Surface reconstruction from point clouds using a novel variational model

Chapter · November 2015

CITATION
1

READS
2,330

1 author:

Jinming Duan
University of Birmingham
114 PUBLICATIONS **1,573** CITATIONS

SEE PROFILE

Surface reconstruction from point clouds using a novel variational model

Jinming Duan, Ben Haines, Wil O. C. Ward and Li Bai

Abstract Multi-view reconstruction has been an active research topic in the computer vision community for decades. However, state of the art 3D reconstruction systems have lacked the speed, accuracy, and ease to use properties required by the industry. The work described in this paper is part of the effort to produce such a multi-view reconstruction system for a UK company. A novel variational level set method is developed for reconstructing implicit surfaces from unorganised point clouds. A distance function is calculated for the point cloud using a new 3D fast sweeping algorithm and used to find a good initial surface close to the point cloud. A novel variational model is then used to evolve this initial surface. The model consists of three energy terms to ensure accurate and smooth surface reconstruction whilst preserving the small details of the object and increasing reconstruction speed. The model also completely eliminated the need for reinitialisation associated with the level set method. Implementation details of the variational model are given and the functions of the three energy terms of the model are illustrated through numerical experiments. Gradient descent optimisation is used to find the minimum of the proposed variational model and accurately reconstruct the surface. The proposed method is validated and experiments show that the proposed method outperformed the state of the art surface reconstruction approaches.

Jinming Duan
School of Computer Science, University of Nottingham, UK. e-mail: psxjd3@nottingham.ac.uk

Ben Haines
School of Computer Science, University of Nottingham, UK. e-mail: psxbh@nottingham.ac.uk

Wil O. C. Ward
School of Computer Science, University of Nottingham, UK. e-mail: psxwowa@nottingham.ac.uk

Li Bai
School of Computer Science, University of Nottingham, UK. e-mail: Bai.Li@nottingham.ac.uk

1 Introduction

To reconstruct an accurate and smooth 3D surface from a point cloud is a challenging problem as the point cloud consists of scattered points, unorganised and unconnected. In general, there exist two categories of surface representations: explicit or implicit representation. Explicit representation [1–3] describes the location of points on a surface as well as the local geometry of a surface in a explicit manner. It can be accurate, but less robust and less flexible in handling arbitrary and dynamically changing surface topology. Implicit representation [4–7] usually considers a surface as the zero level set of a higher dimensional implicit function (i.e. level set function). In addition to the topological flexibility and robustness, it is more suitable for noisy and non-uniform point clouds [6, 7].

The level set method [8] is a powerful technique to track dynamic interfaces. Surface reconstruction methods [4, 5] using level sets involves the construction of an initial surface which then evolves towards the dataset according to a set of partial differential equations. One of the most successful level set based surface reconstruction methods was proposed by Zhao et al in [4, 5]. Though their model works quite well, there are some disadvantages. First, periodical reinitialisation is needed in their model to keep the evolving level set close to the signed distance function to maintain stable surface evolution and desirable results. However, this procedure is tedious, expensive and may even cause the final surface to shrink to a undesirable position [9]. Second, their model is non-convex, so the results are sensitive to the initial condition. A fast tagging algorithm is developed in [5] to make the initial surface lie close to the true surface. However, if the sampling of the data contains small features or concave regions, the evolving surface often gets stuck in local minimum even if the initial surface is very close to the true surface.

In this paper, a novel variational model is introduced for surface reconstruction from point clouds which overcomes the problems with the existing implicit reconstruction methods. A explicit 3D fast sweeping algorithm is presented extending the 2D version [10] to construct a distance function for the point cloud and used to find the 3D volume enclosed by the point cloud. A good initial surface close to the point cloud is then found for the variational model to evolve. The variational model consists of three energy terms to ensure the reconstruction (1) is smooth and maintains a signed distance function; (2) preserves the small details of the object; (3) proper reconstruction of concave object regions and speed up surface evolution. Gradient descent optimisation is then used to find the minimum of the proposed variational model and accurately reconstruct the surface. Comparison with popular reconstruction methods, including the Poisson method [11], shows that the proposed method has outperformed the state of the art methods.

