UNIVERSIDADE FEDERAL DE PELOTAS
Centro de Desenvolvimento Tecnológico
Programa de Pós-Graduação em Computação

Dissertação

HCLE: Codificador de *Light Fields* para Altas Taxas de Compressão com Predição Baseada em *Optical Flow* e *Phase Correlation*

Douglas Silva Corrêa

Pelotas, 2021

Douglas Silva Corrêa

HCLE: Codificador de *Light Fields* para Altas Taxas de Compressão com Predição Baseada em *Optical Flow* e *Phase Correlation*

Dissertação apresentada ao Programa de Pós-Graduação em Computação do Centro de Desenvolvimento Tecnológico da Universidade Federal de Pelotas, como requisito parcial à obtenção do título de Mestre em Ciência da Computação.

Orientador: Prof. Dr. Bruno Zatt
Coorientadores: Prof. Dr. Daniel Munari Vilchez Palomino
Prof Dr. Guilherme Ribeiro Corrêa

Pelotas, 2021

Universidade Federal de Pelotas / Sistema de Bibliotecas
Catalogação na Publicação

C823h Corrêa, Douglas Silva

HCLE : codificador de *Light fields* para altas taxas de compressão com predição baseada em *Optical flow* e *Phase correlation* / Douglas Silva Corrêa ; Bruno Zatt, orientador ; Daniel Munari Vilchez Palomino, Guilherme Ribeiro Corrêa, coorientadores. — Pelotas, 2021.

72 f.

Dissertação (Mestrado) — Programa de Pós-Graduação em Computação, Centro de Desenvolvimento Tecnológico, Universidade Federal de Pelotas, 2021.

1. Light fields. 2. Codificador. 3. Predição. 4. Optical flow. 5. Phase correlation. I. Zatt, Bruno, orient. II. Palomino, Daniel Munari Vilchez, coorient. III. Corrêa, Guilherme Ribeiro, coorient. IV. Título.

CDD : 005

Elaborada por Maria Inez Figueiredo Figas Machado CRB: 10/1612

Douglas Silva Corrêa

HCLE: Codificador de *Light Fields* para Altas Taxas de Compressão com Predição Baseada em *Optical Flow* e *Phase Correlation*

Dissertação aprovada, como requisito parcial, para obtenção do grau de Mestre em Ciência da Computação, Programa de Pós-Graduação em Computação, Centro de Desenvolvimento Tecnológico, Universidade Federal de Pelotas.

Data da Defesa: 20 de agosto de 2021

Banca Examinadora:

Prof. Dr. Bruno Zatt (orientador)
Doutor em Computação pela Universidade Federal do Rio Grande do Sul.

Prof. Dr. Paulo Roberto Ferreira Jr.
Doutor em Computação pela Universidade Federal do Rio Grande do Sul.

Prof. Dr. Ricardo Matsumura Araujo
Doutor em Computação pela Universidade Federal do Rio Grande do Sul.

Prof. Dr. Luciano da Silva Pinto
Doutor em Biotecnologia pela Universidade Federal de Pelotas.

AGRADECIMENTOS

Agradeço a todos que passaram pelo meu caminho até aqui. Novamente.

Welcome to your life
There's no turning back
TEARS FOR FEARS - EVERYBODY WANTS TO RULE THE WORLD

"Vou acabar é numa cova", ele diz. "E você também. Todos nós."
DESONRA - J. M. COETZEE

RESUMO

CORRÊA, Douglas Silva. **HCLE: Codificador de *Light Fields* para Altas Taxas de Compressão com Predição Baseada em *Optical Flow* e *Phase Correlation***. Orientador: Bruno Zatt. 2021. 72 f. Dissertação (Mestrado em Ciência da Computação) – Centro de Desenvolvimento Tecnológico, Universidade Federal de Pelotas, Pelotas, 2021.

Os *Light Fields* são uma tecnologia muito promissora para representação de sinais visuais digitais. Com ela, é possível capturar uma representação muito mais fiel da cena comparada às demais tecnologias. A gama de aplicações dos *Light Fields* varia de aplicações médicas até tecnologias imersivas relacionadas a realidade virtual. Porém, foi somente há poucos anos que começaram a ser desenvolvidas tecnologias que viabilizassem a captação de *Light Fields*, como por exemplo a câmera *Lytro Illum*. Mesmo assim, existem muitos desafios relacionados aos *Light Fields*, principalmente no que diz a respeito de sua compressão, já que os mesmos são um tipo de mídia digital que demanda uma enorme quantidade de dados para ser representada. Já existem estudos na literatura que procuram comprimir *Light Fields* para tornar seu uso viável. Algumas soluções utilizam o *Light Field* como pseudo-sequências de vídeo e o utilizam em codificadores de vídeo para explorar ferramentas já conhecidas. Outras desenvolvem técnicas novas que exploram características específicas dos *Light Fields* que não existem em outros tipos de mídias digitais. Porém, poucos trabalhos utilizam predição própria para *Light Fields*, deixando de explorar muitas de suas redundâncias específicas. Dessa maneira, esse trabalho propõe um codificador de *Light Fields* para altas taxas de Compressão com predição baseada em *Optical Flow* e *Phase Correlation* (HCLE). O codificador proposto atinge uma taxa de compressão de até 99,9996% em relação ao *Light Field* original, além de remover artefatos visuais inerentes à captura dos *Light Fields lenslet*, como por exemplo o efeito de *vignetting*. Em relação ao trabalho relacionado, o HCLE consegue reduzir a taxa de compressão em 35,22% para cada SAI, codificando todas as 225 SAIs de um *Light Field lenlet*. Para avaliação da qualidade visual do *Light Field*, foi utilizado o método BRISQUE, atingindo uma melhora de 56,21%. Também foi proposto um esquema para compensação de erro de predição utilizando o HCLE juntamente ao HEVC, tornando o codificador HCLE ainda mais competitivo para altas taxas de compressão.

Palavras-chave: Light Fields. codificador. predição. optical flow. phase correlation.

ABSTRACT

CORRÊA, Douglas Silva. **HCLE: Light Field Encoder for High-Compression Rates using Optical Flow and Phase Correlation-based Prediction**. Advisor: Bruno Zatt. 2021. 72 f. Dissertation (Masters in Computer Science) – Technology Development Center, Federal University of Pelotas, Pelotas, 2021.

Light Field is a very promising technology to represent digital visual signals. With Light Fields it is possible to capture and represent in a more faithful fashion the captured scenes compared to previous imaging technologies. The Light Fields have a extense range of applications, variyng from medical to immersive and virtual reality applications. However, it was only a few years ago that technologies emerged to make possible to capture Light Fields, like Lytro Illum camera. Despite the many challenges related to Light Fields manipulation, mainly related to compression, there are many works focusing on Light Fields compression to make it a feasible technology. Some works use Light Fields as videos to compress it in video encoders, to use the already implemented tools present in these encoders. Others develop new techniques in order to explore unique inherent characteristics of Light Fields. However, there are few works that utilize prediction to compress Light Fields. Thus, this work proposes a Light Field Encoder for High-Compression Rate susing Optical Flow and Phase Correlation-based Prediction (HCLE). The proposed encoder reaches a compression rate of 99.9996% in comparison to the original Light Field, besides removing visual artifacts that lenslet Light Fields originally have. Comparing to the related work, the HCLE reaches a compressiont rate of 35.22% for each SAI, generating all 225 lenslet Light Fields SAIs. To assess the Light Field visual quality, it was used the BRISQUE method, reaching a visual enhancement of 56.21%. It was also proposed a scheme to compensate prediction error utilizing the HCLE along with the HEVC, making the proposed solution more competitive to related works.

Keywords: Light Fields. encoder. prediction. optical flow. phase correlation.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

2D	2 dimensões
3DTV	*3D Television*
4D	4 dimensões
BDBR	*Bjøntegaard-Delta Rate*
BRISQUE	*Blind/Referenceless Image Spatial Quality Evaluator*
CABAC	*Context-adaptive Binary Arithmetic Coding*
DBP	*Decoded Picture Buffer*
DCT	*Discrete Cosine Transform*
HCLE	*High-Compression Light Field Encoding*
HDCA	*High-Density Camera Array*
HEVC	*High-Efficiency Video Coding*
HM	*HEVC Test Model*
HVS	*Human Visual System*
HOF	*High-Order Predictor*
HT-B	*Hexadeca-tree bitplane*
LF	*Light Field*
LOF	*Low-Order Predictor*
OF	*Optical Flow*
ME	*Motion Estimation*
MI	Micro-Imagem
MuLE	*Multidimensional Light Field Encoder Using 4D Transforms and Hexadeca-trees*
PC	*Phase Correlation*
PS	Pseudo-Sequência
PSNR	*Peak Signal-to-Noise Ratio*
QP	*Quantization Parameter*

QPS	Quadros por Segundo
RLE	*Run-Length Encoding*
SAD	*Sum of Absolute Differences*
SSIM	*Structural Similarity Index Measure*
SAI	*Sub-Aperture Image*
TSZ	*Test Zone Search*
UHD	*Ultra High Definition*
VAC	*Vergence-Accommodation Conflict*

SUMÁRIO

1 INTRODUÇÃO

O interesse da humanidade por representar a realidade em sua volta e documentar a história em forma de imagens data de milênios atrás. Evidências mostram que muito antes do humano moderno chegar na Europa, pinturas já eram feitas por Neandertais, datando de aproximadamente 64.000 anos atrás (HOFFMANN et al., 2018). Desde então, a arte plástica, representada por pinturas, esculturas, etc., está presente na vida do ser humano como forma de documentar o mundo que o cerca. Porém, a necessidade de retratar cada vez mais fielmente o mundo sempre será uma necessidade humana.

Porém, não foi há mais de 200 anos que surgiu o primeiro protótipo comercial de fotografia como conhecemos hoje. A daguerreótipo (DAGUERRE, 1839), surgiu como o primeiro processo fotográfico capaz de representar fielmente uma cena. Após isso, a evolução de como representamos e consumimos imagens evoluiu muito rapidamente, comparado ao tempo entre as primeiras pinturas rupestres e os primeiros daguerreótipos.

A partir desse momento, a maneira que capturamos e consumimos as representações da nossa realidade evoluíram drasticamente. Atualmente, imagens e vídeos digitais são facilmente capturados e reproduzidos pelos nossos *smartphones* e televisões. Além disso, a forma como os disseminamos também mudou completamente com os adventos das redes sociais e serviços de *streaming*.

O registro de fatos e momentos se tornou uma parte vital da nossa sociedade com a evolução da representação de imagens e vídeos. Entretanto, torna-se cada vez mais relevante a representação dessas, já que ao longo dos anos a utilização de imagens e vídeos passou a ser algo além de registro de fatos e entretenimento. Medicina, segurança e educação são somente alguns dos muitos nichos em que imagens e vídeos são essenciais (ARGYRIOU et al., 2015).

A importância com que capturamos e compartilhamos o nosso mundo continua crescendo. O volume de vídeos digitais via *streaming* já lidera o tráfego de dados na internet com 57,64% do volume total em 2020 (SANDVINE, 2020). Mas não só isso, cada vez mais temos a necessidade de representar o mais fielmente possível essas

imagens e vídeos. Uma forma de fazer é isso é aumentando a resolução dos vídeos. Hoje em dia é comum encontrarmos televisores e *smartphones* com suporte a vídeos em ultra-alta definição (UHD), que podem variar de 3840×2160 até 7680×4320 *pixels*.

Contudo, o ser humano vive e percebe o mundo em três dimensões (3D), e mesmo com o aumento de resolução e tecnologia dos *displays*, os vídeos e imagens digitais continuam sendo massivamente representados em dispositivos de visualização 2D. Dessa forma, continuamos em busca de ter uma maior fidelidade da representação do nosso mundo pelas imagens digitais. Devido a essa necessidade, os contínuos avanços tecnológicos possibilitam que tenhamos opções com maior imersão e interatividade com as imagens/vídeos digitais. Uma prova disso, é que nos últimos anos houve um aumento considerável no consumo de tecnologias imersivas, com previsão de continuar crescendo nos próximos anos (STATISTA, 2020).

Existem diferentes tecnologias que possibilitam essa imersão do usuário ao consumir imagens/vídeos digitais. Entre elas estão os vídeos 3D e vídeos 360°. Os vídeos 3D são capturados com múltiplas câmeras, deslocadas uma da outra, ou câmeras especiais que capturam diversos pontos de vistas. Isso possibilita que com o a tecnologia de representação adequada, seja possível ver esses diferentes pontos de vistas dando impressão de profundidade à cena. Já os vídeos 360°, são capturados com múltiplas câmeras com lentes de grande ângulo, tipicamente posicionadas em organização esférica, possibilitando a interpolação dessas imagens e formando a representação omnidirecional da imagem.

Enquanto os vídeos 360° estão sendo estudados há poucos anos, os vídeos 3D já são estudados há mais tempo, possibilitando diversos dispositivos com essa tecnologia, como televisores e videogames. Apesar disso, nos últimos anos, essa tecnologia está caindo em desuso devido a alguns motivos (CASS, 2014)(LESWING, 2017):

1. *Vergence-Accommodation Conflict*
   O *Vergence-Accommodation Conflict* (VAC) (HOFFMAN et al., 2008) é um problema causado devido ao objeto em que a visão do usuário converte está atrás ou à frente da cena, ainda assim é necessário focar a visão no anteparo em que a mídia está sendo exibida. Isso leva a uma percepção não natural e, consequentemente, limita a imersão do usuário. Além disso, muitos usuários relatavam problemas, como dor de cabeça, que aconteciam devido ao VAC.

2. Declínio da televisão 3D
   No início de 2010, a popularização da 3DTV (do inglês, *3D television*) ganhou um impulso devido ao filme Avatar (LESWING, 2017), fazendo com que grandes companhias começassem a investir na comercialização dessa tecnologia. Todavia, ao longo dos anos vemos essa tecnologia perdendo cada vez mais espaço no mercado. Alguns dos motivos para isso são listados abaixo:

- Para a utilização em televisores 3D em casa, na maioria das vezes a tecnologia era associada ao *Blu-ray*. Isso acontecia enquanto os serviços de *streaming* ganhavam popularidade, fazendo com que as pessoas não procurassem utilizar a tecnologia 3D. Além disso, na maiorias das vezes, devido a falta de maturidade dos *displays*, a tecnologia também estava aliada ao uso de óculos especiais, o que poderia incomodar ou dificultar o uso para algumas pessoas.
- Na maioria das vezes, a tecnologia 3D era *stereo*, ou seja, possuía somente dois pontos de vista para dar o efeito de profundidade, não possuindo uma imersão satisfatória, além da falta do *motion parallax* nessa tecnologia, que é a diferença de velocidade aparente de um objeto dependendo da sua distância do ponto de observação (BRINKMANN, 2008).
- Algumas pessoas não possuem visão estéreo, principalmente quando possuem problemas de visão associados a somente um olho, fazendo com que a percepção 3D dos vídeos não fosse satisfatória.

Devido a esses e outros problemas, as empresas começaram a investir em aumento de resolução ao invés de melhoria da tecnologia 3D (CASS, 2016). Porém, ainda há a necessidade de tecnologias imersivas para imagens e vídeos digitais para que possamos continuar melhorando a nossa representação do mundo com maior fidelidade. Dessa forma, a tecnologia de *Light Fields* (LF) surge como uma alternativa promissora em relação a outras tecnologias imersivas.

Diferentemente das tecnologias atuais de captura de imagens, onde somente a intensidade da luz é capturada em cada ponto do sensor, nos *Light Fields*, além da intensidade da luz, a direção com que os raios de luz incidem no sensor também é capturada. Desta forma, uma cena capturada como *Light Field* é representada de forma muito mais fiel do que quando se utilizam as tecnologias atuais/convencionais. Além disso, essas características possibilitam uma quantidade de técnicas de pós-processamento muito maior, como mudança de perspectiva, refoco, reiluminação, entre outros (IHRKE; RESTREPO; MIGNARD-DEBISE, 2016).

1.1 Motivação

Os *Light Fields* são baseados na função plenóptica (ADELSON; BERGEN, 1997). A função plenóptica descreve a luz em termos de posição, ângulos, comprimento de onda e tempo. Contudo, representar toda a dimensionalidade da função plenóptica é uma tarefa muito díficil. Dessa maneira, os *Light Fields* utilizam uma função reduzida derivada da função plenóptica que descreve os raios de luz considerando posição e

ângulo de incidência. Devido a grande capacidade de representação de cena obtida com os *Light Fields*, os mesmos podem ser utilizados em diferentes aplicações, desde câmeras comerciais (LYTRO, 2017)(RAYTRIX, 2017) até microscopia (LEVOY et al., 2006).

Mesmo com o avanço recente em termos de *software* e *hardware* relacionado a *Light Fields*, ainda existem muitos desafios a serem explorados. Um dos desafios é relacionado ao volume de dados necessário para representação de um *Light Field*. Considerando uma câmera *Lytro Illum*, cada *Light Field* capturado contém 625$\times$434 micro-imagens de tamanho 15$\times$15 *pixels*. Considerando 3 canais de cor e 10 *bits* por canal, um *Light Field* totalizaria 218,26MB de dados. Sendo assim, para tornar viável a manipulação, transmissão e armazenamento dos *Light Fields* é necessário que hajam soluções eficientes para compressão de *Light Fields* (ZHAO et al., 2016)(CONCEIÇÃO et al., 2018)(CARVALHO et al., 2018).

Existem estratégias para a compressão de *Light Fields* que o transformam em uma pseudo-sequência (PS) de vídeo e utilizam padrões de codificação de vídeos, como o *High Efficiency Video Coding* (HEVC) (JCT-VC, 2019), para fazer a compressão do LF. A vantagem dessas técnicas é utilizar ferramentas presentes nos codificadores de vídeo para fazer a compressão do *Light Field* sem a necessidade de criar novas ferramentas. A desvantagem é não explorar a grande dimensionalidade e as redundâncias presentes no *Light Field*. Sendo assim, estratégias tentam explorar a dimensionalidade do *Light Field* de forma a obter melhores resultados, como por exemplo, o codificador *Multidimensional Light Field Encoder using 4D Transforms and Hexadeca-trees* (MuLE) (CARVALHO et al., 2018). Nesse caso, o codificador explora as redundâncias 4D do *Light Field* utilizando a Transformada Discreta do Cosseno (DCT) em cada dimensão do *Light Field*. O MuLE também faz parte da padronização feita pelo JPEG para codificação de *Light Fields*, chamada de *JPEG Pleno Light Field Encoding*, e é chamado de *4D-Transform mode* (DE OLIVEIRA ALVES et al., 2020). Mesmo sendo uma solução específica para *Light Fields*, ainda há espaço para melhoria da compressão dos LFs utilizando predição para explorar ainda mais a redundância e reaproveitamento de dados.

Ainda que o MuLE compreenda o conceito de exploração de características específicas de *Light Fields*, o mesmo faz isso explorando transformadas 4D. Entretanto, existe espaço para explorar redundâncias em LFs para melhorar ainda mais suas taxas de compressão. Um exemplo disso, seria usando predição em *Light Fields* reaproveitando dados e se aproveitando de redundâncias no LF. Diferentes modelos de predição podem ser empregados para codificação de *Light Fields*. O mais direto é utilizar codificadores já conhecidos, como os codificadores de vídeo (CONCEIÇÃO et al., 2018). Nesse caso, o LF pode ser transformado em uma pseudo-sequência de vídeo para utilizar ferramentas e algoritmos de predição já conhecidos desses codificadores.

Uma outra maneira, seria desenvolver novos tipos de predição explorando redundâncias específicas dos *Light Fields* que não aparecem em vídeo digitais, por exemplo (MONTEIRO et al., 2017).

Mesmo com todos os esforços presentes na literatura com soluções para viabilizar o uso dos *Light Fields*, ainda há espaço para desenvolver predições específicas que exploram suas características únicas de redundância. Sendo assim, uma alternativa para elaboração de predição para *Light Fields* é utilizar *Optical Flow* (OF). O *Optical Flow* representa a distribuição aparente de movimento em uma imagem, sendo que o mesmo pode ser escalado para a disparidade de movimento entre duas imagens. O mesmo já foi empregado para estimação de movimento em imagens estáticas (WALKER; GUPTA; HEBERT, 2015), assim como na predição de movimento em vídeos, onde os quadros consecutivos, na sua maioria, são imagens muito similares (KRISHNAMURTHY; WOODS; MOULIN, 1999)(WEI; YIN; LIN, 2018). Dessa forma, o OF se mostra uma alternativa para explorar características únicas de redundância de um *Light Field*, já que o mesmo possui ângulos de visualização que possuem grande similaridade.

