Image Phylogeny Forests Reconstruction

Filipe de O. Costa, *Student Member, IEEE*, Marina A. Oikawa, Zanoni Dias,
Siome Goldenstein, *Senior Member, IEEE*, and Anderson de Rezende Rocha, *Member, IEEE*

***Abstract*— Today, a simple search for an image on the Web can return thousands of related images. Some results are exact copies, some are variants (or near-duplicates) of the same digital image, and others are unrelated. Although we can recognize some of these images as being semantically similar, it is not as straightforward to find which image is the original. It is not easy either to find the chain of transformations used to create each modified version. There are several approaches in the literature to identify near-duplicate images, as well as to reconstruct their relational structure. For the latter, a common representation uses the parent–child relationship, allowing us to visualize the evolution of modifications as a phylogeny tree. However, most of the approaches are restricted to the case of finding the tree of evolution of the near-duplicates, with few works dealing with sets of trees. Since one set of near-duplicates can contain *n* independent subsets, it is necessary to reconstruct not only one phylogeny tree, but several trees that will compose a phylogeny forest. In this paper, through the analysis of the state-of-the-art image phylogeny algorithms, we introduce a novel approach to deal with phylogeny forests, based on different combinations of these algorithms, aiming at improving their reconstruction accuracy. We analyze the effectiveness of each combination and evaluate our method with more than 40 000 testing cases, using quantitative metrics.**

***Index Terms*— Image forensics, multimedia phylogeny, optimal branching.**

I. Introduction

In RECENT years, sharing digital content became a trivial task due to the technological boom of hardware, software as well as the advent of social networks. Once one image or video is shared by a user, depending on its contents, it can easily *go viral*, being republished by many other users in different channels on the Web. Additionally, with the popularization of image editing software and online editor tools, in most of the cases, not only their exact duplicates will be available, but also manipulated versions of the original source. This scenario easily leads to copyright infringement, sharing of illegal contents and, in some cases, negatively affect or impersonate the public image of people or corporations.

When small changes are applied during the redistribution, usually without interfering in their semantic meaning, they are called *near-duplicate* objects. Several approaches for Near-Duplicate Detection and Recognition (NDDR) have been developed targeting at different applications, such as photography collections arrangement [1], [2], multimedia file matching [3], copyright infringement detection [4], [5] and forgery detection [6], [7]. However, more challenging than identifying different versions of a given document is to investigate the structure of modifications applied to generate these documents and the relationship among them.

In this context, a new research field, called *Multimedia Phylogeny* [8] has arisen, in which researchers are not only interested in determining if objects within a set are near-duplicate of each other, but also in finding their ancestral relationship and the original source (*root* or *patient zero*), reconstructing the order and the transformations that originally created the near-duplicate set. The main idea is inspired by the evolutionary process observed in Biology, in which an organism inherits characteristics from its ancestors, and can also go through mutations over time, producing new species [9]. If we look at the complete structure relating this population, moving from the root to the leaves of the phylogeny tree, we are able to identify the ancestral lineage, the descendants of each ancestor and have an overall idea of their evolution timeline.

A similar structure can be considered for a document that goes under slight modifications over time. If we are able to reconstruct the phylogeny tree of a set of near-duplicate objects, we can easily visualize their past history and ancestry information. In this paper, we will focus on algorithms dealing with the *Image Phylogeny* problem. In this case, we are interested in identifying the relationships among a set of Near-Duplicate Images (NDI), including transformations such as rotation, translation, cropping, brightness and contrast adjustment, among others. Those relationships are represented by an oriented tree, whose root is the least modified image and most probably the image first used to create others in the set, and the leaves, which represent the images with more modifications.

There are several forensic applications to image phylogeny solutions. For instance, consider postings on the Internet of

Manuscript received December 23, 2013; revised May 7, 2014; accepted July 4, 2014. Date of publication July 17, 2014; date of current version September 4, 2014. This work was supported in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior, Brasilia, Brazil, in part by the National Council for Scientific and Technological Development under Grant 307018/2010-5, Grant 304352/2012-8, Grant 306730/2012-0, Grant 477692/2012-5, Grant 477662/2013-7, and Grant 483370/2013-4, in part by the Fundação de Amparo à Pesquisa do Estado de São Paulo under Grant 2010/05647-4 and Grant 2013/08293-7, in part by Microsoft Research, and in part by the European Union through the Reverse Engineering of Audio-Visual Content Data Project, which was supported by the Future and Emerging Technologies Programme within the Seventh Framework Programme for Research of the European Commission, under Grant FET-268478. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Teddy Furon.

The authors are with the RECOD Laboratory, Institute of Computing, University of Campinas, Campinas 13083-970, Brazil (e-mail: fcosta@ic.unicamp.br; marina.oikawa@ic.unicamp.br; zanoni@ic.unicamp.br; siome@ic.unicamp.br; anderson.rocha@ic.unicamp.br).

This paper has supplementary downloadable material available at the IEEE Xplore website. The file consists of further explanations and examples regarding the methods developed in this paper. The material is 800KB in size. Furthermore, the datasets are registered on the address http://figshare.com/articles/Image_Phylogeny_Forests_Reconstruction/1012816, and the source code and documentation are available in a public repository on http://repo.recod.ic.unicamp.br/public/projects.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIFS.2014.2340017

images containing illegal or abusive content (e.g., a person in a bullying situation, fake and defamatory image of celebrities or politicians, etc.), which were redistributed after being modified by different users. With phylogeny algorithms, it becomes easier to understand the evolutionary process among the set of replicated images and trace their past history, allowing us to find the original image (or the least modified), which can also give hints about the creator of these images.

Researchers in this area mainly tried to solve this problem by considering only one tree for each set of near-duplicates [10]–[13]. However, in some cases, we may have multiple sets of NDIs, in which the source images come from different cameras or from the same camera but from a slightly different point in space and time. Therefore, we are not interested in reconstructing a single tree, but a *phylogeny forest*, a set of phylogeny trees, for representing their relationship.

Approaches for finding phylogeny forests are useful for tracing the original documents within a large set of semantically similar images. In other words, if there is more than one original image (the same scene, but taken from another view, for example), phylogeny forest approaches should be used for finding the correct trees of each image.

In this paper, we first make a brief review of the state-of-the-art approaches for reconstructing phylogeny trees, which follow two main strategies: using a modified version of the Kruskal's minimum spanning tree algorithm [14], or optimal branching strategies [13], such as the algorithms proposed independently by Chu & Liu [15], Edmonds [16] and Bock [17] to construct minimum spanning trees on directed graphs with known roots. Usually, each approach for solving the image phylogeny problem is analyzed separately. We extend existing approaches to automatically find forests and, based on the analysis of their properties and complementarity, we also introduce a solution to combine them toward a more stable, robust and effective algorithm. Finally, following the same methodology used in previous approaches, we evaluate the proposed methods using two different datasets comprising more than 40,000 testing cases.

II. Image Phylogeny

The analysis of the content of pairs of multimedia objects to investigate their relationship is a challenging task, since there are several operators that can be applied on one object to change its contents. However, these operators also leave behind useful artifacts that can be explored for reconstructing the evolutionary history of these multimedia objects.

A first definition and formalization of Image Phylogeny was introduced by Dias et al. [11], aiming at identifying the causal relationships among a set of NDIs and the transformations that led from one to the other. Given a set of images known to be near-duplicates, a dissimilarity function d is used to compare each pair of images, returning small values for similar images and large values for more distinct images.

Once the dissimilarity is calculated for all pairs of images, an asymmetric dissimilarity matrix M is constructed, which can be seen as a complete, directed graph with weights on each edge. To reconstruct the image phylogeny tree relating these images, the authors proposed to use the classic Kruskal's minimum spanning tree algorithm [14] adapted for oriented graphs, which was named Oriented Kruskal (OK) algorithm. As a result, images closer to the root of the tree represent the least modified versions, while the leaves represent the images with more significant transformations. Experiments performed using both synthetic images and images collected from the Internet provided good results, though, for large-scale scenarios, the calculation of the entire dissimilarity matrix became an expensive operation. In [18], the authors proposed a solution to this problem, by modifying the original reconstruction algorithm to allow missing values in the dissimilarity matrix and reduce the calculation time.

In the literature, there are also other approaches following a similar trend, aiming at finding the structure of the evolution of images on the Internet [10], [19]–[22]. Extensions to the original image phylogeny algorithm were also proposed for reconstructing the tree of evolution of a set of near-duplicate videos (Video Phylogeny) [12], as well for dealing with more than one phylogeny tree [8], [23]. In addition, the phylogeny of audio clips were also investigated by [24] using the original work proposed by Dias et al. [8].

In the next sections, we will give more details about the basic definitions used in phylogeny algorithms for reconstructing trees and forests, given a set of n near-duplicate images.

A. Image Phylogeny Tree (IPT)

Following the formalization described in [8], let $\mathcal{T}$ be a family of image transformations, and let T be a transformation such that $T \in \mathcal{T}$. The dissimilarity function d between two images $\mathcal{I}_A$ and $\mathcal{I}_B$ can be defined as the lowest value of $d_{\mathcal{I}_A,\mathcal{I}_B}$, such that

$$d(\mathcal{I}_A, \mathcal{I}_B) = \min_{T_{\vec{\beta}}} |\mathcal{I}_B - T_{\vec{\beta}}(\mathcal{I}_A)|_{\text{ point-wise comparison}}\, \mathcal{L}, \quad (1)$$

for all possible values of the parameter β in $\mathcal{T}$. Equation 1 calculates the dissimilarity between the best transformation mapping $\mathcal{I}_A$ onto $\mathcal{I}_B$, according to the family of transformations $\mathcal{T}$ and $\mathcal{I}_B$. Then, the comparison between the images can be performed by any point-wise comparison method $\mathcal{L}$ (e.g., sum of squared differences).

The dissimilarity of each pair of images is calculated by first finding interest points in each pair of images, using SURF [25], which will be used to estimate warping and cropping parameters robustly using RANSAC [26]. Pixel color normalization parameters are calculated using the color transfer technique proposed by Reinhard et al. [27]. Then, one of the images is compressed with the same compression parameters as the other. As a final step, both images are uncompressed and compared pixel-by-pixel according to the minimum squared error (MSE) metric. After calculating the dissimilarity for each pair of images, we have a dissimilarity matrix $M_{n\times n}$, where n is the number of near-duplicates and each region of the matrix represents the dissimilarity between one pair of images.

To reconstruct the IPT associated with the dissimilarity matrix, the authors first proposed the Oriented Kruskal algorithm. Considering a graph G that represents the dissimilarity matrix, the Oriented Kruskal algorithm starts considering each vertex (near-duplicate) of G as one root of the phylogeny tree.

