IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 1 February 2024, accepted 26 February 2024, date of publication 5 March 2024, date of current version 19 March 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3374230

METHODS

Link-Level Traffic Modeling of Medical Extended Reality (MXR) Applications

YONGKANG LIU, (Member, IEEE), AND MOHAMAD OMAR AL KALAA, (Senior Member, IEEE)
Office of Science and Engineering Laboratories (OSEL), Center for Devices and Radiological Health (CDRH), U.S. Food and Drug Administration, Silver Spring, MD 20993, USA

Corresponding author: Yongkang Liu (yongkang.liu@fda.hhs.gov)

ABSTRACT Medical extended reality (MXR) applications are rapidly evolving with innovative use cases in medical training and surgical planning, among others. MXR benefits from seamless connectivity, like that promised by 5G, to facilitate its functionality where users can interact with virtual content, share views, and collaborate. In this paper, we identify practical implementation approaches for MXR connectivity and characterize the traffic models of selected example MXR applications. The traffic models were obtained through a data-oriented workflow to showcase general procedures and considerations in modeling traffic patterns for diverse MXR applications. The proposed models correlate the network traffic with specific MXR deployments (e.g., remote rendering, display mirroring) and events like head rotation, hand movements, and image download. Accordingly, they can be used to formulate the traffic models for other MXR applications to support an application-specific approach to the evaluation of relevant MXR connectivity risks.

INDEX TERMS 5G, medical extended reality, biomedical communications.

I. INTRODUCTION

Medical extended reality (MXR) encompasses augmented reality (AR) and virtual reality (VR) applications in healthcare use cases. The U.S. Food and Drug Administration (FDA) continues to receive MXR-related device submissions for diverse medical uses including in the orthopedic, radiology, physical medicine, ophthalmic, and cardiovascular areas [1]. MXR applications commonly use commercial-off-the-shelf head mounted display (HMD) to overlay virtual content over real images in the case of AR or immerse the user in a virtual environment populated by virtual content in the case of VR [2]. Network connectivity is used to facilitate the communication between MXR system components (e.g., application server, HMD, and/or the other non-HMD clients) and enable the envisioned integration of MXR in use cases like telemedicine and telesurgeries. Accordingly, remote healthcare providers can assist in virtual diagnosis and treatment sessions, coordinate with other specialists, and allow trainees to participate in realistic learning. MXR is used in multiple medical domains to facilitate access to medical information for diagnosis, treatment, and training purposes. For example, the FDA hosts a public list of medical devices that incorporate AR and VR that the agency has reviewed and authorized for marketing [1]. Multi-user coordination and access to remote virtual content are MXR features that rely on network connectivity to support the device-specific communication requirements and enable the intended device function [3].

The need to support diverse use cases requiring different connectivity quality of service (QoS) profiles highlights the importance of communication infrastructure that can be easily adapted to accommodate those profiles. 5G has been identified as one of the promising connectivity solutions for enabling MXR applications. For example, field trials for 5G-enabled medical visualization systems have been conducted by healthcare providers in partnership with 5G service operators [4], [5]. Furthermore, in a recent 5G landscape analysis report [6], the Medical Device Innovation Consortium (MDIC) identified 5G-Enabled Simulation with XR, including telementoring and immersive training, as a 5G-enabled healthcare use case alongside 5G-enabled

The associate editor coordinating the review of this manuscript and approving it for publication was Giambattista Gruosso.

robotics, mobile units, and remote care. The report also discusses knowledge gaps and practical considerations for translating the promises of 5G connectivity into realistic implementations. Those include the need for novel approaches for ongoing QoS monitoring, unestablished and varying communication key performance indicators (KPIs) for 5G-enabled use cases, and the lack of evaluation methods for medical device functions enabled by 5G. As a cellular network serving large geographic areas, 5G enables high-throughput, low-latency connections and facilitates mobile edge computing (MEC), which can contribute to deploying MXR applications in hospitals, training facilities, and remote locations. 5G improvements compared to previous network generations are enabled by a plethora of technical advancements across the radio access and core networks including millimeter wave communication, novel waveforms, and network slicing. Although 5G has been discussed at length as an MXR enabler, existing HMDs are not yet 5G-native and a 5G gateway device (e.g., smartphone, customer on-premises equipment) is needed to facilitate 5G connectivity. Furthermore, XR development environments are equipped to incorporate communication using technologies like local area networks (LAN) and Wi-Fi. A clear distinction between 5G and LAN technologies is the limited user control over 5G network configurations and operating conditions leading to a third party having the primary influence over the QoS that an MXR application receives.

Although discussions of the MXR connectivity aspects have been largely in the 5G context, there is a lack of representative traffic models that address MXR use-cases and deployment features, which constitute foundational knowledge of MXR system behavior, regardless of the enabling connectivity modality. MXR traffic models can help with the design and evaluation of MXR devices. For example, device developers can use them to inform their understanding of the QoS needs for a device, which influences the selection of a wireless technology that can support those needs as recommended in the FDA guidance document *Radio Frequency Wireless Technology in Medical Devices* [7]. Network administrators can also benefit from these models in the planning for network resource allocation and service provisioning. In this paper, we take a step toward bridging the gap in traffic modeling by proposing flexible MXR traffic models that can inform the development, evaluation, and realistic deployment of MXR devices. To do so, we develop MXR application examples that capture the diverse deployment options of MXR in realistic scenarios where the HMD device as the anchor node interacts with different companion devices. The traffic models were obtained through a data-oriented workflow to showcase general procedures and considerations in modeling traffic patterns for diverse MXR applications. Example equipment, deployments, and MXR applications were used for model development. The proposed models attempt to capture small-scale components of MXR applications like hand movements, head rotation, image download, etc. to facilitate the simulation of other MXR applications and support an application-specific approach to the evaluation of relevant connectivity risks. Accordingly, the contributions of this paper are three-fold. First, we investigate the link traffic patterns in connected MXR applications by developing an end-to-end workflow; second, we collect and characterize the traffic patterns in representative use cases using data-oriented methods; third, we reproduce the traffic loads by implementing the obtained traffic models and validating the simulated traffic.

Table 1 lists the abbreviations used in this paper. The remainder of the paper is organized as follows. Section II summarizes works related to connectivity in MXR and relevant traffic modeling literature. The system model used for identifying different MXR use scenarios is introduced in Section III. The overview of the adopted modeling approach is explained in Section IV, which is followed by Section V where the data collection activities are detailed. Section VI reports the results and validation work. Section VII concludes the paper.

II. RELATED WORK

The subject of this manuscript is the development of MXR application traffic models. A review of MXR use cases is included in [8]. Our previous work in [9] reported the integration of selected MXR services in a 5G testbed to document practical implementation challenges and observe realistic MXR connectivity characteristics. The experimental MXR application developed and described in [9] was used to generate MXR network traffic in the work detailed in this paper.

MXR connectivity is commonly discussed in the context of 5G networks [6]. The 3rd generation partnership project (3GPP) has published a technical report (TR) on XR over 5G [10], which addresses typical XR applications and their QoS benchmarks. In another 3GPP TR, use cases are categorized and 5G system requirements are identified to support XR applications [11]. However, investigations of traffic models in realistic MXR application and the corresponding connectivity requirements are not yet published in the literature. Although XR applications, of which MXR is a subset, share many commonalities, MXR introduce technical evaluation challenges for spatial image quality, temporal image quality, and usability [3]. Additionally, low-latency transmissions with high-reliability connection support have been identified as elements of interest to enable connected MXR applications [6]. Notably, many MXR applications use the frameworks offered by generic XR development platforms, e.g., Unreal Engine and Unity, which were designed and optimized for multi-player, interactive network applications like online gaming [12]. As a result, existing reports of XR traffic modeling are based on XR games and easily accessible hardware like smartphones. Accordingly, the authors in [13] proposed a traffic model for VR applications based on traffic traces of playing a VR game on the SteamVR platform. Data was solely collected from the application waiting room, which was

TABLE 1. List of abbreviations.

Abbrev.	Meaning
3GPP	3rd generation partnership project
5G	Fifth-generation (mobile communication)
ACK	Acknowledgement (message)
API	Application programming interface
AR	Augmented reality
ARP	Address Resolution Protocol
CDF	Cumulative Distribution Function
CDRH	Center for Devices and Radiological Health
CSV	Comma-separated values
CT	Computed tomography
DHCP	Dynamic Host Configuration Protocol
DT	Downstream Traffic
ELTA	Exploratory Link Traffic Analaysis
FDA	Food and Drug Administration
FFH	Full Function Headsets
FOV	Field of View
fps	frames per second
HDI	Human Device Interaction
HL2	Hololens2 (a HMD device example)
HMD	Head-mounted display
HMS	Headset as a Mirroring Server
HTTPS	Hypertext Transfer Protocol Secure
IBT	Inter-burst time
IFT	Inter-frame time
i.i.d.	Independent and identically distributed
I/O	Input and/or output
IP	Internet Protocol
ITBT	Inter-traffic-block time
Kbps	Kilobits per second
KPI	Key performance indicator
LAN	Local area network
Mbps	Megabits per second
MDIC	Medical Device Innovation Consortium
MEC	Mobile Edge Computing
MXR	Medical extended reality
OC	Observing client
OSEL	Office of Science and Engineering Laboratories
OSI	Open System Interface
PCAP	Packet Capture
PDU	Protocol Data Unit
PMF	Probability Mass Function
RFH	Reduced Function Headset
SBC	Super-Burst Cycle
TAP	Test access point
TB	Traffic Block
TBS	Traffic Block Size
TCP	Transmission control protocol
TR	Technical Report
UDP	User datagram protocol
UE	User equipment
UI	User Interface
UR	Unreal Engine
UT	Upstream Traffic
VR	Virtual reality
XR	Extended reality

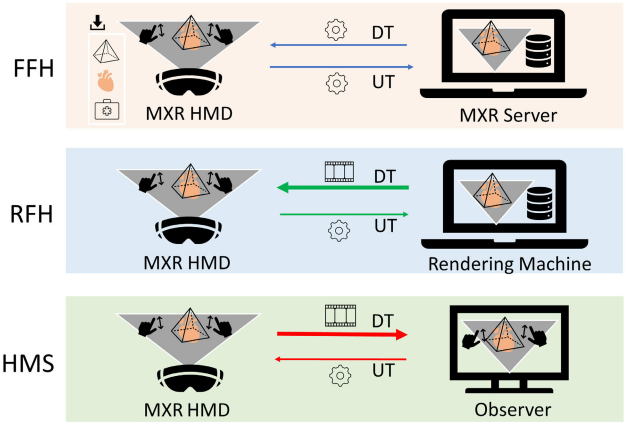

FIGURE 1. Example MXR application scenarios and associated traffic flows. Full Function Headsets, Reduced Function Headsets, and Headset as a Mirroring Server are abbreviated by FFH, RFH, and HMS, respectively. Thick arrows indicate the traffic direction with high usage for each scenario while thin arrows indicate the opposite. The video icon refers to primarily video content flow and the gear icon refers to all other types of traffic including control and data exchange.

rendered and streamed from a PC rendering server to a smartphone acting as a passive VR headset. Frame periods were modeled as independent and identically distributed (i.i.d.) Logistic distributed random variables. Video frame sizes were modeled using a Gaussian Mixture Model. This work was further expanded in [14] with additional traffic traces incorporating user interactions. Gaming XR applications often involve fast-switching backgrounds and vigorous actions, which differ from the common operations in MXR use cases. For example, in pre-op planning sessions, the patient's AR content is cast onto a body phantom for surgeons to discuss and practice the possible approaches to surgical procedures [15]. The user may focus on the operation area in the filed of view (FOV) for a while without many actions. The corresponding network traffic would only communicate limited changes in user motion and background variations.

Traffic models, like those reported in this manuscript or the ones reported in [13] and [14] can be used with the traffic generation modules in network simulators (e.g., ns-3 [16]) to study network capacity and as a step to characterize the system reaction to connectivity failure modes (e.g., congestion, delays on certain interfaces, channel access conflicts). Additionally, the modeling results can enrich the emulation capability of network evaluation testbeds [17], [18] by facilitating the implementation of simple test objects focused on the communication aspects of realistic devices. The following sections will introduce the system model for connected MXR applications and the steps taken to develop traffic models corresponding to realistic MXR scenarios.

III. SYSTEM MODEL

We consider a system model that identifies use cases and corresponding network services supporting MXR applications. As illustrated in Figure 1, the system model addresses the

following use cases based on the role of MXR headset serving as the primary user interface (UI): Full Function Headsets (FFH), Reduced Function Headsets (RFH), and Headset as a Mirroring Server (HMS). The remainder of this section is dedicated to discussing these three use cases and how the user input/output (I/O) traffic data flows were captured and characterized.

A. FULL FUNCTION HEADSETS (FFH)

This case involves MXR HMD nodes and remote application servers. MXR FFH are equipped with full virtual content rendering capabilities, i.e., a headset can independently render and present virtual content on its local display. The information used for rendering MXR content is prepared and locally cached while the HMD communicates with the remote application server to exchange status updates and other system wide information, e.g., environment changes or administrator's messages, as well as changes in peer nodes if multiple clients are active [19]. The deployment of FFH in an MXR application requires the least network bandwidth compared to other use cases as most content, e.g., patient's medical profile, are downloaded directly to the HMD, which reduces the burden on the connectivity infrastructure when multiple clients collaborate in the same virtual environment. An FFH node may also request additional MXR information on the fly, e.g., updated patient X-ray images, from external medical data sources [9]. However, such on-demand data transmissions could be fulfilled with intermittent, short sessions that do not stress the network for long periods.