1.2 Objetivos

Apresentadas as necessidades para viabilizar tecnologias imersivas viáveis, as problemáticas envolvidas e algumas abordagens utilizadas para resolver, o objetivo central dessa dissertação é: *propor e desenvolver um codificador com perdas para Light Fields explorando predição baseada em Optical Flow e Phase Correlation focando em altas taxas de compressão*.

Como objetivos específicos, tem-se os seguintes pontos:

1. Descrever as redundâncias em *Light Fields* que possam ser exploradas para viabilizar sua compressão eficiente.

2. Propor um modelo de predição para *Light Fields* baseado em *Optical Flow* e *Phase Correlation*.

3. Desenvolver um esquema de codificação completo, desde a manipulação do *Light Field* original até a geração do *bitstream*, integrando o modelo de predição proposto ao codificador HEVC para codificação residual.

4. Desenvolver um decodificador capaz de reconstruir, com perdas, o *Light Field* a partir do *bitstream* gerado pelo codificador desenvolvido.

1.3 Organização

Este trabalho está organizado em seis capítulos. O Capítulo 2 apresenta o referencial teórico necessário para o desenvolvimento e entendimento do trabalho. São abordados conceitos básicos relacionados a *Light Fields* incluindo tipos de *Light Fields*, sua captura, nomenclaturas e problemas relacionados a transmissão e armazenamento. O Capítulo 2 ainda mostra conceitos básicos de codificação de vídeos que serão úteis para o entendimento do esquema de codificação de erros de predição proposto neste trabalho, além de conceitos de *Optical Flow* e *Phase Correlation*, que são utilizados no preditor proposto. Por fim, o capítulo finaliza com uma visão geral de trabalhos relacionados à codificação de *Light Fields* apontando suas vantagens e desvantagens.

O Capítulo 3 mostra a visão geral do codificador proposto, passando pelas análises que foram feitas para chegar nos algoritmos de cada uma das etapas de codificação e decodificação. O Capítulo 4 mostra um esquema desenvolvido para melhorar a qualidade geral dos *Light Fields* codificados com o esquema proposto baseado na codificação residual. O Capítulo 5 mostra os resultados experimentais obtidos e como os mesmo foram organizados. Por fim, o Capítulo 6 conclui esta dissertação com uma síntese do trabalho e aponta trabalhos futuros.

2 REFERENCIAL TEÓRICO

Antes de abordar as questões envolvidas nas proposições desta dissertação, serão explicados alguns conceitos necessários para a sua compreensão. Neste capítulo, serão abordados conceitos relacionados a *Light Fields* e suas aplicações atuais, passando pelo conceito geral de *Light Field*, como o mesmo se aplica às mídias digitais atuais, até os desafios relacionados a sua aquisição e viabilização. Após isso, será apresentada uma breve explicação sobre codificação de vídeos digitais, que serve como base para o codificador proposto, passando por redundâncias que podem ser exploradas nesse tipo de mídia. Continuando, serão explicados os conceitos relacionados à redundância que podem ser aplicadas nos dados referentes aos *Light Fields*. Além disso, também serão explicados os conceitos relacionados ao *Optical Flow* e *Phase Correlation*. Por fim, serão apresentados alguns trabalhos da literatura que propõem diferentes abordagens para compressão de *Light Fields*.

2.1 *Light Fields*: conceitos e nomenclaturas

2.1.1 Função Plenóptica

Os *Light Fields* derivam da função plenóptica (ADELSON; BERGEN, 1997). A função plenóptica é um modelo matemático que descreve toda a informação visual existente em qualquer ponto do espaço e qualquer instante de tempo. Para acomodar todas essas informações visuais, é necessário uma função de 7 dimensões.

$$LF = P(x, y, z, \theta, \phi, \gamma, t) \tag{1}$$

A Equação 1 expressa a função plenóptica. A descrição da luz se dá pelos raios de luz em toda posição possível (x, y, z), ângulo de incidência (θ, ϕ), comprimento de onda (γ) de cada raio de luz e o tempo (t) que o mesmo incide.

O interesse pela função plenóptica acontece principalmente de forma conceitual. Sendo assim, é necessário assumir algumas simplificações na função plenóptica para ser possível representar os LFs de forma digital.

2.1.2 *Light Fields* digitais

Lidar com a função plenóptica completa é uma tarefa difícil devido a grande dimensionalidade e falta de discretização de algumas dimensões. Sendo assim, para ser possível tratar computacionalmente e de forma eficiente os *Light Fields*, é necessário adicionar algumas restrições para simplificar a função plenóptica:

1. *Light Fields* estáticos não têm variação no tempo. Sendo assim, é possível remover a varável t da equação. *Light Fields* dinâmicos já são explorados na literatura (WILBURN et al., 2005) e serão utilizados como exemplos para possíveis problemáticas.

2. Para discretizar o comprimento de onda γ, utilizamos espaços de cores como RGB ou YCbCr.

3. Assume-se que os raios descritos na função plenóptica que incidem de todas as posições possíveis (x, y, z), propagam-se somente na direção de um ponto de vista, ou seja, na posição do dispositivo que captura o *Light Field*. Sendo assim, não há necessidade de possuir a profundidade descrita na função (z)

Com todas as simplificações citadas acima, a função plenóptica é simplificada como descrita na Equação 2:

$$LF = P(x, y, \theta, \phi) \tag{2}$$

Porém, como estamos lidando com *Light Field* como uma mídia digital, a equação é simplificada, discretizando θ e ϕ, já que os pontos de vista serão finitos e definidos pela forma que o dispositivo captura o *Light Field*, como mostrado na Equação 3:

$$LF = P(r, s, x, y) \tag{3}$$

Com a função plenóptica reduzida para uma função de 4 dimensões, é possível representar de forma digital os *Light Fields*. No caso da Equação 3, (r, s) denotam os ângulos de incidência possíveis, enquanto (x, y) denotam a posição de incidência da luz. Da mesma forma que em uma imagem digital, os LFs possuem três canais de cores, cada canal possui sua própria função.

Mesmo com as simplificações, os *Light Fields* ainda possuem muitas vantagens sobre o 3D convencional, possibilitando, por exemplo, mudança de perspectiva, refoco, entre outros (IHRKE; RESTREPO; MIGNARD-DEBISE, 2016). Isso faz com que os *Light Fields* sejam uma tecnologia imersiva muito promissora, já que a gama de aplicações é muito ampla.

2.1.3 Aquisição de *Light Fields*

Os *Light Fields* podem ser capturados atualmente por três principais maneiras:

1. *Array* de câmeras:
   Câmeras são dispostas lado a lado formando os diferentes pontos de vista possíveis. Dessa maneira, as câmeras precisam estar precisamente espaçadas entre si e sincronizadas para a aquisição fiel do *Light Field*. Para esse caso, os *datasets* capturados são chamados de HDCA (*High Density Camera Array*).

2. *Software*:
   Os *Light Fields* ainda podem ser artificialmente gerados a partir de computação gráfica, via *softwares* de modelagem. Os *Light Fields* gerados de forma artificial via *software* são chamados de *Light Fields* sintéticos.

3. Câmeras *Light Field*:
   Para esse caso, câmeras específicas para aquisição de *Light Fields* foram desenvolvidas. Exemplos delas são a *Lytro Illum* (LYTRO, 2017) e a *Raytrix* (RAYTRIX, 2017). Como as câmeras foram desenvolvidas especificamente para captação de *Light Fields*, essa maneira é a mais viável de capturar os mesmos, já que não é necessário a calibração e sincronização necessárias no *Array* de câmeras, assim como não é necessário ser gerado artificialmente via *software*. No caso de *Light Fields* capturados por essas câmeras específicas, os *Light Fields* são chamados de *lenslet*.

No caso dessas câmeras específicas para captação de *Light Fields*, as mesmas possuem um *array* de micro-lentes. Dessa forma, a lente principal captura os raios de luz vindo do objeto a ser capturado como mostra a Figura 1. Após isso, um *array* de micro-lentes direciona os raios de luz vindo de ângulos diferentes, gerando assim todos os ângulos possíveis que a câmera pode capturar, limitado pelo número de micro-lentes que o *array* possui.

Como as câmeras específicas para *Light Field* são a opção mais viável para capturar *Light Fields*, foram escolhidos como foco desse trabalho os *datasets* gerados a partir dessas câmeras, os *Light Fields lenslet*.

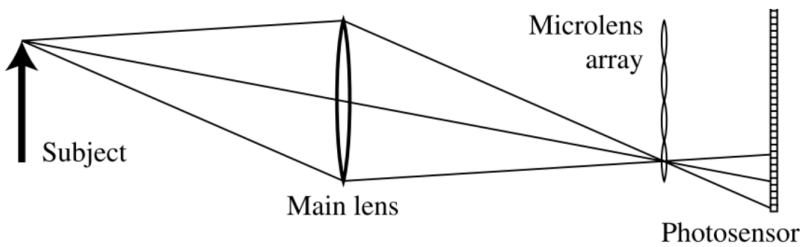

Figura 1 – Esquemático de uma câmera *Light Field* (NG et al., 2005)

2.1.4 Visualização e nomenclaturas dos *Light Fields*

Como as câmeras específicas para *Light Fields* são as opções mais viáveis para capturar esse tipo de mídia, os *datasets* escolhidos para ser o foco desse trabalho são os do tipo *lenslet*. Os *Light Fields lenslet* mais facilmente encontrados são capturados pela câmera *Lytro Illum*. A câmera *Lytro* possui um *array* de 15×15 micro-lentes, enquanto captura um ângulo completo com resolução de 625×434 *pixels*. Como os *Light Fields* derivam de uma função 4D, a visualização do mesmo torna-se um tarefa difícil de se imaginar, já que diferentes recortes podem ser feitos. Nesse trabalho, serão abordados dois tipos de corte: o primeiro é fixando os pontos de vista disponíveis (x, y) e variando os ângulos (r_0, s_0), o segundo é a inversão do primeiro caso, fixando o ângulo (r, s) e variando os pontos de vista (x_0, y_0).

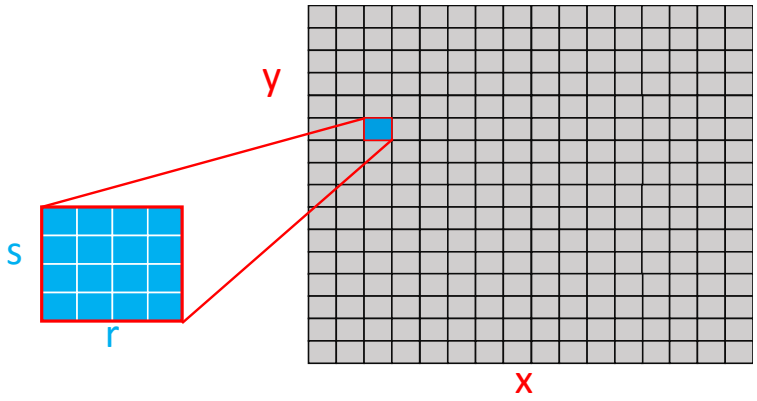

Figura 2 – Visualização conceitual de um *Light Field*

A Figura 2, mostra um modelo conceitual onde podem ser vistas as 4 dimensões de um *Light Field* de forma 2D. Na imagem, é possível notar que temos todos os pontos de vista (x, y) e todos os ângulos (r, s) representados por um *Light Field*.

Para imaginar um *Light Field* de forma que consigamos visualizar de maneira 2D, recortes nessas 4 dimensões podem ser feitos. Por exemplo, pode-se fixar os pontos de vistas e modificar os ângulos. Assim como também é possível fixar os ângulos enquanto modificam-se os pontos de vistas.

2.1.4.1 Visualização dos ângulos possíveis

A forma mais familiar de se visualizar um *Light Field* é decompondo o mesmo em suas porções 2D, fixando o ponto de vista, ou seja, fixando o ângulo. Nesse caso, como os ângulos são fixos (r_0, s_0) e variamos os pontos de vista (x, y), são geradas imagens 2D chamadas de SAIs (*Sub-Aperture Images*), como mostra a Figura 3. Cada SAI representa um diferente ângulo de vista que pode ser obtido fixando o ponto de vista.

Na Figura 3(a) podemos ver como um *Light Field* pode ser decomposto em suas porções 2D e visualizado de forma integral. Nesse caso, cada SAI representa um

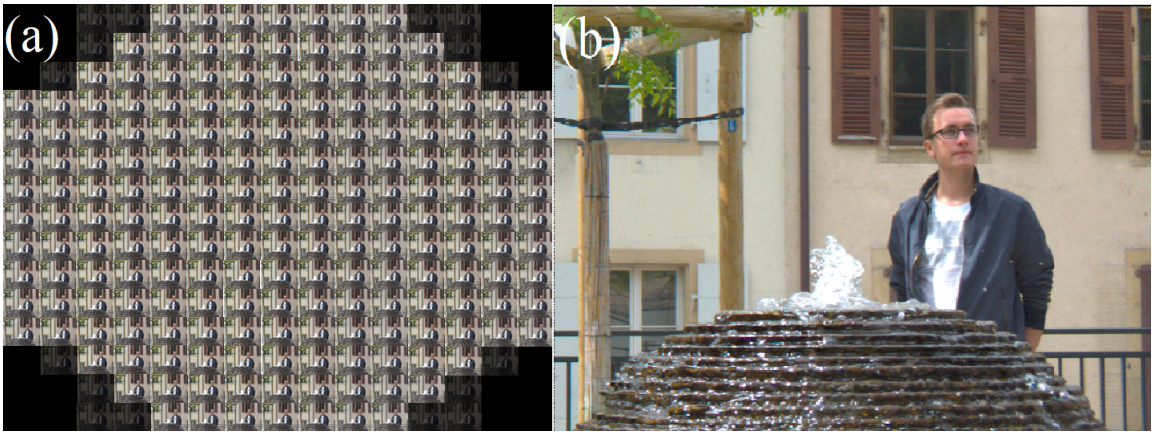

Figura 3 – (a) *Light Field Fountain_and_Vincent 2* decomposto em suas respectivas SAIs (b) SAI central.

ângulo capturado por uma câmera *Lytro Illum*. Cada SAI é um diferente ângulo de um mesmo ponto de vista, totalizando 225 ângulos diferentes. As SAIs das bordas que são totalmente pretas ou possuem efeito de opacidade são causadas pela captação de um *Light Field* com a câmera *Lytro Illum* e ocorre em todos os *Light Fields* capturados com essa câmera. Esse efeito de opacidade nas bordas do *Light Field* é chamado de *vignetting*.

Já a Figura 3(b) mostra somente uma SAI do *Light Field*. As SAIs não precisam ser visualizadas de maneira integral, podendo também ser visualizada somente uma SAI por vez. Sendo assim, todas as posições para um mesmo ângulo são visualizadas. Essa forma é a mais familiar para se visualizar um *Light Field*, já que nesse caso a SAI é uma imagem 2D convencional, com resolução de 625$\times$434 *pixels*.

Pode-se fazer uma relação com a Figura 3(a) com a Figura 2, onde temos todos os ângulos (r, s) representados em uma matriz. Dessa forma, podemos indexar todas as SAIs de um *Light Field* de acordo com a Equação 3. Por exemplo, a SAI central de um *Light Field* pode ser denotada pela Equação 4, onde (x, y) representam todos os pontos possíveis, simplificando para:

$$LF = P(7, 7) \tag{4}$$

2.1.4.2 Visualização dos pontos de vista

Além da visualização dos cortes 2D de um *Light Field*, o mesmo também pode ser visualizado com sua representação 4D em um espaço 2D, como mostra a Figura 3. Na Figura 3(a) uma porção do *Light Field Fountain_and_Vincent 2* é mostrado, onde é possível visualizar as micro-imagens (MI). Cada MI representa a luz incidente em um ponto no espaço para os diferentes ângulos de incidência. Ou seja, cada *pixel* de MI é o *pixel* colocalizado em diferentes SAIs. A Figura 4(b) mostra somente uma micro-imagem 15$\times$15 *pixels*, que é o tamanho padrão para a câmera *Lytro Illum*. As

bordas pretas ao redor das micro-imagens são devido à utilização da câmera *Lytro Illum*, e estão diretamente relacionadas as SAIs pretas e opacas da Fig. 1(a) do efeito de *vignetting*.

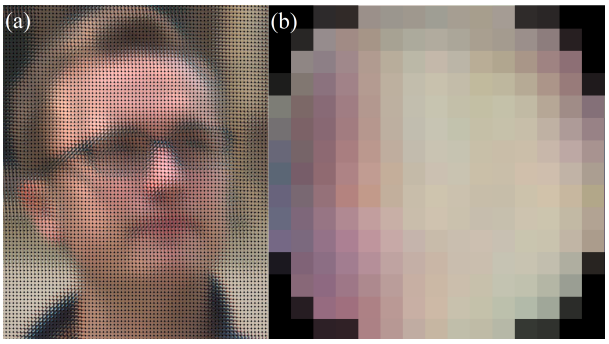

Figura 4 – (a) Porção do *Light Field Fountain_and_Vincent 2* visualizado a partir de suas micro-imagens (b) Micro-imagem.

2.1.5 Compressão

Visualizando o *Light Field* de diferentes maneiras, é possível perceber que o LF contém uma enorme quantidade de dados. Para exemplificar, caso seja necessário enviar o *Light Field* em sua forma crua, seria necessário enviar uma imagem de 9375$\times$6510 *pixels*. Considerando uma câmera *Lytro Illum*, cada *Light Field* capturado contém 625$\times$434 micro-imagens de tamanho 15$\times$15 *pixels*, considerando 10 *bits* por amostra e três canais de cores, um *Light Field* totalizaria 218,26MB de dados. Tomando um vídeo *Light Field* de duas horas e 30 quadros por segundo (QPS), o vídeo totalizaria aproximadamente 45TB. Com as tecnologias atuais de armazenamento e transmissão, seria inviável armazenar ou transmitir esses dados.

Com os dados apresentados acima, fica claro que é necessário aplicar técnicas de compressão para tornar os *Light Fields* uma tecnologia viável. Várias tipos de técnica já são aplicados e pesquisados na literatura, alguns desses casos serão apresentados no fim desse capítulo.

2.2 Fundamentos em compressão de imagens

Como esse trabalho tem como um dos objetivos o desenvolvimento de um codificador de *Light Field*, é essencial uma visão global nos conceitos por trás de compressão de imagens e vídeos. O exemplo usado será de vídeos digitais, já que existem mais tipos de redundância dentro de um vídeo digital que podem ser exploradas para reaproveitar dados. Essa seção fará uma apresentação do que é um vídeo digital. Após isso, serão apresentados os tipos de redundância que podem ser exploradas dentro de um vídeo digital. Por fim, um modelo genérico de codificador será apresentado, já que o modelo do codificador proposto segue um modelo parecido.

2.2.1 Vídeos Digitais

Um vídeo digital é composto por uma sequência de imagens que são capturadas a partir de uma cena e amostradas em uma determinada taxa, essa taxa é chamada de taxa de quadros por segundo. O QPS pode variar dependendo da aplicação, mas usualmente é utilizado na faixa de 15 até 60 QPS. Cada imagem nessa sequência, chamado de quadro, é composta por elemento onde a informação da imagem é salvo, o *pixel*. Outra característica importante em um vídeo é sua resolução. A resolução de um vídeo é definida pela quantidade de *pixels* verticais e horizontais em cada quadro. Novamente, a resolução do vídeo dependerá da aplicação, como por exemplo, em dispositivos móveis, onde o tamanho da tela é menor, uma resolução menor e um menor QPS não será tão prejudicial na experiência final do usuário. Ao contrário de aplicações onde o tamanho de exibição é grande, como por exemplo em um cinema, onde um resolução menor será mais perceptível.

2.2.1.1 Redundância de Dados em Vídeos Digitais

Como o objetivo de um codificador de vídeo é comprimir e codificar o vídeo para um tamanho aceitável para que seja possível manipulá-lo, armazená-lo e distribuí-lo, é necessário que a quantidade de dados necessário na representação do vídeo seja diminuída. Sendo assim, o codificador de vídeo pode se aproveitar de dados redundantes existentes dentro de um vídeo para aumentar a sua compressão. É considerado dado redundante aquele que não adiciona novas informações relevantes na representação de uma imagem. Na codificação de vídeos existem três diferentes tipos de redundância de dados: a redundância espacial, redundância temporal e redundância entrópica. Cada uma dessas redundância serão exploradas nas próximas subseções (AGOSTINI, 2007).