2 Preprocessing

2.1 Distance function for point cloud using fast sweeping

Given an unorganised 3D point cloud $\{x_i\}$, its distance function $d(x)$ satisfies the following Eikonal equation

$$|\nabla d(x)| = f(x), x \in \Omega \backslash \{x_i\} \tag{1}$$

with $f(x) = 1$ and

$$d(x_i) = 0, x \in \{x_i\}$$

(1) is a typical partial differential equation and it can be solved efficiently by using fast sweeping algorithm proposed by Zhao for two-dimensional (2D) problems [10]. In this paper, the 2D fast sweeping algorithm is extended to 3D. To do so, the Godunov upwind difference scheme is used to discretise (1) as follows:

$$\left[(d^n_{i,j,k} - d^n_{xmin})^+\right]^2 + \left[(d^n_{i,j,k} - d^n_{ymin})^+\right]^2 + \left[(d^n_{i,j,k} - d^n_{zmin})^+\right]^2 = f^2_{i,j,k} \tag{2}$$

In equation (2), $d^n_{xmin} = min(d^n_{i,j+1,k}, d^n_{i,j-1,k})$, $d^n_{ymin} = min(d^n_{i+1,j,k}, d^n_{i-1,j,k})$, $d^n_{zmin} = min(d^n_{i,j,k+1}, d^n_{i,j,k-1})$ and $x^+ = \begin{cases} x & x > 0 \\ 0 & x \leq 0 \end{cases}$. Boundary conditions need to be handled in the computational grid space, and one-sided upwind difference is used for each of the 6 boundary faces of the grid space. For example, at the left boundary face, a one-sided difference along the x direction is computed as follows

$$\left[(d^n_{i,1,k} - d^n_{i,2,k})^+\right]^2 + \left[(d^n_{i,1,k} - d^n_{ymin})^+\right]^2 + \left[(d^n_{i,1,k} - d^n_{zmin})^+\right]^2 = f^2_{i,1,k}$$

d^n_{xmin}, d^n_{ymin} and d^n_{zmin} are then sorted in increasing order and the sorted version is recorded as a_1, a_2 and a_3. So, the unique solution to (2) is given as follows:

$$d^{n+1}_{i,j,k} = min(d^n_{i,j,k}, \widetilde{d}_{i,j,k}) \tag{3}$$

where $\widetilde{d}_{i,j,k}$ is a piecewise function containing three parts

$$\widetilde{d}_{i,j,k} = \begin{cases} \dfrac{a_1 + a_2 + a_3 + \sqrt{3f^2_{i,j,k} - (a_1 - a_2)^2 - (a_1 - a_3)^2 - (a_2 - a_3)^2}}{3} \\ \dfrac{a_1 + a_2 + \sqrt{2f^2_{i,j,k} - (a_1 - a_2)^2}}{2} \\ a_1 + f_{i,j,k} \end{cases}$$

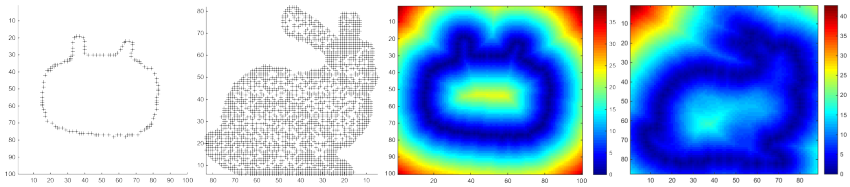
The three parts correspond to the following intervals, respectively

$$f^2_{i,j,k} \geq (a_1 - a_3)^2 + (a_2 - a_3)^2$$
$$(a_1 - a_2)^2 \leq f^2_{i,j,k} < (a_1 - a_3)^2 + (a_2 - a_3)^2$$

$$f_{i,j,k}^2 < (a_1 - a_2)^2$$

To solve (3), which is not in analytical form, the fast Gauss-Seidel iteration with alternating sweeping orderings is used. For initialization, the points of the unorganised point cloud are set to zero, and the rest of the points are set to large values. Specifically, the whole 3D grid is traversed in the following orders.

$$(1)\ i = 1:M, j = 1:N, k = 1:H;\ (2)\ i = M:1, j = N:1, k = H:1$$
$$(3)\ i = M:1, j = 1:N, k = 1:H;\ (4)\ i = 1:M, j = N:1, k = H:1$$
$$(5)\ i = M:1, j = N:1, k = 1:H;\ (6)\ i = 1:M, j = 1:N, k = H:1$$
$$(7)\ i = 1:M, j = N:1, k = 1:H;\ (8)\ i = M:1, j = 1:N, k = H:1$$



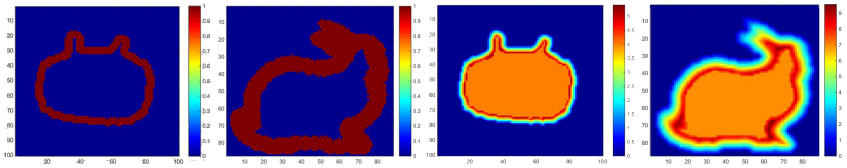
(a) (b) (c) (d)

Fig. 1 Calculating the distance function of the original point cloud. (a)-(b) are 2D and 3D point cloud respectively. (c) is the distance function for (a). (d) is a slice of the distance function for (b).

