Proceedings of ALGORITMY 2009
pp. 176–184

ALGORITHM FOR TOPOLOGICAL CHANGES OF PARAMETRICALLY DESCRIBED CURVES

PETR PAUŠ* AND MICHAL BENEŠ*

Abstract. This contribution presents an algorithm for topological changes of curves solved by a parametric method. The algorithm improves standard parametric method and allows merging and splitting closed and open curves. The algorithm is mainly designed to allow for topological changes which may occur during dislocation dynamics. We consider a family of closed or open smooth curves $\Gamma(t) : S \to \mathbb{R}^2$, $t \geqq 0$. The curves are driven by the normal velocity v which is the function of curvature κ and the position vector $x \in \Gamma(t)$. In this case the equation is defined as: $v = -\kappa + F$. The motion law is treated using direct approach solved by backward Euler semi-implicit scheme. Numerical stability is improved by tangential redistribution of curve points which allows long time computations and better accuracy.

Key words. Mean Curvature Flow, Dislocation Dynamics, Parametric approach

AMS subject classifications. 35L65, 76M12, 80A20

1. Introduction. The evolving curves can be mathematically described in several ways. One possibility is to use the *level-set method* [1, 2, 3, 13], where the curve is defined by the zero level of some surface function. One can also use the *phase-field method* [4]. Finally, it is possible to use the *direct (parametric) method* [5, 6] where the curve is parametrized in usual way. Parametric method is accurate and fast but the main disadvantage is that it does not allow topological changes of curves, i.e. merging or dividing. If we want to allow for such changes, we have to implement a particular algorithm for such situation. We restrict ourselves to the initial condition in the form of one or more disjoint non-selfintersecting curves.

2. Parametric description. When using the parametric approach, the planar curve $\Gamma(t)$ is described by a smooth time-dependent vector function

$$X : S \times I \to \mathbb{R}^2,$$

where $S = [0, 1]$ is a fixed interval for the curve parameter and $I = [0, T]$ is the time interval. The curve $\Gamma(t)$ is then given as the set

$$\Gamma(t) = \{X(u, t) = (X^1(u, t), X^2(u, t)), u \in S\}.$$

The curve evolve according to the equation of motion

$$v = -\kappa + F, \tag{2.1}$$

where v is the normal velocity of the curve evolution, κ is the curvature, and F is the forcing term.

The evolution law (2.1) is transformed into the parametric form. The unit tangential vector $\vec{T}$ is defined as $\vec{T} = \partial_u X / |\partial_u X|$. The unit normal vector $\vec{N}$ is perpendicular to the tangential vector and $\vec{N} \cdot \vec{T} = 0$ holds. In case of closed curve, $\vec{N}$ is the outer

*Department of Mathematics, Faculty of Nuclear Sciences and Physical Engineering, Czech Technical University, Prague vector to the interior of the curve. In case of open curve, $\vec{N}$ has a selected, pre-defined direction (e.g., upwards). The orientation of curves is clockwise.

The curvature κ is defined as

$$-\kappa = \frac{\partial_u X^{\perp}}{|\partial_u X|} \cdot \frac{\partial_{uu} X}{|\partial_u X|^2} = \vec{N} \cdot \frac{\partial_{uu} X}{|\partial_u X|^2},$$

where $X^{\perp}$ is a vector perpendicular to X. The normal velocity v is defined as the time derivative of X projected into the normal direction,

$$v = \partial_t X \cdot \frac{\partial_u X^{\perp}}{|\partial_u X|}.$$

The equation (2.1) can now be written as

$$\partial_t X \cdot \frac{\partial_u X^{\perp}}{|\partial_u X|} = \frac{\partial_{uu} X}{|\partial_u X|^2} \cdot \frac{\partial_u X^{\perp}}{|\partial_u X|} + F,$$

which holds provided

$$\partial_t X = \frac{\partial_{uu} X}{|\partial_u X|^2} + F(u,t) \frac{\partial_u X^{\perp}}{|\partial_u X|}. \tag{2.2}$$

This equation is accompanied by the periodic boundary conditions for closed curves, or by fixed-end boundary condition for open curves, and by initial condition. These conditions are considered similarly as in [5].

