Multi-Reconstruction from Points Cloud by Using a Modified Vector-Valued Allen–Cahn Equation

Jin Wang 1 and Zhengyuan Shi 2,*

Abstract: The Poisson surface reconstruction algorithm has become a very popular tool of reconstruction from point clouds. If we reconstruct each region separately in the process of multi-reconstruction, then the reconstructed objects may overlap with each other. In order to reconstruct multicomponent surfaces without self-intersections, we propose an efficient multi-reconstruction algorithm based on a modified vector-valued Allen–Cahn equation. The proposed algorithm produces smooth surfaces and closely preserves the original data without self-intersect. Based on operator splitting techniques, the numerical scheme is divided into one linear equation and two nonlinear equations. The linear equation is discretized using an implicit method, and the resulting discrete system of equation is solved by a fast Fourier transform. The two nonlinear equations are solved analytically due to the availability of a closed-form solution. The numerical scheme has merit in that it can be straightforwardly applied to a graphics processing unit, allowing for accelerated implementation that performs much faster than central processing unit alternatives. Various experimental, numerical results demonstrate the effectiveness and robustness of the proposed method.

Keywords: multimaterial; surface reconstruction; Allen–Cahn equation; mean curvature flow

1. Introduction

Multimaterial reconstruction from multi-labeled point clouds is of significant importance in many disciplines, such as mechanical engineering [1,2]. Such point clouds are sampled by a multi-camera vision system and provide the local connectivity of objects. For instance, in the reversed engineering problem, multicomponent surfaces/volumes of the object should be reconstructed without self-intersections so that a multimaterial 3D printer can process these models. The challenges in multi-reconstruction mainly lie in the following two aspects: how to handle the contacting regions and how to maintain the original topology. Up until now, many reconstruction methods have been developed to address modeling from point clouds. In addition, modeling multi-labeled domains have been extensively studied in reconstructing surfaces or volumes from sets of planar cross-sections. The authors proposed a phase field framework for 3D surface reconstruction from parallel slices based on the modified Cahn–Hilliard equation [3–5]. All of the above-mentioned reconstruction methods can process one region or separated regions but cannot process contacting regions. If we reconstruct surfaces from each single-labeled point, the resulted objects may overlap with each other (see Figure 1). Here, the surface/volumes-overlapping problem refers to the phenomenon that the volume voxel is defined for more than one value for the label. Hence, we need to develop an efficient algorithm to reconstruct multicomponent surfaces from point clouds without overlapping regions.
In this paper, we propose an efficient multi-reconstruction method from point clouds. The proposed method is based on the Allen–Cahn (AC) equation [7], which has the motion of the mean curvature flow [8–11]. Under the mean curvature flow, the reconstructed surface is smooth. The Allen–Cahn equation has been applied to a wide range of problems such as crystal growth [12–14], image analysis [15–18], triply-periodic minimal surface [19,20], and volume reconstruction [21–23], and topology optimization [24]. In this paper, we use the Allen–Cahn equation in different contexts and applications. To our best knowledge, the presented approach is the first algorithm using the modified vector-valued Allen–Cahn equation for multi-reconstruction from point clouds. Our method can guarantee the production of well-reconstructed multicomponent surfaces without self-intersections. Our main contributions in this paper are as follows:

- The first algorithm using a modified vector-valued Allen–Cahn equation for multi-reconstruction from point clouds, which can reconstruct multicomponent surfaces without self-intersections;
- Based on operator splitting techniques, the proposed numerical scheme is simple to implement;
- The algorithm can be straightforwardly applied to a graphics processing unit (GPU), allowing for accelerated implementation that performs many times faster than other central processing units (CPU).

The remainder of the paper is organized as follows. Section 2 briefly reviews some related works. In Section 3, we introduce the preliminary. In Section 4, the governing equations for the multi-reconstruction from a point cloud are presented. Section 5 presents the operator splitting scheme, and Section 6 states the numerical results of several computational examples. Conclusions are drawn in Section 7.

2. Related Work

2.1. Implicit Surface Reconstruction

Implicit methods usually deal with data imperfections [25,26]. Most of the implicit implemented function are based on the signed distance function, radial basis functions, piecewise polynomial functions, or indicator functions. Hoppe et al. [27] defined the signed distance field by using the tangent plane projection. In [28], Carry et al. employed the polyharmonic radial basis function to reconstruct the manifold surfaces. This technique is well-suited to deal with non-uniformly sampled point clouds. However, it fails to reconstruct a proper surface from the point clouds with noises. Ohtake et al. [29] used the piecewise quadratic function and partition-of-unity weights to capture the local surface shape. This approach has the benefit of sharp feature reconstruction. Manson et al. [30] presented a streaming method for surface reconstruction from large point sets by using...
the wavelets method. The Poisson method [6] is a popular implicit method because of its resilience to data noises. However, it tends to smooth geometric details. To overcome this defect, Kazhdan and Hoppe [31] extended the Poisson method by explicitly incorporating the points as interpolation constraints by formulating a screened Poisson surface reconstruction method. Bretin et al. proposed a variational approach for the reconstruction of a volume from slices by using a minimizer of a geometric regularity criterion, Willmore energy, with inclusion–exclusion constraints associated with the cross-sections. In [32], the authors considered the problem of 3D shape reconstruction from multimodal data, given uncertain calibration parameters. To analytically and compactly represent the object, they exploited a parametric level set method, which utilized ellipsoidal radial basis functions. Kim and Lee proposed a modified Cahn–Hilliard equation for three-dimensional volume reconstruction from 2D slices [4]. In order to accurately satisfy the constraints while obtaining a smooth result, they applied a presmoothing procedure based on anisotropic diffusion to the slices.