2.2.1.2 Redundância Espacial

A redundância espacial em um vídeo digital pode ser vista como a similaridade entre *pixels* ou regiões dentro de um único quadro. A correlação entre *pixel* ou regiões similares pode ser facilmente percebida no domínio espacial em regiões muito homogêneas de um quadro. Isso pode ser exemplificado como uma região da imagem que possui pouca variação de cor, textura ou luminosidade, ou seja, mais homogênea. Em uma região homogênea os *pixels* tendem a possuir valores muito similares ou até mesmo iguais, isso significa que o codificador de vídeo pode reaproveitar dados para representar determinadas regiões da imagem. Em um codificador de vídeo, essa etapa é chamada de codificação intra quadro. A redundância espacial de um vídeo também pode ser explorada no domínio das frequências, para isso, é necessário fazer a transformação da imagem do domínio espacial para o domínio das frequências.

2.2.1.3 Redundância Temporal

Como dito anteriormente, um vídeo digital costuma ter uma taxa de 15 a 60 QPS. Isso significa que um vídeo pode mostrar até 60 quadros em apenas um segundo. Considerando o ritmo com que objetos se movem usualmente no nosso cotidiano, pode-se inferir que existe pouca diferença entre um quadro e outro temporalmente falando. A redundância temporal pode ser exemplificada como um objeto que se move, que considerando o QPS, move-se pouco de um quadro para outro. Isso implica que *pixels* ou regiões de um vídeo podem ter tido seus valores pouco alterados ou até mesmo continuarem imutáveis, ou ainda uma região da imagem e seus *pixels* continuam exatamente os mesmos, porém com sua posição horizontal e vertical modificadas. Outro exemplo de redundância temporal, pode ser uma leve diferença na iluminação em uma determinada parte da imagem. Consequentemente, a exploração da redundância temporal em vídeos digitais leva a altas taxas de compressão em um vídeo, visto que acontece com frequência dada a taxa de QPS dos vídeos digitais atuais e também a possível falta de quebra de contexto entre quadros muito próximos. Analogamente com a redundância espacial, a etapa responsável pela exploração da redundância temporal é a codificação inter quadros.

2.2.1.4 Redundância Entrópica

A redundância entrópica está relacionada com a probabilidade de ocorrência de símbolos no sinal. Isso significa que o codificador de vídeo diminui a redundância de símbolos utilizando menos *bits* para representar os símbolos que ocorrem frequentemente, e mais *bits* para representar os símbolos que ocorrem raramente.

2.3 Modelo Genérico de um Codificador de Vídeo

Para ficar mais fácil de entender o codificador proposto e desenvolvido, será apresentado antes um modelo genérico de codificador de vídeo, que apresenta uma organização parecida. Para isso, nesta seção, utiliza-se um modelo genérico que se aproxima muito dos atuais codificadores de vídeos.

A Figura 5 mostra o modelo utilizado nesta seção para definir e explicar as etapas básicas de um codificador de vídeos.

É possível observar que existem duas entradas para o codificador de vídeo, a primeira é do quadro a ser codificado e a segunda de um quadro de referência que está na memória externa do dispositivo utilizado para codificar o vídeo. O quadro de referência é utilizado na codificação do quadro atual. Além disso, existe uma saída que é o *bitstream*. Este *bitstream* será enviado para o decodificador de vídeo que fará o processo de reconstrução de cada quadro para o vídeo ser exibido em algum dispositivo.

Primeiramente, o quadro atual a ser codificado é separado em blocos de mesmo

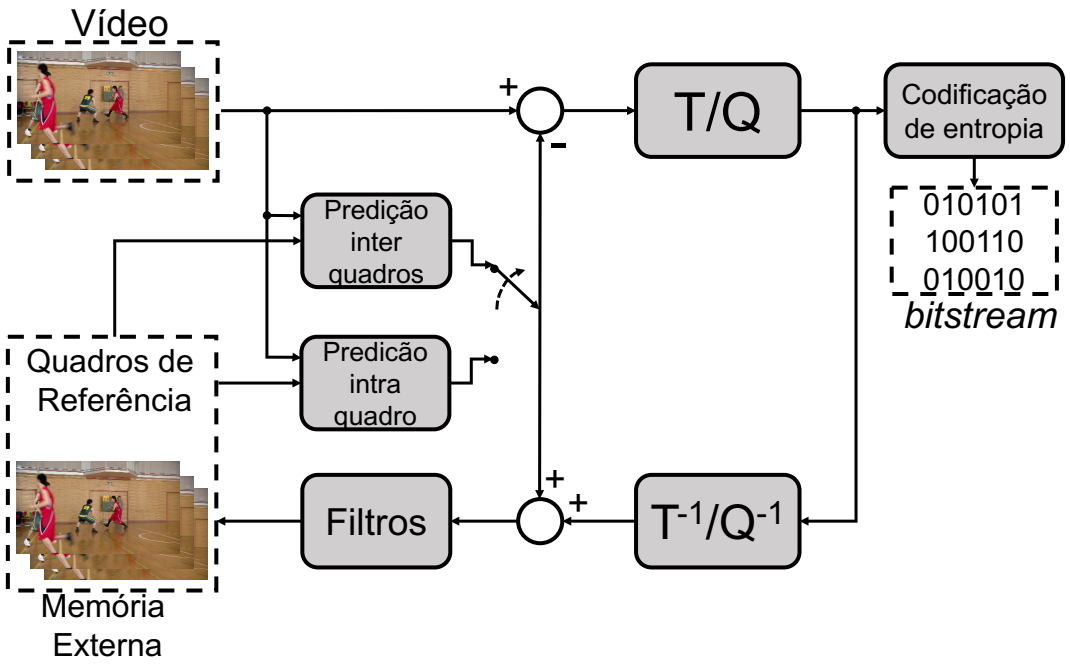

Figura 5 – Modelo genérico de um codificador de vídeos

tamanho, como mostrado na Figura 6, e após isso cada bloco é codificado separadamente. Cada bloco do quadro atual pode ser codificado utilizando a predição intra-quadro ou a predição inter-quadros. A predição intra-quadro é responsável por identificar e eliminar a redundância espacial procurando similaridades em blocos dentro do quadro atual que está a ser codificado.

Já a predição inter-quadros é responsável por identificar eliminar a redundância temporal procurando similaridades entre blocos no quadro atualmente sendo codificado e no quadro de referência já codificado que está armazenado na memória externa, como pode ser visto pelo quadrado preto indicado na Figura 7. É dentro da etapa de predição inter quadros que está presente a estimação de movimento (ME).

A ME é responsável pela detecção de movimentação entre quadros para achar blocos similares entre o quadro atual e o quadro de referência. Existem diversos algo-

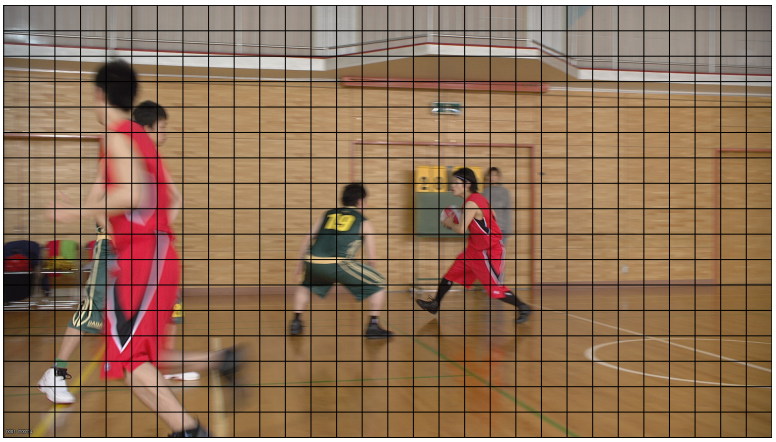

Figura 6 – Quadro do vídeo *BasketballDrive* separado em blocos do mesmo tamanho

ritmos que podem ser implementados para fazer a busca dentro da ME. Um exemplo básico pode ser visto também na Figura 7, onde o quadro atual procura pelo bloco mais parecido dentro de uma área de busca no bloco de referência. Assim, pode codificar somente a diferença, chamada de resíduo, entre os dois blocos e descartar a informação redundante. A escolha entre predição intra-quadro ou inter-quadros dependerá das características do vídeo.

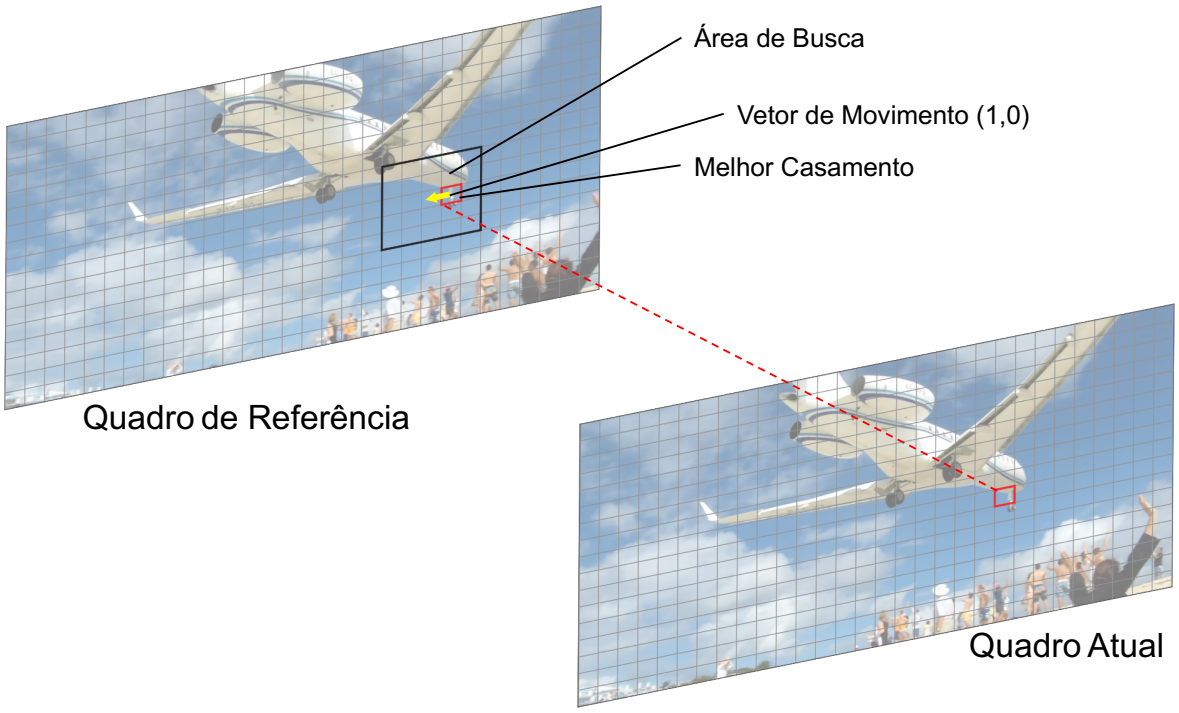

Figura 7 – Representação da estimação de movimento

Após passar pelas etapas de predição intra-quadro e inter-quadros, o bloco a ser codificado é subtraído do bloco predito e inicia o processo de codificação residual. Dentro da etapa de codificação residual existem duas principais etapas: a transformação e quantização. Na etapa de transformação, os resíduos são transformados do domínio espacial para o domínio das frequências. Após a transformação, os resíduos, já no domínio das frequências, passam pela quantização. Essa etapa é responsável por remover ou atenuar frequências que são pouco relevantes para o olho humano. Como o HVS (*Human Visual System*) é menos sensível a altas frequências, ao removê-las é possível alcançar alta compressão do vídeo a um baixo custo na qualidade final do vídeo. Porém, de qualquer maneira, adiciona perdas na codificação já que não é possível recuperar o sinal original.

Após o processo de codificação residual, cada bloco codificado passa pela codificação de entropia. Essa etapa, como dito anteriormente (Seção 2.2.1.4), é responsável por explorar a redundância entrópica. Ou seja, a codificação de entropia será responsável por representar o máximo possível de informação por símbolos codificados,

gerando o *bit stream* a ser transmitido ou decodificado.

As etapas de de transformação inversa e quantização inversa assim como os filtros são etapas do processo de decodificação que são necessárias dentro da etapa de codificação. As duas estão presentes pois é necessário reconstruir os quadros que serão utilizados como referência na codificação dos quadros subsequentes.

2.4 Redundâncias em *Light Fields*

Após a visão geral de codificação de vídeos, nesta seção serão apresentadas redundâncias específicas que podem ser exploradas em *Light Fields*. Mais especificamente, serão abordados dois tipos de redundância: redundância angular e redundância espacial.

2.4.1 Redundância Espacial

Como visto na Seção 2.3, os codificadores atuais de vídeo podem aplicar dois tipos de predição: predição intra-quadro e predição inter-quadros. A predição intra-quadro se refere a explorar redundâncias espaciais dentro de um mesmo quadro. Já a predição inter-quadros procura explorar redundâncias temporais buscando regiões similares de um quadro em outros quadros em diferentes instantes de tempo.

Diferentemente da redundância espacial explorada em vídeos, a redundância espacial funciona de uma maneira diferente em *Light Fields*, já que o *Light Field* pode ser manipulado em recortes diferentes, como explicado nas visualizações da Seção 2.1.4. Para exemplificar, será utilizado o corte explicado na Seção 2.1.4.2 para facilitar o entendimento da redundância espacial em *Light Fields*.

Tomando como exemplo as MIs que já foram mostradas na Figura 4, obtemos uma representação do 2D *Light Field* onde possuímos todos os ângulos e pontos de vistas. Uma parte dessa imagem é mostrada na Figura 8.

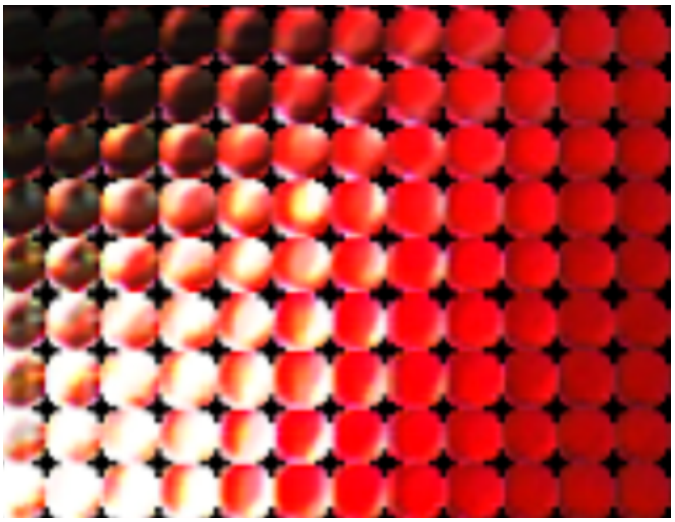

Figura 8 – MIs do *Light Field Bikes*

Na Figura 8, estão apresentadas algumas MIs do *Light Field Bikes* que representam vários ângulos (r_0, s_0) e várias posições (x_0, y_0). É possível notar que várias das MIs possuem o conteúdo muito similar, enquanto outros possuem o conteúdo divergente. Nesse caso, diferentemente da redundância espacial em vídeo digitais, que explorava redundância de dados dentro de um mesmo quadro, a redundância espacial em *Light Fields* explora a similaridades de um *Light Field* considerando pontos de vistas diferentes.

A predição intra-quadro de codificadores de vídeos digitais explora esse tipo de redundância copiando ou reaproveitando dados de regiões similares dentro de um mesmo quadro. Porém, no caso dos *Light Fields*, essa predição poderia ser feita, por exemplo, copiando ou reaproveitando dados entre MIs similares.

Alguns trabalhos da literatura já aplicam preditores para *Light Fields* explorando esse tipo de redundância (MONTEIRO et al., 2017), que será explicado com maior profundidade na seção 2.6.

2.4.2 Redundância Angular

Como o própria nome já diz, a redundância angular explora similaridades entre ângulos diferentes de um *Light Field*. Para exemplificar, será utilizado o corte explicado na Seção 2.1.4.1, onde possuímos variação dos ângulos mantendo o mesmo ponto de vista, visualizando as SAIs de um *Light Field*, como na Figura 3.

A Figura 9(a) representa a SAI central do *Light Field Stone_Pillars_Outside*, enquanto a Figura 9(b) representa a SAI logo a direita da SAI central, ou seja, a SAI de coordenadas (8, 7), isto é, são ângulos diferentes para o mesmo ponto de vista.

É possível notar pela Figura 9 que as SAIs são muito similares, sendo necessário analisar com maior atenção para começar a notar as diferenças entre elas. Sendo assim, uma maneira de explorar a redundância angular dentro de um *Light Field*, seria copiar ou reaproveitar informações similares que existem dentro das SAIs apresentadas. Um método similar entre à predição inter-quadros que codificadores de vídeos digitais utilizam.

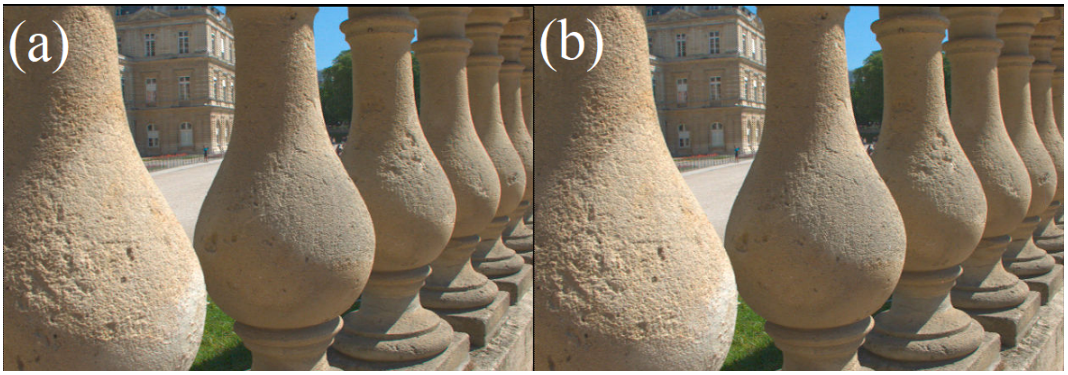

Figura 9 – (a) SAI central (b) SAI (8, 7) do *Light Field Stone_Pillars_Outside*

Pelo fato de ser um método similar à predição inter-quadros empregada em codificadores de vídeos, trabalhos da literatura transformam as SAIs de um *Light Field* em uma pseudo-sequência de vídeo e utilizam codificadores de vídeos, como por exemplo o HEVC (*High Efficiency Video Coding*) (SULLIVAN et al., 2012), para se aproveitar das ferramentas de predição já implementadas e conhecidas e fazer a compressão de *Light Fields* (CONCEIÇÃO et al., 2018). O problema é que as ferramentas de predição foram desenvolvidas para vídeos digitais e podem não explorar características específicas presentes nos *Light Fields*.

Para explorar características específicas de um *Light Field*, foram utilizados os algoritmos de *Optical Flow* e *Phase Correlation* no codificador proposto deste trabalho.

2.5 *Optical Flow* e *Phase Correlation*

Diferentemente de vídeos digitais, onde que de um quadro para outro objetos podem se mover em diversas direções, aproximar-se ou se distanciar da tela, desaparecer ou aparecer, ou até mesmo ocorrerem mudanças bruscas de contexto (e.g. a cena de um filme mudar de um interior para um exterior, mudando completamente o contexto entre as cenas), os *Light Fields* não possuam essas mudanças aleatórias.

Para tratar essas mudanças aleatórias de contexto de um vídeo digital, os codificadores de vídeo utilizam algoritmos de busca exaustiva para encontrar blocos mais similares entre si, ou até mesmo algoritmos rápidos como o *Test Zone Search* (TZS) para fazer a busca. Já nos *Light Fields*, essas mudanças de cena se dão pela distância entre as SAIs, no caso dos *lenslet*, ou a distância entre as câmeras, no caso dos *Ligth Fields* HDCA. Sendo assim, os objetos se moverão sempre na mesma direção dependendo de quais ângulos se modificam dentro de um *Light Field*. Usando a Figura 3 onde temos a visão integral das SAIs, os objetos se moverão mais ou menos, mas sempre na mesma direção, se usarmos como referência a SAI central em relação às SAIs (8, 7) e (14, 7). O mesmo pode ser dito de movimentos entre SAIs horizontalmente ou verticalmente.

