On-edge 2D-to-3D generative pipeline for seamless instance transformation

Jirayu Petchhan[1], Surasachai Doungtap[2]
[1]Department of Computer Engineering, School of Engineering, King Mongkut's Institute of Technology Ladkrabang, Bangkok, Thailand
[2]International Graduate Program of Electrical Engineering and Computer Science, National Taipei University of Technology, Taipei, Taiwan

Article Info

Article history:

Received Jun 5, 2025
Revised Sep 17, 2025
Accepted Sep 27, 2025

Keywords:

3D reconstruction
Edge computing
End-to-end pipeline
Intelligent space
Virtual reality

ABSTRACT

Despite ongoing challenges with fragmented workflows, latency in device imports, and the main issue of limitations in object reconstruction functionality, relying on imperfect extraction networks remains an impractical solution for scalable object generation. To deal with these constraints, we proposed an end-to-end pipeline that leverages a re-designed self-consistency mechanism—aimed at reducing discrimination, along with the beneficial enhancement from level-set projection and gradient-surface orthogonality. In addition, our approach designs dynamic 3D object creation with minimal manual effort by unifying surface topology and optimizing data loading, enabling a streamlined reconstruction process and more flexible object projection. Our method supports rapid, resource-efficient mesh reconstruction and consistently demonstrates performance improvements across multiple instance benchmarks, covering virtual projection tasks. Improvements in mesh topology reconstruction, as measured by the L1 Chamfer distance (CD) metric, are consistently higher, while the system also achieves significant transmission speedups—up to 56.5×—near-instant importing—along with lowering latency in practical rendering on virtual reality (VR) devices. This result highlights that refining mesh binding improves re-creation fidelity. Our approach to scalability leads to faster user engagement and allows automated deployment without requiring human intervention during importing.

Corresponding Author:

Jirayu Petchhan
Department of Computer Engineering, School of Engineering
King Mongkut's Institute of Technology Ladkrabang
No.1, Chalong Krung 1 Alley, Ladkrabang, Bangkok, 10520, Thailand
Email: jirayu.pe@kmitl.ac.th

1. INTRODUCTION

Virtual-physical space and digital twins (DTs) have grown continuously and expeditiously with applications across intelligent spaces, smart appliances, and intelligent commerce [1]–[3]. Many DT concepts draw from the industrial internet of things (IoT), focusing on creating advanced simulation models that use real-time data streams to enhance performance [4].

During the past few years, 3D modeling of tangible things has been a challenging task with uses in computational graphics, gaming, virtual reality (VR), augmented reality (AR), and commercial employment [5]. Usually, 3D parametric representations were created manually using laborious and costly methods such as scanning for 3D models or computer-aided design (CAD) modeling [6]. Implicit sparse point representations (point clouds) are typically used for building models, with programs creating apparel structures from this data [7]. Public online repositories also provide models of real-world apparel. However, the effortless and flawless 2D-to-3D conversion continues to be a race [8].

Computer-assisted learning, such as deep learning, has emerged as a computational tool for generating accurate 3D models from images or texts, offering the potential to revolutionize how 3D structures are created by taking matters in 2D detail. Convolutional neural networks (CNNs), which are frequently employed for thorough extraction, are specifically designed to adapt to target tasks. In particular, our emphasis on reconstruction tasks follows this study. For instance, the multi-view stereo (MVS) [9] enables the prediction of depth maps from a single image for CNNs, while refining the estimations to create complete 3D models. This approach has led to significant advancements in the precision and completeness of generated 3D models [10]. In contrast, they proposed viable proof-of-concept approach when applied to extraction via neural nets.

With the advancement of CNNs, surface reconstruction from 3D point clouds is essential for fields like 3D vision, robotics, and graphics, bridging raw sensor data and editable surfaces [11]. While unsigned distance functions (UDFs) [12]–[14] are used for surface reconstruction, their effectiveness is limited by the discontinuous nature of point clouds, hindering smooth distance field generation, even with large-scale meshes or ground truth values. Consequently, UDF techniques often fail to extract surfaces directly. In the practice of current states and study problems, several internal and external factors influence the use of computation to reconstruct and map the virtual world. Key considerations include whether the granularity of generated data should be evaluated across multiple regenerative models. Next is the smoothness of data mapping, such as reducing reconstruction and projection time [15], seamless automatic projection [16], or even the appeal of the user experience, like gamification [17], to entice people to drive their lives through game-like playing. All of which are crucial for effective facilitation.

However, the previous and existing studies proposed principally theoretical computational perspectives to build higher-dimensional or smoothly reconstructive entities. Some studies have actively applied practical computation to real-world use cases. Consequently, most studies are still stuck on utlizing robust computation, taking much time in running and inference, and relying on manual intervention in some processes. This study looks at reducing many multi-resource consumption usage in practice, such as shifting from top-down to applying on-edge and/or on-device computing and reliance on human-based management.

Therefore, our study proposes extending 2D-to-3D generation for multiple scenarios to address these problems. Our approach aims to adapt simple low-dimensional instances, like 2D objects, sourced from everyday life, to reconstruct them into higher-dimensional shapes seamlessly, that is, 3D objects, and instantly map them into a cyber space, without human intervention. This further leads to adjusting the decisions, like playing games, through VR glasses. Empowering user-generated content (UGC) and generating items from any mobile device to create personalized scenes in virtual space adds both direct and indirect value to the cyber world, as well as inspires individuals to engage in these spaces as they would in the physical world, to overcome existing limitations.

To clarity, this article is arranged as follows: section 2 reviews recent studies on virtual-physical spaces and deep generative models. Section 3 presents our method, while section 4 outlines the experimental results, including the reconstruction and visualization phases, and comparative benchmarking. Finally, section 5 discusses our conclusions, limitations, and leads to future research directions.

2. RELATED WORK

Beginning with the overview of digitalization, numerous studies have highlighted the importance of DTs in human existence [18], [19]. For instance, healthcare DTs use machine learning to aid doctors with medical recommendations and minimally invasive procedures. In smart home advancement, DT study enables collaboration with mobile assistive robots to develop smart applicant systems [20]. In smart agriculture, these technologies lead to cyber-physical systems (CPS), helping farmers optimize resource and equipment usage [21].

Metaverse and DTs research provides a framework for this cultural heritage metaverse, highlighting crucial elements and describing the mapping between actual and cyber cultural heritage worlds. The majority of academics concentrated on the emerging trend of Metaverse, which is a digital environment [22], [23]. Some studies [24], [25] provide a three-layer structure that connects the physical world to virtual space through user interaction. It also investigates the security and privacy concerns associated with employing DTs in the Metaverse. Some discussed digitization and the influence of DTs on the expansion of progressive physical-intelligent cities, alongside the key materials and fundamental technologies of DT-supported smart communities. Urban building was organized into four steps: vision first, IoT perception, application update, and simulation. An approach [26] for designing and implementing DTs for equipment combat damage testing was also developed. This development was important for furthering DT combat emergence and validation [27].

A study explored key aspects of digital reality with 3D parametric modeling and graphic designing, where visual realism can impact real-time interaction [28]. It outlined the VR development process, modern 3D modeling, and texture painting techniques, while also focusing on approaches to enhance interaction speed [29]. Also, AR was discussed and shown in applications using 3D and 4D modeling, with beneficial graphical applications [30]. Historical image textures were applied to add the fourth dimension, namely time duration, to contemporary 3D models. Subsequently, homogenizing the models in graphic software and exporting them into a graphic-displaying space to create the reference surface led to the formed diagram that outlines the required features and functionalities [31].

A recent study focused on 3D reconstruction from implicit sparse points [32]. Implicit-based generative networks (a.k.a. neural implicit function) were built to identify vector-field distance function, handling and calculating both sign values from the point of the nearest measurement points, or occupancies, before using the marching cubes algorithm to obtain a massive mesh topology of a continuous isometric surface from an independent scalar field.

A previous technique used an encoder or an optimization to incorporate the shape in a global latent code, which was afterwards regenerated using a decoder [33]. Some techniques advocate employing more latent codes to record local shape priors, resulting in more exact geometry [34]. The sparse point cloud is first partitioned into uniform grids or native patches, leading to a neural-based model that extracts latent coding for each grid/patch. Further, [35]–[37] presented that the signed function made a more detailed map of the creatable floor plan. Millane *et al.* [38] utilized graphic processing to accelerate the localization map further. Some current studies [39], [40] have described agnostic learning to produce signed distance fields (SDF) while employing properly designed network initialization, gradient restrictions, or geometrical regularization [41]. Neural-pull [42] produced a SDF by creating adjacent rooms on their surface. While study [43] depended on a parameterized surface reconstruction technique, it was hard to form a higher-dimensional shape of stuff. Although one of the studies tried to produce SDFs, it was unable to reconstruct complex structures with open or multi-layer surfaces. All these utilized an implicit generative function [44] to prepare the implicit surface to generate explicit 3D instances easily afterwards.

On the other hand, UDF can construct surfaces directly from gradients. In this period, unsigned distance fields (UDFs) also expanded regularization, such as neural vector fields (NVF) [45], consistency-aware distance [46] (CAP-UDF), learning to anchor (anchor-UDF) [47], and fast meshing to make 3D surfaces [48] (MeshUDF). Most recently, level-set UDF [49] has been proposed using the technique of level-set surface projection, such as illustrating in UDFs, mapping points to the nearest surface using gradient estimation and projection. The projection increases computational complexity. However, leveraging level-set projection gradients can enhance mesh quality, making them a promising approach.

Including the perspective of human-computer interaction, many studies have advanced where they are being integrated with real-world applications for illustration and interaction, for example, computational sensing on 3D-MR systems [50], reconstruction specifically for projecting onto cyber worlds [51], and even hand-sculpting 3D objects for studying real cyber environments [52], [53]. Some studies have worked on applying many computational intelligence variations through various network topologies [54]. However, one missing thing is the motivation for accessibility and automatic creation without human intervention.

In summary, most of the previous studies focus on the utilization of high computational usage for only a single constructive task, namely either point-cloud or full-mesh generation only, even though a study has established an end-to-end pipeline for full-form mesh construction, but they always face a high-computation burden. Likewise, the part of applying edge environment like projecting, most of them still meet limitations of several usage, and they also depend on requiring manual management. Our approach is proposed to tackle all the above current states. Our approach is a fully computational pipeline with lower resource usage to support an edge-computation mechanism. Our technique, following the challenge of UDF techniques, especially the adoption into edge world applications, extends the learning function of an UDF from an implicit 3D generative function, such as sparse point clouds, to create new 3D continuous surfaces. The import was made seamlessly by mapping 3D objects into virtual space. We also summarize the recent studies and our proposed framework in Table 1. Thereby, our proposed highlights have been presented as follows:

- The proposed framework, which is self-controversial to learn different isosurfaces of self, to address our problem setting, becomes an entirely transformable function from the captured low-dimensional physical- or virtual-space instances to the existing-captured shape-like explicit higher-dimensional instances.
- Besides, a given exemplary scenario has been showcased to prove two manifolds: less-time consumption and reliable performance for end-to-end regeneration through our aggregated near-device computation and our instantly transformative framework.

