Virtual Reality over Wireless Networks: Quality-of-Service Model and Learning-Based Resource Management

Mingzhe Chen*, Walid Saad†, and Changchuan Yin*
*Beijing Laboratory of Advanced Information Network, Beijing University of Posts and Telecommunications, Beijing, China 100876,
Emails: chenmingzhe@bupt.edu.cn, ccyin@ieee.org.
†Wireless@VT, Bradley Department of Electrical and Computer Engineering, Virginia Tech, Blacksburg, VA, USA, Email: walids@vt.edu.

***Abstract*—In this paper, the problem of resource management is studied for a network of wireless virtual reality (VR) users communicating over small cell networks (SCNs). In order to capture the VR users' quality-of-service (QoS) in SCNs, a novel VR model, based on multi-attribute utility theory, is proposed. This model jointly accounts for VR metrics such as tracking accuracy, processing delay, and transmission delay. In this model, the small base stations (SBSs) act as the VR control centers that collect the tracking information from VR users over the cellular uplink. Once this information is collected, the SBSs will then send the three-dimensional images and accompanying audio to the VR users over the downlink. Therefore, the resource allocation problem in VR wireless networks must jointly consider both the uplink and downlink. This problem is then formulated as a noncooperative game and a distributed algorithm based on the machine learning framework of echo state networks (ESNs) is proposed to find the solution of this game. The proposed ESN algorithm enables the SBSs to predict the VR QoS of each SBS and is guaranteed to converge to a mixed-strategy Nash equilibrium. The analytical result shows that each user's VR QoS jointly depends on both VR tracking accuracy and wireless resource allocation. Simulation results show that the proposed algorithm yields significant gains, in terms of VR QoS utility, that reach up to 22.2% and 37.5%, respectively, compared to Q-learning and a baseline proportional fair algorithm. The results also show that the proposed algorithm has a faster convergence time than Q-learning and can guarantee low delays for VR services.**

I. Introduction

Virtual reality (VR) services will enable users to experience and interact with virtual and immersive environments through a first-person view [2]. For instance, individuals can use a VR device to walk around in a fully immersive world and travel to any destination, within the confines of their own home. Compared to a static high definition (HD) video, a VR video is generated based on the users' movement such as head and eye movements. Therefore, generating a VR video requires tracking information related to the users' interactions with the VR environment. In consequence, the tracking accuracy and the delay of tracking information transmission will significantly affect the creation and transmission of VR videos hence affecting the users' experience. Such tracking delay does not exist for regular HD videos and, thus, constitutes a fundamental difference between VR and HD videos. If VR devices such as HTC Vive [3] rely on wired connections to a VR control center, such as a computer, for processing their information, then the users will be significantly restricted in the type of actions that they can take and the VR applications that they can experience. To enable truly immersive VR applications, one must deploy VR systems [4] over wireless cellular networks. In particular, VR systems can use the wireless connectivity of small cell networks (SCNs) [4] in which small cell base stations (SBSs) can act as VR control centers that directly connect to the VR devices over wireless links, collect the tracking[1] information from the VR devices, and send the VR images to the VR devices over wireless links. However, operating VR devices over SCNs faces many challenges in terms of tracking, low delay (typically less than 20 ms), and high data rate (typically over 25 Mbps) [4].

A. Related Work

The existing literature has studied a number of problems related to VR such as in [2], [4]–[13]. The authors in [2] and [4] provided qualitative surveys that motivate the deployment of VR over wireless systems, but these works do not provide any mathematically rigorous modeling. In [5]. the authors proposed a distortion-aware concurrent multipath data transfer algorithm to minimize the end-to-end video distortion. The work in [6] deveploped a modeling-based approach to optimize the high frame rate video transmission over wireless networks. However, the works in [5] and [6] only consider the video content transmission which cannot be directly applied to the VR content transmission since the VR contents are generated based on the users' tracking information. In [7], the authors proposed a streaming scheme that delivers only the visible portion of a 360° video based on head movement prediction. Meanwhile, the authors in [8] developed an algorithm for generating high-quality stereo panoramas. The work in [9] proposed a real-time solution that uses a single commodity RGB-D camera to track hand manipulation. In [10], a reinforcement learning algorithm is proposed to guide a user's movement within a VR immersive environment. The authors in [11] proposed an approach based on the three-dimensional (3D) heat maps to address the delay challenges. The work in [12] designed several experiments for quantifying the performance

This work was supported in part by the National Natural Science Foundation of China under Grant 61671086 and Grant 61629101, the 111 Project under Grant B17007, the Director Funds of Beijing Key Laboratory of Network System Architecture and Convergence under Grant 2017BKL-NSAC-ZJ-04, by BUPT Excellent Ph.D. Students Foundation, and by the U.S. National Science Foundation under Grants IIS-1633363, CNS-1460316, and CNS-1617896.

A preliminary version of this work was published in the IEEE GLOBECOM conference [1].

[1]Here, tracking pertains to the fact that the immersive VR applications must continuously collect a very accurate localization of each user including the positions, orientation, and eye movement (i.e., gaze tracking).

of tile-based 360° video streaming over a real cellular network. In [13], the authors performed a WiFi experiment for wireless VR for a single user within a single room. However, beyond the survey in [4], which motivated the use of VR over wireless and the WiFi experiment for a single VR user in [13], the majority of existing VR works in [2], [7]–[12] focus on VR systems that are deployed over wired networks and, as such, they do not capture any challenges of deploying VR over wireless SCNs. Moreover, most of these existing works [2], [4], [7]–[13] focus only on improving a single VR quality-of-service (QoS) metric such as tracking or generation of 3D images. Indeed, this prior art does not develop any VR-specific model that can capture all factors of VR QoS and, hence, these existing works fall short in addressing the challenges of optimizing VR QoS for wireless users.

B. Main Contributions

The main contribution of this paper is a novel framework for enabling wireless cellular networks to integrate VR applications and services. To our best knowledge, *this is the first work that develops a comprehensive framework for analyzing the performance of VR services over cellular networks.* Our main contribution include:

- We propose a novel VR model based on *multi-attribute utility theory* [14], to jointly capture the tracking accuracy, transmission delay, and processing delay thus effectively quantifying the VR QoS for all wireless users. In this VR model, the tracking information is transmitted from the VR users to the SBSs over the cellular uplink while the VR images are transmitted in the downlink from the SBSs to their users.
- We analyze resource (resource blocks) allocation *jointly* over the uplink and downlink. We formulate the problem as a noncooperative game in which the players are the SBSs. Each player seeks to find an optimal spectrum allocation scheme to optimize a utility function that captures the VR QoS.
- To solve this VR resource management game, we propose a learning algorithm based on echo state networks (ESNs) [15] that can predict the VR QoS value resulting from resource allocation and reach a mixed-strategy Nash equilibrium (NE). Compared to existing learning algorithms [16]–[18], the proposed algorithm has lower complexity, requires less information due to its neural network nature, and is guaranteed to converge to a mixed-strategy NE. One unique feature of this algorithm is that, after training, it can use the stored ESN information to effectively find an optimal converging path to a mixed-strategy NE.
- We perform fundamental analysis on the gains and trade-offs that stem from changing the number of uplink and downlink resource blocks for each user. This analytical result shows that, in order to improve the VR QoS of each user, we can improve the tracking system or increase the number of the resource blocks allocated to each user according to each user's specific state.

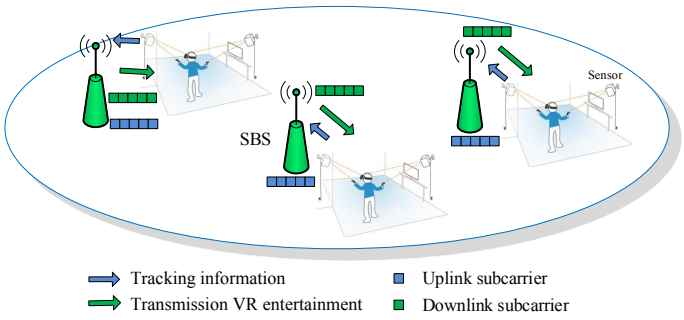


Fig. 1. A network of immersive VR application.

- Simulation results show that the proposed algorithm can yield, respectively, 22.2% and 37.5% gains in terms of VR QoS utility compared to Q-learning and proportional fair algorithm. The results also show that the proposed algorithm significantly improves the convergence time of up to 24.2% compared to Q-learning.

The rest of this paper is organized as follows. The problem is formulated in Section II. The resource allocation algorithm is proposed in Section III. In Section IV, simulation results are analyzed. Finally, conclusions are drawn in Section V.

II. System Model and Problem Formulation

Consider the downlink transmission of a cloud-based small cell network (SCN)[2] servicing a set $\mathcal{U}$ of V wireless VR users via a set $\mathcal{B}$ of B SBSs [20]. Here, we focus on entertainment VR application such as watching immersive videos and playing immersive games [7]. VR allows users to be immersed in a virtual environment within which they can experience a 3D and high-resolution 360° vision with 3D surround stereo. In particular, immersive VR will provide a 360° panoramic image for each eye of a VR user. Compared to a conventional 120° image, a 360° panoramic image enables a VR user to have a surrounded vision without any dead spots. However, a 360° VR image needs more pixels than a traditional two-dimensional image, and, hence, VR transmission will require more stringent data rate [4] and delay (less than 20 ms) requirements than traditional multimedia services. Moreover, for an HD video, we only need to consider the video transmission delay as part of the QoS. In contrast, for a VR video, we need to jointly consider the video transmission delay, tracking information transmission delay, and the delay of generating the VR videos based on the tracking information. Note that, for VR applications, the transmission delay and processing delay will directly determine how each VR video is transmitted and, thus, they are a key determinant of the VR video quality.

The SBSs adopt an orthogonal frequency division multiple access (OFDMA) technique and transmit over a set of uplink resource blocks $\mathcal{S}^{\mathrm{u}}$ and a set of downlink resource blocks $\mathcal{S}^{\mathrm{d}}$, as shown in Fig. 1. The uplink resource blocks are used to

[2]Since next-generation cellular networks will all rely on a small cell architecture [19], we consider SCNs as the basis for our analysis. However, the proposed VR model and algorithm can also be used for any other type of cellular networks.

TABLE I
LIST OF NOTATIONS

Notation	Description	Notation	Description
V	Number of users	$S^{\mathrm{d}}, S^{\mathrm{u}}$	Number of downlink and uplink resource blocks
B	Number of SBSs	$\mathcal{S}^{\mathrm{d}}, \mathcal{S}^{\mathrm{u}}$	Sets of downlink and uplink resource blocks
$\boldsymbol{W}_{\text{in}}$	Input weight matrix	D_{ij}^{T}	Transmission delay between user i and SBS j
P_B	Transmit power of SBSs	d_{ij}	Distance between user i and SBS j
$\mathcal{A}_j$	Set of SBS j's actions	c_{ij}	Data rate of user i associated with SBS j
$\boldsymbol{a}_j$	One action of SBS j	$\boldsymbol{s}_{ij}^{\mathrm{d}}, \boldsymbol{s}_{ij}^{\mathrm{u}}$	Resource allocation vector
$\boldsymbol{a}_{ji}$	Action i of SBS j	$\boldsymbol{d}_j$	Downlink resource allocation of SBS j
$\boldsymbol{W}_{\text{out}}$	Output weight matrix	$\bar{u}_j$	Average utility function of SBS j
N_w	Number of reservoir units	$\hat{u}_j$	Utility function of SBS j
K_i	Tracking accuracy	γ_K	Maximal tracking inaccuracy
D_i	Total delay of user i	$\boldsymbol{v}_j$	Uplink resource allocation of SBS j
$\boldsymbol{\chi}$	Vector of user's localization	$U_i\left(D_i, K_i\right)$	Total utility function of user i's VR QoS
λ	Learning rate	$U_i\left(D_i \mid K_i\right)$	Conditional utility function of user i's VR QoS
D_i^{P}	Processing delay of user i	V_j	Number of users associated with SBS j
$\boldsymbol{W}$	Reservoir weight matrix	$\lvert\mathcal{A}_j\rvert$	Number of SBS j's actions
$\boldsymbol{\mu}_j$	Reservoir state of SBS j	$\boldsymbol{a}_{-j}$	Actions of all SBSs other than SBS j
A	Data size of tracking vector	L	Maximum Data size of each VR image

transmit the data that is collected by the VR sensors placed at a VR user's headset or near the VR user while the downlink resource blocks are used to transmit the image displayed on each user's VR device. We define the coverage of each SBS as a circular area of radius r_B and we assume that each SBS will only allocate resource blocks to the users located in its coverage range. Table I provides a summary of the notations used hereinafter.

A. VR Model

For our VR model, we consider delay and tracking accuracy as the main VR QoS metrics of interest. Based on the accurate localization of each user, the SBS can build the immersive and virtual environment for each user. Among the components of the VR QoS, a delay metric can be defined to capture two key VR service requirements: high data rate and low delay. Next, we will explicitly discuss all the components of the considered VR QoS metrics.