2.2 Calculating volumetric data from distance function

After eight directional sweeping using fast Gauss-Seidel iteration, the distance function $d(x)$ in the Eikonal equation (1) can be solved. Fig. 1 shows two examples for 2D and 3D data. $d(x)$ is zero at the points of the original point cloud and its values are very small at the points close to the original point cloud, and very large at the points far away from the original point cloud. Based on this, a closed annular binary image $I(x)$, shown in the first two images in Fig. 2, can be obtained by thresholding the distance function $d(x)$.

With the annular binary image $I(x)$, the fast sweeping algorithm is again applied to find the volumetric data $u(x)$ enclosed by the point cloud, as shown in the last two images in Fig. 2. First, $f(x) = I(x)$ is taken in the right-hand side of (3.1) instead of $f(x) = 1$. Second, in order to calculate the objective function $d(x)$ (it is not distance function when $f(x) \neq 1$) in (1), zero values are assigned to the grid points in the 6 boundary faces and very large values assigned to other grid points for eight directional sweeping Gauss-Seidel iterations. Once the objective function $d(x)$ is found, so the volumetric data $u(x) = d(x)$. The volumetric data $u(x)$ is used to find a good initial surface to speed up variational level set evolution in section 2.3.

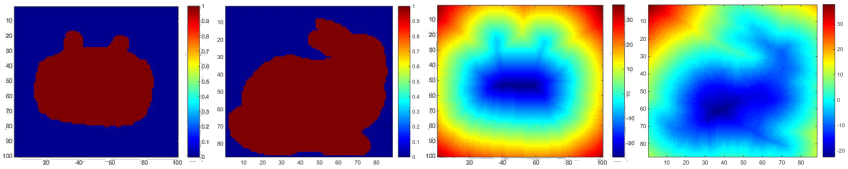


(a) (b) (c) (d)

Fig. 2 Obtain volumetric data from the annular binary image. (a)-(b) are two annular binary images. (c)-(d) are corresponding volumetric data of (a) and (b) respectively. (b) and (d) are one slice of its corresponding 3D data.

2.3 Surface initialization using volumetric data

The volumetric data $u(x)$ is first thresholded to obtain a new binary image $s(x)$ shown in the first two images in Fig. 3. A simple algorithm (such as March Cube) can be employed to find all the points $\{\widetilde{x}_i\}$ on the boundary of the object in the binary image $s(x)$. These new points $\{\widetilde{x}_i\}$ are very close to the original unorganised point cloud $\{x_i\}$. The fast sweeping algorithm is applied again on the new point cloud $\{\widetilde{x}_i\}$ to obtain a new signed distance function ϕ_0 using the sign information obtained from the binary image $s(x)$ (i.e. inside is negative and outside is positive) as a good initialization for the level set evolution in section 3.



(a) (b) (c) (d)

Fig. 3 Calculate a signed distance function from the binary image. (a)-(b) are two binary images. (c)-(d) are corresponding signed distance maps of (a) and (b) respectively. (b) and (d) stand for one slice of the 3D data.

3 The proposed method

Minimisation of the following variational energy functional is proposed to reconstruct the surface from point clouds $\{x_i\}$.

$$E(\phi, c_1, c_2) = E_R(\phi) + \lambda E_I(\phi, c_1, c_2) + \beta E_B(\phi) \tag{4}$$

Each term of the energy functional targets a different aspect of the problem. The first term, E_R is a regularization term which keeps the surface of the object smooth while keeping the level set function ϕ as a signed distance function. The second data fitting term E_I incorporates the information derived from the dataset, where c_1 and

c_2 represent the mean values inside and outside of the zero level set of ϕ. The third balloon force term E_B includes the area/volume information inside the zero level set of ϕ. The details of the energy terms in (4) are given as follows.

$$E_R(\phi) = \int_\Omega d(x)|\nabla H(\phi)| + \frac{\mu}{2}\int_\Omega (|\nabla\phi| - 1)^2 \tag{5}$$

where $d(x)$ is the distance function calculated from the original point cloud $\{x_i\}$ in section 2.1. The first term in this functional is equivalent to GAC (geodesic active contour) model [12]. It is the weighted (by $d(x)$) length/area of the boundary by using the co-area formula for TV (total variation) [13]. The second term keeps level set function ϕ as a sign distance function, thus eliminating the need of reinitialization and producing desirable reconstruction. μ is positive penalty parameter controlling the degree of penalizing the deviation of ϕ from a signed distance function. Larger μ leads to more similarity between ϕ and signed distance function. (5) improves methods in [5] by using a variational level set without reinitialisation.