B. REDUCED FUNCTION HEADSETS (RFH)

Compared to FFH, RFH stands for the type of user nodes enabled with reduced XR features, e.g., due to HMD resource limitations and power constraints. In this case, also referred to as remote rendering in the literature [20], the HMD primarily serves as an I/O device at the user-end. In the inbound traffic, the MXR content is rendered remotely by an external machine with relevant compute and communication resources, e.g., mobile edge server, and streamed to the headset to be displayed in the user's FOV. In the outbound traffic, the HMD updates the rendering engine with local status changes captured by its embedded sensors and cameras. Deploying RFH simplifies the user-end equipment design requirements, e.g., hardware, software, battery, and form factor, at the cost of increased network traffic loads and more stringent performance requirements for transmitting live, remotely rendered virtual content. The rendering engine can be deployed in the vicinity of the user or remotely at further geographic locations, which places higher bandwidth requirements on the link to the end user compared to FFH applications. In the 5G network, the rendering machine itself can be deployed either as the UE or on the server side. In the former, the communications between the rendering node and the MXR server are similar to the FFH case, which offloads the display and sensing capability to another field device; in the latter, the traffic is routed through the wired network segment of the 5G network as a part of MEC or cloud services [21].

C. HEADSET AS A MIRRORING SERVER (HMS)

The FFH and RFH use cases address communication scenarios in MXR applications involving the end-to-end connection between an MXR user node and the enabling remote services. Mirroring the headset's view to third-party observing clients (OC) is another use case that offers complementing features to interactive MXR functions. In this case, the headset serves as a mirroring server (HMS) to share the user's view with one or multiple external observers through a live video stream. Many XR devices and development platforms provide application program interfaces (API) in support of this feature [22], [23]. HMS is useful in healthcare training programs for relaying the mentor's instructions or guided therapy sessions for monitoring the session progress [15].

As the HMD device is usually deployed as a field node, e.g., a user equipment (UE) node in the 5G network, the above FFH, RFH, and HMS use cases highlight the diverse data flows that need to be supported by the connectivity infrastructure in both the uplink and downlink. The traffic patterns of individual services identified in Fig. 1 can also serve as elements in formulating the aggregated patterns of service combinations in hybrid use cases. For example, realistic MXR applications can adopt a hybrid scenario where all three types of traffic streams could be implemented simultaneously to serve specific application purposes [2].

IV. MODELING METHOD OVERVIEW

MXR applications are generally IP-based, where the application traffic can be generated in a variety of forms and rates, which can be constant or fluctuating with time. Modeling the traffic flows is intended to depict deterministic generation patterns or stochastic distributions of traffic blocks (TB) in communication links carrying MXR application messages. A traffic model considers two primary variables: the traffic block size (TBS) and the inter-traffic-block time (ITBT). The former indicates the size of information generated at each message instance; the latter measures the length of the interval between any two consecutive blocks. The traffic modeling effort reported herein is focused at the Application Layer per the open system interconnection (OSI) 7-layer protocol stack so that the resulting traffic models represent the application's unique traffic needs and patterns for communications regardless of the enabling connectivity modality.

The data-oriented traffic modeling process shown in Fig. 2 was used to study the MXR use cases specified in Section III. The process can be decomposed into five steps: traffic measurement, data pre-processing, data exploratory analysis, traffic modeling, and validation. Specifically, MXR traffic in the link of each application use case was measured and collected. Raw data in terms of packet captures (PCAP) was then processed by a network protocol parsing tool

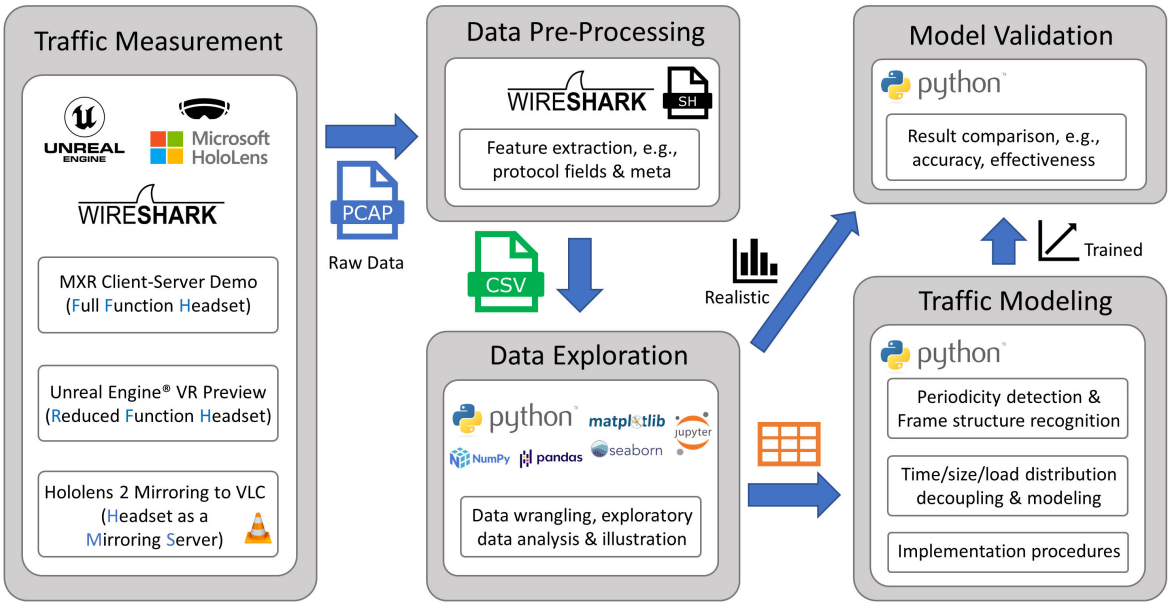

FIGURE 2. Data-oriented workflow for modeling MXR traffic.

to extract packet header field information and meta data. Next, exploratory data analysis was performed to obtain preliminary traffic pattern features, e.g., statistical or time-series. The patterns then informed the development of traffic model(s) that depict how the application data was generated using variables for TBS and ITBT. Model verification was then performed by comparing with measured traffic data for the considered scenario.

The modeling process allows the initial characterization of basic traffic flow features and the subsequent detection of repeated, statistically significant occurrences of unique traffic instances, their frequency, and triggering conditions. Notably, the traffic capture on a link can be composed of multiple parallel data flows for different purposes. These can be decomposed into unique flows, and each characterized with a standalone model. However, the decomposition of flows can be facilitated with the prior knowledge of communicating services and their anticipated behavior. The following sections expand on the techniques and outcomes of individual steps in the traffic modeling process.

Notably, it is difficult to generalize the traffic observations of an investigated connected medical device and consider those observations applicable to an entire class of devices. For example, claiming that the observed traffic for the studied example MXR application in this paper is representative of all MXR applications on the market. The intended use of a certain device, its design, use of data, and how it incorporates the connectivity infrastructure are factors that influence the device connectivity requirements and the traffic patterns it generates [24].

V. TRAFFIC DATA COLLECTION AND WRANGLING

The traffic data collection was performed at the FDA 5G lab in Silver Spring, MD, USA. The three MXR use cases

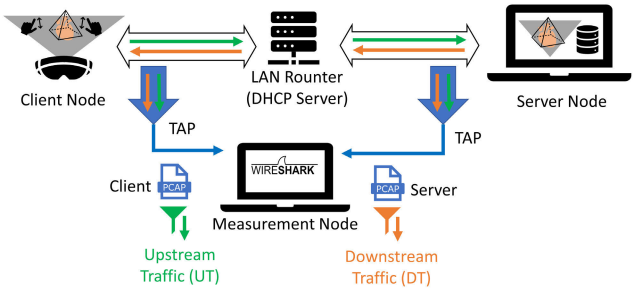

FIGURE 3. The MXR link traffic measurement setup illustrated for the FFH use case.

specified in Section III were configured in representative experimental setups. Fig. 3 illustrates the measurement setup used for the FFH links as an example; the other setups were similar but followed their respective specification for downstream and upstream traffic links. The application traffic generation and associated data sets for each use case are summarized in Table 2 in addition to a summary of the tools used for data collection and analysis.

A Microsoft Hololens 2 (HL2) was used as an example HMD. The HL2 is an XR HMD that is supported by multiple XR development platforms, e.g., Unreal Engine (UR) and Unity. Traffic data was collected from MXR application examples developed on Unreal Engine 4.27.2 [9]. FFH data set was collected using an application example of a multi-user virtual clinical room with CT image display capability; RFH data set was collected using another application example running the VR Preview mode of the UR editor. HMS data set was collected by streaming the headset display to external observing nodes while the HL2 was running the FFH application. Traffic was captured in a LAN setup consisting of Ethernet connections between the headset and the remote

TABLE 2. Summary of MXR data sets.

		FFH	RFH	HMS
Setup	Hardware	Server: Dell Latitude 7760 HMD: Microsoft Hololens2	Renderer: Dell Latitude 7760 HMD: Microsoft Hololens2	HMD: Microsoft Hololens2 Client: Dell Latitude 5500
	Software	Unreal Engine 4.27.2	Unreal Engine 4.27.2	Server: Device Portal Client: VLC player
	Measurement, Analysis, and Modeling	SharkTAP CC (× 2) + Dell Latitude 7750 Wireshark 2.6.10 Python 3.8.5		
Traffic	Technical Enabler	Client-server MXR, Unreal Engine	"VR Preview" by Unreal Engine Editor	Device Portal (Web UI)
	In-Service Actions	User entries/exits, MXR display requests, and angular size changes in the FOV.	Session ON/OFF, MXR content loading and size/position control, and angular size changes in FOV.	Mirroring ON/OFF, user FOV dynamics per FFH.
	Downstream	Game states and actor updates, bi-way lightweight	Rendered MXR video frames	Mirrored video frames (fragmented MP4)
	Upstream		User input and environmental updates	N/A
	Features	Max. local processing; Min. network consumption; On-demand medical profile caching	Min. user device dependencies; Max. network consumption	One-way video streaming
Data Set	DT@Server UT@Client	∼420 sec traffic trace with 7 application states, ∼60 sec per state.	∼680 sec traffic trace with 8 application states, ≥30 sec per state.	∼1090 sec traffic trace

node (i.e., server, render, observer) through an Ethernet router that was running a Dynamic Host Configuration Protocol (DHCP) service for assigning the headset's IP address. The inbound and outbound traffic flows at the headset and remote nodes in each measurement were captured by test access point (TAP) devices that copy all through traffic streams to a separate measurement computer running Wireshark. The user and server side traffic copies were saved as PCAP files for post analysis and modeling. Accordingly, the traffic modeling addressed the MXR application—not the underlying connectivity—that represented common MXR features (e.g., downloading images from a server and displaying them to the user in a virtual environment). This facilitates the study of MXR systems communication behavior in contrast to stressing the network to identify connectivity-specific capacity and failures. Therefore, the resulting models can help with simulating MXR applications over a variety of connectivity enablers like LAN, Wi-Fi, or 5G.

The data wrangling starts with the raw PCAP files. To identify and depict traffic patterns, especially inter-block time distributions, of interest was packet copies that were captured at the source node and stamped with the time when they were first observed on the link. For example, in the FFH use case, the dowlink traffic analysis considers the packets captured at the MXR server while the uplink traffic is modeled based on the packets observed at the HMD. In each data set, traffic flows were cleaned by filtering out irrelevant protocol packet captures, e.g., Address Resolution Protocol (ARP) and other network discovery messages, through Wireshark protocol filters. Afterwards, a Bash script was executed to call the tshark program [25] for extracting the header field information from each packet and save the values of selected fields into a comma-separated values (CSV) file wherein each column stores values of a recorded header field and each row stores all field values associated with a single packet. The tabular CSV data was then imported into a Python program for pattern analysis and modeling.

VI. EXPLORATORY LINK TRAFFIC ANALYSIS (ELTA) AND TRAFFIC PATTERN MODELING

The exploratory link traffic analysis (ELTA) aims to explore traffic patterns and their features in the collected data sets. Traffic features investigated in ELTA include time-series characteristics of TBS/IBT variables, their statistics, and their joint distribution. This process employs visualization tools and methods to illustrate samples of the studied metric. ELTA outcomes provide preliminary insight into the studied data, e.g., the basic shape of variable spaces, that suggests new candidate features to explore for characterizing traffic patterns. In addition, metrics obtained by ELTA provide the basic description of the observed traffic pattern that can help with studying model accuracy, i.e., if simulated traffic by a model achieves similar behavior to realistic traffic, e.g., values in the expected ranges, ratio of discrete values, sample densities per clustered values.

A summary of the process is as follows. First, the traffic data space (e.g., time range, frame size set, rate amplitudes, and time-size correlations) is illustrated. Afterwards, the presence of traffic blocks is verified, i.e., aggregated consecutive message frames indicated by common packet features like flag values and indicator bits. Next, focus is concentrated on the majority of traffic data and non-significant contributors to the traffic load are removed by noting frames occurrence frequency. Multiple passes through these steps are performed to refine the model. Exploratory methods leveraging

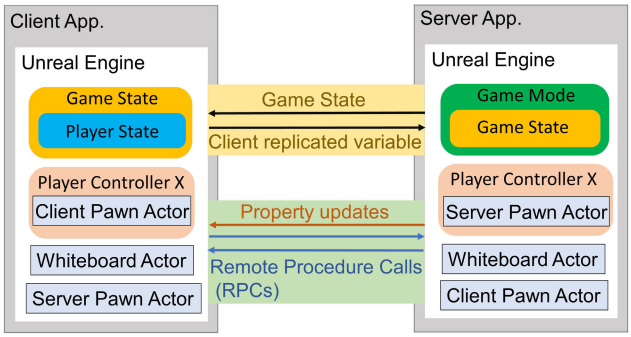

FIGURE 4. Client-Server communications in connected XR applications based on Unreal Engine.

