Universidade Federal de Santa Catarina
Campus Reitor João David Ferreira Lima
Departamento de Engenharia Elétrica e Eletrônica

Bruna Suemi Nagai

Hardware dedicado para transformação Affine em codificação de vídeo

Florianópolis
2022

Bruna Suemi Nagai

Hardware dedicado para transformação Affine em codificação de vídeo

Ante-projeto de Trabalho de Conclusão de Curso apresentado à Universidade Federal de Santa Catarina como parte dos requisitos necessários para a obtenção do título de Bacharel em Engenharia Eletrônica.
Orientador: Prof. Dr. Mateus Grellert

Universidade Federal de Santa Catarina
Campus Reitor João David Ferreira Lima
Departamento de Engenharia Elétrica e Eletrônica

Florianópolis
2022

Bruna Suemi Nagai

Hardware dedicado para transformação Affine em codificação de vídeo

Ante-projeto de Trabalho de Conclusão de Curso apresentado à Universidade Federal de Santa Catarina como requisito parcial para a obtenção do título de Bacharel em Engenharia Eletrônica.

Prof. Dr. Mateus Grellert
Universidade Federal de Santa Catarina
Orientador

Florianópolis, 13 de março de 2023

Resumo

A codificação de vídeo faz parte do cotidiano de muitas pessoas, pois sempre que acessamos uma rede social ou fazemos uma vídeo chamada, foi necessário que houvesse a codificação e decodificação do vídeo para que o usuário final conseguisse assistí-lo. Essa tecnologia está em constante evolução e cada vez mais temos algoritmos sofisticados que aumentam a complexidade computacional, fazendo com que a demanda por métodos mais eficientes aumente. Assim, este trabalho tem como objetivo realizar uma revisão bibliográfica e investigar otimizações implementáveis com *hardware* dedicado, focando na etapa da transformação *affine* em codificação de vídeo, que pertence a uma das etapas mais custosas do processo. Após a implementação, serão estimados os resultados de área, potência e frequência de operação da arquitetura proposta.

Palavras-Chave: 1. Codificação de vídeo. 2. VLSI. 3. Hardware. 4. Affine. 5. Compensação de Movimento

Lista de figuras

Figura 1 – Acessos realizados ao *Facebook, Netflix* e *YouTube*, pré e pós a pandemia do Covid-19. 6
Figura 2 – Esquemático do funcionamento básico do codificador VVC. 9
Figura 3 – Diferença entre (a) AMC e (b) TMC. 10
Figura 4 – Arquitetura proposta para a DAMC-Net. 12
Figura 5 – Fluxo estimado de trabalho. 15

Sumário

1 INTRODUÇÃO 6
1.1 Objetivos e motivação 7

2 REFERENCIAL TEÓRICO 8
2.1 Codificação de vídeo 8
2.2 Transformação Affine 8
2.2.1 Geração dos vetores de movimento 10
2.2.2 Interpolação 11

3 REVISÃO DO ESTADO DA ARTE 12
3.1 *Deep Affine Motion Compensation Network for Inter Prediction in VVC* 12
3.2 *An Efficient Four-Parameter Affine Motion Model for Video Coding* 13
3.3 *Fast Affine Motion Estimation for Versatile Video Coding (VVC) Encoding* 13
3.4 Análise dos artigos estudados 14

