# An ADMM-based scheme for distance function approximation 

Alexander Belyaev ${ }^{1}$ (1) $\cdot$ Pierre-Alain Fayolle ${ }^{2}$

Received: 6 March 2019 / Accepted: 24 July 2019/ Published online: 19 August 2019
© The Author(s) 2019


#### Abstract

A novel variational problem for approximating the distance function (to a domain boundary) is proposed. It is shown that this problem can be efficiently solved by ADMM. A review of several other variational and PDE-based methods for distance function estimation is presented. Advantages of the proposed distance function estimation method are demonstrated by numerical experiments. Applications of the method to the problems of surface curvature estimation and computing the skeleton of a binary image are shown.


Keywords Distance function • Variational methods • Distance transform • Skeleton • Curvature

## 1 Introduction

Fast and accurate estimation of the distance to a surface (the distance to a curve in 2D) is important for a number of applications including redistancing or reinitialization for level-set methods [17], wall distance models in turbulence modeling [23, 29], heterogeneous material modeling in computational mechanics [6], medial axis transform and meshing [31], FEM extensions [2, 16], robot navigation and mapping [9], and

[^0]various computer graphics studies [10, 34, 35]. In particular, variational and PDEbased methods for distance function estimation are currently a subject of intensive research [3, 4, 11, 22, 24].

In this paper, we propose a variational method for accurate distance function estimation. The core of our approach consists of formulating a new variational problem, whose solution is the distance function and solving this variational problem numerically by ADMM [19]. We review several other variational and PDE-based methods for distance function estimation, such as the recent geodesics-in-heat method [11, 12]. We show how to discretize the variational problem by the finite element method, and by using finite difference. In particular, we present the Matlab code for implementing our method (as well as the other variational methods) for computing the distance transform of a 2D binary image. We compare the results obtained by our approach with those of the other methods, and also show how our method can be used for estimating surface curvatures and approximating the skeleton of a 2D binary image.

## 2 Distance function computation

### 2.1 Distance function properties

Consider a domain $\Omega \subset \mathbb{R}^{m}$ bounded by $\partial \Omega$ oriented by its inner normal $\boldsymbol{n}$. Denoted by $|\boldsymbol{q}|$, the magnitude of vector $\boldsymbol{q}=\left(q_{1}, \ldots, q_{m}\right)^{T},|\boldsymbol{q}|=\sqrt{q_{1}^{2}+\cdots+q_{m}^{2}}$. Let $d(\boldsymbol{x})$ be the signed distance function from $\partial \Omega$. The distance function satisfies the eikonal equation

$$
\begin{equation*}
|\nabla d|=1 \quad \text { in } \quad \Omega \tag{1}
\end{equation*}
$$

and boundary conditions

$$
\begin{equation*}
d=0, \quad \partial d / \partial \boldsymbol{n}=1, \quad \text { and } \quad \partial^{k} d / \partial \boldsymbol{n}^{k}=0, \quad k=2,3, \ldots \quad \text { on } \quad \partial \Omega \tag{2}
\end{equation*}
$$

Typically, (1) is used with the first (Dirichlet) boundary condition in (2).

### 2.2 Screened Poisson distance function approximations

A simple PDE-based approach to estimate $d(\boldsymbol{x})$ consists of considering a Dirichlet boundary value problem for a screened Poisson equation

$$
\begin{equation*}
w(\boldsymbol{x})-t \Delta w(\boldsymbol{x})=0 \quad \text { in } \quad \Omega, \quad w=1 \quad \text { on } \quad \partial \Omega, \tag{3}
\end{equation*}
$$

where $t$ is a small positive parameter. As shown in [30, Theorem 2.3],

$$
\begin{equation*}
\lim _{t \rightarrow 0} \sqrt{t} \ln [1 / w(\boldsymbol{x})]=d(\boldsymbol{x}) \tag{4}
\end{equation*}
$$

Thus, $\sqrt{t} \ln [1 / w(\boldsymbol{x})]$ approximates $d(\boldsymbol{x})$ and the parameter $t$ controls the approximation accuracy. This approximation of the distance function was used in [28] for detecting skeletal structures in grayscale images.

