly segment the mushroom foot, and particularly the leaf at the bottom. This result is also partially found in the directed graph case, which suggests that directional information is useful to segment thin elongated objects. The example of the last row is probably the one that shows the best the benefits of the introduction of directional information into the INH model. In fact, the arm of the boy is difficult to segment in an semi-supervised framework due to its color proximity with the background and the lack of seed pixels in this region. However, our approach exploiting the DINH model was the best at segmenting the arm, confirming that directional information can help at detecting elongated objects. The directed graph can also improve the arm segmentation, but we can see that it also shares segmentation errors with the undirected graph structure (the region below the armpit of the boy is not correctly segmented, and the face region is imprecise). This kind of difficulty can easily be tackled with the help of user interaction, by prompting the user to submit seed pixels at critical points. In the last example it will be relevant to mark some pixels of the boy's arm as object pixels in order to properly guide the segmentation.

These results clearly show that directed hypergraph models can provide a richer model of representation compared to simple undirected hypergraphs and graphs. If the error rate improvement between INH and DINH models can be considered as slight, this can be explained by the fact that the DINH is based on the INH, and further directed hypergraph models (not necessarily based on previously existing hypergraph representations) have to be investigated. In conclusion, our model based on a directed hypergraph outperforms the ones based on a different structure, and particularly the undirected hypergraph case, within our iterative learning setting. Consequently, we showed that the introduction of directed hypergraphs in computer vision and particularly in image representation is relevant, which was the main purpose and contribution of this paper.

6.2. Comparison with state-of-the-art

Secondly, we compared the performance of our approach with several state-of-the-art supervised segmentation algorithms. These results are compiled in Table 2. The first row refer to an algorithm based on a Gaussian Mixture Markov Random Fields (GM-MRF) interpretation on a graph [50]. The second row is an algorithm that exploits the hypergraph random walk matrix [8] based on a hypergraph representation of an image generated with superpixels [53]. The third one is not based on a global energy minimization method like the others, but on a local filtering of the label space [52]. The fourth row is our proposed approach. The fifth row is Grady's random walker algorithm [33] also exploiting a random walk interpretation of the learning step, but solved by a set of equations obtained from the Laplacian of the underlying graph. The algorithm on the sixth row also exploits the graph Laplacian, but with a statistical transductive interpretation [35]. It was also shown in [35] that this algorithm reduces to Grady's random walker if a given parameter is set to a particular value (which explains that the two reported error rates are the same). Finally, the seventh and last row refers to an algorithm again exploiting hypergraphs with a superpixels generation step, but in a much richer manner than in [18], since the given hypergraph uses real-valued incidence matrices. Here the learning step is handled by hypergraph interpolation involving a hypergraph Laplacian computation. All the error rates

