González Morín, D.; Pérez, P.; García Armada, A. Toward the distributed implementation of immersive augmented reality architectures on 5G networks, in *IEEE Communications Magazine, vol. 60, no. 2, Feb. 2022, pp. 46-52*

DOI: 10.1109/MCOM.001.2100225

Toward the Distributed Implementation of Immersive Augmented Reality Architectures on 5G Networks

Diego González Morín, Pablo Pérez and Ana García Armada

Abstract

Augmented Reality (AR) has been lately presented as one of the key technology fields in which 5G networks can become a disruptive tool, rising interest from both industry and academia. The main goal of this article is to extend the current state of the art of distributed AR studies and implementations by extracting the main AR algorithms offloading requirements individually. This extension is further achieved by depicting the data flow between these algorithms and their hardware requirements. From the obtained results, we estimate a preliminary set of network Key Performance Indicators (KPIs) for a subset of three examples of distributed AR implementations highlighting the necessity of 5G technologies and their ecosystem to unveil the full potential of AR. Finally, and based on these KPIs, we propose a set of 5G configuration parameters for a successful distributed AR implementation. As most of the described algorithms are also used in VR applications, our contributions can facilitate future distributed implementations of both AR and VR applications.

Introduction

The improvement of Augmented Reality (AR) and Virtual Reality (VR) technologies has enabled the proposition of novel use cases which entail a new manner of interacting with the real world and with each other. While VR aims to allow the user to fully immerse in a virtual scenario AR targets to blend the real scenario with overlaid virtual content, greatly enhancing the user experience. However, there are still several technological constraints, such as limited computing power or battery life, which hinder AR and VR applications to reach their full potential. The fifth generation of mobile telecommunication networks (5G) includes a set of three services: enhanced mobile broadband (eMBB), ultra-reliable low-latency communications (URLLC), and massive machine-type communications (mMTC). Both eMBB and URLLC 5G services match AR and VR applications' extremely high communication data rate requirements under demanding low latency constraints. From the resource allocation perspective, both eMBB and URLLC are competing services which consume resources from each other. Consequently, smart network designs, including multi-access edge computing (MEC), are required for immersive and enhanced AR/VR experiences.

AR and VR network requirements are mostly use-case or application dependent. There are plenty of research studies [1] of 5G-based wireless VR implementations and architectures, which commonly focus on the high-quality and low-latency downlink video-stream. However, it is more seldom to find research avenues focused on distributed AR implementations on 5G networks.

The future of AR aims to create an enhanced reality which will reveal new ways of human to human interaction, learning or entertaining. In industry, for instance, AR devices could display on-site real-time instructions to an operator, or automatically detect safety failures or incorrect procedures. On the social side, AR can add an extra dimension to the current remote human communications: in realistic 3D avatar-based real-time communication users would be able to simultaneously interact with virtual objects or, as a longer-term goal, with each other, demolishing the distance barriers in human communication.

AR applications should not only properly place the virtual content but allow both the user and the real scenario to interact with it. This requires the real environment to be analyzed as accurately as possible and in real-time, requiring multiple heavy duty algorithms, such as semantic segmentation and 3D reconstruction, to concurrently run in real-time. Even though there are some real-time state-of-the-art implementations of these processes [2][3], they require modern hardware which is usually not portable and energy consuming. Besides, the AR content should be rendered with a motion to photon delay below 15 ms [4], deadline to which most of the previously mentioned processes need to comply. Even more constraint is the case in which the user needs to interact with a virtual object: the virtual object's reaction and haptic feedback latencies must be smaller than 10 ms [4].

These tight processing deadlines along with very demanding hardware requirements justify the idea of offloading some or all these algorithms from the device. 5G eMBB and URLLC services theoretically allow to upload the data streams from the different sensors and receive the processed results while satisfying the tight real-time requirements. To that end, the most latency-critical processes should be handled as close to the device as possible: MEC can be considered a pivotal piece in the distributed implementation of AR applications. There are some studies [5] which thoroughly analyze the distributed implementation of AR, proposing several offloading architectures and some use case-dependent requirements. While the proposed use cases' requirements and their link to proposed 5G solutions and architectures are well described, [5] lacks a study on how and where the individual algorithms should run, and what are the individual offloading requirements. Each individual AR application requires different combinations of AR algorithms. Knowing the specific offloading requirements for each algorithm is a key factor for an optimal architecture design: different offloading combinations can be deployed for the same use case or application depending on the available network resources. Consequently, our work can be considered as an extension of the current state of the art with the following contributions:

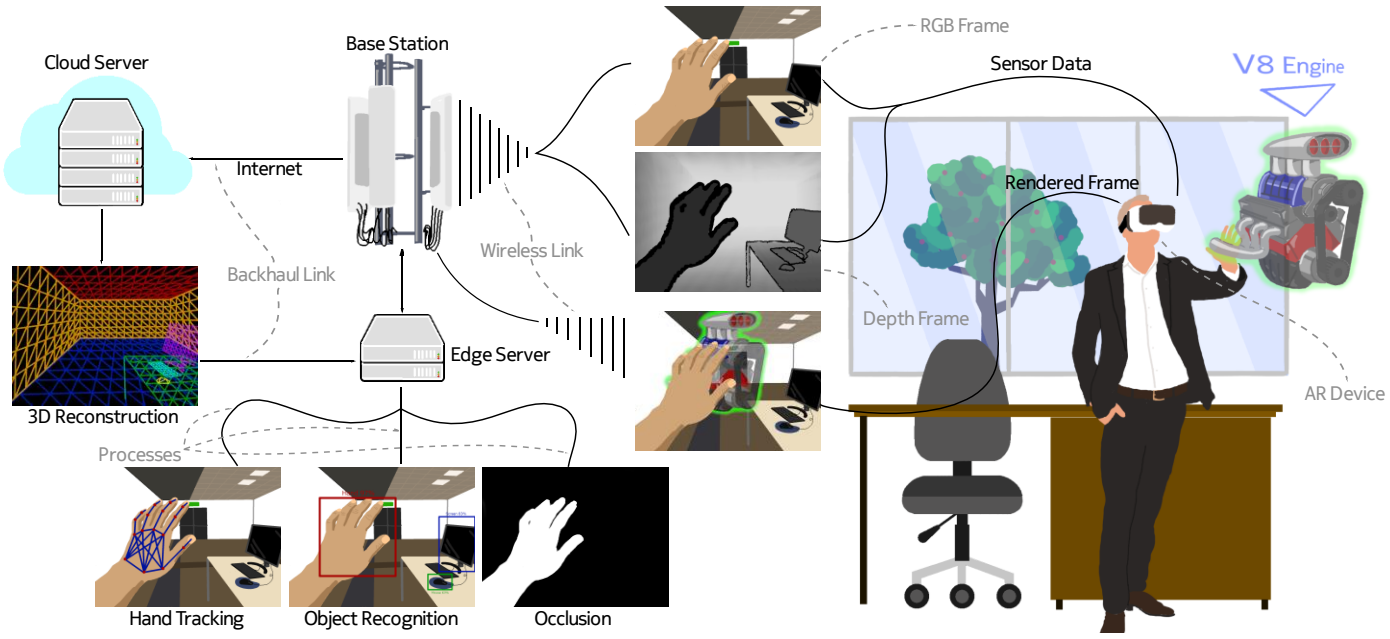

Figure 1: Simplified offloading architecture for a complex Augmented Reality application.

- A thorough and in-depth analysis of the main AR algorithms and the data flow between these algorithms, pinpointing which of them are robust candidates to be offloaded from the AR devices and under which network conditions.
- A set of latency and throughput requirements for different offloading scenarios.
- Aligned with the architectures described in [5], we propose an AR offloading architecture for a set of representative algorithms.
- An enabling network configuration tightly linked with the standards which satisfy the offloading requirements.

Our final goal is to facilitate distributed AR deployments by characterizing which algorithms should run where and what are their offloading requirements given an offloading architecture. Besides, the individual analysis of these algorithms can be useful for successful offloading in other fields such VR and autonomous driving. An example architecture is depicted in Fig. 1, which represents the analyzed algorithms and general network elements.

A Breakdown of Augmented Reality Algorithms

Any complex AR application involves numerous algorithms and processing blocks which interact with each other. To study the challenges and requirements of AR offloading we need to understand each process individually and the data flow between them.

Sensor Capture and Preprocessing

In every AR application data is periodically gathered from the different sensors and shared between the processing blocks. Cutting edge AR devices include several sensors: multiple RGB cameras, one or more depth cameras, microphones, etc. The recently released Hololens 2 [6] is a perfect example of such sensor capture complexity. Running at 60 Hz, the Hololens 2 is producing around 500 Mbps of raw sensor data.

The stability of the virtual content placement is directly affected by the Frames Per Second (FPS) reached by the device, as humans can perceive visual changes as fast as 13 ms [7]. Hence, the AR device update time should be as close as possible to this reference threshold. In practice, it is considered that 60 FPS is the minimum update frequency to have a tolerable motion-to-photon latency, and current devices typically work at such frame rate. Any AR offloading attempt must ensure a frame update time of less than 16.6 ms.

Simultaneous Localization and Mapping

One key feature of any AR device is the capability of accurately estimating its own pose in real-time using the available sensor data. This pose is used to anchor the virtual content to the real scenario, so that the device movement does not affect the pose and behavior of the virtual objects. This estimation is done using Simultaneous Localization and Mapping (SLAM) algorithms [8]. The minimum set of sensors required for a robust AR-targeted SLAM algorithm, is an RGB camera and an IMU. In some examples, the SLAM algorithm is also fed with the depth information, increasing the algorithm's accuracy and robustness. On the other hand, the output bitrate could be neglected as it is only composed by some few dozens of floats for every processed frame.

