



MODELING AND EVALUATION OF ADAPTIVE 360-DEGREE VIDEO STREAMING WITH TILES

HENRIQUE DOMINGUES GARCIA

**DOCTORAL DISSERTATIONS
EM ELECTRICAL ENGINEERING**

ELECTRICAL ENGINEERING DEPARTMENT

FACULDADE DE TECNOLOGIA

UNIVERSIDADE DE BRASÍLIA

**Universidade de Brasília
Faculdade de Tecnologia
Electrical Engineering Department**

**Modeling and Evaluation of Adaptive 360-Degree Video Streaming
with Tiles**

Henrique Domingues Garcia

**DOCTORAL DISSERTATIONS SUBMETIDA AO PROGRAMA DE PÓS-GRADUAÇÃO
EM ENGENHARIA ELÉTRICA DA
UNIVERSIDADE DE BRASÍLIA COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE DOCTOR.**

APROVADA POR:

**Nome, Título (Instituição)
(Orientador)**

**Nome, Título (Instituição)
(Co-orientador)**

**Nome, Título (Instituição)
(Examinador Externo)**

**Nome, Título (Instituição)
(Examinador Externo)**

**Nome, Título (Instituição)
(Examinador Interno)**

Brasília/DF, Abril de 2024.

FICHA CATALOGRÁFICA

GARCIA, H. D.

Modeling and Evaluation of Adaptive 360-Degree Video Streaming with Tiles. [Brasília/DF] 2024.

xxx, nnnp., 210 x 297 mm (ENE/FT/UnB, Doctor, Doctoral dissertations, 2024).

Universidade de Brasília, Faculdade de Tecnologia, Electrical Engineering Department.

Electrical Engineering Department

- | | |
|---------------|--------------------|
| 1. Keyword | 2. Keyword |
| 3. Keyword | 4. Keyword |
| 5. Keyword | 6. Keyword |
| 7. Keyword | 8. Keyword |
| I. ENE/FT/UnB | II. Título (série) |

REFERÊNCIA BIBLIOGRÁFICA

GARCIA, H. D. (2024). Modeling and Evaluation of Adaptive 360-Degree Video Streaming with Tiles. Doctoral dissertations, Publicação PPGEE.XXXXX/2024, Electrical Engineering Department, Universidade de Brasília, Brasília, DF, xxxxpx.

CESSÃO DE DIREITOS

AUTOR: H. D. Garcia

TÍTULO: Modeling and Evaluation of Adaptive 360-Degree Video Streaming with Tiles.

GRAU: Doctor

ANO: 2024

É concedida à Universidade de Brasília permissão para reproduzir cópias deste Doctoral dissertation e para emprestar ou vender tais cópias somente para propósitos acadêmicos e científicos. O autor reserva outros direitos de publicação e nenhuma parte deste trabalho pode ser reproduzida sem autorização por escrito do autor.

H. D. Garcia

Universidade de Brasília (UnB)

Campus Darcy Ribeiro

Faculdade de Tecnologia - FT

Electrical Engineering Department(ENE)

Brasília - DF CEP 70919-970

Para Clarisse.

AGRADECIMENTOS

Obrigado!

ABSTRACT

Regardless of the idiom chosen, the Abstract in English must be provided.

Keywords: Keywords go here.

RESUMO

Mesmo que a Tese ou Dissertação seja redigida em inglês, é necessário fornecer o resumo em português.

Palavras-chave: Escrever palavras-chave.

SUMÁRIO

Sumário	i
Lista de Figuras	iii
Lista de Tabelas	iv
Lista de Símbolos	v
Glossário	vi
Capítulo 1 – Introdução	1
Capítulo 2 – Modelagem do sistema: 360 video streaming	5
2.1 O vídeo esférico	5
2.1.1 video360tools	5
2.1.2 Projeções	8
2.1.2.1 Projeção Equirretangular	8
2.1.2.2 Projeção Cubemap	9
2.1.3 O Viewport	10
2.2 Qualidade objetiva do vídeo	12
2.2.1 Qualidade para o cliente	12
2.2.2 Qualidade para o servidor	13
2.2.2.1 MSE/PSNR	15
2.2.2.2 S-MSE/S-PSNR	16
2.2.2.3 WS-MSE/WS-PSNR	16
2.2.3 Estatísticas	17
Capítulo 3 – Avaliação e resultados	19
3.1 Caracterização dos elementos do servidor	20
3.2 Caracterização dos elementos do cliente	26
3.3 Cenários avaliados	28

3.3.1	Estatísticas dos ladrilhos por ladrilhamento	28
3.3.2	Estatísticas por nível de qualidade e ladrilhamento	30
3.3.3	Estatísticas dos <i>Chunks</i> para o envio de todos os ladrilhos	31
3.3.4	Ladrilhos vistos (por viewport)	32
Capítulo 4 – Conclusão e Trabalhos Futuros		34
4.1	Trabalhos futuros	35
4.2	Cronograma	35
References		36

LISTA DE FIGURAS

1.1 (a) Viewport em um streaming monolítico: desperdício ao requisitar pixels não vistos (em vermelho). (b) Um vídeo ladrilhado no formato 4×4 com região de interesse (em verde) e viewport.	2
2.1 Os Sistemas de Coordenadas	6
2.2 Projeção Equirretangular mapeada no plano da imagem.	8
2.3 Projeção Cubemap e a disposição das seis faces no plano da imagem.	9
2.4 Extração do viewport usando projeção retilinear ou gnomônica.	10
2.5 Avaliação de qualidade objetiva do viewport.	13
2.6 Tiles que deverão ser solicitados ao servidor e que serão assistidos durante um <i>chunk</i>	14
2.7 Avaliação de qualidade objetiva da projeção.	14
3.1 Fluxograma para captura de métricas de desempenho.	20
3.2 Dispersão do SI e TI para as projeções avaliadas	23
3.3 Mecanismo para seleção de ladrilhos.	27
3.4 Mapa da velocidade angular média de todos os voluntários para todos os grupos de vídeo. Extraído de (NASRABADI <i>et al.</i> , 2019)	28
3.5 Boxplot do tempo de decodificação, taxa de bits e erro organizado por ladrilhamento.	30
4.1 Modelo de simulação que será usado para avaliação	35

LISTA DE TABELAS

2.1	Tabela de coordenadas (X, Y, Z) em função de (U, V)	10
2.2	Tabela de condições e valores de $face, u, v$ em função de (X, Y, V)	10
3.1	Vídeos usados no experimento	21
3.2	Detalhes do ladrilhamento e resoluções resultantes.	24
3.3	Parâmetros de codificação.	25
3.4	Valores de média e desvio padrão das métricas analisadas de acordo com o la- drilhamento, abrangendo todas qualidades.	29
3.5	Correlação entre o tempo de decodificação do bloco e a taxa de bits do bloco. . .	31

LISTA DE SÍMBOLOS

ε_r Relative electric permittivity [p.u.]

GLOSSÁRIO

3LPE Three-layer polyethilene

CAPÍTULO 1

INTRODUÇÃO

A recente divulgação do novo óculos de realidade virtual (em inglês HMD - Head Mounted Display) da Apple impulsiona mais uma vez o mercado de aplicações de realidade virtual (em inglês VR - Virtual Reality), juntando-se a dispositivos como o popular Oculus Meta Quest 3 da Meta. Além destes dispositivos tudo-em-um, é possível utilizar o próprio celular acoplado em um suporte especial na cabeça, como o Google Cardboard e o Samsung GearVR, para facilmente usufruir de aplicações VR, como jogos e vídeos 3D e esféricos.

De todas as aplicações, o tráfego de vídeo esférico é um dos mais exigentes em termos de largura de banda. Nesta aplicação, o vídeo é reproduzido na casca interna de uma esfera e o usuário é posto no centro. Usando informações dos sensores de movimento do HMD, o usuário tem liberdade para olhar ao redor movendo a cabeça em três graus de liberdade.

Atualmente, o streaming de vídeos esféricos depende principalmente da arquitetura de streaming de vídeo 2D tradicional. Assim, toda a esfera de vídeo precisa primeiro ser projetada em um plano usando alguma técnica de projeção (por exemplo, equirretangular ou cubemap) antes de ser comprimida com um codificador de vídeo 2D padrão, como H.264 ou VP9. Desta forma, para que o vídeo no HMD exiba uma boa qualidade, a projeção do vídeo deve possuir uma resolução de pelo menos 4K (2160 linhas), gerando alta taxa de transmissão, exigindo grande capacidade de processamento e armazenamento (AFZAL *et al.*, 2017). Codificadores tradicionais como H.264 geralmente não são usados para resoluções maiores que 4K, sendo necessário também novas tecnologias como o codificador H.265 para a compressão de vídeos em resoluções superiores.

Aplicativos como YouTube e Facebook geralmente usam as projeções equirretangular (do inglês ERP - Equirectangular Projection) ou projeção CubeMap (do inglês CMP - CubeMap Projection), codificados com H.264 ou VP9 e empregam o protocolo aberto da MPEG Dynamic Adaptive Streaming over HTTP (DASH). No DASH o servidor disponibiliza o vídeo

em segmentos de duração fixa, geralmente de 4 a 8 segundos de vídeo, codificados em múltiplas qualidades/taxas de bits (quanto maior a qualidade, maior a taxa de bits do vídeo). No lado do cliente, o aplicativo utiliza um algoritmo de Adaptação da Taxa de Bits (do inglês ABR - Adaptive Bitrate) para decidir qual a melhor qualidade considerando os recursos do dispositivo, como largura de banda disponível, resolução da tela, capacidade de processamento, decodificador disponível, bateria, etc.

No entanto, ao contrário dos vídeos 2D, um usuário só pode visualizar uma fração de todo o vídeo esférico devido à limitação do campo de visão humano (FOV), conforme mostrado na Figura 1.1(a). Normalmente, um HMD tem um FoV de $120^\circ \times 90^\circ$, o que corresponde a apenas 16,6% da esfera (AFZAL *et al.*, 2017). Consequentemente, vários recursos do sistema serão desperdiçados se todo o vídeo esférico for transmitido (também conhecido como streaming "monolítico").

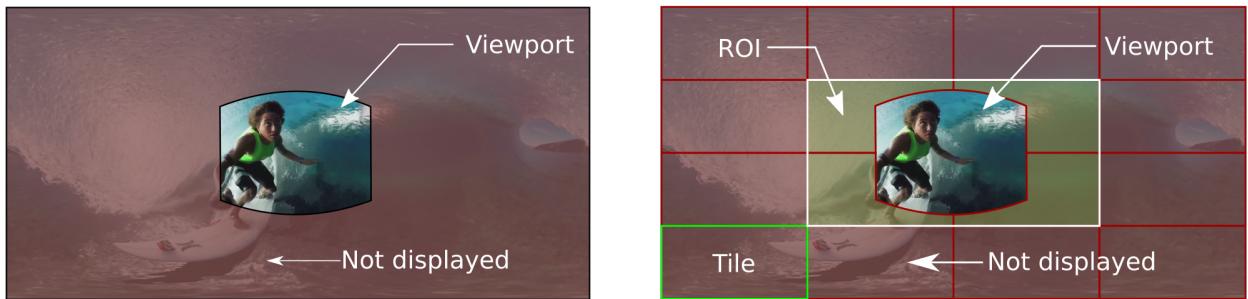


Figura 1.1. (a) Viewport em um streaming monolítico: desperdício ao requisitar pixels não vistos (em vermelho). (b) Um vídeo ladrilhado no formato 4×4 com região de interesse (em verde) e viewport.

Para economizar recursos, vários trabalhos (ZARE *et al.*, 2016; GRAF *et al.*, 2017; HOSSEINI; SWAMINATHAN, 2017; XIAO *et al.*, 2018) adotaram uma abordagem adaptativa à janela de visualização para o streaming de vídeos de 360° . Neste caso, a projeção do vídeo é segmentada espacialmente em ladrilhos independentes, de modo que o cliente possa solicitar apenas aqueles que abrangem as viewports previstas para os próximos segundos. A figura 1.1(b) apresenta esta abordagem utilizando um padrão de ladrilho 4×4 com um ROI e uma janela de visualização correspondente.

