

Will Edge Computing Enable Location-based Extended/Mixed Reality Mobile Gaming? Demystifying Trade-off of Execution Time vs. Energy Consumption

Aleksandr Ometov, Jari Nurmi

Tampere University, Korkeakoulunkatu 1, Tampere, 33720, Finland

Abstract

The trailblazing development in mobile and wearable-based gaming dictates both the support of new technology enablers to allow for current demand and the development of modern computational offloading strategies to decrease the energy of handheld devices and maintain the energy emissions caused both by computation and transmission of data. Modern cellular networks already provide some support for proximity-based gaming, e.g., Ingress, PokemonGo, and The Witcher: Monster Slayer, among others. However, the demand of users is pushing the boundaries toward full-immersive Extended and Mixed Reality (XR/MR) experiences. Thus, computational offloading to the wireless network Edge becomes inevitable to keep the immersion high. This paper aims to analyze the impact of computational offloading (and, thus, execution time) on energy consumption. Computationally demanding games are analyzed for cases run locally, sent to a conventional remote server (cloud), offloaded to the user-owned more energy-independent device, or to the network edge. The results show that Edge computing operates the most efficiently regarding the trade-off between energy spent for execution vs. data transmission. It is also noted that distance to the edge node remains one of the critical factors affecting energy consumption.

Keywords

Mobile Gaming, Computational offloading, Edge computing, Cloud computing, Extended Reality

1. Introduction

One direction of current mobile gaming development is using handheld and wearable devices for game execution, e.g., in specialized areas or even outdoors [1, 2]. Naturally, mobile gaming could be executed locally on the device or via the remote server located in the cloud [3]. But both scenarios require the support of two fundamental enablers: powerful computational nodes and wireless networks [4]. However, technological advancement allows offloading the computationally demanding tasks to the network infrastructure-based servers closer to the user.

In many cases, handheld devices cannot provide satisfactory (console, computer-like, or Virtual Reality-like) immersion in a standalone mode as those are heavily limited by computational capacity, battery power, and communications capabilities [5]. According to Ericsson, the


WIPHAL 2023: *Work-in-Progress in Hardware and Software for Location Computation*, June 06–08, 2023, Castellon, Spain

✉ aleksandr.ometov@tuni.fi (A. Ometov); jari.nurmi@tuni.fi (J. Nurmi)

ORCID 0000-0003-3412-1639 (A. Ometov); 0000-0003-2169-4606 (J. Nurmi)

© 2023 Copyright for this paper by its authors.

Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

 CEUR Workshop Proceedings (CEUR-WS.org)

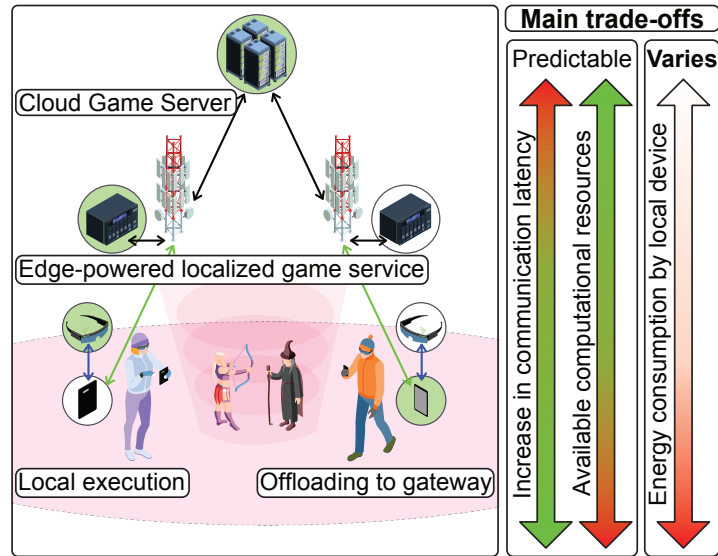


Figure 1: Reference architecture for XR/MR Mobile Gaming scenario: (1) – Local execution; (2) – Offloading to the gateway; (3) – Offloading to the cloud server; (4) – Offloading to the edge server.

latter is becoming less important with the development of cellular networks beyond 5G, which are expected to enable true-to-life worlds, sensory-rich feedback, and the end of lag [6] by delivering more efficient and on-the-fly computational offloading capabilities for content-heavy applications [7].

Overall, computational offloading moves some computing tasks from a local energy-dependent device to a remote computing location, e.g., a remote cloud server, as shown in Figure 1. Offloading is especially beneficial for resource-constrained devices, e.g., low-power Internet of Things (IoT) devices, wearables, or smartphones, or when the device must temporarily carry out tasks that are not within its capabilities [8]. For many typical smartphones, mobile gaming also falls within this scope. Naturally, this process requires support for efficient wireless connectivity.

