Extending the Differential Image Foresting Transform to Root-based Path-cost Functions with Application to Superpixel Segmentation

Marcos A.T. Condori*, Fábio A.M. Cappabianco†, Alexandre X. Falcão‡ and Paulo A.V. Miranda*
*University of São Paulo, Institute of Mathematics and Statistics, São Paulo, SP, Brazil.
†Universidade Federal de São Paulo, Instituto de Ciência e Tecnologia, São José dos Campos, SP, Brazil.
‡University of Campinas, Institute of Computing, Campinas, SP, Brazil.
{mtejadac,pmiranda}@vision.ime.usp.br cappabianco@unifesp.br afalcao@ic.unicamp.br

***Abstract*—The Image Foresting Transform (IFT) is a general framework to develop image processing tools for a variety of tasks such as image segmentation, boundary tracking, morphological filters, pixel clustering, among others. The Differential Image Foresting Transform (DIFT) comes in handy for scenarios where multiple iterations of IFT over the same image with small modifications on the input parameters are expected, reducing the processing complexity from linear to sublinear with respect to the number of pixels. In this paper, we propose an enhanced variant of the DIFT algorithm that avoids inconsistencies, when the connectivity function is not monotonically incremental. Our algorithm works with the classical and non-classifical connectivity functions based on root position. Experiments were conducted on a superpixel task, showing a significant improvement to a state-of-the-art method.**

I. Introduction

The Image Foresting Transform (IFT) algorithm [1] is a framework to develop image processing operators that has been used in a number of successful applications including morphological filters [2], boundary tracking [3] [4], region segmentation [5] and classification [6]. The initial usage of the IFT algorithm was restricted to the so called *smooth connectivity functions* [1], since IFT with this class of connectivity functions was guaranteed to generate a forest (i.e., a graph with no cycles) of paths with optimum costs. Afterwards, *non-smooth connectivity functions* were proposed [7]. Even though the IFT with these functions does not always output an optimum-path forest, its resulting spanning forest still presents interesting properties from the perspective of the desired applications, such as edge persecution [8] and superpixel segmentation [9]. Indeed, in some cases, the optimal results have shown to be optimum according to other criteria, such as a cut measure in the image graph [5] [10] [11].

The proposed IFT algorithm runs in linear time with the number of pixels, when the priority queue is implemented by bucket sorting, or linearithmic time for a heap-based implementation. In applications that require user interaction, such as seed-based interactive image segmentation, the Differential Image Foresting Transform (DIFT) [12] algorithm allows to modify segmentation (i.e., reconstruct the optimum-path forest), by changing the seed set along iterations, without starting over the process. The DIFT algorithm uses partial results from previous iterations and makes changes only over the required regions of the image, presenting much higher performance than the IFT, with sublinear-time execution. However, the current DIFT algorithm can generate a series of anomalies, such as the presence of cycles in the output graph, when the connectivity function is not monotonically incremental[1].

This paper presents a novel differential algorithm, named *DIFTmod*, which maintains the same time complexity order of the original DIFT and handles a family of non-smooth connectivity functions, refered to as *root-based path-cost functions*. We also demonstrate the potential of DIFTmod in the context of superpixel segmentation, improving a state-of-the-art method in this area.

The paper is organized as follows: In Section II, we present the IFT and DIFT algorithms. Section III shows the anomalies of the original DIFT algorithm as associated with root-based path functions. The proposed algorithm, called DIFTmod, is presented in Section IV. In Section V, we show an application of the proposed modified DIFT for superpixel segmentation. We discuss the experiments and results in Section VI. Finally, we state the conclusions in Section VII.

II. Technical background

A. Image Foresting Transform (IFT)

An image can be interpreted as a weighted graph $G = \langle \mathcal{I}, \mathcal{A} \rangle$, whose nodes are the image pixels in its image domain $\mathcal{I} \subset \mathbb{Z}^2$, and whose arcs are the ordered pixel pairs $\langle s, t \rangle \in \mathcal{A}$ (e.g., 4-neighborhood or 8-neighborhood).

A path $\pi = \langle t_1, \ldots, t_n \rangle$ is a sequence of adjacent pixels (i.e., $\langle t_i, t_{i+1} \rangle \in \mathcal{A}$, $i = 1, 2, \ldots, n-1$) with no repeated vertices. The notation $\pi_t = \langle t_1 = r, \ldots, t_n = t \rangle$ indicates a path with terminus at a pixel t and the notation $\pi_{r \rightsquigarrow t}$ indicates a path from a pixel r (origin/root) to a pixel t (destination node). A path is also said *trivial*, when $\pi_t = \langle t \rangle$. A path $\pi_t = \pi_s \cdot \langle s, t \rangle$ indicates the extension of a path π_s by an arc $\langle s, t \rangle$. $\Pi(G)$ indicates the set of all paths in a graph G.

[1] A monotonically incremental path cost function is a particular case of smooth function [1].

A *label map* is a function $L : \mathcal{I} \to \{0, ..., l\}$ where $l - 1$ is the number of classes/clusters in the image.

A *predecessor map* is a function $P : \mathcal{I} \to \mathcal{I} \cup \{nil\}$ that assigns to each pixel t in $\mathcal{I}$ either some other adjacent pixel or a distinctive marker $nil \notin \mathcal{I}$ — in which case t is said to be a *root* of the map. A *spanning forest* is a predecessor map which contains no cycles. For any pixel $t \in \mathcal{I}$, a spanning forest P defines a path π_t^P recursively as $\langle t \rangle$ if $P(t) = nil$, and $\pi_s^P \cdot \langle s, t \rangle$ if $P(t) = s \neq nil$.

