Pixel-Level Snakes for the CNN Universal Machine: On Chip Implementation

David L. Vilariño and Csaba Rekeczky
Analogical and Neural Computing System Laboratory
Computer and Automation Institute
Hungarian Academy of Sciences,
Kende u. 13-17, H-1111 Budapest, Hungary.

Abstract

In this paper, a new algorithm for the cellular active contour technique called pixel-level snakes is proposed. The motivation is twofold: On the one hand a higher efficiency and flexibility in the contour evolution towards the boundaries of interest is pursued. On the other hand a higher performance and suitability for its hardware implementation onto a CNN chip-set architecture is also required. To this end the algorithm proposed in [9] is completely revised and its limitations are discussed. Based on this analysis the contour evolution is improved and a new approach to manage the topological transformations is incorporated. Furthermore new capabilities in the contour guiding are introduced by the incorporation of inflating/deflating terms based on the balloon forces for the parametric active contours. The entire algorithm has been implemented on a CNN-UM chip set architecture (ACE4K [10], ACE-BOX [1]) for which the results of the time performance measurements are also given. To illustrate the validity and efficiency of the new scheme several examples are discussed including real applications from medical imaging.

keywords: Cellular Neural Networks, CNN Universal Machine, Analogic Algorithms, Cellular Active Contours, Deformable Models.

---

* On leave from the Department of Electronics and Computer Science. University of Santiago de Compostela. E-15782 Santiago de Compostela, Spain.

Contents

1 Introduction

Since they were introduced by Kass *et al.* [8] active contours have become a popular tool in multiple image processing tasks like segmentation, tracking and modelling. An active contour is defined as an elastic curve which deforms controlled by image features and shape constraints to adapt itself to the boundaries of the objects of interest. The active contour techniques are usually classified as either energy-based (parametric models, [8]) or level-set based (implicit models, [4], [11]). The first, also called snakes, are physically motivated and represent the contours explicitly as parameterized curves in a Lagrangian formulation. The second are based on the theory of curve evolution and implement the curves implicitly as a level set of a higher order function which evolves according to an Eulerian formulation. This classification obeys to the contour representation, implementation and capabilities from the operation point of view (particularly, the capability to manage changes in the contour topology). Nevertheless almost all of them have in common high computational requirements which might limit their use in those tasks needing fast time response. This inconvenience is alleviated by the development of new strategies for the numerical simulation of the equations which govern the dynamic of the curve evolution [2, 20, 25, 3, 15] which usually lead towards a compromise between processing speed and flexibility in the contour evolution.

As a difference with the commented strategies the cellular active contours (CAC) appear originally intended to resolve the high computational cost inherent to the classical active contour techniques. They are based on a pixel-level discretization of the contours and on a massively parallel computation on every contour cell which lead to a high speed processing without penalizing the efficiency of the contour location.

Up to the present two different cellular active contour approaches have been proposed. In [17] an active wave computing approach is introduced. This consists of a topographic non-iterative region propagation technique where the contours are defined by the boundaries of trigger waves. Therefore, as in the level-set approaches, the contour evolution is implicitly represented as a wavefront propagation. This approach has demonstrated a high flexibility in the contour evolution and like the implicit models gives a simple solution to the changes of topology required when two different wavefronts collide. Nevertheless in this kind of techniques sophisticated stop criteria are usually required to conveniently control the wave-front propagation which may increase considerably the computation complexity in real applications.

In [9] the called pixel-level snakes (PLS) are addressed. They represent a topographic iterative active contour technique where the contours are explic-

itly represented and evolve towards (local) minimal distance curves based on a metric defined as a function of the features of interest. In the PLS method contours are guided by local information and regularizing terms dependent on the contour curvature.

Since PLS were introduced in [21], the associated algorithm has undergone different improvements and new capabilities were incorporated in several steps. In [22], a new guiding term derived from the contour itself was embodied allowing to keep smooth the contour shape and providing a higher robustness against noise. In [9], new modules were included allowing to handle the topological transformations usually required when multiple active contours are evolving simultaneously.

Keeping in mind the characteristics of the commented CAC techniques, we propose an improved algorithm for PLS which combines the contour evolution of the previous PLS schemes with the region propagation addressed in [17]. The result is a novel approach which performs a better contour evolution and a more efficient management of the topological transformations.

All the steps of the proposed algorithm consist of simple local dynamic convolutions and morphological hit and miss operations together with simple arithmetic and logical operations. Therefore the proposed system meets the requirements to be implemented on a CNN chip-set architecture based on the CNNUM concept [19]. The algorithm has been already tested on the 64x64 CNNUM chip (ACE4K, [6, 10]) within the ACE-BOX computational infrastructure [1]. All the examples used to illustrate the capabilities of the algorithm are experimental results from the on-chip implementation.

The remainder of the paper is organized as follows: In Section 2, a brief revision of the PLS strategy and the main operations of the algorithm are addressed. In Section 3 the new algorithm is introduced showing the contributions and comparing the efficiency of the proposed algorithm with the previous one. In Section 4 the projection on the CNNUM is described. In Section 5 some real-life examples of application are discussed and finally, the main conclusions are drawn in Section 6.

2 Pixel-Level Snakes

In the context of PLS, the active contours are represented as sets of 8-connected activated pixels in a binary image called contour image. This has the same dimensions as the image containing the objects or regions to be defined. The contour evolution consists on an iterative process of activation and deactivation of the contour image pixels along the four cardinal directions. These operations are driven by external information extracted

from the image under processing and internal information derived from the contours themselves. The result is equivalent to contour shifts towards final shapes and locations according to the requirements of the guiding information.

The guiding information can be interpreted as scalar potential fields extended to all pixels of the image under processing. This includes:

- The external potential, that takes lower values near the edges. This will guide the contours to the boundaries of the objects into the image.
- The internal potential, extracted from the contour image. This will try to keep smooth the contour shape.

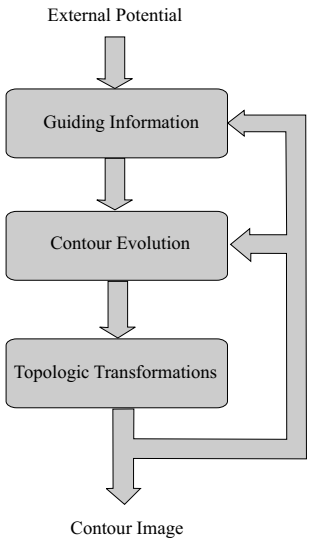


Fig. 1: Diagram showing the main modules and their interactions for a PLS algorithm.

Conceptually the PLS consist of three different modules which interact dynamically (Fig. 1):

1. A module responsible to extract the information to guide the contour evolution. This includes the derivation of the internal potential from the contour image and the combination with the external potential from the image under processing.
2. A module dedicated to the contour evolution. This consists of an iterative operation of pixel-to-pixel shift of the contours driven by the guiding information.

3. A module undertaken to handle the possible collision and the required topologic transformations between contours.

In the following, the main operations to be performed by the modules in Fig. 1 are introduced. A comprehensive description can be found in [23].

2.1 Guiding information

The components of the so-called guiding forces along the direction under processing are derived from the external and internal potential matrices by simple directional gradient operations. A positive force should indicate a valid direction for the contour evolution. However, in a pixel-level iterative technique only the sign of the component of the guiding forces along the direction under exploration is actually needed. Therefore, in this stage a thresholding operation is also included. These operations are gathered into the so-called guiding force extraction module (GFE). In short, the output of this GFE module will represent a binary map with activated pixels in those locations where the potential is decreasing along the direction under study (thus the contour evolution is allowed towards those directions). Fig. 2 illustrates the operations in the GFE module by means of an example of contour evolution based on external potential.

The external potential should be defined in such a way that the boundaries of interest coincide with the valleys of the potential field. This is strongly dependent on the particular application and represents an external input to the PLS algorithm. On the other hand the internal potential is derived directly from the active contours. The internal potential estimation (IPE) consists of a recursive low-pass filtering or diffusion operation acting on the contour image. The result is a real-valued array characterized by lower potential values at the cavities of the contour and higher outside [22, 23]. Therefore a directional gradient operation acting on this array will originate positive internal forces which push to reduce the local curvature and therefore to smooth the contour shape [22]. This idea is illustrated in Fig. 3, where one contour is guided by only this kind of internal information. It is well known that a planar closed contour whose evolution relies only on the local curvature will adopt a circular shape and finally will collapse. This behavior is observed with the proposed internal potential estimation which demonstrates the curvature dependence of the approach.

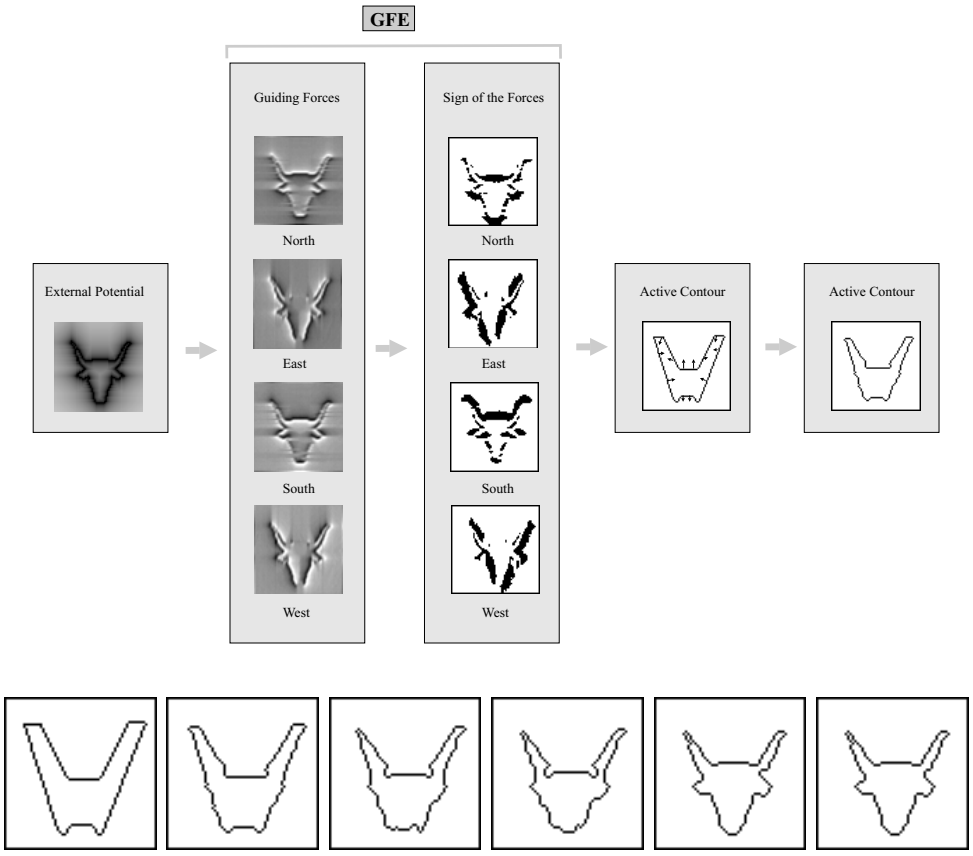


Fig. 2: (Upside): Guiding force extraction (GFE) from an external potential field. Lower potential is represented by lower intensity. By means of directional gradients the component of the guiding forces for each direction is obtained. The sign of these forces will indicate the correct direction to move the active contour. (Downside): Several snapshots of the contour evolution guided by only the external information.