$$E_I(c_1, c_2, \phi) = \int_\Omega Q(c_1, c_2)H(\phi) \tag{6}$$

where $Q(c_1, c_2) = (c_1 - u(x))^2 - (c_2 - u(x))^2$, and $u(x)$ is the 2D/3D image computed from section 2.2. This term follows the work of the two-phase piecewise constant Chan-Vese model [14] and encourage each region of the reconstruction to have an approximately constant value. By incorporating the region-based information, this term has much larger convergence range. Thus it can help ease some local minimum problems and improve the accuracy of reconstruction by capturing small features of the object with homogeneous intensity values. The penalty parameter λ on this term in (4) should be positive.

$$E_B(\phi) = \int_\Omega d(x)H(-\phi) \tag{7}$$

This energy functional is the weighted area/volume of region $\Omega_\phi^- \triangleq \{x : \phi(x) < 0\}$. This term is introduced to speed up surface evolution as well as segment concave objects. The parameter β on this term in (4) can be positive or negative depending on whether inside or outside of the zero level set is defined as positive. In this paper, if the initial boundary is placed outside the object, the coefficient take positive values, so that the zero level set points can shrink during level set evolution. If the initial boundary is placed inside the object, the coefficient take negative values in order to expand the boundary.

Equation (4) is a multivariate minimization problem usually solved by an optimization procedure. First ϕ is fixed to optimise c_1 and c_2 as follows

$$c_1 = \frac{\int_\Omega uH(\phi)}{\int_\Omega H(\phi)};\ c_2 = \frac{\int_\Omega u(1-H(\phi))}{\int_\Omega (1-H(\phi))}$$

Then c_1 and c_2 are fixed using the following gradient descent flow starting with $\phi = \phi_0$ in section 2.3 to minimize (4)

$$\frac{\partial \phi}{\partial t} = \left(\nabla \cdot \left(d \frac{\nabla \phi}{|\nabla \phi|} \right) - \lambda Q(c_1, c_2) + \beta d \right) \delta(\phi) + \mu \left(\Delta \phi - \nabla \cdot \left(\frac{\nabla \phi}{|\nabla \phi|} \right) \right) \quad (8)$$

In practice, the Heaviside function $H(\phi)$ and Dirac function $\delta(\phi)$ in (4) and (8) are usually approximated by their regularized version with a small positive regularized number ε

$$H_\varepsilon(\phi) = \frac{1}{2} + \frac{1}{\pi} arctan\left(\frac{\phi}{\varepsilon} \right)$$

$$\delta_\varepsilon(\phi) = \frac{1}{\pi} \frac{\varepsilon}{\varepsilon^2 + \phi^2}$$

4 Experimental results

In this section, some 2D and 3D reconstruction results are presented. In Fig. 4, a 2D contour is given of the dataset shown in Fig. 1(a). The initial contour obtained in the preprocessing step is very close to the true surface, which can speed up convergent. As the evolution proceeds, the reconstruction by Zhao's method [4,5] loses the small features of the original data (i.e. the two convex parts). However, the term (6) in the proposed model is able to preserve these features.

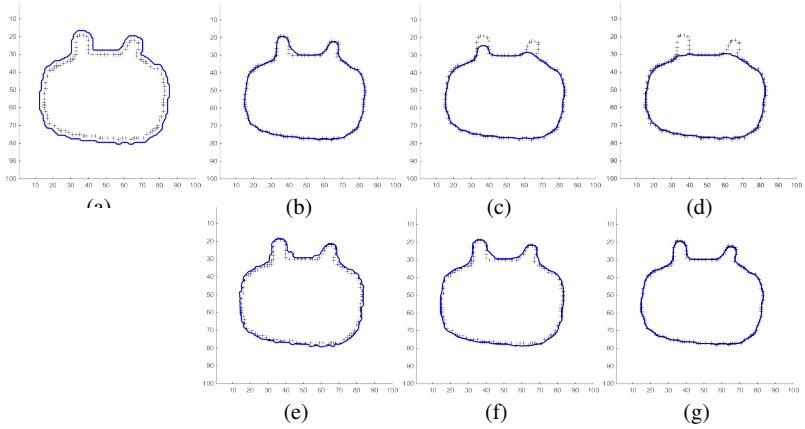


Fig. 4 Compare Zhao's method [4, 5] with the proposed model (4) with $\beta = 0$ in 2D. (a): same initialisation for methods to be compared; (b)-(d): intermediate and final results by Zhao's method; (e)-(g): intermediate and final results by the proposed method with $\beta = 0$.