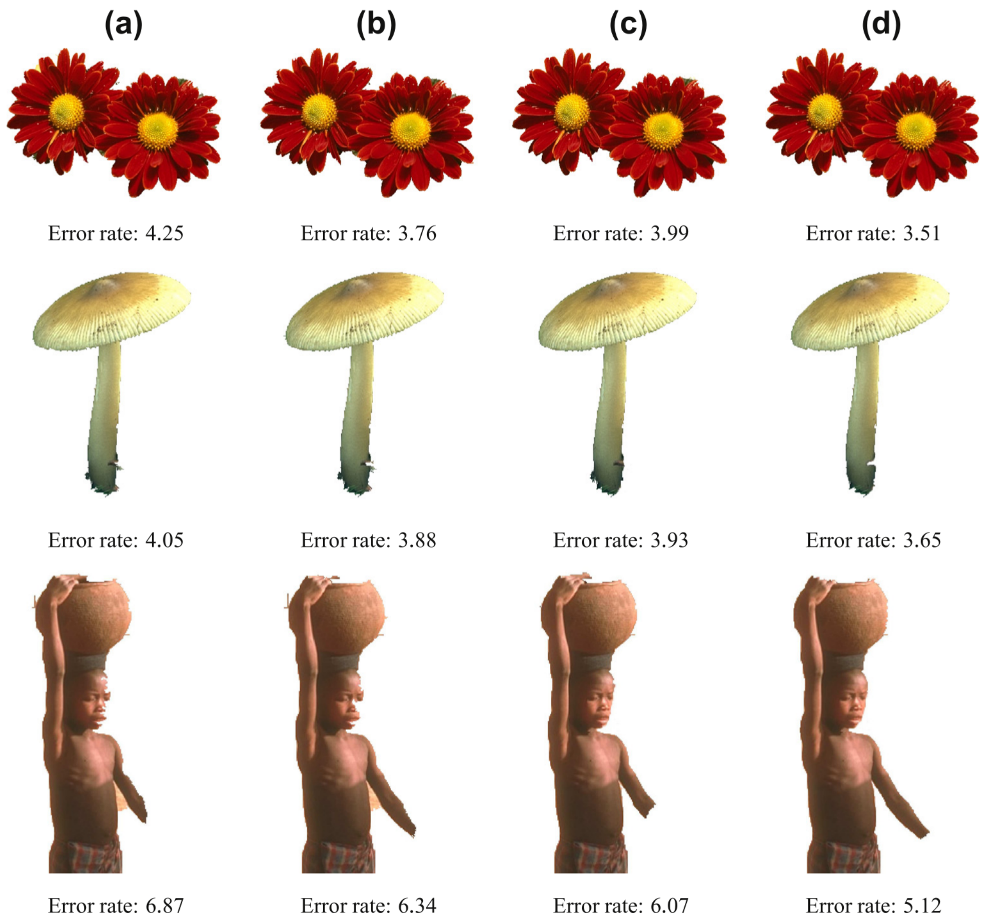

Fig. 2. Comparison of close-up foreground segmentation results from (a) an undirected graph model, (b) a directed graph model, (c) an INH model, and (d) our proposed directed hypergraph model. The error rates obtained for each result are also displayed. The original images, trimaps and ground truth are shown in Fig. 1 (first three rows).

found in Table 2 are reported on the GrabCut dataset using the original seeds under the same conditions, in [50] for the GM-MRF, in [18] for the superpixels hypergraph, in [52] for the cost volume filtering, in [35] for the Laplacian regularizer and Grady's random walker, and in [14] for the probabilistic hypergraph.

We can see from these results that the error rates obtained with our approach also compare well to other state-of-the-art image segmentation algorithms, showing the reliability of our global framework. The results obtained by the superpixels hypergraph [18] (despite using a hypergraph representation) can be explained by the fact that computing a superpixels representation of the image helps at reducing the size of the problem, but also introduces a hard labeling constraint between the pixels among the same superpixel. Consequently, the error produced by the segmentation algorithm for a single superpixel affects all the pixels that the superpixel represents, and then the error rate increases. If the method based on probabilistic hypergraph produced way better results, this can be explained by the fact that the hypergraph model used is built in a completely different way and exploits real-valued incidence matrices.

One should note that existing methods such as the Laplacian regularizer [35] and Grady's random walker [33] can reach more satisfactory results. The main reason for this is that these methods are based on a direct calculation of the labels of the unlabeled pixels (instead of approaching the solution with an iterative setting), generally using powerful tools based on matrix operations and equations solvers, thanks to the good properties displayed by the Laplacian matrix. This is also the case for the algorithm exploiting probabilistic hypergraphs. These methods can provide a solution that satisfies directly the optimization learning problem, which can explain the better results, but generally come along with a heavy computational burden [35]. These difficulties are not present in a label propagation approach. Furthermore, such a Laplacian representation has to be symmetric in order to allow a regularization or an eigen-decomposition. At that time, it does not exist a proper definition of such a Laplacian matrix for directed hypergraphs, which can be the subject of future research in this field. Recall that the primary objective of this work was to judge the reliability of using directed hypergraphs in the computer vision domain, and that we did not intend to design a segmentation algorithm outperforming the actual best methods in this field. Obviously, our method is not able to compete in terms of quality of segmentation with the ones that can directly find a global optimum to the underlying energy minimization problem. Nevertheless, the error rates of our algorithm are reasonably close to the better ones, which shows that our system is reliably efficient, and justifies further research work for improving its quality.