A simple heuristic behind (4) can be found, for example, in [20] and uses the Hopf-Cole transformation [13]. It is easy to check that substituting $w(\boldsymbol{x})=$ $\exp \{-v(\boldsymbol{x}) / \sqrt{t}\}$ in (3) yields $|\nabla v|^{2}=1+\sqrt{t} \Delta v$, which for small $t$ can be considered as a regularized version of eikonal (1).

The geodesics-in-heat method [11, 12], a highly popular distance function approximation, is a close relative of the above approach extended to non-flat geometry. The method is based on the observation that $w(\boldsymbol{x})$ and $v(\boldsymbol{x})$ have the same level sets and construct a distance function approximation $u(\boldsymbol{x})$ as the solution of the following minimization problem

$$
\begin{equation*}
\int_{\Omega}|\nabla u-\boldsymbol{m}|^{2} d \boldsymbol{x} \rightarrow \min , \quad u=0 \text { on } \partial \Omega \tag{5}
\end{equation*}
$$

where $\boldsymbol{m}=\nabla w /|\nabla w|$ and $w(\boldsymbol{x})$ is the solution to (3).
Practical implementation of the geodesics-in-heat method includes numerical solutions to the screened Poisson (3) and the Poisson equation corresponding to the variational problem (5).

It is also interesting that (3) combined with either (4) or (5) can serve as a warm start to iterative methods for computing the distance function.

Although both the methods, (4) and (5), are based on screened Poisson (3), to distinguish them we refer to the former as the screened Poisson method and to the latter as the heat method. In Section 4, we deal with approximate distance transforms for 2D binary images and provide the reader with simple MATLAB-based implementations of both methods.

### 2.3 Proposed distance function approximation

We now turn to the description of our approach. A key observation is given by the following proposition.

Proposition 1 The distance function $d(\boldsymbol{x})$ delivers the solution to the following energy minimization problem

$$
\begin{equation*}
\int_{\Omega} \phi d x \longrightarrow \max , \quad \text { where } \quad \max _{\Omega}|\nabla \phi| \leq 1 \quad \text { and } \quad \phi=0 \quad \text { on } \quad \partial \Omega \tag{6}
\end{equation*}
$$

Proof Indeed, in order to maximize $\int_{\Omega} \phi d \boldsymbol{x}$, the function $\phi(\boldsymbol{x})$ has to grow as fast as possible and its gradient magnitude $|\nabla \phi|$ achieves at each point $\boldsymbol{x}$ its maximal allowed value. Thus, the solution to (6) satisfies

$$
|\nabla \phi|=1 \text { in } \Omega \quad \text { and } \quad \phi=0 \text { on } \partial \Omega
$$

and, therefore, is the distance function $d(\boldsymbol{x})$.
The constrained energy minimization problem (6) can be reformulated as the following unconstrained minimization problem

$$
\begin{equation*}
F(\phi)+G(\nabla \phi) \rightarrow \min , \tag{7}
\end{equation*}
$$

where

$$
F(\phi)=-\int_{\Omega} \phi d \boldsymbol{x}, \quad \phi=0 \text { on } \partial \Omega, \quad G(\boldsymbol{q})= \begin{cases}0 & \text { if }\|\boldsymbol{q}\|_{L^{\infty}} \leq 1 \\ +\infty & \text { otherwise }\end{cases}
$$

We solve (7) numerically by ADMM. The corresponding augmented Lagrangian has the form

$$
\begin{equation*}
L(\phi, \boldsymbol{q}, \sigma)=\int_{\Omega}\left(-\phi+G(\boldsymbol{q})+\sigma \cdot(\nabla \phi-\boldsymbol{q})+\frac{r}{2}|\nabla \phi-\boldsymbol{q}|^{2}\right) d \boldsymbol{x}, \tag{8}
\end{equation*}
$$

where $\sigma$ is the vector of Lagrange multipliers and $r>0$ is a regularization parameter. Now ADMM for (8) yields the following iterative process:

```
repeat
```

