Learning for new generation video coders

Yiqun Liu

COLLEGE MATHS, TELECOMS
DOCTORAL INFORMATIQUE, SIGNAL
BRETAGNE SYSTEMES, ELECTRONIQUE

Université de Rennes

THÈSE DE DOCTORAT DE

L'UNIVERSITÉ DE RENNES

ÉCOLE DOCTORALE N° 601
Mathématiques, Télécommunications, Informatique, Signal, Systèmes, Électronique
Spécialité : *Signal, Image, Vision*

Par
Yiqun LIU

Learning for new generation video coders

Complexity Reduction of Inter coding for VVC Codec

Thèse présentée et soutenue à Rennes, le 11 Decembre 2023
Unité de recherche : INRIA Rennes - Bretagne Atlantique

Rapporteurs avant soutenance :

Marco CAGNAZZO Professeur, Université de Padoue
Mounir KAANICHE Professeur, Université Sorbonne Paris Nord

Composition du Jury :

Président :	Daniel MENARD	Professeur, INSA de Rennes
Examinateurs :	Marco CAGNAZZO	Professeur, Université de Padoue
	Mounir KAANICHE	Maître de Conférences, HDR, Université Sorbonne Paris Nord
	Daniel MENARD	Professeur, INSA de Rennes
	Enzo TARTAGLIONE	Maître de conférences, Télécom-Paris
Dir. de thèse :	Christine GUILLEMOT	Directrice de Recherche, INRIA Rennes
Co-dir. de thèse :	Thomas GUIONNET	Ingénieur de recherche, Ateme

Invité(s) :

Aline ROUMY Directrice de Recherche, INRIA Rennes
Mohsen ABDOLI Ingénieur de recherche, XiaoMi
Marc RIVIERE Ingénieur de recherche, ATEME

Acknowledgement

I would like to express my sincere gratitude to my supervisor, Prof. Christine GUILLEMOT, Prof. Aline ROUMY and Dr. Thomas GUIONNET, for guiding and supporting me during my PHD studies. They have put a lot of efforts in helping me with my research by providing valuable guidance and suggestions.

I would also like to thank Mohsen ABDOLI, Marc RIVIERE and Hadi AMIRPOUR. They have had inspiring meetings and discussions with me to help me with my research.

I want to express my gratitude to my family and friends for their consistent backing and motivation during my thesis. Your affection and encouragement have been the driving force behind my determination and creativity.

Table of Contents

RÉSUMÉ EN FRANÇAIS

Contexte

La consommation de vidéos en ligne a explosé ces dernières années, en raison de multiples facteurs qui ont remodelé notre vie quotidienne. Premièrement, l'émergence de plateformes vidéo en ligne à la demande (e.g Netflix, TikTok) et de services de streaming live ont rendu les contenus vidéo plus accessible que jamais. D'autre part, le streaming de jeux et de sports en temps réel a acquis une immense popularité, contribuant de manière significative à l'augmentation du trafic vidéo. De plus, la demande croissante de contenus de meilleure qualité a conduit à la création et à la diffusion de vidéos en très haute résolution (e.g 4K). De nouvelles expériences immersives, en cours de développement, visent à améliorer l'expérience utilisateur au prix d'une consommation élevée de ressources de calcul et de bande passante. Enfin, la transition mondiale vers le travail à distance après la pandémie de COVID-19 repose sur des outils de communication vidéo pour les activités liées au travail. Les vidéoconférences, les webinaires et les réunions virtuelles sont devenus des éléments essentiels de la culture du travail moderne, augmentant le trafic vidéo en raison du besoin de connectivité constante. Ensemble, ces tendances ont entraîné une augmentation exponentielle du trafic vidéo. D'après une enquête de Sandvine [1], le trafic vidéo a été augmenté de 24% en 2022 et représente 65% du trafic Internet total en 2023.

L'augmentation de la bande passante Internet a considérablement facilité l'accès à ces contenus sur une variété d'appareils, mais ne suffit pas à soutenir l'accélération de la consommation de contenus vidéos. C'est pourquoi l'évolution des normes de compression est d'une importance majeure. Depuis la norme H.261 en 1988, les chercheurs et ingénieurs n'ont cessé de travailler à l'amélioration et à la proposition de nouveaux codecs vidéo pour s'adapter à de nouvelles contrainte en terme de définition, fréquence des images, d'espace colorimétrique, ...

Des premières normes comme AVC aux codecs récents comme HEVC et VVC, tous ces codecs ont conservé jusqu'à maintenant la structure de codage hybride basée sur les blocs, tout en améliorant considérablement l'efficacité d'encodage à qualité constante. Cette amélioration des performances d'encodage est due à des algorithmes avancés, à

une compensation de mouvement améliorée, à une meilleure prédiction et à un codage entropique plus efficace. Le dernier codec vidéo, VVC a été finalisé en juillet 2020. Il permet de réduire d'environ 40% les débits par rapport au HEVC à qualité constante. Plus précisément, VVC se distingue par son efficacité à transmettre du contenu haute résolution, y compris des vidéos 4K et 8K, ainsi que du contenu immersif.

Néanmoins, ce gain sur les performances de codage de VVC se fait au prix d'une complexité de calcul largement accrue. Une approche pour évaluer la complexité d'un codec consiste à comparer le temps d'encodage de son implémentation de référence avec les codecs précédents. Concrètement, le temps d'encodage de VVC est de 6,6 à 25,8 fois plus long que celui de HEVC [2] selon les configurations d'encodage, ce qui limite son adoption notamment dans les scénarios de diffusion vidéo en continue. Cette complexité a également un impact sur la consommation énergétique aussi bien pour l'encodage que le décodage. Enfin, elle rend l'implémentation d'un encodeur basse latence très difficile. Ces limites sont détaillées dans la section suivante.

Motivations et Objectifs

Selon l'étude Cisco [3], le nombre d'appareils connectés au réseau pourrait atteindre 29,3 milliards d'ici 2023. Une majorité de ces équipements sont des systèmes embarqués sur lesquels une grande quantité de données vidéo est traitée. D'après le rapport de mobilité d'Ericsson [4], le trafic vidéo représente 71% de l'ensemble du trafic de données mobiles en 2023. Cette proportion devrait atteindre 80% en 2028. Selon l'enquête [5] menée auprès des consommateurs du monde entier, les répondants ont regardé en moyenne 17 heures de contenu vidéo en ligne par semaine en 2023. Une grande partie de ces équipements mobiles utilise une batterie comme source d'énergie, batterie qui se déchargera plus vite lors des tâches de traitement vidéo utilisant des codecs plus complexes. Ainsi, il est primordial de réduire la complexité de ces nouveaux codecs tout en minimisant l'impact sur le compromis débit-distorsion.