A visual comparison of results obtained with our approach and Grady's random walker [33] can be found in Fig. 3, which displays close-ups of the segmentation foregrounds for Grady's random walker and our label propagation approach (both with INH and DINH models) corresponding to the images in the two last rows in Fig. 1. The result from the top row clearly shows the advantage of a direct computation of the label memberships. The boundaries of the foreground object are correctly located in the case of Grady's random walker, as they are very imprecise with our approach. In fact, the label memberships are slow to converge in these particular regions where the complex texture in the background makes the label diffusion difficult. Consequently, the algorithm will be stopped because it will detect only a slight change in the label memberships between two iterations, even if their optimal value has not been reached. But the result in the bottom row shows the advantage of using our directed hypergraph representation, where the use of directional penalties allows to obtain a better segmentation of the right (in the image) leg of the sheep, whereas the two other approaches are unable to compute a proper segmentation due to the lack of seed pixels in this region.

In addition, our approach has been found to be reasonably efficient in terms of computation time, thanks to the iterative label propagation behavior we adopted. With our C++ implementation on an Intel Xeon 2.67 GHz machine, the algorithm takes between 0.85 and 5.35 seconds to converge, depending on the image size, with the GrabCut [32] dataset. The algorithm is slightly faster in the graph case, because by construction the number of neighbors to consider at each iteration for each pixel is higher in the hypergraph case. The computation times are way better than those reported for the Laplacian regularizer in [35] and the hypergraph interpolation in [14], which confirms the fact that powerful learning tools based on matrix operations or equations solving are computationally intensive. However, these methods can rely on many optimization solutions [54].

6.3. Multi-label segmentation

Our segmentation approach has also two practical advantages. First, it computes real-valued label memberships for every pixel

Table 2
Comparison of the average error rates obtained on the GrabCut dataset [32] between our proposed approach and different state-of-the-art algorithms. The value obtained by our proposed approach is highlighted in bold.

Algorithm	Average error rate (%)
GM-MRF [50]	7.9
Superpixels hypergraph [18]	7.3
Cost volume filtering [52]	6.2
Proposed (DINH)	**6.15**
Grady's random walker [33]	5.4
Regularized Laplacian [35]	5.4
Probabilistic hypergraph [14]	5.33

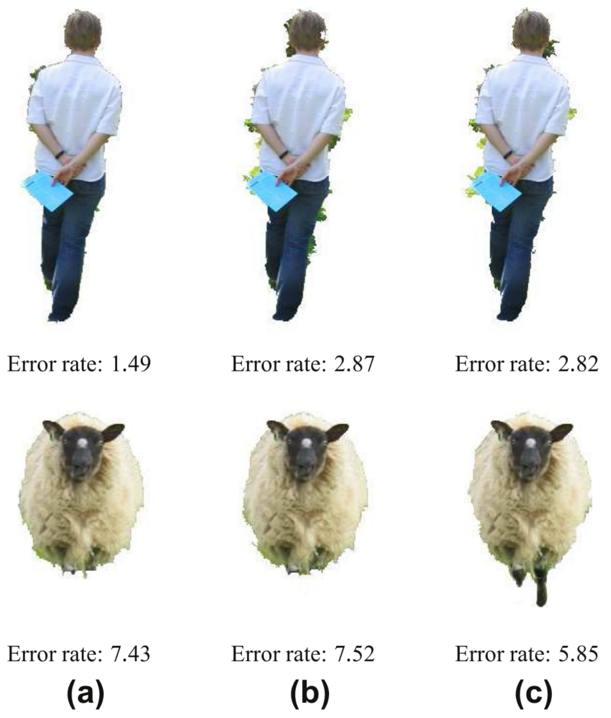

Fig. 3. Comparison of foreground segmentation results from (a) Grady's random walker algorithm [33], and our label propagation approach using (b) an undirected hypergraph (INH) model and (c) our proposed directed hypergraph (DINH) model. The error rates are also displayed. The original images, trimaps and ground truth are shown in Fig. 1 (two last rows).