Nessa linha, Qian et al. (QIAN *et al.*, 2018) propuseram uma solução baseada em ladrilhos voltada para dispositivos móveis comuns. O algoritmo ABR proposto otimiza o número e o tamanho (ou seja, a qualidade) dos blocos a serem solicitados com base em uma restrição que leva em consideração o diversas características do vídeo, estimados a partir de médias

de amostra avaliadas em tempo de execução, entre outros parâmetros. Consequentemente, a restrição sobre a qual uma decisão ótima é tomada assume que um único valor “típico” do tempo de decodificação do bloco é válido para todos os tamanhos pesquisáveis de blocos (isto é, qualidades). Tal suposição pode gerar problemas de ocupação do *buffer* que podem levar a interrupções indesejáveis na reprodução. Portanto, embora a solução do sistema dos autores ofereça um excelente desempenho, acreditamos que seu trabalho também lançou alguma luz sobre a necessidade de investigar mais detalhadamente as características dos ladrilhos. Na verdade, o status de ocupação do *buffer* é fundamental no projeto de algoritmos ABR (HUANG *et al.*, 2014), e entender seu comportamento por meio de filas ou teoria de controle provou ser uma ferramenta poderosa (HUANG *et al.*, 2014; YIN *et al.*, 2015; SPITERI *et al.*, 2016; YADAV *et al.*, 2017) . Para ajudar nisso, é necessário algum conhecimento estatístico do comportamento do tráfego de entrada/saída do *buffer* de um cliente. Embora considerável atenção na literatura tenha sido dada à caracterização da largura de banda da rede em vídeos esféricos com ladrilhos, nenhum trabalho anterior tratou da caracterização do modelo de transmissão de vídeo que correlacione a qualidade percebida pelo cliente com a qualidade das projeções codificadas no servidor.

Neste trabalho apresentamos a modelagem de um sistema de transmissão de vídeo 360 graus codificado com HEVC, considerando diferentes tipos de segmentação espacial e qualidade em streaming de vídeo sob demanda com DASH. Estimamos a da função de densidade com base em uma série de experimentos realizados em um conjunto de vídeos de 360° com diferentes características espaciais, temporais e observacionais. Além disso, investigamos até que ponto o tempo de decodificação do bloco está correlacionado com a taxa de bits do bloco (no nível do bloco), para que o streaming de vídeo baseado em DASH possa possibilitar o uso de tal informação para otimizar a transmissão de vídeo. Esses dados podem ser usados para emular a reprodução de uma sessão de vídeo 360 para que possamos analisar o desempenho dos algoritmos ABR.

A metodologia deste trabalho envolve o cálculo de métricas estatísticas de sistemas de transmissão de vídeo, que serão utilizadas para a avaliação do desempenho de streaming de vídeo em 360 graus para diferentes algoritmos de adaptação de taxa de bits (ABR). As métricas consideradas incluem tempo de decodificação, taxa de transmissão e qualidade medida em MSE

e outras métricas de erro próprias para vídeo esféricos. O modelo é utilizado para emular um usuário assistindo a uma seção de vídeo, capturando traços de rede para avaliação de desempenho e otimizando a qualidade da transmissão de forma eficiente.

O restante deste estudo está organizado da seguinte forma: No Capítulo 2, é realizada uma revisão do estado da arte para sistemas de transmissão de vídeo de 360 graus com vídeo segmentado baseado em ladrilhos, considerando mecanismos de seleção de blocos, previsão de janela de visualização e bancos de dados disponíveis. O capítulo 3 concentra-se na modelagem dos objetos e métricas considerados em sistemas de transmissão. O capítulo 4 relata as análises de diferentes cenários e, por fim, no capítulo 5 são apresentadas as conclusões do trabalho desenvolvido e as propostas de trabalhos futuros para a continuação desta pesquisa.

CAPÍTULO 2

MODELAGEM DO SISTEMA: 360 VIDEO STREAMING

2.1 O VÍDEO ESFÉRICO

O vídeo esférico, diferente do vídeo tradicional plano, é um tipo de vídeo que pode ser projetado na casca interna de uma esfera, onde o usuário, localizado no centro, possui três graus de liberdade para olhar ao redor movimentando a cabeça. A captura do vídeo começa em um arranjo de câmeras tradicionais que captura o ambiente ao redor. As várias imagens das câmeras são sincronizadas e costuradas em uma esfera formando uma única imagem. Para utilizar os codificadores de vídeo populares como HEVC, AV1, VP9, etc, o vídeo precisa ser mapeado em um plano através de técnicas de projeção cartográficas, como a projeção equirretangular (ERP) e projeção cubemap (CMP) para ser comprimido, encapsulados em arquivos MP4, por exemplo e armazenados ou transmitidos pela rede. Desta forma, para poder manipular os pixels no domínio da projeção e da esfera, extrair o viewport e calcular a qualidade do vídeo, foi necessário desenvolver uma nova biblioteca chamada `video360tools`¹ que realiza o mapeamento entre diferentes sistemas coordenadas para as projeções equirretangular e cubemap. Para isto foi definido a relação entre os sistemas de coordenadas cartesiano, horizontal e sistema de coordenadas de corpo conforme descrito a seguir.

2.1.1 `video360tools`

O sistema de coordenadas cartesiano, dextrógiro, representado na figura 2.1(a) é usado para descrever e operar a esfera no espaço de 3 dimensões através de operações de álgebra linear. O vídeo esférico é representado com o usuário no centro de uma esfera de raio 1. Na posição inicial, o eixo Z aponta para a frente, o eixo X aponta para a direita e o eixo Y aponta para baixo. Como o campo de visão do usuário é limitado pelo sistema visual humano, ele visualiza

¹<https://github.com/henriquedgarcia/video360utils>

apenas uma parte da esfera, chamada de Viewport.

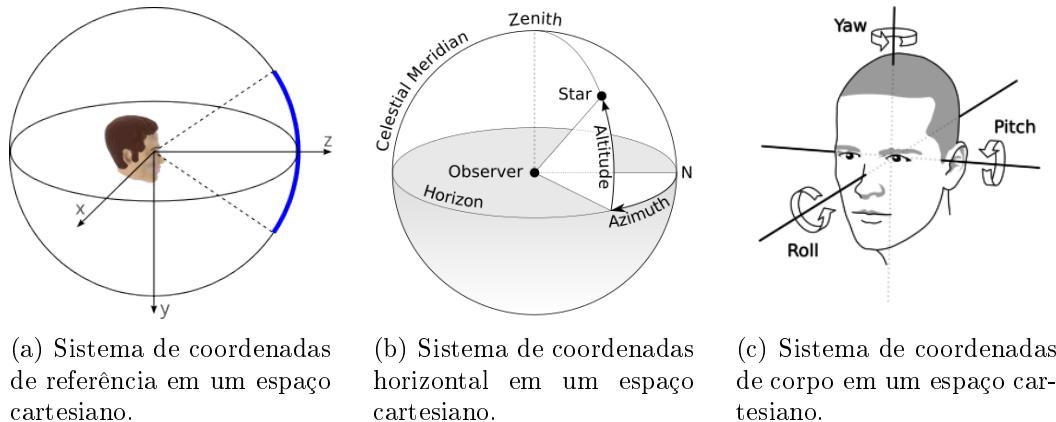


Figura 2.1. Os Sistemas de Coordenadas

Para representar um ponto na esfera do vídeo, uma variação do sistema de coordenadas esféricas, chamado de sistema de coordenadas horizontal é usado como intermediário para operações entre projeções. O sistema de coordenadas horizontal, muito usado na astronomia para observação de corpos celestes, utiliza ângulos para representar a posição dos pixels na esfera do vídeo. Como mostra a figura 2.1(b), neste sistema, as coordenadas são relativas ao horizonte (plano ZX) e a posição inicial, isto é, o eixo Z positivo.

A elevação varia de -90° a 90° , sendo que o sinal positivo aponta para acima do horizonte, enquanto que o azimute varia de -180° a 180° , sendo que o sinal positivo está à direita do eixo Z. Desta forma, o pixel na posição (elevação, azimute) igual a $(0,0)$ encontra-se no ponto $(x=0, y=0, z=1)$. As equações para conversão das coordenadas cartesianas em coordenadas horizontais são:

$$r = \sqrt{x^2 + y^2 + z^2} \quad (2.1)$$

$$\text{Azimute} = \sin^{-1} \left(\frac{-y}{r} \right) \quad (2.2)$$

$$\text{Elevação} = \tan^{-1} \left(\frac{x}{z} \right) \quad (2.3)$$

$$(2.4)$$

O sistema de coordenadas de corpo, também usado para modelagem de aeronaves é usado para modelar o movimento da cabeça do usuário e sua orientação coincide com o sistema de coordenadas horizontal. Este sistema consiste de ângulos relativos à posição inicial (elevação=0

e $\text{azimute}=0$ no sistema horizontal, ou o vetor $(0, 0, 1)$ no sistema cartesiano) e é usado para representar o movimento da cabeça do usuário usando ângulos de Euler, como mostra a figura 2.1(c). Por se tratar de rotação, os valores dos ângulos podem assumir qualquer valor (em radianos).

Assim, rodar o usuário em uma direção é equivalente a rodar a esfera na direção oposta. O movimento de *pitch* é realizado olhando em direção ao eixo X e rodando a esfera. O movimento de *yaw* é realizado rodando a esfera em relação ao eixo Y e o *roll* é realizado rodando a esfera em relação ao eixo Z. É convencionado que todas as rotações em sentido horário são positivas e em sentido anti-horário são negativas. Assim, olhar para cima possui um *pitch* positivo, olhar para a direita possui um *yaw* positivo e rodar a cabeça em sentido horário possui um ângulo *roll* positivo.

A nova posição dos pixels na posição (x', y', z') da esfera é dada pelo produto matricial da posição dos pixels por uma matriz de rotação R, conforme equação abaixo.

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = R \times \begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix}$$

A matriz de rotação para os ângulos (*yaw*, *pitch*, *roll*) é dada pelo produto matricial das matrizes de rotação do eixo X, Y e Z na seguinte ordem:

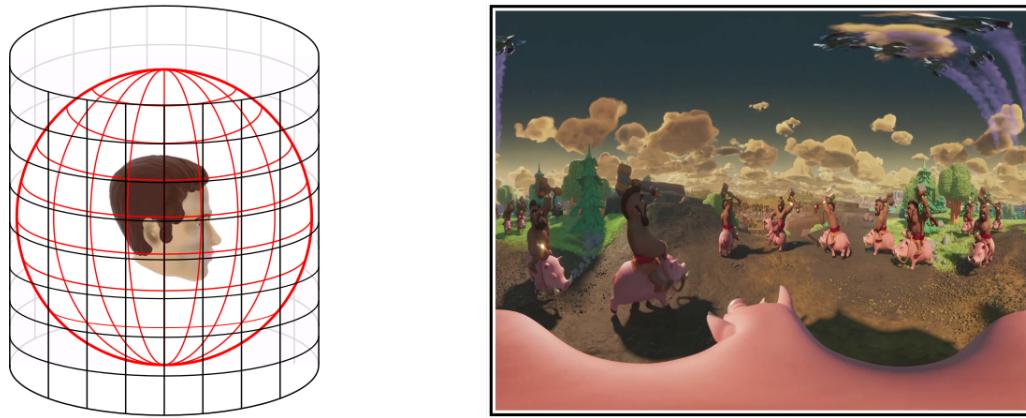
$$R = Y \times X \times Z$$

As matrizes individuais são:

$$R_X = \begin{bmatrix} 1 & 0 & s0 \\ 0 & \cos(\text{pitch}) & -\sin(\text{pitch}) \\ 0 & \sin(\text{pitch}) & \cos(\text{pitch}) \end{bmatrix}$$

$$R_Y = \begin{bmatrix} \cos(\text{yaw}) & 0 & \sin(\text{yaw}) \\ 0 & 1 & 0 \\ -\sin(\text{yaw}) & 0 & \cos(\text{yaw}) \end{bmatrix}$$

$$R_Z = \begin{bmatrix} \cos(\text{roll}) & -\sin(\text{roll}) & 0 \\ \sin(\text{roll}) & \cos(\text{roll}) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$



Equirectangular

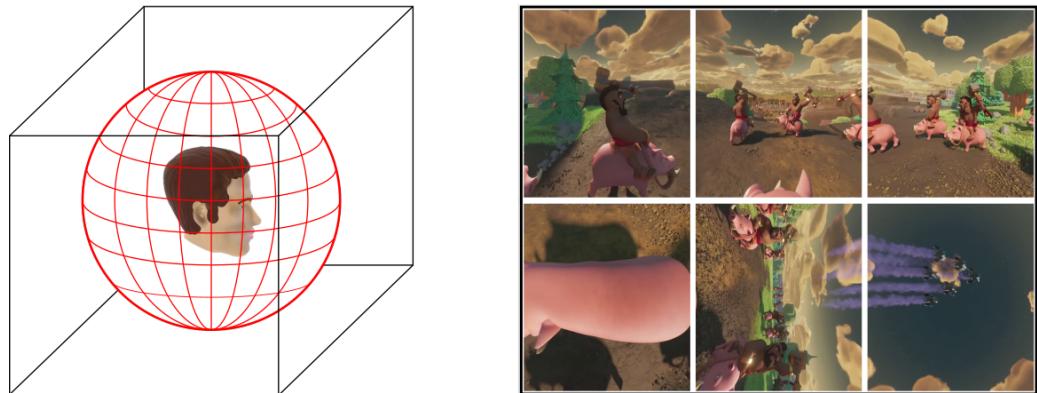
Figura 2.2. Projeção Equirretangular mapeada no plano da imagem.