Fig. 5 shows the reconstructed 3D surface of the Bunny point cloud shown in Fig. 1(b). Both Zhao's and Poisson failed to reconstruct the Bunny's ears and feet,

while the proposed method succeeded. The reconstruction by Poisson also loses some texture and looks smoother than that of the proposed method. This demonstrates the effectiveness of the volumetric data fitting term (6) in the proposed model.

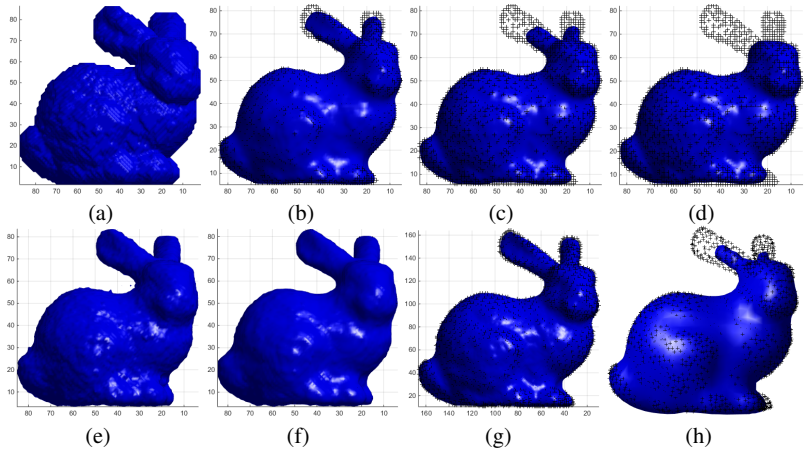


Fig. 5 Comparison with state of the art. (a): same initialisation for models to be compared; (b)-(d): intermediate and final results by Zhao's method [4, 5]; (e)-(g): intermediate and final results by the proposed model (4) with $\beta = 0$; (h): Reconstruction by Poisson [11].

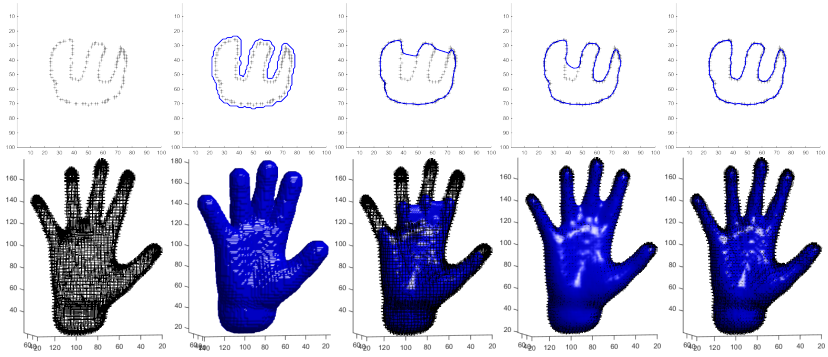
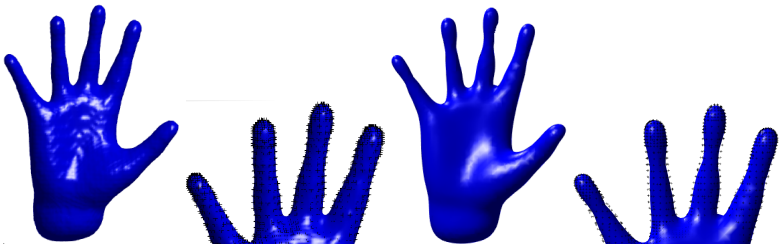


Fig. 6 Effectiveness of the balloon force term (7) in the proposed model using 2D and 3D datasets. 1st column: original data points; 2nd column: initialisation obtained in the first step; 3rd column: results by Zhao's method; 4th column: results by the proposed model (4) without using balloon force term (i.e. $\beta = 0$ in (4)); 5th column: results by the proposed model (4).

Fig. 6 shows the reconstruction results of a 2D concave object and a 3D human hand that contains small details (i.e. fingers) and concave regions (i.e. the spaces between fingers). Zhao's method gets stuck in the concave region and also loses the fingers. The proposed model within only (6) term can preserve the fingers and partially go down the concave region. The proposed model with both terms (6) and (7) succeeds in preserving all features as well as the concave regions in both 2D and

3D cases. This validates the capability of the balloon force term (7) in the proposed model.



(a) (b) (c) (d)

Fig. 7 Comparison with Poisson. (a) and (b) are reconstructions by the proposed model; (c) and (d) are reconstructions by the Possion method; (b) and (d) are zoomed-in version of (a) and (c), with the original data points added.