Dessa maneira, optou-se pela utilização de *Optical Flow* e *Phase Correlation* para explorar essas diferenças fixas que existem entre diferentes SAIs de um *Light Field* na etapa de predição do codificador proposto neste trabalho.

2.5.1 *Optical Flow*

O *Optical Flow* (OF) (HORN; SCHUNCK, 1981) representa a distribuição aparente de movimento em uma imagem. Isto é, comparando duas imagens, como por exemplo dois quadros consecutivos de um vídeo onde não há mudança brusca de contexto, o OF representa o movimento aparente dos objetos de uma cena de um quadro para o outro. O OF é sempre calculado sobre um imagem em relação a outra imagem de

referência. Como os *Light Fields lenslet* possuem movimentos que são definidos pela distância entre as SAIs, o OF é uma boa maneira para calcular o movimento aparente dentro de um *Light Field*.

No contexto desta dissertação, o OF foi utilizado para calcular disparidades entre SAIs diferentes. Quando o OF é calculado, são fornecidos vetores de movimento para cada *pixel* da SAI, ou seja, um componente x e um componente y para cada vetor. A Figura 10 mostra o vetores do OF das SAIs (8, 7) e (6, 7), respectivamente, em relação a SAI central apresentados nas SAIs para melhor visualização.

Figura 10 – (a) OF da SAI (8, 7) (b) OF da SAI (6, 7) em relação a SAI central para o *Light Field Vincent_Fountain2*

Pela Figura 10 é possível notar que o movimento entre as SAIs é majoritariamente definido pela relação angular que a SAI calculada possui em relação a SAI de referência. No caso da Figura 10(a), a SAI (8, 7) está localizado no ângulo imediatamento à direita da SAI central. Sendo assim, é possível ver que o OF na sua maioria possui os vetores com a maior parte no componente x.

Já na Figura 10(b), como a SAI (6, 7) está angularmente posicionado imediatamente à direita da SAI central, os vetores do OF ainda possuem quase na totalidade toda a magnitude no componente x, porém com o módulo invertido em relação ao OF da SAI (8, 7).

Como o cálculo do OF não se dá de maneira exata, vários métodos estão disponíveis na literatura para a aproximação do OF. No contexto desta dissertação, o método escolhido foi o *Horn-Schunk* (HORN; SCHUNCK, 1981), já que o mesmo já é amplamente difundido no meio científico e mostra que possui uma baixa taxas de erro mesmo comparado a outros métodos de aproximação de OF (HARTMANN et al., 2018).

Como o comportamento do OF se mostra regular quando calculado entre SAIs de um *Light Field*, o mesmo pode ser utilizado para explorar suas características específicas, como por exemplo, o movimento regular entre diferentes SAIs. Dessa forma, o OF é uma boa alternativa para um possível algoritmo de predição em *Light Fields*, já que explora de maneira eficiente as redundâncias angulares citadas na Seção 2.4.2.

2.5.2 *Phase Correlation*

Phase Correlation (PC) representa o movimento de translação entre duas imagens (KUGLIN; HINNE, 1975). Diferentemente do OF, o PC representa a disparidade, em termos de *pixels*, de uma imagem para outra com somente um vetor (componentes x e y).

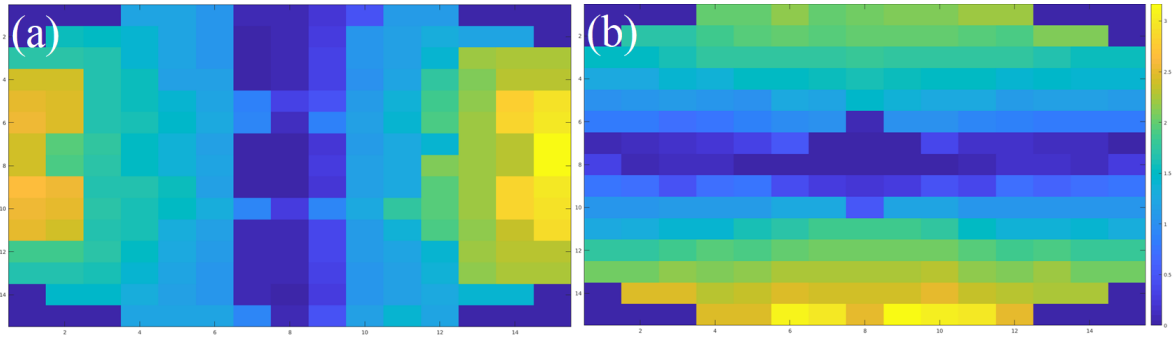

Figura 11 – (a) PC horizontal (b) PC vertical das SAIs em relação a SAI central para o *Light Field Fountain_Vincent2*

Na Figura 11, que mostra o PC de todas as SAIs calculadas em relação a SAI central, é possível notar que para as SAIs que estão deslocadas horizontalmente em relação a SAI central (Figura 11(a)), possuem um deslocamento regular, tanto para esquerda como para a direita, enquanto as SAIs verticalmente alinhadas, possuem menos deslocamento. Também é possível notar que quanto mais próximo da SAI central, menor o PC das SAIs. O mesmo se repete para a Figura 11(b), porém, nesse caso, para as SAIs verticalmente deslocadas em relação a SAI central.

Juntamente ao OF, o PC pode ser últil também para explorar redundâncias angulares nos *Light Fields* compensando possíveis erros no cálculo do OF, já que possui somente um vetor de movimento para toda a imagem. Assim, em regiões da imagem que são totalmente regulares e possuem diferentes OF, o PC pode corrigir esse erro sendo utilizado, por exemplo, para escalar os vetores do OF.

2.6 Trabalhos Relacionados

Como discutido em seções anteriores, existem muitas problemáticas relacionadas aos *Light Fields*, desde captação a transmissão. Um dos principais problemas, é a representação de um *Light Field* de forma digital, já que o mesmo pode ocupar armazenamento fora da realidade atual de *hardwares* disponíveis. Dessa forma, existem muitos trabalhos na literatura que utilizam diferentes métodos para compressão de *Light Fields*. Para dar uma visão geral do que é proposto no meio científico para *Light Fields*, os trabalhos foram separados em dois tipos: métodos que usam características genéricas de compressão de imagens digitais e métodos que utilizam características específicas dos *Light Fields*.

2.6.1 Métodos genéricos para compressão de *Light Fields*

Trabalhos que usam métodos genéricos para compressão de *Light Fields* são aqueles que utilizam ferramentas já conhecidas, como por exemplo, codificadores de vídeos. Nesse caso específico, as SAIs de um *Light Field* podem ser transformadas em uma pseudo-sequência (PS) de vídeo e codificado com algum codificador de vídeos. Como o codificador de vídeo utilizará a predição intra-quadro e inter-quadros, e nesse segundo caso, o número de quadros em que a busca é feita é limitado, a ordem em que a pseudo-sequência de vídeo é gerada mudará o resultado da codificação.

Uma pseudo-sequência de vídeo pode utilizar diferentes ordens, como por exemplo ordem *raster*, circular ou zigue-zague. Para tentar melhorar como a sequência é gerada, em (CONCEIÇÃO et al., 2018) é utilizado um novo algoritmo que define a ordem da pseudo-sequência de vídeo pela maior similaridade entre as SAIs, definida pela *Sum of Absolute Differences* (SAD).

Algoritmo 1: Algoritmo LF-CAE (CONCEIÇÃO et al., 2018)

```
Data: sais_list: SAIs do Light Field
Result: ordered_sais_list: pseudo-sequência com SAIs ordenadas por
        similaridade
ordered_sais_list.add(central_sai)
while length(sais_list) > 0 do
    i = 0
    while i < length(sais_list) do
        curr_sai = sais_list[i]
        sad = compare_sais(curr_sai, ordered_sais_list)
        if sad < previous_sad then
            best_sai = curr_sai
            previous_sad = sad
        end
        ordered_sais_list.add(best_sai)
        sais_list.remove(best_sai)
    end
end
```

O Algoritmo 1 é um pseudo-algoritmo adaptado de (CONCEIÇÃO et al., 2018) para gerar uma pseudo-sequência de vídeo que agrupe as SAIs com maior similaridade em sequência para tirar maior proveito da predição inter-quadros do HEVC.

Nesse algoritmo, a SAI central é adicionada como primeira da pseudo-sequência de vídeo. Após isso, todas as SAIs disponíveis são comparadas com as SAIs que serão usadas como referência (SAIs já adicionadas na PS). A SAI que possuir o menor SAD é adicionada como próxima na PS de vídeo. O processo é repetido após todas

as SAIs terem sido adicionadas na PS de vídeo.

O método LF-CAE apresenta uma redução de 6,34% em relação a ordem *raster* e 20,5% em relação a ordem circular em termos de BDBR (*Bjøntegaard-Delta Rate*). Mesmo possuindo bons resultados, o trabalho em questão utiliza o HEVC para codificar a PS de vídeo gerada, não explorando características específicas de *Light Fields*.

2.6.2 Métodos específicos para compressão de *Light Fields*

Métodos específicos para compressão de *Light Fields* são aqueles que usam características específicas dos *Light Fields* para gerar novas solução para compressão dos mesmos, em vez de usar métodos de codificação já conhecidos, como o HEVC. Nas subseções seguintes serão apresenta métodos que se enquadram nessa categoria, porém utilizam diferentes abordagens com e sem predição.

2.6.2.1 Predição em Light Fields

Um exemplo de método que usa predição para *Light Fields*, é o método desenvolvido em (MONTEIRO et al., 2017). Nesse método, é feita uma predição das MIs com base em duas etapas: *Low-Order Predictor* (LOF) e *High-Order Predictor* (HOF).

A primeira etapa de LOF, é responsável por achar o bloco de MIs que possui a maior similaridade com os bloco a ser codificado. Essa busca é feita de forma exaustiva e indica um bloco de referência que será usado na etapa de HOF.

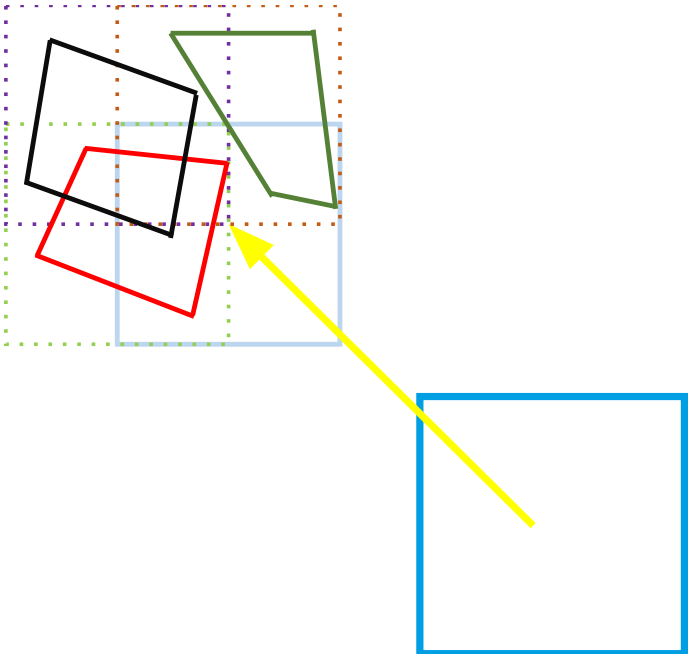

Figura 12 – Etapa de HOF em (MONTEIRO et al., 2017)

A Figura 12 mostra a etapa de HOF presente em (MONTEIRO et al., 2017). O quadrado azul da figura indica o bloco de MIs a ser predito (o bloco possui 4 MIs de tamanho). A seta amarela indica o bloco de referência encontra na etapa de LOF. A

partir do bloco de referência, 3 blocos são gerados, deslocando o bloco de referência nas 3 direções indicadas por 1 MI. Após isso, são gerados quadriláteros a partir de transformações geométricas do bloco de referência. O quadrilátero que possuir a menor distorção em relação ao bloco de referência é utilizado como bloco predito.

Como esse método utiliza características específicas de *Light Fields*, atinge uma redução de 33,35% de *bitrate* em relação ao HEVC comum, mostrando que utilizar características específicas do *Light Field* para gerar etapas de predição é um método promissor para compressão de *Light Fields*.

2.6.2.2 Método específico sem predição

Ainda existem soluções que exploram características intrínsecas de um *Light Field* mesmo sem fazer predição. É o caso da solução proposta em (CARVALHO et al., 2018), o *Multidimensional Light Field Encoder Using 4D Transforms and Hexadeca-trees* (MuLE).

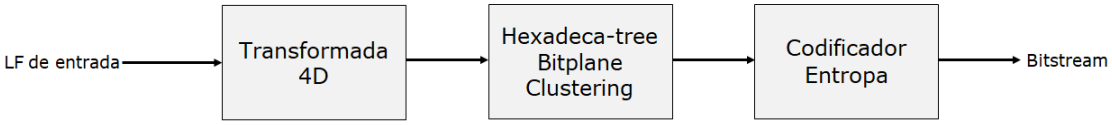

Figura 13 – Codificador MuLE adaptado de (CARVALHO et al., 2018)

A primeira etapa do codificador MuLE é separar o LF de entrada em blocos 4D de $15\times15\times13\times13$ *pixels*. Após isso, a segunda etapa do codificador é uma DCT-4D (*Discrete Cosine Transform 4D*). A DCT-4D aplica uma DCT em cada dimensão do LF de entrada. Após a etapa de transformada 4F, é aplicada uma etapa de quantização que é chamada de *Hexadeca-tree bitplane* (HT-B), maiores detalhes podem ser encontrados em (CARVALHO et al., 2018). Por fim, um codificador de entropia baseado em codificação aritmética é empregado para gerar o *bitstream* do *Light Field* codificado.

Pelo fato de explorar de forma orgânica as redundâncias 4D de um *Light Field*, o MuLE atinge resultados melhores de BDBR em relação ao HEVC e VP9. Porém, o MuLE não explora nenhum tipo de predição fazendo uso das redundâncias espaciais e angulares presentes nos *Light Fields*.

3 CODIFICADOR DE *LIGHT FIELDS* COM PREDIÇÃO BASEADA EM *OPTICAL FLOW* E *PHASE CORRELATION*

As seções anteriores mostraram todo o referencial teórico sobre *Light Fields* e as problemáticas envolvidas em capturar, representar, armazenar e comprimi-los. Também foram apresentados conceitos de compressão de imagens digitais, que podem servir como base para codificação de *Light Fields*. Foi explicado que os conceitos atuais que existem para compressão de *Light Fields*, mesmo funcionando, deixam espaço para explorar características inerentes de um *Light Field*, como redundância especial e redundância angular. Também foram apresentadas técnicas que exploram as redundâncias e características específicas dos *Light Fields*, o *Optical Flow* e o *Phase Correlation*.

O principal objetivo dessa dissertação é propor um codificador para *Light Fields* focado em altas taxas de compressão, utilizando métodos que explorem as particularidades presentes nos *Light Fields* que já foram explicadas em seções anteriores, o mesmo foi desenvolvido como um *codec* (codificador e decodificador). Sendo assim, propõe-se um codificador completo para *Light Fields* que utiliza predição baseada em *Optical Flow* e *Phase Correlation*. O codificador proposto foi nomeado de HCLE (do inglês *High-Compression Light-Field Encoding*), pois o mesmo atinge altas taxas de compressão. Ainda, elimina alguns problemas relacionados aos *Light Fields lenslet* como, por exemplo, o efeito de *vignetting*.

Esse capítulo será organizado da seguinte forma: será apresentada uma visão geral do codificador, primeiro mostrando a etapa de codificação e após a etapa de decodificação. Após a visão geral, cada etapa da codificação e da decodificação será explicada mais detalhadamente, juntamente com as decisões de projetos tomadas.

3.1 Visão geral do *coded*

Como o HCLE foi desenvolvido como um *codec*, primeiramente será apresentado uma visão geral da etapa de codificação seguida da explicação de cada passo presente nela. Após isso, a etapa de decodificação e seus passos será detalhada.

3.1.1 Etapa de Codificação

A etapa de codificação é responsável por comprimir o *Light Field*, isto é, o *Light Field* que será utilizado como entrada, no caso do HCLE precisa estar representado em sua visualização de SAIs (ver Seção 2.1.4.1), será codificado e a etapa de codificação gerará o *bitstream* comprimido desse *Light Field*. A Figura 14 mostra a visão geral da etapa de codificação do HCLE com todos os seus passos.

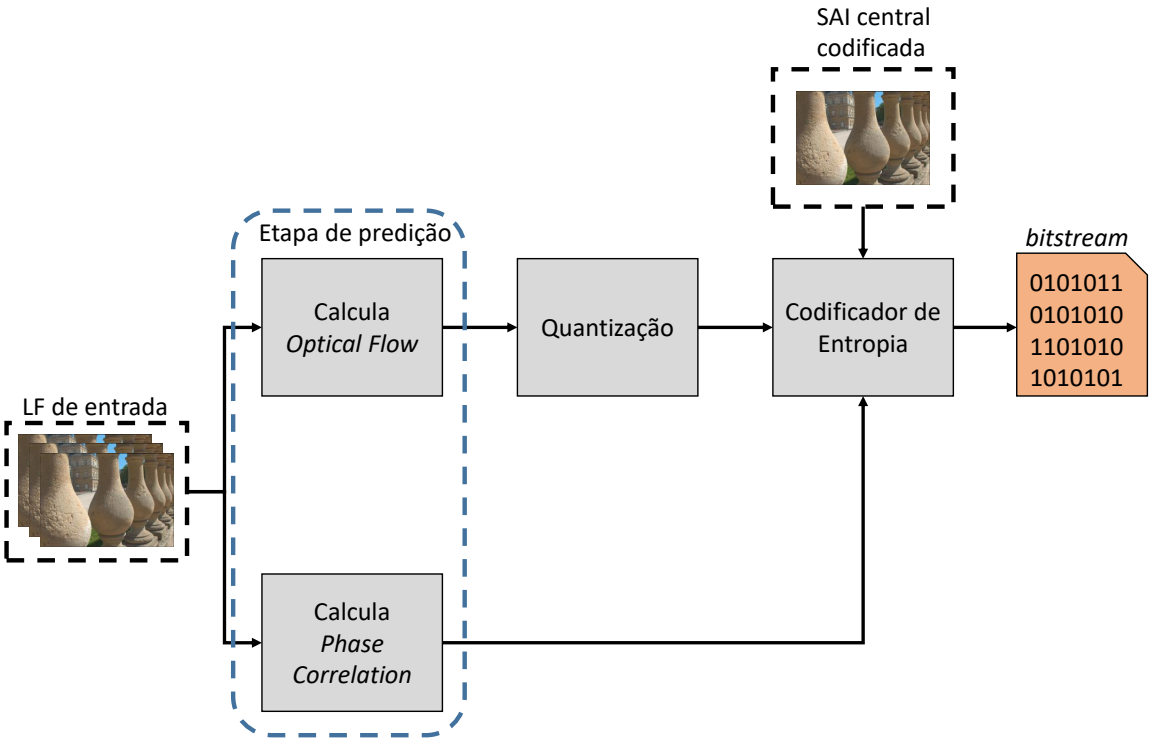

Figura 14 – Visão geral das etapas de codificação do HCLE

A Figura 14 apresenta diferentes etapas de processos do codificador HCLE:

1. Predição: responsável por gerar a predição baseada em *Optical Flow* e *Phase Correlation*. Essa etapa possui duas etapas menores:
   - Calcula *Optical Flow*: Calcula o *Optical Flow* para cada amostra das SAIs.
   - Calcula *Phase Correlation*: Calcula o *Phase Correlation* para cada SAI.
2. Quantização: essa etapa é utilizada para remover informações desnecessárias ou irrelevantes do cálculo de *Optical Flow*, diminuindo a quantidade de símbolos que precisam ser representados e adicionados ao *bitstream*.
3. Codificador de Entropia: responsável por gerar o *bitstream* com todas as informações necessárias (*Optical Flow* quantizado, *Phase Correlation* e a SAI central codificada).

3.1.1.1 Predição

A Predição do HCLE é composta por duas etapas: Calcula *Optical Flow* e Calcula *Phase Correlation*.

A etapa Calcula *Optical Flow* é responsável por calcular o *Optical Flow* das SAIs em relação a SAI central (8, 8). Relembrando da Seção 2.5.1, o OF calcula o movimento aparente entre duas imagens. Esse movimento é dado por um vetor (componentes V_x e V_y). Sendo assim, é possível visualizar esses vetores gerados pelo OF em uma SAI, como mostra a Figura 15.