4 METODOLOGIA 15

5 CRONOGRAMA 16

6 RECURSOS NECESSÁRIOS 17

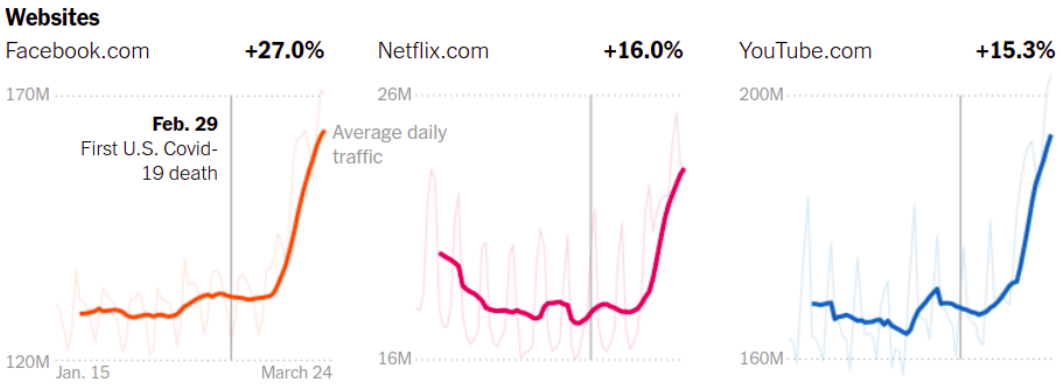
REFERÊNCIAS BIBLIOGRÁFICAS 18

1 Introdução

Nos últimos anos, acompanhamos um grande aumento no uso de plataformas digitais, como redes sociais, meios de comunicação e serviços de *streamings*. Segundo (STATISTA, 2022), o Facebook contou com mais de 1,9 bilhões de usuários ativos por mês no início de 2022. A Netflix, uma das grandes empresas no ramo de *streaming*, possui sozinha, mais de 220 milhões de assinantes (STOLL, 2022) e o YouTube chega a processar cerca de 500 horas de vídeos a cada minuto (CECI, 2022). Diante desses dados, podemos ter uma percepção do quanto a aquisição e reprodução de vídeos são processos relevantes e recorrentes no cotidiano das pessoas no mundo todo. A Cisco, em seu relatório anual publicado em 2018, previa que em 2022 teríamos cerca de 4,8 bilhões de usuários de *internet*, dos quais 82% de tal tráfego global correponderia a vídeos (CISCO, 2020).

A pandemia causada pelo Covid-19 em 2020 intensificou o aumento na quantidade de acessos à *websites* com conteúdos em vídeo, como mostra no gráfico da Figura 1, onde foram analisados três grandes empresas de tecnologia: Facebook, Netflix e YouTube. Nota-se que tanto a geração quanto o consumo de mídias digitais é recorrente e faz parte do cotidiano das pessoas, visto o grande número de tráfego nos *websites* analisados, onde são postados e reproduzidos vídeos a todo momento. Portanto cada vez mais é esperado que se tenham avanços tecnológicos disponíveis no mercado que atendam à qualidade, sem muito consumo de energia, principalmente quando se tratam de *smartphones*.

Figura 1 – Acessos realizados ao *Facebook, Netflix* e *YouTube*, pré e pós a pandemia do Covid-19.



Fonte: 1.

Os codificadores e decodificadores (codecs) de vídeo são algoritmos responsáveis pelo processamento dos dados, ou seja, fazem a compressão do vídeo para que este seja transmissível, armazenável e reproduzível com os recursos disponíveis. Para tal fim, os codecs utilizam técnicas específicas como estimação e compensação de movimento, aplicação do

cálculo de transformadas, codificação entrópica, quantização, implementação de filtros e por fim, o arquivo comprimido resultante é o *bitstream*.

As tecnologias emergentes com formatos de vídeos em altas resoluções maiores do que o convencional *high definition* (HD), como de 4K, 8K e até 16K (15360 pixels $\times$ 8640 pixels) e dispositivos de realidade augmentada, requerem o processamento de uma quantidade enorme de dados em tempo real, que novamente enfatiza a demanda pela aceleração em *hardware* para o processamento atingir taxas de compressão relativamente maiores.

Tendo em mente que compressão deveria ser mais eficaz, foi desenvolvido um novo padrão de codificador, finalizado em julho de 2020 e chamado de *Versatile Video Codec* (VVC). Esse padrão é fruto dos esforços conjunto do *Video Coding Experts Group* (VCEG) e *Moving Picture Experts Group* (MPEG). O VVC obtém uma taxa de bits 50% menor do que seu predecessor, o *High Efficiency Video Coding* (HEVC), a uma mesma qualidade de vídeo (BROSS et al., 2021). Além disso, suporta compressões para resoluções de até 16K, fazendo jus ao seu nome "versátil" já que o objetivo era conseguir englobar aplicações tecnológicas mais versáteis e emergentes (BROSS et al., 2021).

1.1 Objetivos e motivação

Diante do que foi exposto, observa-se que existe uma demanda por tecnologias de aquisição e reprodução de vídeos com taxa de compressão suficiente para possibilitar o armazenamento e transmissão dos dados. Complementar a isso, foi detalhado que os padrões de codificadores modernos, como o VVC, possuem diversos recursos adicionais para propiciar compressões mais inteligentes, porém a um custo de aumento na complexidade computacional. Um caso relevante é a transformação *affine* que realiza múltiplas multicações para cada quadro de vídeo analisado. Sendo assim, se torna evidente a motivação do trabalho já que um uma implementação *Very Large Scale Integration* (VLSI) dedicado fazendo as operações com maior eficiência.