A *connectivity function* $f : \Pi(G) \to \mathcal{V}$ computes a value $f(\pi_t)$ for any path π_t, in some totally ordered set $\mathcal{V}$ of cost values. A path π_t is *optimum* if $f(\pi_t) \leq f(\tau_t)$ for any other path τ_t in G. For applicable connectivity function f, the *image foresting transform* (IFT) takes an image graph $G = \langle \mathcal{I}, \mathcal{A} \rangle$, and returns an *optimum-path forest* P — i.e., a spanning forest where all paths π_t^P for $t \in \mathcal{I}$ are optimum. The IFT is a generalization of Dijkstra's algorithm to *smooth* connectivity functions [1]. As shown by Frieze [13], the original proof of Dijkstras algorithm is easily generalized to monotonic-incremental (MI) connectivity functions, which conforms to $f(\pi_s \cdot \langle s, t \rangle) = f(\pi_s) \odot \langle s, t \rangle$, where $f(\langle t \rangle)$ is given by an arbitrary handicap cost, and $\odot : \mathcal{V} \times \mathcal{A} \to \mathcal{V}$ is a binary operation that satisfies the conditions: $x' \geq x \Rightarrow x' \odot \langle s, t \rangle \geq x \odot \langle s, t \rangle$ and $x \odot \langle s, t \rangle \geq x$, for any x, $x' \in \mathcal{V}$ and any $\langle s, t \rangle \in \mathcal{A}$.

Figure 1 presents the IFT algorithm. It computes a path-cost map V, which converges to V_{opt} for smooth functions [1]. It is also optimized for handling infinite costs, by storing in the priority queue $\mathcal{Q}$ only the nodes with finite-cost path, assuming that $V_{opt}(t) < +\infty$ for all $t \in \mathcal{I}$. In the case of non-monotonically incremental functions, the *state test* $t \notin \mathcal{K}$ (Line 10) is required to ensure that each pixel will be removed from Q (Line 7) just once during execution of the algorithm. In the case of MI functions, this state test is considered only a performance improvement [1] to avoid the computation of the path-cost in Line 11.

B. Differential Image Foresting Transform (DIFT)

In interactive segmentation based on markers, the user can add markers to and/or remove markers from previous interactions in order to improve the results. In this context, the DIFT enables the computation of a differential segmentation that aims at fixing wrongly segmented parts from previous iterations. For that purpose, the DIFT algorithm makes use of a *root map* as a function $R : \mathcal{I} \to \{r_0, \ldots, r_m\}$ that assigns some root marker r_i, $i \in \{0, 1, 2, ..., m\}$ to each pixel t in $\mathcal{I}$, such that $\pi_t^P = \langle r_i, \ldots, t \rangle$. At each iteration, the DIFT applies the following rules:

- Addition of markers: Suppose the insertion of a new marker r' during an iteration, such that $\pi_{r' \rightsquigarrow s}$ becomes an optimum path to a pixel s (possibly $s = r'$). If the new path through s extended by the arc $\langle s, t \rangle$ offers a lower cost $f(\pi_{r' \rightsquigarrow s} \cdot \langle s, t \rangle)$ to t than its current one in $V(t)$, then the path π_t^P will be updated as $\pi_{r' \rightsquigarrow s} \cdot \langle s, t \rangle$ and t will be inserted in Q. However, in the case that the offered cost $f(\pi_{r' \rightsquigarrow s} \cdot \langle s, t \rangle)$ is the same as its current value in $V(t)$ and the predecessor $P(t)$ equals s, the segmentation label $L(t)$ will not be updated and t will not be reinserted in Q as required. Hence, without the test of the predecessor $P(t) = s$ (Line 16 of DIFT algorithm in Figure 2), the path suffix of $\pi_{r' \rightsquigarrow t}$ will not be updated, which might cause an inconsistency in the label map $L(P(t)) \neq L(t)$. This inconsistency is described in details in Section II-B1.
- Removal of marker: In the case of a removed root r, the entire tree of the marker/seed r must be removed, creating a set of frontier pixels $F = \{t : \langle s, t \rangle \in \mathcal{A} \wedge r = R(s) \wedge R(t) \neq R(s)\}$. These frontier pixels will be inserted in the priority queue to start a new dispute for the conquest of the removed area. Figure 3 shows how the procedure is executed for marker removal.
- Simultaneous addition and removal of markers: In this case, the removal is executed first, defining the frontier pixels, the new markers are initialized, removing from F all new markers that eventually also belong to it.

Input: Image graph $G = \langle \mathcal{I}, \mathcal{A} \rangle$,and function f.
Output: Optimum-path forest P and the path-cost map V.
Auxiliary: Priority queue $\mathcal{Q}$, variable tmp, and set $\mathcal{K}$.

1. $\mathcal{K} \leftarrow \emptyset$
2. **for each** $t \in \mathcal{I}$ **do**
3. Set $P(t) \leftarrow nil$, $V(t) \leftarrow f(\langle t \rangle)$
4. **if** $V(t) \neq +\infty$ **then**
5. insert t in $\mathcal{Q}$
6. **while** $\mathcal{Q} \neq \emptyset$ **do**
7. Remove s from $\mathcal{Q}$ such that $V(s)$ is minimum.
8. Add s to $\mathcal{K}$.
9. **for each** pixel t such that $\langle s, t \rangle \in \mathcal{A}$ **do**
10. **if** $t \notin \mathcal{K}$ **then**
11. Compute $tmp \leftarrow f(\pi_s^P \cdot \langle s, t \rangle)$.
12. **if** $tmp < V(t)$ **then**
13. Set $P(t) \leftarrow s$, $V(t) \leftarrow tmp$.
14. Set $L(t) \leftarrow L(s)$.
15. **if** $t \notin \mathcal{Q}$ **then**
16. insert t in $\mathcal{Q}$.

Fig. 1. Original IFT Algorithm proposed in [1]

The original DIFT algorithm does not have the state test $t \notin \mathcal{K}$, because it was developed only for MI functions. Given that we are interested in functions that are not MI, we are presenting DIFT with the state test, referred to as DIFT* (Figure 2). However, this generates some inconsistencies:

1) Problems with the state test: DIFT* is able to process only the necessary pixels in each interaction. Still, under certain circumstances it generates inconsistencies between the map of labels and the map of predecessors. An inconsistent label map contains a pixel s such that $L(R(s)) \neq L(s)$, that is, the label of s is distinct from the label of its root.