In some cases, the SLAM algorithm output is used for 3D reconstruction, requiring the generated point cloud and RGB/depth key frames to be exported, considerably increasing the output bit rate. Offloading the SLAM process would require both the 5G eMBB and URLLC capabilities to fulfil the constraint update times.

3D Reconstruction

3D reconstruction is a crucial step for a successful immersive experience: the AR engine uses the 3D model of the world to realistically place and anchor the virtual content allowing realistic interactions with the real scenario. Besides, 3D reconstruction can be used to partially hide the virtual objects that are placed behind a real object in a process called AR occlusion. Incorrect AR occlusion handling might produce the virtual content to be perceived unrealistically, with inconsistent size and position.

Furthermore, 3D reconstruction can be considered an extension of SLAM algorithms to an extent in which some

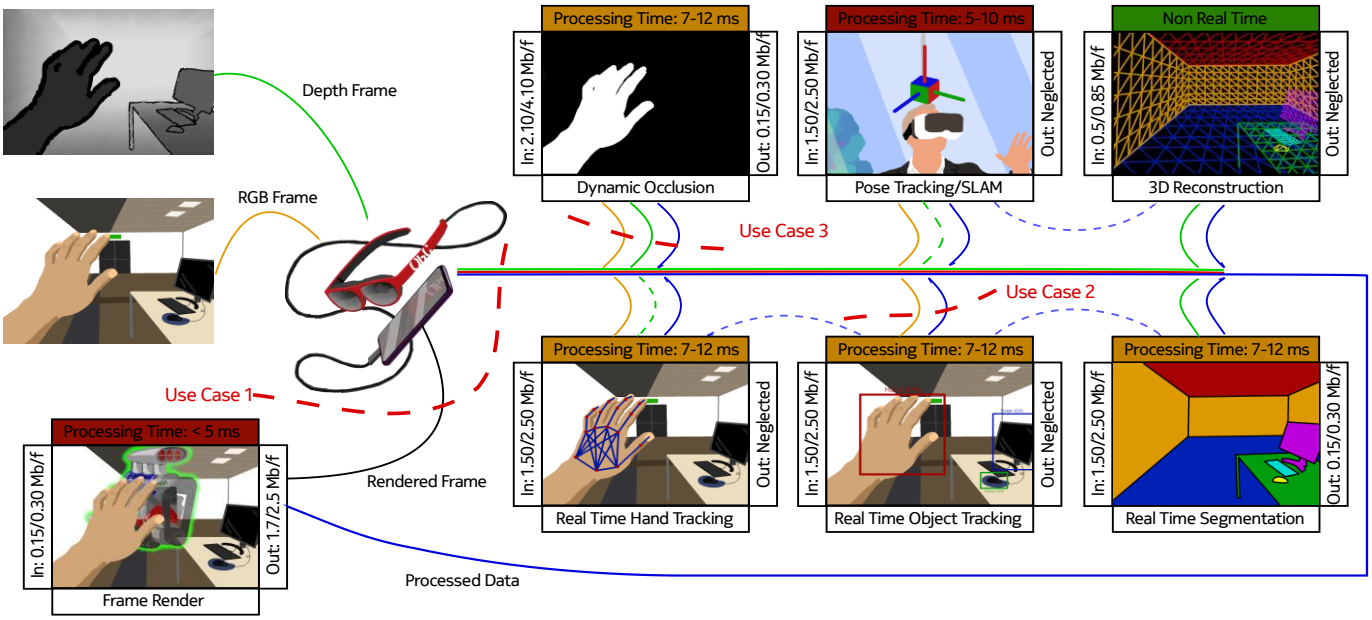

Figure 2: Simplified AR architecture including estimated processing times and required input and output data

SLAM algorithms simultaneously handle the pose tracking and the 3D updates along with the corresponding color textures. It is not efficient to continuously export the entire mesh, but only the sections which are modified. The offloading requirements are less restrictive as the generated mesh updates can be transmitted at a lower rate, 0.5-1 Hz, and higher latencies (below a second), allowing this process to be offloaded to a further server. Finally, the final 3D reconstruction could be stored on the device or a server after each session and be reloaded in a posterior session.

Semantic Understanding

Semantic understanding algorithms obtain semantic information from the real scene, such as which real objects are in the scene along with their shape, pose, and size. In our analysis, we focus in three of the most important semantic understanding algorithms: object detection, hand tracking, and semantic segmentation.

Object detection is a very populated research topic for its importance on uncountable fields such as autonomous driving or AR. Consequently, its state of the art has improved rapidly, reaching real time performance on standalone devices with limited computing power such as smartphones. However, the hardware usage is very high while achieving lower detection accuracies than in GPU-enabled computing systems.