The development of cellular networks beyond 5G (towards 6G) already allows for most human-driven use cases [9], i.e., for the connectivity to remote servers, thus, unloading the local GPU/CPU and potentially extending battery life. Naturally, it comes with a trade-off for transmission overheads and delays. Still, it comes with support for the actual user mobility and proximity-based communication enablers – their efficient convergence is the cornerstone of mobile gaming [10].

Overall, mobile games that use the player’s location as a crucial gameplay component are referred to as location or proximity-based games. The Witcher: Monster Slayer and PokemonGo are well-known examples of mobile location-based games. Here, players fight or capture XR/MR or virtual creatures while exploring the real world, as well as collectively competing with each other [11].

Naturally, the location-based game requires some knowledge of its’ location, i.e., it uses Other Global Navigation Satellite Systems (GNSS), commonly referred to as the main representative of GPS. The cellular network identifies the player’s location. It is further used for gameplay

interactions, e.g., fixing the unique location-bonded objectives on the map. This produces a distinctive gaming experience that combines the virtual and actual worlds. But, at the same time, it brings additional load to the cellular network as the gameplay-related data and the user telemetry needs to be transmitted to the game server (following the traditional gaming model). This might congest the network, especially in places with many players along with normal cellular users, increasing network traffic and occupying more wireless mediums.

To add more oil to the fire, proximity-based games often require more than just an intermittent cellular link but a reliable communications channel [12]. This requirement comes from the need for real-time interactions between the virtual world and the real one. Moreover, some level of communication should be maintained even when the player isn't actively playing for, e.g., random gamification encounters. Increased communications overhead will eventually increase the battery discharge rate and battery lifetime.

Proximity games empowered by computational offloading have their pros and cons. On the one hand, this might result in heavier network load, thus, higher licensed wireless medium use. Still, on the other hand, it can also enable a completely new level of thrilling gaming experience that fuses the virtual and actual worlds.

To summarize, cellular networks already play a crucial role in offloading computing tasks for mobile gaming. The offloading is currently done to the game server. At the same time, new cellular network standardization activities also allow offloading the data to the closest and more energy-independent device in the network, e.g., a gateway on an Edge-powered cellular Base Station (BS) [13].

This work aims to study the impact of energy consumption if the computationally demanding task was executed locally on a handheld/wearable device, offloaded to the closest user-carried gateway, or executed on the network edge BS. This work only focuses on the energy consumption of the local end devices, not the infrastructure nodes, as those are assumed to have a constant power supply.

The rest of the paper is organized as follows. Section 2 provides the background information on the applicability of computational offloading to mobile gaming and related benefits. Next, Section 3 outlines the system model used to simulate offloading the computationally expensive gaming task to the network Edge. Further, numerical results are summarized in Section 4. Challenges and future perspectives are provided in Section 5. The last section provides the conclusions and outlines future work direction.

2. Motivation and background

Computational offloading, being still in its infancy, can play a significant role in proximity-based gaming by allowing the game to carry out complex calculations and data processing on remote servers rather than on the player's device [5].

First, it can enable more sophisticated proximity-based games, which frequently require processing large amounts of data, such as mapping information and real-time location data, to offer improved game features. The advanced game can offer more sophisticated features and a more engaging gaming experience by shifting this processing to distant servers. This is especially important for Mixed/Augmented Reality scenarios, as those require both immersion

and heavy traffic processing.

Additionally, it might enable enhanced performance on low-end devices, i.e., on the player's device, proximity-based games can operate more quickly and smoothly by offloading computationally demanding tasks to remote servers [14]. As a result, improved frame rates, less lag, and a better gaming experience can all be obtained. Edge computing nodes can give proximity-based games more computing power, leading to better performance and gaming experience for the players.

The decreased data usage may also serve as a catalyst for networks that go beyond 5G, which is particularly important for cellular network operators. Proximity-based games can lessen the amount of data that must be transmitted over the cellular network while loading the backhaul by offloading computationally demanding tasks to distant servers. This can lessen the player's data usage to better use network resources, lower user data plans, and reduce the load on the constrained cellular network resources that operate in licensed bands.

Next, it will enable lower latency in localized gaming that doesn't require any connectivity to the server. Edge computing can significantly reduce latency in proximity-based gaming by having computing resources closer to the end users. Real-time conversations and an immersive gaming experience may result from this.

Edge computing may also improve proximity-based games' security and privacy by processing sensitive data locally rather than sending it to a distant cloud server. By doing this, the sensitive data could be preserved locally, thus, limiting potential scopes of attacks. However, that requires a separate set of ethical studies.

In summary, computational offloading can significantly contribute to proximity-based gaming by allowing the game to perform difficult calculations and data processing on distant servers. This may lead to enhanced game features, better performance, fewer hardware requirements, and less data usage.