O projeto de Trabalho de Conclusão de Curso terá como objetivo o desenvolvimento de um bloco em VLSI dedicado para os cálculos da transformação *affine*, considerando a arquitetura de quatro e seis parâmetros. Além disso, será implementados a interpolação a partir dos vetores de movimentos gerados pela transformação e pelo bloco do quadro de referência. Desse modo, também será necessário elaborar o *buffer* para armazenar os valores intermediários da interpolação, no caso de duas passadas, horizontal e vertical. Serão realizadas as verificações funcionais da arquitetura através de simulações e posteriormente será realizada a síntese lógica para verificar a frequência de operação, potência e área estimada.

2 Referencial Teórico

2.1 Codificação de vídeo

Após a captura do vídeo por câmeras digitais, de *smartphones* por exemplo, os codificadores são encarregados pela compressão desses dados. Durante essa etapa, são implementados algoritmos e utilizados componentes de *hardware* dedicado para tal processamento e assim, possibilitar sua transmissão ou armazenamento eficiente. Já o decodificador é o incumbido pela conversão reversa do *bitstream* para viabilizar a reprodução do vídeo em um *smartphone*, televisão ou computador. Portanto, percebe-se que tanto a codificação quanto a decodificação devem seguir o mesmo padrão. A Figura 2 detalha as etapas típicas da codificação de um vídeo utilizando o padrão VVC.

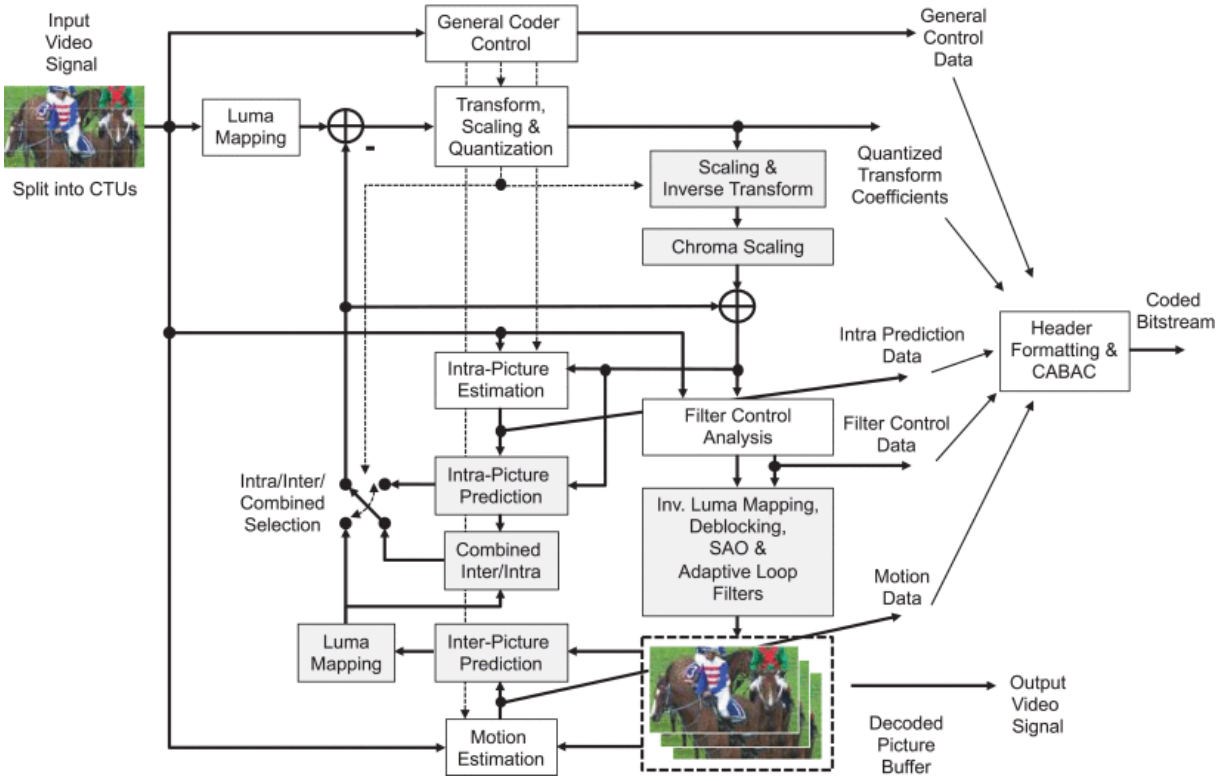
Pode-se resumir a descrição do processo a partir da codificação convencional, onde há métodos examinando cada um dos quadros e sub-regiões dentro deles (blocos) em busca de informações de pouca relevância para o sistema visual humano. O início da codificação se inicia com o sinal de entrada sendo o próprio vídeo, que é separado em quadros no tempo e posteriormente ainda são divididos em blocos menores com poucos pixels, com características similares. Cada bloco, quadrado ou retangular, é representado por alguns pixels, que têm informações como luminância e cor. Então, tais blocos passam pelas etapas de exploração espacial (predição intra quadros) e exploração temporal (predição inter quadros). O resíduo, gerado a partir da subtração de algum bloco anteriormente predito e o bloco atual, passa pelas etapas de cálculo de transformada e quantização, e sua saída avança para o codificador entrópico. Assim, o *bitstream* é gerado, podendo ser transmitido ou armazenado com maior facilidade.

