DeepJoin: Learning a Joint Occupancy, Signed Distance, and Normal Field Function for Shape Repair

NIKOLAS LAMB, Clarkson University, USA
SEAN BANERJEE, Clarkson University, USA
NATASHA KHOLOGADE BANERJEE, Clarkson University, USA

Fig. 1. From an input fractured shape, DeepJoin estimates latent codes for a complete shape and break surface using occupancy (Occ), signed distance function (SDF), and normal field (NF) samples, and predicts a restoration shape to repair the input fractured shape using the estimated latent codes.

We introduce DeepJoin, an automated approach to generate high-resolution repairs for fractured shapes using deep neural networks. Existing approaches to perform automated shape repair operate exclusively on symmetric objects, require a complete proxy shape, or predict restoration shapes using low-resolution voxels which are too coarse for physical repair. We generate a high-resolution restoration shape by inferring a corresponding complete shape and a break surface from an input fractured shape. We present a novel implicit shape representation for fractured shape repair that combines the occupancy function, signed distance function, and normal field. We demonstrate repairs using our approach for synthetically fractured objects from ShapeNet, 3D scans from the Google Scanned Objects dataset, objects in the style of ancient Greek pottery from the QP Cultural Heritage dataset, and real fractured objects. We outperform three baseline approaches in terms of chamfer distance and normal consistency. Unlike existing approaches and restorations using subtraction, DeepJoin restorations do not exhibit surface artifacts and join closely to the fractured region of the fractured shape. Our code is available at: https://github.com/Terascale-All-sensing-Research-Studio/DeepJoin.

CCS Concepts: • Computing methodologies → Shape analysis, Neural networks.

Additional Key Words and Phrases: Shape Representation, Implicit, Repair, Fracture, Deep Learning

1 INTRODUCTION

Household objects often undergo damage, e.g. denting, part loss, weathering, or fracturing. Fractured objects may be reassembled if all of their parts can be found. However, in some cases parts are destroyed during the fracture, e.g. the sugar jar in Figure 2(a), preventing reassembly of the object. Other damage, such as weathering, cannot be repaired using reassembly. Instead the object must be rebuilt using other materials. Users are likely to want to repair an object if the object has some sentimental or functional value. Repair is also necessary if the object is no longer manufactured. With the proliferation of 3D printing, the field of computational fabrication is ideally positioned to enable the repair of damaged objects by generating 3D printable repair parts.

Existing approaches that repair a single damaged object [Antlej et al. 2011; Rengier et al. 2010; Schilling et al. 2014; Scopigno et al. 2011; Seixas et al. 2018] require modeling a repair part by hand using 3D design tools. While users may attempt to manually repair an object if the process is straightforward, repairing large or complex fractures, e.g. the cup in Figure 2(a), is outside the scope of an average user. Most automated repair approaches, e.g. for aerospace components [Gao et al. 2008; Zheng et al. 2006] and medical implants [Harrysson et al. 2007; Witek et al. 2016], are domain-specific and unlikely to generalize. Other automated approaches require fractures to be symmetric to existing object parts [Gregor et al. 2014; Papaioannou et al. 2017], or need to be fed a complete 3D proxy object [Lamb et al. 2019]. Though 3D-ORGAN [Hermoza and Sipiran...}
2 RELATED WORK

Fractured Shape Restoration. Most existing approaches require a user to generate a repair part manually [Antle et al. 2011; Rengier et al. 2010; Schilling et al. 2014; Scopigno et al. 2011; Seixas et al. 2018], which is out of the scope of the average user. Some approaches repair symmetric objects using reflection followed by subtraction [Gregor et al. 2014; Papaioannou et al. 2017]. These approaches cannot repair asymmetrical objects or objects that have undergone symmetric damage. The approach of Lamb et al. [2019] requires a ground truth complete proxy shape as input, which may be unobtainable e.g. in the case of a rare or specialized object. 3D-ORGAN [Hermoza and Sipiran 2018] performs shape completion from a fractured shape without requiring the shape to be symmetric. However, as 3D-ORGAN encodes shapes as 32^3 resolution voxels, their restorations cannot accurately represent the fractured surface and cannot be closely joined to the fractured shape. DeepJoin infers a restoration shape directly without geometric constraints, and generates high-resolution restorations that fit closely to the fractured shape.

