mathematics

MDPI

Article

A Game-Theoretic Approach for Rendering Immersive Experiences in the Metaverse

Anjan Bandyopadhyay [1], Ansh Sarkar [1], Sujata Swain [1], Debajyoty Banik [1], Aboul Ella Hassanien [2], Saurav Mallik [3,*], Aimin Li [4,5] and Hong Qin [6,*]

1 School of Computer Science and Engineering, Kalinga Institute of Industrial Technology, Odisha 751024, India
2 Faculty of Computer and AI, Cairo University, Giza 12613, Egypt
3 Department of Environmental Health, Harvard T H Chan School of Public Health, Boston, MA 02115, USA
4 School of Computer Science and Engineering, Xi'an University of Technology, Xi'an 710049, China
5 School of Precision Health, The University of Texas Health Science Center at Houston, Houston, TX 77030, USA
6 Department of Computer Science and Engineering, University of Tennessee at Chattanooga, Chattanooga, TN 37403, USA
* Correspondence: sauravmtech2@gmail.com or smallik@hsph.harvard.edu (S.M.); hong-qin@utc.edu (H.Q.)

Abstract: The metaverse is an upcoming computing paradigm aiming towards blending reality seamlessly with the artificially generated 3D worlds of deep cyberspace. This giant interactive mesh of three-dimensional reconstructed realms has recently received tremendous attention from both an academic and commercial point of view owing to the curiosity instilled by its vast possible use cases. Every virtual world in the metaverse is controlled and maintained by a virtual service provider (VSP). Interconnected clusters of LiDAR sensors act as a feeder network to these VSPs which then process the data and reconstruct the best quality immersive environment possible. These data can then be leveraged to provide users with highly targeted virtual services by building upon the concept of digital twins (DTs) representing digital analogs of real-world items owned by parties that create and establish the communication channels connecting the DTs to their real-world counterparts. Logically, DTs represent data on servers where postprocessing can be shared easily across VSPs, giving rise to new marketplaces and economic frontiers. This paper presents a dynamic and distributed framework to enable high-quality reconstructions based on incoming data streams from sensors as well as to allow for the optimal allocation of VSPs to users. The optimal synchronization intensity control problem between the available VSPs and the feeder network is modeled using a simultaneous differential game, while the allocation of VSPs to users is modeled using a preference-based game-theoretic approach, where the users give strict preferences over the available VSPs.

Keywords: metaverse; LiDAR; game theory; digital twins; DSIC mechanism; virtual service provider

MSC: 91-10

Citation: Bandyopadhyay, A.; Sarkar, A.; Swain, S.; Banik, D.; Hassanien, A.E.; Mallik, S.; Li, A.; Qin, H. A Game-Theoretic Approach for Rendering Immersive Experiences in the Metaverse. *Mathematics* **2023**, *11*, 1286. https://doi.org/10.3390/math11061286

Academic Editor: Francesco Calimeri

Received: 17 January 2023
Revised: 3 March 2023
Accepted: 4 March 2023
Published: 7 March 2023

1. Introduction

The word metaverse itself is a word formed by the conjunction of meta and verse which means beyond the universe. The metaverse is a newly developed and partially adopted concept of 3D cyberworlds or universes that we can create based upon the concept of virtual reality (VR) [1] and augmented reality (AR) [2] to extend immersive experiences to users; see Figure 1. With major commercial interest being vested in the space from companies such as Meta and Roblox, it would not be wrong to say that the metaverse has gained traction recently and shows no signs of slowing down. As this space is currently being built, a lack of coherence can be seen in terms of the vision toward which these major industry players are progressing. While companies such as Meta are more focused on

building a VR-based social media platform, Roblox and Sandbox are focusing on enabling the creation of user-generated games along with digital worlds and assets tradeable as NFTs on global marketplaces. This lack of coherence or structure has been attributed to a lack of definition, as the metaverse is still being built and no concrete vision exists. This paper aims at presenting a hierarchical model that could act as a potential framework for the implementation and development of the metaverse, by leveraging a multitude of virtual service providers (VSPs) and connecting them to an interconnected network of LiDAR clusters.

LiDAR stands for light detection and ranging and represents a remote-sensing methodology that leverages the properties of pulsed laser light to measure the distances and precise spatial locations of points in the surrounding environment. The received light pulses are combined with data collected across other LiDAR sensors and used to generate precise, three-dimensional point clouds depicting the shape of the surrounding surfaces and objects.

The use of LiDAR in this context was primarily governed by its natural ability to scan and work with real-time three-dimensional, point-cloud-oriented data and the simplicity of applications that leverage multiple such clouds to form high-resolution three-dimensional models that are quintessential to the formation of a metaverse.

A common strategy followed by new entrants aimed toward market capture is to disseminate services for free, over a limited period of time for people to get comfortable with the technology, after which, a basic or minimal charge can be levied to continue the service. A similar strategy was used by Jio India by adopting a freemium model and providing customers with free internet services for the initial three months after which a minimal fee was charged to retain the customers. This led to rapid market capture translating to roughly 50 million subscribers within 83 days of launch. In this text, we assume that the VSPs being considered are following a similar model, wherein they provide free services over an initial time span but later resort to the collection of a minimal fee for the continuation of the service.

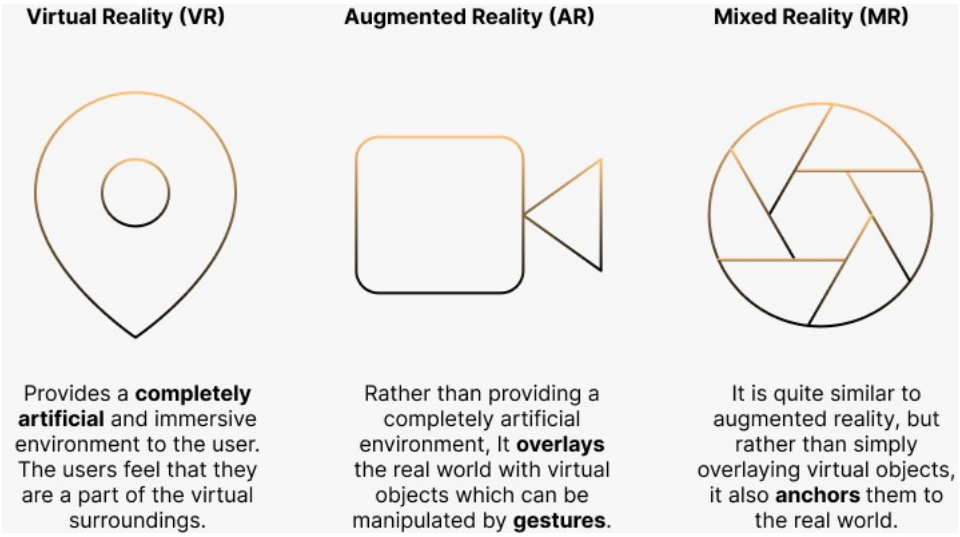

Figure 1. A quick and concise comparison of the various types of artificial reality paradigms currently in existence.

The interaction between the interconnected network of LiDAR sensors (the ground or base layer) with the VSPs (virtual service providers) is modeled as a double-sided matching [3,4] problem wherein every LiDAR sensor chooses a VSP to send its data for processing, based on its preference list. These data are mostly in the form of three-dimensional point clouds or coordinates which can later be reconstructed to get a 3D model. On the other hand, the VSPs also try to choose the best possible LiDAR sensor to accept data from, based on their own preferences. The incoming data streams undergo processing on the servers which try to reconstruct the best-resolution environments possible.

Multiple VSPs together form a distributed service-providing network (something similar to CDNs, content delivery networks). The users then choose or request VSPs based on their preferences, which are modeled in this paper as a single-sided matching problem. Once all the allocations have been completed, the user can enjoy the best possible immersive reality experience possible at their location. The entire paper revolves around the implementation and simulations related to the above-mentioned game-theoretic approach to get the most stable/optimum mapping of all the entities involved in the system.

Another key application of the proposed framework can be found in the field of computational bioinformatics [5–9] (Figure 2), where processes such as gene selection, deep-learning-based gene association, etc., can be divided into smaller tasks and run parallelly across servers in the cloud. The laborious and expensive task of screening a potentially vast number of drug-target protein combinations with biological tests was tackled in [10], whereas the use case of drug discovery and drug repurposing was explored in [11,12]. Novel mechanisms were researched in [13] under which distributed systems could collaborate based on a consensus mechanism for gene selection while [14] extended the same research to DNA microarray data. Apart from drug discovery and gene selection, [15] brought to light the potential applications of efficient distributed computing in gene-expression-based profile exploration. A more practical approach to this application was taken by [16], which explored the implementation of such classification and profiling algorithms for gene expression on public clouds. This, in turn, displayed the intersection that the proposed system in this paper has with the computational and mathematical modeling of biology and their implementation in a highly parallelized and distributed environment.