Selon [6], les centres de données représentent 2,7% de la demande d'électricité en 2018. Une grande partie de ces centres de données sont dédiés à l'encodage ou au transcodage vidéo. L'utilisation d'un codec plus léger peut réduire efficacement la consommation énergétique du traitement vidéo, réduisant ainsi une grande partie du coût énergétique des plateformes de streaming comme Netflix. En 2022, Netflix [7] estime la consommation de ses centres de données à 36110 mégawattheures. Au vu de ces chiffres, il est clair que

la réduction de la complexité des codecs vidéo peut réduire considérablement à la fois la consommation énergétique de ces centres de données et les émissions de gaz à effet de serre.

Les codecs vidéo sont utilisés dans différents contextes, avec des contraintes très différentes. Pour l'encodage de vidéos destinées à une plateforme de vidéo à la demande, l'efficacité d'encodage est bien plus importante que le temps d'encodage. En effet, dans ce scénario, quelques encodages sont nécessaires à la distribution à des millions d'utilisateurs. Dans le cas de la diffusion vidéo en continue, le temps et la latence sont favorisés à l'efficacité d'encodage. Dans ce contexte, et malgré ses promesses de réduction de débit, la complexité du codec VVC est un frein majeur à son adoption. La complexité de VVC, en particulier pour le processus d'encodage, doit encore être réduite. Le partitionnement dans Versatile Video Coding (VVC) est le principal contributeur à sa complexité. En particulier, il a été observé dans [8] que l'encodeur VVC consacre 97% de son temps d'encodage à la recherche de la partition optimale. Par conséquent, les méthodes de partitionnement émergent comme les approches les plus prometteuses pour accélérer l'ensemble du processus d'encodage de VVC.

L'objectif de cette thèse est de proposer des methodes de partitionnement rapides pour réduire efficacement la complexité d'encodage de VVC. Notre objectif est d'accélérer le processus de partitionnement dans deux cas particuliers: le cas d'un encodage isolé et le cas d'une série d'encodages d'une même séquence en multi-débit. Dans le premier cas, nous proposons de limiter l'espace de recherche de partitionnement d'un encodeur VVC à l'aide d'un réseau de neurones convolutionnel (CNN). Dans le deuxième cas, nous proposons une méthode pour exploiter les choix de partitionnement obtenus suite à un encodage d'une séquence à un certain débit pour les encodages de la même séquence à d'autres débits. L'objectif général des méthodes de partitionnement rapide ci-dessus est d'accélérer le processus de codage avec une perte de qualité raisonnable afin de maintenir le gain de codage du codec VVC par rapport aux codecs précédents.

Contributions

Ce manuscrit comprend trois chapitres distincts correspondant à trois contributions principales et une analyse statistique en Annexe.

Étant donné que VVC est un codec vidéo récemment finalisé, il n'y a que peu d'études existantes dans la littérature sur la réduction de la complexité de VVC par rapport

aux codecs précédents, et encore moins sur l'accélération d'encodage inter. Cependant, l'accélération d'encodage inter est essentiel pour le déploiement du codec VVC dans une application industrielle. Dans cette thèse, nous contribuons à l'accélération de l'encodage inter en VVC par des méthodes de partitionnement rapide. Nos travaux ont été intégrés dans différentes implémentations de VVC. Les tests montrent que nos méthodes surpassent l'état de l'art, contribuant au passage à l'encodage en temps réel VVC. Les trois principales contributions sont décrites ci-dessous:

Accélération du partitionnement de l'inter dans VVC basée sur un CNN léger

Pendant le processus d'encodage, de nombreuses partitions sont testées dans la recherche de partitionnement, représentant la majorité de la complexité. Par conséquent, nous proposons une méthode basée sur un CNN pour réduire l'espace de recherche du partitionnement inter afin d'accélérer les traitements. Notre méthode opère au niveau CTU, où chaque CTU est divisé en une grille fixe de 8×8 blocs. Chaque cellule de la grille contient des informations sur la profondeur de partitionnement dans cette zone spécifique. Au cours du processus d'optimisation taux-distorsion, nous utilisons un CNN léger pour prédire cette grille, dans le but de limiter la recherche de split MT et d'éviter les partitions inutiles qui ont peu de chances d'être sélectionnées. Les résultats expérimentaux montrent que la méthode proposée atteint une accélération allant de 17% à 30% avec une légère perte de qualité d'encodage dans le RAGOP32 de l'implémentation VTM10 en tant que VVC.

Partitionnement rapide de l'encodage inter du VVC par CNN basé sur le champ de mouvement multiscale

Sur la base de la contribution précédente, nous proposons une nouvelle structure de Convolutional Neural Network (CNN) et un nouvel algorithme de partitionnement rapide mieux adapté au schéma de partitionnement de VVC pour réduire davantage l'espace de recherche du partitionnement inter. Notre approche implique le développement d'un CNN basé sur U-Net, qui prend en entrée un champ de vecteurs de mouvement à plusieurs échelles au niveau du Coding Tree Unit (CTU). L'objectif de l'inférence du CNN est de prédire le chemin de partitionnement optimal lors du processus Rate-Distortion Optimization (RDO). Pour ce faire, notre CNN divise le CTU en grilles et prédit la profondeur

de partitionnement Quadtree (QT), ainsi que les décisions de split Multi-type Tree (MT) pour chaque cellule de la grille. De plus, nous introduisons un algorithme d'accélération qui utilise les prédictions du CNN à chaque niveau de partitionnement pour éliminer les chemins de partitionnement peu probables. Pour équilibrer la complexité et l'efficacité, nous concevons un schéma de sélection par seuil adaptatif qui offre la scabilité. Les résultats expérimentaux montrent que la méthode proposée permet d'obtenir une accélération allant de 16,5% à 60,2% avec une baisse raisonnable d'efficacité, surpassant les solutions dans l'état de l'art en termes de compromis entre l'accélération et l'efficacité.