Hand tracking is a particular use case of object detection in which the user hands along with the individual fingers' joints and their poses are tracked. It is a key step to allow seamless interactions between the user and the virtual objects and GUIs.

Semantic segmentation can be understood as a natural extension of object detection algorithms: once the object is detected, its accurate shape and 2D position is extracted. This step can be used, for instance, to handle the occlusion of dynamic objects like the user's hands.

When these algorithms are used for real-time tasks, the latency requirements are given by the device update rate, as the estimated result is used to render the immediate next frame. The current research trend of semantic understanding is to study how to significantly decrease the size of these networks while preserving their accuracy. Two key examples of these shallow networks are the YOLO-Tiny which runs at 244Hz on a GPU, while YOLO-LITE reaches 20 FPS running on a CPU [11]. While in the object detection and hand tracking cases the output bit rate can be neglected, the semantic segmentation's output consists of a pixel classification mask per frame considerably increasing the output bit rate. Offloading these algorithms requires the combination of intensive computing resources and low latency data transmission, demanding a near MEC with the URLLC service to offload this process from the device.

Real-time Dynamic Occlusion

Dynamic occlusion handling is a tough task which is still considered an open AR problem. If the virtual content is not properly occluded behind a moving object, for instance the user's hands, it would produce a wrong perception of its position within the real world, ruining the experience. It requires a perfect segmentation of the moving object, with very constrained processing times. The input is usually composed by the RGB and depth. The output consists of just the occlusion mask which can be considered a high resolution grayscale feed. Some state of the art examples can run in real time, as in [12], reaching 60 Hz on a high-end GPU.

These demanding hardware and processing requirements impose the usage of a dedicated GPU-enabled MEC. Furthermore, both the URLCC and eMBB services are necessary to satisfy the throughput and latency requirements. Real-time dynamic occlusion is a perfect example of a demanding machine learning algorithm whose analysis can be later used as a reference for offloading other complex AR algorithms.

AR Engine and Frame Rendering

The AR engine simulates the physics of the virtual content and renders a new frame accordingly. Contrary to VR, in AR this step is not extremely demanding, allowing the devices to be completely wireless. However, targeting to decrease the size and weight of the device, it is natural to consider the possibility of offloading these steps. The AR engine is fed with all the processed data in all the previously described

	Data	Rate (Hz)	Frame Size (Mbit)	Bit Rate (Mbps)
(1)	RGB + IMU	60	2.50	150
	RGB + D + IMU	60	3.83	230
	Pose + RGB	20	2.50	50
	Pose + RGB + D	20	3.83	77
(2)	RGB + Pose	20	2.50	50
	RGB + Pose + D	20	3.83	77
	3D Mesh + Tex	1	<<1	<<1
(3)	RGB	60	2.50	150
	Bounding Box	60	<<1	<<1
(4)	RGB	60	2.50	150
	Semantic Mask	60	0.32	20
(5)	RGB + Depth	60	3.83	230
	Occlusion Mask	60	0.32	20
(6)	Masks + Pose	60	0.32	20
	HD Frame	60	2.50	150

Table 1: Update rates and, input (green) and output (blue) bitrates for a set of Augmented Reality algorithms: (1) SLAM, (2) 3D Reconstruction, (3) Object Detection, (4) Semantic Understanding, (5) Dynamic Occlusion, (6) AR Rendering

algorithms, while the output is the high definition rendered frame to be displayed

The latency requirements are severe as no frame can be skipped or received with a delay greater than a frame's period, demanding the URLLC service to ensure an extremely low latency and reliable connectivity between a nearby MEC and the device.

AR Algorithms Summary

From the AR breakdown, we extracted a set of input and output data rates and latency requirements of each of the described algorithms, gathered in Table 1. To estimate these values, we use the Hololens 2 [6] as the reference device, with RGB and depth resolutions of 1920x1080 and 720x720 respectively and a refresh rate of 60 Hz. As indicated in [13] we assumed the individual frames are compressed in JPEG with a quality corresponding to a pixel weight of 1.2 bit, from which we estimate the values summarized in Table 1. Besides, Fig. 2 depicts the described algorithms, along with their input and output frame sizes and processing times.

Hardware Requirements

The state of the art of the described algorithms involves a wide range of different approaches, demanding the hardware requirements to be specific for each individual solution. SLAM algorithms [8] do not need high-end hardware to run at rates higher than 20 Hz. On the contrary, even in optimized hand segmentation algorithms [14] the processing hardware must be equipped with a dual high-end GPU with at least 12 GB of memory and 3,584 cores for inference times below 30 ms.

