CPM: A Deformable Model for Shape Recovery and Segmentation Based on Charged Particles

Andrei C. Jalba, Michael H.F. Wilkinson, *Member*, *IEEE*, and
Jos B.T.M. Roerdink, *Senior Member*, *IEEE*

Abstract—A novel, physically motivated deformable model for shape recovery and segmentation is presented. The model, referred to as the charged-particle model (CPM), is inspired by classical electrodynamics and is based on a simulation of charged particles moving in an electrostatic field. The charges are attracted towards the contours of the objects of interest by an electrostatic field, whose sources are computed based on the gradient-magnitude image. The electric field plays the same role as the potential forces in the snake model, while internal interactions are modeled by repulsive Coulomb forces. We demonstrate the flexibility and potential of the model in a wide variety of settings: shape recovery using manual initialization, automatic segmentation, and skeleton computation. We perform a comparative analysis of the proposed model with the active contour model and show that specific problems of the latter are surmounted by our model. The model is easily extendable to 3D and copes well with noisy images.

Index Terms—Deformable model, charged-particle system, electrostatic field, Coulomb force, segmentation, shape recovery, skeleton.

1 Introduction

An important goal in computer vision is to recover the shapes of the objects of interest from visual data. Deformable models introduced by Kass et al. [1] and generalized to the 3D case by Terzopoulos et al. [2] have become extremely popular, offering a unified framework which combines knowledge from geometry, physics, and approximation theory [3]. They have been extensively used in shape recovery and medical imaging (see [3], [4] for recent surveys). Other applications range from geometric modeling [5] and computer animation [6] to texture segmentation [7] and object tracking [8], [9].

A popular deformable model is the snake model [1], describing a closed parametric curve that deforms dynamically and moves towards the desired image features under the influence of internal and external forces, appearing in an energy functional to be minimized. The internal forces keep the contour smooth, while the external forces attract the snake towards lines, edges, or other low-level image features. Amini et al. [10] pointed out some shortcomings of the original algorithm and proposed an improvement based on discrete dynamic programming. Although their approach is more stable and allows the inclusion of hard constraints in the energy functional, the method is time consuming and needs careful setting of parameters. A fast, greedy active contour algorithm was proposed by Williams and Shah [11]. Their method retains the improvements of Amini's algorithm, but is more than an order of magnitude faster. Leymarie and Levine [12] presented a detailed analysis of the snake model, emphasizing its limitations and shortcomings, and proposed an improved termination criterion and a method based on a discrete scale-space representation useful for object tracking. Grzeszczuk and Levin [13] controlled the evolution of the active contour by a simulated annealing process which causes the contour to settle into the global minimum of an image-derived energy function. Peterfreund [14] proposed the velocity snake for boundary tracking of nonrigid objects, by applying velocity control to the evolving contour. Ngoi and Jia [15] presented an active contour model for colour region extraction in natural scenes. Wong et al. [16] developed the segmented snake model and demonstrated its ability to handle objects with sharp corners. Their method involves a recursive split-and-merge procedure that divides a contour into segments, each of which defines the contour locally. In [17], an active contour model based on the B-spline representation and multiple-stage energy minimization was proposed. Xu and Prince [18] proposed a new external force, which they called gradient vector flow (GVF), and showed that GVF has a large capture range and is able to move the snake into boundary concavities. An accurate and high-speed active contour model based on a reformulation of the internal energy was introduced by Mokhtarian and Mohanna [19], by removing the curvature part and using curvature scale-space filtering for smoothing.

All aforementioned deformable models cannot handle topological changes of the underlying shapes. Several methods have been proposed to address this limitation. Malladi et al. [20] proposed a geometric formulation of the active contour model, based on level sets. As shown in [21], [4], this method suffers from boundary leakage in the vicinity of blurred edges. In an attempt to overcome this problem, Caselles et al. [21] reformulated the snake evolution as an optimization of the total gradient along the snake, which led to a curvature evolution implementation. Since the literature on geometric active contours and surfaces based on level sets is vast, we refer to [4] for a recent and exhaustive review. McInerney and Terzopoulos

• *The authors are with the Institute for Mathematics and Computing Science, University of Groningen, PO Box 800, 9700 AV Groningen, The Netherlands. E-mail: {andrei, michael, roe}@cs.rug.nl.*

Manuscript received 18 Dec. 2003; revised 30 Mar. 2004; accepted 1 Apr. 2004.
Recommended for acceptance by R. Basri.
For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number TPAMI-0436-1203.

[22] proposed a new class of deformable models by introducing an affine cell image decomposition. This mechanism induces an iterative reparameterization that enables parametric deformable surfaces to modify their topology. Vasilescu and Terzopoulos [23] used spring-mass models in the context of surface reconstruction.

A completely different approach to shape deformation is based on particle systems. Such systems have been introduced in computer graphics by Reeves [24] to model natural phenomena such as fire and waterfalls. In these models, particles move under the influence of force fields but do not interact with each other. Szeliski and Tonnensen [25] proposed a system of oriented, interacting particles to model deformable surfaces. Each particle is considered as a surface element and has an associated rotation matrix. The authors define coplanar and cocircular energies which tend to align the particles on the surface of a plane or a sphere, respectively. The interacting-particle system can be used to split, join, or extend surfaces without the need for manual intervention.

In this paper, we introduce a novel, physically motivated deformable model for shape recovery and segmentation based on charged particles. The charges are attracted towards the contours of the objects of interest by an electric field, whose sources are computed based on the gradient-magnitude image. The electric field plays the same role as the potential force in the snake model, while internal interactions are modeled by repulsive electrostatic forces, referred to as Coulomb forces. The method needs an initialization step, but we will show that this step is much less critical than in the snake model. Unlike the active contour model, our model allows charges to be placed entirely inside an object, outside on one side of the object, or can cross over parts of boundaries. In contrast to attractive forces based on the squared gradient-magnitude image [1] which act only in small vicinities along boundaries of objects, the electric field exhibits increased capture range because of its long range attraction, and enhanced robustness against boundary leakage. Due to the combined effect of external interactions of particles with the electrostatic field and internal repelling forces between them, particles follow paths along object boundaries and, hence, converge without difficulty into deep boundary concavities or internal boundaries separating embedded objects. Moreover, the method is insensitive to initialization and can adapt to topological changes of the underlying shape.

The remainder of this paper is organized as follows: In Section 2, we review the snake model and discuss variations and some inherent problems with this model. The charged-particle model (CPM) is introduced in Section 3. We give ample descriptions of all its constituent parts and implementation details in Section 4, which also contains a multiresolution setting useful for dealing with noisy gray-scale images. In Section 5, we present experimental results, in three different settings: 1) shape recovery, 2) automatic segmentation, and 3) skeleton computation. We include some initial 3D examples. Also, the behavior of the proposed model in the presence of noise and its sensitivity to parameter changes are analyzed. Finally, conclusions, current limitations and a discussion of possible extensions are given in Section 6.

2 The Active Contour Model

In this section, we briefly review the active contour model and include a discussion of the model variations and additions, in so far as relevant to this paper.

2.1 The Classical Snake

The traditional active contour model is a closed parametric curve $\boldsymbol{C}(s) = (x(s), y(s))$, $s \in [0,1]$, which can change shape and position in order to minimize the energy functional [1]

$$E = \oint_0^1 \mathrm{d}s \; \alpha E_{int}(\boldsymbol{C}(s)) + \beta E_{im}(\boldsymbol{C}(s)) + \gamma E_{con}(\boldsymbol{C}(s)), \quad (1)$$

with $E_{int}(\boldsymbol{C}(s)) = |\boldsymbol{C}'(s)|^2 + w|\boldsymbol{C}''(s)|^2$, where α, β, w, and γ are scalar weights. The movements of the snake are governed by: 1) its internal energy E_{int}, 2) the image energy E_{im}, which tends to move the snake towards image features, and 3) a user-defined energy term E_{con}, specifying some constraints. The internal energy term E_{int} ensures curve regularity and is computed as the sum of two terms. The first term is *tension* and acts as an elastic force which shortens the curve, while the second term—*stiffness*—enforces smoothly changing contours, and makes the snake behave like a thin plate. Here, a first difficulty occurs with respect to setting the parameter w of the snake. Clearly, too much tension makes the snake deform too easily, stopping at false edges, while too much stiffness makes the snake too rigid, preventing it to follow contours with large changes in curvature. Initially, Kass suggested for the image-energy term a potential based on the squared gradient-magnitude image [1], i.e., $E_{im} = -|G_\sigma * \nabla I|^2$, where G_σ is a Gaussian kernel with standard deviation σ and "*" denotes convolution.

2.2 Problems, Variations, and Additions

There are several problems related to the traditional snake model, which we briefly mention here. For a more extensive discussion, see [18], [21], [26], [27].

1. *Curve collapse*: In the absence of image energy, the curve tends to *collapse* [28]. One may add an internal pressure term forcing the contour to expand [29], but this requires the initial contour to be placed completely inside the target object. Moreover, as pointed out in [18], the snake may get outside the contour of the target object in places with weak response of the gradient operator (*boundary leakage*). Although the dynamic pressure models proposed in [30], [31] were designed to solve the latter problem, the first still remains.
2. *Sensitivity to initialization*: The initial contour should be close to the true boundary in order to be attracted. Several techniques have been proposed to rectify this problem, e.g., multiscale methods [32], [19], pressure forces and distance potentials [29], and, more recently, gradient vector flow (GVF) snakes [18]. Basically, the GVF approach replaces the classical external force with a vector field exercising strong forces near edges and also extends the gradient map further into homogeneous regions by a generalized diffusion process.

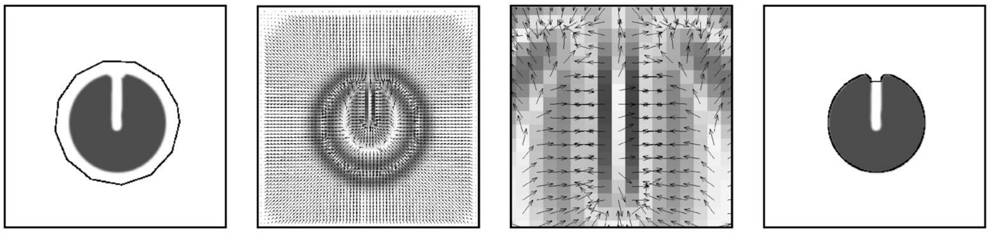


Fig. 1. Convergence of the GVF snake into boundary concavities. Left-to-right: initialization; the GVF field; zoom-in on the GVF field; result.

3. *Convergence into boundary concavities*: If the external force is weak, the snake is not pulled inside concavities and will stop advancing. Although there are several approaches [29], [33] which try to solve this problem, most of them do not give satisfactory results [18]. The GVF method was shown to remedy this problem to some extent [18], [34], but it still does not work well for thin and long boundary concavities. As the example of Fig. 1 shows, the GVF snake stops at the "entrance" of the concavity because there is no attractive force to pull the snake inside; see, especially, the zoom-in on the GVF field.
4. *Termination criterion*: The deformation is supposed to stop when the global energy minimum is reached. However, the snake can be trapped in local minima, and global minimization techniques such as simulated annealing [35] or dynamic programming [36] need to be used.
5. *Topological changes*: Splitting and merging of contours can be addressed by considering the active contour (or surface) as the zero level set of a higher-dimensional function, cf. Malladi et al. [20] and Caselles et al. [37]. The evolving curve $\boldsymbol{C}$ is embedded as the zero level set of a 2D scalar function $\phi(\boldsymbol{x}, t)$, satisfying the evolution equation

$$\frac{\partial \phi}{\partial t} = \hat{F}_i(x,y)(w_0 + w_1 \kappa(x,y))|\nabla \phi|, \qquad (2)$$

where w_0 and w_1 are constants, κ is the level set curvature and $\hat{F}(x,y)$ is an extension of $F(x,y) = (1 + |\nabla G_\sigma * I(x,y)|)^{-m}$ to all level sets. This approach works well for objects with steady edges, but otherwise it suffers from boundary leakage. This happens because the multiplicative term only slows down the evolving contour near edges, without stopping it completely. To overcome this problem, Caselles et al. [21] introduced geodesic active contours by adding an extra stopping term which is meant to pull back the contour if it passes the boundary. However, as shown by Siddiqi et al. [26], this formulation is still affected by the boundary-leakage problem, which was further addressed by adding an "area minimizing term," providing an additional attractive force when the evolving front is in the vicinity of an edge. As noted by the authors, this active contour can still leak through larger boundary gaps.