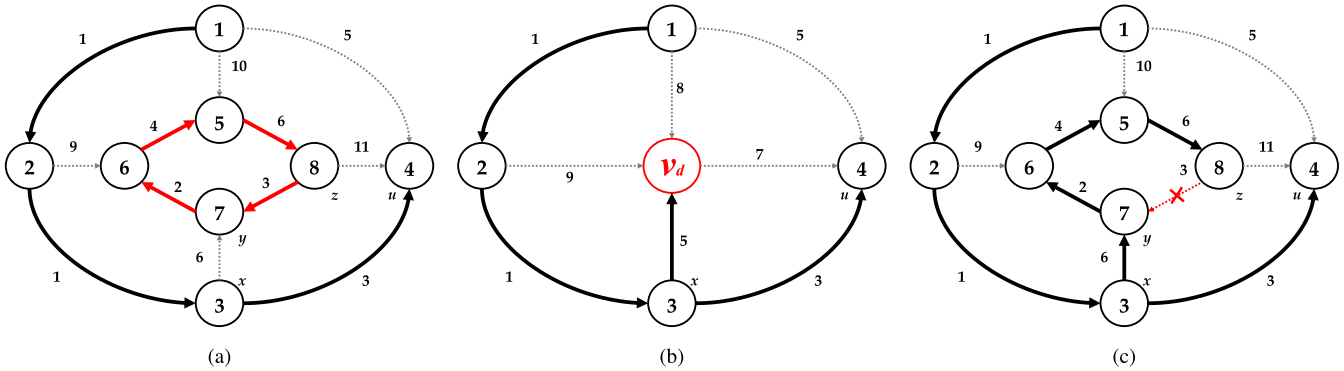


Fig. 1. Optimum branching algorithm simulation. (a) Starting from node 1, the edge with the lowest cost arriving in each node v is selected. Next, the algorithm needs to deal with the circuit in red. (b) A dummy node v_d is created, and the edge weights of the nodes connected to v_d are updated. (c) Finally, after processing all nodes within and outside the circuit, the algorithm recursively finds the optimal branching.

Then, the algorithm sorts all edges (i, j) of G and analyzes each position according to the sorted order, joining different trees and checking whether or not one edge can be added in the tree.

More recently, Dias et al. presented in [13] other approaches for IPT reconstruction, comparing their performance with the Oriented Kruskal algorithm. The first method is a variation of the algorithm for minimum spanning tree of Prim [28], adapted for oriented graphs. This Oriented Prim approach initially receives the dissimilarity matrix M and, for each possible root out of n, it reconstructs the tree rooted at the chosen node and calculates the cost for building such tree. After evaluating all possible roots, the final IPT is the one with the lowest reconstruction cost, which is calculated by summing the weight of the edges in the tree. However, this algorithm presented low effectiveness for the tree reconstruction, considering the ancestry relationship among the images.

The second approach presented in [13] is based on the Optimum Branching (OB) algorithm developed independently by Chu and Liu [15], Edmonds [16] and Bock [17]. Considering $G = (V, E)$ be the input graph and r one node of the phylogeny tree, the first step in this algorithm consists of assuming the node r as a possible root, as done in the Oriented Prim Algorithm. As an example, suppose that vertex 1 in Figure 1 was chosen as the initial root r. Then, for each node $v \in V \mid v \neq r$, the edge arriving at v with the lowest cost is selected. If there is no circuit in the found graph, the algorithm returns such branching. Otherwise, it deals with it by creating a dummy node v_d. For each edge connecting a node x outside the circuit and a node y in the circuit, the weight w is updated by taking into account $w(x, y)$ plus the lowest edge weight in the circuit minus the weight of the edge arriving at y in the circuit. In this example, $w(3, 7) = 6$, the lowest edge weight inside the cycle is $w(7, 6) = 2$, and the weight of the edge that arrives at the vertex $y = 7$ is $w(8, 7) = 3$. Thus, $w(3, v_d) = 6 + 2 - 3 = 5$. On the other hand, to update each edge connecting a node z inside the circuit and a node u outside the circuit, it is necessary to take into account $w(z, u)$ plus the lowest edge weight in the circuit minus the weight of the edge arriving at z inside the circuit. For instance, $w(v_d, 4) = w(8, 4) + w(7, 6) - w(5, 8) = 11 + 2 - 6 = 7$.

After processing all nodes within and outside the circuit, the algorithm recursively finds the optimal branching A_{i+1} in this graph, resulting in the updated graph depicted in Figure 1(b). Next, the algorithm needs to cope with the circuit itself. This is achieved by first discarding the edge in the current optimum branching connecting a node x to the dummy node v_d and updating such edge with the correct one in the original graph. In this case, the edge $(3, v_d)$ is replaced by edge $(3, 7)$ in the final branching. In addition, the algorithm updates all edges in the optimum branching within the circuit that do not arrive in y (in the example, the edge $(8, 7)$ is removed). Finally, the algorithm just needs to update possible edges from nodes within the circuit to outside that are part of the optimum branching. The algorithm ends with the graph in Figure 1(c) with solid lines highlighting the edges that belong to the optimum branching.

The algorithm's complexity is given by the recurrence $T(V, E) = T(V_0, E_0) + O(V, E)$, where $O(V, E)$ is the complexity of trying to generate one tree. As in each recursive call the number of vertices is reduced in, at least, one unit, $|V_0| < |V|$, we have that $T(V, E) = O(V(V + E))$. In our case, as $E = \Theta(V^2)$ (complete graph), then $T(V, E) = O(V^3)$. A faster implementation provided by Tarjan [29] also exists, which runs in $O(V^2)$, for dense graphs [13].

This new approach outperformed the Oriented Kruskal algorithm, being more effective for finding the roots and the ancestry relationships, either considering complete trees or with missing links in the phylogeny tree. This happens due to the different properties presented by each algorithm. The Oriented Kruskal algorithm is a greedy algorithm, that is, it makes a local optimal choice (the lowest cost edge) at each stage to find the minimum branching. On the other hand, the Optimum Branching algorithm is an exact algorithm, always finding one optimum global solution (it considers the whole dissimilarity matrix to make decisions about the phylogeny reconstruction), which leads to better results in most of the cases. More details about the optimum branching algorithm can be found in [13].

B. Image Phylogeny Forest (IPF)

In some cases in multimedia phylogeny, instead of one tree, we may find m trees representing the ancestry relationship in

a set of near-duplicate images. This happens when we have multiple images with the same semantic content, but that are not directly related to each other. For instance, images from the same scene taken with different cameras, or with the same camera but using different settings and in slight different positions. In this case, each tree in the forest represents the structure of transformations and the evolution of one subset of near-duplicate images, while the forest comprises distinct subsets of near-duplicate images which are semantically similar [23].

A first approach for reconstructing an IPF from a set of semantically similar images extended the idea of reconstructing an IPT from a set of near-duplicate images using the Oriented Kruskal algorithm. In this approach, Dias et al. [8] considered that to reconstruct m trees, the algorithm would stop when the structure of relationship reached $n - m$ edges where n is the number of images in the set. Although providing good results, the algorithm required from the user the number of trees to look for in the forest. However, this information is not always available, and without knowing the number of trees to reconstruct, the performance of the algorithm decreased with the number of trees, specially regarding the correct number of roots and the ancestry relationship in the IPF [23].

To enable automatic IPF reconstructions, Dias et al. presented in [23] a modified version of the Oriented Kruskal algorithm originally used to reconstruct IPTs. To create the new approach, named *Automatic Oriented Kruskal* (AOK), three parameters are required as input: the number of semantically similar images n, an $n \times n$ dissimilarity matrix M built upon these images and a parameter γ_{AOK}, calculated beforehand and defined as the number of standard deviations used to limit the number of edges to be included in the forest.

The AOK algorithm keeps track of the variance of processed edges and only adds a new one to the forest if the weight of the current edge is lower than γ_{AOK} times the standard deviation of the processed edges up to that point. This parameter γ_{AOK} is related to a threshold point τ_{AOK} that selects only edges that belong to valid trees. To define its value, a study about the behavior of the dissimilarity values of valid trees and forests was performed. It was observed that a Log-Normal distribution can reasonably describe the data regardless of the number of trees in the forest and the type of image capture (single/multiple cameras). The threshold $\tau_{\text{AOK}} = \mu_{\text{AOK}} + \gamma_{\text{AOK}} \times \sigma_{\text{AOK}}$ was used, where μ_{AOK} represents the average and σ_{AOK} the standard deviation of the weight of the edges already selected. After testing for different values, it was defined that $\gamma_{\text{AOK}} = 2$. In terms of complexity, and considering the same notation used for the analysis of OB algorithm, AOK algorithm has the same complexity of the original Kruskal algorithm: $O(V^2 \log V)$. Further details and a step-by-step algorithm can be found in [23].

III. Automatic Reconstruction of Image Phylogeny Forests

Without user intervention, the IPF reconstruction algorithm relies on the choice of a good threshold point to correctly decide the number of trees in the forest in advance. In this section, we propose new methods to automatically decide the number of trees in a forest, through the analysis of the dissimilarity matrix values.

Algorithm 1 Automatic Optimum Branching

Input: number of n semantically similar images, $n \times n$ dissimilarity matrix M and the number of standard deviations γ_{AOB} used as limit (default: $\gamma_{\text{AOB}} = 2$)
Output: reconstructed IPF $forest$

1: **for** $i \in [1..n]$ **do** ▷ Initialization
2: $forest[i] \leftarrow i$
3: **end for**
4: $n_{edges} \leftarrow 0$ ▷ Number of processed edges
5: $x_1 \leftarrow x_2 \leftarrow 0$ ▷ Variables to calculate σ dynamically
6: $G \leftarrow Graph(M)$
7: $B \leftarrow OB_{min}(G)$ ▷ Minimum branching
8: $B' \leftarrow$ **Sort** edges (i, j) of B by weight in non-decreasing order
9: **for** $(i, j) \in B'$ **do**
10: **if** $(n_{edges} > |B'|/2)$ **then**
11: $\sigma_{\text{AOB}} \leftarrow \sqrt{\frac{x_1 - \left(\frac{x_2^2}{n_{edges}}\right)}{n_{edges} - 1}}$
12: **if** $(w(i, j) - last > \gamma_{\text{AOB}} \times \sigma_{\text{AOB}})$ **then**
13: **break**;
14: **end if**
15: **end if**
16: $n_{edges} \leftarrow n_{edges} + 1$ ▷ Updates auxiliary variables
17: $last \leftarrow w(i, j)$ ▷ Last processed edge
18: $x_1 \leftarrow x_1 + w(i, j)^2$
19: $x_2 \leftarrow x_2 + w(i, j)$
20: $forest[j] \leftarrow i$ ▷ Adds new edge to the forest
21: **end for**
22: **return** $forest$ ▷ Returns the final forest

A. Automatic Optimum Branching (AOB)

Following a successful approach to automatically decide the number of trees in an IPF using the Oriented Kruskal algorithm, one might wonder what happens if we apply a similar process to the optimum branching algorithm explained in Section II-A. Thus, in this section, we extend upon this idea, proposing the Automatic Optimum Branching algorithm (AOB), with the necessary modifications to deal with its particularities.

Algorithm 1 describes our proposed implementation, which requires the following parameters: the number n of semantically similar images, an $n \times n$ dissimilarity matrix M built upon these images, and the parameter γ_{AOB}, which is the number of standard deviations used to limit the number of edges to be included in the forest.

