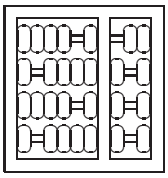


INSTITUTO DE COMPUTAÇÃO
UNIVERSIDADE ESTADUAL DE CAMPINAS

Object detection using tree pruning

A.X. Falcão *P.A.V. Miranda*
F.P.G. Bergo

Technical Report - IC-05-30 - Relatório Técnico

December - 2005 - Dezembro

Object detection using tree pruning

Alexandre X. Falcão, Paulo A.V. Miranda and Felipe P.G. Bergo
Institute of Computing, University of Campinas
{afalcao,paulo.miranda,felipe.bergo}@ic.unicamp.br

Abstract

We propose a new approach for object detection using the image foresting transform (IFT). For a given image with seed pixels inside the object, the IFT computes an optimum-path forest in a graph derived from the image, such that object and background are connected by a few optimum paths (*leaking paths*). The leaking paths cross the object's boundary through its "most weakly connected" parts (called *leaking pixels*). The topology of the forest is exploited to identify the leaking pixels and eliminate their subtrees, such that the remaining forest defines the object. The method is called *tree pruning*. We present tree pruning with an automatic approach to detect leaking pixels, which does not require parameters; show several examples; discuss its limitations; and evaluate the method for automatic detection of license plates using a database with 990 images.

1 Introduction

We consider the problem of precisely defining the spatial extent of a desired object in a given image, namely object detection. Object detection is a hard and important problem in image analysis, which very often requires user intervention [10, 5, 9, 2, 12, 6]. We present a method, called *tree pruning*, that reduces user intervention to the selection of a few pixels inside the object. The major advantages of tree pruning are: it runs in linear time, it is free of *ad-hoc* parameters, it is extensive to multidimensional images, and it allows automatic solutions when the internal seeds can be estimated. For example, we can use approaches that locate the object in the image [17] to estimate seeds. Candidate seeds can otherwise be used to obtain a set of possible objects, and the desired one can be chosen based on object features or objective functions [14, 11]. In the worst case, candidate objects can be used as effective initializations for deformable models [10, 3, 5, 4].

Tree pruning uses the *Image Foresting Transform* (IFT)— a general approach to the design of image processing operators based on connectivity [8]. In the IFT, an image is interpreted as a graph whose nodes are image pixels and whose arcs are defined by an *adjacency relation* between pixels. For a given set of *seed pixels* and suitable *path-cost function*, the IFT computes a minimum-cost path forest in the graph whose roots are drawn from the seed set. Each tree in the forest consists of pixels more strongly connected to its root than to any other seed. In tree pruning, the seeds must be chosen inside the object and the choice of the path-cost function intends to connect object and background by a few optimum paths (*leaking paths*), which cross the object's boundary through its "most weakly connected" parts (called *leaking pixels*). The topology of the forest is exploited to identify the leaking pixels and eliminate their subtrees, such that the remaining forest defines the object.

In comparison to region-growing approaches that use optimum paths from internal seeds [15], the criterion to disconnect object and background is not based on the costs of the optimum paths but on the topology of the forest. Optimum paths that reach object pixels are assumed to not pass through the background, instead of having costs strictly lower than the costs of paths that reach the background.

In tree pruning, internal seeds compete among themselves and only a few seeds become roots of leaking paths. Other approaches based on optimal seed competition [1, 16, 13] can be roughly described in three steps: (i) seed pixels are selected inside and outside the objects, (ii) each seed defines an *influence zone* that contains pixels which are more strongly connected to that seed than to any other, and (iii) each object is defined by the union of the influence zones of its internal seeds. The absence of boundary information and/or heterogeneity of the background usually cause invasion (leaking) of object seeds in influence zones of background seeds and vice-versa. Tree pruning differs from these approaches in two aspects: it does not require external seeds (simplifying seed selection) and exploits the leaking problem to solve segmentation.

In comparison with graph-cut approaches that use terminal nodes and hard constraints (seed pixels) [2], tree pruning does not require external seeds, uses a simpler image graph, and runs in time proportional to the number of pixels.

Tree pruning was first presented by Falcao et al. [7], with two approaches to detect leaking pixels. One is interactive where the user can visually identify leaking pixels and select them with the mouse pointer. The other is automatic, but relies on *ad-hoc* parameters. In the present paper, we introduce a new automatic approach for detection of leaking pixels, which is free of parameters, discuss advantages and limitations of the method, and evaluate it for a real application.

