Scalable and Distributed Optimization of Shared 3D Object Quality for Large-Scale Hybrid-Metaverses

Yui Maruyama
Osaka University
Osaka, Japan
y-maruyama@ist.osaka-u.ac.jp

Tatsuya Amano
Osaka University
Osaka, Japan
t-amano@ist.osaka-u.ac.jp

Hirozumi Yamaguchi
Osaka University
Osaka, Japan
h-yamagu@ist.osaka-u.ac.jp

***Abstract*—Hybrid-metaverses, integrating physical and virtual spaces, face a critical challenge in managing shared 3D object quality across multiple users with diverse preferences and limited network resources. This paper addresses the problem of allocating limited bandwidth for transmitting point cloud representations while maximizing overall user satisfaction. We propose a distributed optimization method that dynamically adjusts 3D object quality based on contextual importance, available resources, and user preferences. Our approach uses Input Convex Neural Networks (ICNN) to model user utility functions and employs the Alternating Direction Method of Multipliers (ADMM) for distributed optimization. Key advantages include scalability, adaptability, and improved quality of experience. Evaluation using open dataset demonstrates significant improvements in user satisfaction and resource utilization compared to baseline approaches. Our method achieves 93-94.6% accuracy in modeling user utility and shows up to 60% faster convergence for scenarios with 30 users, contributing to the balance between high-fidelity representation and efficient data management in hybrid-metaverses.**

***Index Terms*—3D point cloud, hybrid-metaverse, distributed optimization, Input Convex Neural Networks**

I. Introduction

The Metaverse, an immersive virtual shared space, has increasingly served as a critical technology for remote communication, bridging the social distance caused by the pandemic and others. Platforms like cluster [1] utilize cutting-edge VR/AR/XR technologies to create immersive virtual environments that closely resemble physical spaces.

Recent advancements in 3D sensing technologies have introduced a new concept: the integration of real-world spaces into virtual environments through real-time sensing [2]–[5]. This approach, which we term a "hybrid-metaverse," transcends traditional avatar-based communication by capturing and incorporating physical living spaces into the metaverse in real-time. For instance, Physically Grounded Metaverse Applications (PMAs) propose metaverse environments based on spaces dynamically scanned by LiDAR sensors [4]. Amano et al. [3] further explore this concept, introducing an extended metaverse where high-resolution 3D point cloud from users' physical environments are captured and integrated into shared spaces in real-time, enabling more immersive hybrid experiences. This hybrid approach enables more immersive experiences in collaborative applications, blurring the line between physical and virtual realities and opening new possibilities for remote collaboration, education, and social interaction.

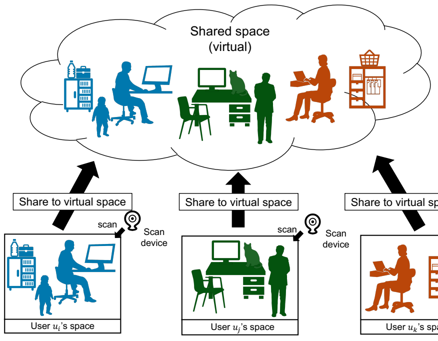


Fig. 1: The emerging concept of hybrid-metaverse

In these hybrid-metaverses, users' physical spaces are continuously captured and digitized in real-time. Typically, this is achieved using multiple fixed RGB-D cameras or LiDAR sensors to scan the environment, generating detailed 3D point cloud data. A crucial aspect of these systems is their ability to process this data at the object level. Through real-time semantic segmentation and object tracking, the captured point cloud is divided into separate streams for individual objects - such as people, furniture, or handheld items. This object-based approach allows for more efficient data management and transmission, as well as finer control over the quality of different elements in the shared virtual space. However, the sheer volume of this object-level 3D data and the need for real-time updates still present significant challenges, especially when dealing with multiple users and diverse objects in a bandwidth-constrained network environment.

The core challenge in hybrid-metaverse systems is optimizing the quality of shared 3D objects while operating within network bandwidth constraints. This optimization aims to maximize the overall utility of all users, which we define in this paper as the degree to which users can share their physical space as desired with others. Ideally, all objects would be shared at maximum quality, but network limitations make this impossible. The complexity of this task arises from several factors. Users have different priorities for object quality in the shared virtual space, and these priorities change depending on the context. For example, in a remote exercise class, the quality and frame rate of human avatars might be more important than the resolution of background objects. In contrast, a virtual cooking class might require high-resolution representations of cooking utensils and ingredients. The importance of different objects also varies based on the current activity within the metaverse. During a virtual tour, the quality of architectural details might be crucial, while a collaborative design session would prioritize the fidelity of 3D models being manipulated. This context-dependent nature of user utility is difficult to capture with simple, linear models. Moreover, the available bandwidth between users is often limited and may fluctuate, necessitating dynamic adjustments to 3D object quality to maintain a satisfactory experience for all users. Ultimately, user utility is determined by how well they can share their space according to their preferences in each specific context, given these context-specific quality parameters and network constraints.

To address these issues, we propose an approach that integrates machine learning with distributed optimization techniques. Our strategy employs Input Convex Neural Networks (ICNNs) to accurately model the utility functions of users. These neural networks are adept at grasping the intricate, non-linear dynamics between the quality of objects, the context of their use, and user satisfaction levels. To accommodate the demands of large-scale metaverse environments, where centralized optimization is not feasible, we utilize a distributed optimization framework using the Alternating Direction Method of Multipliers (ADMM). This method enables individual users to optimize their personal utility functions, ensuring that each participant can adjust the quality of the virtual objects they interact with according to their own requirements and the current bandwidth availability. Simultaneously, the system maintains a balance, ensuring that the collective adjustments adhere to the optimal network performance criteria. This setup not only enhances user satisfaction but also promotes fairness and efficiency in resource usage, particularly evident during large virtual gatherings where varied object quality needs are dynamically managed.