Esse resíduo quantizado passa pela quantização reversa e transformada reversa para obter o resíduo reconstruído, que é somado às amostras anteriormente preditas para gerar as amostras reconstruídas. Em seguida, o resultado ainda passa por um *deblocking filter* (DBF) e por um *sample adaptive offset* (SAO) com o intuito de suavizar os artefatos adicionados pela quantização e pelo fato de o processamento ser realizado em blocos (CORREA et al., 2016). Por fim, a amostra reconstruída pode ser armazenada em um *buffer* que servirá de base para as próximas predições, como quadros de referência.

2.2 Transformação Affine

Durante a codificação do quadro em análise, é feita a predição inter quadros, na qual há a etapa de Estimação de Movimento (ME), uma das maiores consumidoras de

Figura 2 – Esquemático do funcionamento básico do codificador VVC.

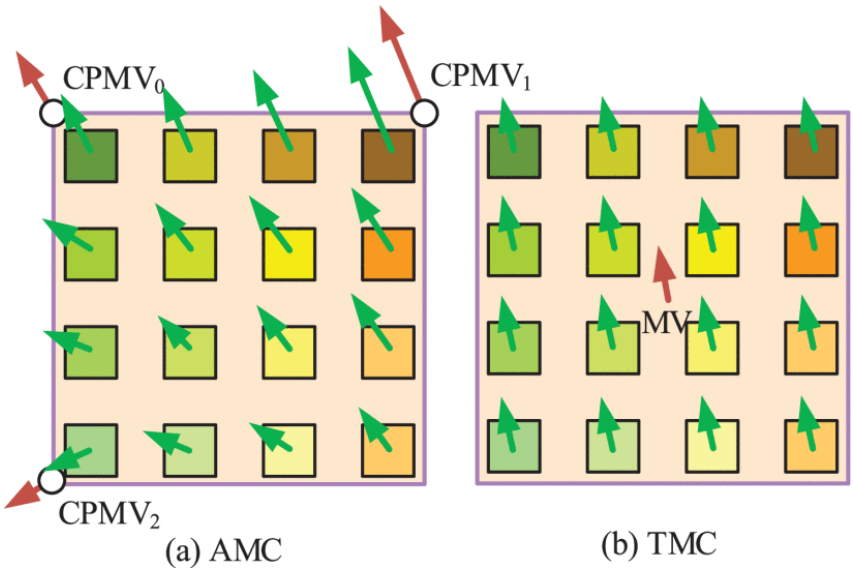


Fonte: (BROSS et al., 2021).

tempo, podendo chegar a até 80% do tempo total de codificação no modo *full-search* (ZHANG; ZHANG; FAN, 2018). Tal esforço computacional se deve ao fato de que a ME é responsável por procurar o melhor bloco, por critério de semelhança, que será utilizado como referência na Compensação de Movimento (MC) posteriormente (SEIDEL et al., 2014). Para a realização da MC convencional do VVC, é necessário interpolar as amostras do bloco em análise como se fosse uma reamostragem em escala (BROSS et al., 2021). Porém, esse padrão de codificação suporta também um modo de Compensação de Movimento Affine (AMC) que possibilita a análise de movimentos mais complexos que ocorrem entre quadros e não se limita a movimentos simples de translação que são calculados pela Compensação de Movimento Translacional (TMC) (LI et al., 2018).