1) Tracking Model: For any VR QoS metric, tracking consists of position tracking and orientation tracking [7]. VR tracking directly affects the construction of the users' virtual environment since the SBSs need to use each user's localization information to construct the virtual environment. Hereinafter, we use the term "localization information" to represent the information related to the user's location and orientation. The localization data of each user is used as the primary component of tracking [4]. The tracking vector of each user i is $\boldsymbol{\chi}_i = [\chi_{i1}, \chi_{i2}, \chi_{i3}, \chi_{i4}, \chi_{i5}, \chi_{i6}]$, where $[\chi_{i1}, \chi_{i2}, \chi_{i3}]$ represents the position of each VR user while $[\chi_{i4}, \chi_{i5}, \chi_{i6}]$ represents the orientation of each user. Here, we note that the position and orientation of each user are determined by the SBS via the information collected by the sensors. For each VR user i, the tracking accuracy $K_i\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right)$ can be given by:

$$K_i\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right) = 1 - \frac{\left\|\boldsymbol{\chi}_i\left(\boldsymbol{\gamma}_i^{\mathrm{u}}\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right)\right) - \boldsymbol{\chi}_i^R\right\|}{\max\limits_{\boldsymbol{s}_{ij}^{\mathrm{u}}}\left\|\boldsymbol{\chi}_i\left(\boldsymbol{\gamma}_i^{\mathrm{u}}\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right)\right) - \boldsymbol{\chi}_i^R\right\|}, \tag{1}$$

where $\boldsymbol{\chi}_i\left(\boldsymbol{\gamma}_i^{\mathrm{u}}\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right)\right)$ is the tracking vector transmitted from the VR headset to the SBS over wireless links that depends on the signal-to-interference-plus-noise (SINR) ratio. $\boldsymbol{s}_{ij}^{\mathrm{u}} = \left[s_{ij1}^{\mathrm{u}}, \ldots, s_{ijS^{\mathrm{u}}}^{\mathrm{u}}\right]$ represents the vector of uplink resource blocks that SBS j allocates to user i with $s_{ijk}^{\mathrm{u}} \in \{1, 0\}$. Here, the uplink (downlink) resource blocks are equally divided into S^{u} (S^{d}) groups. Hereinafter, the term *resource block* refers to one of those groups. $s_{ijk}^{\mathrm{u}} = 1$ indicates that resource block k is allocated to user i. $\boldsymbol{\gamma}_i^{\mathrm{u}}\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right) = \left[\gamma\left(s_{ij1}^{\mathrm{u}}\right), \ldots, \gamma\left(s_{ijS^{\mathrm{u}}}^{\mathrm{u}}\right)\right]$ represents the SINR of SBS j with resource blocks $\boldsymbol{s}_{ij}^{\mathrm{u}}$ where $\gamma\left(s_{ijk}^{\mathrm{u}}\right) = \frac{P_U h_{ij}^k}{\sigma^2 + \sum\limits_{l \in \mathcal{U}_k, l \neq i} P_U h_{il}^k}$ is the SINR between user i and SBS j over resource block k. Here, $\mathcal{U}_k$ is the set of users that use uplink resource block k. P_U is the transmit power of user i (assumed to be equal for all users), σ^2 is the variance of the Gaussian noise and $h_{ij,\tau}^k = g_{ij,\tau}^k d_{ij}^{-\beta}$ is the path loss between user i and SBS j over resource block k with $g_{ij,\tau}^k$ being the Rayleigh fading parameter, d_{ij} the distance between user i and SBS j, and β the path loss exponent. In $\boldsymbol{\chi}_i\left(\boldsymbol{\gamma}_i^{\mathrm{u}}\left(\boldsymbol{s}_{ij,\tau}^{\mathrm{u}}\right)\right)$, the resource block vector $\boldsymbol{s}_{ij}^{\mathrm{u}}$ determines the uplink SINR of the tracking vector's transmission. The tracking vector will be subject to bit errors and, hence, it will depend on the SINR and the resource block vector $\boldsymbol{s}_{ij}^{\mathrm{u}}$. $\left\|\boldsymbol{\chi}_i\left(\boldsymbol{\gamma}_i^{\mathrm{u}}\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right)\right) - \boldsymbol{\chi}_i^R\right\|$ represents the wireless tracking inaccuracy. $\boldsymbol{\chi}_i^R$ is obtained from the users' force feedback and transmitted via a dedicated wireless channel. Force feedback [21] represents the feedback that the users send to the SBSs whenever they are not satisfied with the displayed VR image. $\boldsymbol{\chi}_i^R$ is only transmitted from the users to the SBSs when the users feel uncomfortable in the environment. Since $\boldsymbol{\chi}_i^R$ is not transmitted every time slot, the SBSs can use orthogonal resource blocks over a dedicated channel to transmit $\boldsymbol{\chi}_i^R$. Thus, $\boldsymbol{\chi}_i^R$ is generally much more accurate than $\boldsymbol{\chi}_i\left(\boldsymbol{\gamma}_i^{\mathrm{u}}\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right)\right)$. Note that, (1) is based on normalized root mean square errors [22], which is a popular measure of the difference between two datasets. From (1), we can see that, when the SINR increases, the bit error rate will decrease and, hence, $\left\|\boldsymbol{\chi}_i\left(\boldsymbol{\gamma}_i^{\mathrm{u}}\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right)\right) - \boldsymbol{\chi}_i^R\right\|$ decreases and the tracking accuracy improves.

2) Delay: Next, we define the delay component that consists of the transmission delay and processing (and computing)

delay. The transmission delay of each user i will be:

$$D_{ij}^{\mathrm{T}}\left(\boldsymbol{s}_{ij}^{\mathrm{d}},\boldsymbol{s}_{ij}^{\mathrm{u}}\right)=\frac{L}{c_{ij}\left(\boldsymbol{s}_{ij}^{\mathrm{d}}\right)}+\frac{A}{c_{ij}\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right)}, \tag{2}$$

where L is the maximum size of VR image that each SBS needs to transmit to its users, A is the size of the tracking vector that each user needs to transmit to the associated SBS, and $c_{ij}\left(\boldsymbol{s}_{ij}^{\mathrm{d}}\right)=\sum_{k=1}^{S^{\mathrm{d}}} s_{ijk}^{\mathrm{d}} B_{\mathrm{R}}\log_2\left(1+\gamma\left(s_{ijk}^{\mathrm{d}}\right)\right)$ is the downlink rate of user i. $c_{ij}\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right)=\sum_{k=1}^{S^{\mathrm{u}}} s_{ijk}^{\mathrm{u}} B_{\mathrm{R}}\log_2\left(1+\gamma\left(s_{ijk}^{\mathrm{u}}\right)\right)$ is the uplink rate of user i. $\boldsymbol{s}_{ij}^{\mathrm{d}}=\left[s_{ij1}^{\mathrm{d}},\ldots,s_{ijS^{\mathrm{d}}}^{\mathrm{d}}\right]$ is the vector of resource blocks that SBS j allocates to user i with $s_{ijk}^{\mathrm{d}}\in\{0,1\}$. $s_{ijk}^{\mathrm{d}}=1$ indicates that resource block k is allocated to user i. $\gamma\left(s_{ijk}^{\mathrm{d}}\right)=\frac{P_B h_{ij}^k}{\sigma^2+\sum_{l\in\mathcal{B}_k,l\neq j}P_B h_{il}^k}$ is the signal-to-interference-plus-noise ratio between user i and SBS j over resource block k with $\mathcal{B}_k$ being the set of the SBSs that use downlink resource block k. B_{R} is the bandwidth of each resource block, P_B is the transmit power of SBS j (assumed to be equal for all SBSs).

In the VR QoS, the *processing delay* primarily stems from the tracking accuracy. To properly capture the processing delay, we define a vector $\boldsymbol{l}\left(\boldsymbol{\chi}_i\left(\boldsymbol{\gamma}_i^{\mathrm{u}}\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right)\right)\right)$ that represents a VR image of user i constructed by the tracking vector $\boldsymbol{\chi}_i\left(\boldsymbol{\gamma}_i^{\mathrm{u}}\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right)\right)$. $\boldsymbol{l}\left(\boldsymbol{\chi}_i\left(\boldsymbol{\gamma}_i^{\mathrm{u}}\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right)\right)\right)$ will be typically generated before the SBSs receive the force feedback tracking vector $\boldsymbol{\chi}_i^R$, as the VR system will use the historical tracking information to predict the future tracking vector. When the SBSs receive $\boldsymbol{\chi}_i^R$, they must construct $\boldsymbol{l}\left(\boldsymbol{\chi}_i^R\right)$ based on $\boldsymbol{\chi}_i^R$. The simplest way of constructing the VR image $\boldsymbol{l}\left(\boldsymbol{\chi}_i^R\right)$ is to correct it from $\boldsymbol{l}\left(\boldsymbol{\chi}_i\left(\boldsymbol{\gamma}_i^{\mathrm{u}}\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right)\right)\right)$ to $\boldsymbol{l}\left(\boldsymbol{\chi}_i^R\right)$. The processing delay will then represent the time that an SBS must spend to change the VR image from $\boldsymbol{l}\left(\boldsymbol{\chi}_i\left(\boldsymbol{\gamma}_i^{\mathrm{u}}\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right)\right)\right)$ to $\boldsymbol{l}\left(\boldsymbol{\chi}_i^R\right)$. The bits that the SBSs use to change the VR image from $\boldsymbol{l}\left(\boldsymbol{\chi}_i\left(\boldsymbol{\gamma}_i^{\mathrm{u}}\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right)\right)\right)$ to $\boldsymbol{l}\left(\boldsymbol{\chi}_i^R\right)$ can be calculated by using the motion search algorithm [23]. The motion search algorithm can find the different pixels between $\boldsymbol{l}\left(\boldsymbol{\chi}_i\left(\boldsymbol{\gamma}_i^{\mathrm{u}}\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right)\right)\right)$ and $\boldsymbol{l}\left(\boldsymbol{\chi}_i^R\right)$ and, hence, an SBS can directly replace those different pixels [23]. In our model, the VR sensors can accurately collect the VR users' movement [24]. Moreover, if the SBSs receive accurate tracking data, they can accurately extract each user's location and orientation. Hence, the tracking accuracy depends only on the data error that is caused by the uplink transmission over the wireless link. To compute the processing delay, we first assume that the total amount of computational resources available at each SBS is M which is equally allocated to the associated users. M represents the number of bits that can be processed by each SBS which is determined by each SBS's central processing unit (CPU). Then, the processing delay can be given by:

$$D_i^{\mathrm{P}}\left(K_i\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right)\right)=\frac{\upsilon\left(\boldsymbol{l}\left(\boldsymbol{\chi}_i\left(\boldsymbol{\gamma}_i^{\mathrm{u}}\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right)\right)\right),\boldsymbol{l}\left(\boldsymbol{\chi}_i^R\right)\right)}{M/N_j}, \tag{3}$$

where $0\leq\upsilon\left(\boldsymbol{l}\left(\boldsymbol{\chi}_i\left(\boldsymbol{\gamma}_i^{\mathrm{u}}\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right)\right)\right),\boldsymbol{l}\left(\boldsymbol{\chi}_i^R\right)\right)\leq L$ represents the number of bits that must be changed when SBS j transforms the VR image from $\boldsymbol{l}\left(\boldsymbol{\chi}_i\left(\boldsymbol{\gamma}_i^{\mathrm{u}}\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right)\right)\right)$ to $\boldsymbol{l}\left(\boldsymbol{\chi}_i^R\right)$. $\upsilon\left(\cdot\right)$ depends on the image size, the number of bits used to store a pixel, and the content of the VR image and is the result of the motion search algorithm. Here, we adopt the motion search algorithm of [25] to determine $\upsilon\left(\boldsymbol{l}\left(\boldsymbol{\chi}_i\left(\boldsymbol{\gamma}_i^{\mathrm{u}}\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right)\right)\right),\boldsymbol{l}\left(\boldsymbol{\chi}_i^R\right)\right)$. When the SINR $\boldsymbol{\chi}_i\left(\boldsymbol{\gamma}_i^{\mathrm{u}}\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right)\right)$ of user i increases, the bit errors in $\boldsymbol{\chi}_i\left(\boldsymbol{\gamma}_i^{\mathrm{u}}\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right)\right)$ will decrease. In consequence, $\upsilon\left(\boldsymbol{l}\left(\boldsymbol{\chi}_i\left(\boldsymbol{\gamma}_i^{\mathrm{u}}\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right)\right)\right),\boldsymbol{l}\left(\boldsymbol{\chi}_i^R\right)\right)$ will decrease and, hence, the processing delay decreases. Here, the SINR $\boldsymbol{\chi}_i\left(\boldsymbol{\gamma}_i^{\mathrm{u}}\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right)\right)$ of user i depends on the resource blocks $\boldsymbol{s}_{ij}^{\mathrm{u}}$ allocated to user i. When the deviation between $\boldsymbol{l}\left(\boldsymbol{\chi}_i\left(\boldsymbol{\gamma}_i^{\mathrm{u}}\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right)\right)\right)$ and $\boldsymbol{l}\left(\boldsymbol{\chi}_i^R\right)$ increases, more data is needed to correct the image. $\upsilon\left(\boldsymbol{l}\left(\boldsymbol{\chi}_i\left(\boldsymbol{\gamma}_i^{\mathrm{u}}\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right)\right)\right),\boldsymbol{l}\left(\boldsymbol{\chi}_i^R\right)\right)\leq L$ is a constraint that captures the maximum number of bits that must be corrected. N_j is the number of the users associated with SBS j and $\frac{M}{N_j}$ is the computation resources allocated to any user i's VR session. From (3), we can see that the processing delay depends on the tracking accuracy, the number of the users associated with SBS j, and the resource blocks allocated to user i. The total delay of each user i will hence be:

$$D_i\left(\boldsymbol{s}_{ij}^{\mathrm{d}},\boldsymbol{s}_{ij}^{\mathrm{u}}\right)=D_i^{\mathrm{P}}\left(K_i\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right)\right)+D_i^{\mathrm{T}}\left(\boldsymbol{s}_{ij}^{\mathrm{d}},\boldsymbol{s}_{ij}^{\mathrm{u}}\right). \tag{4}$$

B. Utility Function Model

Next, we use the framework multi-attribute utility theory [14] to define a utility function that can effectively capture VR delay and tracking. Using multi-attribute utility theory, we construct a total utility function that jointly considers the delay and tracking of the VR QoS. Conventional techniques for defining a utility function, such as by directly summing up delay and tracking are only valid under the assumption that delay and tracking are independent and that their relationship is linear [26, Chapters 2 and 3]. However, for VR, the relationship between the delay and tracking is not necessarily linear nor independent. Therefore, we use multi-attribute utility theory [14] to define the utility function. The defined utility function can assign to each delay and tracking components of the VR QoS a unique utility value without any constraints.