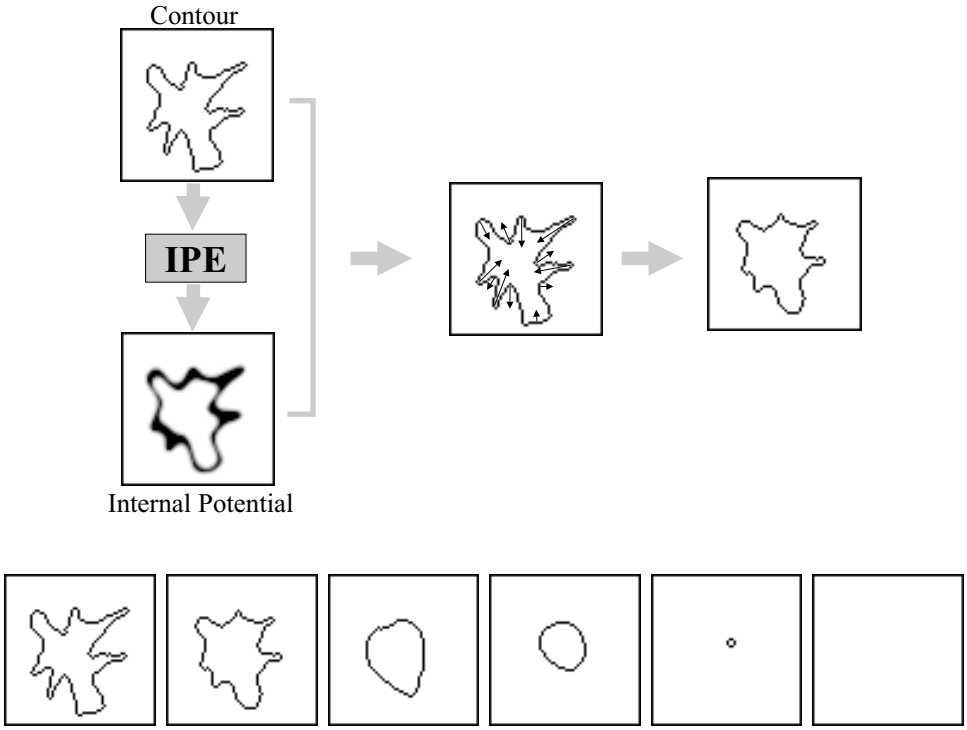


Fig. 3: (Upside): Generation of the internal potential for the PLS. Lower potential is represented by lower intensity. A directional gradient operation will originate forces proportional to the local curvature which guide the contour evolution in order to regularize its shape. (Downside): Several snapshots of the contour evolution guided by only internal information.

Therefore, the combination of both external and internal potential would provide more robustness to the contour evolution against noise, as it is illustrated in Fig. 4. There is not an exact rule to determine both the number of diffusion steps and the influence (weight) of this kind of potential respect to the external potential. Like in classical active contour techniques they must be determined heuristically. However, a great precision in the internal potential estimation is actually not required. The main objective of the internal potential use is to reach a better behavior in the processing of noised information. If the internal potential is weakly weighed with regards the external potential, anchored pixels can appear in the locations perturbed by noise. However this situation leads to a sharp contour shapes and, as a consequence, to the increasing of the internal potential which eventually can overcome the noised external potential.

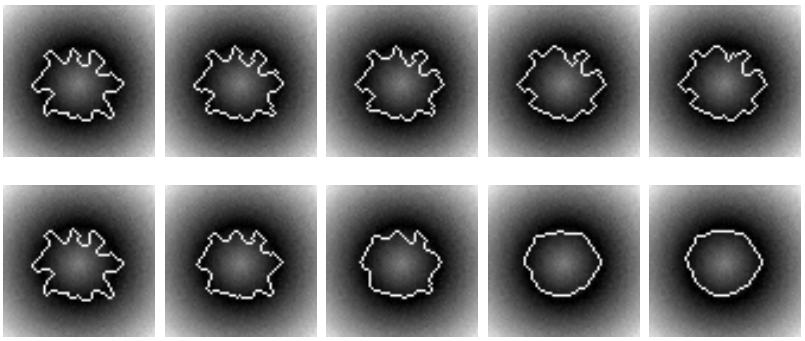


Fig. 4: (Upside): Contour evolution based on noised external potential (additive Gaussian noise $\sigma = 100$, into the usual range for gray-scale images, [0, 255]). (Downside): Contour evolution based on noised external potential combined with internal potential.

2.2 Contour evolution

To produce the desired contour evolution a directional contour expansion (DCE) of the active contours followed by a directional contour thinning (DCT) are carried out along the direction under processing. The process is driven by the GFE output in such a way that the contour evolution can only affect those locations which coincide with activated pixels in the GFE output. In summary, the goal after each cycle (four iterations, one for each cardinal direction) is to obtain new well-defined contours slightly shifted and/or deformed based on the guiding information from *GFE* in order to come closer and fit themselves to the boundaries which define the regions of interest.

2.3 Topologic transformations

When the number of active contours does not coincide with the number of objects into the scene the collision between different contours (or different parts of the same contour) may occur. Due to the characteristics of evolution and the nonparametric nature of the pixel-level snakes the required changes of topology can be suitably approached by simple inspections of the contour map. This capability notably increases the set of tasks where the PLS can be applied. In [9] a solution to manage the changes of topology supported by local CNN-operations was proposed. The approach is based on preventing the collisions between contours and a controlled split of these contours followed by merging of the new ones before the collision. In the following Section, this approach is analyzed and its limitations are discussed. Then, a more efficient strategy to approach the changes of topology is proposed.

3 Improved PLS algorithm

The algorithm of PLS addressed in [23] has demonstrated a good performance in multiple applications where active contour techniques are frequently used. Nevertheless, in its initial form, it has some drawbacks and exceptions which reduce the efficiency of the algorithm. In this Section, these limitations and exceptions are analyzed and illustrated by examples. Then several changes are proposed which result in a higher performance of the PLS. The modifications affect the three modules described in the previous Section. In the sequel the contributions are discussed and compared to the previous structure.

3.1 Guiding information

Numerous refinements have been proposed in the literature to improve the robustness and stability of the active contour techniques. Working towards this direction in [5] a *balloon force* has been introduced to the parametric snake model. This new term comes from represent an anisotropic pressure potential that controls the evolution of the contours helping them to trespass spurious isolated weak image edges and counteracts their tendency to shrink (due to the internal forces). The snake becomes more robust with respect to the initial curve position and the image noise. However, in a practical situation, it has to be decided whether an inflationary or deflationary force is required. Both implicit and pixel-level active contours can easily incorporate this kind of guiding terms. The Implicit models implement them as geometric-independent advection terms [14]. The pixel-level snakes can effectively inflate (deflate) the contours by adding higher (lower) potential terms to those locations inside the closed curves with respect to those situated outside. The process is mainly supported by means of a weighted hole filling operation as it is illustrated in Fig. 5. The sign of the weight constant will determine the inflating or deflating nature of the potential. Note that to define correctly the balloon forces the contour locations must be weighted with the higher potential.

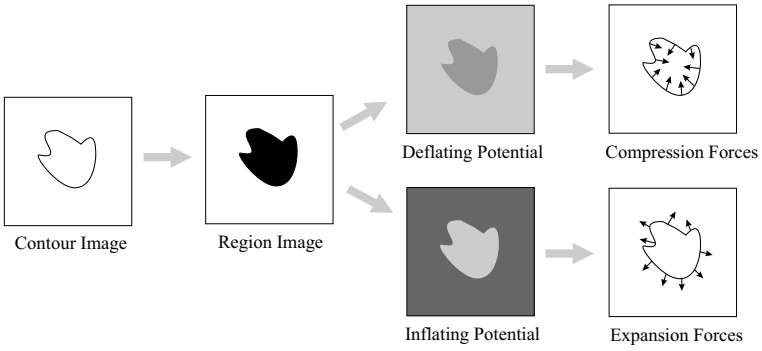


Fig. 5: Generation of inflating/deflating potentials for the PLS. Lower potential is represented by lower intensity. A directional gradient operation will originate forces guiding the contour evolution outwards (inflation) or inwards (deflation).

Therefore this kind of potential term can help to put the active contour closer to the boundaries of interest from locations where the external forces are too weak (Fig. 6).

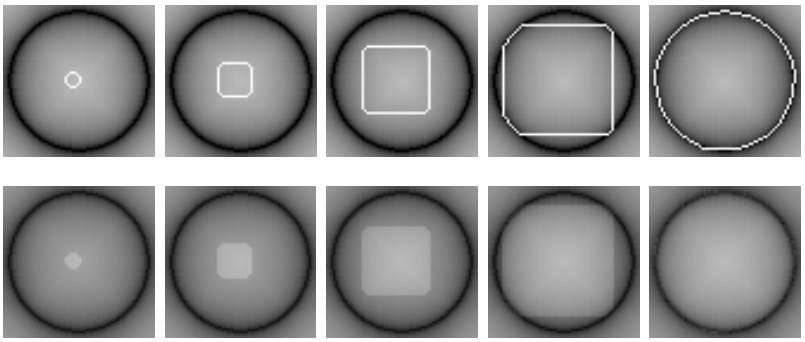


Fig. 6: (Upside): Several snapshots of the evolution of an active contour superposed to the external potential field. (Downside): Combination of the external potential and inflating potential. While the external forces are too weak (far from the boundaries of interest) the inflating forces dominate the contour evolution. When the active contour comes closer to the object, the external forces become strong enough to control the evolution.

3.2 Contour evolution

As it has been commented in the previous Section, the contour evolution is based on iterative operations of expansion and thinning extended along the four cardinal directions conducted by the binary map from GFE. Given a direction of processing, a directional contour expansion (DCE) is carried out only in those locations where the local potential is decreasing (i.e. associated with positive guiding forces). The subsequent directional contour thinning (DCT) operates on the DCE binary output deactivating those pixels situated in locations of locally decreasing potential which do not entail a rupture of the contour connectivity. The combination of both operations produces the contour evolution towards decreasing potential. However this is somehow restricted mainly for two reasons (Fig. 7):

1. The expansion and the thinning operations affect different pixels and therefore rely on different guiding information. Note in the example in Fig. 7 that the pixels activated by DCE are different to those deactivated by DCT.

2. The DCE operation is constrained to only contour duplications: Only the expansion is produced from those locations where the contour is one-pixel wide along the direction of processing. This is required to avoid the generation of ill-defined contours as a consequence of the different control for DCE and DCT.

