Computers, Materials & Continua
DOI:10.32604/cmc.2021.017948
Article

Tech Science Press

Bit Rate Reduction in Cloud Gaming Using Object Detection Technique

Daniyal Baig[1], Tahir Alyas[1], Muhammad Hamid[2], Muhammad Saleem[3], Saadia Malik[4], Nadia Tabassum[5,*] and Natash Ali Mian[6]

[1]Department of Computer Science, Lahore Garrison University, Lahore, 54000, Pakistan
[2]Department of Statistics and Computer Science, University of Veterinary and Animal Sciences, Lahore, 54000, Pakistan
[3]Department of Industrial Engineering, Faculty of Engineering, Rabigh, King Abdulaziz University, Jeddah, 21589, Saudi Arabia
[4]Department of Information Systems, Faculty of Computing and Information Technology-Rabigh, King Abdulaziz University, Jeddah, 21589, Saudi Arabia
[5]Department of Computer Science, Virtual University of Pakistan, Lahore, 54000, Pakistan
[6]School of Computer and Information Technology, Beaconhouse National University Lahore, 53700, Pakistan
[*]Corresponding Author: Nadia Tabassum. Email: nadiatabassum@vu.edu.pk
Received: 15 January 2021; Accepted: 18 March 2021

Abstract: The past two decades witnessed a broad-increase in web technology and on-line gaming. Enhancing the broadband confinements is viewed as one of the most significant variables that prompted new gaming technology. The immense utilization of web applications and games additionally prompted growth in the handled devices and moving the limited gaming experience from user devices to online cloud servers. As internet capabilities are enhanced new ways of gaming are being used to improve the gaming experience. In cloud-based video gaming, game engines are hosted in cloud gaming data centers, and compressed gaming scenes are rendered to the players over the internet with updated controls. In such systems, the task of transferring games and video compression imposes huge computational complexity is required on cloud servers. The basic problems in cloud gaming in particular are high encoding time, latency, and low frame rates which require a new methodology for a better solution. To improve the bandwidth issue in cloud games, the compression of video sequences requires an alternative mechanism to improve gaming adaption without input delay. In this paper, the proposed improved methodology is used for automatic unnecessary scene detection, scene removing and bit rate reduction using an adaptive algorithm for object detection in a game scene. As a result, simulations showed without much impact on the players' quality experience, the selective object encoding method and object adaption technique decrease the network latency issue, reduce the game streaming bitrate at a remarkable scale on different games. The proposed algorithm was evaluated for three video game scenes. In this paper, achieved

14.6% decrease in encoding and 45.6% decrease in bit rate for the first video game scene.

Keywords: Video encoding; object detection; bit rate reduction; game video motion estimation; computational complexity

1 Introduction

The concept of cloud computing is being employed in online cloud gaming systems; games have become an important part of our society. The term Cloud Gaming is forthcoming to be one of the most significant terms in on-line gaming as it executes distributed computing benefits for gaming. The main thought is to run the game on the cloud gaming server and render the game scenes to the client with updated messages. The client's device is just required to process a particular measure of information that mirrors client's responses to the game. The gigantic measure of information preparing, for example, 3D illustrations rendering, are not important to be done on the customer's system. The gushing of a game should be possible normally by spilling items and designs to the player alongside refreshed messages. This methodology devours a great deal of transmission capacity and necessities higher prerequisites for clients' systems. Another way is finished by encoding the game as a video stream to the customer system.

The new pattern of on-line gaming and the term of cloud gaming is to limit reliance on the end-client equipment. The end-client ought not to endure the colossal measure of graphics handling on his/her device. Present day video gaming pattern manages the game as a video arrangement not as a game and can be called as a Game as Video service. In gaming as a video service encoding and rendering of the game scene is performed on the server side and a video stream is transferred to the client device where it is decoded. Many cloud gaming service which are compatible with windows, for example, GeForce now had been working to improve the hindrances in the gaming services. As the internet capabilities are increasing each year cloud gaming will be becoming more relevant as of now many big companies are moving their games to the cloud services and offering the facility on per monthly basis.

Cloud gaming gives numerous advantages by decreasing equipment required to run a game and protecting rights of game engineers, lamentably, a few issues progressed while applying this procedure. Data transmission impediments and transmission inertness influenced ineffectively on the nature of the administration. This issue happens as the most pieces of rendering are done on the cloud and not on the client's computer as in the ordinary way. Cloud gaming concept involves rendering the game as a video over the internet. Streaming of the video and receiving commands from the users causes latency and increases the encoding time required to process the video before transferring the video to the client. Frame by Frame video is rendered to the client. This increases the time complexity for the processing. But moving games to the cloud saves game engineers efforts by eliminating the chance of piracy and games can be developed more platform independent. As the cloud gaming is enhancing but the high internet speed of almost 12 Mbps and high bit rate is significantly reducing the market for cloud gaming for example Onlive failed because of high demand and company not having enough servers to support the demand. This problem occurs as the most parts of displaying are carried out on the cloud environment and not on the customer's device as in the conventional way.

Many of the Convolutional neural networks are mostly being used for some specific system detections. Some CNN networks are utilized for car parking detection, collision detection and certain object detections. As compared to yolo v4 tiny these detectors are slow. For the cloud gaming video detections a real time object detector is required which utilizes the gpu for faster object detections. In cloud gaming environment latency delay and encoding time plays an important role. Many of latest CNN models require a huge time to train but Yolo v4 can be trained on one conventional gpu as shown in Fig. 1. Yolo V4 can be trained on custom dataset and pre trained dataset for fast detections. Yolo v4 runs twice as fast compared to efficientDet. The requirement to embed machine learning and cloud gaming is to use a fast operating detector which works both with custom and pre trained weights [1].