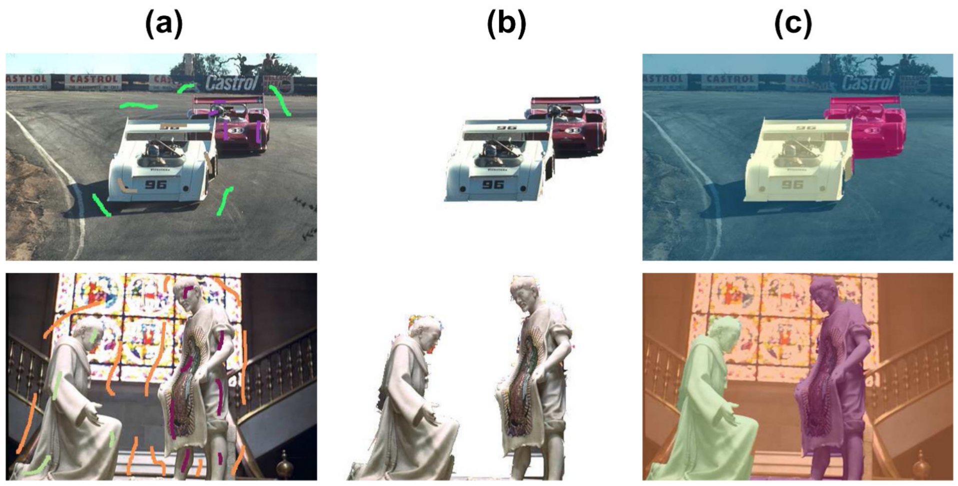

Fig. 4. Some examples of multi-label segmentation with our proposed approach using a DINH model. (a) Original image with user-supplied seeds. (b) Foreground region of the segmentation corresponding to the two object labels. (c) Original image with superimposed final labels.

(on the contrary of some algorithms like graph cuts [38] that only provides labels), that can serve as confidence scores or for alpha matting. And second, the mathematical formulation of the segmentation method and the development of its update rule (see Section 5) has been done by considering the possibility of having more than two labels (background/foreground), as some algorithms only consider the two-labels case. Consequently, our algorithm is directly applicable to multi-label segmentation.

Fig. 4 presents some examples of such multi-label segmentation obtained with our label propagation approach and our DINH model. One can see that with a small amount of user-supplied seeds, our algorithm is able to achieve perceptually acceptable results, even if some defects can be observed, especially when looking at the foreground region of the image at the bottom row, which are essentially due to the complex texture present in the background around the statues' boundaries.

7. Conclusions

In this paper, we investigated the utilization of directed hypergraphs in the computer vision domain. We proposed a generalization of undirected hypergraphs into directed hypergraphs, and defined a formulation of a random walk in a directed hypergraph. We proposed a DINH (Directed Image Neighborhood Hypergraph) model as a directed hypergraph representation of the image content, and adapted it to a semi-supervised image segmentation problem using a transition matrix computed from our random walk formulation. Experiments on the standard Microsoft GrabCut [32] datasets showed encouraging results in comparison with algorithms using undirected and directed graphs and undirected hypergraph-based image representation. In particular, results show that the introduction of directional information unto a hypergraph model can help at improving the segmentation results due to an asymmetric weighting of the hyperarcs. Consequently, we introduced the directed hypergraph concept in the computer vision domain and showed its relevance within this context.