2.1.2 Projeções

2.1.2.1 Projeção Equirretangular

A projeção equirretangular é uma técnica de mapeamento cartográfico de uma esfera para um plano retangular, preservando a proporção dos elementos na imagem, ou seja, os pixels na imagem mapeada têm uma relação de aspecto constante. Isso significa que a imagem resultante tem a mesma largura para altura em todos os pontos, o que simplifica sua representação digital e facilita o processo de visualização em plataformas de vídeo e dispositivos, como mostra a figura 2.2.

No mapeamento de um vídeo 360° usando a projeção equirretangular, a esfera do vídeo é desdobrada em um plano 2D retangular, onde cada ponto na esfera é mapeado para um ponto correspondente na imagem retangular. Primeiro, a posição do pixel na esfera em coordenadas cartesianas é convertido para o sistema de coordenadas horizontal usando as equações 2.4. Em seguida, as coordenadas são normalizadas em um plano uv onde u e v variam de 0 a 1, o eixo u aponta para a direita e o eixo v aponta para baixo usando as equações 2.7. Por fim as coordenadas (m, n) na imagem são dadas multiplicando u e v pela largura (W) e altura (H) da imagem, respectivamente. O truncamento é realizado afim de se aplicar a interpolação por vizinho mais próximo.



Cubemap

Figura 2.3. Projeção Cubemap e a disposição das seis faces no plano da imagem.

$$u = \frac{\text{Azimute}}{2\pi} + 0.5 \quad (2.5)$$

$$v = \frac{-\text{Elevação}}{\pi} + 0.5 \quad (2.6)$$

$$(2.7)$$

2.1.2.2 Projeção Cubemap

A projeção cubemap é outra técnica comumente usada em vídeos 360° para mapear um ambiente tridimensional em uma representação bidimensional, atualmente adotado pelo YouTube e pelo Facebook. Ao contrário da projeção equirretangular, que mapeia a esfera em um plano retangular, a projeção cubemap mapeia a esfera em seis faces quadradas de um cubo de lado igual a 2. As seis faces correspondem a visão ao redor do ponto de captura. As faces geralmente são organizadas em uma grade de 3x2 formando um único arquivo de imagem. As três imagens superiores são as faces da esquerda, frente e direita, Enquanto as faces de baixo são rotacionadas 90° em sentido horário e correspondem as faces de cima, de trás e de baixo. A figura 2.3 mostra um exemplo de mapeamento para projeção Cubemap (CMP).

Como forma de representação auxiliar, cada face é mapeada em um plano uv conforme a tabela 2.1, onde as coordenadas u e v variam de -1 a +1, com u apontando para direita e v apontando para baixo. Por fim, as coordenadas uv são convertidas para posições dos pixels pela equação $(uv + 1) \times \frac{A}{2} - 0.5$, onde A é o número de pixels em uma face do quadrado, e repositionado na imagem. O processo inverso, de mapeamento do plano uv para o espaço

cartesiano é feito pela tabela 2.1.

Tabela 2.1. Tabela de coordenadas (X, Y, Z) em função de (U, V)

Face	X	Y	Z
0	-1.0	v	u
1	u	v	1.0
2	1.0	v	-u
3	-u	1.0	v
4	-u	v	-1.0
5	-u	-1.0	-v

Tabela 2.2. Tabela de condições e valores de $face, u, v$ em função de (X, Y, V)

Condição	Face	u	v
$ X \geq Z $ e $ X \geq Y $ e $X < 0$	0	$\frac{Z}{ X }$	$\frac{Y}{ X }$
$ Z \geq X $ e $ Z \geq Y $ e $Z > 0$	1	$\frac{X}{ Z }$	$\frac{Y}{ Z }$
$ X \geq Z $ e $ X \geq Y $ e $X > 0$	2	$\frac{-Z}{ X }$	$\frac{Y}{ X }$
$ Y \geq X $ e $ Y \geq Z $ e $Y < 0$	3	$\frac{-X}{ Y }$	$\frac{Z}{ Y }$
$ Z \geq X $ e $ Z \geq Y $ e $Z < 0$	4	$\frac{-X}{ Z }$	$\frac{Y}{ Z }$
$ Y \geq X $ e $ Y \geq Z $ e $Y > 0$	5	$\frac{-X}{ Y }$	$\frac{-Z}{ Y }$

2.1.3 O Viewport

O viewport é definido como a porção da esfera que é visível pelo usuário. Como a visão humana (FOV - *Field of View*) é limitada, os HMD reproduzem um FOV de aproximadamente $120^\circ \times 90^\circ$. Para isto os ponto da esfera que pertencem ao viewport são projetados a um plano tangente a esfera no ponto $P(0, 0, 1)$ utilizando a projeção Gnomônica ou Retilínea², como pode ser visto na figura 2.4.

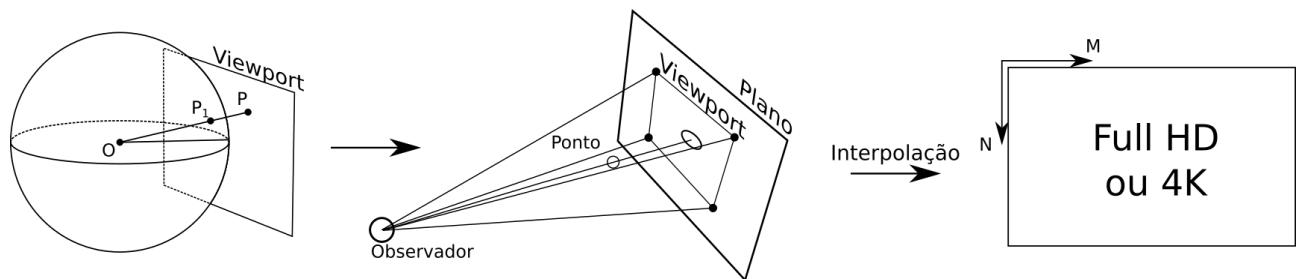


Figura 2.4. Extração do viewport usando projeção retilínea ou gnomônica.

Primeiramente, a matriz da imagem do viewport é pre alocada de acordo com a resolução exibida para o usuário. Em vez do valor do pixel, cada célula da matriz é preenchida com a

²<https://www.wolframalpha.com/input/?i=rectilinear+projection>

posição do pixel na esfera usando coordenadas horizontais, de $-fov_X$ a $+fov_X$ e de $-fov_Y$ a $+fov_Y$. Em seguida, cada coordenada é convertida para o sistema cartesiano usando as equações 2.4. Finalmente as coordenadas cartesianas são convertidas para o domínio da projeção, seja equirretangular ou cubemap. Obviamente os pontos não serão os mesmos, então é usado algum método de interpolação, com interpolação por vizinho mais próximo ou interpolação linear. Agora basta mapear os pixels do viewport com os pixels da projeção.

Por outro lado, para verificar quais ladrilhos pertencem ao viewport, é preciso verificar se um pixel da projeção está sendo visualizado. Para isto, consideramos que o viewport compreende o espaço interno da intersecção de quatro planos que passam pelo centro da esfera e possuem inclinação relativas ao FOV do viewport e ao eixo Z. Por exemplo, um FOV de $120^\circ \times 90^\circ$ está limitado a 60° à direita e -60° à esquerda do eixo Z e 45° acima e -45° abaixo do eixo Z. Estes planos definem os limites superiores, inferiores, esquerdo e direito do viewport. Estas normais devem apontar para o lado de fora do viewport, assim o viewport compreende apenas os pontos que estão abaixo de todos os planos ao mesmo tempo. As normais que definem os planos são descritas na equação 2.8, onde fov_X e fov_Y são os ângulos horizontal e vertical do FOV, respectivamente.

$$N = \begin{bmatrix} 0 & -\cos\left(\frac{fov_Y}{2}\right) & -\sin\left(\frac{fov_Y}{2}\right) \\ 0 & \sin\left(\frac{fov_Y}{2}\right) & -\cos\left(\frac{fov_Y}{2}\right) \\ \cos\left(\frac{fov_X}{2}\right) & 0 & -\sin\left(\frac{fov_X}{2}\right) \\ -\sin\left(\frac{fov_X}{2}\right) & 0 & -\cos\left(\frac{fov_X}{2}\right) \end{bmatrix} \quad (2.8)$$

Após o movimento de cabeça, as normais são rotacionadas de acordo com as coordenadas de corpo $H(yaw, pitch, roll)$ e assumem nova posição N_R dado pelo produto matricial do vetor de normais com a matriz de rotação $N_R = \mathbf{N} \times \mathbf{R}$. Um pixel \mathbf{P} qualquer no espaço cartesiano pertencerá ao viewport se o produto interno $N \cdot T \dot{P} \leq 0$ para todas as normais, ou seja, estão abaixo das normais após as normais forem rotacionadas durante o movimento da cabeça.

2.2 QUALIDADE OBJETIVA DO VÍDEO

2.2.1 Qualidade para o cliente

No processo de preparação para o *streaming* de vídeos esféricos, a esfera de vídeo passa por um mapeamento em um plano, introduzindo distorções que variam ao longo do plano. Ao utilizar métodos de compressão com perda, os codificadores geralmente aplicam a quantização de forma uniforme em todo o quadro, podendo impactar mais severamente em regiões distorcidas pela projeção. Após a decodificação, os pixels renderizados são remapeados de volta para a esfera, e quaisquer artefatos de codificação, como blocados, aliasing, efeito de *Gibbs*, borrado, etc, também serão distorcidos.

Assim, a qualidade objetiva da viewport é definida como a disparidade entre o viewport extraído do vídeo codificado e o viewport de um vídeo de referência. A Figura 2.5 ilustra como medir a qualidade objetiva do *viewport*, considerando a esfera sem degradação e a esfera recuperada após a compressão. A viewport precisa ser extraída com base na posição da cabeça do usuário em ambos os vídeos de referência e codificado, assim o MSE e SSIM devem ser calculados entre esses quadros. O MSE foi usado no lugar do PSNR, pois o PSNR deve ser calculado apenas após o cálculo da média do MSE de todos os quadros de um vídeo, como mostra a documentação do filtro de PSNR do ffmpeg³. As operações de cálculo MSE e do SSIM foram calculadas usando a biblioteca scikit-image. Apesar do MSE e o SSIM falhem ao tentar medir qualidade sob certas condições, elas são suficientes para se medir distorções produzidas por artefatos esperados como bordas de ladrilho, borrado e blocado. (WANG *et al.*, 2004)

Quando o vídeo é segmentado em ladrilhos, é necessário reconstruir a projeção com os ladrilhos que são visualizados durante todo o período de duração de um *chunk*. Como o usuário está movimentando a cabeça, durante um *chunk* os ladrilhos vistos podem se modificar. Para o DASH o *chunk* inteiro precisa estar no *buffer* do cliente para que apenas alguns quadros possam ser decodificados. A aplicação cliente precisa saber de antemão quais ladrilhos serão vistos e então solicitá-los ao servidor. Por exemplo, na figura 2.6 vemos uma projeção equirretangular segmentada em 3x3 ladrilhos. No instante 0 o usuário está visualizando os ladrilhos 3, 4, 6,

³<https://ffmpeg.org/ffmpeg-filters.html>

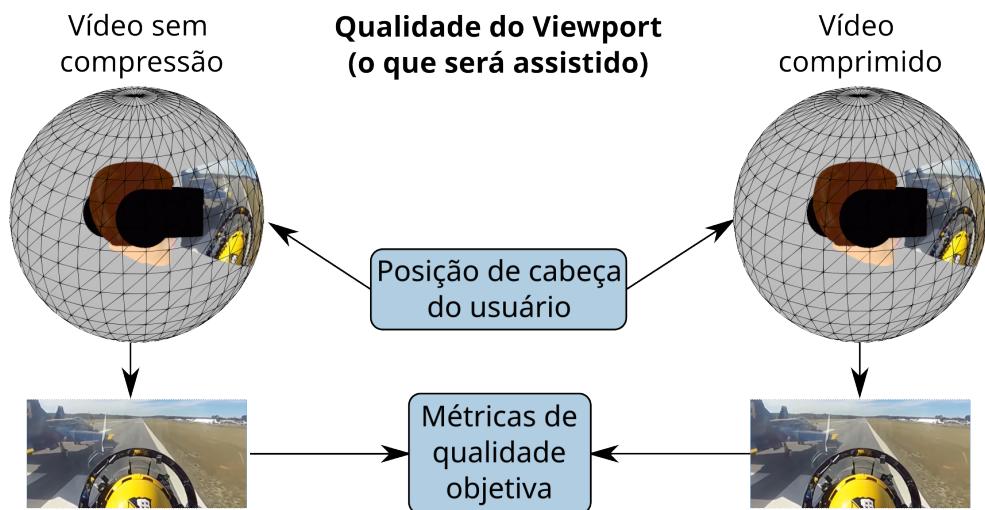


Figura 2.5. Avaliação de qualidade objetiva do viewport.