– Several actual implementations have been conducted the garment, object, and scene tasks have been used to conduct comparisons from various perspectives.

Table 1. The comparative summary of recent studies and ours

Studies	Learning strategies	End-to-end 2D-to-3D transformation a visualization into virtual environments		Compatibility with edge environments	
		2D-to-3D surface extraction	3D seamless texturing and visualization	Edge role	Reconstructive time consumption
Implicit 3D generation [44]	S	Δ (2D images to implicit 3D points)	-	-	High ↑
Explicit 3D generation: SDF [32]–[42], UDF [45]–[49]	S (Except [49]: IS)	Δ (Implicit-to-explicit 3D surface)			
3D-VR/AR/MR [50]–[53]	-	Δ (Hand sculpting)	Δ (Visualization only)	Δ (Edge equipment used for projection only)	
Doungtap *et al.* [55]	Implicit & explicit 3D: all S	✓	Δ (Discrete transformation)		
Ours	Implicit 3D: S Explicit 3D: SS	✓	✓ (Seamless transformation)	✓ (Edge machines used for training & inference)	Lower ↓

Note: The learning strategies of the reconstruction model function are denoted by supervised (S), implicit-supervised (IS), and self-supervised (SS). Including the symbols of -, Δ, ✓ represent none, part of, and include all, respectively.

3. OUR APPROACH

In this section, we are describing our approach in three parts. Firstly, the proposed framework, which includes the higher dimension point cloud implicit function, a less-time explicit 3D regenerative function, and a transformation task into virtual space, is concluded as an all-in-one approach. Next, we delve into our modified approach of implicit regenerative function based on UDFs that are further compatible with on-edge real-time instance generation instantly. Lastly, transformation and visualization techniques are used to handle seamless physical-to-cyber mapping continuously.

3.1. Fully transformative function into computing infrastructure

Recent computation intelligence is truly based on deep learning that focuses on neural network development, particularly in virtual-physical space applications, like segmentation [56] and object detection [57]. With the rise of transitional learning, including transfer learning and deep generative modeling [58], research in this field continues to expand. Generative models like GANs and VAEs learn the training data distribution for image generation to generate new, similar samples. An exciting approach is the UDF, represented as voxel grids or 3D textures storing distance values. UDFs, widely used in graphics and geometric modeling, enable shape representation and object rendering. Thus, generative models should learn UDF distributions to generate novel samples.

Earlier, we proposed the collaboration of machine-to-machine 2D-to-3D translation from the large-dedicated server with the storage and distributed computing, processing client data, as mentioned in Figure 1(a), to illustrate this infrastructure. With this current state, it was evident that a massive transmission cost occurred for inbound-outbound offloading. The most difficult challenge is developing new reconstructed samples, prioritizing computational efficiency over lightness.

Thus, our proposed approach in Figure 1(b) is presented by an enhanced framework that completely separates storage in the cloud from computing at the edge. This relies on discrete data transfer for storage outside the sample computation time from on-edge to inter-mobile on-device. Our combined techniques become our fully transformative function. This method improves overall quality and reliability by ensuring the final output is based on a richer, more detailed dataset, which is critical for high-fidelity reconstructions. Shifting from a server-centric to an edge service-to-device transmission model provides several key advantages that are particularly impactful for applications requiring real-time performance. Including obviously reducing the transport time and the regenerative sampling time can be significantly reduced.

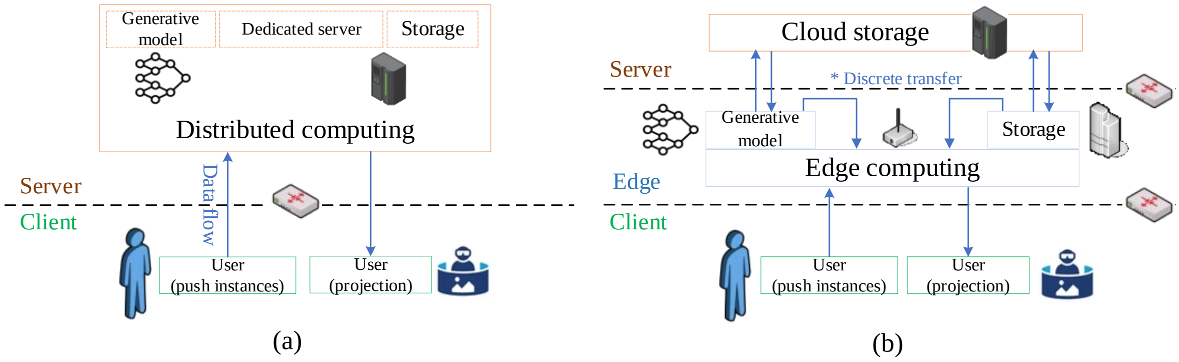

Figure 1. Comparison of framework architectures; (a) previous cloud computing framework and (b) proposed at-the-edge inter-mobile computing framework

3.2. Sparse data points-to-fully explicit instances: self-consistency-based unsigned distance fields

Initially, we still relied on the beneficial implicit function of 2D cyber- or physical-space capture images from various circumstances, and all these are transformable into point cloud coordination-based samples through the pixel-rearranged implicit function [44]. Besides, the highlight this effectively fastens and rivets many points formed as a hint cloud, reconstructing it as a higher-dimensional point cloud object (in this study we pin at three-dimension shape at most) for afterward processing.

Previously, many studies relied heavily on pinning, anchoring, or field aids, leading to prolonged training times and inefficient resource usage. UDF is a scalar field where each point in space stores the minimum distance to a surface, but without distinguishing between the inside and outside of the object. Our previous employment, through the application of CAP-UDF approach [46], enhances model learning and object reconstruction by directly leveraging the gradient field with UDFs-based sparse point rearrangement and consistency-realized gradient fields $\nabla f(\cdot)$. Additionally, continuing from earlier, the surface projection with level-set illustration [49] also works with UDFs, moving a point in space onto the closest surface defined by the zero-level set of the distance field. This is typically done by finding the closest surface, estimating the gradient, and projecting their computed gradient onto the surface, respectively; nevertheless, the two mentioned UDFs, these higher performances come at the cost of increased computational complexity and resource consumption. Fortunately, we still foresee that level-set projection gradient utilizes the advantage of both gradient computation and projection to create a finer and more comprehensive mesh surface, so we see that it can be utilized.

Technically, the estimation of cost function is estimated by the $E_{(p,\ q)\sim(P,\ Q)}(\cdot)$ formulation. This estimation is implemented into the pulling enhancement for learning unsigned distances from raw point clouds without ground truth. Given a raw point cloud in real-number-based 3D space $P=\{p \in \mathbb{R}^3\}^N$ with all of N points, samples of a set of queries in real-number-based 3D space $Q=\{q \in \mathbb{R}^3\}^M$ surrounding the point clouds P. The f function realizes the unsigned distances between queries Q and the shape described by original point clouds P. The goal is to complete full-form mesh objects with indiscriminate typology. In processing, given a query input q_i, resurfacing relocates it against the gradient direction $\nabla f(q_i)$ at q_i, Eventually, the objective loss proceeds the risk minimization between predicted moved queries $\hat{Q}$ and the raw point cloud P, becoming the cost function through Chamfer distance (CD) function as formulated in (1):

$$L_{CD} = E_{(p,q)\sim(P,Q)}\left(\sum_{i\in M}\min_{j\in N}\left\|p_j-\hat{q}_i\right\|_2 + \sum_{j\in N}\min_{i\in M}\left\|p_j-\hat{q}_i\right\|_2\right) \tag{1}$$

where $\hat{q}_i = q_i - f(q_i)\cdot\left(\frac{\nabla f(q_i)}{\|\nabla f(q_i)\|_2}\right)$

Lastly, the Level-Set UDFs technique not only mitigates the issues of fragmentation and discontinuity but also improves the overall performance in various applications, such as point-based cloud upsampling and normal estimation. By projecting non-zero level sets onto the zero-level set with gradient constraints, they align gradients and correct errors, enhancing continuity. Through the perspective of multiple regularization:

a. Projection loss aims to improve feature representation by enforcing orthogonality among class features, enhancing inter-class separation and intra-class clustering. This loss is particularly effective in scenarios with adversarial attacks and label noise, demonstrating robustness across diverse tasks like image recognition and domain generalization, as indicated in (2).

b. Gradient-surface orthogonal loss focuses on aligning gradients across different level sets in UDFs, which helps achieve smoother and more continuous representations of surfaces. Utilizing gradient constraints corrects errors in unsigned distances, improving surface reconstruction outcomes.
c. Surface regularize loss penalizes irregularities in surface representations, promoting smoother transitions and continuity in learned models. This aligns with the goals of the other two losses, enhancing overall model performance in tasks involving complex geometries. Technically, all of them were adjusted using control weights to regulate the effect of resurface gradient activation.

$$L_{proj} = E_{(p,q)\sim(P,Q)}\left(\sum_i^N exp(\lambda|f(q_i)|)\cdot\left(1-\left|\frac{\nabla f(q_i)\cdot\nabla f(\hat{q}_i)}{\|\nabla f(q_i)\|_2\cdot\|\nabla f(\hat{q}_i)\|_2}\right|+\sum_i^N|f(p_i)|\right)\right) \quad (2)$$

Where the function leads to the gradient at queries Q and the gradient at their projections $\hat{Q}$, both moving towards the zero-level set. The exponential term determines the optimizing velocity queries Q and gradients $\nabla f(Q)$ closer to the zero-level set via λ hyper-parameter. Nonetheless, the projection with the queries and gradients is finally degraded due to the unsigned distance limitation; thus, the behind-the-scenes term, namely, the consistency of correcting the zero-level set. Consistency constraint distances are added to assist in generalizing and shaping distance values.

$$L_{orth} = E_{(p,q)\sim(P,Q)}\sum_j^M\left(1-\left|\frac{\nabla f(q_i)\cdot\tau(q_i,p_i)}{\|\nabla f(q_i)\|_2\cdot\|\tau(q_i,p_i)\|_2}\right|\right) \quad (3)$$

Where the orthogonal surface cost is shaped by a similar projection loss but changed to an orthogonal style. Instead of the gradient trajectories, the variable is replaced with the vectorized direction $\tau(\cdot,\ \cdot)$. This vectorized optimization uses query q_i and raw point p_i to account for inconsistencies in the learning function between the closest point p_i and the closest predicted point $\hat{q}_i$.

$$L_{surf} = E_{q\sim Q}\sum_j^M\left(f(q_j)\right) \quad (4)$$

Where the surface regularization depends on the computation of exact point q_j, extracted as several field surfaces and averaged on all samples to give a much easier and smoother shaping surface, on the other hand, utilizing exact points to assist is analogous to calculating the uncertainty on certainty. We can conclude that it is due to supervised certainty. Straightforwardly, the relationship of valuable ground truth of existing point clouds present in this regularization also relies on weekly partial supervision to facilitate the smoothing of this gradient-assisted surface enhancement.