Minimize (8) w.r.t. $\phi(\boldsymbol{x})$ by solving $-r\left(\Delta \phi_{k+1}-\operatorname{div} \boldsymbol{q}_{k}\right)=1+\operatorname{div} \sigma_{k}$ in $\Omega$, $\phi_{k+1}=0$ on $\partial \Omega ;$

Perform the projection $\boldsymbol{q}_{k+1}=P_{B}\left(\nabla \phi_{k+1}+\sigma_{k} / r\right)$, where
$P_{B}(z)=\left\{\begin{array}{ll}z & \text { if }\|z\|_{L^{\infty}} \leq 1 \\ z /\|z\|_{L^{\infty}} & \text { otherwise }\end{array} ;\right.$
Update the Lagrange multipliers by $\sigma_{k+1}=\sigma_{k}+r\left(\nabla \phi_{k+1}-\boldsymbol{q}_{k+1}\right)$;
until convergence

The iterative approach presented here shares a lot of similarities with the numerical method proposed in [15], where ADMM is used to numerically compute the solution to a $p$-Poisson problem $-\Delta_{p} u=1$. The limit as $p \rightarrow \infty$ of the solution to this $p$ Poisson problem, with vanishing Dirichlet boundary condition, is $d(\boldsymbol{x})$, the distance
to the boundary function [5]. The main difference with the approach proposed here lies in the second step of minimizing (8) by ADMM, where a simple projection can now be used instead of numerically computing the root of a polynomial.

## 3 Numerical experiments: distance to a polygonal mesh using FEM and application to curvature computation

We consider first the problem of computing the distance function to $\partial \Omega$, a surface (curve in 2D), represented by a triangle mesh (a polygonal chain in 2D).

As described in Section 2, we use ADMM to minimize (7). This gives an iterative process, where the first step involves solving a Poisson problem of the form: $-\Delta \phi=f$. This Poisson problem is discretized and solved by the finite elements method. The computational domain bounded by $\partial \Omega$ is represented by a triangle mesh in $2 D$ or a tetrahedral mesh in 3D. Linear basis functions are used at each node of the triangulation. The solution to the Poisson problem is obtained from numerically solving a linear system $\boldsymbol{A} \Phi=\boldsymbol{b}$, where the sparse matrix $\boldsymbol{A}$, corresponding to a discretization of the Laplacian, is the same for each iteration of ADMM and can thus be prefactored (for example with the Cholesky decomposition).

### 3.1 Distance computation

Figure 1 illustrates the result obtained by our approach on a 2D polygonal domain with complex geometry. The left image visualizes the exact distance obtained by computing the minimum distance to any boundary segment. The middle image presents the distance computed by solving (7). The right image demonstrates how the relative residual error

$$
\left\|\phi_{k+1}-\phi_{k}\right\|_{2} /\left\|\phi_{k}\right\|_{2}
$$

decreases with each iteration when solving (7) by ADMM. One can observe that (7) solved numerically by ADMM demonstrates a good convergence and is capable to deliver an accurate approximation of the distance function.


Fig. 1 Left: The exact distance computed as the min distance to the boundary segments. Middle: Distance to the boundary obtained from solving (7) by ADMM. Right: Relative residual error over a few ADMM iterations


Fig. 2 Exact and approximate distance for two tetrahedral meshes, Armadillo and Fertility. Filled contour plots on a planar slice of the domain $\Omega$ are shown. Left: The exact distance computed as the min distance to the boundary triangles. Middle: Distance to the boundary obtained from solving (7) by ADMM. Right: Relative residual error over a few ADMM iterations

As seen in Fig. 2, a similar performance of (7) with ADMM is achieved for 3D models. In this figure, we visually compare on a planar slice of $\Omega$ the function obtained by numerically solving (7) and the exact distance, which is obtained by computing at a given node of the tetrahedral mesh the minimal distance to the set of triangles corresponding to the input surface $\partial \Omega$.