Partial Shape Completion. Though not directly related to our work, a large body of prior work has focused on performing shape completion from partial inputs using deep neural networks. Many approaches use point clouds [Achlioptas et al. 2018; Dai et al. 2017; Han et al. 2017; Liu et al. 2020; Pan et al. 2021; Sarmad et al. 2019; Son and Kim 2020; Yuan et al. 2018] due to their compactness. However, point clouds cannot intrinsically represent closed surfaces, which are necessary to generate repair parts that may be 3D printed. Approaches that predict meshes directly [Groueix et al. 2018; Yu et al. 2022] struggle to reconstruct complex shapes [Mescheder et al. 2019], and cannot represent shapes of arbitrary topology. Voxel-based approaches [Brock et al. 2016; Sharma et al. 2016; Smith and Meger 2017; Wu et al. 2016; Zhang et al. 2018] become computationally intractable at high resolutions. Though approaches have reduced the memory footprint of voxels using hierarchical models [Dai et al. 2020, 2018] and sparse convolutions [Dai et al. 2020; Yi et al. 2021], these approaches discretize the output space, rendering them incapable of representing high frequency geometry.

Many recent shape completion approaches encode shapes implicitly using the signed distance function (SDF) [Chabra et al. 2020; Duggal et al. 2022; Hao et al. 2020; Lin et al. 2020; Ma et al. 2020; Park et al. 2019; Sitzmann et al. 2020; Tretschk et al. 2020; Xu et al. 2020; Yang et al. 2021; Zheng et al. 2021], the occupancy function [Chen and Zhang 2019; Chibane et al. 2020a; Genova et al. 2020; Jia and Kyan 2020; Liao et al. 2018; Lionar et al. 2021; Mescheder et al. 2019; Peng et al. 2020; Poursaeeed et al. 2020; Sulzer et al. 2022; Yan et al. 2022b,a], or the unsigned distance function [Chibane et al. 2020b; Tang et al. 2021; Venkatesh et al. 2020]. DeepSDF [Park et al. 2019] introduced an autocoder architecture that uses maximum a posteriori estimation to perform shape completion by estimating a latent code to fit a set of SDF samples. Approaches have extended the
autodecoder architecture by incorporating meta-learning [Sitzmann et al. 2020], encoding shapes at multiple resolutions [Hao et al. 2020], and proposing incremental loss functions [Duan et al. 2020]. Venkatesh et al. [2020] propose a joint shape function using the unsigned distance and NF, and do not learn the occupancy function. We demonstrate that excluding the occupancy function results in inaccurate restoration shapes with a high chamfer distance.

One could imagine an approach that identifies and removes the fractured region of the fractured shape to create a partial shape, applies a shape completion approach, and performs subtraction of the fractured shape from the complete shape to obtain a restoration shape. As we demonstrate in Section 7.2, approaches based on subtraction produce artifacts on the surface of the fractured shape. Attempts to remove artifacts by discarding components that have a volume less than a threshold cannot eliminate connected artifacts. Our approach automatically generates restoration shapes that join closely to the fractured shape without producing artifacts.

3 REPRESENTING FRACTURED SHAPES

We represent complete, fractured, and restoration shapes as point sets \( C, F, R \) respectively. For a given shape \( S \in \{ C, F, R \} \), we represent the occupancy of a point \( x \) as \( o_S(x) \in \{0, 1\} \), where \( o_S(x) = 1 \) if \( x \) is inside \( S \) and \( 0 \) if \( x \) is outside or on the boundary of \( S \). We exclude the boundary of \( S \) to prevent a point from being inside multiple shapes simultaneously, e.g., inside \( F \) and \( R \). We define the break surface, shown in Figure 3 in blue, as a 2D surface that intersects \( F \) at the fractured region. As shown in Figure 3, we define the break shape, \( B \), as the set of points on the same side of the break surface as the fractured shape. Theoretically the break shape \( B \) has an infinite size. In practice, we limit the break shape to a occupy a unit cube. We define the fractured surface, shown in Figure 3 in green, as the surface shared by the fractured shape and the break surface. We express the fractured shape as the intersection of the complete shape and the break shape, i.e. \( F = C \cap B \). Similarly, we express the restoration shape as the intersection of the complete shape and the complement of the break shape, i.e. \( R = C \cap B' \).