Our image segmentation application is the first attempt of using directed hypergraphs in image analysis problems. If the results converge to an improvement of the performance compared to algorithms using traditional undirected hypergraphs, our proposition is an open system and several ways can be imagined to get more satisfying results. The comparison with several state-of-the-art supervised segmentation algorithms showed that our iterative approach cannot compete in terms of quality with methods that are able to directly compute an optimal solution to the energy minimization framework, using powerful graph-based tools such as Laplacian regularization [33,35]. The translation of such tools to the undirected hypergraph case is difficult at the moment, due to the asymmetric nature of directed hypergraphs and the lack of prior theoretical work in this field. Proposing coherent definitions of matrix-based representation of directed hypergraphs such as Laplacian matrices can be a direction to explore. Furthermore, different directed hypergraph models for image representation have to be investigated, and particularly for the case of color images. An interesting idea can be to model a video (or an image sequence) by the means of directed hypergraphs, since directional features such as the optical flow can easily be computed directly from the data.

Long-term further work in this direction will include the application of directed image hypergraph models in an unsupervised segmentation framework, for example by adapting existing hypergraph-based algorithms like spectral clustering [8] or reductive clustering methods [15] to the directed case.

References

[1] S.E. Schaeffer, Graph clustering, Computer Science Review 1 (1) (2007) 27–64, http://dx.doi.org/10.1016/j.cosrev.2007.05.001. ISSN: 1574-0137.
[2] A. Torsello, F. Escolano, L. Brun (Eds.), Graph-Based Representations in Pattern Recognition, 7th IAPR-TC-15 International Workshop, GbRPR 2009, Venice, Italy, May 26–28, 2009, Proceedings of Lecture Notes in Computer Science, vol. 5534, Springer, 2009. ISBN: 978-3-642-02123-7
[3] C. Berge, Hypergraphs, North Holland, Amsterdam, 1989.
[4] A. Bretto, Hypergraph Theory: An Introduction, Mathematical Engineering, Springer-Verlag, New York Incorporated, 2013. ISBN: 9783319000794.
[5] G. Karypis, R. Aggarwal, V. Kumar, S. Shekhar, Multilevel hypergraph partitioning: applications in VLSI domain, IEEE Transactions on Very Large Scale Integrated Systems 7 (1) (1999) 69–79. doi:http://dx.doi.org/10.1109/92.748202 ISSN 1063-8210.
[6] B.B. Cambazoglu, C. Aykanat, Hypergraph-partitioning-based remapping models for image-space-parallel direct volume rendering of unstructured grids, IEEE Transactions on Parallel and Distributed Systems 18 (1) (2007) 3–16, http://dx.doi.org/10.1109/TPDS.2007.253277. ISSN: 1045-9219.
[7] M. Koyuturk, C. Aykanat, Iterative-improvement-based declustering heuristics for multi-disk databases, Information Systems 30 (1) (2005) 47–70.
[8] D. Zhou, J. Huang, B. Scholkopf, Learning with hypergraphs: clustering, classification, and embedding., Advances in Neural Information Processing Systems, vol. 19, MIT Press, 1601.
[9] A. Bretto, J. Azema, H. Cherifi, B. Laget, Combinatorics and image processing, Graphical Models and Image Processing 59 (5) (1997) 265–277, http://dx.doi.org/10.1006/gmip.1997.0437. ISSN: 1077-3169.
[10] S. Rital, A. Bretto, D. Aboutajdine, H. Cherifi, Application of adaptive hypergraph model to impulsive noise detection, in: Computer Analysis of Images and Pattern, Lecture Notes in Computer Science, Springer, Berlin/Heidelberg, 2001, pp. 555–562.