The solution of (2.2) exhibits a natural redistribution property which is useful for short-time curve evolution [7, 9]. The value of tangential force contained in the second derivative term is given by

$$\alpha_0 = \frac{\partial_{uu} X \cdot \partial_u X}{|\partial_u X|^3}.$$

For long time computations with time and space variable external force $F(X,t)$, the algorithm for curvature adjusted tangential velocity is used. This algorithm moves points along the curve according to the curvature, i.e., areas with higher curvature contain more points than areas with lower curvature. To incorporate a tangential redistribution, a tangential term α has to be added to the equation (2.2).

$$\partial_t X = \frac{\partial_{uu} X}{|\partial_u X|^2} - \alpha \frac{\partial_u X}{|\partial_u X|} + F(u,t) \frac{\partial_u X^{\perp}}{|\partial_u X|}. \tag{2.3}$$

This improves numerical stability and also accuracy of computation. The term α is chosen in such a way that it vanishes at the curve end points. Details of definition of α are described in [8].

3. Numerical scheme. For numerical approximation we consider a regularized form of (2.3) which reads as

$$\partial_t X = \frac{\partial_{uu} X}{Q(\partial_u X)^2} - \alpha \frac{\partial_u X}{Q(\partial_u X)} + F(u,t) \frac{\partial_u X^{\perp}}{Q(\partial_u X)}, \tag{3.1}$$

where $Q(x_1, x_2) = \sqrt{x_1^2 + x_2^2 + \varepsilon^2}$ is a regularization term and ε a small parameter. We use backward Euler semi-implicit scheme for the numerical solution of the differential equation (2.3). The first derivative is discretized by backward difference as follows

$$\partial_u X|_{u=jh} \approx \left[\frac{X_j^1 - X_{j-1}^1}{h}, \frac{X_j^2 - X_{j-1}^2}{h} \right],$$

and the second derivative as

$$\partial_{uu} X|_{u=jh} \approx \left[\frac{X_{j+1}^1 - 2X_j^1 + X_{j-1}^1}{h^2}, \frac{X_{j+1}^2 - 2X_j^2 + X_{j-1}^2}{h^2} \right].$$

The approximation of the first derivative is denoted as $X_{\bar{u},j}$ and the second derivative as $X_{\bar{u}u,j}$.

The semi-implicit scheme for equation (2.3) has the following form

$$X_j^{k+1} - \tau \frac{X_{\bar{u}u,j}^{k+1}}{Q^2(X_{\bar{u},j}^k)} + \tau\alpha_j \frac{X_{\bar{u},j}^{k+1}}{Q(X_{\bar{u},j}^k)} = X_j^k + \tau F(X_j^k, k\tau) \frac{X_{\bar{u},j}^{\perp k}}{Q(X_{\bar{u},j}^k)},$$
$$j = 1, \cdots, m-1, k = 0, \cdots, N_T - 1, \quad (3.2)$$

where $Q(x_1, x_2)$ is a regularization term, $X_{\bar{u},j}^{\perp}$ is a vector perpendicular to $X_{\bar{u},j}$, and α_j is redistribution coeficient. The term ε serves as a regularization to avoid singularities when the curvature tends to infinity.

$X_j^k \approx X(jh, k\tau)$, τ is a time step and N_T is the number of time steps. The matrix of the system (3.2) for one component of X^{k+1} has the following tridiagonal structure:

$$\begin{pmatrix} 1 + \frac{2t}{h^2Q^2} - \frac{t\alpha}{hQ} & \frac{-t}{h^2Q^2} & 0 & \cdots \\ \frac{-t}{h^2Q^2} + \frac{t\alpha}{hQ} & \ddots & \ddots & \ddots \\ 0 & \ddots & & \\ \vdots & \ddots & & \end{pmatrix}.$$

The scheme (3.2) is solved for each k by means of the matrix factorization.

4. Topological changes. The algorithm we present is not supposed to be universal for every situation and possibility. Main purpose is to simulate topological changes that can happen during dislocation dynamics (see [10]), i.e., topological changes such as merging or splitting of curves, closing of open curves, etc. As the initial condition, we consider only curves which do not intersect itself and do not touch each other. The orientation of curves is clockwise. That is The parametric approach does not handle them intrinsically, and we therefore need an additional algorithm allowing for such changes of discretized curves. The algorithm is designed for topological changes of curves which touch only in one point. More complex changes may be accomplished by multiple application of the algorithm in one timestep. We will describe it later.

