

Hybrid Edge-Cloud Computational Offloading for XR Medical Applications

Daria Alekseeva, *Member, IEEE*, Aleksandr Ometov, *Senior Member, IEEE*
Tampere University, Finland

Corresponding author's email: daria.alekseeva@tuni.fi

Abstract—Emerging Extended Reality (XR) applications, particularly in eHealth, offer new opportunities in digital healthcare, such as XR-assisted surgery. Nevertheless, the XR use case is set to extremely high standards to ensure safety, high quality of the medical service, and high user experience. Hence, workload-intense and latency-hungry XR applications force researchers to find ways to process data efficiently. Even though local processing advances data safety because no data is shared with the third-party device, it demands some computational capabilities, directly affecting the battery. Since Mobile Cloud Computing (MCC) or Mobile Edge Computing (MEC) provides a computationally rich server, the proposed hybrid model allows for controlling decision strategies and managing safety and response time according to the task. A hybrid offloading strategy decreases system latency by 77% compared to MCC, 60% improvements to local processing, and 11% enhancement to MEC offloading. The proposed hybrid system reduces the delay by providing computational resources closer to users but can be strained under high workloads.

Index Terms—MEC; MCC; Extended Reality; Remote Surgery; Video transmission; Traffic analysis

I. INTRODUCTION

Half a century ago, imagining to have a video call was still in fiction books. Ten years ago, advancements in wireless technology reached the level of high-reliable communications and widespread broadband access led to a boom in real-time presence applications [1]. Today, almost everyone carries this capability in their pocket or on their wrist, going beyond what mid-20th century science fiction imagined.

Modern communication systems of today, e.g., 5G, in turn, represent a mix of new technologies and improvements to core components from previous generations of wireless networks. The expected Key Performance Indicators (KPIs) for future 6G systems, such as data rates, spectral efficiency, coverage, energy efficiency, latency, and reliability, are set by strict performance standards, driven by the potential for new and innovative uses. Extended Reality (XR) experiences are among these potential applications, and they significantly influence the demands on future network infrastructures [2].

Those XR paradigms include a range of virtual-content technologies, such as Augmented Reality (AR), Virtual Reality (VR), and Mixed Reality (MR) [3]. Each of these technologies has unique features. AR allows real-time interaction with digital elements overlaid on the physical environment, letting users engage with augmented interfaces and control connected devices, creating immersive and dynamic experiences. VR creates a completely immersive digital environment that

simulates physical presence, enabling immersive storytelling experiences. MR systems can understand and map real-world surroundings in real-time, allowing virtual entities to interact with and adapt to the physical environment. This field also spans over emergency and medical domains [4] tremendously through such promising scenarios as Remote Surgery or Video Presence, that are already being standardized [5].

With 3GPP standardization advancements in mind, Fig. 1 depicts the chain of XR-Assisted Surgery video delivery. It could be read as follows: the surgeon, patient, and medical staff prepare for the procedure despite being separated by geographical distances. Equipped with a Head-Mounted Display (HMD) and controllers, the surgeon experiences full immersion in the remote surgery room. At the same time, cameras capture 8K live-stream video and transmit it directly to a remote server via the network. Simultaneously, the surgeon's console sends control messages to robotic surgery equipment through the communication link, ensuring seamless coordination.

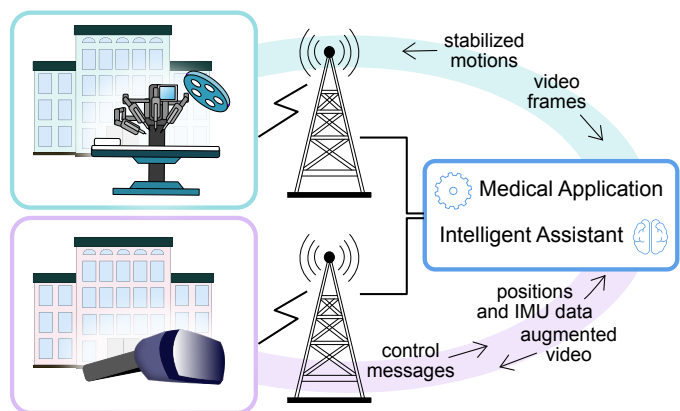


Fig. 1: Video delivery chain and architecture for the XR-assisted surgery use case