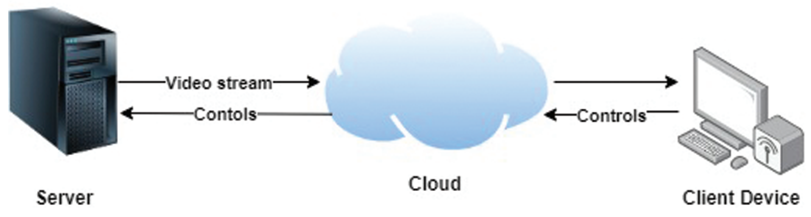

Figure 1: Cloud gaming conceptual model of interaction between user and server

The strategy of configuration of a video sequence from one sequence to another is known as video encoding. Video encoding can also be described as ‘transcoding’ or ‘video Compression.’ During recording particular format and specification is given by device. By using an internal encoding process we will be able to compress huge video files that need high amounts of data capacity. The proposed method is using videopro a video encoder which is used to compress video sequences. Basic encoding format being utilized is H.264/AVC which are mostly used for cloud game video encoding. Our system will work independent of the compression tool being used.

We have evaluated the performance measure on the game mafia one which is an open world game. The metrics used for measuring the game include encoding time, streaming bit rate and video size in mbs. The results in 5 experimental analysis display the improvements we had achieved. The rest of the paper is organized as in the Section 2 we will discuss the related work on the cloud gaming, in third section we will discuss the components and the architecture design. In the fourth section of the research dataset and the performance evaluation is discussed [2].

2 Related Word

In the video encoding mechanism video games can be streamed as a video sequence, to implement the sequence there is a need to change the game scene so it can be utilized in distinct domain. The test demonstrated an increase in the player’s quality of experience. The methodology in the paper changes the game parameters to enhance quality of experience. Bandwidth adapting technique for NVIDIA GeForce now game streaming is conducted in the research [3]. The study demonstrated the issues of network constraints and defines a video content adaption algorithm keeping in mind the limitations of the network of the user. The study helped in benchmarking in cloud games for researcher and player. A practical experiment is performed to evaluate quality of playing games using a video compression algorithm. In [4] an adaption technique was presented which used player current activity to reduce the streaming bit rate and changing in the encoding parameters of the video sequence.

In [5] the researcher skipped the motion estimation by using the game engine information for speeding up the architecture. In [6], an algorithm was presented to enhance the video encoding time by using the current activity of the player and completely removing the unnecessary objects from the game scene the algorithm the significant improvement was noted in bit rate reduction. The research conducted that the mechanism can achieve better result if size is not changed constantly.

The upcoming researches displays that changing the mechanism can greatly increase the on line cloud video gaming speed. In [7], the research significantly decreases the deformation of the communication channel in cloud gaming. A new request dispatching algorithm was discussed which used game attention model for example when the player is driving a car the main focus is on the whether the fuel will run out or not the rest of the scenes had less significance so the objects which are of less significance were completely removed rather than lowering the quality.

The study [8] suggested a layered approach for the reduction of the cloud gaming bandwidth the cloud server can send enhancement layer information to the client to improve the quality of the base layer. The results in this work suggested the opportunity to extend layered coding with other rendering pipeline design, such as low polygon models, shaders, and global illuminations. An optimized selection and game streaming technique is implemented so that only certain objects are selected and sent to the mobile device while keeping the constraints in mind like download speed and battery limit.

The paper [9] discussed through experimentation that by using a hybrid cloud edge gaming system the latency can be improved in comparison with the conventional cloud gaming architectures. A performance evaluation hybrid cloud gaming edge system is discussed in the paper [10] deploying this method the latency on the cloud gaming client can be reduced According to preliminary simulation results it was observed that when increasing renderers the latency is reduced, however from a certain point forward the reduction rate is significantly reduced as well. The research study [11] the researchers have adopted a progressive adaptive download strategy by which users can download adaptive content for cloud gaming depending on the bit rate requirements. A cognitive decomposed approach to reduce the resource allocation is implemented in [12] the authors increased supporting devices in order to reduce the network bandwidth. Cloud gaming has attracted a lot of gamers in research [13] a cooperative cloud gaming architecture to improve bandwidth is discussed. A visual system approach is applied by using a eye tracker device to only rendered the most important scene in research [14]. Edge computing is used to leverage from 4g in order to reduce bandwidth utilization in [15] edge computing introduces the concept of adaptive bit rate allocation in cloud gaming. Many factors affect the gameplay in cloud gaming in [16] Quality of experience is carried to better understand the contributing factors. Cloud gaming uses the Gpu virtualization technique for gameplay but local gpu can also be utilized for game play [17] which uses the processing power of local gpu to enhance the quality of experience 5g technology will play an important role in cloud gaming future [18] if 5g will be readily available for everyone. The research [19] evaluates the use of QOE-Aware algorithm to improve the gaming speed. The server location plays an important role in quality of gaming for a particular user [20].

3 Dataset and Performance Evaluation

3.1 Dataset

The datasets used for image and video detections include MS-coco dataset trained on 82 classes. Yolo make predictions using set of candidate regions. The detections in yolo utilize features from the entire image globally. Yolo is built for fast detection purposes and running time of fps is equal to 45 fps and yolo tiny of about 155 fps. We are training our custom dataset on yolo tiny v4. The reason for choosing yolo is it is faster as compared to other neural networks. Yolo has been used extensively with imagenet dataset which contains over 9000 object categories. After evaluation of the game scenes we decided to create a custom dataset to improve our detections and only select the objects in the scene that need to be adapted. To fit perfectly for the purpose of cloud gaming. The game scenes contained a lot of background objects that did not serve a high purpose in the task accomplishment by the user.

All the unimportant objects for example trees, signs, background buildings and drums are the main source of high streaming bit rate and high encoding time. After gathering the dataset data augmentation steps are performed on the dataset to increase the amount of images and to improve on our predictions. Those objects which will directly fit in the detection criteria will be selected for example only the objects which had threshold value of 75 percent or more are adapted in the new scene which will be rendered to the client. The dataset accommodate four classes after properly evaluating the game scenes. Class 1 will be trees represented by t1 represented in figure class t1 as shown in Tab. 1.