In order to construct the total utility function, we first define a conditional utility function for the delay component of the VR QoS [14]. In the total utility function, both tracking and delay will contribute to the utility value. In contrast, in the conditional utility function of delay, only delay contributes to the utility function and the tracking value is assumed to be given. The method that uses a conditional utility function to derive the total utility function is analogous to the approach used in probability theory to derive a joint probability distribution from conditional probability distributions [27]. For VR, the total utility function of user i is given by $U_i\left(D_i\left(\boldsymbol{s}_{ij}^{\mathrm{d}},\boldsymbol{s}_{ij}^{\mathrm{u}}\right),K_i\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right)\right)$. The conditional utility function of delay, $U_i\left(D_i\left(\boldsymbol{s}_{ij}^{\mathrm{d}},\boldsymbol{s}_{ij}^{\mathrm{u}}\right)\middle|K_i\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right)\right)$, represents the total utility function given a certain value of the tracking accuracy. Based on [14], the conditional utility function of

delay for user i can be given by:

$$U_i\left(D_i\left(\boldsymbol{s}_{ij}^{\mathrm{d}},\boldsymbol{s}_{ij}^{\mathrm{u}}\right)\middle|K_i\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right)\right) = \begin{cases} \frac{D_{\max,i}\left(K_i\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right)\right)-D_i\left(\boldsymbol{s}_{ij}^{\mathrm{d}},K_i\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right)\right)}{D_{\max,i}\left(K_i\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right)\right)-\gamma_D}, & D_i\left(\boldsymbol{s}_{ij}^{\mathrm{d}},K_i\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right)\right)\geq\gamma_D, \\ 1, & D_i\left(\boldsymbol{s}_{ij}^{\mathrm{d}},K_i\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right)\right)<\gamma_D, \end{cases} \tag{5}$$

where γ_D is the maximal tolerable delay for each VR user (maximum supported by the VR system being used) and $D_{\max,i}\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right) = \max_{\boldsymbol{s}_{ij}^{\mathrm{d}}}\left(D_i\left(\boldsymbol{s}_{ij}^{\mathrm{d}},\boldsymbol{s}_{ij}^{\mathrm{u}}\right)\right)$ is the maximum delay of VR user i given $\boldsymbol{s}_{ij}^{\mathrm{u}}$. Here, $U_i\left(D_{\max,i}\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right)\middle|K_i\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right)\right)=0$ and $U_i\left(\gamma_D\middle|K_i\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right)\right)=1$. From (5), we can see that, when $D_i\left(\boldsymbol{s}_{ij}^{\mathrm{d}},\boldsymbol{s}_{ij}^{\mathrm{u}}\right)<\gamma_D$, the conditional utility value will remain 1. This is due to the fact that, when delay $D_i\left(\boldsymbol{s}_{ij}^{\mathrm{d}},\boldsymbol{s}_{ij}^{\mathrm{u}}\right)$ reaches the delay requirement γ_D, the utility value will reach its maximum of 1. Since delay and tracking are both dominant components, we can construct the total utility function, $U_i\left(D_i\left(\boldsymbol{s}_{ij}^{\mathrm{d}},\boldsymbol{s}_{ij}^{\mathrm{u}}\right),K_i\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right)\right)$, that jointly considers the delay and tracking based on [14]. Here, a *dominant component* represents the component that will minimize the total utility function regardless of the value of other components. For VR QoS, delay and tracking are both dominant components. For example, the VR QoS will be minimized when the value of the delay function is at a minimum regardless of the value of tracking accuracy. The use of dominant components such as delay and tracking will simplify the formulation of the total utility function [14]. Therefore, the total utility function of tracking and delay will be [14]:

$$\begin{aligned} U_i\left(D_i\left(\boldsymbol{s}_{ij}^{\mathrm{d}},\boldsymbol{s}_{ij}^{\mathrm{u}}\right),K_i\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right)\right) &= U_i\left(D_i\left(\boldsymbol{s}_{ij}^{\mathrm{d}},\boldsymbol{s}_{ij}^{\mathrm{u}}\right)\middle|K_i\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right)\right)U_i\left(K_i\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right)\right), \\ &\overset{(a)}{=}\left(1-\frac{\left\|\boldsymbol{\chi}_i\left(\boldsymbol{\gamma}_i^{\mathrm{u}}\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right)\right)-\boldsymbol{\chi}_i^R\right\|}{\max\limits_{\boldsymbol{s}_{ij}^{\mathrm{u}}}\left\|\boldsymbol{\chi}_i\left(\boldsymbol{\gamma}_i^{\mathrm{u}}\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right)\right)-\boldsymbol{\chi}_i^R\right\|}\right)\frac{D_{\max,i}\left(K_i\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right)\right)-D_i\left(\boldsymbol{s}_{ij}^{\mathrm{d}},K_i\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right)\right)}{D_{\max,i}\left(K_i\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right)\right)-\gamma_D}. \end{aligned} \tag{6}$$

where $U_i\left(K_i\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right)\right)$ is the utility function of tracking accuracy. (a) is obtained by substituting (1) and (5) into (6). From (6), we can see that, the vector of resource blocks allocated to user i for data transmission, $\boldsymbol{s}_{ij}^{\mathrm{d}}$, and the resource blocks allocated to user i for obtaining the tracking information, $\boldsymbol{s}_{ij}^{\mathrm{u}}$, jointly determine the value of the total utility function. Moreover, this total utility function can assign a unique value to each tracking and delay component of the VR QoS.

C. Problem Formulation

Our goal is to develop an effective resource allocation scheme that allocates resource blocks so as to maximize the users' VR QoS. This maximization jointly considers the coupled problems of user association, uplink resource allocation, and downlink resource allocation. This optimization problem can formalized as follows:

$$\max_{\boldsymbol{s}_{ij}^{\mathrm{d}},\boldsymbol{s}_{ij}^{\mathrm{u}},\mathcal{U}_j}\sum_{t=1}^{T}\sum_{j\in\mathcal{B}}\sum_{i\in\mathcal{U}_j}U_{it}\left(D_i\left(\boldsymbol{s}_{ij}^{\mathrm{d}},\boldsymbol{s}_{ij}^{\mathrm{u}}\right),K_i\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right)\right) \tag{7}$$

$$\text{s. t.} \quad |\mathcal{U}_j|\leqslant V_j, \quad \forall j\in\mathcal{B}, \tag{7a}$$

$$s_{ij,k}^{\mathrm{d}}\in\{0,1\}, \quad \forall i\in\mathcal{U},\forall j\in\mathcal{B}, \tag{7b}$$

$$s_{ij,k}^{\mathrm{u}}\in\{0,1\}, \quad \forall i\in\mathcal{U},\forall j\in\mathcal{B}, \tag{7c}$$

where $\mathcal{U}_j$ is the set of the VR users associated with SBS j, $|\mathcal{U}_j|$ is the number of the VR users associated with SBS j, V_j is the number of the VR users located in the coverage of SBS j. U_{it} is the utility value of user i at time t. (1a) indicates that the number of VR users associated with SBS j must not exceed the number of VR users located in the coverage of SBS j. (1b) and (1c) indicate that each downlink resource block $s_{ij,k}^{\mathrm{d}}$ or uplink resource block $s_{ij,k}^{\mathrm{u}}$ can be allocated to one VR user.

In (7), the VR QoS of each SBS depends not only on its resource allocation scheme but also on the resource allocation decisions of other SBSs. Consequently, the use of centralized optimization for such a complex problem is not possible as it is largely intractable and yields significant overhead. To overcome this challenge, we formulate a noncooperative game $\mathcal{G}=\left[\mathcal{B},\{\mathcal{A}_j\}_{j\in\mathcal{B}},\{u_j\}_{j\in\mathcal{B}}\right]$ between the SBSs that can be implemented in a distributed way. Each player j has a set $\mathcal{A}_j=\left\{\boldsymbol{a}_{j1},\ldots,\boldsymbol{a}_{j|\mathcal{A}_j|}\right\}$ of $|\mathcal{A}_j|$ actions. In this game, each action of SBS j, $\boldsymbol{a}_j=(\boldsymbol{d}_j,\boldsymbol{v}_j)$ consists of: (i) downlink resource allocation vector, $\boldsymbol{d}_j=\left[\boldsymbol{s}_{1j}^{\mathrm{d}},\ldots,\boldsymbol{s}_{V_jj}^{\mathrm{d}}\right]$. $\boldsymbol{s}_{ij}^{\mathrm{d}}=\left[s_{ij1},\ldots,s_{ijS^{\mathrm{d}}}\right]$ represents the vector of downlink resource blocks that SBS j allocates to user i and $s_{ijk}\in\{1,0\}$ with $\sum_{i=1}^{V_j}s_{ijk}^{d}=1$. $s_{ijk}=1$ indicating that channel k is allocated to user i and $s_{ijk}=0$ otherwise. V_j is the number of all users in the coverage area of SBS j, and (ii) uplink resource allocation vector, $\boldsymbol{v}_j=\left[\boldsymbol{s}_{1j}^{\mathrm{u}},\ldots,\boldsymbol{s}_{V_jj}^{\mathrm{u}}\right]$ with $\sum_{i=1}^{V_j}s_{ijk}^{\mathrm{u}}=1$. $\boldsymbol{a}=(\boldsymbol{a}_1,\boldsymbol{a}_2,\ldots,\boldsymbol{a}_B)\in\mathcal{A}$, represents the action profile of all players and $\mathcal{A}=\prod_{j\in\mathcal{B}}\mathcal{A}_j$.

To maximize the VR QoS of each user, the utility function of each SBS j can be given by:

$$u_j\left(\boldsymbol{a}_j,\boldsymbol{a}_{-j}\right)=\sum_{i=1}^{V_j}U_i\left(D_i\left(\boldsymbol{s}_{ij}^{\mathrm{d}},\boldsymbol{s}_{ij}^{\mathrm{u}}\right),K_i\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right)\right), \tag{8}$$

where $\boldsymbol{a}_j\in\mathcal{A}_j$ is an action of SBS j and $\boldsymbol{a}_{-j}$ denotes the action profile of all SBSs other than SBS j. Let $\pi_{j,\boldsymbol{a}_i}=\lim_{T\to\infty}\frac{1}{T}\sum_{t=1}^{T}\mathbb{1}_{\{\boldsymbol{a}_j=\boldsymbol{a}_{ji}\}}=\Pr\left(\boldsymbol{a}_j=\boldsymbol{a}_{ji}\right)$ be the probability that SBS j uses action $\boldsymbol{a}_{ji}$. Hence, $\boldsymbol{\pi}_j=\left[\pi_{j,\boldsymbol{a}_{j1}},\ldots,\pi_{j,\boldsymbol{a}_{j|A_n|}}\right]$ will be a probability distribution for SBS j. We assume that the VR transmission is analyzed during a period that consists of T time slots. Therefore, the average value of the utility function can be given by:

$$\begin{aligned} \bar{u}_j\left(\boldsymbol{a}_j,\boldsymbol{a}_{-j}\right) &= \lim_{T\to\infty}\frac{1}{T}\sum_{t=1}^{T}u_j\left(\boldsymbol{a}_j,\boldsymbol{a}_{-j}\right) \\ &= \sum_{\boldsymbol{a}\in\mathcal{A}}\left(u_j\left(\boldsymbol{a}_j,\boldsymbol{a}_{-j}\right)\prod_{k\in\mathcal{B}}\pi_{k,\boldsymbol{a}_k}\right). \end{aligned} \tag{9}$$

To solve the proposed resource allocation game, we use the concept of a mixed-strategy NE, formally [28]:

Definition 1. (mixed-strategy Nash Equilibrium): A mixed strategy profile $\boldsymbol{\pi}^* = (\boldsymbol{\pi}_1^*, \ldots, \boldsymbol{\pi}_B^*) = \left(\boldsymbol{\pi}_j^*, \boldsymbol{\pi}_{-j}^*\right)$ is a *mixed-strategy Nash equilibrium* if, $\forall j \in \mathcal{B}$ and $\boldsymbol{\pi}_j$, we have:

$$\bar{u}_j\left(\boldsymbol{\pi}_j^*, \boldsymbol{\pi}_{-j}^*\right) \geq \bar{u}_j\left(\boldsymbol{\pi}_j, \boldsymbol{\pi}_{-j}^*\right), \tag{10}$$

where $\bar{u}_j\left(\boldsymbol{\pi}_j, \boldsymbol{\pi}_{-j}\right) = \sum_{\boldsymbol{a} \in \mathcal{A}} u_j\left(\boldsymbol{a}\right) \prod_{k \in \mathcal{B}} \pi_{k,\boldsymbol{a}_k}$ is the expected utility of SBS j selecting the mixed strategy $\boldsymbol{\pi}_j$.

The mixed-strategy NE for the SBSs represents a solution at which each SBS j can maximize the average VR QoS for its users, given the actions of its opponents.