For a shape \( S \in \{ C, F, R \} \), we define the SDF value of a point \( x \) as \( s_S(x) \in \mathbb{R} \), the signed distance from \( x \) to the surface of \( S \). The value of \( s_S(x) \) is negative inside the shape and positive outside. We define the NF value of a point \( x \) as \( n_S(x) \in S^3 \), where \( n_S(x) \) is the normal vector of the closest point to \( x \) on the surface of \( S \), and \( S^3 \) is the unit sphere. We define the SDF \( s_B(x) \) and the NF \( n_B(x) \) for the break shape as the signed distance to the fractured surface, shown in Figure 3 in green, and the normal vector of the closest point to \( x \) on the surface of the fractured shape respectively. The value of \( s_B(x) \) is negative on the fractured side and positive on the restoration side.

The set definitions of \( F \) and \( R \) as \( F = C \cap B \) and \( R = C \cap B' \) imply that the point occupancies \( o_F(x) \) and \( o_R(x) \) for \( F \) and \( R \) can be expressed as a logical conjunction of occupancy in the complete shape, and occupancy in the break and the break shape complement, i.e. as \( o_F(x) = o_C(x) \land o_B(x) \) and \( o_R(x) = o_C(x) \land \neg o_B(x) \). The symbols \( \land \) and \( \neg \) represent the logical and and not operators. We relax these logical relationships to work with continuous values using the product T-norm [Gupta and Qi 1991], as

\[
\begin{align*}
o_F(x) &= o_C(x) o_B(x) & (1) \\
o_R(x) &= o_C(x) (1 - o_B(x)) & (2)
\end{align*}
\]

Figure 4 demonstrates the dependency of \( o_F(x) \) and \( o_R(x) \) on \( o_C(x) \) and \( o_B(x) \) given by Equations (1) and (2). In the figure, \( o_F(x) \) and \( o_R(x) \) can be seen as using the break shape and the inverse break shape to carve out the complete shape from a CSG perspective.

To compute the SDF value \( s_F(x) \) for the fractured shape from the SDF values for the complete and break shapes, we express \( s_F(x) \) as

\[
\begin{align*}
s_F(x) &= \begin{cases} 
  s_B(x), & \text{if } o_B(x) = 0 \text{ or } s_B(x) > s_C(x) \\
  s_C(x), & \text{otherwise}.
\end{cases} & (3)
\end{align*}
\]

Visually, Figure 4 demonstrates how the SDF values for \( s_F(x) \) are obtained from the break shape for points outside the break shape and close to the break surface, i.e. when \( s_B(x) > s_C(x) \), following a CSG perspective in SDF space [Breen et al. 2000], similar to the perspective for occupancy. We express the SDF value \( s_B(x) \) for the restoration shape in terms of the complete and break shapes as

\[
\begin{align*}
s_B(x) &= \begin{cases} 
  -s_B(x), & \text{if } o_B(x) = 1 \text{ or } -s_B(x) > s_C(x) \\
  s_C(x), & \text{otherwise}.
\end{cases} & (4)
\end{align*}
\]

We negate values for the break shape in Equation (4) compared to Equation (3) as the restoration shape is on the opposite side of the break surface from the fractured shape. Figure 4 illustrates how the SDF values for \( s_B(x) \) are obtained from the inverted break shape for points outside the break shape and near the break surface where the SDF for the break shape is more positive than the complete shape, i.e. when \( -s_B(x) > s_C(x) \) holds. Computing the SDF for the fractured and restoration shapes as a function of the occupancy for the break shape allows our approach to treat the estimation of the complete and break SDF as two partial shape completion problems, where the partial shapes are defined by the break occupancy.