Table 1: Class representation of each object in the game

Class	Object
T1	Trees
B1	Yellow drums
S1	Sign boards
B2	Background buildings
Stop sign	All stop signs in game
Fire hydrant	Red fire hydrants
Bench	Un-important bench
Phone	Phone booth
S-Lights	Street lights

The second class will contain all the yellow drums which do not play a major role in the task completion by adding bits to the game scene. Class representation is b1 for the yellow drums represented in Fig. 2. Third class being created is the background buildings which do not add much value will the user is playing but adds data to the game scene. We are working with constraint that we need to adapt the scene in accordance with the cloud gaming environment. To reduce the encoding time and streaming bit rate as much as possible. Building class will be represented as b2. After working on the game we adapted the signs which most of the buildings contained. The class was represented as s1 which is based on the signs on the buildings which did not add much value to the task completion. The game environment contained a lot of background buildings which were adding more bits and increasing the data size in the game scene. As, the game is being adapted to fit the need of the cloud gaming environment we added many classes in to our dataset which objects were removed before rendering the game scene to the client device. Mostly the game had objects like traffic cones, lights background buildings, drums which did not add much value to the game experience. So we created all those classes in our dataset.

Before streaming the scene the input tolerance is about 200–300 ms. So during that time frame object detection and adaption is performed. All the adaptations are performed on the class based system. While training yolo v4 tiny with our own custom dataset of four classes we encountered one false positive value in which the class s1 was wrongly classified as b1 class. For the training purpose each class was trained for 2000 iteration and until the class function loss reached to 0.00012.

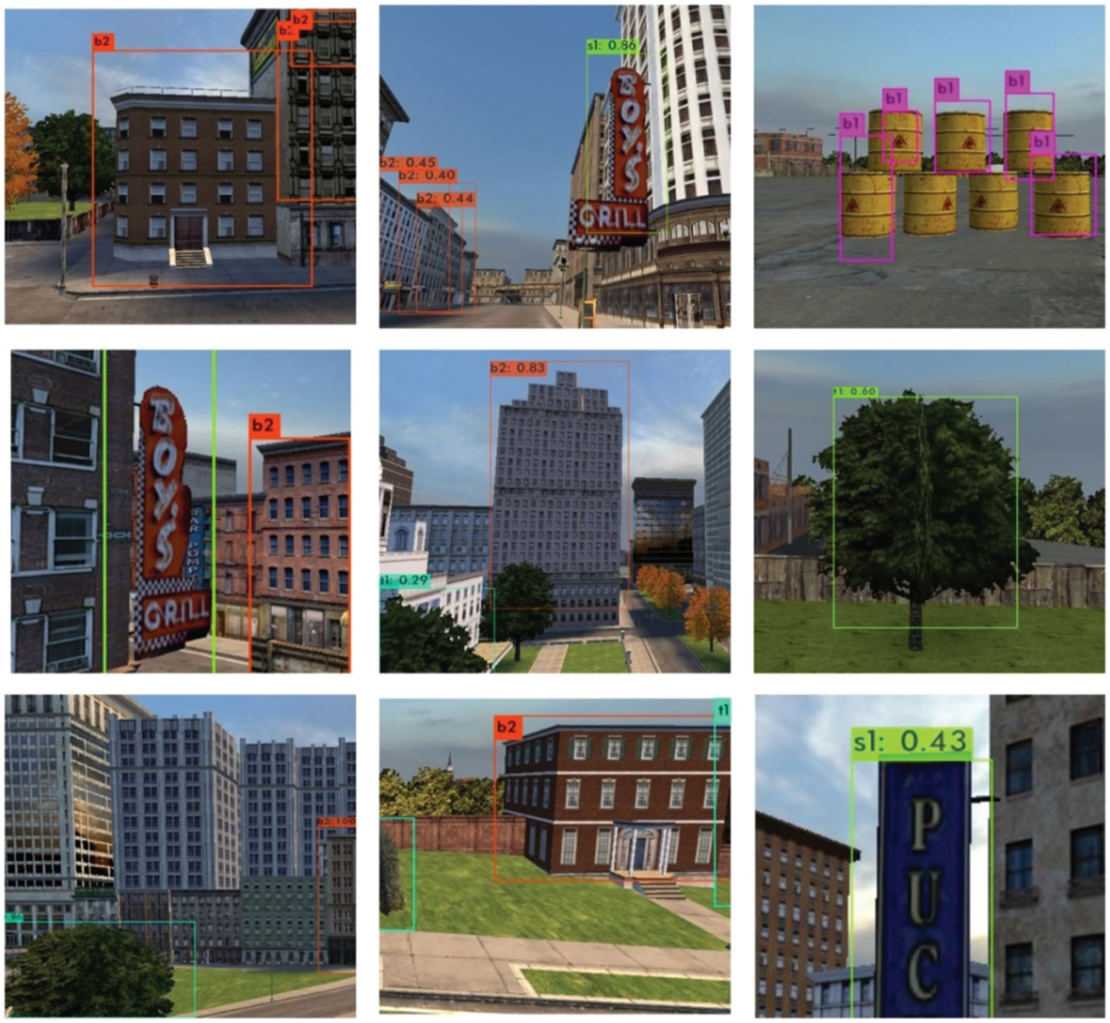

Figure 2: Classes of objects

3.2 Yolo V4 Architecture

There are two types of detector single step and double step detector. Single step detectors have an edge of speed optimality. The predictions are quicker on the single stage detector. Yolo v4 is an enhancement of yolo v3 Referenced in Fig. 3. Two modifications are made in the neck and backbone of the yolo v4 architecture. pre-trained are used to train the backbone. For the mix up augmentation weighted linear interpolation of two existing images is used. We took two images and did linear combination of them. Mix-up reduces the memorization of corrupt labels, increases the robustness. Two images are mixed with weights: λ and $1-\lambda$. λ is generated from symmetric beta distribution with parameter alpha.

$$\hat{x} = \lambda x_i + (1-\lambda)x_j$$
$$\hat{y} = \lambda y_i + (1-\lambda)y_j$$
$$\text{where } \lambda \in [0, 1] \text{ is a random number} \quad (1)$$

