SEGMENTATION OF COLOR IMAGES USING MEAN CURVATURE FLOW AND PARAMETRIC CURVES

Petr Pauš
Czech Technical University in Prague
Trojanova 13
120 00 Prague, Czech Republic

Shigetoshi Yazaki
Meiji University
1-1-1 Higashi-Mita, Tama-ku, Kawasaki-shi
Kanagawa 214-8571, Japan

Abstract. Automatic detection of objects in photos and images is beneficial in various scientific and industrial fields. This contribution suggests an algorithm for segmentation of color images by the means of the parametric mean curvature flow equation and CIE94 color distance function. The parametric approach is enriched by the enhanced algorithm for topological changes where the intersection of curves is computed instead of unreliable curve distance. The result is a set of parametric curves enclosing the object. The algorithm is presented on a test image and also on real photos.

1. Introduction. Numerical methods are widely used to perform automatic image segmentation, for example in the medical field (magnetic resonance images, computed tomography) to determine, e.g., the shape or volume of a ventricle, or for classification of objects in real world photos such as historical buildings and monuments. The real world photos are usually not gray-scale but contain additional color information which should be taken into account to achieve better segmentation results. The idea is to convert color image to gray-scale image by the color distance function developed by CIE (The International Commission on Illumination) [5] instead of simple conversion to gray-scale and then use the algorithm for a gray-scale image segmentation.

From the mathematical point of view, we consider smooth curves which evolve in time and at a finite time enclose the object which is to be segmented. The objects are segmented by a direct (parametric) method using an improved algorithm for topological changes based on [7, 8, 9].

The goal of the algorithm is to enclose areas of similar color into a set of closed curves. It is difficult to define an exact criterion for real world photo segmentation, therefore, we will try to match natural segmentation for human eyes.

2020 Mathematics Subject Classification. Primary: 68U05, 65D18; Secondary: 14H50.
Key words and phrases. Image segmentation, mean curvature flow, parametric curves, finite volume method, color images.

* Corresponding author: Petr Pauš.
2. Motion equation. The curve motion is described by the parametric method which is fast to compute since the problem is only one dimensional (as opposed to, e.g., level-set method [6]). The resulting piece-wise linear curve is given by a sequence of points and can easily be stored in some vector image format, such as SVG. On the other hand, this approach does not handle topological changes of the curve (splitting of a curve into several new curves or merging multiple curves into one).

In our approach, a smooth curve $\Gamma$ used for segmentation evolves according to the mean curvature flow equation [3, 4, 10]

$$v_\Gamma = -\kappa_\Gamma + F,$$

where $v_\Gamma$ is the normal velocity, $\kappa_\Gamma$ is the curvature, and $F$ is the forcing term acting on the curve in the normal direction. The term $F$ depends on the intensity of the segmented image, i.e.,

$$F(\vec{x}) = F_{\text{max}} - (F_{\text{max}} - F_{\text{min}}) \frac{I(\vec{x})}{255},$$

where $\vec{x}$ is the position vector, $F_{\text{max}}$ and $F_{\text{min}}$ are constants defining the maximal and minimal value for the forcing term, and $I(\vec{x})$ is the intensity of the image. We consider 8-bit color depth of the image, therefore, the $I(\vec{x})$ is divided by $2^{8} - 1 = 255$.

We consider a time interval $I_t = [0, T_{\text{max}}]$, a fixed interval $I_u = [0, 1]$ for the curve parameter $u$, and a vectorial mapping $\vec{X} : I_u \times I_t \rightarrow \mathbb{R}^2$, $\vec{X} = \vec{X}(u,t)$.

For closed curves, a circle of length 1 denoted $S^1$ is taken as $I_u$ to ensure all necessary properties of the problem (such as continuous derivatives everywhere, periodicity, periodic boundary conditions). The curve $\Gamma$ at time $t$ denoted as $\Gamma_t$ is given as

$$\Gamma_t = \{ \vec{X}(u,t) = (^1X(u,t), ^2X(u,t)), u \in I_u \},$$

where $^1X(u,t), ^2X(u,t)$ are the components of the position vector $\vec{X}(u,t)$.