Figura 15 – Recorte da SAI central do *Light Field Fountain_Vincent2* com o OF da SAI (6, 7)

A Figura 15, mostra um recorte da SAI central do *Light Field Fountain_Vincent2* com o OF representado para visualização em relação à SAI (6, 7). A SAI (6, 7), relembrando da Figura 2, é a SAI imediatamente à esquerda da SAI central. Pela Figura 15, parece que somente alguns *pixels* da SAI possuem vetor de OF, porém, o OF é calculado para todas as amostras de uma imagem, isto é, todos os *pixels*. O que acontece, é que regiões mais homogêneas, como a parede ao fundo da pessoa, não possuem movimento aparente, já que são regiões muito uniformes. Isso faz com que os vetores de OF dessa região possuam magnitudes irrisórias, nem mesmo sendo possíveis de visualizar na representação da imagem. Esse comportamento acontece tanto para SAIs horizontalmente quanto verticalmente alinhadas.

Outro ponto a ser salientado, é que existem pouquíssimos vetores que possuem direção vertical. Isso acontece pois a SAI (6, 7) está à esquerda da SAI central, ou seja, o ângulo variou somente horizontalmente, fazendo com que os objetos da cena também só apresentem deslocamento nessa mesma orientação. O poste posicionado à esquerda da Figura 15 representa bem esse comportamento, onde todos os vetores do OF estão praticamente perpendiculares ao poste. Isso significa que a maioria dos vetores desse OF possuem o componente V_x com alta magnitude relativa, enquanto o componente V_y possui um valor irrisório.

Considerando o âmbito digital, os componentes do vetor do *Optical Flow* são calculadores e armazenados em valores *double*, que por sua vez possuem 8 *bytes* de

tamanho na maioria das linguagens de programação. Considerando *Light Fields lenslet* que possuem 225 SAIs no total de resolução 625$\times$434 cada, e que o OF possui 625$\times$434 vetores para cada SAI, sendo que cada vetor possui duas componentes V_x e V_y que são um *double* de 8 *bytes*, seria necessário 972,16MB para armazenar todas as informações de OF de todas as SAIs, excluindo a SAI central, o que vai contra a ideia de compressão de *Light Fields*.

Porém, como explicado a partir da Figura 15, SAIs que se encontram horizontalmente ou verticalmente alinhadas, possuem praticamente toda a magnitude do vetor em somente uma das componentes. Isso significa que para essas SAIs, somente a componente de interesse pode ser utilizada no OF. Considerando que o OF é calculado a partir da SAI central nessa etapa, um total de 28 SAIs poderiam usar a informação de somente um componente dos vetores do OF. Com essa modificação, os dados de OF a serem armazenados cairia para um tamanho de 911,4MB.

Mesmo com essas simplificações, ainda há espaço para melhorias na etapa de Calcula *Optical Flow*. Como dito anteriormente na Seção 2.1.3, os *Light Fields lenslet* capturam os diferentes ângulos a partir de um *array* de micro-lentes igualmente espaçados. Considerando isso, pode-se assumir que o movimento dos objetos em cena irá variar de forma regular entre uma SAI e outra, quando as mesmas são verticalmente ou horizontalmente alinhadas, variando um pouco dependendo da profundidade do objeto na cena. Com isso em mente, a ideia básica da Predição no HCLE é calcular somente o OF das SAIs alinhadas verticalmente e horizontalmente à SAI central (considerando somente uma das componentes dos vetores), e gerar as SAIs intermediárias, ou seja, aquelas que não estão alinhadas à SAI central, a partir da interpolação das duas SAIs alinhadas verticalmente e horizontalmente à essa SAI a ser predita. Sendo assim, somente o OF de 28 SAIs, considerando somente a componente de interesse, seriam necessários, diminuindo a quantidade de dados de OF a serem transmitidos para 60,7MB. Porém, ainda há mais uma simplificação que pode ser feita considerando esse movimento regular e unidirecional citado entre as SAIs.

A Figura 16 mostra a última otimização que pode ser feita no cálculo do OF para a transmissão mínima de dados do *Light Field Codificado*. Como consideramos nessa etapa de predição o movimento regular dos objetos em cena entre diferentes ângulos do *Light Field*, somente o OF das SAIs marcadas em vermelho na Figura 16 em relação a SAI central (marcada em verde) pode ser calculado, o restante das SAIs podem ser preditas a partir da interpolação de todas as SAIs verticalmente e horizontalmente colocalizadas. Isso reduz a quantidade de dados de OF a serem enviados para 8,68MB, uma redução de 105 vezes em relação à previsão original de 911,4MB.

Porém, o OF é uma aproximação do movimento aparente entre duas imagens, isso significa que a mesma não fornece esse movimento em termos de *pixel*. Para contornar esse problema, é utilizada a etapa de Calcula *Phase Correlation*.

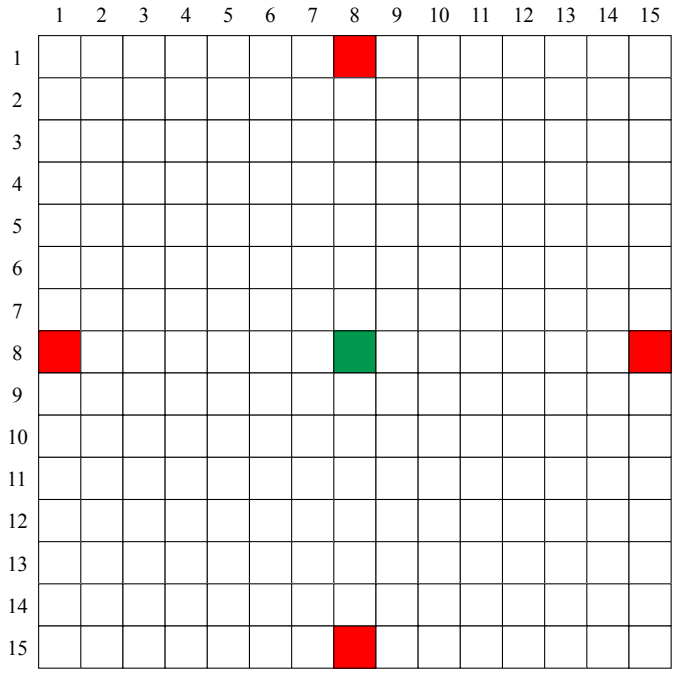

Figura 16 – SAIs utilizadas no cálculo do *Optical Flow* na etapa de Predição do HCLE

O PC, relembrando a Seção 2.5.2, mede o movimento translacional entre duas imagens. Diferentemente do OF, o PC fornece a disparidade de movimento em termos de *pixels*. Além disso, o cálculo do PC de todas as SAIs em relação a SAI central fornece mais evidências do movimento regular e unidirecional entre SAI, como mostra a Figura 11, onde para SAIs alinhadas horizontalmente em relação a SAI central não existe disparidade vertical e para SAIs alinhadas verticalmente não existe disparidade horizontal.

Sendo assim, os valores de OF que são normalizados com base nos valores de PC para estarem em termo de *pixels*. Para isso, são calculados os valores de PC para as SAIs alinhadas em relação a SAI central, como mostra a Figura 17.

Considerando que cada SAI possui um valor de PC calculado, considerando novamente os 8 *bytes*, os 28 PCs adicionariam um *overhead* de 224 *bytes* a serem transmitidos. Com essas informações, mais as informações explicadas anteriormente presentes na Figura 16, é possível fazer a predição de todas as SAIs, fazendo primeiro o deslocamento da SAI central para gerar as SAIs alinhadas a ela, e, após isso, a interpolação para gerar as SAIs intermediárias. Essa etapa será apresentada quando for explicada a etapa de decodificação que possui um passo específico para fazer esse processo e reconstruir todas as SAIs a partir das informações de predição geradas nesse passo de predição da etapa de codificação do HCLE.

3.1.1.2 Quantização

Mesmo com todas as otimizações para redução de dados a serem transmitidos feitas na etapa de Predição, ainda há mais otimizações que são feitas na etapa de

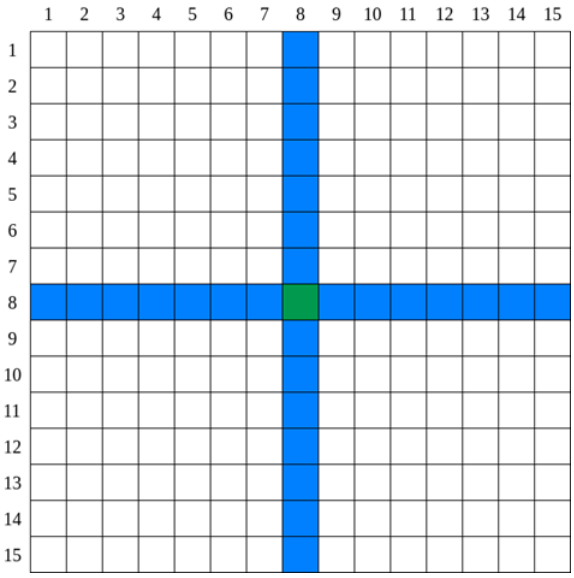

Figura 17 – SAIs utilizadas no cálculo do *Phase Coorrelation* na etapa de Predição do HCLE

quantização. A etapa de quantização é executada em cima dos valores de OF e serve para remover valores irrisórios ou desnecessários dos seus vetores.

Como dito anteriormente, o OF é calculado para cada *pixel* de cada SAI, gerando uma grande quantidade de dados. Além disso, retomando o que foi explicado em relação a Figura 15, a maioria dos vetores, principalmente em partes homogêneas, possui valores irrisórios a ponto de não serem possíveis de visualizar quando são representados sobre uma SAI. Isso significa que a maioria desses valores pode ser removida, já que não possuem informações relevantes para o processo de predição. Ou seja, a quantização remove valores que são considerados irrisórios, fazendo com que a quantidade de símbolos que precisam ser representados na codificação de entropia (ver Seção 2.2.1.4) seja drasticamente reduzido.

No caso da etapa que quantização do HCLE, é definido um limiar que definirá se o valor será eliminado ou não. A etapa de quantização recebe os valores de OF calculados e verifica todos os vetores para cada *pixel* da SAI, eliminando os valores que ficaram abaixo do valor modular do limiar.

A Figura 18 mostra um exemplo de quantização. No caso do exemplo, está re-

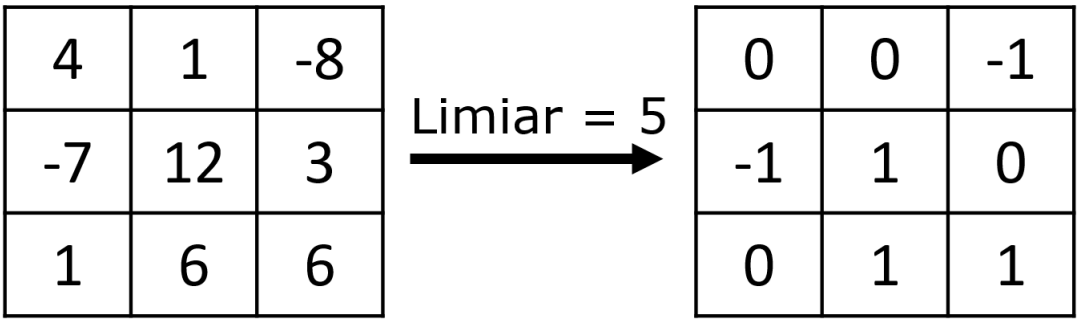

Figura 18 – Exemplo de quantização para o valor de limiar 5

presentada uma matriz de vetores 3$\times$3 que passará por um processo de quantização considerando um limiar de valor 5. O processo de quantização considera somente o módulo do vetor, ou seja, os valores negativos da imagem serão considerados somente em módulo no momento em que passarem pelo processo de quantização. Dessa forma, quando a matriz passar pelo processo de quantização, todos os vetores do OF que possuírem magnitude menor que o limiar serão transformados para 0. Ademais, considerando o deslocamento regular e unidirecional já explicado anteriormente (3.1.1.1), os vetores que possuem magnitude maior que o limiar são transformados em 1. Isso acontece, pois durante o processo de reconstrução das SAIs, durante a etapa de decodificação, utiliza o valor do PC para normalizar os vetores de OF, já que o movimento dos objetos em cena é regular.

Dessa forma, considerando os 8 *bytes* para cada vetor do OF, totalizando 2^{64} valores possíveis a serem representados, o número de símbolos a serem representados pelo codificador de entropia é reduzido para 3 (-1, 0 e 1). Assim, a codificação de entropia consegue gerar um *bitstream* de forma muito eficiente ao reduzir o número de *bits* para representar cada símbolo.

3.1.1.3 Codificador de Entropia

Como explicado na Seção 2.3, o codificador de entropia é responsável por gerar o *bitstream* agrupando todos os símbolos possíveis de maneira eficiente para tentar reduzir a quantidade de dados a serem armazenados.

A etapa de codificação de entropia do HCLE utiliza um algoritmo RLE (*Run-Length Encoding*) para a codificação binária das informações de OF, já que a mesma é uma codificação sem perdas muito eficiente quando há poucos símbolos e longas sequências de um mesmo símbolo. O algoritmo RLE foi escolhido por dois motivos:

1. Com as simplificações devido ao movimento uniforme e unidirecional dos *Light Fields lenslet*, foi possível representar os vetores do OF com apenas 3 símbolos, sendo necessários apenas 2 *bits* para representar os símbolos

2. Como citado e analisado anteriormente, regiões homogêneas de um *Light Field* vão possuir vetores do OF com valores irrisórios, que serão substituídos por 0 na etapa de quantização. Sendo assim, grandes porções das matrizes de OF possuírão 0 nessas regiões homogêneas, ou possuirão muito valores seguidos de 1 ou -1 dependendo da posição do ângulo codificado em relação a SAI central

Esses motivos fazem com que o algoritmo RLE seja muito eficiente para a codificação entrópica das matrizes de OF, reduzindo muito a quantidade de dados a serem transmitidos no *bitstream*. A organização de *bits* para os vetores de OF é mostrado na Figura 19.

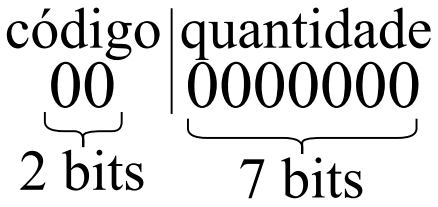

Figura 19 – Organização em *bits* do algoritmo RLE do codificador de entropia

Como somente 3 símbolos são possíveis para os vetores de OF após a etapa e quantização, o código que representa o símbolo no *bitstream* pode ser representado por apenas 2 *bits*, como mostrado na Figura 19. Após isso, os próximos 7 *bits* representam a quantidade de símbolos repetidos referentes ao código dos 2 *bits* anteriores.

A codificação de entropia completa o *bitstream* com a SAI central codificada com alguma outra técnica, como por exemplo o JPEG2000, e as informaçõe de PC sem nenhum tipo de compressão.

3.1.1.4 Codificação por Blocos

Uma outra otimização que pode ser feita no HCLE para reduzir o número de dados a ser enviado pelo *bitstream* é a codificação por blocos. Nesse caso, os valores das matrizes de OF são modificados para serem o mesmo para todos os *pixels* dentro de um bloco. Isso reduz o tamanho do *bitstream* pois haverá maior repetição de valores, fazendo com que o codificador de entropia consiga comprimir mais eficientemente.

O HCLE possui suporte para quatro tamanhos de bloco: 4×4, 8×8, 16×16 e 32×32. Para cada tamanho de bloco, podem ser utilizadas duas técnicas de definição do valor de vetor do bloco: média e mediana. No caso da média, o valor médio de OF dentro do bloco é utilizado em todos os *pixels*. No caso da mediana, o valor mediano é usado.

Quanto maior o tamanho do bloco, menos informação será enviada no *bitstream*, assim como a codificação será mais rápida, já que a predição, que será explicado mais a frente, será feita inteiramente para um bloco.

3.1.2 Etapa de Decodificação

A etapa de decodificação do HCLE é responsável por receber o *bitstream* e fazer a reconstrução do *Light Field*.

A Figura 20 mostra a visão geral da etapa de decodificação do HCLE. A entrada da etapa de decodificação é o *bitstream* gerado pela etapa de codificação. Os passos para a reconstrução do *Light Field* na etapa de decodificação do HCLE são:

1. Decodificador de entropia: é responsável por separar as informações do *bitstream* e aplicar a decodificação de entropia em cima das matrizes de OF

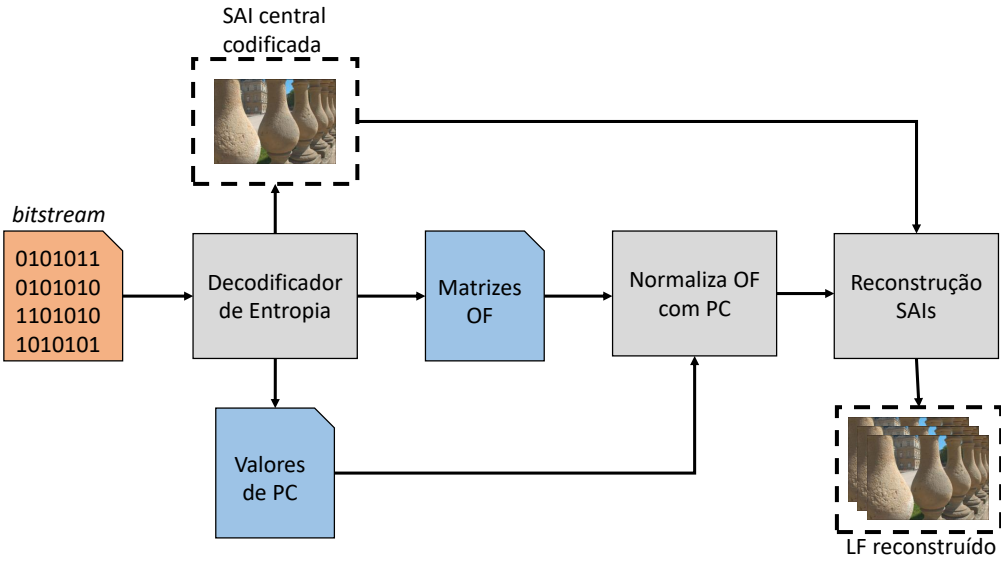

Figura 20 – Visão geral da etapa de decodificação do HCLE

2. Normaliza OF com PC: faz a normalização dos vetores de OF com base nos valores de PC correpondentes

3. Reconstrução SAIs: faz a reconstrução das SAIs com base na SAI central e as predições feitas na etapa de codificação com um algoritmo de interpolação para gerar as SAIs intermediárias

3.1.2.1 Decodificador de Entropia

A primeira ação do decodificador de entropia é separar os 3 tipos de dados que o *bitstream* carrega: SAI central codificada, matrizes de OF e valores de PC. Para as matrizes de OF, é necessário fazer a decodificação de entropia de acordo com o padrão da codificação feito na Seção 3.1.1.3. A decodificação de entropia é feita simplesmente lendo o código e colocando a quantidade indicada a seguir pelos próximos 7 *bits*, respeitando a resolução da SAI de 625×434 *pixels*.

Com esses 3 dados decodificados do *bitstream*, o decodificador continua para a próxima etapa que é a de normalização dos vetores de OF com o valor de PC.

3.1.2.2 Normaliza OF com PC

A etapa de normalização dos vetores de OF com os valores de PC transforma os valores quantizados pela etapa de codificação (-1, 0 e 1) no valor de deslocamento calculado na etapa de cálculo do PC. Essa normalização é feita em cima das 4 matrizes de OF de acordo com a Figura 16. Após isso, é gerado uma matriz de OF para cada uma das 28 SAIs restantes de acordo com a Figura 17. Um exemplo pode ser visto na Figura 21.