Lines 1-3 initialize the vector $forest$ with n initial trees, each tree containing a vertex representing an image. In our implementation, we used the same tree representation introduced in [23], in which each position from $forest[i]$ denotes the parent of node i; $i = \{0..n - 1\}$. For instance, $forest[0] = 3$ means that edge $(3, 0)$ exists in the forest. Lines 4-6 initialize the auxiliary variables n_{edges}, which is used to keep track of the number of accepted edges, the variables x_1 and x_2 to calculate the standard deviation of accepted edges, and G to represent the graph constructed using the values from the dissimilarity matrix M. In Line 7, G is used as input for the Optimum Branching (OB) algorithm, returning its minimum branching in B. Subsequently, the edges of B are sorted into

non-decreasing order in Line 8 and used to decide the number of trees in this forest in the `for` loop in Lines 9-21.

In Line 10, the evaluation of the standard deviation only starts when the number of processed edges is greater than half of the edges of the minimum branching. This was done to avoid removing edges that are still valid to our approach, since with fewer values used for the calculation of the standard deviation, it is not possible to obtain a stable result.

In the next step, the algorithm updates the number of accepted edges and their standard deviation, includes the new edge to the $forest$ vector, and goes back to the beginning of the loop. Otherwise, the algorithm leaves the loop and the final result stored in vector $forest$ is returned in Line 22. In our approach, we use $\gamma_{\text{AOB}} = 2$ as the default value (for more details about how this value was obtained, please refer to Section V).

Algorithm 2 Extended Automatic Optimum Branching

Input: number of n semantically similar images, $n \times n$ dissimilarity matrix M and the vector $forest$ with the IPF reconstructed using AOB
Output: reconstructed forest $finalForest$

1: **for** $i \in [1..n]$ **do** ▷ Initialization
2: $finalForest[i] \leftarrow i$
3: **end for**
4: $G \leftarrow Graph(M)$
5: **for** $i \in [1..n]$ **do**
6: **for** $j \in [1..n]$ **do** ▷ Keeps valid connections
7: **if** $(Roots(forest, j) = Roots(forest, i))$ **then**
8: $G'(i, j) \leftarrow G(i, j)$
9: **end if**
10: **end for**
11: **end for**
12: $B'' \leftarrow OB_{min}(G')$ ▷ Execute OB again
13: **for** $(i, j) \in B''$ **do**
14: $finalForest[j] \leftarrow i$
15: **end for**
16: **return** $finalForest$ ▷ Returns the final forest

B. Extended Automatic Optimum Branching (E-AOB)

Up to this point, it is possible to obtain a first result for the IPF using AOB algorithm. However, after some experiments, we noticed that the IPF reconstruction could be further improved by also executing the OB algorithm on each tree belonging to this forest. One important thing to notice in this process is the fact that AOB algorithm considers all edges to construct the minimum branching. Once we remove some edges to build the forest, we create several partitions that are independent of each other. If these partitions are analyzed separately, the OB algorithm will choose edge connections that are optimal considering only the edges that belong to the current partition. This re-execution characterizes our Extended Automatic Optimum Branching (E-AOB) algorithm. It is also important to notice that, if we apply this re-execution using the Automatic Oriented Kruskal (AOK), due to its greedy heuristic, the result will not change, with AOK selecting the same edges to generate the forest.

The implementation of E-AOB in Algorithm 2 requires as input the dissimilarity matrix M representing the relationship among the images and the initial forest obtained with the execution of AOB algorithm. Similar to the AOB algorithm, Lines 1-3 initialize the vector $finalForest$ with n initial trees, each tree containing a vertex representing an image, and Line 4 initializes the variable G used to represent the graph constructed using the values from the dissimilarity matrix M. Lines 5-13 uses an auxiliary variable G' to store only the weights of the edges connected in the current forest, obtained from the vector $forest$ and representing subgraphs of graph G. In Line 14, AOB algorithm is executed again, returning the optimum branching for each of the trees in this forest. Lines 15-17 updates the variable $finalForest$ with this new forest, which is returned by the algorithm in Line 18.

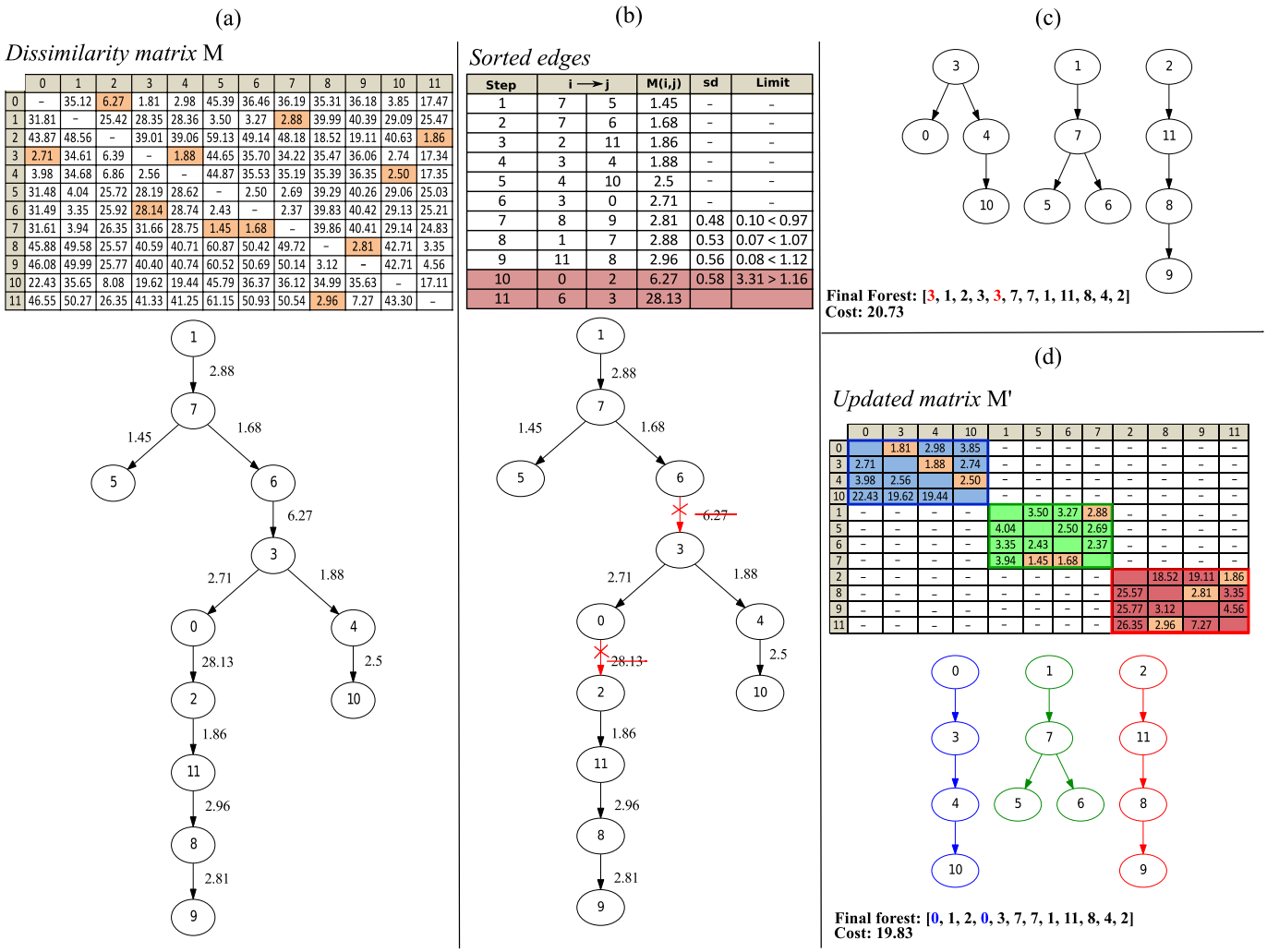
Figure 2 illustrates the execution of our proposed algorithm for a toy example with $n = 12$ semantically similar images related by the dissimilarity matrix M shown in Figure 2(a). After executing the OB algorithm once, we obtain the minimum branching tree B shown in Figure 2(a). We sort its edges from the lowest to the highest weight to decide which edges should be deleted to construct the forest. The table in Figure 2(b) shows the dynamic calculation of the standard deviation, which starts from Step 7 with edge $(8, 9)$, since $|B'|/2 = 12/2 = 6$. In this case, the current standard deviation of the edges is $\sigma_{\text{AOB}} \approx 0.48$. Since $w(8, 9) - w(3, 0) = 2.81 - 2.71 = 0.10$ and $0.10 < 2 \times 0.48 \approx 0.97$, then this edge can be accepted and the algorithm continues to the next edge. In Step 8, the edge $(1, 7)$ updates σ to 0.53 and since $w(1, 7) - w(8, 9) = 2.88 - 2.81 = 0.07$ and $0.07 < 2 \times 0.53 \approx 1.07$, the algorithm proceeds to the next edge until it reaches edge $(0, 2)$. For this edge, the standard deviation is $\sigma_{\text{AOB}} \approx 0.58$ and $w(0, 2) - w(11, 8) = 6.27 - 2.96 = 3.31$ and $3.31 > 2 \times 0.25 \approx 1.16$, which is above the allowed limit. Hence, this edge is discarded and the algorithm stops, returning the forest depicted in Figure 2(c).

Based on the forest reconstructed in this first part, we can apply the E-AOB method, updating the dissimilarity matrix to keep only the edges connected in the current forest and executing the OB algorithm again. The final result for the forest is depicted in Figure 2(d), where each tree is represented by the same color of its corresponding sub-matrix used for the reconstruction. With the execution of this additional step in AOB, it is possible to correct the position of nodes 0 and 3, whose parent-child relationship was reversed in the initial IPF reconstructed by AOB.

IV. Exploring Multiple Combinations

As mentioned in the previous sections, there are basically only one method in the literature currently being explored for the reconstruction of IPFs, the Automatic Oriented Kruskal (AOK). In addition, with the two extensions we introduced in Section III, we have AOB and E-AOB, which are extensions of the Optimum Branching algorithm proposed in [13].