We describe tree pruning in Section 2 with several examples that illustrate its advantages and limitations. Section 3 evaluates the method for automatic detection of license plates using a database with 990 images. We state conclusion in Section 4.

2 Tree-pruning segmentation

We start from a rationale similar to that of a watershed transform [1, 16]. In the watershed transform by markers, an object's boundary is a watershed line resulting from a simulated flooding. The method assumes internal and external seed pixels (markers) with one source of water at the location of each seed; simulates a flooding process over the topographic surface of a *gradient-like image*; and erects barriers (watershed lines) wherever two bodies of water coming from internal and external sources meet.

In tree pruning we have only internal sources and let the water leak to the background through the lower pixels on the object's boundary. In practice, boundaries do not usually have the same height and these leaking pixels are not so many. That is, the streams of water reaching the background usually pass through a few leaking pixels. The identification of the leaking pixels allows the erection of a barrier at their location, separating the water inside from the water outside the object.

The gradient-like image must be higher on the object's boundary than inside the object and, at least, in a neighborhood outside it. As we will see, this *gradient condition* is important for the effectiveness of the method independent of the seed location.

In the following sections we instantiate the IFT [8] for tree pruning, present the new approach for automatic detection of leaking pixels, discuss how to create gradient-like images that satisfy

the above condition, and show several examples that illustrate the advantages and limitations of the method.

2.1 The image foresting transform for tree pruning

A gradient-like image is interpreted as a directed graph, whose nodes are the image pixels and whose arcs are defined by an irreflexive 4− or 8-connected adjacency relation between pixels (Figure 1a).

A *path* is a sequence of adjacent pixels and a *path-cost function* c assigns to each path π a *path cost* $c(\pi)$. A path π is *optimum* if $c(\pi) \leq c(\tau)$ for any other path τ with the same destination of π. We can simulate the flooding process using path-cost function $c_{\max}$ and seed set S inside the object:

$$c_{\max}(\pi) = \begin{cases} \max_{\forall p \in \pi}\{G(p)\} & \text{if } org(\pi) \in S \\ +\infty & \text{otherwise} \end{cases} \quad (1)$$

where $G(p)$ is the intensity of pixel p in a gradient-like image G and $org(\pi)$ is the origin of path π.

In these conditions, the IFT computes in linear time an optimum path from S to every pixel p [8]. These paths form an optimum-path forest rooted in S and are stored in a *predecessor map* P— a function that assigns to each pixel $p \notin S$ its predecessor $P(p)$ in the optimum path from S and a marker *nil* when $p \in S$. Figure 1b shows an example where the predecessor map represents a single optimum-path tree rooted at the seed shown in Figure 1a.

Object and background in Figure 1b are connected by pixel (4,4), which represents the lowest intensity on the object's boundary (i.e. a leaking pixel). Clearly, this result will depend on the gradient condition. If it is satisfied, the object can be obtained by eliminating the subtrees of the leaking pixels (Figure 1c).

When two optimum paths reach an ambiguous pixel p with the same minimum cost, p is assigned to the first path that reached it. That is, we break ties using *first-in-first-out* policy in the IFT algorithm [8]. After leaking, this policy makes ambiguous the pixels in the neighborhood outside the object, ramifying the leaking paths into several branches and avoiding that other paths reach the background (see Figure 1b). This allows to exploit the topology of the forest for detection of leaking pixels.

2.2 Automatic detection of leaking pixels

Let B be a set of background pixels outside the object. For example, the image's border. All paths that reach the image's border B pass through the leaking pixels (Figure 1b). We compute the number of descendants that every pixel of the forest has in B to obtain a descendant map D (Figure 1d). Note that, by following backwards the optimum path in P from any pixel of B to its respective root, the first occurrence of the maximum descendant count is at a leaking pixel (i.e., $p = (4, 4)$ with $D(p) = 36$ in Figure 1d).

Moreover, since all leaking pixels must have descendants in B, the sum of $D(p)$ for all leaking pixels p is the cardinality of B. Therefore, multiple leaking pixels can be automatically detected by repeating the above procedure for every pixel in B until we achieve the cardinality of B.

Figure 2 illustrates an example of license plate detection, where the gradient-like image is the magnitude of the Sobel's gradient. Note that the gradient condition is not satisfied, but the method works as long as we do not select a seed inside a letter or digit of the plate.