2.2. Multi-Reconstruction

Volume fractions and implicit functions are typical approaches for defining the multilabeled domains. Hirt and Nichols [33] developed an efficient interface reconstruction method by treating complicated free boundary configurations as the material interfaces. To obtain the finer interface details, the authors utilized information about the position of the material centroid for the interface reconstruction [34–36]. Lemoine et al. [37] proposed a fast analytic reconstruction formula replacing the original minimization stage in the framework of the moment-of-fluid method. This algorithm can provide accurate results with a lower computational cost. Kikinzon et al. [38] extended the moment-of-fluid method to simultaneously establish the geometry and topology. Since the modified moment-of-fluid method does not require data from its neighbors, the modified moment-of-fluid method is well suitable for massively parallel computing. Yuan et al. [39] introduced a class of object-space multiphase implicit functions that are capable of accurately modeling objects with multiple internal regions. Zhang and Qian [40] resolved the topology ambiguity method by analyzing a hybrid Octree method with both cubic and tetrahedral leaf cells. This method can apply to both homogeneous and multimaterial domains. Da et al. [41] presented the first non-manifold triangle mesh tracking method to simultaneously maintain intersection-free meshes. Their methods support multimaterial remeshing and topological operations. In the context of multi-reconstruction from cross-sections, Liu et al. [42] proposed an extension of the projection-based approach that can deal with curve networks of arbitrary topology on cross-section planes with arbitrary orientations. Bermano et al. [43] used multiple implicit functions to extract multi-labeled material interfaces from sampled planar cross-sections of arbitrary orientation. Huang et al. [44] developed the first reconstruction algorithm for multi-labeled material interfaces that allows for explicit topology control. Li et al. [45] constructed 3D models of multi-labeled objects from sets of cross-sections by employing a multicomponent Cahn–Hilliard system. This algorithm is simple to implement and produces smooth surfaces. However, all the mentioned multi-labeled material reconstruction methods cannot directly reconstruct surfaces from multi-labeled point clouds. To our best knowledge, the presented approach is the first algorithm using the modified vector-valued Allen–Cahn equation for multi-reconstruction from point clouds. Our method can guarantee the production of well-reconstructed multicomponent surfaces without self-intersections.

3. Preliminary: Poisson Surface Reconstruction

The Poisson reconstruction method creates smooth surfaces [6] by using an indicator \( \psi \) (defined as 1 at the voxel inside the solid \( M \) and 0 at the voxel outside the solid \( M \)).
Given the vector field \( v(x) \in \mathbb{R}^3 \) defined by the data samples \( S \), the goal of the Poisson reconstruction method is to find the scalar function \( \psi(x) \) by minimizing

\[
\mathcal{E}_1(\psi) = \int_\Omega \| \nabla \psi(x) - v(x) \|^2_2 dx.
\]  

(1)

Here, \( x \in \Omega \) and \( \Omega \subset \mathbb{R}^3 \). \( \nabla \) is the gradient operator and \( \| \cdot \|_2 \) is the \( L_2 \) inner product. By applying the divergence operator to form the Poisson equation, the governing equation can be obtained by

\[
\Delta \psi = \nabla \cdot v,
\]  

(2)

where \( \Delta \) and \( \nabla \cdot \) are the Laplacian and divergence operators, respectively. The vector field \( v \) is defined by convolving the normal field with a Dirac delta function:

\[
v(x) = \int_{\partial M} \delta(x - p)n(p)dp \approx \sum_{p \in S} \delta(x - p)n(p),
\]  

(3)

where \( n(p) \) is the inward surface normal at \( p \in S \), \( \delta \) is the Dirac delta function, and \( \partial M \) is the boundary of the solid \( M \). After solving Equation (2), we can obtain the solution \( \psi \), which is also called the indicator function [46–50]. Then, we define the reconstructed surface \( S \) as the half isosurface of the scalar function \( \psi \).

