Interactive 3D Segmentation of Brain MRI with Differential Watersheds

Felipe P.G. Bergo *Alexandre X. Falcão*

Technical Report - IC-03-16 - Relatório Técnico

July - 2003 - Julho

Interactive 3D Segmentation of Brain MRI with Differential Watersheds

Felipe P.G. Bergo   Alexandre X. Falcão

bergo@seul.org, afalcao@ic.unicamp.br
Institute of Computing, State University of Campinas (Unicamp)
C.P. 6176, Campinas, SP, 13084-971, Brazil

Abstract

We approach here the interactive 3D segmentation problem in the context of the *image foresting transform* (IFT)— a graph-based tool to develop image processing operators— by introducing the IFT$^-$ algorithm to compute sequences of IFTs in a differential way. We instantiate the IFT$^-$ to be a watershed transform and validate it for segmentation of brain MR-images. The new method reduces by a factor of 8.4 the user's waiting time during segmentation compared to the non-differential watershed approach. It also provides a revertible operator that leads to higher efficiency gains than our previous differential approach, the IFT$^+$.

1 Introduction

Consider the problem of partitioning a scene into *influence zones* associated with a *seed set* given by the user, where the zone of each seed consists of the voxels that are "more closely connected" to that seed than to any other, in some appropriate sense. The segmentation is defined by assigning distinct labels to seeds that belong to distinct objects in the scene. Several effective approaches, such as watershed transform [1, 2, 3, 4, 5, 6] and relative fuzzy connectedness [7, 8], fall in this general model. Very often the user has to add and/or remove seeds (i.e. their influence zones) to correct the segmentation result. The time required to process the whole scene usually compromises the immediacy of response, making interactive segmentation a tedious task. The most efficient watershed algorithms, for example, take about 33 seconds to process a 256^3 scene on an 1.5GHz Pentium-4 PC [6, 9].

We have approached the above problem in the context of the *image foresting transform* (IFT)— a graph-based tool to develop image processing operators [10]. The IFT defines an *optimum-path forest* in a graph, whose nodes are the voxels and whose arcs are defined by an *adjacency relation* between voxels. The cost of a path in this graph is determined by an application-specific *path-cost function.* The roots of the forest are drawn from the given *seed set.* For suitable path-cost functions, the IFT assigns one minimum-cost path from the seed set to each voxel, in such a way that their union is an oriented forest, spanning the whole scene. The influence zone of each root is, by definition, its optimum-path tree in the forest. We have presented the IFT$^+$ algorithm to compute sequences of IFTs in a differential way [9], where the user is allowed to add and remove trees of the forest in time proportional to the number of voxels in these trees. That is, corrections during segmentation take time proportional to the number of voxels in the affected regions of the scene. We have instantiated the IFT$^+$ to be a watershed transform and showed that it reduces by a factor of 6 the user's waiting time during segmentation [9]. However, the IFT$^+$ algorithm cannot revert the user's actions during segmentation, because each tree removal turns all leaf voxels of its adjacent trees into new seeds.

We propose here a better and more intuitive differential solution, the IFT$^-$ algorithm, which allows the remaining roots to compete for the voxels of the removed tree without spawning new trees. The IFT$^-$ also provides an *undo* feature with no extra storage requirements, allowing the user to recover from mistakes. We instantiate it to be a watershed transform and show that it provides higher efficiency gains compared to the non-differential watershed approach for interactive 3D segmentation of brain MR-images.

2 Notation and Definitions

A *scene* $\mathbf{S}$ is a pair $(\mathcal{S}, S)$ consisting of a finite set $\mathcal{S}$ of *voxels* (points in Z^3), and a mapping S that assigns to each voxel v in $\mathcal{S}$ a *scalar* $S(v)$ in some arbitrary value space.

An *adjacency relation* $\mathcal{A}$ is an irreflexive binary relation between voxels of $\mathcal{S}$. We are interested here in symmetric adjacencies only. Once the adjacency relation $\mathcal{A}$ has been fixed, the scene $\mathbf{S}$ can be interpreted as a graph whose nodes are the voxels and whose arcs are the voxel pairs in $\mathcal{A}$.