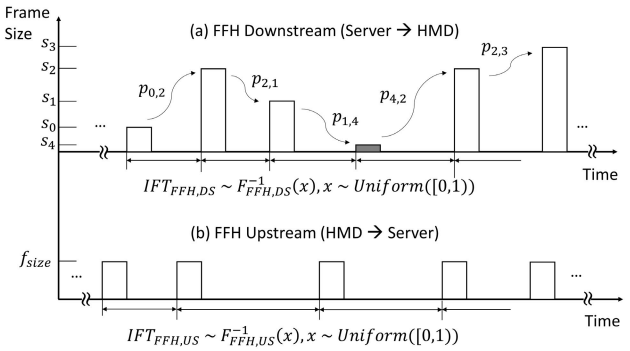

FIGURE 5. Illustration of FFH traffic patterns. Five frame sizes are considered in DT, which are s_0 of 10 bytes, s_1 of 21 bytes, s_2 of 37 bytes, s_3 of 48 bytes, and s_4 of 0 byte for the "silent" frames.

engineering knowledge are leveraged including payload exploration, frequency check of prefix values, throughput weight calculation, and outlier removal. The result is a set of observations as summarized in Table 3.

Because the considered use cases exhibited unique traffic patters, the following subsections will be dedicated to detailing the results for each use case. The validation results of individual link traffic models will be discussed along with implementation examples. The validation objective is to characterize the capability of the developed models to generate link traffic loads at the same or similar pace as the realistic applications. In the following discussion, "frame" denotes the minimum traffic unit for application layer messages (e.g., the discrete MXR service data and signaling) while "packet" is used in a more general way to name any formatted data below the application layer, e.g., TCP/UDP/IP/Ethernet packets. If multiple frames are identified to be generated and transmitted sequentially in an organized manner, the term "burst" is used to denote the frame groups.

A. FFH TRAFFIC PATTERN ANALYSIS

The traffic packets between the FFH HMD and the MXR server are identified as UDP datagrams, which are light-weight, connection-less IP-based traffic packets. Fig. 4 shows the routine and on-demanded message exchanges between MXR client and server nodes in the Unreal Engine application. Readers can refer to the technical documentation of Unreal Engine that introduces common communication messages between client-server pairs [26].

Encrypted UDP packets carry the application data. Furthermore, there is no publicly-available technical reference explaining the structure of Unreal Engine messages or details about its driver design for HL2. Therefore, no attempts were made to recover the upper-level message content during data analysis. With the objective of recognizing traffic patterns and developing models to characterize the network traffic load, the pattern analysis and modeling were performed solely based on the protocol data unit (PDU) payload information contained in UDP datagrams. Downstream and upstream traffics were analyzed separately, which uncovered the traffic patterns described in Table 3 and illustrated in Fig. 5. Notably, message frames contained in individual UDP packet payloads were considered traffic blocks in both the downstream and upstream links. Accordingly, the FFH downstream traffic (DT) model is based on the distributions of frame size and inter-frame time (IFT), which are statistically decoupled from each other. On the contrary, the FFH upstream traffic (UT) model is based on its own IFT distribution with a single, fixed frame size.

1) FFH DOWNSTREAM TRAFFIC (DT) ANALYSIS

Examining the downstream traffic pattern in the FFH data set, i.e., data transmitted from the MXR server to the HMD, revealed that most observed frames — carried by UDP packets — were less than 100 bytes. The top four frequently observed data frame sizes, i.e., 10, 21, 37, and 48 bytes, represented the majority of data and accounted for 99.81% of the total packets and 99.51% of the traffic load in a measurement period of around 420 seconds. The results of ELTA for this data set are illustrated in Fig. 6 including frames time variation and statistics. IFT outliers, i.e., the instances that were either too short (< 5 msec) or too long (> 30 msec), were removed in the illustrations, noting that they occupied less than 0.066% in the total data. DT traffic between the client and server presented a frequent but lightweight communication pattern. The throughput in the link was observed at < 25 Kbps.

The joint distribution of the frame size and IFT shown in Fig. 6b illustrates the clustered traffic pattern. However, it does not indicate whether the frame size can be modeled independently from the IFT. To explain the discrete frame sizes and clusters in the IFT dimension, frames prefix were inspected, i.e., the few bytes at the beginning of the UDP payload, to document that DT data were formatted in segments where larger-sized frames carried more segments than smaller frames. A simple model can help explain the segmented data in DT messages. Suppose that there are two variables, (A, B), denoting for example updates to the environment and objects in the XR application, respectively. Binary states, i.e., ON (1) and OFF (0), can be used to denote

TABLE 3. Summary of traffic patterns and modeling on the MXR data sets.

		Downstream Traffic (Server to Client)	Upstream Traffic (Client to Server)
FFH Server: MXR Server Client: HMD	Pattern	UDP messages with encrypted PDU; Message frames as traffic blocks; Frame size decoupled from inter-frame time (IFT); "Silent" frames assumed; Application-state-specific variation: negligible.	UDP messages with encrypted PDU; Message frames as traffic blocks; Fixed frame size with one unified size value; Application-state-specific variation: negligible.
	Modeling	Markov chain on frame size state transitions; Non-normal, empirical distribution on IFT.	Single, fixed frame size; Non-normal, empirical distribution on IFT.
RFH Server: MXR rendering machine Client: HMD	Pattern	UDP message with encrypted PDU; Frame combinations (head/core/tail) as bursts; Bursts as traffic blocks; TBS decoupled from IBT.	UDP message with encrypted PDU; Frame combinations (head/tail) as bursts; Bursts as traffic blocks; TBS decoupled from IBT.
	Modeling	Empirical, continuous distributions for application-state-specific TBS; Non-normal, empirical distribution on IBT per video frame rate.	Empirical, discrete distributions for application-state-specific TBS; Non-normal, empirical distribution on IBT per video frame rate.
HMS Server: HMD Client: Mirroring player	Pattern	Fragmented MP4 video frames in multiple TCP messages with encrypted PDU; Frame combinations (head/core/tail) as bursts; Bursts as traffic blocks; TBS coupled with IBT in super-burst cycles (SBC) under rate control; application-state-specific variation: negligible.	TCP ACK messages only
	Modeling	Empirical distributions for aggregated block size/delay in a SBC; Simulate TBS/IBT in batches per SBC.	N/A.

the presence of updates from each variable or environment component in a frame when it is generated. Therefore, (0, 0) means neither variable has a change to report, (1, 0) means A has an update, (0, 1) means B has an update, and (1, 1) means both have updates. Consequently, an update from a single variable introduces a marginal load, e.g., A with 11 bytes and B with 27 bytes. Therefore, a basic data load of 10 bytes can be associated with (0, 0), (1, 0) represents the basic load plus A's update resulting in a total of 21 bytes, (0, 1) is for the basic load plus B's update resulting in a total of 37 bytes, and (1, 1) generates the largest load of $10 + 11 + 27 = 48$ bytes. Accordingly, multiple application status variables were present that are routinely updated through the communication link. However, it was not necessary to transmit the values of all variables if they have no updates to report.

Random variables like IFT follow a continuous distribution unless the generation time is arbitrarily manipulated, e.g., being truncated by separate segments. Without additional prior knowledge of the traffic generation scheme used by the application, an assumption was made that the application may skip some update cycles as maintained by local timers when no new information was generated to update the receiver. Accordingly, "silent" frames were introduced and populate into the frame intervals with long IFT values, e.g., selecting the interval samples $>$ 15 msec. In such cases, the traffic generator would omit the outstanding update slot and reload the timer waiting for the next update instance. This can explain the IFT clustered distribution shown in Fig. 6b as the data contained not only the interval instances between any two consecutive frames but also the accumulated delay involving silent frames in between. Multiple silent frames can be inserted into one long interval, however, only the case with up to two consecutive silent frames in the same interval was considered. The IFT distribution based on the reshaped frame samples is shown in Fig. 7.

A Markov chain model was used to depict the transition probabilities between consecutive frames in which the states were in the extended frame size set $S = \{s_0,\ s_1,\ s_2,\ s_3,\ s_4\}$ containing the four detected frame sizes s_0 to s_3 (representing 10, 21, 37, and 48 bytes, respectively) plus a zero size state s_4 for all silent frames. The model can be denoted as follows.

$$\bar{p} = \bar{p} \times A \tag{1}$$

$$\bar{p} = \left[p_0,\ p_1, \ldots,\ p_{|S|}\right] \tag{2}$$

$$A = [p_{i,j}]_{|S|\times|S|} = \begin{bmatrix} p_{0,0}, & p_{0,1}, & \ldots, & p_{0,|S|-1} \\ p_{1,0}, & p_{1,1}, & \ldots, & p_{1,|S|-1} \\ \ldots, & \ldots, & \ldots, & \ldots \\ p_{|S|-1,0}, & p_{|S|-1,1}, & \ldots, & p_{|S|-1,|S|-1} \end{bmatrix} \tag{3}$$

$$p_{i,j} = Pr\{s_{next} = s_j | s_{current} = s_i\}\ for\ s_i, s_j \in S \tag{4}$$

$$\Sigma_{j=0}^{|S|} p_{i,j} = 1, s_i \in S \tag{5}$$

$p_{i,j}$ is the transition probability between states i and j. A is the Markov Chain transition matrix. $\bar{p}$, the stationary distribution of states S, was calculated as [0.242, 0.037, 0.478, 0.05, 0.193] per the DT data set.

IFT values are found not normally distributed. Therefore, the CDF curve of this empirical distribution can be directly used in practice to randomly generate IFT instances (e.g., importing the CDF curve into a network simulator and calling the simulator built-in interpolation functions). Alternatively, it is also plausible to construct a look-up table for the

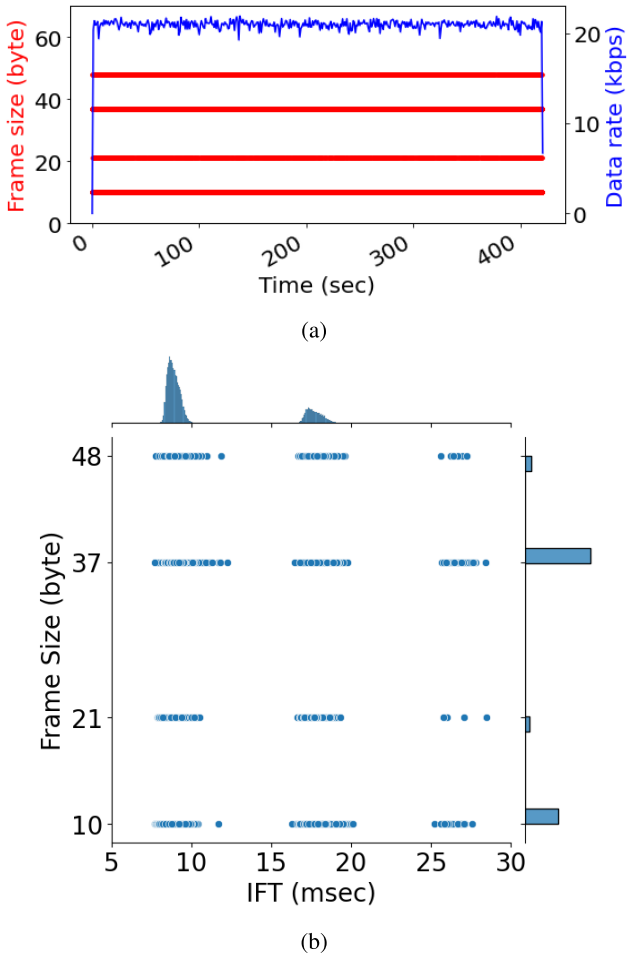

FIGURE 6. Example findings in ELTA on the FFH DT data: (a) frame size (in red) and rate (in blue) versus time and (b) joint frame size and IFT distribution.

quantile function (i.e., inverse CDF) of the IFT to improve the computation efficiency in the simulation. The state-wise distribution plotted in Fig. 7b demonstrates that IFT values do not exhibit significant changes associated with specific frame sizes. Therefore, the frame size and IFT represented in the data set are independently distributed, which permits the decoupled selection of their values.

In addition, seven application states (AS) were exercised during a measurement window of 420 sec to inspect traffic variations following user actions. Table 4 lists the user actions associated with individual application states, each of which was observed for around 60 sec. No significant DT/UT throughput variations were observed between different AS. It echos the earlier discussion on the client-server communication design adopted by the FFH test case where the link is designed and optimized with minimum network bandwidth consumption so that sporadic, on-demand requests may produce minor marginal increments in the aggregated traffic need. The traffic model can capture the average transmission load level and depict variations in the frame sequence.

TCP sessions for downloading medical images from an Apache server were also captured. The traffic was on-demand

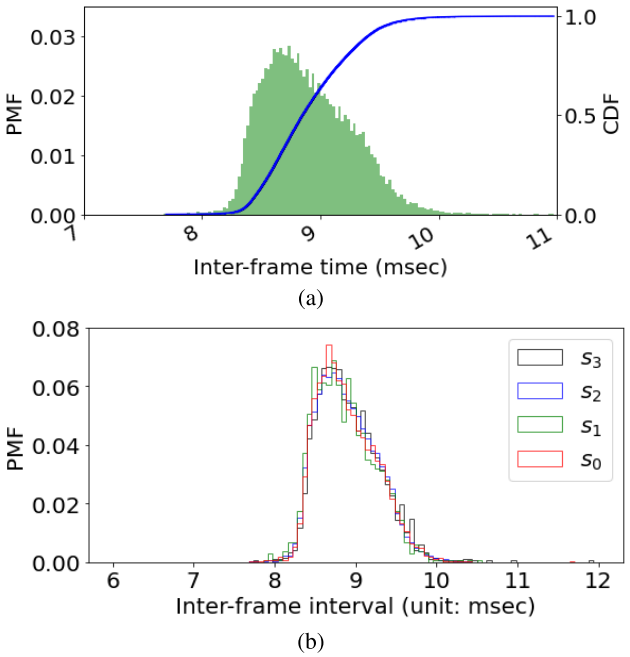

FIGURE 7. IFT distribution in FFH DT: (a) the entire data set and (b) state-wise data sets.