7 e 8, porém, devido ao movimento da cabeça, antes de se concluir a reprodução do *chunk*, o usuário passa a ver os ladrilhos 0, 1, 3, 4, 6 e 7. Os ladrilhos 0 e 1 passaram a ser vistos e o ladrilho 8 deixou de ser visto. Os ladrilhos 0 e 1 precisam já ter sido baixados e decodificados para que o usuário veja seu conteúdo nos instantes finais do *chunk*.

Em um cenário com decodificadores operando em paralelo o tempo para a decodificação dos chunks do viewport será igual ao maior tempo de decodificação entre todos os tiles selecionados. Porém, em um ambiente com threads simples, o tempo de decodificação será igual a soma de todos os tempos de decodificação. A taxa de bits necessária para a reprodução do viewport será igual soma de todos os ladrilhos que foram visualizados durante a duração de um *chunk*.

2.2.2 Qualidade para o servidor

Por outro lado, para sistemas de transmissão adaptativos como o MPEG DASH, a qualidade do vídeo codificado está diretamente correlacionada com sua taxa de bits. Esta correlação baseia-se no pressuposto de que uma taxa de bits mais elevada corresponde a uma melhor qualidade. A lógica por trás dessa relação reside no fato de que a qualidade medida, normalmente avaliada usando métricas como PSNR ou MSE, quantifica o erro induzido pela quantização no processo de compressão com perdas. Neste contexto, o aumento da quantização leva a uma perda de detalhes e a uma maior compressão, reduzindo a qualidade e também a taxa de bits.

No entanto, é importante notar que as métricas SSIM e PSNR podem não ser adequadas

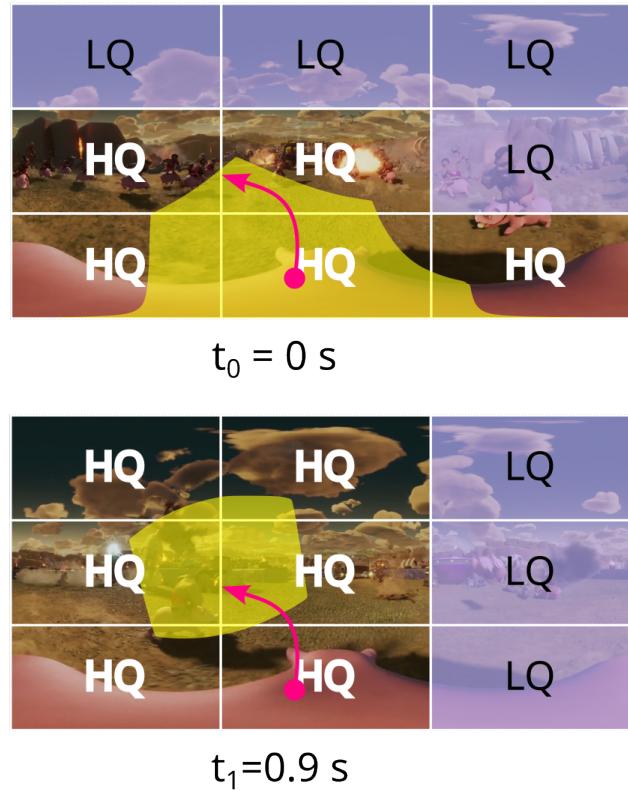


Figura 2.6. Tiles que deverão ser solicitados ao servidor e que serão assistidos durante um *chunk*.

para avaliar diretamente a qualidade da projeção, especialmente considerando as distorções introduzidas durante o processo de projeção. Consequentemente, métricas de qualidade esféricas foram desenvolvidas para avaliar o nível objetivo de qualidade de uma projeção. A parte superior da Figura 2.7 ilustra como a “qualidade objetiva da projeção” deve ser medida. Métricas de qualidade devem ser aplicadas entre a projeção antes da codificação (vídeo de referência) e a projeção decodificada após a compressão (vídeo degradado).

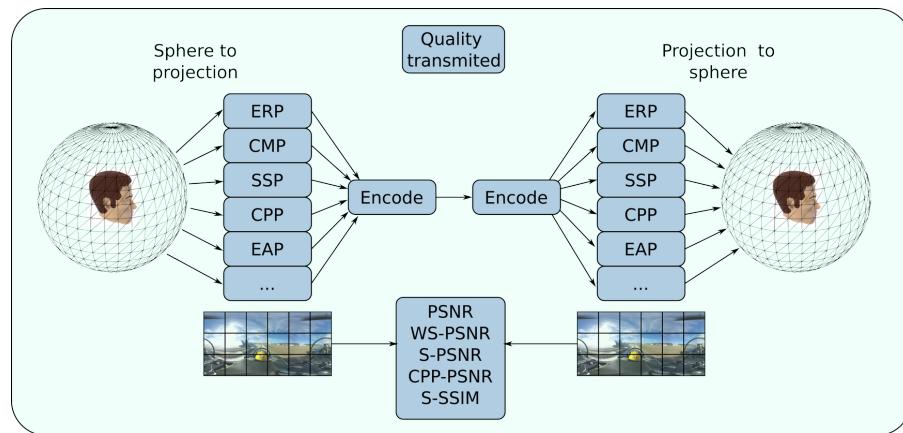


Figura 2.7. Avaliação de qualidade objetiva da projeção.

Conseguir um vídeo codificado de qualidade alta requer minimizar as perdas a um nível que

não seja perceptível o suficiente para causar desconforto. A qualidade objetiva da projeção é definida especificamente como a disparidade entre o vídeo codificado e o vídeo de referência da perspectiva da projeção. Consequentemente, a qualidade objectiva da projeção de vídeo difere da qualidade objetiva da janela de visualização. O que os algoritmos convencionais de taxa de bits adaptativa (ABR) podem considerar satisfatório no domínio de codificação/transmissão pode não estar necessariamente alinhado com a qualidade ideal no domínio da esfera.

Para formular um algoritmo eficaz de adaptação de taxa de bits (ABR) adaptado para streaming de vídeo em 360°, é necessário um estudo aprofundado para compreender a relação entre a qualidade da janela de visualização e a qualidade da projeção. Esse mapeamento torna-se crucial no processo de tomada de decisão para troca de qualidade, reconhecendo que o usuário e o algoritmo ABR operam em planos distintos (TRAN *et al.*, 2017; XU *et al.*, 2020).

2.2.2.1 MSE/PSNR

PSNR, ou *Peak Signal-to-Noise Ratio*, e o MSE (*Mean Squared Error*) são métricas amplamente utilizadas para avaliar a qualidade de imagens e vídeos reconstruídos ou compactados. Quantifica a relação entre a potência máxima possível de um sinal (neste contexto, uma imagem ou vídeo) e a potência do ruído corruptor que impacta a qualidade do sinal. As duas métricas se relacionam através da seguinte fórmula:

$$MSE = \frac{1}{M \times N} \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} (y(i,j) - y'(i,j))^2 \quad (2.9)$$

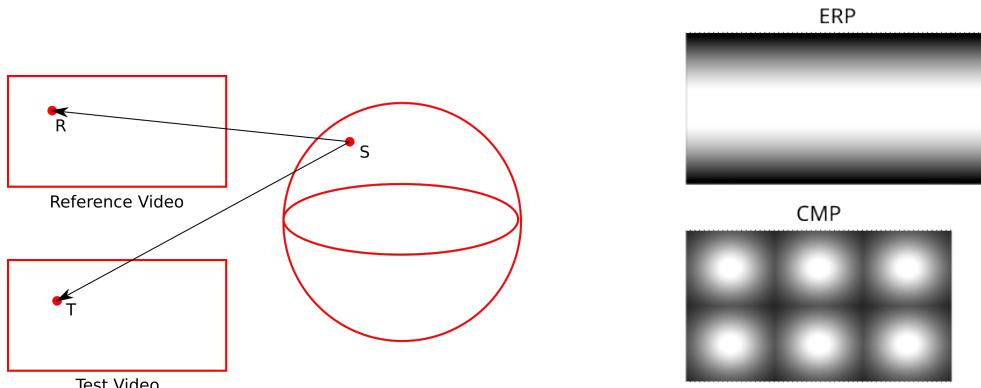
$$PSNR = 10 \times \log_{10} \left(\frac{MAX_I^2}{MSE} \right) \quad (2.10)$$

Primeiro, o MSE é calculado para todos os quadros do vídeo e, no final da reprodução, a média do MSE é usado para calcular o PSNR (*Peak Signal-to-Noise Ratio*). Para um vídeo segmentado em ladrilhos, considere que L_{quadro} é a lista de quadros tocados pelo viewport ao longo da reprodução do vídeo e $MSE_{frame}(i)$ é o MSE de um quadro. A qualidade de uma seção de vídeo assistido pelo usuário, é definida pelo PSNR utilizando a média do MSE de todos quadros que aparecerem no viewport, como mostra a equação 2.11.

$$\bar{MSE} = \sum_{i=0}^{N_{frames}} \frac{MSE_{frame}(i)}{N_{frames}} \quad (2.11)$$

2.2.2.2 S-MSE/S-PSNR

O S-PSNR emprega uma lista de 655.362 pontos equidistantes na esfera para identificar os pixels correspondentes na projeção original e na projeção degradada para em seguida calcular o MSE para todos estes pontos. Este processo é ilustrado na Figura 2.8(a). Encontra-se o ponto na projeção correspondente ao ponto da esfera e então calcula-se o erro. Desta forma é possível comparar duas projeções diferentes. Apesar desta métrica se preocupar em uniformizar as amostras no domínio da esfera, onde o usuário realmente interage, quanto maior a resolução da projeção, mais pixels serão desprezados no cálculo da métrica.



(a) O S-PSNR busca os pixels na projeção de referência “R” e a projeção degradada “T” correspondentes aos pontos “S” pré-definidos e equidistantes na esfera.

(b) Mapa de pesos para a projeção Equirretangular e Cubemap usado pela métrica WS-MSE

2.2.2.3 WS-MSE/WS-PSNR

O WS-MSE e o WS-PSNR incorporam pesos para cada pixel da projeção ao calcular MSE, conforme expresso nas Equações 2.12 e 2.13. Considerando uma projeção com resolução MxN, o WS-MSE primeiro calcula a média ponderada do quadrado das diferenças dos pixels y e y' na posição (i,j) para todos os pixels de cada quadro. O cálculo do WS-PSNR só é aplicado sobre a média do WS-MSE de todos os quadros.

$$MSE = \frac{\sum_{i=0}^{M-1} \sum_{j=0}^{N-1} (y(i,j) - y'(i,j))^2 \times w(i,j)}{\sum_{i=0}^{M-1} \sum_{j=0}^{N-1} w(i,j)} \quad (2.12)$$

$$WS-PSNR = 10 \times \log_{10} \left(\frac{MAX_I^2}{MSE} \right) \quad (2.13)$$

Os pesos de cada pixel dependem da sua posição na projeção e do tipo de projeção. Projeções como ERP esticam mais os polos enquanto a projeção Cubemap distribui as faces na imagem de forma arbitrária. As equações 2.14 e 2.15 delineiam os pesos para as projeções equirretangular (ERP) e cubemap (CMP). No caso de uma projeção Cubemap, considera-se que todas as faces do cubo são quadradas e possuem resolução $A \times A$. O resultado pode ser visto na figura 2.8(b)

$$w_{ERP}(i,j) = \cos \left(\frac{(j + 0.5 - N/2)\pi}{N} \right) \quad (2.14)$$

$$w_{CMP}(i,j) = \left(1 + \frac{d^2(i,j)}{r^2} \right)^{-\frac{3}{2}} \quad (2.15)$$

$$d^2(i,j) = (i + 0.5 - \frac{A}{2})^2 + (j + 0.5 - \frac{A}{2})^2 \quad (2.16)$$

2.2.3 Estatísticas

Determinar as estatísticas das métricas é muito importante afim de entender seu comportamento e distribuição ao longo da reprodução para diferentes tipos de conteúdos. A distribuição das métricas de tempo de decodificação, taxa de bits e qualidade objetiva foram analisadas considerando que as distribuições são positivas e contínuas, descartando distribuições discretas e distribuições sobre a reta real. Considerando também as distribuições disponíveis nos pacotes estatísticos mais comuns, como SciPy e Matlab, as distribuições analisadas são: Burr Tipo XII, Birnbaum-Saunders, Gamma, Inversa Gaussiana, Rayleigh, Log Normal, Generalized Pareto, Pareto, Half-Normal, e Exponencial. O processo de ajuste mais comum se baseia na estimação dos parâmetros por máxima verossimilhança e para a avaliação das distribuições foi utilizada o erro quadrático médio (RMSE).