III. Echo State Networks for Self-Organizing Resource Allocation

Next, we introduce a learning algorithm used to solve the VR game and find its NE. Since we consider both uplink and downlink resource allocation, the number of actions will be much larger than conventional resource allocation scenarios that typically consider only uplink or downlink resource allocation. Therefore, as the number of actions significantly increases, by using traditional game-theoretic algorithms such as fictitious play [29], each SBS may not be able to collect all of the information used to calculate the average utility function. Moreover, using such conventional game-theoretic and learning techniques, the SBSs will typically need to re-run the entire steps of the algorithm to reach a mixed-strategy NE as the states of the users and network vary. Hence, the delay during the convergence process of such algorithms may not be able to satisfy the QoS requirement of a dynamic VR network. To satisfy the QoS requirement for the VR transmission of each SBS, we propose a learning algorithm based on the powerful framework of *echo state networks* (ESN) [30]. The proposed ESN-based learning algorithm enables each SBS to predict the value of VR QoS that results from each action and, hence, can reach a mixed-strategy NE without having to traverse all actions. Since the proposed algorithm can store the past ESN information, then, it can find an optimal convergence path from the initial state to a mixed-strategy NE. Compared to our work in [31] that is based on two echo state network schemes, the proposed algorithm relies on only one echo state network, which reduces complexity. Next, we introduce the components of an ESN-based learning algorithm and, then, we introduce its update process.

A. ESN Components

The proposed ESN-based learning algorithm consists of five components: a) agents, b) inputs, c) ESN model d), actions, and e) output, defined as follows.

- *Agent*: The agents in our ESN are the SBSs in the set $\mathcal{B}$.
- *Actions:* Each action $\boldsymbol{a}_j$ of SBS j jointly considers the uplink and downlink resource blocks, which is given by:

$$\boldsymbol{a}_j = (\boldsymbol{d}_j, \boldsymbol{v}_j) = \left[\ \boldsymbol{s}_{1j}^{\mathrm{d}} \cdots \boldsymbol{s}_{V_j j}^{\mathrm{d}}, \boldsymbol{s}_{1j}^{\mathrm{u}} \cdots \boldsymbol{s}_{V_j j}^{\mathrm{u}}\ \right]^{\mathrm{T}}. \tag{11}$$

In order to guarantee that any action always has a non-zero probability to be chosen, the ε-greedy exploration [32] is adopted. This mechanism is responsible for selecting the actions that each SBS will perform during the learning process while harmonizing the tradeoff between exploitation and exploration. Therefore, the probability with which SBS j chooses action i will be given by:

$$\pi_{j,\boldsymbol{a}_j} = \begin{cases} 1 - \varepsilon + \frac{\varepsilon}{|\mathcal{A}_j|}, & \arg\max\limits_{\boldsymbol{a}_j \in \mathcal{A}_j} \hat{u}_j(\boldsymbol{a}_j), \\ \frac{\varepsilon}{|\mathcal{A}_j|}, & \text{otherwise}, \end{cases} \tag{12}$$

where $\hat{u}_{\tau,j}\left(\boldsymbol{a}_j\right) = \sum_{\boldsymbol{a}_{-j} \in \mathcal{A}_{-j}} u_j\left(\boldsymbol{a}_j, \boldsymbol{a}_{-j}\right) \pi_{-j,\boldsymbol{a}_{-j}}$ is the expected utility of an SBS j with respect to the actions of its opponents, $\mathcal{A}_{-j} = \prod_{k \neq j, k \in \mathcal{B}} \mathcal{A}_k$ is the set of actions other than the action of SBS j and $\pi_{-j,\boldsymbol{a}_{-j}} = \prod_{k \in \mathcal{B}, k \neq j,} \pi_{k,\boldsymbol{a}_k}$. $\hat{u}_{\tau,j}\left(\boldsymbol{a}_j\right)$ is the marginal probability distribution over the action set of SBS j. $\hat{u}_{\tau,j}\left(\boldsymbol{a}_j\right)$ is the average utility function over all SBSs other than SBS j while $\bar{u}_j\left(\boldsymbol{a}_j, \boldsymbol{a}_{-j}\right)$ is the average utility value over all SBSs. From (12), we can see that each SBS will assign the highest probability, $1 - \varepsilon + \frac{\varepsilon}{|\mathcal{A}_j|}$, to the action that results in the maximum utility value, $\hat{u}_{\tau,j}$. For other actions, the SBS will assign the probability $\frac{\varepsilon}{|\mathcal{A}_j|}$. The value of ε determines the convergence speed. In this case, as each SBS maximizes the utility $\hat{u}_{\tau,j}$, the average utility $\bar{u}_j$ reaches maximum. Note that, (12) is used to find the optimal action for each SBS during the training stage. After training, the SBSs will directly select the optimal action that can maximize $\bar{u}_j$. To capture the gain that stems from the change of the number of resource blocks allocated to each user i, we state the following result:

Theorem 1. The gain of user i's VR QoS due to the change of the number of resource blocks allocated to user i is:
i) The gain of user i that stems from the change of the number of uplink resource blocks allocated to user i, ΔU_i^{u}, is:

$$\Delta U_i^{\mathrm{u}} = \frac{\left(\frac{L}{\max\limits_{\boldsymbol{s}_{ij}^{\mathrm{d}}}(c_{ij}(\boldsymbol{s}^{\mathrm{d}}))} - \frac{L}{c_{ij}\left(\boldsymbol{s}_{ij}^{\mathrm{d}}\right)}\right) K_i\left(\boldsymbol{s}_{ij}^{\mathrm{u}} + \Delta \boldsymbol{s}_{ij}^{\mathrm{u}}\right)}{\max\limits_{\boldsymbol{s}_{ij}^{\mathrm{d}}}\left(D_{ij}^{\mathrm{T}}\left(\boldsymbol{s}^{\mathrm{d}}, \boldsymbol{s}_{ij}^{\mathrm{u}} + \Delta \boldsymbol{s}_{ij}^{\mathrm{u}}\right)\right) + D_i^{\mathrm{P}}\left(\boldsymbol{s}_{ij}^{\mathrm{u}} + \Delta \boldsymbol{s}_{ij}^{\mathrm{u}}\right) - \gamma_D}$$
$$- \left(\frac{L}{\max\limits_{\boldsymbol{s}_{ij}^{\mathrm{d}}}\left(c_{ij}\left(\boldsymbol{s}^{\mathrm{d}}\right)\right)} - \frac{L}{c_{ij}\left(\boldsymbol{s}_{ij}^{\mathrm{d}}\right)}\right) \frac{K_i\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right)}{\max\limits_{\boldsymbol{s}_{ij}^{\mathrm{d}}}\left(D_{ij}^{\mathrm{T}}\left(\boldsymbol{s}^{\mathrm{d}}, \boldsymbol{s}_{ij}^{\mathrm{u}}\right)\right) + D_i^{\mathrm{P}}\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right) - \gamma_D}, \tag{13}$$

ii) The gain of user i that stems from the change of number of downlink resource blocks allocated to user i, ΔU_i^{d}, is:

$$\Delta U_i^{\mathrm{d}} = \begin{cases} \frac{K_i\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right) L}{\left(D_{\max,i}\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right) - \gamma_{i,D}\right) c_{ij}\left(\boldsymbol{s}_{ij}^{\mathrm{d}}\right)}, c_{ij}\left(\Delta \boldsymbol{s}_{ij}^{\mathrm{d}}\right) \gg c_{ij}\left(\boldsymbol{s}_{ij}^{\mathrm{d}}\right), \\ \frac{K_i\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right) L c_{ij}\left(\Delta \boldsymbol{s}_{ij}^{\mathrm{d}}\right)}{\left(D_{\max,i}\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right) - \gamma_{i,D}\right) c_{ij}\left(\boldsymbol{s}_{ij}^{\mathrm{d}}\right)}, c_{ij}\left(\Delta \boldsymbol{s}_{ij}^{\mathrm{d}}\right) \ll c_{ij}\left(\boldsymbol{s}_{ij}^{\mathrm{d}}\right), \\ \frac{K_i\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right) L}{\left(D_{\max,i}\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right) - \gamma_{i,D}\right)} \times \frac{c_{ij}\left(\Delta \boldsymbol{s}_{ij}^{\mathrm{d}}\right)}{c_{ij}\left(\boldsymbol{s}_{ij}^{\mathrm{d}}\right)^2 + c_{ij}\left(\boldsymbol{s}_{ij}^{\mathrm{d}}\right) c_{ij}\left(\Delta \boldsymbol{s}_{ij}^{\mathrm{d}}\right)}, \text{else}. \end{cases} \tag{14}$$

Proof. See Appendix A. □

From Theorem 1, we can see that the tracking accuracy, K_i, and the number of uplink resource blocks allocated to user i, will directly affect the VR QoS gain of user i. Therefore, in

order to improve the VR QoS of each user, we can either improve the tracking system or increase the number of the resource blocks allocated to each user according to each user's specific state. Moreover, the gain due to increasing the number of downlink resource blocks depends on the values of data rates $c_{ij}\left(\Delta \boldsymbol{s}_{ij}^{\mathrm{d}}\right)$ and $c_{ij}\left(\boldsymbol{s}_{ij}^{\mathrm{d}}\right)$. Hence, the proposed learning algorithm needs to choose the optimal resource allocation scheme to maximize the VR users' QoS.

• *Input:* The ESN input is a vector $\boldsymbol{x}_{\tau,j} = [x_1, \cdots, x_B]^{\mathrm{T}}$ where x_j represents the index of the probability distribution that SBS j uses at time τ. $\boldsymbol{x}_{\tau,j}$ is then used to estimate the value of $\hat{\boldsymbol{u}}_j$ that captures the average VR QoS of SBS j.

• *ESN Model:* For each SBS j, the ESN model is a learning architecture that can find the relationship between the input $\boldsymbol{x}_{\tau,j}$ and output $\boldsymbol{y}_{\tau,j}$, thus building the function between the SBS's probability distribution and the utility value. Mathematically, the ESN model consists of the output weight matrix $\boldsymbol{W}_j^{\mathrm{out}} \in \mathbb{R}^{|\mathcal{A}_j| \times (N_w + B)}$ and the dynamic reservoir containing the input weight matrix $\boldsymbol{W}_j^{\mathrm{in}} \in \mathbb{R}^{N_w \times B}$, and the recurrent matrix $\boldsymbol{W}_j = \begin{bmatrix} w_{11} & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & w_{N_w N_w} \end{bmatrix}$ with N_w being the number of the dynamic reservoir units. To guarantee that the ESN algorithm can predict the utility values, N_w must be larger than the size of the input vector $\boldsymbol{x}_{\tau,j}$ [22]. Here, the dynamic reservoir is used to store historical ESN information that includes input, reservoir state, and output. Note that the historical ESN information can be used to find a fast converging process from the initial state to the mixed-strategy NE. Here, the number of actions for each SBS determines the output weight matrix and recurrent matrix of each ESN. Next, we derive the number of the actions of each SBS j, $|\mathcal{A}_j|$.

Proposition 1. Given the number of the downlink and uplink resource blocks, S^{d} and S^{u}, as well as the users located in the coverage of SBS j, V_j, the number of actions for each SBS j, $|\mathcal{A}_j|$, is given by:

$$|\mathcal{A}_j| = \begin{pmatrix} S^{\mathrm{d}} - 1 \\ |\mathcal{N}(V_j)| - 1 \end{pmatrix} \sum_{\boldsymbol{n} \in \mathcal{N}(V_j)} \prod_{i=1}^{V_j - 1} \begin{pmatrix} n_i \\ S^{\mathrm{u}} - \sum_{k=1}^{i-1} n_i \end{pmatrix},$$

where $\begin{pmatrix} x \\ y \end{pmatrix} = \frac{x(x-1)\cdots(x-y+1)}{y(y-1)\cdots 1}$ and $\mathcal{N}(V_j) = \left\{ \boldsymbol{n} | \sum_{i=1}^{V_j} n_i = S^{\mathrm{d}}, n_i > 0 \right\}$ with $|\mathcal{N}(V_j)|$ being the number of elements in $\mathcal{N}(V_j)$.

Proof. See Appendix B. □

Based on Proposition 1, we can determine the matrix size for both $\boldsymbol{W}_j^{\mathrm{out}}$ and $\boldsymbol{W}_j$. From Proposition 1, we can see that, as the number of users increases, the number of actions increases. Moreover, the increasing number of resource blocks will also increase the number of actions. From Proposition 1, we can also see that the number of actions in the uplink is much larger than the number of actions in the downlink. This is due to the fact that, in the uplink, the interference of each user changes as the resource blocks allocated to each user vary. However, in the downlink, the actions will not affect the interference of each user.

TABLE II
ESN-BASED LEARNING ALGORITHM FOR RESOURCE ALLOCATION

Inputs: Mixed strategy $\boldsymbol{x}_{\tau,j}$
Initialize: $\boldsymbol{W}_j^{\mathrm{in}}$, $\boldsymbol{W}_j$, $\boldsymbol{W}_j^{\mathrm{out}}$, and $\boldsymbol{y}_j = 0$.
for each time τ **do**.
(a) Estimate the value of the utility function based on (16).
if $\tau = 1$
(b) Set the mixed strategy $\boldsymbol{\pi}_{\tau,j}$ uniformly.
else
(c) Set the mixed strategy $\boldsymbol{\pi}_{\tau,j}$ based on (12).
end if
(d) Broadcast the index of the mixed strategy to other SBSs.
(e) Receive the index of the mixed strategy as input $\boldsymbol{x}_{\tau,j}$.
(f) Perform an action based on $\boldsymbol{\pi}_{\tau,j}$ and calculate the actual utility value.
(e) Update the dynamic reservoir state based on (15).
(g) Update the output weight matrix based on (17).
end for

• *Output:* The output of the ESN-based learning algorithm at time t is a vector of utility values $\boldsymbol{y}_{\tau,j} = \left[y_{\tau,j1}, y_{\tau,j2}, \ldots, y_{\tau,j|\mathcal{A}_j|}\right]$. Here, $y_{\tau,ji}$ represents the estimated value of utility $\hat{u}_{\tau,j}\left(\boldsymbol{a}_{ji}\right)$.