To validate our approach, we conducted extensive experiments using three distinct physical space scenes constructed from the SUN RGB-D dataset, simulating both online meeting and party contexts. Our evaluation demonstrates that the PICNN effectively models complex user utility functions while maintaining the convexity required for ADMM optimization, with 93-94.6% of test data falling within the 95% confidence interval across different scenes. The distributed method shows significant advantages over centralized approaches in terms of scalability and efficiency. As the number of users increases, our method exhibits better convergence speed and lower communication overhead. Notably, for scenarios with 12 or more users, our distributed approach consistently achieves faster convergence than the centralized method, with up to 60% reduction in convergence time for 30 users. These results highlight the effectiveness of our approach in managing shared object quality for large-scale hybrid metaverse applications, offering improved scalability and resource utilization while maintaining high accuracy in representing user satisfaction.

II. Related Work

Technologies for smooth information sharing and communication in remote spaces are rapidly advancing. In VR, techniques such as 360-degree field of view expansion and multi-user spatial sharing have been proposed and implemented. For example, in reference [6], communication technologies for scalable 360-degree video, such as video tiling and viewpoint-adaptive resource allocation, are provided.

Various studies have been conducted on methods for transferring 3D volumetric streaming data, such as point clouds, over networks [7], [8]. According to an ITU-T report summarizing new services and requirements for networks in 2030 [9], it is anticipated that more immersive and rich interactive services, such as holographic communication, tele-haptics, and volumetric streaming, will become widespread. The requirements of these services, such as bandwidth and latency, suggest that addressing them solely through end-to-end or overlay approaches is challenging, and integration with in-network functions is necessary. Volumetric streaming is generally intended for reception by MR or VR devices, enabling users on the receiving end to stream 3D video with six degrees of freedom (6DoF). In particular, point cloud compression and streaming are areas of active interest, with standardization efforts being led by ISO/IEC MPEG [10]. Existing research on point cloud optimization and encoding often takes the approach of controlling the quality of point clouds based on the viewport of the receiver, bandwidth, and client buffering [8], [11].

A notable example of hybrid metaverse research is the work by Amano et al. [3]. They propose an extended metaverse environment that seamlessly blends physical and virtual spaces. Their system utilizes high-resolution 3D sensing technologies to capture users' physical environments in real-time and integrate them into shared virtual spaces.

A key feature of their approach is the focus on visual privacy control, allowing users to manage privacy at an object level within the shared space. This research is closely related to our work on 3D object quality optimization, as it addresses the critical balance between user satisfaction and privacy protection in hybrid metaverse environments. Both studies contribute to the development of more immersive and secure collaborative virtual experiences.

III. System Overview and Problem Formulation

A. System Overview

The hybrid-metaverse system we propose in this paper aims to seamlessly integrate multiple physical spaces into a shared virtual environment, enabling rich, immersive interactions between remote users. This system leverages advanced 3D sensing technologies and real-time data processing to create a dynamic, object-based representation of each user's physical surroundings within the shared virtual space.

At its core, our hybrid-metaverse system consists of three main components: 1) real-time 3D capture and segmentation of physical spaces, 2) efficient data transmission and management across the network, and 3) adaptive quality control for shared objects. These components work together to create a flexible, scalable platform that can accommodate multiple users while optimizing the use of available network resources.

Each user's physical space is continuously scanned by fixed RGB-D cameras, generating a detailed 3D point cloud representation of the user and their surroundings. This raw data undergoes real-time semantic segmentation and tracking on the transmission side, effectively dividing the captured environment into discrete object-based point cloud streams.

This object-level approach to data management offers several advantages. It allows for selective sharing of objects within the metaverse, giving users control over which elements of their physical space they wish to make visible to others. Furthermore, it enables fine-grained control over the quality parameters of each shared object. These parameters include point cloud resolution, transmission frame rate, and compression method, all of which can be dynamically adjusted based on factors such as user preferences, current context, and available network bandwidth.

In multi-user scenarios, the system employs a hybrid architecture for data transmission and management. A central server aggregates and manages object attribute information, including object types and their spatial placement within the shared virtual space. However, to reduce server load and minimize latency, the actual point cloud streams are transmitted directly between users in a peer-to-peer fashion. This approach strikes a balance between centralized coordination and distributed data transfer.

On the receiving end, users obtain both the point cloud streams and the associated attribute and placement information. Their local systems then reconstruct these objects within the shared metaverse space, creating a cohesive virtual environment that integrates elements from multiple physical locations.

A critical aspect of this system is its need for adaptive quality management. Given the substantial data rates associated with high-quality point cloud transmission, the sending side must continuously adjust the quality of each shared object to adhere to the capacity constraints of the inter-user communication channels. This dynamic quality control is essential for maintaining system performance and ensuring a smooth user experience within the bandwidth limitations of the network.

B. Problem Formulation

Figure 2 illustrates the object quality optimization problem in our hybrid-metaverse system.

In the formulation, each physical space contains a single user. We denote the total number of users as N (≥ 1), with the set of users represented as $U = \{u_1, ..., u_i, ..., u_N\}$. Each user u_i shares M_i (≥ 1) objects in the metaverse, forming the set $O_i = \{o_{i,1}, ..., o_{i,l}, ..., o_{i,M_i}\}$. These objects are visible to all other users in the shared virtual environment.