The equation of the mean-curvature flow (1) describing the motion of the segmentation curve can be transformed into the parametric form as follows. The unit tangential vector $\vec{m}$ is defined as $\vec{m} = \frac{\partial_u \vec{X}}{|\partial_u \vec{X}|}$. The unit normal vector $\vec{n}$ in the curve plane is perpendicular to the tangential vector and $\vec{n} \cdot \vec{m} = 0$ holds, where "$\cdot"$ means the scalar product. In the case of a closed curve, $\vec{n}$ is the vector pointing into the interior of the curve.

The curvature $\kappa_\Gamma$ and the normal velocity $v_\Gamma$ can be derived according to [11]. The mean curvature flow equation (1) can be transformed into a partial differential equation

$$\partial_t \vec{X} = \frac{1}{|\partial_u \vec{X}|} \partial_u \left( \frac{\partial_u \vec{X}}{|\partial_u \vec{X}|} \right) + \alpha \frac{\partial_u \vec{X}}{|\partial_u \vec{X}|} + F(\vec{X}) \frac{\partial_u \vec{X} \perp}{|\partial_u \vec{X}|},$$

where the term $\alpha$ denotes the redistribution of the discretization points (see [11] or [1] for the derivation) which is necessary for long time computations and numerical stability of the solution. The term $\alpha$ in the tangential direction to the curve does not affect the shape of the curve but controls the distance of discretization points during the numerical computation. There are several ways how to incorporate tangential redistribution. Either, it is possible to keep the distance the same during the computation (i.e., uniform redistribution) or accumulate more discretization points in the areas with higher curvature to obtain higher accuracy of the object
description. Only the uniform redistribution for image segmentation is considered in this paper. Furthermore, we consider only closed curves. Therefore, the equation is accompanied by the periodic boundary conditions $\vec{X}(0, t) = \vec{X}(1, t)$, $\vec{m}(0, t) = \vec{m}(1, t)$.

3. **Treatment of color images.** In the case of color image segmentation, at first, we transform the color image into a gray-scale image using the color difference function $\Delta E$. We chose a CIE94 version of the $\Delta E$ color difference function. The details about the function can be found in [5]. Before the segmentation, the reference color is chosen. Then using the color difference formula $\Delta E$, we transform each pixel of the original image into gray-scale according to the color difference from the reference color. The distance value provided by the $\Delta E$ function is then mapped to the discrete 8-bit pixel intensity $[0, 255]$.

Color images are usually stored in RGB (red-green-blue) color space of the pixels. To use the $\Delta E$ function, it is necessary to convert the image to Lab color space [2]. Let us choose two colors in Lab color space, i.e., $(L_1, a_1, b_1)$ and $(L_2, a_2, b_2)$. Then, their distance $\Delta E$ reads as [5]

$$\Delta E = \left( \frac{\Delta L}{k_L S_L} \right)^2 + \left( \frac{\Delta C_{ab}}{k_C S_C} \right)^2 + \left( \frac{\Delta H_{ab}}{k_H S_H} \right)^2,$$

where

$$\Delta L = L_1 - L_2, \quad C_1 = \sqrt{a_1^2 + b_1^2},$$

$$C_2 = \sqrt{a_2^2 + b_2^2}, \quad \Delta C_{ab} = C_1 - C_2,$$

$$\Delta a = a_1 - a_2, \quad \Delta b = b_1 - b_2,$$

$$\Delta H_{ab} = \sqrt{\Delta a^2 + \Delta b^2 - \Delta C_{ab}^2}, \quad S_L = k_L = k_C = k_H = 1,$$

$$S_C = 1 + 0.045 C_1, \quad S_H = 1 + 0.015 C_1.$$  

Constants in the terms $S_C$ and $S_H$ are determined empirically by the CIE group. They incorporate the color dependence of the human vision.