A *path* π is a sequence of distinct voxels $\langle v_1, v_2, \ldots, v_n\rangle$ where $(v_i, v_{i+1}) \in \mathcal{A}$ for $1 \leq i \leq n-1$. The path is *trivial* if $n = 1$. If π is a path that ends at a voxel v, and τ is a path that begins at v, we denote by $\pi \cdot \tau$ the concatenation of the two paths, with the two joining instances of v merged into one. A *path-cost function* f assigns to each path π a *path cost* $f(\pi)$, in some totally ordered set $\mathcal{V}$ of cost values. Function f must satisfy certain conditions established by Falcão et al. in [10]. We denote the maximum element in $\mathcal{V}$ by $+\infty$. A path π is *optimum* if $f(\pi) \leq f(\pi')$ for any other path π' that ends at the same final voxel v of π, regardless of its starting point.

A *predecessor map* is a function P that assigns to each voxel v in $\mathcal{S}$ either some other voxel in $\mathcal{S}$, or a distinctive marker *nil* not in $\mathcal{S}$ — in which case v is said to be a *root* of the map. A *spanning forest* is a predecessor map which contains no cycles — in other words, one which takes every voxel to *nil* in a finite number of iterations. For any voxel $v \in \mathcal{S}$, a spanning forest P defines a path $P^*(v)$ recursively as $\langle v\rangle$ if $P(v) = nil$, and $P^*(u) \cdot \langle u, v\rangle$ if $P(v) = u \neq nil$. We will denote by $P^0(v)$ the initial voxel of $P^*(v)$. An *optimum-path forest* is a spanning forest P where $P^*(v)$ is optimum, for every voxel v.

We are interested in paths that start in a distinguished set $\mathcal{M}_I \subset \mathcal{S}$, the *seed voxels* (here given by the user). The user also assigns to each seed $v \in \mathcal{M}_I$ a label $\lambda(v)$. This restriction is implemented by choosing path cost $f(\langle v\rangle) = +\infty$ for all $v \notin \mathcal{M}_I$. We define $\mathcal{M}_R$ the set of voxels marked by the user, whose trees are to be removed from the forest. The IFT computes an optimum-path forest for a given scene $\mathbf{S}$, adjacency relation $\mathcal{A}$, suitable path-cost function f, labeling function λ, and seed set $\mathcal{M}_I$, and outputs an *annotated scene* that assigns to each voxel $v \in \mathcal{S}$ three additional attributes: its predecessor $P(v)$ in the optimum path, the cost $C(v)$ of that path, and a corresponding root label $L(v)$.

3 The IFT$^-$ Approach

The key idea in the IFT$^-$ approach is to allow addition and removal of influence zones (optimum-path trees) in the annotated scene in differential and revertible ways. When a new voxel is added to the seed set $\mathcal{M}_I$, it is supposed to define a new optimum-path tree by invading the influence zone of other roots. When an optimum-path tree is removed from the forest, its voxels become available for a new dispute among the remaining roots. A *symmetric* adjacency is crucial to guarantee that all available voxels will be reachable from the remaining roots.

The input and output of the IFT$^-$ algorithm are annotated scenes. Initially, $\mathcal{M}_R$ is empty and the annotated scene consists only of trivial trees, such that $C(v) = f(\langle v \rangle)$ and $P(v) = nil$, for all voxels $v \in \mathcal{S}$. The first IFT$^-$ will change the initial annotation into an optimum-path forest. For each subsequent addition and/or removal of trees, the IFT$^-$ will modify the annotated scene in a differential way.

Algorithm 1 – IFT$^-$

INPUT:	Scene **S**, cost map C, root label map L, predecessor map P, labeling function λ, path-cost function f, symmetric adjacency relation $\mathcal{A}$, set $\mathcal{M}_I$ of seed voxels, set $\mathcal{M}_R$ of marking voxels;
OUTPUT:	**S**, C, P and L;
AUXILIARY:	Priority Queue Q.

```
1.  $(C, P, \mathcal{F}) \leftarrow$ClearTrees$(C, P, \mathcal{A}, \mathcal{M}_I, \mathcal{M}_R)$;
2.  While $\mathcal{M}_I$ is not empty, Do
3.      Remove any $v$ from $\mathcal{M}_I$;
4.      If $f(\langle v \rangle) < C(v)$ Then
5.          Set $C(v) \leftarrow f(\langle v \rangle)$, set $P(v) \leftarrow nil$ and set $L(v) \leftarrow \lambda(v)$;
6.          Insert $v$ in $Q$;
7.  While $\mathcal{F}$ is not empty, Do
8.      Remove any $v$ from $\mathcal{F}$, insert $v$ in $Q$;
9.  While $Q$ is not empty, Do
10.     Remove from $Q$ a voxel $u$ such that $C(u)$ is minimum;
11.     For Each $v$ such that $(u, v) \in \mathcal{A}$ where $C(v) > C(u)$ or $P(v) = u$, Do
12.         Compute $cost \leftarrow f(P^*(u) \cdot \langle u, v \rangle)$;
13.         If $cost < C(v)$ or $P(v) = u$, Then
14.             If $v \in Q$ Then Remove $v$ from $Q$;
15.             Set $P(v) \leftarrow u$, $C(v) \leftarrow cost$ and $L(v) \leftarrow L(u)$, insert $v$ in $Q$;
```