3. System model

Based on the general wearable architecture, we assume a scenario involving a consumer Augmented Reality (AR) glasses [15] as the wearable device paired with the user's flagship 2023 smartphone. The smartphone is more powerful in terms of computing and energy resources than the AR device. Moreover, the smartphone is assumed to have a reliable wireless link to the cellular BS, which, in turn, has Edge computing capabilities.

Following the literature, we use a broader steady-state computing approach to express the game tasks. Here, a single state could be defined as a subprocess with the load aspect: data needed to be processed D (in bits); and the computing aspect: C – the number of CPU cycles/bit required to process the subprocess [16]. The calculation of C is based on various hardware parameters, e.g., memory, execution time, thread CPU time, number and type of instructions, and function calls [17].

The system could be described in four scenarios: (1) – Local execution on wearable; (2) – offloading to the gateway; (3) – offloading to the Cloud (located in 300km from the BS and with the gateway in 50m); (4) – offloading to the Edge with varying distance to the BS. The latter two cases may help save the energy resources of a device where the game is executed and reduce

latency for better experience and immersion.

As a metric of interest, we focus on the normalized power consumption and task completion times for the abovementioned scenarios. Naturally, the analyzed device may perform local processing, resulting in increased task execution time due to low processing power (if it even fulfills the game's requirements, potentially being a low-end one), thus, degrading the overall user experience. Alternatively, it could be offloaded to a more computationally powerful user device, e.g., a tablet or laptop carried along via proximity-based and network-assisted wireless link or an Edge server with comparatively higher computing resources at the expense of the additional power expended by the device in transmitting data to the task executor and receiving the processed results.

Nonetheless, if the task is offloaded to the Edge server from the end device, there is a need for a gateway node to act as a relay. The smartphone will serve as a relay node, receive the input data from the end device, and forward it to the edge server and vice versa to communicate the results via a long-range wireless link. Based on the technological development state-of-the-art, communications-wise and in contrast to conventional wearables, many latest AR/VR devices are equipped with multiple connectivity options such as Bluetooth Low Energy (BLE), Wi-Fi, mmWave, and/or LTE communication interfaces [18]. Therefore, we assume that the wearable device connects to the user's smartphone over Wi-Fi, further accessing the Edge server through a cellular LTE network.

Therefore, for an outdoor scenario, we assume that the AR device connects to the user's smartphone over conventional short-range IEEE 802.11 protocol (a.k.a., Wi-Fi), further accessing the Edge server through a cellular network.

Moreover, we assume that the subprocess (task) is already atomic and cannot be divided into smaller processes. We consider a dataflow of the AR device per second as the task D . Due to the nature of the game quest, we assume that the generated data size (e.g., video or image processing) is much higher than the resulting data size (e.g., providing a decision about the quest completion). Thus, transferring the resulting data takes many orders of magnitude lower demand on the communications link than the generated one and, therefore, could be neglected [19].

The performance evaluation results are based on the publicly available framework [27]. The system parameters are provided in Table 1. Basically, the task is a junk of data that needs to be processed. Notably, increasing the task data higher than the computation intensity in CPU cycles per bit does not affect the offloading until the system reaches the saturation of the traffic channel, i.e., when it fills the bottleneck of either IEEE 802.11 or LTE channels. However, our derivations are analytical and do not consider those scenarios.

3.1. Local execution

In the first scenario, the task is executed locally on the device [22], and the corresponding time is thus

$$T_i^w = \frac{D_i C_i}{F_i^w}, \quad (1)$$

where D_i is the input data size of task i in bits, C_i is the number of CPU cycles/bit required to execute the task i , and F_i^w denotes the processing power available on the wearable device in

Table 1
Main system parameters

Parameter	Value	Ref.
Input data size	20 MB	n/a
Task computational intensity	10^3 cycles/bit	[20]
Computational capacity of AR device	1GHz	[15]
Computational capacity of the Gateway	2.2GHz	[21]
Computational capacity of the Edge server	20GHz	[22]
Computational capacity of the Cloud server (e.g., Amazon Sumerian)	172.8GHz	[23]
Idle power consumption by the Wi-Fi module	0.177W	[24]
Transmit power consumption by the Wi-Fi module	1.094W	[24]
Receive power consumption by the Wi-Fi module	0.867W	[24]
Distance between the device and the gateway	0.8m	[25]
Distance between the used and the BS	50, 100, 200, 400, m	n/a
Distance between the BS and the Cloud	300km	n/a
Idle power consumption by the LTE module	0.03W	[26]
Transmission power consumption by the LTE module	0.1995W	[22]

terms of CPU cycles per second.