Xu et al. [27] formulated their parametric GVF active contours in the level set framework, in an attempt to combine the advantages of both methods: increased capture range, ability to converge into boundary concavities, increased robustness against boundary leakage conferred by the GVF field, and automatic topology adaptation offered by the level set method. The result was the geometric GVF active contour, formulated as

$$\frac{\partial \phi}{\partial t} = w\kappa|\nabla \phi| - \boldsymbol{g} \cdot \nabla \phi, \qquad (3)$$

where w is a constant and $\boldsymbol{g}$ is the GVF field. Although the GVF field does increase the robustness with respect to the initial placement of the contour, and bridges boundary gaps, there are cases when changes in topology cannot be handled correctly. For example, in the central region separating the objects shown in Fig. 2, the GVF field has direction tangential to the zero level set, and in such a case there is no force to push or pull the level set. If a constant pressure term (similar to w_0 in (2)) is added to the model, the topological problem can be solved, but the level set advances into boundary gaps (see the example shown in Fig. 2b).

Despite all improvements mentioned above, several difficulties with the geometric active contour model still remain. As mentioned in [4], level sets remove one of the major drawbacks of parametric snakes, i.e., sensitivity to initialization, provided that the initial contour is *symmetrically initialized* with respect to the boundaries of the object of interest. Symmetric initialization ensures that the evolving contour reaches all object boundaries almost at the same time. However, if this is not the case and the contour is initialized closer to one part of the boundary than the others, the contour crosses over the first portion of the object boundary. This happens because the stopping factor ($F(x,y)$ in (2)) is small but nonzero. If this factor is made smaller, i.e., by choosing a larger value of m, then the level set will stop at noisy edges since the noise would be amplified. On the contrary, if m is low, weak boundaries are not detected well and they will be crossed over by the evolving contour, see Fig. 3.

Internal boundaries separating *embedded objects* are not captured by a single active contour [20], [4]. In these cases, one needs multiple initializations, with at least one expanding active contour per object.

Thus, we conclude that in the initialization phase, extensive user interaction is required, while the most serious drawback of level set methods is contour leakage through boundary gaps, a problem which arises especially in realistic imagery, such as in ultrasound, MR, and CT images.

3 The Charged-Particle Model (CPM)

In this section, we introduce a physically motivated deformable model for shape recovery based on charged particles. The model is inspired by classical electrodynamics

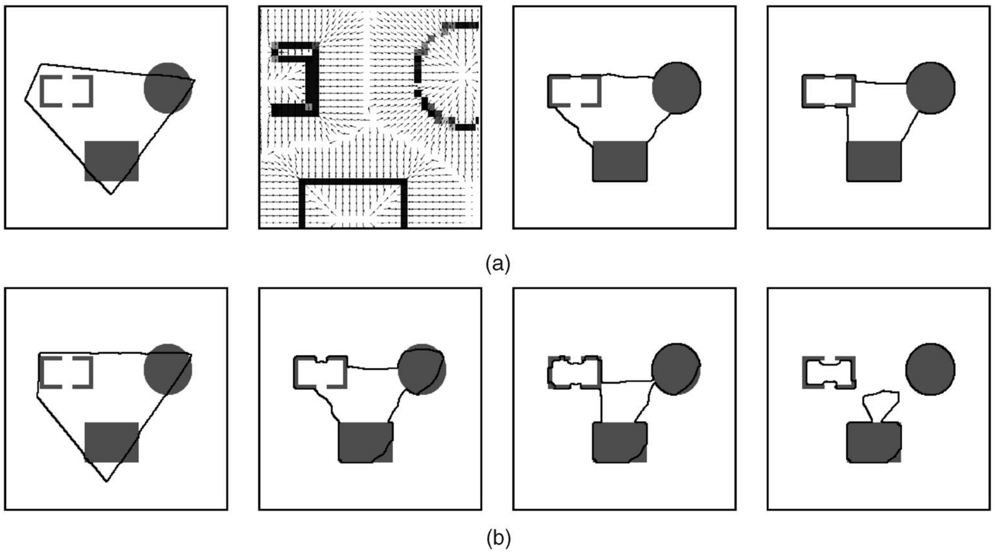


Fig. 2. (a) The geometric GVF snake does not handle topological changes correctly; left-to-right: initialization; zoom-in on the GVF field; snapshots. (b) With a pressure term the topological problem can be solved, but the boundary-leakage problem reappears; left-to-right: initialization; successive snapshots.

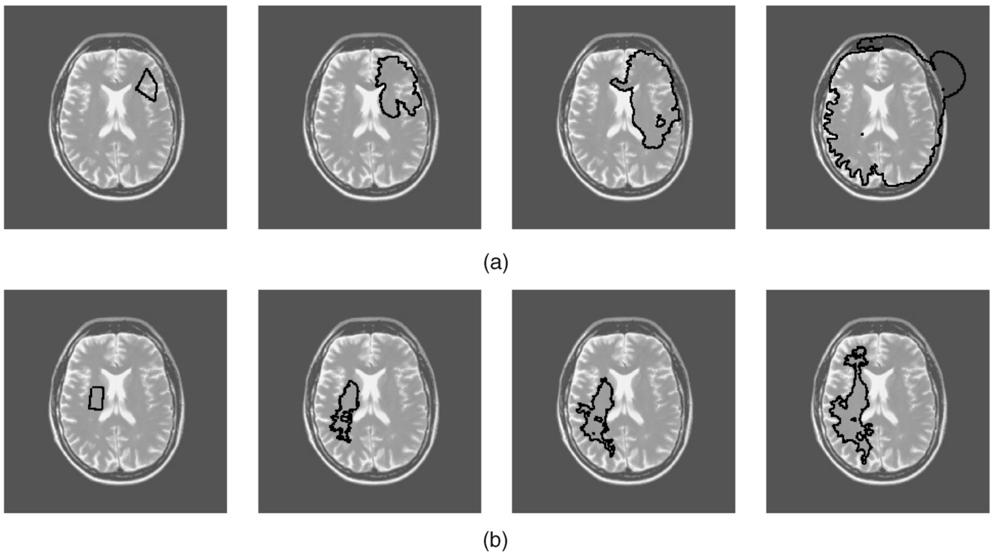


Fig. 3. Geodesic snakes: difficulty of choosing an appropriate stopping factor. (a) For small m, weak boundaries are not detected well. (b) For large m, the level set is stopped at noisy edges.

and is based on a simulation of charged particles moving in an electrostatic field.

3.1 Formulation

Consider a system of N positively charged particles p_i with electric charges q_i, $i = 1 \dots N$. These charges freely move in an external electrostatic field $\boldsymbol{E}$, generated by fixed, negative charges placed at each pixel position of the input image with charge magnitude proportional to the edge-map of the input image. Each free particle q_i moves under the influence of two forces: 1) internal Coulomb force, $\boldsymbol{F}_c$, due to the interaction of the particle with other free particles, and 2) external Lorentz force, $\boldsymbol{F}_l$, due to the electric field generated by the fixed charges e_i, see Fig. 4.

The resulting force $\boldsymbol{F}$ acting on a particle p_i located at position vector $\boldsymbol{r}_i = [x_i, y_i]$ is

$$\boldsymbol{F}(\boldsymbol{r}_i) = \boldsymbol{F}_c(\boldsymbol{r}_i) + \boldsymbol{F}_l(\boldsymbol{r}_i), \tag{4}$$

where $\boldsymbol{F}_c$ is the Coulomb force, and $\boldsymbol{F}_l$ is the Lorentz force acting on particle p_i with charge q_i given by

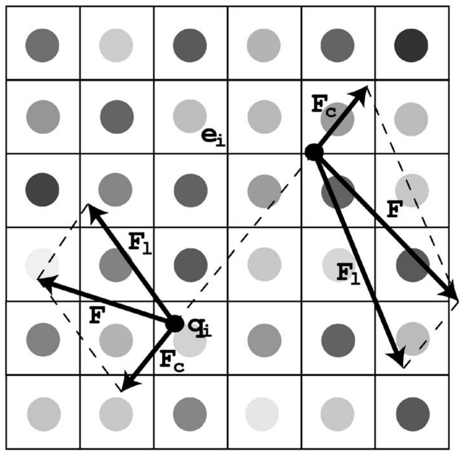
$$\boldsymbol{F}_l(\boldsymbol{r}_i) = q_i(\boldsymbol{E}(\boldsymbol{r}_i) + \boldsymbol{v}_i \times \boldsymbol{B}(\boldsymbol{r}_i)). \tag{5}$$

Here, $\boldsymbol{v}_i$ is the speed of the particle and $\boldsymbol{B}(\boldsymbol{r}_i)$ is the magnetic field. In the absence of a magnetic field ($\boldsymbol{B} = 0$) the force becomes

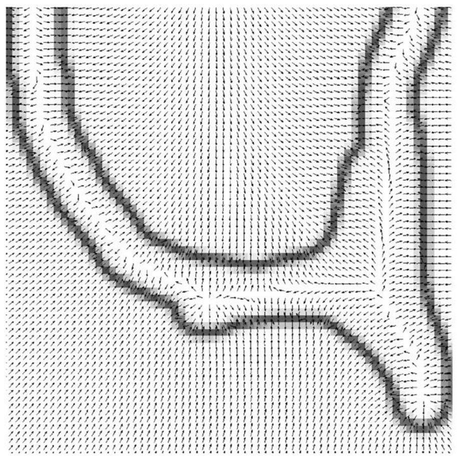
$$\boldsymbol{F}_l(\boldsymbol{r}_i) = q_i\boldsymbol{E}(\boldsymbol{r}_i) \tag{6}$$

and has direction parallel to that of the electric field $\boldsymbol{E}$. Note that although a magnetic field arises at particle position $\boldsymbol{r}_i$ due to all other moving particles, we neglected this field in our formulation because we are only interested in the equilibrium positions.

