TCP-Based Distributed Offloading Architecture for the Future of Untethered Immersive Experiences in Wireless Networks

Diego González Morín
diego.gonzalez_morin@nokia-bell-labs.com
Nokia XR Lab
Madrid, Madrid, Spain

Manuel J. Lopéz Morales
jlopez@tsc.uc3m.es
Universidad Carlos III
Madrid, Madrid, Spain

Pablo Pérez
pablo.perez@nokia-bell-labs.com
Nokia XR Lab
Madrid, Madrid, Spain

Alvaro Villegas
alvaro.villegas@nokia-bell-labs.com
Nokia XR Lab
Madrid, Madrid, Spain

ABSTRACT

Task offloading has become a key term in the field of immersive media technologies: it can enable lighter and cheaper devices while providing them higher remote computational capabilities. In this paper we present our TCP-based offloading architecture. The architecture, has been specifically designed for immersive media offloading tasks with a particular care in reducing any processing overhead which can degrade the network performance. We tested the architecture for different offloading scenarios and conditions on two different wireless networks: WiFi and 5G millimeter wave technologies. Besides, to test the network on alternative millimeter wave configurations, currently not available on the actual 5G millimeter rollouts, we used a 5G Radio Access Network (RAN) real-time emulator. This emulator was also used to test the offloading architecture for an simulated immersive user sharing network resources with other users. We provide insights of the importance of user prioritization techniques for successful immersive media offloading. The results show a great performance for the tested immersive media scenarios, highlighting the relevance of millimeter wave technology for the future of immersive media applications.

CCS CONCEPTS

• **Networks → Network experimentation**; **Network performance analysis**; **Wireless access networks**; Network reliability; • **Information systems → Video search**; • **Computer systems organization → Embedded systems**; *Redundancy*; Robotics.

KEYWORDS

virtual reality, augmented reality, offloading, wireless access networks

IMX '22, June 22–24, 2022, Aveiro, JB, Portugal
© 2022 Association for Computing Machinery.
ACM ISBN 978-1-4503-9212-9/22/06…$15.00
https://doi.org/10.1145/3505284.3529963

ACM Reference Format:
Diego González Morín, Manuel J. Lopéz Morales, Pablo Pérez, and Alvaro Villegas. 2022. TCP-Based Distributed Offloading Architecture for the Future of Untethered Immersive Experiences in Wireless Networks. In *ACM International Conference on Interactive Media Experiences (IMX '22), June 22–24, 2022, Aveiro, JB, Portugal.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3505284.3529963

1 INTRODUCTION

Immersive media technologies aim to enable new manners for humans to interact with the real world, virtual objects and scenarios, and with each other, being Virtual (VR) and Augmented Reality (AR) the best known examples of such technologies. VR can be defined as the ecosystem of technologies which allows humans to fully immerse in a virtual scenario being completely or partially isolated from the real world. On the other hand, the goal of AR is to incorporate virtual objects to the local reality allowing new means of interacting with the real world. Between these two boundary immersive technologies, there are other solutions which provide other level of immersiveness such as Distributed Reality (DR) [31]. The goal of DR is to merge different remote local realities into one single shared immersive experience, which requires the local scenarios to be captured, understood, segmented and shared, considerably increasing the technological complexity. The evolution of immersive solutions can pave the road for novel use cases and applications in fields such as education, industry or human to human interactions.

The interest in immersive media technologies has exponentially increased over the last decade, recently boosted by the announcement of Facebook targeting a new manner of social interaction in what they called the Metaverse [1]. The increased interest has produced a huge investment in these technologies, which has enabled lighter and more affordable devices, and novel algorithms, improving the overall immersiveness and user experience. The increased investment in VR technologies have enable devices with unmatched levels of resolution: Varjo XR-3 [2] provides a visual resolution of 70 pixels per degree, comparable to the human eye's resolution. However, ultra-high resolution is not the only key factor for a successful VR experience: the sense of embodiment is crucial as it heavily affects VR's user experience [15] and requires demanding state of the art algorithms such as hand tracking or

[1] https://about.facebook.com/
[2] https://varjo.com/products/xr-3/

scene recognition. Similarly, AR applications should not only render high resolution virtual content, but accurately and seamlessly place it on the top of the real scenario. Thus, the real scenario has to be analyzed accurately in real-time. This process requires complex and demanding algorithms such as semantic recognition and segmentation, hand tracking, or 3D reconstruction. While there are already some examples of these algorithms [33][8] running in real-time in high-end hardware, these implementations can not run in real-time in wireless AR devices. Expanding the current boundaries of immersive technologies requires the usage of high-end hardware with powerfull GPUs for real-time rendering and machine learning (ML) processing. Consequently, the most technologically advanced immersive devices are still tethered, bulky, uncomfortable and expensive. The release of the fifth generation of telecommunication networks (5G) have brought the exploration of distributed solutions for immersive technologies as an enabler of wireless, lighter, and more affordable devices while increasing the overall immersiveness and user experience. The goal is to offload some or all of the heavy processing tasks to a nearby server increasing the available computing power while reducing the immersive device processing requirements. Both AR and VR devices handle intense data rates in a limited span of time: the update rate must be kept above 60 Hz to ensure a successful experience and avoid any discomfort, such as motion sickness, to the user [34]. Consequently, successful task offloading for VR and AR applications requires a robust network which can satisfy the extremely tight latency and throughput requirements. 5G's enhanced mobile broadband (eMBB) and ultra-reliable low-latency communication (URLLC) services provide high throughput and extremely low latency which can enable offloaded immersive solutions. The network architecture must ensure low-latency communication along the entire offloading pipeline: this can only be achieved by placing the processing servers, referred as multi-access edge computing (MEC) systems in 5G, as close as possible to the immersive device.

Both AR and VR offloading have been previously studied from a theoretical point of view. In [7], [5] and [19] different resource allocation schemes and procedures for successful VR offloading are described. Furthermore, in [26], the authors propose the use of different slicing schemes to efficiently exploit both URLCC and eMBB services to achieve the throughput and latency VR offloading requirements. Some studies [17] aim to apply ML approaches to optimize the resource usage toward low latency and reliable VR offloading schemes. Finally, the usage of millimeter wave spectrum is theoretically studied in [10] for VR offloading, in which the authors concluded that up to 4 immersive users could be successfully provided with sufficiently high throughput and low latency using a single millimeter wave access point.

Fully immersive applications must provide the users with a high sense of embodiment and presence [21], which requires complex algorithms such as hand tracking or egocentric human segmentation [12]. The theoretical network requirements, in terms of latency and downlink and uplink throughput, for successfully offloading some or all of these and other complex algorithms in different scenarios are proposed in [22].

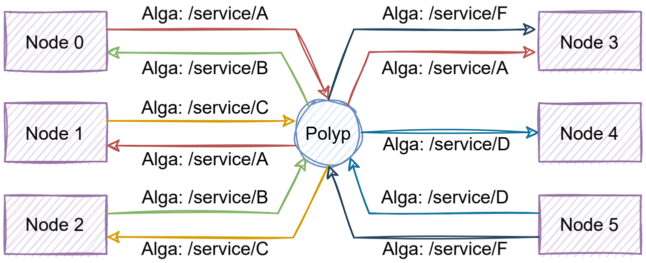

Figure 1: Simple example of a distributed system implemented with the architecture main components: Alga and Polyp.