Likewise, for an overview, while these losses show promise, we still envision the challenge in the current state of them that the reliance on specific loss functions may limit flexibility in adapting to various datasets or tasks, suggesting a need for further exploration of hybrid approaches that combine multiple loss strategies for optimal performance. To address this, we proposed the modification of both the explicit function learning and implicit regularization to enhance to more robust and accurate reconstruction of the higher-dimension instances that are composed of two novel manifolds as follows: Firstly, the cost refinement using reliable unsigned distance surface leads to determining the cost distance between the strong and weak consistency; nonetheless, due to the direct distance minimization problem, which includes the problem of concentration, predicting an unsigned distance surface is quite uncertain.

To tackle this issue, we consider that cost optimization of consistently learned unsigned distance textures is a potential approach to understand and improve itself following the protocol of self-supervision tasks. The weak- and strong-consistency learning approach minimizes the bias of the representations to make the unsigned distance measure closer to each other as formulated in (5) and (6):

$$\omega_t = \beta\cdot\omega_{t-1}+(1-\beta)\cdot f(q^w) \quad (5)$$

$$L_{sc} = E_{(q^w,q^s)\sim(Q^W,Q^S)}\sum_j^M\left(\left\|f\left(q_j^s, log\left(\frac{1}{\omega_t}\right)\right)-f\left(q_j^w, log\left(\frac{1}{\omega_t}\right)\right)+\xi\right\|_2\right) \quad (6)$$

Where the present momentum of ω_t is used to reweight the trading-off between the previous predicted unsigned distance ω_{t-1} and the present unsigned distance q^w to regulate the change of the distance field, which is gradually preserved from earlier. The self-pairwise squared distance loss $\|.\|_2$ is utilized to measure the distance between two objects that are different from each other. Figuring out the logarithm of retaining weight log(1/*ω*) provides a rational approach to the required distances, facing mildly and strongly unsigned distance outputs extracted by an explicit generative function.

Once, we observed that the typical neural network in use for translating from three multi-coordinated points to the moved samples was still questionable. In the existing method, misunderstandings and mistakes cause continuous ripples when generating polygon meshes. We therefore have a minor modified surface extraction on the way, that is, the use of a one-dimensional depth-wise separable convolution block is more realizable from three-axis coordinate clouds in UDFs becomes a confident moved query. All these enhanced functions are visualized in Figure 2.

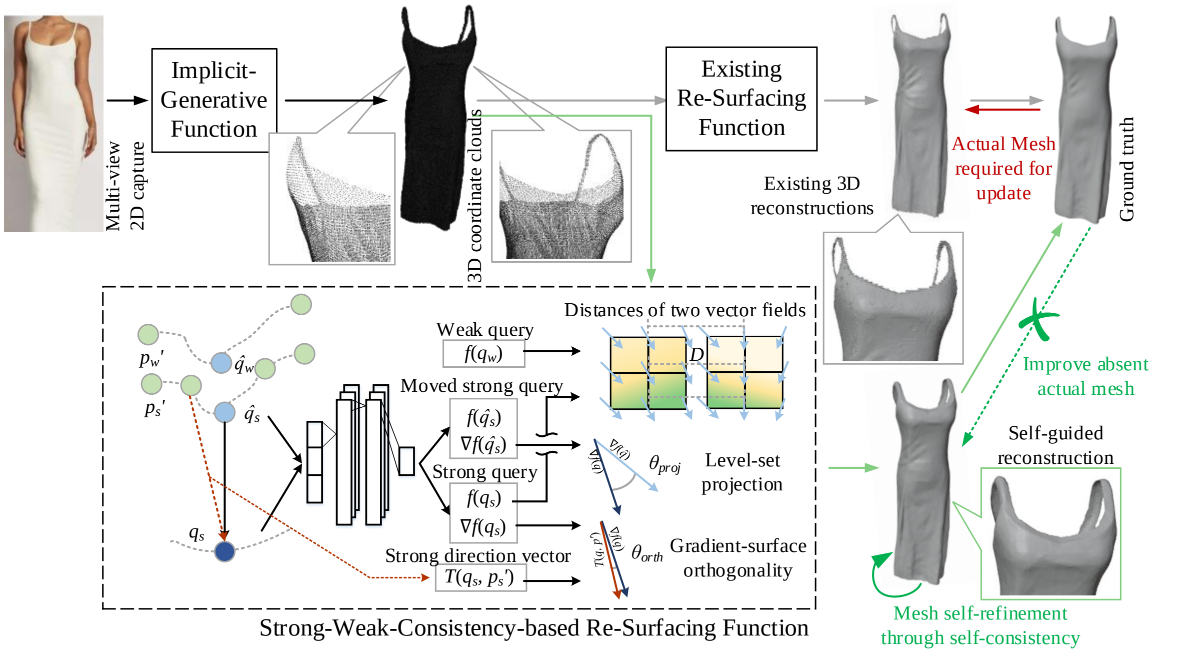
Multi-view 2D capture
Implicit-Generative Function
3D coordinate clouds
Existing Re-Surfacing Function
Existing 3D reconstructions
Actual Mesh required for update
Ground truth
Improve absent actual mesh
Self-guided reconstruction
Mesh self-refinement through self-consistency
Distances of two vector fields
Weak query
$f(q_w)$
Moved strong query
$f(\hat{q}_s)$
$\nabla f(\hat{q}_s)$
Strong query
$f(q_s)$
$\nabla f(q_s)$
Strong direction vector
$T(q_s, p_s')$
D
p_w'
p_s'
$\hat{q}_w$
$\hat{q}_s$
q_s
θ_{proj}
Level-set projection
θ_{orth}
Gradient-surface orthogonality
Strong-Weak-Consistency-based Re-Surfacing Function

Figure 2. Our enhanced resurfacing function with strong-weak uniformity added to the regularization of the level-set projection to enable self-supervision-based resurfacing functionality

The aggregated function in topology mesh optimization is finalized as the base learning of inter-mesh distance, level-set projection, vector-to-surface orthogonality, and our proposed self-discriminable consistency, combined as formulated in (7), each of which regularization term includes the control weights to adjust the influence of dealing with gradient direction in gradient contour space from typical Chamfer-L1 optimization.

$$\underset{j\in N}{min}\underset{i\in M}{min} L_{agg}(p_s, q_s, q_w; i, j)$$
$$= L_{CD}(p_s, q_s; i, j) + \tau_p L_{proj}(q_s; \nabla f(q_s)) \quad (7)$$
$$+\tau_o L_{orth}(q_s; \nabla f(q_s)) + \tau_s L_{self}(q_s, q_w; ; \nabla f(q_s), ; \nabla f(q_w))$$

3.3. Auto-transformation topology and objective function

Dynamic loading of assets from clouds into VR systems enhances scalability and maintains flexibility in content delivery. The VR object application employs dynamic asset loading from cloud-based storage, allowing users to access and interact with a wide variety of garments without the need to pre-package assets within the application. This architecture enables on-demand streaming of models, textures, and related metadata, thereby reducing the initial application size, improving loading performance, and facilitating remote updates.

The system is implemented using Unity’s Addressable Asset System, which supports asynchronous loading of assets via unique identifiers. An object asset prefabs are pre-configured, optimized, and exported as an addressable bundle. These bundles are then uploaded to a cloud content delivery network (CDN) like Firebase Storage or Amazon S3. The application dynamically fetches and instantiates the required assets during runtime based on user interactions, avatar configurations, or stylistic preferences. This dynamic loading process consists of the following steps:

Object models, materials, and animations are labeled and bundled in asset labeling and packaging using Unity's Addressable. For cloud deployment, bundles are uploaded to a remote server or CDN with public or secure access endpoints. While a runtime request, when a user selects an item, such as a jacket or pair of shoes, the system invokes a cloud request using the asset's unique addressable key. Including asynchronous loading, assets are streamed and instantiated in the scene with appropriate positioning, scaling, and reconstructed poly-mesh simulation behaviors. Furthermore, caching and reuse, loaded assets are stored locally in the device cache for future sessions to minimize redundant data transfer.

This approach allows for real-time customization, seasonal content rotation, and personal wardrobe configuration, all without requiring a full application update. Furthermore, it supports extensibility for UGC and marketplace features in future iterations. By decoupling asset distribution from application builds, dynamic cloud loading significantly improves maintainability, performance, and user experience, critical factors in immersive VR environments where responsiveness and personalization are paramount. We have summarized all the 3D-mapping topology in Figure 3.

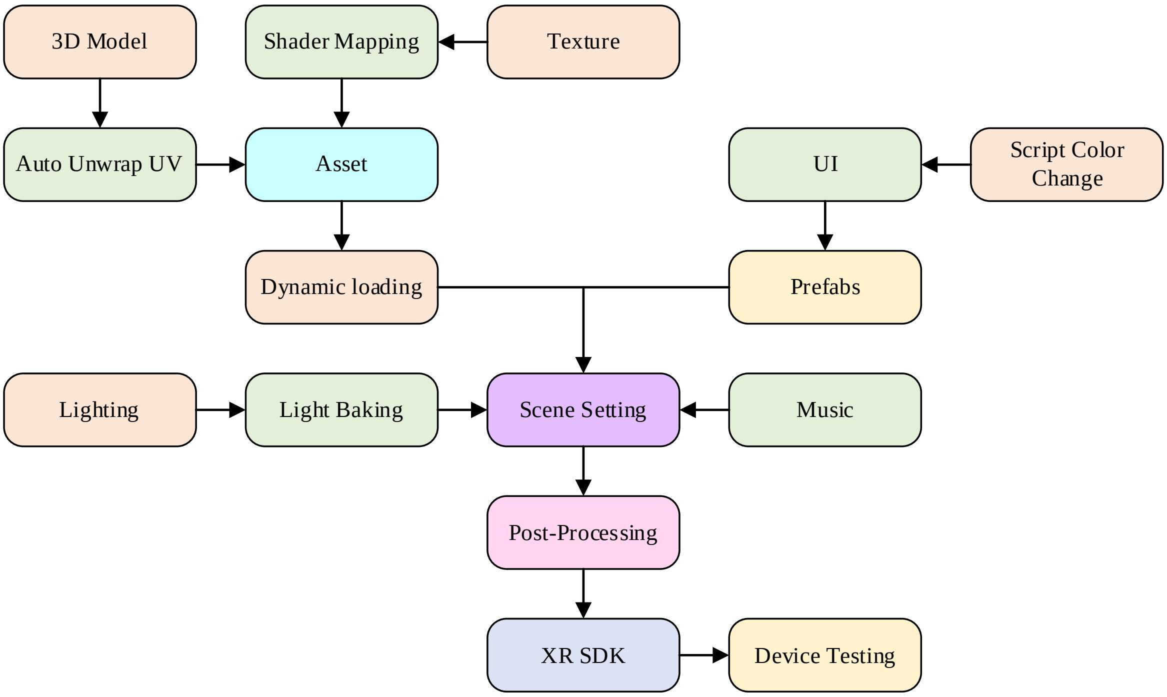

Figure 3. The mesh topology of wrapping and loading our designed automated instance transformation, moving into an interaction space

Our system architecture is built around an end-to-end service that combines comprehensive 2D-to-implicit 3D-to-explicit-3D reconstruction $\forall\Phi(\cdot)$ with our aggregated and minor adjusted functions. Algorithm 1 describes our approach, which has been treated as two states for the edge server and client sides. For instance, User U with mobile device B is sent by the 3D captured objects I. It is shifted to edge-side computers E_g. At a computation event, the proposed transformative pipeline for inter-mobile reconstruction and interaction functions by first generating a purpose-built, interactive instance from captured data. This process is initiated by an event, which triggers the *Fully_Reconstruction_n_Transformation_Phase*. The algorithm regenerates an implicit 2D-to-3D function to obtain well-learned representations from randomized captured instances. An explicit 3D function is then used to reconstruct the mesh, with its topology and learnable representations being optimized from the aggregated data. Thus, in summary, a 3D point cloud is generated first using the implicit 3D generative function, and then our modified function is used to rebuild fully-formed 3D regenerative objects.