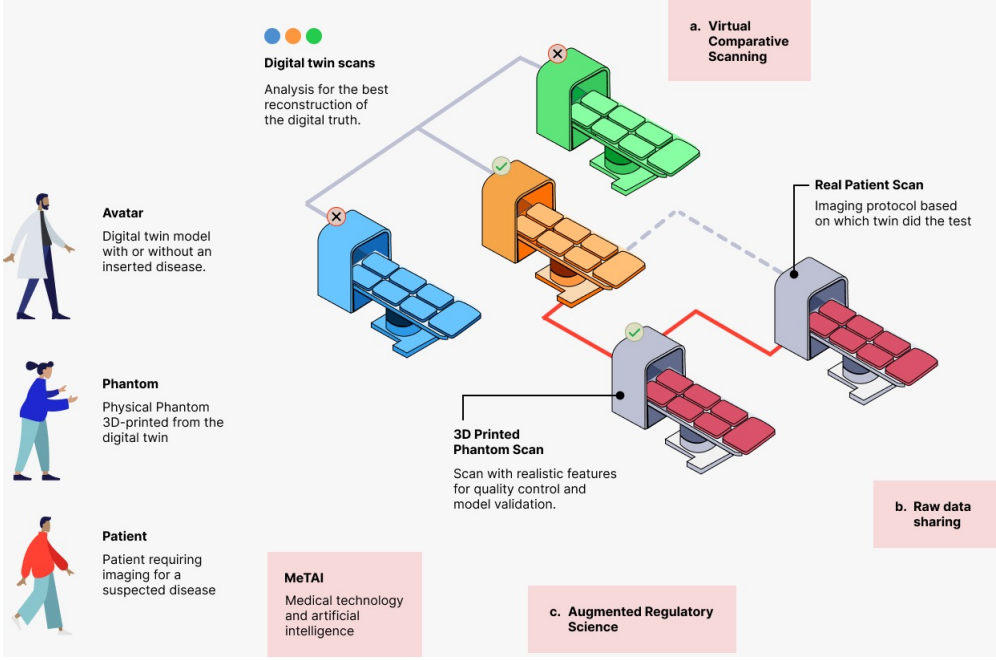

Figure 2. Metaverse in healthcare applications.

2. Literature Review

The world has shifted towards establishing an online presence. With classrooms and meetings being held online via platforms such as Google Meet and Zoom, everything ranging from the ordering of food to watching movies has adopted an online outlook. The interactivity and immersiveness of these platforms are what the metaverse aims to enhance by building a virtual environment around the user and making them feel their presence at the site of the action. A lot of texts exist on the topic and it is important to understand the foundations laid by them before proposing new frameworks aimed toward improving allocations within the ecosystem powering the metaverse [17–20].

Neal Stephenson in the year 1992 published his science fiction book titled Snow Crash, introducing the world to the metaverse [21]. The concept was elaborated as an omnipresent virtual environment that coexisted with the physical world where the primary communication agents were digital avatars controlled by their human analogs. Since its initial appearance, a wide range of concepts have been used for the somewhat abstract nature of the metaverse representing a computer-rendered and computer-maintained reality, enabling features including but not limited to lifelogging [22], social and collective virtual space [23], the introduction of spatial internet [24], translating the concept of a mirror world to reality [25], and even the possibility of an omniverse, representing space for simulation and collaboration [26]. Numerous online events have displaced in-person social gatherings, such as the virtual commencement held by UC Berkeley in 2021 [27] and the virtual concert hosted by Fortnite in 2019 that was reportedly watched by 10.7 million people [28]. The idea behind the metaverse has emerged and undergone development by academia in conjunction with industries vesting commercial interest since 2019 and can be used to describe these virtual occurrences as the real-life implementation of computer-simulated cyberspace interwoven with physical reality [29]. It represents a developing platform where independent creators create digital environments and bridge the gap between the physical world and cyberspace. Every virtual world is rendered by a virtual service provider (VSP) that offers targeted virtual services to their users, such as virtual worlds (e.g., VR chat 1), as well as enabling virtual safaris and sightseeing [30], in-person experience via virtual theme parks [31], non-fungible token (NFT)-based gaming platforms, social networking, and other computer-rendered reality applications. Metaverse adopters would be represented by avatars (their digital twins) via which they could interact across multiple virtual spaces with cross-platform user accounts [32]. This could open new fields for identity management platforms, providing platform-independent identities for seamless access to services across the metaverse. These services might appear to be comparable to the many computer-mediated worlds that exist today [23]. However, the bulk of today's virtual worlds is run autonomously, limiting user access and the sharing of digital assets across platforms. Since each platform uses a different engine, a user cannot move their virtual resources to different platforms thereby introducing an element of discontinuity in the user experience.

Three-dimensional reconstruction technologies would form a key part in the development of the metaverse [33] especially in order to provide real-time data via communication channels between digital twins (DTs) and their real-world counterparts. LiDAR sensors are one such class of sensors that could enable accurate real-time 3D reconstructions of surroundings [34]. A sample rendering of a 3D LiDAR scan (https://www.realserve.co.nz/reality-capture/3d-laser-scanning/, accessed on 15 January 2023) inside a factory depicting one of its few industrial use cases. It is essentially a method for determining distances by targeting objects and surfaces with lasers and then measuring the time taken for the reflected light to be received at the receiver, based on which it forms a point cloud that can be further refined to enable object or surface reconstruction. The concept of real-time 3D LiDAR-based reconstruction was presented in [35] by combining pre- and post-processing techniques and methodologies and applying them on batches of scanned point-cloud data received from live LiDAR feeds that were variably processed. The unorganized data obtained in the form of cloud datasets, and further used as transitional data models in applications, usually contain considerable amounts of noise, due to variations of local point density and incomplete, overlapping, or missing data, primarily caused due to the scattering characteristics of the environment [36]. Methods for dealing with these irregularities were presented in [37]. LiDAR sensors are sensors that could essentially be placed anywhere. They could be fixed to the ground or be flown with aerial vehicles [38,39] and underwater devices [40,41]. The reconstruction of man-made structures such as buildings [42] and other landscape features [43] was also discussed in [44,45] and novel algorithms involving the segmentation of triangles from a 3D mesh have been put forth.

This paper aims at establishing a model that can enable the interoperability of these digital assets, including the digital avatars and associated identities belonging to users, thereby imparting a far better and more seamless experience than would otherwise have been possible.

3. System Model and Problem Formulation

The vastly complex problem of the proper allocation of resources in the metaverse has two major constituents to it, the first one models the allocation of users to their most-preferred VSPs based on their preference list in order to enable faster response times, efficient dissemination of services and better immersive experiences.

The second component involves a double-sided matching of the interconnected network of LiDAR clusters to corresponding virtual service providers (VSPs) in order to enable the storage and sharing of digital twins while also resulting in an increase in net continuous geospatial area covered and the quality of the rendered outputs by pooling the feeds from multiple LiDAR sensors together, running appropriate reconstruction algorithms for cleaning irregularities in scanning and therefore obtaining the best-quality three-dimensional rendered outputs of entire regions. These data obtained from across the feeder network are used to extend services and immersive experiences to the users while also sustaining online marketplaces involving transactions related to the lending of DTs across multiple providers.

Figure 3 depicts the overall framework which we propose in this paper. The following sections expand on this further and define in a detailed manner, the various entities and actions involved in each of the above-mentioned environments.

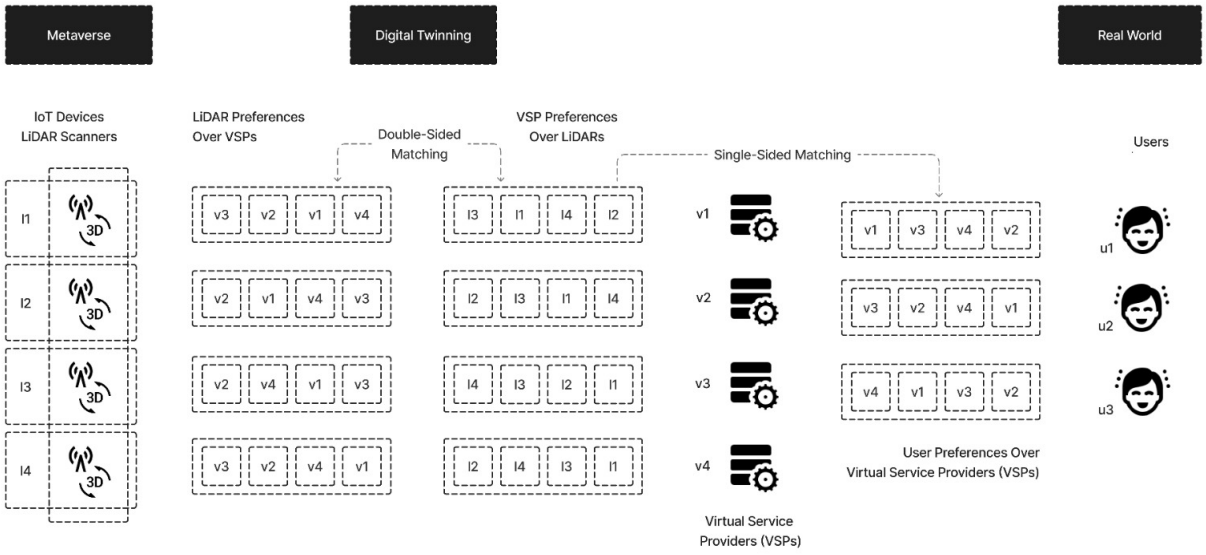

Figure 3. An overview of the proposed framework.

3.1. LiDAR–VSP Matching

The first stage of our proposed framework involves the matching of a LiDAR sensor (from a set of LiDAR sensors) to the best available VSP (from a set of VSPs) and vice versa based on preference lists provided by each entity from both sets. It implies that this part of the model can be defined using the foundational theories available to generate a matching that is stable and exhibits a one-to-one allocation of resources (one LiDAR sensor is mapped to one VSP). A double-sided matching algorithm is used in order to obtain the optimum allocation of LiDAR sensors to VSPs. Each type of agent maintains a strict preference list over the entities on the other side, which is taken into consideration while allocating the resources.

Two disjoint sets of LiDAR sensors and virtual service providers (VSPs), respectively, constitute the model. We assume that all the VSPs are homogeneous, offer the same data-processing facilities, and are equal in number to the count of LiDAR sensors present in the network being considered. Moreover, every LiDAR sensor is assumed to be located in a separate geospatial area, in a way that ensures the maximization of the net geographical area covered.

Even though the specific case of $m = n$ is being considered here, where m denotes the count of LiDAR sensors contained in the disjoint set L and n denotes the count of VSPs contained in the disjoint set V, cases where $m \neq n$ can also be handled with ease by the application of the proposed algorithm over a subset of the entities. The set containing the minimum number of entities is taken as a whole and a subset of the other disjoint set containing the same number of elements is selected, and both are then passed to the proposed algorithm.