2 OFFLOADING ARCHITECTURE

On the other hand, it is seldom to find research examples of distributed implementations or theoretical studies for AR offloading. A relevant survey of AR offloading is [28], in which the authors thoroughly describe the main offloading requirements for mobile AR continued by the description of different possible offloading architectures for a set of relevant use cases. The requirements of each AR offloading use cases are given as a whole, rather than analyzing the individual requirements of each involved algorithm. On the other hand, in [23] the authors analyze the requirements, in terms of throughput and latency, for each individual relevant AR algorithm while describing the network architecture and configuration which can fulfill such requirements in a set of scenarios.

While most of the state of the art has focused on the theoretical study of possible architectures and their requirements for successfully offloading immersive applications in 5G or beyond. Some relevant research examples have actually implement, describe an test different offloading architectures. In [14] the authors proposed an efficient offloading architecture specifically optimized or mobile AR. Besides, authors in [18] propose an offloading architecture carefully designed towards energy consumption reduction. Finally in [25], a reinforcement-learning self-optimizing distributed offloading architecture is proposed. However, none of these research examples focused on thoroughly bench-marking their proposed architectures in terms of provided throughput and latency.

In this paper we describe our novel yet simple offloading architecture from a practical point of view. Besides, we present field results of our offloading architecture for different scenarios and offloading algorithms. We aim to give a practical overview of a functional offloading implementation and present some preliminary results while highlighting what other improvements are still required both from the network and the architectures sides. The main contributions of the paper are listed below:

- Design and implementation details of our novel offloading architecture: built on top of the TCP protocol. The proposed architecture handles multiple streams from different users, allowing concurrent offloading services and traffic routing while aiming for a low latency and reliable data exchange.
- Description of the experimental approach for testing our architecture in different offloading scenarios and wireless networks: we describe the set of experiments we carried out

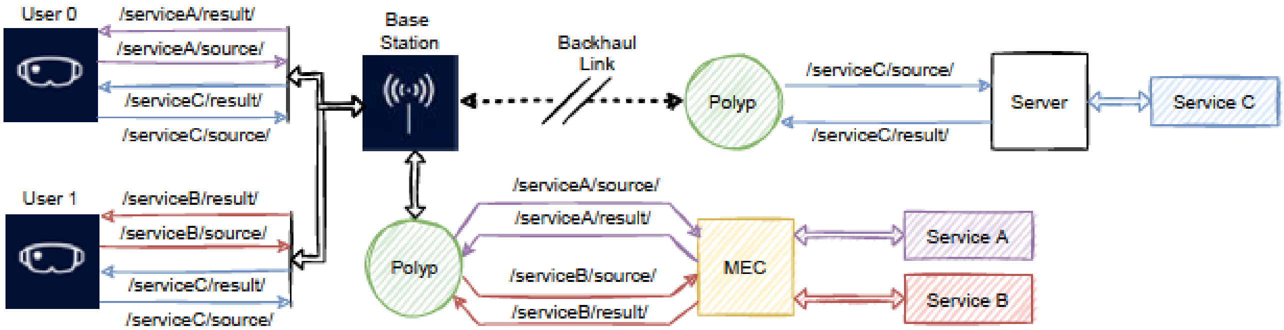

Figure 2: A distributed offloading implementation for immersive applications using the proposed architecture and 5G.

to show the performance of both our architecture and the network itself. The main goal of these experiments is to show what can currently be achieved and what is still lacking from both the network and architecture.

- Brief presentation of the obtained results: we show and analyze the obtained results, paving the ground for future researchers.

The main goal of the implemented architecture is to ensure a reliable, scalable and flexible offloading tool optimized for low latency communication. We understand the proposed architecture as a service provider for immersive media applications: the architecture back-end provides a set of services to which the immersive device subscribes to. Consequently, the data flow follows a publisher-subscriber approach, which facilitates the dissemination of the information in distributed systems [6] as the one we are proposing. We refer as nodes to each individual component of the distributed architecture which subscribes or publishes to an available data channel (see Fig. 1). The available data channels advertised by other nodes are referred as topics. Our architecture is composed of two main components which are in charge of efficiently distributing the information between the different nodes:

- Alga: is the core communication custom library which allows the direct data exchange between nodes. Alga allows a node to publish or subscribe to one or more topics, handling the data reception and transmission.
- Polyp: is the traffic routing agent which maps the publishers with their correspondent subscribers. Polyp receives publishing or subscribing petitions from the connected nodes and routes the traffic accordingly: it receives packets from the nodes, check their destination and route them to all the targeted nodes. Consequently, polyp has to be designed and implemented to ensure high data managing efficiency to avoid any unnecessary delays.

The idea of the proposed architecture is that the potential offloaded algorithms are offered by a MEC or a distributed system of processing servers as services. Each of the offered services has two unique topics assigned, one for the input and one for output. Each immersive application can publish their sensor data to arbitrary services, and can subscribe to their output using their unique output topic (see Fig. 2). With this approach not only the scalability of the proposed pipeline is ensured but it also simplifies the implementation efforts for the immersive applications providers.

To keep the communication latencies low, there should be at least one Polyp instance running as close to the MEC and the serving base station as possible. Polyp is in charge of routing the traffic from and to the services running in different instances within the nearby MEC. Besides, our architecture is designed to handle other services which doesn't require real-time processing: for these services, Polyp can communicate with other Polyp instances through the internet. In this scenario, the immersive application can stream their sensor data to further servers to do non real-time heavy processing, such as photogrammetry realistic reconstruction [20]. Consequently, the proposed architecture can simultaneously handle real-time and non-critical services using the same protocol and data flow. A simple representation of the presented offloading scenario on an example 5G network is depicted in Fig. 2.

2.1 Alga: Implementation Details

Alga is implemented as a reliable publisher-subscriber communication library based in TCP protocol. We decided to use TCP rather than UDP or other UDP-based protocols to prioritize reliability over maximizing the throughput. In both AR and VR lost frames suppose a great degradation of the user experience, specially for the most latency-critical algorithms such as VR rendering or hand segmentation. Alga is implemented using a well-known and well documented TCP-based communication library: ZeroMQ [3]. ZeroMQ has been widely tested and bechmarked, showing outstanding performance both in terms of latency and throughput [29].

ZeroMQ allows to simply connect to an arbitrary endpoint or bind to a socket and start receiving and sending data through that port. Binding is a key functionality as it allows to listen to all the packets coming through an arbitrary port regardless the endpoint source. Besides, ZeroMQ already allows to configure ports as publishers or subscribers, discarding the received packets which do not correspond to the subscribed topic. This feature, combined with the binding capabilities, allows a fast implementation and an optimal performance as it discriminates the packets coming to a bind port by topic, avoiding any extra processing on the binding side.

