J Math Imaging Vis 26: 309–325, 2006

DOI: 10.1007/s10851-006-9004-6

A New Active Convex Hull Model for Image Regions

NIKOLAY M. SIRAKOV
Department of Mathematics and Computer Science, Texas A&M University–Commerce, P.O. Box. 3011, Commerce, TX 75429-3011
Nikolay_Sirakov@tamu-commerce.edu

Published online: 14 August 2006

Abstract. This paper presents a new active convex hull model with the following advantages: invariant with respect to the number of pixels to be enveloped; the number of time iterations is invariant, with respect to the image size; time-cheap for large image regions. The model is based on the geometric heat differential equations, derived from parabolic equation, and parameterized by arc length. To prevent the active contour from intruding into concavities and evolve it to the proper convex hull we use a vector field given as a difference between normal and tangent forces. The vector field is also used to segment an image to shells, such that a single region is present in each shell. A penalty function is developed to stop evolvement of those arc segments, whose vectors encountered boundary points of an image region. Based on the model a discrete algorithm is designed and coded by *Mathematica 5.2*. A condition is developed, with respect to the image size, to guarantee stable convergence of the active contour to the convex hull of the desired region. To validate the advantages and contributions a set of experiments is performed using synthetic, groundwater and medical images of different size and modalities. The paper concludes with a discussion and comparison of the active convex hull model with set of existing convex hull algorithms.

Keywords: heat differential equation, normal and tangent forces, active convex hull model, image processing

1. Introduction

This paper presents a new Active Convex Hull Model (ACHM), which is based on the geometric heat differential equation [12, 20, 29]. The convex hull (CH) is a fundamental construction for mathematics [6] and computer science, which has many useful applications in the practice and science. For instance, the CH lies behind the basis of many optimizing methods and algorithms including the simplex method. In [7] the CH is used for analysis of spectrometry data, while in [6] is applied to determine Delaunay triangulation, Voronoi Diagrams and power diagrams. Aurenhammer [3] describes a set of practical problems, that can be reduced to CH: mesh generation, file searching cluster analysis, crystallography, metallurgy, urban planning, cartography, image processing, and numerical integration.

CH methods and algorithms are reported in [4, 17, 26], and more algorithms can be found in the web site of Sunday [31]. Some of the most efficient, with respect to the runtime, CH algorithm is called *Quickhull* and is introduced in [6]. It is a variation of the Clarkson Shor's algorithm [14], and has successful extension and useful applications to dimensions greater then 3. Recently, four space efficient CH algorithms, two *in-place* and two *in-situ*, were published in [8]. Also, *ConvexHull* operators are provided by *Mathematica and Matlab*. The latter is based on the randomized *Quickhull* algorithm.

A survey of the the CH methods shows that their calculation complexity depends on the number of points (image pixels) to be enveloped by the CH. Therefore these algorithms will work relatively slow in processing large image regions. Moreover, image processing is needed to extract the coordinates of the points (image pixels) to be enveloped by the CH.

Unlike the above mentioned methods the proposed ACHM is invariant with respect to the number of points

(pixels) to be enveloped. The calculation complexity of ACHM depends on the number of vectors in the vector field (VF) and the distance to be traveled by them. In other words, speaking in terms of images, the calculation complexity (the run time) of the conventional CH algorithms depends on the size of the foreground whereas ACHM's calculation complexity (the run time) depends on the size of the image background. This feature makes ACHM conceptually new CH algorithm. To develop ACHM the geometric heat differential equation (DE) [12, 13, 19–22, 29, 30] is employed to evolve a closed, smooth curve in the direction of a VF given as a difference between normal and tangent components (forces). The normal component is presented as a derivative of the tangent vector normalized by the image size. The tangent component is a product of the tangent vector and the distance between the ends of the arc segment.

In the past decade applications of DE and variational methods led to a number of significant results in Image Processing [29], filtering, interpolation, segmentation [33, 34], shape smoothing [24], and computer vision [11]. Also, 2D and 3D active contour models are experiencing successful applications to the emerging area of content based image retrieval (CBIR) [2, 10, 34]. The present paper illustrates a new application of the active contours to find the CHs of the image regions subject of user interest. The CHs show the objects' location in the image and are used for search space partitioning of the corresponding image database [25, 27].

Two key difficulties one can face when employing parametric active contours [32]:

- First, the initial contour must be close to the true boundary, or else it will likely converge to wrong result;
- Second, the active contours have difficulties progressing into boundary concavities.

Several methods have been developed to tackle the above problems [1, 15, 16], but they experience a disadvantage of efficiently solving only one of the two problems. Subsequently, in [32] a model is developed that provides an elegant solution to both problems.

The proposed ACHM is capable of quickly enveloping the CH of an image region, providing the following advantages: time-cheap algorithm; a large capture range, which could inscribe the entire image; stability convergence condition; ability to work with gray level images of varying sizes, proportionally increasing the speed of stable convergence (the length of the time step) if the image size increases. Thus, ACHM provides a solution of the first but does not consider the second problem at all. As an input information ACHM requires the gray level difference between foreground and background. The listed features make the ACHM suitable for the applications mentioned earlier in this page [25, 27]. The use of CH for image regions/image indexing leads to significant reduction (by a factor of roughly hundreds) of the number of image database regions necessary to traverse and compare to the query region. This advantage is important for extensive databases because matching the query shape against every image database region could be infeasible even with powerful computers and recognition engines.

Using *Mathematica 5.2* a prototype tool is developed for implementing ACHM and verifying its advantages. A set of experiments was performed using groundwater, medical X-ray, MRI, PET and synthetic gray level images of varying sizes. Subsequently we began developing the tool in C++ programming language.

The paper is organized as follows: Section 2 develops the continuous model, and proves it is a CH model. In the next section, we provide the discrete model and algorithm, and determine the maximal error of boundary approximation. Section 4 develops a stability condition that prevents the curve from self-intersecting, and guides its convergence to the proper CH. The condition also provides a formula for evaluating the maximal time step, that will keep the convergence stable, depending on the image sizes. In Section 5 we provide the calculation complexity, and the truncation error of the algorithm. In the next section we describe a set of experimental results produced by ACHM using MRI, X-ray, PET and synthetic images. The latter set is considered also to confirm that ACHM's time of convergence (number of time iterations) is invariant with respect to the image size. The 7th section uses VF to develop an algorithm capable of segmenting an image to a set of CHs, if multiple regions are present. Section 8 discusses the contribution, advantages and disadvantages of ACHM and compares them with a set of conventional CH methods from the literature. The final section marks the future directions of our research.

2. Active Convex Hull Model

This section develops the continuous ACHM starting with the parabolic DE [5]:

$$\frac{\partial u(t,\, p)}{\partial t} = \alpha^2 \frac{\partial^2 u(t,\, p)}{\partial p^2}, \tag{1}$$

where p is a space parameter, whereas t is a time parameter. If we replace, in Eq. (1), the function $u(t, p)$ with a position parametric vector function $f(t, p)$ of a particle, we will receive $f_t = \alpha^2 f_{pp}$, which we can interpret as $f_t = \alpha^2 a(p)$, where $a(p)$ is the acceleration of the particle with respect to p. Further we present the acceleration $a(p)$ by using normal and tangent components (forces):

$$\frac{\partial f}{\partial t} = a(p) = k|v(p)|^2\vec{N} + |v(p)|'\vec{T}. \tag{2}$$

In the above equation $\alpha = 1$, k denotes the curvature, $\vec{T}$ is the tangent vector, whereas $\vec{N}$ is the normal vector, and $|v(p)|$ shows the speed of the particle.

Consider a closed, smooth, and convex curve C parameterized by $C(t, p) = r(t, p) = \langle X(t, p), Y(t, p)\rangle$ in the domain $[-1, 1] \times [-1, 1]$, where $t \in [0, \infty)$ parameterizes the family and $p \in [0, 2\pi]$ parameterizes the particular curve. If t is fixed, then the particular curve is described by the following smooth vector function $r(t, p) = r(p) = X(p)i + Y(p)j$, and its partial derivatives $r_p = X_p i + Y_p j, r_{pp} = X_{pp} i + Y_{pp} j$, where $|r_p| = \sqrt{X_p^2 + Y_p^2}$. Replacing in Eq. (1) the function $u(t, p)$ with a $C(t, p) = r(t, p)$, we receive the geometric heat DE, which presents a vector flow (VF) [12, 20, 21, 29]:

$$\frac{\partial C}{\partial t} = \alpha^2 \frac{\partial^2 r}{\partial p^2} = \alpha^2 \frac{\partial}{\partial p}\frac{\partial r}{\partial p} = \alpha^2 \frac{\partial \vec{T}}{\partial p}, \tag{3}$$

Further we parameterize the curve $C(t, p)$ with respect to an arc length, since the arc length arises naturally from the shape of the curve and does not depend on a particular coordinate system. Following this reasoning, we compute the length of a curve's arc segment $s(p) = \int_0^p |r_p(u)|du$ and use it to express p in terms of s, as $p = p(s)$. We then re-parameterize the vector function $r(t, p)$ with respect to the arc length s and determine its derivatives:

$$r(t, s) = r(s) = x(s)i + y(s)j,$$
$$r_s = x_s i + y_s j, \quad r_{ss} = x_{ss} i + y_{ss} j.$$

Taking into account that in case of arc length parametrization

$$|r_s| = \sqrt{x_s^2 + y_s^2} = 1, \tag{4}$$

we arrive at the following computationally cheaper form of the well-known curvature formula:

$$k(s) = |r_s(s) \times r_{ss}(s)| \tag{5}$$

Now the right side of Eq. (3) is stated in terms of $x(s)$ and $y(s)$ derivatives:

$$\frac{\partial C}{\partial t} = \alpha^2 \frac{\partial \vec{T}}{\partial s} = \alpha^2 (x_{ss} i + y_{ss} j). \tag{6}$$

Equation (6) evolves the curve C according to the derivative of the unit tangent vector with respect to arc length. Consider the following lemma:

Lemma 2.1. *The product of the curvature and the normal vector equals the derivative of the tangent vector:* $d\vec{T}(p(s))/ds = k\vec{N}$, *where* $\vec{T}(p(s)) = x_s i + y_s j$

Proof: Take into account that:

$$k(p(s)) = \frac{|\vec{T}_p(p(s))|}{|r_p(p(s))|}, \quad \vec{N}(p(s)) = \frac{\vec{T}_p(p(s))}{|\vec{T}_p(p(s))|},$$
$$\frac{ds}{dp} = |r_p(p(s))|.$$

Multiplying the left and right sides of k and $\vec{N}$ we arrive at:

$$k\vec{N} = \frac{|\vec{T}_p|}{|r_p|}\frac{\vec{T}_p}{|\vec{T}_p|} = \frac{\vec{T}_p}{|r_p|} = \frac{d\vec{T}}{dp}\Big/\frac{ds}{dp} = \frac{d\vec{T}}{ds}.$$

□

Employing the above Lemma to Eq. (3) we receive:

$$\frac{\partial C}{\partial t} = \alpha^2 k\vec{N}. \tag{7}$$

Taking into account that: $k(s) = |x_s y_{ss} - x_{ss} y_s|$, and $\vec{N}(s) = -y_s i + x_s j$, we receive:

$$\alpha^2 k\vec{N} = \alpha^2(-y_s|x_s y_{ss} - x_{ss} y_s|i + x_s|x_s y_{ss} - x_{ss} y_s|j), \tag{8}$$

which represents a VF [12, 20, 21, 29], and another form of Eq. (6).

