Semantic Communication for Capacity-aware Remote Collaboration

Tatsuya Amano Srikant Manas Kala Teruhiro Mizumoto Hirozumi Yamaguchi
Graduate School of Information Science and Technology
Osaka University, Suita, Japan

***Abstract*—The global spread of coronavirus has sparked a considerable interest in technologies that facilitate seamless communication between users which are physically or spatially distant. Using current remote collaboration systems that utilize 3D sensing with LiDAR and depth cameras, point cloud streaming, and MR/VR devices, distant users can communicate with each other as if they did in person. However, these systems may violate users' privacy since they can share information of their entire personal space with other users. In addition, although various point cloud compression methods have been proposed, remote transmission of 3D scenes still requires significant bandwidth. This paper proposes a 3D spatial data sharing system based on the paradigm of "semantic communication", i.e., controlling communication in the units of semantic objects. Our system understands the semantics of the scene and leverages point cloud streaming, thereby enabling users to assert fine-grained control over their privacy. Further, the system adaptively controls the size of the data frame based on network capacity and scene context. The experimental results show that the network delay can be reduced by 96%. We have also tested our system in a commercial 4G network, showing that 3-D spatial sharing with point clouds over severe networks is possible.**

***Index Terms*—Remote Collaboration, Semantic Communication, Point Cloud, MR/VR**

I. Introduction

The worldwide spread of COVID-19 has profoundly impacted the way people communicate with each other due to the limitations imposed by social distancing. The isolation caused by social distancing not only makes it difficult to maintain social relationships, but also causes indirect psychological depression due to restrictions on outings and outdoor activities. Furthermore, the decrease in face-to-face or in-person communication has led to delays in academic timelines, the isolation of elderly people, and the fragmentation of families [1]. Although the restrictions are expected to be relaxed after the pandemic, countermeasures against various infectious diseases, including COVID-19, are still essential. Therefore, there is a need for technologies that offer a sense of spatial proximity which fulfills the human need to feel "connected" even in situations where physical separation is necessary.

Given this social context, remote collaboration systems that allow spatially separated users to communicate with each other as if they were face-to-face have been attracting a lot of attention. In particular, recent developments in mixed reality (MR) and virtual reality (VR) have made it possible to display 3D representations, such as point clouds, meshes, and avatars, of remote users with immersive telepresence. Additionally,

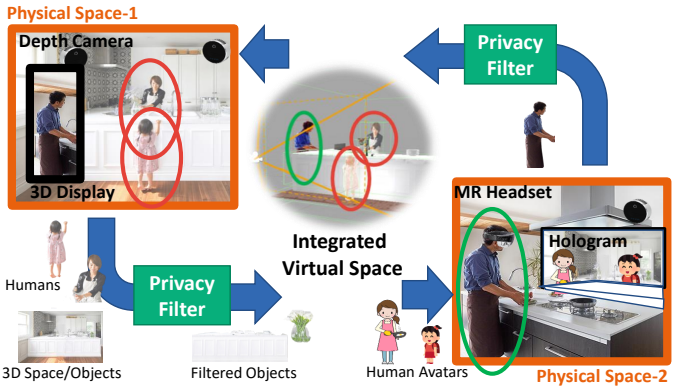


Fig. 1. Remote collaboration system based on 3D space sensing

advances in sensing and spatial recognition technologies have made it possible to capture highly accurate 3D user presence and share a sense of presence with remote users. These technological developments enable remote collaboration systems that permanently "connect" physically distant spaces. This is different from traditional online conferencing systems, which basically depend on an RGB camera and a microphone. In the near future, it is expected that the performance of 5G and 6G communications will be leveraged to share not only 3D scenes, but also human perception and sensation.