Formally, we define a set of size n as $L = \{l_1, l_2, \ldots, l_n\}$ denoting the set of all LiDAR sensors, where l_i represents the ith LiDAR sensor in the set. We similarly define a set $V = \{v_1, v_2, \ldots, v_n\}$ denoting the set of all VSPs, where v_i represents the ith VSP in the set. Both L and V are disjoint and finite. Let us consider an example where $n = 4$, i.e., we have four sensors which are to be mapped to four VSPs and vice versa.

Example 1. *Consider the finite sets of $n = 4$ elements L of sensors containing the elements $\{l_1, l_2, l_3, l_4\}$ and V of VSPs containing the elements $\{v_1, v_2, v_3, v_4\}$. Given below are the preference matrices containing the strict preference ordering of both entities over one another.*

In Table 1, a random sample preference matrix is given containing the preference lists for the four available sensors. The order is further depicted below where $v_j \succ_{l_i} v_k$ implies that the ith LiDAR sensor l_i prefers the jth VSP v_j over the kth VSP v_k, where $j \neq k$.

- $v_1 \succ_{l_1} v_3 \succ_{l_1} v_2 \succ_{l_1} v_4$
- $v_2 \succ_{l_2} v_4 \succ_{l_2} v_1 \succ_{l_2} v_3$
- $v_4 \succ_{l_3} v_1 \succ_{l_3} v_3 \succ_{l_3} v_2$
- $v_2 \succ_{l_4} v_1 \succ_{l_4} v_4 \succ_{l_4} v_3$

Similarly, in Table 2, a sample preference matrix is given containing the preference lists for the four available VSPs over the set of LiDAR sensors. The order is depicted below where $l_j \succ_{v_i} l_k$ implies that the ith VSP v_i prefers the jth LiDAR sensor l_j over the kth LiDAR l_k, where $j \neq k$.

- $l_3 \succ_{v_1} l_4 \succ_{v_1} l_2 \succ_{v_1} l_1$
- $l_2 \succ_{v_2} l_3 \succ_{v_2} l_4 \succ_{v_2} l_1$
- $l_4 \succ_{v_3} l_1 \succ_{v_3} l_3 \succ_{v_3} l_2$
- $l_4 \succ_{v_4} l_3 \succ_{v_4} l_1 \succ_{v_4} l_2$

Table 1. LiDAR preferences over VSPs.

l_1	l_2	l_3	l_4
v_1	v_2	v_4	v_2
v_3	v_4	v_1	v_1
v_2	v_1	v_3	v_4
v_4	v_3	v_2	v_3

Table 1 represents the preferences of the set of LiDAR sensors, $L = \{l_1, l_2, l_3, l_4\}$, over the set of available VSPs, $V = \{v_1, v_2, v_3, v_4\}$. The preferences are arranged in decreasing order from top to bottom.

Table 2. VSP preferences over LiDAR sensors.

v_1	v_2	v_3	v_4
l_3	l_2	l_4	l_4
l_4	l_3	l_1	l_3
l_2	l_4	l_3	l_1
l_1	l_1	l_2	l_2

Table 2 represents the preferences of the set of VSPs, $V = \{v_1, v_2, v_3, v_4\}$, over the set of available LiDAR sensors, $L = \{l_1, l_2, l_3, l_4\}$. The preferences are arranged in decreasing order from top to bottom.

Table 3 presents some of the terms commonly used in this subsection and what they denote.

Table 3. Symbols and notations.

Symbol	Description
L	Set of all LiDAR sensors, $\{l_1, l_2, \ldots\}$
l_i	Denotes a single LiDAR sensor from the set L
V	Set of all VSPs, $\{v_1, v_2, \ldots\}$
v_i	Denotes a single VSP from the set V
$\succ_{l_i}$	Strict preference ordering of the ith LiDAR l_i
$\succ_{v_i}$	Strict preference ordering of the ith VSP v_i

3.2. VSP–User Matching

The second stage of our framework closely represents an allocation model similar to that of a client to a server in a content delivery network (CDN). A one-to-one, single-sided matching is achieved based on the users' preference list which can be generated based on attributes such as the response time, net server up-time, etc.

Two disjoint sets containing the various virtual service providers (VSPs) and users constitute the overall model depicted here. We assume once again, that all the VSPs are homogeneous in nature and that the preference list provided by the users is strict and complete.

Formally, we define a set of size n as $V = \{v_1, v_2, \ldots, v_n\}$ denoting the set of all VSPs, where v_i represents the ith VSP in the set. We similarly define a set $U = \{u_1, u_2, \ldots, u_n\}$ denoting the set of all users, where u_i represents the ith user in the set. Both V and U are guaranteed to be disjoint and finite. Building upon the assumptions of the previous stage, we also assume that the services provided by each of the VSPs are homogeneous across every provider and are equal to the count of users under consideration.

In a practical scenario, however, it is highly unlikely that the number of VSPs or service providers would be equal to the number of users. In such as case, the proposed algorithm can be applied to the set of available VSPs and a small subset of users such that the count of VSPs is equal to the count of users in the subset under consideration.

Example 2. *Consider the finite sets of $n = 4$ elements V of VSPs containing the elements $\{v_1, v_2, v_3, v_4\}$ and U of users containing the elements $\{u_1, u_2, u_3, u_4\}$. Given below is the preference matrix containing the strict preferences of each and every user over the available VSPs. Since it is a single-sided matching, only the users' preferences matrix is considered during allocation.*

In Table 4, a random sample preference matrix was considered, containing the strict preference lists of each of the users over each of the four available VSPs. The order is

depicted below where $v_j \succ_{u_i} v_k$ implies that the ith user u_i prefers the jth VSP v_j over the kth VSP v_k, where $j \neq k$.

- $v_1 \succ_{l_1} v_3 \succ_{l_1} v_2 \succ_{l_1} v_4$
- $v_2 \succ_{l_2} v_4 \succ_{l_2} v_1 \succ_{l_2} v_3$
- $v_4 \succ_{l_3} v_1 \succ_{l_3} v_3 \succ_{l_3} v_2$
- $v_2 \succ_{l_4} v_1 \succ_{l_4} v_4 \succ_{l_4} v_3$

In Table 5, we define the various terms used in this subsection including their symbolic representations and what they stand for in the context of this paper.

Table 4. User preferences over VSPs.

u_1	u_2	u_3	u_4
v_4	v_1	v_3	v_1
v_2	v_3	v_2	v_4
v_3	v_2	v_1	v_2
v_1	v_4	v_4	v_3

Table 4 represents the strict preferences of the set of users $U = \{u_1, u_2, u_3, u_4\}$ over the set of available VSPs $V = \{v_1, v_2, v_3, v_4\}$. The preferences are arranged in decreasing order from top to bottom.

Table 5 presents some of the terms commonly used in this subsection and what they denote.

Table 5. Symbols and notations.

Symbol	Description
U	Set of all users, $\{u_1, u_2, \ldots\}$
u_i	Denotes a single user from the set U
V	Set of all VSPs, $\{v_1, v_2, \ldots\}$
v_i	Denotes a single VSP from the set V
$\succ_{u_i}$	Strict preference ordering of the ith user u_i

4. Proposed Model and Mechanism

After the unambiguous formulation of the system model, this section introduces our proposed stable matching mechanisms for the allocation of resources in both the LiDAR–VSP as well as the VSP–user game-theoretic environments. As discussed earlier, the allocation of resources in the LiDAR–VSP environment is based on a double-sided allocation scheme using the strict preferences provided by both entities over each other. On the other hand, allocation in the VSP–user environment is based on a single-sided matching mechanism using the strict preferences provided only by the set U of users over the available set V of VSPs. Both these mechanisms are proposed separately in the upcoming subsections and come together to form a single environment and achieve optimal performance in our simulations. During the discussion of the proposed mechanisms, the number of entities present in the disjoint sets is considered to be equal. Ways of handling cases where the sets are not of the same size are discussed under the System Model section.

4.1. LiDAR–VSP Double-Allocation Mechanism (LiV-DAM)

The proposed mechanism for the LiDAR–VSP allocation problem as mentioned earlier is a double-sided stable matching algorithm that aims to achieve an optimal allocation by assigning a unique service provider to every distinct LiDAR sensor and vice versa in a way that maximizes the performance of the network as well as the efficiency of the data

transfer and processing among the entities involved in order to obtain the best possible three-dimensional scans of the physical world and translate them to environments in the metaverse.

We assume that a total of n virtual service providers (VSPs) belonging to the set V share their respective preferences on the opposite set of entities (strict and complete), i.e., over the set of n LiDAR sensors belonging to the set L and similarly every LiDAR sensor provides its own strict preferences over all the available VSPs, thereby forming two preference matrices which are then used to obtain an optimal matching of resources in the system.

Initially, no allocations exist, i.e., no VSP is allocated a LiDAR sensor and no LiDAR sensor is allocated a VSP at the very beginning. Following the initialization of all the entities involved, the available set of LiDAR sensors $L : |L| = n \,\&\, L = \{l_1, l_2, \ldots, l_n\}$, the available set of VSPs $V : |V| = n \,\&\, V = \{v_1, v_2, \ldots, v_n\}$, the preference matrix formed by the strict preferences of each LiDAR sensor over the available set of VSPs $\succ_l = [\succ_{l_1}, \succ_{l_2}, \ldots, \succ_{l_n}]$, and the preference matrix formed by the strict preferences of each VSP over the available set of LiDAR sensors $\succ_v = [\succ_{v_1}, \succ_{v_2}, \ldots, \succ_{v_n}]$ are passed as input parameters to our proposed LiV-DAM algorithm in Algorithm 1. This algorithm represents our proposed allocation mechanism and outputs a list of allocated LiDAR–VSP pairs, $O_{LiV-DAM} = [(l_i, v_j), \ldots]$, where $l_i \in L, v_j \in V, 1 < i, j < n$.