Naturally, the local task execution depends on the hardware specifics, thus, could be depicted as

$$P_i^w = \alpha^w ((V^w)^2 F_i^w), \quad (2)$$

where V^w – voltage of the chip, α^w is the processor capacitance based on the chip [28]. Notably by [29], V could be approximated as proportional to F_i^w , therefore,

$$P_i^w = \alpha^w (F_i^w)^3. \quad (3)$$

Therefore, for an input data size of D bits and the computational intensity of the task C cycles/bit, the energy consumption for executing a task locally on the wearable device, E_w , can be estimated as

$$E_i^w = P_i^w T_i^w = \alpha^w (F_i^w)^2 (D_i C_i). \quad (4)$$

3.2. Task offloading to the gateway

The natural way to offload the task to the closes computationally more powerful device is to execute the task on the user's gateway node, e.g., tablet or smartphone, which is, in turn, connected to the network.

The execution time for the scenario is, naturally, related to the computational time of the offloaded task (depending on the gateway specification), and short-range link transmission to the device (reception of reply could be considered as a negligibly small value)

$$T_i^s = T_i^{s,tx} + T_{s,ex}, \quad (5)$$

where tx denotes the transmission state and ex – computational one.

Naturally, $T_i^{s,tx}$ is related to the physical characteristics of the channel, which could be abstracted as throughput TH_{wifi} based on [30]. For simplicity, we omit the discussion on the estimation and fix the parameters to follow IEEE 802.11 specifications (see Table 1). Thus, $T_i^{s,tx}$, similarly to eq. (1) could be defined as

$$T_i^s = \frac{D_i}{TH_{wifi}} + \frac{D_i C_i}{F_i^s}. \quad (6)$$

Overall energy consumption follows a different trend since it also needs to consider the time when the wearable device is idle while waiting for the computation on a remote node to accomplish, as

$$E_i^s = E_i^{w,tx} + E_{s,ex} + E_{w,idle}, \quad (7)$$

where $E_{s,ex}$ is calculated similarly to eq. (1), $E_{w,idle}$ corresponds to the time when the end device is idling while waiting for the task to be executed remotely, and $E_i^{w,tx}$ is energy spent for transmission (reception by the smartphone does not contribute to the end device energy consumption) as

$$E_i^{w,tx} = \frac{P_{w,tx} D_i}{R_i^w}, \quad (8)$$

and $E_i^{w,idle}$ can be estimated as

$$E_i^{w,idle} = P_i^{w,idle} T_{s,ex}. \quad (9)$$

where $P_i^{w,idle}$ is the power spent in idling.

3.3. Task offloading to the edge and cloud server

The scenario of offloading to the cloud is the third traditional use case often used in current application developments. It usually requires the same communication overheads as offloading to the gateway. Still, it is supplemented by the additional time and energy consumption caused by long-range wireless transmission and a minor transmission over the fiber to the remote server. Offloading to the edge is very similar, excluding the fiber and with lower computational power. Therefore, the set of equations is kept uniform.

The time consumption could be, thus, defined as:

$$T_i^{off} = T_i^{w,tx} + T_i^{s,tx} + T_i^{fi,tx} + T_i^{off,ex}, \quad (10)$$

where $T_i^{s,tx}$ is the transmission time over the long-range link (e.g., LTE), $T_i^{fi,tx}$ is the transmission time over the fiber, and $T_i^{off,ex}$ is the offloaded task computation time. $T_i^{s,tx}$ could be defined in [22]. The related discussion is also omitted for simplicity. At the same time, the throughput TH_{lte} is calculated based on the standard [31] for a non-line-of-sight scenario that takes into consideration the distance between the transmitter and receiver d_{lte} .

For the edge offloading scenarios, $T_i^{fi,tx}$ would be equal to zero.

From the energy consumption perspective, the equations follow the same trend, i.e., the device needs to spend more energy idling while waiting for the reply. However, this dependence is not the same as for time:

$$E_i^{off} = E_i^{w,tx} + E_i^{w,idle}, \quad (11)$$

where $E_i^{w,idle}$ is the corresponding energy consumption while waiting for the reply:

$$E_i^{w,idle} = P_i^{w,idle} (T_i^{s,tx} + T_i^{fi,tx} + T_i^{off,ex}). \quad (12)$$

This way, the energy consumption and task time execution could be calculated for all scenarios. Interestingly, they all follow similar exponential trends with different exponents (2 vs. 3).

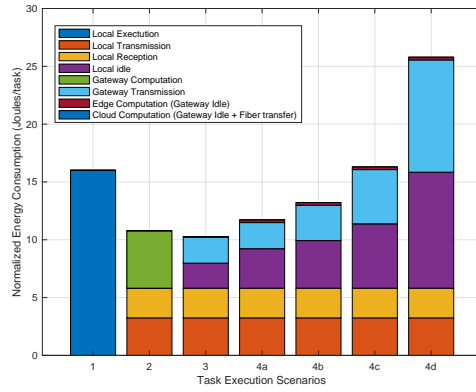
4. Performance evaluation

This section outlines a set of selected execution runs and highlights the main tradeoffs and non-linearities between energy and time consumption.