Algorithm 1. Proposed transformative pipeline for inter-mobile reconstruction and interaction

```
Require        $I$: Randomize captured instances
               $\Phi_I$: Regenerative implicit 2D-to-3D function with well-learned representations
               $\phi_E$: Reconstructive explicit 3D function with learnable representations $\theta_{repr}$
               $\Phi_T$: Seamless transformative function
               $B$: Physical mobile devices (e.g., smartphone and virtual reality equipment)
               $U_P||U_V$: Physical/virtual space through user interaction perspectives
               $Eg$: Edge dedicated server(s) with fully learned transformative function
Output         A purpose-built instance that is interactable in a virtual space
// Mobile side transmits captured sample data to near edges
$Eg \leftarrow (U_P \circ B)(I)$

// Edge server(s) for object transformation purposes
Function Reconstruct($I$) do
 |  // Create from captured data → sparse points → explicit topology objects → importing
 |  Perform sparse points $I_{PC} = \Phi_I(I)$
 |  Optimize reconstructive mesh $\Phi_E \leftarrow \phi_E(I_{PC}, \theta_{repr})$ from aggregated function in (7)
 |  Perform explicit mesh topology $I_{3D} = \Phi_E(I_{PC})$
End Function

// Implement the defined function to transform into a virtual space
OnEvent Fully_Reconstruction_n_Transformation_Phase do
 |  Reconstruct $I_{3D\text{-}V} \leftarrow E_g(\Phi_T \circ \Phi_E \circ \Phi_I)(I_{captured})$
 |  Procedure projecting over $B$ devices with $U$ interacting $U_V \leftarrow (U_P \circ B)(I_{3D\text{-}V})$
End OnEvent
```

Finally, this reconstructed explicit topology is seamlessly projected onto the user's physical or virtual space using a transformative function, which leverages a dedicated edge server for object transformation. The entire procedure facilitates the creation of a fully reconstructed and interactive object that can be seamlessly experienced across various mobile and VR devices. That is, they are then pushed back to auto-map and auto-visualize into virtual fields and made interactable with virtual-projection devices *B*, all this resulting in a single mainstream pipeline known as the reconstructive and transformative phases.

4. RESULTS

To affirm the feasibility of the proposed approach, we have conducted comprehensive experimental settings and results as follows: datasets in use, system implementation details, experiments, analysis and discussion, and real-world scenarios for simulation.

4.1. Datasets

Our verification has been proven using widespread datasets from real-world perspectives and evaluating mapping into VR hardware. A benchmark dataset was built utilizing automobile examples from ShapeNet data [12], [59], which contain many objects. Stanford 3D scene data [60] includes several scenarios scanned in real-world contexts. This includes the apparel data from the Deep Fashion3D [61] dataset, which consists of 2,078 object models generated from authentic outfits, grouped into 10 classes, and 563 garment samples. We have illustrated all datasets in Figure 4.

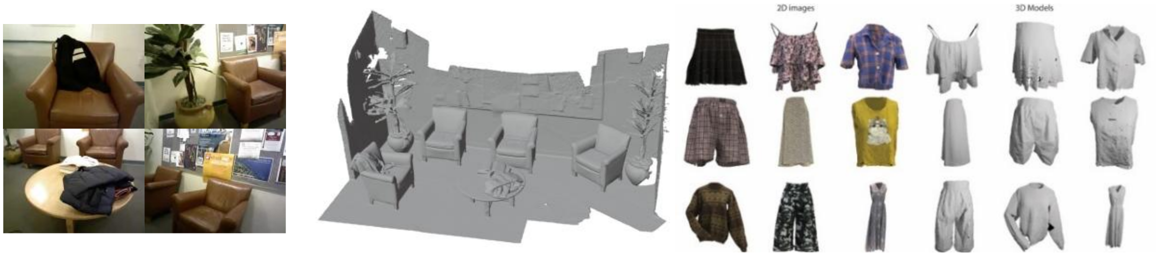

Figure 4. Some captured samples and higher-dimension true-form polygon meshes of scenes from a Stanford 3D scenes dataset (left) and apparel from a Deep Fashion3D dataset (right)

4.2. System implementation details

We have conducted the verification of training and testing through our prepared machines that are composed of; i) typical-performance hardware: CPU i5-13400F with 32 GB installed physical memory and GPU Nvidia RTX 4060Ti with 16 GB physical memory and ii) miserable-performance hardware: CPU i5-10400 with 16 GB physical memory and GPU Nvidia RTX 2060S with 8 GB dedicated memory, all served as our bare metal cloud dedicated servers. The software package is installed to our experimental machine as follows: computing accelerator (CUDA) version 12.1, deep learning accelerator (cuDNN) version 8.9.7, a machine learning environment including PyTorch version 2.5.1, Torchvision 0.20.1, and Chamferdist 1.0.0 for measuring reconstructed polygon mesh distances.

For the preparation of data and the function of implicit and explicit extraction, a pixel-rearranged implicit function [44] is implemented to pre-process low-rank sampled data to become high-rank multi-anchor points of 100-1,000 points. This study is going to compare with the existing studies mentioned in our problem issues from current states as follows: Anchor-UDF [47], CAP-UDF [46], LevelSet-UDF [49], and Doungtap *et al.* [55] clarifying that all it is conducted under the protocol of tasks of supervision and semi-supervision. In contrast, our function stands firmly for the self-supervision task with our proposed strong-weak consistency unsigned distance measures. In this run, we adopt the strong implicit objects for up to 1,000 points per square meter extracting, along with the weak implicit object, around 100/500 points per square meter.

For hyper-parameter setting, we follow the same setting for fairness. The Adam has an initial learning rate of 10^{-3}, a cosine learning rate schedule, and 1,000 warm-up steps. The complete batch running of object and scene reconstruction is conducted in up to 40,000 and 100,000 iterations. While for the surface extraction network, all existing studies use traditional neural networks with 8 layers with skip residual input at the 4-th layer, each of which is composed of 256 neurons. Conversely, our approach adopts the 1D-convolutional layer, formed as a 1D-depthwise separable convolution layer composed of 6 layers and 128 or 256 neurons with twice skipping residual input at the 2-nd and 5-th layers instead. During hyper-parameter optimization, we fine-tuned the regularization control weights—specifically, the projection, orthogonality, and self-consistency weights. For object-level testing, the optimal values are determined to be 10^{-3}, 10^{-4}, and 10^{-2}, respectively, while for scene-level testing, the corresponding values are found to be 2×10^{-3}, 10^{-2}, and 10^{-1}.

Afterwards, to evaluate metrics, we use the CD metrics, which measures the metrics under the main optimization of the 3D-reconstructed protocol. The CD measures the closest point in one set to the other, summing squared distances. Both L1 and L2 normalization distances and projection loss are used for evaluation. Traditional metrics, such as the F-score with thresholds of 0.005 and 0.01, are also considered. Besides, we apply the F-score to measure the categorization performance of object classes.

For the provided topology details, establishing a network for our 2D-to-3D reconstructive object pipeline requires a strategic, multi-stage procedure to ensure low latency and high data throughput. Initially, the user's mobile device acts as a client, capturing 2D images and transmitting this raw data to a nearby edge computer (RTX 2060S) over a low-latency wireless network, such as Wi-Fi 6, being the preferred standard due to its high throughput and low latency. This edge device then performs the initial data processing, converting the images into a pre-processed format like a sparse point cloud. This partially processed data is then securely sent to a powerful central server over a high-bandwidth connection, utilizing an efficient binary protocol like gRPC to minimize transmission overhead. With its powerful generative model, the server reconstructs the final, high-fidelity 3D model, which is stored locally. Finally, upon request, this 3D model is streamed or transmitted from the edge server to the user's VR/AR projection device, ensuring a seamless and immersive interactive experience. A dedicated wireless VR/AR streaming protocol may be used. The total bandwidth required for streaming a 3D model can vary widely, but a sustained transfer rate of at least 150-200 Mbps is a reasonable target to ensure a smooth, lag-free experience. For our generative pipeline, we further adopt the exemplary materials from https://github.com/junshengzhou/LevelSetUDF.

4.3. Experiments

Table 2 presents a comparative evaluation of our method against several state-of-the-art approaches on representative samples from the ShapeNet dataset, specifically a *Garment (Dress)* and a *Vehicle (Truck)*, each trained over 40k iterations. We report four key metrics: Chamfer-L1 distance (↓), Consistency (↓), and F-score at thresholds 0.005 (↑) and 0.01 (↑), where lower Chamfer and consistency errors and higher F-scores indicate superior performance.

For the *Garment (Dress)* sample, our method achieves a Chamfer-L1 distance of 0.001580, which is comparable to top-performing methods such as LevelSet-UDF (0.001516) and CAP-UDF (0.001536), while outperforming all baselines in terms of consistency, achieving a significantly lower error of 5.075×10^{-6}—close to the ground truth (3.001×10^{-6}) and substantially better than others by an order of magnitude. Our approach also yields the highest F-score at the strict 0.005 threshold (0.970), surpassing all compared methods. As visualized in Figures 5(a) to (d), the shoulder section of a given dress deteriorates using existing methods, unless we can reduce the computation nodes, while achieving better reconstructed meshes through the global objects.