Considering a dissimilarity matrix M, AOK algorithm follows a greedy heuristic, while AOB and E-AOB searches for the best global solution for the phylogeny reconstruction. However, all these algorithms assume a perfect dissimilarity calculation, which is not always true. Thus, in some cases, a greedy heuristic may present better results than a global solution. In this section, aiming at exploring these different



	0	1	2	3	4	5	6	7	8	9	10	11
0	–	35.12	6.27	1.81	2.98	45.39	36.46	36.19	35.31	36.18	3.85	17.47
1	31.81	–	25.42	28.35	28.36	3.50	3.27	2.88	39.99	40.39	29.09	25.47
2	43.87	48.56	–	39.01	39.06	59.13	49.14	48.18	18.52	19.11	40.63	1.86
3	2.71	34.61	6.39	–	1.88	44.65	35.70	34.22	35.47	36.06	2.74	17.34
4	3.98	34.68	6.86	2.56	–	44.87	35.53	35.19	35.39	36.35	2.50	17.35
5	31.48	4.04	25.72	28.19	28.62	–	2.50	2.69	39.29	40.26	29.06	25.03
6	31.49	3.35	25.92	28.14	28.74	2.43	–	2.37	39.83	40.42	29.13	25.21
7	31.61	3.94	26.35	31.66	28.75	1.45	1.68	–	39.86	40.41	29.14	24.83
8	45.88	49.58	25.57	40.59	40.71	60.87	50.42	49.72	–	2.81	42.71	3.35
9	46.08	49.99	25.77	40.40	40.74	60.52	50.69	50.14	3.12	–	42.71	4.56
10	22.43	35.65	8.08	19.62	19.44	45.79	36.37	36.12	34.99	35.63	–	17.11
11	46.55	50.27	26.35	41.33	41.25	61.15	50.93	50.54	2.96	7.27	43.30	–

Step	i → j		M(i,j)	sd	Limit
1	7	5	1.45	–	–
2	7	6	1.68	–	–
3	2	11	1.86	–	–
4	3	4	1.88	–	–
5	4	10	2.5	–	–
6	3	0	2.71	–	–
7	8	9	2.81	0.48	0.10 < 0.97
8	1	7	2.88	0.53	0.07 < 1.07
9	11	8	2.96	0.56	0.08 < 1.12
10	0	2	6.27	0.58	3.31 > 1.16
11	6	3	28.13		

	0	3	4	10	1	5	6	7	2	8	9	11
0		1.81	2.98	3.85	–	–	–	–	–	–	–	–
3	2.71		1.88	2.74	–	–	–	–	–	–	–	–
4	3.98	2.56		2.50	–	–	–	–	–	–	–	–
10	22.43	19.62	19.44		–	–	–	–	–	–	–	–
1	–	–	–	–		3.50	3.27	2.88	–	–	–	–
5	–	–	–	–	4.04		2.50	2.69	–	–	–	–
6	–	–	–	–	3.35	2.43		2.37	–	–	–	–
7	–	–	–	–	3.94	1.45	1.68		–	–	–	–
2	–	–	–	–	–	–	–	–		18.52	19.11	1.86
8	–	–	–	–	–	–	–	–	25.57		2.81	3.35
9	–	–	–	–	–	–	–	–	25.77	3.12		4.56
11	–	–	–	–	–	–	–	–	26.35	2.96	7.27	

Fig. 2. Simulation of AOB and E-AOB algorithm to reconstruct an IPF with three trees and 12 semantically similar images, from a 12 × 12 dissimilarity matrix. (a) Optimum Branching calculation. (b) Heaviest edges elimination. (c) IPF with AOB. (d) IPF with Extended AOB.

properties and their complementarity, we propose a combination among the results given by each approach, in such a way that errors introduced by one method can be fixed by other method(s).

A. Fusion Methodology for IPFs Reconstruction

Given n semantically similar images and the reconstructed IPFs of two or more methods to be combined, our approach is based on the idea of finding their most common elements to construct the final IPF. In the first part of our implementation, we calculate (a) the number of roots returned by each forest, (b) the number of times each node appears as a root, and (c) the number of times each edge connecting two nodes in the forest appears. In the last part, we implement the fusion scheme, constructing a new forest whose roots and edges of the trees are the ones with the highest number of votes calculated in the previous step.

However, it is known that the calculation of the dissimilarity among the images is not an exact estimation, given that one of the steps include matching of images, which is not perfect. In our implementation, we take advantage of this flaw in the dissimilarity calculation, and apply different amounts of perturbations through noise addition to the dissimilarity matrix M relating a set of images, generating 100 different variations of M and using them to reconstruct the IPFs. The number of variations was chosen in such a way that it is enough to analyze the consistency among the results and to achieve statistical significance in the analysis.

The noise was created by first calculating the standard deviation σ_M of all values in the dissimilarity matrix M relating each set of images. Then, for each value $m_{ij} \in M$, a different value between $[-k \times \sigma_M, +k \times \sigma_M]$ was randomly chosen using uniform distribution and added to the corresponding m_{ij} entry. In our implementation, $k = 1.5\%$ (for more details about this value, please refer to Section V-A).

Once the number of roots and edges for all forests have been calculated, we calculate the final number of roots r by choosing the median of the number of votes received by all methods in each of the 100 executions.[1] Then, to decide which

[1]There are several alternative strategies to calculate r, such as the average and the most frequent values. In our experiments, the median presented better results during training, but there is still room for further exploration.

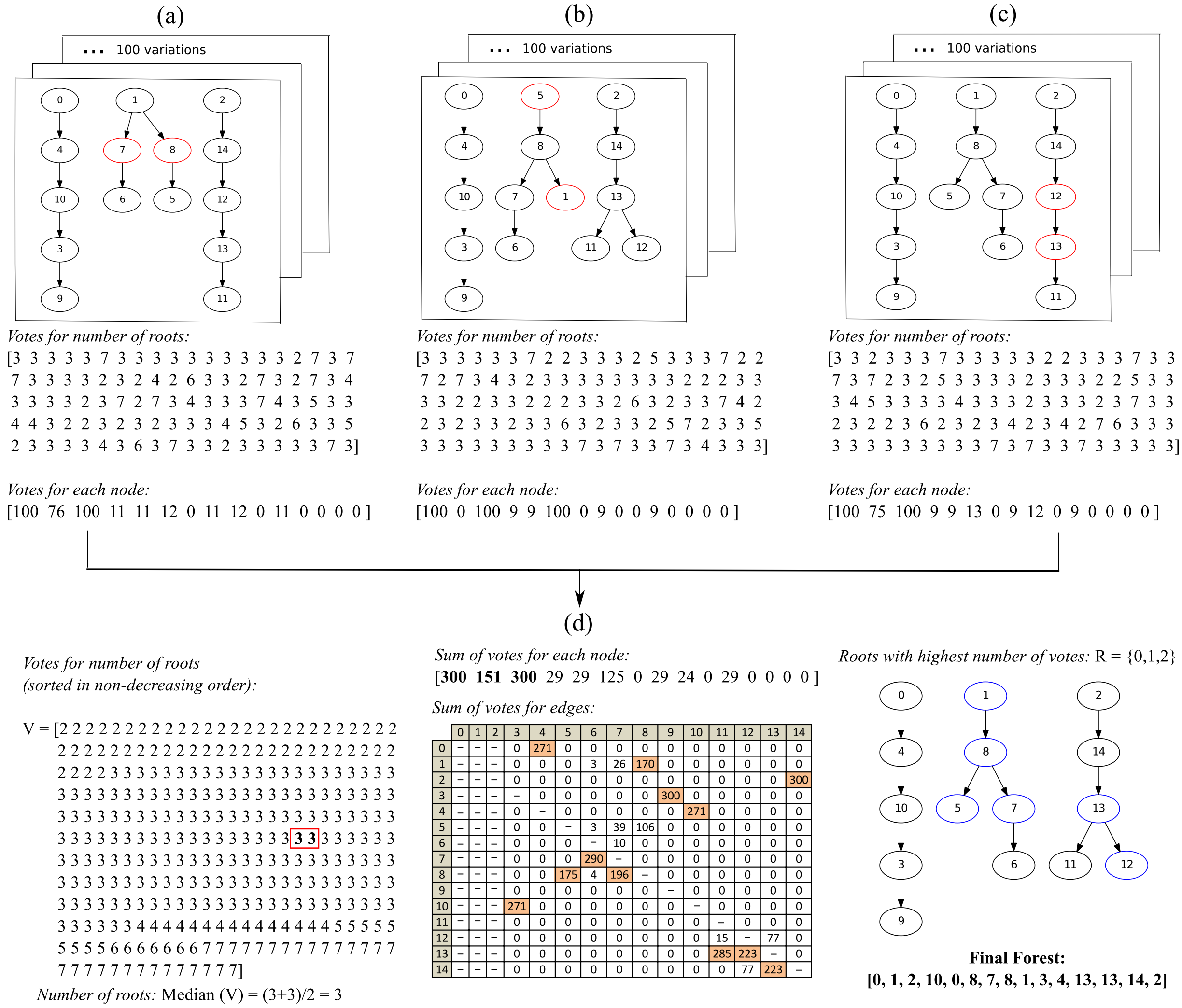


Fig. 3. Reconstructed IPFs for algorithms (a) AOK, (b) AOB, and (c) E-AOB methods used separately. Highlighted in red, the nodes in the wrong position with respect to the ground-truth forest, which can happen in different positions within the 100 variations. In (c) the final forest after fusion of IPF reconstruction algorithms, with the corrected nodes highlighted in blue.

nodes are the roots, we select the r nodes having the highest number of votes. In case there is a tie among the votes, we randomly choose one or some of them, depending on how many roots are yet to be decided. For the edges selection, we sum up the number of times each edge connecting two nodes in each forest appears, constructing a matrix of votes for edges. Once the roots are chosen, we fix these roots and give the matrix of votes for edges as input to a maximum branching algorithm, resulting in the sought combined forest.

For instance, consider the example depicted in Figure 3 with $n = 15$ semantically similar images. The first set shown in Figure 3(a) represents all 100 possibilities of reconstructed IPFs using AOK, with the first IPF illustrating one out of 100 possible reconstructions. The nodes highlighted in red denote a node in the wrong position when compared to the ground-truth forest. Similarly, Figure 3(b) and (c) represent the IPFs using AOB and E-AOB, respectively. Each set of IPFs is followed by the votes for the number of roots and the number of votes each root received.

Let V be a vector storing the number of roots returned by each method for each variation of the IPF. to calculate the number of roots the final IPF should have, we choose the median of V, that is, $Median(V) = 3$. To decide which nodes are the roots, we select the nodes with the highest number of votes, which in this case are Roots $= \{0, 1, 2\}$.

To count the votes for edges, we go through all edges (except for the ones pointing to the roots), increasing their score in a matrix of votes every time the edge (i, j) appears, and summing up the votes for the edges in all forests being combined. Figure 3(d) shows the sum of votes for roots and edges obtained with the combination of AOK $\times$ AOB $\times$ E-AOB, and the result after the fusion. The nodes highlighted in blue in the final forest represent the nodes that were previously in the wrong position (highlighted in red in the forest returned by the first execution of each of the methods), and were corrected after applying our fusion algorithm. Since the forest obtained after the fusion is the same forest than the one used to create this example, it is possible to see the fusion algorithm was able to correct the roots as well as the position of nodes $\{1, 5, 7, 8, 12, 13\}$.

V. Evaluation and Results

For evaluating the reconstructed IPFs, we consider the same quantitative metrics (roots, edges, leaves and ancestry) introduced by Dias et al. [23], considering scenarios where the ground truth is available. The following equation is used:

$$\mathrm{EM}(\mathrm{IPF}_1, \mathrm{IPF}_2) = \frac{S_1 \cap S_2}{S_1 \cup S_2} \tag{2}$$

where EM represents the evaluation metric, IPF_1 is the reconstructed forest with elements represented by S_1, and IPF_2 is the forest ground truth with elements S_2. This equation calculates the intersection of the result returned by IPF_1 in metric EM with respect to the reference forest IPF_2, and normalizes it by the union of both sets. For instance, in the example of Figure 2, the IPF reconstructed by E-AOB is the same of the Ground-Truth Forest (GTF) used to create the example. Therefore, the root metric yields Root (E-AOB, GTF) $= 3/3 = 100\%$. On the other hand, in comparison to the result returned by baseline AOB, we obtain Root (AOB, GTF) $= 2/4 = 50\%$.