Préparation de VVC pour le streaming: une approche d'encodage multi-débit rapide

Les contributions susmentionnées sont implémentées et évaluées à l'aide de l'implémentation de référence de VVC, qui diffère considérablement des implémentations d'encodage en temps réel dans des contextes industriels. L'objectif de cette contribution est d'accélérer le partitionnement dans les scénarios de streaming basés sur une implémentation légère de VVC. Comme mentionné précédemment, l'application industrielle de VVC est un défi pour les fournisseurs de services en raison de sa grande complexité. Ces défis sont encore amplifiés dans les scénarios de streaming où le même contenu doit être encodé à plusieurs débits (représentations) pour s'adapter aux différentes conditions du réseau.

Pour répondre au besoin d'encodage plus rapide de plusieurs représentations dans VVC, nous proposons une méthode qui exploite la carte d'encodage d'une représentation de référence. En utilisant la structure de partitionnement de la représentation de référence, nous accélérons le processus de codage des représentations restantes, appelées représentations dépendantes. Les résultats expérimentaux démontrent une réduction significative du temps d'encodage pour les représentations dépendantes, atteignant une accélération de 40% avec une petite baisse de qualité d'encodage. Cette approche permet aux fournisseurs de services d'encoder plus efficacement plusieurs représentations du même contenu, répondant ainsi aux défis de calcul associés aux codecs avancés dans les environnements de diffusion en continu.

L'annexe de la thèse porte sur l'analyse de la complexité de l'encodeur VVC lors du partitionnement inter, peu exploré dans l'état de l'art contrairement au partitionnement intra. Nous effectuons une analyse statistique de la complexité du codage inter, en nous concentrant spécifiquement sur les tailles de bloc et les modes de codage inter. Cette anal-

yse identifie les tailles de bloc et les modes de codage fréquemment utilisés et ceux qui contribuent aux plus grandes portions de complexité. Notre travail fournit des informations sur les modèles de complexité du codage inter dans VVC, permettant la conception de méthodes d'accélération plus efficaces.

Plan du Manuscrit

Cette section fournit un bref résumé du contenu couvert dans chaque chapitre de ce document:

Dans le ***Chapitre 1***, une introduction générale sur la compression vidéo est donnée. Dans la première section, les caractéristiques de base de la vidéo sont présentées. Ensuite, nous résumons l'évolution des normes de codage vidéo dans la section suivante. Les codecs HEVC et VVC sont également spécifiquement présentés dans cette section. Dans la dernière section, nous présentons brièvement le schéma de codage vidéo de VVC, y compris le schéma de partitionnement, le codage inter/intra, le processus RDO et d'autres informations supplémentaires.

Chapitre 2 donne un aperçu des états de l'art dans nos domaines de recherche. Cette section comprend les travaux connexes dans trois domaines clés: l'analyse de la complexité de VVC, les méthodes de partitionnement rapide et les méthodes de codage multi-débits. Chacune de ces parties correspond à nos contributions aux chapitres 3, 4 et 5, et l'Annexe, respectivement.

Chapitre 3 se concentre sur l'accélération de l'encodage inter par une méthode de partitionnement rapide basée sur l'apprentissage automatique. Un CNN est formé et intégré dans l'encodeur VVC. En utilisant les prédictions CNN, un sous-ensemble de partitions est exclu de RDO, ce qui réduit efficacement l'espace de recherche de partitionnement. Nous obtenons une accélération évolutive de l'encodage VVC sous la configuration inter avec une légère perte.

Dans le ***Chapitre 4***, le travail du chapitre précédent est étendu et encore amélioré. Un CNN avec une nouvelle structure est proposé. Contrairement à la contribution du chapitre 3, ce travail atteint une accélération plus élevée en sautant un plus grand nombre de partitions pendant RDO sur la base de la prédiction du CNN. En implémentant cette méthode dans l'encodeur VVC, nous obtenons une réduction significative de la complexité tout en dépassant l'état de l'art en termes de compromis entre l'accélération et la perte.

Chapitre 5 présente une méthode de partitionnement en multi-débit plus rapide.

Nous exploitons les informations de partition collectées à partir de l'encodage à un débit binaire inférieur pour accélérer l'encodage à un débit binaire supérieur. L'expérience montre que cette méthode offre une accélération considérable sur une implémentation rapide de l'encodeur VVC.

Dans le ***Chapitre 6***, nous concluons notre thèse en résumant nos contributions et en discutant des perspectives futures de nos travaux.

Annexe fournit une étude de complexité sur le codage inter VVC. La complexité d'encodage des différentes tailles de bloc et les modes de codage inter sont étudiés dans ce chapitre. De plus, nous analysons également le taux de sélection pour les tailles de bloc et les modes de codage inter. Sur la base de l'analyse ci-dessus, l'opportunité de réduire la complexité est discutée.

INTRODUCTION

Context

Online video consumption has been surging in recent years, which is driven by multiple factors that have reshaped our daily life. First, the emergence of video-on-demand platforms (*e.g.* Netflix, TikTok) and live streaming services has made video content more accessible than ever before. Furthermore, real-time game and sport streaming have gained immense popularity, contributing significantly to the surge in video traffic. Additionally, the increasing demand for higher-quality content has led to the creation and distribution of very high-resolution videos (e.g., 4K). New immersive experiences, currently in development, aim to enhance the user experience at the cost of high consumption of computational resources and bandwidth. In the end, the global shift towards remote working after the COVID-19 pandemic relies on video communication tools for work-related activities. Video conferencing, webinars, and virtual meetings have become essential components of modern work culture, increasing video traffic due to the need for constant connectivity. Collectively, these trends have led to an exponential increase in video traffic. According to a survey by Sandvine [1], video traffic increased by 24% in 2022 and represents 65% of the total Internet traffic in 2023.

The increase in Internet bandwidth has significantly facilitated access to these contents on a variety of devices but is not sufficient to support the acceleration of video content consumption. This is why the evolution of compression standards is of vital importance.From the H.261 standard in 1988, researchers and engineers have been continuously working on improving and proposing novel video codecs to adapt to new constraints in terms of resolution, frame rate, color space, and more. Since the H.261 standard in 1988, researchers and engineers have continuously worked on improving and proposing new video codecs to adapt to new constraints in terms of resolution, frame rate, color space, ... From early standards like Advanced Video Coding (AVC) to recent codecs like High Efficiency Video Coding (HEVC) and VVC, all of these codecs have retained the hybrid block-based coding structure while significantly improving encoding efficiency at constant quality. This improvement on coding performance is due to advanced algorithms, improved motion

compensation, better prediction, and more efficient entropy coding. The latest video codec is the VVC, finalized in July 2020. VVC achieves roughly 40% bit-rate savings compared to HEVC at constant quality. Specifically, VVC stands out for its efficiency in transmitting high resolution content, including 4K and 8K videos, as well as immersive content.