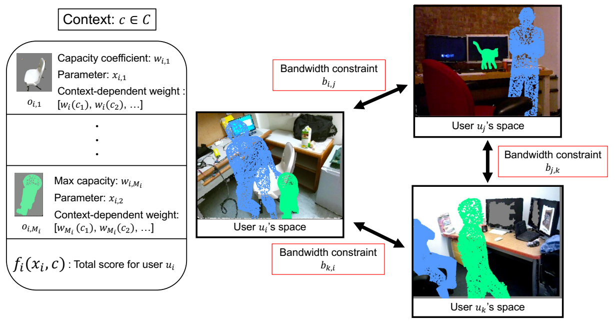

Context: $c \in C$
Capacity coefficient: $w_{i,1}$
Parameter: $x_{i,1}$
Context-dependent weight : $[w_i(c_1), w_i(c_2), ...]$
$o_{i,1}$
Max capacity: w_{i,M_i}
Parameter: $x_{i,2}$
Context-dependent weight: $[w_{M_i}(c_1), w_{M_i}(c_2), ...]$
o_{i,M_i}
$f_i(x_i, c)$: Total score for user u_i
Bandwidth constraint $b_{i,j}$
User u_i's space
User u_j's space
Bandwidth constraint $b_{j,k}$
Bandwidth constraint $b_{k,i}$
User u_k's space

Fig. 2: Problem Formulation

The quality of each shared object $o_{i,l}$ is represented by $x_{i,l}$, with x_i denoting the quality vector for all objects shared by user u_i. While in practice, object quality is determined by multiple parameters such as point cloud resolution and transmission frame rate, we simplify this to a single parameter $x_{i,l} \in [0, 1]$ for our model. The communication capacity required to share $o_{i,l}$ at quality $x_{i,l}$ is given by $x_{i,l} \cdot w_{i,l}$, where $w_{i,l}$ is a predefined capacity coefficient for each object. We assume bidirectional communication channels between users, with the capacity limit between users u_i and u_j denoted by $b_{i,j}$.

The user utility function $f_i(x_i, c)$ quantifies the satisfaction that user u_i derives from the quality of shared objects $x_i = (x_{i,1}, ..., x_{i,M_i})$, where $c \in C$ represents the current context or communication scenario from a set of predefined contexts $C = \{c_1, c_2, ..., c_K\}$. Each element of C corresponds to a specific type of virtual interaction or activity. For instance, c_1 might represent an "online meeting" context, and c_2 a "virtual social gathering" context. This context variable allows the utility function to adapt to different situations, where the importance of different objects may vary significantly. For example, in an online meeting context (c_1), high-quality representation of participants' faces and any shared documents might be prioritized. In contrast, in a virtual social gathering context (c_2), the quality of full-body avatars and surrounding environmental objects might take precedence. Importantly, even when the same physical objects are shared across different time, their contribution to user utility may change dramatically based on the current context. A higher value of $f_i(x_i, c)$ indicates greater user satisfaction within the given context.

The global optimization problem for our hybrid-metaverse system is formulated in Eqn. (1). Here, c is a constant parameter representing the fixed context, and the optimization is performed over the variables x_i for all users u_i. The objective is to maximize the average utility across all users, which we express as a minimization problem by inverting the sign of the utility scores.

$$\begin{aligned}
&\min \frac{1}{N}\sum_{i=1}^{N} -f_i(x_i,c) \\
&\text{subject to } \sum_{l=1}^{M_i} w_{i,l}\cdot x_{i,l} + \sum_{l=1}^{M_j} w_{j,l}\cdot x_{j,l} \le b_{i,j}, \\
&\qquad\qquad \forall i,j \in \{1,\ldots,N\}, i \ne j \\
&\quad 0 \le x_{i,l} \le 1, \\
&\qquad\qquad \forall i \in \{1,\ldots,N\}, \forall l \in \{1,\ldots,M_i\}.
\end{aligned} \tag{1}$$

While this problem could theoretically be solved using centralized optimization if all utility functions f_i were aggregated at a central server, such an approach is impractical in our context. The utility functions in our research require representation by neural networks with substantial model capacities. Furthermore, the dynamic nature of multi-user metaverse environments, with frequent user entries and exits, would necessitate constant updates to the centralized model.

Given these constraints, we propose a distributed optimization method that does not require the sharing of individual utility functions f_i. This approach is better suited to the scalable and dynamic nature of hybrid-metaverse environments.

IV. Proposed method

Our approach employs the Alternating Direction Method of Multipliers (ADMM) to solve the optimization problem defined in Eqn. (1) in a distributed manner, where each user's utility f_i is represented by a neural network. ADMM is an algorithm capable of solving constrained optimization problems by decomposing the objective function into separable parts, alternately optimizing each subproblem under global constraints, and sharing Lagrange multiplier information across subproblems to find the overall solution.

A. ADMM-based distributed optimization

For simplification, we express the capacity constraint between users u_i and u_j using $g_{i,j}(x_i,x_j)$, as defined in Eqn. (3). The context c is assumed to be consistent and fixed during a given optimization process. Therefore, for the sake of simplicity in this section, we denote the utility function $f_i(x_i,c)$ simply as $f_i(x_i)$, with the understanding that c remains constant throughout the optimization.

$$\begin{aligned}
&g_{i,j}(x_i,x_j) \le 0 \\
&\qquad \forall i,j \in \{1,\ldots,N\}, i \ne j
\end{aligned} \tag{2}$$

$$g_{i,j}(x_i,x_j) = \sum_{1\le l\le M_i} w_{i,l}x_{i,l} + \sum_{1\le l\le M_j} w_{j,l}x_{j,l} - b_{i,j} \tag{3}$$

If the constraint is satisfied, $g_{i,j}(x_i,x_j)$ takes a value of 0 or less, and if not satisfied, it takes a positive value.