[3] https://zeromq.org/

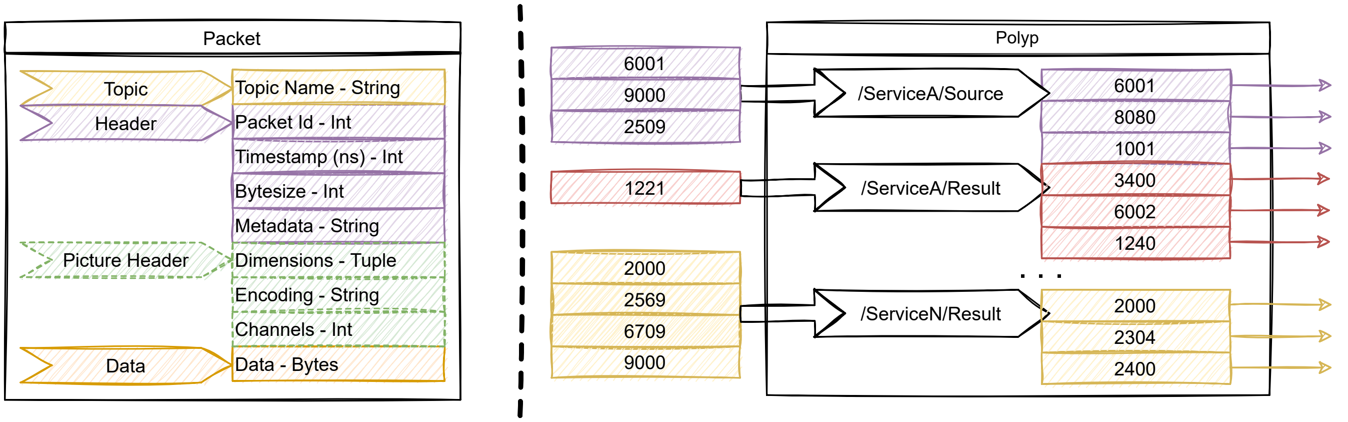

Figure 3: Left: Detailed structure of our custom packets. Right: Schematic simplified representation of Polyp's internal design.

We have implemented three types of publishers/subscribers, which use different kinds of custom packets. All of the packets are divided into three sub-packets: topic, header, and data. The topic is a simple string with the topic to which the socket is subscribed or publishing to. The header includes relevant metadata: packet id, timestamp, byte size, and a free field for other metadata, see Fig. 3 for a detailed structure of our packets. The header and data changes depending on the publisher/subscriber type:

- Picture: it allows to publish or subscribe to three or four channels color images. It implements JPEG encoding and decoding capabilities, and is able of transforming the image data to bytes, and reconstructing the image when received. The header in this case also adds the image metadata: height, width, channels, and codification format. The data sub-packet includes the image (encoded or not) bytes.
- Unsigned 8 Bits Picture: is specifically designed for semantic segmentation algorithms offloading, in which the result is a single channel 8 bits frame. The implementation also allows single channel JPEG encoding and decoding. The header in this case removes the channel information as it is no longer required.
- Metadata: this type is designed to transmit metadata such as configuration information, position and orientation updates or the result from algorithms such as object tracking which outputs just positions, orientations and sizes. The header in this case is the originally described one. The data in this case is sent as regular strings.

While ZeroMQ is originally design to block the main thread when receiving or sending data, we have built both the publishers and subscribers to allow both synchronous and asynchronous communication. In the synchronous mode, both the receiving and sending steps block the main process. In the asynchronous mode, we create an independent thread for each opened socket so that that receiving and sending steps do not block the main process. Besides, we have implemented and efficient callback system for the subscribers. Any custom method can be attached to such callback to handle the income messages at will.

2.2 Polyp: Implementation Details

Polyp has been designed to efficiently route traffic coming from a socket to another arbitrary socket. The main logic is implemented using ZeroMQ, as it provides sufficiently low latency with extremely low processing overhead [29]. Besides, the topic discrimination and binding capabilities already implemented by ZeroMQ facilitates the implementation of Polyp. Polyp binds to a set of arbitrary ports to which the nodes and services connect to receive or send data through an arbitrary topic, and the packets are discriminated by topic automatically by ZeroMQ.

The key component of Polyp is the mapping between the topics and destination ports. When a new service or node is added to the system, Polyp receives the relevant information: subscriber/publisher(s) topic(s) and the correspondent endpoint(s). Polyp then maps these topics with such ports so it only requires the topic information from the incoming packets to correctly route them, see Fig. 3 for an schematic representation of Polyp's inner processes.

3 EXPERIMENTS AND RESULTS

While architecture has been already tested in field VR offloading scenarios in wireless networks [13], we wanted to benchmark our architecture in different scenarios and wireless networks. Consequently, we designed our experimental setup to test a wide range of combination of scenarios and offloaded algorithms. With this experiments, we aim to understand the current limitations of our implementation and study how we and other researchers can move toward an even more optimal offloading solution.

We focused in three main offloading scenarios. In all the cases, the uplink feed is the sensor data, which we consider to be just a single camera feed. The main difference between the scenarios is the resulting data, which is sent back through the downlink stream to the device:

- Scenario A - Full offloading: in this scenario the donwlink side is composed by the rendered frame, which is sent back to the device. This is the most latency-critical scenario, as lost or late frames can produce nausea or discomfort to the user.

Table 1: Decomposition of the source and results frame size used in each experiment.

	A - Rendering Offloading				B - Segmentation Offloading				C - Metadata Offloading		
	UL		DL		UL		DL		UL		DL
	Pixels	Mbits	Pixels	Mbits	Pixels	Mbits	Pixels	Mbits	Pixels	Mbits	Mbits
R1	3840x2160	8.14	3840x2160	8.14	1920x1080	2.81	1920x1080	2.09	1920x1080	2.81	<0.001
R2	960x540	1.00	3840x2160	8.14	960x540	1.00	960x540	0.83	960x540	1.00	<0.001
R3	540x270	0.35	1920x1080	2.81	540x270	0.35	540x270	0.32	540x270	0.35	<0.001

- Scenario B - Real-time segmentation offloading: in this case, the downlink stream includes the individual frames resulting from the segmentation algorithm.
- Scenario C - Light downlink algorithms: some heavy algorithms, such as object tracking [27] or simultaneous localization and mapping (SLAM) [9], only output some metadata as the only results. This metadata is sent back to the device.

The last two scenarios are less restrictive in terms of latency if their implementation include any latency correction algorithm, allowing to have end to end latencies above the sampling rate period. Even though we plan to extend the functionality of the architecture to support other transport protocols, such as UDP and RTP which can be specially optimal for some of the offloading scenarios, we just focus on the current implementation which relies on TCP. Similarly, we consider offloading scenarios in which the frames, both the source (uplink frame sent from the client) and result frames (downlink frame sent from the offloading server), are sent individually. However, we plan to extend, as a short term future step, the presented results by testing other packetization and encoding/decoding schemes more optimal for video streaming, such as RTP in combination with High Efficiency Video Coding (HEVC) techniques. Eventhough current AR and VR devices already include time warping capabilities to reduce the effects or rendering delays or jitter, its still crucial to decrease latencies to the minimum, specially in scenario A. The greater this latencies are, the harder it is for the time warping algorithm to overcome its effect. If the offloading architecture adds extra latency, the correction effect of the time warping algorithm would decrease, so we aim for the architecture to not add extra frames of latency which could degrade the experience even if time warping is available.