4. Methodology

Using the Poisson surface reconstruction [6], the indicator function of \( i \)th material can be determined by a discrete function \( \psi_i(x) \) in the domain \( \Omega \). Here, \( \psi_i(x) = 1 \) if the voxel \( x \) is determined as a volume voxel, otherwise \( \psi_i(x) = 0 \). As mentioned in the above section, \( \psi_i \) contains lots of good voxels and few self-intersection regions. We want to obtain a new smooth discrete function \( \phi_i(x) \), which approaches the given \( \psi_i(x) \) without self-intersections. To realize this goal, we propose the following total free energy

\[
\mathcal{E}(\phi) = \sum_{i=1}^{N} \int_{\Omega} \left( \frac{F(\psi_i)}{\varepsilon^2} + \frac{1}{2} |\nabla \phi_i|^2 + \frac{\lambda}{2} |\phi_i - \phi_l|^2 \right) dx
\]  

(4)

Here, the double-well potential function \( F(\psi_i) = \phi_i^2(\psi_i - 1)^2/4 \) can enforce \( \phi \) to be approximately 0 or 1. \( \varepsilon \) is the positive constant and relates to the phase transition width. \( \lambda \) is the positive constant. Minimizing the first two terms of the functional \( \mathcal{E}(\phi) \) is equivalent to finding the steady-state solution of the gradient flow equation:

\[
\frac{\partial \phi_i(x,t)}{\partial t} = -\frac{F(\phi_i(x,t))}{\varepsilon^2} + \Delta \phi_i(x,t), \quad 0 < t \leq T.
\]  

(5)

Here, \( t \) is the time variable. Equation (5) is the well-known Allen–Cahn equation [7], which has the motion of mean curvature [8]. This ensures that the obtained surface is smooth. Minimizing the third term of the functional \( \mathcal{E}(\phi) \) enforces that the new smooth discrete function \( \phi_i(x) \) is almost the same as those in the originally known volume \( \psi_i \).

The sum of the different material components should be equal to 1, i.e., \( \sum_{i=1}^{N} \phi_i = 1 \), which implies that any voxel is defined as one label (See Figure 2). Therefore, if we minimize the functional \( \mathcal{E} \) and keep \( \sum_{i=1}^{N} \phi_i = 1 \), the obtained surface will be similar to \( \psi_i \) and will be smooth without self-intersections. To minimize the proposed functional \( \mathcal{E} \), we consider the gradient descent method in the \( L_2 \) space:

\[
\frac{\partial \phi_i(x,t)}{\partial t} = -\frac{F(\phi_i(x,t))}{\varepsilon^2} + \Delta \phi_i(x,t) + \beta(x) + \lambda(\psi_i(x) - \phi_i(x,t)),
\]  

(6)

\[
\phi_i(x,t) \text{ is } \Omega - \text{periodic},
\]  

(7)

\[
\phi_i(x,0) = \psi(x).
\]  

(8)
where
\[
\beta = \frac{1}{N} \sum_{i=1}^{N} \frac{F'(\phi_i)}{e} + \frac{1}{N} \lambda \left( 1 - \sum_{i=1}^{N} \phi_i \right). \tag{9}
\]

This Lagrangian multiplier guarantees that the constraint \( \sum_{i=1}^{N} \phi_i = 1 \) is satisfied:
\[
\frac{\partial}{\partial t} \sum_{i=1}^{N} \phi_i = \sum_{i=1}^{N} \frac{\partial \phi_i}{\partial t} = - \sum_{i=1}^{N} \frac{F'(\phi_i)}{e^2} + \sum_{i=1}^{N} \beta + \Delta \sum_{i=1}^{N} \phi_i + \lambda \left( \sum_{i=1}^{N} \phi_i - \sum_{i=1}^{N} \phi_i \right)
\]
\[
= - \sum_{i=1}^{N} \frac{F'(\phi_i)}{e^2} + N\beta + \lambda \left( \sum_{i=1}^{N} \phi_i - 1 \right) = 0. \tag{10}
\]

The total energy can be decreased over time as follows:
\[
\frac{dE}{dt} = \int_{\Omega} \sum_{i=1}^{N} \left( \frac{F'(\phi_i)}{e^2} - \Delta \phi_i + \lambda (\psi_i - \phi_i) \right) \frac{\partial \phi_i}{\partial t} \, dx
\]
\[
= \int_{\Omega} \sum_{i=1}^{N} \left( - \frac{\partial \phi_i}{\partial t} + \beta \right) \frac{\partial \phi_i}{\partial t} \, dx
\]
\[
= - \int_{\Omega} \sum_{i=1}^{N} \left( \frac{\partial \phi_i}{\partial t} \right)^2 \, dx + \beta \int_{\Omega} \frac{\partial}{\partial t} \sum_{i=1}^{N} \phi_i \, dx
\]
\[
= - \int_{\Omega} \sum_{i=1}^{N} \left( \frac{\partial \phi_i}{\partial t} \right)^2 \, dx \leq 0.
\]
Here, we have used the periodic boundary conditions for \( \phi_i \). It should be noted that the solution of Equations (6)–(8), which are derived from a constrained gradient flow of the free energy functional, is unique.