4. **Numerical treatment.** As stated before, we use the direct (parametric) method. The direct approach is treated by the flowing finite method, and the tri-diagonal matrix is solved by the method of factorization (see [8]).

Our goal is to find a piece-wise linear curve $P_j$ approximating $\Gamma_t$ using segments $[X^j_i, X^j_{i+1}]$, $i = 1, \ldots, N - 1$ at the discrete time level $t = t_j$, where $t_0 = 0$ and

$$t_j = \sum_{k=1}^{j} \Delta t_k$$

with the time increment $\Delta t_k > 0, k = 0, 1, \ldots$. The curve $P^{j+1}$ is then determined by data $P^j$ at the previous time level.

At first, the initial values for $\{r^0_i\}$ and $\{\vec{p}_i^0\}$ are computed from the initial data according to (5). For $i = 1, 2, \ldots, N$ the required quantities read as follows:

$$r^j_i = |\vec{p}_i|, \quad r^{ij}_i = (r^j_i + r^{i+1}_i)/2, \quad \vec{p}_i = (p_{i_1}, p_{i_2}) = \vec{X}^j_i - \vec{X}^j_{i-1}.$$  

(5)
The discretization of the equation of motion (3) produces a system of linear equations
\[-a_i^{j+\frac{1}{2}} \Delta t_k \vec{X}_i^{j+1} + (1 + b_i^{j+\frac{1}{2}} \Delta t_k) \vec{X}_i^{j+1} - c_i^{j+\frac{1}{2}} \Delta t_k \vec{X}_i^{j+1} = \vec{X}_i^{j} + F(\vec{X}_i^{j}) \vec{N} \Delta t_k \]
where \(i = 1, \ldots, N\), \(6\)

\[b_i^{j+\frac{1}{2}} = a_i^{j+\frac{1}{2}} + c_i^{j+\frac{1}{2}},\]

\[a_i^{j+\frac{1}{2}} = \frac{1}{r_{i}^{j}} \left( \frac{1}{r_{i}^{j}} - \frac{\alpha_{i}^{j+1}}{2} \right), \quad c_i^{j+\frac{1}{2}} = \frac{1}{r_{i}^{j}} \left( \frac{1}{r_{i}^{j}} + \frac{\alpha_{i}^{j+1}}{2} \right).\]

The terms \(\alpha_{i}^{j+1}\) describe the tangential force on each discretization point and are computed according to [11]. The initial condition is given as an \(N\)-sided polygonal curve \(X_0 = \mathcal{P}^0\), and periodic boundary conditions are incorporated.

5. **Topological changes in parametric approach.** Unlike the level-set method [6], the parametric approach does not handle splitting or merging curves when they touch themselves. We developed a numerical algorithm to cover this disadvantage (see [7] and improved version in [9]) for two open curves which touch themselves during the evolution. In this contribution, we consider one closed curve that, under the influence of the image intensity, touches itself, and we propose better treatment of the splitting phenomenon compared to [7].

![Figure 1. Algorithm for topological changes of a closed curve Γ which overlaps itself under the external force. The intersections are computed and the overlapping segments of the curve are removed. The resulting two closed curves continue evolution in time.](image)

Let us consider one non-intersecting closed curve Γ discretized as \(W = \{\vec{w}_1, \vec{w}_2, \ldots, \vec{w}_n\}\) in \(\mathbb{R}^2\) clockwise (Figure 1a). The curve evolves independently according to the equation (3) under the influence of the external force generated by the image intensity. The evolution continues until a certain time step \(t_{k-1}\) is reached. This is the last time step where the curve does not intersect (overlap) itself (Figure 1a). In the following time step \(t_k\) the curve already overlaps and the algorithm for topological changes is executed.