B. ESN-Based Learning Algorithm for Resource Allocation

We now present the proposed ESN-based learning algorithm to find a mixed strategy NE. The proposed learning algorithm can find an optimal convergence path from any initial state to a mixed-strategy NE. In particular, the proposed algorithm enables each SBS to reach a mixed-strategy NE by traversing minimum number of strategies after training. In order to find the optimal convergence path, the proposed algorithm must store the past ESN information that consists of input, reservoir states, and output. The past ESN information from time 0 up until time τ is stored by the dynamic reservoir state $\boldsymbol{\mu}_{\tau,j}$. The dynamic reservoir state of SBS j at time τ is:

$$\boldsymbol{\mu}_{\tau,j} = f\left(\boldsymbol{W}_j \boldsymbol{\mu}_{\tau-1,j} + \boldsymbol{W}_j^{\mathrm{in}} \boldsymbol{x}_{\tau,j}\right), \tag{15}$$

where $f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$ is the tanh function. From (15), we can see that the dynamic reservoir state consists of the past dynamic reservoir states and the mixed strategy at time τ. Thus, the dynamic reservoir state actually stores the mixed strategy from time 0 to time τ. Based on the dynamic reservoir state, the proposed ESN algorithm will combine with the output weight matrix to estimate the value of conditional utility value. The estimation of the utility value can be given by:

$$\boldsymbol{y}_{\tau,j} = \boldsymbol{W}_{\tau,j}^{\mathrm{out}} \begin{bmatrix} \boldsymbol{\mu}_{\tau,j} \\ \boldsymbol{x}_{\tau,j} \end{bmatrix}, \tag{16}$$

where $\boldsymbol{W}_{\tau,j}^{\mathrm{out}}$ is the output weight matrix at time slot τ. To enable the ESN to use reservoir state $\boldsymbol{\mu}_{\tau,j}$ to predict the utility value, $\hat{u}_{\tau,ji}$, due to action $\boldsymbol{a}_{ji}$, we must train the output matrix $\boldsymbol{W}_j^{\mathrm{out}}$ using a linear gradient descent approach, which is:

$$\boldsymbol{W}_{\tau+1,ji}^{\mathrm{out}} = \boldsymbol{W}_{\tau,ji}^{\mathrm{out}} + \lambda \left(\hat{u}_{\tau,ji} - y_{\tau,ji}\left(\boldsymbol{x}_{\tau,j}^{j}, \boldsymbol{a}_{ji}\right)\right) \boldsymbol{\mu}_{\tau,j}^{\mathrm{T}}, \tag{17}$$

where $\boldsymbol{W}^{\text{out}}_{\tau,ji}$ is row i of $\boldsymbol{W}^{\text{out}}_{\tau,j}$, λ is the learning rate, and $\hat{u}_{\tau,ji}$ is the actual utility value. Here, $y_{\tau,ji}$ is the estimation of the utility value $\hat{u}_{\tau,ji}$. Table II summarizes our algorithm.

C. Convergence of the ESN-Based Learning Algorithm

Now, we analyze the convergence of the proposed ESN-based learning algorithm.

Theorem 2. The proposed ESN-based learning algorithm converges to the utility value, $\hat{\boldsymbol{u}}_j$, if any following conditions is satisfied:
i) λ is a constant and $\min_{\boldsymbol{W}^{\text{in}}_{ji},\boldsymbol{x}_{\tau,j},\boldsymbol{x}'_{\tau,j}} \boldsymbol{W}^{\text{in}}_{ji}\left(\boldsymbol{x}_{\tau,j}-\boldsymbol{x}'_{\tau',j}\right) \geq 2$, where $\boldsymbol{W}^{\text{in}}_{ji}$ represents the row i of $\boldsymbol{W}^{\text{in}}_{j}$.
ii) λ satisfies the Robbins-Monro conditions [33] ($\lambda(t) > 0, \sum_{t=0}^{\infty}\lambda(t) = +\infty, \sum_{t=0}^{\infty}\lambda^2(t) < +\infty$).

Proof. See Appendix C. □

From Theorem 2, we can see that the convergence of the proposed algorithm depends on the values of the input weight matrix $\boldsymbol{W}^{\text{in}}_{ji}$ and the input $\boldsymbol{x}_{\tau,j}$. These values also affect the capacity of the ESN's memory. Here, the memory of a ESN represents the ability that an ESN can store the past ESN information. Therefore, the proposed algorithm of SBS j can converge to the conditional utility function $\hat{\boldsymbol{u}}_j$ by choosing appropriate $\boldsymbol{W}^{\text{in}}_{ji}$ and $\boldsymbol{x}_{\tau,j}$. Indeed, the proposed learning algorithm can converge to $\hat{\boldsymbol{u}}_j$ even when $\boldsymbol{W}^{\text{in}}_{ji}$ and $\boldsymbol{x}_{\tau,j}$ are generated randomly. This is due to the fact that the probability of $\boldsymbol{u}_{\tau,j} = \boldsymbol{u}'_{\tau',j}$ is particularly small since $\boldsymbol{u}_{\tau,j}$ has a larger number of elements (i.e. more than 500). Here, we note that, compared to our work in [31] that uses two echo state networks to guarantee convergence, the proposed ESN algorithm uses only one echo state network and is still guaranteed to converge as shown in Theorem 2. Moreover, this new proof of convergence, we can invoke our result in the fact that the algorithm will reach a mixed NE follows directly from [31, Theorem 2] to guarantee that the convergence point will be a mixed NE, as follows.

Corollary 1. The ESN-based learning algorithm of each SBS j converges to a mixed-strategy Nash equilibrium, with the mixed strategy probability $\boldsymbol{\pi}^*_j$, $\forall j \in \mathcal{B}$.

Based on (12), each SBS will assign the highest probability to the action that results in the maximum value of $\hat{u}_{j,\max} = \max_{\boldsymbol{a}_j \in \mathcal{A}_j} \hat{u}_{\tau,j}(\boldsymbol{a}_j)$. Therefore, when each SBS j reaches the optimal mixed strategy $\boldsymbol{\pi}^*_j$ and the ESN reaches the maximal utility $\hat{\boldsymbol{u}}_j$, the maximum value of the average utility $\bar{u}_j$ can be given by $\left(1-\varepsilon+\frac{\varepsilon}{|\mathcal{A}_j|}\right)\hat{u}_{j,\max}$. Since the mixed-strategy NE depends on the utility values that are not unique, then, the resulting mixed-strategy NE is not unique. While characterizing all possible NEs is challenging analytically, in our simulations in Section IV, we will analyze the performance of different NEs.

D. Implementation and Complexity

The proposed algorithm can be implemented in a distributed way. At the initial state, the reservoir and output of the ESN will be zero. During each iteration, the output and reservoir of the ESN will be updated based on (15)-(16). Based on the ESN's output, each SBS will update its mixed strategy and broadcast the index of this mixed strategy to other SBSs. Compared to the size of VR content and tracking information, the size of the index of the strategy is very small (it can be represented with less than 16 bits). Therefore, these interactions between SBSs are independent of the network size and, hence, they incur no notable overhead. The resulting mixed-strategy NE depends on the utility value resulting from each action of each SBS.

The objective of our game is to find the equilibrium mixed strategy for each SBS. Hence, the complexity of the proposed algorithm depends on the number of mixed strategies. Based on (12), we can see that the number of mixed strategies is equal to the number of actions. Since the worst-case for each SBS is to traverse all actions, the worst-case complexity of the proposed algorithm is $O(|\mathcal{A}_1| \times \cdots \times |\mathcal{A}_B|)$. However, the worst-case complexity pertains to a rather unlikely scenario in which all SBSs choose their optimal probability strategies after traversing all other mixed strategies during each period τ and, hence, the probability of occurrence of the worst-case scenario is $\left(1-\frac{\varepsilon}{|\mathcal{A}_1|}\right)^{|\mathcal{A}_1|-1} \times \cdots \times \left(1-\frac{\varepsilon}{|\mathcal{A}_B|}\right)^{|\mathcal{A}_B|-1}$[3]. Therefore, the proposed algorithm will converge faster than the worst-case with probability $1-\left(1-\frac{\varepsilon}{|\mathcal{A}_1|}\right)^{|\mathcal{A}_1|-1} \times \cdots \times \left(1-\frac{\varepsilon}{|\mathcal{A}_B|}\right)^{|\mathcal{A}_B|-1}$. Moreover, as the number of SBSs increases, based on Proposition 1, the number of actions for each SBS significantly decreases. Thus, the worst-case complexity of the proposed algorithm will also decrease. In addition, our ESN-based learning algorithm can use the past information to predict the value of utility $\hat{\boldsymbol{u}}_j$ and, hence, each SBS can obtain the VR QoS of each SBS without implementing all of its possible actions, which will also reduce the complexity of the proposed algorithm. During the convergence process, the proposed algorithm needs to harmonize the tradeoff between exploration and exploitation. Exploration is used to enable each SBS to explore actions so as to find a better solution. Exploitation refers to the case in which each SBS will use the current optimal action at this iteration. This tradeoff is controlled by the ε-greedy exploration specificed in (12). If we increase the probability of exploration, the proposed algorithm will use more actions and, hence, the number of iterations that the proposed algorithm requires to reach a mixed-strategy NE will decrease. However, increasing the probability of exploration may reduce the VR QoS of each user when the selected action is worse than the current optimal action.

[3] Based on (12), for each SBS, the probability that the optimal action is not selected at each iteration is $\left(1-\frac{\varepsilon}{|\mathcal{A}_j|}\right)$ and hence, the probability that the optimal action is selected at the last iteration is $\left(1-\frac{\varepsilon}{|\mathcal{A}_j|}\right)^{|\mathcal{A}_j|-1}$. Therefore, the probability of all SBSs select their optimal action at last iteration is $\left(1-\frac{\varepsilon}{|\mathcal{A}_1|}\right)^{|\mathcal{A}_1|-1} \times \cdots \times \left(1-\frac{\varepsilon}{|\mathcal{A}_B|}\right)^{|\mathcal{A}_B|-1}$.

TABLE III
SYSTEM PARAMETERS

Parameter	Value	Parameter	Value	Parameter	Value
F	1000	P_B	20 dBm	v	5
B	4	S^{u}	5	S^{d}	5
N_w	1000	σ^2	-95 dBm	r_B	25 m
N_v	6	λ	0.03	A	100 kB

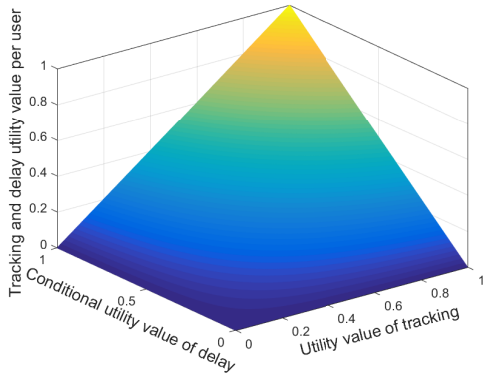


Fig. 2. Total VR QoS utility of each user vs. the tracking and delay utilities. Here, total VR QoS utility refers to (6).

IV. Simulation Results and Analysis

For our simulations, we consider an SCN deployed within a circular area with radius $r = 100$ m. $U = 25$ users and $B = 4$ SBSs are uniformly distributed in this SCN area. The HTC Vive VR device is considered in our simulations and, hence, the number of pixels for a panoramic image is 1920×1080 and each pixel is stored in 16 bits [34]. The flashed rate which represents the update rate of a VR image, is 60 images per second and the factor of compression is 150 [4]. Since two panoramic images consist of one VR image (one panoramic image per eye), the rate requirement for wireless VR transmission will be 25.32 Mbit/s[4]. The bandwidth of each resource block is set to 10×180 kHz [35]. We use a standard H.264/AVC video codec. The VR video is encoded at 60 frames per second and the encoding rate of each VR video is 4 Mbps. The detailed simulation parameters are listed in Table III. For comparison purposes, we use three baselines: a) Proportional fair algorithm in [36] that allocates the resource blocks based on the number of the resource blocks needed to maximize each user's utility value, b) Q-learning algorithm in [37] that has considered the historical utility value to estimate the future utility value, and c) Optimal heuristic search algorithm. Note that, in order to compare with the proportional fair algorithm, all SBSs choose the action with highest probability among the mixed strategy when the proposed algorithm reaches a mixed-strategy NE. This is used for all results in which we compare to proportional fair. Here, the users' localization data is measured from actual wired HTC Vive VR devices and the wireless transmission is simulated, in order to compute the tracking accuracy. All statistical results are averaged over a large number of independent runs.

[4]Here, for each second, each SBS needs to transmit $1920 \times 1080 \times 16 \times 60 \times 2 = 3,981,312,000$ bits $= 3796.875$ Mbits to each user. Since the factor of compression is 150, the rate requirement is $3796.875.75/150 = 25.3125$ Mbit/s.

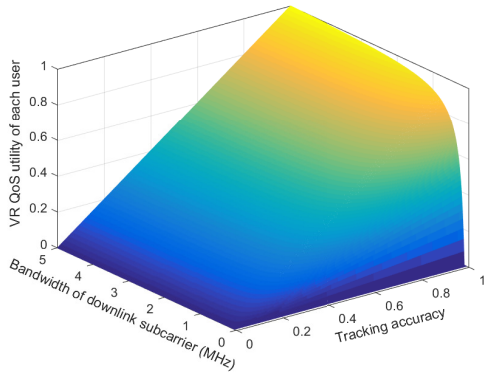
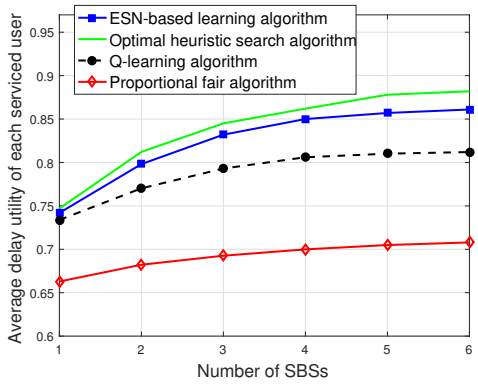


Fig. 3. Total VR QoS utility of each user vs. the tracking and delay utilities. Here, total VR QoS utility refers to (6).