Note that Eq. (8) is computationally cheaper with respect to the equation, that will be obtained if p parametrization is used. But comparing Eqs. (8) and (6), we observe that the latter contains less arithmetical operations. Therefore, unlike [12, 20, 21, 29] from now on we will use Eq. (6) instead of Eq. (8).

Denote by P a penalty function to be used by the model to stop evolvement of all arc segments whose normal vectors $d\vec{T}/ds$ have reached the boundary of a region within the image $I(x, y)$. A replacement of α^2 with P in Eq. (6) yields:

$$\frac{\partial C}{\partial t} = P\frac{d\vec{T}}{ds} = Pk\vec{N}. \tag{9}$$

The latter equation represents a centripetal force, because the VF has only a normal component (force), which points to the center of the image.

Recall we received Eqs. (2) and 9 from Eq. (1), where we substituted $u(t, p)$ with $f(t, p)$ and $r(t, p)$ respectively. Therefore they present different forms of the parabolic DE introduced by Eq. (1), and we can add to the right side of Eq. (9) a term of the form $c\vec{T}$ (where c is a scalar).

$$\frac{\partial C}{\partial t} = P\frac{d\vec{T}}{ds} + c\vec{T}. \tag{10}$$

Also, studying Eqs. (2) and (9) we observe that $P = |v(s)|^2 = 1$ (see Eq. (4)). It means that $P = 1$ if a particle is moving on the curve $C(t, s)$. Conversely, if the particle hits an obstacle (object) and stops its motion then $|v(s)| = 0$ and $P = 0$. To determine when the particle (the VF) hits an object (obstacle), we design P as a cutoff function of the rate of change of the image function $I(x, y)$ with respect to decreasing s along the VF at the point $r(x(t, s), y(t, s))$ on the curve C at time t:

$$P = P_{\epsilon,\,\epsilon_1}\left(\frac{dI(x, y)}{ds}\right)_{(k\vec{N}+c\vec{T}).r} \to \{0, 1\}. \tag{11}$$

If the rate of change, calculated on an interval $\Delta_t \in (k\vec{N} + c\vec{T})$, is between ϵ and ϵ_1 then $P = 0$ otherwise $P = 1$.

Recall that the tangent vector is given by $\vec{T}(s) = x_s i + y_s j = dr/ds$. Therefore $dr = (x_s ds)i + (y_s ds)j$. Taking into account that $dx/ds = x_s$ and $dy/ds = y_s$ we receive $dx = x_s ds$ and $dy = y_s ds$. Replacing them in the equation for dr we receive:

$$dr = (dx)i + (dy)j. \tag{12}$$

In Eq. (12) dr shows how much r changes if a particle moves a small distance ds along the curve C from a point to another point nearby. Using Eqs. (12) and (4) we determine:

$$|dr| = \sqrt{(dx)^2 + (dy)^2} = |ds|\sqrt{x_s^2 + y_s^2} = |ds|. \tag{13}$$

Without any restriction of the reasoning we replace in Eq. (10) the coefficient c with $-|dr|$ and arrive at the following innovative form of the geometric heat DE:

$$\frac{\partial C}{\partial t} = P\frac{d\vec{T}}{ds} - |dr|\vec{T}. \tag{14}$$

The tangent vector $-|dr|\vec{T}$ prevents the active CH from intruding into concavities of an image region when convex portions of its shape are reached. This property shows that Eq. (14) defines a new Active Convex Hull Model (ACHM). To prove it we consider the well known CH definition:

Definition 1. CH of a set of points is a polygonal area of a smallest length, containing the points, so that any pair of points within the area have the line segment between them contained entire inside the area.

By definition, an image region is called object, if it consists of one connected set of pixels (points), whose gray level does not exceed a given interval $[\epsilon, \epsilon_1]$ and whose dimensions are larger then given thresholds Δ_t and Δ_s. The latter we call a radial distance, and it shows the number of vectors which hit a region, while Δ_t denotes the minimal interval (number of pixels) a vector should penetrate an image region in order to stop evolvement of the arc the vector is connected with. Follows that the CH definition applies to image objects, and denote by $d(P_i)$ the distance from a boundary point to the initial active contour $C(0, s)$, which is large enough to inscribe the entire image ($[-1, 1] \times [-1, 1]$). The distance is employed hereafter to formulate the notion local extreme point, used in slightly different sense by [6].

Definition 2. A boundary point E_i is a local extreme point on an arc segment s if $d(P_j) > d(E_i)$ holds for any point P_j, which belongs to s and is different from E_i.

Consider that E_j and E_i are extreme points on a boundary arc segment b_s such that $d(E_j) > d(E_i)$ (the reverse case leads to the same reasoning). Follows from Eq. (14) that the active CH will first hit E_i and then E_j. Also, the active curve segment C_s, between them, is convex (Fig. 6(b)). Since $P = 0$ at E_j and E_i the Eq. (14) is of the following form, at these points: $\partial C/\partial t = -|dr|\vec{T}$. During the time of further evolution every point on C_s, except the points which coincide with E_j and E_i, will converge guided by Eq. (14) (Figs. 6(c–e). Those points on C_s, which coincide with

E_j and E_i, will stay motionless because no normal force is acting upon them. Thus C_s will converge to the straight segment between E_j and E_i (Fig. 6(f)). Once the straight segment is reached by C_s the term $k\vec{N} = d\vec{T}/ds$ in Eq. (14) will become zero, because the curvature $k = 0$ on a straight segment. Therefore no normal force (component) will be acting at the points on C_s. Follows that C_s will stop its further evolution, because the normal force is zero on E_jE_i.

Consider consecutive extreme boundary points E_1, $E_2, \ldots, E_k$ reached by members of the VF. Follows that every triplet E_{i-1}, E_i, E_{i+1} for $i = 2, \ldots, k-1$ satisfies the following condition: the number $d(E_i)$ is either between $d(E_{i-1})$ and $d(E_{i+1})$ or grater than both of them. If E_i does not satisfy the condition, E_i is not on the CH, because it is behind the straight segment $E_{i-1}\ E_{i+1}$, in the direction of the VF, and can not be reached by the VF. Follows that the extreme points $E_1E_2 \ldots E_k$ constitute a CH. Since they are boundary point follows that the polygon is the minimal, which contains the image region (Figs. 1–5).

In order to make the capture range ($C(0, s)$) of ACHM large enough to inscribe the entire image we use the tangent vector normalized with respect to the image sizes as is given below:

$$\vec{\tau}(s) = \frac{2x_s}{nc}i + \frac{2y_s}{nr}j. \tag{15}$$

In Eq. (15) nc denotes the number of columns, whereas nr denotes the number of rows in the image. Replacing in Eq. (14) the derivative of $\vec{T}$ with the derivative of $\vec{\tau}$ we obtain the image form of the ACHM:

$$\frac{\partial C}{\partial t} = P\frac{d\vec{\tau}}{ds} - |dr|\vec{T}. \tag{16}$$

Note that now the penalty function P is defined along the vector $\vec{v} = d\vec{\tau}/ds - |dr|\vec{T}$. Equation (16) defines a VF with two forces: normal force of the vector field (NFVF), determined by $Pd\vec{\tau}/ds$; and tangent force of the vector field (TFVF) determined by $-|dr|\vec{T}$. Thus, Eq. (16) represents a new innovative ACHM, whose capture range depends on the image size and can be set as far as we want from the targeted object. If we present $d\vec{\tau}/ds$ and $\vec{T}$ in the terms of the derivatives of $x(s)$ and $y(s)$ we arrive at the following ACHM form:

$$\frac{\partial C}{\partial t} = \left(P\frac{2x_{ss}}{nc} - |ds|x_s\right)i + \left(P\frac{2y_{ss}}{nr} - |ds|y_s\right)j \tag{17}$$

The next section discusses ACHM implementation by the Forward Difference Method.