Similar to the SDF, the NF value at a point \( x \) for a shape \( S \) is given by the closest point on the surface of \( S \) to \( x \). We use the same approach as in Equations (3) and (4) to define the fractured NF \( n_F(x) \)
and restoration NF \( n_R(x) \) in terms of \( C \) and \( B \), as

\[
\begin{align*}
n_F(x) &= \begin{cases} n_B(x), & \text{if } o_B(x) = 0 \text{ or } s_B(x) > s_C(x) \\
n_C(x), & \text{otherwise}, \end{cases} \\
n_R(x) &= \begin{cases} -n_B(x), & \text{if } o_B(x) = 1 \text{ or } s_B(x) > s_C(x) \\
n_C(x), & \text{otherwise}. \end{cases}
\end{align*}
\]

We negate the NF for the break shape in the definition of \( n_R(x) \) in Equation (6), as normals on the restoration shape at the fracture are oriented in the opposite direction of normals on the break surface. Figure 4 shows that the normals for the fractured and break shape are the same outside of the break shape and near the break surface, i.e. where \( s_B(x) > s_C(x) \) holds. Normals for the restoration shape are also the opposite of the break shape inside the break and near the break surface, i.e. where \( -s_B(x) > s_C(x) \) holds.

We represent the joint occupancy, SDF, and NF function \( f^{SB} \) for a shape \( S \in \{ C, B \} \) using a neural network based on the autoencoder architecture of DeepSDF [Park et al. 2019] as shown in Figure 5. We condition the autoencoder for \( S \) by associating \( S \) with a latent code \( z_S \in \mathbb{R}^p \) fed as input to the network, where \( p \) is the size of the code. We use \( p = 128 \) and 64 for the complete and break shapes respectively. We use \( f^C(z_S, x) \), \( f^B(z_S, x) \), and \( f^{SB}(z_S, x) \) to refer to the occupancy, SDF, and NF values predicted at point \( x \).

4 NETWORK OPTIMIZATION

To train our networks, we use a dataset of sample shapes where each sample is a tuple \( \{ C, F, R, B \} \) corresponding to a complete, fractured, restoration, and break shape. We perform optimization over the network parameters, and over the complete and break shape codes. For a shape \( S \), we use the notation \( o_S(x), s_S(x), \) and \( n_S(x) \) to represent the ground truth values for occupancy, the SDF, and NF respectively. During training, we optimize the loss

\[
L_{\text{train}} = \sum_{x \in Z_C, x \in Z_B} L_{CB} + L_F + L_R + \lambda_{\text{reg}} L_{\text{reg}},
\]

where \( Z_C \) and \( Z_B \) are the sets of complete and break shape latent codes. We define the loss \( L_{CB} \) for the complete and break shapes as

\[
L_{CB} = \frac{1}{|X|} \sum_{x \in X} \{ BCE(f^C_{\text{reg}}(z_S, x), o_S(x)) + \}
\]

\[
\lambda_s ||f^C_{\text{reg}}(z_S, x) - s_S(x)||_1 + \lambda_n ||f^C_{\text{reg}}(z_S, x) - n_S(x)||_2, \quad (8)
\]

where \( BCE \) is the binary cross-entropy loss and \( X \) is a set of probing sample points. We define the fractured shape loss \( L_F \) as

\[
L_F = \frac{1}{|X|} \sum_{x \in X} \{ BCE(f^C(z_C, x), f^C_{\text{reg}}(z_B, x), o_F(x)) + \}
\]

\[
\lambda_s L_{F_C} + \lambda_n L_{F_n}, \quad (9)
\]