Algorithm 1 LiDAR-VSP Double Allocation Mechanism (LiV-DAM)

Input: $L = \{l_1, l_2, \ldots, l_n\}$; $V = \{v_1, v_2, \ldots, v_n\}$; $\succ_l = [\succ_{l_1}, \succ_{l_2}, \ldots, \succ_{l_n}]$; $\succ_v = [\succ_{v_1}, \succ_{v_2}, \ldots, \succ_{v_n}]$
Output: $O_{LiV-DAM} \leftarrow \phi$
begin
$L' = L$
for each $i \in L$ **do**
  $O_{LiV-DAM}[i] \leftarrow \phi$
end for
for each $i \in V$ **do**
  $\overline{O}_{LiV-DAM}[i] \leftarrow \phi$
end for
while $L' \neq \phi$ **do**
  $l^* \leftarrow rand\_pick(L')$
  $v^* \leftarrow extract(\succ_{u^*})$
  if $\overline{O}_{LiV-DAM}[v^*] == \phi$ **then**
    $\overline{O}_{LiV-DAM}[v^*] \leftarrow l^*$
    $L' \leftarrow L' \setminus \{l^*\}$
  else
    $\mathring{l}^* \leftarrow \overline{O}_{LiV-DAM}[v^*]$
    if $l^* \succ_{v^*} \mathring{l}^*$ **then**
      $O_{LiV-DAM}[s^*] \leftarrow l^*$
      $L' \leftarrow L' \setminus \{l^*\}$
      $L' \leftarrow L' \cup \{\mathring{l}^*\}$
    end if
  end if
end while
return $O_{LiV-DAM}$
end

- **Initialization:** In the first stage of the algorithm, the initial allocations are set to **NULL** or ϕ denoting that no allocations exist between the LiDAR sensors and VSPs at the beginning. A copy of the set of LiDAR sensors is maintained in a new variable in order to make sure that the original one is not modified or overwritten as the algorithm progresses.

- **Random allocation:** The allocation phase of the LiV-DAM algorithm stores data about which services have been allocated to which LiDAR. The pseudocode makes it clear that the termination of the while loop happens when there are no more LiDAR sensors left to allocate services to. One of the LiDAR sensors is then selected randomly from the temporary set of available LiDAR sensors L' and stored in l^*. This initial random allocation undergoes reallocations as the algorithm progresses.
- **LiV-DAM allocation:** After the completion of the random allocation phase, the $extract()$ function is then called in order to retrieve the most preferred service using the preference profile of each LiDAR $l^* \in L'$ and is stored in v^*. If the check inside the if statement present inside the while loop evaluates to false, this means that a particular LiDAR sensor has multiple proposed VSPs available. This competitive environment is resolved via a check made in the else section of the outermost **if–else** block based on the preference ordering of the LiDAR sensors. Once this conflict is resolved, the final allocation is held in the output data structure $O_{LiV-DAM}$, which is then returned after completion of the algorithm.

Example 3. *The functioning of the LiV-DAM algorithm is explained in detail and depicted in Figure 4, where the number of LiDAR sensors is assumed to be equal to the count of VSPs with* $n = 5$.

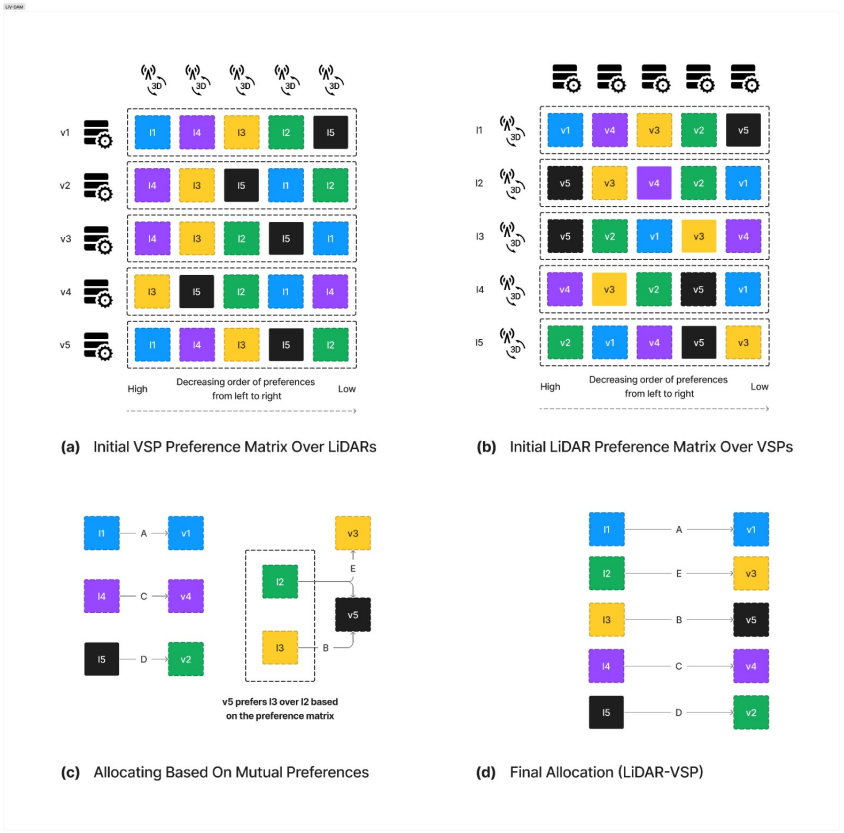

Figure 4. The detailed functioning of the LiV-DAM algorithm (Algorithm 1) where the number of LiDAR sensors is equal to the number of VSPs ($n = 5$).

The profile consisting of strict preferences given by the set of LiDAR sensors L and the set of available VSPs V is depicted in Figure 4. It is to be noted that, in our case, LiDAR sensors requested available virtual service providers (VSPs). Following the pseudocode for our algorithm, we randomly selected LiDAR l_1 and temporarily allocated it to the VSP v_1. In a similar fashion, LiDAR l_3 was randomly picked and allocated to VSP v_5. The next time, LiDAR l_4 was chosen and then matched to VSP v_4, and lastly LiDAR l_5 was allocated

the VSP v_2. Thus, each of the LiDAR sensors l_1, l_2, l_3, and l_4 requested their most-preferred VSPs from their profiles consisting of strict and complete preferences. After this, it was checked whether any VSP among v_1, v_2, v_3, v_4, and v_5 received multiple requests from the set of LiDAR sensors L. Now, it can be seen that in the first iteration of LiV-DAM VSP, v_5 received a request from LiDAR sensors l_2 and l_3. As each VSP could be assigned to only one LiDAR sensor (one-to-one mapping), the competition between LiDAR l_2 and l_3 underwent resolution by once again leveraging the strict and complete preference profile of the VSP under consideration, i.e., v_5. From the preference profile provided by the service provider v_5, it can be seen that LiDAR l_3 was preferred when compared to LiDAR l_2. Hence, the VSP rejected LiDAR l_2's offer. Now, as the LiDAR l_2 did not get its most preferred VSP, i.e., v_5, from its preference profile, LiDAR l_2 requested the next most-preferred VSP, i.e., v_3 based on its preference profile ordering which was available and hence was matched. It can now be seen that all the LiDAR-VSP pairs were allocated, and the allocation could be proven to be stable as there did not exist any blocking pairs.

Time-complexity analysis: In LiV-DAM, we consider the presence of n entities of either type in the form of two separate disjoint sets thereby resulting in a time-complexity analysis in terms of n. The worst-case scenario causing the maximum time complexity for the proposed algorithm occurs when the random allocation yields an initial mapping in which all the LiDAR sensors are mapped to their least preferred choice of virtual service provider (VSP). This would require the algorithm to make a total of n iterations for each LiDAR sensor until they reach the most preferred optimal allocation. Therefore, the worst-case time complexity of LiV-DAM is when we make a total of $|L| * |V| = n * n = n^2$ iterations thereby resulting in a time complexity of $O(n^2)$.

4.2. VSP–User Single-Allocation Mechanism (VU-SAM)

The allocation mechanism used in the second part of our proposed framework was termed VU-SAM (VSP-user single-allocation mechanism). It is a truthful mechanism that assumes an initial random one-to-one allocation of VSPs to users. Once again, based on our assumptions that the count of users is equal to the count of virtual service providers (VSPs), it is guaranteed that a one-to-one allocation of distinct VSPs to correspondingly distinct users is possible. The proposed algorithm then either lets some allocations stay as is or reallocates particular VSPs to particular users based on the preference matrix formed by listing out the preference list of every user over every available VSP.

The proposed allocation mechanism/algorithm in Algorithm 2 takes in the set of users $U = \{u_1, u_2, \ldots, u_n\}$, the set of available VSPs $V = \{v_1, v_2, \ldots, v_n\}$, and the preference matrix formed by the strict preference lists of all the users over each of the available VSPs $\succ_u = [\succ_{u_1}, \succ_{u_2}, \ldots, \succ_{u_n}]$ and outputs an allocation list containing user–VSP pairs, $O_{VU-SAM} = [(u_i, v_j), \ldots]$, where $u_i \in U, v_j \in V, 1 < i, j < n$.

The entire algorithm has been divided into 5 major components which are outlined and elaborated following the pseudocode.