Let us consider two closed or open curve parametrizations discretized as $X = \{x_1, x_2, \cdots, x_n\}$ and $Y = \{y_1, y_2, \cdots, y_m\}$ in $\mathbb{R}^2$. Curves evolve independently according to the equation (2.3). The algorithm for merging two curves is as follows:

1. Compute the distance between X and Y and find one point from each curve where the minimum is reached. Let us denote the distance as d, the point from X as x_{max} and from Y as y_{max}.
2. Check if the distance d between curves is smaller than a given tolerance δ. If not, compute new timestep and go to 1.
3. Create new empty curve Z. We must take into account the type of merged curves. Merging two closed curves will produce one closed curve. Merging one open and one closed curve will produce one open curve and merging two open curves will produce two open curves.
4. Copy points from X from the begining (i.e., from $x1$) up to x_{max} to Z.
5. Copy points from Y from y_{max} up to the end (i.e., up to y_m) to Z.
6. Copy points from Y from the begining (i.e., from y_1) up to y_{max} to Z.
7. Copy points from X from x_{max} up to the end (i.e., up to x_n) to Z.
8. Delete X and Y.
9. Compute a new timestep for Z and go to 1.

We also consider that one curve can intersect itself and thus split itself into 2 parts. Let us consider a closed or open curve discretized as $X = \{x_1, x_2, \cdots, x_n\}$. The curve evolves independently according to the equation (2.3). The algorithm for splitting into two curves is as follows:

1. Compute the distance between points in X and find two points where the minimum was reached. Let us denote the distance as d, and the points as x_{max1} and x_{max2}. We do not consider several points in the neighbourhood of each point when measuring the distance to avoid finding minimal distance for two neighbor points. The number has to be computed according to the value of a given tolerance δ (see the next step). We recommend to omit all points with the distance smaller than at least 4δ.
2. Check if the distance d between points is smaller than a given tolerance δ. If not, compute new timestep and go to 1.
3. Create two new empty curves X_{new1} and X_{new2}. If X is an open curve, X_{new1} will be open and X_{new2} closed curve. If X is a closed curve then X_{new1} and X_{new2} will be closed curves.
4. Copy points from X from the begining (i.e., from x_1) up to x_{max1} to X_{new1}.
5. Copy points from X from x_{max1} up to x_{max2} to X_{new2}.
6. Copy points from X from x_{max2} up to the end (i.e., up to x_n) to X_{new1}.
7. Delete X.
8. Compute new timestep for X_{new1} and X_{new2} and go to 1.

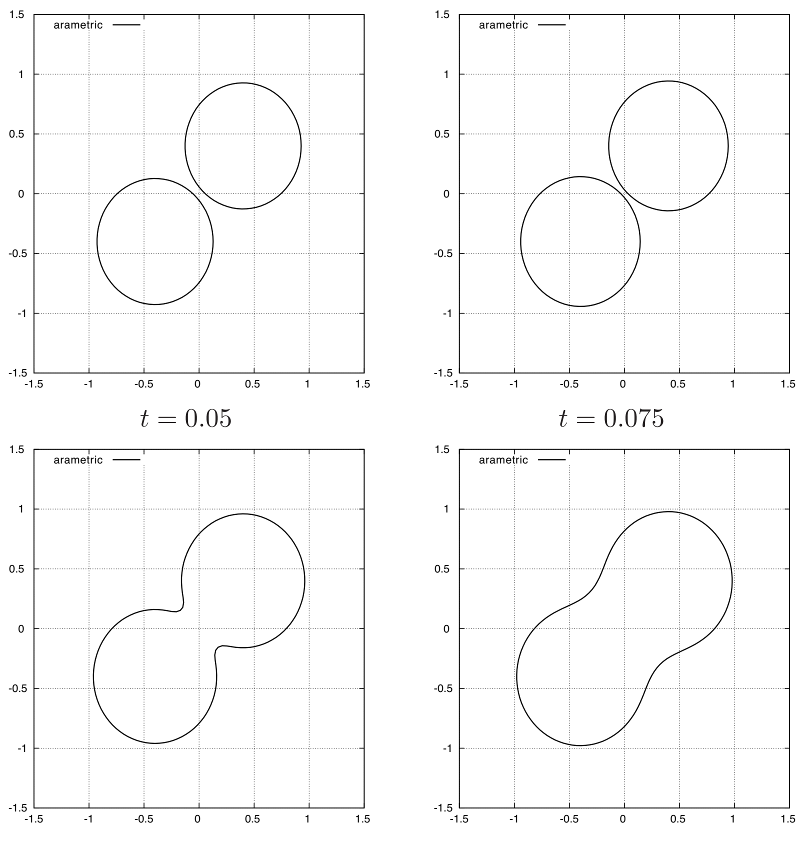
The points x_{max1}, x_{max2}, x_{max}, and y_{max} are omited during copying. It is important to copy all points in the same order as source curves to avoid problems with the direction of normal vector. The orientation and normal vector of the new curve are the same as for the original curve. Therefore, it may happen that the normal vector direction will be inward for the new curve.