However, this gain in coding performance of VVC is at the cost of a largely increased computational complexity of coding. One approach to evaluate the complexity of codec is to compare the encoding time of its reference implementation with previous codecs. Concretely, the encoding time of VVC is from 6.6 to 25.8 times longer than that of HEVC [2] depending on coding configurations, which limits its adoption, especially in video streaming scenarios. This complexity also has an impact on energy consumption, both for encoding and decoding. Moreover, it makes the implementation of a low-latency encoder very challenging. These limitations are detailed in the following section.

Motivations and Objectives

According to the Cisco study [3], the number of devices connected to the network could reach 29.3 billion by 2023. The majority of these devices are embedded systems on which a large amount of video data is processed. According to the Ericsson Mobility Report [4], video traffic is estimated to account for 71% of all mobile data traffic in 2023. This proportion is expected to reach 80% in 2028. In a survey [5] conducted among consumers worldwide, respondents watched an average of 17 hours of online video content per week in 2023. A significant portion of these devices relies on a battery as their power source, and this battery will drain faster during video processing tasks that use more complex codecs. Therefore, it is crucial to reduce the complexity of these new codecs while minimizing the impact on the bitrate-distortion trade-off.

According to [6], data centers account for 2.7% of electricity demand in 2018. A large part of these data centers are dedicated to video encoding or transcoding. Using lighter video codecs can effectively reduce the energy consumption of video processing, reducing a large part of the energy cost for streaming platforms such as Netflix. In 2022, Netflix estimated the power consumption of its data centers to be 36,110 megawatt-hours. Given these figures, it is evident that reducing the complexity of video codecs can significantly decrease both the energy consumption of these data centers and greenhouse gas emissions.

Video codecs are used in various contexts, each with different constraints. For encoding

videos destined for video-on-demand platforms, encoding efficiency is far more important than encoding speed. In this scenario, only a few encodings are needed to distribute content to millions of users. On the other hand, in the case of live video streaming, encoding speed and low latency take precedence over encoding efficiency. In this context, despite despite its bitrate reduction compared to previous codecs, the complexity of the VVC codec is a significant obstacle to its adoption. The complexity of VVC, especially in the encoding process, still needs to be reduced. The partitioning scheme in VVC is the main contributor to its complexity. In particular, it has been observed in [8] that the VVC encoder dedicates 97% of its encoding time to searching for the optimal partition. Consequently, fast partitioning methods emerge as the most promising approaches to speed up the whole VVC encoding process.

Therefore, the objective of this thesis is to propose fast partitioning methods to efficiently reduce the encoding complexity of VVC. Our objective is to accelerate the partitioning process in two specific cases: single encoding and multi-rate encoding of the same video. In the first case, we propose limiting the partitioning search space of a VVC encoder using a CNN. In the second case, we propose a method to leverage the partitioning decisions obtained from encoding a sequence at a certain bitrate for encodings of the same sequence at other bitrates. General purpose of above fast partitioning methods is to accelerate the encoding process with reasonable quality loss so that the coding gain of VVC codec comparing to previous codecs is maintained.

Contributions

This manuscript consists of three distinct chapters corresponding to three main contributions and a statistical analysis in Annex. The latter is a preliminary study that analyzes the frequency and complexity of different inter coding modes and Coding Unit (CU) sizes in VVC to identify acceleration opportunities. Since VVC is a video codec recently finalized, there are only a few existing studies on VVC complexity reduction comparing to previous codecs, and even fewer on the acceleration of inter coding, in the literature. However, fast inter coding is essential for the deployment of VVC codec in industrial application. In this thesis, we contribute to the acceleration of inter coding in VVC by fast partitioning methods. Our work has been integrated into different implementations of VVC. Tests show that our methods outperform state of the art, contributing to the move for VVC real-time encoding. The three principle contributions are outlined

below:

Light-weight CNN-based VVC Inter Partitioning Acceleration

During the encoding process, numerous partitions are tested in the partitioning search, accounting for the majority of complexity [8]. Consequently, we propose a CNN-based method to reduce the search space of inter partitioning to speed up the inter coding process. Our method operates at the CTU level, where each CTU is divided into a fixed grid of 8×8 blocks. Each cell in the grid contains information about the partitioning depth within that specific area. During the rate-distortion optimization process, we employ a lightweight CNN to predict this grid, aiming to limit the search for MT splits and avoid unnecessary partitions that are unlikely to be selected. Experimental results demonstrate that the proposed method achieves acceleration ranging from 17% to 30% with slight encoding quality loss in the RandomAccess Group Of Picture 32 (RAGOP32) of the VVC Test Model (VTM)10 as VVC implementation.

VVC Fast Inter Partitioning by Multi-Scale Motion Field Based CNN

Based on the previous contribution, we propose a novel CNN structure and a new fast partitioning algorithm more adapted to VVC partitioning scheme to further reduce the search space of inter partitioning. Our approach involves the development of a U-Net-based CNN, which takes a multi-scale motion vector field as input at the CTU level. Our CNN divides the CTU into grids and predicts the QT partitioning depth, as well as the MT split decisions for each cell in the grid. In such way, optimal partition paths of the CTU are predicted. Additionally, we introduce an efficient partition pruning algorithm that utilizes the CNN predictions at each partitioning level to skip unnecessary partition paths. To balance complexity and efficiency, we design an adaptive threshold selection scheme that offers scalability. Experimental results show that the proposed method achieves an acceleration ranging from 16.5% to 60.2% with a reasonable efficiency drop, exceeding those of other state-of-the-art solutions.

Preparing VVC for Streaming: A Fast Multi-Rate Encoding Approach

The aforementioned contributions are implemented and evaluated using the reference implementation of VVC, which differs significantly from real-time encoding implementations in industrial contexts. The target of this contribution is to speed up the partitioning in streaming scenarios based on a light-weight implementation of VVC. As previously mentioned, the industrial application of VVC is a challenge for service providers due to its high complexity. These challenges are further amplified in streaming scenarios where the same content needs to be encoded at multiple bitrates (representations) to accommodate varying network conditions.