The first argument to \( BCE \) corresponds to the definition for fractured occupancy from Equation (1). We represent the reconstruction error for the fractured shape SDF \( L_{F_C} \), as

\[
L_{F_C} = \begin{cases} ||f^{SB}_{\text{train}}(z_B, x) - s_F(x)||_1 & \text{if } f^{SB}_{\text{train}}(z_B, x) \leq \mu \text{ or } \\
||f^{C}_{\text{train}}(z_C, x) - s_F(x)||_1 & \text{otherwise}.
\end{cases}
\]

and the reconstruction error for the fractured shape NF \( L_{F_n} \), as

\[
L_{F_n} = \begin{cases} ||f^{SB}_{\text{train}}(z_B, x) - n_F(x)||_2 & \text{if } f^{SB}_{\text{train}}(z_B, x) \leq \mu \text{ or } \\
||f^{C}_{\text{train}}(z_C, x) - n_F(x)||_2 & \text{otherwise}.
\end{cases}
\]

Equations (10) and (11) use the definitions for the fractured shape SDF and NF given in Equations (3) and (5). We use \( \mu = 0.5 \) as a
We evaluate our approach on four datasets.

\[ L_R = \frac{1}{|X|} \sum_{x \in X} BCE \left( f_C^R(z_C, x) (1 - f_B(z_B, x), a_R(x))^+ \right) + \lambda_a L_a + \lambda_n L_n \].

The first argument to BCE in Equation (12) corresponds to the definition of the restoration shape SDF and NF.

\[ L_{Ra} = \begin{cases} 
|f_B(z_B, x) - s_R(x)|_1 & \text{if } f_B(z_B, x) > \mu \\
\|f_C^R(z_C, x) - s_R(x)|_1 & \text{otherwise},
\end{cases} \]

and the error for the restoration shape NF \( L_{Rn} \) as

\[ L_{Rn} = \begin{cases} 
|f_B(z_B, x) - n_R(x)|_2 & \text{if } f_B(z_B, x) > \mu \\
\|f_C^R(z_C, x) - n_R(x)|_2 & \text{otherwise},
\end{cases} \]

which use the definitions for the restoration shape SDF and NF from Equations (4) and (6). We negate the SDF and NF value for the predicted break shape in Equations (13) and (14) as the restoration lies on the opposite side of the break surface from the fractured shape. We define the regularization \( L_{reg} \) as

\[ L_{reg} = ||z_B|| + ||z_C||. \]

We use the Adam optimizer \cite{Kingma-Ba-2014}.

5 INFERRING RESTORATION SHAPES

At inference time our approach generates a restoration shape for a novel fractured shape by performing optimization over occupancy, SDF, and NF samples from the fractured shape to obtain complete and break shape codes. We use the code to predict the SDF value for the restoration shape. During inference we optimize the loss

\[ L_{inf} = L_F + \lambda_{reg} L_{reg}. \]

We find that the surface for the break shape estimated in occupancy space may deviate slightly from the surface in SDF space, causing artifacts if Equation (4) is used to reconstruct the restoration mesh. To prevent artifacting, when generating the restoration mesh we obtain SDF values using the CSG equation for Boolean subtraction \cite{Breen+etal-2006} in SDF space, i.e. \( s_R(x) = \max(f_C^R(z_C, x), -f_B(z_B, x)) \), in place of Equation (4). To obtain a restoration mesh, we perform Marching Cubes \cite{Lorenz-Cline-1987} on a 256^3 grid of points.

6 DATA PROCESSING AND APPLICATIONS

We evaluate our approach on four datasets.

**ShapeNet.** We use 3 ShapeNet \cite{Chang+etal-2015} classes corresponding to commonly fractured objects, i.e. jars, bottles, and mugs, and 5 classes with more complex geometry, i.e. airplanes, chairs, cars, tables, and sofas. We train one network per ShapeNet class.

**Google Scanned Objects dataset.** The dataset \cite{GoogleResearch-2022} contains 3D scanned meshes of household objects such as shoes, pots, and plates. We train one network for the entire dataset.

**QP Cultural Heritage dataset.** The dataset \cite{Koutsoudis+etal-2009} contains artist designed meshes in the style of Greek pottery. We use all meshes for testing on a network trained on ShapeNet jars.

