A hierarchical approach to high resolution edge contour reconstruction *

Stefano Casadei

e-mail: casadei@lids.mit.edu
Massachusetts Institute of Technology
Cambridge Ma, 02139

Sanjoy K. Mitter

e-mail: mitter@lids.mit.edu
Massachusetts Institute of Technology
Cambridge Ma, 02139

Abstract

Efficient edge detection algorithms such as Canny's fail near curve singularities. Moreover, the standard linking algorithms used on top of these detectors often fail because of instabilities in the tracking process (due to multiple responses to the same edge and interference of nearby edges). We propose a hierarchical approach to edge detection based on a graph stabilization method that allows bifurcation resolution in stages. Curve singularities are recovered at the last stage by using "top-down" feedback to select the best curve connections.

1 Introduction

The simplest and most widely used method to detect edges in an image consists in seeking points where the brightness variation is above a threshold or locally maximum in the direction of the gradient [1, 8, 14]. Curves representing edges are then obtained from these points by a "greedy" linking algorithm which links each point to one of its neighbors according to orientation similarity or maximum gradient intensity.

One well-known limitation of this approach is that it fails near curve singularities such as corners and T-junctions. Another difficulty is that curve tracking can be "instable", namely a small localization error at some point can disrupt completely the tracking process (see figure 1).

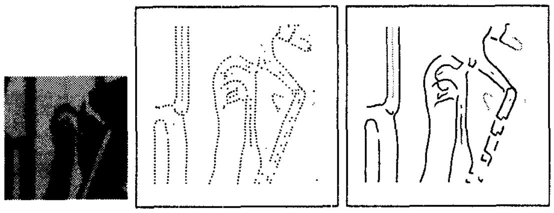


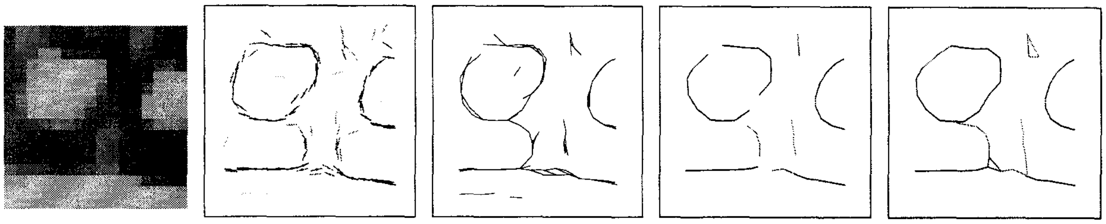
Figure 1: Canny's edge detector algorithm with subpixel accuracy (center) followed by greedy edge linking (right). Edges are tracked by choosing the neighboring edge-point most aligned with the local estimate of edge orientation. Notice that the algorithm fails near curve singularities and edge tracking is instable when multiple responses to the same edge or nearby interfering edges are present.

Specialized techniques have been proposed to recover singular edge-points [4, 16, 15]. However, these methods are computationally expensive because they require either filtering the image with operators tuned to all orientations [16, 14] or solving an optimization problem with several variables at every location in the image [4, 15]. Moreover, the problem of integrating these points into a unified curve representation is not addressed by these approaches.

To make edge tracking more robust and less prone to instable behavior, one has to examine and integrate edge information in large enough "contextual" neighborhoods. Relaxation labeling [7, 13] is a way of doing this. The approach suggested in [13, 19], which also estimates curvature, is capable of representing multiple orientations at the same point and therefore curve singularities. Their lateral maximization procedure presents some similarities to the arc suppression algorithm used here. Probabilistic relaxation methods have also been proposed [6, 11]. The main drawback of relaxation techniques is that their iterative nature entails high computational costs.

To include global information more efficiently, a hierarchical approach based on recursive grouping can be used [10, 12, 5, 17]. The idea is to use a hierarchy of descriptors of increasing complexity and spatial ex-

*Research supported by US Army grant DAAL03-92-G-0115, Center for Intelligent Control Systems



(b) Edge-points. (c) Edge-segments. (d) Regular curves. (e) Linked curves.