Na Figura 21 é considerada um exemplo com uma matriz de OF quantizada e com o valor do PC sendo 3. Os valores que antes eram -1 passam a ser -3 e os valores que

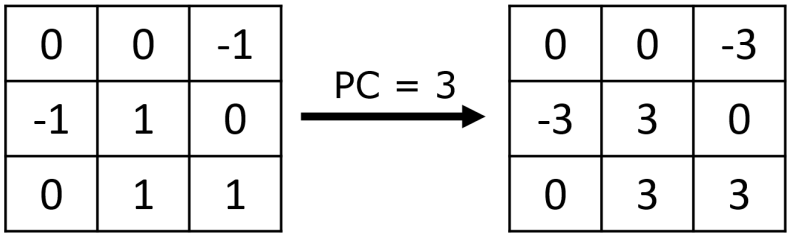

Figura 21 – Exemplo de normalização de uma matriz de OF com base no valor do PC

eram 1 passam a ser 3. Essa normalização serve para gerar o deslocamento em *pixel* necessário para gerar a SAI na próxima etapa do decodificador, seguindo a premissa de que o deslocamento dos objetos do *Light Field* é regular.

Esse processo é feito com a matriz de OF de um ângulo extremo, indicado em vermelho na Figura 16, gerando assim todas as outras matrizes de OF verticalmente ou horizontalmente alinhada à essa SAI extrema, de acordo com a normalização do PC referente a SAI a ser gerada e a matriz de OF.

3.1.2.3 Reconstrução das SAIs

A etapa de reconstrução das SAIs, como o próprio nome já diz, é responsável por gerar todas as SAIs com base nas informações que foram codificadas.

O primeiro passo dessa etapa é reconstruir as SAIs alinhadas horizontalmente e verticalmente com a SAI central. Parar isso, é feito um deslocamento dos *pixels* da SAI central de acordo com as 28 matrizes de OF geradas na etapa anterior. Com essas 28 SAIs reconstruídas, é possível fazer a reconstrução das SAIs intermediárias, aquelas que não estão alinhadas horizontalmente ou verticalmente em relação a SAI central. Para isso, é usado um algoritmo de interpolação que é mostrado na Figura 22 e é aplicado para *pixel* da SAI central de acordo com os vetores de OF da SAI a ser gerada.

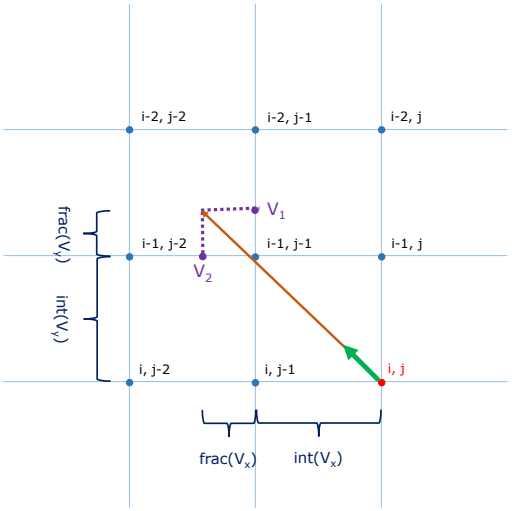

Figura 22 – Representação de interpolação para predizer as SAIs intermediárias

Para facilitar o entendimento dessa etapa, será usado como exemplo um caso de geração de SAI intermediária mostrado na Figura 23. Nesse exemplo, a SAI a ser predita é a SAI marcada em vermelho, que usará como referência um vetor gerado pelas componentes V_x e V_y das matrizes das SAIs indicadas pelas setas vermelhas.

Para um *pixel* (i, j) a ser predito, o vetor gerado pelas duas matrizes de OF é indicado pela seta verde na Figura 22, assim como esse mesmo vetor normalizado pelos valores de PC é indicado pela seta laranja. Como o vetor normalizado pode ser um valor de ponto flutuante, é necessário gerar duas amostras intermediárias de acordo com o valor do vetor. No exemplo citado, essas amostras intermediárias, indicadas por V_1 e V_2 na Figura 22 são geradas com base na Equação 5 e Equação 6, com base nos *pixels* $P(i-1, j-2)$, $P(i-1, j-1)$ e $P(i-2, j-1)$.

$$V_1 = (1 - frac(V_y)) \cdot P_{i-int(V_y),j-int(V_x)} + frac(V_y) \cdot P_{i-int(V_y)-1,j-int(V_x)} \tag{5}$$

$$V_2 = (1 - frac(V_x)) \cdot P_{i-int(V_y),j-int(V_x)} + frac(V_x) \cdot P_{i-int(V_y),j-int(V_x)-1} \tag{6}$$

Após as amostras intermediárias V_1 e V_2 serem geradas, o próximo passo do algoritmo de interpolação é gerar o *pixel* interpolado, na posição (i, j). Para isso, a equação 7 é utilizada. Os pesos dados para os valores de V_1 e V_2 são multiplicados por um α, que corresponde a um peso para a soma de $frac(V_y)$ e $frac(V_x)$ seja 1.

$$P_{i,j} = \alpha \cdot frac(V_y) \cdot V_2 + \alpha \cdot frac(V_x) \cdot V_1 \tag{7}$$

O valor de α é calculado de acordo com a Equação 8.

$$\alpha = \frac{1}{frac(V_x) + frac(V_y)} \tag{8}$$

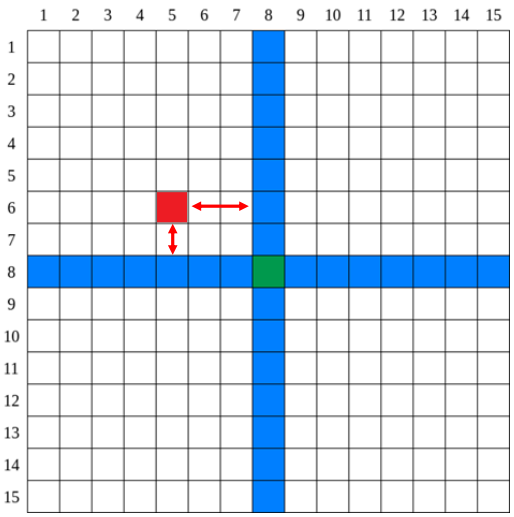

Figura 23 – Exemplo de geração de SAI intermediária. SAI marcada em vermelha a ser predita

O algoritmo de interpolação é feito para cada *pixel* das SAIs a serem preditas com base na SAI central, com tratamento especial para bordas superior e esquerda, onde os valores da SAI central é copiado. Quando o vetor ultrapassa os limites da SAI o valor do *pixel* colocalizado na SAI de referência é copiada na nova SAI.

Além disso, os vetores do OF podem ser utilizados a nível de bloco. Isso significa que o mesmo vetor é utilizado para um bloco, em vez de um *pixel*. A vantagem disso é que predizer um bloco em vez de um *pixel* diminui drasticamente a quantidade de vetores a serem enviados para o *bitstream*, além de diminuir o tempo total de reconstrução, já que o algoritmo de interpolação será aplicado inteiro para um bloco de uma vez, em vez de *pixel* a *pixel*.

3.2 Considerações Finais

Este capítulo apresentou o codificador HCLE proposto e o seu funcionamento detalhado. O codificador proposto é capaz de codificar um *Light Field lenslet* utilizando predição baseada em *Optical Flow* e aproveitando-se principalmente da premissa de que a movimentação dos objetos entre na cena de um LF é regular e unidirecional. Com isso, é possível reduzir drasticamente a quantidade de dados de OF e PC a serem enviados pelo *bitstream*, fazendo com que o codificador possa atingir altas taxas de compressão, que serão mostradas no capítulo de resultados.

Além de explorar características únicas dos *Light Fields lenslet*, o codificador proposto HCLE ainda possui etapas completas de codificação e decodificação, podendo ainda pode ser configurável para possuir diferentes tamanhos de bloco, diferentes tipos de seleção de vetor do bloco e também diferentes limiares para a etapa de quantização.

4 ESQUEMA PARA CODIFICAÇÃO DO ERRO DE PREDIÇÃO

Mesmo que o HCLE seja um codificador completo e consiga gerar todas as SAIs de um *Light Field*, ele não é capaz de compensar perdas causadas pela codificação. Para compensar o erro de predição, seria necessário adicionar uma etapa que reconstruísse cada bloco do LF com o erro da predição além do bloco predito. Com a adição de uma etapa de codificação do erro de predição, seria possível aumentar a qualidade geral da imagem, já que essa etapa compensaria os erros de predição com um acréscimo no *bitrate*.

Para facilitar essa etapa de codificação do erro de predição, foi utilizado o HEVC, que já possui de maneira intrínseca essa etapa ao fazer as etapas de predição. Porém, o HEVC possui diversos modos de predição diferentes, tanto intra-quadro como inter-quadros. Se fosse gerada uma pseudo-sequência com o *Light Field* gerado pelo HCLE e codificado com o HEVC, muitos modos de predição seriam adicionados à codificação, além da predição baseada em *Optical Flow* e *Phase Correlation*. Porém, como queremos avaliar somente a predição do HCLE sem esses modos extras do HEVC, o HEVC foi modificado para comportar o *Light Field* gerado pelo HCLE como o único modo de predição possível, sendo possível avaliar efetivamente a predição do HCLE. Uma visão geral do esquema necessário para gerar essa codificação de erro de predição é apresentado na Figura 24, onde etapas do HCLE e do HEVC são intercaladas.

A Figura 24 mostra o esquema para codificação de erro de predição com a utilização do HCLE e HEVC, onde os blocos da cor azul são etapas do HCLE e os blocos em verde são etapas do HEVC. A primeira etapa do esquema proposto é codificar o LF com o HCLE, o mesmo processo que já foi explicado na Seção 3. Como o HCLE precisa de SAI central codificada para a geração do *bitstream*, foi utilizado como método de compressão da SAI central o próprio HEVC utilizando apenas predição intra (quadro I). Com as SAIs sintetizadas, o *bitstream* gerado pelo HCLE e a SAI central codificada, é possível fazer a geração das pseudo-sequências de vídeo e utilizar a sinalização da predição inter-quadros do HEVC para sinalizar a predição HCLE.

4.1 Modificação no HEVC para Suportar a Predição HCLE

O HEVC possui diferentes modos de predição intra-quadro e inter-quadros. Para avaliar as SAIs sintetizadas pelo HCLE como preditor, optamos por modificar o HEVC para que a predição inter-quadros não utilize os métodos próprios, mas seja utilizada para sinalizar a predição os LFs sintetizados pelo HCLE.

Então, no esquema proposto, o LF de entrada original assume o papel de vídeo a ser codificado, enquanto o LF predito pelo HCLE assume o papel de vídeo de referência para ser usado durante a etapa de predição do HEVC. Para conseguir utilizar o LF codificado pelo HCLE, o *buffer* do HEVC que guarda os quadros codificados para serem usados como referência durante a predição inter-quadros, o DBP (*Decoded Picture Buffer*), foi modificado para armazenar o LF predito pelo HCLE.

Além disso, vários parâmetros do codificador HEVC precisaram ser modificados para garantir a predição de forma coerente com o HCLE. O primeiro deles, é permitir somente um quadro de referência durante a codificação inter-quadros. Outra modificação necessária é garantir que um bloco da SAI a ser codificado é comparado com o mesmo bloco na SAI de referência. Para isso, os vetores da estimação de movimento utilizados sempre possuem componentes x e y igual a zero. Dessa maneira, é garantido que a SAI a ser codificada utiliza a SAI colocalizada sintetizada pelo HCLE como referência para a etapa de estimação de movimento.

A Figura 25 mostra um exemplo de como o HEVC utiliza as pseudo-sequências de vídeo para fazer a codificação inter-quadros. A imagem mostra na parte superior o LF de entrada, sendo $P(i, j)$ a SAI central que não é necessária ser codificada na etapa de predição inter-quadros, já que a mesma é codificada externamente pelo próprio HEVC utilizando predição intra-quadro. Já para o restante das SAIs, o DBP armazena suas SAIs colocalizadas codificadas pelo HCLE. Para cada SAI, a predição inter-quadros busca do DBP a SAI correspondente e subtrai os blocos colocalizados

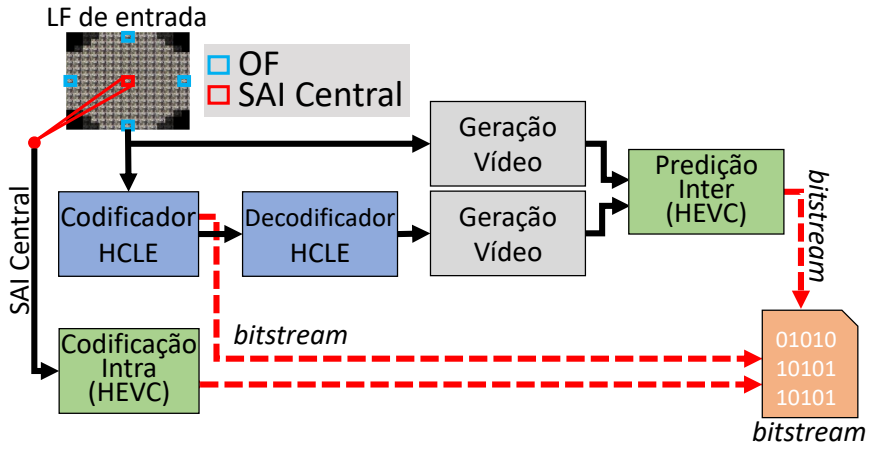

Figura 24 – Esquema para codificação de erro de predição com HCLE e HEVC

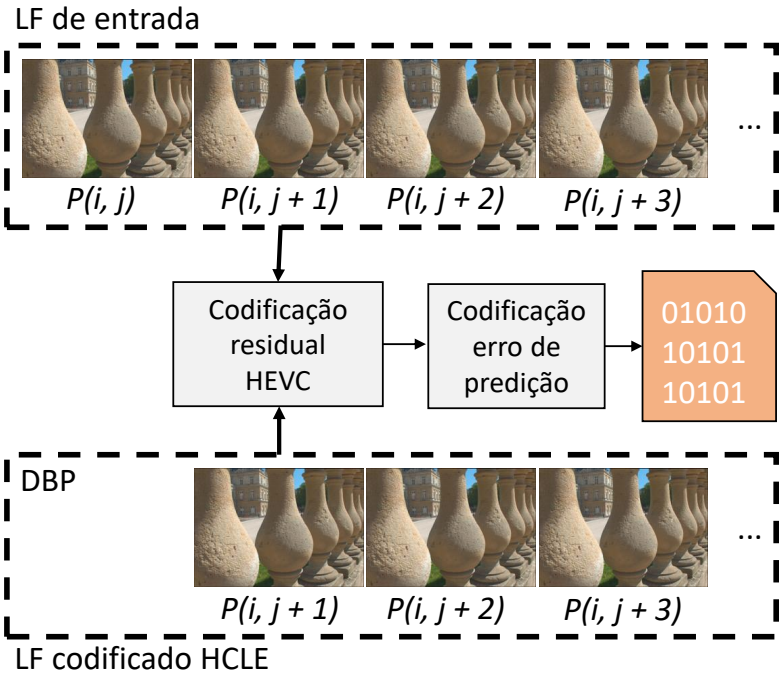

Figura 25 – Exemplo da codificação inter-quadros e como as referências são usadas com base no DBP

(vetores de movimento zerados), gerando o erro de predição de cada bloco e para cada SAI. Após isso, o codificador faz o resto da codificação, identificada pelo bloco Codificação erro de predição na Figura 25. Como o próprio codificador do HEVC faz a etapa de reconstrução, é possível gerar os resultados de qualidade durante a própria codificação.

Dessa forma, o codificador HEVC subtrai a SAI a ser codificada da SAI de referência, gerando o erro de predição que é codificado pelo HEVC. Mesmo possuindo *overhead* devido a sinalização da estimação de movimento e as SAIs de referência, como a informação é idêntica para todos os blocos e SAIs, as mesmas são eficientemente codificadas pelo codificador de entropia do HEVC. Além disso, esse esquema adiciona também o controle de qualidade pelo parâmetro de quantização (QP), que poderá ser ajustado para atingir maiores taxas de compressão, mas qualidades mais baixas, ou o contrário.

Com o esquema proposto para codificação de erro de predição utilizando o HCLE e o HEVC, é possível atingir melhoria na qualidade visual da imagem devido a compensação de erro de predição na etapa de reconstrução.

5 RESULTADOS EXPERIMENTAIS

Este capítulo mostrará todos os resultados experimentais obtidos com o codificador proposto HCLE e o esquema para codificação de erro de predição com HCLE e HEVC. O capítulo está separado de forma a primeiramente explicar a configuração em que os testes foram realizados, passando, após isso, pelos resultados e vantagens do codificador HCLE e os resultados da codificação de erro de predição.

5.1 Configuração dos Experimentos

Todos os resultados experimentais foram simulados em um computador rodando o *Ubuntu* 20.04 LTS e uma configuração com um processador *Intel Core i7-8700@3,20GHz* e 16GB de memória RAM. Os testes seguiram as CTCs (*Common Test Conditions*) formalizados pela iniciativa JPEG Pleno (JBIG; JPEG, 2019). Como dito anteriormente, o foco desse trabalho está nos *Light Fields lenslet*. Sendo assim, os *Light Fields* indicados para esse *dataset* nas CTCs do JPEG Pleno são: *Bikes, Danger_de_Mort, Fountain_Vincent2* e *Stone_Pillar_Outside*. A Figura 26 mostra a SAI central para cada um desses LFs. Estes *Light Fields* possuem SAIs com resolução de 625×434 *micro-imagens* com resolução 15×15 *pixels*.

Figura 26 – SAI central de todos os *Light Fields* utilizados nos teste de acordo com a CTCs do JPEG Pleno: (a)*Bikes* (b)*Danger_de_Mort* (c)*Fountain_Vincent2* (d)*Stone_Pillar_Outside*

As métricas de qualidades que devem ser avaliadas segundos as CTCs são: PSNR (*Peak Signal-to-Noise Ratio*) e SSIM (*Structural Similarity Index Measure*) (WANG et al., 2004). O SSIM considera elementos estruturais da imagem para tentar indicar a degradação de uma imagem em relação a outra. Além disso, as condições de teste também definem *bitrates* alvos, chamado de bpp (*bits* por *pixel*), para as comparações utilizarem uma mesma taxa de compressão e ser possível analisar a qualidade dos *Light Fields* para pontos fixos. Esses bpps alvos são: 0,001, 0,005, 0,02, 0,1 e 0,75.

Os resultados de qualidade dos *Light Field* foram comparados em termos de PSNR e SSIM, como definido nas CTCs. Além disso, também foi avaliada a eficácia da solução como preditor através da avaliação da energia do sinal do erro, e entropia do erro, onde o erro é o *Light Field* original subtraído do *Light Field* predito. Por último, foi avaliada a eficiência de codificação do HCLE, avaliando o tamanho do *bitstream* gerado e o tempo para codificar um *Light Field*. Para a comparação ser coerente, o MuLE foi ajustado para possuir os resultados o mais próximos possíveis em relação ao HCLE, então daqui para a frente, todas as comparações feitas com o MuLE estão ajustadas para o mesmo ponto de operação em relação HCLE, seja em termos de qualidade ou compressão. Esse ponto de operação foi atingido para a qualidade em PSNR o mais próximo possível do codificador HCLE, fazendo com que a *Hexadeca-tree* do MuLE sempre faça o particionamento até o nível 19 ($B_{min} = 19$) (ver (CARVALHO et al., 2018) para maiores detalhes).

5.2 Resultados do Codificador HCLE

Relembrando o que foi explicado na Seção 3.1.1.2, o codificador proposto HCLE possui uma etapa de quantização, responsável por eliminar vetores do *Optical Flow* para que a codificação de entropia atinja uma taxa de compressão maior. Além disso, a Seção 3.1.1.4 também explicou que a codificação pode ser feita a nível de blocos, ou seja, um vetor de OF pode ser selecionado para todas as amostras de um mesmo bloco, já que isso atinge maiores taxas de compressão e também se aproveita das similaridades dos vetores em uma mesma região, principalmente regiões homogêneas.

Com isso em mente, o HCLE foi testado a nível de *pixel* e a nível de bloco. Para o teste a nível de *pixel*, os limiares testados para quantização dos vetores do OF foram $0,1$, $0,01$, $0,001$ e $0,0001$. Já a nível de bloco, os limiares testados foram de 0 e $0,0001$. Além disso, os tamanhos de blocos utilizados foram de 4×4, 8×8, 16×16 e 32×32. Os métodos para escolha dos vetores do OF dentro de um bloco foram a média dos vetores e a mediana dos vetores. Todas as tabelas apresentadas nessa Seção são referentes à média em relação a todos os LFs.

Além disso, todas as comparações são feitas através do *software* de referência

do MuLE, que foi implementado em *C++*, enquanto o HCLE foi implementado em *Matlab*, já que a plataforma possui facilidade para lidar com operações relacionadas a matrizes. A SAI central foi codificada com o JPEG2000 e *compression rate* de 50.