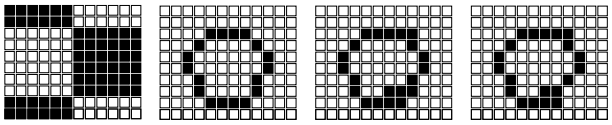


Fig. 7: Example of a contour evolution for the processing of the North direction. (a): Output of the *GFE* module. (b): Contour image before the DCE and the DCT operations. (c): Contour image after the DCE operation. The expansion affects those locations with positive guiding forces (dark pixels in the *GFE* output) whose activation produces only two-pixel wide contours. (d): Contour image after the DCT operation. The thinning only affects those contour pixels with positive guiding forces whose deactivation does not provoke connectivity breakpoints.

This approach leads to an inadequate contour evolution when the contour is anchored in one or more pixels. This ill operation is illustrated in Fig. 8. In the example, a contour evolves towards the North direction as a consequence of some potential field. The contour pixel filled in gray represents an unaccessible location (*e.g.* a noisy pixel). Since the contour expansion can only produce two pixel-wide contours the evolution is deviated with a non-null angle. This deviation is what we call the *scattering effect*. Fig. 13 shows a real experiment for a similar situation of that illustrated in Fig. 8 where the predicted scattering effect appears.

The scattering effect has some important consequences. The presence of noise into the external potential image can lead to the appearance of anchored contour pixels, which produces sharp contour shapes. These should be smoothed by the internal potential which is dependent on the local curvature. Nevertheless, the effect of the internal potential is weakened by the scattering effect which leads to concavities with higher curvature radios. Another important limitation strongly related with the scattering effect, appears in those applications where evolutions along very narrow cavities are involved. The restricted contour expansion impedes to reach deep locations along cavities with less than five pixels of width, as it is illustrated in Fig. 9. The example shows a contour flowing to the North along a four pixel-wide cavity

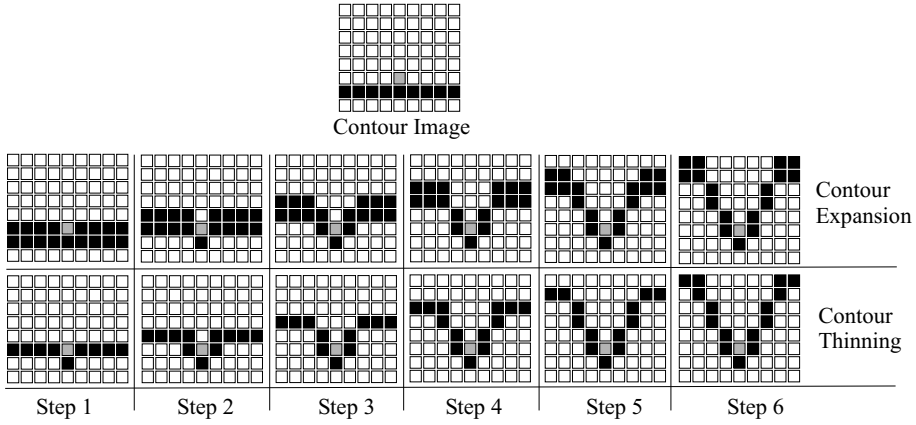


Fig. 8: Illustration of the scattering effect. Black pixels represent contour pixels. The gray pixel represents a location unaccessible for the contour. Therefore the contour is forced to surround that location resulting in a deviation in the trajectory of the contour evolution (scattering).

based on the rules which define the DCE and the DCT operations. In Fig. 14 the results from an equivalent real experiment are showed.

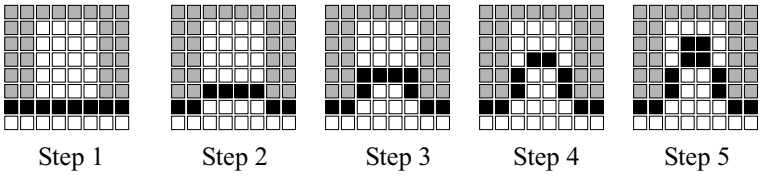


Fig. 9: Illustration of the evolution along narrow cavities. Black pixels represent contour pixels. The gray pixels define the cavity. Based on the rules for the DCE and DCT is not possible the contour evolution along cavities with less than five pixel of width.

As we have seen the scattering effect is a consequence of the restriction in the contour expansion (only two pixel-wide contours can be generated from this operation) and this constraint is a consequence of the different control for the expansion and the thinning steps. We have noted that a constrained thinning operation is actually not required. Since both the expansion and the thinning operation operate consecutively along the same direction the pixels affected for the contour thinning are mainly those pixels which have enforced the contour expansion of the previous step. Therefore the non-constrained thinning operation will lead to the effective contour evolution and makes the

requirement of only contour duplications in the DCE stage unnecessary. In Fig. 10 the operation of the new contour shift approach is illustrated.

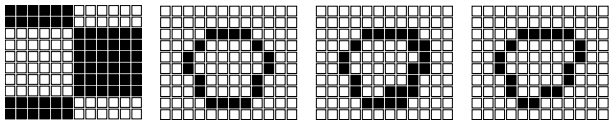


Fig. 10: Example of a contour evolution for the processing of the north direction. (a): Output of the *GFE* module. (b): Contour image before the DCE and the DCT operations. (c): Contour image after the DCE operation. The expansion affects those locations with positive guiding forces (dark pixels in the *GFE* output) neighboring to some contour pixel along the direction under processing. (d): Contour image after the DCT operation. Now the thinning operation is only constrained to keep well-defined the continuity of the contour.

This small but critically important change in the contour evolution relies only on one external constraint (the DCE driving) leading to a more efficient operation. Furthermore, since the contour expansion is not restricted to duplications only, the scattering effect disappears as it is illustrated in Fig. 11. In Fig. 13 an equivalent on-chip experiment is showed.

Therefore the contour evolution is more robust against noise because anchored contour pixels provoke sharper cavities than in the previous version of the algorithm generating a higher internal potential into the cavities. Furthermore, the inhibition of the scattering effect also allows the active contours to evolve along deep and very narrow cavities as it is illustrated in the example in Fig 12. Note that the pixel level snakes cannot flow along cavities with less than three pixels of width. This is a inferior limit to allow well defined contours. In Fig. 14 it is showed how with the new approach the contours can flow along narrower cavities than with the initial formulation.

With the modifications described above not only a higher efficiency in the contour evolution is achieved but also the PLS algorithm is also simplified. Now, the DCE module is supported by only one 3x3 directional template per direction instead of the four operations required for the structure in [9] or the 5x5 directional template in [22].

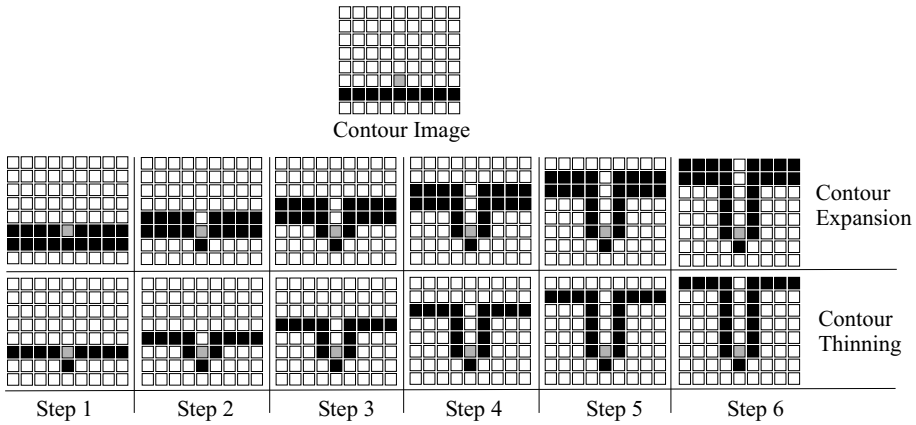


Fig. 11: Illustration of the scattering effect. Black pixels represent contour pixels. The gray pixel represents a location unaccessible for the contour. Therefore the contour is forced to surround that location but now the trajectory of the contour trajectory is not deviated.

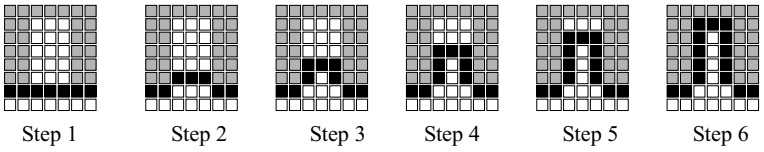


Fig. 12: Illustration of the evolution along narrow cavities. Black pixels represent contour pixels. The gray pixels define the cavity. Based on the new rules for DCE and DCT is possible the contour evolution along cavities with at least three pixel of width.

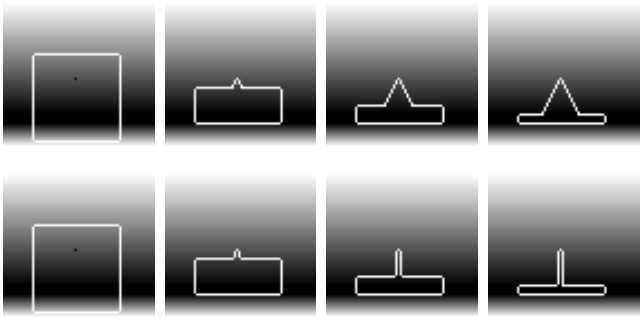


Fig. 13: In this example an active contour evolves towards lower intensities. One pixel in the upside of the contour becomes anchored based on the guiding information which causes the scattering effect from the original PLS algorithm (upside row). This effect disappears in the improved version since the new contour evolution relies on a non-constrained thinning operation and on a more flexible directional contour expansion (downside row).

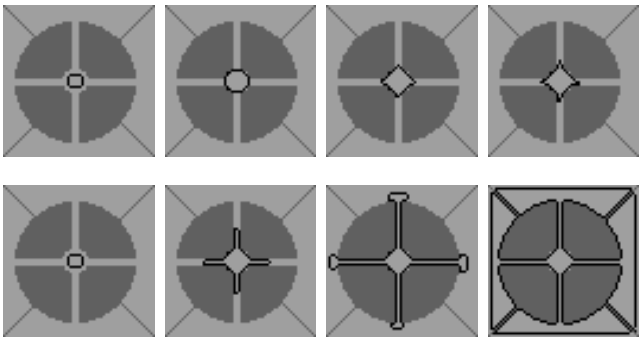


Fig. 14: The scattering effect impedes the contour evolution along very narrow cavities with the original algorithm (upside row). Since the contour expansion is not restricted to only duplications the contour evolution along very narrow cavities becomes more efficient with the new approach (dowside row).

3.3 Topologic transformations

The method to manage topologic transformations addressed in [9] is supported by two main operations: Avoiding all uncontrolled collisions between contours and controlling contour splits and merges during the evolution to provoke the required change of topology. In order to make clear the main shortcomings of this method we now describe the explicit steps of the implementation proposed in previous studies [9, 23].