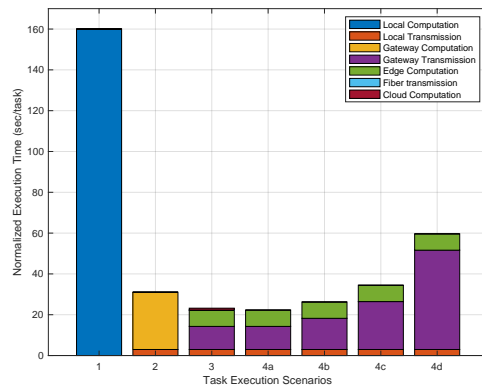
Figure 2(a) shows the energy consumption results for a fixed D of 20mb/task, corresponding to the minimum AR-like dataflow. Not-normalized value for the local execution scenario is 0.16. For AR offloading scenario, the computational trade-offs in terms of energy consumption, we have cases when execution locally may be more beneficial than computational offloading. For example, local execution (1) is almost ten times less energy-consuming than executing it on the gateway device (2) due to relatively similar hardware capabilities (the gateway is only 5 times more powerful than the glass). Cloud offloading for this scenario still seems to be the best solution energy-wise, but that changes tremendously concerning the execution time. Notably, depending on the distance to the BS (scenarios 4a-d), the offloading orchestrator may decide to execute the task locally or offload it to the Edge or Cloud. The simulations also proved that the given range of task intensity does not significantly affect the execution/transmission energy consumption. Thus, related plots were excluded for the sake of space.

Simultaneously, the results were obtained for the task execution time (normalized per task), see Figures 2(b). Not-normalized value for the local execution scenario is 160. Time-wise, the results of energy and time vary a lot. For example, the application parameters may require execution latency beyond a certain threshold, e.g., in mission-critical or latency & reliability scenarios [32]. In this case, local execution may quickly step out of the option. The decision would be left for the orchestrator to either try to execute the command on the gateway or offload it to the Edge or Cloud (while those are clearly affected by the wireless propagation as the main driver for increased latency).

An example of the task operation may also be based on the device's computational intensity, which depends on the hardware parameters, see Figure 3(a). Here, the less computationally powerful the device is – the more impact on the system operation is observed. The capabilities of AR devices degrade the fastest. At the same time, the almost infinite computational power of the Cloud only has a minor effect. Naturally, as the task remains the same, the impact on the communication side is fixed, also visible from Edge-related dashed curves, which have almost no effect on the distance to the BS, see Figure 3(b). The reason behind this may be natural – the actual execution of the task on any device takes a significantly longer time than the transmission.



(a) Normalized energy consumption



(b) Normalized time consumption

Figure 2: Normalized results for 20 mb/task: (1) – Local execution on wearable; (2) – Offloading to the gateway; (3) – Offloading to the cloud server (fixed distance to BS of 50m); (4a-d) – Offloading to the edge server

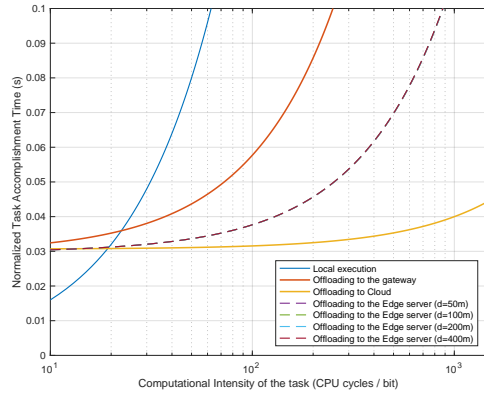
That may change for cases of more heavy traffic, e.g., high-resolution VR scenarios. However, that is unlikely to be considered as mobile gaming but rather ones in indoor [33].

In summary, the application and actual operation scenario remain the main drivers for deciding whether to process the task locally or offload it to a remote location. At the same time, state-of-the-art devices already provide very high computational capabilities. Simultaneously, the bottlenecks of battery and communications links would remain present for a very long time.