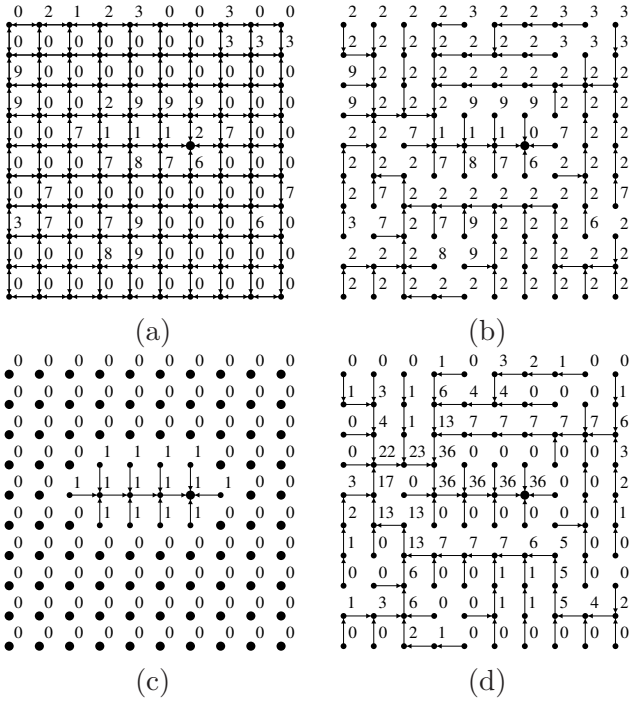


Figure 1: (a) Four-connected graph where the numbers are gradient intensities and the bigger dot denotes a seed inside an object. (b) An optimum path tree for $c_{\max}$ rooted at that seed. (c) The object can be detected by eliminating the subtree of the leaking pixel at position (4,4). (d) The number of descendants in the image's border B for the optimum path tree of Figure 1b. The first maximum in the way back from any pixel in B to the root is at (4,4).

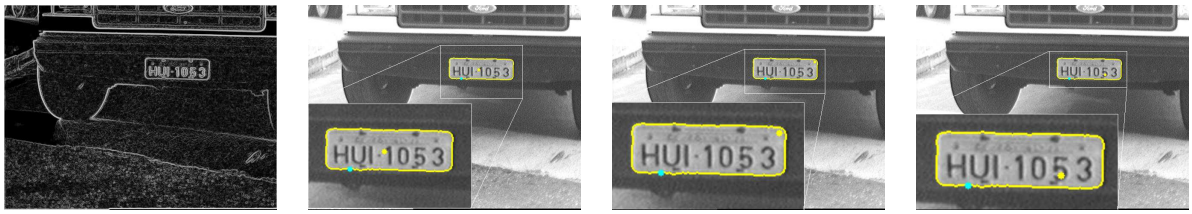
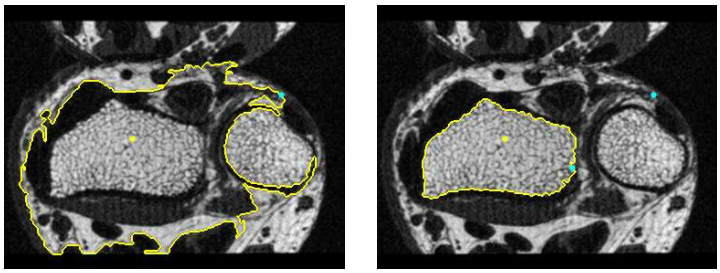


Figure 2: (a) The magnitude of the Sobel's gradient of an image for license plate detection. (b)–(d) The plate is automatically detected for different seed locations (yellow dots) using (a) as gradient-like image. The leaking pixel is always at a same place (cyan dots).



(a) (b)

Figure 3: An MR image of a wrist where its boundary can be detected with two iterations of the method for a single seed (yellow dot). (a) The first iteration uses the image's border as set B and automatically detects an external boundary with a single leaking pixel (cyan dot). (b) The external boundary is used as set B to automatically detect the object's boundary with a single leaking pixel (cyan dot) in the second iteration.

Figure 3 illustrates other situation where the object's boundary is inside (constraint by) a false boundary and all paths that reach the image's border pass through a leaking pixel (cyan dot) of the external boundary. In this case, we can not use the image's border as set B to detect the object, but we can execute the method twice. In the first iteration, we use the image's border to detect the external boundary, and we detect the object in the second interaction by using the external boundary as set B.