Table 2. Evaluation of metrics over exemplary dress and vehicle runs under 40k iterations (sampling of ShapeNet data)

	Garment (dress sample)				Vehicle (truck sample)			
Studies	Chamfer-L1 ↓	Consistency ↓	F-score-0.005 ↑	F-score-0.01 ↑	Chamfer-L1 ↓	Consistency ↓	F-score-0.005 ↑	F-score-0.01 ↑
Ground truth	0.001212	3.001×10^{-6}	0.9897	0.999	0.004502	7.221×10^{-5}	0.9495	0.999
Implicitly [44]	0.005122	4.209×10^{-4}	0.854	0.912	0.00601	13.101×10^{-5}	0.654	0.899
Anchor-UDF [47]	0.001734	1.211×10^{-4}	0.922	0.954	0.005277	11.792×10^{-5}	0.85	0.902
CAP-UDF [46]	0.001536	1.060×10^{-4}	0.94	0.995	0.005014	9.811×10^{-5}	0.8405	0.991
Doungtap *et al.* [55]	0.001534	1.034×10^{-4}	0.949	0.996	0.005001	9.798×10^{-5}	0.8516	0.993
LevelSet-UDF [49]	**0.001516**	6.296×10^{-5}	0.957	**0.997**	0.004986	8.637×10^{-5}	0.9193	0.997
Ours	0.00158	**5.075×10^{-6}**	**0.97**	**0.997**	**0.004919**	**8.496×10^{-5}**	**0.9229**	**0.998**

Note: ↑ and ↓ stand for the highest one and the lowest one, those of which are the best capability that should be performed.

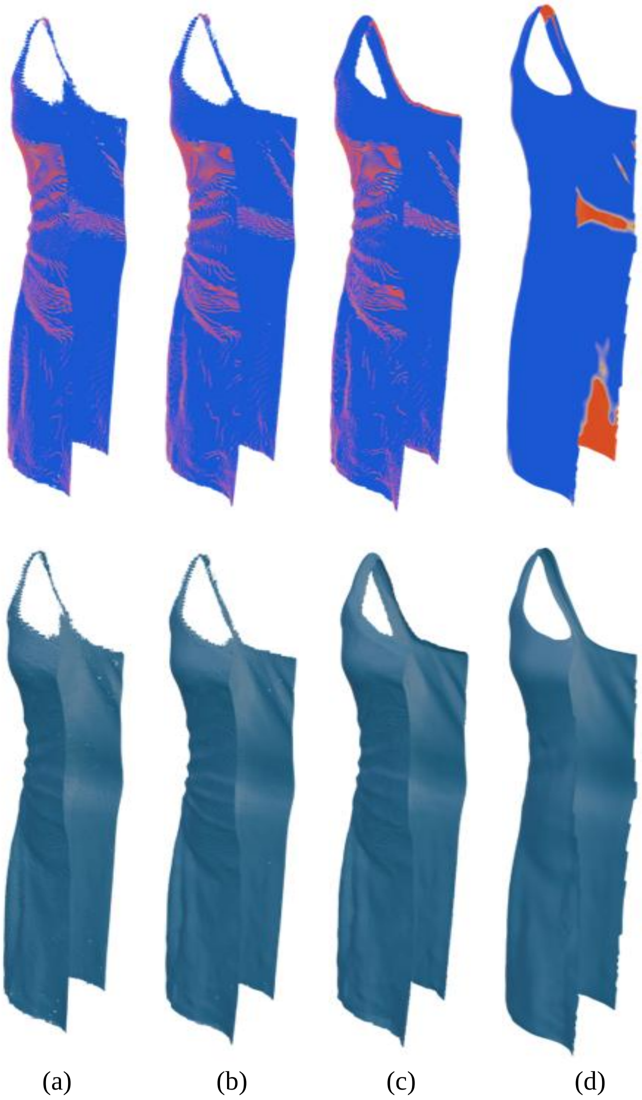

Figure 5. Heatmap and explicit dresses illustrated by; (a) CAP-UDF, (b) LevelSet-UDF, (c) ours, and (d) an actual poly-mesh dress from left to right columns, respectively, sampling on the ShapeNet dataset

On the *Vehicle (Truck)* sample, our method again delivers strong results, with the lowest Chamfer-L1 error (0.004919), a low consistency error (8.496×10^{-5}), and the highest F-scores at both thresholds (0.9229 at 0.005 and 0.9985 at 0.01). Compared to the same existing approaches, our method demonstrates a consistent advantage in reconstructive accuracy and surface completeness across both object categories. Including the illustration in Figures 6(a) to (d), which shows that the platform for stepping onto the truck is mostly away from the existing method, our method gets even more detailed in its reconstruction. These results validate the effectiveness of our approach in producing high-fidelity and topologically consistent 3D reconstructions, with notable improvements especially in strict evaluation regimes.

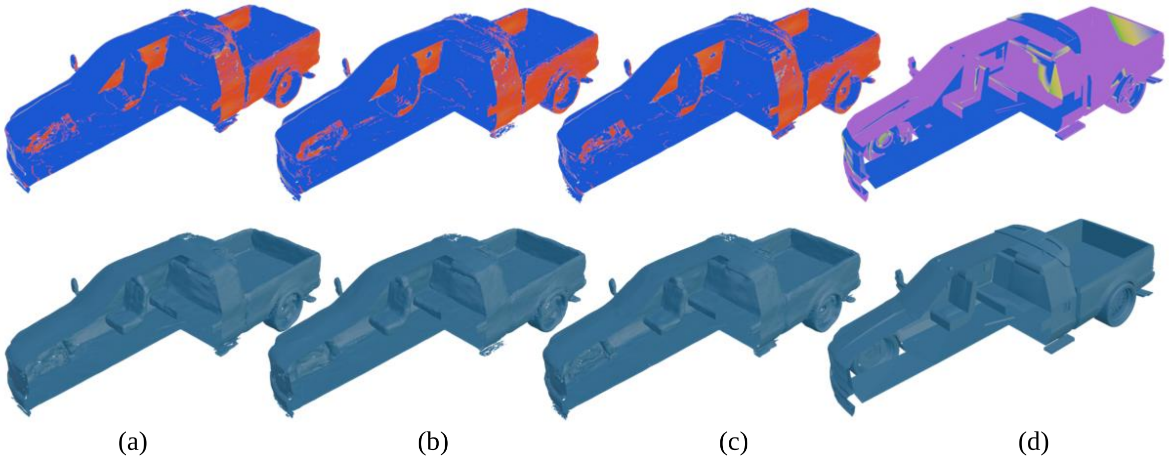

Figure 6. Heatmap and explicit trucks illustrated by; (a) CAP-UDF, (b) LevelSet-UDF, (c) ours, and (d) an actual poly-mesh vehicle from left to right columns, respectively, sampling on ShapeNet dataset

To assess the scalability and robustness of our method in complex environments, we conducted experiments on a wide-area *lounge scene* sampled from the Stanford 3D dataset. The evaluation was performed with a dense point distribution of 1,000 points per square meter and trained over 100k iterations. Table 3 reports quantitative comparisons with several state-of-the-art methods using Chamfer-L1, Chamfer-L2 ($\times 10^{-6}$), and surface consistency metrics (↓ denotes better performance).

Table 3. Evaluation over the exemplary specific indoor scene of a lounge scene with 1,000 points per square meter runs under 100k iterations (sampling of Stanford 3D dataset)

Studies	Chamfer-L1 ↓	Chamfer-L2 (10^{-6}) ↓	Consistency ↓
Ground truth	0.0019	0.1575	0.0005
Implicitly [44]	0.0078	0.8315	0.00031
Anchor-UDF [47]	0.0045	0.5411	0.0002
CAP-UDF [46]	0.0028	0.3254	0.00012
Doungtap *et al.* [55]	0.0028	0.324	0.00012
LevelSet-UDF [49]	0.0027	0.281	0.0001
Ours	**0.0026**	**0.2587**	**0.00009**

Our method achieves the best overall performance across all metrics. Specifically, it records the lowest Chamfer-L1 distance of 0.00257 and Chamfer-L2 distance of 0.2587×10^{-6}, indicating improved geometric fidelity in capturing detailed spatial structures of the indoor environment. Our approach achieves the highest consistency with a minimal error of 0.00009, outperforming the previous best (LevelSet-UDF [49], 0.00010).

Based on a subjective visual assessment of Figures 7(a) to (d), our object reconstructions are better than the reconstructions in the previous studies for both the overall indoor scene and a specific area, like the potted plant and the sofa. Our adjustment produces smoother, more complete objects, especially the trunk, leaves, and the furniture next to it, which are not fused. The reconstructions are less noisy and resemble the proper reference more closely.

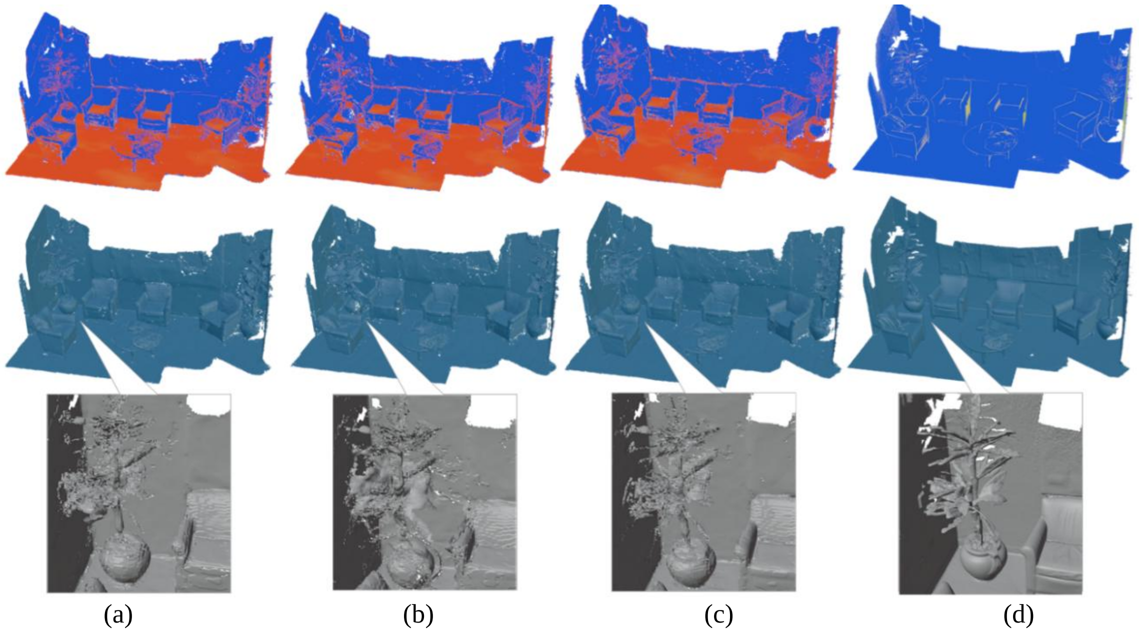

Figure 7. Heatmap and explicit scenes illustrated by; (a) CAP-UDF, (b) LevelSet-UDF, (c) ours, and (d) a true poly-mesh wide scene from left to right columns, respectively, sampling on the Stanford 3D dataset

These results underscore the effectiveness of our method in handling high-density, large-scale reconstruction tasks with superior accuracy and structural stability. The performance gains, especially in complex geometry and occlusions typical of indoor scenes, demonstrate the method's practical applicability to real-world 3D scanning and scene understanding tasks.