To address the need for faster encoding of multiple representations in VVC, we propose a method that takes advantage of the encoding map of a reference representation. By utilizing the partitioning structure of the reference representation, we accelerate the encoding process of the remaining representations, known as dependent representations. Experimental results exhibit a significant reduction in encoding time for the dependent representations, achieving a 40% speedup with a small encoding quality drop. This approach enables service providers to encode multiple representations of the same content more efficiently, thereby addressing the computational challenges associated with advanced codecs in streaming environments.

The last stage of the thesis focuses on the statistical analysis of the VVC encoder during inter coding, which is less explored in the state of the art compared to intra coding. We conduct a statistical analysis of inter coding complexity, specifically focusing on CU sizes and inter coding modes. This analysis identifies frequently used CU sizes and coding modes, as well as those that contribute to the largest portions of complexity. Our work provides insights into the complexity patterns of inter coding in VVC, enabling the design of more efficient acceleration methods.

Outline

This section provides a brief summary of the content covered in each chapter of this document:

In ***Chapter 1***, an overview of video compression is given. In the first section, the basic characteristics of video are presented. Then we summarize the evolution of video

coding standards in the following section. The HEVC and VVC codec is also specifically introduced in this section. In the last section, we briefly present the video coding scheme of VVC, including the partitioning scheme, inter/intra coding, RDO process and other additional information.

Chapter 2 provides an overview of the current state-of-the-art methods in our research domains. This section comprises the related work in three key areas: complexity analysis of VVC, fast partitioning methods, and multi-rate coding methods. Each of these parts corresponds to our contributions in Chapter 3, Chapter 4 and 5, and Annex, respectively.

Chapter 3 focus on accelerating inter coding by machine-learning-based fast partitioning method. A CNN is trained and integrated into VVC encoder. Utilizing CNN predictions, a subset of partitions is excluded from RDO, effectively reducing the partitioning search space. We achieve scalable acceleration of VVC encoding under RandomAccess (RA) configuration with slight loss.

In ***Chapter 4***, the work in previous chapter further improved. A CNN with novel structure is proposed. In contrast to the contribution in Chapter 3, this work attains higher acceleration by skipping a larger number of partitions during RDO based on the prediction of the CNN. By implementing this method in VVC encoder, we achieve significant complexity reduction while surpassing the state of the art in terms of trade-off between acceleration and loss.

Chapter 5 presents a multi-rate faster partitioning method. We leverage the partition information collected from the encoding at lower bitrate to speed up the encoding at higher bitrate. Experiment shows that this method offers a considerable speed-up on a fast implementation of VVC encoder.

In ***Chapter 6***, we conclude our thesis by summarizing our contributions and discussing the future perspectives of our work.

Annex provides a statistical analysis on VVC inter coding. The encoding complexity of different CU sizes and inter coding modes are investigated in this chapter. In addition, we also analyze the rate of selection for CU and inter coding modes. Based on the above analysis, the opportunity for complexity reduction of VVC inter coding is discussed.

Chapter 1

Video Coding

In 2022, the number of Internet users watching streaming or downloading videos at least once a month surpassed 3 billion [9]. For each video viewer, the average time spent on video consumption is one hundred minutes per day [10]. Such a huge demand for video transmission puts pressure on network bandwidth, while uncompressed High Definition (HD) video footage can take up to 10 GB of space per minute. As a consequence, it is critical that researchers develop efficient video coding standards to compress the video and make its transmission affordable for the Internet infrastructure. In addition, video coding standards set a unified standard in the broadcast chain. To be more precise, the video stream encoded using a specific coding standard by the broadcaster can be decoded and played on any consumer device that supports that particular standard.

Moving Picture Experts Group (MPEG) is a working group of International Organization for Standardization (ISO) and International Electrotechnical Commission (IEC). This group has been continuously working on providing video coding standard specifications. As a result of the collaboration between MPEG and International Telecommunication Union - Telecommunication Standardization Sector (ITU-T), Joint Video Exploration Team (JVET) was created in 2017 to develop the latest video coding standard: VVC. Like previous coding standards, VVC adopts the general block-based hybrid video coding scheme.

In this chapter, we first present the characteristics of video files. We then briefly introduce the coding standards, including HEVC and VVC. In Section 1.3, the above-mentioned coding scheme is first introduced as the background of this thesis. Since this thesis focuses on the VVC standard, coding modules in VVC that are relevant to this thesis are presented. These modules include partitioning, inter coding, intra coding, and the RDO process. To highlight the novel features of VVC, we compare it with HEVC in this section. Finally, introductions to the temporal coding structure and quality evaluation metrics are provided as additional information for a better understanding of our work.

1.1 Video Characteristics

A video file can be characterized by its resolution, bit-depth, colour sampling, and video format etc. In this section, we will explain briefly these concepts before moving on to the topic of video compression.

Firstly, a video is made up of sequential frames and each frame contains a matrix of pixels with W columns and H rows denoted by $W \times H$. The dimension of the frame is defined as the spatial resolution of the video. Resolution directly influences the clarity and realism of the video. In general, a higher resolution corresponds to a higher level of clarity in the video. Conventionally, the most frequently utilized resolutions are as follows.

— 3840x2160 (*e.g.*, 4k, Ultra High Definition (UHD))
— 2560x1440 (*e.g.*, 1440p, Quad High Definition (QHD))
— 1920x1080 (*e.g.*, 1080p, Full High Definition (FHD))
— 1280x720 (*e.g.*, 720p, HD)
— 832x480 (*e.g.*, 480p, Standard Definition (SD))
— 416x240 (*e.g.*, 240p, Low Definition (LD))

Regarding the pixel format, YUV (*i.e.* YCbCr) is the color representation widely used for color TV standards, which consists of three components for each pixel: Y is the luma component and U and V are the chroma components. Pixel components can also be referred to as channels. Chroma subsampling involves encoding images with reduced resolution for chroma information compared to luma information, based on the fact that the human visual system is less sensitive to color differences in comparison to luminance. Consequently, the chroma samples are stored with reduced resolution as illustrated in Figure 1.1. Chroma format could be expressed by $j{:}a{:}b$. The j represents the number of luminance samples per row, and a and b indicate the number of unique samples for the upper row and the lower row, respectively. Generally, the most common chroma expressions are 4:4:4, 4:2:2 and 4:2:0.