Fig. 7 shows that Poisson result is smoother than that of the proposed method, but some small details/texture, i.e. palm prints, are also smeared by the method. Fig. 7(b) and (d) showa that Poisson result also misses several original data points, and the reconstructed fingers become thinner than those by the proposed method. The proposed method thus performs better than the Poisson method.

5 Conclusion

In this paper, a novel variational level set method is proposed to reconstruct implicit surfaces from unorganised point clouds for industrial applications. Implementation details of the variational model are given and the functions of the three energy terms of the model are illustrated through numerical experiments. Major advantages of the proposed method over existing approaches are that it produces a smooth reconstruction whilst preserving the small details of the original object. It also handles concave regions of the object well, and without the need for reinitialisation and getting stuck at the local minimum commonly associated with the implicit reconstruction methods. Experimental results show that the proposed method outperforms the state of the art methods.

References

1. Edelsbrunner, H., Mücke, E.P.: Three-dimensional alpha shapes. ACM T. Graphic. 13(1), 43–72 (1994)

2. Amenta, N., Bern, M., Kamvysselis, M.: A new Voronoi-based surface reconstruction algorithm. Proceedings of the 25th annual conference on Computer graphics and interactive techniques, 415–421 (1998)
3. Boissonnat, J.D.: Geometric structures for three-dimensional shape representation. ACM T. Graphic. 3(4), 266–286 (1984)
4. Zhao, H.K., Osher, S., Merriman, B., Kan, M.: Implicit and nonparametric shape reconstruction from unorganized data using a variational level set method. Comput. Vis. Image Und. 80(3), 295–314, Elsevier (2000)
5. Zhao, H.K., Osher, S., Fedkiw, R.: Fast surface reconstruction using the level set method. Variational and Level Set Methods in Computer Vision, 2001. Proceedings. IEEE Workshop on, 194–201, IEEE (2001)
6. Ye, J., Bresson, X., Goldtein, T., Osher, S.: A fast variational method for surface reconstruction from sets of scattered points. CAM Report, 10(01), (2010)
7. Liang, J., Park, F., Zhao, H.K.: Robust and efficient implicit surface reconstruction for point clouds based on convexified image segmentation. J. Sci. Comput. 54(2-3), 577–602, Springer (2013)
8. Osher, S., Sethian, J.A.: Fronts propagating with curvature-dependent speed: algorithms based on Hamilton-Jacobi formulations. J. Computational Phys. 79(1), 12–49, Elsevier (1988)
9. Li, C.M., Xu, C.Y., Gui, C.F., Fox, M.D.: Level set evolution without re-initialization: a new variational formulation. Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on, 1, 430–436, IEEE (2005)
10. Zhao, H.K.: A fast sweeping method for eikonal equations. Math. Comput. 74(250), 603–627 (2005)
11. Kazhdan, M., Bolitho, M., Hoppe, H.: Poisson surface reconstruction. Proceedings of the fourth Eurographics symposium on Geometry processing (2006)
12. Caselles, V., Kimmel, R., Sapiro, G.: Geodesic active contours. Int. J. Comput. Vis. 22(1), 61–79, Springer (1997)
13. Rudin, L.I., Osher, S., Fatemi, E.: Nonlinear total variation based noise removal algorithms. Physica D: Nonlinear Phenom. 60(1), 259–268, Elsevier (1992)
14. Chan, T.F., Vese, L.A.: Active contours without edges. Image Process., IEEE Trans. 10(2), 266–277, IEEE (2001)

6 Appendix

In this section, in order to evolve level set function ϕ in (8), we present the discretisation for $\nabla \cdot \left(d \frac{\nabla \phi}{|\nabla \phi|} \right)$ and $\nabla \cdot \left(\frac{\nabla \phi}{|\nabla \phi|} \right)$ in three-dimensional case based on the finite difference scheme. In detail, let $\Omega \rightarrow \mathbb{R}^{MNL}$ denote the three-dimensional grid space with size MNL. The second order coupled (with distance function d) curvature term $\nabla \cdot \left(d \frac{\nabla \phi}{|\nabla \phi|} \right)$ at voxel (i, j, k) can be discretised as follows