The first operation to carry out the topologic transformations should consist of avoiding the possible collision between contours relying on a preestimation of the locations where a collision could occur. This action is relatively easy to implement because the contours move as the effect of activation and deactivation of pixels in the contour image. Thus, the contours evolve pixel to pixel, which allows to estimate the contour location and shape in the next iteration. The collision point detection (CPD) is carried out by a simple pattern recognition which takes as input the binary contour image and returns a binary image with white pixels in those locations where a collision between contours can appear in the next iteration. Therefore, by the projection of this binary map onto the output of the *GFE* module, the pixel activation can be avoided on those conflictive locations and consequently the contour collision will be prevented. In a physical context, the effect of this operation is equivalent to the generation of an infinite potential barrier between contour pieces to avoid the collision.

Now it is possible to take advantage of these *collision points* to realize a controlled split of the old contours and merging of the new ones. The operations to be implemented in order to approach the topologic transformations are illustrated with an example in Fig. 15 and are as follows:

1. The set of collision points which can guarantee a correct contour separation by only local operations are selected.

2. The split of the old contours is carried out by deactivating the neighboring pixels in the direction under processing (vertical direction in the example), with respect to those collision points selected in the previous step.

3. The generation of the new contours are made by activating the neighboring pixels in the direction under processing (horizontal direction in the example), with respect to the collision points selected in the previous step.

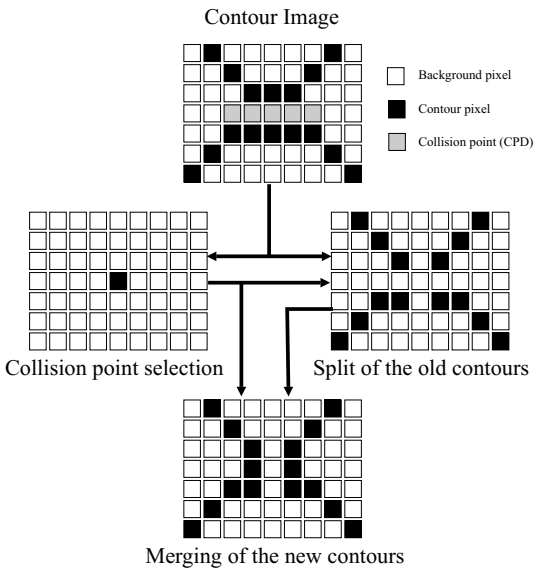


Fig. 15: Example illustrating the operation in the management of topological transformations proposed in [9] (North direction). First, those collision pixels which can support the topological transformation by interacting only with their first order neighborhood are selected. Following the split of the old contours and the merge of the new ones are carried out.

This strategy to tackle the topological transformations has demonstrated a good performance in multiple applications [23]. Nevertheless, as it can be seen in the example of Fig. 15, not all the collision points can guarantee either the correct split of the old contours or the correct merge of the new ones based on the described operations. Therefore some required topological transformations might not be attended. This is illustrated in Fig. 17 with an example where an active contour evolves to define five different objects. In this example changes of topology are required, however they are not attended because the proper definition of the new desired active contours cannot be guaranteed with the described local operations supported by any of the predicted collision points.

On the other hand, the changes of topology are based on *possible* collision points instead of real collisions between contours which have been previously prevented. Therefore there might be certain situations where topological transformations occur even though they are not actually required as it is illustrated in the example in Fig. 18. This ill-function can lead to a bad operation when a contour evolves along very narrow cavities and two pieces of a contour come very close to each other. If the separation between them is only one pixel wide, an undesired topologic transformation can provoke the

stop of the contour evolution as it is illustrated in the example of Fig. 19.

The commented strategy represents a compromise between complexity and accuracy in the management of topological transformations into the contour context where a considerable effort is required to keep well defined the resulting contours. This requirement is dramatically relaxed into a region propagation framework where the contours are defined as frontier pixels of regions into the image space [17]. When two contours (or two parts of one contour) collide the collision points no longer belong to the set of frontier pixels of the associated regions and consequently they do not belong to the set of contour pixels. We propose to handle the changes of topology within the region propagation context by means of the three operations indicated in Fig. 16. The contours are transformed into regions by means of a hole filling operation followed by a one-step morphological opening (erosion+dilation). In the last step the region contours are obtained by a binary edge detection which extracts the set of frontier pixels of the regions.

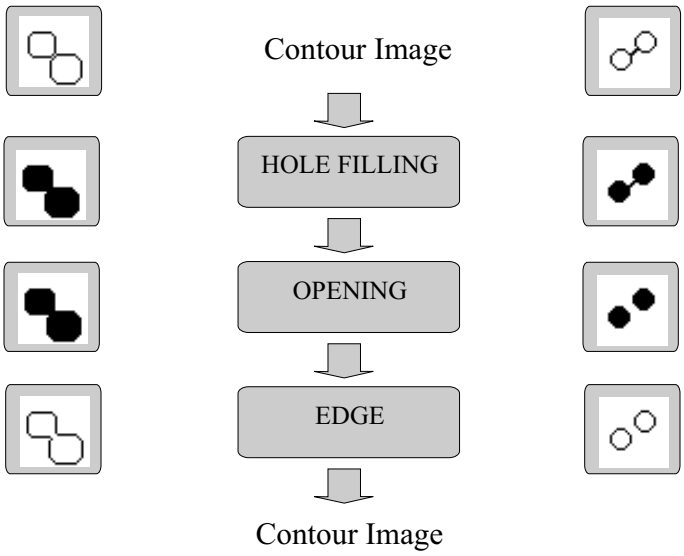


Fig. 16: Flow diagram of the operations for the new topologic transformations module.

This strategy to manage the topologic transformations has a clearly higher performance than that reported in [9] and handles properly the changes of topology *whenever* they are required as it is illustrated in the example of the Fig. 17.

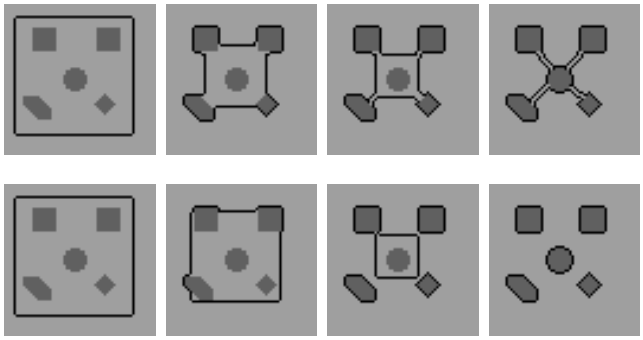


Fig. 17: (Upside row): In this example, the required topological transformations cannot be approached by only local operations based on the proposal in [9]. (Downside row): The new proposal enforces changes of topology whenever they are required.

On the other hand, since now the handling of the topologic transformation is supported by real collision between contours, the changes of topology are attended *only* when they are actually required (Fig. 18).

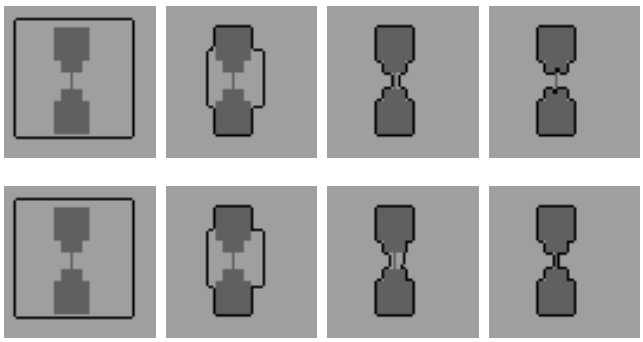


Fig. 18: (Upside row): In this example, two parts of a contour come to each other so close that several pixels between them are marked as collision points. Therefore with the original PLS algorithm a change of topology appears even though it is not required. (Downside row): With the new proposal only changes of topology occur when real collisions appear.

Therefore the PLS methodology with the new strategy to move the active contours allow to evolve the contours along very narrow cavities as it is illustrated in Fig. 19.

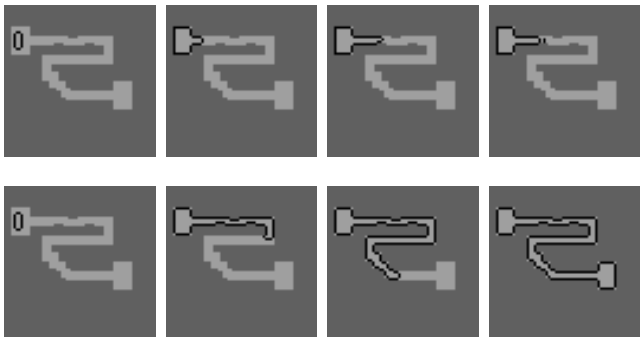


Fig. 19: (Upside row): In this example, two parts of a contour come to each other so close that a change of topology provokes a bad operation. (Downside row): With the new approach no changes of topology appear and the contour can evolve along the cavity.

The new proposal outperforms the contour-based approach not only concerning the efficiency but also in the simplicity of the implementation: Only three isotropic 3x3 linear templates are required instead of the 18 directional templates reported in [9]. Nevertheless this new approach presents an exception: those particular cases where one or several contours appear to be completely surrounded by another active contour cannot be directly managed with the proposed strategy because the cross along the region context would eliminate the internal active contours. On the other hand this behavior can also provide a more robust contour evolution by the absorbtion of artifacts in outwards evolutions in some practical situations as it is illustrated in Fig. 20.

4 Implementation

A flow diagram containing all the operations of the new algorithm for PLS proposed in this paper is showed in Fig. 21. For the CNN operations the initial state and the external input are labelled with A and B respectively. The result of those CNN operations where the A or B labels are missing is independent of the initial state or the input. The only exception is the Hole Filling operation where a +1 initial state should be imposed for all the cells [12]. The collision point detection module (CPD) responsible to prevent

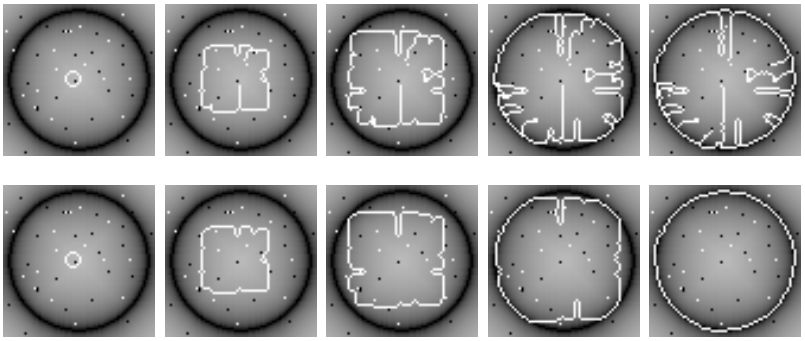


Fig. 20: Contour evolution based on inflating potential and external potential corrupted by noise (binary salt and pepper). (Upside row): Without topologic transformation capabilities the contour is anchored in the corrupted locations. (Downside row): The changes of topology based on the new module permit to leap these locations when they are surrounded.

the collision between contours is no longer required to manage the topologic transformations thus its implementation could be unnecessary. Nevertheless it has been included into the general structure of the algorithm to attend those possible applications where the contour topology should be preserved [23, 7]. Note that the result of this operation could also be used to control the operation of the module responsible for the topological transformations: only changes of topology should be checked when collision points are predicted with the CPD operation.