Figure 2: The hierarchy of descriptors computed from a 16 × 16 subimage of the telephone in figure 8.

tent. The descriptors at some level of this hierarchy are obtained by grouping elements in the representation at the previous level. Grouping allows many descriptors to be replaced by a single token so that the contextual neighborhood can be progressively enlarged without causing a combinatorial explosion of the search space. Ambiguities and inconsistencies in the representation are gradually eliminated as one moves up in the hierarchy.

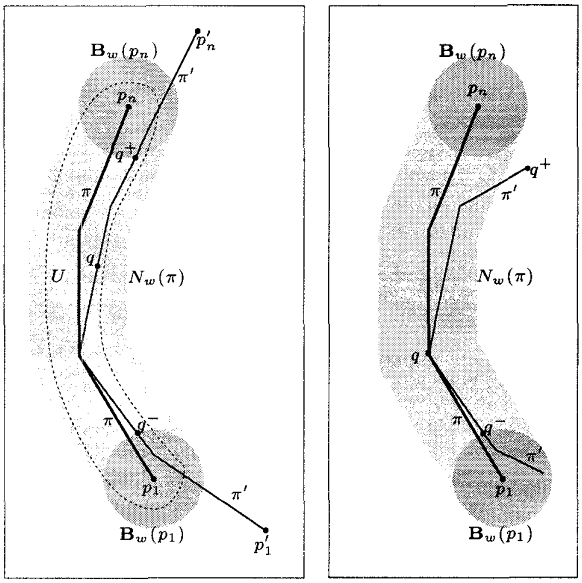
A shortcoming of most existing hierarchical algorithms is that computation is purely "bottom-up" and relies completely on a symbolic edge-point representation such as Canny's. Moreover, the goal is often limited to finding large and global structures in the image at a coarse scale.

The aim of this paper is to propose a hierarchical algorithm for edge detection where top-down feedback is used to select hypotheses about curve singularities. The final result is a curve representation of edges which includes also fine details such as corners and T-junctions. This hierarchy contains five levels (see figure 2): *brightness data, edge-point hypotheses, edge-segment hypotheses, regular curves, linked curves.* The lowest levels (2nd and 3rd) typically contain many ambiguities about edge localization (i.e. multiple responses to the same edge) which are resolved at the higher levels. The goal of the 4th level is to guarantee stability and smoothness of the tracking process. This is done by *separating* the problem of recovering singularities from the problem of edge tracking. The problems of graph stabilization and edge tracking are briefly described in section 2 (see [3] for a complete treatment). Finally, the 5th level recovers singularities and bridges small gaps by hypothesizing and testing all possible short-range curve connections (section 3). A brightness model is constructed for each hypothesis and tested against the raw data. This provides a "top-down" feedback into the final decision process. Experimental results are shown in figure 8.

2 Stabilization of curve tracking

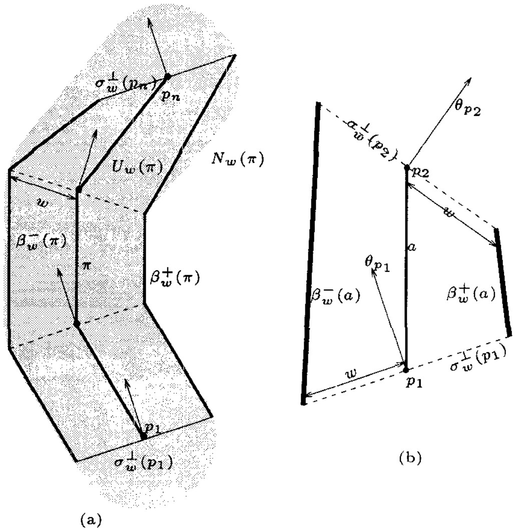
As figure 1 illustrates, one the problems in reconstructing curves from local observations is to make the edge tracking process *stable.* Namely, every path starting near a curve should remain close to it. This is a difficult problem in the presence of noise because stability is a global property (curves can be arbitrarily long) whereas the data is derived locally. The input data to the tracking algorithm (namely the data flowing into the 4th level of the hierarchy) can be represented as a *vector graph* (P, V, A) where P is the set of edge-point candidates; V is a discrete vector field on P; and $A \subset P \times P$ is a graph structure on P. See [3] for more details. The edge-segments in figure 2(c) are the arcs of this vector graph (the orientation of the vector field is shown in figure 2(b)). Recently, we have developed a theory on how to stabilize a vector graph by means of "arc suppression" so that all the arcs necessary to track curves are retained in the graph [3]. The proposed algorithm is capable to reconstruct the set of all "true" regular curves Γ up to some bounded localization error, under the assumption that disturbances are also bounded. A key ingredient of this theory is the definition of a *stable graph* (see figures 3 and 4). Roughly speaking, a graph is stable if every path π in it is an *attractor*, namely if every path π' which contains a point in the *attraction basin* of π does not diverge from π. After computing a stable and *complete*[1] subgraph, the algorithm at the 4th level of the hierarchy extracts a set of disjoint paths by using dynamic programming to minimize a total curvature cost. This set contains an approximation to every "true" curve. The computational complexity of the algorithm is linear in the cardinality of P.