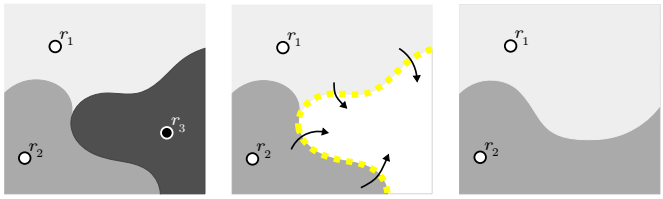
Figure 5 presents an example in which an inconsistency in the map of labels L happens after an execution of the DIFT*. It uses the cost function f_{max} in Equation 1, where $\delta(s, t)$ is the weight on arc $\langle s, t \rangle$ and $\mathcal{H}(s)$ a handicap value.

$$\begin{aligned} f_{max}(\langle s \rangle) &= \mathcal{H}(s) \\ f_{max}(\pi_s \cdot \langle s, t \rangle) &= \max\{f_{max}(\pi_s), \delta(s, t)\} \end{aligned} \quad (1)$$

Input: Image graph $G = \langle \mathcal{I}, \mathcal{A} \rangle$, maps P, R, L, V, labeling function λ, sets of new markers $\mathcal{S}'$ and remove pixels M_R.
Output: Maps $P,\ R,\ L$ and V.
Auxiliary: Priority queue $\mathcal{Q}$, sets frontier F and states $\mathcal{K}$.

1. $\mathcal{Q} \leftarrow \emptyset,\ \mathcal{K} \leftarrow \emptyset$
2. $V, P, F \leftarrow$ DIFT-TreeRemoval $(V, L, P, R, M_R, \mathcal{A})$
3. $F \leftarrow F \setminus S'$
4. **for each** $s \in S'$ **do**
5. **if** $f(\langle s \rangle) < V(s)$ **then**
6. $V(s) \leftarrow f(\langle s \rangle), L(s) \leftarrow \lambda(s), P(s) \leftarrow nil$
7. $R(s) \leftarrow s$ and insert s in $\mathcal{Q}$ whith cost $V(s)$
8. **for each** $s \in F$ **do**
9. Insert s into $\mathcal{Q}$ with cost $V(s)$
10. **while** $\mathcal{Q} \neq \emptyset$ **do**
11. Remove s from $\mathcal{Q}$ such that $V(s)$ is minimum.
12. Add s to $\mathcal{K}$
13. **for each** t such that $\langle s, t \rangle \in \mathcal{A}$ **do**
14. **if** $t \notin \mathcal{K}$ **then**
15. Compute $tmp \leftarrow f(\pi_s^P \cdot \langle s, t \rangle)$
16. **if** $tmp < V(t)$ **or** $P(t) = s$ **then**
17. $P(t) \leftarrow s,\ V(t) \leftarrow tmp$
18. $L(t) \leftarrow L(s),\ R(t) \leftarrow R(s)$
19. **if** $t \notin \mathcal{Q}$ **then**
20. Insert t in $\mathcal{Q}$

Fig. 2. DIFT* Algorithm



(a) (b) (c)

Fig. 3. Representation of the procedure adopted by DIFT for removal of a root. a) Result of initial interaction with the user with three root markers. b) Removal of the root r_3 and creation of the frontier by DIFT-TreeRemoval Procedure (see Figure 4). c) Result of path propagation from the frontier pixels. Note that only a subset of the graph has to be modified.

Given the initial image graph in Figure 5(a), an initial segmentation is executed producing the maps in Figure 5(b) which contains two markers in green and orange. In a second interaction showed in Figure 5(c), the orange marker and its subtree are removed and a blue marker is inserted. The frontier pixels of the removed subtree and the inserted marker are show in yellow. Figure 5(d) shows the moment in which an inconsistency occurs in the label map during the execution of the DIFT*. Pixel s leaves the queue Q before pixel t and change its state, because we employed FIFO tie-break police. Later, when pixel t leaves the queue, it can not propagate its label to s since s was already evaluated. Therefore t is the predecessor of s, but they have distinct labels in the end of DIFT* execution (see Figure 5(e)). Figure 5(f) shows the expected non-inconsistent result after running the IFT

Input: Maps $P,\ R,\ L,\ V$ and set M_R of remove markers.
Output: Maps $P,\ R,\ L,\ V$ and set F of frontier pixels.
Auxiliary: FIFO queue T and set M of roots of trees marked.

1. $F \leftarrow \emptyset,\ M \leftarrow \emptyset$
2. **while** $M_R \neq \emptyset$ **do**
3. Remove any s from M_R
4. $r \leftarrow R(s),\ M \leftarrow M \cup \{r\}$
5. **if** $V(r) \neq +\infty$ **then**
6. Insert r in T, $V(r) \leftarrow +\infty,\ P(r) \leftarrow nil$
7. **while** $T \neq \emptyset$ **do**
8. Remove s from T
9. **for each** t such that $\langle s, t \rangle \in \mathcal{A}$ **do**
10. **if** $P(t) = s$ **then**
11. $V(t) \leftarrow +\infty,\ P(t) \leftarrow nil$
12. $R(t) \leftarrow nil,\ L(t) \leftarrow nil$
13. **else**
14. **if** $R(t) \notin M$ **then**
15. $F \leftarrow F \cup \{t\}$

Fig. 4. DIFT-TreeRemoval Procedure

algorithm with the same two markers.

In [14] the following solution alternatives were analyzed (for functions monotonically increasing).

1) Set the priority policy of $\mathcal{Q}$ as LIFO.
2) Consider the addition and removal operation as two independent operations.
3) Do not use the state test in the DIFT* algorithm.

These alternatives do not work properly with DIFT* algorithm for more general functions, which is the objective of this work.

III. DIFT* FOR ROOT BASED PATH FUNCTIONS

In recent years IFT algorithm employed non-smooth functions in order to improve the results for several applications [7] [15]. With this kind of path function, the use of the state test is mandatory in order to avoid reprocessing pixels and the occurrence of infinite loops. In this section we will show how DIFT* algorithm fails to output consistent cost and label maps when executed with some kinds of non-smooth path functions. The inconsistency in the map of cost values happens when DIFT [12] or DIFT* generates a cost map that does not correspond to any cost map generated through IFT (considering different tie-breaking policies). We propose a novel solution to correct the presented issues that extends the DIFT* algorithm.