The dark gray items in Fig. 21 represent the external data provided by the user. They include:

- Input images: The external potential image (gray-level image) and the initial contour image (binary image).
- Weights: k_{ext}, k_{int} and k_{inf} weigh the influence in the contour evolution of the external potential, internal potential and inflating (deflating) potential respectively.
- Switches: s_{inf} selects between inflating (+1) and deflating (-1) potential in the balloon potential estimation module (BPE). s_{cpd} activates (-1) or inhibits (+1) the operation of the collision point detection.

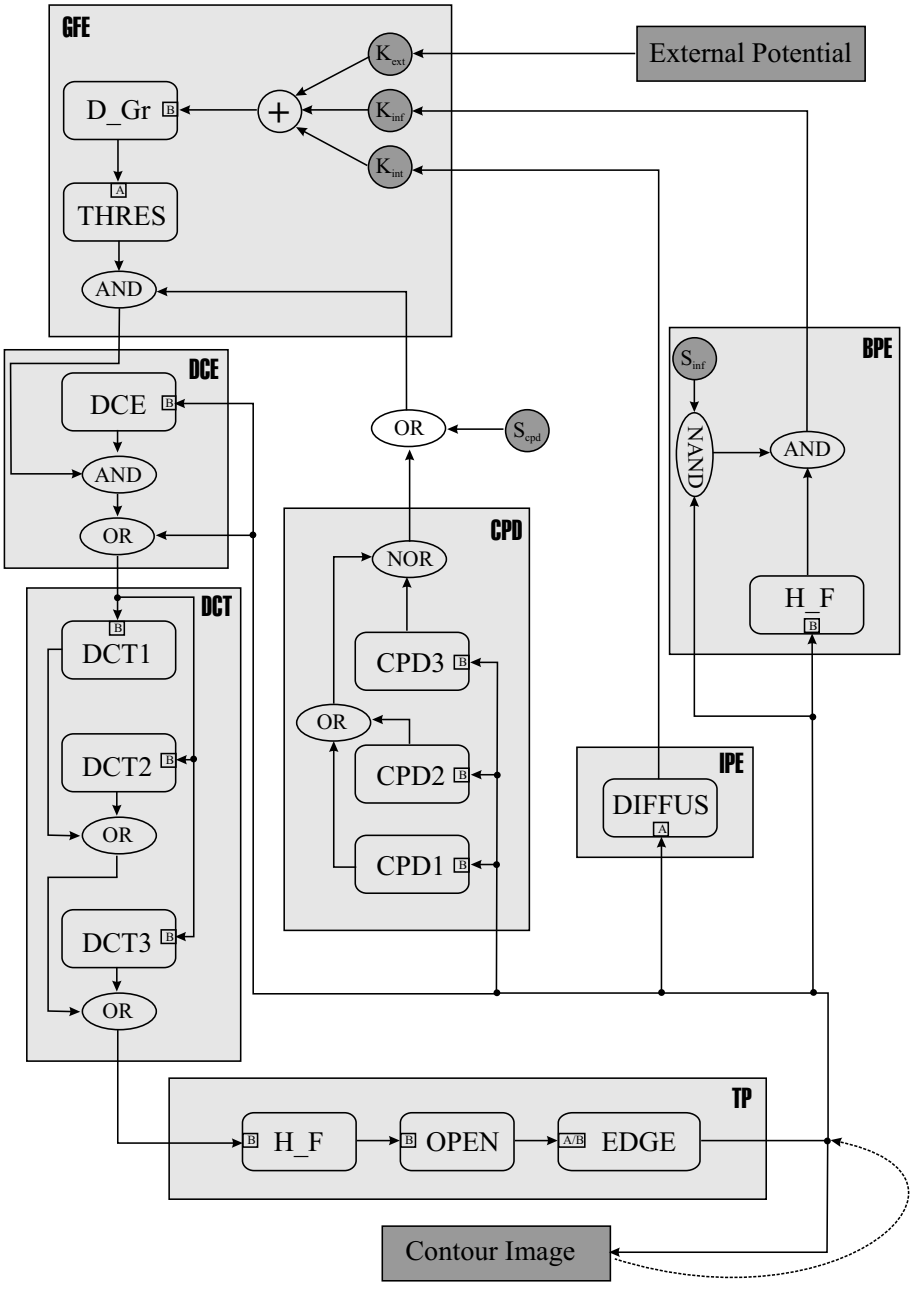


Fig. 21: Flow diagram of the new PLS algorithm containing all the implemented CNN operations (ACE4K [10], ACE-BOX [1]). The active contour image is externally provided for the first iteration.

The external potential field must be defined to have the potential valleys in the boundary of the objects of interest. This definition strongly depends on each application, and usually generates several false positives (due to noise or another non-interest objects) and false negatives (parts of the desired object contour go unnoticed). Often, different image features are combined depending on the application domain and the objects to be segmented [3, 24]. However, the best way of integrating different features remains an open problem and the priority (weight) of image features is still unclear. Another key point is the definition of the initial contours. Since the contour evolution is based on the flow through decreasing potential fields which are estimated based on local rules, the efficiency of the algorithm decreases with the distance from the potential valleys. There is not a general rule to impose the initial contours. It depends on the application and the previous knowledge of the objects to be defined. The use of the balloon potentials can help to make the operation less sensitive with the contour initialization. However, even so, some previous knowledge about the objects to be delimited are required in order to put the initial contour inside (when inflating potentials are used) or outside (when deflating potential are used) the region of interest.

The configuration of the switches (s_{inf} and s_{cpd}) can be lead to very different behaviors for the same input images (external potential and initial contour) which allows to approach more complex tasks based on multiple PLS-operations by changing automatically the configuration of these parameters as it will be showed in the next section.

The templates for the DCE, DCT and CPD operations are derived relying on the local rules showed in Fig. 22. The directional gradient (D_Gr) is performed by an approximation of the Sobel operator. The remainder of the processing steps consists of simple binary logical operations and well-known propagative (thresholding, one-step opening, edge detection) and non-propagative (diffusion, hole filling) analogic CNN operations. In order to make the paper self-contained the templates for all the operations in the algorithm are included in Appendix I.

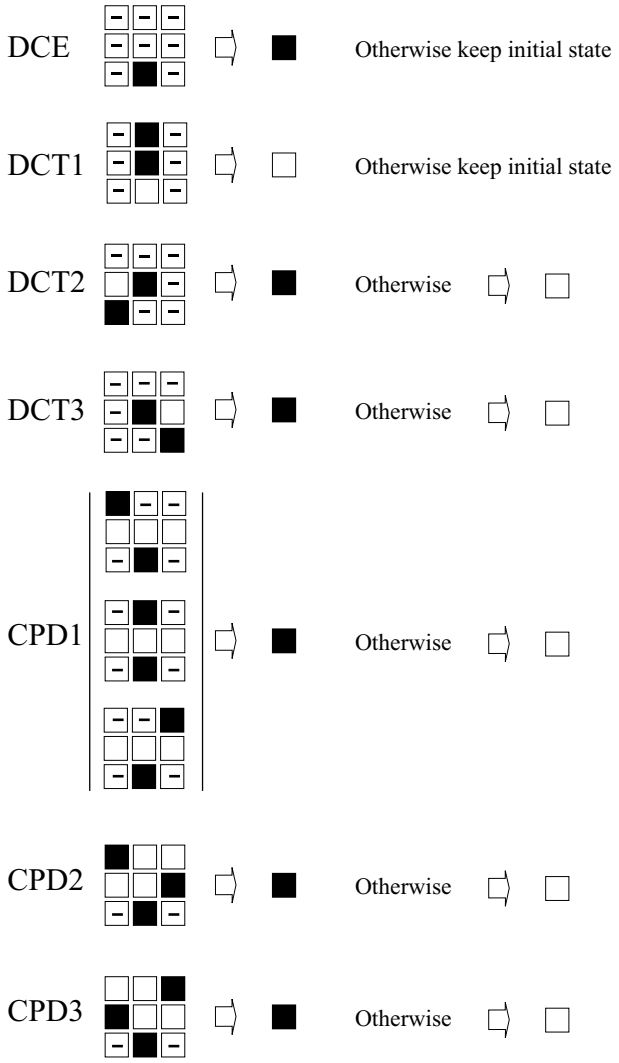


Fig. 22: Patterns and local for the templates of the directional contour expansion and thinning and for the collision point detection along the North direction. For the processing along the other three cardinal directions the resulting templates should be suitably rotated. Note that the *initial state* always makes reference to the previous state of the central cell of the corresponding pattern.

All the steps of the proposed algorithm consist of simple local dynamic convolution and morphological hit and miss operations together with simple arithmetic and logical operations. Therefore the proposed algorithm meets the requirements of the current CNNUM implementations. The algorithm has been implemented and tested on the 64x64 CNNUM, ACE4K ([6, 10]) within the ACE-BOX computational environment [1]. In Table 1, the execution times for each module of the algorithm in Fig. 21 are gathered.

	GFE	**DCE**	**DCT**	**TP**	**BPE**	**IPE**
Time (μs)	250	60	160	760	750	40

Tab. 1: Execution time of all major processing modules in the PLS algorithm extracted from the implementation on the ACE4K (one iteration along a cardinal direction).

Note that the estimation of the internal potential and the inflating/deflating potential (IPE and BPE) are required only once per cycle (one cycle represents four iterations, one for each cardinal direction). Furthermore and as a difference with the original PLS algorithm, the topologic transformations can be checked once per cycle or even less since now the changes of topology are correctly performed in any time after the collision between contours. Therefore the new algorithm with all the functionalities requires less than $4ms$ to complete one cycle running in the 64x64 CNNUM chip. We have observed that in real time applications like video object segmentation and tracking less than ten iterations per frame are usually needed. Therefore, even with the full version of the algorithm the processing of 25 frame/s is feasible. Furthermore in the most of the practical cases not all the functionalities are needed at the same time which can considerably reduce the computational effort.

5 Applications

In order to illustrate the capabilities of the new algorithm in real applications we will show some examples including the processing of binary and gray-scale inputs. First, we will show how the algorithm can be used to find the shortest path in a binary labyrinth where the evolution along very narrow cavities based on inflating and deflating forces is required. Furthermore, the capability to detect and prevent collisions between contours plays an important role in this application. Following we will show examples of applications involving gray-level inputs in medical image processing, a framework where the active contour techniques are most frequently applied [13]. To this end different capabilities of the PLS-algorithm are exploited, including the management of topologic transformations.