**Real Fractured Objects.** We fracture and scan 3 mugs and use 2 items from Lamb et al. \cite{Lamb+etal-2021}. To test them against networks trained on synthetically fractured ShapeNet mugs and jars respectively.

As the ShapeNet, Google, and QP datasets do not contain fractured meshes, we synthetically fracture meshes from these datasets using the fracturing approach described by Lamb et al. \cite{Lamb+etal-2021}. To generate closed meshes we use the approach of Stutz and Geiger \cite{Stutz-Geiger-2020}. We normalize meshes so they occupy a unit cube. We perform a fracture retention test by fracturing each mesh using a randomized geometric primitive 15 times. If between 5% and 20% of the vertices of the mesh are not removed after 15 attempts we discard the mesh. As the bottles, jars, and mugs ShapeNet classes have less than 600 samples we fracture meshes from these classes 3, 3, and 10 times respectively. We fracture all other meshes once. We retain 24,208 out of 26,166 meshes from ShapeNet, 1,042 out of 1,298 meshes from the Google dataset, and 335 out of 408 meshes from the QP dataset.

Though meshes from the Google dataset are oriented upright they are not facing in a uniform direction. We augment the Google dataset by randomly rotating meshes around the ground plane normal by 90 degrees. We partition the ShapeNet and Google datasets using a 70%/10%/20% train/validation/testing split. To obtain ground truth break surfaces we fit a thin-plate spline (TPS) \cite{Duchon-1977} to the fractured region of each mesh such that the domain of the spline corresponds to a plane that is fitted to the fracture region vertices. We use the TPS to partition sample points into two sets, and denote the set that intersects with the fractured shape as the break shape. We discuss point our point sampling method in the supplementary.

7 RESULTS

We use the chamfer distance (CD), as defined by Park et al. \cite{Park+etal-2019} and the normal consistency (NC), as defined by Mescheder et al. \cite{Mescheder+etal-2019} to evaluate the overall accuracy of predicted restoration shapes. In Section 7.2, we compare our approach to methods based on subtraction from completed shapes. Restorations generated using subtraction tend to exhibit physically implausible surface artifacts. We contribute the non-fractured region error (NFRE) metric to evaluate the degree of surface artifacting. To compute the NFRE we sample \( n \) points on the non-fractured region of the fractured shape and on the predicted and ground truth restoration shapes, and compute the percentage of points on the non-fractured region with a nearest neighbor on the predicted restoration that is closer than \( \eta \) and a nearest neighbor on the ground truth restoration farther than \( \eta \). We use \( n = 30,000 \) and \( \eta = 0.02 \). For success the NFRE and CD is low and the NC is high. In 1.2% of cases, our approach generates an empty restoration. Where applicable, we show the non-empty percentage, (NE%), i.e. the percentage of restorations generated.
We evaluate the impact of learning a joint function for multiple features by training our approach to represent fractured shapes using joint functions for the SDF alone, for occupancy alone (‘Occ’), SDF and NF (‘SDF+NF’), occupancy and SDF (‘Occ+SDF’), and for all three features (‘Occ+SDF+NF’). We show the CD and NE% for each variation of our approach in Table 1, over all restorations predicted by each approach. We do not show results for SDF as we find that the network is not able to learn a stable representation for the break shape and predicts no restorations. As shown in Table 1, using occupancy gives a low CD of 0.099. However, it often fails to converge, with a NE% of 89.4%. Though SDF+NF shows a higher CD than Occ, it predicts restorations more often. Using Occ+SDF shows lower CD compared to Occ, and predicts more restorations than SDF+NF, with a NE% of 98.9%. Adding NF gives lowest CD and a similar NE%, showing that NF improves restoration fidelity.