3. The Discrete Model and Algorithm

To implement the ACHM one can use Eq. (17) and employ first and second finite (central) differences to approximate the first and second derivatives of x and y with respect to s. To develop the discrete ACHM and algorithm, we use Eq. (16), and employ first order differences to approximate the derivatives:

$$x_t \approx \frac{x(t+\delta, s) - x(t, s)}{\delta},$$
$$y_t \approx \frac{y(t+\delta, s) - y(t, s)}{\delta},$$
$$\frac{d\vec{\tau}(s)}{ds} \approx \frac{\vec{\tau}(t, s+\Delta s) - \vec{\tau}(t, s+\Delta s)}{\Delta s}.$$

Also, considering Eq. (13) we receive $|\Delta s| = |\Delta r| = |r(s+\Delta s) - r(s)|$, which is the distance between two points on the active CH. Now we rewrite Eq. (16) in the form:

$$\begin{aligned}\langle x(t+\delta, s), y(t+\delta, s)\rangle &\approx \langle x(t, s), y(t, s)\rangle \\ &+ \frac{\delta}{\Delta s}P\langle\vec{\tau}(x(t, s+\Delta s), y(t, s)) \\ &- \vec{\tau}(x(t, s), y(t, s)), \vec{\tau}(x(t, s), y(t, s+\Delta s)) \\ &- \vec{\tau}(x(t, s), y(t, s))\rangle - \delta\Delta s\langle x_s(t, s), y_s(t, s)\rangle.\end{aligned}$$

The latter equation guides the curve evolvement to the CH of a desired image region, and leads to the development of the following Active Convex Hull Forward Difference Algorithm:

$$r_i^{j+1} \approx r_i^j + V_i^j, \quad \text{where} \quad V_i^j = \delta^j P_i^j \partial\vec{\tau}_i^j - \delta^j s_i^j \vec{T}_i^j. \tag{18}$$

Since $\delta^j < 1$, follows that $|v_i^j| = |\delta^j P_i^j \partial\vec{\tau}_i^j - s_i^j\vec{T}_i^j| < |V_i^j|$. Since v^j needs less arithmetic operations and does not restrict our reasoning we will use it for the Forward Difference Algorithm (FDA):

$$r^{j+1} \approx r^j + v^j. \tag{19}$$

In Eq. (19) we add a shorter vector, but can compensate it by using larger δ^j. In the above expressions we used the same notation as in the continuous case: $r_i^j = r^j(t^j, s_i^j)$, for $i = 1, \ldots, n$, and $r_0^j = r_n^j$; j parameterizes the family, while i parameterizes the particular curve; $s_i^j = |r_i^j - r_{i-1}^j|$ is the i-th discrete space step at time j; $\partial\vec{\tau}_i^j \approx (\vec{\tau}_i^j - \vec{\tau}_{(i-1)}^j)/s_i^j$; and $\vec{T}_i^j \approx (r_i^j - r_{i-1}^j)/s_i^j$ for $i = 1, \ldots, n$, where $\vec{T}_0^j = \vec{T}_n^j$, and $\vec{\tau}_0^j = \vec{\tau}_n^j$.

The penalty function P_i^j is defined by the following piecewise function:

$$P_{\epsilon,\epsilon_1} = P_i^j(k) = \begin{cases} 1 & \text{if } PS_i^j(k) < \epsilon \text{ or } PS_i^j(k) > \epsilon_1 \\ 0 & \text{otherwise,} \end{cases} \tag{20}$$

where $\epsilon < \epsilon_1$ are thresholds defined by the user and show the difference between the background and foreground. The penalty sum $PS_i^j(k) = \sum_{m=k}^{k+\Delta_t} de_i^j(m)$, for $k = 0, \ldots, |r_i^{j+1} - r_i^j| - \Delta_t$, and $de_i^j(m) = |I_i^j(m) - I_i^j(m-1)|$. In the latter expression, $I_i^j(m)$ is the gray level value of the mth pixel that lies on the ith normal vector at time j. It follows from Eq. (20), that the algorithm is marching $|r_i^{j+1} - r_i^j| - \Delta_t$ pixels on each vector, stops at each pixel m and "looks" Δ_t pixels ahead. If the rate of change on the interval Δ_t belongs to $[\epsilon, \epsilon_1]$, the marching stops and the boundary approximation is set at the pixel m. Otherwise, go to pixel $m+1$ and repeat the algorithm. Follows from the above concepts that the maximal error of boundary approximation generated by the VF, defined by Eq. (19), is Δ_t.

4. The Stability Convergence Condition

In this section we develop a condition that relates the space step s and time step δ in a way, which ensures stable convergence of the curve C to the CH of the desired image region. Consider Eq. (19) to set up the following inequalities:

$$\begin{aligned} |r_i^{j+1}| &\approx \left|r_i^j + \delta^j P_i^j \partial\vec{\tau}_i^j - s_i^j \vec{T}_i^j\right| < \left|r_i^j + \frac{\delta^j P_i^j \partial \vec{T}_i^j}{m}\right| \\ &= \left|r_i^j + \lambda^2 \delta^j \partial \vec{T}_i^j\right|, \end{aligned} \tag{21}$$

The very right part represents the FDA of the geometric heat DE in the following form:

$$\frac{dC}{dt} = \lambda^2 \frac{d\vec{T}}{ds} = \lambda^2 k \vec{N}, \quad \text{where} \quad \lambda^2 = P/m \quad \text{and} \quad m = \min(nc/2, nr/2). \tag{22}$$

Using a statement from [5] we determine that the FDM of Eq. (22) stably converges to its solution if:

$$\delta^j < \left(s_{ave}^j\right)^2 2\lambda^2 \quad \text{where} \quad \left(\sum_{k=1}^{n} s_i^j\right) \Big/ n = s_{avg}^j, \tag{23}$$

Thus, we arrive at the following form of the Stability Convergence Condition (SCC) (23):

$$\delta^j < m\left(s_{ave}^j\right)/2. \tag{24}$$

where we considered that $P = 1$ at the time of convergence. To compute δ^j which will keep stable convergence of the FDA 19 to the CH we use:

$$\delta^j = c s_{ave}^j, \quad \text{where} \quad c < m/2. \tag{25}$$

Also, Model 16 together with SCC 24 make the convergence time (number of time iterations j) invariant with respect to the image size. The latter means that if we consider an image of size $n \times n$ and enlarge it to a size $nk \times nk$ the ACHM will increase the speed of stable convergence approximately *k-times* and the run time to determine the CH for both images is approximately the same. An experimental verification of this conclusion is given in Section 6.

5. Calculation Complexity and Truncation Error

Considering the Active Convex Hull Forward Difference Algorithm 19 and SCC 24, we computed that the Number of Arithmetical Operations (NAO) that ACHM takes to determine the CH of an image region is approximately $50Jn$, which is on the order of $O(Jn)$. The symbol J denotes the number of iterations (the number of time steps) taken by ACHM to reach the CH, and n denotes the number of arc segments, or vectors, on the curve.

Consider the ACHM defined by Eq. (17) and finite differences to approximate the derivatives. Then the following equations are used to determine the truncation errors that come from derivatives approximation [5]: $[r(t+\delta, s) - r(t,s)]/\delta = \partial C/\partial t + \delta 2/r_{tt}(t_*, s)$, $[x(t, s+\Delta s) - x(t,s)]/\Delta s = x_s + \Delta s/2 x_{ss}(t, s_*)$, $[x(t, s+\Delta s) - 2x(t,s) + x(t, s-\Delta s)]/(\Delta s)^2 = x_{ss} + (\Delta s)^2/12 x^{(IV)}(t, s_*)$.

The valve t_* is an internal point on the vector, while s_* is an internal point on the arc segment. Follows that the truncation error that will be generated approximating $\partial C/\partial t$ is on order of $O(\delta)$; for x_s is on order of $O(\Delta s)$, and for x_{ss} is on order of $O((\Delta s)^2)$. Analogously, we see that y_s has a truncation error $O(\Delta s)$, and y_{ss} has $O((\Delta s)^2)$. Follows that the truncation error of the FDM that implements ACHM is $O(\delta + \Delta s)$. The same truncation error will be generated if Backward Differences are employed to approximate the derivatives. But if one uses Central Differences the truncation

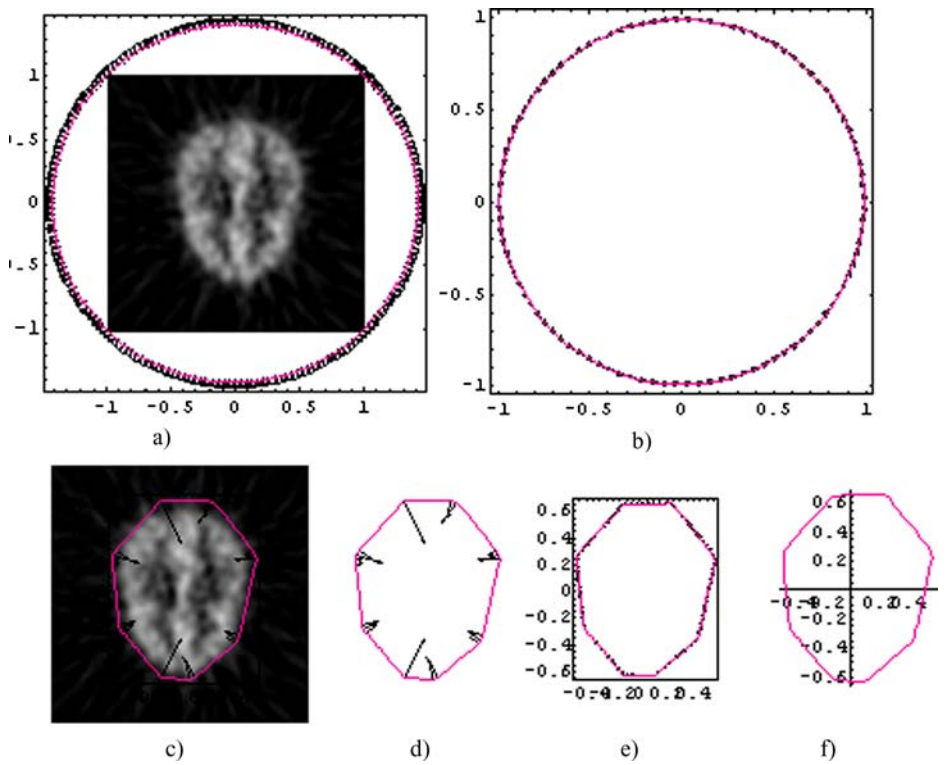

Figure 1. (a) A PET image of size 128×128, the initial CH, and the NFVF; (b) the initial TFVF; (c) the object, the active CH, and NFVF after 2000 iteration; (d) The active CH, together with the normal force after 2000 iterations; (e) the tangent force; (f) the CH alone.

error of ACHM is on the order of $O(\delta^2 + (\Delta s)^2)$. It gives a better approximation, but requires five knots to approximate the derivatives.