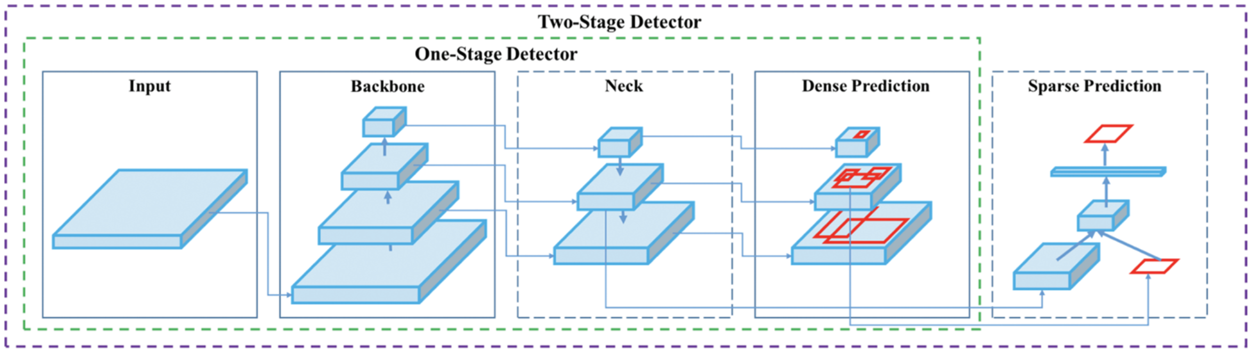

Figure 3: Object detection two stage detector

3.3 Yolo V4 Design

The yolo v4 is made up of three different parts backbone, neck and head.

a) Backbone

In the backbone CSPDarknet53 is used as a feature extractor model which uses GPU for extraction.

b) Neck

In yolo v4 a spatial pyramid pooling and Path aggregation network (PAN) is used. PAN is a modified version as compared to previously used.

c) Head

In Yolo v4 same head is used as the yolo v3 dataset Manipulation can be expressed as following

1) Backbone (detector)

Yolo v4 back bone architecture mainly consists of three parts 1) bag of freebies 2) bag of specials 3) *CSPDarknet53*.

The bag of freebies methods are utilized to increase or decrease the training data. We have used data augmentation to increase the size of our dataset by adding different parameters to change size shape and orientation of the images. Different types of data augmentations are applied for photometric distortion, Geometric distortion and mix up

In our research we are using Yolo v4 because CSPDarknet53 shows better results in detecting objects than CSPReNext50.

3.4 Photometric Distortion

Photometric distortion is used to create different sets of images by changing the brightness, contrast in order to display the same image in many different ways.

3.5 Geometric Distortion

The geometric distortion technique is used to change shape size and rotate the images.

3.6 Focal Loss

Focal loss is created to address the one stage scenario due to extreme imbalance in classes during training. Focal loss function is based on the cross entropy by introducing a (1 − pt)^gamma coefficient as shown in Fig. 4. This coefficient allows focusing the importance on the correction of misclassified in Eq. (2).

$$\begin{aligned} &\mathrm{CE}\,(p_{\mathrm{t}}) = -\log\,(p_{\mathrm{t}}) \\ &\mathrm{FL}\,(p_{\mathrm{t}}) = -\,(1-p_{\mathrm{t}})^{\gamma}\log\,(p_{\mathrm{t}}) \end{aligned} \tag{2}$$

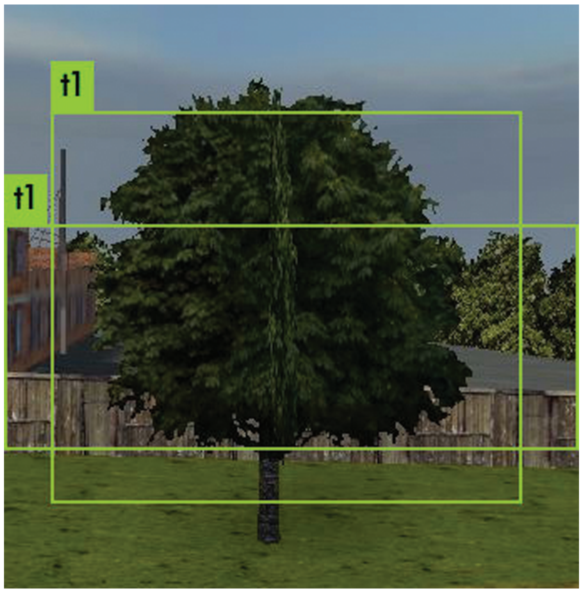

Figure 4: IOU loss

3.7 IoU Loss

Most detection models use bounding box to predict of an item. But for equality evaluation L2 standard is used. L2 standard minimizes the errors on large bounding boxes. This problem is addressed using Iou Loss for more accurate bounding boxes in Eq. (3).

$$x_t = \begin{bmatrix} \text{Ground truth}: \tilde{x} = \left(\tilde{x}_t, \tilde{x}_b, \tilde{x}_l, \tilde{x}_r\right) & \\ \text{Prediction}: x & = (x_t, x_b, x_l, x_r) \\ x_b & \\ \cdot\ell_2\ \text{loss} = \| x_t - x_r \|_2^2 & \\ \cdot\text{IoU loss} = -\ln \dfrac{\text{Intersection}\,(x_t, x_r)}{\text{Union}\,(x_b, x_l)} & \end{bmatrix} \tag{3}$$

Compared to the l2 loss, we can see that instead of optimizing four coordinates independently, the IoU loss considers the bounding box as a unit. Thus the IoU loss could provide more accurate bounding box prediction than the l2 loss.

3.8 CSPDarknet53

Dense Net builds using previous input and synchronization with current input before moving to a compact layer. Each layer of the Dense Net phase consists of a dense block and a transition layer, and each dense block is composed of dense layers k as shown in Fig. 5. Dense layer removal will be combined with dense layer insertion, and the combined result will be dense layer insertion (i + 1) in Eq. (4).

$$\begin{aligned} x_1 &= w_1^* x_0 \\ x_2 &= w_2^* [x_0 x_1] \\ &\vdots \\ x_k &= w_k^* [x_0, x_1, \ldots, x_{k-1}] \end{aligned} \tag{4}$$

2) Neck (detector)

The purpose of the neck is to collect feature maps from different layers.

3.9 SPP

The fully connected network requires a fixed size so we need to have a fixed size image, when detecting objects we don't necessarily have fixed size images. The problems cause the images to upscale, the output of CNN we have our feature map these features are created by different filters which can detect circular geometric shapes. We have a filter able to detect circular shapes, this filter will produce a feature amp highlighting. The spatial pyramid pooling layer allows creating fixed length features relevant to our feature maps. The components of neck flow down and up among all the layers and connect only minor layers of the convolutional network.