7.2 Comparison: Fracture Removal and Shape Completion

As no generalizable high-resolution shape repair approaches exist, we compare our approach to three baseline approaches that automatically remove the fractured region and perform shape completion. The baseline approaches use the existing shape completion approaches of DeepSDF [Park et al. 2019], ONet [Mescheder et al. 2019], and ESSC [Zhang et al. 2018]. For each shape completion approach we create a partial input shape by removing the fractured region. To generate partial shapes, we train a point cloud classifier based on PointNet++ [Qi et al. 2017] to classify the fractured region. Our classifier obtains a test accuracy of 97.1%.

We train DeepSDF to reconstruct complete shapes from SDF point samples. We create a partial shape for inference by removing input sample points that have a nearest neighbor in the fractured region of the fractured shape identified by our classifier. We train ONet to reconstruct complete shapes from complete point clouds. We create a partial point cloud for inference by removing points that are classified as belonging to the fractured region identified by our classifier. 3D-ORGAN [Hermoza and Sipiran 2018] generates low-resolution voxelized restoration shapes from fractured shapes. We find that the approach is unstable and does not converge during training. To provide a fair comparison to a voxel-based approach,
we use ESSC [Zhang et al. 2018] to perform partial shape completion. We train ESSC to reconstruct complete voxel grids from partial voxel grids at $32^3$ spatial resolution to match the resolution of 3D-ORGAN. We use an input resolution of $128^3$ encoded with the flipped-truncated SDF (FTSDF) [Song et al. 2017]. During inference, we create a partial input by computing the FTSDF with respect to a mesh with the fractured region, identified by our classifier, removed.

For each baseline approach we obtain a restoration shape by subtracting the predicted complete shape from the input fractured shape in occupancy space. For DeepSDF and ONet we obtain a mesh using Marching Cubes for $256^3$ sample points. As restoration shapes must be closed meshes and not voxel grids, we generate a mesh for ESSC using Marching Cubes at $32^3$ resolution. To mitigate surface artifacts, for each of the baseline approaches we automatically remove connected components from the predicted restoration shape that have a volume less than $\delta$, where we use $\delta = 0.01$.

Table 2 summarizes metrics over non-empty restorations predicted by our approach. Comparative approaches generate all non-empty restorations. DeepJoin outperforms baseline approaches in terms of the mean CD, NC, and NFRE. DeepSDF predicts restorations with a relatively low CD of 0.072 and high NC of 0.583 compared to the other baselines. However, as shown by the NFRE of 0.181 in Table 2, restorations predicted by DeepSDF exhibit artifacts on the surface of the fractured shape e.g. for the bottle, sofa, and pot in Figure 6. Complete shapes predicted by ONet are less accurate than DeepSDF, resulting in restoration shapes with large artifacts that often cover most of the surface of the fractured shape, as shown in Figure 8, and by the NFRE of 0.634. ESSC shows a the lowest mean CD for 2 classes, and often predicts restoration shapes in the correct location and without surface artifacts. However, the restoration shapes do not precisely match the ground truth, as shown by the NC of 0.289, do not join to the fractured shape, e.g. the cutaway in Figure 8, and are too small to fully restore the fractured shape due to their coarse structure. DeepJoin generates accurate restoration shapes that do not exhibit artifacts, as shown in Figure 8.