Algorithm 2 – ClearTrees

Input:	Cost map C, predecessor map P, symmetric adjacency $\mathcal{A}$, set $\mathcal{M}_I$ of seed voxels, set $\mathcal{M}_R$ of marking voxels;
Output:	C, P, and set $\mathcal{F}$ of frontier voxels;
Auxiliary:	FIFO Queue T, Set $\mathcal{B}$ of frontier candidates.

```
1.  $\mathcal{F} \leftarrow \emptyset$;
2.  For Each $v \in \mathcal{M}_R$, Do
3.      $u \leftarrow P^0(v)$;
4.      If $C(u) \neq +\infty$ Then
5.          Insert $u$ in $T$, set $C(u) \leftarrow +\infty$, and set $P(u) \leftarrow nil$;
6.  While $T$ is not empty, Do
7.      Remove $u$ from $T$;
8.      For Each $v$ such that $(u,v) \in \mathcal{A}$, Do
9.          If $P(v) = u$ Then
10.             Set $C(v) \leftarrow +\infty$, set $P(v) \leftarrow nil$, insert $v$ in $T$;
11.         Else $\mathcal{B} \leftarrow \mathcal{B} \cup \{v\}$;
12. While $\mathcal{B}$ is not empty, Do
13.     Remove any $v$ from $\mathcal{B}$;
14.     If $P(v) \neq nil$ and $v \notin \mathcal{M}_I$ Then
15.         $\mathcal{F} \leftarrow \mathcal{F} \cup \{v\}$;
```

The key difference between the IFT$^-$ algorithm and the IFT$^+$ [9] is the different treatment of the user-added seeds ($\mathcal{M}_I$) and the frontier voxels ($\mathcal{F}$) found by the tree removal algorithm (lines 2–6 and 7–8 of Algorithm 1). In the IFT$^-$ approach, the voxels in the frontier region between removed and non-removed trees are not reset, and therefore do not become new roots. They only represent the propagation front of their trees. In the IFT$^+$ approach, these voxels become new roots and every iteration indeed increases the number of trees in the forest, preventing the user from reverting the operation. The IFT$^-$ allows both growth and reduction of the number of trees in the forest in a revertible way. That is, adding a root in one iteration and removing its tree in the next, and vice-versa, will revert the annotated scene to a state equivalent to the original one. (The forest may not be exactly the same, but it is a valid optimum-path forest for the remaining roots.) In practice, this provides an *undo* feature in the algorithm.

The auxiliary procedure (Algorithm 2) that builds the set $\mathcal{F}$ from $\mathcal{M}_R$ differs from the one used in the IFT$^+$ [9] in the separation of seeds from frontier voxels. The $\mathcal{F}$ set is no longer built in $\mathcal{M}_I$, and the $v \notin \mathcal{M}_I$ test in line 14 ensures that $\mathcal{F}$ and $\mathcal{M}_I$ are disjoint sets, preventing the double queueing of a same voxel.

The priority queue Q can be implemented in such a way that Algorithms 1 and 2 will run in time $O(|\mathcal{S}|)$ for adjacency relations that lead to sparse graphs and integer path costs with limited increments [10]. In practice, all iterations except the first will run in time proportional to a number of voxels much less than $|\mathcal{S}|$.