TABLE 4. Application states in the FFH traffic measurement test.

Application State	ID	User Action
Blank Image Plane on Display	1	HL2 tester looks towards the empty image plane in the view.
Static Image Display	2	The tester watches the 2D CT image displayed on the plane.
Image Switches on Display	3	HL2 and server take turns to cycle new images displayed on the plane every 3 -5 seconds.
Single-hand moves in the view	4	The tester shows one hand in the view.
Double-hand moves in the view	5	The tester shows both hands in the view.
Image being out of sight	6	The tester rotates by 90 degree to keep the image plane out of sight.
Image being in and out of sight	7	The tester switches the view by moving head up/down or side to side to alternate the image plane in and out the sight, each with a 2-3 sec stay.

once the client HMD requested the patient profile that was not available in its local cache. The request was fulfilled with a one-time Hypertext Transfer Protocol Secure (HTTPS) session, which can be modeled as sporadic bursts for the client-server handshakes or be emulated as a short HTTPS session by built-in HTTPS protocol modules in network simulators, e.g., NS-3. Assuming that patient-related medical data has been collected, verified, and made available before starting MXR sessions, the on-demand medical data communication is out of scope of the traffic modeling effort in this paper while noting that models for this type of traffic can be implemented as add-ons when needed.

2) FFH UPSTEAM TRAFFIC (UT) ANALYSIS

FFH UT was inspected through similar steps to the DT case to observe that UT packets were also UDP datagrams with a

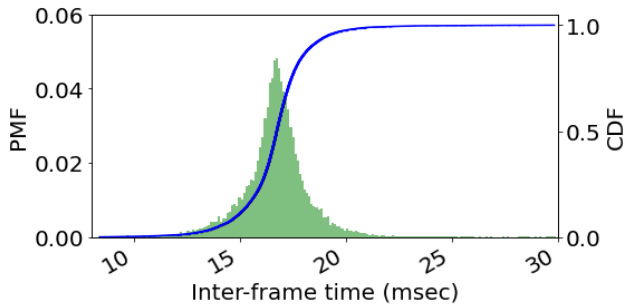

FIGURE 8. IFT distribution in FFH Upstream link.

single packet size of 39 bytes dominating the data set. The IFT distribution is shown in Fig. 8.

3) FFH TRAFFIC MODELS AND IMPLEMENTATION EXAMPLES

The DT model can be implemented using the steps proposed in Algorithm 1. In this implementation, the initial frame size is randomly selected following the stationary probability and the time and size of the following frame instances are estimated according to the Markov chain state transitions and the IFT CDF curve, respectively. Specifically, the parameters in the Markov chain for the frame size transitions include the state set S denoting the frame size, the stationary probability vector $\overline{p}$ in Eq. 2, and the state transition probability matrix A in Eq. 3; the IFT CDF curve is presented as $F_{IFT} = Pr\{IFT < t\}$ along with its inverse function F_{IFT}^{-1}, i.e., the quantile of a given probability number; the algorithm stops at a predetermined simulation time T_{stop}.

Algorithm 1 FFH Downstream Traffic Frame Generation

Input: $S, \overline{p}, A, F_{IFT}, T_{stop}$
Output: $[(f_{time}^i, f_{size}^i)]_{i=0}^n$

$f_{time}^0 \leftarrow 0$
$r \leftarrow X \sim U(0,1)$
if $\Sigma_0^{i-1} p_i \leq r < \Sigma_0^i p_i$ **then**
  $f_{size}^0 \leftarrow s_i$
$k \leftarrow 0$
while $f_{time}^k < T_{stop}$ **do**
  $r \leftarrow X \sim U(0,1)$
  $f_{time}^{k+1} \leftarrow f_{time}^k + F_{IFT}^{-1}(r)$
  $s \leftarrow S.index(f_{size}^k)$
  $r \leftarrow X \sim U(0,1)$
  if $\Sigma_0^{j-1} p_{s,j} \leq r < \Sigma_0^j p_{s,j}$ **then**
    $f_{size}^{k+1} \leftarrow s_j$
  $k \leftarrow k+1$

Implementing the FFH UT pattern discussed above is simpler than FFH DT as there is only one frame size variable to be simulated, which can be summarized by the steps shown in Algorithm 2.

FFH downstream traffic simulated by Algorithm 1 was compared to the actual traffic data set in Fig. 9. The error

Algorithm 2 FFH Upstream Traffic Frame Generation

Input: $F_{IFT}, T_{stop}, f_{size}$
Output: $[(f_{time}^i, f_{size})]_{i=0}^n$

$f_{time}^0 \leftarrow 0$
$k \leftarrow 0$
while $f_{time}^k < T_{stop}$ **do**
  $r \leftarrow X \sim U(0,1)$
  $f_{time}^{k+1} \leftarrow f_{time}^k + F_{IFT}^{-1}(r)$
  $k \leftarrow k+1$

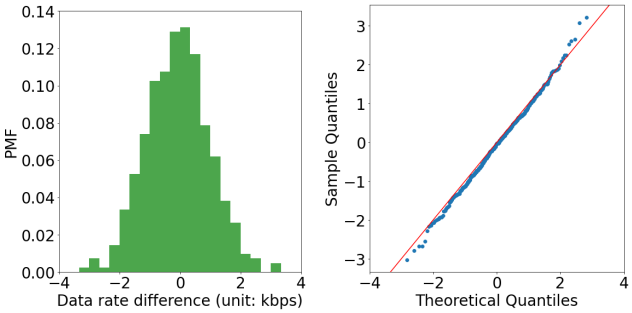

FIGURE 9. Verification of the FFH Downstream traffic model. Left—distribution of the traffic load difference, and Right—the p-p plot of the load difference distribution.

in observed data rates between the two sets has a mean of 0.07 Kbps and standard deviation of 1.04. The p-p plot, which is a visual measure for how closely observations in two data sets align, illustrates the close agreement between the simulated and measured sets. Similarly, comparing traffic simulated by Algorithm 2 with actual FFH UT measurements uncovers a mean data rate error of of 0.06 Kbps and 0.32 standard deviation.

B. RFH TRAFFIC PATTERN ANALYSIS

Similar to the FFH analysis, the outcome of inspecting RFH data set is illustrated in Fig. 10. RFH links carry different application messages compared to the FFH links resulting in unique traffic patterns as will be discussed hereafter.

1) RFH DT ANALYSIS

The remote rendering machine transmits data frames in RFH DT to the headset containing processed MXR content, which may vary with the user's input that encompasses HMD embedded sensors and cameras. To investigate traffic load variations according to application states, we implement the states shown in Table 5 as a representation of user actions used in MXR.

Fig. 11 illustrates select ELTA results in the RFH DT data set. The RFH downstream traffic is also composed of UDP messages, which can be observed with discrete size ranges: low (<150 bytes), medium (around 400 to 800 bytes), and large (800+ bytes) as shown on Fig. 11 demonstrating a scatter plot of the UDP payload frame size with time. Inspecting the prefix sequence of the payload data indicated that these frames belong to different message types, which

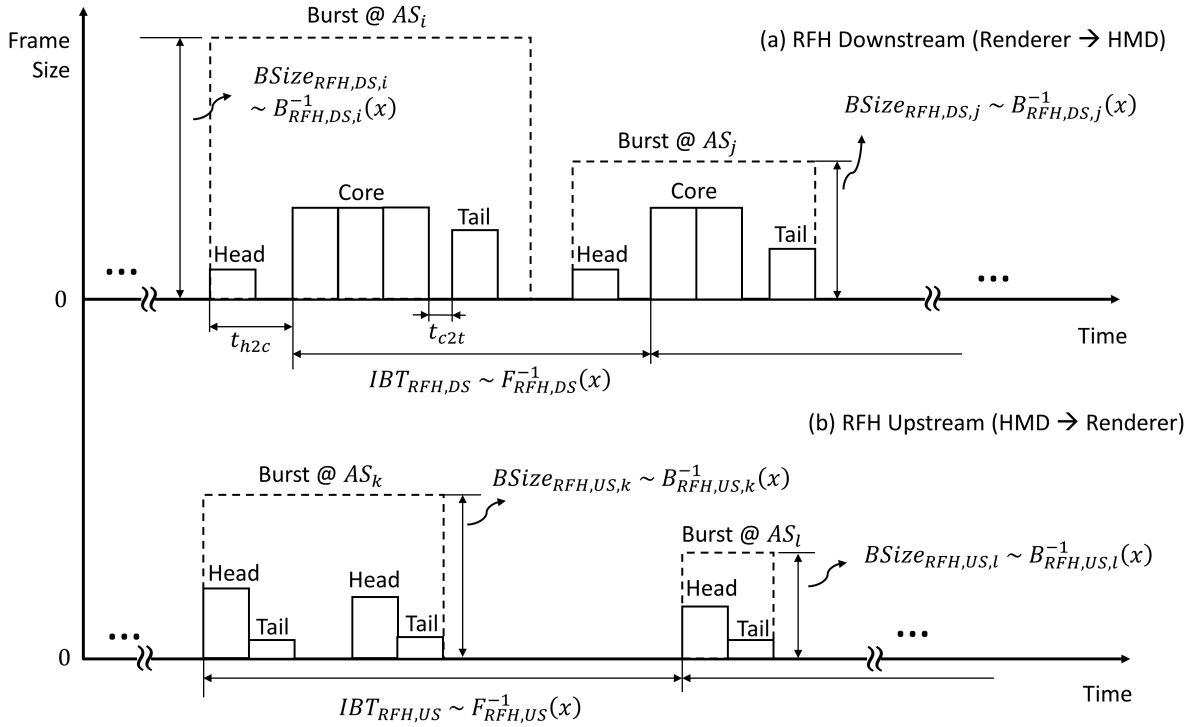

FIGURE 10. RFH traffic patterns illustrated: (a) Downstream and (b) Upstream.

TABLE 5. Application states in the remote rendering test with RFH.

App. State	ID	Actions
HMD Connection Established	0	Remote rendering machine is connected to the HMD.
Remote Rendering Enabled	1	"VR Preview" is enabled, but no XR content is initiated.
Static XR Content Displayed	2	A static XR image is created and displayed on the HMD.
User Input Detected, Single-Hand	3	One user hand is shown and detected by the HMD.
User Input Detected, Double-Hand	4	Two user hands are shown and detected by the HMD.
XR Content Control, Single-Hand	5	The user manipulates the XR content with one hand.
XR Content Control, Double-Hand	6	The user manipulates the XR content with both hands.
XR on Display w/ HMD Rotation	7	The user rotates the HMD to change the XR content in the view.

TABLE 6. Frame types in RFH downstream per the frame prefix.

Frame Type	Subset Frame Ratio	Subset Load Ratio
0x8023	98.32%	98.63%
0x8000	0.86%	0.05%
0x8065	0.65%	1.14%
Others	0.17%	0.18%

can be clustered into parallel streams. ELTA results also indicated that the traffic pattern varies with the active AS. In Fig. 11, the medium and large frames exhibit unique patterns depending on the presence of user hand input in consecutive application states, where deviations from the trend for accumulated data can be observed for example between $t = 100$ s and $t = 200$ s reflecting AS 3 and AS 4 per Table 5. Traffic load variations in response to different states are captured in Fig. 11b, where similar observations are made. In addition, the 2-D scatter plot that considers frame size and IFT variables further supports that RFH DT data constitutes compound traffic streams. Accordingly, the modeling effort benefits from decomposing the aggregated traffic stream into several statistically significant pattern elements to reduce the model complexity and improve its accuracy.

Accordingly, the DT data prefix was inspected up to the first 40 bytes of the UDP payload and results reported in Table 6. Frames with a "0 × 8023" prefix dominated the observations with 98.32% of total frames and 98.63% of total loads. Frames with "0 × 8065' prefix followed with 0.65% of total frames and 1.14% of total loads. Other frames only occupied 0.23% of the total load and were not considered in the traffic modeling, which focused on characterizing two independent traffic streams.

The "0 × 8065" frames were present in the RFH DT data set at all times, even before the XR content was introduced to the link. Accordingly, it is plausible that these are control messages, which is further supported by observing their constant frame size of 1380 bytes and IFT = 1 sec, which resembles heart beat signals at 1 Hz. The data rate was calculated around 11 kbps.