Now, assume that at each pixel position $\boldsymbol{R}_k, k = 1, \dots, M$ of the edge-map function $f(x, y)$ given by $|\nabla(G_\sigma(x, y) * I(x, y))|$ of an image I of size M pixels, a fixed negative electric



(a)



(b)

Fig. 4. The CPM. (a) Forces act on free particles with charge q_i (indicated by small black dots) which move in the electric field generated by fixed charges e_i (indicated by gray dots); different gray values represent different charge magnitudes. (b) Example of electrostatic field E generated by fixed charges.

charge $e_i = -f(x_i, y_i) \leq 0$ is placed. By the superposition principle, the electric field at position $\boldsymbol{r}$ is given by

$$E(\boldsymbol{r}) = -\nabla\phi_0(\boldsymbol{r}) = -\nabla\left(\sum_{k:\boldsymbol{R}_k\neq\boldsymbol{r}}^{M} \frac{e_k}{4\pi\epsilon_0}\frac{1}{|\boldsymbol{r}-\boldsymbol{R}_k|}\right), \tag{7}$$

where $\phi_0(\boldsymbol{r})$ is the electric potential at position $\boldsymbol{r}$, and ϵ_0 is the electric permittivity of free space. Notice that, if the position $\boldsymbol{r}$ coincides with a grid position, say $\boldsymbol{R}_i$, then this position is excluded in the summation. Evaluating the gradient in (7) and substituting $E(\boldsymbol{r}_i)$ in (6), it follows that

$$\boldsymbol{F}_l(\boldsymbol{r}_i) = q_i \sum_{k:\boldsymbol{R}_k\neq\boldsymbol{r}_i}^{M} \frac{e_k}{4\pi\epsilon_0}\frac{\boldsymbol{r}_i-\boldsymbol{R}_k}{|\boldsymbol{r}_i-\boldsymbol{R}_k|^3}. \tag{8}$$

The Coulomb force acting on particle p_i with charge q_i is the sum of the Coulomb forces generated by all other free particles taken in isolation, i.e.,

$$\boldsymbol{F}_c(\boldsymbol{r}_i) = q_i \sum_{j\neq i}^{N} \frac{q_j}{4\pi\epsilon_0}\frac{\boldsymbol{r}_i-\boldsymbol{r}_j}{|\boldsymbol{r}_i-\boldsymbol{r}_j|^3}. \tag{9}$$

Assuming that all free particles have the same positive charge $q_i = q$, (4) can be rewritten as

$$\boldsymbol{F}(\boldsymbol{r}_i) = k\left(q^2\sum_{j\neq i}^{N}\frac{\boldsymbol{r}_i-\boldsymbol{r}_j}{|\boldsymbol{r}_i-\boldsymbol{r}_j|^3} + q\sum_{k:\boldsymbol{R}_k\neq\boldsymbol{r}_i}^{M} e_k\frac{\boldsymbol{r}_i-\boldsymbol{R}_k}{|\boldsymbol{r}_i-\boldsymbol{R}_k|^3}\right), \tag{10}$$

with $k = \frac{1}{4\pi\epsilon_0}$. The major difference between the two terms in (10) is that the Lorentz force reflects particle-mesh or external attractive interactions and is computed in the image domain, while the Coulomb force represents particle-particle or internal repelling interactions. Since the distribution of fixed charges e_i reflects the strength of the edge map and the electric force is "inverse-square," i.e., it decays with the squared distance, the electrostatic field has large values near edges and small values in homogeneous regions of the objects present in the input image.

In the implementation, which is discussed in more detail below, we precompute the field E at the grid positions $\boldsymbol{R}_k$, and then interpolate between the field values at these positions in order to obtain the field at any position $\boldsymbol{r}_i$ of a freely moving particle p_i (see Sections 3.2 and 4).

The total energy of the system is the summation of all particle energies, i.e.,

$$E_p(\boldsymbol{r}_1,\ldots,\boldsymbol{r}_N) = \frac{1}{2}\sum_{i=1}^{N}\left(w_1\sum_{j\neq i}^{N}\frac{1}{|\boldsymbol{r}_i-\boldsymbol{r}_j|} - w_2\sum_{k:\boldsymbol{R}_k\neq\boldsymbol{r}_i}^{M}\frac{e_k}{|\boldsymbol{r}_i-\boldsymbol{R}_k|}\right), \tag{11}$$

where $w_1 = kq^2$ and $w_2 = kq$ are weights.

3.2 Particle Dynamics

Having defined the energy associated with our system, we can derive its equations of motion. The variations of particle potentials with respect to positions produce forces acting on particles. The standard approach is to consider the Newtonian equations of motion, and to integrate the corresponding system of differential equations in time, i.e.,

$$\boldsymbol{F}(\boldsymbol{r}_i) = w_1\boldsymbol{F}_c(\boldsymbol{r}_i) + w_2\boldsymbol{F}_l(\boldsymbol{r}_i) - \beta\boldsymbol{v}_i \tag{12}$$

$$\begin{cases} \frac{d\boldsymbol{r}_i(t)}{dt} = \boldsymbol{v}_i \\ \frac{d\boldsymbol{v}_i(t)}{dt} = -\nabla\phi(\boldsymbol{r})|_{\boldsymbol{r}=\boldsymbol{r}_i} = \boldsymbol{F}(\boldsymbol{r}_i(t)), \end{cases} \tag{13}$$

where ϕ is the particle potential and $\boldsymbol{r}_i$, $\boldsymbol{v}_i$, and $\boldsymbol{a}_i$ are its position, speed, and acceleration, respectively (the mass of the particle has been set to 1). Notice that compared to (4), (12) has an additional term, $\boldsymbol{F}_{damp}(\boldsymbol{r}_i) = -\beta\boldsymbol{v}_i$, the damping (or viscous) force which is required for the particles to attain a stable equilibrium state, which minimizes their potential energies, see (11). The problem with integrating (13) is the circular dependency between $\boldsymbol{r}_i(t)$ and $\boldsymbol{F}(\boldsymbol{r}_i(t))$, which restricts the methods for numerical integration which can be used. However, even though the force $\boldsymbol{F}(\boldsymbol{r}_i(t))$ is unknown, at least its first few derivatives are easily found, which means that Taylor series can be used to approximate the function.

In the interest of simplicity, all particles are advanced using the same time-step parameter Δt. Therefore, at each time step $t_{j+1} = t_j + \Delta t$ the force $\boldsymbol{F}(\boldsymbol{r}_i(t))$ acting on each particle p_i is computed according to (12), in order to obtain the acceleration $\boldsymbol{a}_i$ of the particle. Then, one can use Euler's method [38] to advance the current velocity and position of the particle over the time step. Unfortunately, this method may result in large errors, which accumulate in time.

Therefore, we have developed an efficient alternative method based on Taylor expansion which makes use of previously computed values for acceleration and speed (i.e., a multistep method). It advances the current velocity and position over one time step as follows:

$$\begin{aligned} \boldsymbol{r}(t+\Delta t) &= \boldsymbol{r}(t) + \Delta t\ \boldsymbol{v}(t) + \frac{1}{2}\Delta t^2\ \boldsymbol{a}(t) + \frac{1}{6}\Delta t^3\ \frac{d\boldsymbol{a}(t)}{dt} \\ \boldsymbol{v}(t+\Delta t) &= \boldsymbol{v}(t) + \Delta t\ \boldsymbol{a}(t) + \frac{1}{2}\Delta t^2\ \frac{d\boldsymbol{a}(t)}{dt}. \end{aligned} \tag{14}$$

Since the term in $d\boldsymbol{a}(t)/dt$ is expensive to compute, we approximate it using (quadratic) Lagrange polynomials $\boldsymbol{L}(t)$, at times $\{t_i, t_i - \Delta t, t_i - 2\Delta t\}$. Taking the derivative of the polynomials, and evaluating at $t = t_i$, the desired term is obtained as

$$\begin{aligned} &\left.\frac{d\boldsymbol{a}(t)}{dt}\right|_{t=t_i} = \left.\frac{d\boldsymbol{L}(t)}{dt}\right|_{t=t_i} = \\ &\frac{1}{2\Delta t}(3\boldsymbol{a}(t_i) - 4\boldsymbol{a}(t_i - \Delta t) + \boldsymbol{a}(t_i - 2\Delta t)). \end{aligned} \tag{15}$$

This computation is fast and stable with respect to one of the worst form of errors occurring whenever a particle is very close to an object boundary. Since we use constant time steps, it may happen that at the next time step the particle would cross over the edge, where it is in fact supposed to stop. Using the above method, the acceleration of the particle reverses sign in the vicinity of the edge and, therefore, the particle eventually stops at the edge, after a few small oscillations.

The more accurate Runge-Kutta method [38] could be used, resulting in better convergence and larger time steps, but at the expense of increased computational cost.

3.3 Comparative Analysis of CPM and Snake Model

The two energy functionals in (1) and (11), corresponding to the snake model and to the CPM, respectively, have terms which represent internal and external interactions. In the snake model, the internal energy is the sum of a tension term, which acts as an elastic force and a rigidity term, which discourages bending. Tension evenly spaces snaxels along the snake contour.

In the CPM, the Coulomb force plays the same role, as we will show next. Free particles move down gradients of the potential energy in order to attain a minimum potential energy state. Since both the electrostatic field $\boldsymbol{E}$ and Coulomb force are conservative, each particle will move in the direction that most reduces its potential energy. Since we set $w_2 > w_1$ and both forces in (12) are normalized, the movements of the particles are primarily towards positions of minimum potential energy due to the field $\boldsymbol{E}$.

The orientation of the electric field $\boldsymbol{E}$ at a point corresponds to the direction in which the electric potential decreases most rapidly, and a positive charge placed at that point will be accelerated in this direction. In our case, negative charges e_i proportional to the edge-map function generate the field and, therefore, its direction will be toward the nearest edge with the highest magnitude. This implies that the free particles will be attracted towards these locations and eventually stop there, after few possible oscillations (see the discussion under (15)). Next, the particles will *advance along the boundary* of the object, in an attempt to minimize their potential energy due to the repelling Coulomb force between the free particles. Hence, they will finally occupy evenly spaced positions along the boundary. In our opinion, the motion capability of particles along boundaries of objects constitutes an important advantage over the snake model. This is the reason why the particles are able to converge into small and thin concavities, and are much less sensitive to the initial positioning, unlike most snakes (see Sections 2.2 and 5).

An example is shown in Fig. 5. Charged particles are initialized by sampling the circle shown in the bottom-right part of the first image. The attractive electric field $\boldsymbol{E}$ is shown in the second image. All other images are snapshots obtained at time steps $10, 20, \dots 60$ of the numerical integration. Note that the particles are first attracted towards the boundary of the "U-shaped" object. As soon as they arrive at the contour, they start advancing along the boundary, until they reach evenly spaced positions (see the last image). Note that some particles may depart outside the image space and, in this case, they are simply removed from subsequent computations.

The second term (stiffness) in the snake model serves to maintain smoothly changing contours. However, as shown in [28], curvature minimization provides smoothness by flattening, and it conflicts with the intention to segment nonrectangular objects. The CPM does not have a similar term. This means that, in order to achieve a larger degree of smoothing than that conferred by evenly spaced particles, other methods must be invoked (e.g., Gaussian filtering of the input image, spline interpolation).

One problem remains to be addressed before we can use the CPM for shape detection. Up to now, the model consists of N disconnected particles which are able to advance along the boundary of the objects of interest. However, the output of the method should consist of connected contours of the target objects. We will address this problem in the next subsection.

3.4 Curve (Surface) Reconstruction

So far, our particle system does not provide us with explicit representations of object boundaries. This problem can be thought of as that of curve or surface reconstruction from unorganized points. This is an important problem studied now for many years. Early efforts relied on heuristics and assumed that the input points are sampled uniformly from the unknown surface. Major breakthroughs were achieved by Hoppe et al. [39] and later by Amenta et al. [40] ([41] for curve reconstruction), who designed algorithms that guarantee correct reconstruction even with nonuniform (but sufficiently dense) samples, and which require the selection of a single parameter–the sample density.

In all experiments reported below, we use the algorithms by Amenta et al. [41], [40] to reconstruct the curves and surfaces.