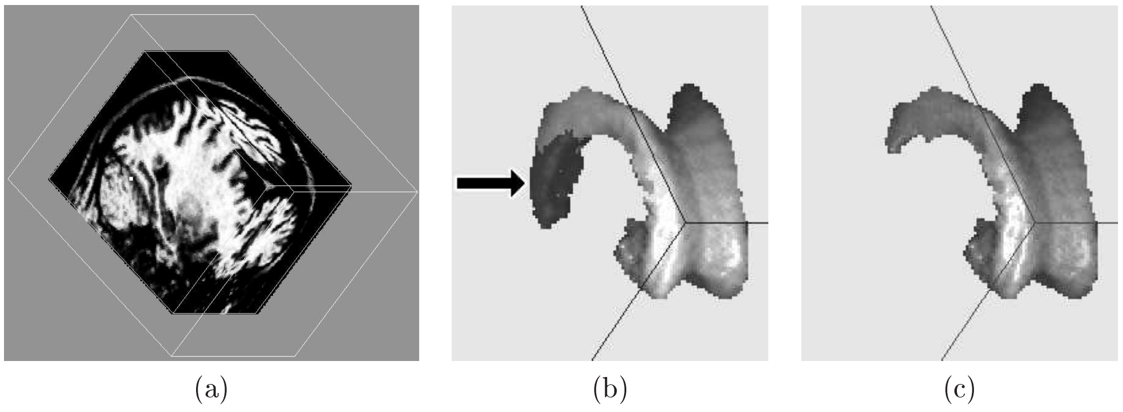
4 Application to Interactive 3D Segmentation

To evaluate the IFT$^-$ approach, we have instantiated it to be a differential watershed transform [1, 6]. In this case, a scene is a 4D surface where the brightness is the altitude of the voxels. An optimum-path between two voxels is defined as one with minimum height, where the height of a path is the maximum intensity of its voxels. Thus, the path-cost function represents the height of a path. The *IFT$^-$-watershed* uses a 6-neighborhood relation, and a path-cost function f defined as:

$$\begin{aligned} f(\langle v \rangle) &= 0, \text{ if } v \in \mathcal{M}_I, \text{ and } +\infty \text{ otherwise,} \\ f(\pi \cdot \langle u, v \rangle) &= \max\{f(\pi), S(v)\} \end{aligned}$$

where $S(v)$ is the brightness of voxel v in a gradient scene, which results from a preprocessing step. The preprocessing applies a Gaussian stretching to the original image, followed by a low-pass convolution filter, and morphological gradient computation. This was chosen to make more effective the segmentation of four objects of interest: cerebellum, lateral ventricles, pons-medulla and the rest of the brain.

Our implementation allows the user to add and remove trees of the forest by selecting seed and marking voxels over orthogonal cuts of the scene (Fig. 1a). After the first iteration, further selection can be performed over surface renditions and 3D views that combine partial segmentation results and anatomic information (Figs. 1b–c). Each label (object) is identified by a distinct color in the user interface. The user can perform as many iterations as desired, allowing him/her to choose the trade-off between the quality of segmentation and the time required to accomplish it.



(a) (b) (c)

Figure 1: (a) orthogonal cut view of the scene, used to make the initial selection of seeds. (b) example of selection of one marking voxel for tree removal over a 3D view of a partial segmentation of the lateral ventricles. (c) Result of the tree removal iteration. Inserting again the roots of the trees highlighted in (b) to the $\mathcal{M}_I$ set and running another iteration over the scene (c) reverts it to the state shown in (b).

5 Experimental Validation

The goals of the experiments were to evaluate the viability of the IFT$^-$-watershed for 3D segmentation and visualization at interactive speeds in a common PC with no specialized hardware, and its efficiency gains over the traditional non-differential approach [1, 6].

We selected ten T1-weighted MR scenes of the head from ten different patients with no known anomalies, acquired with 7.1×10^6 voxels (181x217x181, 12-bit) each, and used the IFT$^-$-watershed transform to segment lateral ventricles, cerebellum, pons-medulla and the rest of the brain. We used one 1.5 GHz Pentium-4 PC with 1.25 GB RAM running Linux to conduct our experiments (see Table 1).

Table 1: Experimental results.