In this context, the subproblem for user u_i is expressed by Eqn. (4). In the original problem (1), each variable $x_{i,l}$ is given independently according to each user's utility function. Therefore, the subproblem involves maximizing the local utility function f_i for user u_i under the capacity constraints of all communication channels between user u_i and other users. For this section, we assume that the utility functions f_i for each user have already been obtained.

$$\begin{aligned}
&\min\ -f_i(x_i) \\
&\text{subject to } g_{i,j}(x_i,x_j) \le 0, \\
&\qquad\qquad \forall j \in \{1,\ldots,N\}, i \ne j \\
&\quad 0 \le x_{i,l} \le 1, \\
&\qquad\qquad \forall i \in \{1,\ldots,N\}, \forall l \in \{1,\ldots,M_i\}.
\end{aligned} \tag{4}$$

To facilitate optimization, we introduce slack variables $s_{i,j}(\ge 0)$ for the inequality constraints related to $g_{i,j}(x_i,x_j)$ and convert them into equality constraints. The constraints defined in Eqn. (2) can be rewritten as follows:

$$\begin{aligned}
&h_{i,j}(x_i,x_j,s_{i,j}) = 0, \\
&\text{where,} \\
&h_{i,j}(x_i,x_j,s_{i,j}) = g_{i,j}(x_i,x_j) + s_{i,j} \\
&s_{i,j} \ge 0
\end{aligned} \tag{5}$$

The augmented Lagrangian function L for the system-wide optimization problem is given by Eqn. (6). Here, $\mu_{i,j}$ represents the Lagrangian multipliers for the capacity constraints between user u_i and u_j. Additionally, ρ is a constant penalty parameter used in the optimization process.

$$\begin{aligned}
L(x,\mu,s) = &\sum_{1\le i\le N} -f_i(x_i) \\
&+ \sum_{1\le i\le N-1}\sum_{i\le j\le N} \mu_{i,j} h_{i,j}(x_i,x_j,s_{i,j}) \\
&+ \sum_{1\le i\le N-1}\sum_{i\le j\le N} \frac{\rho}{2} h_{i,j}(x_i,x_j,s_{i,j})^2
\end{aligned} \tag{6}$$

Similarly, the augmented Lagrangian function L_i for the subproblem of user u_i is given by Eqn. (7).

$$\begin{aligned}
L_i(x,\mu_i,s_i) = &-f_i(x_i) \\
&+ \sum_{1\le j\le N, i\ne j} \mu_{i,j} h_{i,j}(x_i,x_j,s_{i,j}) \\
&+ \sum_{1\le j\le N, i\ne j} \frac{\rho}{2} h_{i,j}(x_i,x_k,s_{i,j})^2
\end{aligned} \tag{7}$$

In the ADMM framework, global optimization under the overall constraints is achieved by alternately minimizing the augmented Lagrangian function L_i for each subproblem and updating the slack variables and Lagrange multipliers. The update formulas for these parameters in this problem are given by Eqn. (8).

$$\begin{aligned}
x_i^{k+1} &= \text{argmin}_x(L_i(x^k,\mu_i^k,s_i^k)) \\
s_{i,j}^{k+1} &= \max(0, s_{i,j}^k - h_{i,j}(x_i^{k+1},x_j^{k+1},s_{i,j}^k) - \frac{\mu_{i,j}^k}{\rho}) \\
\mu_{i,j}^{k+1} &= \mu_{i,j}^k + h_{i,j}(x_i^{k+1},x_j^{k+1},s_{i,j}^{k+1})
\end{aligned} \tag{8}$$

Here, k represents the step number in the optimization process. The initial values for all parameters are assumed to be shared among all users. Each user performs the aforementioned steps individually, updating the parameters sequentially from top to bottom in the equations.

At each step, user u_i determines the optimal x_i by minimizing the augmented Lagrangian function, while keeping all parameters other than their own objective function variables x_i fixed. During this process, the constraints on x_i (i.e., $x_i \in [0, 1]$) are considered.

Under the optimal value x_i obtained, user u_i sends the value of $\sum_{1 \leq l \leq M_i} w_{i,l} x_{i,l}$ from Eqn. (3) to all other users. This represents the total communication capacity used by user u_i in that step. Similarly, user u_i calculates $g_{i,j}(x_i, x_j)$ and $h_{i,j}$ independently based on this information received from other users, and uses these to update the slack variables and Lagrangian multipliers according to Eqn. (8).

Based on the updated parameters, the next update step is performed. Through this independent optimization of subproblems and communication of constraint-related information between users, convergence to the solution of the overall optimization problem is achieved.

In the ADMM framework, it is known that if the objective function f_i in the subproblems is a convex function, convergence is achieved in a finite number of steps.

B. Utility Function Modeling with Input Convex Neural Networks

User satisfaction (utility) in shared spaces with varying object quality is complex and challenging to model simply. This study employs neural networks to represent the utility function f_i. For the ADMM algorithm to guarantee convergence, each subproblem's objective function must be convex. Therefore, we aim to model the user's utility as a convex function using a specialized neural network architecture. While a basic neural network can be made convex by using non-negative weights and convex activation functions, this approach significantly reduces the network's expressive power.

To overcome this limitation, we adopt the Partially Input Convex Neural Network (PICNN) [12]. The PICNN architecture, illustrated in Fig. 3, maintains high expressiveness while ensuring convexity with respect to specific inputs.