[1] Namely, all the arcs necessary to reconstruct the true curves belong to this subgraph.



(a) (b)

Figure 3: The path π in (a) is an attractor because every other path π' containing a point q in $U \subset N_w(\pi)$ does not diverge from π. (Here $N_w(\pi)$ denotes the set of points with distance from π less than the scale parameter w). More precisely, the maximal subcurve of π' containing q and disjoint from the balls $\mathbf{B}_w(p_1)$ and $\mathbf{B}_w(p_n)$ (the curve between q^- and q^+) belongs to U. The path π in (b) is *not* an attractor (for the given w).



(a) (b)

Figure 4: (a): The attraction basin $U_w(\pi)$ constructed by the stabilization algorithm for the path π. Its boundary is composed of four parts $\partial U_w(\pi) = \beta_w(\pi) = \beta_w^-(\pi) \cup \sigma_w^\perp(p_n) \cup \beta_w^+(\pi) \cup \sigma_w^\perp(p_1)$. The two lateral components of this boundary are given by $\beta_w^\pm(\pi) = \cup_{a \in A_\pi} \beta_w^\pm(a)$ where A_π are the arcs of π and $\beta_w^\pm(a)$ are the lateral segments of a shown in (b). Notice that the orientation of the vector field at the end-points of an arc (θ_{p_1} and θ_{p_2}) is used to determine the vertices of its lateral segments. Notice also that each point in $U_w(\pi)$ has distance from π less than w, that is, $U_w(\pi) \subset N_w(\pi)$. If an arc of the graph intersects $\beta_w(\pi)$, then a ***stability violation*** occurs and the stabilization algorithm suppresses one of the two arcs involved in the violation [3, 2].

3 Recovering singularities

The curve representation computed by the 4th level of the hierarchy is disconnected at corners (high curvature points), junctions and sometimes also far from edges singularities. A parameter which determines how difficult it is to fix these disconnections is the size of these gaps. It turns out that many of these gaps are only a few pixels wide and therefore they can be "bridged" with local information only. Unfortunately, local information is not sufficient to find always the correct result[2]. However, it makes sense to fill as many gaps as possible without resorting to global information which can lead to high computational costs. This is consistent with the hierarchical point of view, according to which global information should be introduced as gradually as possible and local processing should always be exploited maximally to provide global stages with as much information as possible. Then one can use feedback from these global stages to correct errors at the lower levels. The experimental results presented in figure 8 demonstrate that many gaps and junctions can be reconstructed correctly by using information in small neighborhoods.

The local algorithm for curve linking described here uses three inputs: the brightness image; the set of edge-points P from the 2nd level of the hierarchy; and the regular curves from the 4th level. The search for curve connections, called *links*, is conducted from the end-points of simple curves, called *terminators*. Three types of links are possible.

- T-T link. This is a direct link from a terminator t_1 to another terminator t_2. It consists of one "central" segment (*bridge*) $\overline{t_1 t_2}$ plus two segments $\overline{t_1 p_1}$, $\overline{t_2 p_2}$, called *extensions*, lying on the curves which t_1, t_2 are terminators of (see figure 5).

[2] See for instance the petals in the second row of figure 8 where, due to the high density of curve terminators, curves are sometimes linked in the wrong way.

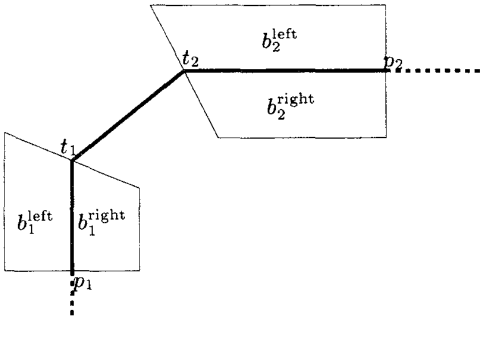


Figure 5: The regions used to estimate the brightness around a link.