The Medical Application (MA), deployed on an Edge, i.e., Mobile Edge Computing (MEC), or Cloud, i.e., Mobile Cloud Computing (MCC), server and connected via wireless link, processes video streams with ultra-low latency, provides feedback based on surgeon tools positions and Inertial Measurement Unit (IMU) data, and offers haptic feedback to enhance the immersive experience. Machine Learning (ML)-enabled functionalities filter motion

control commands to stabilize surgery, ensuring precise and controlled movements even in remote settings. However, from an engineering standpoint, the video flow involves a multi-step process, including image capture, processing, encoding, transfer, decoding, and display, requiring meticulous management to ensure seamless transmission and optimal user experience [6]. Those bring a completely new set of requirements to be considered during the real system implementation.

The system's response time induces delays at each relay, reducing seamless playback quality. This paper proposes a MEC–MCC hybrid offloading model tailored for XR video streaming. The proposed strategy aims to minimize the system's response time while avoiding a significant increase in power consumption.

The rest of the paper is organized as follows. Section II collects performance requirements for immersive applications. Section III provides the system model overview. Further, Section IV provides results related to the latency-critical operation, as well as proposes a hybrid model for packet processing. The last section concludes the paper.

II. PERFORMANCE REQUIREMENTS

The present section collects the main technical recommendations for XR services in casual and medical domains. It briefly overviews normative documents declared by standards development organizations.

A set of recommendations has been applied to the XR systems to enable immersive and pleasant communication between users without irritations, interruptions, or malaises. Active VR users have noticed the sickness caused by delays between vision and head movements when watching or playing in the HMD. To handle motion sickness, the recommendation to motion-to-photon latency lays around 10 ms, being aware that some humans are more sensitive to this kind of delay [6].

Another way to handle motion sickness is by increasing the filming rate (frames per second), simultaneously ensuring smooth and seamless playback for the user. The higher fps rates fit more precisely with the vestibular-ocular reflex's latency requirements. To escape the motion sickness, the XR application in the medical domain demands up to 120 fps video service and ultra-low end-to-end latency [7]. The HMDs currently available on the market supports up to 90 fps.

XR recommends transmitting the uncompressed video frames, dramatically increasing the system load. Nowadays, existing codecs, i.e., HEVC (H.265), transmit video with the rate of up to 16 Mbps for 4K Ultra High Definition (UHD), up to 80 Mbps for 8K [8]. Since there is a need to design new codecs to apply to the High Definition (HD) 360 VR stream, the estimated bit rate expected to reach the 30 and 120 Gbps for the uncompressed 4K and 8K, respectively.

Regarding power consumption requirements, 3GPP suggests sensing and display units consume less than 1 W. There are no specific requirements for processors and communication units' consumption as they highly depend on the receiving load (e.g.,

a high-resolution image to render), the distance to the access point, and bit rates [9].

Recently emerged computing paradigms offer vast computing capacity, which opens new capabilities to medical applications. The offloading strategies for medical cases is a promising research direction because the predicted growth of the XR-enabled scenarios need enormous computing power, mass storage of medical data, and trust sharing of medical information, which a cloud platform could provide. Nevertheless, running remote XR-assisted surgery requires fulfilling the latency and reliability parameters by, for example, developing new offloading approaches.

III. SYSTEM MODEL

This section outlines the system model for the XR video stream. It starts with the model for packet generation, communication model, rendering model, and offloading models. The hybrid offloading model aims to minimize system delays.

A. Data generation

Assume, that the camera captures images with the resolution $m \cdot n$ and γ frame per second is a filming rate. Also assume that 1 pixel is equal to q bits. Therefore, the video bitrate is $V = m \cdot n \cdot q \cdot \gamma$ bits per second and the system workload is $w = V \cdot t$, where t is a simulation time. For transporting, the workload is divided into packets, assuming that the Maximum Transmission Unit (MTU) size limits the packet size. This leads to the delay on packet's on arrivals, notated as λ .

B. Communication model

Communication model for cellular wireless connection is presented using Nyquist model under idealized conditions:

$$C^{cell} = 2 \cdot B \cdot \log M \cdot h(t), \quad (1)$$

where B is a bandwidth, MHz; M is a modulation; and $h(t)$ is a channel condition during time t , where $h(t) \in H = \{M_1, \dots, M_N\}$. The future channel condition state is presented as a discrete stochastic process with a finite number of modulation states from Quadrature Phase Shift Keying (QPSK) to Quadrature Amplitude Modulation (QAM)–256, i.e., by applying Finite State Markov Channel (FSMC).