We decided to test the selected scenarios in three different networks to understand how our architecture adapts to their particularities. Our goal is to understand the limitations of both our architecture and the network in each offloading scenario:

1. WiFi: we decided to test our architecture on a WiFi network as this technology is well stablished as the most used wireless network for indoor tasks. The outstanding performance of the newest releases, such as the release 802.11ax which allow throughputs way higher than 1 Gbps [4], allow user to be considered as a viable network for immersive media offloading. In our experiments we use a Netgear R6400 router.
2. Millimiter Wave (mmW) 5G Network: mmW technology is considered to become one of the key enablers of novel technologies, including VR and AR offloading, over the coming years. Consequently, we decided to test our architecture on a mmW prototype we have access to. The setup we used incorporates 8 subcarriers with 100 MHz of bandwidth each. It is configured with numerology 3 which corresponds to a carriers sub-spacing of 120 KHz. Only two of the available subcarriers are configured to allow uplink traffic, using a 1UL:4DL (1 uplink slot granted for every 4 downlink slots assigned) TDD configuration. We used the Askey RTL6305 mmW modem which, by the time we carried out the experiments, were not capable of aggregating the uplink subcarriers. Consequently, on the uplink side only 100 MHz of bandiwdth with TDD 1UL:DL were available. The experimental setup is using 256-QAM modulation. Millimiter wave communication is a recently introduced technology which is still in a very early stage, both from the modems and base station sides. However, we still considered relevant for the research ecosystem to test the architecture on our mmW setup as we could gain insights on how to optimize our architecture for the future of this groundbreaking telecommunication technology.
3. 5G-RAN Emulator: we decided to use our in-house developed 5G-RAN emulator[11] to test the network with different and currently not possible configurations. More specifically, mmW possible configurations are still limited, constraining the uplink scheduling grants. Besides, mmW commercial modems haven't reached their full potential yet. For this reason, we have decided to emulate in real-time a mmW base station with optimal configuration parameters. The used emulator captures IP packets using Netfilter Queues [24]. These IP packets are sent to the emulator in real-time, which queue them and models what happen with these packets (drop or release) and when. The emulator accurately models the RLC, PDCP, MAC and Physical Layers from a system perspective using the models and implementation details described in the specifications [2][1][3]. The main goal is to emulate, with a high level of accuracy, how our architecture performs on a mmW setup better configured and optimized for both uplink and downlink communication. Besides, the emulator allow to model other latencies such as the link latency between base station and a nearby MEC. The emulator runs on a Aorus Laptop with 16 GB of RAM and an Intel® Core™ i7-10870H CPU @ 2.20GHz × 16.

3.1 Architecture Setup

We decided to test an architecture setup in which a device is offloading an algorithm to a MEC. The MEC has several offloading services available, and an instance of Polyp runs on it, handling and routing the income data to and from the target service. Fig. 4 depicts the architecture setup and data flow we have used for the experiments.

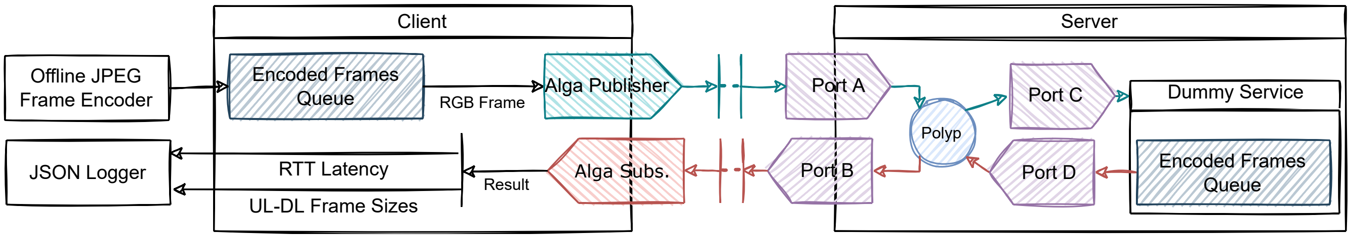

Figure 4: Detailed data flow of the used experimental setup.

Table 2: Summary of each network setup's Iperf performance.

1 - WiFi			2 - mmW			3 - 5G-RAN Emulator		
DL (Mbps)	UL (Mbps)	Ping (ms)	DL (Mbps)	UL (Mbps)	Ping (ms)	DL (Mbps)	UL (Mbps)	Ping (ms)
~200	~200	<4	~3000	>80	<12	>3500	>1500	6

Even though we are aware of the importance of the selected packetization schemes or video encoding/decoding techniques to the overall behavior of the architecture, our main goal for the proposed experiments is to benchmark the throughput and latency capabilities of our implementation. Consequently, we decide to send the frames individually and remove all the processing overhead non related to the architecture itself, such as the encoding and decoding steps. Therefore, we used the same 4K ($3840x2160$) ten seconds video for all the experiments. We resized each frame of the video to resolutions 2, 4 and 8 times smaller than the original, for a total of 4 videos. Then, we encoded each frame of each video to JPEG and store them in memory. These files are used both in the server and client to load the frames that are sent each time. The same process is done for the frames corresponding to the resultant segmentation mask in the offloading scenario B: each frame is offline masked, according to a random color mask, and encoded to single channel JPEG. The process is repeated for the 4 chosen resolutions.

While the architecture is also prepared to receive and send packets asynchronously, to isolate and fairly evaluate each individual frames transmission, the data flow is synchronous (see Fig. 4) and follows the next scheme:

- When the result from the previously transmitted frame is received, the client takes an encoded frame from the stored files and send it through its publisher to an arbitrary port.
- Polyp, running in the same machine as the server, receives the packet and retransmits it to the receiving port to which the target service is attached.
- The target service receives the packet on the subscriber. In this case, we use a dummy service which just discards the packet and immediately loads an encoded frame from the previously created files and send it back to the client. If we are on the offloading scenario C, only random metadata of fixed size (360 bytes) is sent back.
- Polyp re-routes the received reply from the server to the correspondent receiving port on the client side.
- The client receives the reply, discards it, log the round trip time and uplink and downlink frame sizes, and re-initializes the loop.

3.2 Polyp Evaluation

The first step was to evaluate the overhead introduced by Polyp's packet routing. As we consider Polyp to be running in the MEC which is offering different offloading service, we can assume Polyp's routing is done on the local host network. Consequently, we decided to run the the entire proposed experimental pipeline in the same computer for different source and result frames and trace the overhead latencies added by Polyp. We repeated the experiments 4 times, one for each resolution. In all the 4 rounds of experiments, the time overhead mean was in all cases smaller than 1 ms.

After this simple experiment we can conclude that in this type of setup, in which Polyp is running in the same machine as the offloading services, there is no relevant time overhead added by Polyp. Therefore, in the following experiments we do not use Polyp to route the packets to the dummy server, and we directly connect or bind the server's and client's publishers and subscribers.

3.3 Alga Results

We didn't evaluate all the possible combinations of source and results frames. On the contrary, we chose only the most representative combinations, shown in Table 1. The sizes shown in Table 1 are the mean sizes of all the JPEG-encoded frames with the given resolution. As the in scenario B the downlink frames are single channel JPEG-encoded masks, we can observe their sizes to be smaller than the ones with same resolutions in the other scenarios. We have three different resolution setups for each scenario. Table 1 defines the IDs for each experiment, being R1, R2 and R3 the 3 resolution combinations for each offloading scenario, A, B or C.