### 3.2 Comparison to other approaches

In Figs. 3 and 4, we compare the distance function estimation results achieved by solving (7) with those obtained by using the heat method (3) and (5). The results are shown on a planar slice of the domains.

As seen in Fig. 3, the distance function approximation obtained from minimizing (7) delivers the best result. This is also confirmed by Fig. 4, where the point-wise relative error w.r.t. the exact distance

$$
|u(\boldsymbol{x})-d(\boldsymbol{x})| / d(\boldsymbol{x})
$$

is visualized on a planar slice of the domain. Here $u(\boldsymbol{x})$ is the distance function approximation obtained by either (7) with ADMM, or the heat method (3) and (5).

One can notice that (7) delivers the best approximation. Some small regions with relatively high errors are likely due to the numerical solution of the ADMM


Fig. 3 Left: Distance function approximation by the heat method [11]. Middle: Our new variational problem (7) solved numerically by ADMM. Right: The exact distance to the boundary. Results are visualized on a planar slice of the domain
sub-problems by the finite element method. Mesh refinement and mesh moving techniques can be used to improve the results.

### 3.3 Application to surface curvature estimation

We can further use the computed distance to approximate the curvature of the boundary surface $\partial \Omega$. Surface curvature estimation is important for many computer graphics and geometric modeling applications [8] and remains a subject of intensive research [21, 27] (see also references therein). It turns out that the distance function from a surface contains full information about the surface curvatures. Namely, the eigenvalues of the Hessian $H$ of the distance function $d$ at a boundary point $\boldsymbol{y}$ are $\left\{\kappa_{1}, \kappa_{2}, 0\right\}$ and the corresponding eigenvectors are $\left\{\boldsymbol{t}_{1}, \boldsymbol{t}_{2}, \boldsymbol{n}\right\}$, where $\boldsymbol{t}_{1}, \boldsymbol{t}_{2}$ are the principal curvature directions at $\boldsymbol{y}$ and $\boldsymbol{n}$ the outward normal to the surface at $\boldsymbol{y}$. See, for example, [18, Section 14.6].


Fig. 4 Point-wise relative error w.r.t. the exact distance for the distance approximation obtained by the heat method [11] (two left images) and our variational problem (7) solved numerically by ADMM (two right images) visualized on a planar slice of the domain

Given an approximation of the distance, computed by numerically solving (7), and sampled on a tetrahedral mesh, we first approximate the gradient of the distance function $\nabla d$ at each node of the tetrahedral mesh as the normalized weighted average of the gradients in each incident tetrahedron (the volume of the tetrahedron is used as the weight). By repeating this procedure for each component of $\nabla d$, we obtain an expression of the Hessian of the distance function at each node of the tetrahedral mesh. We then compute the non zero eigenvalues of the Hessian at each boundary node (nodes of the tetrahedral mesh that are on the surface boundary).

Figure 5 illustrates results obtained by this approach for computing the mean curvature, $H=\left(\kappa_{1}+\kappa_{2}\right) / 2$ of some surfaces.

## 4 Approximate distance transforms for 2D binary images and skeleton computation

### 4.1 Approximate distance transforms for 2D binary images

Constructing exact and approximate distances from boundaries of objects is also important for a number of image processing applications [14], where it is usually called the distance transform. In particular, shape skeletonization, which itself has numerous applications in image processing and computer graphics [26], can be used as a quality indicator for distance function approximations.

In this section, we focus on computing approximate distance functions for the boundaries of objects represented by 2D binary images. Since the most popular way to encode 2D images consists of representing them using grids of numbers (pixels), we deal with finite difference approximations. We present matlab-based implementations of the screened Poisson method (3) and (4), heat method (5) of Crane, Weischedel, and Wardetzky [11, 12], and our method (7). We demonstrate that typically our method produces a more accurate approximation of the distance function. Finally, we present a simple way to extract skeletons of binary shapes.