3.5 Controlling the Complexity of the CPM

The naive evaluation of the Coulomb force $\boldsymbol{F}_c$ in (10) at all particle positions requires $O(N^2)$ operations, where N is the

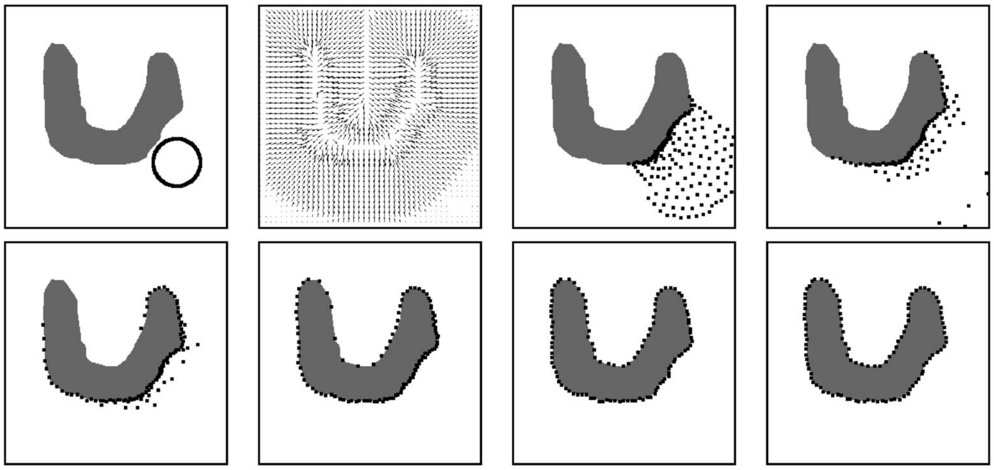


Fig. 5. Convergence of the particles. Left-to-right, top-to-bottom: input image and initialization; electric field $\boldsymbol{E}$; snapshots at time steps $t = 10, 20, \dots 60$.

number of free charged particles. Moreover, a straightforward computation of the electric field $\boldsymbol{E}$ is of order $O(M^2)$, where M is the number of pixels in the input image. For large input images (or volumes), which usually require a large number of particles, the force evaluation in (10) can become prohibitively expensive. Therefore, we describe efficient methods used to compute all constituent parts of the model.

3.5.1 Computation of the Coulomb Force

An efficient approach to approximate the Coulomb force between free particles is to use a *k-d tree* [42] data structure to partition space, such that responses to queries about the k nearest neighbors of a particle can be obtained in logarithmic time. At each time step, after the tree is constructed, the Coulomb force at each particle position is approximated using its $k = 10$ percent nearest particles. In this way, the complexity of force evaluation is reducible to $O(N \log N)$.

3.5.2 Computation of the Electric Field

The electrostatic field (see (7)) is computed using the so-called "particle-particle particle-mesh" (PPPM) method from molecular dynamics [43]. The field is precomputed, and its values are kept in two matrices (with the same sizes as the input image I), one for each component of the 2D vector field.

The basic idea of PPPM is to split the Coulomb potential representing the force between the fixed charges into a short range direct interaction part (PP) and a contribution from the mesh (PM). The Coulomb energy, at vector position $\boldsymbol{R}_i = [x_i, y_i]$ is

$$W_i = \frac{1}{2}\sum_{j \neq i}^{M} W_{ij}^{direct} + W_i^{mesh}. \tag{16}$$

The direct part of the Coulomb energy of a pair of charges (e_i, e_j) separated by $\boldsymbol{R}_{ij}$ is given by the Coulomb energy minus a correction term

$$W_{ij}^{direct}(\boldsymbol{R}_{ij}) = \begin{cases} e_i e_j \left(\frac{1 - W^c(\boldsymbol{R}_{ij})}{|\boldsymbol{R}_{ij}|} \right) & |\boldsymbol{R}_{ij}| \leq R_c \\ 0 & |\boldsymbol{R}_{ij}| \geq R_c, \end{cases} \tag{17}$$

where R_c is the direct interaction cut-off radius. The direct contribution vanishes at R_c and, for $|\boldsymbol{R}_{ij}| \geq R_c$, there is only a mesh contribution. The correction term $W^c(\boldsymbol{R})$ compensates for the portion of the interaction already covered by the mesh potential (see (22) below).

The mesh potential ϕ^{mesh} is obtained by solving Poisson's equation on the grid

$$\nabla^2 \phi(\boldsymbol{R}) = -\rho(\boldsymbol{R}), \tag{18}$$

where $\rho(\boldsymbol{R})$ and $\phi(\boldsymbol{R})$ are the charge density and the electric scalar potential at grid point $\boldsymbol{R}$, respectively. The charge density $\rho(\boldsymbol{R})$ is defined as the charge per grid cell area, and is computed in two steps. In the first step, we use the linear charge assignment scheme in [43]. Every charge e_i is distributed over its eight surrounding grid points, and the charge at grid point $\boldsymbol{R}_i$ is computed as

$$q(\boldsymbol{R}_i) = \sum_{k=1}^{M} e_k H(\boldsymbol{R}_i - \boldsymbol{R}_k). \tag{19}$$

Here, $H = H_x H_y$ is the weight of a charge located at $\boldsymbol{R}$,

$$H_x(R_{i,x} - R_x) = \begin{cases} 1 - \frac{|R_{i,x} - R_x|}{h_x} & |R_{i,x} - R_x| < h_x \\ 0 & |R_{i,x} - R_x| \geq h_x, \end{cases}$$

where M is the size of the input image I, h_x is the mesh grid spacing in dimension x, and H_y is defined similarly. In the second step, the charges are spread over a larger neighborhood of grid points, in order to produce a smooth total charge distribution. This step is implemented using the approach in [44], in which the charges are spread by a diffusion process. The method proceeds by solving Poisson's equation, (18), on the mesh. Then, the mesh-energy term is computed as a weighted sum over the same grid points used in the first charge-assignment step

$$W_i^{mesh} = e_i \sum_k H(\boldsymbol{R}_i - \boldsymbol{R}_k)\phi(\boldsymbol{R}_k) - W_{Gauss,i}^{self}, \tag{20}$$

where $W_{Gauss,i}^{self}$ is a correction term for the mesh energy which a particle experiences from its own charge distribution (the self-energy). This constant term per particle is given by

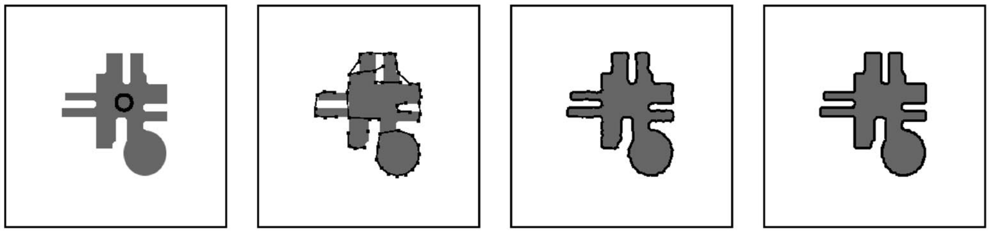


Fig. 6. Left-to-right: input image and initialization; incorrectly recovered shape from 58 particles without particle addition; correct shape recovery using dynamic addition and deletion of particles; correct shape recovery (with particle addition and deletion) with only five particles used for initialization.

$$W^{self}_{Gauss,i} = e_i^2/\sigma, \tag{21}$$

where $\sigma = 2\sqrt{DN_t}$, D is the diffusion coefficient, and N_t is the number of time steps of the diffusion process (N_t and D can be deduced using the cut-off radius parameter R_c, see [44] for details). For a Gaussian charge distribution, obtained by solving a linear diffusion equation, the correction term $W^c(\boldsymbol{R})$ in (17) is given by

$$W^c_{Gauss}(\boldsymbol{R}) = \boldsymbol{E}(\boldsymbol{R}/\sigma), \tag{22}$$

where "$\boldsymbol{E}$" denotes the error function. Note that we have neglected all constants in the description of the PPPM method.

Finally, the electric field $\boldsymbol{E}$ at grid points is computed by substituting the electric potential (given by $\phi(\boldsymbol{R}_i) = W(\boldsymbol{R}_k)/e_k$, $k = 1, 2, \ldots, M$) in (7). In order to obtain the field at any position $\boldsymbol{r}_i$ of a freely moving particle p_i, interpolation between the values at these grid positions is used.

In our implementation, the cut-off radius R_c is set to 20. We compute an approximate solution to the Poisson's equation using $N_p = 50$ iterations of a standard successive over-relaxation (SOR) technique [38]. Hence, the total time taken for computing the electric force is linear in the number M of pixels of the input image.

3.6 Dynamic Particle Creation and Removal

Our particle system can be used to model a wide variety of shapes, of arbitrary topology, provided that the number of free particles in the initialization phase is sufficient. Otherwise, if the number of particles is too small, a rough and/or incorrect approximation of the boundary is found. An example of this behavior is shown in Fig. 6. The first image shows the initialization (58 particles are created by sampling the circle shown in the middle of the object), while the next image shows the (incorrect) recovered shape using the curve reconstruction algorithm by Amenta et al. [41].

This problem is addressed by enhancing the particle system with two heuristic rules controlling the addition and deletion of particles. The *addition rule* postulates that the particles are in a near-equilibrium state with respect to the Coulomb and external potentials and checks if any two neighboring particles have a large enough spacing between them to add a new particle. If the particles are at a distance d (in pixels) such that $d_{min} < d \leq d_{min} + d_{max,t}$, a new particle is created at the midpoint, with velocity and acceleration computed as the average of those of the two neighboring particles. The time-dependent parameter $d_{max,t}$ is a simulated-annealing term which is gradually reduced during evolution. The annealing schedule for this term lowers its value according to $d_{max,t+\Delta t} = a \cdot d_{max,t}$, where $d_{max,t}$ is the value at time t. The *deletion rule* simply checks if two neighboring particles are at a distance d such that $d < 1/2d_{min}$, and in this case one of the particles is deleted.

With these rules we can automatically recover correct boundaries of objects, under the assumption that the set of edge points of the input image does not have gaps larger than the maximum acceptable value for the sample-density parameter of the curve (surface) reconstruction methods (see Section 3.4). If this assumption fails, the reconstructed contours will have gaps. The third image in Fig. 6 shows the correctly recovered shape. The last image was obtained using only five particles in the initialization phase.

4 Implementation

4.1 The Pseudocode of the CPM

The pseudocode of the CPM algorithm is given in Algorithm 1; it consists of two parts: 1) computation of the Lorentz force, 2) particle dynamics and reconstruction. Without loss of generality, the algorithm assumes a 2D input signal, although it can be easily extended for volumetric data.

In Part 1, after regularization (convolution with a Gaussian kernel), the gradient map $[g_x, g_y]$ is computed (line 2) and passed as input to the PPPM method, which in turn outputs the potential ϕ at each location of the image. The electric field is given by the gradient of the electric scalar potential ϕ (see (7)), from which the Lorentz force is precomputed (see (6)), and its components are kept in two matrices, one for each component.