A Figura 3 mostra a diferença entre a AMC e a TMC, onde está representado um macrobloco contendo 4x4 blocos menores. A AMC precisa de múltiplos vetores de movimento de controle, chamados de $CPMV_0$, $CPMV_1$ e $CPMV_2$ na Figura 3, para gerar um Vetor de Movimento (MV), enquanto a TMC considera apenas um MV de um bloco para representar o MV de todos os outros blocos do conjunto, sem a necessidade de gerar outros vetores (JIN et al., 2022). É devido a essa diferença na quantidade de MV utilizados que se tem diferentes representações de movimento para cada tipo de MC, já que transformação *affine* é uma transformação geométrica que preserva o paralelismo e as

Figura 3 – Diferença entre (a) AMC e (b) TMC.



Fonte: (JIN et al., 2022)

linhas, mas que é capaz de rotacionar, dilatar, translacionar e podar as imagens conforme o movimento detectado.

Esses recursos adicionais aumentam a complexidade de compressão, no entanto o VVC simplifica essa técnica ao utilizar macroblocos a partir dos blocos de pixels, podendo ter dimensões 4x4 até 128x128 blocos de pixels (HUANG et al., 2021). Existem dois tipos de AMC para o VVC, um modelo de quatro parâmetros que pode produzir translação, rotação e escalonamento utilizando dois vetores base. O outro tipo de AMC tem seis parâmetros e implementa as mesmas operações que o de quatro, adicionado a poda e transformação de proporção, utilizando três vetores base (LI et al., 2018).

2.2.1 Geração dos vetores de movimento

Para cada um dos blocos componentes do macrobloco deve haver um MV associado dependente dos ($CPMVs$), que na Figura 3 estão representados com setas verdes. As equações que regem a gereção de tais vetores para o modelo de quatro parâmetros estão descritas em Eq. 2.1 e Eq. 2.2, e de seis parâmetros são as Eq. 2.3 e Eq. 2.4. O vetor de movimento horizontal gerado é representado por $MV^h_{(x,y)}$ e o vertical, $MV^v_{(x,y)}$. E sendo os vetores de movimento dos pontos de controle representados como $CPMV_0$, $CPMV_1$ e $CPMV_2$, diferindo as componentes horizontais e verticais.

$$MV^h_{(x,y)} = \frac{CPMV^h_1 - CPMV^h_0}{W-1}x + \frac{CPMV^v_1 - CPMV^v_0}{W-1}y + CPMV^h_0 \tag{2.1}$$

$$MV^v_{(x,y)} = \frac{CPMV^v_1 - CPMV^v_0}{W-1}x + \frac{CPMV^h_1 - CPMV^h_0}{W-1}y + CPMV^v_0 \tag{2.2}$$

$$MV^h_{(x,y)} = \frac{CPMV^h_1 - CPMV^h_0}{W-1}x + \frac{CPMV^h_2 - CPMV^h_0}{H-1}y + CPMV^h_0 \tag{2.3}$$

$$MV^v_{(x,y)} = \frac{CPMV^v_1 - CPMV^v_0}{W-1}x + \frac{CPMV^v_2 - CPMV^v_0}{H-1}y + CPMV^v_0 \tag{2.4}$$

2.2.2 Interpolação

A partir dos $MV_{(x,y)}$ gerados, ainda é necessário fazer a interpolação para gerar os subpixels fracionários de cada bloco pois a precisão do vetor de movimento é de 1/16. Para isso são utilizado quinze filtros para cada uma dessas precisões fracinárias, onde cada filtro possui *8-taps*, ou seja, oito coeficientes que já são determinados e constantes. A Equação 2.5 demonstra como uma interpolação ocorre. A saída I pode se referir tanto a uma coordenada no eixo x ou y, onde seu valor representa o pixel fracionário gerado pela interpolação. Os coeficientes estão representados de C_0 a C_7 (pois são $8 - taps$) e as entradas correspondem aos valores P_0 a P_7.