A. Problems

Firstly, we will study the behavior of some functions that depend on properties of the root propagating the paths. We will consider the function f_{euc} in Equation 2 and the function f_{abs_add} in Equation 3 which is a simplification of the function used in [9].

$$\begin{aligned} f_{euc}(\pi_t = \langle t \rangle) &= \begin{cases} 0 & \text{if } t \in \mathcal{S} \\ +\infty & \text{otherwise} \end{cases} \\ f_{euc}(\pi_{r \rightsquigarrow s} \cdot \langle s, t \rangle) &= \|t - r\|^2 \end{aligned} \tag{2}$$

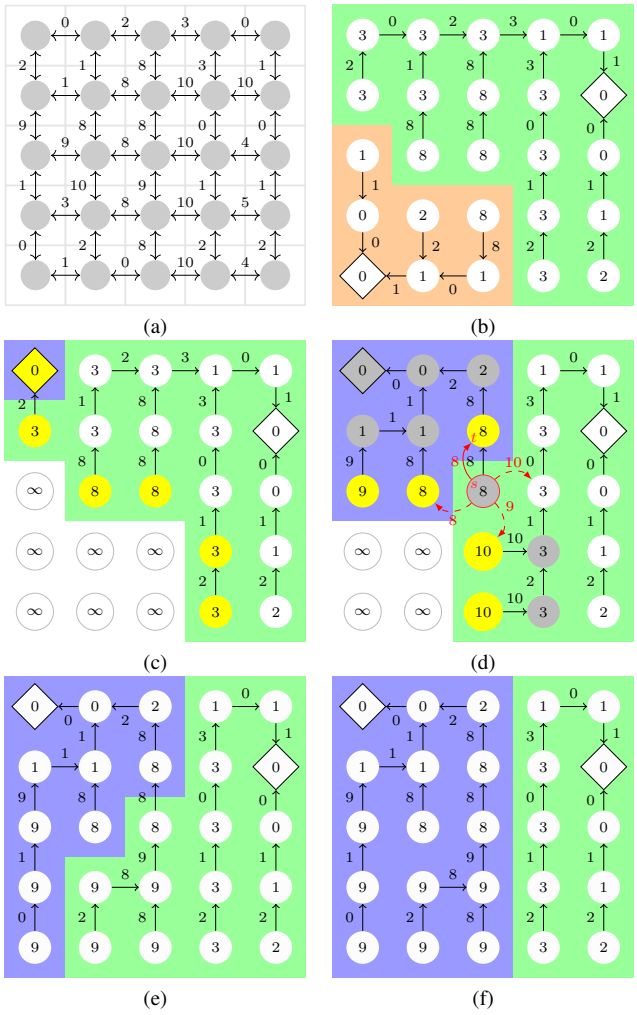


Fig. 5. Execution of DIFT* Algorithm, with roots in diamond shape. Yellow pixels are in the priority queue $\mathcal{Q}$ and gray pixels have already been evaluated. a) Graph with 4-neighborhood adjacency and weights in the edges. b) First interaction completed with f_{max} function, Equation 1. c) Addition of a new marker (upper corner) and frontier set wraps the removed orange zone. d) Some steps later the pixel s is evaluated, propagates a wrong label and changes its state, afterward, t predecessor of s, will be evaluated and will not modify s. e) Result obtained after two interactions through DIFT*, s and its predecessor t have different labels. f) Desired result.

$$\begin{aligned} f_{abs\_add}(\pi_t = \langle t \rangle) &= \begin{cases} 0 & \text{if } t \in \mathcal{S} \\ +\infty & \text{otherwise} \end{cases} \\ f_{abs\_add}(\pi_{r \rightsquigarrow s} \cdot \langle s, t \rangle) &= f_{abs\_add}(\pi_s) + |I(r) - I(t)| + 1 \end{aligned} \tag{3}$$

Equation 2 consists in the Euclidean distance path function. It may generate an inconsistency in the cost map from the second interaction of the DIFT*. Suppose the first iteration of the DIFT* with a single marker presented in Figure 6(b) over an input image graph of Figure 6(a). Now suppose the blue marker (r') is inserted at the second iteration of the DIFT* as showed in Figure 6(c). After a few steps, pixel s offers t a higher cost (i.e., 5) than it current one (i.e., 4), but by the *test of predecessor*, pixel t is conquered by the path $\pi_{r' \rightsquigarrow s,t}$, as showed in Figure 6(e). This leads to an inconsistent cost map since pixel t will have a higher cost compared to the result obtained by two executions of the IFT (compare Figures 6(e) and (f)).

Applying DIFT* algorithm with path function f_{abs_add} described in Equation 3 may output a graph with cycles, which violates the fact that the IFT algorithm should result in an spanning path forest. Suppose we execute the first iteration of DIFT* over the image graph in Figure 7(a). It will produce the forest in Figure 7(b). Then, the orange marker and its tree are removed, and a new green marker is inserted in the beginning of the second interaction, as showed in Figure 7(c). The yellow pixels belong to the frontier of the removed tree. After a few propagations, Figure 7(d) displays the propagation of pixel s offering t a higher cost than its current one. Still, by the predecessor test, pixel t is conquered by path $\pi_s \cdot \langle s, t \rangle$ resulting in the graph of Figure 7(e). Note that the pixel with red bound in Figure 7(e) leaves the queue before pixel t, conquering it again, and generating a cycle showed in Figure 7(f). Note that the connected component to which t belongs does not even have a root node.

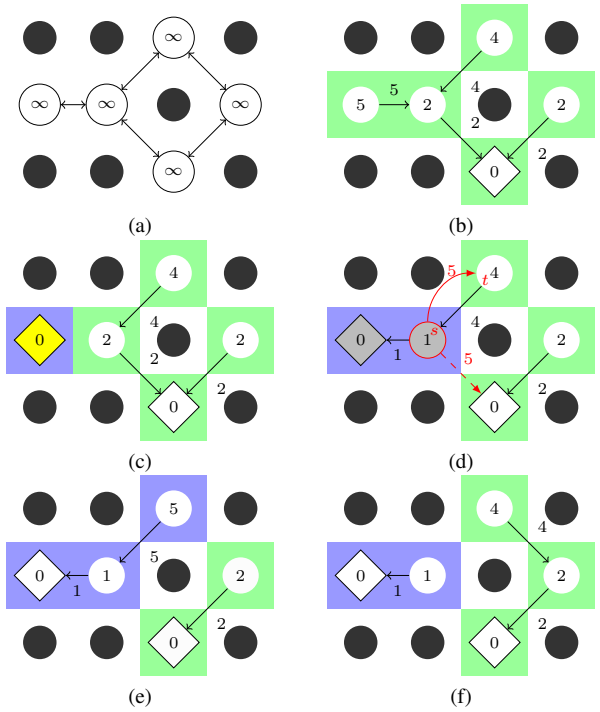


Fig. 6. DIFT* algorithm, roots in diamond shape, yellow pixels are in the priority queue $\mathcal{Q}$ and gray pixels have already been evaluated. a) Graph with 8-neighborhood adjacency and the initial cost map. b) Labels after first interaction completed with f_{euc} function. c) Addition of a new marker. d) Some steps later the pixel s offers t at a higher cost than $V(t)$ but by the test of predecessor t is conquered. e) Result obtained after two interactions through DIFT*. f) Desired result.