A correlação entre as métricas é feita entre todos os chunks de cada ladrilho. Como a correlação mede a dependência entre duas variáveis e as métricas variam de forma diferente

para cada tipo de ladrilhamento e para cada qualidade de codificação, a correlação deve ser medida considerando apenas as ladrilhos de uma mesma qualidade e de uma mesma região de cada vídeo. Assim, as métricas de 60 chunks para cada ladrilho, para cada qualidade e para cada vídeo puderam ser correlacionada. Por fim a média de todas as correções são consideradas ao se comparar diferentes cenários.

CAPÍTULO 3

AVALIAÇÃO E RESULTADOS

Para otimizar o streaming de vídeo esférico é essencial conhecer suas principais características que afetam seu desempenho e sua qualidade. Para caracterizar o tráfego de vídeo panorâmico de 360 graus ladrilhado com DASH-SRD, é essencial replicar as condições de transmissão, codificação e segmentação do vídeo durante uma seção do cliente. Além disso, as métricas de avaliação devem ser relacionadas a experiência do usuário e do servidor. Para o cliente queremos medir o tempo de decodificação, a taxa de bits e a qualidade do vídeo exibido no viewport. Para o servidor, queremos que a melhor qualidade seja entregue com a menor taxa de bits possível e para isso avaliamos métricas de qualidade de codificação tradicional e de vídeo esférico e sua correlação com a taxa de bits.

Em um sistema de streaming adaptativo como DASH, os chunks do vídeo devem ser codificados de forma que possa ser decodificado de forma independente do chunk anterior, isto é, não tenham dependência temporal. Para isso, o GOP (*Group of pictures*) usado em codificação preditiva, precisa ser do tipo fechado e com o numero de quadros fixo igual a duração do *chunks*. Além disso, para evitar que haja artefatos nas bordas dos ladrilhos durante a reprodução, os ladrilhos precisam ser codificados de forma que a qualidade seja aproximadamente a mesma entre todos os ladrilhos de um quadro ao longo de sua reprodução. Geralmente os vídeos que serão transmitidos devem ser codificados baseado em taxa média para que o buffer de reprodução possa ser facilmente modelado. Porém, se o vídeo for segmentado em ladrilhos, cada ladrilho possuirá uma complexidade diferente, por exemplo, ladrilhos que mapeiam o topo ou a base da esfera tendem a retratar o chão ou o céu, que possuem poucos detalhes, sendo assim a compressão é alta, mesmo usando pouca quantização. Por outro lado, ladrilhos que mapeiam regiões da esfera com alta atividade temporal e espacial, como movimento, pessoas, texturas, etc, tentam a ser difícil de comprimir. Assim, os ladrilhos de um vídeo não devem possuir a mesma taxa de bit, mas sim a mesma distorção de codificação. Em contra partida, a taxa de

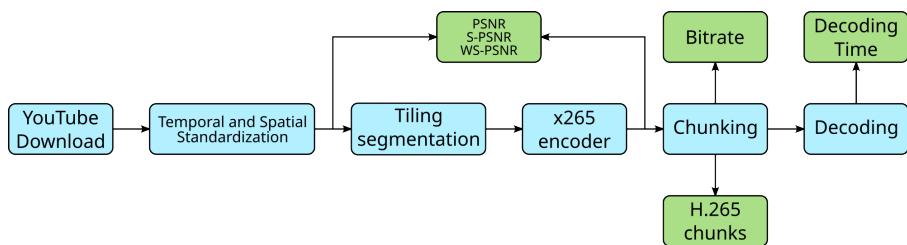


Figura 3.1. Fluxograma para captura de métricas de desempenho.

bits poderá variar muito dependendo do conteúdo do vídeo, inserindo mais complexidade na modelagem do buffer de reprodução.

3.1 CARACTERIZAÇÃO DOS ELEMENTOS DO SERVIDOR

O fluxo de trabalho da caracterização é visto na figura 3.1. Os vídeos selecionados São os mesmos empregados no estudo de Nasrabadi (NASRABADI *et al.*, 2019). Seis vídeos foram gravados pessoalmente pelo autor em projeção equirretangular. Os demais vídeos foram obtidos do YouTube em projeção cúbica. O processo de download foi executado usando a ferramenta JDownloader¹ e, devido a variabilidade de qualidades e formatos abrangendo da plataforma, demos preferência a vídeos codificados com h.264, resolução de 4K e 30 fps (quadros por segundo). Apesar disto, os vídeos estavam em duas projeções diferentes e possuíam grande variabilidade de resoluções e taxas de quadro. Isto nos levou a padronizá-los afim de podermos comparar suas características.

Assim, topes os vídeos foram convertidos para as projeções cubemap e equirretangular usando o codificador de referência HM 16.9 com a biblioteca 360lib da MPEG². Em seguida, convertidos para as resoluções 3240×2160 (CMP) e 4320×2160 (ERP), com taxa de quadros igual a 30 fps. A proporção da projeção foi definida dependendo da projeção utilizada. Para a projeção equirretangular, a proporção é de 2:1, pois ela mapeia os 360° em torno da esfera e os 180° de polo a polo. Já a projeção cúbica apresenta uma proporção de 3:2, pois cada uma das seis faces do cubo é projetada a 90° tanto na latitude quanto na longitude. Por questões práticas, os vídeos foram comprimidos sem perda usando codificador x265 do ffmpeg com CRF igual a 0 e armazenados em arquivos mp4.

¹<https://jdownloader.org/>

²https://jvet.hhi.fraunhofer.de/svn/svn_360Lib/

A Tabela 3.1 exibe a lista dos cinquenta e seis vídeos utilizados. Os vídeos estão codificados com CRF 28 com a configuração padrão do codificador x265. Conforme estudo de Nasrabadi, os vídeos foram classificados de acordo com o movimento da câmera e o numero de objetos na cena. O número de objetos na cena foram definidos pelo autor. O valor da coluna "Grupo" é composto pela primeira inicial do tipo de movimento de câmera (fixo, horizontal, vertical, rotacional, múltiplo) e numero de objetos na cena (nenhum, simples, múltiplo). Assim, o grupo "VM"corresponde ao movimento Vertical com Múltiplos objetos.

Tabela 3.1: Vídeos usados no experimento

Grupo	Nome	Projeção	SI	TI	Taxa de Bits (Mbps)
FN	montana	ERP	33.4	0.34	3.033
		CMP	35.7	0.19	2.811
	sunset	ERP	15.6	0.44	1.571
		CMP	18.7	0.46	1.442
FS	closet_tour	ERP	51.0	1.65	2.349
		CMP	57.2	1.95	2.276
	video_04	ERP	49.9	1.19	2.875
		CMP	50.9	1.11	2.245
Continua na próxima página					

Tabela 3.1 – Continuação da página anterior

Grupo	Nome	Projeção	SI	TI	Taxa de Bits (Mbps)
FM	dubstep_dance	ERP	32.9	6.14	2.771
		CMP	38.0	6.28	2.516
	rhinos	ERP	38.5	0.71	3.799
		CMP	39.8	0.63	3.251
HN	drone_footage	ERP	30.3	5.49	5.153
		CMP	37.2	5.80	4.334
	petite_anse	ERP	41.4	7.84	5.913
		CMP	39.4	7.50	4.919
HS	cable_cam	ERP	49.7	17.54	18.790
		CMP	53.1	18.47	17.554
	motorsports_park	ERP	35.6	8.15	3.545
		CMP	41.7	8.88	3.413
HM	chariot_race	ERP	39.1	16.87	8.034
		CMP	44.8	18.99	7.519
	nyc_drive	ERP	82.7	24.59	12.176
		CMP	92.7	27.72	11.565
VN	elevator_lift	ERP	52.9	8.46	2.325
		CMP	60.3	9.59	2.100
	glass_elevator	ERP	74.6	9.08	5.432
		CMP	82.1	11.82	5.974
VM	drone_video	ERP	77.0	15.73	7.831
		CMP	84.5	18.58	7.431
	drop_tower	ERP	41.0	4.95	3.141
		CMP	45.0	4.27	2.839
RN	video_19	ERP	44.7	7.80	1.310
		CMP	48.8	9.01	1.645
	video_20	ERP	31.8	5.36	0.787
		CMP	36.8	6.03	0.925
RS	penthouse	ERP	42.2	1.56	1.421
		CMP	48.3	1.61	1.458
	video_22	ERP	34.7	5.59	1.055
		CMP	40.6	6.56	1.166
RM	video_23	ERP	45.6	10.95	3.398
		CMP	49.7	12.28	3.231
	video_24	ERP	41.8	9.43	1.802
		CMP	48.7	11.11	2.097
MN	angel_falls	ERP	24.3	1.89	2.690
		CMP	24.2	1.95	2.233
	three_peaks	ERP	25.9	3.30	1.693
		CMP	30.4	3.06	1.473
MS	drone_chases_car	ERP	28.9	12.10	5.839
		CMP	29.6	13.12	5.251
	wingsuit_dubai	ERP	68.8	18.98	10.012
		CMP	66.8	16.86	7.881

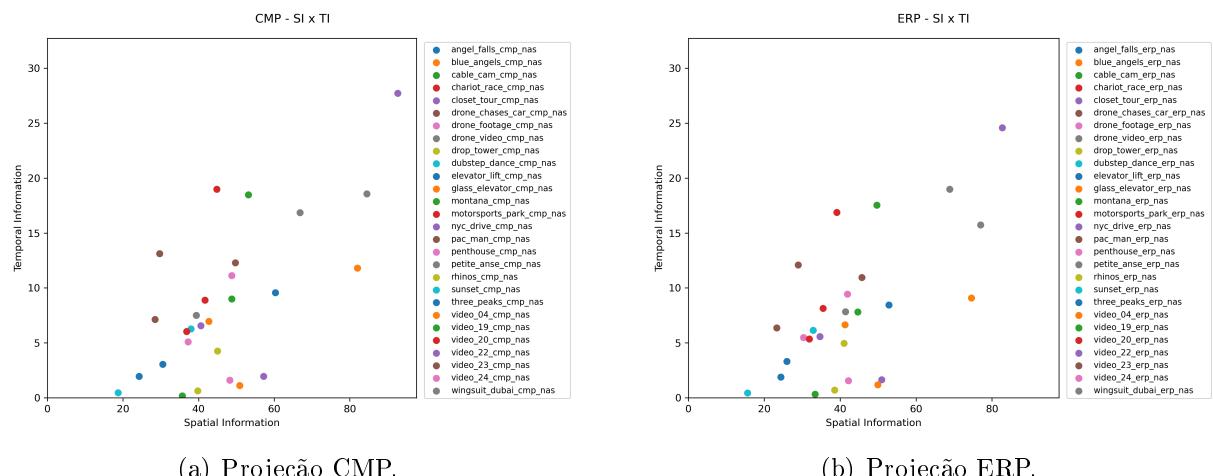
Continua na próxima página

Tabela 3.1 – Continuação da página anterior

Grupo	Nome	Projeção	SI	TI	Taxa de Bits (Mbps)
MM	blue_angels	ERP	41.2	6.65	3.766
		CMP	42.7	6.94	3.520
	pac_man	ERP	23.2	6.36	1.262
		CMP	28.4	7.14	1.193

As colunas "SI" e "TI" mostram informações espaciais (SI) e temporais (TI), calculadas segundo a recomendação ITU-T P.910, porém, utilizando a mediana em vez do valor máximo, pois os vídeos são longos e possuem picos que não caracterizam o vídeo em si. Os índices de correlação de Pearson entre SI e taxa de bits, assim como entre TI e taxa de bits, são respectivamente de 0,519 e 0,759 para projeção cúbica e 0,528 e 0,773 para projeção equirretangular, indicando uma influência razoável tanto de SI quanto de TI na taxa de bits, com ênfase à atividade temporal.