3.10 Head (Detector)

The role of the head in the case of a one stage detector is to perform dense prediction.

Bag of freebies.

Loss of CIoU introduces two new concepts compared to Loss of IoU. The first concept is the concept of intermediate points, which is the distance between the point of the actual binding institution and the point of the binding center of the obligation in Eq. (5).

$$\mathcal{L}_{CIoU} = 1 - IoU + \frac{\rho^2\left(\mathbf{b}, \mathbf{b}^{gt}\right)}{c^2} + \alpha v \tag{5}$$

b and bgt mean points between B and Bgt, $\cdot$ $(\cdot)$ Euclidean distance, c sellable length of the smallest closure box covering two boxes, α is a positive trading parameter, and v measures the consistency of aspect ratio in Eq. (6).

$$v = \frac{4}{\pi^2}\left(\arctan\frac{w^{gt}}{h^{gt}} - \arctan\frac{w}{h}\right)^2 \tag{6}$$

W presents the height and width of the bounding box.

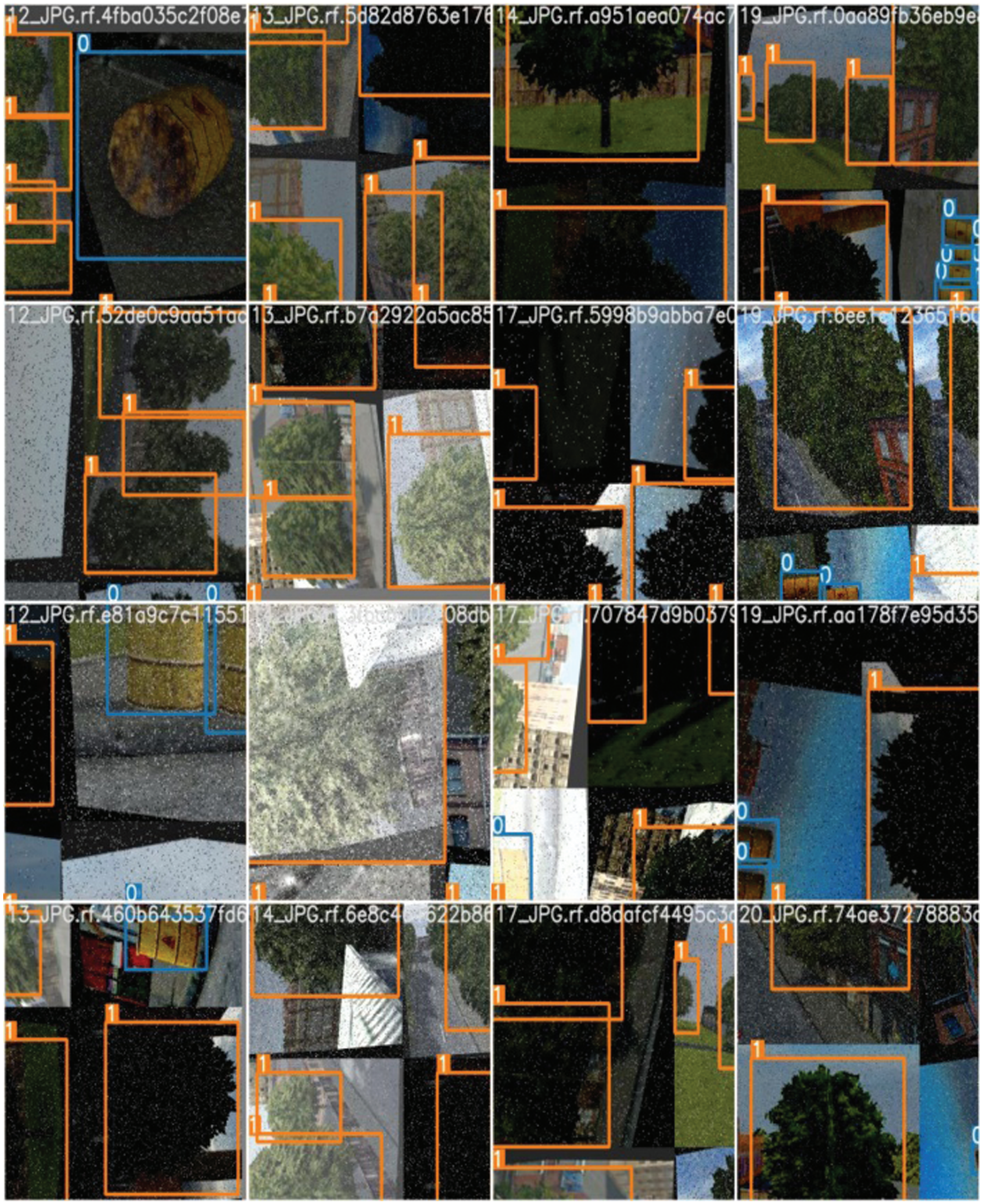

Figure 5: Mosaic data augmentation

3.11 CmBN (Cross mini Batch Normalization)

Performing Batch Normalization does not work when the size of the batch becomes smaller. Standard deviation estimates and meanings are reduced by sample size.

To solve this problem, Taylor's polynomials are used to measure any work that is permanently divided in Eq. (7).

$$f(x) = f(c) + f'(c)(x-c) + \frac{f''(c)}{2!\,(x-c)^2} + \cdots + \frac{f^{(n)}(c)}{n!\,(x-c)^n} + R_n(x) \tag{7}$$

The classic way to normalize a batch in Eq. (8):

$$\hat{x}_{t,i}(\theta_t) = \frac{x_{t,i}(\theta_t) - \mu_t(\theta_t)}{\sqrt{\sigma_t(\theta_t)^2 + \epsilon}} \tag{8}$$

The mini batch normalization is defined in Eq. (9):

$$\hat{x}^l_{t,i}(\theta_t) = \frac{x^l_{t,i}(\theta_t) - \overline{\mu}^l_{t,k}(\theta_t)}{\sqrt{\overline{\sigma}^l_{t,k}(\theta_t)^2 + \epsilon}} \tag{9}$$

3.12 Mosaic Data Augmentation

Mosaic data augmentation combines the training images into one ratio. The helps model to learn to identify object at a smaller scale. It also increases the model accuracy into localize different types of images in different parts of the frame in Fig. 5.