Scene	# of Iterations	First Iteration Time	Correction Times Min–Max(Avg)	Response Avg. Time	Total Time for Segmentation
1	35	18.86"	1.58"–1.84" (1.67")	2.51"	21'48"
2	40	16.64"	1.59"–1.82" (1.65")	2.35"	21'28"
3	23	17.42"	1.55"–1.71" (1.60")	2.48"	15'16"
4	29	17.46"	1.56"–1.81" (1.62")	2.11"	14'23"
5	32	18.78"	1.57"–1.77" (1.62")	2.40"	17'03"
6	39	17.85"	1.57"–1.74" (1.61")	2.15"	17'20"
7	31	18.47"	1.56"–1.95" (1.63")	2.06"	15'11"
8	35	20.87"	1.56"–2.24" (1.66")	2.39"	17'41"
9	23	18.86"	1.59"–2.09" (1.69")	2.13"	11'43"
10	34	18.59"	1.69"–2.49" (1.79")	2.16"	26'01"

Differential iterations (corrections) were performed until visual inspection of orthogonal cuts showed no relevant innacuracies (see Fig. 2). Combined with a fast 3D projection method, an IFT$^-$-based segmentation tool can process and display corrections in about 2 seconds (5th column of Table 1). In a non-differential approach, each correction would require the same time of the first iteration (3rd column of Table 1), leading to response times around 20 seconds including 3D visualization. Note that the interactive segmentation with the non-differential watershed approach would become lenghty and tedious. Considering the total CPU time spent by the IFT$^-$ and the non-differential IFT algorithms, the IFT$^-$ approach provided an efficiency gain of 8.40, on average (see Table 2).

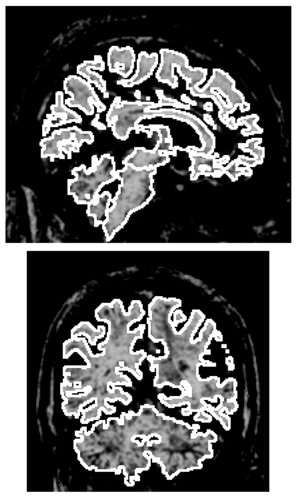
The efficiency gain of the IFT$^-$ is also higher than the one obtained with the IFT$^+$ approach (5.75, as shown in [9]). With the IFT$^+$, the number of trees never decreases with each iteration, and the annotated scene quickly becomes oversegmented. Further corrections become harder due to the increased difficulty of visualizing, selecting and removing undesired and/or misplaced influence zones— more time is spent selecting seeds than performing correction iterations.

Oversegmentation is not an issue with the IFT$^-$, where mistakes can be readily undone and tree removal actually reduces the number of trees in the forest, allowing the user to

Table 2: Experimental Efficiency Gain of the IFT^- Approach

Scene	IFT^- CPU time	IFT CPU time	Efficiency Gain	Scene	IFT^- CPU time	IFT CPU time	Efficiency Gain
1	75.54”	660.10”	8.74	6	79.06”	696.15”	8.81
2	80.90”	665.60”	8.23	7	67.43”	572.57”	8.49
3	52.68”	400.66”	7.61	8	77.41”	730.45”	9.44
4	62.88”	506.34”	8.05	9	56.00”	433.78”	7.75
5	69.02”	600.96”	8.71	10	77.55”	632.06”	8.15

proceed with further corrections as much as she/he desires.


(a)

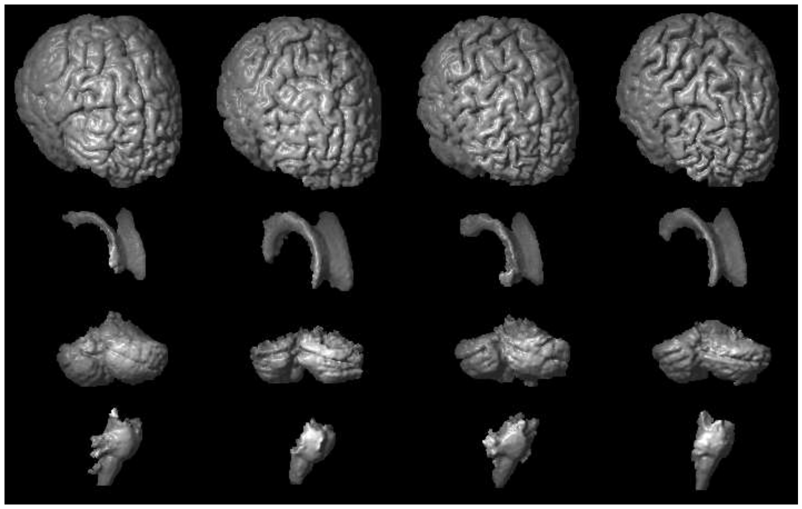

(b)

Figure 2: (a) Examples of orthogonal views with border highlighting used for inspection of segmentation accuracy. (b) 3D renditions of segmented structures in 4 of the 10 test subjects. Top to bottom: brain, lateral ventricles, cerebellum and pons-medulla; Left to right: subjects 1, 5, 7 and 9.