Examinando o gráfico de dispersão de SI e TI na Figura 3.2, observamos um padrão bem distribuído entre os vídeos: a taxa de bits média é de 4,22 Mbps, o SI médio é de 46,3 e o TI médio é de 8,5. Os índices de correlação de Pearson entre SI e taxa de bits, bem como entre TI e taxa de bits, são de 0,530 e 0,765, respectivamente, significando uma influência razoável tanto de SI quanto de TI na taxa de bits.

**Figura 3.2.** Dispersão do SI e TI para as projeções avaliadas

Para replicar uma condição comparável DASH-SRD, os ladrilho passaram pelo ladrilhamento antes da codificação, garantindo que cada ladrilho fosse espacialmente independente. Para cada vídeo, geramos quatro novas sequências de vídeo, cada uma com padrões de os la-

drilhos conforme a tabela 3.2. Quanto maior o numero de ladrilhos, menor o numero de pixel em cada ladrilho, reduzindo o espaço de busca do compensador de movimento do codificador e reduzindo a eficiênci da compressão. A projeção cúbica, tendo proporção menor tende a ser uma representação mais compacta, apresentando menor taxa de bits, porém com maior complexidade de compressão. Apesar de quanto menor os ladrilhos, menos pixels não visto são processador, este procedimento produz redução na eficiênci do codificador. Além disso, quanto mais ladrilhos forem solicitados, serão necessárias múltiplas instâncias de um decodificador no cliente e mais complexo será o processo, pois para reproduzir um único quadro do ladrilho, todo o chunk precisa ser decodificado.

Tabela 3.2. Detalhes do ladrilhamento e resoluções resultantes.

Ladrilhamento	Quantidade de ladrilhos	Projeção	Pixels por ladrilho
1x1	1	CMP	4320x2160
		ERP	3240x2160
3x2	6	CMP	1440x1080
		ERP	1080x1080
6x4	24	CMP	720x540
		ERP	540x540
9x6	54	CMP	480x360
		ERP	360x360
12x8	96	CMP	360x270
		ERP	270x270

Subsequentemente, utilizamos o codificador HEVC x265 no FFmpeg³ para padronizar e codificar cada ladrilho, empregando seis valores distintos do Fator de Taxa Constante (CRF) para controlar a qualidade/taxa de bits: 16, 22, 28, 34, 40 e 46. O valor de CRF mais baixo (mais alto) de 16 (46) corresponde à maior (menor) qualidade de vídeo. O valor padrão de 28 é recomendado pelo codificador por seu equilíbrio entre qualidade e taxa de compressão. Vale ressaltar que cada incremento de 3 no valor de CRF resulta em uma redução pela metade na taxa de bits; em nosso conjunto, a taxa de bits diminui quatro vezes para os valores de CRF especificados. Os parâmetros de codificação do vídeo estão detalhados na Tabela 3.3. O GOP do vídeo foi fixado em 30 quadros, o parâmetro para detecção de cortes de cena foi desativado para que o GOP não varie dentro de um chunk e o parâmetro --no-info"foi utilizado para omitir cerca de 200 bytes de informações sobre o codificador no cabeçalho mp4 e assim reduzir o overhead dos vídeos. Esta quantidade é relativamente significativa quando se trata de

³<https://ffmpeg.org/>

chunks muito pequenos com baixa complexidade de codificação e alta compressão. Além disso os vídeos foram processados com os parâmetros -tune psnr" para comparação mais precisa das métricas de distorção, como MSE e PSNR.

Resolução (Projeção)	4320 × 2160 pixels (Cubemap), 3240 × 2160 pixels (Equirectangular)
Taxa de Quadro	30 fps
GOP	30 quadros
Duração do Chunk	1 segundo
Qualidade (CRF)	0, 16, 22, 28, 34, 40 e 46
Padrão de Ladrilho	1 × 1, 3 × 2, 6 × 4, 9 × 6, 12 × 8
Codificador/Decodificador	x265/FFmpeg 5.0-static
Configuração do x265	keyint=30:min-keyint=30:open-gop=0:scenecut=0:info=0
Sistema Operacional	Ubuntu 22.04

Tabela 3.3. Parâmetros de codificação.

Após a codificação, usando o programa GPAC⁴, cada ladrilho passou por uma segmentação temporal em fragmentos de 1 segundo, equivalentes a um GOP (Grupo de Quadros) completo de 30 quadros. Essa escolha considera que a previsão de movimento da cabeça tende a ser eficaz dentro de janelas de previsão de aproximadamente 1-2 segundos (QIAN *et al.*, 2016). Subsequentemente, esses fragmentos foram encapsulados em um arquivo MP4 para facilitar a decodificação individual, acarretando em um overhead de cerca de 100-1000 bytes para incluir um cabeçalho MP4 adicional de "box moov" por fragmento. Apesar do overhead, esta abordagem permite a decodificação independente de cada chunk por qualquer tocador de vídeo. Desta forma, cada ladrilho de cada vídeo em cada projeção, compreende 60 fragmentos decodificáveis em seis qualidades diferentes, resultando em um total de 3.648.960 *chunks* para análise.

O processo de decodificação de vídeo ocorreu em um computador desktop i7-4770 de 3,4 GHz equipado com 16 GB de RAM, rodando Linux Ubuntu 22.04. O decodificador nativo do FFmpeg foi utilizado, empregando apenas uma thread para a decodificação. As medidas de tempo de decodificação coletadas apresentam precisão de 1 ms e são relativas ao tempo de usuário (*user time*), o que representa o tempo da CPU, excluindo operações do kernel como chamadas de sistema, focando especificamente em tarefas de decodificação, como operações matriciais, por exemplo. Cada fragmento de um dado ladrilho passou por decodificação cinco vezes para produzir um tempo médio de decodificação. Devido a natureza multiprocesso e da hierarquia de memória, foi possível observar uma variação do tempo de decodificação de até

⁴<https://github.com/gpac/gpac/>

93,75%. Além disso, medimos a taxa de bits de cada *chunk* simplesmente multiplicando o tamanho do arquivo por oito, já que cada *chunk* possui exatamente 1 segundo.

A seguir, para cada quadro de cada chunk, métricas objetivas de qualidade, incluindo SSIM, MSE, WS-MSE e S-MSE, serão calculadas comparando os quadros decodificados os vídeos codificados com CRF 0. As métricas são armazenadas em um JSON em uma estrutura de árvores, em que cada nível representa respectivamente a projeção, nome do video, padrão de ladrilhamento, crf, ladrilho e chunk. No caso do tempo de decodificação, foi armazenado um valor para cada decodificação.

Para a análise das distribuições estatística dos chunks codificados, empregamos a biblioteca Python SciPy. Por padrão, a SciPy utiliza do método de Estimação por Máxima Verossimilhança para determinar os parâmetros da distribuições de probabilidade de densidade (PDF). As distribuições analisadas são: Burr Tipo XII , Birnbaum-Saunders, Gamma, Inversa Gaussiana, Rayleigh, Log Normal, Generalized Pareto, Pareto, Half-Normal, e Exponencial⁵. Como ferramenta auxiliar foi utilizado o pacote Fitter para testar todas as distribuições⁶. Além das distribuições foi analisada a correlação entre as métricas e para a avaliação dos erros, foi utilizada a raiz do erro quadrático médio (RMSE). Nos resultados produzidos pela Scipy, os valores são normalizados e os parâmetros de deslocamento (*loc*) e escala (*scale*) devem ser aplicados à distribuição normalizada para obter a ajustada, conforme a equação 3.1.

$$\text{fitted_pdf}(x) = \left(\frac{1}{\text{scale}} \right) \text{normalized_pdf} \left(\frac{x - \text{loc}}{\text{scale}} \right). \quad (3.1)$$

3.2 CARACTERIZAÇÃO DOS ELEMENTOS DO CLIENTE

Para a caracterização do que o usuário experimenta, a posição da cabeça do usuário é utilizada para identificar as ladrilhos necessárias para a reconstrução da viewport. Foi desenvolvido uma biblioteca para conversão de coordenadas na esfera e um mecanismo para seleção de ladrilhos com base na posição da cabeça do usuário. Este mecanismo foi implementado em Python, e, quando fornecido o padrão de ladrilhamento, dimensões da projeção e posição da viewport, ele retorna uma lista de ladrilhos que estão visíveis na viewport. Existem vários métodos para

⁵<https://docs.scipy.org/doc/scipy/reference/stats.html>

⁶<https://fitter.readthedocs.io/>

selecionar e priorizar ladrilhos (NGUYEN *et al.*, 2020). Porém, neste trabalho, focamos em considerar apenas os ladrilhos visíveis, essenciais para a construção da viewport, conforme ilustrado na Figura 3.3. Este mecanismo foi aplicado com um banco de dados de posição da cabeça do usuário para calcular uma lista de ladrilhos que aparecem na viewport durante a reprodução.

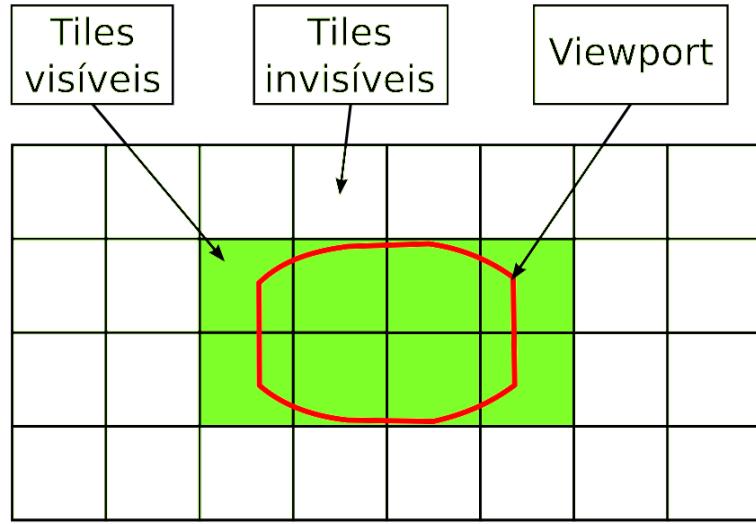


Figura 3.3. Mecanismo para seleção de ladrilhos.

O banco de dados de movimentos da cabeça foi gerado por Nasrabadi (NASRABADI *et al.*, 2019). Este banco de dados compilou a posição central do viewport na esfera para 60 voluntários (30 voluntários por vídeo), representados em quatérnions e em vetores cartesianos a uma taxa variável. Utilizando marcas de tempo foi feita a interpolação para que todas as amostras estejam à taxa de 30 Hz. A figura 3.4, extraída do trabalho de Nasrabadi, fornece um mapa da velocidade angular média para todos os usuários em cada vídeo. Como observado pelo autor, a velocidade angular média dos vídeos depende do usuário, o que significa que um usuário com velocidade alta em um vídeo tende a exibir velocidade alta em todos os vídeos.

A seguir, considerando os ladrilhos selecionados na etapa anterior, são reconstruídos a projeção e geradas viewports tanto de vídeos não compactados (CRF 0) quanto de vídeos codificados com qualidade variável para uma mesma região, com base nas regiões que os indivíduos visualizaram anteriormente. nas regiões da projeção que o ladrilho não foi selecionado é preenchido com zero. A criação da viewport é realizada utilizando a projeção Gnomônica (chamada também de projeção retilinear) restrita pelo campo de visão (FOV) do dispositivo. Nossa janela de visualização é configurada com um FOV horizontal e vertical de 100°x90°, um valor aproximado

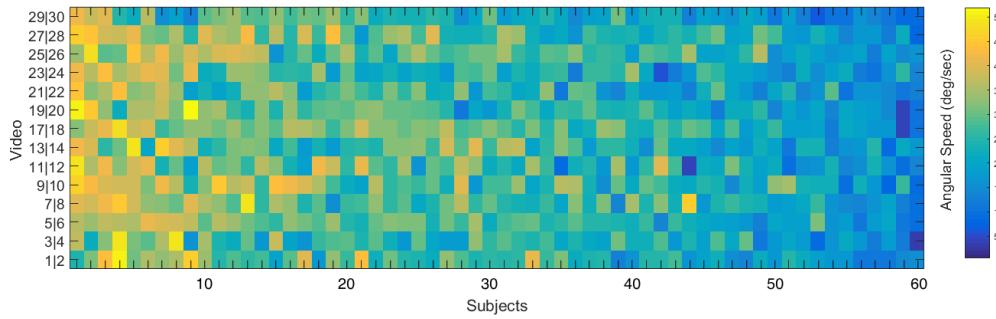


Figura 3.4. Mapa da velocidade angular média de todos os voluntários para todos os grupos de vídeo. Extraído de (NASRABADI *et al.*, 2019)

daqueles adotados pela linha Oculus Quest⁷. A seguir, será calculado o PSNR entre o quadro da viewport criado com ladrilhos compactados e o quadro da viewport criado com ladrilhos não compactados (com CRF 0).