<table>
<thead>
<tr>
<th>Method</th>
<th>bottles</th>
<th>cars</th>
<th>chairs</th>
<th>jars</th>
<th>mugs</th>
<th>planes</th>
<th>sofas</th>
<th>tables</th>
<th>Mean</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeepSDF</td>
<td>0.024</td>
<td>0.025</td>
<td>0.109</td>
<td>0.134</td>
<td>0.943</td>
<td>0.954</td>
<td>0.076</td>
<td>0.129</td>
<td>0.072</td>
</tr>
<tr>
<td>NFRE</td>
<td>0.084</td>
<td>0.158</td>
<td>0.316</td>
<td>0.217</td>
<td>0.092</td>
<td>0.125</td>
<td>0.215</td>
<td>0.238</td>
<td>0.181</td>
</tr>
<tr>
<td>ONet</td>
<td>0.183</td>
<td>0.122</td>
<td>0.160</td>
<td>0.133</td>
<td>0.150</td>
<td>0.121</td>
<td>0.148</td>
<td>0.190</td>
<td>0.144</td>
</tr>
<tr>
<td>NFRE</td>
<td>0.829</td>
<td>0.753</td>
<td>0.518</td>
<td>0.538</td>
<td>0.523</td>
<td>0.787</td>
<td>0.649</td>
<td>0.473</td>
<td>0.634</td>
</tr>
<tr>
<td>ESSC</td>
<td>0.175</td>
<td>0.069</td>
<td>0.167</td>
<td>0.056</td>
<td>0.052</td>
<td>0.126</td>
<td>0.152</td>
<td>0.122</td>
<td>0.155</td>
</tr>
<tr>
<td>NFRE</td>
<td>0.108</td>
<td>0.388</td>
<td>0.153</td>
<td>0.414</td>
<td>0.481</td>
<td>0.138</td>
<td>0.251</td>
<td>0.377</td>
<td>0.289</td>
</tr>
<tr>
<td>Ours</td>
<td>0.084</td>
<td>0.018</td>
<td>0.089</td>
<td>0.099</td>
<td>0.027</td>
<td>0.033</td>
<td>0.077</td>
<td>0.126</td>
<td>0.062</td>
</tr>
<tr>
<td>NFRE</td>
<td>0.687</td>
<td>0.749</td>
<td>0.567</td>
<td>0.558</td>
<td>0.783</td>
<td>0.660</td>
<td>0.537</td>
<td>0.505</td>
<td>0.631</td>
</tr>
<tr>
<td>Mean</td>
<td>0.052</td>
<td>0.021</td>
<td>0.042</td>
<td>0.054</td>
<td>0.008</td>
<td>0.024</td>
<td>0.064</td>
<td>0.040</td>
<td>0.038</td>
</tr>
</tbody>
</table>

Table 2. Chamfer distance (CD), normal consistency (NC) and non-fracture region error (NFRE) for baseline approaches and our approach. Best metric values are bolded. Mean is computed over class means.

Fig. 8. From left to right: input fractured shapes (gray) and restorations (red) from DeepSDF, ONet, ESSC, DeepJoin, and ground truth.

8 CONCLUSION AND FUTURE WORK

In this paper we present DeepJoin, an approach to automatically generate restoration shapes from input fractured shapes by deconstructing the fractured shape into corresponding complete and break shapes. We contribute a novel method to encode a shape using a joint function for occupancy, signed distance, and NF and use this encoding to regress a high resolution restoration shape. Our approach overcomes the disadvantages of prior work using voxels for automated shape repair, and plays an important role in enabling consumer-driven object repair using additive manufacturing.

Limitations. In 1.2% of cases our approach fails to generate the restoration shape, e.g. on the left of Figure 9, when the break surface is predicted away from the fractured shape. However, as our approach also predicts a complete shape, it enables fallback to generating a restoration using subtraction. As we represent break surfaces using thin-plate splines, our approach is unable to accurately represent concave breaks. In future work, we will explore using more generalizable representations for the break surface, such as non-uniform rational basis splines (NURBS) [Piegl and Tiller 1996]. Our approach may predict a restoration shape that is smaller than the ground truth, e.g. the cabinet in Figure 9, or may predict the restoration shape in the wrong location if the fractured region is small, e.g. the TV stand in Figure 9. However, these repair parts can still be used to partially repair the fracture, e.g. for the table.

Future Work. Though we focus on single component fractures, future work can explore simultaneous estimation of inter-part transformations to facilitate multi-fracture assembly and repair. In future work, we will investigate automated repair deformation near the fracture surface to enable fine-precision joins of 3D prints for physical water-tightness. Though our fractured shapes simulate the physical water-tightness, though our fractured shapes simulate the surface roughness found in ceramic or earthenware objects, proliferation of our approach requires large-scale analysis of fracture across diverse materials. Our future work will contribute a comprehensive fracture dataset of 3D scans of physically fractured objects with accurate waterproofing, to facilitate data-driven study of fracture.