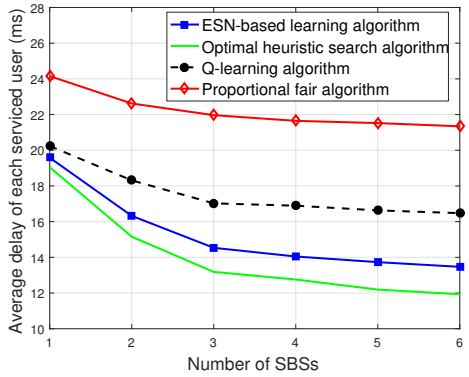
Fig. 2 shows how each user's VR QoS utility varies as the tracking and delay utilities change. In Fig. 2, different colors indicate different total VR QoS utilities. From Fig. 2, we can see that, when the delay (tracking) utility is 0, the total VR QoS utility will be 0 regardless of the tracking (delay) value. Thus, both tracking accuracy and delay will affect the VR QoS. In Fig. 2, we can also see that only when both tracking and delay utilities are 1, the total VR QoS utility is maximized. This is due to the fact that the multi-attribute utility theory model assigns to each tracking and delay components of the VR QoS a unique value. Clearly, it is clear that, the proposed total utility function can effectively capture the VR QoS.

Fig. 3 shows how each user's total utility changes as function of the tracking accuracy and the bandwidth of each downlink resource block group (Fig. 3 uses a similar color legend as Fig. 2). From Fig. 3, we can see that when the bandwidth of downlink resource block group (tracking accuracy) is 0, the total VR QoS utility is 0 regardless of the tracking accuracy (bandwidth of a downlink resource block group). This is due to the fact that the VR QoS depends on both delay and tracking. This corresponds to a scenario in which SBS j has enough downlink bandwidth to send a VR image to the user while the tracking information is inaccurate. In this case, SBS j cannot construct the accurate VR image due to the inaccuracy of user's localization and, hence, the VR QoS of this user will be 0. Fig. 3 also shows that, when the tracking accuracy is 1 and the bandwidth of a downlink resource block group is over 4 MHz, the total VR QoS utility will be maximized. This verifies the result of Theorem 1.

In Fig. 4, we show how the average delay utility for each user varies as the number of SBSs increases. From Figs. 4(a) and 4(b), we can see that, as the number of SBSs increases, the average delay utility for each serviced user increases and the transmission delay of each user decreases. This is due to the fact that as the number of SBSs increases, the number of users located in each SBS's coverage decreases and, hence, the average delay utility increases. However, as the number of SBSs keeps increasing, the average delay utility increases slowly. This stems from the fact that the interference from the SBSs to the users increases as the number of SBSs continues to increase. Fig. 4(b) also shows that the proposed algorithm



(a) Average delay utility of each serviced user



(b) Average delay of each serviced user

Fig. 4. Delay for each serviced user vs. the number of SBSs.

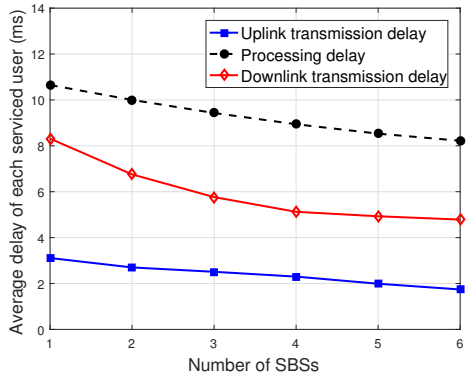


Fig. 5. Delay for each serviced user vs. the number of SBSs.

achieves up to 19.6% gain in terms of average delay compared to the Q-learning algorithm for the case with 6 SBSs. In Fig. 4(b), we can also see that the proposed ESN-based learning algorithm enables the wireless VR transmission to meet typical delay requirement of VR applications that consists of both the transmission delay and processing delay (typically 20 ms [38]). These gains stem from the fact that the proposed algorithm uses the past ESN information stored at the ESN model to find a better solution for the proposed game.

Fig. 5 shows how the transmission and processing delays change as the number of the SBSs varies. From Fig. 5, we can see that all delay components of each user decrease as the number of SBSs increases. This is due to the fact that,

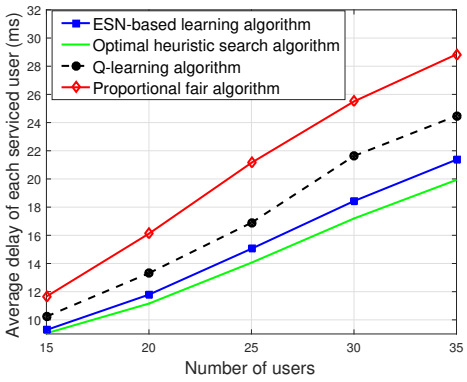


Fig. 6. Average delay for each serviced user as the number of users varies.

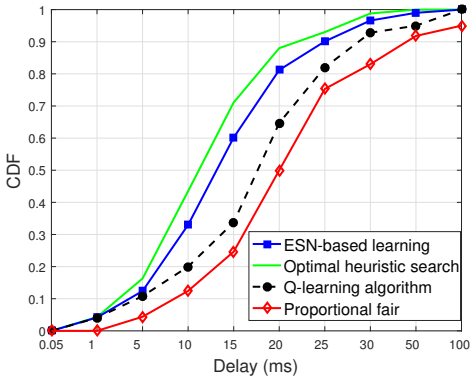
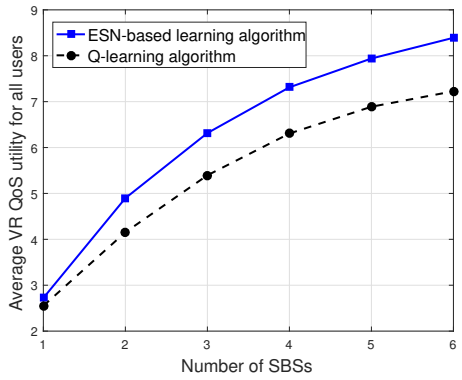


Fig. 7. CDFs of the delay resulting from the different algorithms.

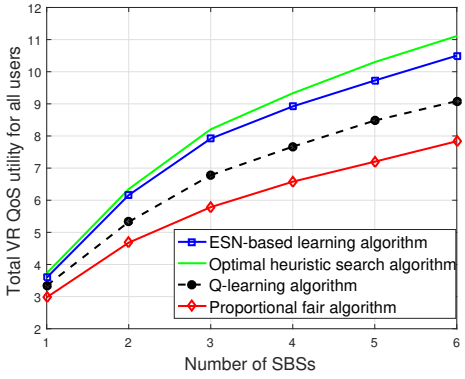
as the number of SBSs increases, the users will have more SBS choices and the distances from the SBSs to the users decrease, and, hence, the SINR and the potential resources (resource blocks) allocated to each user will increase.

In Fig. 6, we show how the average delay for each serviced user changes as the number of VR users varies. From Fig. 6, we can see that, as the number of VR users increases, the average delay of each VR user increases. This is due to the fact that, as the number of users increases, the number of resource blocks that is allocated to each VR user decreases. Fig. 6 also shows that the deviation between the proposed algorithm and Q-learning increases as the number of VR users increases. This is due to the fact that, as the number of users increases, the number of the users associated with each SBS increases. In consequence, the number of actions for each SBS increases and, hence, the SBSs need to record more QoS values resulting from these actions. Compared to Q-learning that uses a matrix to record the QoS values, the proposed algorithm uses an ESN to approximate the function of QoS values and, hence, the proposed algorithm can record more QoS utility values compared to Q-learning. Fig. 6 also shows that the proposed algorithm can yield 29.8% gain of the average delay compared to proportional fair algorithm. This gain stems from the fact that the proposed algorithm can find the relationship between resource block allocation strategies and utility values so as to maximize the utility values.

Fig. 7 shows the cumulative distribution function (CDF) for the total delay resulting from all of the considered schemes.



(a) Average total VR QoS utility for all users.



(b) Total VR QoS utility for all users

Fig. 8. VR QoS for all users vs. the number of SBSs. Here, total VR QoS utility refers to (8). The proportional fair algorithm and optimal heuristic search do not use mixed strategies and, hence, they are not shown in Fig. 8(a).

In Fig. 7, we can see that, the delay of 100% of users resulting from all of the considered algorithms will be above 0.05 ms. This is due to the fact that, the delay requirement of each user is higher than 0.05 ms and, hence, when the user's delay requirement is satisfied, the SBSs will allocate the resource blocks to other users. Fig. 7 also shows that the proposed approach improves the CDF of up to 25% and 50% gains at a delay of 20 ms compared to Q-learning and proportional fair algorithm. These gains stem from the fact that the proposed algorithm can estimate the VR QoS resulting from each SBS's actions accurately and, hence, can find a better solution compared to Q-learning and proportional fair algorithms. Fig. 7 also shows that there exists a delay variance for users who achieve the 20 ms delay target. If needed, the network can reduce this variance by adjusting the size of the resource blocks.

Fig. 8 shows how the VR QoS for all users changes as the number of SBSs varies. From Figs. 8(a) and 8(b), we can see that both total utility values and average total utility values (at the mixed-strategy NE) of all considered algorithms increase as the number of SBSs increases. This is due to the fact that, as the number of SBSs increases, the number of users located within the coverage of each SBS increases and the distances from the SBSs to their associated users decrease. Fig. 8(a)

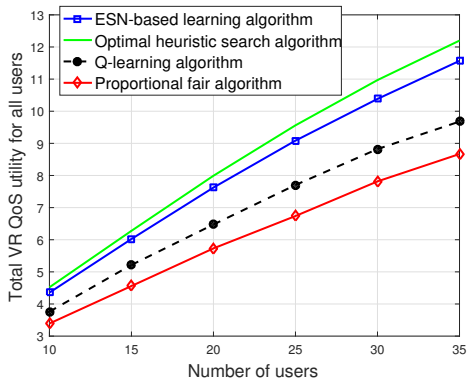


Fig. 9. The total VR QoS utility for all users vs. total number of users. Here, the total VR QoS utility refers to (8).

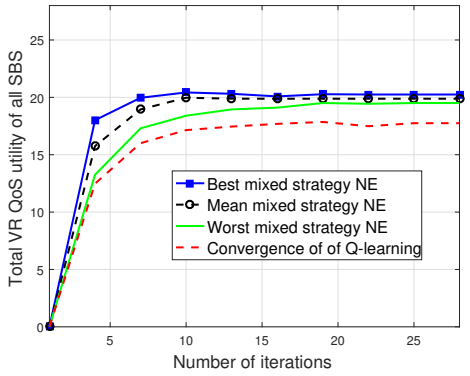


Fig. 10. Convergence of the proposed algorithm and Q-learning. Here, total VR QoS utility refers to (8)

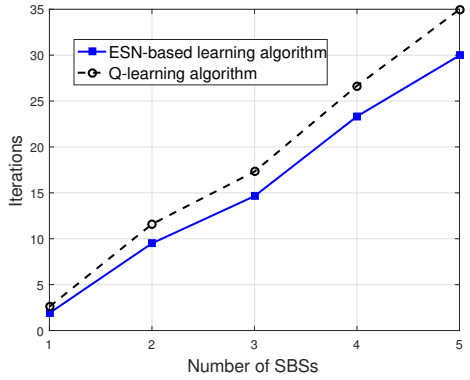
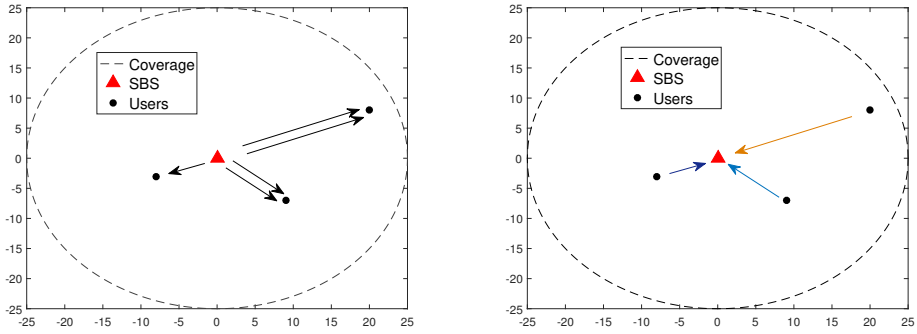


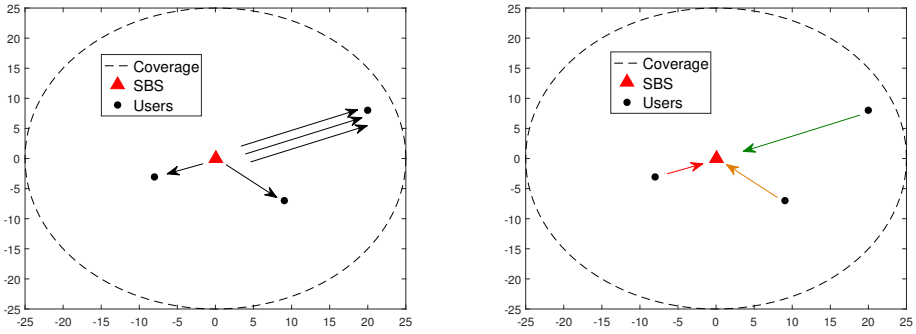
Fig. 11. The convergence time as a function of the number of SBSs.

shows that the proposed algorithm can yield up to of 15.3% gain in terms of the average of total VR QoS utility compared to the Q-learning for the case with 5 SBSs. In Fig. 8(b), we can also see that the proposed ESN-based learning algorithm achieves, respectively, up to 17.1% and 36.7% improvements in terms of the total utility value compared to Q-learning and proportional fair algorithms for the case with 4 SBSs. These gains are due to the fact that our ESN algorithm can store the past ESN information and use it to build the relationship between the input and output. Hence, the proposed learning algorithm can predict the output (utility value) and, hence, find a better solution for allocating resources.