3.13 DIoU-NMS

NMS (Non-Maximum Suppression) is used to remove the boxes that represent the same object while keeping the one that is the most precise compared to the real bounding box in Eq. (10).

$$\mathcal{R}_{DIoU} = \frac{\rho^2\left(\mathbf{b}, \mathbf{b}^{gt}\right)}{c^2} \tag{10}$$

where b and bgt denote the central points of B and Bgt, $\rho(\cdot)$ is the Euclidean distance, and c is the diagonal length of the smallest enclosing box covering the two boxes.

4 Object Detection and Adaption Algorithm

The major thought supporting this research paper is to render the functional 3 Dimensional game object on the cloud provider instead of rendering it on the end user system. User is mainly focused on objects that get the task accomplished and keep the overall gaming experience enjoyable. An adapted video of the game scene is transmitted to the end user where only the relevant objects are displayed with updated game messages and updated controls. The list which is created of the important objects is dynamic and is based on the objects detected and according to the player's actions in the current game scene. Important objects like avatar, gun, and enemy are given high priority.

5 Experiments and Analysis

This article identification strategy is utilized and applied on well-known open world game mafia city of lost heaven.

5.1 Object Detection on Mafia 1 City of Lost Heaven

The model of Yolo v4 has been applied on the frame of a random gameplay video and the model detects the main features in the scene. Model has identified the main objects in the image as our main purpose is to detect the main objects of our game and then apply video encoding on that detected video sequence to increase the game efficiency in less optimal internet conditions.

The game was played for 30 s and recorded by using movavia cam. Game resolution was set at 1024×920 pixels. The frame rate was 45 fps. As it is clear in the redone sequence the macro blocks present in the adapted scene are significantly less as compared to original scene for example the drums which can be seen has a threshold values of 98%. So the drum sets perfectly in our criteria for adaption. So, the drum marked with class name b1 has been excluded from the scene as shown in Fig. 6. The encoded scene has much less macro blocks and the overall encoding time is reduced by using the detection algorithm.

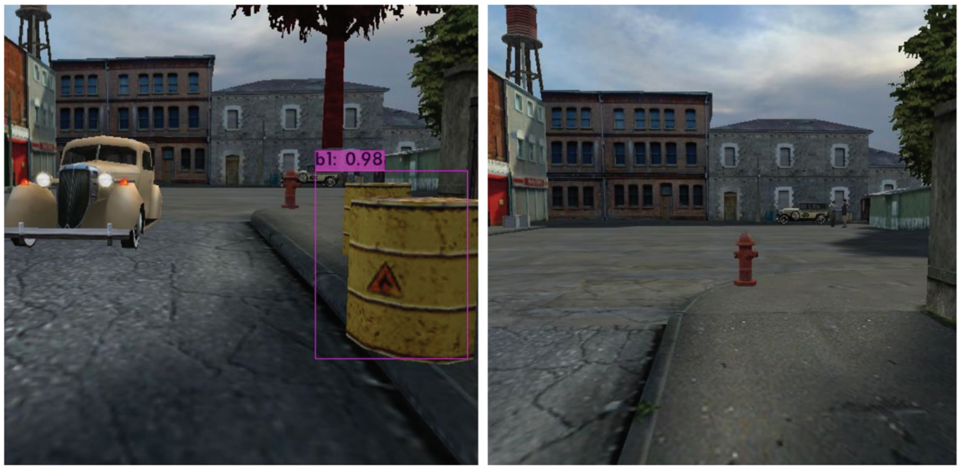

Figure 6: Game scene before and after removal

The second video scene was selected in the mission two. The environment in mission two is mostly large buildings. So, we performed our adaptation algorithm on the video scene. Detections were made on the basis of threshold values of more than 75 percent. Two main background items were detected. The scene contains fewer amounts of pixels. The task accomplishment and story line is not affected in any manner of the game scenes a shown in Fig. 7.

5.2 Object Detection on Mafia 1 Mission Little Italy

In the video sequence of the game the yolo v4 model has detected the objects in the game sequence and all those objects that are relevant which play a role in the game. The adapted scene which displays a scene computationally less complex as compared to original scene but the task accomplishment is not disturbed in any manner.

A lot of buildings and trees are encountered during the game play. These objects are adding a lot data into the game scene. Objects which are detected in the detection layer are tree which are constantly repeating. The trees were detected with high threshold value of 91 percent as shown in Fig. 8.