The algorithm can handle the evolution of more than two curves. At each timestep it performs the procedure described above for each pair of curves which allows it to merge or divide more curves at a single timestep or to merge or divide one curve more times in a single timestep. Let us consider r open or closed parametrized curves $X_1, X_2, \cdots, X_r$. Multiple curves can be treated as follows:

```
1. For $k = 1$ to $r - 1$ do:
2.     For $l = k + 1$ to $r$ do:
3.         Perform the merging algorithm described above for $X_k$ and $X_l$ without
```

```
   computing a new timestep.
4.         If curves were merged, go to 1, else continue the FOR loops.
5. For $k = 1$ to $r$ do:
6.      Perform the algorithm for dividing curves on $X_k$ without computing a
   new timestep.
7.      If a curve was divided, go to 5, else continue the FOR loop.
8. Compute a new timestep for all curves.
```

The algorithm has a few drawbacks. If the parameter δ is small (i.e., we need a higher accuracy of merging or splitting), the curve must have fine discretization which causes higher computation times. For a small δ, the algorithm also needs a small timestep, otherwise curves may cross each other without merging or splitting. Recommended value of δ is about 10^{-4} or 10^{-5}.



$t = 0.05$ $t = 0.075$

$t = 0.1$ $t = 0.125$

Fig. 5.1: Merging of two circles, $F = 2.5, t \in (0, 1.25), \delta = 5 \cdot 10^{-4}$

5. Results of numerical simulation. The algorithm described above was implemented and tested in various situations. The first example shows merging of two circles (Figure 5.1). The circles with the diameter of 1 are located at [0.4, 0.4], resp. [-0.4, -0.4] and expand under $F = 2.5$. Each circle is discretized by 75 points. The threshold δ is set to $5 \cdot 10^{-4}$.