6 Conclusion

We introduced a new *differential image foresting transform* (IFT^-), which computes sequences of IFTs in differential and revertible ways, and instantiated it to be a watershed operator. We evaluated the IFT^--watershed in the context of interactive 3D segmentation of brain MR-images, and showed that the IFT^--watershed is, on average, 8.4 times faster than the non-differential approach. The revertibility property of the IFT^- makes segmentation easier, more intuitive and faster than with the IFT$^+$ approach [9], by providing a stable tree removal method without oversegmentation. For 7-Mvoxel scenes, the IFT$^-$-watershed provides feedback for differential segmentation corrections in about 2.5 seconds, including 3D visualization, on a 1.5 GHz Pentium 4 PC. We can conclude that the main contribution of the IFT$^-$ is the considerable reduction in the response time to user's actions, which makes it viable for routine use in clinical settings.

Our current research aims at improving and automating scene preprocessing for other structures in the brain, and creating new methods for 3D segmentation based on the IFT framework. We also intend to conduct further experiments to evaluate the accuracy, precision, and efficiency gains of the IFT$^-$.

Acknowledgments

Thanks to Dr. Fernando Cendes and Dr. Li Li Min (Department of Neurology, State University of Campinas) for providing the data sets and useful feedback in our segmentation results. Bergo thanks CAPES for his scholarship. Falcão is grateful to CNPq (Proc. 302966/02-1) for the financial support.

References

[1] S. Beucher and F. Meyer. The morphological approach to segmentation: The watershed transformation. In Edward R. Dougherty, editor, *In Mathematical Morphology in Image Processing*, chapter 12, pages 433–481. Marcel Dekker, Inc., New York, NY, 1993.

[2] L. Vincent and P. Soille. Watersheds in digital spaces: An efficient algorithm based on immersion simulations. *IEEE on Pattern Analysis and Machine Intelligence*, 13(6):583–598, June 1991.

[3] W.E. Higgins and E.J. Ojard. Interactive morphological watershed analysis for 3D medical images. *Computerized Medical Imaging and Graphics*, 17(4/5):387–395, 1993.

[4] G. Bueno, O. Musse, F. Heitz, and J.P. Armspach. Three-dimensional segmentation of anatomical structures in MR images on large data bases. *Magnetic Resonance Imaging*, 19:73–88, 2001.

[5] R.J. Lapeer, A.C. Tan, and R. Aldridge. Active watersheds: Combining 3D watershed segmentation and active contours to extract abdominal organs from MR images. In *5th MICCAI, Lecture Notes in Computer Science 2488*, pages 596–603, Tokyo, 2002. Springer-Verlag.

[6] R.A. Lotufo and A.X. Falcão. The ordered queue and the optimality of the watershed approaches. In *Mathematical Morphology and its Applications to Image and Signal Processing*, volume 18, pages 341–350. Kluwer Academic Publishers, Palo Alto, USA, June 2000.

[7] P.K. Saha and J.K. Udupa. Relative fuzzy connectedness among multiple objects: theory, algorithms, and applications in image segmentation. *Computer Vision and Image Understanding*, 82:42–56, 2001.

[8] T. Lei, J.K. Udupa, P.K. Saha, and D. Odhner. Artery-vein separation via MRA - An image processing approach. *IEEE Transactions on Medical Imaging*, 20(8), 2001.

[9] A.X. Falcão and F.P.G. Bergo. The iterative image foresting transform and its application to user-steered 3D segmentation. In *Proceedings of SPIE on Medical Imaging*, volume 5032, pages 1464–1475, San Diego, CA, May 2003.

[10] A.X. Falcão, J. Stolfi, and R.A. Lotufo. The image foresting transform: Theory, algorithms and applications. Technical Report IC-03-04, IC-Unicamp, April 2003. to appear on IEEE Trans. on Pattern Analysis and Machine Intelligence.