To define FSMC, assume that there are two types of channel – the slow-varying and rapid-varying channel. A slow-varying channel assumes that the modulation on the next state remains the same with higher probability, then it will change. Transition matrix for slow varying channel is:

$$\tau^{slow} = \begin{bmatrix} 0.8 & 0.1 & 0.1 & 0 \\ 0.1 & 0.8 & 0.1 & 0 \\ 0 & 0.1 & 0.8 & 0.1 \\ 0 & 0.1 & 0.1 & 0.8 \end{bmatrix}. \quad (2)$$

Columns and rows corresponds to the modulation states from QPSK to QAM–256, which corresponds to 5G modulation orders. Another type represents a channel with the rapid modulation change, where the modulation rather change the

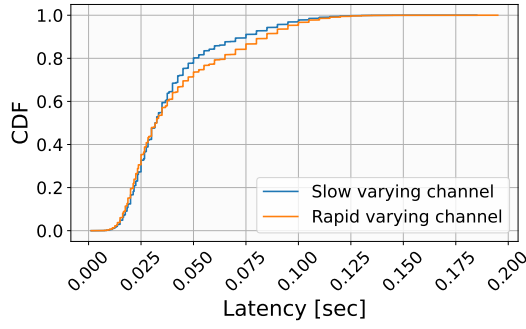


Fig. 2: Communication delay for 16 Mbps video stream with slow and rapid varying channel

order then stay the same. Transition matrix for rapid varying channel is:

$$\tau^{rapid} = \begin{bmatrix} 0.1 & 0.3 & 0.3 & 0.3 \\ 0.3 & 0.1 & 0.3 & 0.3 \\ 0.3 & 0.3 & 0.1 & 0.3 \\ 0.3 & 0.3 & 0.3 & 0.1 \end{bmatrix}. \quad (3)$$

System's delay, that was spent for wireless communication is approximated by following:

$$T_{comm}^{cell} = \frac{w}{C^{cell}}, \quad (4)$$

where w is the workload, Mb; and C^{cell} is a throughput for cellular or wired connectivity, Mbps.

Fig. 2 shows the calculated communication delay for 4K video stream for slow and rapid varying channel. As the difference is not significant, the slow varying channel is used further.

In this work, assume that wired connectivity provided by 10Gbps multi-mode optic fiber. System's delay, that was spent for transmission by optic fiber is approximated by following:

$$T_{comm}^{optic} = \frac{s}{v}, \quad (5)$$

where s is distance between transmitter and receiver, in meters; and v is a speed of light, $3 \cdot 10^8$ meters per second.

C. Rendering model

Rendering in XR systems is the process in real-time for video of converting 3D models into 2D, adding data to the image, and other processes [9]. In this work, the rendering model covers all the system's required computations. The rendering latency highly depends on the processing capabilities of the device. In theory, it could be presented as a computing process:

$$T_{comp} = \frac{w \cdot \sigma}{f}, \quad (6)$$

where f is the device's Central Processing Unit (CPU) frequency in cycles/sec, σ is the device's processor capabilities in cycles/bit, and w is the workload, bits.

D. Offloading model

Assume that there are three potential locations for data offloading: offload to Cloud, i.e., MCC, offload to a nearby Edge server, i.e., MEC, and process on the device locally, i.e., on the HMD. Local processing advances security because no data is shared with the third-party device. Nevertheless, local executions demand some computational capabilities, directly affecting the battery. The end-to-end latency contains only computational delay, which depends on the computational capabilities of the device. Alternatively, a computational rich MCC or MEC grants access to unlimited computational and storing resources but potentially holds other challenges, like intelligent user allocation, communication reliability, and security [10], [11]. The end-to-end delay model for edge computing consists of sum Uplink (UL) and Downlink (DL) communication delays and rendering latency. Similarly, the total latency for cloud computing is a sum of UL and DL communication delays, which consist of wireless hop and wired connectivity to the remote Data Center and rendering latency.

The application denotes a sequence of tasks, or in other words, a sequence of packets, defining a system workload w_i . Each packet can be processed locally, i.e., on HMD or outside the device, i.e., on the MCC, or MEC. The offloading decision is denoted as k_i in the sequence of the computing decisions K , and determined using heuristic approach. The offloading decision is possibly one of the following $k_i \in \{-1, 0, 1\}$, where $k_i = -1$ means that the executed on the HMD, $k_i = 0$ if the task processed on the Edge server, i.e., MEC, and $k_i = 1$, if the task is processing in the Cloud, i.e., MCC. Summarizing above, the application response time is presented in (7).