5.1 Binary input: shortest path problem

PLS can be applied to resolve the shortest path problem in wide binary labyrinths. The strategy is based on the approach proposed in [16] and later developed in [18] for narrow labyrinths composed by one pixel wide four connected paths. There the labyrinth under study is explored by a travelling wave initiated at the source point. Then, the wave-front evolving on the shortest path will reach the target point without collision. Any waves initiated at a junction located on the shortest path will reach the end of the branch or collide with other wave(s). If the paths are cut at the collision points then all closed loops will be destroyed. Finally, all the branches are pruned in such a way that if there is a unique solution the result will be the shortest path.

In the PLS framework, an active contour surrounding the source point is guided by an inflating potential field through the labyrinth ($s_{inf} = +1$, Fig. 21). The CPD operation prevents the possible collisions when different parts of the active contour meet each other ($s_{cpd} = -1$). Finally, the contour is anchored in the target and the guiding information is switched to a deflating potential ($s_{inf} = -1$). Fig. 23 illustrates the commented strategy with two examples.

Note that at least a three-pixel wide labyrinth is required for the propagation of well-defined active contours. This strategy can guarantee a valid solution if the width of the labyrinth is *4N-1* in all locations where N represents the number of different solutions. Fig. 24 shows an example of a seven-pixels wide labyrinth with two possible solutions. As it can been observed the width of the paths guarantees that at least one (or even two) contours can evolve along them.

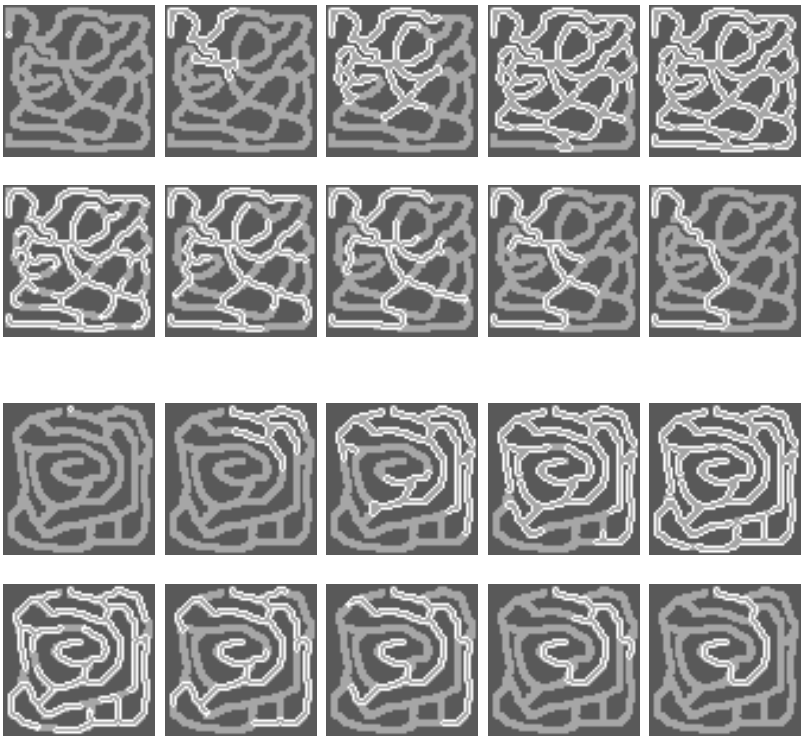


Fig. 23: Contour evolution through binary labyrinths. Initially, the contour evolution is based on an inflating potential field. When the target is reached the guiding information is switched to deflating potential to compress the contour and finally define the shortest path.

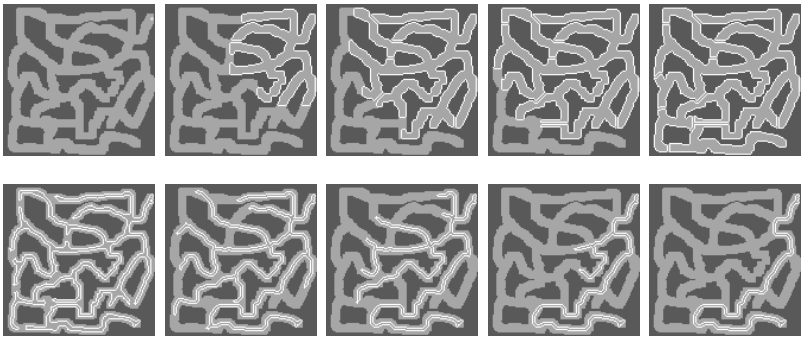


Fig. 24: Contour evolution through a 7-pixels wide labyrinth. In this example there are two different valid solutions. The algorithm guarantees to result in at least one of the shortest paths.

The PLS-based approach of the shortest path problem is particularly advantageous in sparse labyrinths mainly present in applications like robot path planning where different obstacles should be avoided to reach the target. As a difference with the narrow-labyrinth techniques the PLS approach acts directly on the real scenario to find the optimal path. The search can be based on the following metrics:

- City-block distance: $d((x_1, y_1), (x_2, y_2)) = |x_1 - x_2| + |y_1 - y_2|$
- Maximum distance: $d((x_1, y_1), (x_2, y_2)) = max\{|x_1 - x_2|, |y_1 - y_2|\}$

The City-block or Manhattan distance comes as result of estimating the inflating potential (BPE module in Fig. 21) only once each cycle whereas the Maximum distance is a consequence of the inflating potential estimation in every iteration (i.e. along each cardinal direction, four times per cycle). Fig. 25 shows both metrics implemented with the PLS.

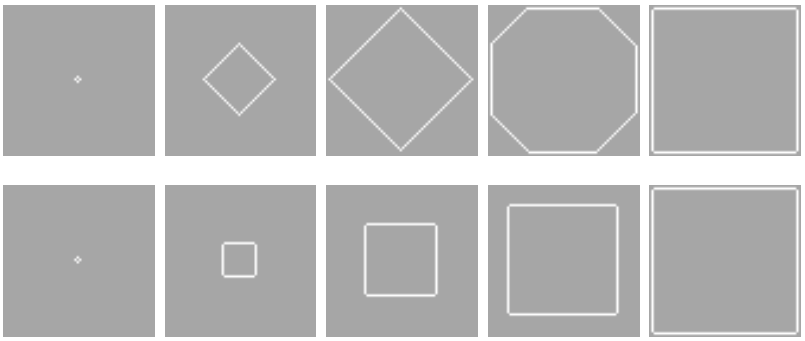


Fig. 25: Several snapshots of a contour evolution based on inflating potential. (Upside row): City-block distance resulting from one BPE per cycle (iter=1, 15, 30, 45, 61). (Downside row): Maximum distance resulting from one BPE per iteration (iter=1, 7, 15, 23, 31).

The choice of the type of distance to be implemented is critical. The shortest path depends strongly on such selection as it is illustrated in Fig. 26.

Note that the shortest path is defined based on the orientation of the obstacles when they are avoided along a trajectory. Therefore the one-pixel wide path surrounded by the final contour does not necessary have a minimal length. Nevertheless this can be approached by a third step of processing where the CPD operation is inhibited ($s_{CPD} = +1$). As a consequence the contour collides with itself giving place to an open contour anchored in both

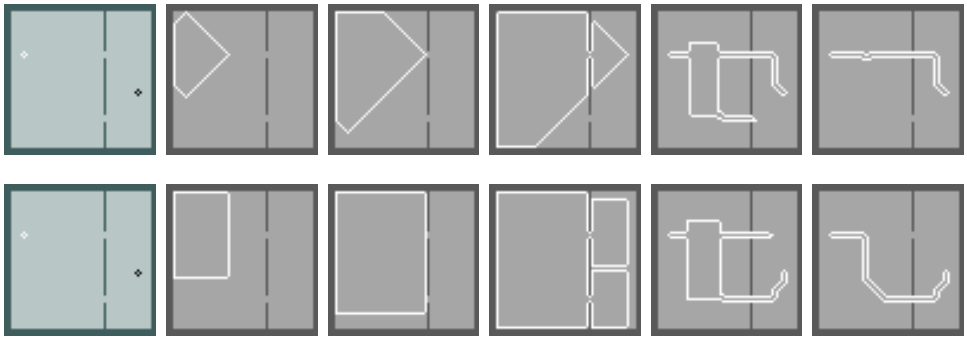


Fig. 26: Several snapshots of a contour evolution to define the shortest path from the start point and the target (surrounded in white and black respectively in the first picture of both sequences). (Upside row): Shortest path based on the City-block distance. (Downside row): Shortest path based on the Maximum distance.

the starting and target points. Now, the action of the internal potential will tighten the contour going closer to the minimal length path. This operation is illustrated in Fig. 27 where quasi-optimal routes are approached from the results in Fig 26.

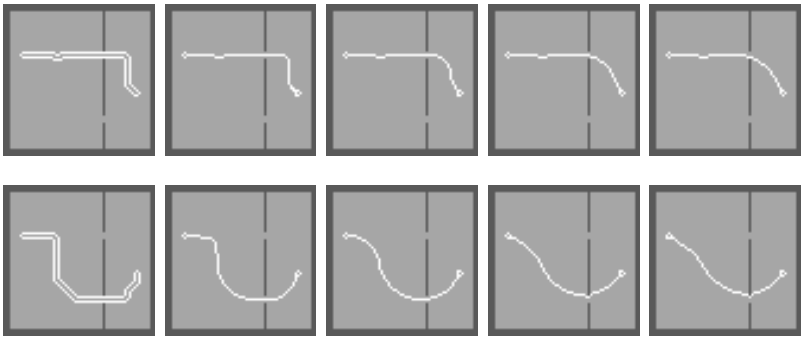


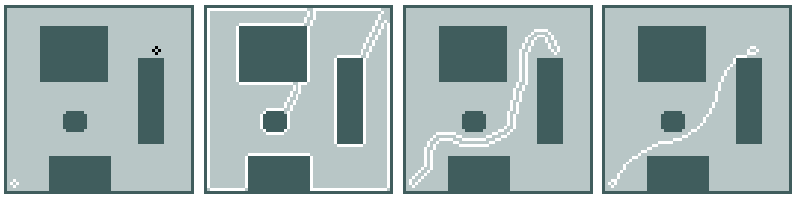
Fig. 27: Several snapshots of a contour evolution towards quasi-optimal paths from the results in Fig 26. The collision of the contour produces an open contour anchored in the ends (start and targetpoints). Therefore the internal potential will smooth the contour constrained by the obstacles into the scene.