[11] A. Bretto, L. Gillibert, Hypergraph-based image representation, in: L. Brun, M. Vento (Eds.), Graph-Based Representations in Pattern Recognition, 5th IAPR InternationalWorkshop, GbRPR 2005, Poitiers, France, April 11–13, 2005, Proceedings, vol. 3434 of Lecture Notes in Computer Science, Springer, 2005, pp. 1–11. ISBN: 3-540-25270-3.
[12] H. Bunke, P. Dickinson, M. Kraetzl, M. Neuhaus, M. Stettler, Matching of hypergraphs: algorithms, applications, and experiments, in: Applied Pattern Recognition, Studies in Computational Intelligence, vol. 91, 2008, pp. 131–154.
[13] Y. Huang, Q. Liu, D. Metaxas, Video object segmentation by hypergraph cut, in: IEEE Conference on Computer Vision and Pattern Recognition, 2009, pp. 1738–1745, http://dx.doi.org/10.1109/CVPR.2009.5206795. ISSN: 1063-6919.
[14] L. Ding, A. Yilmaz, Interactive image segmentation using probabilistic hypergraphs, Pattern Recognition 43 (5) (2010) 1863–1873, http://dx.doi.org/10.1016/j.patcog.2009.11.025. ISSN: 0031-3203.
[15] A. Ducournau, A. Bretto, S. Rital, B. Laget, A reductive approach to hypergraph clustering: an application to image segmentation, Pattern Recognition 45 (7) (2012) 2788–2803.
[16] A. Ducournau, S. Rital, A. Bretto, B. Laget, A multilevel spectral hypergraph partitioning approach for color image segmentation, in: IEEE International Conference on Signal and Image Processing Applications, Kuala Lumpur, Malaysia, 2009, pp. 419–424, http://dx.doi.org/10.1109/ICSIPA.2009.5478690.
[17] A. Bretto, A. Ducournau, S. Rital, A hypergraph reduction algorithm for joint segmentation and classification of satellite image content, in: Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications, vol. 6419 of Lecture Notes in Computer Science, 2010, pp. 38–45.
[18] L. Ding, A. Yilmaz, Image segmentation as learning on hypergraphs, in: Proceedings of the 2008 Seventh International Conference on Machine Learning and Applications, IEEE Computer Society, Washington, DC, USA, 2008, pp. 247–252, http://dx.doi.org/10.1109/ICMLA.2008.17. ISBN: 978-0-7695-3495-4.
[19] G. Ausiello, Directed hypergraphs: data structures and applications, in: M. Dauchet, M. Nivat (Eds.), CAAP '88, Lecture Notes in Computer Science, vol. 299, Springer, Berlin/Heidelberg, 1988, pp. 295–303, http://dx.doi.org/10.1007/BFb0026111. ISBN: 978-3-540-19021-9.
[20] G. Gallo, G. Longo, S. Pallottino, S. Nguyen, Directed hypergraphs and applications, Discrete Applied Mathematics 42 (1993) 177–201, http://dx.doi.org/10.1016/0166-218X(93)90045-P. ISSN: 0166-218X.
[21] D. Klein, C.D. Manning, New Developments in Parsing Technology, Chapter: Parsing and hypergraphs, Kluwer Academic Publishers, Norwell, MA, USA, 2005. ISBN: 1-4020-2293-X, 351-372.
[22] J. Oliveira, J. Jones-Oliveira, D. Dixon, C. Bailey, D. Gull, Hyperdigraph-theoretic analysis of the EGFR signaling network: initial steps leading to GTP: Ras complex formation, Journal of Computational Biology 11 (5) (2004) 812–842.
[23] A. Volpentesta, Hypernetworks in a directed hypergraph, European Journal of Operational Research 188 (2) (2008) 390–405.
[24] P. Kaiser, Y. Louët, A. El Sahili, An optimization algorithm for SDR multi-standard systems unsing directed hypergraphs, Frequenz – Journal of RF-Engineering and Telecommunications 66 (09 & 10) (2012) 251–260, http://dx.doi.org/10.1515/freq-2012-0047. ISSN: 2191-6349.