For our experiments in a controlled scenario, we consider images taken with a single camera (OC) and with multiple cameras (MC) having similar scene semantics (the main content of the image is the same, but with small variations in the camera parameters, such as viewpoint, zoom, etc.). Three different datasets were used in our experiments, with images in the JPEG format:

- **Training Dataset**: It represents a small exploratory set with $2 \times 3^3 \times 4 \times 1 \times 10 = 2,160$ forests, containing images from OC and MC scenarios, taken from three different cameras, three different scenes, three images per camera, four forest sizes $|F| = \{2..5\}$, one topology, and 10 random variation of parameters for creating the near-duplicate images. A topology refers to the form of the trees in a forest.
- **Dataset A**: comprises images from OC and MC scenarios, three different scenes, three different cameras, three images per camera, four forest sizes $|F| = \{2..5\}$, four different tree topologies, and ten random variations of parameters for creating the near duplicate images. Therefore, the dataset comprises $2 \times 3^3 \times 4 \times 4 \times 10 =$ 8,640 forests. The Training Dataset as well as Dataset A are the same used by [23].
- **Dataset B**: comprises images randomly selected from a set of 20 different scenes, 10 different cameras, 10 images per camera, 10 different tree topologies, 10 random variations of parameters for creating the near duplicate images, and forests with 10 trees each. For each of the cases, OC and MC, a total of 2,000 forests within this set were randomly selected with forests of size $|F| = \{2..10\}$. Therefore, this set consisted of $2 \times 2{,}000 \times 9 = 36{,}000$ forests.

The image transformations used to create the near-duplicates are the same used in [8]: resampling, cropping, affine warping, brightness and contrast adjustment, and lossy compression using the standard lossy JPEG algorithm.

The experiments have two phases: first, we analyze the effects of choosing different values for parameter γ_{AOB} (threshold calculation), and the value of parameter k (amount of noise to be introduced in all variations of the dissimilarity matrix). Second, we evaluated the robustness of the approaches proposed in this paper in a controlled scenario. For the tests using the OB algorithm, we consider a black-box implementation[2] that follows Tarjan's description [29].

A. First Phase: Calculation of Parameters γ_{AOB} and k

The experiments described in this section were all performed using the Training Dataset previously described. To find the value of parameter γ_{AOB}, we studied the behavior of AOB using Algorithm 1, varying the thresholds in the interval $[\mu_{\mathrm{AOB}} - 2\sigma_{\mathrm{AOB}}, \mu_{\mathrm{AOB}} + 2\sigma_{\mathrm{AOB}}]$, separated by steps of 0.1. From this experiment, we defined $\tau_{\mathrm{AOB}} = \mu_{\mathrm{AOB}} + (2 \times \sigma_{\mathrm{AOB}})$, and as a consequence, $\gamma_{\mathrm{AOB}} = 2$.

The parameter k used to add noise in the dissimilarity matrices were chosen according to the formula described in Section IV-A. The parameter k was tested in the interval $[1\%, 10\%]$, separated by steps of 0.5%. Best results were presented for $k = 1.5\%$ of the σ_M value (recall that σ_M is the standard deviation of the input dissimilarity matrix).

B. Second Phase: Validation in a Controlled Scenario

In the second phase of our experiments, we analyzed the robustness of AOK, AOB, and E-AOB in two parts: (a) using each method separately and (b) their possible combinations C = {AOK × AOB, AOK × E-AOB, AOB × E-AOB, AOK × AOB × E-AOK} for Datasets A and B.

To directly compare the algorithms, the error variation $\Delta error$ was calculated with respect to each metric (roots, edges, leaves and ancestry), using the same equation introduced by Dias et al. [13]:

$$\Delta error_{metric}(\mathrm{M1},\mathrm{M2}) = \left(\frac{1 - \mathrm{M1}_{metric}}{1 - \mathrm{M2}_{metric}}\right) - 1 \tag{3}$$

where M1 represents the method being evaluated in comparison to method M2. For instance, on Table I, on column $\Delta error$ (E-AOB, AOK), M1 represents E-AOB, M2 represents AOK, and each of the columns **Roots, Edges, Leaves** and **Ancestry** represents the results for $\Delta error_{roots}$, $\Delta error_{edges}$, $\Delta error_{leaves}$, and $\Delta error_{ancestry}$, respectively.

A $\Delta error < 0$ value means that algorithm M1 performed better than M2, being able to reduce the error. Table I compares the methods AOB and E-AOB against AOK, with E-AOB algorithm outperforming the other two methods in both datasets, A and B, regardless of the number of trees in the forest. On average, E-AOB algorithm reduces the error for finding the roots in about 41% for metric roots, 7% for edges, 10% for leaves and 11% for ancestry in dataset A. For dataset B, the average of the error reduction is approximately 20% for metrics roots, leaves and ancestry, and 18% for edges.

A Wilcoxon signed-rank test was performed for all metrics in dataset B, comparing (i) AOK with AOB, and (ii) AOK with E-AOB. In Table I, results for this test are described in its last row, with the blue dots indicating that difference among the results are statistically significant, at 95% confidence level,

[2] Available at http://edmonds-alg.sourceforge.net

TABLE I

COMPARISON AMONG AOK [23] AND THE VARIATIONS OF AOB ALGORITHM. (a) SEMANTICALLY SIMILAR IMAGES FROM THE SCENARIO USING A SINGLE CAMERA (OC). (b) SEMANTICALLY SIMILAR IMAGES FROM THE SCENARIO USING MULTIPLE CAMERAS (MC)

(a)

	AOK				**AOB**				**E-AOB**				$\Delta error$ **(E-AOB, AOK)**			
	Dataset A															
$\|F\|$	**Roots**	**Edges**	**Leaves**	**Ancestry**	**Roots**	**Edges**	**Leaves**	**Ancestry**	**Roots**	**Edges**	**Leaves**	**Ancestry**	**Roots**	**Edges**	**Leaves**	**Ancestry**
2	0.834	0.788	0.818	0.740	0.842	0.805	0.834	0.755	0.911	0.809	0.839	0.773	-46.60%	-9.80%	-11.50%	-12.60%
3	0.882	0.823	0.827	0.798	0.898	0.833	0.842	0.814	0.938	0.838	0.847	0.829	-47.40%	-8.20%	-11.70%	-15.50%
4	0.870	0.831	0.820	0.826	0.861	0.835	0.827	0.828	0.914	0.839	0.833	0.843	-33.80%	-4.70%	-6.80%	-10.00%
5	0.883	0.780	0.812	0.762	0.887	0.788	0.824	0.778	0.930	0.791	0.829	0.787	-39.80%	-5.10%	-8.60%	-10.70%
	Dataset B															
$\|F\|$	**Roots**	**Edges**	**Leaves**	**Ancestry**	**Roots**	**Edges**	**Leaves**	**Ancestry**	**Roots**	**Edges**	**Leaves**	**Ancestry**	**Roots**	**Edges**	**Leaves**	**Ancestry**
2	0.803	0.852	0.838	0.806	0.786	0.865	0.857	0.822	0.833	0.872	0.864	0.841	-15.44%	-14.02%	-16.26%	-17.80%
3	0.850	0.874	0.856	0.845	0.808	0.887	0.882	0.852	0.885	0.896	0.891	0.879	-23.17%	-17.52%	-24.16%	-21.60%
4	0.854	0.878	0.859	0.848	0.799	0.893	0.886	0.856	0.892	0.903	0.897	0.884	-25.85%	-20.51%	-26.62%	-24.00%
5	0.854	0.879	0.859	0.844	0.768	0.892	0.884	0.843	0.889	0.903	0.896	0.878	-23.91%	-20.14%	-26.27%	-21.74%
6	0.849	0.880	0.859	0.842	0.760	0.893	0.881	0.839	0.879	0.904	0.894	0.872	-20.29%	-20.11%	-24.53%	-19.22%
7	0.832	0.880	0.856	0.830	0.750	0.894	0.878	0.833	0.868	0.905	0.891	0.864	-21.49%	-20.84%	-24.40%	-20.31%
8	0.799	0.883	0.856	0.817	0.723	0.899	0.880	0.824	0.841	0.910	0.894	0.855	-20.84%	-22.81%	-26.16%	-20.87%
9	0.778	0.884	0.856	0.802	0.704	0.899	0.879	0.810	0.822	0.910	0.892	0.840	-19.58%	-22.36%	-25.51%	-19.28%
10	0.755	0.883	0.854	0.784	0.682	0.898	0.877	0.793	0.802	0.908	0.890	0.823	-19.12%	-21.30%	-24.31%	-18.20%
	Wilcoxon				×	•	•	•	•	•	•	•				

(b)

	AOK				**AOB**				**E-AOB**				$\Delta error$ **(E-AOB, AOK)**			
	Dataset A															
$\|F\|$	**Roots**	**Edges**	**Leaves**	**Ancestry**	**Roots**	**Edges**	**Leaves**	**Ancestry**	**Roots**	**Edges**	**Leaves**	**Ancestry**	**Roots**	**Edges**	**Leaves**	**Ancestry**
2	0.830	0.787	0.817	0.739	0.837	0.805	0.833	0.755	0.908	0.809	0.839	0.773	-45.80%	-10.30%	-12.00%	-12.80%
3	0.883	0.822	0.822	0.801	0.873	0.831	0.837	0.811	0.936	0.837	0.845	0.832	-45.00%	-8.60%	-12.70%	-15.80%
4	0.887	0.830	0.817	0.833	0.835	0.830	0.822	0.821	0.925	0.838	0.832	0.846	-34.00%	-4.70%	-8.10%	-8.30%
5	0.898	0.782	0.814	0.775	0.868	0.786	0.824	0.777	0.937	0.791	0.831	0.794	-38.30%	-4.30%	-9.10%	-8.30%
	Dataset B															
$\|F\|$	**Roots**	**Edges**	**Leaves**	**Ancestry**	**Roots**	**Edges**	**Leaves**	**Ancestry**	**Roots**	**Edges**	**Leaves**	**Ancestry**	**Roots**	**Edges**	**Leaves**	**Ancestry**
2	0.810	0.859	0.838	0.818	0.765	0.865	0.848	0.819	0.830	0.875	0.856	0.844	-10.42%	-11.03%	-11.19%	-14.69%
3	0.854	0.874	0.852	0.843	0.782	0.879	0.863	0.840	0.876	0.891	0.875	0.872	-14.93%	-13.46%	-15.24%	-18.75%
4	0.872	0.878	0.854	0.855	0.781	0.883	0.863	0.850	0.890	0.896	0.876	0.884	-14.53%	-14.48%	-15.14%	-20.19%
5	0.882	0.877	0.853	0.856	0.776	0.884	0.867	0.849	0.903	0.897	0.879	0.887	-17.63%	-16.89%	-18.11%	-21.07%
6	0.895	0.877	0.854	0.863	0.772	0.885	0.866	0.852	0.911	0.898	0.880	0.890	-15.47%	-17.69%	-18.19%	-19.61%
7	0.897	0.880	0.855	0.867	0.778	0.889	0.868	0.859	0.917	0.903	0.883	0.896	-19.53%	-18.79%	-19.33%	-22.32%
8	0.900	0.886	0.856	0.872	0.785	0.895	0.871	0.866	0.922	0.908	0.886	0.901	-22.59%	-19.73%	-20.85%	-23.16%
9	0.893	0.887	0.860	0.867	0.774	0.897	0.874	0.860	0.915	0.909	0.889	0.896	-20.58%	-19.43%	-20.77%	-21.28%
10	0.885	0.888	0.862	0.859	0.768	0.896	0.875	0.852	0.909	0.908	0.889	0.886	-20.48%	-18.27%	-19.42%	-19.24%
	Wilcoxon				×	•	•	×	•	•	•	•				

in favor of AOB or E-AOB, while the red crosses represent statistical difference in favor of the baseline AOK. In case (i), statistical difference in favor of AOK was found for metric roots in the OC scenario, and for metrics roots and ancestry in the MC scenario. These results show that AOB is only able to improve the results of AOK regarding the metrics edges and leaves. On the other hand, when comparing AOK and E-AOB (Scenario ii), all differences are statistically significant in favor of E-AOB, confirming this method has better performance than the state-of-the-art method presented in the literature [23].