4.4. Analysis and discussion

Self-Consistency Capability: to evaluate the effectiveness of the proposed self-consistency regularization, we delve into the object-shaped comparison, like mesh-to-mesh distance, utilized by the L1-distance between predicted outputs and corresponding targets across two different categories: *Vehicle (Van)* and *Garment (Dress)*. As clearly tabulated in Table 4, incorporating our self-regularization significantly improves performance, for example, in 1,000 points per square meter, particularly in the *Garment (Dress)* category. Without self-regularization, the L1-distance for *Vehicle (Van)* and *Garment (Dress)* is 0.00595 and 0.00917, respectively. With the proposed self-regularization, the L1-distance for *Vehicle (Van)* slightly decreases to 0.00586, and *Garment (Dress)* is substantially reduced to 0.00160. These results demonstrate that our regularization approach enhances consistency and reduces prediction error, especially for categories exhibiting higher variability or deformation.

Table 4. Ablation studies and computation complexity with and without further self-regularization

Techniques	L1-distance (1,000 points/m^2)			L1-distance (10,000 points/m^2)		
	Vehicle (Van)	Garment (Dress)	Computational usage (GB)	Vehicle (Van)	Garment (Dress)	Computational usage (GB)
w/o self-regularization ($ChamferL_1 + L_{proj} + L_{orth}$)	0.00595	0.00917	3.9~6.8	0.00798	0.01251	6.1~7.8
w/ proposed self-regularization ($ChamferL_1 + L_{proj} + L_{orth} + L_{self}$)	**0.00586**	**0.00160**	**2.1~3.2**	**0.00670**	**0.00255**	**3.6~4.5**

Nevertheless, as expected, adding our self-regularization term, L_{self}, increases the computational overhead. For a point cloud density of 1,000 points per square meter, the baseline method (without self-regularization) required 3.9-6.8 GB of GPU memory. In contrast, the method incorporating L_{self} just consumed 2.1-3.2 GB. This usage represents a moderate decrease in memory usage. Even though it is primarily attributed to the storage of intermediate data and gradients necessary for calculating the additional loss term, we still utilize the memory computation more efficiently. This trend is even more pronounced with higher density point clouds. At 10,000 points per square meter, the memory usage for the baseline method rose to 6.1-7.8 GB, while our full function required 3.6-4.5 GB. The consistent decrease in memory consumption across both density levels confirms that the self-regularization term diminishes a measurable computational burden to the training process.

Resource Usage and Computational Complexity: Table 5 compares the proposed method with an existing baseline regarding model size, memory bandwidth, computational complexity, and runtime efficiency under the surface extraction setting. Our method demonstrates substantial improvements in both resource efficiency and computational scalability. Specifically, the number of parameters is reduced from approximately 463.1k to 147.2k, and the bandwidth usage decreases from 1.78 MB to 0.57 MB. Moreover, our method improves computational complexity from quadratic time, *$O(n^2)$*, in the existing approach to linear *L* time, *O(L)*.

Table 5. Model complexity (surface extraction and function perspectives)

Approach	Surface extraction setting			40k-iter time batch run (min) ↓	
	#Parameters (x 10^3 unit) ↓	Network bandwidth (MB) ↓	Complexity	Vehicle (Van)	Garment (Dress)
Level projection [49]	~463.1	1.78	*$O(n^2)$*	~18.0	~23.5
Ours	**~147.2 (x 1/3.2)**	**0.57 (x 1/3.1)**	***O(L)***	**~13.5 (x 1/1.3)**	**~12.3 (x 1/1.9)**

These improvements are reflected in the runtime performance as well. For the *Vehicle (Van)* category, batch run time is reduced from approximately 18.0 minutes to 13.5 minutes. For the *Garment (Dress)* category, the runtime decreases substantially—from 23.5 minutes to 12.3 minutes. These results highlight the efficiency and scalability of our approach, making it suitable for real-time or resource-constrained applications without compromising performance.

Resource Usage from a Practical Deployment Perspective: from conceptual dynamic loading to further assess the practicality of our method, Table 6 presents a comparative analysis of resource consumption and processing efficiency in a real-world use case, specifically for the *Garment (Dress)* category aspect. A comparison is made between an existing high-performance server-based method and our proposed approach, which is designed for edge deployment. While the baseline leverages a powerful dedicated server equipped with typical hardware and achieves a reconstruction time of approximately 12.3 minutes, it incurs a high transmission cost of about 11.3 minutes. It relies on human intervention for visualization by hand, making it less suitable for scalable or automated deployment.

Table 6. Completed the up-down processing run through the given dress example

Approach	Resource		Thorough end-to-end pipeline processing time usage (min)			
	Topology	Accelerator	Reconstruction time cost ↓	Transmission time cost ↓	Visualization time cost ↓	Total ↓
[55]	Server	RTX 4060Ti	~12.3	~11.3	N/A (Depends on time taken by man)	
Ours	Edge	RTX 2060S	**~21.2 (x 1.7)**	**~0.2 (x 1/56.5)**	**~0.1**	**~21.5**

In contrast, our method operates effectively on a significantly less powerful edge device with performance-limited hardware. Despite a longer reconstruction time (up to 21.2 minutes), our approach gives substantial advantages in downstream efficiency. The transmission cost is drastically reduced to around 0.2 minutes, and the visualization cost is negligible, just 0.1 minutes following the automatic importing function as designed, leading to a total end-to-end processing time of around 21.5 minutes. This demonstrates that our method is well-suited for low-bandwidth, edge-level deployment scenarios, offering a favorable trade-off between computational load and operational autonomy. The transmission cost is evaluated under typical 100/1000 Gbps bandwidth settings, consistent across both approaches.

Rendering Latency in Practical Use: before assessing runtime performance, we analyzed the geometric complexity of the generated mesh topologies, as it plays a crucial role in memory efficiency and rendering speed. In 3D graphics, vertices represent points in space, edges are straight lines connecting pairs of vertices, and faces (typically triangles) are planar surfaces defined by sets of edges. Together, these elements define the structure and density of a 3D mesh, with denser topologies often leading to higher computational and memory demands. As summarized in Table 7, our method produced the most compact mesh representation, with the fewest vertices (822.03K), edges (1,024.45K), and faces (408.95K), while also achieving the lowest memory consumption at 57.3 MB. These outputs indicate a more efficient representation without sacrificing structural integrity.

Table 7. Memory allocation in processing and visualizing the exemplary lounge scene

Approach	Mesh topology memory usage (MB) ↓	Vertices (x10^3) ↓	Edges (x10^3) ↓	Faces (a.k.a. triangles) (x10^3) ↓
[46]	57.5	827.62	1,031.52	411.80
[55]	57.9	835.10	1040.36	415.01
[49]	58.0	837.83	1,044.03	416.74
Ours	**57.3**	**822.03**	**1,024.45**	**408.95**

Following this, we conducted runtime performance testing using the Meta Quest 3 (which includes Snapdragon XR2 Gen2 processor and 8 GB of memory, rendering budget for 90 FPS and 11 ms at the default) to evaluate real-time rendering capabilities. As shown in Figure 8, our method attained the highest estimated frame rate reduced by highly rendering cost, such as topology bandwidth, vertex cost, and triangle cost for supporting rendering object-reconstructed task, reaching 58.5 FPS, outperforming other approaches such as the consistent-aware field [46] (58.3 FPS), Doungtap *et al.* [55] (58.1 FPS), and level-set projection [49] (58.0 FPS). These results highlight the effectiveness of our mesh optimization strategy in reducing computational overhead, enabling smoother and more responsive performance in memory-constrained, real-time environments.

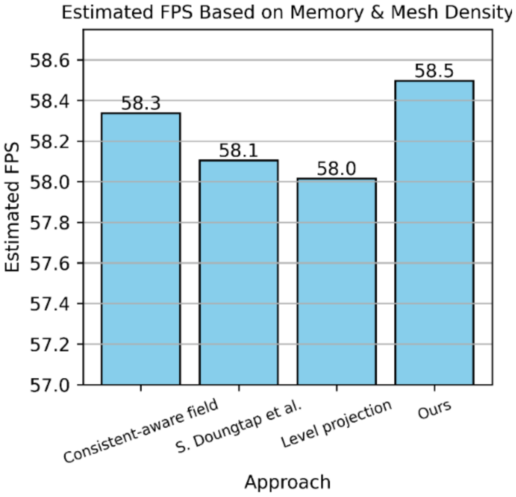

Figure 8. Estimated FPS calculated between memory usage and topology density run through Meta Quest 3 equipment

Discussion and Interpretation of Key Findings: we found that self-consistent contrastive learning correlates with the feasibility of better generalization and regeneration of higher-dimensional objects more smoothly and robustly, for instance, the relevance of findings between fitting and 3D sampling on an exemplary scene in Table 3 against Figure 7. The aggregated method in this study tended to have an inordinately smaller proportion of several resource perspectives; however, it still required a few higher computational complications to learn surfacing. Our findings indicate that higher algorithm complexity is not associated with poor performance in resurfacing from implicit clouds. The proposed method may benefit from debiasing the instances without negatively affecting the traditional query vectorization. In the meantime, this study investigated a comprehensive point-to-mesh object reconstruction and practical visualization on mobile-edge topology.

However, additional in-depth studies may be required to confirm its actual uses and long-term robustness, particularly regarding the proposed self-consistent algorithm. Our findings show that adopting the approach to any constructive environment is more resilient than traditional methods. Future challenges may look into higher efficiency and seamless methods concurrently, such as for producing re-texturing and naturally moving surfaces. Furthermore, in aspects of edge-networking mechanism, the cross-on-device bi-direction transfer may benefit this proposed reconstructive network in real-world implementation concretely.

Deformation Scenario: in addition, we perform object reconstruction from the captured incomplete point set, such as those obstructed by obstacles, causing deformation as shown in Figures 9(a) to (e). Our approach demonstrates superior performance in reconstructing the fine-grained details of the non-rigid object. While the level-set projection method captures the general form, its output shows noticeable surface artifacts and a lack of detail in the garment's folds. In contrast, our approach produces a cleaner, smoother mesh that more faithfully preserves the cloth's intricate geometry and subtle deformations. Although this qualitative improvement highlights the robustness and precision of our method in handling the challenges of complex point cloud data, some minor surface irregularities are still present when compared to the flawless ground truth model. Technically, our reconstruction method does not have a human workload to correct some surface modifications. Even though it is better than the existing methods, this limitation of some defective surface areas may pose a challenge for the future to fulfill the surface with a self-awareness learning function automatically.