The algorithm for splitting one open curve is as follows (Splitting algorithm):
1. At time \( t_k \) (Figure 1b), find two intersections of polygonal curve \( X \) with vertices from \( W \) using Intersection algorithm (below). If no intersection occurs, go to step 7.

2. Denote the starting points of intersecting line segments as \( \vec{w}_1, \vec{w}_2, \vec{w}_3, \vec{w}_4 \) according to Figure 1b. Note that the curve is discretized clockwise.

3. Create two new closed curves \( Y = \{\vec{y}_1, \vec{y}_2, \ldots, \vec{y}_n\} \) and \( Z = \{\vec{z}_1, \vec{z}_2, \ldots, \vec{z}_m\} \), where \( \vec{n} = n - i4 + i1 \) and \( \vec{m} = i3 - i2 \).

4. Fill the curve \( Y \) with proper data, i.e., set the values \( y_j, j = 1, \ldots, \vec{n} \) as follows: \( y_j = w_j, \quad j = 1, \ldots, i1 \) (blue part of the original curve) and \( y_{i1+j} = w_{i4+j}, \quad j = 1, \ldots, n - i4 \) (magenta part of the original curve). Simply put, copy the discretization points \( \vec{w}_1, \ldots, \vec{w}_{i1} \) (blue) to \( Y \) and append additional points \( \vec{w}_{i4+1}, \ldots, \vec{w}_n \) (magenta) to \( Y \) which will produce a closed curve with \( \vec{n} = n - i4 + i1 \) discretization points.

5. Fill the curve \( Z \) with proper data, i.e., set the values \( z_j, j = 1, \ldots, \vec{m} \) as follows: \( z_j = w_{i3+j}, \quad j = 1, \ldots, i3 - i2 \) (red part of the original curve). Simply put, copy the discretization points \( \vec{w}_{i2+1}, \ldots, \vec{w}_{i3} \) (red) to \( Z \) which will produce another closed curve with \( \vec{m} = i3 - i2 \) discretization points.

6. Delete the curve \( W \) and continue the computation with closed curves \( Y \) and \( Z \).

7. Proceed to a new time step for \( X \), resp., \( Y \), \( Z \), set \( k = k + 1 \), and go to step 1.

The algorithm for finding the intersections (Intersection algorithm) takes sequentially segments from the beginning of the curve and computes intersections with all other curve segments. Here, we consider periodicity, i.e., \( n + 1 = 1 \). Segment indices of first two intersections are returned.

1. For \( j = 1, \ldots, n \) do:
   2. For \( k = n, \ldots, j + 2 \) do:
   3. Take line segments \( s_j = (w_j, w_{j+1}) \) and \( s_k = (w_k, w_{k+1}) \).
   4. Compute the intersection of two line segments \( s_j \) and \( s_k \).
   5. If the intersection occurs, set \( i1 = j \) and \( i4 = k \) and continue loops.
   6. If the intersection occurs again, set \( i2 = j \) and \( i3 = k \) and go to 9.
   7. End k.
   8. End j.
   9. Return values \( i1, i2, i3, i4 \) and perform splitting algorithm.

If the curve overlaps itself two or more times in one time step, we apply the algorithm sequentially until all intersections are treated. That is, Intersection algorithm finds two intersections and Splitting algorithm splits the curve into two new closed curves. Intersection algorithm and Splitting algorithm is then applied to each new curve recursively until all intersections are treated. There is always even number of intersections since we do not consider touch as an intersection.

After splitting the curve into two new curves, two discretization points near the intersection may become very close to each other. However, the tangential redistribution resolves this issue after several time steps.