Fig. 5 Mean curvature for the bunny and the fertility models, computed from the eigenvalues of the Hessian of the distance function
matlab allows for a very simple implementation of the screened Poisson based and heat distance function approximations discussed in Section 2.2. Since in this section we use them in our numerical experiments and comparison, we present their implementations below. First, our matlab implementation of (3) and (4) for approximating the distance to the shape of a binary image looks as follows:

```
function U = screened_Poisson_dist(I, t)
[row,col] = find(I > 0);
v = (1: length(row))';
G = zeros(size(I));
for i = 1:length(row)
    G(row(i), col(i)) = v(i);
end
D = delsq(G);
N = sum(G(:)>0);
rhs = ones(N,1); % right-hand side = 1
u}=-\textrm{sqrt}(\textrm{t})*\operatorname{log}(1-(\textrm{t}*\textrm{D}+\operatorname{speye}(\textrm{N}))\\mathrm{ rhs );
U = G;
U(G>0) = u(G(G>0));
end
```

Next, our matlab implementation of the heat method [11, 12] solving (3) and minimizing (5) reads as follows:

```
function U = heat_dist(I,t)
[row,col] = find(I > 0);
v = (1: length(row))';
G = zeros(size(I));
for i = 1:length(row)
    G(row(i),col(i)) = v(i);
end
D = delsq(G);
N = sum(G(:)>0);
rhs = ones(N,1); % right-hand side = 1
u = (t*D+speye(N))\rhs; % solving screened Poisson
U = G;
U(G>0) = u(G(G>0));
[Ux,Uy] = gradient(U);
g = sqrt(Ux.*Ux+Uy.*Uy);
Ux = Ux./g;
Uy = Uy./g;
div = -divergence(Ux,Uy);
w = D\div(G>0); % solving Poisson equation
U(G>0) = w(G(G>0));
end
```

Finally, a simple implementation of the ADMM-based minimization of (7) in MATLAB, relying on its built-in functions, is shown below

```
function U = our_dist(I, niter, r)
[row,col] = find(I > 0);
v = (1:length(row))';
G = zeros(size(I));
for i = 1:length(row)
    G(row(i),col(i)) = v(i);
end
D = delsq(G); % 5-point negative Laplacian
N = sum(G(:) 0); % Number of interior points
f1 = ones(N,1)./r;
sigma = zeros(size(I,1),size(I,2),2); % multipliers
[dI(:,:,1), dI(:,:,2)] = gradient(I);
for i = 1:niter
    q = dI - sigma./r;
    gu = repmat(sqrt(q(:,:,1).^2+q(:,:,2).^2),[11 1 2]);
    xi = q;
    xi(gu>1) = q(gu>1)./gu(gu>1);
    sigma = sigma + r .* (xi - dI);
    xi2 = xi + (1.0/r).*sigma;
    % solve -delta u = -div (\xi + \sigma/r) + f/r
    div = divergence(xi2(:,:,1), xi2(:,:,2));
    rhs = div(G>0) + f1;
    u = D rhs;
    U = G;
    U(G>0) = full(u(G(G>0)));
    [dI(:,:,1), dI(:,:,2)] = gradient(U);
end
end
```

In the numerical experiments presented below, we compare these three methods (the screened Poisson (3) and (4), the heat method (3) and (5), and the proposed approach (7)) for computing the distance to the boundary of binary shapes. We set $t=0.6$ for both the screened Poisson and heat methods, as choosing a smaller value may lead to instabilities. We set $r=10$ and use 50 iterations for our method.

In our experiments, we use two simple geometric shapes (mushroom and keyhole) and four more complex shapes from the Kimia-99 dataset [25]. Figure 6 illustrates the results obtained by these methods on the test shapes. The bottom three rows show the point-wise absolute error of the screened Poisson, the heat method, and our approach, respectively, compared with the exact distance. Quantitative results are provided in Table 1, where the RMS and maximum error for each of the test shapes is provided.