The "0 × 8023" frames exhibited varying frame sizes and IFTs. Notably, these frames, especially the larger ones

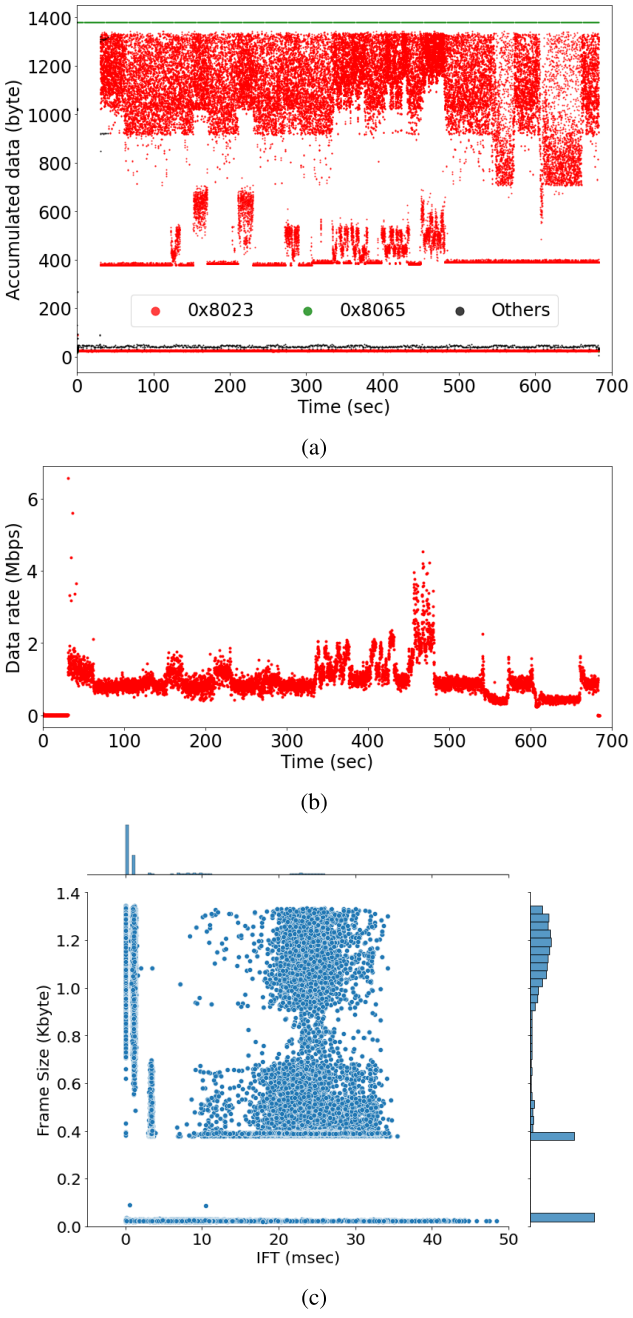

FIGURE 11. Selected ELTA findings in the RFH DT data: (a) frame size variation with time, (b) data rate in time (per 0.1 sec), and (c) joint size and IFT distribution.

(> 800 bytes), were commonly within discrete "bursts". The frames belonging to a burst were generated within 10s of microseconds while the inter-burst interval (IBT) was at the scale of 10s of milliseconds. Within a burst, not all frames were of the same size or generated at the same interval. As depicted in Fig. 10, there are three types of frames in a burst, namely "head", "core", and "tail", following their size and relative timing. Head frames were observed at the beginning of a burst with the smallest frame size among the three types (<100 bytes). Core frames were the primary frames with the largest frame size (>800 bytes) and were observed as multiple consecutive frames. Tail frames appeared at the end of a burst with a moderate frame size (around 400–800 bytes).

Differences among the sub-burst frames can also be verified by their unique labels embedded in data messages, where head frames shared a common field "0 × 0000" immediately after the prefix "0 × 8023". Values of the same field in core and tail frames were found to increment by one in each new instance, which counted for core and tail frames separately. In addition, core and tail frames could be differentiated by the 24-th hex digit in their data: the former with "0 × 0" (or "b0000") and the latter with "0 × 2" (or "b0010"), which was a one-bit flag toggled for different frame types.

Fig. 12 illustrates the sub-burst frames with time. Tail frames were clearly correlated with the display of content responding to user's input, e.g., highlighting the user's hand(s) once detected, as such frame sizes increased significantly for application states when the user's input was activated. Core frames can be correlated to the display of XR content, e.g., 2D/3D virtual objects, as they carried the most significant portion of the link traffic load.

Core frames were periodically generated with a frequency of ∼30 Hz, which also reflects the 30 frames per second (fps) refresh rate of the default configuration in the UR editor. This is illustrated in Fig. 13 plotting the distribution of inter-core sequence intervals, i.e., the interval between two core frame sequences as shown in Fig. 10. Notably, there are spikes on the shoulders of the interval distribution at a distance of around 1 msec. Causes for this jitter can include delays in frame generation associated with delay compensation mechanisms, internal clock drift, and the non-realtime system threads at the rendering machine,[1] as well as buffering delays at the LAN router. At least one head or tail frames is observed in every burst and both were commonly present. Fig. 14 illustrates the intra-burst interval distributions including the head-to-core intervals and tail-to-core intervals. Notably, the negative values shown in Fig. 14(b) indicate that tail frames were generated earlier than core frames in some bursts.

The above observations inspired the focus on depicting the burst-wise interval and load patterns in the traffic model. Therefore, the periodicity of core frames was used to represent the inter-burst time because the majority of burst load occurred at and around core frames. Tail frames were also observed to appear before the core frame sequence when head frames were missing in these bursts. However, tail frames were generally distributed around core frames within a few milliseconds as shown in Fig. 14b. Although head frames were independently distributed in time as shown in Fig. 14, their relatively small frame sizes, i.e., 20+ bytes (<1% of the burst size), could be neglected in the load analysis.

The burst size was calculated as the sum of the corresponding head/core/tail frame sizes. Since we observed that the

[1] Windows 10 on a Dell laptop with a NVIDIA RTX A3000 GPU.

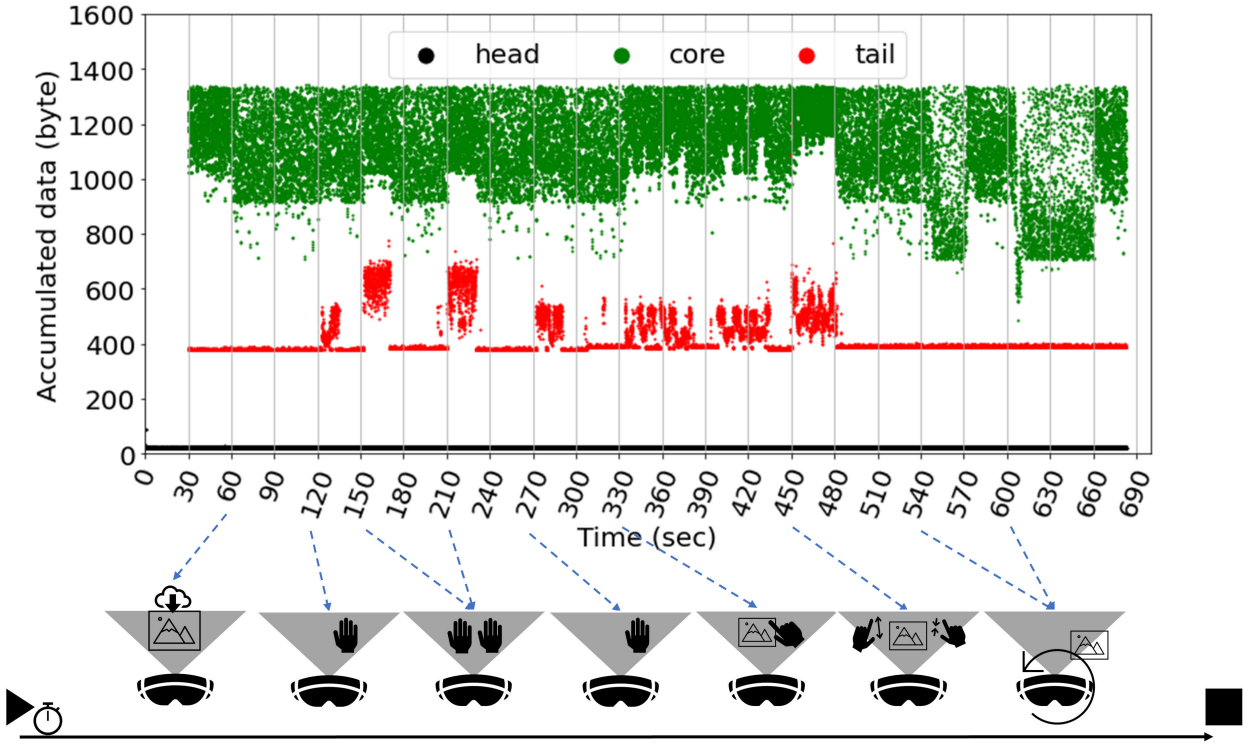

FIGURE 12. RFH Downstream sub-burst (head/core/tail) frames in time for the dominant "0 × 8023" frames.

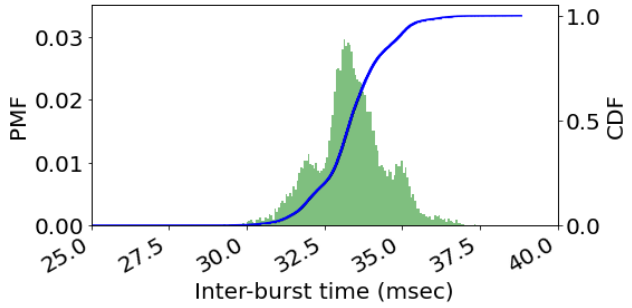

FIGURE 13. RFH DT inter-core sequence interval distribution for the dominant "0 × 8023" frames.

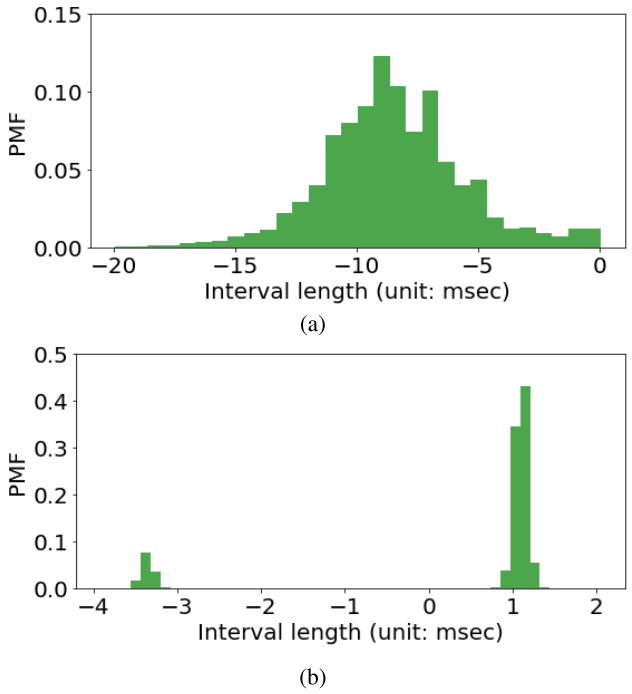

FIGURE 14. Distributions of intra-burst frame intervals in RFH DT bursts for the dominant "0 × 8023" frames: (a) head-to-core intervals and (b) core-to-tail intervals.

RFH DT data rate also varied with the active application state, we factored in these states when obtaining burst statistics. We then focused on analyzing variations of the burst size distribution with specific application states as introduced in Table 5. State-wise burst size distributions are shown in Fig. 15. The IBT variation with state switches were then investigated to document that IBT instances were centered around 33.3 msec for the 30 fps refresh rate, which did not change with changing application states.

2) RFH UT ANALYSIS

Table 7 illustrates dominant frame types RFH UT data as distinguished by the prefix values stored in the first 4 hex digits of frame data. For the purpose of traffic load modeling, the top two frame types in the load, i.e., "0 × 8000" and "0 × 8065", were considered which occupied up to 98.32% of the total upstream load. Among the selected frames,

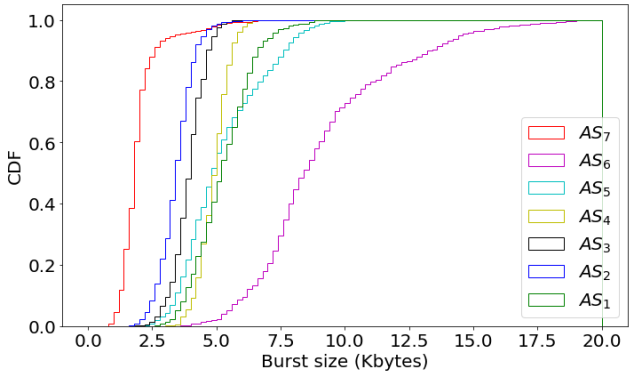

FIGURE 15. RFH DT state-wise burst size distributions for the dominant "0 × 8023" frames in CDF.

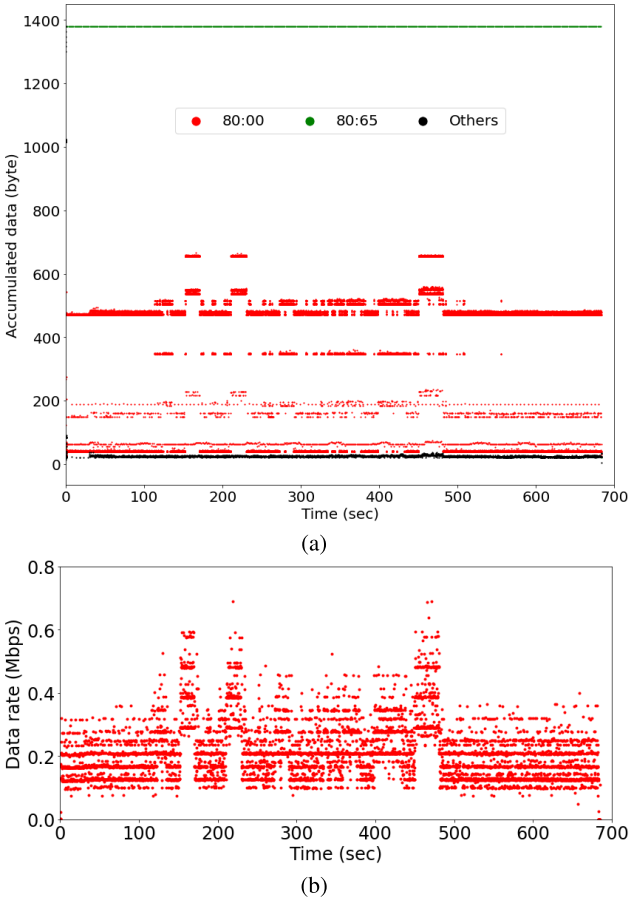

FIGURE 16. ELTA results for the RFH UT data: (a) frame size variation with time and (b) data rate in time (per 0.1 sec).

TABLE 7. Frame types in RFH upstream per the frame prefix.