IV. Proposal DIFTmod Algorithm

We propose here a novel algorithm, DIFTmod (Figure 8), to reinforce the *predecessor test*. The objective is to continue using the *state test*, having a high performance solution and to be able to deal with the previously reported problems,

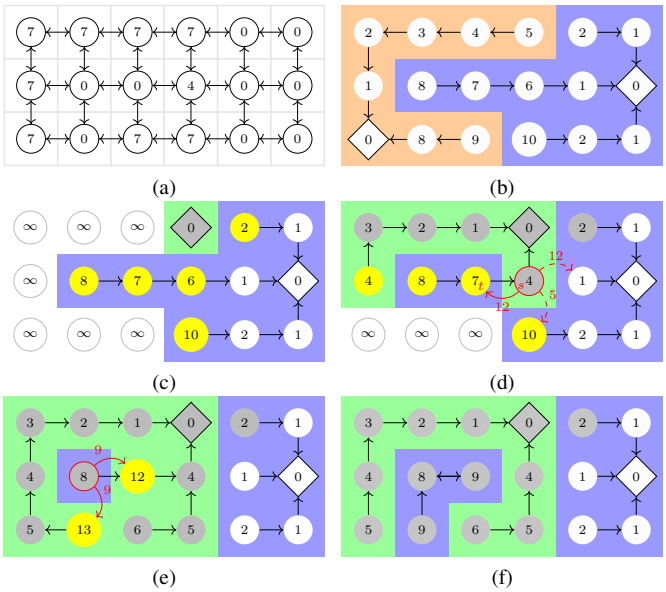


Fig. 7. Inconsistency of DIFT* algorithm with function f_{abs_add}, and roots in diamond shape. Gray pixels have already been evaluated. a) Graph with 4-neighborhood adjacency and intensities in the nodes. b) Labels after first interaction completed. c) Remove of orange tree and addition of marker (green label), yellow pixels belong to the frontier set. d) Some steps later, pixel s offers pixel t higher cost than its current one ($V(t)$) but by the test of predecessor t is conquered. Yellow pixels are in the priority queue $\mathcal{Q}$. e) Red circle around the node that leaves the queue before node t, generating a cycle in the graph. Yellow pixels are in the priority queue $\mathcal{Q}$. f) Incorrect resulting cycle graph obtained after two interactions of DIFT*.

including the issue reported in the Section II-B1 and also with non-smooth path functions presented in Section III.

Moya's alternatives [14], presented in Section III-A, do not support our objective. LIFO policy does not output an equitable segmentation. Performing the marker addition and removal in two separated operations increases the amount of processing, since nodes may be processed more than once. Finally, removing the *state test* is not an alternative again because of the reduced performance due to the multiple processing of nodes among other problems.

Therefore, we propose the following alternative: if the cost offered by path $\pi_{r' \rightsquigarrow s} \cdot \langle s,t \rangle$ to pixel t is equal to the current cost of t and both pixels have different labels $L(s) \neq L(t)$ and $P(t) = s$, t is placed as an extension of the path and t is inserted in the priority queue $\mathcal{Q}$ with a policy LIFO (only for this insertion, after this insertion the priority queue works with policy FIFO). This prevents path suffixes of path $\pi_{r' \rightsquigarrow t}$, having a higher priority with the same value of $V(t)$ in $\mathcal{Q}$ to be evaluated before the pixel t.

Now, to prevent problems related to functions which employ properties of roots, as explained in Section III, we included other conditions to the *predecessor test* of the DIFT* algorithm. The idea is to verify if the subtree of a pixel s, being evaluated, is in an inconsistent state. A subtree rooted in t is inconsistent if the cost offered from a path π_s extended to t is greater than the current cost of π_t and s is the predecessor of t.

Procedure RemoveSubTree in Figure 9, releases the entire subtree, converting its pixels to trivial trees of infinite cost, and transforms all neighboring pixels into frontier pixels, inserting them in $\mathcal{Q}$.

Input: Image graph $G = \langle \mathcal{I}, \mathcal{A} \rangle$, maps P, R, L, V, labeling function λ, sets of new markers $\mathcal{S}'$ and remove pixels M_R.
Output: Maps P, R, L and V.
Auxiliary: Priority queue $\mathcal{Q}$, sets frontier F and states $\mathcal{K}$.

1. $\mathcal{Q} \leftarrow \emptyset$, $\mathcal{K} \leftarrow \emptyset$
2. $V, P, F \leftarrow$ DIFT-TreeRemoval $(V, L, P, R, M_R, \mathcal{A})$
3. $F \leftarrow F \setminus S'$
4. **for each** $s \in S'$ **do**
5. **if** $f(\langle s \rangle) < V(s)$ **then**
6. $V(s) \leftarrow f(\langle s \rangle), L(s) \leftarrow \lambda(s), P(s) \leftarrow nil$
7. $R(t) \leftarrow R(s)$ and insert s in $\mathcal{Q}$ with cost $V(s)$
8. **for each** $s \in F$ **do**
9. Insert s into $\mathcal{Q}$ with cost $V(s)$
10. **while** $\mathcal{Q} \neq \emptyset$ **do**
11. Remove s from $\mathcal{Q}$ such that $V(s)$ is minimum.
12. Add s to $\mathcal{K}$
13. **for each** pixel t such that $\langle s,t \rangle \in \mathcal{A}$ **do**
14. **if** $t \notin \mathcal{K}$ **then**
15. Compute $tmp \leftarrow f(\pi_s^P \cdot \langle s,t \rangle)$
16. **if** $tmp < V(t)$ **then**
17. $P(t) \leftarrow s, V(t) \leftarrow tmp$
18. $L(t) \leftarrow L(s), R(t) \leftarrow R(s)$
19. **if** $t \notin \mathcal{Q}$ **then**
20. Insert t in $\mathcal{Q}$ with policy FIFO
21. **else**
22. **if** $P(t) = s$ **then**
23. **if** $tmp > V(t)$ **then**
24. DIFT*-RemoveSubTree$(V, L, P, R, \mathcal{Q}, \mathcal{K}, t)$
25. **else**
26. **if** $tmp = V(t)$ and $L(t) \neq L(s)$ **then**
27. $L(t) \leftarrow L(s), R(t) \leftarrow R(s)$
28. **if** $t \notin \mathcal{Q}$ **then**
29. Insert t in $\mathcal{Q}$ with policy LIFO