6. Experimental Results

On the basis of the FDA 19 and SCC 24, we developed a prototype tool (Active Convex Hull Tool-ACHT) using *Mathematica5.2*. Recall that given an initial convex curve, Algorithm 19 will evolve it in an inward direction until the CH of the image region is encountered. Employing *ACHT*, experiments were performed using X-ray, MRI, and PET medical images as well as a set of synthetic images of varying sizes. We begin with the following initial parametrization of the convex curve C defined by $r(0, p) = \langle R\cos p, R\sin p\rangle$, where $p \in \{0, \ldots, (2\pi/n)i, \ldots, (2\pi/n)(n-1)\}$, and R is the radius. Then following the concept from Section 2 we switch to arc length parametrization. You can tell from above, a circle is used as an initial curve. But Fig. 3 shows that ACHM was successfully applied to rectangular images. Since ACHM defined by Eq. (16) and SCC 24 allow us to set the initial contour as far as we want, we used a radius $R = \sqrt{2}$ to inscribe the entire domain $[-1, 1] \times [-1, 1]$. In this case the area outside the image but in the circle is considered as background.

Figure 1(a) depicts a PET image of size 128×128. The initial curve has a radius $R = \sqrt{2}$ and is divided to 40 arc segments (the number of inward vectors is the same) that yields $s^0_{ave} = 0.22$ in the domain. Employing SCC 24 and considering that $m = 64$, we computed that $\delta^0 < 32(0.22)^2$, and used $\delta^j = 16(0.22)^2$. In Fig. 1(a) are shown, also the initial active CH and the normal force of the vector field (NFVF), whereas the initial tangent force of the vector field (TFVF) is given in Fig. 1(b). The image, the active CH together with the NFVF after 2000 time iterations are shown Fig. 1(c). Figure 1(d) gives the active CH together with the NFVF after 2000 iterations, while the tangent force is presented in Fig. 1(e). The CH, of the image object is shown in Fig. 1(f). The minimal perimeter of the CH, in the domain $[-1, 1] \times [-1, 1]$, is 3.666, whereas in the image is 235 pixels, and was obtained after 2000 iterations, which gives $(4)10^6$ arithmetical operations. It means that performing more iterations will not lead to any change of the CH length and shape. Note that in Figs. 1(c) and (d) part of the NFVF vanished, because the curvature at straight segments is 0.

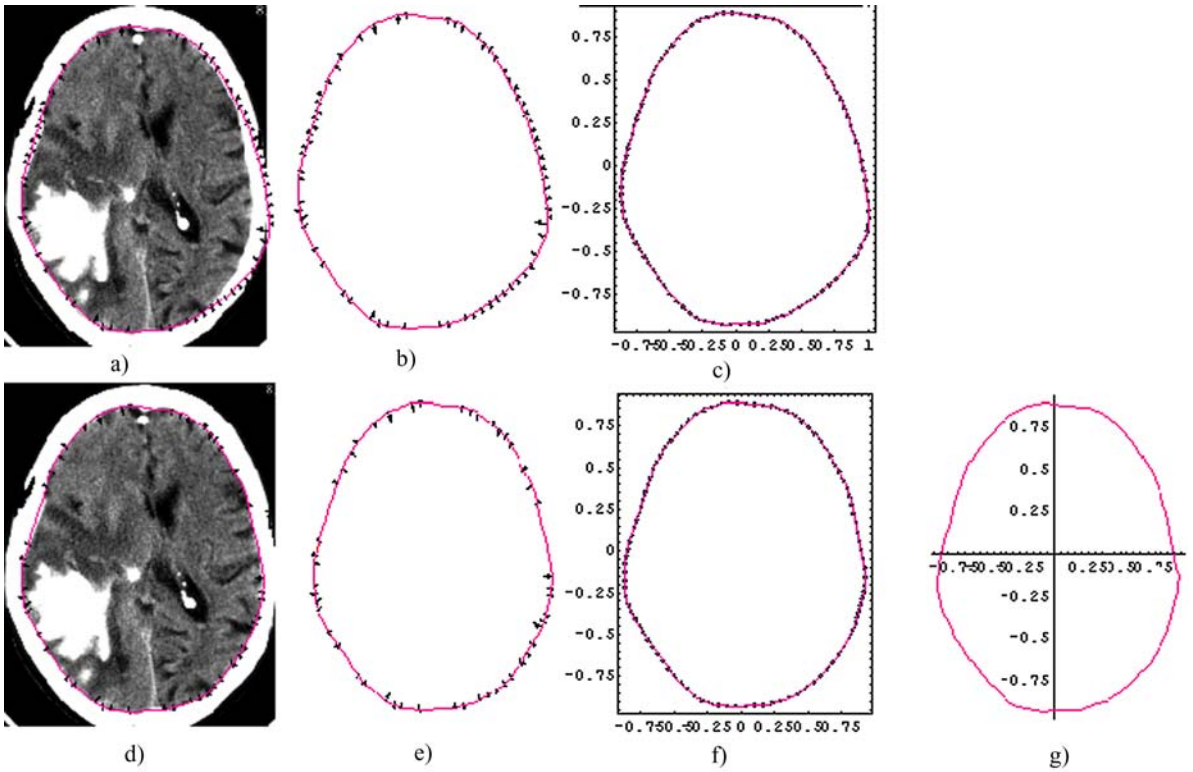

Figure 2. (a) An 256×256 MRI image, the active CH, and NFVF after 500 iterations; (b) the active CH, together with NFVF; (c) the TFVF; (d) the object, the active CH, and NFVF after 1000 iteration; (e) The active CH, and the NFVF after 1000 iterations; (f) the tangent force; (g) the CH alone.

In the same figures some members of the normal force stretched, because the curvature at the extreme point is larger then the curvature at the same curve point before the "contact" with the extreme point. As you may tell from Figs. 1(b) and (e) the tangent force did change the shape and the length but not its dimension. The latter follows also from Eq. (16).

InFig. 2(a) we show an MRI image of size 256×256 of a Parenchymal Hemorrhage. The initial curve has a radius $R = \sqrt{2}$ and is divided to 100 arc segments that yields $s^0_{ave} = 0.088$ in the domain, which equals to approximatly 12 pixels. Employing SCC 24 and considering that $m = 128$, we computed that $\delta^0 < 64(0.088)^2$, but used $\delta^j = 33(0.088)^2$. In Fig. 2(a) are shown also the active CH and NFVF after 500 iterations, while in Fig. 2(b) they are given alone. The TFVF is presented after 500 iterations in 2(c), whereas the image, the active CH together with the NFVF after 1000 iterations are shown Fig. 2(d). Figure 2(e) gives the active CH together with the NFVF, whereas the tangent force is presented in Fig. 2(f). The CH, of the image object, alone is shown in Fig. 2(g). The minimal perimeter of the CH, in the domain $[-1, 1] \times [-1, 1]$, is 5.64462, whereas in the image is 723 pixels, and was obtained after 1000 iterations, which gives $(5)10^6$ arithmetical operations. Once again in Figs. 2(b) and (e) part of the NFVF vanished, because the curvature at straight segments is 0. In the same figures some members of the normal force stretched, because the curvature at the extreme point is larger then the curvature at the same curve point before the "contact" with the extreme point. As one can tell from Figs. 2(c) and (f) the TFVF did change the shape but not its dimension.

Figure 3(a) shows an X-ray image of size 301×500. The initial curve has again a radius $R = \sqrt{2}$ but the VF contains 160 vectors, which yields $s^0_{ave} = 0.055$ in the domain, and approximately 14 pixels in the image. To provide a stable convergence to the CH of the hand we used $\delta^j = 33(0.055)^2$. In Fig. 3 we present the object, the active CH, the NFVF, and the TFVF after 1000 and 7000 iteration, when the active CH reached its minimal perimeter 5.9938, whose size in the image is 1498 pixels. *ACHT* performed approximately $(56)10^6$ arithmetical operations to determine this CH.