However, as 3D space-sharing systems for remote collaboration become more sophisticated and involve a stronger presence, they may unintentionally violate user privacy. In physical space, people naturally share visual, auditory, tactile, and other senses, and they can recognize which information about themselves is shared with others and which is not. Therefore, people can authoritatively control their own privacy-related information [2]. On the other hand, in a virtual space generated by a telecommunication system based on 3D sensing, the captured data and information are unilaterally shared with other users without considering the user's willingness to share. Figure 1 shows an example of a cooking-lecture delivered through a remote collaboration system. Spatial objects and information required for the context of cooking (e.g., things in the kitchen) should be shared seamlessly, but information pertaining to each family's personal space and that of other residents should be filtered and not shared.

Another problem in the recent remote collaboration systems is related to the optimization of 3D representations, such as point cloud, for network communication. For example, the

978-1-6654-6975-3/22/$31.00 ©2022 IEEE

Intel Realsense L515 RGB-D sensor generates about 120,000 points per frame at $30\,\mathrm{FPS}$. Thus, sending the source data requires about $550\,\mathrm{Mbps}$ of network bandwidth, even with one sensor. Although the standardization of point cloud stream compression is in progress [3], a method for adaptive compression and optimization of point clouds based on the 3D scene context has not yet been established.

This paper bridges these gaps by proposing a new architecture for remote collaboration systems. The proposed architecture envisions a semantic understanding of the physical space through multiple sensors. Our system recognizes objects in 3D space, separates their meaning (bounding boxes, bone pose, etc.) from their representation (point cloud, mesh, human avatar, etc.), and transmits and receives each object separately. The approach makes it possible to control connections on an object-by-object basis, allowing users fine-grained privacy control, such as hiding certain objects or changing their representation (e.g., replacing certain object point cloud with existing static mesh object).

To evaluate the performance and feasibility of spatially aware semantic communication, we implemented the remote collaboration system based on the proposed architecture. The results show that the total network-delay can be reduced by 96% using two types of point cloud optimization in a Wi-Fi environment. We have also tested our system in a commercial 4G network, showing that 3-D spatial sharing with point clouds over severe networks is possible.

II. Related Works

Technologies that facilitate remote communication and collaboration are developing rapidly. Microsoft Mesh [4] enables remote collaboration in a 3D space using a Hololens headset. In this Mesh, other remote users appear as 3D avatars superimposed on the user's real space via the headset. There are also other types of systems, such as Mozilla Hub, that allow multiple users of VR headsets to collaborate in a unified virtual world [5]. A tele-immersive system proposed in [6] utilizes a single RGB-D camera and a motion sensor to capture human motions and colored point clouds, and then sends them to a remote VR user to enable communication including gestures. Our goal is to extend the state-of-the-art to realize a system that can sense and integrate multiple physical spaces in real time, and communicate within the unified virtual space through VR, MR, or ordinary displays.

Various studies have been conducted on the transfer of 3D volumetric representation streams, such as point clouds, over networks [7], [8]. According to the ITU-T report on new services and capabilities needed in 2030, richer, more immersive, and interactive services such as holographic, haptic, and volumetric communication are expected to become widespread [9]. It is suggested that the requirements of these services (e.g., bandwidth and latency) are difficult to meet in an end-to-end or overlay fashion and must be coordinated with functions in the network. Volumetric streaming generally assumes MR and VR equipment as the receiving device and allows receiver-side users to move in six degrees of freedom. In particular, point cloud compression and streaming have attracted a lot of interest, with ISO/IEC MPEG standardization underway [3]. Existing studies on point cloud optimization or encoding take the approach of controlling the quality of point cloud based on the receiver viewport, bandwidth, and client buffering [8], [10].

Semantic communication focuses on transferring the human interpretation of data (e.g., speech scripts) rather than on the communicated data itself (e.g., voice signals) [11]. By leveraging the significance and utility of information, more human-oriented compression and optimization is possible. It also facilitates a natural introduction of privacy controls into communications [12].

Thus, understanding the scene is a key function in semantic communication, and recent advances in deep learning have made it possible to extract the semantics of the scene from the point clouds. PointNet [13] is one of the most famous approaches, which can deal with point clouds using neural networks without converting them to other representations.PointNet++ [14] aims to improve accuracy by capturing the local structure of point clouds through the hierarchical application of PointNet. Further, VoteNet [15] improves the performance with the detection of the object's centroid by the voting mechanism.