During the algorithm, some discretization points are removed. If the splitting occurs many times, it may result in poor discretization of the resulting segmentation. However, in practice, usually only two points are removed during each splitting. If necessary, we can easily apply algorithms for discretization refinement, e.g., adding new discretization point in the middle of each segment.
In the previous algorithm for topological changes of parametric curves described in [7], the splitting criterion was based on the distance of the discretization points of a closed curve. If the distance of two non-neighboring points was smaller than some constant or time variable \( \delta \), the splitting occurred. The main disadvantage was how to choose this \( \delta \). For large values, the curve topology changes sooner than it should and it is necessary to eliminate false detection of splitting for finely discretized curve (as described in [7]). On the other hand, for small values of \( \delta \) the algorithm may fail because splitting is not detected, especially for larger time step or sparser discretization. The crucial problem of finding proper \( \delta \) is eliminated in our new algorithm because it now detects the intersection of the curve. The disadvantage is that if the time step is large, more points may be removed, as described in the previous paragraph.

**Figure 2.** Original image with white background (left), gray-scale intensity image from a red color (middle), and simple conversion to gray-scale and inversion (right).

6. **Image segmentation by the means of the direct approach.** We applied the algorithm to several test images and the results are provided in this section. Let us start with an artificial test image. Firstly, the input image Figure 2 (left) is converted to the Lab color space and the reference color is chosen (for example red). Then the grey-scale distance image is computed using the color distance function \( \Delta E \) (4) for each pixel as seen in Figure 2 (middle). Colors optically closer to red for human eyes have higher intensity. One can see that the white background was transformed to dark gray and blue colors are even closer to black.

**Figure 3.** Artificial color image segmentation with the red reference color.

The image segmentation itself starts with a circle large enough to enclose most of the image. The curve then shrinks or expands and sticks to the edges of high intensity areas. The progress of the segmentation is illustrated in Figure 3. The
final segmentation consists of 8 circles enclosed by the black curves (the small black ellipse in the dark yellow circle will disappear in time).

The segmentation result depends on the choice of the reference color and also on the constants $F_{\text{max}}$ and $F_{\text{min}}$ in the forcing term equation (2). For all computations we used symmetric values $F_{\text{max}} = 1.2$ and $F_{\text{min}} = -1.2$. The segmentation result shows that colors similar to the red color are enclosed in curves while others not.

**Figure 4.** Comparison of the color distance segmentation (left) and simple gray-scale conversion segmentation (right).

It is possible to compare the segmentation by color distance function with the simple conversion of the original image to gray-scale (see Figure 2 middle and right). One can see that for the simple conversion, we cannot choose the dominant color and that is why we always obtain same segmentation. The gray-scale image was inverted since we need to have the dark background for segmentation. The segmentation results for both cases are presented in Figure 4. On the left, only colors close to red are segmented, while on the right, even the black or blue colors are segmented which is not the expected result.

**Figure 5.** Original yellow flower photo (left), the distance image from a yellow color (middle), and a simple conversion to gray-scale (right).

The algorithm was tested also on real world photos. Figure 5 (left) shows the yellow flower, its color distance conversion from the yellow color (middle), and a simple conversion to gray-scale (right). Directly from the intensity images we can guess which segmentation will provide better results. The color distance function faded all colors that differ too much from yellow. On the other hand, simple gray-scale conversion provided rather poor contrast for a yellow color. The segmentation
results are presented in Figure 6. The color distance segmentation is on the left and the simple conversion to gray-scale on the right. The figures show the last time step of the computation when the curve motion stopped.

Figure 6. Comparison of the color distance segmentation (left) and simple gray-scale conversion segmentation (right) for a yellow flower.

The last example illustrates the segmentation of a cloud (Figure 7) which contains only shades of blue color. Even in this situation the color distance function converts the image to gray-scale better than the simple conversion. Again, the original image is on the left, the color distance conversion in the middle, and the simple conversion on the right. The segmentation process is presented in Figure 8.

Figure 7. The photo of a cloud and its distance image from the almost white (very light light blue) color.

Figure 8. Segmentation of the original cloud image.
Results and discussion. The computations show that the algorithm for the color image segmentation works sufficiently well, at least from the optical evaluation by eyes. It is possible to simply select the desired color to be segmented and the related color distance intensity image is computed. This intensity image then serves as an input data for the algorithm based on [1]. In addition to [1], we incorporated the improved algorithm for topological changes which eliminated the biggest flaw of the previous version [7] (the choice of the distance parameter triggering the splitting algorithm).