Here, T_{comp}^{local} is time for processing on a device, sec; p_{ex} is the power that was spent for computing, Wh; $T_{comp}^{MEC, MCC}$ is time for MEC and MCC computation, sec; p_i is the idle power state, Wh; T_{comm} is a latency of transmission data from the last Base Station (BS) to the device, sec; p_{RX} is the DL transmission power, Wh.

The components of the response time and power states are illustrated in Fig. 3. During the remote computation, the device is using the idle power state p_i , i.e., power saving mode of light sleep to save energy. Noticeably, the device uses the receiving power state $p_{RX/TX}$ when sending or receiving data. In the MCC model, the device sends data to the first BS in the transmitting power state, then it switches to the idle state, as it does not care about where data is transferring after the BS gets data till it reaches Cloud. Finally, execution power state p_{ex} is a power consumption of the CPU unit. In addition, assume that $p_i < p_{ex} < p_{RX}$ [12].

Power consumption is $E = \int P dt$, where P – the power that was used for the task in Watts. Considering above, the energy consumption model is presented in (8). The offloading decision strategy can be seen as an optimization function trying to minimize the system's response time without significant impact on energy consumption.

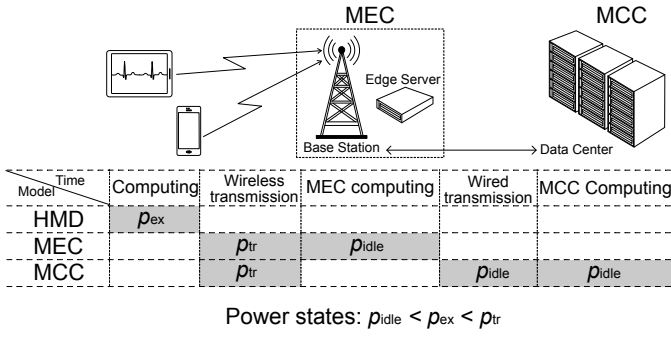


Fig. 3: Power states in three scenarios: local processing, MEC and MCC [13]

IV. TIMING ANALYSIS

Assume the packet is processed locally, on the HMD, or offloaded to MCC, or MEC server. When the packet arrives at the processing node (i.e., device or server), it might be processed immediately or wait in the queue for processing. This describes the delay in the queue for processing, which linearly depends on the node's processing capability. The accumulated queuing delay affects the total system latency. Simulations parameters are shown in the Table I.

A. Numerical results

Fig. 4a presents the response time and power consumption for local, MEC and MCC offloading, considering the queue for processing. It influences the Quality of Experience (QoE), as the motion-to-photon latency delay is 10 ms. In addition to that, the packet might not be relevant to the process, assuming the video stream, and it consumes more power. Offloading the processing to a resource-rich server helps to avoid long queues and, hence, potential failures of packet processing; also, they can increase the number of resources if required due to their scalability. Moving away from the Cloud node adds more communication delay to MCC. Fig. 4b presents simulations with the 1000 km distance to the Data Center, showing a significant decrease in the response time compared to 50 km in Fig. 4a. Nevertheless, both figures show that processing on MEC is more efficient than processing the task locally on the device.

Fig. 5 shows the dependency for the last packet MCC response time on the payload size and distance. Payload size is limited by MTU. While payload size does not affect the delay much, the distance dramatically increases the communication delay and strongly affects the overall system time.

The packet processing queue will dramatically increase the waiting time and power consumption and decrease the QoE, respectively, due to the packet delays. Despite the high computational capabilities of the Cloud, the overall response time suffers from a long communication path, which in turn depends on the distance between the user and the Cloud server. In this case, MEC benefits by providing the user with more computationally rich nodes in one hop. Resource-rich servers can handle longer queues; increasing the workload up to 30 Gbps for a 360 video stream will significantly affect their performance. Thus, parallel processing codecs, intelligent strategies for offloading, and algorithms are needed to reduce the workload without decreasing the QoE.

B. Hybrid offloading strategy

Assume that the first packet is always processed locally to exclude the communication delay for the first packet. The next decision is to offload the task, which can change to MEC or MCC, or stay on the device. Fig. 6 presents simulations for $N_{packets} = 1000$, $s = 200$ km.