Algorithm 2 VSP-User Single-Allocation Mechanism (VU-SAM)

Input: $U = \{u_1, u_2, \ldots, u_n\}$; $V = \{v_1, v_2, \ldots, v_n\}$; $\succ_u = [\succ_{u_1}, \succ_{u_2}, \ldots, \succ_{u_n}]$
Output: $O_{VU-SAM} \leftarrow \phi$
begin
Initialize $O_{VU-SAM}[i] = 0 \; \forall \, i \in \{1, 2, \ldots, n\}$
$A \leftarrow U$
$E \leftarrow \{u_1, u_2, \ldots, u_n, v_1, v_2, \ldots, v_n\}$
for each $u_i \in U$ **do**
  $O_{VU-SAM}[i] \leftarrow rand(distinctj : v_j \in V)$
end for
repeat
  $G^d \leftarrow generate\_digraph(\succ_u, E, O_{VU-SAM})$
  $A_c \leftarrow detect\_cycle(G^d)$
  for all $(u_i, v_j) \in A_c$ **do**
    $O_{VU-SAM}[i] \leftarrow j$
    $A \leftarrow A \setminus \{u_i\}$
    $V \leftarrow V \setminus \{v_j\}$
    $E \leftarrow E \setminus \{u_i, v_j\}$
  end for
  $update\_profile(\succ_u)$
until $(A \neq \phi)$ **return** O_{VU-SAM}
end

Explanation of various steps in VU-SAM:

- **Initialization:** The algorithm commences its execution with the setting of the entire allocation list to **NULL** or ϕ. This denotes that at the very beginning, no allocations exist between any user and VSP or vice versa. A copy of the set of users U is maintained in a new variable A. The purpose of this redundant copy will become clear shortly. All of the list positions in O_{VU-SAM} are set to 0 in order to denote the same. In the graph G^d, all the users as well as the VSPs are denoted by nodes or vertices connected by a finite number of edges. All these nodes are stored in the entities list E during the execution of the algorithm. After the initial setup of all the required variables, the algorithm then randomly allocates a virtual service provider (VSP) $v_j \in V$ to each available user $u_i \in U$.
- **DiGraph creation:** After the random allocation of VSP v_j to a user u_i, inside the do–while (repeat–until) loop, a call to the function $generate\_digraph()$ is initiated. The preference matrix formed by the strict and complete preference ordering of the users $\succ_u = [\succ_{u_1}, \succ_{u_2}, \ldots, \succ_{u_n}]$, the set of all nodes belonging to the graph $E \in G^d$, and the initially allocated random configuration of user–VSP pairs are passed during the function call.
  At the very beginning of the $generate\_digraph()$ function, the selection of edges is carried out from the initial random allocation connecting a user $u_i \in U$ to a VSP $v_j \in V$, given in O_{VU-SAM} and adding them to the graph G^d. The second stage of the function then selects edges connecting user $u_i \in U$ to VSP $v_j \in V$ based on the users' preference profile $\succ_{u_i}$. This digraph is then stored in G^d for further processing and manipulation.
- **Cycle detection:** The DiGraph (directed graph) obtained in the previous step, G^d, is then passed to the $detect\_cycle()$ function in order to detect cycles of finite length representing cyclic dependencies which can be corrected via mutual exchange in order to obtain an optimal allocation. A random vertex $x \in G^d$ is first selected, and outgoing edges are traced until the repetition of a vertex is encountered, thereby indicating the presence a cycle C^d. Upon detection of such a cycle, the VSPs are reallocated to the users by mutually exchanging their previous allocations.

- **Removing allocated entities:** After the detection of the finite directed cycle C^d, all the vertices $x \in C^d$ are removed from A: $A \leftarrow A \setminus \{u_i\}$), V: $V \leftarrow V \setminus \{v_j\}$) as well as the complete vertex set E: $E \leftarrow E \setminus \{u_i, v_j\}$).
- **Profile update:** The $update\_profile()$ function is then passed the users' strict preference matrix $\succ_u = [\succ_{u_1}, \succ_{u_2}, \ldots, \succ_{u_n}]$. The preference matrix $\succ_u$, set of entities E, temporary copy of the set of users A, and the set of available VSPs V are updated in this phase, and new allocations are made based on preferences. This process repeats itself until $A \neq \phi$ i.e., until all the users are allocated a distinct VSP according to their preference profiles.

Example 4. *The details of the VU-SAM algorithm and its functioning are elaborated and illustrated in Figure 5, where once again, similar to the case considered in LiV-DAM, the numbers of VSPs and users were assumed to be equal to* $n = 5$.

The strict and complete preference profiles given by the set of users U is presented. Following the execution of the algorithm from the provided pseudo code, a random directed edge representing the initial allocation was created mapping the following pairs to each other $(VSP, User)$ pair: (v_3, u_1), (v_5, u_2), (v_1, u_3), (v_2, u_4), and (v_4, u_5). Now, upon the execution of the $create\ digraph()$ function, let us say that a user u_1 was selected from the list of non-allocated users and offered a connection request to its most preferred VSP. For user u_1, based on the preference profile, this would be denoted by an offer to VSP v_4. A directed edge between (v_4, u_1) was therefore placed. This process was repeated one by one for all the users, choosing the next most-preferred and available VSP and resulted in the creation of directed edges between all the user–VSP pairs. The $detect\_cycle()$ function was then called and cycles $\{u_2, v_1, u_3, v_5\}$ and $\{u_1, v_4, u_5, v_3\}$ were detected, removed, and underwent a mutual exchange in order to achieve optimal allocations. The final state of allocations can be seen at the end of Figure 5.

Time-complexity analysis: In VU-SAM, we generalized the number of edges and vertices to be equal, which therefore allowed us to analyze the time complexity based solely on the count of vertices in the graph. During initialization, the for-loop used takes $O(V^2)$ time, where V denotes the count of vertices in the graph. The outer for-loop loops over the set of all users once and the $rand()$ function takes $O(V)$ time to generate a random allocation. Similarly, the creation of the digraph requires the algorithm to go through all the V vertices and the adjacency list against each vertex may have at most V vertices, thereby bringing the time complexity to $O(V^2)$. The cycle detection section of the algorithm on the other hand, takes a linear time O(V), as it is based on the concept of DFS and linearly visits every vertex. The **repeat–until** section consists of a for loop that is executed a total of V times (assuming the count of edges is same as the count of vertices). The update profile function generally involves removing the allocated vertices from the users' preference matrix, which calls for the manipulation of a two-dimensional List and hence an $O(V^2)$ time complexity. Every iteration would result in the removal of a user–VSP pair, thereby effectively removing two vertices from the graph. Therefore, a total of $O(V)$ iterations are possible. This, along with the $O(V^2)$ time complexity of the $update\_profile()$ section, brings the algorithm's total time complexity to $O(V^3)$. Therefore, the proposed VU-SAM algorithm exhibits a time complexity of $O(V^3)$.

Figure 5. The detailed functioning of the VU-SAM algorithm (Algorithm 2) where the number of users is equal to the number of VSPs ($n = 5$).

5. Lemmas, Propositions and Proofs

Lemma 1. *The maximum number of offers extended during runtime of the LiV-DAM algorithm is always less than or equal to* n^2.

Proof. Every offer (possible allocation) from a LiDAR to a VSP and vice versa is represented by a directed edge in the DiGraph formed during the execution of the algorithm. Since the count of LiDAR sensors is n and similarly, the count of virtual service providers (VSPs) is also n, the maximum number of possible offers can be depicted by a complete graph, containing n^2 edges and therefore n^2 offers. □

Proposition 1. *The LiV-DAM algorithm is stable by design.*

Proof. If the LiV-DAM algorithm were not stable, there would exist an arbitrary LiDAR sensor $l_i \in L$ that would be assigned to a VSP $v_j \in V$ where $v_i \in V$ would be its most preferred option according to its preference profile over the set of all available VSPs, i.e., $v_i \succ_{l_i} v_j$ even though v_i would still be available for allocation to l_i. During the execution of the LiV-DAM algorithm, LiDAR l_i offers its data to VSPs in the order of decreasing preferences in its preference profile. This implies that at any point in time, the offer extended by the LiDAR will always be to its next most preferred VSP. If v_i is more preferred than v_j according to the preference profile of LiDAR l_i and has not yet been rejected (allocated to some other LiDAR sensor), it implies that v_i occurs before v_j in the preference list of LiDAR l_i, thereby making it impossible for v_j to be allocated, provided v_i is still available. Hence,

we can conclude that there exists no pair (l_i, v_j) $l_i \in L$ and $v_j \in V$ such that they prefer each other but are still not allocated or assigned. Therefore, LiV-DAM is stable. □

Proposition 2. *A perfect matching is returned by the LiV-DAM algorithm between the set of all LiDAR sensors L and the set of all available VSPs V.*

Proof. Let us assume that the returned set of allocations A is not perfect. In such a case, there must exist at least one unallocated LiDAR sensor l_i. According to the algorithm, this LiDAR sensor would have expressed its interest to every available VSP due to its strict preference profile over each and every service provider, before termination of the algorithm. Moreover, every VSP receives at least one proposal (the first random allocation), which implies that all of the VSPs are from the initialization stage, allocated to a distinct LiDAR. Therefore, this further implies that all n VSPs are mapped, and hence by virtue of the one-to-one mapping there should exist n allocated LiDAR sensors. This contradicts our initial assumption that at least one of the LiDAR sensors remains unallocated, thereby proving by the method of contradiction that the returned allocation A is a perfect matching. □

Proposition 3. *The allocations returned after the execution of the LiV-DAM algorithm are stable in nature but do not represent the core allocation.*

Proof. Let A denote the set of all allocations made during the runtime of the algorithm which is not stable and is coalition-dominated, made up of pairs or standalone independent entities. Therefore, no allocations exist in the game's core that can be unstable. On the other hand, we know that the set of all allocations that are non-dominated by coalitions of arbitrary size is referred to as the core; therefore, the stable allocations or stable mappings might result in a strict containment of the core. However, in the present scenario involving a one-to-one matching, the case differs and hence does not represent the core. □

Proposition 4. *A dominant-strategy incentive-compatible (DSIC) characteristic is depicted by the VU-SAM algorithm.*