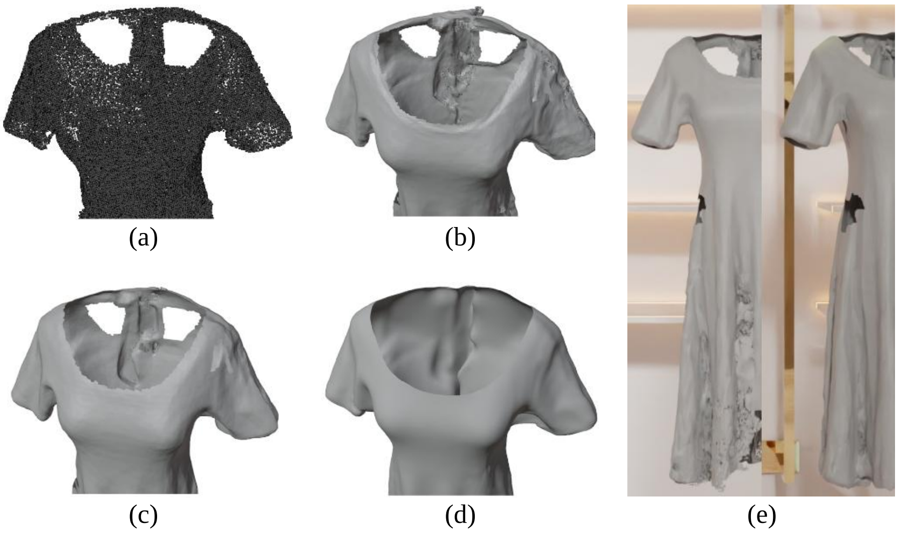

Figure 9. Comparison of apparel reconstruction from; (a) a deformed point cloud apparel, (b) level-set projection method, (c) ours, (d) absolute ground truth, and (e) reconstructive objects from an existing study (left) and ours (right) projected onto a scene

5. CONCLUSION

From the current state, creating new higher-dimensional assets is still scattered, and importing devices takes considerable time due to both downstream data transfer and visualization. With our continuous end-to-end pipeline, we first propose a self-supervised new object creation by reducing discrimination between the two regenerated instances to create a more flexible and unified topology surface, alongside creating a further structure to wrap and load dynamically on-the-fly for easy import instead of manual intervention. Recent observations indicate that the use of strong-weak consistent querying. Our key findings offer definitive proof that this phenomenon is linked to better reconstruction alteration, rather than being adopted by only the traditional querying method. The bench tests show that the implementation is resource-dependent and performs efficiently from multiple perspectives. This processing includes using the new user-facing edge network to create higher-resolution instances. Key findings are that it can be more human-centric and seamless through the following: In terms of efficiency, the generalization achieves mostly the lowest mesh reconstruction loss for the references given. Transporting 3D objects onto virtual projection devices, it is 56.5 times faster. Mapping data into those without human intervention takes just 0.1 seconds, including rendering speed, fewer generated mesh topologies, and higher FPS for visualization. All of this allows the proposed approach to demonstrate the initial capabilities of effortless creation and conversion systems.

However, our following concerns still exist: the ability to persuade people to use the behavioral engagement tool remains uncertain. Using gamified incentives may help facilitate access. The terms of object re-creation functions included, but continuous improvement is still required to strengthen the mesh topology network potential further after extraction.

ACKNOWLEDGMENTS

This study utilized computing resources provided by the Software and Artificial Intelligence Group (SAIG) laboratory. This article has been verified for linguistic accuracy by the Office of Academic Journal Administration from King Mongkut's Institute of Technology Ladkrabang.

FUNDING INFORMATION

This study is financially supported by King Mongkut's Institute of Technology Ladkrabang Research Fund with Grant Number: KREF186728.

AUTHOR CONTRIBUTIONS STATEMENT

This journal uses the Contributor Roles Taxonomy (CRediT) to recognize individual author contributions, reduce authorship disputes, and facilitate collaboration.

Name of Author	C	M	So	Va	Fo	I	R	D	O	E	Vi	Su	P	Fu
Jirayu Petchhan	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Surasachai Doungtap		✓	✓	✓		✓				✓	✓			

C : **C**onceptualization
M : **M**ethodology
So : **So**ftware
Va : **Va**lidation
Fo : **Fo**rmal analysis

I : **I**nvestigation
R : **R**esources
D : **D**ata Curation
O : Writing - **O**riginal Draft
E : Writing - Review & **E**diting

Vi : **Vi**sualization
Su : **Su**pervision
P : **P**roject administration
Fu : **Fu**nding acquisition

CONFLICT OF INTEREST STATEMENT

Authors state no conflict of interest.

DATA AVAILABILITY

The data that support the findings of this study are openly available in [12], [59]–[61].

REFERENCES

[1] M. Grieves and J. Vickers, "Digital Twin: Mitigating Unpredictable, Undesirable Emergent Behavior in Complex Systems," in *Transdisciplinary Perspectives on Complex Systems*, Springer International Publishing, 2016, pp. 85–113, doi: 10.1007/978-3-319-38756-7_4.

[2] E. VanDerHorn and S. Mahadevan, "Digital Twin: Generalization, characterization and implementation," *Decision Support Systems*, vol. 145, p. 113524, Jun. 2021, doi: 10.1016/j.dss.2021.113524.

[3] M. A. Hamzaoui and N. Julien, "Social Cyber-Physical Systems and Digital Twins Networks: A perspective about the future digital twin ecosystems," *IFAC-PapersOnLine*, vol. 55, no. 8, pp. 31–36, 2022, doi: 10.1016/j.ifacol.2022.08.006.

[4] M. Liu, S. Fang, H. Dong, and C. Xu, "Review of digital twin about concepts, technologies, and industrial applications," *Journal of Manufacturing Systems*, vol. 58, pp. 346–361, Jan. 2021, doi: 10.1016/j.jmsy.2020.06.017.

[5] J. Collins *et al.*, "ABO: Dataset and Benchmarks for Real-World 3D Object Understanding," in *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2022, pp. 21094–21104, doi: 10.1109/cvpr52688.2022.02045.

[6] I. Špelic, "The current status on 3D scanning and CAD/CAM applications in textile research," *International Journal of Clothing Science and Technology*, vol. 32, no. 6, pp. 891–907, Oct. 2019, doi: 10.1108/ijcst-07-2018-0094.

[7] R. H. Helle and H. G. Lemu, "A case study on use of 3D scanning for reverse engineering and quality control," *Materials Today: Proceedings*, vol. 45, pp. 5255–5262, 2021, doi: 10.1016/j.matpr.2021.01.828.

[8] K. Son and K. Lee, "Effect of Tooth Types on the Accuracy of Dental 3D Scanners: An In Vitro Study," *Materials*, vol. 13, no. 7, pp. 1-13, Apr. 2020, doi: 10.3390/ma13071744.

[9] J. Liu *et al.*, "PlaneMVS: 3D Plane Reconstruction from Multi-View Stereo," in *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2022, pp. 8655–8665, doi: 10.1109/cvpr52688.2022.00847.

[10] D. Cernica, I. Benedek, S. Polexa, C. Tolescu, and T. Benedek, "3D Printing—A Cutting Edge Technology for Treating Post-Infarction Patients," *Life*, vol. 11, no. 9, pp. 1-14, Sep. 2021, doi: 10.3390/life11090910.

[11] Y. Guo, H. Wang, Q. Hu, H. Liu, L. Liu, and M. Bennamoun, "Deep Learning for 3D Point Clouds: A Survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 12, pp. 4338–4364, Dec. 2021, doi: 10.1109/tpami.2020.3005434.

[12] J. Chibane and A. Mir, "Neural Unsigned Distance Fields for Implicit Function Learning," *arXiv*, 2020, doi: 10.48550/arXiv.2010.13938.

[13] R. Venkatesh *et al.*, "Deep Implicit Surface Point Prediction Networks," in *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, Oct. 2021, pp. 12633–12642, doi: 10.1109/iccv48922.2021.01242.

[14] X. Long *et al.*, "NeuralUDF: Learning Unsigned Distance Fields for Multi-View Reconstruction of Surfaces with Arbitrary Topologies," in *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2023, pp. 20834–20843, doi: 10.1109/cvpr52729.2023.01996.

[15] Z. Peng *et al.*, "RTG-SLAM: Real-time 3D Reconstruction at Scale using Gaussian Splatting," in *Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers*, Jul. 2024, pp. 1–11, doi: 10.1145/3641519.3657455.

[16] H. Shu *et al.*, "Seamless augmented reality integration in arthroscopy: a pipeline for articular reconstruction and guidance," *Healthcare Technology Letters*, vol. 12, no. 1, Jan. 2025, doi: 10.1049/htl2.12119.

[17] B. Rodriguez-Garcia, J. M. Ramírez-Sanz, I. Miguel-Alonso, and A. Bustillo, "Enhancing Learning of 3D Model Unwrapping through Virtual Reality Serious Game: Design and Usability Validation," *Electronics*, vol. 13, no. 10, pp. 1-21, May 2024, doi:

10.3390/electronics13101972.
[18] B. R. Barricelli, E. Casiraghi, J. Gliozzo, A. Petrini, and S. Valtolina, "Human Digital Twin for Fitness Management," *IEEE Access*, vol. 8, pp. 26637–26664, 2020, doi: 10.1109/access.2020.2971576.
[19] W. Shengli, "Is Human Digital Twin possible?," *Computer Methods and Programs in Biomedicine Update*, vol. 1, 2024, doi: 10.1016/j.cmpbup.2021.100014.
[20] X. Li, J. Cao, Z. Liu, and X. Luo, "Sustainable Business Model Based on Digital Twin Platform Network: The Inspiration from Haier's Case Study in China," *Sustainability*, vol. 12, no. 3, pp. 1-26, Jan. 2020, doi: 10.3390/su12030936.
[21] A. R. Al-Ali, R. Gupta, T. Z. Batool, T. Landolsi, F. Aloul, and A. Al Nabulsi, "Digital Twin Conceptual Model within the Context of Internet of Things," *Future Internet*, vol. 12, no. 10, pp. 1-15, Sep. 2020, doi: 10.3390/fi12100163.
[22] M. Štroner, T. Křemen, and R. Urban, "Progressive Dilution of Point Clouds Considering the Local Relief for Creation and Storage of Digital Twins of Cultural Heritage," *Applied Sciences*, vol. 12, no. 22, pp. 1-18, Nov. 2022, doi: 10.3390/app122211540.
[23] F. Niccolucci, A. Felicetti, and S. Hermon, "Populating the Data Space for Cultural Heritage with Heritage Digital Twins," *Data*, vol. 7, no. 8, pp. 1-28, Jul. 2022, doi: 10.3390/data7080105.
[24] Z. Lv, W.-L. Shang, and M. Guizani, "Impact of Digital Twins and Metaverse on Cities: History, Current Situation, and Application Perspectives," *Applied Sciences*, vol. 12, no. 24, pp. 1-22, Dec. 2022, doi: 10.3390/app122412820.
[25] S. Ashraf, "A proactive role of IoT devices in building smart cities," *Internet of Things and Cyber-Physical Systems*, vol. 1, pp. 8–13, 2021, doi: 10.1016/j.iotcps.2021.08.001.
[26] Z. Lv, L. Qiao, Y. Li, Y. Yuan, and F.-Y. Wang, "BlockNet: Beyond reliable spatial Digital Twins to Parallel Metaverse," *Patterns*, vol. 3, no. 5, pp. 1-9, May 2022, doi: 10.1016/j.patter.2022.100468.
[27] M. Song, Q. Shi, Q. Hu, Z. You, and L. Chen, "On the Architecture and Key Technology for Digital Twin Oriented to Equipment Battle Damage Test Assessment," *Electronics*, vol. 12, no. 1, pp. 1-25, Dec. 2022, doi: 10.3390/electronics12010128.
[28] Y. M. Tang, K. M. Au, H. C. W. Lau, G. T. S. Ho, and C. H. Wu, "Evaluating the effectiveness of learning design with mixed reality (MR) in higher education," *Virtual Reality*, vol. 24, no. 4, pp. 797–807, Feb. 2020, doi: 10.1007/s10055-020-00427-9.
[29] M. Livesu, S. Ellero, J. Martínez, S. Lefebvre, and M. Attene, "From 3D models to 3D prints: an overview of the processing pipeline," *Computer Graphics Forum*, vol. 36, no. 2, pp. 537–564, May 2017, doi: 10.1111/cgf.13147.
[30] D. Fritsch and M. Klein, "3D and 4D modeling for AR and VR app developments," in *2017 23rd International Conference on Virtual System &amp; Multimedia (VSMM)*, Oct. 2017, pp. 1–8, doi: 10.1109/vsmm.2017.8346270.
[31] I. Garcia-Dorado, D. G. Aliaga, S. Bhalachandran, P. Schmid, and D. Niyogi, "Fast Weather Simulation for Inverse Procedural Design of 3D Urban Models," *ACM Transactions on Graphics*, vol. 36, no. 2, pp. 1–19, Apr. 2017, doi: 10.1145/2999534.
[32] M. Kazhdan and H. Hoppe, "Screened poisson surface reconstruction," *ACM Transactions on Graphics*, vol. 32, no. 3, pp. 1–13, Jun. 2013, doi: 10.1145/2487228.2487237.
[33] R. Chabra *et al.*, "Deep Local Shapes: Learning Local SDF Priors for Detailed 3D Reconstruction," in *Computer Vision – ECCV 2020*, Springer International Publishing, 2020, pp. 608–625, doi: 10.1007/978-3-030-58526-6_36.
[34] T. Li, X. Wen, Y.-S. Liu, H. Su, and Z. Han, "Learning Deep Implicit Functions for 3D Shapes with Dynamic Code Clouds," in *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2022, pp. 12830–12840, doi: 10.1109/cvpr52688.2022.01250.
[35] J. Ortiz *et al.*, "iSDF: Real-Time Neural Signed Distance Fields for Robot Perception," *Robotics: Science and Systems*, Jun. 2022, doi: 10.15607/rss.2022.xviii.012.
[36] J. Shim, C. Kang, and K. Joo, "Diffusion-Based Signed Distance Fields for 3D Shape Generation," in *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2023, pp. 20887–20897, doi: 10.1109/cvpr52729.2023.02001.
[37] M. M. Johari, C. Carta, and F. Fleuret, "ESLAM: Efficient Dense SLAM System Based on Hybrid Representation of Signed Distance Fields," in *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2023, pp. 17408–17419, doi: 10.1109/cvpr52729.2023.01670.
[38] A. Millane *et al.*, "nvblox: GPU-Accelerated Incremental Signed Distance Field Mapping," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*, May 2024, pp. 2698–2705, doi: 10.1109/icra57147.2024.10611532.
[39] M. Atzmon and Y. Lipman, "SAL: Sign Agnostic Learning of Shapes From Raw Data," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2020, pp. 2562–2571, doi: 10.1109/cvpr42600.2020.00264.
[40] Y. L. M. Atzmon, "SALD: Sign Agnostic Learning with Derivatives," *arXiv*, 2020, doi: 10.48550/arXiv.2006.05400.
[41] Y. L. A. Gropp, L. Yariv, N. Haim, and M. Atzmon, "Implicit Geometric Regularization for Learning Shapes," *arXiv*, 2020, doi: 10.48550/arXiv.2002.10099.
[42] M. Z. B. Ma, Z. Han, and Y.-S. Liu, "Neural-pull: Learning signed distance functions from point clouds by learning to pull space onto surfaces," *arXiv*, 2021, doi: 10.48550/arXiv.2011.13495.
[43] R. Sharma and P. Abrol, "Parameter Extraction and Performance Analysis of 3D Surface Reconstruction Techniques," *International Journal of Advanced Computer Science and Applications*, vol. 14, no. 1, 2023, doi: 10.14569/ijacsa.2023.0140135.
[44] C.-H. Lin, C. Kong, and S. Lucey, "Learning Efficient Point Cloud Generation for Dense 3D Object Reconstruction," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, Apr. 2018, doi: 10.1609/aaai.v32i1.12278.
[45] X. Yang, G. Lin, Z. Chen, and L. Zhou, "Neural Vector Fields: Implicit Representation by Explicit Learning," in *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2023, pp. 16727–16738, doi: 10.1109/cvpr52729.2023.01605.
[46] J. Zhou, B. Ma, S. Li, Y.-S. Liu, Y. Fang, and Z. Han, "CAP-UDF: Learning Unsigned Distance Functions Progressively From Raw Point Clouds With Consistency-Aware Field Optimization," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 46, no. 12, pp. 7475–7492, Dec. 2024, doi: 10.1109/tpami.2024.3392364.
[47] F. Zhao, W. Wang, S. Liao, and L. Shao, "Learning Anchored Unsigned Distance Functions with Gradient Direction Alignment for Single-view Garment Reconstruction," Oct. 2021, doi: 10.1109/iccv48922.2021.01244.
[48] B. Guillard, F. Stella, and P. Fua, "MeshUDF: Fast and Differentiable Meshing of Unsigned Distance Field Networks," in *Computer Vision – ECCV 2022*, Springer Nature Switzerland, 2022, pp. 576–592, doi: 10.1007/978-3-031-20062-5_33.
[49] J. Zhou, B. Ma, S. Li, Y.-S. Liu, and Z. Han, "Learning a More Continuous Zero Level Set in Unsigned Distance Fields through Level Set Projection," in *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, Oct. 2023, pp. 3158–3169, doi: 10.1109/iccv51070.2023.00295.
[50] Y. M. Tang and H. L. Ho, "3D Modeling and Computer Graphics in Virtual Reality," in *Mixed Reality and Three-Dimensional Computer Graphics*, IntechOpen, 2020, doi: 10.5772/intechopen.91443.
[51] L. Van Holland, P. Stotko, S. Krumpen, R. Klein, and M. Weinmann, "Efficient 3D Reconstruction, Streaming and Visualization

of Static and Dynamic Scene Parts for Multi-client Live-telepresence in Large-scale Environments," in *2023 IEEE/CVF International Conference on Computer Vision Workshops (ICCVW)*, Oct. 2023, pp. 4260–4274, doi: 10.1109/iccvw60793.2023.00460.

[52] S. Sangamuang, N. Wongwan, K. Intawong, S. Khanchai, and K. Puritat, "Gamification in Virtual Reality Museums: Effects on Hedonic and Eudaimonic Experiences in Cultural Heritage Learning," *Informatics*, vol. 12, no. 1, pp. 1-23, Mar. 2025, doi: 10.3390/informatics12010027.

[53] S. Münster *et al.*, "A Digital 4D Information System on the World Scale: Research Challenges, Approaches, and Preliminary Results," *Applied Sciences*, vol. 14, no. 5, pp. 1-48, Feb. 2024, doi: 10.3390/app14051992.

[54] T. Cerquitelli, M. Meo, M. Curado, L. Skorin-Kapov, and E. E. Tsiropoulou, "Machine learning empowered computer networks," *Computer Networks*, vol. 230, p. 109807, Jul. 2023, doi: 10.1016/j.comnet.2023.109807.

[55] S. Doungtap, J. Petchhan, V. Phanichraksaphong, and J.-H. Wang, "Towards Digital Twins of 3D Reconstructed Apparel Models with an End-to-End Mobile Visualization," *Applied Sciences*, vol. 13, no. 15, pp. 1-18, Jul. 2023, doi: 10.3390/app13158571.

[56] S. Klingenberg, R. Fischer, I. Zettler, and G. Makransky, "Facilitating learning in immersive virtual reality: Segmentation, summarizing, both or none?," *Journal of Computer Assisted Learning*, vol. 39, no. 1, pp. 218–230, Sep. 2022, doi: 10.1111/jcal.12741.

[57] Y. Ghasemi, H. Jeong, S. H. Choi, K.-B. Park, and J. Y. Lee, "Deep learning-based object detection in augmented reality: A systematic review," *Computers in Industry*, vol. 139, p. 103661, Aug. 2022, doi: 10.1016/j.compind.2022.103661.

[58] L. Regenwetter, A. H. Nobari, and F. Ahmed, "Deep Generative Models in Engineering Design: A Review," *Journal of Mechanical Design*, vol. 144, no. 7, Mar. 2022, doi: 10.1115/1.4053859.

[59] A. X. Chang *et al.*, "Shapenet: An information-rich 3d model repository," *arXiv*, 2015, doi: 10.48550/arXiv.1512.03012.

[60] Q.-Y. Zhou and V. Koltun, "Dense scene reconstruction with points of interest," *ACM Transactions on Graphics*, vol. 32, no. 4, pp. 1–8, Jul. 2013, doi: 10.1145/2461912.2461919.

[61] H. Zhu *et al.*, "Deep Fashion3D: A Dataset and Benchmark for 3D Garment Reconstruction from Single Images," in *European Conference on Computer Vision*, 2020, pp. 512–530, doi: 10.1007/978-3-030-58452-8_30.

BIOGRAPHIES OF AUTHORS

Jirayu Petchhan received the B.E. (Hons. I) degree in Control Engineering and M.E. degree in Instrumentation Engineering from the School of Engineering, King Mongkut's Institute of Technology Ladkrabang, Thailand, in 2017 and 2019, respectively, and the Ph.D. degree in Electrical Engineering with the College of Electrical Engineering and Computer Science, National Taiwan University of Science and Technology, Taipei City, Taiwan, R.O.C. in 2024. He was a system integration engineer with Contrologic Company Limited, Thailand, for two years, and he is currently a lecturer with the department of Computer Engineering at King Mongkut's Institute of technology Ladkrabang. His research interests include machine learning, transfer learning, language models, and artificial intelligence of things. He can be contacted at email: jirayu.pe@kmitl.ac.th.

Surasachai Doungtap received B.Sc. in Computer Game Multimedia from Rungsit University and M.Sc. in Interaction Design & Innovation International Program from National Taipei University of Technology, in 2012 and 2020, respectively, and he is currently pursuing the Ph.D. degree in Computer Science and Information Engineering with International Graduate Program Electrical Engineer and Computer Science, National Taipei University of Technology, Taipei City, Taiwan, R.O.C. He possesses over a decade of experience as a 3D artist in the gaming, advertising, and animation sectors, having contributed to several projects, including AAA games, and received multiple accolades. His research interests include computer vision, computer graphics, UI/UX design, and human-computer interaction, especially, related to virtual/augmented/mixed reality. He can be contacted at email: t110999402@ntut.org.tw.