The heat method (5) demonstrates a better stability than the screened Poisson one (3) and (4). In particular, the latter collapses for some images tested above if we


Fig. 6 Distance estimation on test images. First row (from top to bottom): input binary images. Second row: the exact distance from the boundary. Third row: absolute error for the screened Poisson based distance function approximations (4). Fourth row: absolute error for the heat distance function approximations (5). Fifth row: absolute error for the proposed distance function approximations (7)
set $t=0.5$, while the heat method demonstrates slightly better performance. On the other hand, our method (7) can also benefit from a more accurate selection of the regularization parameter $r$. For example, setting $r=7$ makes our method the absolute RMS and maximum error winner for all the images considered above.

Table 1 RMS and maximum errors for the images in Fig. 6

|  | RMS error |  |  | Maximum error |  |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: |
|  | Varadhan | Heat | Ours | Varadhan | Heat | Ours |
| Mushroom | 0.35 | 0.26 | 0.16 | 1.72 | 1.84 | 0.79 |
| Keyhole | 0.26 | 0.22 | 0.14 | 1.63 | 1.25 | 0.69 |
| Hand | 0.25 | 0.26 | 0.16 | 1.67 | 1.86 | 0.96 |
| Cat | 0.17 | 0.19 | 0.14 | 1.21 | 1.77 | 1.37 |
| Donkey | 0.17 | 0.18 | 0.12 | 1.50 | 1.63 | 0.99 |
| Fish | 0.10 | 0.12 | 0.07 | 1.02 | 1.40 | 1.01 |

Bold font is used to indicate the best results

### 4.2 2D binary image skeletonization

The proposed approach for distance estimation can be used to compute skeletons of binary images. The skeleton, or medial axis, is an important shape descriptor, which was introduced more than fifty years ago [7] and which remain a topic of active research, see, for example, [1, 26, 32, 33]. The skeleton of a 2D shape can be defined as the locus of centers of inner bitangent circles. Alternatively, given the distance function to the boundary of a shape, the skeleton can be defined as the set of inner distance function singularities. In this study, we use a smooth distance function for constructing an approximate skeleton of a binary image.

Since the magnitude of the gradient of the distance function is equal to one everywhere except at the distance function singularities, it is natural to expect that the points where the gradient of a smooth distance function is small form a thick/fuzzy version of the skeleton. Once such a thick/fuzzy skeleton is extracted, one can get a one-pixel-wide skeleton by applying MATLAB's bwmorph function to the thick/fuzzy skeleton.

In Fig. 7, we demonstrate how this gradient-of-smooth-distance-function approach works and compare it with the standard thinning-based binary image skeletonization procedure consisting of applying bwmorph to the whole binary image. Given a binary image, we start from using (7) for computing a smooth distance function $u(\boldsymbol{x})$. Then, the fuzzy-skeleton function and its normalized version

$$
\begin{equation*}
S(\boldsymbol{x})=\sqrt{u(\boldsymbol{x})\left(1-|\nabla u(\boldsymbol{x})|^{2}\right)} \quad \text { and } \quad S_{n}(\boldsymbol{x})=S(\boldsymbol{x}) / \max _{\boldsymbol{x}}[S(\boldsymbol{x})] \tag{9}
\end{equation*}
$$

are evaluated. Here, the factor $\sqrt{u(\boldsymbol{x})}$ is used to suppress image skeletal structures appearing due to small perturbations of the image boundary. As demonstrated by the


Fig. 7 Using the distance approximation (7) for skeleton extraction. Top row: our distance function approximation (7) is used to compute a gradient-based function $S_{n}(\boldsymbol{x})$ defined by (9). Middle row: the skeletons obtained by applying simple thresholding and thinning to $S_{n}(\boldsymbol{x})$. Bottom row: the skeletons obtained by thining the original binary images with bwmorph
top row images of Fig. 7, $S_{n}(\boldsymbol{x})$ delivers a good detection of skeletal structures of the binary images. Applying simple thresholding and thinning by bwmorph yields one-pixel-wide skeletons (the middle row of Fig. 7), which are of higher quality than those generated by thinning the original binary images with bwmorph (the bottom row).