III. System Overview

The proposed architecture is shown in Fig.2. For simplicity of explanation, we assume one sender and one receiver in this example, and we will later explain how we can extend the architecture for multi-user scenarios. The sender's 3D scene is captured by synchronized multiple sensors and depth cameras as colored 3D point clouds. The overall outline is that the point cloud streams are sent to the receiver system via the network, and the receiver can view the scene through the MR/VR headset.

Our system is based on the concept of semantic communication that distinguishes *semantics* and *representation* of an object, and transfers them with different connections. The (spatial) semantics of an object are basically its 3D bounding box and object type (human, chair, etc.). More "fine-grained" semantics rather than the 3D bounding box can also be considered if available. For example, the posture of a human object can be captured with a body tracking algorithm, which is also regarded as a spatial semantic instead of the 3D bounding box. The representation is a point cloud that corresponds to the object.

The sender system first applies semantic segmentation to the point cloud of the input scene. Segmentation results in a set of spatial semantics (3D bounding boxes and types) and a set of representations (point clouds) of the segments. Next, privacy filtering is applied to spatial semantics and representations. This either removes the objects or modifies their representations according to user-specified rules. For example, a sender may wish to hide the details of people outside the kitchen and show only their presence and behavior, by using (a) raw point clouds without colors, or (b) avatars.

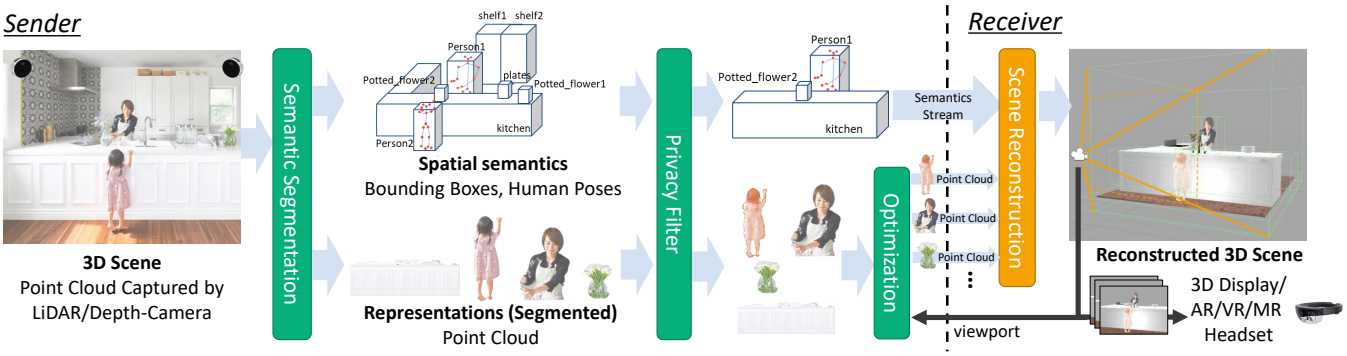


Fig. 2. Spatial data sharing based on semantic communication (one-way)

This rule-based object filtering allows users to ensure their privacy. We assume this rule is predefined by the user for each object category.

The stream of filtered spatial semantics is sent over the network to the receiver. The corresponding point cloud representations are sent as well. However, it is often difficult to send the entire point cloud as is due to the network capacity and data volume. Therefore, our system applies two types of optimization for point clouds. The optimizations leverage network capacity measurements and obtained spatial semantics that provide the types of spatial objects and events of human-object interactions. Details are provided in Section V.

The receiver system gets the spatial semantics and reconstructs an entire 3D scene representation based on it. The receiver can choose the representation of each spatial semantic. For example, with human poses and object bounding boxes in the spatial semantics, the receiver can visualize the person's movements by applying a humanoid avatar to the received pose information. They can also spatially fill the bounding boxes of static objects with 3D mesh object data. Alternatively, everything can be represented in text or 2D images. These processes generate a 3D scene, and by setting a viewport (viewing frustum) within the scene, the receiver can see them via MR/VR or an ordinary display.