The PICNN takes inputs x and y, guaranteeing convexity only with respect to y. Each hidden layer k is divided into a convex part z_k and a non-convex part u_k. The network's output is defined by the following recursive equations:

$$\begin{aligned}
u_{i+1} =& \tilde{g}_i(\tilde{W} i u_i + \tilde{b} i) \\
z_{i+1} =& g_i(W_i^{(z)}(z_i \circ [W_i^{(zu)} u_i + b_i^{(z)}]+)+ \\
& W_i^{(y)}(y \circ (W_i^{(yu)} u_i + b_i^{(y)})) + W_i^{(u)} u_i + b_i) \\
f(x, y; \theta) =& z_k, u_0 = x
\end{aligned}$$

In this formulation, z_{i+1} comprises three components: the relationship between the non-convex part u_i and the convex part z_i, the interaction between the non-convex part u_i and

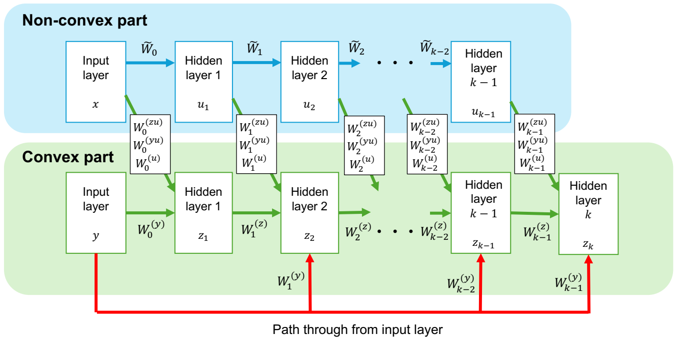


Fig. 3: Architecture of PICNN

the input y, and information solely from the non-convex part. The non-convex part u_i plays a crucial role in enhancing the expressive power of the network, allowing the PICNN to capture complex, non-convex relationships in the input data, which are then transformed into convex representations.

The PICNN architecture incorporates several key features. Only the weights $W_i^{(z)}$ in each layer are constrained to be non-negative, as only z_i needs to be a convex function for each i. The activation functions g_i for the convex parts are selected to be convex functions. Passthrough connections $W_i^{(y)}$ (for $i = 1, 2, ..., k-1$) apply weights directly to the network's input and pass it through to subsequent hidden layers, starting from the second hidden layer onwards. These passthrough connections help overcome the difficulty of training identity mappings in deep networks, especially under the non-negative weight constraint of PICNNs. Additionally, weight parameters $W_i^{(u)}$ transmit information from the non-convex parts of each layer to the convex parts. This combination of features allows the PICNN to maintain both expressiveness and the required convexity properties.

As a result, the output becomes a convex function of the input y, while generally remaining non-convex with respect to x. This partial convexity allows the network to model complex utility functions that may have non-convex dependencies on certain inputs, while still maintaining the convexity properties required for optimization with respect to the key variables.

In our application, the input x represents the context, while y represents the object quality parameters. The PICNN ensures that the output (user utility) is convex with respect to y for any given x. This property guarantees that, for any fixed context, the utility function is convex with respect to the quality parameters we aim to optimize.

By adopting the PICNN, we can accurately represent complex user preferences that may change significantly across different contexts, while ensuring that the optimization problem remains tractable within the ADMM framework. This approach provides a flexible and powerful way to model user satisfaction in dynamic, shared virtual environments, where the impact of object quality on user utility may vary greatly depending on the context, but the optimization process requires convexity with respect to these quality parameters.

V. Evaluation

A. Experimental Setup

To evaluate the effectiveness of our proposed method in large-scale hybrid metaverse scenarios, we constructed a comprehensive experimental environment using open-source 3D point cloud datasets. We built three distinct physical space scenes (Scene A to C) using point cloud data, as illustrated in Fig. 4. These scenes were constructed from the SUN RGB-D dataset [13]–[15], which provides both point cloud data and bounding box information for object segmentation.

We considered two distinct contexts for each scene in our hybrid metaverse setting: online meetings and online parties. For each context, we comprehensively set the quality parameters (resolution and frame rate) for each object shared by the users. To determine the true user utility, we employed a multi-step process that accurately reflects the user's perception of shared objects in the virtual space.

For each shared object, we created a video sequence from the user's perspective in the virtual space, showing the object at various quality levels with different resolutions and frame rates. We then utilized VMAF (Video Multimethod Assessment Fusion), a perceptual video quality assessment algorithm developed by Netflix, to evaluate these video sequences. VMAF provides a score that closely approximates human perception of video quality, making it particularly suitable for our hybrid metaverse scenario.

The true user utility is defined as the weighted sum of VMAF scores for all objects shared by that user. We predefined the weights for each object according to the context, reflecting the varying importance of different objects in various scenarios. This approach allows us to capture the nuanced differences in object importance between, for example, an online meeting and an online party.

Mathematically, we express the utility function for a user u in context c as:

$$U(u,c) = \sum_{i \in O_u} w_i(c) \cdot \mathrm{VMAF}(V_i(r_i, f_i))$$

Where N is the number of objects in the scene, $w_i(c)$ is the context-dependent weight for object i, r_i is the downsampling rate of object i, f_i is the frame rate of object i (for moving objects), and $V(r_i, f_i)$ is the Video Multimethod Assessment Fusion score. In this equation, O_u represents the set of objects shared by user u, $w_i(c)$ is the predefined context-dependent weight for object i, $V_i(r_i, f_i)$ is the video sequence of object i as seen from the user's perspective with resolution r_i and frame rate f_i, and $\mathrm{VMAF}(V_i(r_i, f_i))$ is the VMAF score for this video sequence.