2.3 Gradient-like images

In order to satisfy the gradient condition, preprocessing is sometimes necessary to enhance the boundary of interest in detriment of other boundaries. For example, we first assign a membership value for each pixel with respect to the object based on image features.

Let $\mathbf{x}_p$ be a feature vector computed at a given pixel p; μ_p and $\mathbf{\Sigma}_p$ be mean and covariance matrices of the feature vectors $\mathbf{x}_q$ computed at all pixels q within an adjacency radius around p; and T be a set of training pixels, selected in object regions that have different image features. For a given pixel $s \in T$, a membership value $R_s(p)$ is computed for every image pixel p.

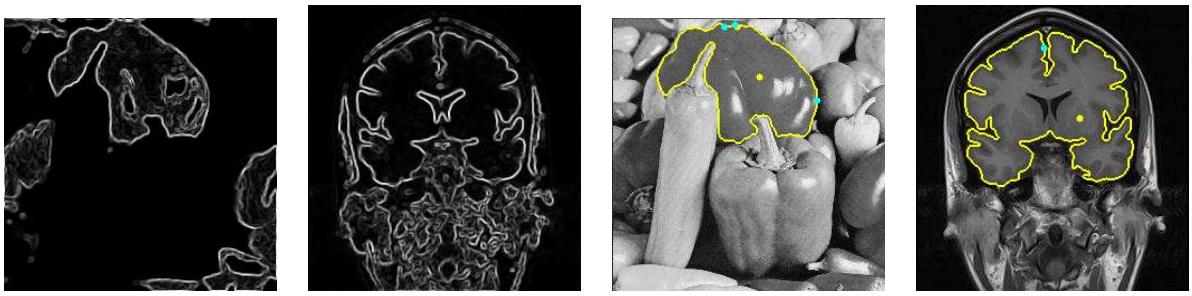
$$R_s(p) \quad = \quad \exp\left(-\frac{1}{2d}(\mathbf{x}_p - \mu_s)^t \mathbf{\Sigma}_s^{-1}(\mathbf{x}_p - \mu_s)\right) \tag{2}$$

where $d > 1$ takes into account the absence of statistical information (e.g., we use $d = 10$). We also set a distinct adjacency radius for each pixel $s \in T$, making it as largest as possible, in order to compute the best estimation for μ_s and $\mathbf{\Sigma}_s$ inside the object region that includes s. A region map R is obtained as

$$R(p) \quad = \quad \max_{\forall s \in T}\{R_s(p)\}. \tag{3}$$

Finally, we apply a median filter on R to make it more homogeneous and use a gradient magnitude of R as gradient-like image (see Figure 4). In these examples, we used three training pixels and two normalized attributes within $[0, 1]$ for the feature vectors of Equation 2: the original red and green values (Figure 4a); and brightness and gradient magnitude (Figure 4b).

In order to demonstrate the effectiveness of tree pruning, we will only use simple gradient-like images created by computing the magnitude of the Sobel's gradient on the original image.

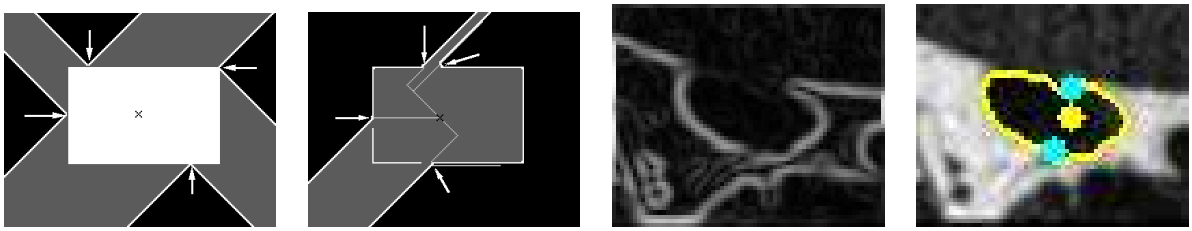


(a) (b) (c) (d)

Figure 4: (a)–(b) Gradient-like images based on region maps for the image of peppers and an MR-brain image. (c)–(d) Results of segmentation for both images, where the seeds are manually selected (yellow dots) and the leaking pixels (cyan dots) are automatically detected.