Firstly, in the MCC model, the distance between the user and the data center has a significant impact. Secondly, the local processing is limited by the device's computational capabilities and battery size, while other strategies are relatively unlimited. Thirdly, the MEC model benefits from close-to-user implementation. Finally, the proposed hybrid strategy shows a significant improvement in latency.

The hybrid model benefits in total power consumption processing in the HMD. Nevertheless, the MCC and MEC models remain more effective in energy consumption. However, the relative power consumption, presented in Fig. 6, shows that the hybrid strategy advances MCC and MEC on average. In Fig. 6, clear transitions between states are visible. The median consumed power per packet for the hybrid model is $2.3 \cdot 10^{-10}$ mW, while for MCC and MEC, they are $1.1 \cdot 10^{-9}$ mW and $6.6 \cdot 10^{-10}$ mW respectively, which holds great potential for future implementation.

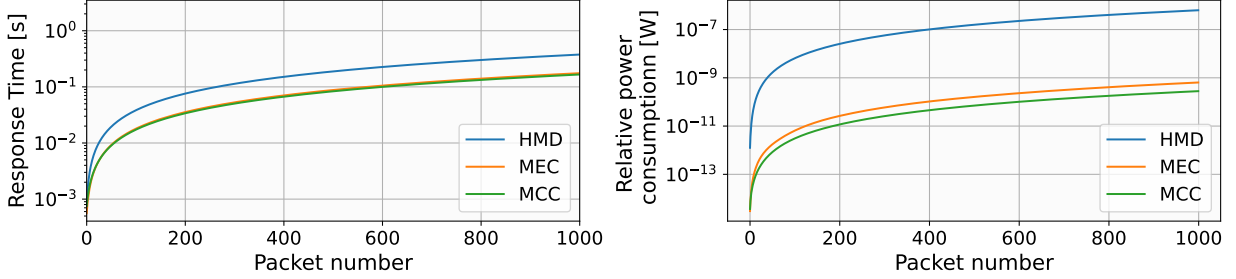
$$T = \sum_i \left(\frac{1-k}{2} |k| \cdot T_{comp}^{local}(t_i) + (2^{1-|k|} - 1) \cdot T_{comp}^{MEC}(t_i) + \frac{k+1}{2} k \cdot T_{comp}^{MCC}(t_i) + |k+1| \cdot T_{comm} \right), \quad (7)$$

$$P = \sum_i \left(p_{ex} \cdot \frac{1-k}{2} |k| \cdot T_{comp}^{local}(t_i) + p_i \left((2^{1-|k|} - 1) \cdot T_{comp}^{MEC}(t_i) + \frac{k+1}{2} k \cdot (T_{comp}^{MCC}(t_i) + T_{comm}) \right) + p_{RX} \cdot \frac{|k+1|}{2^k} \cdot T_{comm} \right), \quad (8)$$

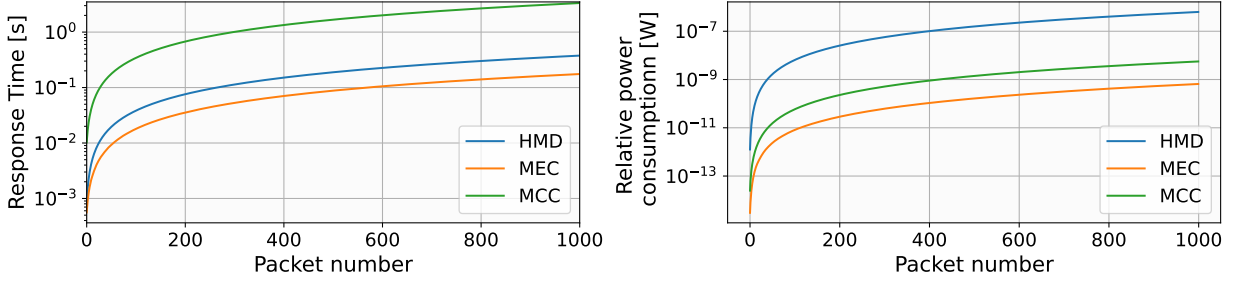
$$k_i = \begin{cases} -1, & \text{if processed on the device} \\ 0, & \text{if offloaded to the MEC server} \\ 1, & \text{if offloaded to the MCC server} \end{cases}$$