Algorithm 1 CPM pseudocode

```
Part 1: Computation of the electric field $E$
Input: $f$—a gray-scale image.
Output: $[F_{l,x}, F_{l,y}]$—a vector-valued image representing
        the Lorentz force.

1: $f' := \mathrm{GaussianFiltering}(f)$
2: $[g_x, g_y] := \mathrm{GradientMap}(f')$
3: $e := -\mathrm{Magnitude}(g_x, g_y)$
4: $\phi := \mathrm{PPPMPotential}(e)$
5: $[E_x, E_y] := -\mathrm{GradientMap}(\phi)$
6: $[F_{l,x}, F_{l,y}] := \mathrm{LorentzForce}(E_x, E_y)$

Part 2: Particle dynamics and reconstruction
Input: $[F_{l,x}, F_{l,y}]$.
Output: a set of contours.
1: Set parameters $t, \Delta t, a, d_{min}, d_{max,0}, w_1, w_2, \beta$
2: $[p_1, p_2, \ldots, p_N] := \mathrm{InitParticles}()$
```

```
3: repeat
4:   $[F_{c,x}, F_{c,y}] := \mathrm{EvaluateCoulombForce}(p_1, p_2, \ldots, p_N)$
5:   $[F_x, F_y] := w_1[F_{c,x}, F_{c,y}] + w_2[F_{l,x}, F_{l,y}]$
6:   $\mathrm{InsertAndDeleteParticles}(d_{min}, d_{max})$
7:   $\mathrm{UpdateParticles}(p_1, p_2, \ldots, p_N, F_x, F_y, \beta, \Delta t)$
8:   $d_{max} := a \cdot d_{max}$
9:   $t := t + \Delta t$
10: until $\mathrm{Convergence}(p_1, p_2, \ldots, p_N)$
11: $\mathrm{Reconstruction}(p_1, p_2, \ldots, p_N)$
```

In Part 2 of the algorithm, in the initialization phase (line 1), the parameters of the method are set. Throughout the paper we use the same values (unless stated otherwise): $\Delta t = 0.5$, $d_{min} = 2.0$, $d_{max,0} = 3.0$, $a = 0.99$, $w_1 = 1.6$, $w_2 = 1.75$, $\beta = 0.8$. The time-step parameter Δt is related to the convergence speed of the free particles. A small value of this parameter leads to slow convergence, while a large value improves the convergence speed. However, the parameter cannot be set to an arbitrarily large value because during time integration the particles may cross over edges (see Section 3.2). The next three parameters control the density of particles: d_{min} was chosen such that the smallest distance between particles is one pixel; $d_{max,0}$ and a were set such that they represent an acceptable tradeoff between increased computation time and high density of particles. The same observations made above for the time step parameter hold also for the parameters w_1 and w_2; in addition, as discussed in Section 3.3, the values of these parameters must be set such that $w_2 > w_1$. The damping factor β has a regularization role, and it turned out that suitable values are $\beta < w_1$; see also the discussion in Section 5.1.4. As a general remark, we observed in our experiments that the setting of all parameters is not critical and can be adapted easily in order to obtain the desired level of accuracy (i.e., resolution), see also Section 5.5.

After setting the parameters, the particles are initialized, either manually or automatically (see Section 5), and then their positions, velocities and accelerations are updated at each time step (in the *repeat-until* loop). The Coulomb force is evaluated as described in Section 3.5.1, and the resulting force $\boldsymbol{F}$ at each particle position is computed by summation of the Coulomb and Lorentz forces (line 5). The required field values at positions of free particles are obtained by interpolation between the precomputed field values at grid positions. Particle insertion and deletion (line 6) is performed as discussed in Section 3.6. (In our actual implementation, this step is performed during the evaluation of the Coulomb force, when the neighbors of each particle are available.) Then, the particles are advanced (line 7) according to (14), cf. Section 3.2. Particle convergence (line 10) is detected by computing the mean absolute difference at time step t_k between all vector positions of particles at the previous T time steps; this difference can be expressed as

$$\epsilon_{t_k} = \frac{1}{N}\left|\sum_{i=1}^{T}(-1)^i \sum_{j=1}^{N}\left|\boldsymbol{r}_j^{t_{k-i}}\right|\right|, \tag{23}$$

where $\boldsymbol{r}_j^{t_{k-i}}$ is the vector position at time step t_{k-i} of particle p_j; in our implementation $T = 4$ and $\epsilon = 0.001$. Then, the stopping criterion $T_s : \epsilon_{t_k} < \epsilon$ is evaluated and, if it holds, reconstruction is performed using the algorithms by Amenta et al. ([41] for curve reconstruction and [40] for surface reconstruction).

4.2 A Multiscale Setting

A multiscale approach for the CPM solves the problem at successive scales iteratively. First, the solution is found at a coarse scale, on a small image which needs only few particles. This solution is used as initialization at the next finer scale, and the process is repeated. By propagating the result from the coarsest scale to the finest scale, the final curve is obtained on the initial image without any loss of precision. The multiscale algorithm can be summarized as follows:

1. Build the pyramid of images from scale 0 (original image) to scale S (coarsest scale).
2. For s decreasing from $S - 1$ to 0, evolve the particles at scale s until convergence is achieved, and denote by C_s the solution (i.e., particle positions) obtained using the projection (rescaling) of C_{s+1} to level s as initialization.
3. Do reconstruction at scale $s = 0$.

For the purpose of simplicity we use a standard Gaussian pyramid [45] with three levels, although other pyramids (e.g., wavelet pyramids) may be used as well.

The main advantage of this approach is a reduction of the computing costs. Initial convergence leads to a rough estimation of the boundary and is achieved at a coarse scale, at which the number of particles is small. At finer scales, the particles are already close to the boundary, and the number of iterations needed to achieve convergence is expected to be small. Moreover, the multiscale setting greatly improves the behavior of the CPM in the presence of noise, as shown in Section 5.6.

5 Results

In this section, we show several results obtained using the CPM on binary and gray-scale images. Three different settings are considered: 1) shape recovery using manual initialization, 2) segmentation using automatic initialization, and 3) skeleton extraction. Also, we present initial 3D shape recovery and segmentation results. Note that all experiments were conducted using the multiscale approach (see Section 4.2).

5.1 Shape Recovery

All shape-recovery experiments were carried out using manual initializations. We will treat in each subsection a specific problem of the active contour model.

5.1.1 Initialization

In the first experiment, we present results obtained using the CPM with different initializations on binary images. Examples are shown in Fig. 7. Because of lack of space, we skipped intermediate snapshots and show only initializations and final results. The initializations in Fig. 7 are very difficult (if not impossible) to handle by most snakes because parts of the boundary are inside the initial snake contour, while others are outside. For the CPM, however, these initializations possess no problem and the silhouettes of all objects are recovered.

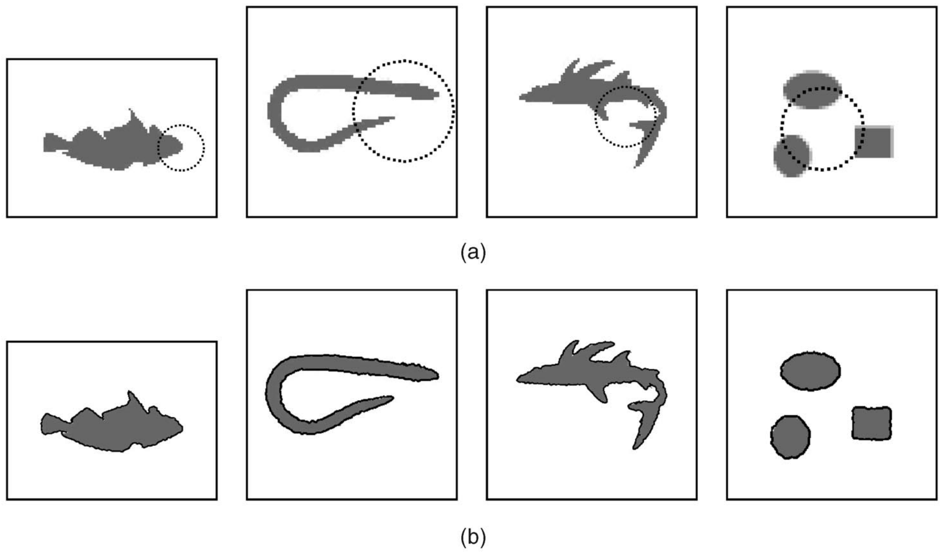


Fig. 7. Initialization robustness of the CPM. (a) input images and initializations; (b) reconstructed curves.

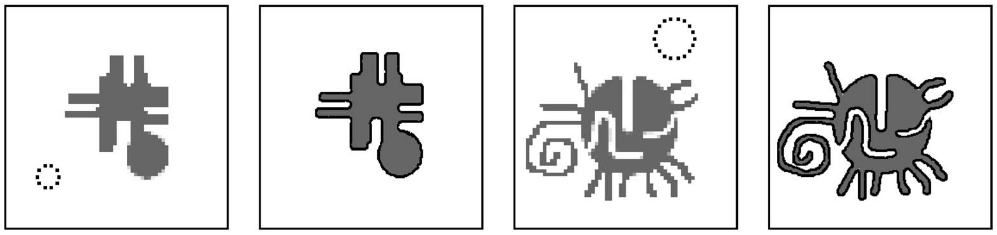


Fig. 8. Convergence into boundary concavities. Initializations and reconstructed curves for two complex objects.

5.1.2 Convergence into Boundary Concavities

In this experiment we study the convergence of the CPM into thin and long boundary concavities. The results are shown in Fig. 8. Even with a difficult initialization, the particles correctly recover the shapes of both objects. This happens because the particles can advance along boundaries of objects, even in thin and long concavities, due to the repelling force between them (see Section 3.1).

5.1.3 Embedded Objects and Topology Changes

An important characteristic of the CPM is that its particle system can adapt to the unknown topology. The free particles arrive at the boundaries of the objects in the image, regardless of the topologies of the objects. Moreover, embedded objects can be captured using a single initialization.

Some results are shown in Fig. 9. The first three examples in this figure were correctly segmented using only a single initialization. As shown in the last example, multiple initializations can also be performed. Note that the recovered contours are not always closed (see the first, third, and fourth examples), they may also form T-junctions (see the first example) and, in our opinion, this represents an advantage over active contours.

5.1.4 Boundary Leakage

This is not an issue for the CPM. If object boundaries have gaps, the edge strength is small and the particles will not converge to these locations. The output contours will be either open or closed, depending on the lengths of the gaps. In Fig. 10, we show results for three different values of the damping coefficient β. For $\beta = 0.6$, the particles cross over the border separating the two main regions of the input image, and all three objects are detected. When $\beta = 1.0$, the particles cross over boundaries with weak edge responses (i.e., the right part of the border), and are attracted to the boundary of the circle. For larger values of β, the particles converge only to the border separating the two regions and to the elliptical object near the initial set of particles. Hence, one can control the behavior of the particles according to the desired output. However, for automatic processing a parameter-free method is desirable, which our method is not.

Results obtained using the CPM for a number of additional input images are shown in Fig. 11. In the first row, the input images and the initializations are shown; the second row contains the resulting contours found by the method.

Although the first image (angiogram) is quite noisy and difficult (because of the thin and elongated structures), the CPM method yields quite a good result, being able to enclose the whole artery. The next images are difficult to segment even by advanced segmentation methods because of the texture present either in the background (first and third images) or in the foreground (the fourth image). Still, the CPM method behaves quite well, being able to output useful contours which enclose most parts of the objects present in the images.

5.2 Automatic Segmentation

Our next experiment is segmentation using (trivial) automatic initializations. Note that the values of some parameters of the method were decreased ($w_1 = 0.6, w_2 = 0.7, \beta = 0.4$), in

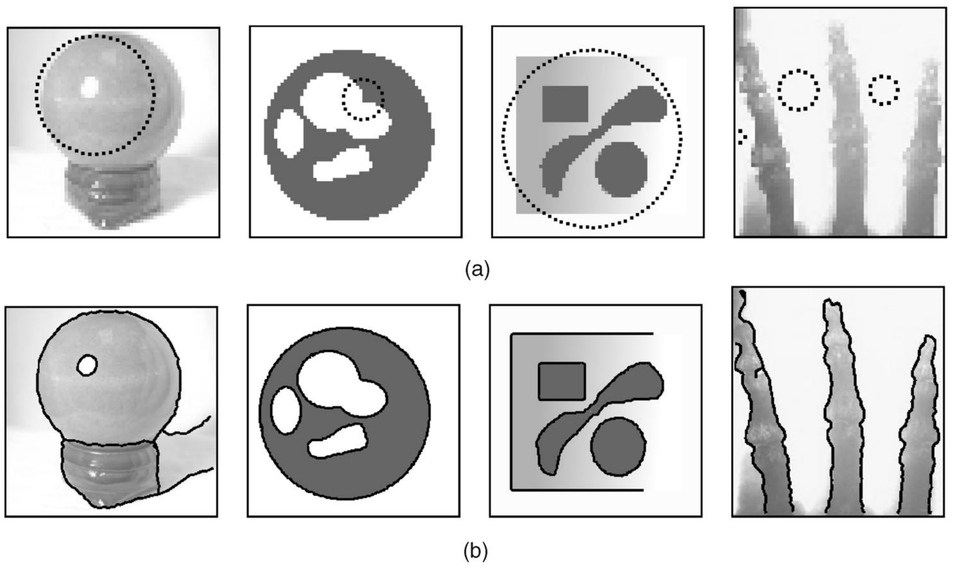


Fig. 9. Embedded objects and topology changes. First row: input images and initializations; second row: reconstructed curves.