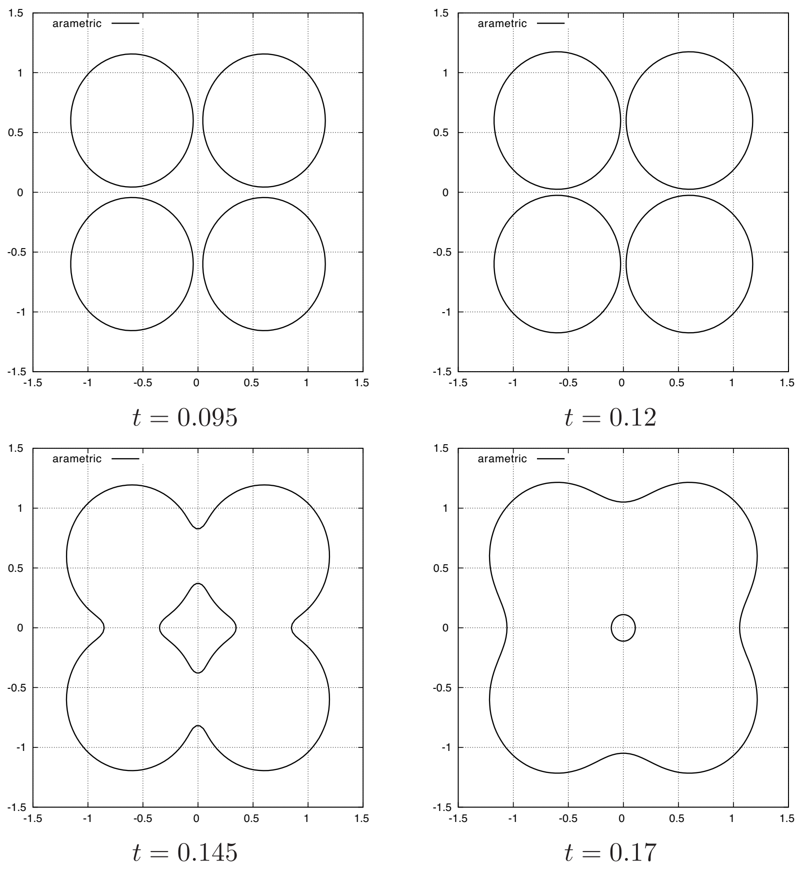


Fig. 5.2: Merging of four circles, $F = 2.5, t \in (0, 0.17), \delta = 5 \cdot 10^{-4}$

Figure 5.2 illustrates merging of four circles. The circles with the diameters of 1 are located at [0.6, 0.6], [-0.6, -0.6], [0.6, -0.6] and [-0.6, 0.6] and expand under $F = 2.5$. Each circle is discretized by 75 points. Under these settings, the curves should touch in the same time. Since the algorithm cannot handle more than one touch point, everything is done in steps. At first, two circles are merged, then merged with the third one, and then with the fourth one. Merging happens always only in one point. After merging, there is only one curve which touches itself. Now the algorithm for dividing curve starts and splits the curve into two closed curves. As a consequence, the normal vector of the outer curve is outward and for the inner curve inward. The inner curve then shrinks quickly to a point. A comparison with level-set method will be presented in [14].

Figure 5.3 illustrates merging of two circles and one open curve. The circles with the diameters of 1 are located at [0.6, 0.6] and [-0.6, 0.6], the open curve has fixed ends at [-1, -0.7] and [1,-0.7]. All curves expand under $F = 2.5$. Each curve is discretized by 75 points. The curves merge into one open curve.

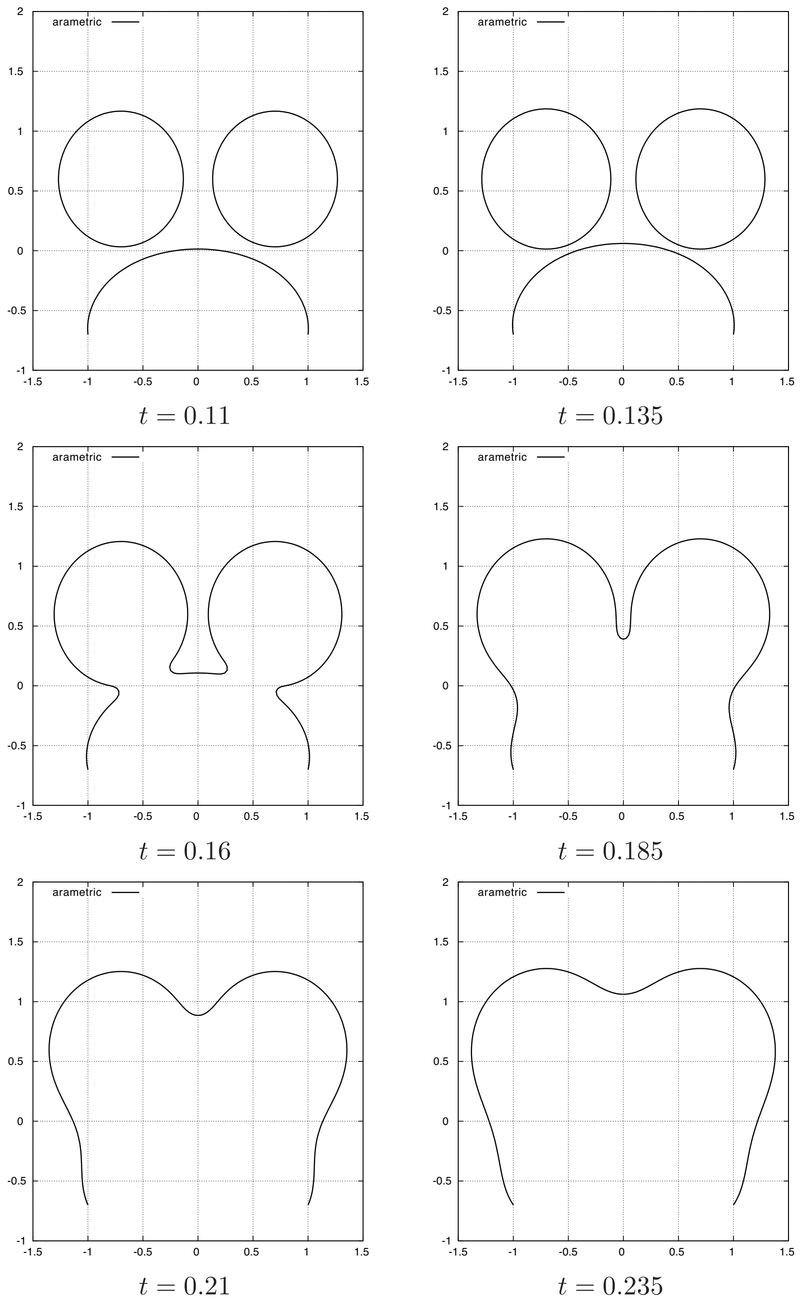
The last example (Figure 5.4) illustrates the expansion of an open curve which at a certain time touches itself and divide into one open and one closed curve. This computation is similar to dislocation dynamics, namely the Frank-Read source [11, 12]. The half-circle has fixed ends at [-1, 0] and [1,0] and expands under $F = 2.5$. The number of discretization points is 200. This computation requires tangential redistribution of points because if the redistibution is not used, areas near the ends of the curve contain very few points and the algorithm fails.