Figure 2. Reconstructed surface in the cut view. The red and blue regions represent the different materials. The yellow region represents the overlapping region.

5. Numerical Method

We utilize an efficient operator splitting scheme for Equation (6) on the discrete domain \( \Omega = (0, L_x) \times (0, L_y) \times (0, L_z) \). Let \( h = L_x/N_x = L_y/N_y = L_z/N_z \) be the uniform grid size. Let \( x_p = ph, y_q = qh, z_r = rh, 1 \leq p \leq N_x, 1 \leq q \leq N_y, \) and \( 1 \leq r \leq N_z \). Let \( x_{pqr} = (x_p, y_q, z_r) \) and \( \phi^k_{t_{pqr}} \) be approximations of \( \phi_t(x_{pqr}, k\Delta t) \), where \( \Delta t \) is the time step. We split the original problem (6) into a sequence of simpler problems as
\[
\left\{ \begin{array}{l}
\frac{\partial}{\partial t} \phi^1_t(x, t) = \lambda(x) (\psi(x) - \phi^1_t(x, t)), \quad t \in ((k-1)\Delta t, k\Delta t], \\
\phi^1_t(x, (k-1)\Delta t) = \phi_1(x, (k-1)\Delta t),
\end{array} \right. \tag{11}
\]
\[
\left\{ \begin{array}{l}
\frac{\partial}{\partial t} \phi^2_t(x, t) = \Delta \phi^2_t(x, t) + \beta(x), \quad t \in ((k-1)\Delta t, k\Delta t], \\
\phi^2_t(x, (k-1)\Delta t) = \phi^1_t(x, (k\Delta t),
\end{array} \right. \tag{12}
\]
and
\[
\begin{aligned}
\frac{\partial}{\partial t} \phi_t^3(x, t) &= -\frac{f'(\phi_t^3)}{e^{\xi t}}, \\
\phi_t^3(x, (k-1)\Delta t) &= \phi_t^3(x, k\Delta t).
\end{aligned}
\]
(13)

Here, \(\phi_t^1, \phi_t^2,\) and \(\phi_t^3\) denote the solutions of the subproblem (11)–(13), respectively. For a fixed \(x\), we can see that Equation (11) is a separable ordinary differential equation, i.e.,
\[
\lambda dt + d\phi/(\phi - \psi) = 0.
\]
We can obtain the following solution:
\[
\phi_{t,pqr}^{1,k+1} = e^{-\lambda \Delta t} \phi_{t,pqr}^k + \left(1 - e^{-\lambda \Delta t}\right) \psi_{t,pqr}.
\]
(14)

Next, we employ an implicit method to Equation (12):
\[
\frac{\phi_{t,pqr}^{2,k+1} - \phi_{t,pqr}^{1,k+1}}{\Delta t} = \Delta \phi_{t,pqr}^{2,k+1} + \rho_{t,pqr}^{1,k+1}.
\]
(15)

This discrete scheme can be solved by a fast Fourier transform as follows. The discrete Fourier transform \(\hat{\phi}_{t,lmn}\) for \(l = 1, \ldots, N_x, m = 1, \ldots, N_y,\) and \(n = 1, \ldots, N_z\) is defined as
\[
\hat{\phi}_{t,lmn} = \mathcal{F}(\phi_{t,pqr}) = \sum_{p=1}^{N_x} \sum_{q=1}^{N_y} \sum_{r=1}^{N_z} \phi_{t,pqr} e^{-i(\xi_l x_p + \eta_m y_q + \gamma_n z_r)},
\]
(16)

where \(\xi_l = 2\pi(l-1)/L_x, \eta_m = 2\pi(m-1)/L_y,\) and \(\gamma_n = 2\pi(n-1)/L_z,\) respectively. The inverse discrete Fourier transform is
\[
\phi_{t,pqr} = \mathcal{F}^{-1}(\hat{\phi}_{t,lmn}) = \sum_{l=1}^{N_x} \sum_{m=1}^{N_y} \sum_{n=1}^{N_z} \hat{\phi}_{t,lmn} e^{i(\xi_l x_p + \eta_m y_q + \gamma_n z_r)}.
\]
(17)

We apply the Fourier transform to both sides of Equation (15):
\[
\mathcal{F}(\phi_{t,pqr}^{2,k+1}) - \mathcal{F}(\phi_{t,pqr}^{1,k+1}) = -\left(\xi_l^2 + \eta_m^2 + \gamma_n^2\right) \mathcal{F}(\phi_{t,pqr}^{2,k+1}) + \mathcal{F}(\rho_{t,pqr}^{1,k+1}).
\]
(18)