5.2.1 Resultados de Qualidade da Predição

A primeira análise feita não é comparada ao MuLE, já que se refere à métricas de predição, e posteriormente o *Light Field* codificado pelo HCLE será utilizado dentro do HEVC como preditor. Para analisar a eficiência do LF codificado pelo HCLE como preditor, foram medidas algumas métricas de predição para o HCLE. As métricas testadas foram energia do erro e entropia do erro.

A energia do erro, mede a energia do sinal de erro, sendo que o erro é a diferença entre o LF original e o LF codificado. A fórmula da energia do erro pode ser vista na Equação 9.

$$E = \sum_{r=0}^{14}\sum_{s=0}^{14}\sum_{i=0}^{434}\sum_{i=0}^{625}(|LF(r,s,i,j) - LF'(r,s,i,j)|)^2 \tag{9}$$

Pela Equação 9, vemos que quanto menor o valor de energia, mais próximo do LF de referência (LF) o LF codificado (LF') está, já que erro é a diferença entre LF e LF'. Dessa forma, o erro que possuir a menor energia, também está mais próximo do LF original.

Como o LF original possui efeito de *vignetting* para SAIs das bordas, como explicado na Seção 2.1.4.2, essas SAIs foram excluídas da análise de métricas de predição. Sendo assim, a Equação 9, não considera todos os valores nos dois primeiros somatórios.

Para a entropia do erro a mesma ideia é utilizada. A Equação da entropia do erro é indicada abaixo:

$$H = \sum_{0}^{14}\sum_{0}^{14}\sum_{k} p_k \log(p_k) \tag{10}$$

O valor de p_k indicado na Equação 10 é o histograma normalizado de níveis de cada canal de cor do LF, ou seja, a probabilidade de cada nível do histograma acontecer dentro do LF. Quanto menor a entropia do LF, mais maior potencial de compressão existe no dado.

A Tabela 1 mostra os resultados do codificador proposto em termos de métricas de predição, energia do erro e entropia do erro, mostrando a média para todos os LFs. O melhor resultado do codificador HCLE acontece para a técnica de seleção do vetor do bloco *Média*, *Limiar* $0,0001$ e bloco 16$\times$16 com 1,66E+04. Já na entropia o codificador proposto atinge um mínimo de 2,7123. Pela tabela, é possível perceber que há diminuição da entropia para tamanhos de blocos maiores na técnica *Mediana*.

Isso provavelmente se deve ao fato de que utilizar muitos vetores de OF pode fazer com que a predição atingida seja pior, pois possui mais valores que adicionam erro à predição em relação ao LF original.

Tabela 1 – Resultados de métricas de predição para o HCLE

Técnica	Limiar	Bloco	Energia Erro	Entropia Erro
Média	0	4×4	1,72E+04	2,7474
		8×8	1,70E+04	2,7485
		16×16	1,66E+04	2,7446
		32×32	1,68E+04	2,7489
	0,0001	4×4	1,72E+04	2,7473
		8×8	1,70E+04	2,7479
		16×16	1,66E+04	2,7449
		32×32	1,68E+04	2,7496
Mediana	0	4×4	1,71E+04	2,7344
		8×8	1,74E+04	2,7432
		16×16	1,70E+04	2,7205
		32×32	1,70E+04	2,7123
	0,0001	4×4	1,71E+04	2,7347
		8×8	1,75E+04	2,7428
		16×16	1,71E+04	2,7174
		32×32	1,71E+04	2,7113
Pixel	0,0001	1×1	1,75E+04	2,7272
	0,001	1×1	1,77E+04	2,7277
	0,01	1×1	1,80E+04	2,7445
	0,1	1×1	1,99E+04	2,807

O capítulo 4 mostra um esquema de compensação de erro de predição, utilizando o *Light Field* codificado pelo HCLE como o único preditor e aproveitando-se das ferramentas de compensação já implementadas no HEVC. Dessa forma, será possível adicionar pontos de operação para o HCLE, assim como melhorar a qualidade objetiva, já que erros de predição serão compensados. Essa seção, mostrou que tamanhos de blocos maiores fornecem melhores preditores do HCLE, já que diminuem a entropia, eliminando vetores do OF que poderiam adicionar erros à predição, já que possivelmente eram valores irrisórios.

5.2.2 Resultados para Métricas de Qualidade Objetiva

A Tabela 2 mostra os resultados para todos os cenários testados no codificador proposto comparado ao codificador MuLE, e estão organizados da seguinte forma: a coluna de *Técnica* indica qual foi a técnica de escolha do vetor para cada tamanho de bloco, a coluna *Limiar* indica o valor de limiar testado na etapa de quantizacão e a coluna *Bloco* indica o tamanho dos blocos a serem codificados.

O melhor resultado alcançado em termos de PSNR foi para o OF a nível de bloco com técnica da *Mediana* e *Limiar* de $0,0001$. Nesse caso, o Y-PSNR foi de 25,306dB,

Tabela 2 – Resultados de qualidade para o HCLE vs MuLE

Técnica	Limiar	Bloco	Y-PSNR (dB)	Y-SSIM	YUV-PSNR (dB)
Média	0	4×4	25,0654	0,7821	26,9997
		8×8	24,8913	0,7751	26,8673
		16×16	24,6484	0,7644	26,6823
		32×32	24,4189	0,7516	26,5052
	0,0001	4×4	25,0642	0,782	26,9982
		8×8	24,8903	0,7751	26,867
		16×16	24,6537	0,7644	26,6882
		32×32	24,419	0,7514	26,5072
Mediana	0	4×4	25,298	0,7925	27,1823
		8×8	25,2795	0,7938	27,1788
		16×16	25,3024	0,7965	27,1961
		32×32	25,2462	0,7965	27,1537
	0,0001	4×4	25,2957	0,7923	27,1817
		8×8	25,2725	0,7935	27,1736
		16×16	25,3060	0,7965	27,1997
		32×32	25,224	0,7955	27,1379
Pixel	0,0001	1×1	25,1535	0,7882	27,0677
	0,001	1×1	25,1391	0,7863	27,0537
	0,01	1×1	24,8321	0,7659	26,7959
	0,1	1×1	23,4128	0,7192	25,6863
MuLE			24,5503	0,732	26,7023

0,7557dB maior do que o MuLE, e de 0,7995 de SSIM, onde também supera o MuLE sendo 0,0645 maior. Também é possível notar que o resultado da *Mediana* foi o melhor na média, atingindo média de 25,2815dB para *Limiar* de 0, enquanto a *Média* atinge 24,756dB. O OF a nível de *pixel* alcançou, na média, os piores resultados, com uma média de 24,6343dB. No geral, o melhor caso do codificador HCLE obtém um aumento de 0,7557dB para o Y-PSNR, 0,0645 para o SSIM (0,7965 do HCLE para *Mediana*, *Limiar* $0,0001$ e *Bloco* 16×16), e aumento de 0,4974dB para o YUV-PSNR.

Além disso, olhando a Tabela 2 é possível observar que em termos de qualidade o tamanho do bloco não possui impacto significativo, enquanto o *Limiar* possui alto impacto para o OF a nível de *pixel*, já que para o menor *Limiar*, o Y-PSNR foi de 25,1535dB e para o *Limiar* de $0,1$ o Y-PSNR alcançou 23,4128dB. No SSIM a diminuição foi de 0,7882 para 0,7192 do menor para o maior *Limiar*. Isso se deve, principalmente, pelo fato do *Limiar* acabar eliminando muitos valores que poderiam ser relevantes à nível de *pixel* para a etapa de predição, enquanto o tamanho de bloco acaba preservando, em certa quantidade, a relevância dos vetores de OF para o tamanho de bloco e eliminando menos valores importantes na etapa de quantização.

Mais para a frente, na Seção 5.2.2 será explicado e mostrado que o codificador proposto não carrega artefatos presentes no *Light Field* original, causados pela captação, que diminuem bastante a qualidade subjetiva da imagem. Isso faz com que

as métricas de qualidade objetivas, principalmente o PSNR, acusem uma quantidade de ruído na imagem codificada que não existe no *Light Field* original, já que o codificado pelo HCLE não possui esses artefatos. O SSIM, mesmo sofrendo esse mesmo impacto, acaba sendo menor do que o PSNR, já que o SSIM também leva em consideração características estruturais da imagem que o HCLE mantém de acordo com o LF original.

Em todas as métricas de qualidade objetiva testadas, o codificador proposto possuiu melhores resultados em pelo menos um caso comparado ao MuLE.

5.2.3 Resultados de Compressão e Tempo de Codificação

A Tabela 3 mostra os resultados do tamanho do *bitstream* e tempo de codificação para todos os cenários testados. É importante salientar que o tempo de codificação compreende a todo o processo até a geração das SAIs.

Tabela 3 – Resultados de *bitrate* e tempo de codificação para o HCLE vs MuLE

Técnica	Limiar	Bloco	Bitrate (B)	Tempo (s)
Média	0	4×4	154244	122,54
		8×8	95746	48,63
		16×16	55216	27,12
		32×32	41089	20,35
	0,0001	4×4	184414	120,15
		8×8	99407	46,22
		16×16	60467	26,74
		32×32	45185	20,95
Mediana	0	4×4	163040	93,76
		8×8	87510	39,58
		16×16	61735	25,74
		32×32	49563	21,51
	0,0001	4×4	195844	92,86
		8×8	100391	39,76
		16×16	64969	25,02
		32×32	59184	21,8
Pixel	0,0001	1×1	1353650	17,23
	0,001	1×1	1222779	16,88
	0,01	1×1	605643	16,73
	0,1	1×1	153447	16,48
MuLE			47643	187,37

A primeiro ponto a se observar na Tabela 3, é que quanto maior o tamanho do bloco utilizado, menor o *bitrate* e menor o tempo de codificação. A diminuição do *bitrate* acontece devido ao menor número de vetores a serem enviados para o *bitstream*. Em vez de enviar um vetor de OF para cada *pixel*, cada bloco terá somente um valor de um vetor para todos os *pixels*, fazendo com que o codificador de entropia possua mais casos repetidos de um mesmo símbolo seguido. Nesse caso, o maior tamanho

de *bitstream* foi de 1.353.650B para a técnica de escolha de vetor *pixel* e *Limiar* de 0,0001. O codificador proposto também consegue atingir taxas de *bitrate* menor do que o MuLE, considerando que o MuLE codifica 169 SAIs contra 225 do HCLE.

Já a diminuição de tempo acontece pois a predição é feita inteira para um bloco de uma só vez. Então, ao invés de fazer o cálculo para 625×434 *pixels*, o cálculo é feito para blocos que variam de 4×4 até 32×32, reduzindo drasticamente o número de cálculos a serem feitos e, consequentemente, o tempo total de codificação.

Também é importante observar que além do codificador proposto possuir *bitrate* competitivo em relação ao MuLE, para o mesmo ponto de qualidade, o mesmo gera todas as 225 SAIs, não sofrendo do problema de *vignetting* nas SAIs e nem mesmo SAIs completamente pretas, como pode ser observado na Figura 3. Sendo assim, o codificador proposto alcança um menor *bitrate* considerando que possui mais SAIs geradas que o MuLE, podendo ser até 35,22% mais eficiente na compressão de cada SAI do que o MuLE, considerando a compressão para todas as SAIs.

O codificador proposto supera em todos os cenários testados o codificador MuLE no aspecto tempo de codificação. Mesmo o pior caso do codificador proposto, onde o tempo de codificação é 122,54s, supera o MuLE, que precisou de 187,37s para fazer a codificação de um *Light Field*. Já no melhor caso do codificador proposto, o tempo de codificação foi de 20,35s, uma redução de 89,14% em relação ao codificador MuLE.

Na média, o codificador proposto utilizando a técnica de *Mediana* mostrou-se melhor em termos de qualidade, superando todas as outras técnicas e o MuLE em termos de PSNR e SSIM. Para as métricas de qualidade objetiva do LF, o HCLE atingiu no melhor caso um aumento de 0,7557dB para o PSNR e um aumento de 8,67% no SSIM. Já para as taxas de compressão e tempo de codificação, o HCLE atingiu uma melhoria de 13,75% na taxa de compressão gerando todas as 225 SAIs contra 169 do MuLE, uma melhoria de 35,22% na taxa de compressão de cada SAI, além de atingir redução de tempo de codificação de até 89,14%. Em relação ao LF original, que possui tamanho de 9.402.232.992 *bytes* quando capturado com a câmera *Lytro Illum*, o codificador proposto atinge taxa de compressão de até 99.9996%

5.2.4 Resultados de Qualidade Visual dos *Light Fields*

Como explicado na Seção 5.2.2, os *Light Fields lenslet* possuem artefatos na imagem que são inerentes da captação, mais proeminentes nas SAIs mais próximas às bordas. Esses artefatos são carregados pelo MuLE, mas pelo HCLE não são, já que todas as SAIs são preditas a partir da SAI central. Dessa forma, os resultados de métricas objetivas de imagem podem ser prejudicados, já que as métricas utilizam somente dados da imagem, onde esses artefatos podem causar diferenças em relação as SAIs originais. Além disso, para o mesmo ponto de compressão e métricas de qualidade objetiva, o HCLE possui qualidade visual superior ao MuLE. Esse capítulo

não tem a intenção de prover uma análise subjetiva completa dos *Light Fields*, mas sim demonstrar a diferença visual e os artefatos que são eliminados pelo HCLE em relação ao LF original e outras técnicas. A Figura 27 mostra um exemplo disso, comparando a SAI (12, 10) do LF *Stone_Pillars_Oustide* original, codificada pelo HCLE e codificada pelo MuLE.

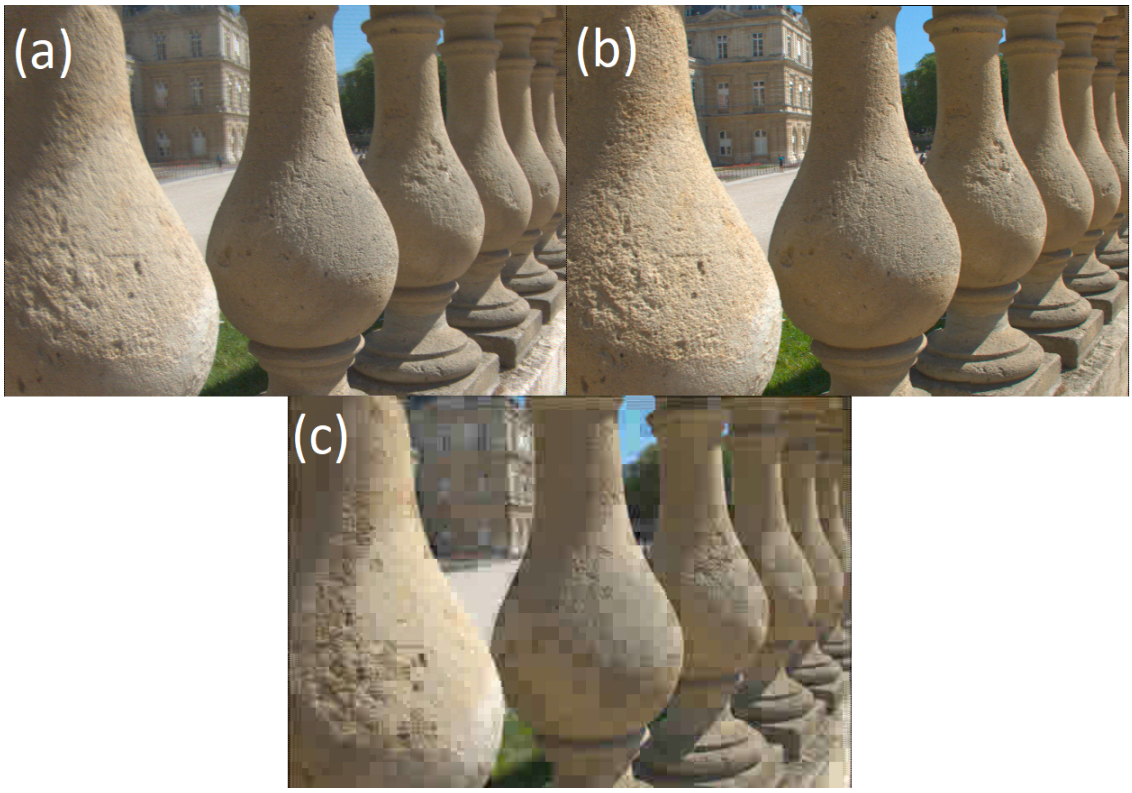

Figura 27 – SAI (12, 10) para o *Light Field Stone_Pillars_Oustide* (a) SAI original (b) SAI codificada pelo HCLE (c) SAI codificada pelo MuLE ($B_{min} = 19$)

Pela Figura 27(a), é possível notar os artefatos que o LF possui, principalmente na parte esquerda superior do LF, onde o prédio ao fundo possui efeito de borrão. Esse tipo de artefato está mais presente nas SAIs mais extremas do LF. A Figura 27(b) possui a mesma SAI codificada pelo HCLE. Nela, é possível ver que os artefatos não são carregados, já que a qualidade da imagem, no geral, é mais agradável, sem possuir os borrões do LF original. Isso acontece devido a predição do HCLE utilizar somente a SAI central como referência, e como essa não possui artefatos, os mesmos não aparecem nas SAIs codificadas pelo HCLE. A Figura 27(c) mostra a mesma SAI codificada pelo MuLE no mesmo ponto de operação de compressão que o HCLE.

É notável a diferença de qualidade a Figura 27(b) e a Figura 27(c), porém, os resultados de métricas objetivas, são muito próximos. Isso acontece devido aos artefatos que não são carregados pelo HCLE, causando diferenças de ruídos fazendo com que, principalmente o PSNR fique abaixo do esperado, enquanto o SSIM possui resultados mais elevados em relação ao MuLE, mas ainda abaixo do esperado já que a qualidade visual da SAI codificada pelo HCLE é alta. A falta da correção do erro de

predição também impacta nesses resultados, por isso o esquema explicado na Seção 4 foi implementado e os resultados serão avaliados mais a frente.

Uma possível avaliação para medir a qualidade dos *Light Fields* codificador pelo HCLE, seria fazer avaliações subjetivas, comparando os LFs gerados pelo HCLE com outras técnicas ou até mesmo contra o *Light Field* original. Porém, para simplificar, já que avaliações subjetivas para LFs ainda não são especificadas, devido a falta de tecnologia para visualização e até mesmo condições de teste, uma ideia é utilizar métricas de qualidade sem referência, onde são medidas perdas na naturalidade da imagem. Para adicionar mais a essa discussão sem necessitar de uma avaliação subjetiva, foram obtidos resultados de métrica de qualidade objetiva sem referência, para complementar os resultados objetivos, considerando mais a qualidade visual do *Light Field* do que simplesmente métricas objetivas que podem estar influenciadas pelos próprios erros das imagens de referência.

O método BRISQUE (Blind/Referenceless Image Spatial Quality Evaluator) (MITTAL; MOORTHY; BOVIK, 2012), foi utilizado para fazer a análise de qualidade visual sem referência. O método BRISQUE não utiliza parâmetros específicos de distorção para avaliar a qualidade. No lugar disso, utiliza estatísticas da cena para calcular possíveis perdas na naturalidade da imagem.

Tabela 4 – Resultados médios de BRISQUE para todos os *Light Fields*

Light Field	Original	HCLE	MuLE
Bikes	26,2627	28,1581	51,7322
Danger_de_Mort	20,0602	21,2776	55,2509
Fountain_Vincent2	26,0629	27,4274	56,0808
Stone_Pillars_Outside	18,2562	16,1511	48,5256

A Tabela 4 mostra os resultados da análise da qualidade visual dos LFs com o método BRISQUE e por ela é possível notar que visualmente o HCLE aproxima-se mais do que o MuLE para o mesmo ponto de operação, já que quanto menor o resultado do BRISQUE, maior qualidade visual da imagem. Para o HCLE, foi utilizado técnica *Media*, *Limiar* de $0,0001$ e bloco 16$\times$16. Para o melhor caso, o HCLE possui até mesmo uma melhor qualidade visual do que o LF original, já que corrige os artefatos presentes nas SAIs do LF. O HCLE apresenta um resultado melhor para todos os LFs em relação ao MuLE.