Proof. The VU-SAM algorithm's incentive-compatible (IC) nature has its foundation in the fact that all users $u_i \in U$ are matched to the most preferred virtual service provider (VSP) based on the preference profiles reported by them over the available VSPs, $v_j \in V$. Suppose A_j is the agent or service provider mapped during the jth iteration, considering that the involved entities do not misreport. Users in A_1, i.e., the first iteration, get their most-preferred VSP and hence have no incentive to misreport. A user $a_i \in A_2$ is not pointed to by any agent $a_j \in A_1$ in the first iteration, otherwise a_i would have been allocated during A_1 rather than A_2. Since a_i is allocated its most preferred choice apart from the virtual service providers (VSPs) already allocated during A_1, misreporting does not provide any sort of benefit, i.e., there exists no incentive to do so. Speaking based on generalization, an agent or VSP a_i of A_j is never allocated during the first $j-1$ cycles of VU-SAM to any user in $A_1 \cup A_2 \cup \cdots \cup A_{j-1}$. Thus, $a_i \in A_j$ cannot be allocated to a service provider that has already been allocated to a user in $A_1 \cup A_2 \cup \cdots \cup A_{j-1}$. Since, VU-SAM results in the matching of the most preferred provider to i , there exists no incentive to misreport. Hence, it is DSIC. □

Proposition 5. *A unique core allocation is returned upon the execution of the VU-SAM algorithm.*

Proof. Consider a subset U^* of users, in order to prove that the VU-SAM algorithm returns a core allocation. □

The user in the ith cycle is denoted by $\Pi(U_i)$. Consider l to be the iteration in which $\Pi(U_l) \cap U^* \neq \phi$, with users $u_l \in U^*$ receiving its service provider for the first time. The

algorithm allocates user u_l to its most preferred virtual service provider (VSP) outside of the VSPs allocated to other users in $U_1 \cup U_2 \cup \cdots \cup U_{l-1}$.

As no $i \in U^* \notin U_1 \cup U_2 \cup \ldots U_{l-1}$, there exists no configuration or mapping in which a reallocation of the VSPs can make i strictly better off. In the VU-SAM algorithm, all users receive their most-preferred choice. Any core allocation must hold true and abide by this property.

Without abiding by this property, an allocation would implicitly denote that the users of U_1 who did not get their first preference would tend towards the formation of a coalition following which a mutual reallocation could result in everyone being strictly better off.

In the same manner, all the users of U_2 are allocated to their most preferred VSP outside of U_1. Since it is known that agreement is guaranteed in every core allocation with respect to the allocations generated for the users of U_1, such an agreement must also exist for the users of U_2, in the absence of which the users of U_2 that were unable to get allocated to their most-preferred choice outside of U_1 can collectively reach a better state via a mutual reallocation. Therefore, conclusively, we can say by induction that the proposed VU-SAM algorithm returns a core allocation that is unique.

6. Experimental Results

This section summarizes the results of simulations that were carried out using sample data points generated by random probability distributions, benchmarked against the performance of the LiV-DAM as well as the VU-SAM algorithms. We compared both our proposed algorithms to the performance of the random-distribution-based allocations, hereafter referred to as random assignment. These simulations were coded and run on a Ubuntu-based machine in Python 3.

6.1. Simulation Setup, Initialization, and Parameters

The simulations depicted below were carried out considering a set of n LiDAR sensors, $l_i \in L, 1 < i < n$ and n virtual service providers (VSPs), $v_i \in V, 1 < i < n$ for simulating the LiV-DAM algorithm. For simulating the VU-SAM algorithm, we once again considered a set of n virtual service providers (VSPs) and a set of n users, $u_i \in U, 1 < i < n$. In the case of the double-sided, LiV-DAM algorithm, all the n LiDAR sensors provided a strict and total preference ordering over the set of VSPs and similarly each of the n VSPs provided their own strict and total preference ordering over the set of all available LiDAR sensors. On the other hand, in the case of the VU-SAM simulations, the preference ordering/preference profile was only provided by the user over the set of available VSPs, by virtue of its single-sided nature. During the course of these simulations, we plotted the preference indices (how preferable the allocations were) obtained from the random assignment algorithm vs. our proposed algorithms.

6.2. Benchmarking: LiV-DAM vs. Random Assignment

We compared the ranks (or preference indices) of the allocated service provider (based on the preference matrix provided by the LiDAR sensors) for individual LiDAR sensors for both algorithms, namely, LiV-DAM and random assignment on the same game instance (i.e., while keeping the parameters constant) and set of entities. In the presented example, the LiDAR sensors were numbered starting from 0 to $n-1$, $L = \{l_0, l_1, \ldots, l_{n-1}\}$ similar to the VSPs that were also numbered from 0 to $n-1$, $V = \{v_0, v_1, \ldots, v_{n-1}\}$. It can be clearly seen in Figure 6 that the scatter plot of magenta-colored pentagons, representing the allocations obtained via LiV-DAM, are much closer to the x-axis. This implies that the resulting LiDAR allocations mapping them to VSPs were extremely close to their preference ordering without a large margin of deviation from their reported preference profiles, i.e., the preference indices or ranks of the allocated VSPs were low indicating a high preference. The yellow stars, which represent the allocations using random assignment, are much more scattered and distributed throughout the x–y plane thereby denoting that the ranks of the allocated VSPs were high, indicating a lower preference by the LiDAR sensors. This

observation begs the conclusion that our proposed LiV-DAM Algorithm had a significantly better performance when benchmarked against the random assignment mechanism on the same set of entities and simulation instance.

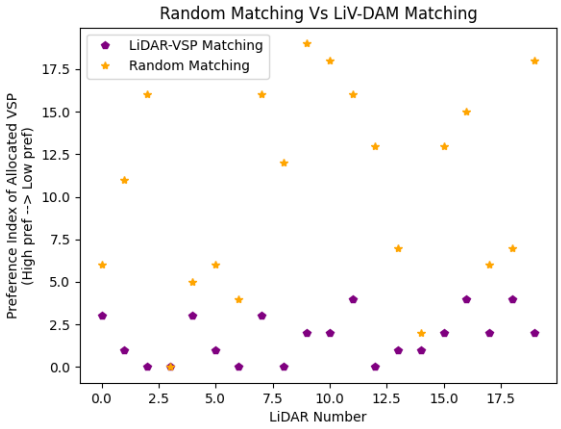

Figure 6. Two-dimensional plot of LiDAR sensors vs. the preference index of the VSP that is allocated to them.

We took this analysis one step further by plotting the three-dimensional scatter plot (Figure 7) in order to factor in not just the preference indices, but also the number of LiDAR sensors which were allocated such preference indices. Once again from the three-dimensional point cloud, we can see that a lot of allocations made by the LiV-DAM algorithm had low indices. On the other hand, the scattered nature of allocations obtained by the random allocation algorithm caused a distributed and nonfrequently occurring distribution of preference indices. In almost all cases, the ranks of the allocated VSPs were lower in LiV-DAM than that obtained in the allocations returned by random assignment, which further verified that LiV-DAM was a much better allocation algorithm when compared to random assignment.

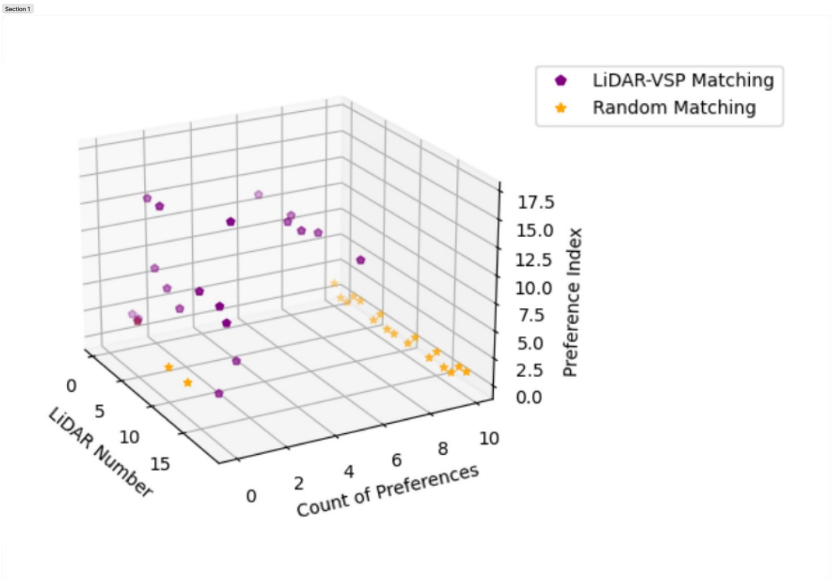

Figure 7. A three-dimensional plot between LiDAR sensors and the preference index of allocated VSP while also factoring in and showing the frequency with which a particular preference index is chosen.

6.3. Benchmarking: VU-SAM vs. Random Assignment

Similar to the benchmarking of the LiV-DAM algorithm against random assignment/allocation, in this case also, we compared the ranks (or preference indices) of the allocated virtual service providers (VSPs) (based on the preference profile reported by the set of users over all the available VSPs) to individual users, for both algorithms, namely, VU-SAM and random assignment on the same game instance (i.e., while keeping the parameters constant) and set of entities. In our example, the users were denoted by integers from 0 to $n-1$, $U = \{u_0, u_1, \ldots, u_{n-1}\}$. Similarly, once again, the VSPs were also numbered in a similar fashion from 0 to $n-1$, $V = \{v_0, v_1, \ldots, v_n\}$. It can be seen from Figure 8 that once again, the scatter plot of pink-colored squares, representing the allocations obtained after the application of VU-SAM, are closer to the x-axis. This implies that the user allocations to VSPs lay extremely close to their preference ordering and did not deviate much with respect to their preference profiles, i.e., the preference indices of the allocated VSPs were significantly lower indicating a high preference from the users. The green hexagons, on the other hand, represent the allocations obtained via random assignment and are much more scattered and distributed throughout the x–y plane, thereby denoting that the ranks of the allocated VSPs were high, indicating a lower preference by the users. This observation once again begs the conclusion that the performance of the VU-SAM Algorithm far surpassed that of the random assignment mechanism for the same set of entities and simulation instances.