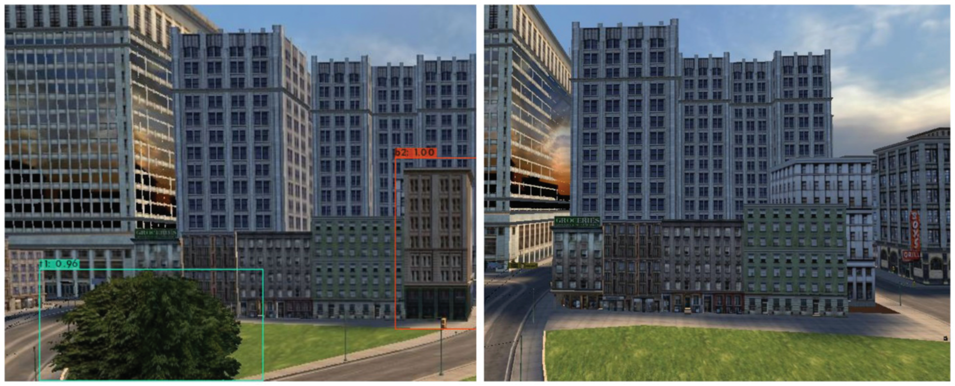

Figure 7: Object detection on game video sequence

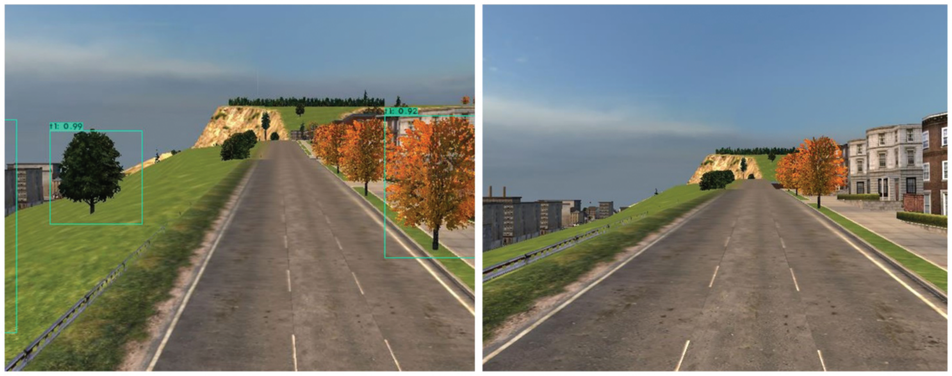

Figure 8: Object detection on game video sequence

6 Experimental Results

The research results will be evaluated on the basis of encoding time, Streaming bit rate and size of the video stream in Tab. 2.

Table 2: Before adaptation of game scenes

	Mission 1	Mission 2	Mission 3
Encoding time	50.4 ms	52.4 ms	54.4 ms
Streaming bit rate	2211 Kbps	2457 Kbps	2266.5 Kbps
Video size	7.55 Mb	9.2 Mb	8.1 Mb

The results of all the three scenes are generated in a controlled environment. The input delay tolerance for open world games like the one we selected is about 200 to 300 ms as shown in Tab. 3. While making our object detections based on our custom dataset time will be added before encoding process. The yolo v4 layer added about 16.3 ms in scene 1, 17 ms were added in detecting objects in scene two and 14.5 ms was added in scene three. After adding the time in the scene one total time taken on the server side will be 42.1 ms, scene two will be 44.8 ms and the third scene will be 42.3 ms.

Table 3: After adaptation of game scenes

	Mission 1	% Decrease	Mission 2	% Decrease	Mission 3	% Decrease
Encoding time	42.1 ms	16.46	44.8 ms	14.50	42.3 ms	22.24
Streaming bit rate	1201 Kbps	45.66	1529 Kbps	38.2	1365 Kbps	39.7
Video size	4.43 Mb	42.8	5.6 Mb	39.1	5 Mb	38.2

As for the streaming bit rate the results will be very high as the streaming phase is on the third step of the system design it will not be affected by the yolo v4 detection time. Delay tolerance is the time it takes for the server and the user to interact. We are evaluating our system for an open world game; hence we had a bigger bracket of time to run our detections and adaptions on the game scene. Our system will perform even better with omnipresent games as delay tolerance is about 1000 ms in those types of games in Tab. 4. Keeping in mind the constraints in cloud gaming the streaming bit rate and decreased video size will greater enhance user ability. While adapting our game scenes we are keeping in mind that task accomplishment should not be affected at any moment of the game play.

Table 4: Delay tolerance for different genre games

	Type of game	Latency
First person shooter	First person	100 ms
Open world games	Third person	200–300 ms
Strategy games	Omnipresent	1000 ms

6.1 *Graphical Representation of Encoding Time Improvement*

The Fig. 9 Shows the encoding time reduction before and after the game scene detection and removing of objects.

6.2 *Streaming Bit Comparison*

The Fig. 10 shows the streaming bit rate before and after adapting the game scene.

7 Simulations

The game was played on parsec cloud hosting service which lets user connect to other devices and host their games for the Devices which have lower hardware and software capabilities. After adaptation of the game scene the game was rendered to the client device in our scenario it is the second laptop which is operating without the gpu before adaption we were achieving 30–45 fps. After smart removal we achieved 50–55 fps on average.

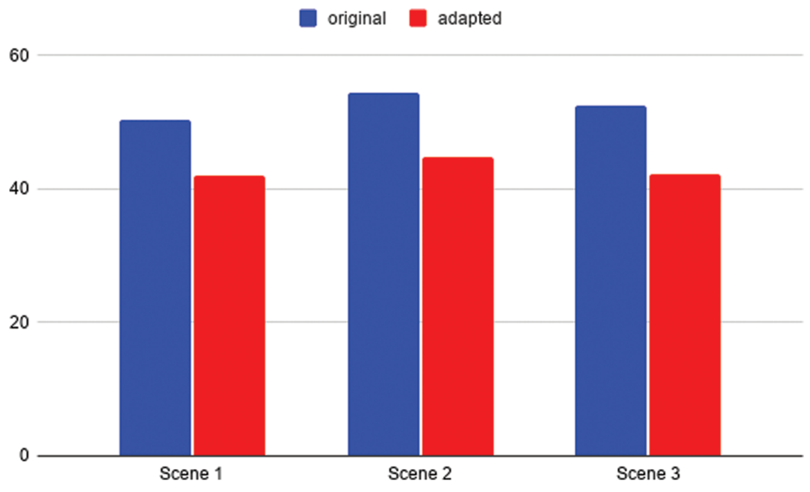

Figure 9: Encoding time reduction for before and after game scene

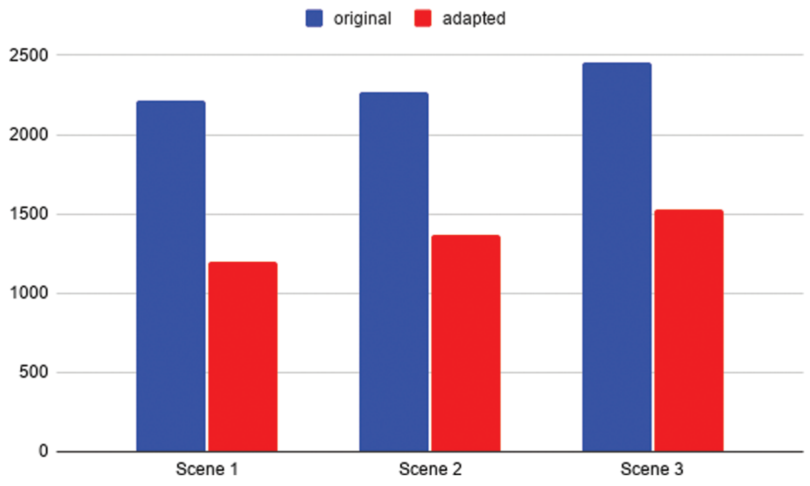

Figure 10: Streaming bit rate reduction for before and after game represents adapted scene