$$I = P_0C_0 + P_1C_1 + P_2C_2 + P_3C_3 + P_4C_4 + P_5C_5 + P_6C_6 + P_7C_7 \tag{2.5}$$

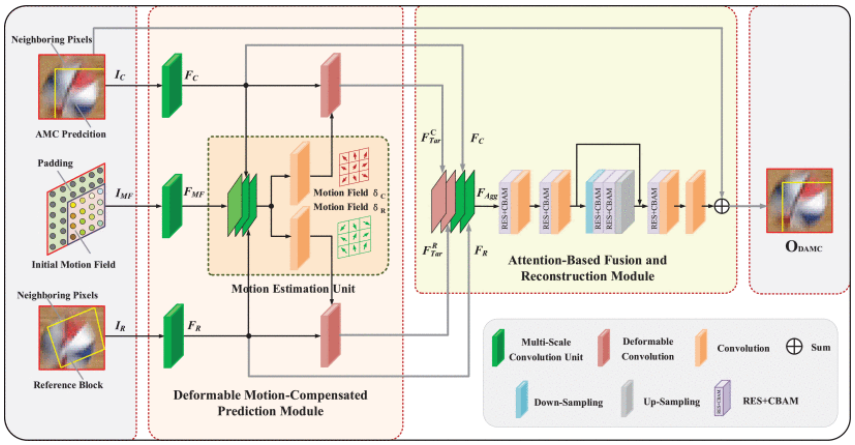
Essa etapa é custosa computacionalmente, pois como apontado pela Equação 2.5, são executadas várias multiplicações e além disso, no piores casos em que há partes fracionárias para ambos os eixos, são necessárias duas passadas de interpolação para cada um dos blocos. Nessa situação, primeiramente são gerados os pixels fracionários horizontais, que servirão de entrada para gerar todos os outros pixels fracionários verticais.

3 Revisão do Estado da Arte

3.1 *Deep Affine Motion Compensation Network for Inter Prediction in VVC*

O artigo (JIN et al., 2022) propôs uma rede de *deep learning* para melhorar a performance da compensação de movimento *affine*, que foi chamado de *Deep Learning Affine Motion Compensation Network* (DAMC-Net). A ideia principal explicada no artigo é compensar o bloco de codificação estimando os campos de movimento gerados por ele em outros domínios e tempos. Essa se mostrou uma alternativa ao AMC existente no VVC, pois a vantagem é que as informações sobre os campos de movimento no DAMC-Net podem ser aprendidos. Assim, o método proposto é capaz de lidar com movimentos mais complexos em vídeos.

Figura 4 – Arquitetura proposta para a DAMC-Net.



Fonte: (JIN et al., 2022)

A Figura 4 mostra como a arquitetura proposta foi esquematizada, onde a primeira entrada IC explora as correlações espaciais ao combinar a previsão da AMC com os pixels vizinhos do bloco. Para a segunda entrada IR, é feita a reconstrução do bloco de referência mais similar ao atual a partir dos vetores de movimento dos pontos de controle, $CPMV$ obtidos pela AMC. E por último, a teceira entrada IMF constitui no campo de movimento inicial que também é derivado dos $CPMV$. Portanto, as três entrdas possuem as informações espaciais e temporais, a partir das quais se extraem *features* no módulo DMCP, que posteriormente passa para o módulo MEU onde serão estimados os campos de movimento. Baseado nessas informações, uma convolucação deformável é utilizada para compensar os movimentos temporais e espaciais das *features* extraídas que

são concatenadas com elas próprias para servirem como entrada do módulo de fusão e reconstrução (ARF). Por fim, gera-se o bloco compensado, chamado de O_{DAMC}.

3.2 *An Efficient Four-Parameter Affine Motion Model for Video Coding*

Em (LI et al., 2018), foi analisado o modelo de quatro parâmetros da AMC que diferentemente do modelo de seis parâmetros, este tipo de transformação requer menor complexidade por possuir apenas quatro graus de liberdade e necessitar de apenas dois vetores de movimento para representá-lo em cada bloco.

Foram propostas duas técnicas para obtenção dos parâmetros da transformação *affine*:

- Predição avançada do vetor de movimento para o *affine* combinado com um algoritmo de *affine* rápida (*fast affine motion estimation*);
- Modelo de fusão *affine* (*merge affine*);
- Modelo com redução de complexidade na interpolação dos sub-pixels.

O primeiro é baseado em técnicas de gradiente para gerar o algoritmo de *affine* que consegue convergir rapidamente, conseguindo reduzir a complexidade de codificação significantemente. A segunda alternativa faz uma fusão na tentativa de reutilizar os parâmetros de movimento do *affine* dos blocos vizinhos ao invés de gerar novos parâmetros, assim este é um modelo mais eficiente já que aproveita essa correlação entre blocos previamente calculados. O último modelo apresentado no artigo propõe duas técnicas de redução de complexidade e de codificação ao aplicar apenas uma passada do filtro de interpolação dos sub-pixels ao invés das duas passadas convencionais. E ainda são considerados a MC baseada em blocos de tamanhos adaptativos ao invés de se basear nos pixels.