| Parameter | Notation | Value |
|--|-------------------|----------------------------------|
| Number of packets | N | 1,000 |
| Packet size | w | 12,000 bits |
| Packet's delay on arrivals | λ | 0.001 ms |
| HMD/MEC/MCC CPU processor frequency | $f_{HMD/MEC/MCC}$ | 35/75/288 GHz [14], [15] |
| Computation to data ratio | σ | $1.1 \cdot 10^3$ cycles/bit [16] |
| CPU power consumption | p_{ex} | 10 Wh |
| Receiver (RX)/Transmitter (TX) power consumption | $p_{RX/TX}$ | 0.04 Wh / 0.02 Wh |
| Idle power consumption | p_i | 0.01 Wh |
| Distance between user and Data Center | s | 400 km |
| Bandwidth UL/DL | B | 40 MHz / 100 MHz |

TABLE I: Simulation parameters for modeling system's response time



(a) System's response time and power consumption for $N_{packets} = 1000$, $s = 50$ km



(b) System's response time and power consumption for $N_{packets} = 1000$, $s = 1000$ km

Fig. 4: System's response time and power consumption for local processing and offloading to MEC and MCC

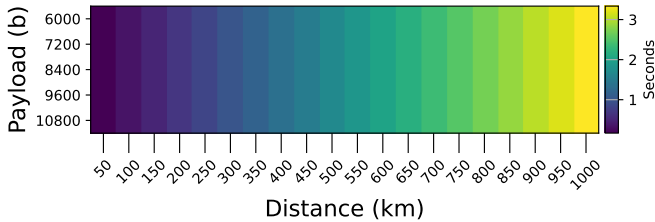


Fig. 5: MCC response time dependency on payload size and distance

V. CONCLUSION

In MA applications related to video processing, queue processing increases waiting time and power consumption and decreases QoE due to packet delays. Although the Cloud has high computational power, its long communication paths cause delays depending on the distance between the user and

the server. MEC offers advantages by providing users with computational resources in one hop, but high workloads can strain these servers. Therefore, parallel processing codecs, intelligent offloading strategies, and workload reduction algorithms are necessary to maintain QoE. As presented in the study, a hybrid strategy can minimize queue delays, improving response time and power consumption compared to local processing.

The hybrid offloading model offers flexibility and efficiency for future immersive scenarios. However, implementing this model in real-world applications presents several challenges and limitations. Firstly, the hybrid model relies heavily on network connectivity to offload tasks to the cloud. Thus, the offloading decisions differ depending on the user's distance from the MCC facilities. Also, efficient resource management between local and remote processing is challenging. Therefore, promising future research directions could include finding more efficient offloading strategies with

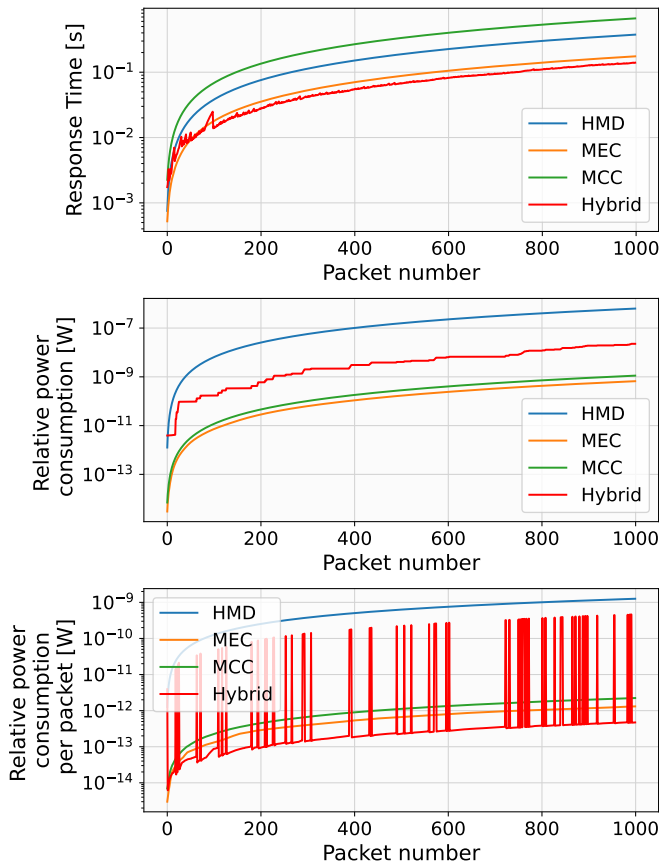


Fig. 6: System's response time and power consumption for hybrid model for $N_{packets} = 1000$, $s = 200$ km