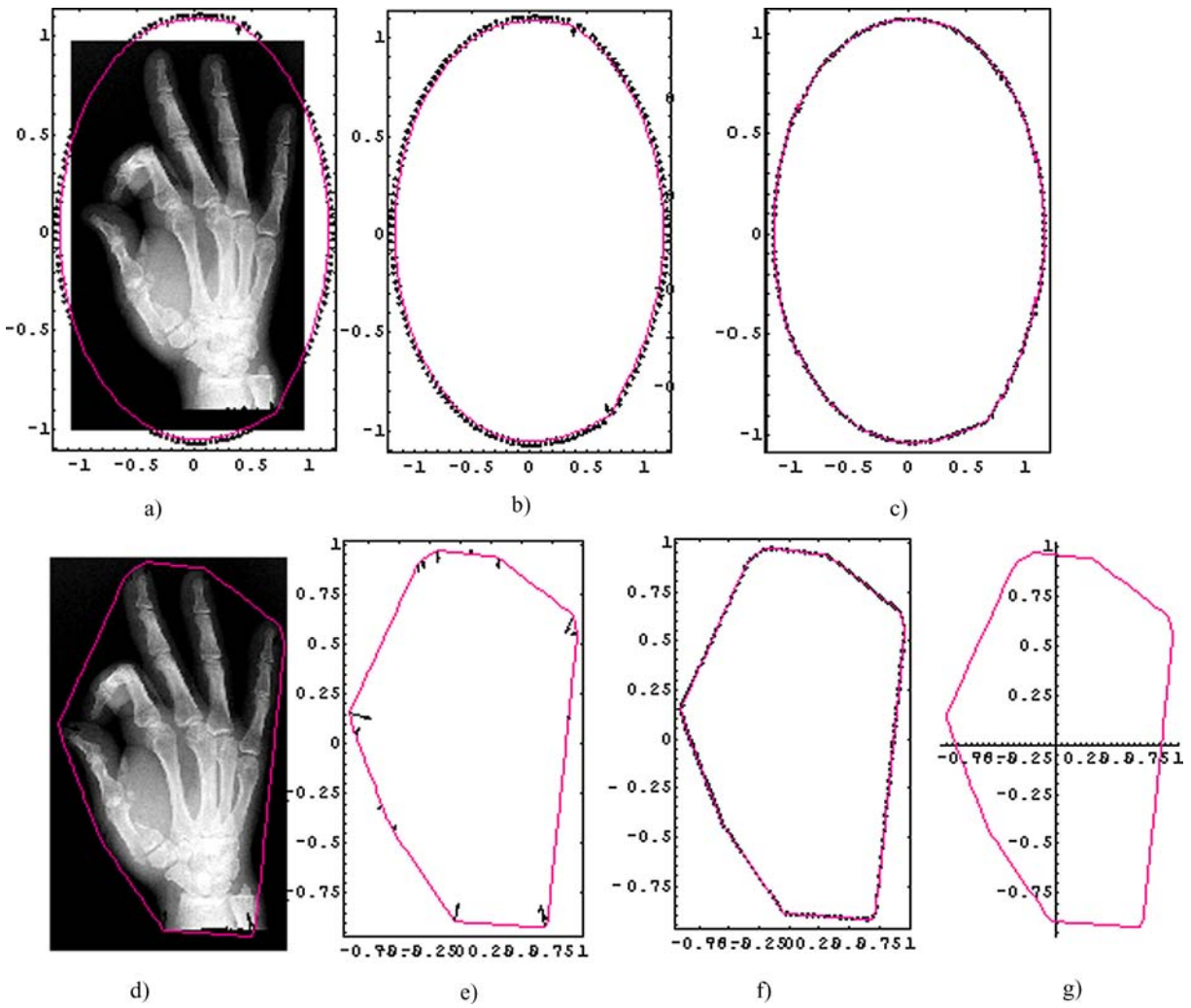

Figure 3. (a) An X-ray image of size 301 × 500, the active CH, and NFVF after 1000 iterations; (b) the CH, and NFVF from (a); (c) the TFVF after 1000 iterations; (d) the image, the active CH, and NFVF after 7000 iteration; (e) The active CH, and NFVF; (f) the tangent force; (g) the CH after 7000 iterations.

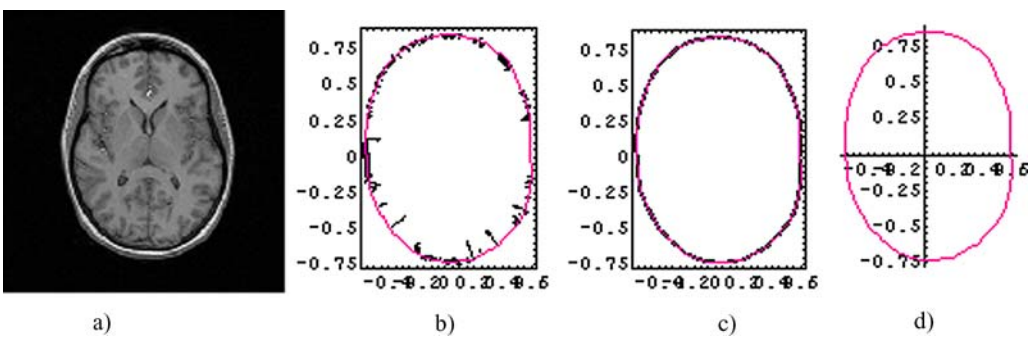

Figure 4. (a) An MRI image of size 256 × 256; (b) the active CH, together with NFVF after 3000 iterations; (c) the TFVF after 3000 iterations; (d) the CH alone.

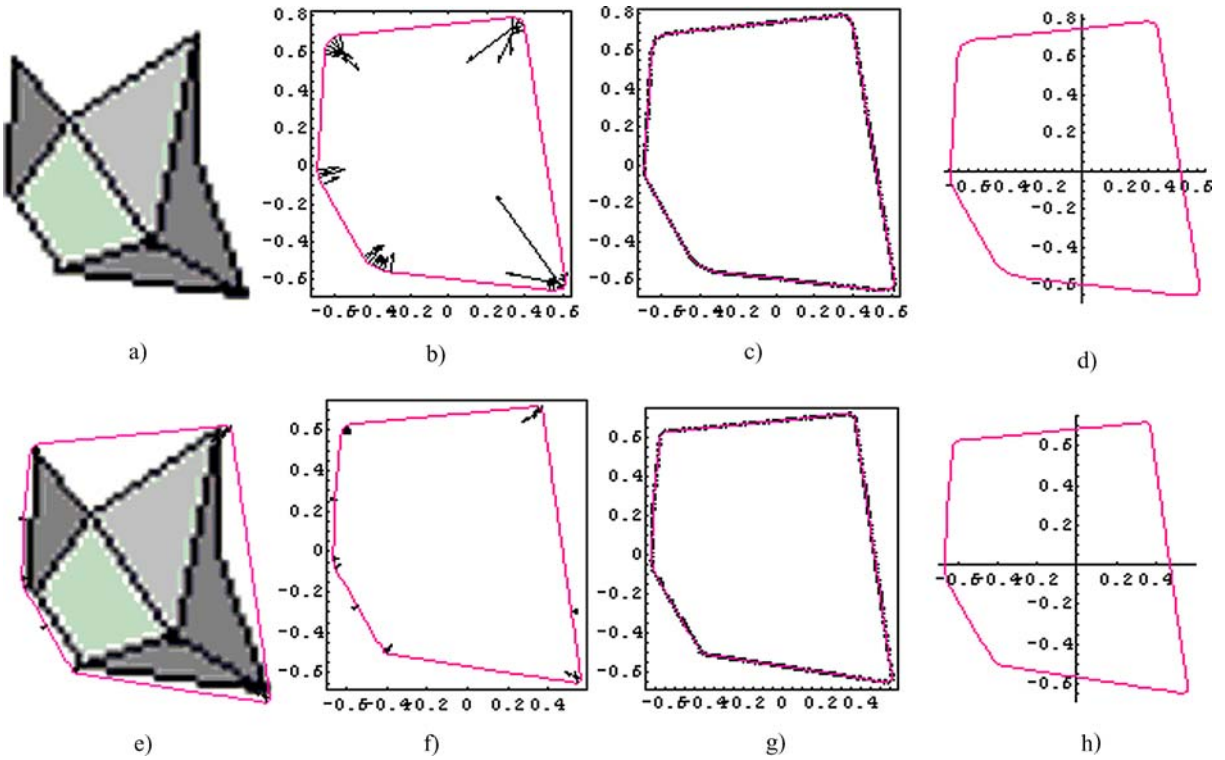

Figure 5. (a) A synthetic image of size 128 × 128; (b), (c) and (d) presents the active CH, together with NFVF; the TFVF; and the active CH alone after 6000 iterations; (e) the image from (a) but enlarged to 3072 × 3072; (f), (g) and (h) presents the active CH, together with NFVF; the TFVF; and the active CH alone after 6000 iterations.

The last experiment we made with medical image considers MRI image of size 256 × 256 (Fig. 4(a)). The minimum perimeter of length 4.43122 (in the domain, which equals to 568 pixels in the image) was reached after 3000 iteration, which gives $(37.5)10^6$ arithmetic operations (Fig. 4(d)). The used VF consists of 250 vectors, which yields $s^0_{ave} = 0.037$. The active CH together with NFVF is shown in Fig. 4(b), whereas Fig. 4(c) presents the TFVF.

Further, we performed a set of experiments in order to demonstrate the capability of *ACHT* to increase the convergence speed (the time step of convergence) with the same ratio as the image size increases. For this purpose we used a synthetic image of size 128 × 128 (see Fig. 5(a)), and its enlarged derivatives 256 × 256, 512 × 512 , 1024 × 1024, 2048 × 2048, and 3072 × 3072 pixels. Note that enlarging the original image the object located there was enlarged with the same ratio. For all experiments we used a VF containing one and the same number of vectors equal to 200, and the initial CH was a circle with a radius $R = \sqrt{2}$. For the first image (128 × 128) we used $\delta^j = 24(0.044)^2$, where $0.044 = s^0_{ave}$, and $24 = c$ is the coefficient in Eq. (25). For the next image whose sizes are twice larger we used $c = 48$. The third image has sizes 4 times larger then the first one and the used coefficient was $c = 96$. The next image is of size 1024 × 1024 and ACHT used $c = 205$, which is slightly larger then $8 * 24 = 192$. To perform the experiments with the two largest images 2048 × 2048, and 3072 × 3072 we increased the VF to $n = 300$. From SCC 24 we receive $c < 512$, but used $c = 490$ to keep stable convergence in the image 2048 × 2048. In what concern the image 3072 × 3072 we received $c < 768$, but used $c = 760$. In all experiments ACHT reached the minimal length of the CH after 6000 iterations. The CHs obtained from both images 128 × 128, and 3072 × 3072 are shown in Figs. 5(d) and (h).

The above results verify that ACHT's number of iterations (time of convergence) is invariant with respect to the image size. In other words, the algorithm performs approximately the same number of arithmetic operations to determine the *CH* in a stretched or shrank images. The maximal step of stable convergence is automatically computed by Eq. (25), selecting c with respect to the image size.