In Fig. 9, we show how the total utility value of VR QoS for all users changes as the total number of users varies. From



(a) Optimal action of the ESN-based algorithm for the users over downlink and uplink



(b) Optimal action of proportional fair algorithm for the users over downlink and uplink

Fig. 12. Optimal actions resulting from the different algorithms (Each black arrow represents the downlink resource blocks while each color arrow represents an group of uplink resource blocks).

Fig. 9, we can see that, as the number of users increases, the total utility values of all considered algorithms increase. This is due to the fact that, in all algorithms, each SBS has a limited coverage area and, hence, the number of users located in each SBS's coverage increases with the network size. Moreover, since each SBS has a limited number of resource blocks, the number of users that can associate with each SBS is also bounded. In particular, as the number of users located within the coverage of a given SBS exceeds the maximal number of users that each SBS can provide service, the SBS will only service the users that can maximize the total utility value. The VR QoS of the remaining users will be 0. In this case, the total utility value will also increase with the number of the users. From Fig. 9, we can also see that the proposed algorithm achieves, respectively, up to 22.2% and 37.5% gains in terms of the total utility value of VR QoS compared to Q-learning and proportional fair algorithms for the case with 35 users.

Fig. 10 shows the number of iterations needed till convergence for both Q-learning and the proposed ESN-based learning approach with different mixed-strategy NEs and Q-learning. In this figure, we can see that, as time elapses, the total VR QoS utilities for both the proposed algorithm and Q-learning increase until convergence to their final values. Fig. 10 also shows that the proposed algorithm for best mixed strategy NE needs 19 iterations to reach convergence while Q-learning needs 25 iterations to reach convergence. Hence, the proposed algorithm achieves 24.2% gain in terms of the number of the iterations needed to reach convergence compared to Q-learning. This is because the ESN in the proposed algorithm can store the SBSs' action strategies and its corresponding total utility values. Fig. 10 also shows that, different mixed-strategy NEs will result in different total utility values of all SBSs. However, the total utility values achieved by these mixed-strategy NEs are very close to each other. Moreover, Fig. 10 also shows that the worst mixed-strategy NE resulting from the proposed algorithm can still achieve 14% gain of the total utility value compared to Q-learning. Hence, regardless of the reached NE, the total utility value of the proposed algorithm will be higher than the utility value achieved by Q-learning.

In Fig. 11, we show how the convergence time changes as the number of SBSs varies. In this figure, we can see that as the number of the SBSs increases, the convergence time of both algorithms increases. Indeed, as the number of SBSs increases, the proposed ESN algorithm will require more time to accurately calculate the VR QoS utility. From Fig. 11, we can also see that as the number of SBSs increases, the difference in the convergence time between the proposed algorithm and Q-learning increases. This stems from the fact that as the number of SBSs increases, the number of actions for each SBS decreases and, hence, the number of output weight matrix used to predict the VR QoS utility for each action decreases.

Fig. 12 shows the optimal actions resulting from the pro-

posed ESN-based learning algorithm and proportional fair algorithm. Here, each color arrow represents a unique group of uplink resource blocks. From Figs. 12(a) and 12(b), we can see that, for downlink resource allocation, the proportional fair algorithm allocates most of the downlink resource blocks to the user located farthest to the SBS while the proposed learning algorithm allocates only two groups of resource blocks to the farthest user. This is due to the fact that the proportional fair algorithm only considers the users' resource blocks demands while the proposed learning algorithm considers how to maximize the total utility values of VR QoS for all associated users. Figs. 12(a) and 12(b) also show that, both the proposed ESN-based learning algorithm and proportional fair algorithm allocate three groups of resource blocks to the farthest user. However, the uplink resource blocks allocated to each user are different. This is due to the fact that, in uplink, the proportional fair algorithm only considers the users' resource blocks demands while ignoring the uplink interference pertaining to the allocation of uplink resource blocks. In this case, the interference of users in uplink will significantly decrease the total VR QoS utility for each user. Note that, since each SBS allocates their all downlink and uplink resource blocks to its associated users, the interference of the users in downlink will not change as the actions vary while the interference in uplink depends on the actions.

V. Conclusion

In this paper, we have developed a novel multi-attribute utility theory based VR model that can capture the tracking and delay components of VR QoS. Based on this model, we have proposed a novel resource allocation framework for optimizing the VR QoS for all users. We have formulated the problem as a noncooperative game between the SBSs that seeks to maximize the average VR QoS utilities for all users. To solve this problem, we have developed a novel algorithm based on the machine learning tools of echo state networks. The proposed algorithm enables each SBS to decide on its actions autonomously according to the users' and networks' states. Moreover, the proposed learning algorithm only needs to update the mixed strategy during the training process and, hence, can quickly converge to a mixed-strategy NE. Simulation results have shown that the proposed VR model can capture the VR QoS in wireless networks while providing significant performance gains.

Appendix

A. Proof of Theorem 1

For i), the gain that stems from increasing the number of uplink resource blocks allocated to user i, ΔU_i^{u}, is:

$$\begin{aligned}
&U_i\Big(D_i\Big(\boldsymbol{s}_{ij}^{\mathrm{d}},\boldsymbol{s}_{ij}^{\mathrm{u}}+\Delta\boldsymbol{s}_{ij}^{\mathrm{u}}\Big),K_i\big(\boldsymbol{s}_{ij}^{\mathrm{u}}+\Delta\boldsymbol{s}_{ij}^{\mathrm{u}}\big)\Big)-U_i\Big(D_i\Big(\boldsymbol{s}_{ij}^{\mathrm{d}},\boldsymbol{s}_{ij}^{\mathrm{u}}\Big),K_i\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right)\Big)\\
=&K_i\left(\boldsymbol{s}_{ij}^{\mathrm{u}}+\Delta\boldsymbol{s}_{ij}^{\mathrm{u}}\right)\times\frac{D_{\max,i}\left(\boldsymbol{s}_{ij}^{\mathrm{u}}+\Delta\boldsymbol{s}_{ij}^{\mathrm{u}}\right)-D_i\left(\boldsymbol{s}_{ij}^{\mathrm{d}},\boldsymbol{s}_{ij}^{\mathrm{u}}+\Delta\boldsymbol{s}_{ij}^{\mathrm{u}}\right)}{D_{\max,i}\left(\boldsymbol{s}_{ij}^{\mathrm{u}}+\Delta\boldsymbol{s}_{ij}^{\mathrm{u}}\right)-\gamma_D}\\
&-K_i\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right)\times\frac{D_{\max,i}\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right)-D_i\left(\boldsymbol{s}_{ij}^{\mathrm{d}},\boldsymbol{s}_{ij}^{\mathrm{u}}\right)}{D_{\max,i}\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right)-\gamma_D},
\end{aligned}\tag{18}$$

where (13) is obtained by substituting (2) and (4) into (18). Since $K_i(x)$ is determined by the bit errors due to the wireless transmission, (13) cannot be further simplified.

For ii), the gain of changing the downlink resource blocks, ΔU_i^{d}, can be given by:

$$\begin{aligned}
\Delta U_i^{\mathrm{d}}&=K_i\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right)\times\frac{D_i\left(\boldsymbol{s}_{ij}^{\mathrm{d}},\boldsymbol{s}_{ij}^{\mathrm{u}}\right)-D_i\left(\boldsymbol{s}_{ij}^{\mathrm{d}}+\Delta\boldsymbol{s}_{ij}^{\mathrm{d}},\boldsymbol{s}_{ij}^{\mathrm{u}}\right)}{D_{\max}\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right)-\gamma_D}\\
&=K_i\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right)\times\frac{D_i^T\left(\boldsymbol{s}_{ij}^{\mathrm{d}}\right)-D_i^T\left(\boldsymbol{s}_{ij}^{\mathrm{d}}+\Delta s_{ij}^{\mathrm{d}}\right)}{D_{\max,i}\left(s_{ij}^{\mathrm{u}}\right)-\gamma_D},\\
&=\frac{K_i\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right)}{D_{\max,i}\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right)-\gamma_D}\times\frac{Lc_{ij}\left(\Delta\boldsymbol{s}_{ij}^{\mathrm{d}}\right)}{c_{ij}\left(\boldsymbol{s}_{ij}^{\mathrm{d}}\right)c_{ij}\left(\boldsymbol{s}_{ij}^{\mathrm{d}}+\Delta\boldsymbol{s}_{ij}^{\mathrm{d}}\right)}\\
&=\frac{K_i\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right)L}{D_{\max,i}\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right)-\gamma_D}\times\frac{c_{ij}\left(\Delta\boldsymbol{s}_{ij}^{\mathrm{d}}\right)}{c_{ij}\left(\boldsymbol{s}_{ij}^{\mathrm{d}}\right)^2+c_{ij}\left(\boldsymbol{s}_{ij}^{\mathrm{d}}\right)c_{ij}\left(\Delta\boldsymbol{s}_{ij}^{\mathrm{d}}\right)}.
\end{aligned}$$

Here, when $c_{ij}\left(\Delta\boldsymbol{s}_{ij}^{\mathrm{d}}\right)\gg c_{ij}\left(\boldsymbol{s}_{ij}^{\mathrm{d}}\right)$, $\frac{c_{ij}\left(\Delta\boldsymbol{s}_{ij}^{\mathrm{d}}\right)}{c_{ij}\left(\boldsymbol{s}_{ij}^{\mathrm{d}}\right)^2+c_{ij}\left(\boldsymbol{s}_{ij}^{\mathrm{d}}\right)c_{ij}\left(\Delta\boldsymbol{s}_{ij}^{\mathrm{d}}\right)}\approx\frac{1}{c_{ij}\left(\boldsymbol{s}_{ij}^{\mathrm{d}}\right)}$, and, consequently, $\Delta U_i^{\mathrm{d}}=\frac{K_i\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right)L}{\left(D_{\max,i}\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right)-\gamma_D\right)c_{ij}\left(\boldsymbol{s}_{ij}^{\mathrm{d}}\right)}$. Moreover, as $c_{ij}\left(\Delta\boldsymbol{s}_{ij}^{\mathrm{d}}\right)\ll c_{ij}\left(\boldsymbol{s}_{ij}^{\mathrm{d}}\right)$, $\frac{c_{ij}\left(\Delta\boldsymbol{s}_{ij}^{\mathrm{d}}\right)}{c_{ij}\left(\boldsymbol{s}_{ij}^{\mathrm{d}}\right)^2+c_{ij}\left(\boldsymbol{s}_{ij}^{\mathrm{d}}\right)c_{ij}\left(\Delta\boldsymbol{s}_{ij}^{\mathrm{d}}\right)}\approx\frac{c_{ij}\left(\Delta\boldsymbol{s}_{ij}^{\mathrm{d}}\right)}{c_{ij}\left(\boldsymbol{s}_{ij}^{\mathrm{d}}\right)^2}$. Thus, $\Delta U_i^{\mathrm{d}}=\frac{K_i\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right)Lc_{ij}\left(\Delta\boldsymbol{s}_{ij}^{\mathrm{d}}\right)}{\left(D_{\max,i}\left(\boldsymbol{s}_{ij}^{\mathrm{u}}\right)-\gamma_D\right)c_{ij}\left(\boldsymbol{s}_{ij}^{\mathrm{d}}\right)}$. This completes the proof.

B. Proof of Proposition 1

To prove Proposition 1, we first need to prove that the number of actions for the users over the downlink is $\binom{S^{\mathrm{d}}-1}{|\mathcal{N}(V_j)|-1}$. Since the SBSs will allocate all downlink resource blocks to their associated users, the interference from each SBS to its associated users is unchanged when the actions change. For example, the interference when SBS j allocates resource block 1 to user i is the same as the interference when SBS j allocates resource block 2 to user i. Therefore, we only need to consider the number of downlink resource blocks allocated to each user and, consequently, the number of actions for the users over the downlink is $\binom{S^{\mathrm{d}}-1}{|\mathcal{N}(V_j)|-1}$. Then, we need to prove that the number of actions for the users over the uplink is $\sum_{\boldsymbol{n}\in\mathcal{N}(V_j)}\prod_{i=1}^{V_j-1}\binom{n_i}{S^{\mathrm{u}}-\sum_{k=1}^{i-1}n_i}$. For each vector $\boldsymbol{n}$, SBS j has $\binom{n_1}{S^{\mathrm{u}}}$ actions to allocate the resource blocks to the first user. Based on the resource blocks allocated to the first user, SBS j will have $\binom{n_2}{S^{\mathrm{u}}-n_1}$ actions to allocate the resource blocks to the second user. For other associated users, the number of actions can be derived using a similar method as the method of the second user. Therefore, the number of actions for the users over uplink is $\sum_{\boldsymbol{n}\in\mathcal{N}(V_j)}\prod_{i=1}^{V_j-1}\binom{n_i}{S^{\mathrm{u}}-\sum_{k=1}^{i-1}n_i}$, and, hence, the number of actions for the users over uplink and downlink is $\binom{S^{\mathrm{d}}-1}{|\mathcal{N}(V_j)|-1}\sum_{\boldsymbol{n}\in\mathcal{N}(V_j)}\prod_{i=1}^{V_j-1}\binom{n_i}{S^{\mathrm{u}}-\sum_{k=1}^{i-1}n_i}$. This completes the proof.