Network Requirements for AR Offloading

In this section we describe what are the actual estimated network requirements to offload some or all of the aforementioned processes while ensuring a satisfactory immersive experience. For our analysis, we simplify the ideal transmission time of a single frame, considering that it is only affected by the amount of data to be sent divided by the effective throughput. We also define the total transmission latency as the aggregation of the ideal transmission time and the total transmission latency. This latency includes not only the propagation latency, but other network-specific latencies such as the ones associated with the hybrid automatic repeat request (HARQ) process, among others.

In AR offloading, the total update time is the aggregation of the sensor feed (uplink) transmission time, data processing, retrieved results (downlink) transmission time and frame display time. The summatory of the algorithms processing, AR rendering and display times is what we refer to as AR processing time. Finding a general and unique value of the AR processing time is extremely hard as it does not only depend on the algorithms that are implemented, but also on the machine capabilities, or available hardware resources. For this reason, we present some illustrative results of a simple optimization problem to find the less restrictive roundtrip transmission latency and uplink and downlink throughput requirements which ensure an update rate of 60 Hz given different possible AR processing times. The optimization problem has been solved using Python's SciPy library with the Nelder-Mean algorithm. The goal is to present the most suitable pairs of peak throughput and latency for both the uplink and downlink streams given a range of AR processing times spanning from 1 ms to 14 ms.

To obtain these values, we designed a straightforward reward function which favors low uplink and downlink throughputs, high round trip latencies, and total transmission times. It includes individual weights to fine-tune the influence of each of these parameters. To obtain bounded results we impose a set of constraints to the optimization solver. First, we constrain the total update time to be smaller than the update period (16.7 ms). Second, we limit the minimum roundtrip latency to be 1 ms, as it is the duration of a subframe according to 5G specifications. Finally, we constrain the maximum total peak throughput to be 10 Gbps.

We consider that 5G networks are initially designed to use the Time Division Duplex (TDD). According to the 3GPP 38.213 specification, the TDD configuration in 5G is very flexible, allowing tens of different slot configurations, which can be fixed or dynamically selected. The slot configuration determines which OFDM symbols are assigned to uplink or downlink within a slot. For extremely demanding applications, the proper selection of the slot configuration is crucial to satisfy the link requirements. To simplify the estimation, we consider the number of slots reserved for downlink and uplink to be balanced in our optimization problem. In a later analysis of the results, we analyze how different configurations comply with the estimated latency and throughput requirements. For our analysis, we consider that no adaptive offloading technique [15] is used as we want to study the most restrictive case in which the environment is highly dynamic. In this case, all the captured frames are sent to the server. We use the estimated uplink and downlink frame sizes included in Table 1 for our analysis.

For the sake of simplicity, we choose the 3 most representative AR offloading scenarios. In use case A, we analyze the full AR offloading scenario, which is necessary to enable ultra-light mobile AR devices with almost no computation capabilities. In the case in which the AR device includes a computationally powerful companion or can

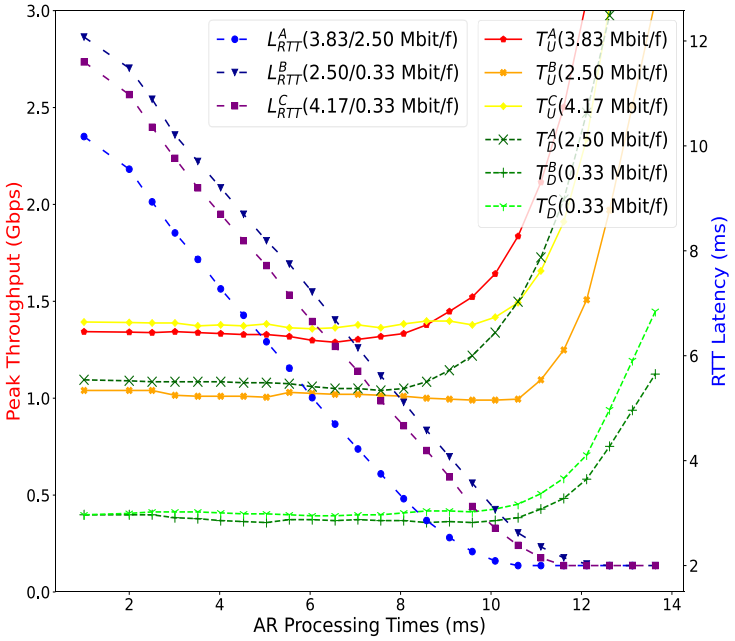

Figure 3: Set of optimal pairs of roundtrip latencies ($RTT - L$) and uplink and downlink peak throughputs (T_U and T_D) for a set of given AR processing times for the 3 use cases running at 60 Hz.

achieve heavier computations, only some of the mentioned algorithms need to be offloaded. We analyze the offloading of two different demanding algorithms separately: object detection and semantic segmentation (use case B) and occlusion handling (use case C), as they are crucial AR computationally intensive algorithms. The red dash lines in Fig. 2 represent which algorithms are offloaded in each use case. Notice that the throughput values are not the mean values but the peak throughput values that the network needs to supply during the frame transmission time.