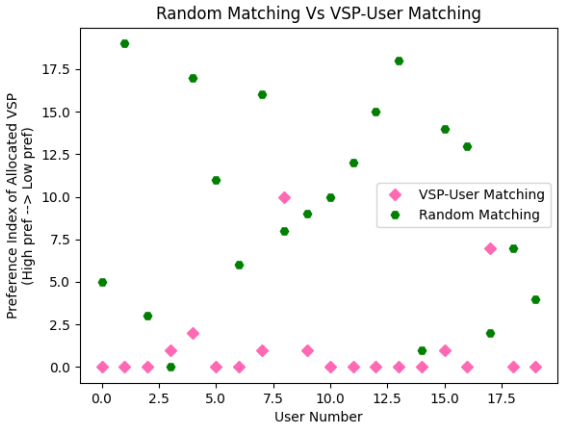

Figure 8. Two-dimensional plot of users vs. the preference index of the VSP that is allocated to them.

We once again obtained a three-dimensional scatter plot (Figure 9) in order to factor in the number of users who were allocated a particular preference index. From the three-dimensional point cloud, it can be clearly seen that a lot of allocations made by the VU-SAM algorithm had low indices. Whereas, on the other hand, the scattered nature of allocations obtained by the random allocation algorithm caused a distributed and not frequently occurring distribution of preference indices. In almost all cases, the ranks of the allocated VSPs were much lower in VU-SAM than that those obtained in the allocations returned by random assignment, further verifying that VU-SAM was a much better allocation algorithm when compared to random assignment.

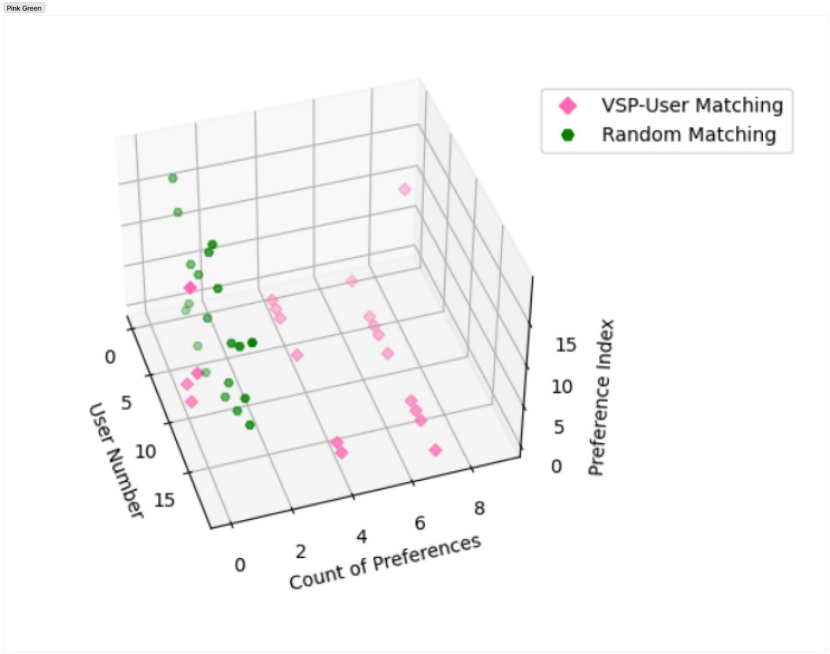

Figure 9. A three-dimensional plot between users and the preference index of allocated VSP while also factoring in and showing the frequency with which a particular preference index is chosen.

7. Conclusions and Future Work

In this paper, we aimed towards establishing and verifying a possible allocation mechanism in a game-theoretic environment between LiDAR sensors (enabling the creation of digital twins), virtual service providers or VSPs (allowing for the processing and proper distribution of data across markets and users), and users (individuals and organizations in the metaverse who leverage services provided by the VSPs) in a free market where VSPs are incentivized by the long-term goal of market capture. We also benchmarked our proposed algorithm against the random assignment of resources, thereby conclusively proving the improvement that our mechanism introduced.

The proposed mechanism here can be extended to other use cases, some of which, such as gene expression, association, signature detection, and drug discovery, was highlighted in the literature review as well. The proposed system design for such a use case can be obtained by simply replacing the LiDAR sensors with their analogous gene sequencer counterparts. The allocation mechanisms even though they would now be applied in a completely different context, would remain constant and enable the efficient allocation of computing resources to the user-defined tasks divided across multiple service providers enabling faster throughput rates and analysis of biological data.

The current model lacks an in-depth monetary analysis of the system, which would play a major factor in a real-life implementation based on which, preferences across different classes of entities will be calculated.

In our future work, we will dive deeper and chalk out the finer aspects and details of enabling a distributed digital-twin-based ecosystem (DDTE) in the metaverse as well as the introduction of economic activities with money as an incentive, further generalizing the cyberspace markets. Our future case studies and simulations will also focus on mapping real-world datasets to our proposed mechanisms and obtaining conclusive real-life numbers as well as the possibility of commerce in a digital marketplace powered by decentralized and distributed technologies, while also diving more into the healthcare or bioinformatics use cases that this mechanism enables. The division of tasks across servers and the efficient allocation of computing resources will be studied in greater depth in a manner that encompasses a multiuse case ecosystem. Moreover, in the future, we will apply our proposed metaverse-based approach in potential drug discovery and molecular

docking as well as medical surgery, which would be mainly beneficial for pharmaceutic and healthcare applications.

Author Contributions: Conceptualization, methodology, and validation: A.B., A.S., S.S. and D.B.; formal analysis and investigation: A.B., A.S., A.E.H., S.M. and H.Q.; resources and writing—review: A.B., A.S., A.E.H., S.M., A.L. and H.Q.; writing—review and editing: A.E.H., S.M., A.L. and H.Q. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Informed Consent Statement: Not applicable.

Data Availability Statement: The supporting data related to this study is openly available at https://github.com/Ansh-Sarkar/LiV_DAM-And-VU_SAM-Simulations (accessed on 6 March 2023).

Acknowledgments: The authors would like to thank all research team members of the connected institutes and universities for their valuable suggestions to improve the standard of the article.

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

References

1. Zheng, J.; Chan, K.; Gibson, I. Virtual reality. *IEEE Potentials* **1998**, *17*, 20–23. [CrossRef]
2. Carmigniani, J.; Furht, B. Augmented reality: An overview. In *Handbook of Augmented Reality*; Springer: Berlin/Heidelberg, Germany, 2011; pp. 3–46.
3. Bandyopadhyay, A.; Mukhopadhyay, S.; Ganguly, U. Allocating resources in cloud computing when users have strict preferences. In Proceedings of the IEEE International Conference on Advances in Computing, Communications and Informatics (ICACCI), Jaipur, India, 21–24 September 2016; pp. 2324–2328.
4. Bandyopadhyay, A.; Mukhopadhyay, S.; Ganguly, U. On free of cost service distribution in cloud computing. In Proceedings of the IEEE International Conference on Advances in Computing, Communications and Informatics (ICACCI), Udupi, India, 13–16 September 2017; pp. 1974–1980.
5. Bansal, G.; Karthik, R.; Chamola, V.; Xiong, Z.; Niyato, D. Healthcare in Metaverse: A Survey on Current Metaverse Applications in Healthcare. *IEEE Access* **2022**, *10*, 119914–119946. [CrossRef]
6. Dai, X.; Spasic, I.; Meyer, B.; Chapman, S.; Andres, F. Machine learning on mobile: An on-device inference app for skin cancer detection. In Proceedings of the IEEE Fourth International Conference on Fog and Mobile Edge Computing (FMEC), Rome, Italy, 10–13 June 2019; pp. 301–305.
7. Sahoo, K.K.; Ghosh, R.; Mallik, S.; Roy, A.; Singh, P.K.; Zhao, Z. Wrapper-based deep feature optimization for activity recognition in the wearable sensor networks of healthcare systems. *Sci. Rep.* **2023**, *13*, 965. [CrossRef] [PubMed]
8. Saladi, S.; Karuna, Y.; Koppu, S.; Reddy, G.R.; Mohan, S.; Mallik, S.; Qin, H. Segmentation and Analysis Emphasizing Neonatal MRI Brain Images Using Machine Learning Techniques. *Mathematics* **2023**, *11*, 285. [CrossRef]
9. Bora, K.; Mahanta, B.L.; Borah, K.; Chyrmang, G.; Barua, B.; Mallik, S.; Das, H.S.; Zhao, Z. Machine Learning Based Approach for Automated Cervical Dysplasia Detection Using Multi-Resolution Transform Domain Features. *Mathematics* **2022**, *10*, 4126. [CrossRef]
10. Zong, N.; Ngo, V. Computational Drug Target Prediction: Benchmark and Experiments. In Proceedings of the 2022 IEEE 10th International Conference on Healthcare Informatics (ICHI), Rochester, MN, USA, 11–14 June 2022; pp. 559–560. [CrossRef]
11. Nordsletten, D.A.; Yankama, B.; Umeton, R.; Ayyadurai, V.A.S.; Dewey, C.F. Multiscale Mathematical Modeling to Support Drug Development. *IEEE Trans. Biomed. Eng.* **2011**, *58*, 3508–3512. [CrossRef] [PubMed]
12. Unni, A.P.; Sreekumar, A.; Balakrishnan, K. Disease Based Computational Drug Repurposing: A Review. In Proceedings of the 2021 5th International Conference on Information Systems and Computer Networks (ISCON), Mathura, India, 22–23 October 2021; pp. 1–6. [CrossRef]
13. Wu, H.-C.; Wei, X.-G.; Chan, S.-C. Novel Consensus Gene Selection Criteria for Distributed GPU Partial Least Squares-Based Gene Microarray Analysis in Diffused Large B Cell Lymphoma (DLBCL) and Related Findings. *IEEE ACM Trans. Comput. Biol. Bioinform.* **2018**, *15*, 2039–2052. [CrossRef]
14. Bolón-Canedo, V.; Sechidis, K.; Sánchez-Maroño, N.; Alonso-Betanzos, A.; Brown, G. Exploring the consequences of distributed feature selection in DNA microarray data. In Proceedings of the 2017 International Joint Conference on Neural Networks (IJCNN), Anchorage, AK, USA, 14–19 May 2017; pp. 1665–1672. [CrossRef]
15. Chen, I. A Novel and Fast Distributed Computation Method for Fisher's Exact Test and Its Application in Gene Expression Profiling Studies. In Proceedings of the 2022 IEEE MIT Undergraduate Research Technology Conference (URTC), Cambridge, MA, USA, 2022; pp. 1–3. [CrossRef]
16. Vecchiola, C.; Abedini, M.; Kirley, M.; Chu, X.; Buyya, R. Gene Expression Classification with a Novel Coevolutionary Based Learning Classifier System on Public Clouds. In Proceedings of the 2010 Sixth IEEE International Conference on e-Science Workshops, Brisbane, QLD, Australia, 7–10 December 2010; pp. 92–97. [CrossRef]