reinforcement learning, i.e., Markov Decision Process (MDP), Q-learning, etc. Finally, scaling the hybrid offloading model to accommodate many devices and users requires robust infrastructure and efficient management strategies. Ensuring consistent performance across various devices and network conditions adds to the complexity to the system. Thus, another future research direction is a deeper exploration of scalability under high workloads.

ACKNOWLEDGMENT

The work of Daria Alekseeva was supported by the Pekka Ahonen Fund, DELTA, and the Doctoral Grant of the Information Technology and Communications Science Faculty at Tampere University.

REFERENCES

- [1] Z. Huang, C. Xiong, H. Ni, D. Wang, Y. Tao, and T. Sun, "Standard Evolution of 5G-advanced and Future Mobile Network for Extended Reality and Metaverse," *IEEE Internet of Things Magazine*, vol. 6, no. 1, pp. 20–25, 2023.
- [2] A. A. Esswie and M. Repeta, "Evolution of 3GPP Standards Towards True Extended Reality (XR) Support in 6G Networks," *arXiv preprint arXiv:2306.04012*, 2023.
- [3] E. Bozkir, S. Özdel, K. H. C. Lau, M. Wang, H. Gao, and E. Kasneci, "Embedding Large Language Models into Extended Reality: Opportunities and Challenges for Inclusion, Engagement, and Privacy," *arXiv preprint arXiv:2402.03907*, 2024.

- [4] K. Zeman, M. Stusek, P. Masek, P. Mlynek, and J. Hosek, "Augmented Reality-Aided Indoor Localization and Navigation for Next-Gen Rescue Services," in *Proc. of 14th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT)*. IEEE, 2022, pp. 202–206.
- [5] 3GPP TS 22.104 version 19.1.0, "Service Requirements for Cyber-Physical Control Applications in Vertical Domains," 2023-09.
- [6] 3GPP TR 26.918 version 15.2.0 Release 15, "Universal Mobile Telecommunications System (UMTS); LTE; Virtual Reality (VR) Media Services Over 3GPP," 2018-07.
- [7] 3GPP TR 22.826 version 17.2.0, "Study on Communication Services for Critical Medical Applications," 2021-03.
- [8] 3GPP TR 26.925 version 16.0.0 Release 16, "5G; Typical Traffic Characteristics of Media Services on 3GPP Networks," 2020-11.
- [9] 3GPP TR 26.928 version 17.0.0 Release 17, "5G; Extended Reality (XR) in 5G," 2022-05.
- [10] Q. He, G. Cui *et al.*, "A Game-Theoretical Approach for User Allocation in Edge Computing Environment," *IEEE Trans. on Parallel and Distributed Systems*, vol. 31, no. 3, pp. 515–529, 2019.
- [11] F. B. Dihn, A. A. Razzac *et al.*, "Adaptive Data Replication for URLLC in Cooperative 4G/5G Networks," in *Proc. of 32nd Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*. IEEE, 2021, pp. 1376–1381.
- [12] H. Wu, K. Wolter *et al.*, "EEDTO: An Energy-Efficient Dynamic Task Offloading Algorithm For Blockchain-Enabled IoT-Edge-Cloud Orchestrated Computing," *IEEE Internet of Things Journal*, vol. 8, no. 4, pp. 2163–2176, 2020.
- [13] D. Alekseeva, A. Ometov, and E. S. Lohan, "Towards the Advanced Data Processing for Medical Applications Using Task Offloading Strategy," in *2022 18th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*. IEEE, 2022, pp. 51–56.
- [14] "1U Rackmount AMD Ryzen Server," <https://www.onlogic.com/eu/store/mk100b-40/>, accessed on October 29, 2024.
- [15] "Amazon EC2 Instance Types," <https://aws.amazon.com/ec2/instance-types/>, accessed on October 29, 2024.
- [16] A. P. Miettinen and J. K. Nurminen, "Energy Efficiency of Mobile Clients in Cloud Computing," in *Proc. of 2nd USENIX Workshop on Hot Topics in Cloud Computing (HotCloud 10)*, 2010.