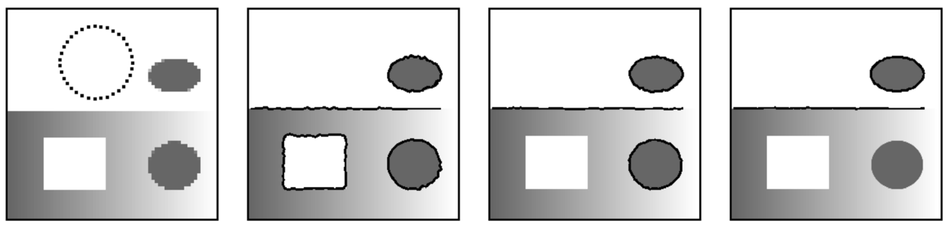


Fig. 10. Boundary leakage. Left-to-right: input image and initialization, results for different values of the damping coefficient: $\beta = 0.6$, $\beta = 1.0$, $\beta = 1.2$.

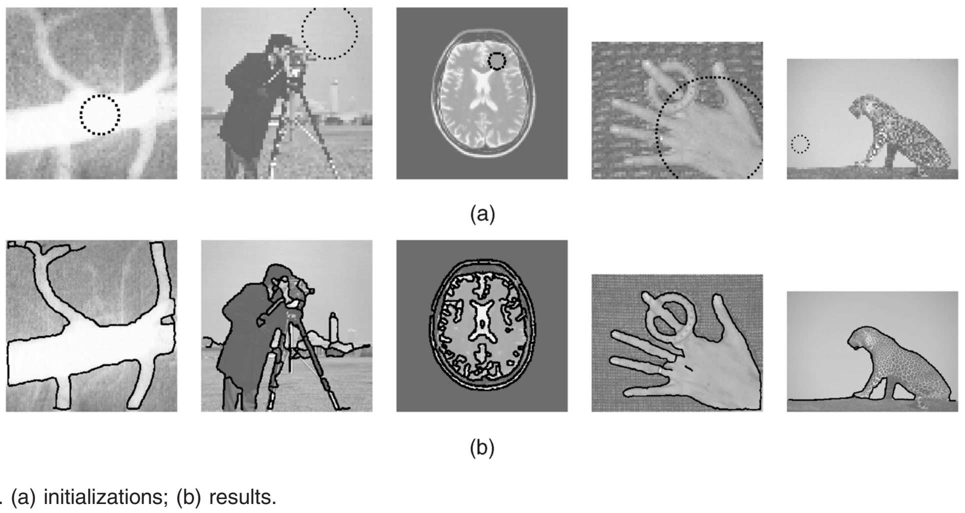


Fig. 11. Other results. (a) initializations; (b) results.

an attempt to reduce the oscillations of the particles near equilibrium positions and, hence, to increase the accuracy of the reconstruction; however, the downside is a slightly increased convergence time.

The first set of segmentation results is shown in Fig. 12. The initializations shown in the first two cases were performed by placing free particles at those locations of the gradient-magnitude image with values above 10 percent of the maximum magnitude. In the next two images, initialization was performed by uniformly spreading particles over the image plane. The most important structures shown in these complex images were correctly recovered, although some gaps in the final contours do exist. To address this problem, one can use more advanced curve reconstruction algorithms, better schemes for filtering, and postprocessing of the contours. Note that, here, no postprocessing of the contours was performed and only Gaussian filtering was used.

Some segmentation results for natural images obtained using the first initialization method are shown in Fig. 13. Natural images are known to be particularly difficult to segment, mostly because of the background texture surrounding the main objects. Without being perfect, the segmentation results shown in both figures are quite good, even though very simple initialization methods were used. We expect that more advanced initializations combined with nonlinear filtering would yield even better results.

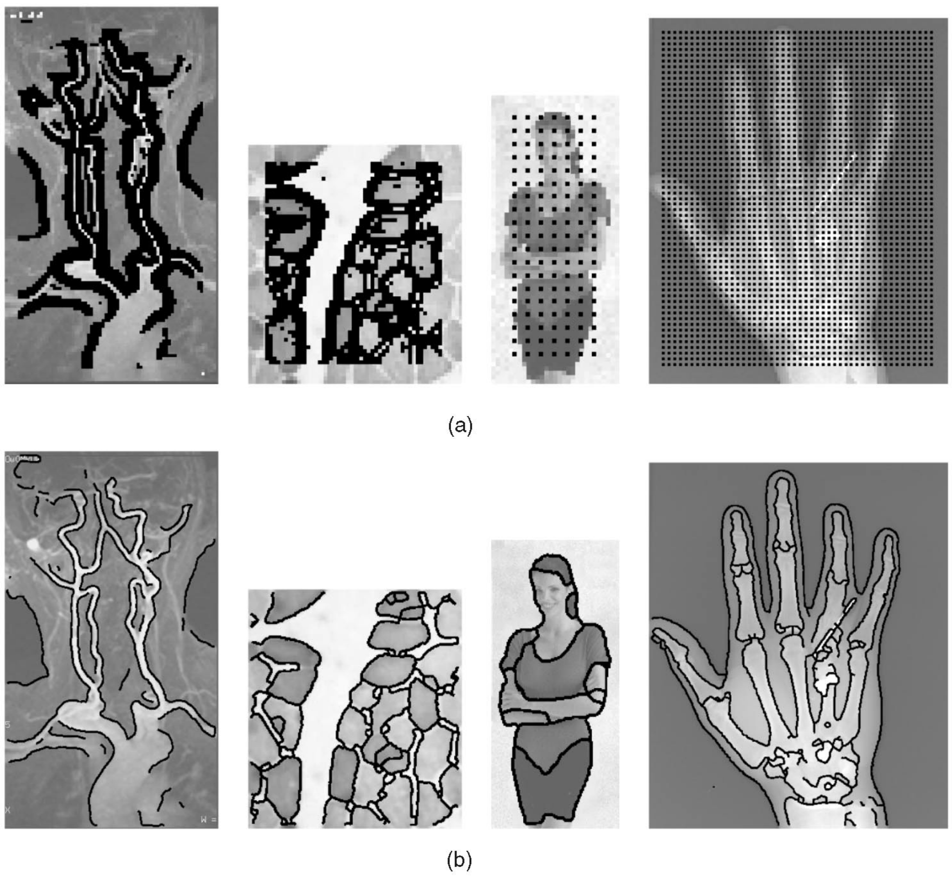


Fig. 12. Automatic segmentation. (a) initializations; (b) results.

5.3 Skeleton Extraction

Another application of the CPM is skeleton computation, based on the observation that local minima of the electric field correspond to locations situated on or near the center of the shape. Therefore, a simple method for skeleton extraction is as follows:

1. Compute the electric field $\boldsymbol{E}$.
2. Detect local minima of the magnitude of the field and assign fixed charges e_i to these locations.
3. Compute once again the electric field to yield the Lorentz force $\boldsymbol{F}_l$.

We use a very simple local-minima detector, which labels points as minima if they have in their (5×5) neighborhood at least 95 percent of points with value greater than or equal to their own values.

Some examples of skeleton extraction using this experimental setup are shown in Fig. 14. The first initialization was done by placing free particles at those locations of the gradient-magnitude image (after the second computation of the electric field) with values above 10 percent of the maximum magnitude. The second initialization was obtained by uniformly spreading free particles over the image plane. The advantage of this method for skeleton computation is that it can be applied to gray-scale images. A possible extension is to augment the skeleton with some information which may be used to reconstruct the curves (similar to the medial axis transform).

5.4 Three-Dimensional Examples

A synthetic shape-recovery experiment using the CPM in 3D was carried out on a 64^3 grid representing a torus, see Fig. 15. The parameters of the model were set to $w_1 = 0.6$, $w_2 = 0.7$, $\beta = 0.4$, and the initialization was obtained by sampling particles on the surfaces of the four spheres shown in the first image. Intermediate reconstruction snapshots are shown in the next images, ending with the correctly recovered shape of the target object—the torus shown in the last image.

A realistic shape-recovery experiment was performed on a 3D MRA (Magnetic Resonance Angiography) volume data set (256^3 voxels), see Fig. 16. A maximum intensity projection (MIP) rendering of this data set is shown in the left image of Fig. 16; note the high amount of noise present in this data set. The next image shows the initialization (625 particles were sampled on the surface of the small sphere shown as wire frame model) superimposed on the recovered surface, consisting of 22,120 vertices (i.e., particles) and 80,306 faces. The last image shows the recovered shape using as initialization 2,500 particles sampled on the surfaces of the four spheres. The final surface has 26,417 vertices and 110,987 faces. Note that no postprocessing (i.e., smoothing or simplification) of the final surfaces was performed. Both results show that the most important structures were correctly recovered, and only a few vessels are disconnected.

The same characteristics of the model observed in 2D are apparent here as well; we could have initialized particles anywhere inside the volume and the result would have been essentially the same.

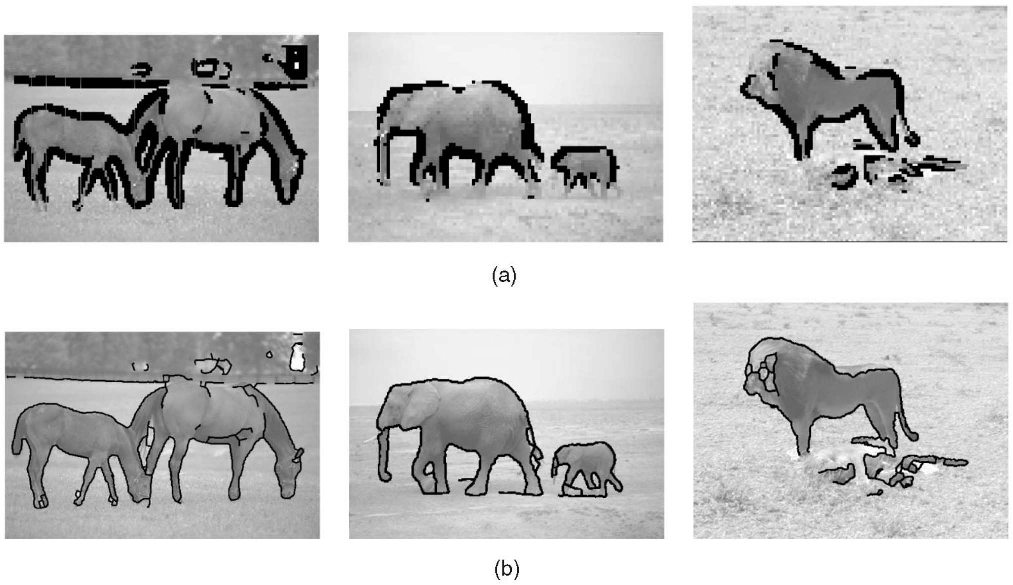


Fig. 13. Segmentation of natural images. (a) initializations; (b) results.