6. Conclusion. This contribution presents the algorithm for topological changes of parametric curves. The capabilities of the algorithm are fitted for simulation of dislocation dynamics. It is not designed to be universal. The main advantage is that it can handle both open and closed curves with a reasonable speed of computation. On the other hand, the algorithm needs a tangential redistribution of points and a higher number of discretization points for good precision. The comparison with other methods such as level set or phase field will be the subject of a future paper.

Acknowledgement. This work was partly supported by the project MSM No. 6840770100 “Applied Mathematics in Technical and Physical Sciences” and by the project No. LC06052 “Nečas Center for Mathematical Modelling” of the Ministry of Education, Youth and Sport of the Czech Republic.

REFERENCES

[1] S. Osher, R. P. Fedkiw: Level set methods and dynamic implicit surfaces, Springer, 2003
[2] J. A. Sethian: Level set methods and fast marching methods, Cambridge University Press, 1999.
[3] G. Dziuk, A. Schmidt, A. Brillard, and C. Bandle: Course on mean curvature flow, Manuscript 75p., Freiburg, 1994.
[4] M. Beneš: Phase field model of microstructure growth in solidification of pure substances, Acta Math. Univ. Comenian 70 (2001), 123-151.
[5] K. Deckelnick, G. Dziuk: Mean curvature flow and related topics, Albert-Ludwigs-Univ., Math. Fak, 2002.
[6] K. Mikula, D. Ševčovič: Computational and qualitative aspects of evolution of curves driven by curvature and external force, Computing and Visualization in Science, Vol. 6 (2004), No. 4, 211–225.
[7] K. Mikula, D. Ševčovič: Evolution of plane curves driven by a nonlinear function of curvature and anisotropy, SIAM Journal on Applied Mathematics Vol. 61 (2001), No. 5, 1473–1501.
[8] D. Ševčovič, S. Yazaki: On a motion of plane curves with a curvature adjusted tangential velocity, arXiv 0.7.11.2568.
[9] P. Pauš: Numerical simulation of dislocation dynamics, MAGIA, Proceedings of Slovak-Austrian Congress, Podbanské (2007), pp. 45–52.
[10] P. Pauš, M. Beneš: Direct approach to mean-curvature ow with topological changes, Proceedings of Czech-Japanese Seminar 2008, Takachiho, Japan (submitted).
[11] T. Mura: Micromechanics of Defects in Solids, Springer, 1987
[12] F. Kroupa: Long-range elastic eld of semi-innite dislocation dipole and of dislocation jog. Phys. Status Solidi 9 (1965), 2732.
[13] M. Beneš, K. Mikula, T. Oberhuber, D. Ševčovič: Comparison study for Level set and Direct Lagrangian methods for computing Willmore flow of closed planar curves, Computing and Visualization in Science (2008) (accepted)
[14] P. Pauš, M. Beneš: Comparison of methods for mean curvature flow, in preparation.