C. Proof of Theorem 2

In order to prove this theorem, we first need to prove that the ESN-based learning algorithm converges to a constant value. Here, we do not know the exact value to which the proposed algorithm converges. Hence, our purpose is to prove that the proposed algorithm cannot diverge. Then, we derive the exact value to which the ESN converges. For i), based on [33, Theorem 8], the conditions of convergence for an ESN are: a) The ESN is k-step unambiguous and b) The ESN-based learning process is k order Markov decision process (MDP). Here, the definition of k-step unambiguous can be given as follows:

Definition 2. Given an ESN with initial state $\boldsymbol{\mu}_{0,j}$, we assume that the input sequence $\boldsymbol{x}_{0,j}, \ldots, \boldsymbol{x}_{\tau,j}$ results in an internal state $\boldsymbol{\mu}_{\tau,j}$, and the input sequence $\boldsymbol{x}'_{0,j}, \ldots, \boldsymbol{x}'_{\tau',j}$ results in an internal state $\boldsymbol{\mu}'_{\tau',j}$. If $\boldsymbol{\mu}_{\tau,j} = \boldsymbol{\mu}'_{\tau',j}$ implies that $\boldsymbol{x}_{\tau-i,j} = \boldsymbol{x}'_{\tau'-i,j}$, for all $i = 0, \ldots, \tau$, then the ESN is k-step unambiguous.

Here, $\boldsymbol{u}_{\tau,j} = \boldsymbol{u}'_{\tau',j}$ can be rewritten as:

$$\begin{aligned}\boldsymbol{u}_{\tau,j} - \boldsymbol{u}'_{\tau',j} &= \boldsymbol{W}_j \left(\boldsymbol{\mu}_{\tau-1,j} - \boldsymbol{\mu}'_{\tau'-1,j}\right) + \boldsymbol{W}_j^{\text{in}} \left(\boldsymbol{x}_{\tau,j} - \boldsymbol{x}'_{\tau',j}\right) \\ &= \begin{bmatrix} w_{11}\left(\mu_{\tau-1,j1} - \mu'_{\tau'-1,j1}\right) \\ \vdots \\ w_{N_w N_w}\left(\mu_{\tau-1,jN_w} - \mu'_{\tau'-1,jN_w}\right) \end{bmatrix} \\ &\quad - \begin{bmatrix} \boldsymbol{W}_{j1}^{\text{in}}\left(\boldsymbol{x}_{\tau,j} - \boldsymbol{x}'_{\tau',j}\right) \\ \vdots \\ \boldsymbol{W}_{jN_w}^{\text{in}}\left(\boldsymbol{x}_{\tau,j} - \boldsymbol{x}'_{\tau',j}\right) \end{bmatrix},\end{aligned}$$

where $\mu_{\tau-1,jk}$ is element k of $\boldsymbol{\mu}_{\tau-1,j}$ and $\mu'_{\tau'-1,jk}$ element k of $\boldsymbol{\mu}'_{\tau'-1,j}$. Since the tanh function in (15) ranges from -1 to 1, the maximum value of $\left(\mu_{\tau-1,jk} - \mu'_{\tau'-1,jk}\right)$ is 2. As $w_{kk} \in (-1, 1), k = 1, \ldots, N_w$, $\max_k w_{kk}\left(\mu_{\tau-1,jk} - \mu'_{\tau-1,jk}\right) < 2$. In this case, if $\boldsymbol{W}_{jk}^{\text{in}}\left(\boldsymbol{x}_{\tau,j} - \boldsymbol{x}'_{\tau,j}\right) \geq 2$, then $\boldsymbol{\mu}_{\tau,j} - \boldsymbol{\mu}'_{\tau',j} \neq 0$. Therefore, if $\boldsymbol{\mu}_{\tau,j} - \boldsymbol{\mu}'_{\tau',j} = 0$, then $\boldsymbol{\mu}_{\tau-1,j} = \boldsymbol{\mu}'_{\tau'-1,j}$ and $\boldsymbol{x}_{\tau,j} = \boldsymbol{x}'_{\tau',j}$. In this case, an ESN is k-step unambiguous when $\boldsymbol{W}_{ji}^{\text{in}}\left(\boldsymbol{x}_{\tau,j} - \boldsymbol{x}'_{\tau,j}\right) \geq 2$. Since the dynamic reservoir can only store limited ESN information [30], the dynamic reservoir state $\boldsymbol{\mu}_{\tau,j}$ only depends on the finite past reservoir states, i.e., $\boldsymbol{\mu}_{\tau-1,j}, \ldots, \boldsymbol{\mu}_{\tau-k,j}$. Moreover, the number of reservoir states and actions in the proposed algorithm is finite. Therefore, the proposed ESN-based algorithm is a k order MDP and, hence, condition 2) is satisfied. For case 2), if the learning rate of the proposed algorithm satisfies Robbins-Monro conditions and the proposed algorithm is a k order MDP, the proposed algorithm will satisfy the conditions in [31, Theorem 1] and, hence, converges to a region. For both cases i) and ii), based on [31, Theorem 1], the proposed ESN-based learning algorithm will converge to the utility value, $\hat{\boldsymbol{u}}_j$. This completes the proof.

REFERENCES

[1] M. Chen, W. Saad, and C. Yin, "Resource management for wireless virtual reality: Machine learning meets multi-attribute utility," in *Proc. of IEEE Global Communications Conference (GLOBECOM)*, Singapore, Dec. 2017.

[2] P. Rosedale, "Virtual reality: The next disruptor: A new kind of worldwide communication," *IEEE Consumer Electronics Magazine*, vol. 6, no. 1, pp. 48–50, Jan. 2017.

[3] HTC, "HTC Vive," https://www.vive.com/us/.

[4] E. Baştuğ, M. Bennis, M. Médard, and M. Debbah, "Towards interconnected virtual reality: Opportunities, challenges and enablers," *IEEE Communications Magazine*, vol. 55, no. 6, pp. 110–117, June 2017.

[5] J. Wu, B. Cheng, C. Yuen, Y. Shang, and J. Chen, "Distortion-aware concurrent multipath transfer for mobile video streaming in heterogeneous wireless networks," *IEEE Transactions on Mobile Computing*, vol. 14, no. 4, pp. 688–701, April 2015.

[6] J. Wu, C. Yuen, N. M. Cheung, J. Chen, and C. W. Chen, "Modeling and optimization of high frame rate video transmission over wireless networks," *IEEE Transactions on Wireless Communications*, vol. 15, no. 4, pp. 2713–2726, April 2016.

[7] F. Qian, L. Ji, B. Han, and V. Gopalakrishnan, "Optimizing 360 video delivery over cellular networks," in *Proc. of All Things Cellular (ATC) Workshop on Operations, Applications and Challenges*, New York, USA, Oct. 2016.

[8] C. Richardt, Y. Pritch, H. Zimmer, and A. Sorkine-Hornung, "Megastereo: Constructing high-resolution stereo panoramas," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, Portland, OR, USA, June 2013.

[9] S. Sridhar, F. Mueller, M. Zollhöfer, D. Casas, A. Oulasvirta, and C. Theobalt, "Real-time joint tracking of a hand manipulating an object from RGB-D input," in *Proc. of European Conference on Computer Vision*, Amsterdam, Netherlands, Oct. 2016.

[10] A. Rovira and M. Slater, "Reinforcement learning as a tool to make people move to a specific location in immersive virtual reality," *International Journal of Human-Computer Studies*, vol. 98, pp. 89–94, Feb. 2017.

[11] T. Pfeiffer and C. Memili, "Model-based real-time visualization of realistic three-dimensional heat maps for mobile eye tracking and eye tracking in virtual reality," in *Proc. of the Ninth Biennial ACM Symposium on Eye Tracking Research & Applications*, Charleston, South Carolina, Mar. 2016.

[12] W. C. Lo, C. L. Fan, S. C. Yen, and C. H. Hsu, "Performance measurements of 360° video streaming to head-mounted displays over live 4G cellular networks," in *Proc. of Asia-Pacific Network Operations and Management Symposium (APNOMS)*, Seoul, South Korea, Sept. 2017.

[13] O. Abari, D. Bharadia, A. Duffield, and D. Katabi, "Cutting the cord in virtual reality," in *Proc. of the ACM Workshop on Hot Topics in Networks*, Atlanta, GA, USA, Nov. 2016.

[14] A. E. Abbas, "Constructing multiattribute utility functions for decision analysis," *INFORMS Tutorials in Operations Research*, pp. 62–98, Oct. 2010.

[15] M. Chen, M. Mozaffari, W. Saad, C. Yin, M. Debbah, and C. S. Hong, "Caching in the sky: Proactive deployment of cache-enabled unmanned aerial vehicles for optimized quality-of-experience," *IEEE Journal on Selected Areas on Communications (JSAC)*, vol. 35, no. 5, pp. 1046–1061, May 2017.

[16] S. B. Chekroun, E. Sabir, A. Kobbane, H. Tembine, E. H. Bouyakhf, and K. Ibrahimi, "A distributed open-close access for small-cell networks: A random matrix game analysis," in *Proc. of International Wireless Communications and Mobile Computing Conference (IWCMC)*, Dubrovnik, Croatia, 2015.

[17] M. Bennis, S. Guruacharya, and D. Niyato, "Distributed learning strategies for interference mitigation in femtocell networks," in *Proc. of IEEE Global Communications Conference (GLOBECOM)*, Kathmandu, Nepal, Dec. 2011.

[18] Q. Zhu, W. Saad, Z. Han, H. V. Poor, and T. Başar, "Eavesdropping and jamming in next-generation wireless networks: A game-theoretic approach," in *Proc. of Military Communications Conference*, Baltimore, MD, USA, 2011, pp. 119–124.

[19] J. G. Andrews, S. Buzzi, W. Choi, S. V. Hanly, A. Lozano, A. C. K. Soong, and J. C. Zhang, "What will 5G be?," *IEEE Journal on Selected Areas in Communications*, vol. 32, no. 6, pp. 1065–1082, June 2014.

[20] M. Mozaffari, W. Saad, M. Bennis, and M. Debbah, "Unmanned aerial vehicle with underlaid device-to-device communications: Performance and tradeoffs," *IEEE Transactions on Wireless Communications*, vol. 15, no. 6, pp. 3949–3963, June 2016.

[21] G. C. Burdea and P. Coiffet, *Virtual Reality Technology*, John Wiley & Sons, 2003.

[22] M. Lukoševicius, *A Practical Guide to Applying Echo State Networks*, Springer Berlin Heidelberg, 2012.

[23] J. F. Yang, S. C. Chang, and C. Y. Chen, "Computation reduction for motion search in low rate video coders," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 12, no. 10, pp. 948–951, Oct. 2002.

[24] "HTC Vive install support," https://support.steampowered.com/kb_article.php?ref=2001-UXCM-4439.

[25] L. Wei, J. Cai, C. H. Foh, and B. He, "QoS-aware resource allocation for video transcoding in clouds," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 27, no. 1, pp. 49–61, 2017.

[26] D. C. Montgomery, E. A. Peck, and G. G. Vining, *Introduction to Linear Regression Analysis*, John Wiley & Sons, 2015.

[27] A. M Mood, "Introduction to the theory of statistics.," 1950.

[28] Z. Han, D. Niyato, W. Saad, T. Basar, and A. HjÃ¸rungnes, *Game Theory in Wireless and Communication Networks: Theory, Models, and Applications*, Cambridge University Press, 2012.

[29] G. Bacci, S. Lasaulce, W. Saad, and L. Sanguinetti, "Game theory for networks: A tutorial on game-theoretic tools for emerging signal processing applications," *IEEE Signal Processing Magazine*, vol. 33, no. 1, pp. 94–119, Jan. 2016.

[30] M. Chen, U. Challita, W. Saad, C. Yin, and M. Debbah, "Machine learning for wireless networks with artificial intelligence: A tutorial on neural networks," *available online: arxiv.org/abs/1710.02913*, Oct. 2017.

[31] M. Chen, W. Saad, and C. Yin, "Echo state networks for self-organizing resource allocation in LTE-U with uplink-downlink decoupling," *IEEE Transactions on Wireless Communications*, vol. 16, no. 1, January 2017.

[32] R. Sutton and A. Barto, *Reinforcement Learning: An Introduction*, MIT Press, 1998.

[33] I. Szita, V. Gyenes, and A. Lőrincz, "Reinforcement learning with echo state networks," *Lecture Notes in Computer Science*, vol. 4131, pp. 830–839, 2006.

[34] Oculus, "Mobile VR media overview," https://developer3.oculus.com/documentation/mobilesdk/latest/concepts/mobile-media-overview/.

[35] M. Bennis, S. M. Perlaza, P. Blasco, Z. Han, and H.V. Poor, "Self-organization in small cell networks: A reinforcement learning approach," *IEEE Transactions on Wireless Communications*, vol. 12, no. 7, pp. 3202–3212, June 2013.

[36] F. Kelly, "Charging and rate control for elastic traffic," *European Transactions on Telecommunications*, vol. 8, no. 1, pp. 33–37, 1997.

[37] M. Bennis and D. Niyato, "A Q-learning based approach to interference avoidance in self-organized femtocell networks," in *Proc. of IEEE Global Commun. Conference (GLOBECOM) Workshop on Femtocell Networks*, Miami, FL, USA, Dec. 2010.

[38] M. Abrash, "What VR could, should, and almost certainly will be within two years," https://media.steampowered.com/apps/abrashblog/Abrash%20Dev%20Days%202014.pdf.