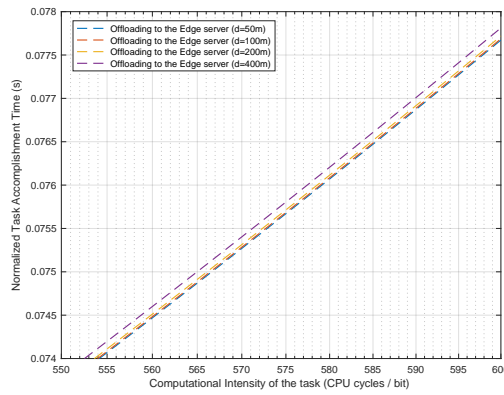
5. Challenges and Discussion

While computational offloading can bring several benefits to proximity-based gaming, several challenges need to be addressed to make it a successful and effective solution:

Network connectivity: Proximity-based gaming requires a relatively constant connection to the



(a) Broad range of intensities



(b) Zoomed range for Edge scenario

Figure 3: Normalized time consumption vs. the device computational task intensity for different offloading scenarios.

network to transmit data between the device and the remote servers, which could be congested very fast for various reasons [34]. This can be challenging in areas with poor network coverage or high congestion, resulting in poor performance and reduced user experience. Solutions like proximity-based communications (e.g., Device-to-Device (D2D) communications) or utilization of higher frequency may solve the congested licensed wireless medium issue [35, 36]. However, it still requires certain connectivity to the cloud/network orchestrator.

Bandwidth requirements: Computational offloading requires significant bandwidth to transmit data between the device and the remote servers. This can be challenging, especially in areas with limited bandwidth or high network congestion. It can result in increased latency and reduced performance. Utilization of heterogeneous networking and improved spacial reuse and prioritization of time-critical traffic may decrease the bandwidth load on the wireless network side. The load balancing behind the backhaul may be solved by efficient data compression on the device side and/or by utilizing Network Function Virtualization (NFV) packet-level

solutions [37]. Revised traffic shaping may also improve the overall system performance.

Execution Latency: Latency, or the time it takes for data to travel between the device and the remote servers supplemented by the execution delay, is an important factor in proximity-based gaming. High latency can result in reduced performance for localized gaming (for cases when the server presence is not mandatory for the entire gaming experience) and a less immersive gaming experience. Novel solutions to orchestrate the games themselves must be developed to efficiently enable localized gaming via, e.g., advanced Mobile Edge Computing focused on reliability and latency [32].

Security and Privacy: Offloading computational tasks to Edge servers can also raise security concerns, as sensitive data, such as the player's location, is transmitted over the network and processed on third-party hardware [38]. This can increase the risk of data breaches and raise privacy concerns. Traditional privacy-preserving approaches and recently booming advanced contact tracing algorithms may be applied to ensure the desired level of privacy (required for a certain application). Secure virtualization and virtual machine deployment on the Edge may be another promising research direction.

Cost: Setting up and maintaining a computational offloading infrastructure can be expensive, requiring significant investment in hardware and software resources. However, the standardization activities set by 3GPP in this direction are already developing at a strong pace [13]. Even today, application developers can use region-specific Cloud infrastructure to reduce the latency of gaming, resulting in a decrease in overall operational costs. New service placement strategies must be developed to fulfill costs, latency, and energy efficiency trade-offs.

Hardware & Software limitations: As general hardware and System-on-Chip (SoC) performance constantly grow, the offloading may be executed even on present hardware. However, with the growing demand for applications, adding more racks on the Edge may become unreasonable [39]. Therefore, new solutions to decrease task complexity need to be developed. Those may cover better predictability approaches already heavily studied, e.g., assisted remote surgeries, by applying Machine Learning (ML). From a hardware perspective, some games may generally require "good enough" results, i.e., allowing some room for inexact or approximate computing.

Overall, computational offloading can bring several benefits to proximity-based gaming. Still, it also presents several challenges that need to be addressed, such as network connectivity, bandwidth requirements, latency, security, and cost. These challenges must be carefully considered and managed to ensure the success and effectiveness of computational offloading in proximity-based gaming.

6. Conclusions

In summary, the future of computational offloading for mobile gaming will likely see continued growth and development, driven by technological advances and the increasing demand for more immersive and engaging mobile gaming experiences.

In this work, we have highlighted that future Mixed/Extended Reality scenarios for mobile gaming have a lot of aspects to consider while developing the computational offloading strategies still have a long path to go through. Especially important, we have shown that the device's

energy consumption in local execution and various offloading cases do not go in hand with latency. Thus, an intelligent approach to set the target function is of out-most need to be defined while developing future location-based games powered by the offloading mechanisms.

As a future work, we aim to understand how location-based gaming would affect the wearable device battery life and study the impact on the cellular network's infrastructure energy consumption, commonly left unnoticed during the analysis. It is mainly motivated by the present energy crisis in Europe. Therefore, we aim to contribute to the sustainable development of the operators' future wireless networks.

Acronyms

5G	5th generation wireless network
AR	Augmented Reality
BLE	Bluetooth Low Energy
BS	Base Station
D2D	Device-to-Device
GNSS	Global Navigation Satellite Systems
IEEE	Institute of Electrical and Electronics Engineers
IoT	Internet of Things
LTE	Long-term evolution
ML	Machine Learning
MR	Mixed Reality
NFV	Network Function Virtualization
SoC	System-on-Chip
Wi-Fi	Wireless Fidelity
XR	Extended Reality

Acknowledgments

The work was supported by Jane and Aatos Erkko Foundation through the CONVERGENCE of Humans and Machines project.)