- T-M-T link. This is a link between two terminators via a "mediator" point m. An element of P, in order to be a mediator, has to lie outside all the attraction basins of the computed curves (see figure 6). The bridge of this link consists of two connected segments, $\overline{t_1m}$, $\overline{mt_2}$. The total number of segments (including the extensions) is 4.
- T-C link. This is a link from a terminator to a point of a curve which is not a terminator. The algorithm examines these links only if a valid link of the other two types is not found.

In this implementation, the search for a mediator, terminator, or curve point is conducted in the 5×5 neighborhood of each terminator (see figure 7).

To determine whether a particular link from a terminator is a good hypothesis, the brightness on each side of the link is examined. More precisely, let $b_1^{\text{left}}(\lambda)$, $b_2^{\text{left}}(\lambda)$, $b_1^{\text{right}}(\lambda)$, $b_2^{\text{right}}(\lambda)$ be the average brightness on the four "corners" of the link $\lambda \in \Lambda_t$, where Λ_t is the set of all T-T and T-M-T links from the terminator t (see figure 5). A homogeneity cost on each side of the link is defined as follows:

$$J^s_{\text{hom}}(\lambda) = |b^s_1(\lambda) - b^s_2(\lambda)|, \qquad s = \text{left,right}$$

The geometric cost of the link is given by:

$$J_{\text{geom}}(\lambda) = c_1 \cdot \text{length}(\lambda) + c_2 \cdot \text{turn}(\lambda)$$

where turn(λ) is the total turn (curvature) of the link. Thus the total cost on each side of the link λ is $J^s(\lambda) = J^s_{\text{hom}}(\lambda) + J_{\text{geom}}(\lambda)$. These two cost functions are minimized separately so that a left and a right link are obtained for each terminator t:

$$\lambda^s(t) = \operatorname*{argmin}_{\lambda \in \Lambda_t} J^s(\lambda), \qquad s = \text{left,right}$$

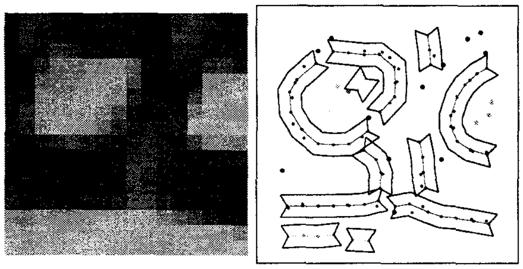


Figure 6: The mediators (large dots) are edge-point hypotheses which lie outside the attraction basins of the computed regular curves.

To avoid inconsistent groupings, only links which are optimal at both terminators are included in the final representation. More precisely, the link $\lambda^{\text{left}}(t)$ is kept only if $\lambda^{\text{right}}(u^{\text{left}}(t)) = t$, where $u^{\text{left}}(t)$ is the terminator at the other end of the link $\lambda^{\text{left}}(t)$. Similarly, $\lambda^{\text{right}}(t)$ is kept only if $\lambda^{\text{left}}(u^{\text{right}}(t)) = t$. If after this final step a terminator is not linked to any other terminator, then the algorithm looks for an optimal T-C link by minimizing a similar cost function.

4 Conclusions

It is often believed that to improve edge detection beyond the performance attained by local algorithms such as Canny's one has to resort to either iterative and computationally expensive methods such as [13] or to methods which require human intervention, such as snakes [9, 18]. In this paper we have proposed a hierarchical approach to edge detection which yields improved performance over Canny's algorithm by using a sequence of local, non-iterative procedures which have linear computational complexity in the number of image pixels. Two key ingredients of this approach are stabilization of edge tracking and model-based feedback to reconstruct curve singularities.

To improve edge detection even further one has to add more levels to the hierarchy so that more global information can be used efficiently. For instance, to improve detection of the flower petals in the second row of figure 8 one can include region descriptors into the process so that curve groupings which give rise to closed and convex boundaries are favored over others. Also, one can increase gradually the gap size at which curves are grouped to deal more robustly with occlusion and shadows and to recover objects with "invisible" edges.