Fig. 8. Algorithm DIFTmod

V. Application to Generation of Superpixels

Superpixels are the result of an over-segmentation of the image and have been used in a wide variety of applications of computer vision, segmentation, object detection and 3d reconstruction. One of the most well-known superpixel algorithms is the Simple Linear Iterative Clustering (SLIC) [16], which divides the image using an adaptive k-means clustering.

The IFT-SLIC algorithm [9] considers the first initialization of clusters in a similar way as SLIC and adapts the superpixels by calculating the IFT with a connectivity function that depends on the root pixel of each superpixel according to Equation 4. The algorithm is repeated a few times (e.g. 10), updating the central position of each superpixel.

Input: Maps V, L, P, R, priority queue $\mathcal{Q}$, set $\mathcal{K}$, adjacency $\mathcal{A}$ and pixel u (root of the subtree to be removed)
Output: Maps V, L, P, R, priority queue $\mathcal{Q}$, set $\mathcal{K}$
Auxiliary: Sets H e H'.

1. $H \leftarrow \emptyset$, $H' \leftarrow \emptyset$
2. Insert u in H e H'
3. **while** $H \neq \emptyset$ **do**
4. Remove a from H and from $\mathcal{K}$
5. $L(a) \leftarrow nil, P(a) \leftarrow nil$
6. $R(a) \leftarrow nil, V(a) \leftarrow +\infty$
7. **if** $a \in \mathcal{Q}$ **then**
8. Remove a from $\mathcal{Q}$
9. **for each** pixel b such that $\langle a, b\rangle \in \mathcal{A}$ **do**
10. **if** $P(b) = a$ **then**
11. Insert b in H
12. **else**
13. Insert b in H'
14. **while** $H' \neq \emptyset$ **do**
15. Remove a from H'
16. **if** $V(a) \neq +\infty$ **then**
17. **if** $a \in \mathcal{Q}$ **then**
18. Update a in $\mathcal{Q}$ with cost $V(a)$
19. **else**
20. Insert a in $\mathcal{Q}$ with cost $V(a)$

Fig. 9. Procedure DIFT*-RemoveSubTree

$$f_D(\pi_t = \langle t \rangle) = \begin{cases} 0 & \text{if } t \in \mathcal{S} \\ +\infty & \text{otherwise} \end{cases}$$
$$f_D(\pi_{r \rightsquigarrow s} \cdot \langle s, t \rangle) = f_D(\pi_s) + (\|I(r) - I(t)\| \cdot \alpha)^{\beta} + d(s, t) \quad (4)$$

A. Differential mode of IFT-SLIC

It is possible to use the DIFT in order to reduce the computation effort at each iteration, but since the path function used by IFT-SLIC is an expansion of the function presented in Equation 3, inconsistencies may occur. With the modifications presented in Section IV though, it is possible to generate consistent results, capable of being reproduced by the execution of the IFT. A demonstration is available at the author's website[2].

VI. Experimental results

To compare the time reduction and the obtained results, we execute the algorithms for different values of the parameters. In the Equation 4 we used 20 samples for α in [0.01,0.2] and β equals to 12.

The number of superpixels can be exact if we distribute uniformly the number of initial superpixels but for the experiment we use as a parameter an approximate number k as it is realized in SLIC. In the experiment, we used the test set of 50 natural images of the public GrabCut dataset [17]. We execute the DIFT* algorithm, the DIFT* algorithm with the proposed modifications (DIFTmod) and the IFT algorithm executed 10 times (IFT 10x) to get an image with the labels of each superpixel.

In order to calculate the accuracy of the new algorithm, we compare the labeled images of the differential algorithms with the result obtained by the IFT 10x. Since the number of superpixels is the same in all results, we match the labels of the superpixels of the differential algorithms with those of the IFT 10x, because the labels of each superpixel may have shifted occasionally during iterations.

For example for the swimmer image in Figure 10, we calculate the images of labels for IFT x10 and DIFTmod algorithms (Figure 10(b)(c), shows the edges of each superpixel). The divergence of the number of non-matching labeled pixels is shown in Figure 10(d), note that the greatest divergence occurs at the borders of each superpixel.

That divergence occurs in areas with little information to define good borders (absence of contrast) or because they are areas of tie-breaking showed in Figure 10(c), that is why we give the tolerance of 1 pixel at the borders and consider the result of Figure 10(e). At the end of the process Figure 10(e) shows the largest differences between the result of the algorithm IFT x10 and the DIFTmod.

As a result of the experiments, we obtain a reduction of processing time. Figure 12(b) shows the time of the first execution of IFT common for the three algorithms (blue zone), the savings ratio of calculating the superpixels of differential mode without inconsistencies and highlighting the performance for the range of alpha values that produce better results $\alpha \in [0.04, 0.08]$ as explored in [9] and [18]. Figure 12 shows the processing time for different values of k. Figure 13 shows the visual results of the DIFT* algorithm and the DIFTmod algorithm.

The percentage of divergence observed in the experiment is shown in Figure 11. For the DIFT* algorithm, the divergence grows according to the increase in the values of the parameters k and α in addition to the probability of having inconsistencies in their result. On the other hand, for the DIFTmod algorithm the divergence remains almost constant, presenting low values ($< 0.05\%$) as compared to total number of pixels in the image.