Then, we can get
\[
\phi_{t,pqr}^{2,k+1} = \mathcal{F}^{-1}(\mathcal{F}(\phi_{t,pqr}^{2,k+1})) = \mathcal{F}^{-1}\left(\frac{\mathcal{F}(\rho_{t,pqr}^{1,k+1})}{\Delta t} + \Delta t \frac{\mathcal{F}(\phi_{t,pqr}^{2,k+1})}{1 + \Delta t (e^{-i\eta_m} + e^{-i\gamma_n})}\right).
\]
(19)

For a fixed \(x\), Equation (13) is also a separable ordinary differential equation, i.e.,
\[
dt/\epsilon^2 - 2d\phi/\phi + 4d\phi/(\phi - 0.5) + 2d\phi/(1 - \phi) = 0.
\]
With the initial condition \(\phi_{t,pqr}^{2,k+1}\), we can obtain the following solution after \(\Delta t\):
\[
\phi_{t,pqr}^{3,k+1} = \frac{1}{2} + \frac{\phi_{t,pqr}^{2,k+1} - 0.5}{\sqrt{e^{\xi t} + (2\phi_{t,pqr}^{2,k+1} - 1)^2 (1 - e^{\xi t})}}.
\]
(20)
Then, we set $\phi_{t}^{k+1} = \phi_{i}^{3k+1}$ and complete the one time step processing. Finally, the proposed scheme can be written as follows:

$$\phi_{t,pyr}^{1k+1} = e^{-\lambda \Delta t} \phi_{t,pyr}^{k} + \left(1 - e^{-\lambda \Delta t}\right) \phi_{t,pyr}$$ \hspace{1cm} (21)

$$\phi_{t,pyr}^{2k+1} = \frac{1}{\mathcal{F}^{-1}} \left( \mathcal{F}(\phi_{t,pyr}^{1k+1}) + \Delta t \mathcal{F}(\phi_{t,pyr}^{1k+1}) \right)$$ \hspace{1cm} (22)

$$\phi_{t,pyr}^{k+1} = \left[ \frac{1}{2} + \frac{\phi_{t,pyr}^{2k+1} - 0.5}{\sqrt{e^{\frac{-\Delta t}{2\Delta^2}} + (2\phi_{t,pyr}^{2k+1} - 1)^2 (1 - e^{-\frac{\Delta t}{2\Delta^2}})}} \right]$$ \hspace{1cm} (23)

Therefore, the numerical scheme (Equations (21)–(23)) is simple to implement because Equation (6) is divided into one linear equation and two nonlinear equations. The linear equation is discretized using an implicit method, and the resulting discrete system of equations is solved by a fast Fourier transform. The two nonlinear equations are solved analytically due to the availability of a closed-form solution. Our method is fast since the computational complexities of two nonlinear equations are $O(N_x N_y N_z)$ and the computational complexity of the linear equation is $O(N_x N_y N_z \log (N_x N_y N_z))$. Therefore, our method is simple and fast. Our method also has merit in that it can be straightforwardly applied to GPU-accelerated implementation by using the MATLAB Parallel Computing Toolbox.

The operator splitting-based hybrid numerical scheme (Equations (21)–(23)) has a good numerical stability. For Equation (21), we get $0 \leq \phi_{t}^{1k+1} \leq e^{-\lambda \Delta t} + (1 - e^{-\lambda \Delta t}) = 1$, because $\phi_{t}^{k} \in [0, 1]$ and $\psi_{i} \in [0, 1]$. Since Equation (22) is a heat-type equation, its implicit numerical scheme can allow the use of a large time step. For Equation (23), we get

$$\phi^{k+1} = \left[ \frac{1}{2} + \frac{\phi^{2k+1} - 0.5}{\sqrt{e^{\frac{-\Delta t}{2\Delta^2}} + (2\phi^{2k+1} - 1)^2 (1 - e^{-\frac{\Delta t}{2\Delta^2}})}} \right] = \begin{cases} 1 & \text{if } \phi^{2k+1} = 1, \\ \leq 1 & \text{if } \phi^{2k+1} \in (0.5, 1), \\ 0.5 & \text{if } \phi^{2k+1} = 0.5, \\ \geq 0 & \text{if } \phi^{2k+1} \in (0, 0.5), \\ 0 & \text{if } \phi^{2k+1} = 0. \\ \end{cases}$$ \hspace{1cm} (24)

Thus, if $\phi_{t}^{2k+1} \in [0, 1]$, then $\phi_{t}^{k+1} \in [0, 1]$. Therefore, the numerical solution $\phi_{i}$ is bounded, which implies that our proposed scheme allows using a larger time step. The hybrid numerical scheme (Equations (21)–(23)) has first-order accuracy in time and second-order accuracy in space because Equation (22) is employed by using an implicit method and Equations (21) and (23) are two ordinary equations. It should be noted that in the analysis, we can not prove the unconditional stability of our method. On the other hand, for the computational simulation, the results with a large time step are generally less accurate than those obtained by using a small time step, because using a larger time step would cause a large error of the numerical solutions. Therefore, a small time step will be used for highly accurate numerical solutions. Therefore, to maintain the numerical accuracy and reduce computational costs, we propose to use $\Delta t = 0.25$ in this paper.