Use Case A. Full Offloading

In this first use case we the full AR processing stack to be offloaded from the device. In this setup the uplink stream is expected to include all the sensor data. On the downlink side, the system is transmitting the rendered scene back to the device. The estimated mean uplink rate for this first use case is between 150 and 230 Mbps for a device running at 60 FPS. On the downlink side, we estimate the transmission to require 2.5 Mbit per frame, as the transmitted data corresponds to the high definition rendered frame. In Fig. 3 we can observe the estimated pairs of round-trip latencies and uplink and downlink peak throughput optimal values which satisfy the 16.6 ms of total update time for different AR processing times. The depicted latency corresponds to the total roundtrip latency, including any latency added by the entire network stack. We can observe that for AR processing times between 6 and 8 ms, the estimated roundtrip latencies are 4 ms and above. Within these bounds, the required peak throughput values lay around 1.3 Gbps on the downlink side and 1.2 on the uplink.

Use Case B: Object Detection and Segmentation

In this case, the only offloaded process is the object detection and segmentation algorithm. The uplink stream corresponds to the RGB and depth feeds, while the downlink only includes the single-channel segmentation mask. This setup requires uplink and downlink rates of around 2.5 Mbit and 0.33 Mbit per frame respectively. To analyze the current use case, we choose the same AR processing times span as in the previous example. As expected, we can observe in Fig. 3 how the downlink peak throughput requirements considerably drop below 0.4 Gbps as the downlink data steam is slimmer than in the previous use case. On the uplink side, the required peak throughput remains high, with estimated values below 1.2 Gbps. The estimated roundtrip latency is higher than 5 ms for AR processing times below 8 ms.

Use Case C: Occlusion Handling Offloading

Occlusion handling algorithms require at least the depth and RGB frames to be constantly transmitted. Besides, some state-of-the-art algorithms use the output from the 3D reconstruction and hand tracking algorithms, which in this case are sent from the device. Consequently, we estimate the required input to weight 4.17 Mbit per frame while the output is lighter: 0.33 Mbit per frame from the high-resolution occluding mask. We can observe in Fig. 3 how the estimated peak throughput and round-trip latency requirements slightly increase compared to the previous use case. However, real-time occlusion is a hard and unsolved problem which we can expect to require higher AR processing times. In that case, the required round-trip latency might drop below 4 ms with uplink peak throughput values above 1.4 Gbps.

5G RAN Configuration

To fulfil the above detailed requirements, we propose an initial 5G RAN configuration which can lead to a successful AR offloading scheme. In general, AR offloading demands tight peak throughput requirements on the uplink side, with values above 1 Gbps. On the downlink side, the required peak throughput can reach similar values. Finally, the required roundtrip network latency might decrease to values below 4 ms if we are constrained to hardware providing AR processing times above 8 ms. These requirements demand a very specific and well-designed network configuration. The required roundtrip latencies are extremely hard to achieve for several reasons. Current RAN are very complex systems with very different processes which add extra latency to the transmission. First, low latency demanding processes must run as close as possible to the gNB, avoiding any backhauling connectivity if possible. Consequently, the use of well-equipped MEC systems is crucial. There are other potential sources of latency such as the HARQ process. This acknowledgement-based error control tool is a key element of current wireless networks. However, it can induce high packet transmission latencies if many retransmissions are required. To avoid this scenario, the modulation and coding scheme (MCS) selection must be well aligned with the current network conditions. Incorrectly selected MCS might produce high error rates and, consequently, higher latencies and peak throughput requirements.

Consequently, the scheduling algorithm must be designed to prioritize the AR user. This prioritization is crucial both to achieve high peak throughput and low latency communication. Packets corresponding to the prioritized user should be allocated as soon as they are ready, along with any necessary packet retransmission. Besides, the TDD scheme should be carefully selected. As there is a higher demand on the uplink side, TDD slot configurations must prioritize the uplink stream. According to the 3GPP TS 38.213 specification, TDD configurations 34 to 42 and 50 to 55