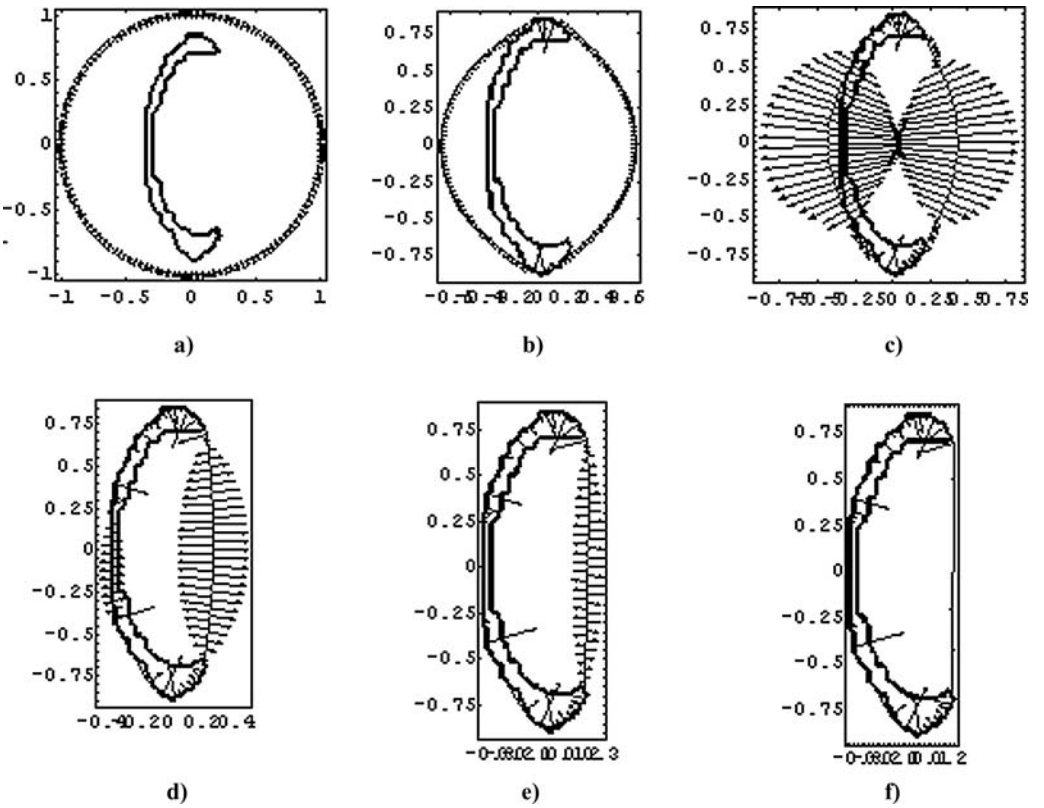

Figure 6. (a) Image of size 140 × 140, together with the initial curve and the NFVF; (b) The object from (a) together with the NFVF and the active CH after 500 iterations; (c) after 1000 iterations; (d) after 2000 iterations; (e) after 3000 iterations; (f) and after 6000 iterations.

In Fig. 6(a) we present a simplified section from a gravel deposit in an image of size 140 × 140, together with the initial curve and the NFVF. Figure 6(b) shows the object, the shape of the active *CH*, and the NFVF after 500 iterations. The curve evolvement and the corresponding NFVF behavior after 1000, 2000, 3000 and 6000 iterations can be observed in Figs. 6(c–f). The used *VF* consists of 140 vectors, and $c = 28$. The obtained minimal length of the *CH* in the domain is 3.93383, which equals approximately 275 pixels.

The experimental result given in Fig. 7 shows that *ACHM* is a model without edges. Figure 7(a) contains a swarm of points in an image of size 64 × 64. The initial curve has a $R = 1$ and is divided into 140 segments, yielding $s^0_{ave} = 0.0634$. Using Inequality 24 we computed that $\delta^0 < 16(0.0634)^2$ and set $c = 13$ to determine δ^j for $j = 1, 2, 3, \ldots$. The active *CH* and part of the NFVF are given in Fig. 7(b) after 1000 iterations, while Fig. 7(c) shows the *CH* together with the normal force after 2500 iterations. Again the vectors normal to straight segments vanished, and those at the contact points enlarged. The *CH* alone is shown in Fig. 7(d), and has a minimal length 5.07757 that is equal to 162 pixels.

The following list of features can be derived from the theoretical concepts and experiments described above. The ACHM and ACHT are capable of:

- working with gray level images of varying sizes (see Figs. 1–5);
- increasing the speed of stable convergence with approximately the same ratio as the image size increases (Indeed for the image 128 × 128 we used $c = 24$; for 256 × 256 we used $c = 48$; for 512 × 512, $c = 96$ was used; whereas for the image 1024 × 1024 the coefficient was 205, and for 2048 × 2048 $c = 490$, for 3072 × 3072 $c = 760$,);
- providing a large capture range that can inscribe the whole image (see Figs. 1, 3);
- CH modeling of objects without edges (see Fig. 7).

The above listed features show that ACHT is suitable for applications to automatic image database mining for regions' location and CH extraction. Consequently, the CH was used in [27], for medical image database

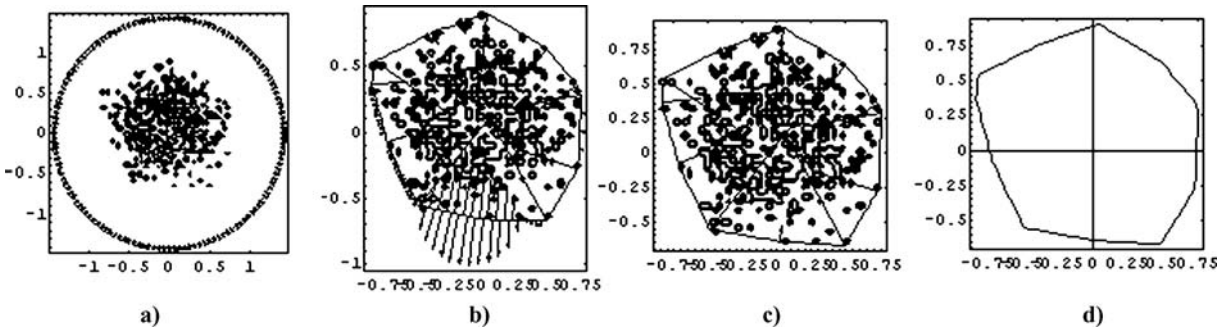

Figure 7. (a) A swarm of points in an image 64 × 64 together with the NFVF, and the initial active CH with a $R = 1$; (b) The active CH, part of the NFVF, and the swarm after 1000 iterations; (c) after 2500 iterations; (d) The CH alone.

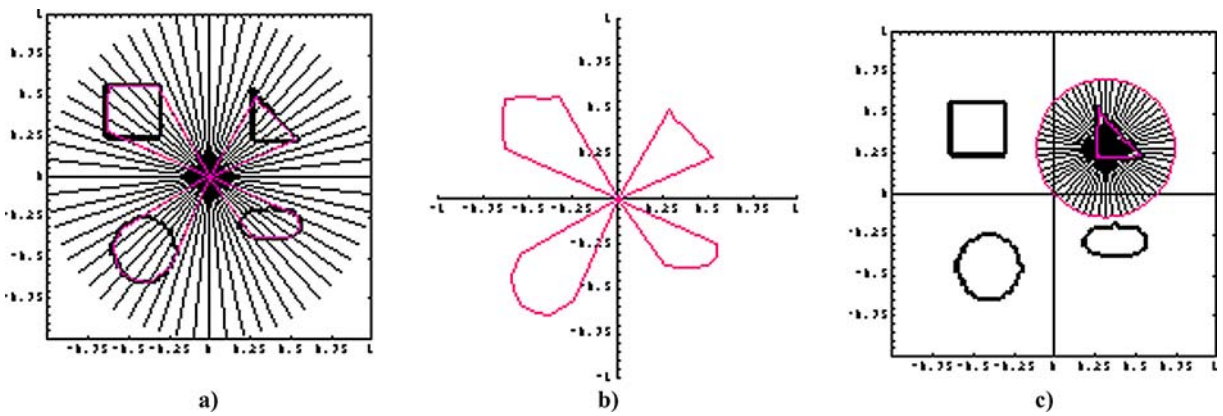

Figure 8. (a) Four image regions; (b) The shell for each region; (c) The new active contour defined for the first shell.

indexing and search space partitioning. The advantage obtained, in this work, is a factor of a hundred in the reduction in the number of image database regions necessary to traverse and compare to a given query region.

7. Image Segmentation Using VF

Studying the images used to perform our experiments one can tell ACHM works if the center of the image is located in the image region under-consideration. Also, if there are multiple image regions they will be enveloped by a single CH. To overcome above disadvantages and segment an image to multiple convex hulls we use the notion *shell* [25]. We define *shells* by employing the NFVF (which is centripetal force) to the initial curve and prolong the vectors all the way to the center of the domain $[-1, 1] \times [-1, 1]$, as is shown in Fig. 8(a). By definition, we call a *shell* an image region bounded by peace of the object's boundary and two normal vectors (see Fig. 8(a) and (b)) tangent to other parts of the boundary. Employing it together with Δ_s and Δ_t, defined in Section 2, the algorithm can determine whether multiple objects are present in an image, or there is only one. If the first holds the algorithm computes the mass center of each *shell*, and place it in the domain $[-1, 1] \times [-1, 1]$, such that the mass center coincides with the center of the domain [28]. Then a new initial curve is defined (see Figs. 8(c) and 9(d)), and the process repeats until a single object remains in each *shell*, where the algorithm runs *ACHT*.

Shell algorithm is coded in C++, and we used it together with ACHT to determine the CH's of the impermeable units (the darkest objects in the image) given in Fig. 9(a). To determine the CH of each impermeable unit we first ran the *shell tool*, which segmented the image to set of shells (Figs. 9(b) and (c)). The segmentation repeats until a single impermeable unit remains in each shell, where a circle is defined (Fig. 9(d)) and ACHT is ran to determine the CH of the impermeable unit, shown together with the corresponding boundary in Fig. 9(e).

Table 1. Shows back to back the time taken by *ConvexHull*, *ACHT* , and *Convex* to obtain the CH of the objects in the images with sizes given in the first column. The shape of the object under-consideration is shown in Figs. 5(a) and (e).

Image size	Mathematica-ConvexHull	ACHT- notebook	Convex-notebook
128×128	0.55 seconds	254 seconds	0.6 seconds
256×256	17.2 seconds	254 seconds	3.25 seconds
512×512	441 seconds	254 seconds	13 seconds
1024×1024	over 1 hour	254 seconds	51.64 seconds
2048×2048	over 1 hour	261 seconds	209 seconds
3072×3072	over 1 hour	261 seconds	Run out of memory after approximately 30 min

8. Advantages and Contributions

The main contributions of this paper are as follows:

- a new time-cheap active convex hull model is developed on the base of parabolic Eq. (1) and using VF with normal and tangent components (forces);
- a condition is developed to guarantee stable convergence of the *ACHM*;
- *ACHM's* time of stable convergence (number of iterations) to the *CH* is invariant with respect to the image size, and the number of points (pixels) to be enveloped;
- the number of arithmetic operations taken by ACHT to find a *CH* is estimated;
- the truncation error and the error of boundary approximation are determined;
- the model uses a new penalty function and provides a large capture range;
- an image segmentation to set of convex hulls is available.