3.3 *Fast Affine Motion Estimation for Versatile Video Coding (VVC) Encoding*

O outro artigo estudado como estado da arte é (PARK; KANG, 2019), onde foi proposto um método de codificação rápida que reduz eficientemente a complexidade do cálculo da *affine* para estimação de movimento (AME) para o VVC, obtendo resultados em 65% menos tempo do que o convencional sem perdas significativas de qualidade.

A proposta se baseia em dois processos principais:

- Eliminação de processos redundates, tanto na AME quanto na AMC;
- Redução da quantidade de quadros de refência utilizados pela AME

O primeiro processo foi feito utilizando um método de terminação antecipada baseada na exploração das unidades de codificação vizinhas ao bloco sendo processado. Para tal, a AME é aplicada em duas partes, na primera, o andamento do algoritmo é finalizado antes de iniciar o processamento da unidade de codificação. O melhor resultado obtido então é analisado para decidir se a AME inteira será pulada ou não. E na segunda parte da proposta do artigo, foram reduzidas a complexidade e o tempo gasto no processamento da AME, já que são menos quadros a serem considerados no cálculo.

3.4 Análise dos artigos estudados

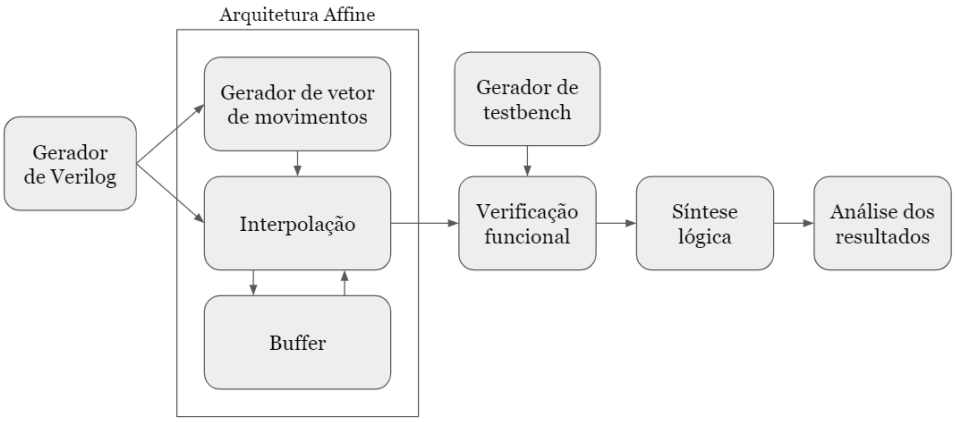
Pela leitura dos arigos explanados acima, pode-se ter uma ideia do que está sendo feito para reduzir a complexidade de computação e tempo para o algoritmo *affine*, onde foram utilizadas técnicas de *deep learning*, terminações antecipadas e técnicas de redução de parâmetros, de quantidade de interpolações e de número de quadros de referências analisados.

Portanto podemos notar a importância de se reduzir os cálculos da *affine* para viabilizar a codificação de vídeo para os usuários finais da tecnologia. Diante disso, existem ainda possibilidades de se explorar métodos de aceleração em *hardware* para a transformação *affine*. Sendo esta solução um componente especializado desse processamento, espera-se que os resultados sejam melhores do que a convencional aplicação do algoritmo em *software*.

4 Metodologia

O desenvolvimento do Trabalho de Conclusão de Curso, como mostrado na Figura 5, se iniciará com mais estudos teóricos sobre a transformação *affine*, que se divide na etapa de geração dos vetores de movimento e na etapa de interpolação. Portanto como forma de estudo, serão implementados algoritmos em *Python* que realizam ambas etapas, assim será possível visualizar os vetores dos pontos de controle e os vetores gerados. Tendo a fundamentação teórica bem estabelecida, será estudada e implementada a arquitetura em *hardware* descrita em *Verilog*.

Figura 5 – Fluxo estimado de trabalho.