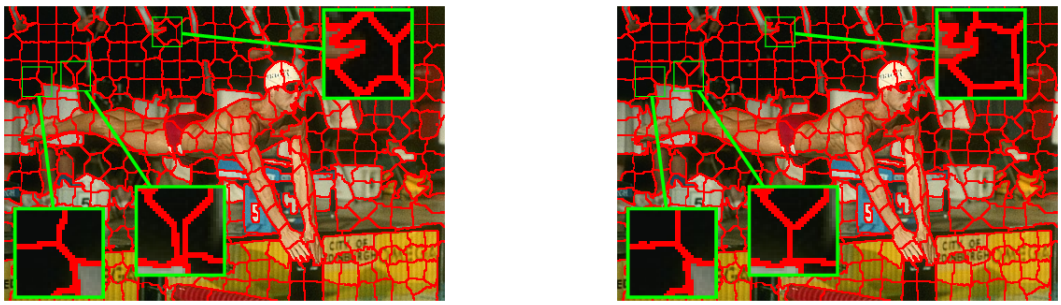
Clearly, the computation of superpixels through the IFT-SLIC algorithm using the proposed DIFTmod algorithm obtains a low computational cost and high precision results compared to IFT x10 algorithm.

VII. Conclusion

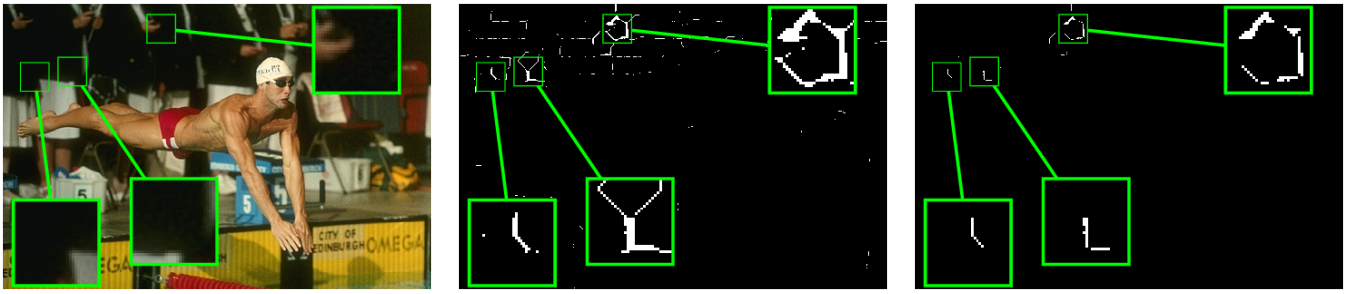
In this paper, we modified the original DIFT algorithm in order to avoid inconsistencies that may occur with classical functions, as well as extending it to other connectivity functions, that depend on the position of the root. The experiments show flawless results with running times very close to the original DIFT. They also show the importance of exploring other types of functions in the IFT framework.

As future works, we intend to explore the differential calculation of the Oriented Image Foresting Transform (OIFT) in 3D images with the DIFTmod algorithm, by adding the support for other types of not MI functions.

[2]http://www.vision.ime.usp.br/~mtejadac/diftmod.html



(a) (b)



(c) (d) (e)

Fig. 10. Graphical measure of accuracy of algorithms. a-b) Show the edges of the superpixels obtained by the algorithms IFT x10 and DIFT-mod respectively, with parameters $k = 450$ and $\alpha = 0.06$. The divergent areas are highlighted with zoom. c) Shows the original image and areas with little information to define exact edges. d) Divergent areas between results of IFT x10 and DIFTmod algorithms. Note that most of the differences occur in areas with low contrast. e) Divergent areas with 1-pixel tolerance (adjacency 4) at the edges of superpixels. White pixels represent notable differences in the algorithm results.

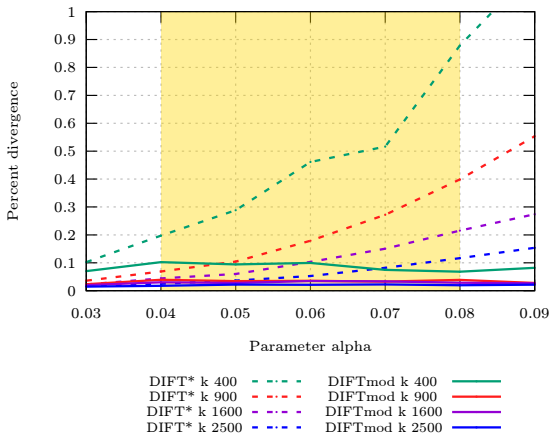


Fig. 11. Percentage of pixels divergence between results of differential algorithms (DIFT*, DIFTmod) and IFT x10.

Acknowledgment

The authors would like to thank CNPq (308985/2015-0, 486083/2013-6, 486988/2013-9, FINEP 1266/13), FAPESP (2011/50761-2), CAPES, NAP eScience - PRP - USP and grant 2016/21591-5, São Paulo Research Foundation (FAPESP) for funding. This research is also part of the FAPESP Thematic Research Project (proc. 2014/12236-1).

References

[1] A. X. Falcão, J. Stolfi, and R. de Alencar Lotufo, "The image foresting transform: Theory, algorithms, and applications," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 1, pp. 19–29, 2004.

[2] R. d. A. Lotufo, A. X. Falcão, and F. A. Zampirolli, "Ift-watershed from gray-scale marker," in *Computer Graphics and Image Processing, 2002. Proceedings. XV Brazilian Symposium on*. IEEE, 2002, pp. 146–152.

[3] A. X. Falcao, J. K. Udupa, and F. K. Miyazawa, "An ultra-fast user-steered image segmentation paradigm: live wire on the fly," *IEEE Transactions on Medical Imaging*, vol. 19, no. 1, pp. 55–62, Jan 2000.

[4] M. A. Condori, L. A. Mansilla, and P. A. Miranda, "Bandeirantes: A graph-based approach for curve tracing and boundary tracking," in *Mathematical Morphology and Its Applications to Signal and Image Processing*, May 2017, pp. 95–106.