To convey the situation in real time, the point cloud of objects is used partially in the reconstruction of the 3D scene. Each bounding box contains URI to access the point cloud stream of the corresponding object in the sender scene. The receiver system opens a new network connection to the sender system as needed, to obtain the real-time point cloud representation of the object in question. The received point cloud is placed on the reconstructed scene based on the corresponding semantics (bounding boxes).

The proposed spatial-sharing architecture scales easily to multi-user scenarios. Each sender only needs a single server that bundles spatial semantics (Fig. 3), and transmits semantics to the "merging server" rather than directly to the receivers. The merging server converts the multiple semantic streams into a single merged stream. The receiver system receives semantics from the merging server, and all other processes are the same as in the above exemplified scenario. Since the

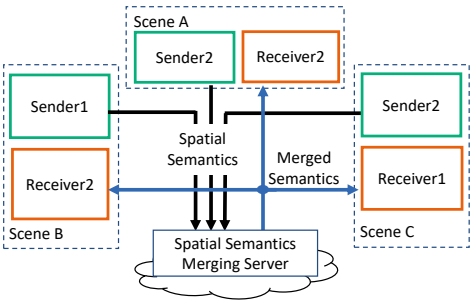


Fig. 3. Flow of the spatial semantics in the bidirectional multiuser scene

data size of semantics is very small, the semantic merging process is fast compared to approaches that integrate multiple point cloud streams.

When bundling semantic streams, the merging server converts each sender's local coordinates in the semantics to the unified global coordinates. For this merging, this paper uses a simple algorithm that an offset presets the relative position of each sender space in the global coordinate, and each coordinate value in semantics is rewritten based on that offset in the real-time bundling process.

IV. Semantic Segmentation Design

In this section, we explain design details of the semantic segmentation, which is applied by a sender to each frame of point clouds captured by sensors in the sender scene. The input point cloud is a colored point cloud that we can acquire by RGB-D cameras. We assume the segmentation is performed at the each client side.

To investigate the performance of semantic segmentation methods and choose the one that best suits our objective, we examined the processing speed of semantic segmentation methods for 3D point clouds and 2D frames with a same machine. The result is shown in Table I. For the 2D segmentation input, RGB data from Azure Kinect was used, which was scaled down to 1280×720. PointNet and VoteNet results from models trained with SUNRGB-D dataset, and the result of Yolov5x and Faster RCNN is by models trained with COCO Image dataset. Azure Kinect uses a combination of a ToF sensor and an RGB camera to capture colored point clouds.

TABLE I
SEMANTIC SEGMENTATION PROCESS TIME

	average inference time per frame [ms]
VoteNet	381
PointNet	512
Yolo v5x (2D)	23
Faster-RCNN (2D)	439

Therefore, applying 2D segmentation to the RGB camera data yields a 3D segmentation result for a single sensor. To use multiple sensors, the segmentation results from each sensor are integrated.

Based on the investigation, we chose Yolov5 based semantic segmentation since its inference time is remarkably short compared to other frameworks. Yolov5 provides object detection with bounding boxes and does not support per-pixel segmentation. To achieve the 3D semantics using Yolov5, we apply the following processes: 1. Fixed objects such as floors, walls, and desks are extracted from the point cloud by prior manual region setting. 2. Euclidean clustering is applied to the remaining point cloud to obtain point cloud segments. 3. Object labels by Yolov5 are given to the segments that intersect the bounding box of the Yolov5 result.

Furthermore, to avoid segmented object jittering, our sender system employs a point cloud-based object tracking method that we have proposed in [16]. The method employs a basic Kalman-filter-based tracking approach.The unique proposition of the method is that it continuously monitors the Kalman filter for spatial merging or splitting of observations to account for noise or occlusion defects in the point cloud data. Thereafter, it uses the results to generate a virtual frame of observations every time.