2.4 Limitations

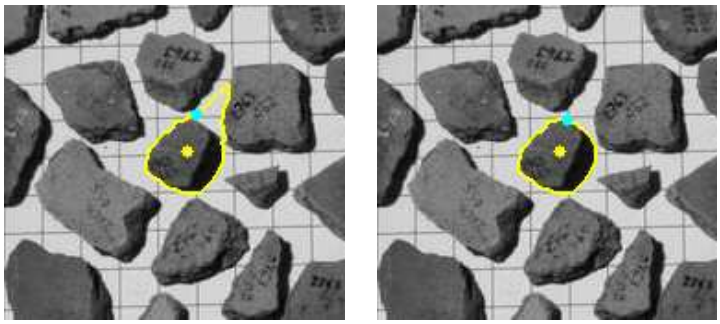
Curiously, the method fails in perfect situations, such as the one illustrated in Figure 5a, because almost all boundary pixels are leaking pixels. A similar problem occurs when the weaker parts of the boundary are perfect gaps (Figure 5b). However, in the presence of noise, the method usually works for boundaries with gaps (Figures 5c and 5d).



(a) (b) (c) (d)

Figure 5: Issues with tree pruning: (a) Perfect object border and (b) perfect gaps after four interactive prunings (arrows). (c) The magnitude of Sobel's gradient applied to an MR-image of a vein in the wrist. (d) Result of segmentation for an internal seed (yellow dot) using (c) as gradient-like image. Two leaking pixels are automatically detected (cyan dots).

Another problem may occur when multiple objects with similar gradient intensities are very close to each other. The absence of space in between the objects to ramify the path at a boundary's pixel creates a false leaking pixel outside the object (Figure 6a). If the gradient condition is satisfied, we may assume that the correct location of a leaking pixel is always at the maximum gradient value in the path segment between the detected pixel and its root. We call this *variant of maximum gradient*. Note that it does not affect the location of real leaking pixels on the object's boundary, but solves the problem as illustrated in Figure 6b.



(a) (b)

Figure 6: Image with one object fragment marked for detection. (a) The detected pixel is outside the fragment due to its proximity to the other fragments. (b) The correct leaking pixel is automatically detected using the variant of maximum gradient.

3 Evaluation

We evaluate the effectiveness of tree pruning for automatic detection of license plates using a database with 990 images of 352×240 pixels each (Figure 7a). The magnitude of the Sobel's gradient is used as gradient-like image (Figure 7b).

In this application, automatic seed selection is a difficult task because any attempt to estimate seeds inside a plate is likely to find seeds in other parts of the image. A natural strategy is to select markers as seed sets (connected components); run tree pruning for each marker separately; score each candidate object; and choose the one with the best score. The score of an object is obtained based on shape features, since the plates are slightly deformed rectangles.

Marker selection and object score are *ad-hoc* processes, but their parameters can be fixed using our prior knowledge about the problem. Better procedures are left for future work.

3.1 Marker selection

Marker selection aims at isolating some pixels around the plate's number, given that they are dark pixels in our database. Figure 7c shows a binary image of those pixels in Figure 7a with value below 30% of its maximum intensity. If no plate is detected at this threshold, the algorithm reduces the threshold to 15% of the maximum intensity and repeats the segmentation process. We apply an erosion with a disk of radius 5.0, followed by a dilation with a disk of radius 7.0 (Figure 7d), and consider the components in Figure 7c which do not belong to Figure 7d as markers. These parameters are too small to compromise the effectiveness of the method (e.g., 93% of the plates have sizes around 2936 pixels). The larger dilation radius was chosen to avoid seeds over the plate's border. We also eliminate components that touch the image's border and components with less than 6 pixels. Markers in the top region of the image (30% of the height) are rejected since there is no license plate there. The resulting markers are finally dilated by a disk of radius 1 to assure some seeds around the plate's number, and labeled with a distinct number (Figure 7e).