We also analyzed the base performance of both the WiFi and mmW setups. We used iperf [4] to estimate the TCP throughput capabilities. We let the test run for $100s$ in each setup, with no other users connected. In the emulator case, we assumed a one way latency between the core and the MEC of $3ms$. The emulator is configured exactly as the actual mmW setup, but assuming the 1UL:4DL TDD configuration is extended along the 8 subcarriers, and the modem is capable of performing carrier aggregation. This

[4] https://iperf.fr/

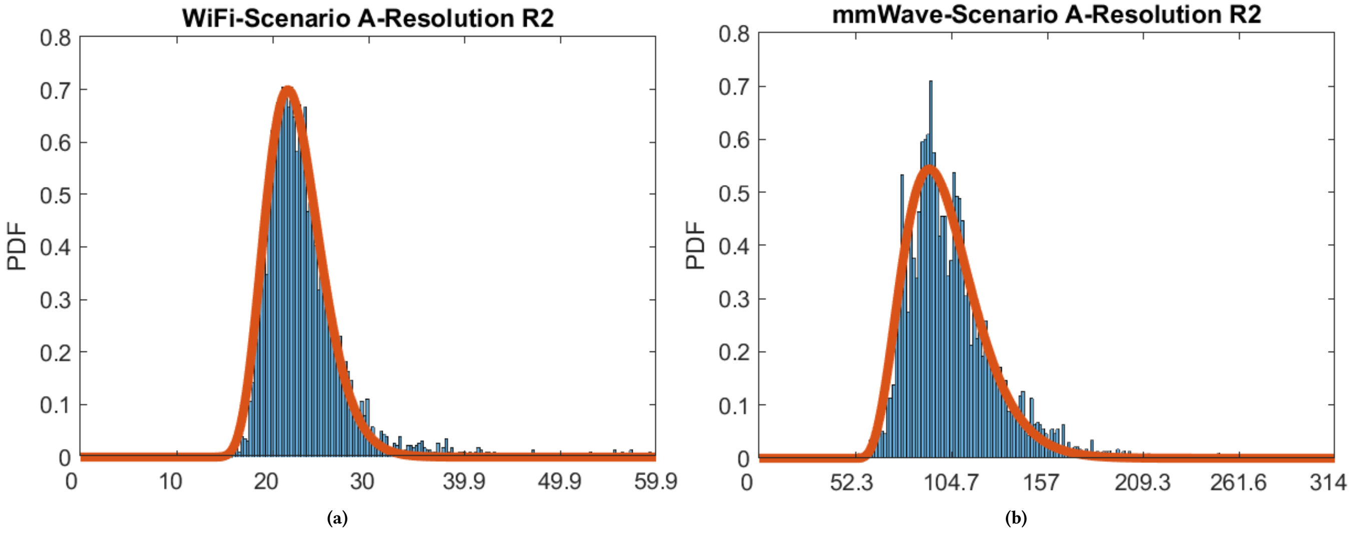

Figure 5: Example of the estimated pdf (in orange) for the obtained round trip times histograms in milliseconds (in blue) for two arbitrarily chosen examples: Scenario A and resolution R2 for both WiFi (a) and the mmW base station setup (b).

configuration multiplies by 8 the actual uplink throughut capabilities of the actual mmW experimental setup. The simulated modem is placed 50 meters away from the base station. This configuration is used in all the emulated experiments. The summary of the achieved TCP performance on Iperf in the three cases is shown in Table 2. The ping values measurements were taken with empty buffers.

We are using the same set frames along the duration of each experiment. As a consequence, we only need to focus on the measured round trip time (RTT) latency on the application layer. For each experiment, we transmitted both ways a total of 10000 frames through the architecture. In each iteration, the round trip time was measured and saved for later analysis. We decided to obtained the RTT latencies' probability density functions for each experiment, giving a statistical view of the RTT performance of our architecture.

By a quick inspection, our intuition indicated that the PDF followed a Gamma distribution [30] with a certain time offset. This intuition matches the fact that any delay in a communication network is defined by a minimum value (the offset), which is caused by physical and computing limitation factors, with a greater occurrence in the delays close to the offset and some values with a much lower occurrence in higher delays. Besides this, our intuition was supported by a study on the prediction of RTT for wireless networks [32]. Below, you can find the definition of the offset Gamma distribution:

$$f_X(x, x_i, \alpha, \theta) = \frac{(x - x_i)^{\alpha-1} e^{-(x-x_i)/\theta}}{\theta^\alpha \Gamma(\alpha)}, \text{with } \Gamma(\alpha) = \int_0^\infty \frac{t^{\alpha-1}}{e^t} dt, \tag{1}$$

with α a shape parameter, θ the scale parameter, β the inverse of the scale parameter, and x_i the offset. These are the parameters that need to be adjusted given the input data. We found that 10000 iterations was enough to accurately adjust an offset Gamma PDF, for each experiment, from the RTTs histogram. To adjust the pdf, we defined an optimization function of the following type:

$$\min_{x_i,\alpha,\theta} \sum_x |f_X(x, x_i, \alpha, \theta) - h_X(x, s)|, \tag{2}$$

where $h_X(x, s)$ is the histogram adjusted to the x axis of the PDF and the experiment s. The optimization problem was solved using Matlab. Fig. 5 shows the obtained RTT histograms and their estimated PDFs from two of the presented experiments. We can observe a great level of agreement between the theoretical PDF and the histogram after solving the optimization problem defined in Eq. 2.

Apart from this, the mean and variance of the RTT have been calculated, since these statistic parameters are the simplest that characterize any distribution, and can be applied with a good agreement for Gamma distributions. The obtained RTT metrics for each resolution and scenario combination and wireless technology experiment is shown in Table 3. Besides, and to show when most of the frames arrives on time in each scenario and wireless technology, we estimated the 95th percentiles shown also in Table 3. Aiming to give a visual insight of the results we built Fig. 6 which depicts the bar plot of the mean and variance RTT from each experiment. Notice that we have added two thresholds in Fig. 6: one is delimiting the maximum RTTs which support hard real-time and the other referring to the soft real-time deadline. We understand hard real-time as the process in which the total end to end latency, including the processing overhead, is smaller than the frame update period (16.6ms). We chose this deadline to be 8ms according to the processing overhead times assumed in [23]. In specific VR offloading applications, as the one described in [13], higher RTTs (<50ms) provides a sufficient quality of experience, as buffering techniques can overcome the delays effects. Consequently, we consider 32 ms as a our reference soft real-time deadline for this particular less constraint offloading applications.

Table 3: Summary of the estimated pdf values for each experiment: rendering (A), segmentation (B) and metadata (C) offloading; and high (R1), medium (R2), and low (R3) frame resolution. See Table 1 for details.