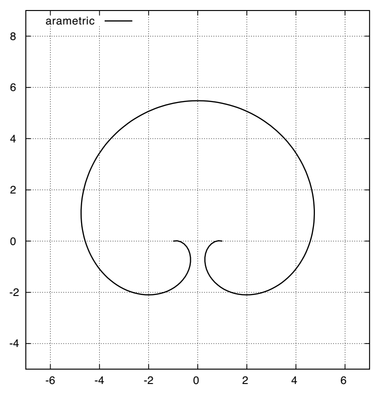


$t = 0.11$ $t = 0.135$

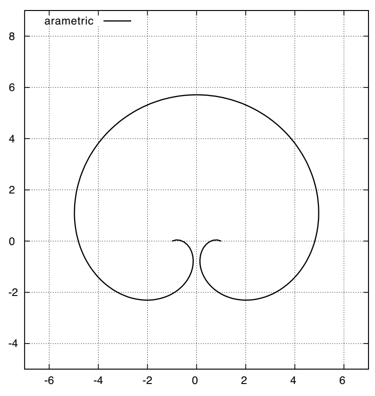
$t = 0.16$ $t = 0.185$

$t = 0.21$ $t = 0.235$

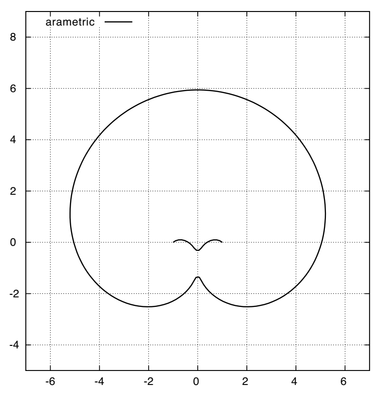
Fig. 5.3: Merging of two circles and one open curve, $F = 2.5, t \in (0, 0.235), \delta = 5 \cdot 10^{-4}$



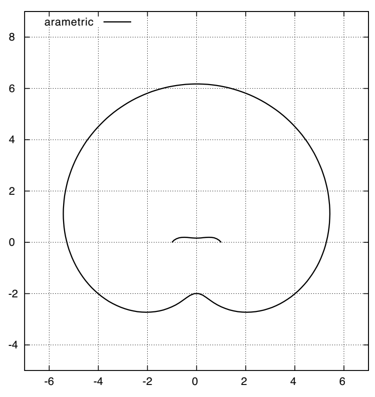
$t = 2.4$



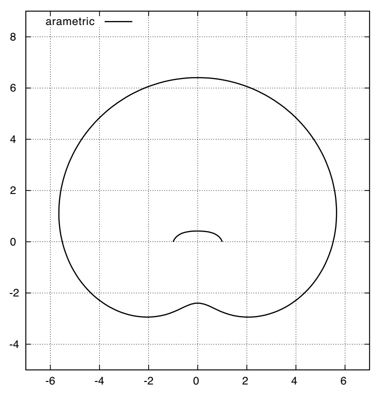
$t = 2.5$



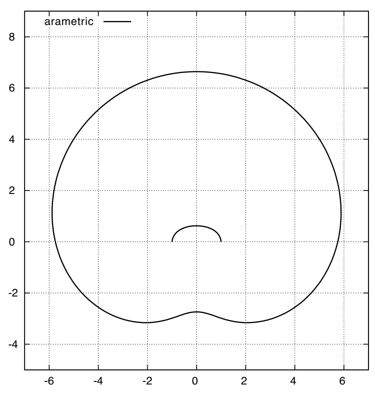
$t = 2.6$



$t = 2.7$



$t = 2.8$



$t = 2.9$

Fig. 5.4: Dividing an open curve, $F = 2.5, t \in (0, 2.9), \delta = 5 \cdot 10^{-4}$