However, there is still room for improvement. Similarly to [1], it is difficult to choose the values of $F_{\text{min}}$ and $F_{\text{max}}$. There are several automatic methods based, for example, on image histogram but we obtained acceptable results by manual selection. This is still work in progress. There is also a problem with darker shades of a reference color. For example, if a bright yellow color was chosen, darker yellow shades are considered rather distant and are not segmented well (see Figure 9). This might require additional research of the color distance function.

Figure 9. Segmentation of the sunflower with different shades of yellow.

Also, the refinement of the curve during computation is needed because in the case of an object with complicated boundary we need to start with rather a high number of discretization points. The computations in this paper were performed using 500 or 1000 discretization points. On the other hand, the computation is fast, around 5 to 10 seconds on an average computer. For example, the flower segmentation in Figure 5 with 500 discretization points took 8.3 seconds on Intel I7 processor (I7-4578U) on a single core.

In this paper, we focused on the idea of incorporating CIE94 color distance function and on the enhancement of the splitting algorithm. However, the comparison with other methods for color image segmentation might be necessary.

Acknowledgments. This work was partially supported by the project No. 15-27178A of the Ministry of Health of the Czech Republic, partially by the NAKI project No. DG18P02OVV015 of the Ministry of Culture of the Czech Republic, and partially supported by JSPS KAKENHI Grant Number 19H01807.

REFERENCES

[1] M. Beneš, M. Kimura, P. Pauš, D. Ševčovič, T. Tsujikawa and S. Yazaki, Application of a curvature adjusted method in image segmentation, Bulletin of the Institute of Mathematics, Academia Sinica (New Series), 2008 (2008), 509–523.
[2] I. C. Consortium and et al., Specification icc. 1: 2004-10, (profile version 4.2. 0.0): Image technology colour management, 2004.

[3] K. Deckelnick and G. Dziuk, Discrete anisotropic curvature flow of graphs, *ESAIM: Mathematical Modelling and Numerical Analysis*, 33 (1999), 1203–1222.

[4] K. Deckelnick and G. Dziuk, Mean curvature flow and related topics, *Frontiers in Numerical Analysis*, Universitext, Springer, Berlin, (2002), 63–108.

[5] R. McDonald and K. J. Smith, Cie94-a new colour-difference formula, *Journal of the Society of Dyers and Colourists*, 111 (1995), 376–379.

[6] S. Osher and R. Fedkiw, *Level Set Methods and Dynamic Implicit Surfaces*, Applied Mathematical Sciences, 153. Springer-Verlag, New York, 2003.

[7] P. Pauš and M. Beneš, Direct approach to mean-curvature flow with topological changes, *Kybernetika (Prague)*, 45 (2009), 591–604.

[8] P. Pauš and M. Beneš, Algorithm for topological changes of parametrically described curves, *Proceedings of ALGORITMY*, (2009), 176–184.

[9] P. Pauš and S. Yazaki, Exact solution for dislocation bowing and a posteriori numerical technique for dislocation touching-splitting, *JSIAM Letters*, 7 (2015), 57–60.

[10] D. Ševčovič, Qualitative and quantitative aspects of curvature driven flows of planar curves, *Topics on Partial Differential Equations, Jindřich Nečas Cent. Math. Model. Lect. Notes, MatFyzPress, Prague*, 2 (2007), 55–119.

[11] D. Ševčovič and S. Yazaki, Evolution of plane curves with a curvature adjusted tangential velocity, *Japan Journal of Industrial and Applied Mathematics*, 28 (2011), 413–442.

Received January 2019; revised February 2020.

E-mail address: pauspetr@fit.cvut.cz
E-mail address: syazaki@meiji.ac.jp