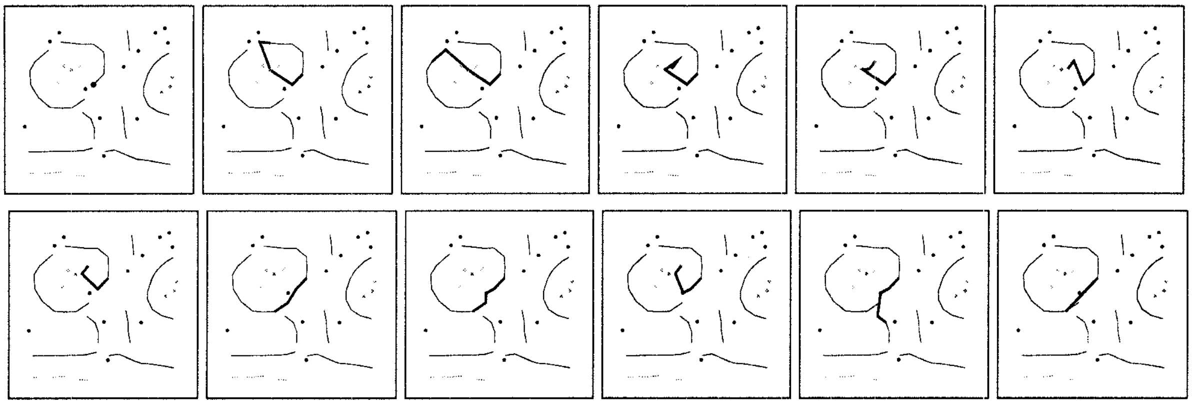


Figure 7: The set of candidate links from a particular terminator. All links except the last one are T-T or T-M-T links. The last one is a link to a point on a curve (T-C link).

References

[1] J. Canny. A computational approach to edge detection. *PAMI*, 8:679–698, 1986.

[2] S. Casadei and S.K. Mitter. Hierarchical curve reconstruction. part 1: bifurcation analysis and recovery of smooth curves. In *ECCV*, 1996.

[3] S. Casadei and S.K. Mitter. Hierarchical image segmentation. part 1: Detection of regular curves, 1996. in preparation.

[4] R. Deriche and T. Blaszka. Recovering and characterizing image features using an efficient model based approach. In *CVPR*, June 15-18 1993.

[5] J. Dolan and E. Riseman. Computing curvilinear structure by token-based grouping. In *CVPR*, 1992.

[6] S. Geman and D. Geman. Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *PAMI*, 6:721–741, November 1984.

[7] E.R. Hancock and J. Kittler. Edge-labeling using dictionary-based relaxation. *PAMI*, 12:165–181, 1990.

[8] R. Haralik. Digital step edges from zero crossing of second directional derivatives. *PAMI*, 6(1):58–68, 1984.

[9] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *Int. J. of Computer Vision*, 1:321–331, 1988.

[10] D. Lowe. *Perceptual Organization and Visual Recognition.* Kluwer, 1985.

[11] J. Marroquin, S. Mitter, and T. Poggio. Probabilistic solution of ill-posed problems in computational vision. *J. Am. Stat. Ass.*, 82(397):76–89, Mar. 1987.

[12] R. Mohan and R. Nevatia. Perceptual organization for scene segmentation and description. *PAMI*, 14, June 1992.

[13] P. Parent and S.W. Zucker. Trace inference, curvature consistency, and curve detection. *PAMI*, 11, August 1989.

[14] P. Perona and J. Malik. Detecting and localizing edges composed of steps, peaks and roofs. In *ICCV*, pages 52–57, Osaka, 1990.

[15] K. Rohr. Recognizing corners by fitting parametric models. *Int. J. of Computer Vision*, 9:3, 1992.

[16] L. Rosenthaler, F. Heitger, O. Kubler, and R. von der Heydt. Detection of general edges and keypoints. In *ECCV*, 1992.

[17] E. Saund. Labeling of curvilinear structure across scales by token grouping. In *CVPR*, 1992.

[18] A. Tannenbaum. Three snippets about curve evolution, 1995.

[19] S. W. Zucker, C. David, A. Dobbins, and L. Iverson. The organization of curve detecion: Coarse tangent fields and fine spline coverings. In *ICCV*, 1988.