6. Experimental Results

In this section, various numerical results will be performed to demonstrate the accuracy and efficiency of our algorithm. We consider the numerical result as a steady-state solution if the relative error for every component is less than a tolerance $\text{tol}$, i.e., $\left\| \phi_{i}^{k+1} - \phi_{i}^{k} \right\|_{2}/\left\| \phi_{i}^{k} \right\|_{2}$ for $i = 1, \ldots, N$. Unless otherwise specified, we use the following parameters: $h = 1$, $\Delta t = 0.25$, $\text{tol} = 1e-5$, $\lambda = 0.1$, and $\epsilon = 0.75$. The reconstructed surface $S$ is defined as the half isosurface of the phase field $\phi_{i}$. Note that the original input points cloud is taken from [51].
6.1. Basic Mechanism of the Algorithm

We start with an example that illustrates the basic mechanism of our algorithm. As is shown in Figure 3a, let us consider two-labeled oriented point samples of two intersecting spheres. The radii of the red sphere and blue sphere are 100 and 75, respectively. Figure 3b,c shows the reconstructed surface obtained by the Poisson reconstruction and our method, respectively. The overlapping region exists in the result of Poisson reconstruction, as shown in the left figure of Figure 4. On the other hand, our proposed method can overcome this difficulty, as shown in the right figure of Figure 4. Figure 4 illustrates the evolution of our modified multicomponent AC equation in the cross-view. From left to right, the illustration represents \( t = 0, 1, \) and 5.5, respectively. It is obvious that our method can reconstruct multicomponent surfaces without overlapping regions.

![Figure 3.](image)

**Figure 3.** (a) The point cloud that is downsampled for display. (b,c) The surfaces that are obtained by Poisson surface reconstruction and our method, respectively.

![Figure 4.](image)

**Figure 4.** The evolution of our method in the cut view. (a–c) The contours from \( t = 0, 1, 5.5 \) in the cross-view, respectively.

6.2. Performance on the Complex Reconstruction

We use the proposed method to reconstruct the surface from the complex multi-labeled point clouds. We compared our results with the results attained by Poisson surface reconstruction in Figure 5a,c, respectively. The overlapping regions are obvious. On the other hand, our method can avoid self-intersections and provide a high-quality repairing result, as shown in Figure 5b,d. The mesh grid size is \( 298 \times 222 \times 156 \). The computation of our method takes about 2.5 s to perform on the GPU.

6.3. Multicomponent Surface Reconstruction

Our method can process the multicomponent surface reconstruction from the multi-labeled point clouds. Figure 6a–c shows the results with two-component, three-component, and four-component reconstructions, respectively. Here, a mesh grid \( 298 \times 156 \times 212 \) is used. Our proposed method achieves fast convergence with few iterations. These simulations demonstrate that our proposed method can perform well for multicomponent surface reconstruction.
Figure 5. (a,c) The results of Poisson reconstruction and our proposed method in the whole view, respectively. (b,d) The results in the cross-view of (a,c), respectively.

Figure 6. Multicomponent reconstruction of our proposed method. (a) Two-component reconstruction. (b) Three-component reconstruction. (c) Four-component reconstruction.

Figure 7 shows the four-component surface reconstruction of the tablet. From left to right, these are the input real point clouds and final reconstructions with different views. As can be observed, our proposed method can handle the complex morphology, and the reconstructed surface is smooth.
6.4. Multi-Reconstruction with Different Density of Point Data

Figure 8 shows our multi-reconstruction results for the happy buddha surface with different densities of input point data. Figure 8a is the initial input data. Note that for the purposes of better visualization, we displayed the points more sparsely than the original density. Observing these results in Figure 8, we can see that with a low density of input point data, our proposed method can still reconstruct the surface with finer details and features. The agreement between the results obtained by different sampling densities suggests that our method can successfully reconstruct the surface with the low sampling density.

6.5. Parameter Sensitivity Analysis

In this section, we will analyze the parameter sensitivity of our proposed algorithm. The fidelity term in Equation (6) keeps the solution close to the original result of the Poisson reconstruction. $\lambda$ balances the motion by the mean curvature flow and the fitting term.
If $\lambda = 0$, (6) becomes the classical Allen–Cahn equation. Consequently, the interface of phase field $\phi_i$ will shrink with time. That is, if $\lambda$ is too small, the detailed information of the original object will be lost, as shown in Figure 9a. On the other hand, if $\lambda$ is too large, the fidelity term is dominant so that $\phi_i \approx \psi_i$ for any time. Thus, the reconstructed surfaces will overlap with each other, as shown in Figure 9c. Therefore, we should choose an appropriate $\lambda$. In this paper, we propose to use $\lambda = 0.1$. The role of $\epsilon$ is related to the material transition width. We take the same initial condition except for different values of $\epsilon$. Observing the results of Figure 10, we can observe that if $\epsilon$ is small, the surface of objects is sharp and the volumes have many holes. If $\epsilon$ is too large, the surface becomes over-smooth and the volumes overlap with each other. In this paper, we propose using $\epsilon = 0.75$.