	WiFi								
	A-1	A-2	A-3	B-1	B-2	B-3	C-1	C-2	C-3
Mean (ms)	36,43	23,54	10,58	25,36	11,61	3,98	8,43	4,23	2,52
95th Percentile	43,35	31,21	13,52	40,92	19,84	5,50	10,47	5,56	3,68
Std (ms)	4,52	5,33	1,86	8,97	4,61	0,97	1,32	1,02	0,86
Offset (ms)	23,75	12,43	7,17	9,63	5,32	2,61	5,88	2,86	1,78
Shape	9,8	11,46	5,96	3,93	2,49	2,56	4,81	3,69	1,09
Scale	0,24	0,17	0,34	0,52	0,74	0,5	0,32	0,18	0,39
	mmWave								
	A-1	A-2	A-3	B-1	B-2	B-3	C-1	C-2	C-3
Mean (ms)	190,53	105,83	51.05	58,22	27,11	19,07	42,80	22,39	17,95
95th Percentile	259,82	153,00	70,62	82,05	36,47	27,46	61,80	27,81	25,36
Std (ms)	41,8	31,04	18,24	35,87	8,12	6,7	11,91	6,35	4,55
Offset (ms)	102,62	51,45	32,72	32,98	21,42	16,85	22,74	8,41	14,98
Shape	4,96	5,79	6,9	2,55	1,1	0,24	3,88	6,66	0,86
Scale	0,5	0,33	0,09	0,23	0,25	1,51	0,38	0,16	0,41
	5G-RAN Emulator								
	A-R1	A-R2	A-R3	B-R1	B-R2	B-R3	C-R1	C-R2	C-R3
Mean (ms)	16,14	10,76	8,67	10,99	8,57	8,51	9,88	8,54	8,5
95th Percentile	17.92	11,69	9,48	12,72	9,44	8,65	10,62	9,41	8,65
Std (ms)	3,14	1,34	0,96	1,49	0,96	0,43	0,82	0,38	0,34
Offset (ms)	14,67	10,38	8,37	10,15	8,34	8,37	9,21	8,32	8,35
Shape	0,96	0,71	1,59	0,59	0,51	1,56	0,82	1,99	1,98
Scale	0,13	0,07	0,02	0,34	0,04	0,06	0,21	0,07	0,09

We can directly observe that the performance on the actual mmWave setup is the poorest one. We were expecting this behaviour as the mmW technology, and specially the experimental setup we have access too, is still improving, with almost no commercial roll-outs worldwide. The available mmW modems are still very limited on the uplink side: the Askey RTL6305 is not performing any succesful carrier agreggation on the uplink. Consequently, the uplink effective bandwidth is limited to less than 30 MHz, considerably reducing the effective throughput. Besides, the fact that our architecture is based on TCP reduces the network exploitation on poorly performing networks: delay and throughput are competing resources in TCP. However we can observe that in some scenarios, the soft real-time deadlines are achieved. Consequently, for some of the proposed scenarios, the current sub-optimal development stage of mmW technologies is already useful for particular immersive offloading tasks.

Our WiFi experimental setup considerably favors our TCP-based architecture as the measured baseline ping end to end latencies were smaller than 4 ms. Consequently, all the experiments but the most demanding in terms of throughput (A-R1) were meeting the soft real-time deadline. Besides, only three experiments showed deadlines above the hard real-time requirements (A-R1, A-R2, B-R1). These limitations could be overcome using a router which implements MIMO or multilink transmission to provide even higher throughputs.

Finally, as we were expecting a priori, the best performance is given by the emulated mmW setup. First, is key to acknowledge that in this setup, the entire architecture pipeline was running on the same machine, removing any possible network degradation that could be produced by any of the network components involved. Besides, the mmW configuration used, which assumed the use of modems capable of performing carrier aggregation, enables almost 200 MHz just for uplink transmission. This allows more than 8 times uplink throughput than in the actual mmW setup. We can observe that only the first scenario and resolution combination (A-R1) is not

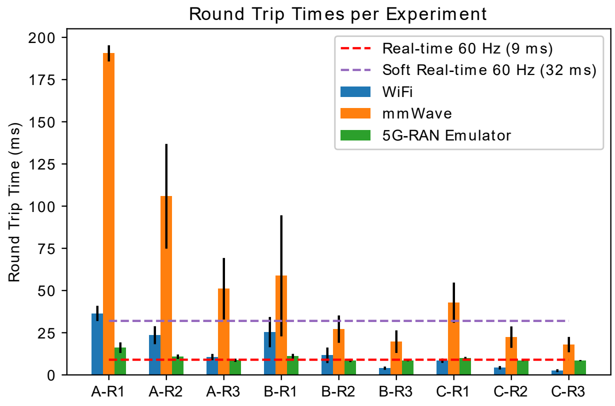

Figure 6: Graphical summary of the round trip time results from all the experiments.

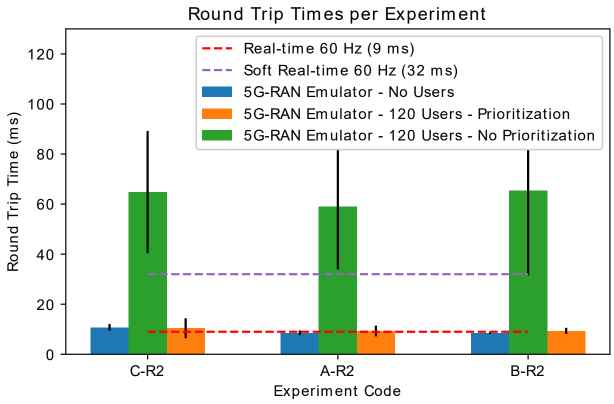

Figure 7: Graphical summary of the round trip time results from all the experiments.

fulfilling the hard real-time requirements. However, we can observe that the mean RTTs are not going below 8-7 ms. This is due to two facts: we modeled a fixed simulated core to MEC one-way latency of 3 ms, and TDD scheduling add extra non-avoidable latencies, specially on the uplink side. This last limitation could be overcome using network slicing which allows greater scheduling flexibility and use case specific optimizations.

3.4 Emulated mmW-based Offloading on a Realistic Scenario

To goal of this last set of experiments is to test our architecture in a more realistic scenario using the 5G-RAN emulator: we simulated multiple users attached to the same base station which are sharing network resources with the virtual immersive user. The goal is to emulate an actual mmW deployment and understand how the performance degrades as other users are connected to the same base station. We included 120 virtual users in the simulator which are consuming 5 Mbps and 1 Mbps of downlink and uplink traffic respectively. We chose these values as they represent the case in which these virtual users are on a video-call, receiving and sending HD 1080 and SD 480 video streams respectively. This number of users was selected as this combination of users and uplink and downlink throughputs sufficiently degrades the overall performance of the offloading architecture. The users are placed randomly within a 1000m radius from the base station. Besides, the emulator has user prioritization capabilities, handled by the emulator's scheduler implementation. Therefore, we tested the scenario with no user prioritization against the one with it. We used proportional fair [16], a well studied solution, as the ruling scheduling algorithm. As we already have the previous results as a baseline, we decided to focus only on the most relevant resolutions for each offloading scenario: R2 in all the offloading scenarios A, B, C from Table 1. We choose this uplink frames resolution (960x540) as it is, along with 1080x720, a typical resolution a sufficiently high resolution for relevant VR offloading applications as the one proposed in [13].