Using as baseline the results presented in Table I, better results for the fusion approach were found for the combination (AOK × AOB × E-AOB). Although the combination using only AOK and E-AOB seems to be a better choice, results of this fusion presented lower results compared to the one using the three methods together.

In our experiments, we noticed that the AOB method is currently important in the fusion approach, mainly because the dissimilarity calculation is not perfect. For instance, in some cases, the dissimilarity between a pair of images may be exchanged: node A is the real parent of node B, but the dissimilarity $d(B, A) < d(A, B)$. This will lead AOK to choose the wrong direction between these images, since it follows a greedy heuristic. Complementary, by taking into account all

TABLE II

RESULTS FOR FUSION (AOK × AOB × E-AOB) AND THE $\Delta error$ IN COMPARISON TO AOK AND E-AOB. (a) SEMANTICALLY SIMILAR IMAGES FROM THE SCENARIO USING A SINGLE CAMERA (OC). (b) SEMANTICALLY SIMILAR IMAGES FROM THE SCENARIO USING MULTIPLE CAMERAS (MC)

(a)

	Fusion (AOK × AOB × E-AOB)				$\Delta error$ **(Fusion, AOK)**				$\Delta error$ **(Fusion, E-AOB)**			
					Dataset A							
$\|F\|$	**Roots**	**Edges**	**Leaves**	**Ancestry**	**Roots**	**Edges**	**Leaves**	**Ancestry**	**Roots**	**Edges**	**Leaves**	**Ancestry**
2	0.917	0.811	0.841	0.777	-50.25%	-10.83%	-12.52%	-14.22%	-6.89%	-1.09%	-1.11%	-1.86%
3	0.943	0.840	0.85	0.833	-52.03%	-9.75%	-13.60%	-17.42%	-8.75%	-1.67%	-2.16%	-2.28%
4	0.926	0.843	0.838	0.848	-43.32%	-6.98%	-10.09%	-12.84%	-14.31%	-2.38%	-3.49%	-3.16%
5	0.932	0.794	0.831	0.789	-42.23%	-6.20%	-9.97%	-11.60%	-3.97%	-1.18%	-1.46%	-1.06%
					Dataset B							
$\|F\|$	**Roots**	**Edges**	**Leaves**	**Ancestry**	**Roots**	**Edges**	**Leaves**	**Ancestry**	**Roots**	**Edges**	**Leaves**	**Ancestry**
2	0.867	0.884	0.879	0.862	-32.34%	-22.02%	-25.61%	-28.64%	-19.99%	-9.31%	-11.17%	-13.18%
3	0.900	0.900	0.896	0.886	-33.08%	-21.03%	-27.74%	-26.15%	-12.90%	-4.25%	-4.71%	-5.81%
4	0.901	0.905	0.898	0.889	-32.19%	-22.25%	-27.83%	-26.82%	-8.55%	-2.19%	-1.65%	-3.70%
5	0.893	0.905	0.897	0.879	-26.25%	-21.61%	-26.84%	-22.20%	-3.07%	-1.84%	-0.78%	-0.60%
6	0.883	0.905	0.894	0.873	-22.45%	-20.89%	-24.73%	-19.83%	-2.71%	-0.97%	-0.27%	-0.75%
7	0.870	0.906	0.892	0.865	-22.88%	-21.59%	-24.78%	-20.86%	-1.77%	-0.95%	-0.51%	-0.70%
8	0.837	0.911	0.894	0.853	-18.95%	-23.59%	-26.40%	-19.68%	2.38%	-1.02%	-0.33%	1.50%
9	0.818	0.911	0.893	0.838	-18.00%	-23.41%	-26.07%	-18.29%	1.96%	-1.36%	-0.76%	1.23%
10	0.796	0.909	0.890	0.820	-16.62%	-22.13%	-24.76%	-16.62%	3.09%	-1.05%	-0.59%	1.93%
	Wilcoxon				•	•	•	•	–	•	•	–

(b)

	Fusion (AOK × AOB × E-AOB)				$\Delta error$ **(Fusion, AOK)**				$\Delta error$ **(Fusion, E-AOB)**			
					Dataset A							
$\|F\|$	**Roots**	**Edges**	**Leaves**	**Ancestry**	**Roots**	**Edges**	**Leaves**	**Ancestry**	**Roots**	**Edges**	**Leaves**	**Ancestry**
2	0.916	0.810	0.840	0.778	-50.90%	-10.97%	-12.83%	-14.70%	-9.35%	-0.70%	-0.96%	-2.15%
3	0.944	0.840	0.848	0.837	-52.24%	-10.34%	-14.34%	-18.23%	-13.13%	-1.86%	-1.87%	-2.86%
4	0.932	0.841	0.835	0.851	-40.42%	-6.62%	-10.16%	-10.97%	-9.77%	-2.01%	-2.21%	-2.96%
5	0.943	0.793	0.833	0.798	-43.95%	-5.23%	-10.21%	-10.20%	-9.16%	-0.98%	-1.26%	-2.09%
					Dataset B							
$\|F\|$	**Roots**	**Edges**	**Leaves**	**Ancestry**	**Roots**	**Edges**	**Leaves**	**Ancestry**	**Roots**	**Edges**	**Leaves**	**Ancestry**
2	0.871	0.889	0.871	0.869	-32.12%	-21.24%	-20.20%	-28.25%	-24.23%	-11.48%	-10.15%	-15.90%
3	0.900	0.898	0.88	0.884	-31.52%	-18.45%	-18.89%	-26.08%	-19.51%	-5.77%	-4.30%	-9.02%
4	0.905	0.899	0.879	0.891	-25.81%	-17.50%	-17.63%	-24.48%	-13.21%	-3.52%	-2.93%	-5.38%
5	0.916	0.900	0.882	0.891	-28.84%	-18.82%	-19.94%	-24.20%	-13.61%	-2.32%	-2.24%	-3.96%
6	0.921	0.901	0.883	0.895	-25.00%	-19.65%	-19.82%	-22.78%	-11.27%	-2.37%	-1.99%	-3.94%
7	0.928	0.905	0.885	0.900	-30.30%	-20.38%	-20.39%	-25.20%	-13.38%	-1.95%	-1.32%	-3.71%
8	0.931	0.910	0.888	0.904	-31.63%	-21.05%	-22.02%	-25.45%	-11.68%	-1.65%	-1.48%	-2.98%
9	0.926	0.911	0.891	0.900	-31.10%	-21.00%	-22.22%	-24.34%	-13.24%	-1.95%	-1.83%	-3.89%
10	0.917	0.910	0.891	0.890	-28.03%	-19.68%	-20.96%	-21.53%	-9.50%	-1.72%	-1.91%	-2.84%
	Wilcoxon				•	•	•	•	•	•	•	•

dissimilarities, AOB may choose the correct direction for some of these cases. With the re-execution performed by E-AOB (on each tree separately), some edges may change their direction to obtain the optimum branching. In some cases, E-AOB chooses edges that AOK also chooses erroneously. Therefore, in the fusion (AOK × E-AOB), since we take into account only the votes of both methods, if both of them have a large number of votes for some wrong edge, it will lead to the wrong result. However, when we include the votes from AOB, combined to the votes correctly obtained by E-AOB, we may be able to keep the right edge direction. If the dissimilarity calculation can be further improved, there is a possibility that the fusion AOK × E-AOB may present better results than the fusion of AOK × AOB × E-AOB. Improvements on this dissimilarity calculation is not included in the scope of this paper, but it is a challenge to be addressed by future research of the community. Results for the fusions (AOK × AOB), (AOK × E-AOB), and (AOB × E-AOB), can be found in the supplementar material.

Table II shows the results for the fusion (AOK × AOB × E-AOB) and the error variation in comparison to AOK and to the current best performing algorithm, E-AOB. In this fusion, OC scenario had lower performance only in dataset B, introducing more error for metrics roots and ancestry when the forest has more than eight trees. However, after running