V. POINT CLOUD OPTIMIZATION

The sender system optimizes the transmitting point cloud of each object. We employ two types of optimization approaches: Object-type-aware optimization (OO) and visibility-aware optimization (VO). Each optimization ultimately determines the spatial downsampling (SDS) rate and the temporal downsampling (TDS) rate of each spatial object o for each connection to receiver i. SDS process is used to reduce the amount of data per transmitted frame. It is implemented by removing some points from the point cloud of the object combining voxel grid filtering [17] and random downsampling. If the data size of one frame is $100\,\mathrm{Mbits}$ and the SDS ratio is 0.1, the data size is reduced to $10\,\mathrm{Mbits}$. TDS means reducing the frequency of point cloud data transmission. For example, if the point cloud source update interval is $100\,\mathrm{ms}$ and the TDS rate of it is 0.5, the transmission interval increases to $200\,\mathrm{ms}$. The premeasured network capacity (bandwidth) between the sender and receiver i is denoted by $\bar{T}_i$ in the following sections.

A. Object Type-aware Optimization

To apply OO, the sending system uses the results of semantic segmentation to classify objects into the following four groups. (1) static objects, (2) persons, (3) objects with which persons interact, and (4) objects with which any person does not interact.

Since persons are the subjects of communication and move with high frequency, the system sends point clouds of them as close to the source rate as possible. Static objects, such as walls and floors, change very little and are of minimal importance. Similarly, objects with which humans interact (hold or touch) are of equal importance to humans because of the possibility of shape change, whereas objects with which they do not interact are less likely to change their appearance and have low importance for humans. Whether each object is interacted with by some persons is determined by whether the bounding boxes of them intersect or not.

Based on the above considerations, the downsampling ratios for each group of objects are determined as follows. The set of objects belonging to group (1) to group (4) is denoted by G_1, G_2, G_3, and G_4, respectively. For an object o, denote the transmission interval of the source by $I(o)$ and the size of the source frame by $F(o)$. The original network capacity required to transmit the point cloud of the object o can be calculated by $F(o)I(o)$ bps. Denoting by $\alpha_{i,g}$ the total downsampling ratio of SDS and TDS for the object o belonging to the group $g \in \{1,2,3,4\}$, the overall throughput after applying the OO can be rewritten by Eqn. (1).

$$T_i(\alpha_{i,1},\alpha_{i,2},\alpha_{i,3},\alpha_{i,4}) = \sum_{g=1}^{4} \alpha_{i,g} \sum_{o \in G_g} I(o)F(o) \tag{1}$$

where

$$\begin{aligned} T_i(\alpha_{i,1},\alpha_{i,2},\alpha_{i,3},\alpha_{i,4}) \leq\ & \bar{T}_i \\ \alpha_{i,1} : \alpha_{i,2} : \alpha_{i,3} : \alpha_{i,4} =\ & A_1 : A_2 : A_3 : A_4 \end{aligned}$$

A_1 ,..., A_4 are predefined system parameters. A_g represents the relative importance of the group G_g. The system determines the total downsampling ratio $\alpha_{i,g}$ that maximizes Eqn. (1) with the linear optimization. Then the SDS ratio $\beta_{i,g}$ and the TDS ratio $\gamma_{i,g}$ for each group can be calculated from $\alpha_{i,g}$ and the predefined ratio parameter B_g in the equation $\alpha_{i,g} = \beta_{i,g}\gamma_{i,g}$ where $\beta_{i,g} : \gamma_{i,g} = 1 : B_g$

The sender system applies SDS and TDS according to the calculated ratios $\gamma_{i,g}$ and $\beta_{i,g}$.