![Figure 9. Illustration for the parameter $\lambda$ sensitivity analysis. (a) $\lambda = 0.01$. (b) $\lambda = 0.1$. (c) $\lambda = 10$.](image)

![Figure 10. Illustration for the parameter $\epsilon$ sensitivity analysis. (a) $\epsilon = 0.25$. (b) $\epsilon = 0.75$. (c) $\epsilon = 2.5$.](image)

### 6.6. Performance of Our Method

Table 1 presents the information on the grid size, iteration numbers, CPU and GPU times. CPU(pro) and GPU(pro) are the times taken to process the multi-reconstruction by
using the computer’s central processing unit and graphics processing unit, respectively. The CPU and GPU times in seconds of our calculations, which are performed in MATLAB, are measured on a desktop with an NVIDIA GeForce GTX 1050Ti and a quad-core Intel Core i5. We can observe that our proposed method obtains fast convergence after a few iterations because the computational complexities of the two nonlinear equations are $O(N_xN_yN_z)$ and the computational complexity of the linear equation is $O(N_xN_yN_z \log (N_xN_yN_z))$.

Furthermore, we can employ a GPU-accelerated method that performs many times faster than CPU-only alternatives. It should be noted that the gain is related to the speed of the computation and our method may perform well in a high-performance computing cluster system.

### Table 1. List of grid size, iterations, CPU times (second), GPU times (second), and errors. CPU(pro) and GPU(pro) are the times taken to process the multi-reconstruction by using the computer’s central processing unit and graphics processing unit, respectively.

<table>
<thead>
<tr>
<th>Case</th>
<th>Grid Size</th>
<th>Iteration</th>
<th>CPU (pro)</th>
<th>GPU (pro)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Figure 4</td>
<td>$(256 \times 256 \times 256)$</td>
<td>24</td>
<td>36.2</td>
<td>9.8</td>
</tr>
<tr>
<td>Figure 5</td>
<td>$(298 \times 222 \times 156)$</td>
<td>12</td>
<td>9.2</td>
<td>2.5</td>
</tr>
<tr>
<td>Figure 6a</td>
<td>$(298 \times 156 \times 212)$</td>
<td>14</td>
<td>9.6</td>
<td>3.0</td>
</tr>
<tr>
<td>Figure 6b</td>
<td>$(298 \times 156 \times 212)$</td>
<td>18</td>
<td>20.6</td>
<td>5.6</td>
</tr>
<tr>
<td>Figure 6c</td>
<td>$(298 \times 156 \times 212)$</td>
<td>30</td>
<td>40.2</td>
<td>11.3</td>
</tr>
<tr>
<td>Figure 7</td>
<td>$(642 \times 476 \times 484)$</td>
<td>20</td>
<td>515.9</td>
<td>133.4</td>
</tr>
<tr>
<td>Figure 8</td>
<td>$(288 \times 642 \times 288)$</td>
<td>13</td>
<td>87.7</td>
<td>23.2</td>
</tr>
<tr>
<td>Figure 9b</td>
<td>$(298 \times 294 \times 240)$</td>
<td>20</td>
<td>34.3</td>
<td>8.8</td>
</tr>
<tr>
<td>Figure 10b</td>
<td>$(256 \times 298 \times 236)$</td>
<td>15</td>
<td>22.4</td>
<td>5.4</td>
</tr>
</tbody>
</table>

### 6.7. Comparisons with Related Works and Accuracy Test

To evaluate the accuracy of our proposed method, we compare our results with the exact solution. Here, the error for every component is defined as $\|\Phi - \sum_{i=1}^{N} \phi_i \|_2$, where $\Phi$ is the result of Poisson reconstruction from the all of the point clouds. We can see that the reconstructed surfaces are close to the exact solution, as shown in Table 2.