In summary in sparse labyrinths a quasi-optimal route can be approached by carrying out consecutively the following processing steps (Fig. 28):

1. Contour evolution based on inflating potential ($s_{inf} = +1$). The collisions should be avoided ($s_{CPD} = +1$).

2. Contour evolution based on deflating potential ($s_{inf} = -1$). The collisions should be avoided ($s_{CPD} = +1$).

3. Contour evolution based on internal potential ($k_{int} > 0$). The collisions should be allowed ($s_{CPD} = -1$).



(a) Start/target **(b)** Step 1 **(c)** Step 2 **(d)** Step 3

Fig. 28: Determination of a quasi-optimal route from a start point (surrounded in white in the first picture) and a given target (surrounded in black in the first picture) by three consecutive PLS-processing based on different switch configuration. (a): Labyrinth with the start point and the target, (b): Exploration step, (c): Route definition step, (d): Path optimization step.

Therefore the complex task of determining a quasi-optimal path in sparse labyrinths (robot path planning) can be approached by three consecutive PLS-operations based on different configurations of the externally accessible parameters. Finally in Fig. 29 quasi-optimal routes are determined from one start point to four different targets. In this case the City-block distance for the first step (exploration) was considered.

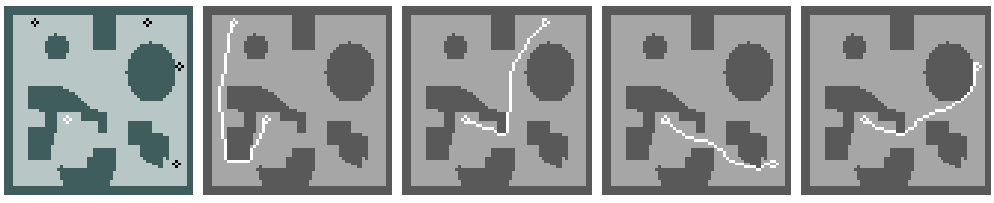


Fig. 29: Shortest paths from a start point (surrounded in white in the first picture) and different targets (surrounded in black in the first picture).

5.2 Gray scale input: medical image processing

In order to illustrate the capabilities of the new algorithm in real applications where gray-level inputs are involved we will show some examples into the framework of medical image processing. First, in Fig. 30 we show several snapshots of the evolution of active contours along blood vessels from a retinal angiography image. In this case the contour evolution is guided by the combination of the intensity image and inflating potential. As it can be seen the operation entails the evolution along narrow cavities and several changes of topology. Note, that in this case the CPD operation should be inhibited.

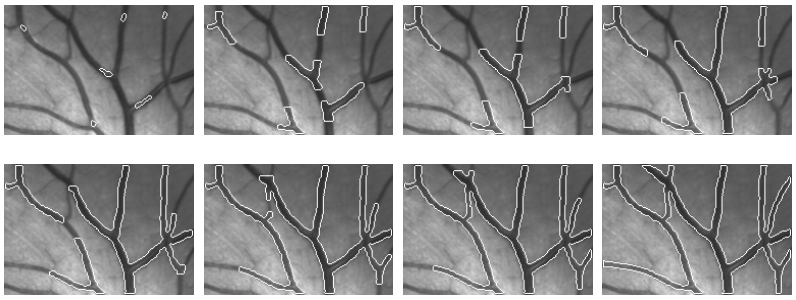


Fig. 30: Example of segmentation of blood vessel in an angiogram of retina from several initial contours. The sequence goes from left to right and top to bottom.

Finally, we have carried out the processing of sequences of ultrasound (US) echocardiography images. The aim of the operation is to define the contour of the left ventricle from the frames as a previous step towards its 3D reconstruction. The contour evolution is guided by a combination of external potential from the US images together with internal potential to keep smooth the contour shape. In order to implement the complete system in the ACE4K, the external potential is derived from the current frame filtered by a local constrained diffusion (see the corresponding template in appendix I) combined with the result of a diffused edge detection onto this image (Fig. 31). The diffused edge detection encourages the contour evolution towards edges and the inverted filtered image reinforces the evolution towards high intensity locations.

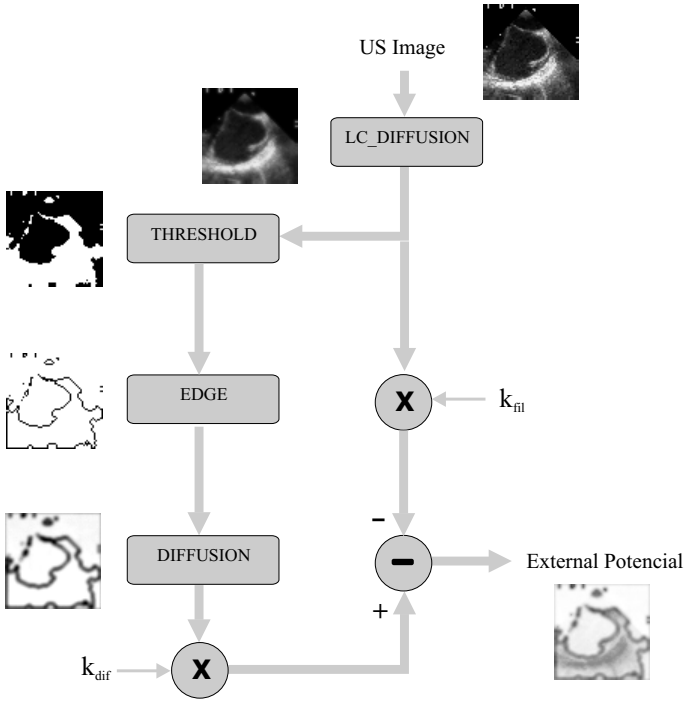


Fig. 31: External potential estimation for the contour location in US images of the human left ventricle. The diffused edge detection generates a potential field with the edges in the valleys (lower real intensity). The substraction of the filtered image encourages the contour evolution to the real high intensity boundaries. k_{fil} and k_{dif} are constants which weigh the filtered image and the diffused edge detection.

Only in the processing of the first frame of each sequence an inflating potential is also considered to put one initial seed situated into the left ventricle close to the boundaries of interest. For the subsequent frames the initial contour coincides with the result from the previously processed frame. In Fig. 32, 33, 34, 35 and 36 several frames of five sequences of processed US images following the commented strategy superimposed to the final contours resulting from the on-chip experiments are showed.

6 Conclusions

Pixel-level snakes represent a cellular active contour technique which has demonstrated a high performance in multiple active contour applications. In this paper, the associated algorithm has been analyzed and some limitations and exceptions have been discussed leading to new proposals to increase the

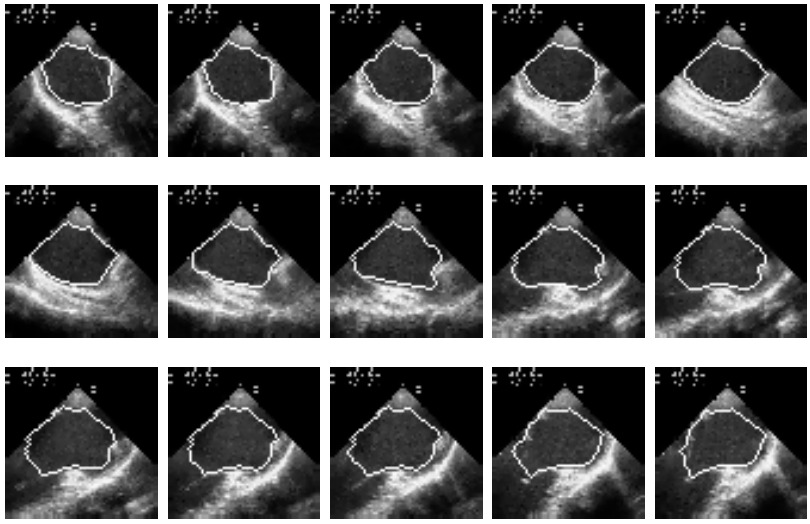


Fig. 32: Contour tracking in ultrasound echocardiography. The sequence goes from left to right and top to bottom.

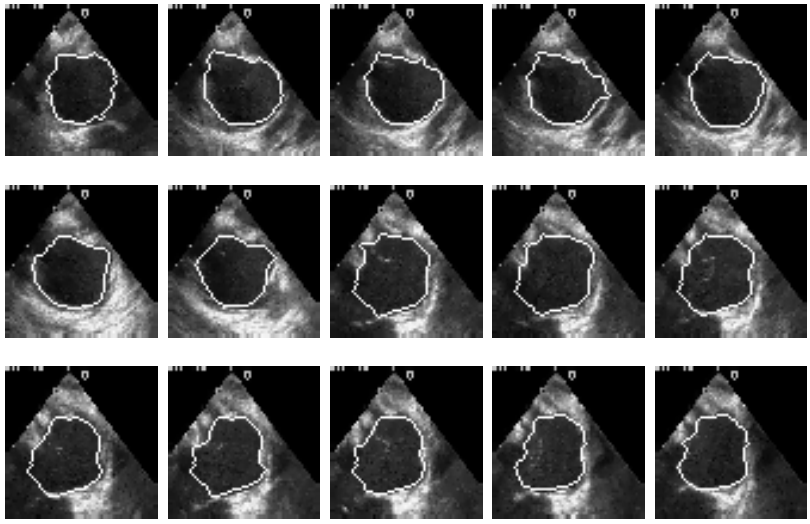


Fig. 33: Contour tracking in ultrasound echocardiography. The sequence goes from left to right and top to bottom.

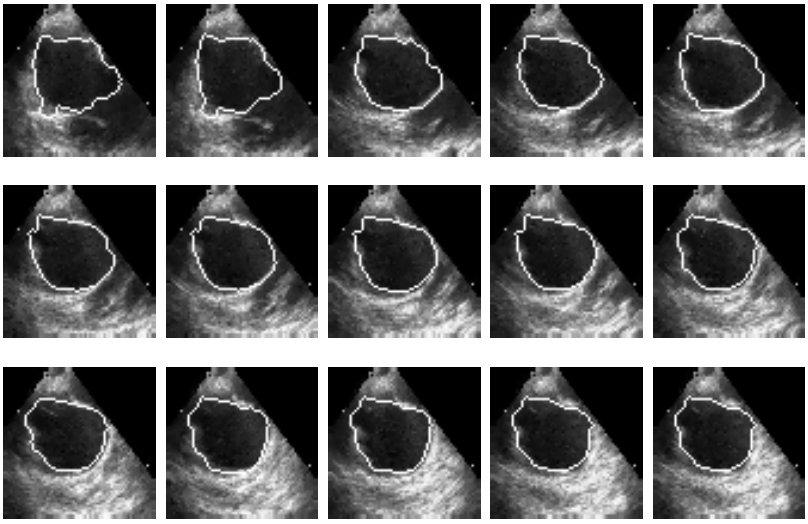


Fig. 34: Contour tracking in ultrasound echocardiography. The sequence goes from left to right and top to bottom.