Essa análise complementa a análise da Seção 5.2.1, já que mostra que os *Light Fields* codificados pelo HCLE possuem um maior qualidade visual da imagem do que os LFs codificados pelo MuLE no mesmo ponto de operação. No melhor caso, para o *Light Field Stone_Pillars_Outside*, houve uma melhoria de 66,71% para a métrica BRISQUE em relação ao MuLE. No pior caso, a melhoria foi de 45,57%. Na média, o codificador HCLE melhora a qualidade visual da imagem em relação ao MuLE pela métrica BRISQUE em 52,22%, enquanto perde em relação ao LF original apenas 1,1%

na média. Porém, as métricas objetivas ainda podem ser melhoradas com o esquema apresentado no Capítulo 4 e os resultados desse esquema serão apresentados na próxima Seção.

5.3 Resultados para o esquema HCLE + HEVC

O esquema para compensação dos erros de predição foi apresentado na Seção 4. Nesse esquema desenvolvido, o LF codificado pelo HCLE é utilizado como predição no HEVC, e a etapa de reconstrução do HEVC fica responsável por compensar os erros de predição do HCLE. Isso é feito para que a qualidade geral da imagem melhore, gerando melhores resultados em termos de métricas objetivas, mais especificamente, o PSNR. Além disso, com esse esquema é possível atingir diferentes pontos de compressão do LF, consequentemente, diferentes qualidades.

As modificações explicadas na Seção 4 foram feitas no *software* de referência do HEVC, o HM (*HEVC Test Model*), na versão 16.20. Para atingir os bpps definidos pelas CTCs, o parâmetro QP foi modificado. As comparações foram feitas em relação ao MuLE para os mesmos bpps. Os bpps podem sofrer pequenas variações devido ao QP do HEVC, mas foram simulados para ficarem o mais próximos possíveis dos bpps alvos. O bpp alvo de $0,001$ foi excluído da análise pois o esquema proposto não foi capaz de atingir essa taxa de compressão. Para a codificação do HCLE, foi escolhida a técnica de *Mediana*, para tamanho de blocos 16$\times$16 e *Limiar* de $0,0001$, já que no geral, foram as configurações que apresentarem os melhores resultados. As pseudo-sequências de vídeo foram geradas com o *sofware ffmpeg* sem subamostragem.

Além disso, um esquema similar so apresentado em (ALVES; PEREIRA; SILVA, 2016) foi utilizado para evitar que as SAIs com artefatos na imagem fossem avaliadas. Para isso, somente as SAIs verticalmente e horizontalmente alinhadas a SAI central foram avaliadas.

A Figura 28 mostra os resultados para o *Light Field Bikes*, onde o eixo *y* mostra os

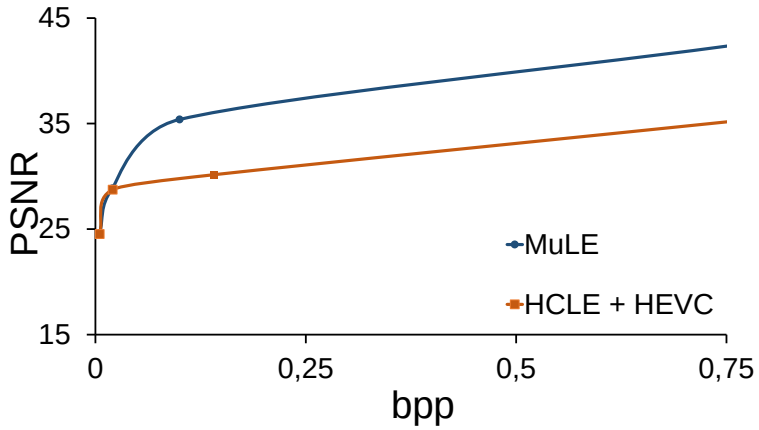

Figura 28 – Resultados de PSNR e bpp obtidos para o *Light Field Bikes*

resultados em termos de PNSR e o eixo *x* mostra os bpps, esse padrão é o mesmo para as próximas figuras. Para o LF *Bikes*, o esquema proposto obteve resultados similares para os bpps mais baixos de $0,005$ e $0,02$. Enquanto que para os outros bpps, fica abaixo do MuLE.

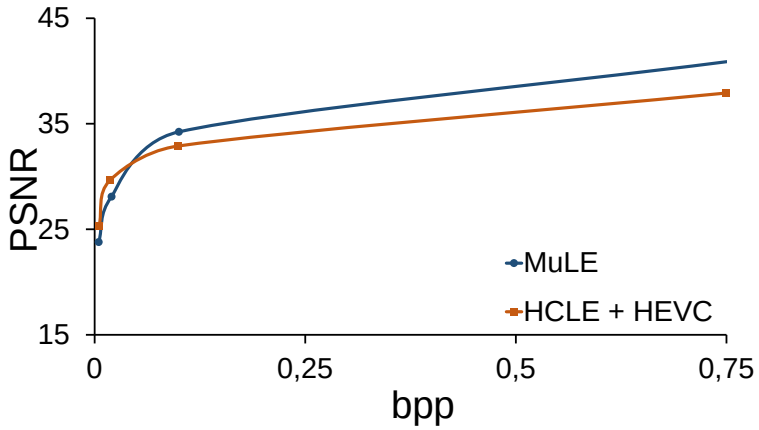

Figura 29 – Resultados de PSNR e bpp obtidos para o *Light Field Dange_de_Mort*

A Figura 29 mostra os resultados para o LF *Danger_de_Mort*, no caso desse LF, o esquema proposta supera o MuLE para bpps mais baixos, já que para os bpps de $0,005$ e $0,02$ os resultados de PSNR ficaram acima dos resultados obtidos com o MuLE. Isso também se reflete nos resultados da Tabela 4, onde o LF *Danger_de_Mort* possui resultado melhor que o *Bikes*.

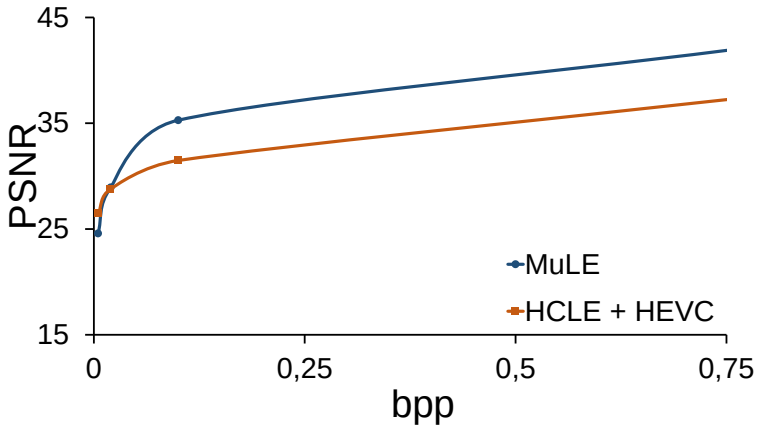

Figura 30 – Resultados de PSNR e bpp obtidos para o *Light Field Fountain_Vincent2*

Já para o LF *Fountain_Vincent2*, onde os resultados são mostrados na Figura 30, o esquema proposto se sai melhor para o menor bpp, já que o PSNR é maior com, virtualmente, nenhuma diferença no bpp. O bpp de $0,02$ atinge um PSNR quase igual ao MuLE, mas ainda sim um pouco abaixo. O restante do comportamento é o mesmo nos demais *Light Fields* testados.

Por fim, o LF *Stone_Pillars_Outside* é o que atinge os melhores resultados, já que para os menores bpps, o mesmo atinge um PSNR consideravelmente maior que o

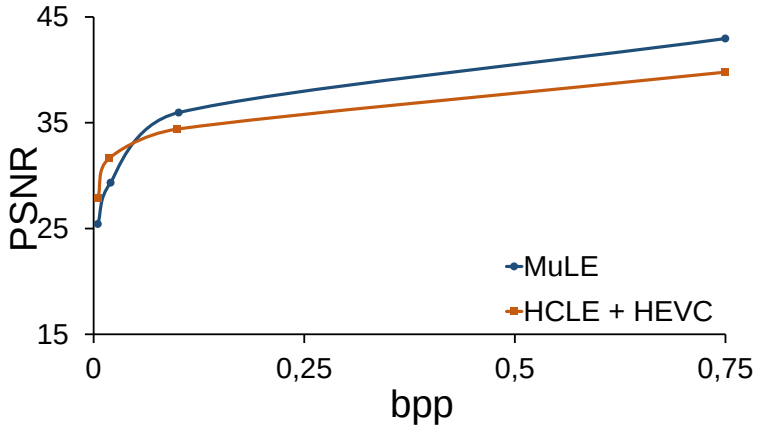

Figura 31 – Resultados de PSNR e bpp obtidos para o *Light Field Stone_Pillars_Outside*

MuLE, como pode ser visto pela Figura 31. Para o restante dos bpps, o comportamento se repete e o PSNR fica abaixo para os mesmos pontos de operação.

Tabela 5 – Resultados de PSNR médio para todos os bpps do MuLE vs HCLE + HEVC

bpp	MuLE	HCLE + HEVC
0,005	24,6082	26,5375
0,02	28,8018	29,3750
0,1	35,2154	32,5625
0,75	42,0201	38,0725

A Tabela 5, mostra o resultado médio considerando todos os LFs do MuLE contra o esquema HCLE + HEVC. Pela Tabela, percebe-se que para *bitrates* baixos, o esquema HCLE + HEVC possui maior qualidade do que o MuLE, já que para os bpps de $0,005$ e $0,01$ o MuLE atinge 24,61dB e 28,80dB, respectivamente, contra 26,54dB e 29,38dB. Para os bpps mais altos, de $0,1$ e $0,75$, o esquema proposto atinge PSNR menor do que o MuLE.

6 CONCLUSÃO

Os *Light Fields* são uma tecnologia extremamente promissora. Já que a gama de aplicações varia desde entretenimento até aplicações médicas. Além disso, os LFs possuem a vantagem de técnicas de processamento mesmo após serem capturados. O aumento de relevância dos LFs se evidencia ainda mais pois o mercado começa a investir cada ano mais nessa tecnologia, já que ela pode ser a próxima grande tecnologia imersiva. Porém, por capturar uma enorme quantidade de dados, ainda há muito a ser pesquisado no âmbito de compressão para que os LFs passem a ser uma opção viável no mercado de tecnologias imersivas.

Muitos trabalhos na literatura já propõem soluções eficientes e viáveis para o problema de compressão de *Light Fields*. Alguns utilizam ferramentas já conhecidas para codificação de imagens digitais, como codificadores de vídeo, transformando o LF em uma pseudo-sequência de vídeo para aproveitar as ferramentas que exploram redundâncias em vídeos dentro do contexto dos LFs. Outras exploram redundâncias específicas dos LFs, que diferem das características dos vídeos digitais, para tentarem melhorar a compressão dessa nova tecnologia. Mesmo assim, não são muitas as soluções que propõem o uso de predição para LFs, já que os mesmos possuem muita redundância de dados, dada a natureza da captação, que é ângulos com deslocamento muito próximos um dos outros.

Dessa forma, esse trabalho propôs um codificador de *Light Fields* com Predição Baseada em *Optical Flow* e *Phase Correlation* para Altas Taxas de Compressão, nomeado de HCLE. O HCLE é um codificador completo que explora as redundâncias específicas de dados dentro de um LF utilizando, principalmente, as técnicas de *Optical Flow* e *Phase Correlation*. O codificador se mostrou eficiente, já que pode atingir taxas de até 99,9996% de compressão do *Light Field* se comparado ao original capturado pela câmera *Lytro Illum*. Além disso, se comparado a trabalho relacionado, o HCLE consegue atingir aumento no PSNR de até 0,7557dB para o mesmo ponto de operação, assim como melhorar a taxa de compressão em 35,22% na média para cada SAI de um LF possuindo tempo de codificação de até 89,14% menor. Não obstante, o HCLE também melhora a qualidade visual do LF eliminando artefatos e o efeito

de *vignetting* no LF que são inerentes de sua captura, atingindo 56,21% de melhora na qualidade visual em termos da métrica BRISQUE, assim como também gera mais SAIs do que o LF original. Também foi apresentado um esquema para compensação do erro de predição, utilizando o HCLE e o HEVC, onde para as taxas de compressão mais altas, o esquema se mostrou mais eficiente do que o trabalho relacionado comparado.

O codificador pode ser utilizado juntamente de outras técnicas e também ainda há explorações que podem ser feitas dentro do HCLE, como agregar diferentes tipos de predição ou melhorar o reaproveitamento dos dados de OF e PC. Porém, mesmo com melhorias a serem exploradas, o codificador HCLE se mostra uma opção viável para codificação de *Light Fields* onde é necessária uma alta de taxa de compressão ou alta velocidade de codificação.

REFERÊNCIAS

ADELSON, E.; BERGEN, J. The Plenoptic Function and the Elements of Early Vision. **Computational models of visual processing**, [S.l.], p.3–20, 08 1997.

AGOSTINI, L. V. **Desenvolvimento de Arquiteturas de Alto Desempenho Dedicadas à Compressão de Vídeo Segundo o Padrão H.264/AVC**. 2007. 172p. Tese (Doutorado) — Instituto de Informática, Univesidade Federal do Rio Grande do Sul, Porto Alegre.

ALVES, G.; PEREIRA, F.; SILVA, E. A. da. Light field imaging coding: Performance assessment methodology and standards benchmarking. In: IEEE INTERNATIONAL CONFERENCE ON MULTIMEDIA EXPO WORKSHOPS (ICMEW), 2016., 2016. **Anais...** [S.l.: s.n.], 2016. p.1–6.

ARGYRIOU, V.; RINCÓN, J. M. D.; VILLARINI, B.; ROCHE, A. **Image, Video & 3D Data Registration**. [S.l.]: John Wiley & Sons, Ltd, 2015.

BRINKMANN, R. **The Art and Science of Digital Compositing**. [S.l.]: Morgan Kaufmann, 2008. 684p.

CARVALHO, M. B. de et al. A 4D DCT-Based Lenslet Light Field Codec. In: IEEE INTERNATIONAL CONFERENCE ON IMAGE PROCESSING (ICIP), 2018., 2018. **Anais...** [S.l.: s.n.], 2018. p.435–439.

CASS, S. **3-D TV is Officially Dead (For Now) and This is Why it Failed**. Disponível em: <https://spectrum.ieee.org/tech-talk/consumer-electronics/audiovideo/3d-tv-is-officially-dead-for-now-and-this-is-why-it-failed>. Acesso em: 2021-07-03.

CASS, S. **Samsung And Philips Sign 3D's Death Warrant**. Disponível em: <https://www.forbes.com/sites/johnarcher/2016/03/04/samsung-and-philips-sign-3ds-death-warrant/?sh=37e5a34361c3>. Acesso em: 2021-07-03.

CONCEIçãO, R.; PORTO, M.; ZATT, B.; AGOSTINI, L. LF-CAE: Context-Adaptive Encoding for Lenslet Light Fields Using HEVC. In: IEEE INTERNATIONAL CONFE-

RENCE ON IMAGE PROCESSING (ICIP), 2018., 2018. **Anais...** [S.l.: s.n.], 2018. p.3174–3178.

DAGUERRE, L. J. M. **An Historical and Descriptive Account of the Various Processes of the Daguerréotype and the Diorama**. [S.l.]: American Photographic Historical Society, 1839. 86p.

DE OLIVEIRA ALVES, G. et al. The JPEG Pleno Light Field Coding Standard 4D-Transform Mode: How to Design an Efficient 4D-Native Codec. **IEEE Access**, [S.l.], v.8, p.170807–170829, 2020.

HARTMANN, C.; WANG, J.; OPRISTESCU, D.; VOLK, W. Implementation and evaluation of optical flow methods for two-dimensional deformation measurement in comparison to digital image correlation. **Optics and Lasers in Engineering**, [S.l.], v.107, p.127–141, Aug. 2018.

HOFFMAN, D. M.; GIRSHICK, A. R.; AKELEY, K.; BANKS, M. S. Vergence–accommodation conflicts hinder visual performance and cause visual fatigue. **Journal of Vision**, [S.l.], v.8, n.3, p.33, Mar. 2008.

HOFFMANN, D. L. et al. U-Th dating of carbonate crusts reveals Neandertal origin of Iberian cave art. **Science**, [S.l.], v.359, n.6378, p.912–915, Feb. 2018.

HORN, B. K.; SCHUNCK, B. G. Determining optical flow. **Artificial Intelligence**, [S.l.], v.17, n.1-3, p.185–203, Aug. 1981.

IHRKE, I.; RESTREPO, J.; MIGNARD-DEBISE, L. Principles of Light Field Imaging: Briefly revisiting 25 years of research. **IEEE Signal Processing Magazine**, [S.l.], v.33, n.5, p.59–69, Sept. 2016.

JBIG; JPEG. **JPEG PLENO LIGHT FIELD CODING COMMON TEST CONDITIONS V3.3**. N18277.ed. [S.l.: s.n.], 2019.

JCT-VC. **Text of ISO/IEC FDIS 23008-2 (4th edition)**. N18277.ed. [S.l.: s.n.], 2019.

KRISHNAMURTHY, R.; WOODS, J.; MOULIN, P. Frame interpolation and bidirectional prediction of video using compactly encoded optical-flow fields and label fields. **IEEE Transactions on Circuits and Systems for Video Technology**, [S.l.], v.9, n.5, p.713–726, 1999.

KUGLIN, C. D.; HINNE, D. C. The phase correlation image alignment method. In: INTERNATIONAL CONFERENCE ON CYBERNETICS AND SOCIETY, 1975. **Proceedings...** [S.l.: s.n.], 1975. p.163–165.

LESWING, K. **3D TV is dead - Business Insider**. Disponível em: <https://www.businessinsider.com/3d-tv-is-dead-2017-1>. Acesso em: 2021-07-03.

LEVOY, M. et al. Light field microscopy. In: ACM SIGGRAPH 2006 PAPERS ON - SIGGRAPH '06, 2006. **Anais...** ACM Press, 2006.

LYTRO. **Lytro**. Disponível em: <https://www.lytro.com/>. Acesso em: 2018-07-03.

MITTAL, A.; MOORTHY, A. K.; BOVIK, A. C. No-Reference Image Quality Assessment in the Spatial Domain. **IEEE Transactions on Image Processing**, [S.l.], v.21, n.12, p.4695–4708, 2012.

MONTEIRO, R. J. S.; NUNES, P. J. L.; RODRIGUES, N. M. M.; FARIA, S. M. M. Light Field Image Coding Using High-Order Intrablock Prediction. **IEEE Journal of Selected Topics in Signal Processing**, [S.l.], v.11, n.7, p.1120–1131, 2017.

NG, R. et al. Light Field Photography with a Hand-Held Plenopic Camera. **Technical Report CTSR 2005-02**, [S.l.], v.CTSR, 01 2005.

RAYTRIX. **RayTrix. 3D light field camera technology**. Disponível em: <http://www.raytrix.de/>. Acesso em: 2021-07-03.

SANDVINE. **2020 Global Internet Phenomena Report**. Disponível em: <https://www.sandvine.com/phenomena>. Acesso em: 2020-31-01.

STATISTA. **Immersive technology consumer market revenue worldwide from 2018 to 2023, by segment**. Disponível em: <https://www.statista.com/statistics/936078/worldwide-consumer-immersive-technology-market-revenue/>. Acesso em: 2021-07-03.

SULLIVAN, G. J.; OHM, J.-R.; 0001, W. H.; WIEGAND, T. Overview of the High Efficiency Video Coding (HEVC) Standard. **IEEE Trans. Circuits Syst. Video Techn**, [S.l.], v.22, n.12, p.1649–1668, 2012.

WALKER, J.; GUPTA, A.; HEBERT, M. Dense Optical Flow Prediction from a Static Image. In: IEEE INTERNATIONAL CONFERENCE ON COMPUTER VISION (ICCV), 2015., 2015. **Anais...** [S.l.: s.n.], 2015. p.2443–2451.

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

WEI, H.; YIN, X.; LIN, P. Novel Video Prediction for Large-scale Scene using Optical Flow. **CoRR**, [S.l.], v.abs/1805.12243, 2018.

WILBURN, B. et al. High performance imaging using large camera arrays. **ACM Transactions on Graphics**, [S.l.], v.24, n.3, p.765–776, July 2005.

ZHAO, S.; CHEN, Z.; YANG, K.; HUANG, H. Light field image coding with hybrid scan order. In: VISUAL COMMUNICATIONS AND IMAGE PROCESSING (VCIP), 2016., 2016. **Anais...** IEEE, 2016.