References

- [1] M. Saaty, D. Haqq, D. B. Toms, I. Eltahir, D. S. McCrickard, A Study on Pokémon GO: Exploring the Potential of Location-based Mobile Exergames in Connecting Players with Nature, in: *Extended Abstracts of the 2021 Annual Symposium on Computer-Human Interaction in Play*, 2021, pp. 128–132.
- [2] Z. Cheng, B. N. Greenwood, P. A. Pavlou, Location-based Mobile Gaming and Local Depression Trends: A study of Pokémon Go, *Journal of Management Information Systems* 39 (2022) 68–101.

- [3] S. Paasovaara, T. Olsson, Proximity-based Automatic Exchange of Data in Mobile Gaming: Studying the Experiences of Streetpass Users, in: Proc. of 9th Nordic Conference on Human-Computer Interaction, 2016, pp. 1–10.
- [4] S. Andreev, J. Hosek, T. Olsson, K. Johnsson, A. Pyattaev, A. Ometov, E. Olshannikova, M. Gerasimenko, P. Masek, Y. Koucheryavy, et al., A Unifying Perspective on Proximity-based Cellular-Assisted Mobile Social Networking, *IEEE Communications Magazine* 54 (2016) 108–116.
- [5] X. Zhang, H. Chen, Y. Zhao, Z. Ma, Y. Xu, H. Huang, H. Yin, D. O. Wu, Improving Cloud Gaming Experience through Mobile Edge Computing, *IEEE Wireless Communications* 26 (2019) 178–183.
- [6] Ericsson: How Will 5G and Edge Computing Transform the Future of Mobile Gaming?, [Online] Available: <https://www.ericsson.com/en/blog/2021/3/5g-edge-computing-gaming> (Accessed June 29, 2023), 2021.
- [7] J. Du, F. R. Yu, G. Lu, J. Wang, J. Jiang, X. Chu, MEC-assisted Immersive VR Video Streaming over Terahertz Wireless Networks: A Deep Reinforcement Learning Approach, *IEEE Internet of Things Journal* 7 (2020) 9517–9529.
- [8] K. Cao, Y. Liu, G. Meng, Q. Sun, An Overview on Edge Computing Research, *IEEE Access* 8 (2020) 85714–85728.
- [9] K. H. Tan, H. S. Lim, K. S. Diong, Modelling and Predicting Quality-of-Experience of Online Gaming Users in 5G Networks, *International Journal of Technology* 13 (2022) 1035.
- [10] Q. Bi, The Proximity Radio Access Network for 5G and 6G, *IEEE Communications Magazine* 60 (2022) 67–73.
- [11] M. Baer, T. Tregel, S. Laato, H. Söbke, Virtually (re) Constructed Reality: The Representation of Physical Space in Commercial Location-based Games, in: Proc. of 25th International Academic Mindtrek Conference, 2022, pp. 9–22.
- [12] E. Cuervo, A. Wolman, L. P. Cox, K. Lebeck, A. Razeen, S. Saroiu, M. Musuvathi, Kahawai: High-quality Mobile Gaming Using GPU Offload, in: Proc. of 13th Annual International Conference on Mobile Systems, Applications, and Services, 2015, pp. 121–135.
- [13] TS 23.558 (Rel. 17), Architecture for Enabling Edge Applications, [Online] Available: <https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3723> (Accessed June 29, 2023), 2022.
- [14] A. Tshipis, K. Oikonomou, Joint Optimization of Social Interactivity and Server Provisioning for Interactive Games in Edge Computing, *Computer Networks* 212 (2022) 109028.
- [15] Google Glass, Technical Specifications, [Online] Available: <https://support.google.com/glass/answer/3064128?hl=en> (Accessed June 29, 2023) (2023).
- [16] X. Lyu, H. Tian, Adaptive Receding Horizon Offloading Strategy Under Dynamic Environment, *IEEE Communications Letters* 20 (2016) 878–881.
- [17] M. Othman, S. A. Madani, S. U. Khan, et al., A Survey of Mobile Cloud Computing Application Models, *IEEE Communications Surveys & Tutorials* 16 (2013) 393–413.
- [18] H. Sun, Z. Zhang, R. Q. Hu, Y. Qian, Wearable Communications in 5G: Challenges and Enabling Technologies, *IEEE Vehicular Technology Magazine* 13 (2018) 100–109.
- [19] Z. Cheng, P. Li, J. Wang, S. Guo, Just-in-Time Code Offloading for Wearable Computing, *IEEE Transactions on Emerging Topics in Computing* 3 (2015) 74–83.
- [20] K. Cheng, Y. Teng, W. Sun, A. Liu, X. Wang, Energy-Efficient Joint Offloading and