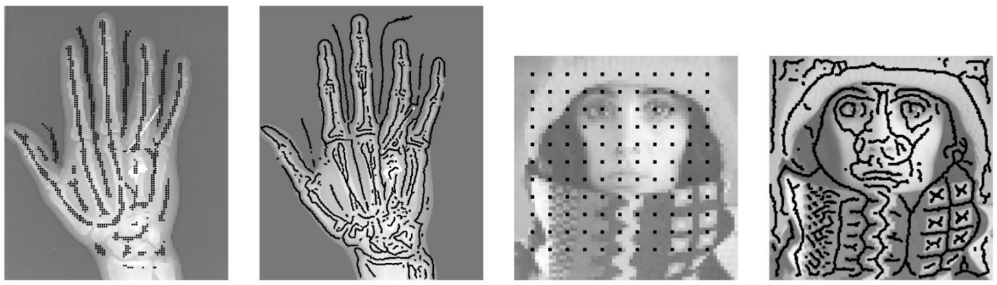


Fig. 14. Two examples of skeleton computation.

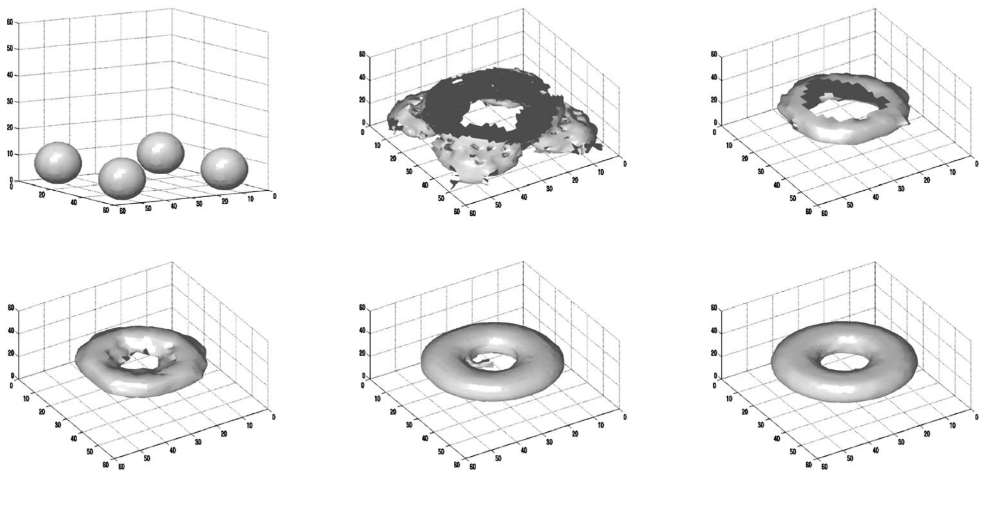


Fig. 15. Synthetic 3D example. Left-to-right, top-to-bottom: initialization, reconstructions at time steps $t = 20, 40, 100, 140, 180$, at different scales.

5.5 Parameter Sensitivity

In this experiment, we consider the behavior of the CPM with varying settings of parameters. Since the choice of the damping factor β was already addressed (see Fig. 10), we limit ourselves to the effect of varying the weight ratio $w_r = w_1/w_2$ and the annealing factor a, see Fig. 17. The results indicate that the parameter setting of the CPM is not critical. There is a wide range from which one can select suitable parameter values. Obviously, the model cannot handle extreme settings (see the right-most images), and in these cases the results may not be usable.

5.6 Coping with Noise

We have studied the behaviour of the CPM in the presence of three types of independently distributed noise: 1) Gaussian noise, 2) uniform noise, and 3) salt and pepper noise. All experiments were carried out on the same binary image, with the same initialization–the circle superimposed

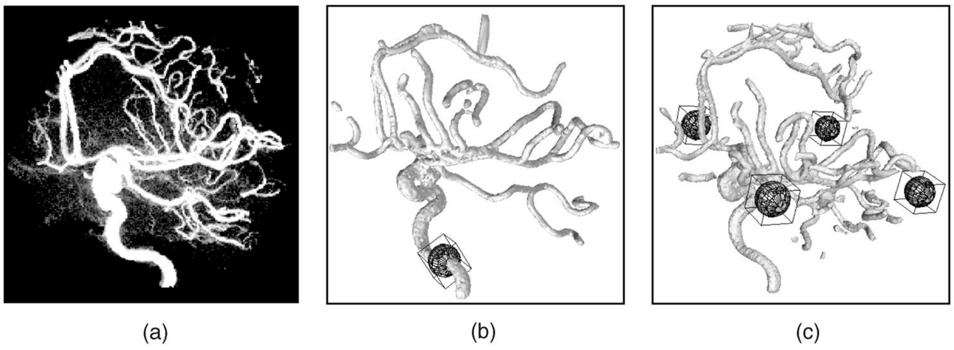


Fig. 16. Realistic 3D shape-recovery example. (a) Maximum intensity projection of the input volume; (b) recovered shapes using particles initialized on the surface of the small sphere shown as wire frame model; (c) recovered shapes using particles initialized on the surfaces of the four spheres shown as wire frame models.

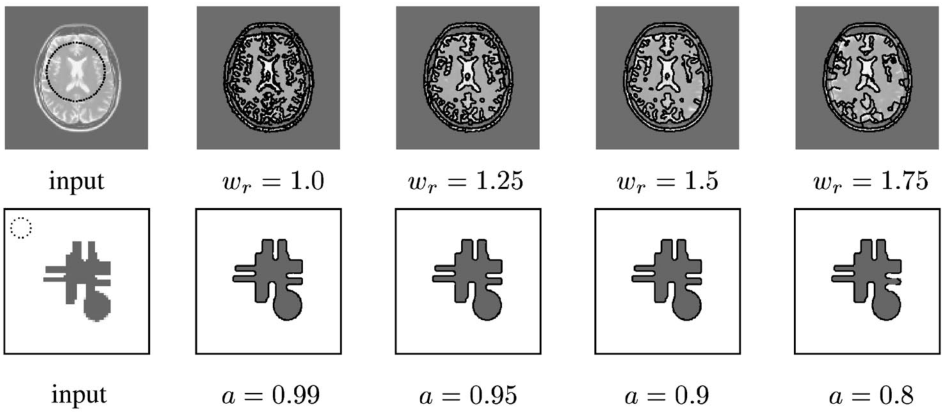


Fig. 17. Sensitivity of the CPM to different parameter settings. In row 1, w_r denotes the weight ratio w_1/w_2.

over the result shown in the first image of Fig. 18. The following noise types were used:

- *Gaussian noise*: zero mean and standard deviations $\sigma = 20, 60, 80, 100$.
- *Uniform noise*: 20, 60, 80, 90 percent of the pixels were replaced by random values between gray level 0 and 255.
- *Salt and pepper noise*: 20, 60, 80, 90 percent pixels were replaced with either 0 or 255, randomly.

The results in Fig. 18 show that the method copes quite well with all three types of noise. In all but two cases, the method succeeds in finding rather good approximations of the contours of the target object. Even in the last image, which is the most difficult case (since the input image is binary and has the largest amount of salt and pepper noise), the method is still able to output a (noisy) contour, although the object is more difficult to recognize.

6 Conclusions and Future Work

We have introduced the CPM (charged-particle model), a new physically motivated deformable model for shape recovery, and demonstrated its flexibility and potential in a wide variety of settings: 1) shape recovery using manual initialization, 2) automatic segmentation, and 3) skeleton computation on gray-scale images. The CPM exhibits some important characteristics:

1. much less sensitivity to initialization than snakes;
2. increased capture range, granted by both Coulomb and Lorentz forces;
3. good convergence into boundary concavities;
4. possibility to handle topological changes;
5. easily extendable to 3D; and
6. good behavior on noisy images.

In our opinion, the most important advantage of the CPM over active contours is that the method can be used for automatic segmentation, even with very simple initialization procedures. Extensive user interaction in the initialization phase is not mandatory, and segmentation can be performed automatically.

Efficient methods were described for fast evaluation of the Coulomb force at all particle positions (computational complexity $O(N \log N)$, with N the number of particles) and computation of the external electric field (complexity $O(M)$, with M number of pixels in the image). Shape reconstruction can be done in $O(N \log N)$ time, using the algorithms of Amenta et al. [40], [41]. Therefore, the overall complexity of the method is dominated by evaluations of Coulomb forces, performed at each time step. In practice, the method is slower compared to most parametric snakes (complexity $O(N)$, with N the number of snaxels), but faster than level-set techniques (even with efficient implementations based on the narrow-band algorithm [20]). For example, the time taken to segment the X-ray hand image from Fig. 12 (resolution 417×510 pixels) using uniformly placed particles, was 45 seconds on a Pentium III at 667 MHz. In 3D, the time taken to segment the

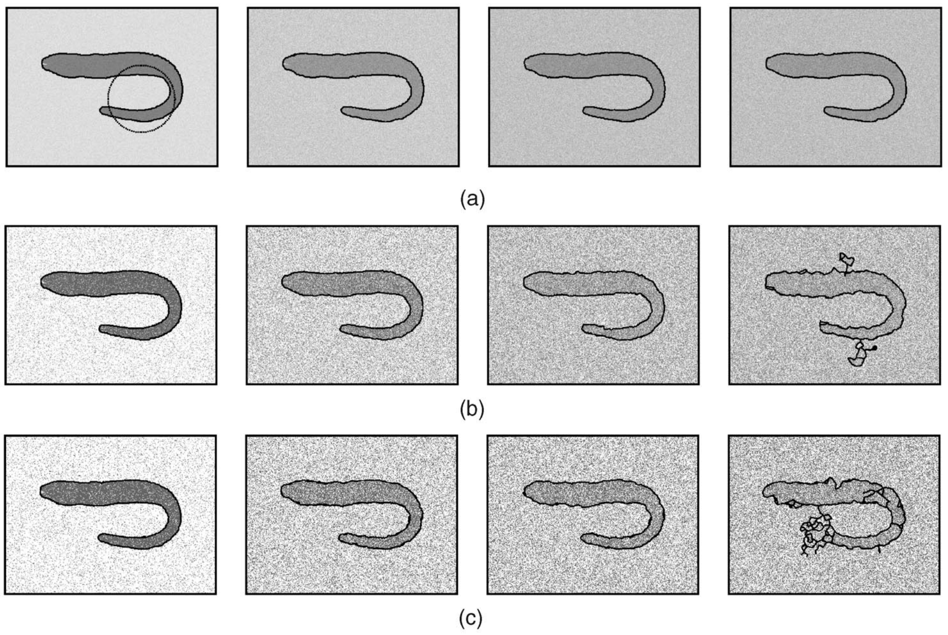


Fig. 18. Noise behavior. First image—common initialization; next Images—results in the presence of: (a) Gaussian noise with $\sigma = 20, 60, 80, 100$; (b) uniform noise with 20, 60, 80, 90 percent replaced pixels; (c) salt and pepper noise with 20, 60, 80, 90 percent replaced pixels.

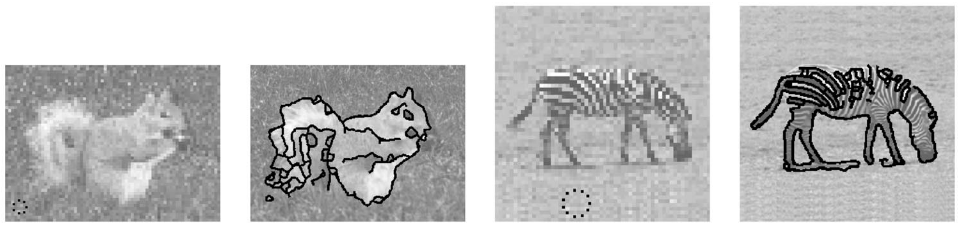


Fig. 19. The CPM may fail if highly textured regions surround or belong to the main object.

angiogram shown in Fig. 16 (256^3 voxels) was around five minutes (depending on the initialization) on an AMD XP2600+ at 1900 MHz.