Frame Type	Subset Frame Ratio	Subset Load Ratio
0x8000	87.62%	93.29%
0x8023	11.04%	0.94%
0x8065	1.07%	5.03%
Others	0.27%	0.74%

"0 × 8065" frames were similar to their downstream peers that appeared once every 1 second as UT heart beat signals, each with a data size of 1380 bytes. Therefore, the focus

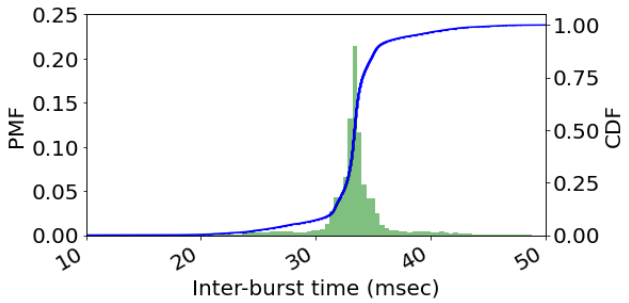

FIGURE 17. RFH upstream IBT distribution for the dominant "0 × 8000" frames.

was on recognizing the pattern of "0 × 8000" frames that varied with time, size, and application states. Inspecting the occurrence frequency of "0 × 8000" frames revealed that they commonly appeared in pairs which could be labeled as "head" and "tail" frames, as shown in Fig 10. Therefore, similar to the downstream terminology, the upstream "burst" also denoted frame clusters in time. Note that a burst may contain one or more frame pairs while all included frames are identified with the same 4 hex-digit index number immediately following the frame type prefix "0 × 8000". This feature was used to identify individual bursts and characterize burst properties.

The burst start time is considered the start time of the first frame in a burst. Fig. 17 illustrates the IBT distribution in UT, which echos the 30 fps refresh rate for the XR content update in the DT and indicates that the RFH node also updated its own status with the rendering machine at the same frequency.

The burst size varied with application state changes in discrete values containing the upstream content (e.g., the captured user input through embedded HMD sensors and cameras). Fig. 19 illustrates the primary burst sizes per application state in terms of probability mass functions (PMF).

3) MODEL IMPLEMENTATIONS

Algorithm 3 presents an example implementation of the traffic model that supports both RFH DT and UT simulations with the difference being the reference burst size and IBT distributions applied in the algorithm. Specifically, for F_{IBT}

Algorithm 3 RFH Traffic Data Generation

Input: $F_{TBS}, F_{IBT}, T_{stop}, f_{AS}(t)$
Output: $[(b^i_{time}, b^i_{size})]^n_{i=0}$

$b^0_{time} \leftarrow 0$
$k \leftarrow 0$
while $b^k_{time} < T_{stop}$ **do**
  $r \leftarrow X \sim U(0,1)$
  $b^{k+1}_{time} \leftarrow b^k_{time} + F^{-1}_{IBT}(r)$
  $s \leftarrow f_{AS}(b^{k+1}_{time})$
  $r \leftarrow X \sim U(0,1)$
  $b^{k+1}_{size} \leftarrow F^{-1}_{TBS}(s,r)$
  $k \leftarrow k+1$

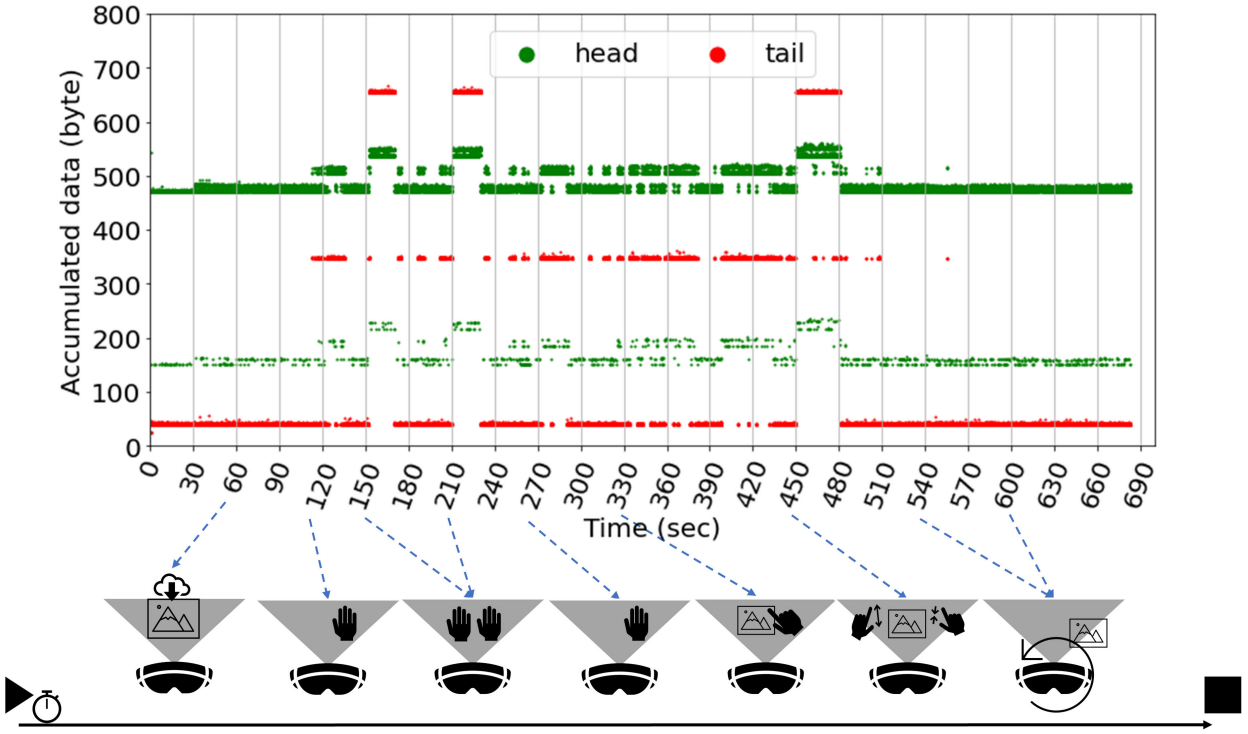

FIGURE 18. RFH Upstream sub-burst (head/tail) frames in time for the dominant "0 × 8000" frames.

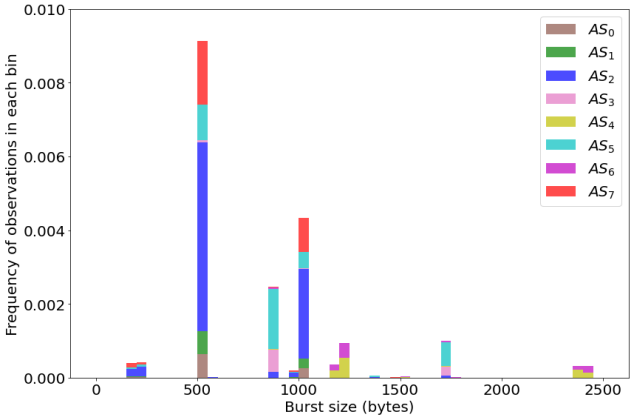

FIGURE 19. Counts of RFH UT burst sizes over application states (displayed with the primary "0 × 8000" frames in 50 bins between 0 and 2500 bytes).

TABLE 8. Simulation results in RFH links.

			Simulated	Measured
AS_2	Trace Length		$\sim 260sec$	
	DT	Rate: kbps (Mean, Std)	838.65, 56.40	839.81, 112.15
		Rate Diff.	$\pm 100\ kbps$: 80.37% $\pm 150\ kbps$: 92.08%	
	UT	Rate: kbps (Mean, Std)	160.76, 15.15	160.83, 20.63
		Rate Diff.	$\pm 30\ kbps$: 86.74% $\pm 40\ kbps$: 95.08%	
AS_6	Trace Length		$\sim 30sec$	
	DT	Rate: kbps (Mean, Std)	2167.87, 155.94	2140.82, 639.62
		Rate Diff.	$\pm 600\ kbps$: 81.25% $\pm 1000\ kbps$: 90.63%	
	UT	Rate: kbps (Mean, Std)	366.40, 30.65	356.40, 69.69
		Rate Diff.	$\pm 40\ kbps$: 87.50% $\pm 80\ kbps$: 93.75%	

that depicts the IBT randomness, the DT traffic simulator uses the empirical CDF shown in Fig. 13 while the UT one uses the IBT model shown in Fig. 17. Since the RFH traffic blocks were discussed in terms of bursts, for F_{TBS} depicting the burst size, the DT simulator selects the state-specific empirical CDF per Fig. 15 while UT adopts the discrete PMF matched with the desired application state s.

DT and UT flows were simulated for application states AS_2 and AS_6 to study the performance of RFH traffic models. Results are plotted in Fig. 20 and Fig. 21 for DT and UT, respectively. Additionally, the simulated traffic traces are compared with the measured ones and the results reported in Table 8.

Specifically, Fig. 20 reports that the DT model generates the application session rate at each state and closely matches the realistic session rate measured at the MXR rendering machine. The deviation in the rate distribution is plotted in Fig. 20(a) and Fig. 20(b) and can be explained by the wide TBS range in DT for dynamic MXR content carried by individual bursts. The needed TBS per burst is not constant, even within the same application state. For instance, AS_2 describes the static operation by the user while staring at a 2-D virtual medical image. However, the user's micro-movements, either intentional or unintentional, could result in changes of the TBS per burst to adjust the FOV display. Furthermore, when the user interacts with virtual

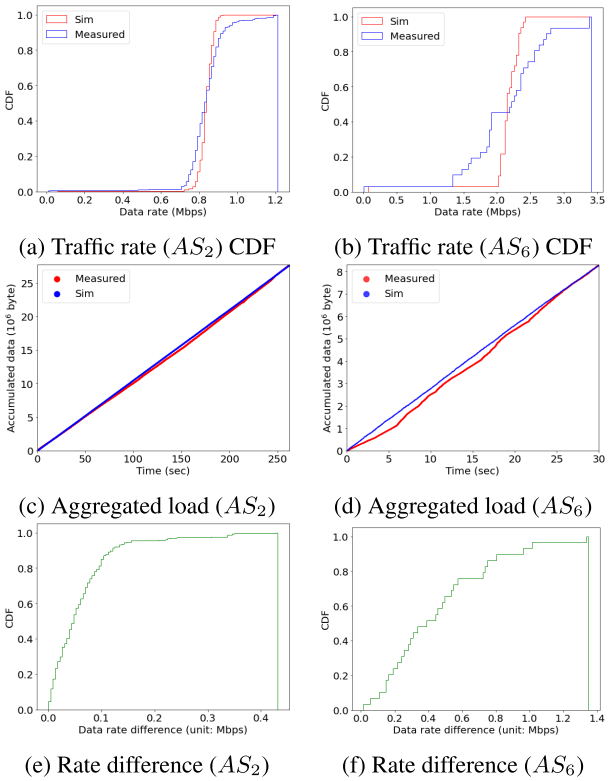

FIGURE 20. Results of simulated RFH DT traces in comparison with the measured traffic traces. AS_2 and AS_6 are presented as examples. Data rate is presented per second.

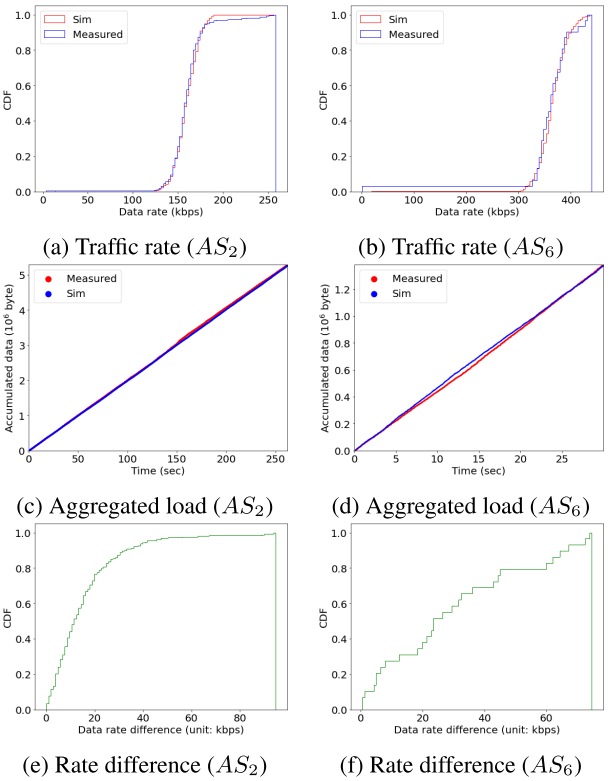

FIGURE 21. Results of simulated RFH UT traces in comparison with the measured traffic traces. AS_2 and AS_6 are presented as examples. Data rate is presented per second.

content, e.g., the double-hand MXR manipulation defined by AS_6, human-device interactions (HDI) would affect the TBS variation at the macro level in time, e.g., TBS instances being correlated in a few seconds. Further improvement of the traffic model accuracy would require additional measurement, analysis, and modeling of HDI behaviors, which is beyond the scope of this paper. Considering the long-term link load performance, Fig. 20(c) and Fig. 20(d) illustrate that the model can match the measured traffic load. Numerical analysis results as reported in Table 8 and indicate that the delta value between the simulated and measured rate stays within a limited range for the majority of the simulated period, which is a fraction of the rate level. For example, in AS_2 the simulated DT rate is limited to ± 100 kbps from the measured rate for 80.37% of the time.

Adopting the same implementation procedures, the UT simulation achieves better performance compared to the DT one as shown in Fig. 21. This is primarily due to the stable UT load per burst instance and the limited TBS values illustrated in Fig. 19.