3.3 CENÁRIOS AVALIADOS

Após a coleta dos dados, o comportamento da qualidade será analisado sob duas perspectivas: uma do ponto de vista de um sistema ABR que considera a correlação da qualidade com a taxa de transmissão do vídeo e outra do ponto de vista do usuário que visualiza o vídeo utilizando óculos de realidade virtual.

3.3.1 Estatísticas dos ladrilhos por ladrilhamento

A tabela 3.4 fornece os valores de média e desvio padrão para o tempo de decodificação e a taxa de bits dos *chunk* para vários padrões de ladrilho. Conforme previsto, a redução no tamanho do ladrilho (correspondendo a um aumento no ladrilhamento do vídeo) menor a taxa de bits e o tempo de decodificação por ladrilho. O desvio padrão da taxa de bits apresenta um valor superior à média, pois esta média inclui todas as qualidades, o que resulta em uma variação de aproximadamente 20 vezes na taxa de bits. Além disso, a projeção cúbica possui uma taxa de bits média aproximadamente 8% menor do que a projeção equirretangular. Isto se deve a sua representação ser mais compacta e exigir menos pixels. Esta diferença também se reflete no tempo de decodificação dos ladrilhos que serão decodificados mais rapidamente. A

⁷<https://risa2000.github.io/hmdgdb/>

tabela também revela que as métricas de qualidade sofrem um sutil aumento com o aumento do ladrilhamento enquanto seu desvio padrão cresce de forma mais agressiva. Esta variação ocorre porque o codificador, configurado com o parâmetro CRF, ajusta a quantização afim de manter a qualidade uniforme⁸ e uma vez que o ladrilhamento isola regiões da projeção que possuem atividade espacial e temporal diferente, o codificação aplicará a quantização de forma diferente, aumentando o desvio padrão e deslocando a média. Já a métrica SSIM não apresenta diferença significativa entre os ladrilhamentos, apesar de seu desvio padrão aumentar com a redução dos ladrilhos. Quando comparamos as métricas de qualidade para as diferentes projeções, vemos que a projeção equirretangular apresenta m

Tabela 3.4: Valores de média e desvio padrão das métricas analisadas de acordo com o ladrilhamento, abrangendo todas qualidades.

Métrica	Ladrilhamento	CMP		ERP	
		Média	Desvio Padrão	Média	Desvio Padrão
Taxa	1x1	6.050.720	10.352.018	6.591.315	11.176.695
	3x2	1.030.891	2.012.731	1.130.073	2.281.468
	6x4	277.689	589.369	301.493	661.050
	9x6	133.972	273.547	144.640	318.278
	12x8	84.727	163.725	90.798	188.049
Tempo	1x1	0,629	0,278	0,757	0,311
	3x2	0,098	0,053	0,122	0,062
	6x4	0,022	0,015	0,026	0,017
	9x6	0,011	0,007	0,012	0,008
	12x8	0,006	0,004	0,008	0,005
MSE	1x1	21,472	33,788	17,782	28,024
	3x2	21,603	39,575	18,155	34,290
	6x4	21,923	45,121	18,409	39,623
	9x6	22,523	48,846	18,853	43,182
	12x8	22,613	50,499	19,034	45,037
SSIM	1x1	0,951	0,050	0,957	0,043
	3x2	0,951	0,060	0,957	0,055
	6x4	0,951	0,068	0,957	0,061
	9x6	0,951	0,071	0,957	0,064
	12x8	0,951	0,073	0,957	0,067
WS-MSE	1x1	21,784	33,965	20,473	33,051
	3x2	21,968	40,410	20,880	39,442
	6x4	22,420	45,615	19,578	42,162
	9x6	22,738	49,216	19,401	44,243
	12x8	22,758	50,740	19,343	45,660
	1x1	21,779	34,009	20,461	33,072

Continua na próxima página

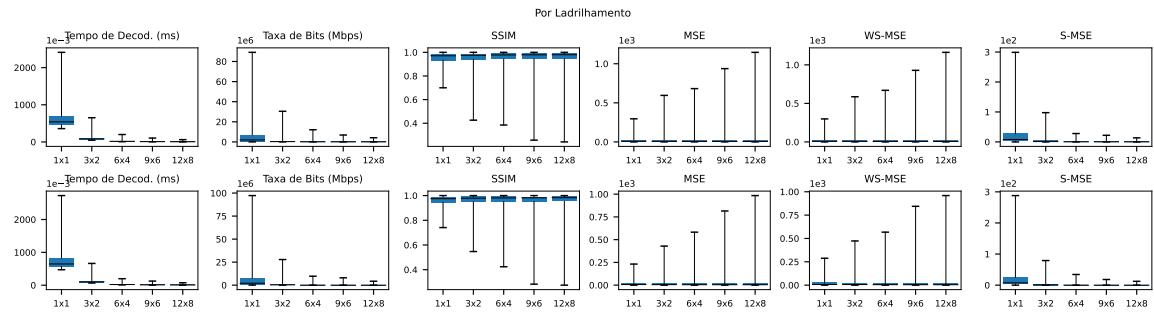
S-MSE

⁸<https://x265.readthedocs.io/en/master/cli.html>

Tabela 3.4 – Continuação da página anterior

	3x2	3,661	6,749	3,477	6,572
	6x4	0,935	1,904	0,883	2,084
	9x6	0,427	0,963	0,401	1,014
	12x8	0,241	0,568	0,228	0,597

A figura 3.5 mostra o boxplot para cada métrica. Cada caixa representa o segundo e o terceiro quartil e a linha interna representa a mediana. Os traços acima e abaixo da caixa representam os valores máximos e mínimos. Apesar do desvio padrão ser alto, a maior parte dos valores estão concentrados em torno da mediana, indicando que, apesar da alta variação nestas métricas, apenas em poucos casos ocorrerão desvios.

**Figura 3.5.** Boxplot do tempo de decodificação, taxa de bits e erro organizado por ladrilhamento.

A Figura ?? descreve, para cada padrão de ladrilho, as três melhores funções de densidade de probabilidade ajustadas à distribuição empírica do tempo de decodificação do ladrilho. A partir dos resultados, a distribuição Log-Normal está entre as três distribuições mais bem ajustadas em 100% dos padrões, enquanto as distribuições Gaussiana Inversa e Birnbaum-Saunders aparecem em 87,5% e 75% dos casos, respectivamente.

3.3.2 Estatísticas por nível de qualidade e ladrilhamento

A Figura ?? ilustra o tempo médio de decodificação do ladrilho e a taxa de bits média por nível de qualidade para cada padrão de ladrilho. A redução na taxa de bits correspondente a uma diminuição na qualidade do vídeo é acompanhada por uma diminuição semelhante, embora menos pronunciada, no tempo de decodificação dos blocos. Por exemplo, no cenário 6×4 , onde a taxa de bits diminui em 97,5% (de 1,91 Mbps para 47,6 Kbps) em uma faixa CRF de 16 a 46, o tempo de decodificação do bloco diminui apenas 63,1% (de 0,067 a 0,025 segundos) para

a mesma faixa CRF.

Analisando os resultados, fica evidente que as três distribuições mais bem ajustadas variam de acordo com o padrão de ladrilho. As frequências com que aparecem entre os três primeiros são as seguintes: Birnbaum-Saunders (79,2%), Gaussiana Inversa (72,9%), Log-Normal (70,8%), Burr Tipo XII (41,7%), Gama (8,33%), Pareto Generalizado (8,3%), Half Normal (6,3%), Exponencial (3,08%) e Rayleigh (2,1%).

Tabela 3.5 fornece os resultados da correlação entre o tempo de decodificação do bloco e a taxa de bits média em todos os casos. Notavelmente, para um determinado padrão de ladrilhos, a correlação tende a aumentar com níveis de qualidade mais elevados. Por outro lado, para um determinado valor de CRF, a correlação diminui à medida que o número de blocos por quadro aumenta.

Como ilustração, considerando o padrão de ladrilhos de 6×3 , a correlação é de aproximadamente 0,728 para CRF = 28. É digno de nota que esse padrão de ladrilhos específico demonstrou alcançar economias substanciais de largura de banda em pesquisas anteriores (GRAF *et al.*, 2017).

Tabela 3.5. Correlação entre o tempo de decodificação do bloco e a taxa de bits do bloco.

Padrão	Qualidade (CRF)					
	16	22	28	34	40	46
1 × 1	0.893	0.839	0.803	0.786	0.704	0.554
3 × 2	0.857	0.781	0.735	0.684	0.610	0.479
4 × 3	0.846	0.782	0.744	0.674	0.584	0.497
6 × 3	0.848	0.788	0.734	0.683	0.581	0.471
6 × 4	0.843	0.789	0.728	0.673	0.566	0.451
6 × 5	0.827	0.761	0.703	0.636	0.537	0.424
6 × 6	0.814	0.748	0.687	0.617	0.520	0.403
7 × 6	0.808	0.743	0.688	0.605	0.505	0.398

3.3.3 Estatísticas dos *Chunks* para o envio de todos o ladrilhos

Agora, nosso foco muda para o tempo necessário para decodificar sequencialmente todos os ladrilhos de um intervalo de tempo, considerando todos os vídeos e níveis de qualidade coletivamente. Desta forma poderemos simular o pior caso, onde o usuário terá que solicitar e processar toda a esfera. A Figura ?? ilustra o tempo médio de decodificação do pedaço e a

taxa de bits média para cada padrão de ladrilho, com barras de erro indicando o desvio padrão.

Como previsto, a taxa de bits apresenta um aumento correspondente ao aumento no número de blocos por quadro. Este resultado é atribuído ao fato de que a segmentação de blocos restringe o espaço de busca para a previsão de movimento do codificador. Inesperadamente, o tempo médio de decodificação do bloco se beneficia da segmentação lado a lado, com valores médios geralmente menores que aqueles observados para o caso 1×1 . Observa-se que o tempo médio de decodificação diminui à medida que o número de ladrilhos por quadro aumenta, atingindo um mínimo no ladrilhamento 6×3 . Esse mínimo é aproximadamente 10,8% menor que o padrão 1×1 , e essa melhoria é alcançada com apenas um aumento de 6,26% na taxa de bits. Além deste ponto, os valores do tempo de decodificação começam a subir novamente. As razões específicas por trás deste comportamento merecem uma investigação mais aprofundada, pois a complexidade do vídeo codificado reduz a medida que os ladrilhos diminuem de tamanho, tornando mais rápido a decodificação do conjunto, porém o número de bits aumenta, o que aumentaria o tempo de decodificação. Contudo, a exploração desta ocorrência está além do escopo deste trabalho.

Nossas medições revelam uma correlação significativa entre o tempo médio de decodificação do chunk e sua taxa de bits média em todos os padrões de ladrilhos. O coeficiente de correlação medido ultrapassa 0,96 em todos os casos. A Figura ?? ilustra as três distribuições de probabilidade com menor erro para os dados coletados em cada padrão de ladrilho.

Notavelmente, as distribuições Gaussiana Inversa e Birnbaum-Saunders estão consistentemente classificadas entre as três primeiras em 100% dos padrões, enquanto a distribuição Log-Normal aparece entre as três primeiras em 87,5% dos padrões. A distribuição Burr Tipo XVII surge como uma candidata potencial para o caso 1×1 . Infelizmente, os valores específicos dos parâmetros de distribuição para este caso não são apresentados devido a limitações de espaço.