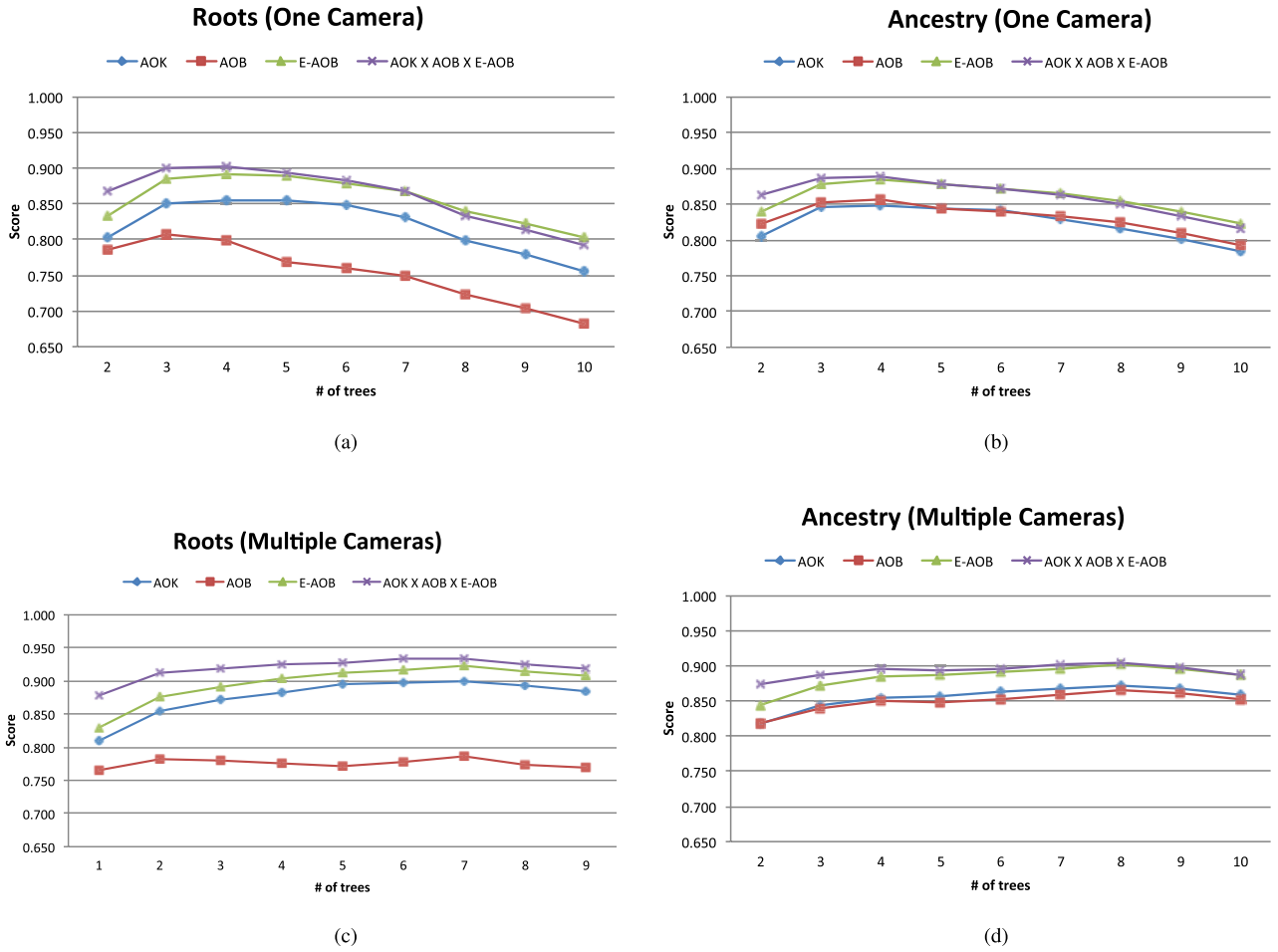


Fig. 4. Comparison among the proposed approaches for scenarios with a single camera (OC) and with multiple cameras (MC), featuring metrics roots and ancestry. (a) Metric: Roots (OC). (b) Metric: Ancestry (OC). (c) Metric: Roots (MC). (d) Metric: Ancestry (MC).

a Wilcoxon signed-rank test for these results, no statistical difference was found among them (represented by the green dashes in the last row of the table). For the other metrics in OC, and all metrics in MC scenario, differences are statistically significant at 95% confidence level in favor of the fusion approach.

The graphics in Figure 4 depict a comparison among the performance of all methods described in Table I and the fusion. Only metrics root and ancestry are the ones featured in these graphics because they are usually the most important in a forensic scenario. For instance, roots are important to find the source of the illegal activity, and with the ancestry relationship we can trace back the users involved in the chain of transformations. In the graphic, it becomes more clear the improvement obtained with E-AOB and the fusion in comparison to AOK and AOB.

Additionally, Figure 5 shows a comparison of the error reduction ($-\Delta error$) between methods E-AOB and the fusion AOK $\times$ AOB $\times$ E-AOB, in comparison to AOK, described in Table II. A closer analysis of the results shows that MC scenario has better performance than OC. In a forensic scenario this is advantageous since, in most of the cases, more than one user may be producing and spreading some illegal content from the same scene, and using different cameras. In this case, it is important to find enough evidences to frame all users involved in the illegal activity (e.g., cases of child pornography, for instance). Furthermore, the MC case is also more common in other scenarios. Consider, for instance, that we want to find all images related to a certain topic. In this case, the images will come from different photographers and higher are the chances they will be using different cameras. Therefore, the correct identification and separation of each group of images plays an important role.

Regarding the methods' running time, the biggest bottleneck corresponds to the dissimilarity calculation, as reconstructing the forest and performing the fusion is much faster. Considering a forest with 100 semantically similar images, the dissimilarity function step spends 38.31 minutes, on average, for all cases. On the other hand, the time to perform the fusion of three methods (including the time spent in 100 perturbations of each method, the votes counting, and the final execution of the fusion) takes, on average, 0.478 seconds. Tables III and IV details the running time for all forest sizes. These experiments were performed in a machine with processor Intel Xeon E5645, 2.40GHz, with 16GB of memory, and running Ubuntu 12.04.4 LTS (GNULinux 3.5.0-49-generic x86_64).

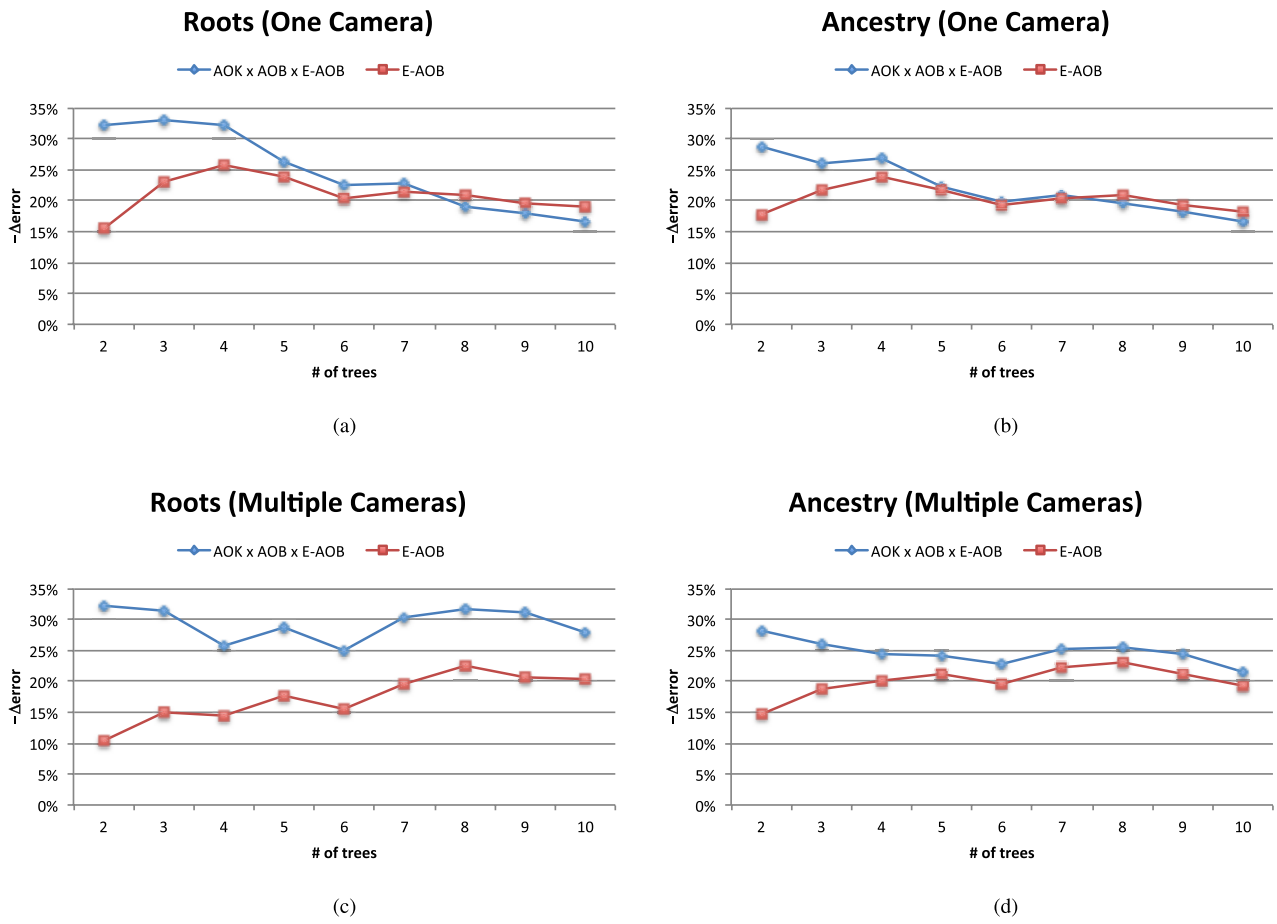


Fig. 5. Comparison among the error reduction ($-\Delta_{error}$) of methods E-AOB and the Fusion (AOK $\times$ AOB $\times$ E-AOB) in comparison to AOK, for metrics roots and ancestry. In (a) and (b), the graphs represent the scenario with images from a single camera (OC), while (c) and (d) show the results for images from the multiple camera (MC) scenario. (a) Metric: Roots (OC). (b) Metric: Ancestry (OC). (c) Metric: Roots (MC). (d) Metric: Ancestry (MC).

TABLE III

AVERAGE TIME (*minutes*) FOR THE DISSIMILARITY CALCULATION AND THE FOREST RECONSTRUCTION (SINGLE EXECUTION)

$\|F\|$	Dissimilarity	AOK	AOB	E-AOB
2	1.77	0.2 $\times 10^{-5}$	30.6 $\times 10^{-5}$	55.4 $\times 10^{-5}$
3	4.28	0.5 $\times 10^{-5}$	46.2 $\times 10^{-5}$	76.7 $\times 10^{-5}$
4	6.45	0.9 $\times 10^{-5}$	57.7 $\times 10^{-5}$	108.6 $\times 10^{-5}$
5	10.23	1.5 $\times 10^{-5}$	77.9 $\times 10^{-5}$	136.1 $\times 10^{-5}$
6	18.40	2.3 $\times 10^{-5}$	95.6 $\times 10^{-5}$	170.7 $\times 10^{-5}$
7	23.12	3.1 $\times 10^{-5}$	117.7 $\times 10^{-5}$	200.6 $\times 10^{-5}$
8	24.42	3.9 $\times 10^{-5}$	140.2 $\times 10^{-5}$	234.7 $\times 10^{-5}$
9	32.13	5.4 $\times 10^{-5}$	164.2 $\times 10^{-5}$	272.5 $\times 10^{-5}$
10	38.31	6.3 $\times 10^{-5}$	191.9 $\times 10^{-5}$	312.7 $\times 10^{-5}$

TABLE IV

AVERAGE TIME (*miliseconds*) CONSIDERING 100 EXECUTIONS TO PERFORM THE FUSION (AOK $\times$ AOB $\times$ E-AOB)

$\|F\|$	AOK	AOB	E-AOB	Votes	Fusion	Total
2	0.30	18.44	33.78	16.99	17.87	87.38
3	0.55	27.53	48.47	22.41	26.73	125.69
4	0.87	37.11	66.10	27.43	35.79	167.30
5	1.28	47.64	81.12	35.34	46.52	211.91
6	1.77	59.14	100.55	40.95	56.77	259.17
7	2.32	72.92	123.14	47.23	68.90	314.51
8	2.94	79.80	143.51	53.56	80.40	360.20
9	3.63	100.08	164.21	63.09	92.32	423.32
10	4.65	115.9	183.41	64.46	109.35	477.77

Finally, to ensure reproducibility of all experiments, all datasets used in the experiments and the entire source code will be freely available. The datasets are registered on the address: http://figshare.com/articles/Image_Phylogeny_Forests_Reconstruction/1012816 under the accession number http://dx.doi.org/10.6084/m9.figshare.1012816. The source code and documentation are available in a public repository in the following address: http://repo.recod.ic.unicamp.br/public/projects.

VI. CONCLUSIONS

In this paper, we introduced three new different approaches to deal with image phylogeny forests. First, we extended the approach developed for phylogeny trees by Dias et al. [13], using Optimum Branching and applying a similar idea used for automatic reconstruction of IPFs that the authors used for their Oriented Kruskal algorithm. By finding a threshold point after analysis of the behavior of valid forests, we made possible the use of the optimum branching algorithm

to deal with forests, resulting in the AOB method. Results were further improved with the proposal of E-AOB, which employs an important additional step of re-execution of the OB algorithm in each tree of the IPF found by AOB further refining the initial results. Both approaches outperformed the state-of-the-art AOK method presented in the literature.