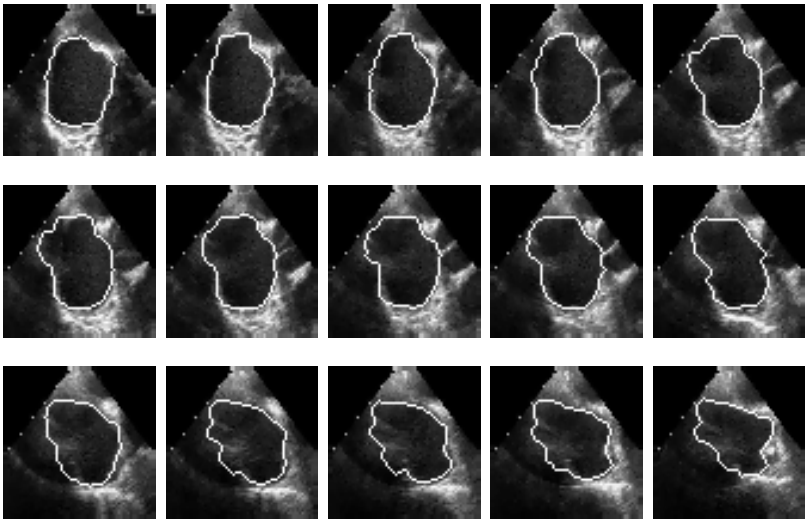


Fig. 35: Contour tracking in ultrasound echocardiography. The sequence goes from left to right and top to bottom.

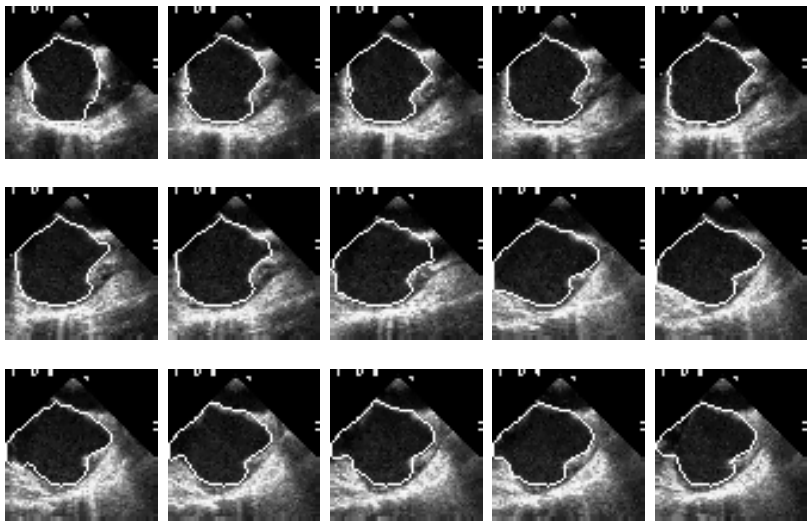


Fig. 36: Contour tracking in ultrasound echocardiography. The sequence goes from left to right and top to bottom.

performance of the PLS. The contributions include a new term to guide the contour evolution as well as a new strategy to manage the topologic transformations. Furthermore, changes have been introduced into the contour evolution module. Altogether these modifications lead to a higher performance and an easier hardware implementation of the algorithm for pixel-level snakes onto a CNN chip set architecture. The algorithm has been already tested on the 64x64 CNNUM, ACE4K [6, 10] within the ACE-BOX computational environment [1]. All the examples used to illustrate the algorithm capabilities represent experimental results from the chip implementation. Furthermore, this implementation has been used to solve a complex task of practical interest, the contour location of US images of the human left ventricle.

Acknowledgment

The research has been completed while the first author was staying at the Computer and Automation Institute of Hungarian Academy of Sciences in Budapest (SZTAKI), in the framework of the EU Centre of Excellence program (EU Supported Visiting Scientist Program, ICAI-CT-2000-70025).

Appendix I. Template derivation

In this Appendix, the templates involved in the PLS algorithm are listed. The templates are expressed in a vector format:

$$\begin{bmatrix} n_0 & n_1 & n_2 \\ n_3 & n_4 & n_5 \\ n_6 & n_7 & n_8 \end{bmatrix} \Rightarrow [n_0, n_1, n_2, n_3, n_4, n_5, n_6, n_7, n_8] \quad (1)$$

Directional templates for the processing along the North direction (for East, West and South directions these templates should be suitability rotated):
D_Gr:
B=0.25*[1,2,1,0,0,0,-1,-2,-1];
DCE:
A=[0,0,0,0,1,0,0,0,0]; B=[0,0,0,0,1,0,0,1,0]
DCT(1):
A=[0,0,0,0,1,0,0,0,0]; B=[0,-1,0,0,2,0,0,1,0]; I=1
DCT(2):
A=[0,0,0,0,1,0,0,0,0]; B=[0,0,0,-1,1,0,1,0,0]; I=-2
DCT(3):
A=[0,0,0,0,1,0,0,0,0]; B=[0,0,0,0,1,-1,0,0,1]; I=1
CPD(1):
A=[0,0,0,0,1,0,0,0,0]; B=0.5*[1,1,1,-3,-3,-3,0,-3,0]; I=-5
CPD(2):
A=[0,0,0,0,1,0,0,0,0]; B=[1,-1,0,-1,-1,1,0,1,0]; I=-5
CPD(3):
A=[0,0,0,0,1,0,0,0,0]; B=[0,-1,1,1,-1,-1,0,1,0]; I=-5

Isotropic templates:
THRES:
A=[0,0,0,0,2,0,0,0,0]
H_F:
A=[0,1,0,1,3,1,0,1,0]; B=[0,0,0,0,4,0,0,0,0]; I=-1
OPEN:
B=[0,1,0,1,1,1,0,1,0]; I={-4,+4}
EDGE:
A=[0,0,0,0,2,0,0,0,0]; B=0.25*[-1,-1,-1,-1,8,-1,-1,-1,-1];
I=-1.5
DIFFUS:
A=0.05*[2,3,2,3,0,3,2,3,2];
LC_DIFFUS:
A=0.05*[2,3,2,3,0,3,2,3,2]; B=0.05*[2,3,2,3,0,3,2,3,2];

References

[1] Analogic Computers LTD: Aladdin Pro R3.0. http://www.analogic-computers.com/, Budapest 2003.

[2] D. Adalsteinsson and J. Sethian. A Fast Level-Set Method for Propagating Interfaces. *Journal of Computational Physics*, 118:269–277, 1995.

[3] A. Blake and M. Isard. *Active Contours*. Springer-Berlag, 1998.

[4] V. Caselles, F. Catte, and F. Dibos. A Geometric Model of Active Contours in Image Processing. *Numer. Math.*, 66, 1993.

[5] L.D. Cohen and I. Cohen. Finite Element Methods for Active Contour Models and Ballons for 2D and 3D Images. *IEEE Trans. Patt. Anal. Machine Intell.*, 15:1131–1147, 1993.

[6] S. Espejo, R. Dominguez Castro, G. Liñan and A. Rodriguez Vazquez. A 64x64 CNN Universal Chip with Analog and Digital I/O. In *Proceedings of 5th International Conference on Electronics, Circuits and Systems, ICECS'98*, pages 203–206, 1998.

[7] X. Han, C. Xu, and J.L. Prince. Topology Preserving Level Set Method for Geometric Deformable Models. *IEEE Trans. Patt. Anal. Machine Intell.*, 25(6):755–768, 2003.

[8] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active Contours Models. *International Journal on Computer Vision*, 1:321–331, 1988.

[9] T. Kozek and D.L. Vilariño. An Active Contour Algorithm for Continuous-Time Cellular Neural Networks. *Journal of VLSI Signal Processing Systems*, 23(2/3):403–414, 1999.

[10] G. Liñan, S. Espejo, R. Dominguez-Castro and A. Rodriguez-Vazquez. ACE4k: An analog I/O 64 x 64 Visual Microprocessor Chip With 7-bit Analog Accuracy. *International Journal Of Circuit Theory and Applications*, 30:89–116, 2002.

[11] R. Malladi, J.A. Sethian, and B.C. Vemuri. Shape Modeling with Front Propagation: A Level Set Approach. *IEEE Trans. Patt. Anal. Mach. Intell.*, 17(2):158–174, 1995.

[12] T. Matsumoto, L.O. Chua, and H. Suzuki. CNN Cloning Template: Connected Component Detector and Hole Filler. *IEEE Trans. on Circuits and Systems*, 37:633–638, 1990.

[13] T. McInerney and D. Terzopoulos. Deformable Models in Medical Images Analysis: A Survey. *Medical Image Analysis Journal*, 1:91–108, 1996.

[14] W.J. Niessen, B.M. ter Haar Romeny, and M.A. Viergever. Geodesic Deformable Models for Medical Image Analysis. *IEEE Transactions on Medical Imaging*, 17(4):634–641, 1998.

[15] N. Paragios and R. Deriche. Geodesic Active Contours and Level Sets for the Detection and Tracking of Moving Objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(3):266–280, 2000.

[16] Cs. Rekeczky, A. Ushida and T. Roska. Analogic CNN Algorithm Exploring the Shortest Path: Possible Applications in Routing Problems. *Technical Report, Institute of Electronics, Information and Communication Engineers NLP '94*, 94(259):61–68, Tokyo, 1994.

[17] Cs. Rekeczky and L.O. Chua. Computing with Front Propagation: Active Contour and Skeleton Models in Continuous-Time CNN. *Journal of VLSI Signal Processing Systems*, 23(2/3):373–402, 1999.

[18] Cs. Rekeczky. Skeletonization and the Shortest Path Problem - Theoretical Investigation and Algorithm for CNN Universal Chips. In *Proceedings of 7th International Symposium on Nonlinear Theory and Applications, NOLTA'99*, 1999.

[19] T. Roska and L.O. Chua. CNN Universal Machine: An Analogic Array Computer. *IEEE Trans. Circuits Syst.*, 40(3):163–173, 1993.

[20] J. Sethian. A Fast Marching Level Set Method for Monotonically Advancing Fronts. In *Proceedings of the National Academy of Sciences*, volume 93, pages 1591–1594, 1996.

[21] D.L. Vilariño, V.M. Brea, D. Cabello and J.M. Pardo. Discrete-Time CNN for Image Segmentation by Active Contours. *Pattern Recognition Letters*, 19(8):721–734, 1998.

[22] D.L. Vilariño, D. Cabello, M. Balsi and V.M Brea. Image Segmentation Based on Active Contours Using Discrete-Time Cellular Neural Networks. In Vedat Tavsanoglu, editor, *Fifth IEEE International Workshop on Cellular Neural Networks and Their Applications*, pages 331–336, 1998.

[23] D.L. Vilariño, D. Cabello, X.M. Pardo and V.M. Brea. Cellular Neural Networks and Active Contours: A Tool for Image Segmentation. *Image and Vision Computing*, 21(2):189–204, 2003.

[24] A. Yuille and P. Hallinan. Deformable Templates. In A. Blake and A. Yuille, editors, *Active Vision*, pages 21–38. MIT Press, 1992.

[25] Y. Zhu and H. Yan. Computerized Tumor Boundary Detection Using a Hopfield Neural Network. *IEEE trans. on Medical Imaging*, 16(1):55–67, 1997.