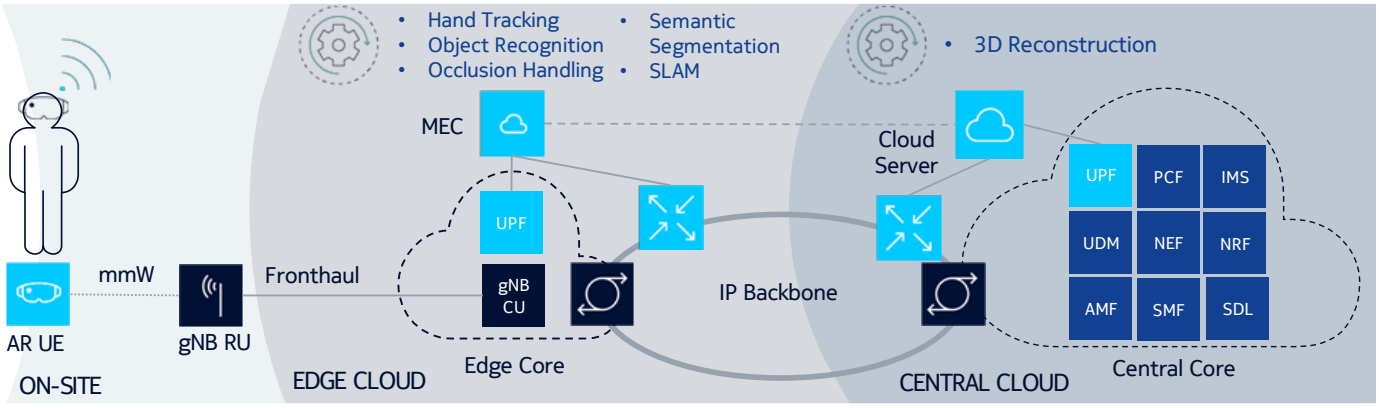

Figure 4: 5G RAN proposed architecture.

prioritize higher the uplink stream. Ideally, the slot configuration should be dynamically selected depending on the network status, connected users and scheduling decisions, as described in the 3GPP TS 38.213 specification.

From the resource allocation side, it is key to select the proper subcarrier spacing (or numerology) to ensure that the network can provide the required data rates. According to the 3GPP TS 38.306 specification, the estimated data rate requirements can be sufficed using a subcarrier spacing of 120 KHz (numerology 3) and a bandwidth of 400 MHz. High numerology also has positive impact on the latency, as they correspond to shorter OFDM symbols.

Given the previous analysis and the tight estimated requirements, we conclude that the throughput requirements can be better covered on high frequency bands: millimeter wave (mmW) along with MIMO and Carrier Aggregation can reach bandwidths above 2 GHz according to the 3GPP TS 38.101 specification, becoming a key enabler of portable immersive AR. Besides, we would need to make use of other crucial capacities of 5G, such as Non-Public Networks (NPNs) which, as described in 3GPP TS 28.807, allow controlled user prioritization and very specific scheduling schemes.

AR offloading may require heavy peaks of upstream throughput with low latency. AR traffic should travel through a specific network slice where RAN resources are properly reserved and QoS policies are established throughout the network. This slice could coexist with other slices in the same NPN with different requirements (e.g., mMTC for IoT devices in a connected factory, and eMBB, mostly with downlink capabilities, for general intranet/internet access).

Finally, as the offloading sets demanding hardware requirements for the MEC, particularly in terms of GPU resources, it is important to use dynamic resource allocation and virtualization. The idle MEC capacity available in low-usage hours (e.g. during the nights) may be dynamically released to other tasks that can be performed offline and scheduled for such hours (e.g. heavy data analytics). Therefore, the AR slice should also be able to acquire and release the computing resources whenever they are needed.

The proposed 5G network architecture for AR offloading, is shown in Fig. 4. We can observe how the latency-constrained algorithms, such as dynamic occlusion, semantic segmentation, and object tracking, run on a MEC with no required backhaul connection. Other non-real-time processes, such as 3D reconstruction, can run on a further server, which involves the backhaul link and the 5G core.

Conclusions

In this article we have analyzed the main technology and network requirements for an immersive AR distributed architecture and how 5G can become a key enabler of fully immersive AR. We have described the current state of the art of AR algorithms, along with their individual required inputs and outputs, and hardware requirements. We have studied the latency and peak throughput requirements necessary to succesfully offload some of the described algorithms. Furthermore, we have numerically exemplified how the latency, throughput and AR processing times are related with each other, proposing possible network configurations that may guarantee them. The estimated required peak throughput and latency values call for the use of 5G milimeter wave spectrum along with massive MIMO, smart resource allocation slicing, and NPNs. Besides, we have described the importance of novel and optimized schedulling approaches to fullfil the tight requirements for AR offloading. Finally, we characterized the parameters and algorithm distribution of a plausible 5G network architecture to enable the presented distributed AR offloading implementation. As a main conclusion we can foresee that the time has come for the implementation of AR on feasible devices thanks to the efficient architecture, throughput and latency enabled by 5G.

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] M. S. Elbamby *et al*, "Toward Low-Latency and Ultra-Reliable Virtual Reality", in *IEEE Network*, vol. 32, no. 2, pp. 78-84, March-April 2018.

[2] H. Zhao *et al.* "ICNet for Real-Time Semantic Segmentation on High-Resolution Images", *ECCV,* 2018

[3] A. Dai *et al.* "BundleFusion: Real-Time Globally Consistent 3D Reconstruction Using On-the-Fly Surface Reintegration". *ACM ToG.* 36, 3, Article 24 (May 2017), 18 pages.