The obtained results are depicted in Fig. 4. We can observe the high performance degradation when other users are consuming resources from the base station and no user prioritization is used: measured RTTs grow 6 times bigger than in the no users setups. However we can observe that there is almost no visual RTT values differences between the user-less and prioritization setups. We can, on the contrary, observe that the standard deviation is slightly higher for the scenario with simulated users. This is justified by the fact that, even though the user prioritization is forcing the scheduler to grant transmission slots to the prioritized user, proportional fair schedulling algorithm guarantees periodic serving to the other users. This produces small peaks of latency on the prioritize user, which, as we can observe in Fig. 7, are small enough to be neglected.

4 CONCLUSIONS

In this paper, we have presented our optimized, TCP-based immersive media offloading architecture. First, we have described the main characteristics and implementation details of the proposed architecture. Besides, we have described the main experiments that were performed to test our implementation. The architecture was tested on both WiFi and a 5G mmW wave setup. As the current development of mmW wave has still not reach its full potential, we have also tested our network on a 5G emulator which is able of modeling a configurable mmW base station. The goal of this last experiment is to test the potential of our offloading architecture on a fully functional mmW setup.

The architecture was tested on different offloading scenarios, frame resolutions, and the three wireless networks setups: WiFI, mmW, and emulated mmW. The results where focused on measuring the round trip time at the application level for each scenario and wireless technology. The obtained results showed a high performance of our architecture in WiFi, meeting at least the soft real time deadline for every test case but one. On the contrary, mmW results were not as successful: the test cases with highest input or output resolutions did not satisfy the soft real time requirements. This was expected as the current development of mmW technologies is currently limited, considerably constraining the uplink throughput capabilities. However, even with the current limitations, mmW can already be used in several offloading use cases according to the results. Finally, using the emulated mmW setup, configured for exploiting the actual throughput capabilities, we managed to obtain outstanding results with our offloading architecture. The only test case in which the emulated mmW setup could not perform in hard real-time is the scenario in which both the uplink and downlink frames' resolution was 4K. From these results, we conclude that the scenario A, in which the rendered immersive scene is sent back to the device, which is unavoidably a hard real-time offloading use case, is not well suited for TCP communication protocol. In this case, other video streaming solutions based in UDP, such as RTP, are probably more suitable. We also plan to explore more recent protocols such as QUIC. With this UDP or derived protocols scenario, we are considering incorporating other multimedia coding standards such as High Efficiency Video Coding (HEVC). Furthermore, we tested our architecture in a realistic mmW rollout, using the 5G mmW emulator. We added multiple virtual users to the emulated scenario, consuming throughput resources from the base station. When no user prioritization was used, the network degradation was such that the measured RTT increased up to 8 times compared to the no user scenario. We performed the same experiments using user prioritization, showing almost an identical performance as the scenario with no users. The experiments with the mmW emulated setup has shown the potential of both mmW technologies and the proposed offloading architecture. The combination of these technologies, and the newest releases of WiFI, can become a key technology enabler for the future of immersive media technologies. These results already give a detailed overview of the throughput and latency capabilities of the proposed architecture for the particular application of immersive media offloading. However, we plan to extend the presented experiments and results in the short term by testing how the selected packetization schemes and encoding/decoding approach affect the overall behaviour of the architecture, both from a quantitative and quality of the immersive experience point of views.

ACKNOWLEDGMENTS

This work has received funding from the European Union (EU) Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie ETN TeamUp5G, grant agreement No. 813391.

REFERENCES

[1] 3GPP. 2022. *NR; Physical channels and modulation.* Technical Specification (TS) 36.211. 3rd Generation Partnership Project (3GPP). https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3213 Version 17.0.0.

[2] 3GPP. 2022. *NR; Physical layer procedures for data.* Technical Specification (TS) 36.214. 3rd Generation Partnership Project (3GPP). https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3216 Version 17.0.0.

[3] 3GPP. 2022. *Study on channel model for frequencies from 0.5 to 100 GHz.* Technical Specification (TS) 36.901. 3rd Generation Partnership Project (3GPP). https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3173 Version 16.1.0.

[4] Muhammad Shahwaiz Afaqui, Eduard Garcia Villegas, and Elena López-Aguilera. 2017. IEEE 802.11ax: Challenges and Requirements for Future High Efficiency WiFi. *IEEE Wireless Communications* 24 (2017), 130–137. https://doi.org/10.1109/MWC.2016.1600089WC

[5] Ahmed A. Al-Habob, Ahmed Ibrahim, Octavia A. Dobre, and Ana García Armada. 2020. Collision-Free Sequential Task Offloading for Mobile Edge Computing. *IEEE Communications Letters* 24, 1 (2020), 71–75. https://doi.org/10.1109/LCOMM.2019.2948179

[6] Roberto Baldoni, Mariangela Contenti, and Antonino Virgillito. 2003. *The Evolution of Publish/Subscribe Communication Systems.* Springer-Verlag, Berlin, Heidelberg, 137–141. https://doi.org/10.5555/1809315.1809344

[7] Min Chen and Yixue Hao. 2018. Task Offloading for Mobile Edge Computing in Software Defined Ultra-Dense Network. *IEEE Journal on Selected Areas in Communications* 36, 3 (2018), 587–597. https://doi.org/10.1109/JSAC.2018.2815360

[8] Angela Dai, Matthias Nießner, Michael Zollhöfer, Shahram Izadi, and Christian Theobalt. 2017. BundleFusion: Real-Time Globally Consistent 3D Reconstruction Using On-the-Fly Surface Reintegration. *ACM Trans. Graph.* 36, 3, Article 24 (may 2017), 18 pages. https://doi.org/10.1145/3054739

[9] Jeffrey Delmerico and Davide Scaramuzza. 2018. A Benchmark Comparison of Monocular Visual-Inertial Odometry Algorithms for Flying Robots. In *2018 IEEE International Conference on Robotics and Automation (ICRA).* IEEE, Brisbane, Australia, 2502–2509. https://doi.org/10.1109/ICRA.2018.8460664

[10] Mohammed S. Elbamby, Cristina Perfecto, Mehdi Bennis, and Klaus Doppler. 2018. Toward Low-Latency and Ultra-Reliable Virtual Reality. *IEEE Network* 32, 2 (2018), 78–84. https://doi.org/10.1109/MNET.2018.1700268

[11] Diego Gonzalez Morin, Manuel J. Lopez Morales, Ana Perez, Pablo Garcia Armada, and Alvaro Villegas. 2022. FikoRE: 5G and Beyond RAN Emulator for Application Level Experimentation and Prototyping. http://arxiv.org/abs/2204.04290

[12] Ester Gonzalez-Sosa, Pablo Pérez, Ruben Tolosana, Redouane Kachach, and Alvaro Villegas. 2020. Enhanced Self-Perception in Mixed Reality: Egocentric Arm Segmentation and Database With Automatic Labeling. *IEEE Access* 8 (2020), 146887–146900. https://doi.org/10.1109/ACCESS.2020.3013016

[13] Diego González Morín, Ester Gonzalez-Sosa, Pablo Pérez, and Alvaro Villegas. 2022. Bringing Real Body as Self-Avatar into Mixed Reality: A Gamified Volcano Experience. In *2022 IEEE Virtual Reality (VR).* IEEE, Online, 3–10.