B. Visibility-aware Optimization

VO is a method that controls the SDS rate based on the distance between the receiver and the object and is often used by recent VR / MR and point cloud based remote collaboration systems. We simply used distance-based SDS according to a survey in [7] that investigated the relationship between point cloud density, distance, and quality of experience. SDS ratio $\delta_{i,o}$ based on the distance $d_{i,o}$ m between the object o and the receiver position i is determined by Eqn. (2).

$$\delta_{i,o} = \begin{cases} 1.0 & (0 \leq d < 3.2) \\ 0.8 & (3.2 \leq d < 4.2) \\ 0.6 & (4.2 \leq d < 5.2) \\ 0.4 & (5.2 \leq d < 6.2) \\ 0.2 & (6.2 \leq d) \end{cases} \tag{2}$$

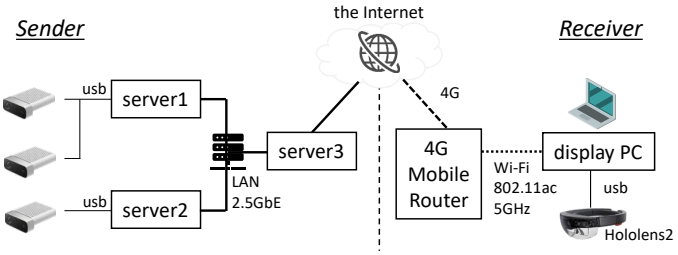
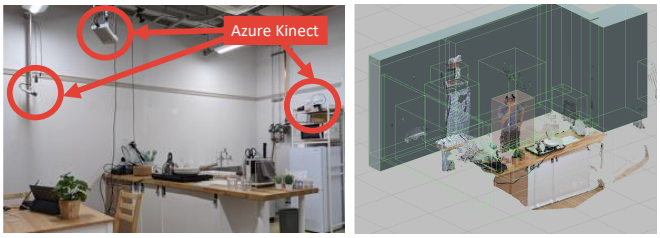


Fig. 4. Network Architecture and Device Connections for the Experiment



(a) Physical Environment (b) Reconstructed Scene

Fig. 5. Experimental Environment of the Sender

When combining OO and VO, for each object, the smaller of $\beta_{i,g}$ and $\delta_{i,o}$ is selected as the valid SDS rate. At the same time, as an optimization based on the field of view, the receiver's frontal viewing angle is assumed to be 120 degrees, and the point clouds of all objects that do not overlap with that field of view are not transmitted to the receiver i.

In order to apply VO, the sender system should know the receiver's viewport, and since there can be delays in sending and receiving this viewport, the sender system predicts the position of the receiver's viewport. The linear movement of each receiver's viewport is tracked and predicted with a Kalman filter. Horizontal position (x_i, y_i) and absolute orientation (θ_x, θ_y) are used as state variables for this spatial tracking.

VI. Performance Evaluation

A. Environmental Setup

We evaluated the communication performance of the proposed system. Figure 4 shows the network setup for the evaluation in a 4G environment, and Figure 5 shows the physical space of the sender to be captured. Three Azure Kinect sensors are used to capture the sender scene. Each sensor is connected to server1 and server2 via usb, and semantic segmentation using Yolov5 is applied to the data from each sensor. All point clouds and segmentation results are merged on server3, and all sender system processes are also executed on server3. From server3, each object's spatial semantics and point cloud are sent to the receiver system via wired Ethernet, Wi-Fi or 4G network.

The receiver system consists of a mobile router to connect 4G, a display PC connected to the router through 802.11ac Wi-Fi, and an MR headset (Hololens 2) connected to the display PC via USB. We kept the distance between the receiver's viewport and the center of the received point cloud at approximately $5\,\mathrm{m}$. In the evaluation with Wi-Fi connection, the display PC is directly connected to the sender system's LAN via Wi-Fi. The CPU on servers 1 and 2 is an Intel Core i7-7820HQ with an NVIDIA GTX1650 GPU, and the CPU on server 3 is an Intel Core i9-10980XE with an NVIDIA RTX6000 GPU.