- Wireless Resource Allocation Strategy in mMulti-MEC Server Systems, in: Proc. of IEEE International Conference on Communications (ICC), IEEE, 2018, pp. 1–6.
- [21] Google Pixel 7, Technical Specifications, [Online] Available: https://store.google.com/product/pixel_7_specs (Accessed June 29, 2023) (2023).
- [22] X. Lyu, H. Tian, C. Sengul, P. Zhang, Multiuser Joint Task Offloading and Resource Optimization in Proximate Clouds, *IEEE Transactions on Vehicular Technology* 66 (2016) 3435–3447.
- [23] Amazon Sumerian, Easily create and run browser-based 3D, augmented reality (AR), and virtual reality (VR) applications, [Online] Available: <https://aws.amazon.com/sumerian/> (Accessed June 29, 2023), 2023.
- [24] Y. Xiao, et al., Modeling Energy Consumption of Data Transmission over Wi-Fi, *IEEE Transactions on Mobile Computing* 13 (2013) 1760–1773.
- [25] Max Roser and others, Human Height-Our World in Data, *Our World in Data* (2019).
- [26] A. Carroll, G. Heiser, et al., An Analysis of Power Consumption in a Smartphone, in: Proc. of USENIX Annual Technical Conference, volume 14, Boston, MA, 2010, p. 21.
- [27] W. B. Qaim, A. Ometov, C. Campolo, A. Molinaro, E. S. Lohan, J. Nurmi, Understanding the Performance of Task Offloading for Wearables in a Two-Tier Edge Architecture, in: Proc. of 13th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT), IEEE, 2021, pp. 1–9.
- [28] Y. Jang, J. Na, S. Jeong, J. Kang, Energy-Efficient Task Offloading for Vehicular Edge Computing: Joint Optimization of Offloading and Bit Allocation, in: Proc. of IEEE 91st Vehicular Technology Conference (VTC2020-Spring), IEEE, 2020, pp. 1–5.
- [29] T. D. Burd, R. W. Brodersen, Processor Design for Portable Systems, *Journal of VLSI Signal Processing Systems for Signal, Image and Video Technology* 13 (1996) 203–221.
- [30] 802.11 OFDM Data Rates, The Math Behind The Numbers, [Online] Available: <https://dot11.exposed/2018/11/29/802-11-ofdm-data-rates-the-math-behind-the-numbers/> (Accessed June 29, 2023), 2018.
- [31] 3GPP TR 36.814, Evolved Universal Terrestrial Radio Access (E-UTRA); Further Advancements for E-UTRA Physical Layer Aspects (Release 9), 2017.
- [32] M. S. Elbamy, C. Perfecto, C.-F. Liu, J. Park, S. Samarakoon, X. Chen, M. Bennis, Wireless Edge Computing with Latency and Reliability Guarantees, *Proceedings of the IEEE* 107 (2019) 1717–1737.
- [33] O. Buruk, M. Salminen, N. Xi, T. Nummenmaa, J. Hamari, Towards the Next Generation of Gaming Wearables, in: Proc. of CHI Conference on Human Factors in Computing Systems, 2021, pp. 1–15.
- [34] S. Paasovaara, E. Olshannikova, P. Jarusriboonchai, A. Malapaschas, T. Olsson, Next2You: A Proximity-based Social Application Aiming to Encourage Interaction between Nearby People, in: Proc. of 15th International Conference on Mobile and Ubiquitous Multimedia, 2016, pp. 81–90.
- [35] D. Moltchanov, A. Ometov, P. Kustarev, O. Evsutin, J. Hosek, Y. Koucheryavy, Analytical TCP Model for Millimeter-Wave 5G NR Systems in Dynamic Human Body Blockage Environment, *Sensors* 20 (2020) 3880.
- [36] A. Ometov, A. Orsino, L. Militano, G. Araniti, D. Moltchanov, S. Andreev, A Novel Security-Centric Framework for D2D Connectivity based on Spatial and Social Proximity, *Computer*

Networks 107 (2016) 327–338.

- [37] P. Tam, S. Math, S. Kim, Optimized Multi-Service Tasks Offloading for Federated Learning in Edge Virtualization, *IEEE Transactions on Network Science and Engineering* 9 (2022) 4363–4378.
- [38] N. Mäkitalo, A. Ometov, J. Kannisto, S. Andreev, Y. Koucheryavy, T. Mikkonen, Safe and secure execution at the network edge: a framework for coordinating cloud, fog, and edge, *IEEE Software* 35 (2018) 30–37.
- [39] H. J. Damsgaard, A. Ometov, J. Nurmi, Approximation Opportunities in Edge Computing Hardware: A Systematic Literature Review, *ACM Computing Surveys* (2022).