[25] X. Sun, Y. Lu, Directed-hypergraph based personalized E-learning process and resource optimization, in: Proceedings of the 2012 Fourth International Conference on Digital Home, ICDH '12, IEEE Computer Society, Washington, DC, USA, 2012, pp. 171–178, http://dx.doi.org/10.1109/ICDH.2012.89. ISBN: 978-0-7695-4899-9.
[26] A. Vasilevskiy, K. Siddiqi, Flux maximizing geometric flows, IEEE Transactions on Pattern Analysis and Machine Intelligence 24 (2002) 1565–1578. http://dx.doi.org/10.1109/TPAMI.2002.1114849 ISSN 0162-8828.
[27] Y. Boykov, G. Funka-Lea, Graph cuts and efficient N-D image segmentation, International Journal of Computer Vision 70 (2006) 109–131, http://dx.doi.org/10.1007/s11263-006-7934-5. ISSN: 0920-5691.
[28] D. Singaraju, L. Grady, R. Vidal, Interactive image segmentation via minimization of quadratic energies on directed graphs, in: CVPR, IEEE Computer Society, 2008.
[29] A. Hakeem, M. Shah, Learning, detection and representation of multi-agent events in videos, Artificial Intelligence 171 (8) (2007) 586–605.
[30] A. Narayanaswamy, A. Merouane, A. Peixoto, E. Ladi, P. Herzmark, U. Von Andrian, E. Robey, B. Roysam, Multi-temporal globally-optimal dense 3-D cell segmentation and tracking from multi-photon time-lapse movies of live tissue microenvironments, Spatio-temporal Image Analysis for Longitudinal and Time-Series Image Data (2012) 147–162.
[31] E.N. Mortensen, W.A. Barrett, Intelligent scissors for image composition, in: Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH '95, ACM, New York, NY, USA, 1995, pp. 191–198. doi:http://doi.acm.org/10.1145/218380.218442 ISBN 0-89791-701-4.
[32] C. Rother, V. Kolmogorov, A. Blake, GrabCut: interactive foreground extraction using iterated graph cuts, in: ACM SIGGRAPH 2004 Papers, SIGGRAPH '04, ACM, New York, NY, USA, 2004, pp. 309–314. doi:http://doi.acm.org/10.1145/1186562.1015720.
[33] L. Grady, Random walks for image segmentation, IEEE Transactions on Pattern Analysis and Machine Intelligence 28 (2006) 1768–1783, http://dx.doi.org/10.1109/TPAMI.2006.233. ISSN: 0162-8828.
[34] F. Wang, X. Wang, T. Li, Efficient label propagation for interactive image segmentation, in: Proceedings of the Sixth International Conference on Machine Learning and Applications, ICMLA '07, IEEE Computer Society, Washington, DC, USA, 2007, pp. 136–141. doi:http://dx.doi.org/10.1109/ICMLA.2007.43 ISBN 0-7695-3069-9.
[35] O. Duchenne, J.-Y. Audibert, R. Keriven, J. Ponce, F. Ségonne, Segmentation by transduction, in: IEEE Conference on Computer Vision and Pattern Recognition, IEEE, Anchorage, USA, 2008, pp. 1–8, http://dx.doi.org/10.1109/CVPR.2008.4587419.
[36] C. Couprie, L. Grady, L. Najman, H. Talbot, Power watershed: a unifying graph-based optimization framework, IEEE Transactions on Pattern Analysis and Machine Intelligence 33 (7) (2011) 1384–1399.
[37] X. Zhu, J. Lafferty, L. Cs, C.M.U. Edu, Semi-supervised learning using gaussian fields and harmonic functions, Machine Learning 20 (2) (2001) 912.
[38] Y. Boykov, M.-P. Jolly, Interactive graph cuts for optimal boundary and region segmentation of objects in N-D images, in: Proceedings of the Eighth IEEE International Conference on Computer Vision (ICCV), vol. 1, 2001, pp. 105–112, http://dx.doi.org/10.1109/ICCV.2001.937505.