We implemented all processes and connections using ROS, a distributed pub/sub middleware. ROS master, an ROS process that manages ROS message topics runs on server3. Note that although the ROS master manages the list of topics, all actual sensor data are sent and received via point-to-point communication of the devices rather than through a broker like MQTT middleware. Also, the point cloud stream of a single object is implemented as a single topic. To cross the NAT in both the sender and receiver, we use a WireGurad VPN.

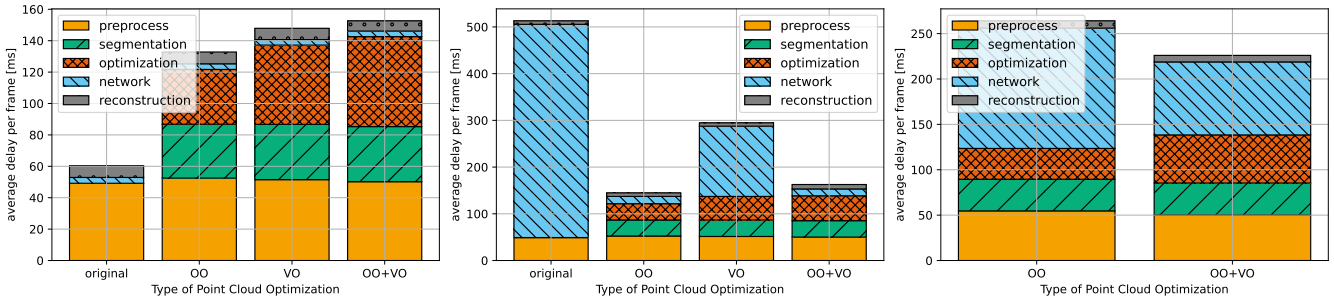
The end-to-end (server3 to display PC) PING latency is $0.35\,\mathrm{ms}$, $3.8\,\mathrm{ms}$ and $94.2\,\mathrm{ms}$ via Ethernet, Wi-Fi and 4G on average, respectively. The downlink bandwidth tested by Iperf3 is $763\,\mathrm{Mbps}$, $229\,\mathrm{Mbps}$ and $32\,\mathrm{Mbps}$ on average via Ethernet, Wi-Fi and 4G, respectively. The predefined parameters of proposed system A_1, A_2, A_3 and A_4 is set to 3,1,3 and 1, respectively. Also all B_g values are set to 0.5. This means that OO applies downsampling by SDS at twice the rate of that by TDS.

In the experimental environment, the total point cloud of the three sensors contains 276,480 points on average, and the source frame rate of the integrated point cloud is 5 FPS. The point cloud format is ROS's PointCloud2 binary format, where each point consists of 16 Bytes XYZ coordinates and 4 Bytes RGBA color information. Therefore, the original point cloud transmission with the source frame rate requires a bandwidth of about $400\,\mathrm{Mbps}$ or more. Our implementation uses ROS custom messages as spatial semantics and employs Websocket protocol for server-to-server communication.

B. Performance on Commercial Mobile Network

Figure 6 shows the delay time for each process from the time a single point cloud is captured to the time it is received and displayed. All results are averages over the initial 600 frames. The preprocess delay includes the time required for the synchronization of multiple sensor data and aggregation to server3. The result of original point cloud streaming is not shown in the figure since sending a single original frame via 4G resulted in a network delay of more than 10 seconds. All metrics, including bandwidth and process delays, were investigated using the rostopic command of ROS.

As shown in Fig. 6(a), the end-to-end bandwidth in the Ethernet environment was enough, and the original source could be sent as is without incurring significant network delay. In a Wi-Fi environment (Fig. 6(b)), where the bandwidth is less than half that of the Ethernet environment, the scene latency increased significantly when sending the original



(a) via Ethernet (LAN) (b) via Wi-Fi (c) via 4G (The results of original and VO are excluded from the figure since they took more than 10 seconds.)