The number of bits used to store each component of the pixel value is defined as the bit depth. Bit depth values typically range from 8 to 16 bits, which are the most commonly used formats in various applications. In this document, the pixel format involved is the YUV format with chroma sampling 4:2:0 and a bit depth of 8 bits or 10 bits, denoted as yuv420p / yuv420p10. Nevertheless, our proposed methods can be applied to other formats.

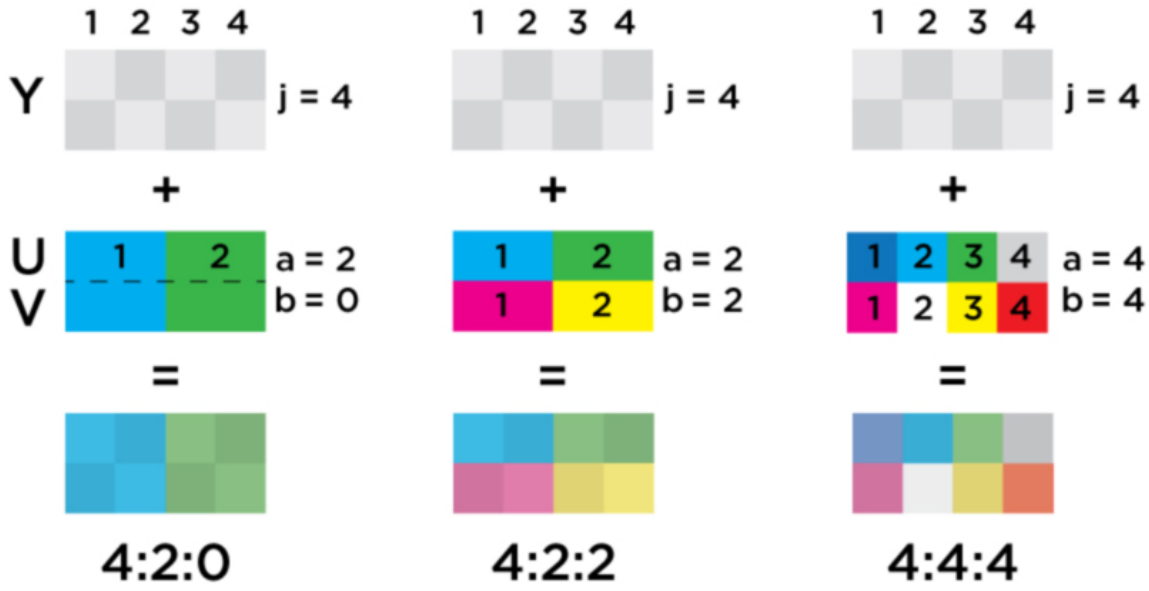


Figure 1.1 – Chrominance sub-sampling [11]

Secondly, the temporal resolution is determined by its frame rate, which is usually expressed as Frames per Second (FPS). The frame rate represents the frequency at which frames change in the video. The first 5 frames of a raw video in the format yuv420p with a resolution of 416x240 at a frame rate of 50 FPS are presented in Figure 1.2, where each frame is refreshed every 0.02 seconds.

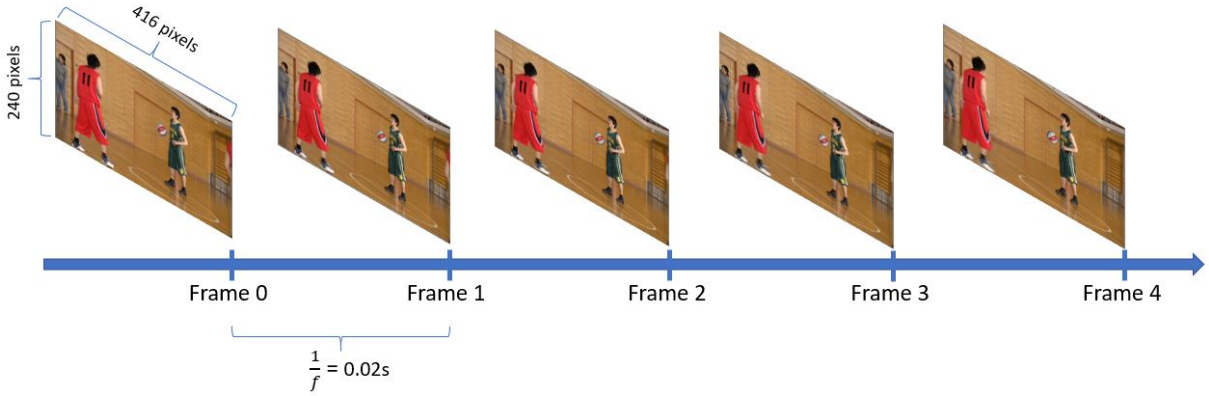


Figure 1.2 – Resolution and frame rate of video

The bandwidth needed for transmitting a raw video could be computed as follows:

$$Bandwidth(bits/s) = \frac{W \times H \times FPS}{8} \times 2 \times (j + a + b) \times bitdepth \tag{1.1}$$

For the 240p video mentioned above, a bandwidth of 57.1 Mb/sec is required. However, for a 4K video at 10 bits, the required bandwidth increases significantly to 5.8 Gb/sec, which is mostly unaffordable based on average user's network conditions. Consequently, the compression capabilities of video codecs are necessary to effectively reduce the bandwidth required for video transmission. In the next section, we will introduce background information about video coding.

1.2 Video Coding Standard

This section provides a brief introduction to the video coding standards and principles of coding relevant to this thesis. In the first part, we present a general overview of the evolution of video coding standards. Then, in the second part, we present the video coding standard that is the focus of our work: VVC. We will begin by introducing HEVC, which was developed prior to VVC, and its reference software, HEVC Test Model (HM). Next, we will introduce VVC and its reference software, VTM, as well as a fast and efficient implementation of VVC called VVenc. Then, VVC is compared to HEVC to reflect its improvements and characteristics. The third part provides background information on video coding. Firstly, the general coding scheme is illustrated. Afterwards, the partitioning scheme used to divide the frame into sub-blocks is explained. The coding of these sub-blocks can be either intra coding (discussed in Section 1.3.3) or inter coding (discussed in Section 1.3.4). Next, the coding results of a search space of coding decisions undergo the RDO process (discussed in Section 1.3.5) and the best coding decisions are selected for encoding. Furthermore, Section 1.3.6 presents the temporal coding structure in VVC, including the type of slice and frame, Group Of Picture (GOP) structure, and coding configurations. Finally, various quality evaluation metrics for assessing the coding loss are described in Section 1.3.7.