[39] X. Bai, G. Sapiro, Geodesic matting: a framework for fast interactive image and video segmentation and matting, International Journal of Computer Vision 82 (2) (2009) 113–132, http://dx.doi.org/10.1007/s11263-008-0191-z. ISSN: 0920-5691.
[40] V. Caselles, R. Kimmel, G. Sapiro, Geodesic active contours, International Journal of Computer Vision 22 (1) (1997) 61–79, http://dx.doi.org/10.1023/A:1007979827043. ISSN: 0920-5691.
[41] J. Kemeny, J. Snell, Finite Markov chains, University series in undergraduate mathematics, VanNostrand, New York, repr edn., 1969.
[42] P. Doyle, J. Snell, Random walks and electric networks, Carus mathematical monographs, Mathematical Association of America, 1984. ISBN: 9780883850244.
[43] M. Szummer, T. Jaakkola, Partially labeled classification with Markov random walks, in: Advances in Neural Information Processing Systems, MIT Press, 2002, pp. 945–952.
[44] S. Kakutani, Markov processes and the Dirichlet problem, Proceedings of the Japan Academy 21 (1945) 227–233.
[45] C. Avin, Y. Lando, Z. Lotker, Simple random walks on radio networks (simple random walks on hyper-graphs), arXiv preprint arXiv:0907.1678.
[46] I. Gorodezky, I. Pak, Generalized loop-erased random walks and approximate reachability, Random Structures & Algorithms, http://dx.doi.org/10.1002/rsa.20460, ISSN: 1098-2418.
[47] A. Bretto, H. Cherifi, D. Aboutajdine, Hypergraph imaging: an overview, Pattern Recognition 35 (3) (2002) 651–658.
[48] X. Zhu, Z. Ghahramani, Learning from labeled and unlabeled data with label propagation, Tech. rep. CMU-CALD-02-107, School of Computer Science, Carnegie Mellon University, Pittsburgh PA, 2002.
[49] H. Minc, Nonnegative Matrices (Wiley-Interscience Series in Discrete Mathematics and Optimization), Wiley-Interscience, 1988. ISBN: 0471839663.
[50] A. Blake, C. Rother, M. Brown, P. Perez, P.H.S. Torr, Interactive image segmentation using an adaptive GMMRF model, in: T. Pajdla, J. Matas (Eds.), ECCV (1), Lecture Notes in Computer Science, vol. 3021, Springer, 2004, pp. 428–441. ISBN: 3-540-21984-6.
[51] A.Y. Yang, J. Wright, Y. Ma, S.S. Sastry, Unsupervised segmentation of natural images via lossy data compression, Computer Vision and Image Understanding 110 (2) (2008) 212–225, http://dx.doi.org/10.1016/j.cviu.2007.07.005. ISSN: 1077-3142.
[52] C. Rhemann, A. Hosni, M. Bleyer, C. Rother, M. Gelautz, Fast cost-volume filtering for visual correspondence and beyond, in: Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition, CVPR '11, IEEE Computer Society, Washington, DC, USA, 2011, pp. 3017–3024, http://dx.doi.org/10.1109/CVPR.2011.5995372. ISBN: 978-1-4577-0394-2.
[53] X. Ren, J. Malik, Learning a classification model for segmentation, in: Proceedings of the Ninth IEEE International Conference on Computer Vision, vol. 1, 2003, pp. 10–17. http://dx.doi.org/10.1109/ICCV.2003.1238308.
[54] L. Grady, A lattice-preserving multigrid method for solving the inhomogeneous Poisson equations used in image analysis, in: D.A. Forsyth, P.H.S. Torr, A. Zisserman (Eds.), ECCV (2), Lecture Notes in Computer Science, vol. 5303, Springer, 2008, pp. 252–264. ISBN: 978-3-540-88685-3.