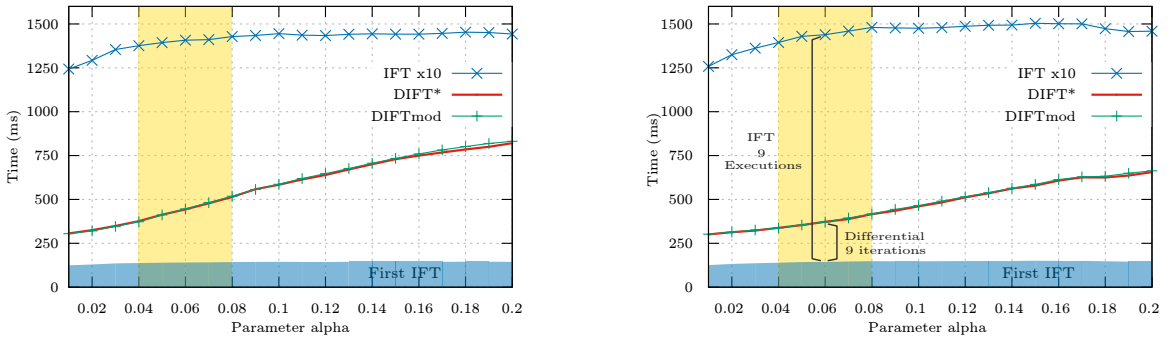
[5] P. A. Miranda and L. A. Mansilla, "Oriented image foresting transform segmentation by seed competition," *IEEE Transactions on Image Processing*, vol. 23, no. 1, pp. 389–398, 2014.

[6] F. Cappabianco, A. Falcão, C. Yasuda, and J. Udupa, "Brain tissue MR-image segmentation via optimum-path forest clustering," *Computer Vision and Image Underst.*, vol. 116, no. 10, pp. 1047–1059, 2012.

[7] L. A. Mansilla, P. A. V. Miranda, and F. A. Cappabianco, "Image segmentation by image foresting transform with non-smooth connectivity functions," in *2013 XXVI Conference on Graphics, Patterns and Images*. IEEE, 2013, pp. 147–154.

[8] P. A. V. Miranda, A. X. Falcao, and T. V. Spina, "Riverbed: A novel user-steered image segmentation method based on optimum boundary tracking," *IEEE Transactions on Image Processing*, vol. 21, no. 6, pp. 3042–3052, June 2012.

[9] E. B. Alexandre, A. S. Chowdhury, A. X. Falco, and P. A. V. Miranda, "IFT-SLIC: A general framework for superpixel generation based on simple linear iterative clustering and image foresting transform," in *2015 28th SIBGRAPI Conference on Graphics, Patterns and Images*, Aug 2015, pp. 337–344.



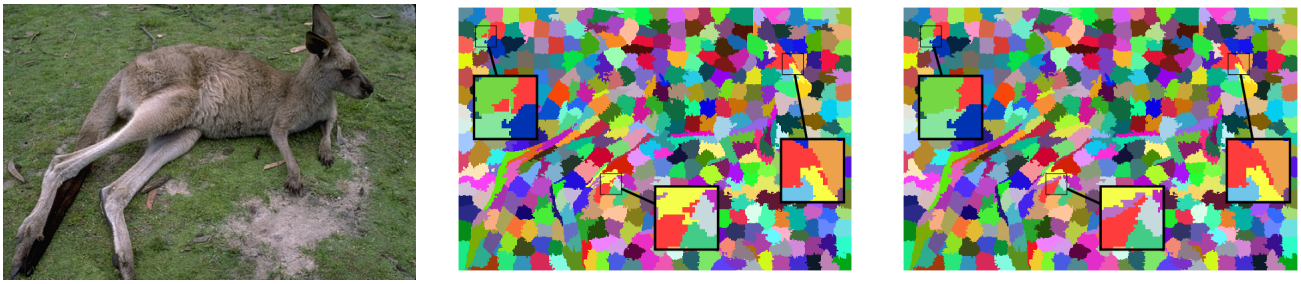
(a) k equals to 400

(b) k equals to 900

(c) k equals to 1600

(d) k equals to 2500

Fig. 12. Performance for the DIFTmod algorithm for differents values of k.



(a) (b) (c)

Fig. 13. Inconsistencies generated for DIFT* algorithm and result of DIFTmod algorithm for parameters $k = 500$ and $\alpha = 0.1$. a) Original image. b) Image with labels of DIFT* algorithm and inconsistencies (Section III-A), with focus on zoomed red pixels. c) Image with labels of DIFTmod algorithm. Inconsistencies observed in b) disappeared with the proposed modifications.

[10] L. A. C. Mansilla and P. A. V. Miranda, *Image Segmentation by Oriented Image Foresting Transform with Geodesic Star Convexity*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 572–579. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-40261-6_69

[11] L. A. C. Mansilla, M. P. Jackowski, and P. A. V. Miranda, "Image foresting transform with geodesic star convexity for interactive image segmentation," in *2013 IEEE International Conference on Image Processing*, Sept 2013, pp. 4054–4058.

[12] A. X. Falcão and F. P. Bergo, "Interactive volume segmentation with differential image foresting transforms," *IEEE Transactions on Medical Imaging*, vol. 23, no. 9, pp. 1100–1108, 2004.

[13] A. Frieze, "Minimum paths in directed graphs," *Operational Research Quarterly*, vol. 28, no. 2, pp. 339–346, 1977.

[14] N. Moya, "Interactive segmentation of multiple 3D objects in medical images by optimum cuts in graph," Master's thesis, Institute of Computing, University of Campinas, Brazil, May 2015.

[15] L. A. C. Mansilla, P. A. V. Miranda, and F. A. M. Cappabianco, "Oriented image foresting transform segmentation with connectivity constraints," in *2016 IEEE International Conference on Image Processing (ICIP)*, Sept 2016, pp. 2554–2558.

[16] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Ssstrunk, "SLIC superpixels compared to state-of-the-art superpixel methods," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 11, pp. 2274–2282, Nov 2012.

[17] C. Rother, V. Kolmogorov, and A. Blake, "Grabcut: Interactive foreground extraction using iterated graph cuts," *ACM Trans. Graph.*, vol. 23, no. 3, pp. 309–314, Aug. 2004.

[18] A. A. Tavares, P. A. Miranda, T. V. Spina, and A. X. Falcão, "A supervoxel-based solution to resume segmentation for interactive correction by differential image-foresting transforms," in *Mathematical Morphology and Its Applications to Signal and Image Processing*, May 2017, pp. 107–118.