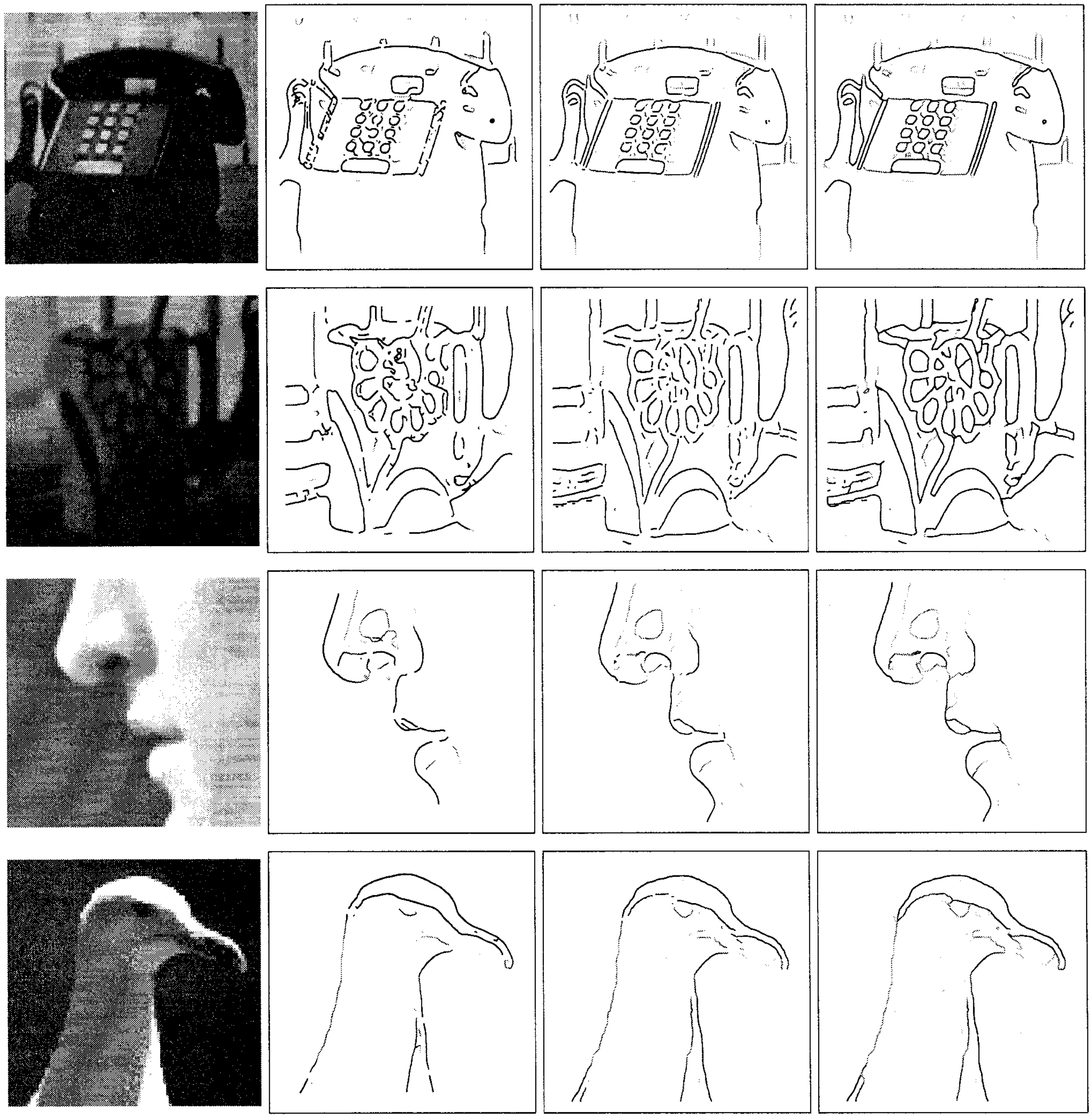


Figure 8: Experimental results. The second column is the result of Canny's edge detector followed by greedy edge linking. The third column is the set of regular curves obtained by the proposed algorithm (the specific implementation of this part of the algorithm is the one described in [2]). Finally, the last column shows the result after curve singularities have been detected.