Recall that we used Eq. (19) to implement the ACHM. But one can also use $r^{j+1} = r^j + V^j$, whose SCC and truncation error are the same, but the NAO is greater by an order of nJ, where n denotes the number of normal vectors in the VF, while J denotes the number of time iterations.

One can find a number of 2D CH algorithms in the literature. For example, Bentley-Faust-Preparata [4] sought a calculation complexity on order of $O(n \log n)$, where n defines the number of the plane points to be enveloped by the CH. Another basic algorithm is Grahams scan algorithm [17, 26], whose calculation complexity is also on order of $O(n \log n)$. A number of CH algorithms together with their run time are listed in the web site of Sunday [31].

Four space-efficient algorithms are discussed in [8], and they have several advantages over the traditional approaches. The first one is that space-efficient algorithms allow for processing a larger data sets. Any algorithm that uses separate input and output arrays will require enough memory to store $2n$ points. In contrast, space-efficient algorithms will require enough memory to store n points plus $O(\log n)$ working space [8]. This property makes them less prone to failure, because they do not require a large amount of memory, that may not be available at runtime.

Regarding space efficiency ACHM works in the domain $[-1, 1] \times [-1, 1]$, where the image is considered. Follows that if the input image is of size $k \times m$ ACHM needs a memory of size 3km bytes. Follows that comparing with the methods given above ACHM is more efficient for large images, where the foreground is larger than the background (Table 1).

The first algorithm, described in [8], is based on Graham's scan in combination with an in-space sorting algorithm, and runs in $O(n \log n)$ time. The second and third are running at $O(n \log h)$, where h is the number of points in the upper "hull". Both are based on algorithms described in [10] and [23] respectively. The forth algorithm is an improvement of the work given in [9], and runs in $O(n \log h)$ time, where h denotes the output size. Recall that ACHM runtime is on order of $O(uJ)$, where u shows the size of the VF, while J gives the number of time iterations.

Some of the most efficient CH algorithms, with respect to the runtime, is called *Quickhull* [6], which lies in the bases of the *Convhull* operator provided by MatLab. *Quickhull* is an empirical algorithm similar to randomized algorithms and Delaunay triangulations, but an advantage of *Quickhull* over randomized algorithms is in the use of less space. An advantage of *Quickhull* over ACHM is that the first one is successfully working in dimensions grater than 3, while ACHM could be extended to maximum 3.

If works in dimension less then or equal to three the runtime of *Quickhull* is $O(n \log r)$, where n shows the

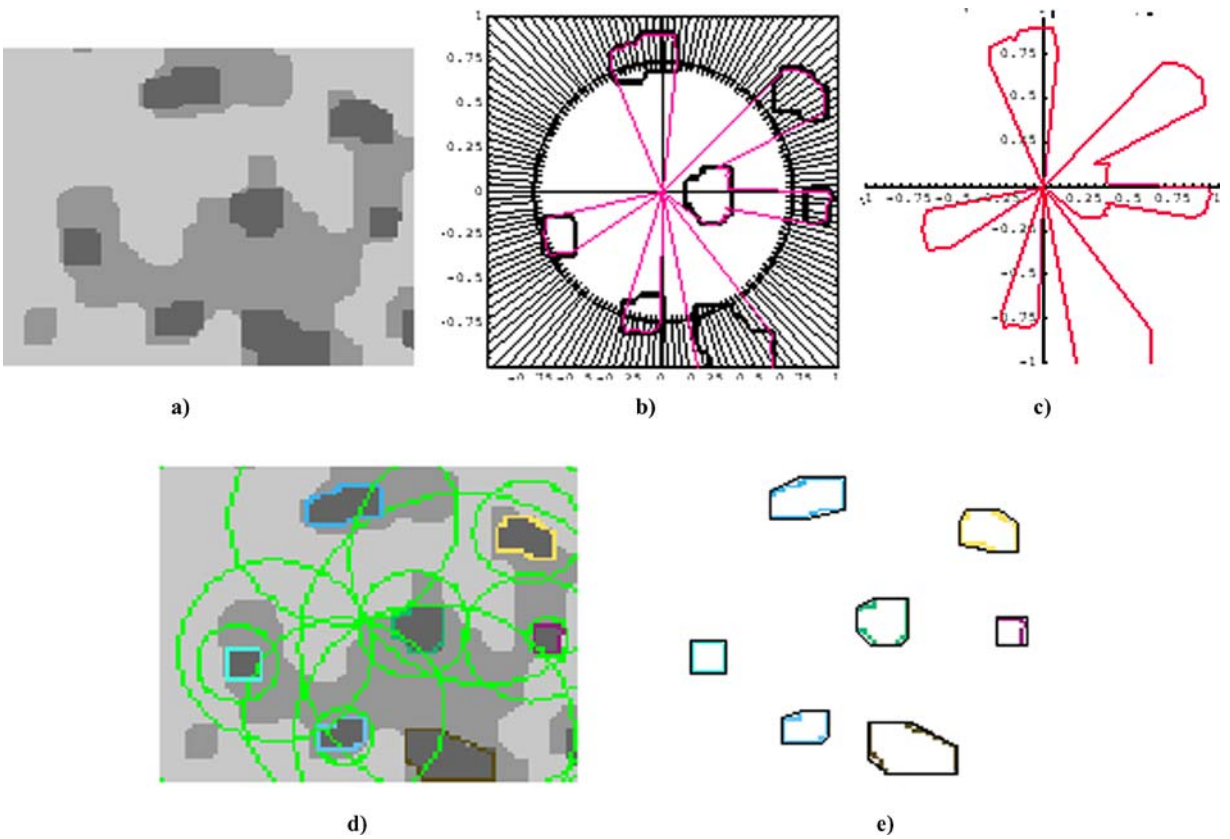

Figure 9. (a) Ground water section, the darkest regions present impermeable units; (b) The VF together with the shells; (c) The shells alone; (d) The circles defined by the shells; (e) the CH together with the boundary of each impermeable unit.

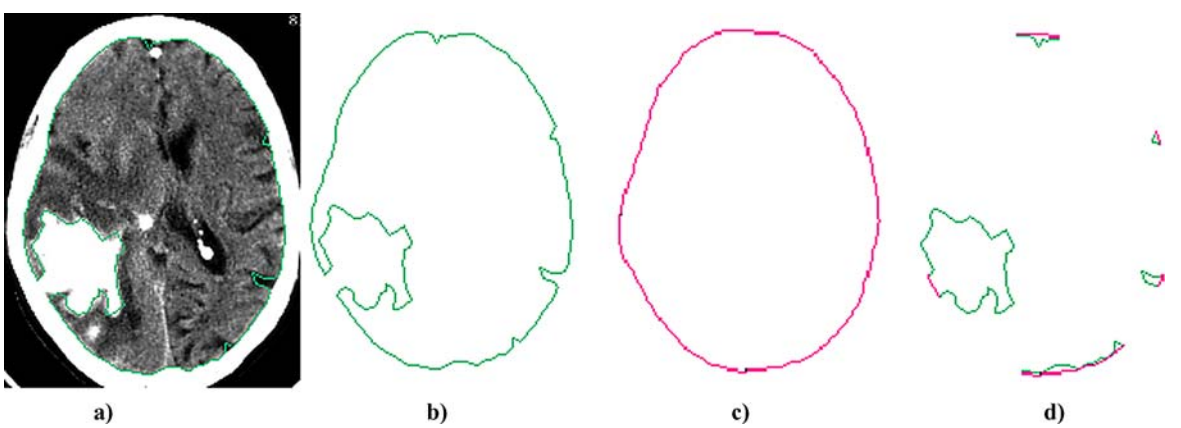

Figure 10. (a) The image from Fig. 2(a); (b) the contour of the parenchyma with a hemorrhage obtained by an active contour; (c) the CH of the parenchyma determined by ACHT; (d) The shape of the extracted hemorrhage and four small concavities.

number of the input points, whereas r gives the number of processed points. For dimensions greater than 3 the runtime is $O(nf_r/r)$, where f_r is the maximum number of facets of r points.

A *ConvexHull* operator is provided also by Mathematica. To run the operator a list of input points is required. To make *ConvexHull* working with images an image processing notebook was developed to extract the objects' points (pixels), whose CH is to be determined, and create the list needed by *ConvexHull*. A set of experiments was performed using the synthetic image of size 128 × 128 (shown in Fig. 5(a)) and its derivatives of sizes 256×256, 512×512, 1024×1024, 2048×2048, and 3072×3072. Recall, enlarging the original image the object located there was enlarged with the same ratio. The above mentioned image processing notebook was used to generate a list of pixels from the object located in each image. The runtime of *ConvexHull* to determine the CH of each object, using the corresponding list, is given in Table 1. In the same table, in column 3, is shown also the runtime taken by ACHM to determine the CH of the same objects. In Table 1 we give also the runtime of an algorithm called *Convex* and posted on the Web page [18] in March 2005. All experiments are made with a Lap Top with CPU 1.8 GHz, and RAM 1 GB.

Studding Table 1, one can tell *ConvexHull* is 500 times faster then ACHM if dealing with the image of size 128×128, but more then 1.7 times slower if dealing with the image of size 512×512. *Convex* is faster then ACHT for all sets of points (pixels) from the images up to 2048×2048, but with the set of points (pixels) from the image 3072×3072 it ran out of memory after approximately 30 minutes. Unlike *Convex*, ACHT took 261 seconds to determine the CH of the object in this image. Note that the number of object's points (pixels) in the image of size 3072×3072 is 576 times larger then the number of object's points (pixels) in the image of size 128×128. Despite of the vast size difference, ACHM took approximately (there is a slight time difference for the very large images) one and the same runtime (number of iterations) to find the CH in all images. This property is a great advantage of ACHM over the CH algorithms discussed above. The advantage comes from the fact that Eq. (16) together with the SCC 24 make the ACHM capable of increasing the convergence speed (the time step) with the same ratio as the image size increases. This property states that the time of stable convergence (number of time iterations) is invariant with respect to the image size (Table 1). The CHs obtained by ACHM from the image 128×128 and 3072×3072 are shown in Figs. 5(d) and (h).