## 5 Conclusion

In this paper, we have proposed a new variational problem for the distance-fromsurface function (7) and showed how it can be efficiently solved using ADMM. We have presented the results of our numerical experiments when the problem is discretized by FEM or finite differences. We have demonstrated advantages of (7) over the heat method [11] and shown that our approach can be used for surface curvature estimation and skeletonization of 2D binary images.

Acknowledgments The Armadillo and the Bunny mesh models are courtesy of the Stanford Graphics Laboratory. The Fertility mesh model is courtesy of AIM@SHAPE. The Kimia-99 binary shape dataset [25] was downloaded from https://github.com/mmssouza/kimia99.

Open Access This article is distributed under the terms of the Creative Commons Attribution 4.0 International License (http://creativecommons.org/licenses/by/4.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license, and indicate if changes were made.

## References

1. Aubert, G., Aujol, J.F.: Poisson skeleton revisited: a new mathematical perspective. J. Math. Imaging Vis. 48(1), 149-159 (2014)
2. Babuška, I., Banerjee, U., Osborn, J.E.: Survey of meshless and generalized finite element methods: a unified approach. Acta Numerica 12, 1-125 (2003)
3. Belyaev, A., Fayolle, P.A.: On variational and PDE-based distance function approximations. Comput. Graph. Forum 34(8), 104-118 (2015)
4. Belyaev, A., Fayolle, P.A., Pasko, A.: Signed $L_{p}$-distance fields. Comput. Aided Des. 45, 523-528 (2013)
5. Bhattacharya, T., DiBenedetto, E., Manfredi, J.: Limits as $p \rightarrow \infty$ of $\Delta_{p} u_{p}=f$ and related extremal problems. Rend. Sem. Mat. Univ. Pol. Torino, Fascicolo Speciale Nonlinear PDEs, 15-68 (1989)
6. Biswas, A., Shapiro, V., Tsukanov, I.: Heterogeneous material modeling with distance fields. Comput. Aided Geom. Des. 21, 215-242 (2004)
7. Blum, H.: A transformation for extracting new descriptors of shape. Models for the perception of speech and visual form 19(5), 362-380 (1967)
8. Botsch, M., Kobbelt, L., Pauly, M., Alliez, P., Lévy, B.: Polygon Mesh Processing. AK Peters/CRC Press (2010)
9. Cadena, C., Carlone, L., Carrillo, H., Latif, Y., Scaramuzza, D., Neira, J., Reid, I., Leonard, J.: Past, present, and future of simultaneous localization and mapping: $t$ the robust-perception age. IEEE Trans. Robot. 32, 1309-1332 (2016)
10. Calakli, F., Taubin, G.: SSD: Smooth signed distance surface reconstruction. Comput. Graph. Forum 30(7), 1993-2002 (2011)
11. Crane, K., Weischedel, C., Wardetzky, M.: Geodesics in heat: a new approach to computing distance based on heat flow. ACM Trans. Graph. 32, 152:1-152:11 (2013)
12. Crane, K., Weischedel, C., Wardetzky, M.: The heat method for distance computation. Commun. ACM 60, 90-99 (2017)
13. Evans, L.C.: Partial Differenetial Equations. American Mathematical Society (1998)
14. Fabbri, R., Costa, L.D.F., Torelli, J.C., Bruno, O.M.: 2D Euclidean distance transform algorithms: a comparative survey. ACM Comput. Surv. 40, 2:1-2:49 (2008)
15. Fayolle, P.A., Belyaev, A.: p-laplace diffusion for distance function estimation, optimal transport approximation, and image enhancement. Comput. Aided Geom. Des. 67, 1-20 (2018)
16. Freytag, M., Shapiro, V., Tsukanov, I.: Finite element analysis in situ. Finite Elem. Anal. Des. 47(9), 957-972 (2011)