Fonte: Autora

Para a elaboração correta da arquitetura do interpolador, será importante incluir um *buffer* para armazenar os pixels fracionários gerados na primeira passda, que serão utilizados no cálculo da segunda passada. Portanto, ainda será necessário estudar sobre o funcionamento desse elemento de memória antes de implementá-lo. Além disso, pode-se utilizar as técnicas para redução do consumo de potência como *clock gating* e *operand isolation*.

A fim de conseguir alterar rapidamente os parâmetros de codificação nos arquivos com descrição em *Verilog*, os *scripts* serão escritos a partir de um Gerador de *Verilog* em *Python*. Do mesmo modo, serão criados os arquivos de *testbench* para a verificação funcional da arquitetura proposta.

Posteriormente, será estudado o fluxo de síntese lógica para circuitos digitais pela e será feita a síntese da arquitetura prosta utilizando as ferramentas da *Synopsys*. Por fim, os resultados de área, potência e tempo serão analisadas para a arquitetura implementada.

5 Cronograma

As atividades foram separadas conforme uma estimativa do que deverá ser feito durante os próximos meses. A Tabela 1 contém os meses e as atividade programadas.

Tabela 1 – Cronograma.

	2022				
Atividades	Ago	Set	Out	Nov	Dez
Estudo teórico: transformação affine e interpolação	X				
Implementação de um algoritmo em software	X				
Estudo teórico: técnicas de baixo consumo de potência		X			
Implementação da arquitetura proposta		X			
Verificação funcional da arquitetura		X	X		
Implementação da síntese lógica			X		
Análise comparativa dos resultados			X	X	
Preparação da defesa do TCC				X	
Defesa do TCC					X

6 Recursos necessários

A seguir, estão listadas os recursos necessários para a realização do Projeto de TCC:

- Referências bibliográficas: livros e artigos da IEEE;
- *Software* para desenvolvimento do *script* em *Verilog*: Quartus II Web Edition;
- *Software* para simulação de estímulos: ModelSim Altera;
- Ferramentas de síntese lógica: Synopsys;
- *Software* para desenvolvimento dos *scripts* em *Python*: Visual Code.

Referências Bibliográficas

BROSS, B. et al. Overview of the versatile video coding (vvc) standard and its applications. *IEEE Transactions on Circuits and Systems for Video Technology*, v. 31, n. 10, p. 3736–3764, 2021. 7, 9

CECI, L. *Hours of video uploaded to YouTube every minute 2007-2020*. 2022. Disponível em: <https://www.statista.com/statistics/259477/hours-of-video-uploaded-to-youtube-every-minute/>. 6

CISCO, U. *Cisco Annual Internet Report (2018-2023) White Paper*. 2020. 6

CORREA, G. et al. *Complexity-Aware High Efficiency Video Coding*. [S.l.]: Springer, 2016. 8

HUANG, Y.-W. et al. Block partitioning structure in the vvc standard. *IEEE Transactions on Circuits and Systems for Video Technology*, IEEE, v. 31, n. 10, p. 3818–3833, 2021. 10

JIN, D. et al. Deep affine motion compensation network for inter prediction in vvc. *IEEE Transactions on Circuits and Systems for Video Technology*, v. 32, n. 6, p. 3923–3933, 2022. 9, 10, 12

LI, L. et al. An efficient four-parameter affine motion model for video coding. *IEEE Transactions on Circuits and Systems for Video Technology*, v. 28, n. 8, p. 1934–1948, 2018. 9, 10, 13

PARK, S.-H.; KANG, J.-W. Fast affine motion estimation for versatile video coding (vvc) encoding. *IEEE Access*, v. 7, p. 158075–158084, 2019. 13

SEIDEL, I. et al. Analysis of pel decimation and technology choices to reduce energy on sad calculation. *Journal of Integrated Circuits and Systems*, v. 9, n. 1, p. 48–59, 2014. 9

STATISTA. *Facebook: number of monthly active users worldwide 2008-2022*. 2022. Disponível em: <https://www.statista.com/statistics/264810/number-of-monthly-active-facebook-users-worldwide/>. 6

STOLL, J. *Netflix subscribers count worldwide 2011-2022*. 2022. Disponível em: <https://www.statista.com/statistics/250934/quarterly-number-of-netflix-streaming-subscribers-worldwide/>. 6

ZHANG, Y.; ZHANG, C.; FAN, R. Fast motion estimation in hevc inter coding: An overview of recent advances. In: *2018 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*. [S.l.: s.n.], 2018. p. 49–56. 9