1.2.1 Evolution of Coding Standards

Video coding is the process of compressing and encoding raw video data to reduce its size while preserving its quality as much as possible. It is an essential technology in various fields, such as streaming and broadcasting. The growing demand for HD and UHD videos

has led to an increasing need for more efficient and advanced video coding algorithms and techniques. The development of video coding standards has been advancing rapidly, as shown in Figure 1.3.

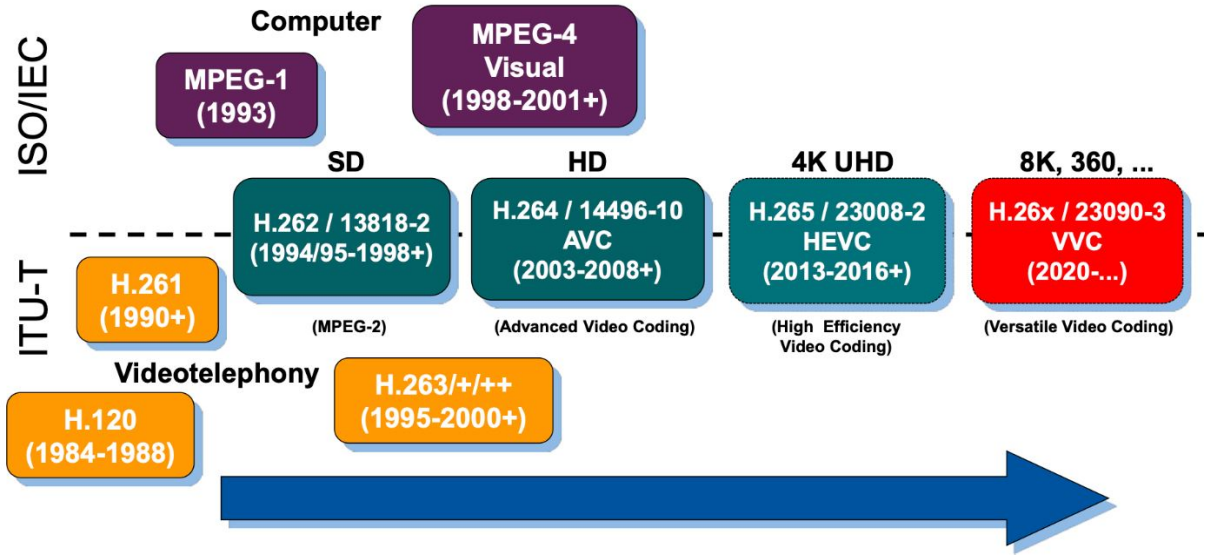


Figure 1.3 – History of international video coding standardization[12]

The most widely used video coding standard [13] in the industry so far remains the H.264/AVC format introduced in 2003. AVC uses block-based hybrid coding, which combines Discrete Cosine Transform (DCT) and motion estimation to achieve high compression efficiency. However, with the emergence of HD and UHD videos, AVC has limitations in terms of compression efficiency.

To address the limitation of coding efficiency, the ITU-T and International Organization for Standardization / International Electrotechnical Commission (ISO/IEC) jointly developed the HEVC standard in 2013. Compared to AVC, HEVC uses advanced techniques such as more variable block sizes, motion vector-based motion estimation, and extended intra prediction modes to achieve better performance. HEVC can achieve a bitrate reduction of 49.3% [14] with the same quality as in AVC.

However, even with advances in HEVC, there is still room for improvement in terms of compression efficiency, especially for low-bit-rate applications. To address this challenge, the ITU-T and ISO/IEC finalized the VVC standard in 2020. VVC introduces various novel coding tools and a more sophisticated partitioning structure. In return, its bit-rate reduction is approximately 50% [2] when compared to HEVC.

In summary, video coding has evolved significantly over the years, with each new gen-

eration of standards providing improved compression efficiency and better performance. From AVC to HEVC and now VVC, the industry continues to push the boundaries of what is possible in terms of video compression and quality. The remainder of this section will focus on providing an overview of the video coding standards HEVC and VVC.

1.2.2 HEVC and VVC

A video coding standard is a document that outlines the structure and syntax of a bitstream and the decoder for video compression. It specifies the output format that an encoder should generate, rather than defining the encoder part, as shown in Figure 1.4. Typically, video coding standards contain a collection of tools that are used for compression. The principal coding standards include AVC, AOMedia Video 1 (AV1), VP9, HEVC, and VVC.

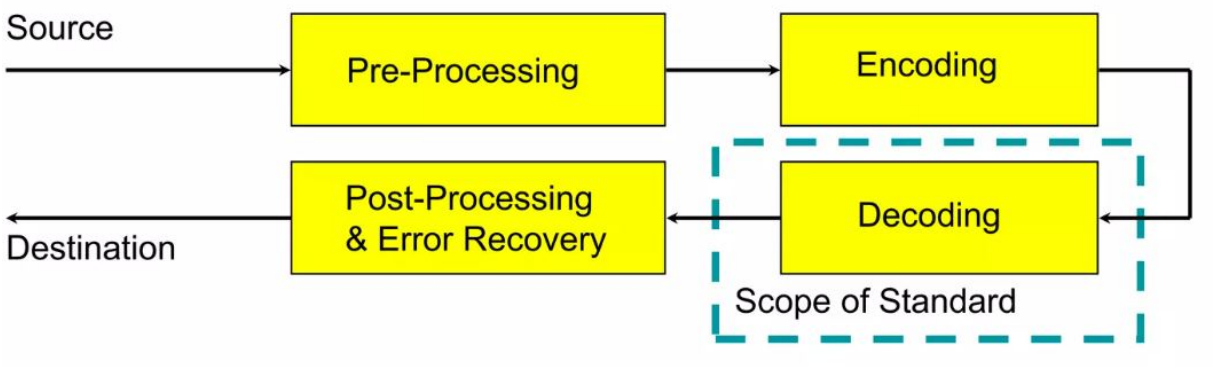


Figure 1.4 – Scope of Video Coding Standard[15]

1.2.2.1 HEVC

HEVC shares the basic coding structure as its predecessor, AVC. The differences are outlined in Table 1.1. These improvements could be summarized as follows.