### Table 2. Accurate evaluation of our method.

<table>
<thead>
<tr>
<th>Case</th>
<th>Figure 4</th>
<th>Figure 5</th>
<th>Figure 6a</th>
<th>Figure 6b</th>
<th>Figure 6c</th>
<th>Figure 7</th>
<th>Figure 8</th>
<th>Figure 9b</th>
<th>Figure 10b</th>
</tr>
</thead>
<tbody>
<tr>
<td>Errors</td>
<td>$1.667 \times 10^{-3}$</td>
<td>$1.078 \times 10^{-2}$</td>
<td>$3.746 \times 10^{-3}$</td>
<td>$4.679 \times 10^{-3}$</td>
<td>$7.187 \times 10^{-3}$</td>
<td>$1.742 \times 10^{-2}$</td>
<td>$9.127 \times 10^{-3}$</td>
<td>$1.050 \times 10^{-2}$</td>
<td>$5.440 \times 10^{-3}$</td>
</tr>
</tbody>
</table>

Based on the Cahn–Hilliard system [18,52,53], Li et al. proposed an efficient and robust algorithm to reconstruct the volumes of multi-labeled objects from sets of cross-sections without overlapping regions, artificial gaps, or mismatched interfaces [45]. Our algorithm can also handle cross-sections wherein different regions have different labels. Firstly, we obtain the point clouds from the given set of slice data by using the MATLAB contour file (See Figure 11a,b). Then, we perform our method to reconstruct the surface, as shown in Figure 11b–d. Figure 11e–h are the two-dimensional results at the slice data $z = 20, 50, 80$, and 110, respectively. Here, a mesh grid $128 \times 128 \times 124$ is used. A similar test has been performed in [45]. Observing these numerical results, we can see that our proposed method is qualitatively in good agreement with the results in [45]. Our method is simpler and easy to perform compared to Li et al.’s method because their method is a fourth-order nonlinear partial differential equation and is hard to solve in an explicit manner. On the other hand, our method is a second-order partial differential equation and is easy to solve in the splitting manner. Note that this comparison is unfair because Li et al.’s method [45] can directly reconstruct the volumes of multi-labeled objects from sets of cross-sections. However, our approach should firstly translate the slice data into the point clouds and then perform the method to reconstruct surfaces, which may lead to missing data during
data transforming. To our best knowledge, the presented approach is the first algorithm using the modified vector-valued Allen–Cahn equation for multi-reconstruction from point clouds.

![Figure 11](image)

**Figure 11.** Comparisons with the related work [45]. (a) Given set of slice data. (b) Transformed point clouds. (c) Reconstructed Schwarz diamond minimal surface. (d) Reconstructed Schoen’s F-RD minimal surface. (e–h) Two-dimensional results at the slice data \( z = 20, 50, 80, \text{ and } 110, \) respectively.

### 7. Conclusions and Future Work

We have shown that surface reconstruction can be expressed as a modified vector-valued Allen–Cahn equation, which obtains a smooth surface due to the motion of the mean curvature flow, and we have demonstrated that this approach can robustly reconstruct multicomponent surfaces without self-intersections. Based on operator splitting techniques, our numerical scheme is simple to implement and achieves fast convergence, which can be regarded as an advantage for real-time application. The relative computational cost is also presented. Our algorithm can also handle cross-sections wherein different regions have different labels. Compared to Li et al.’s method [45], our method is simpler and easier to perform. This comparison is considered unfair because Li et al.’s method [45] reconstructs the multi-labeled volumes from sets of cross-sections, and our method is a multi-reconstruction from point clouds. One limitation of our proposed method is that it suffers from high memory usage. As an example, Figure 7 shows the reconstruction of a stele, constructing the voxel grid at a resolution of \( 642 \times 476 \times 484. \) The reconstruction was computed with 1.72 GB of RAM. Trying to compute an equivalent reconstruction with methods, such as the Octree-based approach, would require constructing at a depth of nine and may require an excess of 0.178 GB of memory. In future work, we will present a parallel Octree-based mesh refinement solver for the current algorithm.

**Author Contributions:** J.W. and Z.S. contributed equally to this work. Methodology, J.W. and Z.S.; software, Z.S.; validation, J.W.; formal analysis, J.W.; data curation, J.W.; writing—original draft preparation, J.W. and Z.S.; writing—review and editing, J.W. and Z.S.; All authors have read and agreed to the published version of the manuscript.

**Funding:** Funds for the Central Universities (No. XTR042019005).

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Informed consent was obtained from all subjects involved in the study.

**Data Availability Statement:** See the Institutional Review Board Statement.
Acknowledgments: The authors are grateful to Y. Li and reviewers, whose valuable suggestions significantly improved the quality of this paper.

Conflicts of Interest: The authors declare no conflict of interest.

References
22. Li, Y.; Kim, J. Fast and efficient narrow volume reconstruction from scattered data. Pattern Recognit. 2015, 48, 4057–4069. [CrossRef]


41. Bermano, A.; Vaxman, A.; Gotsman, C. Online reconstruction of 3D objects from arbitrary cross-sections. ACM Trans. Graph. 2011, 30, 1–11. [CrossRef]

42. Huang, Z.; Zou, M.; Carr, N.; Ju, T. Topology-controlled reconstruction of multi-labelled domains from cross-sections. ACM Trans. Graph. 2017, 36, 1–12. [CrossRef]


47. Marseglia, G.; Medaglia, C.M.; Ortega, F.A.; Mesa, J.A. Optimal alignments for designing urban transport systems: Application to Seville. Sustainability 2019, 11, 5058. [CrossRef]