Further investigations of the CPM are the subject of ongoing research. Of high interest is the applicability of the method for segmentation of medical images. In addition, we will focus on supplementing the skeleton with some information useful in the reconstruction phase. A shortcoming of the current method is that it cannot guarantee that the recovered contours (surfaces) are without gaps. Instead of using Gaussian pyramids, one can use wavelet or other pyramids based on nonlinear diffusion operators. Also, more complex initialization methods for automatic segmentation can be used to shorten the time required by the particles to converge.

Another research direction would be to include more constraints in the model. For example, we would want to integrate some region information, as in [7], which means that shape reconstruction should be done at each time step (which is feasible since the reconstruction algorithms are fast) in order to identify regions. Without this information, the CPM may fail to recover the contours of the main objects, if highly textured regions are present either in the surroundings or in the main objects, see Fig. 19.

We believe that the CPM has wide potential and is flexible enough to be used and adapted to a given task ranging from geometrical modeling to segmentation and object tracking.

References

[1] M. Kass, A. Witkin, and D. Terzopoulos, “Snakes: Active contour models,” *Int'l J. Computer Vision,* vol. 1, pp. 321-331, 1987.

[2] D. Terzopoulos, A. Witkin, and M. Kass, “Constraints on Deformable Models: Recovering 3D Shape and Nonrigid Motion,” *Artificial Intelligence,* vol. 36, pp. 91-123, 1988.

[3] T. McInerney and D. Terzopoulos, “Deformable Models in Medical Image Analysis: A Survey,” *Medical Image Analysis,* vol. 1, pp. 91-108, 1996.

[4] J. Suri, K. Liu, S. Singh, S. Laxminarayan, X. Zeng, and L. Reden, “Shape Recovery Algorithms Using Level Sets in 2-D/3D Medical Imagery: A State of the Art Review,” *IEEE Trans. Information Technology in Biomedicine,* vol. 6, pp. 8-28, 2002.

[5] J.L. Mallet, “Discrete Smooth Interpolation in Geometric Modelling,” *Computer Aided Design,* vol. 24, pp. 178-191, 1992.

[6] D. Terzopoulos, J. Platt, A. Barr, and K. Fleischer, “Elastically Deformable Models,” *ACM Computer Graphics,* vol. 21, pp. 205-214, 1987.

[7] N. Paragios and R. Deriche, “Geodesic Active Contours for Supervised Texture Segmentation,” *Proc. Conf. Computer Vision Pattern Recognition,* pp. 422-427, 1999.

[8] D. Geiger, A. Gupta, L.A. Costa, and J. Vlontzos, “Dynamic-Programming for Detecting, Tracking, and Matching Deformable Contours,” *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 18, no. 5, p. 575, May 1996.

[9] F.F. Leymarie and M.D. Levine, “Tracking Deformable Objects in the Plane Using an Active Contour Model,” *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 15, no. 6, pp. 617-634, June 1993.

[10] A. Amini, T. Weymouth, and R. Jain, “Using Dynamic Programming for Solving Variational Problems in Vision,” *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 12, pp. 855-867, 1990.

[11] D.J. Williams and M. Shah, "A Fast Algorithm for Active Contours and Curvature Estimation," *Proc. Conf. Computer Vision Graphics and Image Processing,* vol. 55, pp. 14-26, 1992.

[12] F. Leymarie and M.D. Levine, "Tracking Deformable Objects in the Plane Using an Active Contour Model," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 15, pp. 617-634, 1993.

[13] R.P. Grzeszczuk and D.N. Levin, ""Brownian Strings": Segmenting Images with Stochastically Deformable Contours," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 19, pp. 1100-1114, 1997.

[14] N. Peterfreund, "The Velocity Snake: Deformable Contour for Tracking in Spatio-Velocity Space," *Computer Vision and Image Understanding,* vol. 73, pp. 346-356, 1999.

[15] K.P. Ngoi and J.C. Jia, "An Active Contour Model for Colour Region Extraction in Natural Scenes," *Image and Vision Computing,* vol. 17, pp. 955-966, 1999.

[16] Y.Y. Wong, P.C. Yuen, and C.S. Tong, "Segmented Snake for Contour Detection," *Pattern Recognition,* vol. 31, pp. 1669-1679, 1998.

[17] M. Wang, J. Evans, L. Hassebrook, and C. Knapp, "A Multistage, Optimal Active Contour Model," *IEEE Trans. Image Processing,* vol. 5, pp. 1586-1591, 1996.

[18] C. Xu and J.L. Prince, "Snakes, Shapes, and Gradient Vector Flow," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 7, no. 3, pp. 359-369, Mar. 1998.

[19] F. Mokhtarian and F. Mohanna, "Fast Active Contour Convergence through Curvature Scale Space Filtering," *Proc. Image and Vision Computing New Zealand,* pp. 157-162, 2001.

[20] R. Malladi, J.A. Sethian, and B.C. Vemuri, "Shape Modeling with Front Propagation: A Level Set Approach," *IEEE Trans. Pattern Analysis Machine Intelligence,* vol. 17, pp. 158-175, 1995.

[21] V. Caselles, R. Kimmel, and G. Sapiro, "Geodesic Active Contours," *Proc. Fifth Int'l Conf. Computer Vision,* pp. 694-699, 1995.

[22] T. McInerney and D. Terzopoulos, "Topology Adaptive Deformable Surfaces for Medical Image Volume Segmentation," *IEEE Trans. Medical Imaging,* vol. 18, pp. 840-850, 1999.

[23] M. Vasilescu and D. Terzopoulos, "Adaptive Meshes and Shells: Irregular Triangulation, Discontinuities, and Hierarchical Subdivision," *Proc. Conf. Computer Vision and Pattern Recognition,* pp. 829-832, 1992.

[24] W.T. Reeves, "Particle Systems—A Technique for Modeling a Class of Fuzzy Objects," *Computer Graphics,* vol. 17, pp. 359-376, 1983.

[25] R. Szeliski and D. Tonnesen, "Surface Modeling with Oriented Particle Systems," *Computer Graphics,* vol. 26, pp. 185-194, 1992.

[26] K. Siddiqi, Y.B. Lauziere, A. Tannenbaum, and S.W. Zucker, "Area and Length Minimizing Flows for Shape Segmentation," *IEEE Trans. Image Processing,* vol. 7, pp. 433-443, 1998.

[27] C. Xu, A. Yezzi, and J.L. Prince, "On the Relationship between Parametric and Geometric Active Contours," *Proc. 34th Asilomar Conf. Signals, Systems, and Computers,* pp. 483-489, 2000.

[28] D. Perrin and C. Smith, "Rethinking Classical Internal Forces for Active Contour Models," *Proc. IEEE Conf. Computer Vision Pattern Recognition,* 2001.

[29] L.D. Cohen, "On Active Contour Models and Balloons," *Proc. Conf. Computer Vision Graphics and Image Processing, Image Understanding,* vol. 53, no. 2, pp. 211-218, 1991.

[30] J. Ivins and J. Porrill, "Active Region Models for Segmenting Medical Images," *First IEEE Int'l Conf. Image Processing,* 1994.

[31] R. Ronfard, "Region Based Strategies for Active Contour Models," *Int'l J. Computer Vision,* vol. 13, no. 2 1994.

[32] B. Leroy, I. Herlin, and L. Cohen, "Multi-Resolution Algorithms for Active Contour Models," *Proc. 12th Int'l Conf. Analysis and Optimization of Systems,* 1996.

[33] L.D. Cohen and I. Cohen, "Finite-Element Methods for Active Contour Models and Balloons for 2-D and 3D Images," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 15, no. 11, pp. 1131-1147, Nov. 1993.

[34] C. Xu and J. Prince, "Generalized Gradient Vector Flow External Forces for Active Contours," *Signal Processing—An Int'l J.,* vol. 71, no. 2, pp. 131-139, 1998.

[35] G. Storvik, "A Bayesian-Approach to Dynamic Contours through Stochastic Sampling and Simulated Annealing," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 16, no. 10, pp. 976-986, Oct. 1994.

[36] D. Geiger, A. Gupta, L.A. Costa, and J. Vlontzos, "Dynamical Programming for Detecting, Tracking and Matching Deformable Contours," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 17, no. 3, pp. 294-302, Mar. 1995.

[37] V. Caselles, F. Catte, T. Coll, and F. Dibos, "A Geometric Model for Active Contours," *Numerische Mathematik,* vol. 66, pp. 1-31, 1993.

[38] W.H. Press, B.P. Flannery, S.A. Teukolsky, and W.T. Vetterling, *Numerical Recipes in C: The Art of Scientific Computing.* Cambridge Univ. Press, 1988.

[39] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle, "Surface Reconstruction from Unorganized Points," *Proc. ACM SIGGRAPH 92 Conf.,* pp. 71-78, 1992.

[40] N. Amenta, M. Bern, and M. Kamvysselis, "A New Voronoi-Based Surface Reconstruction Algorithm," *Computer Graphics,* vol. 32, pp. 415-421, 1998.

[41] N. Amenta, M. Bern, and D. Eppstein, "The Crust and the β-Skeleton: Combinatorial Curve Reconstruction," *Graphical Models and Image Processing,* vol. 60, no. 2, pp. 125-135, 1998.

[42] H. Samet, *The Design and Analysis of Spatial Data Structures.* Addison-Wesley, 1989.

[43] R.W. Hockney and J.W. Eastwood, *Computer Simulation Using Particles.* Taylor & Francis, 1988.

[44] J.V.L. Beckers, C.P. Lowe, and S.W. deLeeuw, "An Iterative PPPM Method for Simulating Coulombic Systems on Distributed Memory Parallel Computers," *Molecular Simulation,* vol. 20, pp. 369-383, 1998.

[45] T. Lindeberg, "Scale-Space Theory: A Basic Tool for Analysing Structures at Different Scales," *J. of Applied Statistics,* vol. 21, no. 2, pp. 224-270, 1994.

Andrei C. Jalba received the BSc (1998) and MSc (1999) degrees in applied electronics and information engineering from the "Politehnica" University of Bucharest, Romania. He is currently pursuing the PhD degree at the Institute for Mathematics and Computing Science of the University of Groningen. His research interests include computer vision, pattern recognition, image processing, and parallel computing.

Michael H.F. Wilkinson received the MSc degree in astronomy from the Kapteyn Laboratory, University of Groningen (RuG) in 1993, after which he worked on image analysis of intestinal bacteria at the Department of Medical Microbiology, RuG. This work form the basis of his PhD at the Institute of Mathematics and Computing Science (IWI), RuG, in 1995. He was appointed as researcher at the Centre for High Performance Computing (also RuG) working on simulating the intestinal microbial ecosystem on parallel computers. During that time he edited the book *Digital Image Analysis of Microbes* (John Wiley, UK, 1998) together with Frits Schut. After this, he worked as a researcher at the IWI on image analysis of diatoms. He is currently assistant professor at the IWI. He is a member of the IEEE.

Jos B.T.M. Roerdink received the MSc (1979) degree in theoretical physics from the University of Nijmegen, the Netherlands. After receiving the PhD degree (1983) from the University of Utrecht and a two-year position (1983-1985) as a postdoctoral fellow at the University of California, San Diego, both in the area of stochastic processes, he joined the Centre for Mathematics and Computer Science in Amsterdam. There he worked from 1986-1992 on image processing and tomographic reconstruction. He was appointed associate professor (1992) and full professor (2003), respectively, at the Institute for Mathematics and Computing Science of the University of Groningen, where he currently holds a chair in scientific visualization and computer graphics. His current research interests include biomedical visualization, neuroimaging, and bioinformatics. He is a senior member of the IEEE.