17. Wang, G.; Badal, A.; Jia, X.; Maltz, J.S.; Mueller, K.; Myers, K.J.; Niu, C.; Vannier, M.; Yan, P.; Yu, Z.; et al. Development of metaverse for intelligent healthcare. *Nat. Mach. Intell.* **2022**, *4*, 922–929. [CrossRef]
18. Petrigna, L.; Musumeci, G. The metaverse: A new challenge for the healthcare system: A scoping review. *J. Funct. Morphol. Kinesiol.* **2022**, *7*, 63. [CrossRef] [PubMed]
19. Taylor, S.; Soneji, S. Bioinformatics and the metaverse: Are we ready? *Front. Bioinform.* **2022**, *2*, 863676. [CrossRef] [PubMed]
20. Garavand, A.; Nasim, A. Metaverse phenomenon and its impact on health: A scoping review. *Inform. Med. Unlocked* **2022**, *32*, 101029. [CrossRef]
21. Joshua, J. Information bodies: Computational anxiety in neal stephenson's snow crash. *Interdiscip. Lit. Stud.* **2017**, *19*, 17–47. [CrossRef]
22. Bruun, A.; Stentoft, M.L. Lifelogging in the wild: Participant experiences of using lifelogging as a research tool. In *IFIP Conference on Human Computer Interaction*; Springer: Berlin/Heidelberg, Germany, 2019; pp. 431–451.
23. Burns, W., III. Everything You Know about the Metaverse is Wrong. 2018. Available online: https://www.linkedin.com/pulse/everything-you-know-metaverse-wrong-william-burns-iii/ (accessed on 15 January 2023).
24. Chayka, K. *Facebook Wants Us to Live in the Metaverse*; The New Yorker: New York, NY, USA, 2021.
25. Kevin, K.; David, G. Ar Will Spark the Next Big Tech Platform—Call It Mirror-World. Available online: https://fabricofdigitallife.com/Detail/objects/3649/ (accessed on 12 October 2022).
26. Chen, B.; Song, C.; Lin, B.; Xu, X.; Tang, R.; Lin, Y.; Yao, Y.; Timoney, J.; Bi, T. A cross-platform metaverse data management system. In Proceedings of the 2022 IEEE International Conference on Metrology for Extended Reality, Artificial Intelligence and Neural Engineering (MetroXRAINE), Rome, Italy, 26–28 October 2022; IEEE: Piscataway, NJ, USA, 2022; pp. 145–150.
27. Watch Blockeley, uc Berkeley's Online Minecraft Commencement. Available online: https://news.berkeley.edu/2020/05/16/watch-blockeley-uc-berkeleys-online-minecraftcommencement/(accessed on 12 October 2022).
28. Fortnite's Marshmello Concert Was the Game's Biggest Event Ever-the Verge. Available online: https://www.theverge.com/2019/2/21/18234980/fortnite-marshmello-concert-viewer-numbers (accessed on 12 October 2022).
29. Lee, L.-H.; Braud, T.; Zhou, P.; Wang, L.; Xu, D.; Lin, Z.; Kumar, A.; Bermejo, C.; Hui, P. All one needs to know about metaverse: A complete survey on technological singularity, virtual ecosystem, and research agenda. *arXiv* **2021**, arXiv:2110.05352.
30. Shurvell, J. Take a Virtual African Safari in 2020 in Real-Time from Your Sofa. Available online: https://www.forbes.com/sites/joanneshurvell/2020/05/15/take-a-virtual-african-safari-in-2020-in-real-time-from-your-sofa/ (accessed on 12 October 2022).
31. Virtual Theme Park Rides You Can Experience from Home—Tips—The Jakarta Post. Available online: https://www.thejakartapost.com/travel/2020/03/31/virtual-theme-park-rides-you-can-experience-from-home.html (accessed on 12 October 2022).
32. Schroeder, R. Social interaction in virtual environments: Key issues, common themes, and a framework for research. In *The Social Life of Avatars*; Springer: Berlin/Heidelberg, Germany, 2002; pp. 1–18.
33. Wang, X.; Chen, Q.; Li, Z. A 3d reconstruction method for augmented reality sandbox based on depth sensor. In Proceedings of the 2021 IEEE 2nd International Conference on Information Technology, Big Data and Artificial Intelligence (ICIBA), Chongqing, China, 17–19 December 2021; IEEE: Piscataway, NJ, USA, 2021; Volume 2, pp. 844–849.
34. Romanoni, A.; Fiorenti, D.; Matteucci, M. Mesh-based 3d textured urban mapping. In Proceedings of the 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Vancouver, BC, Canada, 24–28 September 2017; IEEE: Piscataway, NJ, USA, 2017; pp. 3460–3466.
35. Brock, E.; Huang, C.; Wu, D.; Liang, Y. Lidar-based real-time mapping for digital twin development. In Proceedings of the 2021 IEEE International Conference on Multimedia and Expo (ICME), Shenzhen, China, 5–9 July 2021; IEEE: Piscataway, NJ, USA, 2021; pp. 1–6.
36. Hou, M.; Liu, Y.; Gong, S. Study on the controlled atmospheric refractivity fluctuation impairing the imaging quality of the heterodyne detection lidar. In Proceedings of the 2018 12th International Symposium on Antennas, Propagation and EM Theory (ISAPE), Hangzhou, China, 3–6 December 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 1–4.
37. Kulawiak, M.; Lubniewski, Z. Processing of lidar and multibeam sonar point cloud data for 3d surface and object shape reconstruction. In Proceedings of the 2016 Baltic Geodetic Congress (BGC Geomatics), Gdansk, Poland, 2–4 June 2016; IEEE: Piscataway, NJ, USA, 2016; pp. 187–190.
38. Miao, Y.; Tang, Y.; Alzahrani, B.A.; Barnawi, A.; Alafif, T.; Hu, L. Airborne lidar assisted obstacle recognition and intrusion detection towards unmanned aerial vehicle: Architecture, modeling, and evaluation. *IEEE Trans. Intell. Transp. Syst.* **2020**, *22*, 4531–4540. [CrossRef]
39. Peng, C.-C.; He, R. A concept for high precision digital terrain 3d mapping using UAVs and multiple solid-state lidars. In Proceedings of the 2022 IEEE International Conference on Consumer Electronics-Taiwan, Taipei, Taiwan, 6–8 July 2022; IEEE: Piscataway, NJ, USA, 2022; pp. 457–458.
40. Dubrovinskaya, E.; Dalgleish, F.; Ouyang, B.; Casari, P. Underwater lidar signal processing for enhanced detection and localization of marine life. In Proceedings of the 2018 OCEANS-MTS/IEEE Kobe Techno-Oceans (OTO), Kobe, Japan, 28–31 May 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 1–8.
41. Zhou, G.; Li, C.; Zhang, D.; Liu, D.; Zhou, X.; Zhan, J. Overview of underwater transmission characteristics of oceanic lidar. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2021**, *14*, 8144–8159. [CrossRef]

42. Kurdi, F.T.; Awrangjeb, M.; Liew, A.W.-C. Automated building footprint and 3d building model generation from lidar point cloud data. In Proceedings of the 2019 Digital Image Computing: Techniques and Applications (DICTA), Perth, Australia, 2–4 December 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 1–8.
43. Li, Q.; Lu, L.; Jiang, H.; Huang, J.; Liu, Z. Object-based urban land cover mapping using high-resolution airborne imagery and lidar data. In Proceedings of the 2018 Fifth International Workshop on Earth Observation and Remote Sensing Applications (EORSA), Xi'an, China, 18–20 June 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 1–5.
44. Wang, L.; Chu, C.-H.H. 3d building reconstruction from lidar data. In Proceedings of the 2009 IEEE International Conference on Systems, Man and Cybernetics, San Antonio, TX, USA, 11–14 October 2009; IEEE: Piscataway, NJ, USA, 2009; pp. 3054–3059.
45. Cheng, L.; Tong, L.; Zhao, W.; Liu, Y.; Li, M. Dynamic triangle—Based method for 3d building rooftop reconstruction from lidar data. In Proceedings of the 2011 19th International Conference on Geoinformatics, Shanghai, China, 24–26 June 2011; IEEE: Piscataway, NJ, USA, 2011; pp. 1–4.