3.3.4 Ladrilhos vistos (por viewport)

No primeiro cenário, para cada chunk haverá um conjunto de ladrilhos que aparecem no viewport. Ao longo da duração do chunk o usuário poderá mover a cabeça e o conjunto de

ladrilhos muda. Estes novos ladrilhos precisam já ter sido decodificados completamente para poderem ser exibidos. Desta forma é preciso considerar a trajetória do movimento de cabeça para se requisitar os ladrilhos. Sendo A o conjunto dos ladrilhos que são vistos pelo viewport no quadro f , os ladrilhos vistos lv durante a duração de um chunk será dada pela equação 3.2.

$$lv = \bigcup_{f=1}^{30} A_f \quad (3.2)$$

A taxa de bits de lv será igual a soma da taxa de todos os ladrilhos de lv . Para o tempo de decodificação de lv , podemos ter duas abordagens. Decodificação em série: os ladrilhos de lv são decodificados em série e o tempo total de decodificação de todos será igual a soma do tempo de todos os ladrilhos de lv . A segunda abordagem é a decodificação em paralelo: Neste caso o tempo de decodificação de lv é definido pelo maior tempo de todos os ladrilhos de lv . Já a qualidade, devemos considerar a média do MSE, SSIM, S-MSE e WS-MSE de todos os ladrilhos em lv . Por fim, uma seção de vídeo sv consiste de uma sequencia de lv com tamanho igual ao numero de chunks de um vídeo. Em sv temos todos os ladrilhos que foram vistos ao longo do vídeo separados em blocos de 1 segundo. Cada ladrilho tem associado sua posição espacial e temporal, indicada por um índice na lista de ladrilhos e pelo número do chunk que foi visto ao longo da reprodução. Como temos 30 usuários por vídeo, 28 vídeos, dois tipos de projeção e cinco tipos de ladrilhamento, teremos ao todo 8400 seções que podem ser reproduzidas considerando seis qualidades. Entre as métricas avaliadas, estão o número de ladrilhos utilizados pelo viewport ao longo da reprodução, taxa de bits, tempo de decodificação e qualidade do viewport e dos ladrilhos na projeção. Essa análise também pode ser estendida para diversas técnicas de seleção de ladrilho.

CAPÍTULO 4

CONCLUSÃO E TRABALHOS FUTUROS

Este trabalho apresentou uma modelagem de transmissão de streaming de vídeo em 360 graus com ladrilhos que consideravam diferentes padrões de ladrilhamento em uma ampla faixa de níveis de qualidade de vídeo (isto é, taxas de bits) e propriedades SI/TI. A partir dos resultados, as distribuições Log-Normal, Gaussiana Inversa e Birnbaum-Saunders ajustaram-se melhor aos dados experimentais na maioria dos casos. Tais distribuições são muito flexíveis e interessantes para fins de modelagem matemática (por exemplo, aplicação a modelos de filas). O tempo de decodificação do ladrilho está fortemente correlacionado com a taxa de bits do ladrilhos somente se a qualidade do vídeo for alta, e o grau de correlação diminui se o número de ladrilhos por quadro aumentar (ou seja, alta segmentação do bloco). O padrão de ladrilho 6×3 proporcionou o melhor equilíbrio entre o tempo de decodificação e a taxa de bits média, ao mesmo tempo que apresentou boas propriedades de correlação entre ambas as métricas se altos níveis de qualidade forem usados. Tal informação pode possivelmente ser usada por algoritmos ABR baseados em DASH para inferir o tempo de decodificação dos ladrilhos.

Como trabalho futuro planejamos projetar um algoritmo de adaptação de qualidade utilizando as métricas de desempenho analisadas e então avaliá-lo em um ambiente virtual de simulação de rede usando o simulador NS-3¹ conectado a servidores locais reais rodando em containers Docker. Considerando o registro de movimento de cabeça e que todos os segmentos acessados pelo cliente serão sempre os mesmos para cada usuário, será possível avaliar diferentes tipos de algoritmos sob diferentes infraestruturas sob as mesmas condições de usuário. Então, usando o modelo de qualidade desenvolvido neste trabalho será possível avaliar de forma objetiva a capacidade dos algoritmos se adaptarem às condições de rede em um ambiente bem controlado, como o mostra a figura 4.1.

¹<https://www.nsnam.org/>

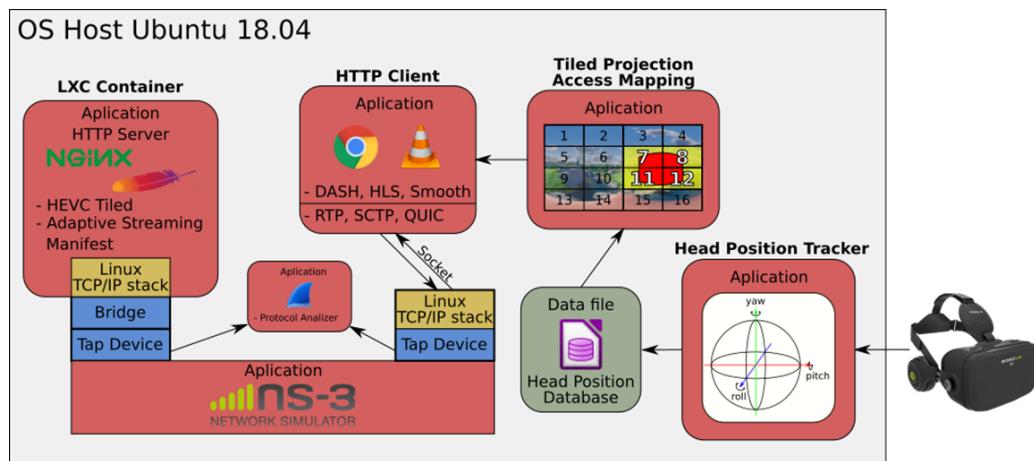


Figura 4.1. Modelo de simulação que será usado para avaliação

4.1 TRABALHOS FUTUROS

4.2 CRONOGRAMA

REFERENCES

- AFZAL, S.; CHEN, J.; RAMAKRISHNAN, K. K. Characterization of 360-degree Videos. In: *Proceedings of the Workshop on Virtual Reality and Augmented Reality Network - VR/AR Network '17*. New York, New York, USA: ACM Press, 2017. p. 1–6. ISBN 9781450350556. Disponível em: <<http://dl.acm.org/citation.cfm?doid=3097895.3097896>>. Cited 2 times in pages 1 and 2.
- CISCO. *2020 Global Networking Trends Report*. [S.l.], 2020. Disponível em: <https://www.cisco.com/c/m/en_us/solutions/enterprise-networks/networking-report.html>. No citation in text.
- GRAF, M.; TIMMERER, C.; MUELLER, C. Towards Bandwidth Efficient Adaptive Streaming of Omnidirectional Video over HTTP. In: *Proceedings of the 8th ACM on Multimedia Systems Conference - MMSys'17*. New York, New York, USA: ACM Press, 2017. p. 261–271. ISBN 9781450350020. Disponível em: <<http://dl.acm.org/citation.cfm?doid=3083187.3084016>>. Cited 2 times in pages 2 and 31.
- HOSSEINI, M.; SWAMINATHAN, V. Adaptive 360 VR video streaming based on MPEG-DASH SRD. In: *Proceedings - 2016 IEEE International Symposium on Multimedia, ISM 2016*. Institute of Electrical and Electronics Engineers (IEEE), 2017. p. 407–408. ISBN 9781509045709. Disponível em: <<https://ieeexplore.ieee.org/document/7823660>>. Cited in page 2.
- HUANG, T.-Y.; JOHARI, R.; MCKEOWN, N.; TRUNNELL, M.; WATSON, M. A Buffer-Based Approach to Rate Adaptation: Evidence from a Large Video Streaming Service. In: *Proceedings of the 2014 ACM conference on SIGCOMM - SIGCOMM '14*. New York, New York, USA: ACM Press, 2014. p. 187–198. ISBN 9781450328364. ISSN 19435819. Disponível em: <<http://dl.acm.org/citation.cfm?doid=2619239.2626296>>. Cited in page 3.
- NASRABADI, A. T.; SAMIEI, A.; MAHZARI, A.; FARIAS, M. C. Q.; CARVALHO, M. M.; MCMAHAN, R. P.; PRAKASH, R. A Taxonomy and Dataset for 360 deg Videos. In: *Proceedings of the 10th ACM on Multimedia Systems Conference - MMSys'19*. [s.n.], 2019. p. 2–7. ISBN 9781450362979. Disponível em: <<http://arxiv.org/abs/1905.03823>> <http://dx.doi.org/10.1145/3304109.3325812> <https://arxiv.org/abs/1905.03823> <http://dx.doi.org/10.1145/3304109.3325812>>. Cited 4 times in pages iii, 20, 27, and 28.
- NGUYEN, D. V.; TRAN, H. T. T.; THANG, T. C. An Evaluation of Tile Selection Methods for Viewport-Adaptive Streaming of 360-Degree Video. *ACM Transactions on Multimedia Computing, Communications, and Applications*, v. 16, n. 1, p. 1–24, apr 2020. ISSN 1551-6857. Disponível em: <<https://dl.acm.org/doi/10.1145/3373359>>. Cited in page 27.
- QIAN, F.; HAN, B.; XIAO, Q.; GOPALAKRISHNAN, V. Flare: Practical Viewport-Adaptive 360-Degree Video Streaming for Mobile Devices. In: *Proceedings of the 24th Annual International Conference on Mobile Computing and Networking - MobiCom '18*. New York, New York, USA: ACM Press, 2018. p. 99–114. ISBN 9781450359030. Disponível em: <<https://dl.acm.org/doi/10.1145/3240508.3240510>>.

//doi.org/10.1145/3241539.3241565http://dl.acm.org/citation.cfm?doid=3241539.3241565>. Cited in page 2.

QIAN, F.; JI, L.; HAN, B.; GOPALAKRISHNAN, V. Optimizing 360 video delivery over cellular networks. In: *Proceedings of the 5th Workshop on All Things Cellular Operations, Applications and Challenges - ATC '16*. New York, New York, USA: ACM Press, 2016. p. 1–6. ISBN 9781450342490. Disponível em: <<http://dl.acm.org/citation.cfm?doid=2980055.2980056>>. Cited in page 25.

SPITERI, K.; URGAONKAR, R.; SITARAMAN, R. K. BOLA: Near-optimal bitrate adaptation for online videos. In: *IEEE INFOCOM 2016 - The 35th Annual IEEE International Conference on Computer Communications*. IEEE, 2016. v. 2016-July, p. 1–9. ISBN 978-1-4673-9953-1. ISSN 0743166X. Disponível em: <<http://ieeexplore.ieee.org/document/7524428/>>. Cited in page 3.

TRAN, H. T. T.; NGOC, N. P.; PHAM, C. T.; JUNG, Y. J.; THANG, T. C. A subjective study on QoE of 360 video for VR communication. In: *2017 IEEE 19th International Workshop on Multimedia Signal Processing (MMSP)*. IEEE, 2017. v. 2017-Janua, p. 1–6. ISBN 978-1-5090-3649-3. Disponível em: <<http://ieeexplore.ieee.org/document/8122249/>>. Cited in page 15.

WANG, Z.; BOVIK, A.; SHEIKH, H.; SIMONCELLI, E. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, v. 13, n. 4, p. 600–612, 2004. Cited in page 12.

XIAO, M.; ZHOU, C.; SWAMINATHAN, V.; LIU, Y.; CHEN, S. BAS-360°: Exploring Spatial and Temporal Adaptability in 360-degree Videos over HTTP/2. *Proceedings - IEEE INFOCOM*, v. 2018-April, p. 953–961, 2018. ISSN 0743166X. Cited in page 2.

XU, M.; LI, C.; ZHANG, S.; CALLET, P. L. State-of-the-Art in 360° Video/Image Processing: Perception, Assessment and Compression. *IEEE Journal of Selected Topics in Signal Processing*, Institute of Electrical and Electronics Engineers (IEEE), v. 14, n. 1, p. 5–26, jan 2020. ISSN 1932-4553. Disponível em: <<https://ieeexplore.ieee.org/document/8960364/>>. Cited in page 15.

YADAV, P. K.; SHAFIEI, A.; OOI, W. T. QUETRA: A Queuing Theory Approach to DASH Rate Adaptation. In: *Proceedings of the 2017 ACM on Multimedia Conference - MM '17*. New York, New York, USA: ACM Press, 2017. p. 1130–1138. ISBN 9781450349062. Disponível em: <<http://dl.acm.org/citation.cfm?doid=3123266.3123390>>. Cited in page 3.

YIN, X.; JINDAL, A.; SEKAR, V.; SINOPOLI, B. A Control-Theoretic Approach for Dynamic Adaptive Video Streaming over HTTP. In: *Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication - SIGCOMM '15*. New York, New York, USA: ACM Press, 2015. p. 325–338. ISBN 9781450335423. Disponível em: <<http://dl.acm.org/citation.cfm?doid=2785956.2787486>>. Cited in page 3.

ZARE, A.; AMINLOU, A.; HANNUKSELA, M. M.; GABBOUJ, M. HEVC-compliant Tile-based Streaming of Panoramic Video for Virtual Reality Applications. In: *Proceedings of the 2016 ACM on Multimedia Conference - MM '16*. New York, New York, USA: ACM Press, 2016. p. 601–605. ISBN 9781450336031. Disponível em: <https://www.researchgate.net/publication/308820373_HEVC-compliant_Tile-based_Streaming_of_Panoramic_Video_>. Cited in page 3.

for _Virtual _Reality _Applications<http://dl.acm.org/citation.cfm?doid=2964284.2967292>>. Cited in page 2.