The context-dependent weights play a crucial role in representing the relative importance of objects in different scenarios. Table I shows the weights assigned to various objects in our scenes for both contexts. For instance, in an online meeting context, a laptop might have a higher weight compared to its weight in an online party context, reflecting its increased importance in a professional setting.

As shown in Tab. I, objects like "person_2" have significantly higher weights in the online party context (0.8) compared to the online meeting context (0.1). Conversely, the "laptop" has a higher weight in the online meeting context (0.6) than in the party context (0.3).

To simulate a more realistic environment, we introduced two moving objects in each scene. One represents the space owner, while the other simulates a potentially distracting element, such as another person or a pet. For these dynamic objects, both downsampling rate and frame rate affect their VMAF scores and, consequently, the overall utility. This comprehensive setup allows us to realistically simulate diverse metaverse environments and user scenarios, accurately model user satisfaction across various object qualities and contexts, and validate the scalability and adaptability of our distributed optimization approach. In the following sections, we will describe how we used this experimental setup to train our PICNN model and evaluate our proposed distributed optimization method.

B. Training Utility Function with PICNN

We train each scene's user utility function as a separate PICNN. Each model takes two types of inputs: the context (1 for meeting or 2 for party) as the non-convex input x, and all object quality parameters (resolution and frame rate) as the convex input y. While frame rate parameters exist for both moving and stationary objects, they only affect utility for moving objects.

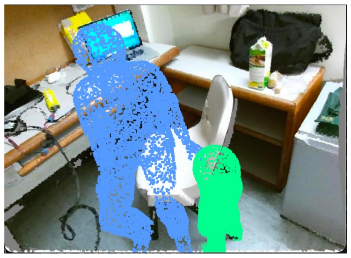
For each scene, we generated a dataset of 10,000 training and 2,000 test samples by randomly sampling context and object parameters, then calculating their corresponding utility scores. All object parameters were normalized to [0, 1]. The PICNN architecture was augmented with an output layer to convert the final convex layer output z_k into a single utility score. The hyper-parameters in the training are shown in Tab.II. The activation function used is ReLU, and the optimizer used is Adam. With this setup, training on Mac Book Air with Apple M1 CPU and 24 GB Memory, took approximately 15 minutes.

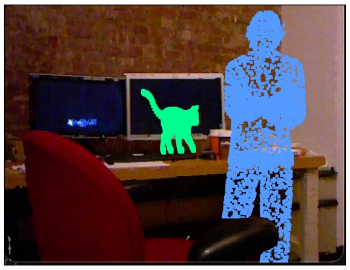
To solve each optimization step in our proposed method, we leverage PyTorch's automatic differentiation to minimize the Lagrangian function containing the neural network, updating parameters through the Adam optimizer.

C. Performance of Trained Utility Function with PICNN

We evaluate how accurately the PICNN was able to learn the utility function. Figure 5 shows the accuracy on the test data for each of the three scene models. The x-axis of each point represents the true utility, while the y-axis represents the utility predicted by the model. The red line represents the theoretical value with the line $y = x$, and the green lines indicate the lower and upper bounds of the 95% confidence interval, respectively.

Table III shows the proportion of data within the 95% confidence interval for each scene. From these results, it can be seen that the model has been able to learn the utility reasonably


(a) Scene A


(b) Scene B

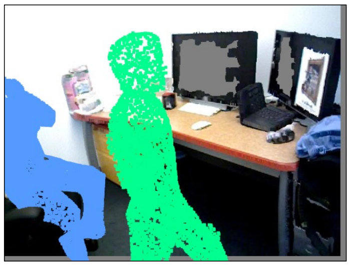

(c) Scene 3

Fig. 4: Physical Scenes for the Evaluation

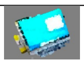
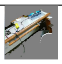
Object name	Image	Max capacity (KB)	Weight for context 1	Weight for context 2
bag		926.20	0.1	0.1
fridge		1179.61	0.1	0.1
laptop		433.92	0.6	0.3
shelf		2247.28	0.2	0.2
table		3030.14	0.4	0.4
chair		1376.95	0.7	0.7
person_2		436.31	0.1	0.8
person_1		457.59	1.0	1.0
bulletin_board		1170.47	0.1	0.2
bag_2		115.17	0.1	0.1
plastic_container		365.48	0.1	0.3

TABLE I: Object table with image, max capacity, and weights for two contexts

name	value
learning rate	0.01
number of epoch	1000
batch size (training data)	128
batch size (test data)	64
number of node (z_i)	100
number of node (u_i)	10
number of hidden_layers (k)	2

TABLE II: Hyperparameter for learning PICNN

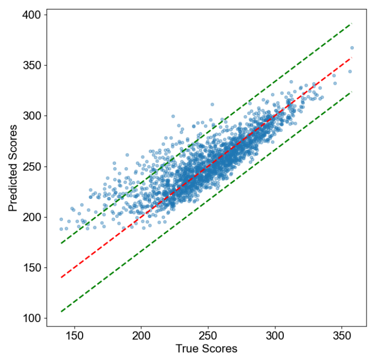
well, despite the constraint that it must be a convex function with respect to the parameters.

D. Communication cost of proposed method

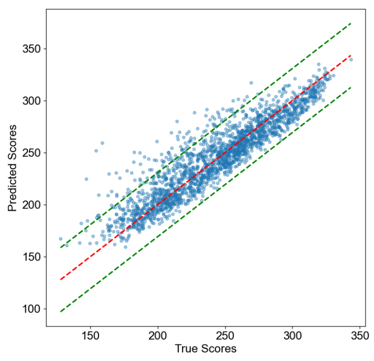
We evaluate the communication cost of the proposed method in comparison with the centralized method. In the proposed method, information about the parameters of other users is required to compute the constraints, and this information needs to be exchanged with other users at each step.