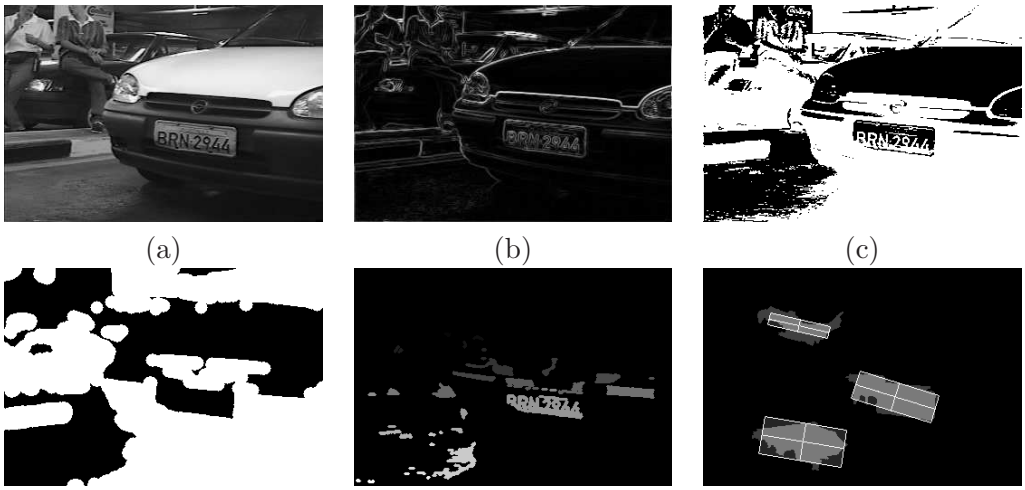


Figure 7: (a) Original image. (b) Gradient image. (c) Image (a) after thresholding. (d) Image (c) after erosion and dilation. (e) Image of labeled markers. (f) The best rectangles for three false candidate objects.

3.2 Object score

Tree pruning is applied for each marker with a distinct label and the resulting object receives a score proportional to its similarity to a rectangular shape. At the end, the object with the best score is selected as a license plate. This process is restricted to an interest region of 200×100 pixels around each marker. We also reject objects with less than 1200 pixels or more than 8200 pixels, since the size of the plates varies between 1559 and 7753 pixels.

First, we find the best fit between each candidate object and a rectangle. To guarantee rotation invariance, we determine the object's major semi-axis by principal component analysis. Then starting at its geometric center, we go along the major semi-axis in both orientations, alternately and at same time, until we reach the background in some orientation. The same process is repeated to the orthogonal semi-axis. These boundary points provide us all necessary information to trace a rectangular model. Figure 7f shows three of these rectangles for false candidates.

Considering each rectangle as a pseudo "ground truth", we compute a score SC for the corresponding object based on false positives FP and false negatives FN, as follows.

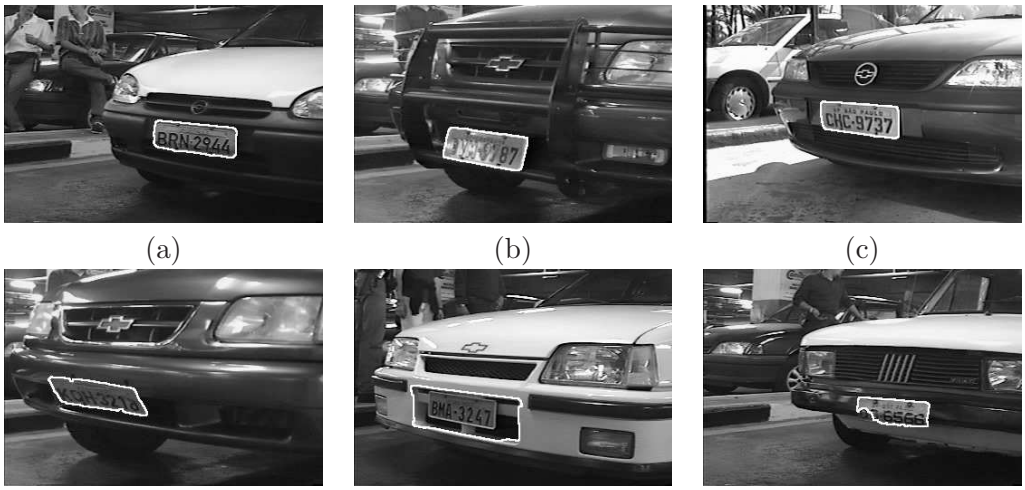
$$\begin{aligned} SC &= (1.0 - \min\{1.0, FP + FN\}) * W \\ W &= \begin{cases} 1.0 & \text{if } 2.4 \leq R \leq 3.0, \\ \exp\left(-(2.4 - R)^2/3.0\right) & \text{if } R < 2.4, \\ \exp\left(-(3.0 - R)^2/3.0\right) & \text{if } R > 3.0. \end{cases} \end{aligned} \tag{4}$$

where R is the aspect ratio of the rectangle and the interval $[2.4, 3.0]$ was chosen in between the minimum and maximum expected aspect ratios of the plates.