8 Conclusion

Object detection and adaption technique was proposed to reduce bandwidth (bitrate) and encoding time for cloud gaming system improvement. Items in the gaming scene which were less important for the overall game play experience were captured and removed for better results. Exploratory outcomes demonstrated that the proposed method prevails to decrease both bandwidth and encoding time by 45.6% and 16.46% separately in mission one. In the second mission, encoding time is reduced by 14.5% and the streaming bit rate is reduced by 38.2%. The last mission achieved a reduction of 39.7% in streaming bit rate and the encoding time was reduced to 22.24%. After adding the time used by yolo v4 tiny the encoding time is still less as compared to other techniques. The detection phase of the Yolo v4-tiny added about 17.7 ms. As we selected an open-world game that has about 500 ms delay tolerance bracket. The algorithm used will add about 17.7 ms while using yolo v4 to detect those objects. In the results phase 17 ms is added in to each scene result.

Acknowledgement: Thanks to our families and colleagues, who provided moral support.

Funding Statement: The authors received no specific funding for this study.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding this study.

References

[1] G. Castañé, H. Xiong, D. Dong and J. Morrison, "An ontology for heterogeneous resources management interoperability and HPC in the cloud," *Future Generation Computer Systems*, vol. 88, no. 7, pp. 373–384, 2018.

[2] B. Bello and M. Aritsugi, "A transparent approach to performance analysis and comparison of infrastructure as a service providers," *Computers and Electrical Engineering*, vol. 69, no. 23, pp. 317–333, 2018.

[3] A. Bochkovskiy, C. Wang and H. Liao, "Yolov4: Optimal speed and accuracy of object detection," *International Journal of Advanced Computer Science and Applications*, vol. 9, no. 2, pp. 94–105, 2020.

[4] M. Suznjevic and M. Homen, "A new triple bargaining game-based energy management scheme for hierarchical smart grids," *IEEE Access*, vol. 7, pp. 161131–161140, 2020.

[5] R. Li, Q. Zheng, X. Li and Z. Yan, "Multi-objective optimization for rebalancing virtual machine placement," *Future Generation Computer Systems*, vol. 105, no. 6, pp. 824–842, 2020.

[6] S. Zahra, M. Khan, M. Ali and S. Abbas, "Standardization of cloud security using mamdani fuzzifier," *International Journal of Advanced Computer Science and Applications*, vol. 9, no. 3, pp. 292–297, 2018.

[7] N. S. Naz, S. Abbas, M. A. Khan, B. Abid and N. Tariq, "Cloud gaming: Architecture and performance," *International Journal of Advanced Computer Science and Applications*, vol. 10, no. 3, pp. 569–577, 2019.

[8] V. Emilia and Balas, "Performance analysis of IoT services based on clouds for context data acquisition," *Computational Intelligence and Complexity*, vol. 786, pp. 81–92, 2018.

[9] C. Seong and N. Cheung, "Bandwidth efficient mobile cloud gaming with layered coding and scalable phone lighting," in *IEEE Int. Conf. on Image Processing*, Paris, France, pp. 606–611, 2014.

[10] D. Logofatu, F. Leon and F. Muharemi, "General video game ai controller-integrating three algorithms to bring a new solution," in *23rd Int. Conf. on System Theory, Control and Computing*, Sinaia, Romania, pp. 856–859, 2019.

[11] F. Chi, X. Wang, W. Cai and V. M. Leung, "Ad-hoc cloudlet based cooperative cloud gaming," *IEEE Transactions on Cloud Computing*, vol. 6, no. 3, pp. 625–639, 2018.

[12] W. Cai, H. Chan, X. Wang and V. Leung, "Cognitive resource optimization for the decomposed cloud gaming platform," *IEEE Transaction Circuits System Video Technology*, vol. 25, no. 12, pp. 2038–2051, 2015.

[13] L. Wang, Y. Ma, J. Yan, V. Chang and A. Y. Zomaya, "Toward multiplayer cooperative cloud gaming," *IEEE Cloud Computing*, vol. 5, no. 5, pp. 70–80, 2018.

[14] G. K. Illahi , T. V. Gemert , M. Siekkinen, E. Masala and A. Oulasvirta, "Cloud gaming with foveated video encoding," *ACM Transactions on Multimedia Computing, Communications and Applications*, vol. 16, no. 1, pp. 1–24, 2020.

[15] Xin Zhang, H. Chen, Y. Zhao and Z. Ma, "Improving cloud gaming experience through mobile edge computing," *IEEE Wireless Communications*, vol. 26, no. 1, pp. 178–183, 2019.

[16] S. Sabet, S. Schmidt and S. Zadtootaghaj, “Delay sensitivity classification of cloud gaming content,” in *Proc. of the 12th ACM Int. Workshop on Immersive Mixed and Virtual Environment Systems*, Istanbul Turkey, pp. 25–30, 2020.

[17] H. Chen, X. Zhang, Y. Xu, Ju Ren and J. Fan, “T-gaming: A cost-efficient cloud gaming system at scale,” *IEEE Transactions on Parallel and Distributed Systems*, vol. 30, no. 12, pp. 2849–2865, 2019.

[18] O. S. P. Pulla, C. Baena, S. Fortes, E. Baena and R. Barco, “Measuring key quality indicators in cloud gaming: Framework and assessment over wireless networks,” *Journal on the Science and Technology of Sensors*, vol. 21, no. 4, pp. 1387–1401, 2021.

[19] I. Slivar, L. S. Kapov and M. Suznjevic, “QoE-aware resource allocation for multiple cloud gaming users sharing a bottleneck link,” in *22nd Conf. on Innovation in Clouds, Internet and Networks and Workshops*, Paris, France, pp. 2849–2865, 2019.

[20] Y. Deng, Y. Li, R. Seet, X. Tang and W. Cai, “The server allocation problem for session-based multiplayer cloud gaming,” *IEEE Transactions on Multimedia*, vol. 20, no. 5, pp. 1233–1245, 2018.