On the other hand, in the centralized optimization method, all users must send their models and constraint conditions (such as the capacity of communication channels with other users) to the server. After all the information has been transmitted to the server, there is no further data to send. Under this assumption, we evaluate the total communication cost of the system for both the centralized method and the proposed method. As an additional assumption, we consider that all numerical values are 32-bit (i.e., 4 bytes), and the size of the model is based on the file size used to store the model.

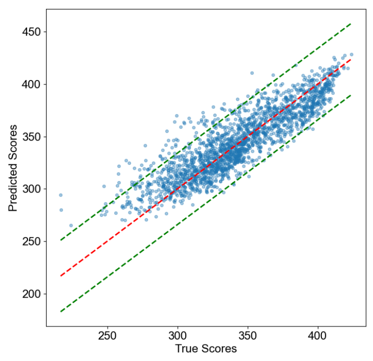
Figure 6 shows the communication overhead until convergence for both the centralized method and the proposed method, with the number of users ranging from 5 to 30. The convergence condition is the same as that described later in Fig. 8. In the centralized method, the communication overhead increases linearly as the number of users grows, since only the utility neural network model is shared. On the other hand, in the proposed method, communication occurs between pairs of



(a) Scene A



(b) Scene B



(c) Scene C

Fig. 5: Performance of PICNN

scene	scene A	scene B	scene C
rate(%)	93.0	94.1	94.6

TABLE III: Percentage of evaluation data points sampled from the true utility function that fall within the 95% confidence interval, as assessed using the trained PICNN model for each scene.

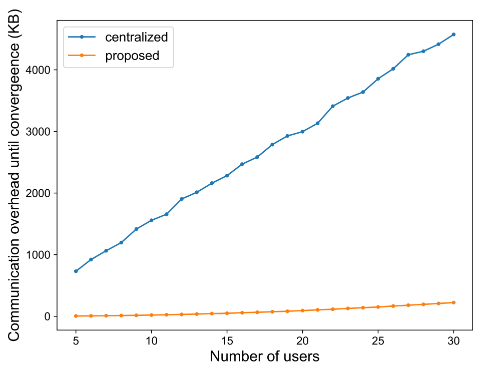


Fig. 6: Comparison of communication overhead until convergence between the centralized algorithm and the proposed method. While the centralized method requires transmission of large neural network models from each user to the server, the proposed method only exchanges small parameter values between users, resulting in lower total communication overhead for the shown range of users (5-30).

users at each step, leading to an increase in communication overhead proportional to the square of the number of users. However, because the communication per step is minimal, the proposed method demonstrates superior performance in terms of communication overhead compared to the centralized method within the range shown in Figure 6.

E. Execution time of proposed method

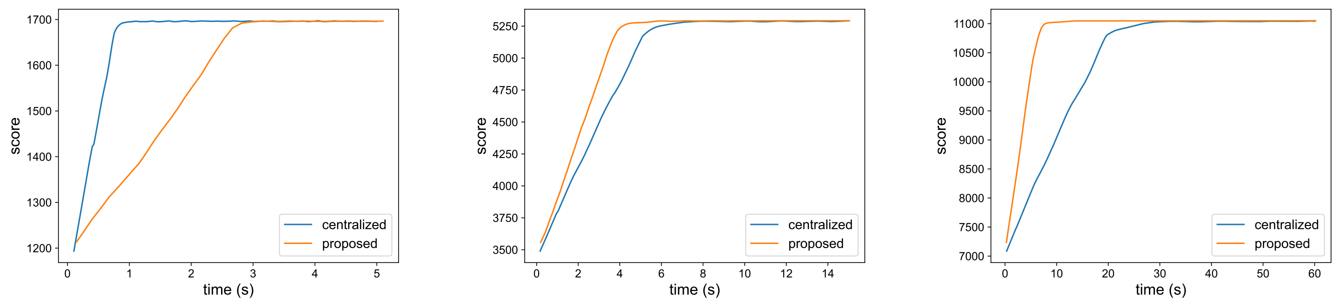
We compared our proposed distributed method against a centralized baseline that simultaneously optimizes parameters for all uesrs by solving Eqn. (1). The centralized method uses the same extended Lagrangian function with Lagrange multipliers μ and slack variables s, following the parameter update procedure in Eqn. (8).

Figure 7 shows the transition of scores for the centralized method and the proposed method when the number of users is 5, 15, and 30. The horizontal axis represents time, and the vertical axis represents the score at that time.

The time is based on the time taken for the optimization part of each step. In the centralized method, communication delays during the initial model sharing and Lagrange multiplier updates at each step are considered. On the other hand, in the proposed method, communication delays for sharing information and delay for parameter update about constraints are taken into account. The communication and parameter update delay is assumed to follow a normal distribution with a mean of 100 ms and a standard deviation of 10.

We set convergence conditions and examined the time required for both the centralized method and the proposed method to converge for each number of users. The results are shown in Fig. 8. The convergence condition was set such that the error between the moving average recorded at each step and that of the previous step was less than 10^{-4}. The step size of the moving average was 128 for the centralized method and 32 for the proposed method. In the proposed method, inference is performed four times for the optimization of each user, so the design ensures that the number of inferences within the window is equal.

From these figures, it is observed that the proposed method shows a slower increase in convergence time as the number of users grows compared to the centralized method. Under this study's convergence conditions, the proposed method converges faster than the centralized method when there are 12 or more users.