We have also explored the idea of combining the best matches among the proposed methods, aiming at reducing the errors introduced by them when they are used independently, and improve the score of all metrics. Thus, we proposed a novel approach through the combination of the reconstructed forests of AOK, AOB and E-AOB methods. Results for this new approach showed better or equivalent performance to the best result achieved so far (E-AOB method), being a competitive solution for the image phylogeny problem.

Our fusion algorithm explores a view of the values of a dissimilarity matrix as random variables. Since the relative ordering between causal pairs of images can change when their difference is small enough, our method introduces robustness to statistical errors in the estimation of the image dissimilarities, and takes away some of the importance of extra fine tuning the dissimilarity calculation. Our approach can be easily extended to an arbitrary number of algorithms.

As future work, we intend to further analyze the behavior of the phylogeny algorithms using other statistical measures for the IPF reconstruction. For instance, Kurtosis is a statistical measure of extreme variation, such that higher values denote a few extreme deviations in the data, while lower values show more frequent, smaller deviations. Variations on the Kurtosis pattern have been previously studied in image forensics to detect, for instance, noise added to the image at various stages of production, which causes changes in the kurtosis [30], or image splicing, by identifying any portions of the image with significantly different Kurtosis values [31]. By exploring an analogous possibility, we aim at finding a relationship between the variation of the Kurtosis of the edges chosen by the phylogeny forest algorithms while reconstructing the forest, and the choice of the number of trees this forest should have.

Extensions to our approaches also include solving the phylogeny forest problem considering other media types, such as videos. We also intend to perform experiments with real cases we may find in the internet (non-controlled scenario) in which we do not have any knowledge about the history of generation of the duplicates.

REFERENCES

[1] A. Jaimes, S.-F. Chang, and A. Loui, "Duplicate detection in consumer photography and news video," in *Proc. ACM Workshop Multimedia Security*, 2002, pp. 423–424.

[2] F. Schaffalitzky and A. Zisserman, "Multi-view matching for unordered image sets, or 'How do i organize my holiday snaps?"' in *Proc. 7th Eur. Conf. Comput. Vis.*, 2002, pp. 414–431.

[3] D.-Q. Zhang and S. F. Chang, "Detecting image near-duplicate by stochastic attributed relational graph matching with learning," in *Proc. 12th Annu. ACM Int. Conf. Multimedia*, 2004, pp. 877–884.

[4] X. Cheng and L.-T. Chia, "Stratification-based keyframe cliques for removal of near-duplicates in video search results," in *Proc. ACM Int. Conf. Multimedia Inform. Retr.*, 2010, pp. 313–322.

[5] Z. X. H. Ling, F. Zou, Z. Lu, and P. Li, "Robust image copy detection using multi-resolution histogram," in *Proc. ACM Int. Conf. Multimedia Inform. Retr.*, 2010, pp. 129–136.

[6] Y. Ke, R. Sukthankar, and L. Huston, "Efficient near-duplicate detection and sub-image retrieval," in *Proc. ACM Workshop Multimedia Security*, 2004, pp. 869–876.

[7] J. Fridrich, D. Soukal, and J. Lukas, "Detection of copy-move forgery in digital images," in *Proc. Digital Forensics Res. Conf.*, 2003, pp. 134–137.

[8] Z. Dias, A. Rocha, and S. Goldenstein, "Image phylogeny by minimal spanning trees," *IEEE Trans. Inf. Forensics Security*, vol. 7, no. 2, pp. 774–788, Apr. 2012.

[9] B. Lewin, *Genes VI*. London, U.K.: Oxford Univ. Press, 1997.

[10] A. D. Rosa, F. Uccheddu, A. Costanzo, A. Piva, and M. Barni, "Exploring image dependencies: A new challenge in image forensics," *Proc. SPIE*, vol. 7541, no. 2, pp. 1–12, 2010.

[11] Z. Dias, A. Rocha, and S. Goldenstein, "First steps towards image phylogeny," in *Proc. IEEE Int. Workshop Inform. Forensics Security*, 2010, pp. 1–6.

[12] Z. Dias, A. Rocha, and S. Goldenstein, "Video phylogeny: Recovering near-duplicate video relationships," in *Proc. IEEE Int. Workshop Inform. Forensics Security*, Dec. 2011, pp. 1–6.

[13] Z. Dias, S. Goldenstein, and A. Rocha, "Exploring heuristic and optimum branching algorithms for image phylogeny," *J. Vis. Commun. Image Represent.*, vol. 24, no. 7, pp. 1124–1134, Oct. 2013.

[14] J. B. Kruskal, "On the shortest spanning subtree of a graph and the traveling salesman problem," *Proc. Amer. Math. Soc.*, vol. 7, no. 1, pp. 48–50, 1956.

[15] Y. J. Chu and T. H. Liu, "On the shortest arborescence of a directed graph," *Sci. Sinica*, vol. 14, no. 4, pp. 1396–1400, 1965.

[16] J. Edmonds, "Optimum branchings," *J. Res. Nat. Inst. Standards Technol.*, vol. 71B, no. 4, pp. 233–240, 1967.

[17] F. Bock, "An algorithm to construct a minimum directed spanning tree in a directed network," *Develop. Oper. Res.*, vol. 1, pp. 29–44, 1971.

[18] Z. Dias, S. Goldenstein, and A. Rocha, "Large-scale image phylogeny: Tracing image ancestral relationships," *IEEE Multimedia*, vol. 20, no. 3, pp. 58–70, Jul./Sep. 2013.

[19] L. Kennedy and S.-F. Chiang, "Internet image archaeology: Automatically tracing the manipulation history of photographs on the web," in *Proc. ACM 16th Int. Conf. Multimedia*, 2008, pp. 349–358.

[20] Z. Fan and R. L. Queiroz, "Identification of bitmap compression history: JPEG detection and quantizer estimation," *IEEE Trans. Image Process.*, vol. 12, no. 2, pp. 230–235, Feb. 2003.

[21] J. Mao, O. Bulan, G. Sharma, and S. Datta, "Device temporal forensics: An information theoretic approach," in *Proc. 16th IEEE Int. Conf. Image Process.*, Nov. 2009, pp. 1501–1504.

[22] J. R. Kender, M. L. Hill, A. Natsev, J. R. Smith, and L. Xie, "Video genetics: A case study from YouTube," in *Proc. Int. Conf. Multimedia*, 2010, pp. 1253–1258.

[23] Z. Dias, S. Goldenstein, and A. Rocha, "Toward image phylogeny forests: Automatically recovering semantically similar image relationships," *Forensic Sci. Int.*, vol. 231, nos. 1–3, pp. 178–189, 2013.

[24] M. Nucci, M. Tagliasacchi, and S. Tubaro, "A phylogenetic analysis of near-duplicate audio tracks," in *Proc. IEEE Int. Workshop Multimedia Signal Process.*, Oct. 2013, pp. 99–104.

[25] H. Bay, A. Ess, T. Tuytelaars, and L. V. Gool, "Speeded-up robust features (SURF)," *Comput. Vis. Image Understand.*, vol. 110, no. 3, pp. 346–359, 2008.

[26] M. A. Fischler and R. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Commun. ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[27] E. Reinhard, M. Ashikhmin, B. Gooch, and P. Shirley, "Color transfer between images," *IEEE Comput. Graph. Appl.*, vol. 21, no. 5, pp. 34–41, Sep./Oct. 2001.

[28] R. C. Prim, "Shortest connection networks and some generalizations," *Bell Syst. Tech. J.*, vol. 36, no. 6, pp. 1389–1401, 1957.

[29] R. E. Tarjan, "Finding optimum branchings," *Networks*, vol. 7, no. 1, pp. 25–35, 1977.

[30] D. Zoran and Y. Weiss, "Scale invariance and noise in natural images," in *Proc. IEEE 12th Int. Conf. Comput. Vis.*, Oct. 2009, pp. 2209–2216.

[31] X. Pan, X. Zhang, and S. Lyu, "Exposing image splicing with inconsistent local noise variances," in *Proc. IEEE Int. Conf. Comput. Photogr.*, Apr. 2012, pp. 1–10.

Filipe de O. Costa received the B.Sc. degree in computer science from the Federal University of Alfenas, Alfenas, Brazil, in 2010, and the master's degree in computer science from the University of Campinas, Campinas, Brazil, in 2012, where he is currently pursuing the Ph.D. degree with the Institute of Computing. His main research interests include digital forensics, computer vision, image processing, and machine learning.

Marina A. Oikawa received the B.Sc. degree in computer science from the Federal University of Para, Belém, Brazil, in 2006, and the master's and Ph.D. degrees in engineering from the Nara Institute of Science and Technology, Ikoma, Japan, in 2010 and 2013, respectively. She is currently a Post-Doctoral Researcher with the Institute of Computing, University of Campinas, Campinas, Brazil. Her main research interests include digital forensics, computer vision, and computer graphics.

Zanoni Dias received the B.Sc. degree and the Ph.D. degree in computer science from the University of Campinas (Unicamp), Campinas, Brazil, in 1997 and 2002, respectively. Since 2003, he has been an Assistant Professor with the Institute of Computing at Unicamp. His main interests include theoretical computer science, bioinformatics, and computational molecular biology.

Siome Goldenstein received the M.Sc. degree in computer science from Pontifícia Universidade Católica do Rio de Janeiro, Rio de Janeiro, Brazil, in 1997, the degree in electronic engineering from the Federal University of Rio de Janeiro, Rio de Janeiro, in 1995, and the Ph.D. degree in computer and information science from the University of Pennsylvania, Philadelphia, PA, USA, in 2002. He is an Associate Professor with the Institute of Computing, University of Campinas, Campinas, Brazil. His research interests lie in computer vision, computer graphics, forensics, and machine learning. He is an Area Editor of three journals, *Computer Vision and Image Understanding*, *Graphics Models*, and the IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY. He has been a Program Committee Member of multiple conferences and workshops, including the local organization of the 2007 IEEE International Conference on Computer Vision in Rio de Janeiro.

Anderson de Rezende Rocha received the B.Sc. degree from the Federal University of Lavras, Lavras, Brazil, in 2003, and the M.S. and Ph.D. degrees from University of Campinas (Unicamp), Campinas, Brazil, in 2006 and 2009, respectively, all in computer science. He is currently an Assistant Professor with the Institute of Computing at Unicamp. His main interests include digital forensics, reasoning for complex data, and machine intelligence. He has been a Program Committee Member in several important Computer Vision, Pattern Recognition, and Digital Forensics events. He is an Associate Editor of the Elsevier *Journal of Visual Communication and Image Representation* and a Leading Guest Editor of EURASIP/Springer *Journal on Image and Video Processing*. He is an Affiliate Member of the Brazilian Academy of Sciences and the Brazilian Academy of Forensics Sciences. He is also a member of the IEEE Information Forensics and Security Technical Committee and a Microsoft Research Faculty Fellow.