[4] F. Zheng et al., "*Minimizing latency for augmented reality displays: Frames considered harmful*" 2014 IEEE ISMAR, Munich, 2014, pp. 195-200.

[5] Y. Siriwardhana, *et al.*, "A Survey on Mobile Augmented Reality With 5G Mobile Edge Computing: Architectures,

Applications, and Technical Aspects," in IEEE Communications Surveys & Tutorials, vol. 23, no. 2, pp. 1160-1192, 2021

[6] Hololens 2—Overview, Features, And Specs | Microsoft Hololens. [online] Available at: https://www.microsoft.com/en-us/hololens/hardware, accessed Dec. 13, 2021

[7] Mary Potter et al. "*Detecting meaning in RSVP at 13 ms per picture*". Attention, perception & psychophysics, Springer (2013).

[8] J. Delmerico and D. Scaramuzza, "*A Benchmark Comparison of Monocular Visual-Inertial Odometry Algorithms for Flying Robots*" 2018 IEEE ICRA.

[9] Raúl Mur-Artal and Juan D. Tardós. "*ORB-SLAM2: an Open-Source SLAM System for Monocular, Stereo and RGB-D Cameras*". IEEE Transactions on Robotics, vol. 33, no. 5, pp. 1255-1262, 2017

[10] Dai, Angela et al. "*BundleFusion: Real-Time Globally Consistent 3D Reconstruction Using On-the-Fly Surface Reintegration*" 2016 ACM ToG

[11] Jonathan Pedoeem and Rachel Huang, "*YOLO-LITE: A Real-Time Object Detection Algorithm Optimized for Non-GPU Computers*", arXiv, 2018

[12] Anna Katharina Hebborn et al., "*Occlusion Matting: Realistic Occlusion Handling for Augmented Reality Applications*", IEEE ISMAR 2017.

[13] G. K. Wallace, "The JPEG still picture compression standard," in IEEE Transactions on Consumer Electronics, vol. 38, no. 1, pp. xviii-xxxiv, Feb. 1992

[14] E. Gonzalez-Sosa, et al., "Enhanced Self-Perception in Mixed Reality: Egocentric Arm Segmentation and Database With Automatic Labeling," in IEEE Access, vol. 8, pp. 146887-146900, 2020

[15d] Luyang Liu, Hongyu Li, and Marco Gruteser, "Edge Assisted Real-time Object Detection for Mobile Augmented Reality", ACM MobiCon, New York, NY, USA, Article 25, 1–16, 2019

Biographies

Diego González Morín (diego.gonzalez_morin@nokia-bell-labs.com) is a Ph.D student at Nokia Bell Labs Spain, enrolled with Universidad Carlos III de Madrid, Spain. He received his B.Sc. and M.Sc. in Industrial Engineering from Universidad Politécnica de Madrid in 2015 and 2018 respectively. In 2018, he received his M.Sc. in Systems, Control and Robotics from Kunliga Tekniska Högskolan (KTH) in Stockholm, Sweden. After receving his M.Sc. degrees, he joined Ericsson Research's Devices Technologies group as a researcher, where his research interest focused on Augmented Reality technologies, field in which he holds 3 patents. From august 2019, he joined Nokia Bell Labs as a Ph.D student. He is currently pursuing his Ph.D. focused on the application of ultra-dense networks for the implementation of distributed media rendering.

Pablo Pérez received the Telecommunication Engineering degree (integrated BSc-MS) in 2004 and the Ph.D. degree in Telecommunication Engineering in 2013 (Doctoral Graduation Award), both from Universidad Politécnica de Madrid (UPM), Madrid, Spain. From 2004 to 2006 he was a Research Engineer in the Digital Platforms Television in Telefónica I+D and, from 2006 to 2017, he has worked in the R&D department of the video business unit in Alcatel-Lucent (later acquired by Nokia), serving as technical lead of several video delivery products. Since 2017, he is Senior Researcher in the Distributed Reality Solutions department at Nokia Bell Labs. His research interests include multimedia quality of experience, video transport networks, and immersive communication systems.

Ana Garcia Armada (S'96-A'98-M'00-SM'08) is a Professor at University Carlos III of Madrid, Spain. She has published approximately 150 referred papers and she holds four patents. She serves on the editorial board of IEEE Trans. on Communications and IEEE Open Journal of the Communications Society. She has served on the TPC of more than 50 conferences, and she has been part of many organizing committees. She has received several awards from University Carlos III of Madrid, including an excellent young researcher award and an award to best practices in teaching. She was awarded the third place Bell Labs Prize 2014 for shaping the future of information and communications technology. She received the outstanding service award from the IEEE ComSoc Signal Processing for Communications & Computing Technical Committee (formerly SPCE). Her research mainly focuses on signal processing applied to wireless communications.