(a) 5 users (b) 15 users (c) 30 users

Fig. 7: Exection time of centralized method and proposed method

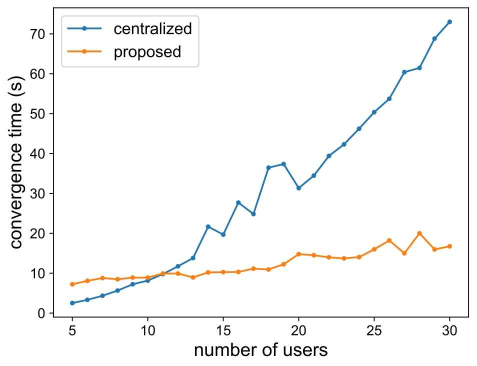


Fig. 8: Convergence time for centralized and proposed method

VI. Conclusion

This paper introduced a distributed optimization method for managing shared object quality in hybrid-metaverse environments. Our approach combines Input Convex Neural Networks for modeling user utility and the Alternating Direction Method of Multipliers for distributed optimization. We evaluated our method using three distinct physical space scenes constructed from the SUN RGB-D dataset, considering both online meeting and party contexts. Experiments demonstrate that our PICNN models accurately represent user utility, with over 93% of test data falling within the 95% confidence interval. Compared to traditional centralized optimization method, our approach shows better scalability, faster convergence, and lower communication overhead as the number of users increases. For scenarios with 12 or more users, our method converges faster than the centralized approach. This work contributes to the field by offering an efficient, adaptive solution for quality management in large-scale hybrid-metaverse applications.

Acknowledgment

This work was partially funded by NICT Commission Research 22608.

References

[1] I. Cluster, "Cluster: Metaverse platform," https://cluster.mu/en/, 2023, [Accessed 01-July-2023].

[2] B. Kang, S. Kang, and I. Hwang, "Ai-driven family interaction over melded space and time," *IEEE Pervasive Computing*, vol. 22, no. 1, pp. 85–94, 2023.

[3] T. Amano, T. Mizumoto, S. Kala, H. Yamaguchi, T. Matsui, and K. Yasumoto, "Visual privacy control for metaverse and the beyond," *IEEE Pervasive Computing*, vol. 23, no. 01, pp. 10–17, jan 2024.

[4] D. I. Gonzalez-Aguirre, J. Perez-Ramirez, J. Felix-Rendon, J. F. Leon, J. Bourgault, J. Z. Esquivel, and L. Nachman, "Robot-based uniform-coverage and high-resolution lidar mapping for physically-grounded metaverse applications," *IEEE Internet of Things Magazine*, vol. 6, no. 1, pp. 40–45, 2023.

[5] S. Sakuma, Y. Mishima, T. Matsui, H. Suwa, K. Yasumoto, T. Amano, and H. Yamaguchi, "3d point cloud-based interaction system bridging physical spaces in virtual environments," in *Proc. of 2024 IEEE International Conference on Pervasive Computing and Communications Workshops and other Affiliated Events*, 2024, pp. 394–396.

[6] J. Chakareski, "Viewport-Adaptive Scalable Multi-User Virtual Reality Mobile-Edge Streaming," *IEEE Trans. on Image Processing*, vol. 29, pp. 6330–6342, 2020.

[7] B. Han, Y. Liu, and F. Qian, "ViVo: Visibility-aware mobile volumetric video streaming," in *Proc. of the International Conference on Mobile Computing and Networking (MobiCom 2020)*, 2020, pp. 137–149.

[8] Z. Liu, Q. Li, X. Chen, C. Wu, S. Ishihara, J. Li, and Y. Ji, "Point Cloud Video Streaming: Challenges and Solutions," *IEEE Network*, vol. 35, no. 5, pp. 202–209, 2021.

[9] ITU-T FG NET-2030 Sub-G2, "New Services and Capabilities for Network 2030: Description, Technical Gap and Performance Target Analysis," ITU-T Deliverable, 2019.

[10] S. Schwarz, M. Preda, V. Baroncini, M. Budagavi, P. Cesar, P. A. Chou, R. A. Cohen, M. Krivokuca, S. Lasserre, Z. Li, J. Llach, K. Mammou, R. Mekuria, O. Nakagami, E. Siahaan, A. Tabatabai, A. M. Tourapis, and V. Zakharchenko, "Emerging MPEG Standards for Point Cloud Compression," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 9, no. 1, pp. 133–148, 2019.

[11] J. Van Der Hooft, T. Wauters, F. De Turck, C. Timmerer, and H. Hellwagner, "Towards 6Dof HTTP adaptive streaming through point cloud compression," in *Proc. of the 27th ACM International Conference on Multimedia (MM2019)*. Association for Computing Machinery, Inc, oct 2019, pp. 2405–2413.

[12] B. Amos, L. Xu, and J. Z. Kolter, "Input convex neural networks," in *Proc. of the 34th International Conference on Machine Learning (ICML 2017)*, vol. 70. PMLR, 2017, pp. 146–155.

[13] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus, "Indoor segmentation and support inference from rgbd images," in *Proc. of the 12th European Conference on Computer Vision (ECCV)*. Springer, 2012, pp. 746–760.

[14] A. Janoch, S. Karayev, Y. Jia, J. T. Barron, M. Fritz, K. Saenko, and T. Darrell, "A category-level 3d object dataset: Putting the kinect to work," *Consumer depth cameras for computer vision: research topics and applications*, pp. 141–165, 2013.

[15] J. Xiao, A. Owens, and A. Torralba, "Sun3d: A database of big spaces reconstructed using sfm and object labels," in *Proc. of the IEEE international conference on computer vision*, 2013, pp. 1625–1632.