$$
\begin{aligned}
\nabla\cdot\left(d\frac{\nabla\phi}{|\nabla\phi|_\varepsilon}\right)_{i,j,k} = {} & d_{i,j+\frac{1}{2},k}\frac{\partial_x^+\phi_{i,j,k}}{\sqrt{\left(\nabla_x^+\phi_{i,j,k}\right)^2+\left(\nabla_y^0\phi_{i,j+\frac{1}{2},k}\right)^2+\left(\nabla_z^0\phi_{i,j+\frac{1}{2},k}\right)^2+\varepsilon^2}} \\
& -d_{i,j-\frac{1}{2},k}\frac{\partial_x^-\phi_{i,j,k}}{\sqrt{\left(\nabla_x^-\phi_{i,j,k}\right)^2+\left(\nabla_y^0\phi_{i,j-\frac{1}{2},k}\right)^2+\left(\nabla_z^0\phi_{i,j-\frac{1}{2},k}\right)^2+\varepsilon^2}} \\
& +d_{i+\frac{1}{2},j,k}\frac{\partial_y^+\phi_{i,j,k}}{\sqrt{\left(\nabla_y^+\phi_{i,j,k}\right)^2+\left(\nabla_x^0\phi_{i+\frac{1}{2},j,k}\right)^2+\left(\nabla_z^0\phi_{i+\frac{1}{2},j,k}\right)^2+\varepsilon^2}} \\
& -d_{i-\frac{1}{2},j,k}\frac{\partial_y^-\phi_{i,j,k}}{\sqrt{\left(\nabla_y^-\phi_{i,j,k}\right)^2+\left(\nabla_x^0\phi_{i-\frac{1}{2},j,k}\right)^2+\left(\nabla_z^0\phi_{i-\frac{1}{2},j,k}\right)^2+\varepsilon^2}} \\
& +d_{i,j,k+\frac{1}{2}}\frac{\partial_z^+\phi_{i,j,k}}{\sqrt{\left(\nabla_z^+\phi_{i,j,k}\right)^2+\left(\nabla_x^0\phi_{i,j,k+\frac{1}{2}}\right)^2+\left(\nabla_y^0\phi_{i,j,k+\frac{1}{2}}\right)^2+\varepsilon^2}} \\
& -d_{i,j,k-\frac{1}{2}}\frac{\partial_z^-\phi_{i,j,k}}{\sqrt{\left(\nabla_z^-\phi_{i,j,k}\right)^2+\left(\nabla_x^0\phi_{i,j,k-\frac{1}{2}}\right)^2+\left(\nabla_y^0\phi_{i,j,k-\frac{1}{2}}\right)^2+\varepsilon^2}}
\end{aligned}
\tag{9}
$$

where ε is a small positive number to avoid division by zero. Note that the half-point difference scheme is used here for (9) in order to satisfy rotation-invariant characteristics. The distance function d on half-points of voxel (i,j,k) are given as

$$
\begin{aligned}
d_{i,j+\frac{1}{2},k} &= \tfrac{d_{i,j+1,k}+d_{i,j,k}}{2}, \quad d_{i,j-\frac{1}{2},k} = \tfrac{d_{i,j-1,k}+d_{i,j,k}}{2} \\
d_{i+\frac{1}{2},j,k} &= \tfrac{d_{i+1,j,k}+d_{i,j,k}}{2}, \quad d_{i-\frac{1}{2},j,k} = \tfrac{d_{i-1,j,k}+d_{i,j,k}}{2} \\
d_{i,j,k+\frac{1}{2}} &= \tfrac{d_{i,j,k+1}+d_{i,j,k}}{2}, \quad d_{i,j,k-\frac{1}{2}} = \tfrac{d_{i,j,k-1}+d_{i,j,k}}{2}
\end{aligned}
$$

The first order forward ∂_x^+ and backward ∂_x^- discrete derivatives along x, y and z directions can be defined as follows

$$
\begin{aligned}
\partial_x^+\phi_{i,j,k} &= \phi_{i,j+1,k}-\phi_{i,j,k}, \quad \partial_x^-\phi_{i,j,k} = \phi_{i,j,k}-\phi_{i,j-1,k} \\
\partial_y^+\phi_{i,j,k} &= \phi_{i+1,j,k}-\phi_{i,j,k}, \quad \partial_y^-\phi_{i,j,k} = \phi_{i,j,k}-\phi_{i-1,j,k} \\
\partial_z^+\phi_{i,j,k} &= \phi_{i,j,k+1}-\phi_{i,j,k}, \quad \partial_z^-\phi_{i,j,k} = \phi_{i,j,k}-\phi_{i,j,k-1}
\end{aligned}
$$

The central differences are applied to approximate the following first order discrete derivatives on the half-points of voxel (i,j,k) in (9).