3.3 Results

The method correctly detected 931 (94.04%) license plates out of 990. Some examples of correct and incorrect segmentations are shown in Figure 8. We could always estimate markers inside the plates. Among the 59 plates not detected, the method failed due to the object score process for 28 plates, and tree pruning failed for the remaining 31 plates (only in 3.13% of the database).

According to a recent publication [18], where the authors use a database with 784 images of 384×288 pixels, our results seem to be good. The images in this database have slightly higher resolution and the license plates seem to appear less deformed in shape. Zheng et al. [18] have evaluated four distinct approaches and found an average rate of 88.83% of correct detections, where their method obtained 99.7% of correct detections and the minimum rate was 82.0%.



(a) (b) (c)

(d) (e) (f)

Figure 8: (a)-(d) Correct detections with scores 0.937, 0.913, 0.902, and 0.767, respectively. (e)-(f) Wrong detections with scores 0.917 and 0.801, respectively.

4 Conclusion

We introduced tree-pruning segmentation with automatic detection of leaking pixels and showed with several examples that it can reduce object detection to the choice of a few internal seeds. Tree pruning was evaluated for automatic segmentation of license plates, where it correctly detected 94.04% of the plates in a database with 990 images.

We are currently evaluating tree pruning in other applications involving 2D and 3D images, and the method has shown to be an effective approach for object detection.

Acknowledgments

The authors thank Roberto Lotufo (FEEC-UNICAMP) for the images of license plates. This work was supported by CNPq (Proc. 302427/04-0), CAPES and FAPESP (Proc. 03/09793-1 and Proc. 03/13424-1).

References

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

[2] Y. Y. Boykov and M.-P. Jolly. Interactive graph cuts for optimal boundary & region segmentation of objects in N-D images. In *International Conference on Computer Vision (ICCV)*, volume 1, pages 105–112, 2001.

[3] L. Cohen. On active contour models and ballons. *Computer Vision, Graphics, and Image Processing: Image Understanding*, 53(2):211–218, 1991.

[4] T. Cootes, G. Edwards, and C.J.Taylor. Active appearance models. In *European Conference on Computer Vision (ECCV)*, volume 2, pages 484–498, 1998.

[5] T. Cootes, C. Taylor, D. Cooper, and J. Graham. Active shape models – their training and application. *Computer Vision and Image Understanding*, 61(1):38–59, 1995.

[6] A. X. Falcão and F. P. G. Bergo. Interactive volume segmentation with differential image foresting transforms. *IEEE Trans. on Medical Imaging*, 23(9):1100–1108, 2004.

[7] A. X. Falcão, F. P. G. Bergo, and P. A. V. Miranda. Image segmentation by tree pruning. In *XVII Brazilian Symposium on Computer Graphics and Image Processing (SIBGRAPI)*, pages 65–71. IEEE, Oct 2004.

[8] A. X. Falcão, J. Stolfi, and R. A. Lotufo. The image foresting transform: Theory, algorithms, and applications. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 26(1):19–29, 2004.

[9] A. X. Falcão, J. K. Udupa, S. Samarasekera, S. Sharma, B. E. Hirsch, and R. A. Lotufo. User-steered image segmentation paradigms: Live-wire and live-lane. *Graphical Models and Image Processing*, 60(4):233–260, 1998.

[10] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *International Journal of Computer Vision*, 1:321–331, 1987.

[11] V. Kolmogorov and R. Zabih. What energy functions can be minimized via graph cuts. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 26(2):147–159, Feb 2004.

[12] H. T. Nguyen, M. Worring, and R. van den Boomgaard. Watersnakes: energy-driven watershed segmentation. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 25(3):330–342, Mar 2003.

[13] 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.

[14] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 22(8):888–905, Aug 2000.

[15] J. K. Udupa and S. Samarasekera. Fuzzy connectedness and object definition: theory, algorithms, and applications in image segmentation. *Graphical Models and Image Processing*, 58:246–261, 1996.

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

[17] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Intl Conf. on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages I–511–I–518, 2001.

[18] D. Zheng, Y. Zhao, and J. Wang. An efficient method of license plate location. *Pattern Recognition Letters*, 2005. Article in press, available at www.sciencedirect.com.