A conclusion one can derive from the above discussion is that *ACHT* is faster than *ConvexHull*, *Convex* and the other CH algorithms from the literaature, if they process large images, where the number of the points (pixels) to be enveloped is larger than the number of the background points (pixels). In the other cases, the conventional *CH* methods may be faster than *ACHT*. The conclusion holds because the *NAO* taken by *ACHT* depends on the size of the VF, and the number of time iterations. In other words, the run time depends on the distance traveled by the active CH. Roughly speaking the NAO taken by ACHT to determine the CH depends on the background. On the other hand, the NAO taken by the above listed conventional algorithms depend on the foreground, the number of points (pixels) to be enveloped by the CH.

Another ACHM's advantage over the conventional methods listed above is in the provided image segmentation to set of convex hulls, if multiple objects are present in the image (see Figs. 8 and 9). Unlike *shell-ACHM* algorithm the conventional methods will envelop the whole set of objects with a single CH.

The next advantage of ACHM over the conventional CH algorithms is that it could be used together with active contours to automatically determine the shape of objects' concavities [28]. To validate this concept an experiment was performed using the image shown in Fig. 2(a). To automatically detect object concavities we employed an active contour based on the geometric heat differential equation in combination with ACHM. The obtained results are shown in Fig. 10. The provided opportunity for automatic extraction of the objects' concavities is an important property with application to Image Database indexing.

Finally we would like to notice that the penalty function defined by Eq. (11) and implemented by Eq. (20), is designed to work with gray level images (see Figs. 1–4) and can be extended to noise cleaning. In this case, the information needed by the function is the gray level intervals, which describe the background, and the minimal sizes Δ_s and Δ_t of the objects under consideration.

A disadvantage ACHM possesses is that it can not be used to determine Delaunay Triangulations and Voronoi Diagrams of the set of foreground points as most of the conventional CH algorithms can do [3, 6, 8].

9. Further Directions

A $C{+}{+}$ version of *ACHT* is to be released. An approach capable of automatic concavities extraction is under-consideration [28]. A work is underway to

extend the penalty function and make it working with color images and noise cleaning. A model is under-development to extend the 2D *ACHM* to the 3D case using level sets.

Acknowledgments

The authors wish to thank the unknown referees for their general and detailed comments and suggestions, which helped very much to improve the presentation of this paper. Thanks to Dr. Richard Kreminski for the help provided in computing the truncation error and the creative discussions on arc length parameterization. Thanks also to Dr. John Neuberger for his discussions on the first version of the manuscript and for coding the initial ACHT notebook. Finally the authors thank Nona Sirakova for text corrections.

References

1. A.J. Abrantes and J.S. Marques, "A class of constrained clustering algorithms for objects boundary extraction," *IEEE Trans. on Image Processing*, Vol. 5, pp. 1507–1521, 1996.
2. F. Ansia et al., "Automatic 3D shape re-construction of bones using active nets based segmentation," *Proc. of the 15th Intl. Conf. on Pattern Recognition*, Vol. 1, pp. 486–489, 2000.
3. F. Aurenhammer, "Voronoi diagrams—A survey of a fundamental geometric data structures," *ACM Comput. Surv.*, Vol. 23, pp. 345–405, 1991.
4. J. Bentley, G.M. Faust, and F. Preparata, "Approximation algorithms for convex hulls," *Comm. ACM*, Vol. 25, pp. 64–68, 1982.
5. R. Burden, J.D. Faires, *Numerical Analysis,* 4th edition, PWS-KENT Publishing Company, US, 1993.
6. C. Bradford Barber, D. Dobkin, and H. Huhdanpaa, "The quickhull algorithm for convex hulls," *ACM Transactions on Mathematical Software*, Vol. 22, No. 4, pp. 469–483, 1996.
7. J. Boardman, "Automating spectral unmixing of AVIRIS data using convex geometry concepts," in *4th JPL Airborne Geoscience Workshop* (Washington, D.C.). JPL, Pasadena, Calif. 1993.
8. H. Bronnimann, J. Iacono, J. Katajainen, P. Morin, J. Morrison, and G. T. Toussaint, "Space-efficient planar convex hull algorithms," *Theoretical Computer Science*, 2003, in Press. http://cgm.cs.mcgill.ca/ godfried/publications/insitu.pdf
9. T. Chan, "Optimal output-sensitive convex hull algorithm in two and three dimensiions," *Discrete and Computational Geometry*, Vol. 18, pp. 433–454, 1996.
10. Y. Chan and S.Y. Kung, "A hierarchical algorithm for image retrieval by sketch," *IEEE Workshop on Multimedia Signal Processing*, pp. 564–569, 1997.
11. T.F. Chan, J. Shen, and L. Vese, "Variational PDE models in image processing," *Notices of the American Math Society*, Vol. 50-1, pp. 14–26, 2003.
12. V. Caselles, R. Kimmel, and G. Sapiro, "Geodesic active contours," *International Journal of Computer Vision*, Vol. 22-1, pp. 61–79, 1997.
13. V. Caselles, F. Catte, T. Coll, and F. Dibos, "A geodesic model for active contours," *Numerishe Mathematik*, Vol. 66, pp. 1–33, 1993.
14. K. Clarkson and P. Shor, "Applications of random sampling in computational geometry," *Discr. Comput. Geom*, Vol. 4, pp. 387–421, 1989.
15. L.D. Cohen and I. Cohen, "Finite element methods for active contour models and balloons for 2D and 3D images," *IEEE Trans. Pattern Anallysis Macine Intell.*, Vol. 15, pp. 1131–1147, 1993.
16. C. Davatzikos and J.L. Prince, "An active contour model foor mapping the cortex," *IEEE Trans. Med. Imaging*, Vol. 14, pp. 65–80, 1995.
17. R.L. Graham, "An efficient algorithm for determining the convex hull of a finite planar set," *Info. Proc. Lett.*, Vol. 1, pp. 132–133, 1972.
18. "http : //forums.wolfram.com/mathgroup/archive/2005/Mar/msg00799.html,"
19. R. Jain, "World-Wide Maze," *IEEE Multimedia*, Vol. 2–3, 1995.
20. S. Kichenassamy, A. Kumar, P. Olver, A. Tannenbaum, and A. Yezzi, "Gradient vector flows and geometric active contour models," *ICCV* 1995, pp. 810–815.
21. B.B. Kimia, A. Tannenbaum, and S.W. Zucker, "Shapes, shocks and deformations," *Int. J of Computer Vision*, Vol. 15, pp. 189–224, 1995.
22. R. Kimmel and G. Sapiro, "Shortening three-dimensional curves via two-dimensional flows," *Int. J of Computers and Mathematics with Applications*, Vol. 29, pp. 49–62, 1995.
23. D.G. Kirkpatrick and R. Siebel, "The ultimate planar convex hull algorithm?," *SIAM Journal of Computing*, Vol. 15, No. 1, pp. 287–299, 1986.
24. J. Lisani et al., "Affine Invariant mathematical morphology applied to a generic shape recognition algorithm," *Mathematical Morphology and its App. to Image and Signal Processing*, J. Goutsias, L. Vincent, D.S. Bloomberg (Eds.), Kluwer Pub. 2000, pp. 91–98.
25. F. Mlsna and N.M. Sirakov, "An intelligent shape features extraction and indexing system for fast medical image retrieval," *IEEE Comp. Society*, US, pp. 172–176, 2004. ISBN:0-7803-8387-7.
26. J. ORourke, *Computational Geometry in C*, Cambridge University Press, US, 1998.
27. N.M. Sirakov and P. Mlsna, "Search space partitioning using convex hull and concavity features for fast medical image retrieval," *IEEE ISBI'04*, April 15–18 2004, Arlington, pp. 796–799, 2004.
28. N.M. Sirakov and I. Simonelli, *A New Automatic Concavities Extraction Model*, Published by IEEE, Computer Society, 2006, USA, pp. 178–182. ISBN: 1-4244-0069-4, Library of Congress N: 2005937121.
29. G. Shapiro, *Geometric Partial Differential Equation and Image Processing,* Cambridge Univ. Press, 2001.
30. J.A. Sethian, "A review of recent numerical algorithms for hypersurfaces moving with curvature dependent flows," *J. Differential Geometry*, Vol. 31, pp. 131–161, 1989.
31. D. Sunday, http://geometryalgorithms.com/Archive/al-gorithm_0109/algorithm_0109.htm, 2001.
32. C. Xu and J.L. Prince, "Snakes, shaps and gradient vector flow," *IEEE Transaction on Image Processing*, Vol. 7, No. 3, pp. 359–369, 1998.
33. L.A. Vese and T.F. Chan, "A multiphase level set framework for image segmentation using the mumford and shah model," *International Journal of Computer Vision*, Vol. 50, No. 3, pp. 271–293, 2002.
34. G. Zamora, H. Sari-Sarraf, and S. Mitra, "Estimation of orientation of cervical vertebrae for segmentation with active shape models," *Proc. SPIE Medical Imaging: Image Proc.*, San Diego, CA, Feb. 17–23, 2001.

Nikolay M. Sirakov received B.S. degree from School of Mathematics and Informatics-Sofia University in 1982, M.S. degree from the same University in the field of Coding Theory in 1983, and Ph.D. degree from Institute of Mechanics-Bulgarian Academy of Sciences in the field of 3D modeling and recognition in 1991.

He had research and teaching positions at the Institute of Mechanics (1984–2000—Associate Professor since 1999), International Lab of Artificial Intelligence-Slovak Academy of Sciences (1990), Instituto Superior Technico–CVRM-Portugal (1993, 1998–2000), and Northern Arizona University (2001–2004). Currently Dr. Sirakov is an Assistant Professor of Mathematics and Computer Science at Texas A&M University Commerce. His research interests fall in Active Contours/Surfaces Models to image segmentation and features extraction, 3D reconstruction and visualization, 2D/3D objects matching and classification, Image Processing and Analysis, and Content Based Image Retrieval. He published over seventy papers and was a co-author of two books in the above listed fields.