$$
\begin{aligned}
\nabla^0_y \phi_{i,j+\frac{1}{2},k} &= \frac{\phi_{i+1,j+1,k}+\phi_{i+1,j,k}-\phi_{i-1,j+1,k}-\phi_{i-1,j,k}}{4} \\
\nabla^0_z \phi_{i,j+\frac{1}{2},k} &= \frac{\phi_{i,j+1,k+1}+\phi_{i,j,k+1}-\phi_{i,j+1,k-1}-\phi_{i,j,k-1}}{4} \\
\nabla^0_y \phi_{i,j-\frac{1}{2},k} &= \frac{\phi_{i+1,j,k}+\phi_{i+1,j-1,k}-\phi_{i-1,j,k}-\phi_{i-1,j-1,k}}{4} \\
\nabla^0_z \phi_{i,j-\frac{1}{2},k} &= \frac{\phi_{i,j,k+1}+\phi_{i,j-1,k+1}-\phi_{i,j,k-1}-\phi_{i,j-1,k-1}}{4} \\
\nabla^0_x \phi_{i+\frac{1}{2},j,k} &= \frac{\phi_{i+1,j+1,k}+\phi_{i,j+1,k}-\phi_{i+1,j-1,k}-\phi_{i,j-1,k}}{4} \\
\nabla^0_z \phi_{i+\frac{1}{2},j,k} &= \frac{\phi_{i+1,j,k+1}+\phi_{i,j,k+1}-\phi_{i+1,j,k-1}-\phi_{i,j,k-1}}{4} \\
\nabla^0_x \phi_{i-\frac{1}{2},j,k} &= \frac{\phi_{i,j+1,k}+\phi_{i-1,j+1,k}-\phi_{i,j-1,k}-\phi_{i-1,j-1,k}}{4} \\
\nabla^0_z \phi_{i-\frac{1}{2},j,k} &= \frac{\phi_{i,j,k+1}+\phi_{i-1,j,k+1}-\phi_{i,j,k-1}-\phi_{i-1,j,k-1}}{4} \\
\nabla^0_x \phi_{i,j,k+\frac{1}{2}} &= \frac{\phi_{i,j+1,k+1}+\phi_{i,j+1,k}-\phi_{i,j-1,k+1}-\phi_{i,j-1,k}}{4} \\
\nabla^0_y \phi_{i,j,k+\frac{1}{2}} &= \frac{\phi_{i+1,j,k+1}+\phi_{i+1,j,k}-\phi_{i-1,j,k+1}-\phi_{i-1,j,k}}{4} \\
\nabla^0_x \phi_{i,j,k-\frac{1}{2}} &= \frac{\phi_{i,j+1,k}+\phi_{i,j+1,k-1}-\phi_{i,j-1,k}-\phi_{i,j-1,k-1}}{4} \\
\nabla^0_y \phi_{i,j,k-\frac{1}{2}} &= \frac{\phi_{i+1,j,k}+\phi_{i+1,j,k-1}-\phi_{i-1,j,k}-\phi_{i-1,j,k-1}}{4}
\end{aligned}
$$

In order to discretise the curvature term $\nabla \cdot \left(\frac{\nabla \phi}{|\nabla \phi|} \right)$, we set $d_{i,j+\frac{1}{2},k} = d_{i,j-\frac{1}{2},k} = d_{i+\frac{1}{2},j,k} = d_{i-\frac{1}{2},j,k} = d_{i,j,k+\frac{1}{2}} = d_{i,j,k-\frac{1}{2}} = 1$ in (9).

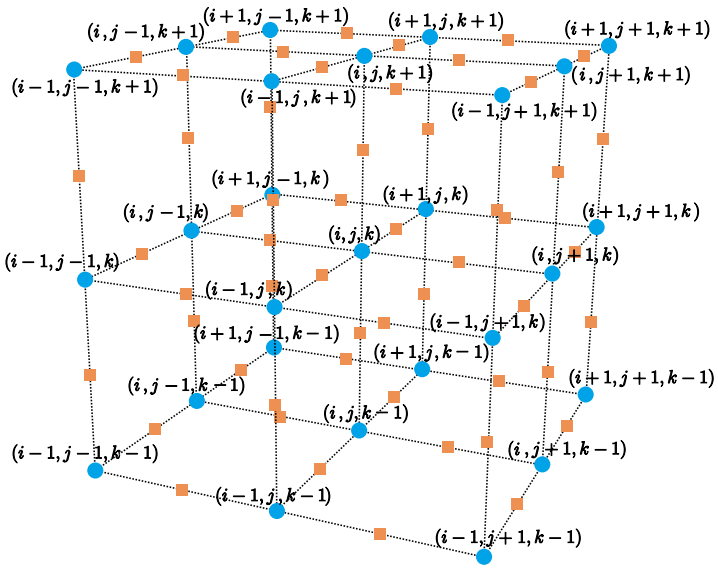


Fig. 8 3D grid space. The blue points represent different voxels, while the yellow points are the half points defined between each two voxels. The figure illustrates how to calculate the discrete differential operators used in equation (9).