C. HMS TRAFFIC PATTERN ANALYSIS

The HL2 HMD provides a streaming service for mirroring its view on the display of a separate user device through the built-in Device Portal API. Accordingly, HMS trace data were collected for both DT and UT. DT is from the mirroring server (i.e., HMD) to the observing client (i.e., a computer running VLC media player [27]). UT is from the player back to the HMD. Unlike traffic streams captured in the previous two cases, mirrored content from the HMD was transmitted through TCP messages. In addition, the UT link carried only acknowledgement (ACK) messages from the player in response to downstream TCP transmissions. Since there is no application data transmitted in the UT link, modeling was done only for DT traffic pattern in HMS.

1) HMS DT ANALYSIS

Fig. 22 illustrates observations of the HMS DT traffic pattern. Starting at the raw PCAP trace data, the TCP protocol analysis tool in Wireshare was used to detect and reproduce each DT stream application frame that was carried by one or multiple TCP messages. Additional time-series analysis using the timestamp of each data frame unveiled that the DT application data was generated in bursts. In each burst, DT data was carried by multiple frames with maximum frame size of 16,408 bytes and inter-frame time within 1 msec. In most burst cases, there were up to five frames. However, in some rare cases, a burst contained up to 14 frames. Since almost all frame elements were generated within 10 msec from the burst start, i.e., the first frame element time, it is still plausible to use the burst as the traffic block to characterize the HMS DT pattern. Accordingly, TBS and IBT distributions, respectively, are characterized in the traffic model.

DT bursts were generated following the IBT distribution shown in Fig. 23. Notably, there were differences in how

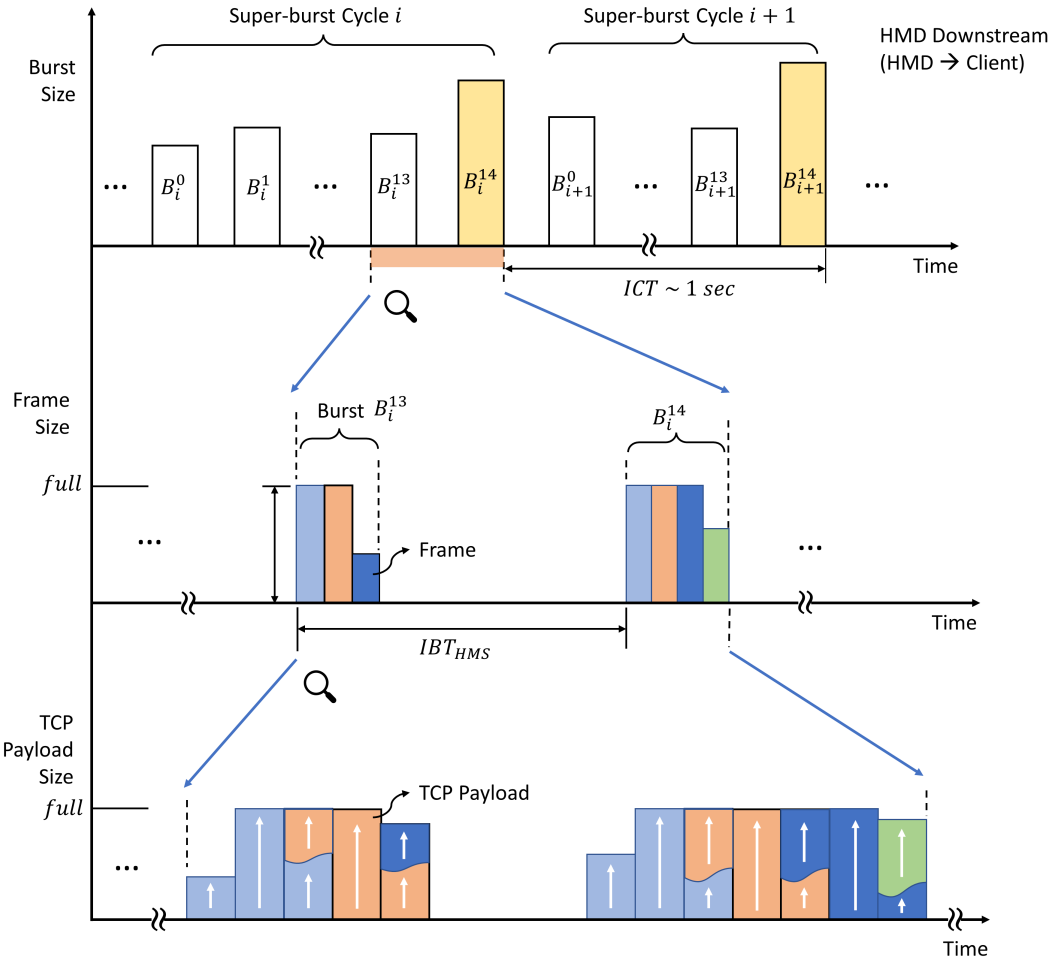

FIGURE 22. HMS traffic pattern illustrated (Downstream only).

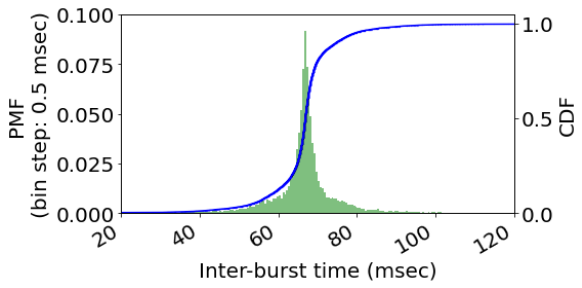

FIGURE 23. IBT distribution of the HMS DT data set.

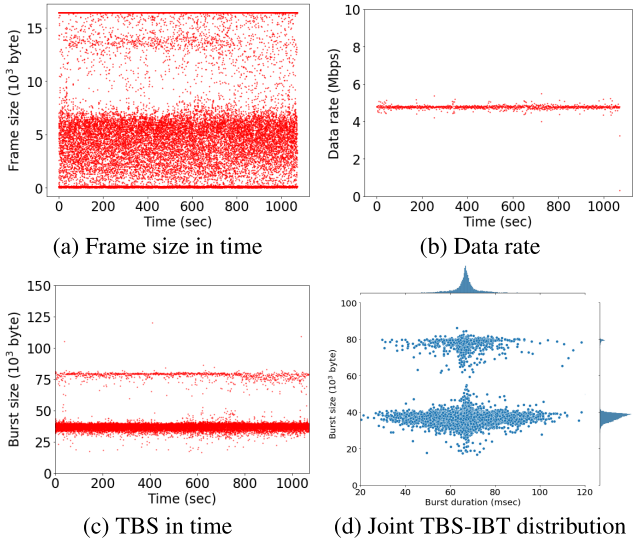

FIGURE 24. ELTA results for the HMS DT data set.

the data was transmitted by the server and the way it was consumed by the client. Fig. 23 indicates that the IBT is around 66 msec (i.e., around 15 Hz) while the VLC player at the receiver replayed the streamed traffic as an MP4 live video stream at the rate of 30 fps, which was twice the traffic rate. There was a noticeable delay of about a few seconds from the headset display to the replicated view at the receiver, which indicates that the "live" traffic was buffered before replay.

Fig. 24 illustrates ELTA results based on frame and burst samples identified in HMS DT data, where IBT and TBS distributions shown in Fig. 24(d) can be directly used to replicate the traffic. The implementation considers independent and identically distributed (IID) random variables for TBS and IBT by running Algorithm 3 with a single application state. However, the IID model for HMS DT does

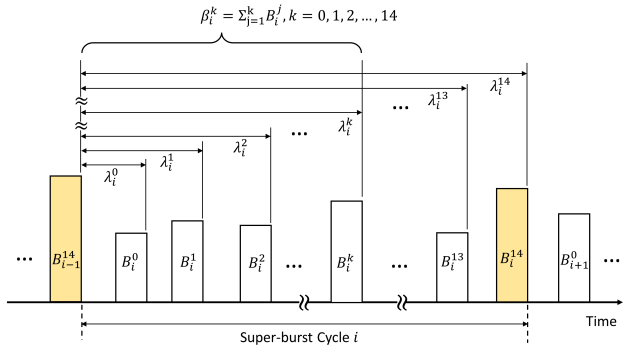

FIGURE 25. Aggregated burst delay and load size illustrated in the HMS DT super-burst cycle (SBC).

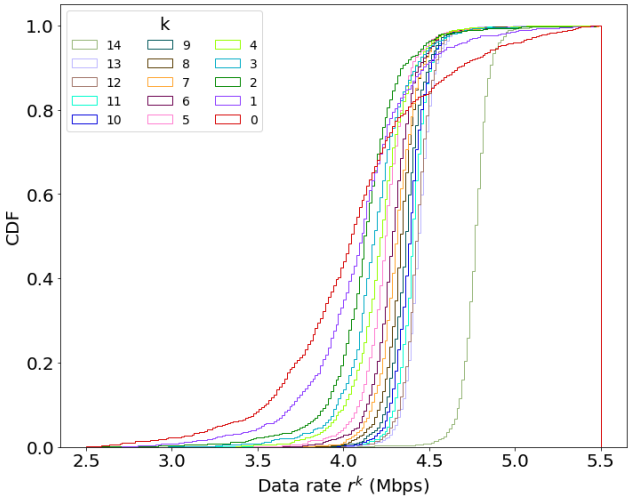

FIGURE 26. Aggregated rate distribution per burst element in a super-burst cycle.

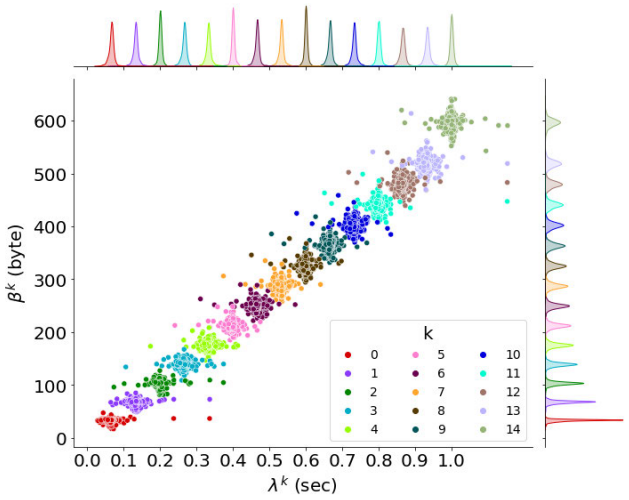

FIGURE 27. Joint distribution of the aggregated burst delay λ and the aggregated burst size β in an SBC in the measured HMS DT data set.

not consider time-series characteristics in the transitions from one burst to the next, which results in significant deviations from the realistic load. Notably, the measured session data rate is ∼5 Mbps. According to the HL2 developer manual, the HMD live stream service is provided with multiple rate options of which we select the one offering the highest rate at around 5 Mbps for the 720P (1280 × 720) resolution quality and 30 fps refresh rate. Accordingly, The HL2 HMD encodes video content into video frames and ensures the transmission rate meets the pre-selected goal. Therefore, we introduce the effect of the rate-control design implemented in this use case leveraging observations in the data set (i.e., absent complete knowledge of the rate control algorithm adopted by the HMD.)

In addition to identifying the data burst structure, a time-series analysis of DT data documents the burst periodicity. Approximately 15 bursts are transmitted in each cycle, which includes an ending burst that is usually bigger than the others. The group of bursts is named a super-burst cycle (SBC) in our model. The first 14 burst elements (i.e, regular bursts in SBC) have a size below 60,000 bytes while the ending burst element (i.e., the "pilot" in SBC) is > 60,000 bytes, which conforms to the observed TBS distribution in Fig. 24d.

The SBC is illustrated in Fig. 25, where the size of each burst element k in an SBC i can be denoted as B_i^k. Accordingly, the aggregated burst size β_i^k is defined as the sum of the sizes of burst elements whose indices are less or equal to k. In addition, the aggregated burst delay λ_i^k is denoted by the delay from the start of the SBC i to the burst k. Therefore, the IBT between the burst k and $k+1$ can be calculated by $\lambda_i^{k+1} - \lambda_i^k$. The aggregated burst rate r_i^k per burst element can also be presented as $r_i^k = \beta_i^k / \lambda_i^k$.

The aggregated rate distribution is plotted in Fig. 26 per the collected data set. As a result of the HMD rate control, each intermediate burst in the SBC is an attempt to tuning the aggregated data rate to the target value (i.e., 5 Mbps, which was measured around 4.8 Mbps in the data set), and the ending burst regulates the aggregated rate with respect to the target. However, there were random events observed that contributed to deviations from the target.

To characterize time variations of bursts under the rate control mechanism in the DT model, the rate control period was assumed to be the same as the SBC, i.e., around 1 sec. The mirroring server can select its control goal for the rate in multiple ways. Absent technical specifications for the actual rate control design implemented in the HL2's mirroring service, the following options were considered in the traffic model.