[14] Jinki Jung, Jaewon Ha, Sang-Wook Lee, Francisco A. Rojas, and Hyun S. Yang. 2012. Novel Applications of VR: Efficient Mobile AR Technology Using Scalable Recognition and Tracking Based on Server-Client Model. *Comput. Graph.* 36, 3 (may 2012), 131–139. https://doi.org/10.1016/j.cag.2012.01.004

[15] Konstantina Kilteni, Raphaela Groten, and Mel Slater. 2012. The Sense of Embodiment in Virtual Reality. *Presence: Teleoperators and Virtual Environments* 21, 4 (2012), 373–387. https://doi.org/10.1162/PRES_a_00124

[16] Raymond Kwan, Cyril Leung, and Jie Zhang. 2009. Proportional fair multiuser scheduling in LTE. *IEEE Signal Processing Letters* 16, 6 (2009), 461–464. https://doi.org/10.1109/LSP.2009.2016449

[17] Ji Li, Hui Gao, Tiejun Lv, and Yueming Lu. 2018. Deep reinforcement learning based computation offloading and resource allocation for MEC. In *2018 IEEE Wireless Communications and Networking Conference (WCNC).* IEEE, Barcelona, Spain, 1–6. https://doi.org/10.1109/WCNC.2018.8377343

[18] Zhiyuan Li, Cheng Wang, and Rong Xu. 2001. Computation Offloading to Save Energy on Handheld Devices: A Partition Scheme. In *Proceedings of the 2001 International Conference on Compilers, Architecture, and Synthesis for Embedded Systems* (Atlanta, Georgia, USA) *(CASES '01).* Association for Computing Machinery, New York, NY, USA, 238–246. https://doi.org/10.1145/502217.502257

[19] Chen-Feng Liu, Mehdi Bennis, and H. Vincent Poor. 2017. Latency and Reliability-Aware Task Offloading and Resource Allocation for Mobile Edge Computing. In *2017 IEEE Globecom Workshops (GC Wkshps).* IEEE, Singapore, 1–7. https://doi.org/10.1109/GLOCOMW.2017.8269175

[20] John McCarthy. 2014. Multi-image photogrammetry as a practical tool for cultural heritage survey and community engagement. *Journal of Archaeological Science* 43 (03 2014). https://doi.org/10.1016/j.jas.2014.01.010

[21] Erin A. McManus, Bobby Bodenheimer, Stephan Streuber, Stephan de la Rosa, Heinrich H. Bülthoff, and Betty J. Mohler. 2011. The Influence of Avatar (Self and Character) Animations on Distance Estimation, Object Interaction and Locomotion in Immersive Virtual Environments. In *Proceedings of the ACM SIGGRAPH Symposium on Applied Perception in Graphics and Visualization* (Toulouse, France) *(APGV '11).* Association for Computing Machinery, New York, NY, USA, 37–44. https://doi.org/10.1145/2077451.2077458

[22] Diego González Morín, Ana García Armada, and Pablo Pérez. 2020. Cutting the Cord: Key Performance Indicators for the Future of Wireless Virtual Reality Applications. In *2020 12th International Symposium on Communication Systems, Networks and Digital Signal Processing (CSNDSP).* IEEE, Online, 1–6. https://doi.org/10.1109/CSNDSP49049.2020.9249445

[23] Diego González Morín, Pablo Pérez, and Ana García Armada. 2022. Toward the Distributed Implementation of Immersive Augmented Reality Architectures on 5G Networks. *IEEE Communications Magazine* 60, 2 (2022), 46–52. https://doi.org/10.1109/MCOM.001.2100225

[24] Netfilter. 2022. Netfilter Queues. https://netfilter.org/projects/libnetfilter_queue/

[25] Zhaolong Ning, Peiran Dong, Xiaojie Wang, Mohammad S. Obaidat, Xiping Hu, Lei Guo, Yi Guo, Jun Huang, Bin Hu, and Ye Li. 2020. When Deep Reinforcement Learning Meets 5G-Enabled Vehicular Networks: A Distributed Offloading Framework for Traffic Big Data. *IEEE Transactions on Industrial Informatics* 16, 2 (2020), 1352–1361. https://doi.org/10.1109/TII.2019.2937079

[26] Jihong Park and Mehdi Bennis. 2018. URLLC-eMBB Slicing to Support VR Multimodal Perceptions over Wireless Cellular Systems. In *2018 IEEE Global Communications Conference (GLOBECOM).* IEEE, Singapore, 1–7. https://doi.org/10.1109/GLOCOM.2018.8647208

[27] Jonathan Pedoeem and Rachel Huang. 2018. YOLO-LITE: A Real-Time Object Detection Algorithm Optimized for Non-GPU Computers. https://doi.org/10.48550/ARXIV.1811.05588

[28] Yushan Siriwardhana, Pawani Porambage, Madhusanka Liyanage, and Mika Ylianttila. 2021. A Survey on Mobile Augmented Reality With 5G Mobile Edge Computing: Architectures, Applications, and Technical Aspects. *IEEE Communications Surveys Tutorials* 23, 2 (2021), 1160–1192. https://doi.org/10.1109/COMST.2021.3061981

[29] P. Sommer, F. Schellroth, M. Fischer, and J. Schlechtendahl. 2018. Message-oriented Middleware for Industrial Production Systems. In *2018 IEEE 14th International Conference on Automation Science and Engineering (CASE).* IEEE, Munich, Germany, 1217–1223. https://doi.org/10.1109/COASE.2018.8560493

[30] H. C. S. Thom. 1958. A Note On The Gamma Distribution. *Monthly Weather Review* 86, 4 (1958), 117 – 122. https://doi.org/10.1175/1520-0493(1958)086<0117:ANOTGD>2.0.CO;2

[31] Alvaro Villegas, Pablo Pérez, and Ester González-Sosa. 2019. Towards a Distributed Reality: A Multi-Video Approach to XR. In *Proceedings of the 11th ACM Workshop on Immersive Mixed and Virtual Environment Systems* (Amherst, Massachusetts) *(MMVE '19).* Association for Computing Machinery, New York, NY, USA, 34–36. https://doi.org/10.1145/3304113.3326111

[32] Shinya Yasuda and Hiroshi Yoshida. 2018. Prediction of round trip delay for wireless networks by a two-state model. In *2018 IEEE Wireless Communications and Networking Conference (WCNC).* IEEE, Barcelona, Spain, 1–6. https://doi.org/10.1109/WCNC.2018.8377039

[33] Hengshuang Zhao, Xiaojuan Qi, Xiaoyong Shen, Jianping Shi, and Jiaya Jia. 2018. ICNet for Real-Time Semantic Segmentation on High-Resolution Images. In

Computer Vision – ECCV 2018, Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss (Eds.). Springer International Publishing, Cham, 418–434. https://doi.org/10.1007/978-3-030-01219-9_25

[34] Feng Zheng, Turner Whitted, Anselmo Lastra, Peter Lincoln, Andrei State, Andrew Maimone, and Henry Fuchs. 2014. Minimizing latency for augmented reality displays: Frames considered harmful. In *2014 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. IEEE, Munich, Germany, 195–200. https://doi.org/10.1109/ISMAR.2014.6948427