17. Gibou, F., Fedkiw, R., Osher, S.: A review of level-set methods and some recent applications. J. Comput. Phys. 353, 82-109 (2017)
18. Gilbarg, D., Trudinger, N.S.: Elliptic Partial Differential Equations of Second Order. Classics in Mathematics. Springer (1998)
19. Glowinski, R.: On alternating direction methods of multipliers: a historical perspective. In: Fitzgibbon, W., Kuznetsov, Y.A., Neittaanmäki, P., Pironneau, O. (eds.) Modeling, Simulation and Optimization for Science and Technology, pp. 59-82. Springer (2014)
20. Gurumoorthy, K.S., Rangarajan, A.: A Schrödinger equation for the fast computation of approximate Euclidean distance functions. In: Scale Space and Variational Methods in Computer Vision (SSMV 2009). LNCS, vol. 5567, pp. 100-111. Springer (2009)
21. Kronenberger, M., Wirjadi, O., Hagen, H.: Empirical comparison of curvature estimators on volume images and triangle meshes. IEEE Transactions on Visualization and Computer Graphics (2018)
22. Lee, B., Darbon, J., Osher, S., Kang, M.: Revisiting the redistancing problem using the Hopf-Lax formula. J. Comput. Phys. 330, 268-281 (2017)
23. Roget, B., Sitaraman, J.: Wall distance search algorithm using voxelized marching spheres. J. Comput. Phys. 241, 76-94 (2013)
24. Royston, M., Pradhana, A., Lee, B., Chow, Y.T., Yin, W., Teran, J., Osher, S.: Parallel redistancing using the Hopf-Lax formula. J. Comput. Phys. 365, 7-17 (2018)
25. Sebastian, T.B., Klein, P.N., Kimia, B.B.: Recognition of shapes by editing their shock graphs. IEEE Trans. Pattern Anal. Mach. Intell. 5, 550-571 (2004)
26. Siddiqi, K., Pizer, S.M.: Medial Representations: Mathematics, Algorithms and Applications, vol. 37. Springer (2008)
27. Tang, Y., Li, H., Sun, X., Morvan, J.M., Chen, L.: Principal curvature measures estimation and application to 3D face recognition. J. Math. Imaging Vis. 59(2), 211-233 (2017)
28. Tari, Z.S.G., Shah, J., Pien, H.: Extraction of shape skeletons from grayscale images. Comput. Vis. Image Underst. 66(2), 133-146 (1997)
29. Tucker, P.G.: Hybrid Hamilton-Jacobi-Poisson wall distance function model. Comput. Fluids 44(1), 130-142 (2011)
30. Varadhan, S.R.S.: On the behavior of the fundamental solution of the heat equation with variable coefficients. Comm. Pure Appl. Math. 20, 431-455 (1967)
31. Xia, H., Tucker, P.G.: Fast equal and biased distance fields for medial axis transform with meshing in mind. Appl. Math. Model. 35, 5804-5819 (2011)
32. Yan, Y., Letscher, D., Ju, T.: Voxel cores: Efficient, robust, and provably good approximation of 3d medial axes. ACM Trans. Graph. 37(4), 44,1-44,13 (2018)
33. Yan, Y., Sykes, K., Chambers, E., Letscher, D., Ju, T.: Erosion thickness on medial axes of 3d shapes. ACM Trans. Graph. 35(4), 38,1-38,12 (2016)
34. Zhu, B., Skouras, M., Chen, D., Matusik, W.: Two-scale topology optimization with microstructures. ACM Trans. Graph. 36(5), 36:1-36,14 (2017)
35. Zollhöfer, M., Dai, A., Innmann, M., Wu, C., Stamminger, M., Theobalt, C., Nießner, M.: Shadingbased refinement on volumetric signed distance functions. ACM Trans. Graph. 34(4), 96:1-96,14 (2015)

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.


[^0]:    Alexander Belyaev
    a.belyaev@hw.ac.uk

    Pierre-Alain Fayolle fayolle@u-aizu.ac.jp

    1 School of Engineering \& Physical Sciences, Heriot-Watt University, Edinburgh, UK
    2 Computer Graphics Laboratory, University of Aizu, Aizu-Wakamatsu, Japan