1) **"SBC ON".** In this rate control option, the traffic model introduces the periodicity of different burst element types, i.e., regular and pilot, defined in the SBC structure. Each burst element uses the specific TBS distribution to determine the value of its size in the simulated traffic.
2) **"SBC, Burst".** Based on the definitions of aggregated burst size and delay, the IBT and TBS for each burst

Algorithm 4 HMS DT Frame Generation (w/ "SBC, Burst")

Input: $\{F_{\lambda,\beta,k}\}_{k=0}^{14}$, T_{stop}
Output: $\left[[(b_{time}^{i*14+k},\ b_{size}^{i*14+k})]_{k=0}^{14}\right]_{i=0}^{n}$

$b_{time}^{0} \leftarrow 0$
$i \leftarrow 0$
while $b_{time}^{i*14} < T_{stop}$ **do**
  $k \leftarrow 0,$
  $r \leftarrow X \sim U(0,1)$
  $(\lambda_i^k, \beta_i^k) \leftarrow F_{\lambda,\beta,k}^{-1}(r)$
  while $k < 15$ **do**
    $k \leftarrow k+1$
    $r \leftarrow X \sim U(0,1)$
    $(\lambda_i^k, \beta_i^k) \leftarrow F_{\lambda \geq \lambda_i^{k-1}, \beta \geq \beta_i^{k-1}, k}^{-1}(r)$
  $k \leftarrow 0$
  while $k < 15$ **do**
    $b_{time}^{i*14+k} \leftarrow b_{time}^{i*14} + \lambda_i^k$
    $b_{size}^{i*14+k} \leftarrow \beta^{k+1} - \beta^k$
    $k \leftarrow k+1$
  $i \leftarrow i+1$

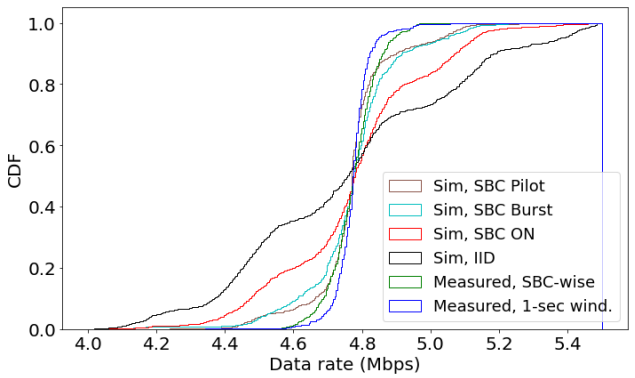

FIGURE 28. HMS DT measured and simulated application rate distributions. "Measured, 1-sec wind." calculates the rate per the 1-sec window; "Measured, SBC-wide" presents the rate in each SBC period; the rate in all simulated cases is calculated per 1-sec window.

element are determined by the joint (λ, β) distribution shown in Fig. 27.

3) **"SBC, Pilot".** Distinguished from "SBC, Burst" by is fixing the aggregated rate for the pilot burst to the rate target.

2) MODEL IMPLEMENTATIONS

Algorithm 4 is introduced to implement the HMS DT model for simulating streaming traffic, which uses the "SBC, Burst" case. Other variants can be similarly implemented. For example, Algorithm 5 can support the "SBC, Pilot" option by assigning the aggregated rate at the Pilot to be closest to the rate target.

Fig. 28 presents distributions of the application rate in the measured and simulated traffic data sets. Notably, the simulated rate using the "SBC, Pilot" model is closer to the measured one compared with other model variants because it considers the most information about the realistic case.

Algorithm 5 HMS DT Frame Generation (w/ "SBC, Pilot")

Input: $\{F_{\lambda,\beta,k}\}_{k=0}^{14}$, R_{target}, T_{stop}
Output: $\left[[(b_{time}^{i*14+k},\ b_{size}^{i*14+k})]_{k=0}^{14}\right]_{i=0}^{n}$

$b_{time}^{0} \leftarrow 0$
$i \leftarrow 0$
while $b_{time}^{i*14} < T_{stop}$ **do**
  $k \leftarrow 0$
  $r \leftarrow X \sim U(0,1)$
  $(\lambda_i^k, \beta_i^k) \leftarrow F_{\lambda,\beta,k}^{-1}(r)$
  while $k < 15$ **do**
    $k \leftarrow k+1$
    if $k == 14$ **then**
      $(\lambda_i^k, \beta_i^k) \leftarrow \underset{(\lambda,\beta)}{\operatorname{argmin}} |\beta/\lambda - R_{target}|$ for $(\lambda, \beta) \in F_{\lambda \geq \lambda_i^{k-1}, \beta \geq \beta_i^{k-1}, k}^{-1}$
    else
      $r \leftarrow X \sim U(0,1)$
      $(\lambda_i^k, \beta_i^k) \leftarrow F_{\lambda \geq \lambda_i^{k-1}, \beta \geq \beta_i^{k-1}, k}^{-1}(r)$
  $k \leftarrow 0$
  while $k < 15$ **do**
    $b_{time}^{i*14+k} \leftarrow b_{time}^{i*14} + \lambda_i^k$
    $b_{size}^{i*14+k} \leftarrow \beta^{k+1} - \beta^k$
    $k \leftarrow k+1$
  $i \leftarrow i+1$

VII. CONCLUSION

Link-level traffic models offer foundational knowledge in how MXR applications communicate and their associated needs for connectivity QoS. In this paper, we have developed an end-to-end data-oriented modeling workflow for measuring, analyzing, and reproducing link traffic loads of MXR application. The diversity of traffic loads in selected MXR implementation examples was observed and documented, which enriches existing literature reports on XR traffic patterns, commonly focused on gaming. Algorithms for model implementation were proposed and their overall performance was evaluated. Results indicate that our traffic models can emulate link-level traffic loads in individual application scenarios including projected variations in time and application states. This work contributes to the development and evaluation of novel MXR devices. It can assist MXR device developers in identifying their applications' transmission behaviors and needs and selecting relevant connectivity modalities and configurations to support those needs. It also provides reference traffics to network engineers for designing and optimizing connectivity solutions relevant to emerging MXR use cases and aid in the development and integration of existing or future protocols and standards for network QoS. Future work includes the integration of developed models into the evaluation of connected MXR applications enabled by 5G and next generation Wi-Fi.

ACKNOWLEDGMENT

The authors would like to thank Cheng-Yu Cheng for developing the MXR demo during his ORISE fellowship at FDA/CDRH/OSEL and also would like to thank Matthew Johnson, Ryan Beams, and Wei-Chung Cheng (FDA/CDRH/OSEL) for their valuable comments and suggestions on the MXR DUT application design.

DISCLAIMER

The mention of commercial products, their sources, or their use in connection with material reported herein is not to be construed as either an actual or implied endorsement of such products by the Department of Health and Human Services. This article reflects the views of the authors and should not be construed to represent FDA's views or policies.

REFERENCES

[1] U.S. Food and Drug Administration. (2022). *Augmented Reality and Virtual Reality in Medical Devices*. [Online]. Available: https://www.fda.gov/medical-devices/digital-health-center-excellence/augmented-reality-and-virtual-reality-medical-devices

[2] W. López-Ojeda and R. A. Hurley, "Extended-reality technologies: An overview of emerging applications in medical education and clinical care," *J. Neuropsychiatry Clin. Neurosciences*, vol. 33, no. 3, p. 177, Jul. 2021.

[3] R. Beams, E. Brown, W.-C. Cheng, J. S. Joyner, A. S. Kim, K. Kontson, D. Amiras, T. Baeuerle, W. Greenleaf, R. J. Grossmann, and A. Gupta, "Evaluation challenges for the application of extended reality devices in medicine," *J. Digit. Imag.*, vol. 35, no. 5, pp. 1409–1418, Apr. 2022.

[4] Verizon. (2023). *With 5G-Enabled Hospitals for Veterans, a Better Future of Healthcare is on the Horizon*. [Online]. Available: https://www.verizon.com/business/resources/casestudies/5g-enabled-hospitals-for-veterans.pdf

[5] *Tim Uses 5G to Enable Remote Eye Surgery at Bari Polyclinic*. Accessed: Oct. 3, 2023. [Online]. Available: https://tecknexus.com/5gusecase/tim-uses-5g-to-enable-remote-eye-surgery-at-bari-polyclinic/

[6] Medical Device Innovation Consortium (MDIC). (2022). *Landscape Analysis of 5G in Healthcare*. [Online]. Available: https://mdic.org/resource/landscape-analysis-of-5g-in-healthcare/

[7] U.S. Food and Drug Administration (FDA). (2013). *Radio Frequency Wireless Technology in Medical Devices—Guidance for Industry and FDA Staff*. [Online]. Available: https://www.fda.gov/media/71975/download

[8] F. M. García, R. Moraleda, S. Schez-Sobrino, D. N. Monekosso, D. Vallejo, and C. Glez-Morcillo, "Health-5G: A mixed reality-based system for remote medical assistance in emergency situations," *IEEE Access*, vol. 11, pp. 59016–59032, 2023.

[9] C.-Y. Cheng, Y. Liu, M. Johnson, R. Beams, and M. O. A. Kalaa, "Assessing 5G connectivity in medical extended reality (MXR) applications: A testbed approach," in *Proc. IEEE Healthcom*, Dec. 2023, pp. 1–6.

[10] *Study on XR (Extended Reality) Evaluations for NR*, document 3GPP TS38.838, 2022.

[11] *Extended Reality (XR) in 5G*, document 3GPP TS26.928, 2020.

[12] F. Metzger, S. Geißler, A. Grigorjew, F. Loh, C. Moldovan, M. Seufert, and T. Hoßfeld, "An introduction to online video game QoS and QoE influencing factors," *IEEE Commun. Surveys Tuts.*, vol. 24, no. 3, pp. 1894–1925, 3rd Quart., 2022.

[13] M. Lecci, A. Zanella, and M. Zorzi, "An ns-3 implementation of a bursty traffic framework for virtual reality sources," in *Proc. Workshop NS-3*, Jun. 2021, pp. 73–80.

[14] M. Lecci, M. Drago, A. Zanella, and M. Zorzi, "An open framework for analyzing and modeling XR network traffic," *IEEE Access*, vol. 9, pp. 129782–129795, 2021.

[15] E. Rojas-Muñoz, M. E. Cabrera, D. Andersen, V. Popescu, S. Marley, B. Mullis, B. Zarzaur, and J. Wachs, "Surgical telementoring without encumbrance: A comparative study of see-through augmented reality-based approaches," *Ann. Surg.*, vol. 270, no. 2, pp. 384–389, Aug. 2019.

[16] M. Lecci and M. Drago. *An Implementation of a Traffic Model for VR Applications on NS-3*. Accessed: Mar. 11, 2024. [Online]. Available: https://github.com/signetlabdei/ns-3-vr-app

[17] National Science Foundation (NSF). (2023). *Platforms for Advanced Wireless Research (PAWR)*. [Online]. Available: https://advancedwireless.org/

[18] Y. Liu and M. O. A. Kalaa, "Testbed as a RegUlatory science tool (TRUST): A testbed design for evaluating 5G-enabled medical devices," *IEEE Access*, vol. 11, pp. 81563–81576, 2023.

[19] Microsoft. *Mixedreality-Unreal-Samples/Hl2Collab At Master Microsoft/Mixedreality-Unreal-Samples*. Accessed: Mar. 11, 2024. [Online]. Available: https://github.com/microsoft/MixedReality-Unreal-Samples/tree/master/HL2Collab

[20] X. Hou, Y. Lu, and S. Dey, "Wireless VR/AR with edge/cloud computing," in *Proc. 26th Int. Conf. Comput. Commun. Netw. (ICCCN)*, Jul. 2017, pp. 1–8.

[21] P. Hande, P. Tinnakornsrisuphap, J. Damnjanovic, H. Xu, M. Mondet, H. Y. Lee, and I. Sakhnini, "Extended reality over 5G—Standards evolution," *IEEE J. Sel. Areas Commun.*, vol. 41, no. 6, pp. 1757–1771, Jun. 2023.

[22] Microsoft. *Mixed Reality Streaming*. Accessed: Mar. 11, 2024. [Online]. Available: https://learn.microsoft.com/en-us/windows/mixed-reality/develop/advanced-concepts/device-portal-api-reference#mixed-reality-streaming

[23] Holo-Light GmbH. *Interactive Streaming for Augemnted Reality (ISAR) Software Development Kit*. Accessed: Mar. 11, 2024. [Online]. Available: https://github.com/Holo-Light-GmbH/ISAR-SDK-Trial

[24] H. N. Qureshi, M. Manalastas, A. Ijaz, A. Imran, Y. Liu, and M. O. Al Kalaa, "Communication requirements in 5G-enabled healthcare applications: Review and considerations," *Healthcare*, vol. 10, no. 2, p. 293, Feb. 2022.

[25] Wireshark Foundation. *Tshark(1) Manual Page*. Accessed: Mar. 11, 2024. [Online]. Available: https://www.wireshark.org/docs/man-pages/tshark.html

[26] *Networking Overview*. Accessed Sep. 13, 2023. [Online]. Available: https://docs.unrealengine.com/4.27/en-U.S./InteractiveExperiences/Networking/Overview/

[27] *VLC Media Player*. Accessed: Nov. 2, 2023. [Online]. Available: https://www.videolan.org/vlc/

YONGKANG LIU (Member, IEEE) received the Ph.D. degree from the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada, in 2013. He is currently a Staff Fellow with the Center for Devices and Radiological Health (CDRH), U.S. Food and Drug Administration (FDA). Before joining FDA, he was with the National Institute of Standards and Technology (NIST), Gaithersburg, MD, USA, on wireless research in vertical sector applications, e.g., process/automation control and industrial robotics. His current research interests include investigating novel uses of wireless technology (5G/Wi-Fi/bluetooth) in healthcare applications and test methods to promote safe and effective use of wireless technology in medical devices.

MOHAMAD OMAR AL KALAA (Senior Member, IEEE) received the bachelor's degree in electronics and telecommunication from Damascus University, Damascus, Syria, in 2008, the M.E. degree in advanced telecommunication from École Nationale Supérieure des Télécommunications de Bretagne, Brest, France, in 2012, and the M.Sc. and Ph.D. degrees in electrical and computer engineering from the University of Oklahoma, Norman, OK, USA, in 2014 and 2016, respectively. He is currently an Electrical Engineer with the Center for Devices and Radiological Health (CDRH), U.S. Food and Drug Administration (FDA). His research interests include the healthcare applications enabled by wireless technology, wireless coexistence of technologies in unlicensed bands, and wireless medical device testing methodologies. He served as the Co-Chair for the Medical Device Innovation Consortium (MDIC) 5G-Enabled Medical Device Working Group and the Secretary for the ANSI C63.27 Standard for Evaluation of Wireless Coexistence Working Group.