Fig. 6. Process and Network Delays

source data. On the other hand, when OO was applied, the network latency did not increase. OO ensures that the total necessary bandwidth is within the premeasured maximum capacity by adaptively downsampling point clouds based on per-object type priority. Since VO is not based on capacity, the average total downsampling rate was only about 60% in this experimental setting.The combination of VO and OO successfully reduced latency from $456.3\,\mathrm{ms}$ to $15.0\,\mathrm{ms}$ by 96% in a Wi-Fi environment. Although the point clouds are significantly downsampled, the quality of the reconstructed scene for humans was maintained thanks to semantic communication. The results in the 4G network environment are also shown in Fig. 6(c). On this network, it took more than 10 seconds to send a single frame of point cloud with the original source and VO applied only.

VII. Conclusion

In this paper, we proposed an architecture to share spatial data using 3D sensing technologies. The proposed communication architecture, which enables control on an object-by-object basis, has been confirmed to enable smooth 3-D scene remote sharing even on commercial mobile networks with relatively narrow bandwidth. We applied rule-based algorithms to privacy control and semantics merging in this paper, but in future works we will replace these with methods based on reinforcement learning and spatial understanding to evaluate overall performance. We also plan to integrate point cloud compression techniques and conduct the evaluations in terms of the quality of experience.

Acknowledgment

The work was supported by “Research and Development of Information and Communication Technologies that Contribute to Countermeasures against Infectious Diseases (222-C03)”, the Commissioned Research of the National Institute of Information and Communications Technology (NICT), JAPAN.

References

[1] I. Ali and O. M. Alharbi, “Covid-19: Disease, management, treatment, and social impact,” *Science of the total Environment*, vol. 728, p. 138861, 2020.

[2] H. Regenbrecht, S. Zwanenburg, and T. Langlotz, “Pervasive Augmented Reality - Technology and Ethics,” *IEEE Pervasive Computing*, vol. PP, pp. 1–8, 2022.

[3] 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.

[4] Microsoft, “Mesh (Preview) Overview.” https://docs.microsoft.com/en-us/mesh/overview, 2022. [Accessed 20-June-2022].

[5] P. Knierim, T. Kosch, and A. Schmidt, “The Nomadic Office: A Location Independent Workspace Through Mixed Reality,” *IEEE Pervasive Computing*, vol. 20, no. 4, pp. 71–78, 2021.

[6] X. Lu, J. Shen, S. Perugini, and J. Yang, “An Immersive Telepresence System using RGB-D Sensors and Head Mounted Display,” nov 2015.

[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)*, pp. 137–149, 2020.

[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, Oct. 2019.

[10] 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)*, pp. 2405–2413, Association for Computing Machinery, Inc, oct 2019.

[11] Z. Weng and Z. Qin, “Semantic communication systems for speech transmission,” *IEEE Journal on Selected Areas in Communications*, vol. 39, pp. 2434–2444, 2021.

[12] G. Shi, Y. Xiao, Y. Li, and X. Xie, “From Semantic Communication to Semantic-Aware Networking: Model, Architecture, and Open Problems,” *IEEE Communications Magazine*, vol. 59, no. 8, pp. 44–50, 2021.

[13] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, “PointNet: Deep learning on point sets for 3D classification and segmentation,” in *Proc. of the 30th IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2017)*, pp. 77–85, 2017.

[14] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, “PointNet++: Deep hierarchical feature learning on point sets in a metric space,” in *Proc. of the Advances in Neural Information Processing Systems (NIPS 2017)*, pp. 5100–5109, 2017.

[15] C. R. Qi, O. Litany, K. He, and L. J. Guibas, “Deep hough voting for 3d object detection in point clouds,” in *Proc. of the IEEE International Conference on Computer Vision*, 2019.

[16] R. Ukyo, T. Amano, A. Hiromori, and H. Yamaguchi, “Pedestrian tracking in public passageway by single 3d depth sensor,” in *Proc. of the 4th International Workshop on Pervasive Computing for Vehicular Systems (PerVehicle 2022)*, pp. 581–586, 2022.

[17] R. B. Rusu and S. Cousins, “3d is here: Point cloud library (pcl),” in *2011 IEEE International Conference on Robotics and Automation*, pp. 1–4, 2011.