— More flexible partitioning, from large to small partition sizes
— Greater flexibility in prediction modes and transform block sizes
— More complex interpolation and in-loop filters
— Enhanced prediction and signalling of prediction mode and motion vectors
— Features enabling efficient parallel coding

Table 1.1 – Comparison between HEVC and AVC

	AVC	HEVC
Motion Prediction	Motion vector prediction	Advanced motion vector prediction (spatial and temporal)
Intra Modes Number	9	35
Inter Modes Number	7	24
Prediction Unit Size	16x16 to 4x4	64x64 to 4x4
Transform Unit Size	8x8 to 4x4	32x32 to 4x4
Block size	16x16	64x64
Supported Resolution	Up to 4K	Up to 8K
Supported FPS	Up to 60FPS	Up to 300FPS
Bit-depth	8 bit	10 bit

1.2.2.2 VVC

A major difference when comparing VVC with HEVC is the newly adopted QuadTree with nested Multi-type Tree (QTMT) partitioning structure instead of the QT partitioning structure of HEVC. In addition to the partitioning structure, new intra/inter prediction modes, novel intra/inter coding tools, and improved transform and in-loop filters have been integrated into VVC to achieve higher coding efficiency. A detailed comparison is given in Table 1.2.

Table 1.2 – Comparison between VVC and HEVC

	HEVC	VVC
Partition structure	Quad Tree, max block size 64x64	QTMT, max block size 128x128
Intra Prediction	35 Intra Prediction Mode (IPM)s	67 IPMs, intra coding tools (*i.e.* Intra SubPartition (ISP) [16], Multi Reference Line (MRL) [17])
Inter Prediction	Skip mode, Merge mode	Skip mode, Merge mode, motion prediction tools (*i.e.* Affine Motion Estimation (AME) [18], Bi-Directional Optical Flow (BDOF) [19])
Transform	Square transform	Square and rectangular transform, Multiple Transform Set (MTS) [20]
In-loop filter	deblocking filter, Sample Adaptive Offset (SAO)	adaptive deblocking filters, SAO, Adaptive Loop Filter (ALF) [21]

1.2.2.2.1 VTM

VTM is the reference software for VVC codec. A reference software includes both encoder and decoder parts. It serves as a valuable tool for users of a video coding standard, facilitating the establishment and testing of conformance and interoperability. Additionally, it is helpful for educating users and showcasing the capabilities of the standard. Hence, this software is made available at [22] to support the study and implementation of VVC.

From VTM1 [23] in 2018 to VTM20 [24] in 2023, VTM versions have continuously evolved. Various new coding tools have been proposed and integrated into VTM. After the finalization of the VVC standard in 2020 with VTM10, JVET continued to make trivial improvements and bug fixes since then. Therefore, the differences in complexity and coding performance between various VTM versions are not negligible before VTM10.

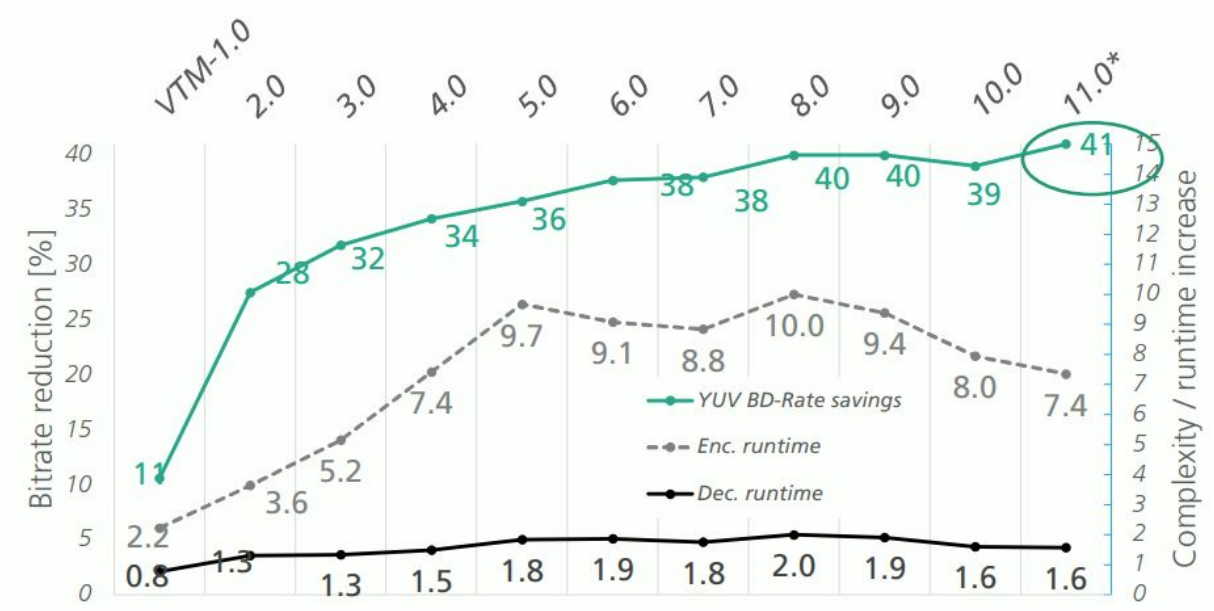


Figure 1.5 – VTM versions vs HM. [25]

We observed in 1.5 that the variation in encoding runtime varies substantially after VTM6, although its coding performance remains relatively stable. During the evolution of VTM, papers on VVC complexity reductions are consequently based on different versions of VTM, making the comparison of the state of the art a challenging task. In this thesis, our method is based on VTM10. We have performed several reimplementations of our methods in different versions of VTM used in state-of-the-art papers to ensure that the

comparison of results is within the same context and fair. The involved VTM versions for these reimplementations are VTM6, VTM8, and VTM11. In VTM10, we have also reproduced the result of a state-of-the-art approach originally implemented in VTM8. In the Annex, a statistical analysis is carried out on VTM15.

1.2.2.2.2 VVenc

The Versatile Video Encoder (VVenC), proposed by Fraunhofer, is a fast and efficient real-world VVC encoder implementation. The corresponding decoder implementation is Versatile Video Decoder (VVdeC). This encoder implementation has the following main features in comparison to the VTM encoder:

— Encoder implementation with slower, slow, medium, fast and faster as predefined quality/speed presets
— Perceptual optimization to improve subjective video quality
— Rate control supporting variable bit rate encoding at the frame level
— Fine-grained control of the encoding process by enhanced encoder interface

As observed in Figure 1.6, VVenC1.3.0 at the slower preset achieves the same coding performance as VTM14.2, while its speedup in terms of encoding time is 17 times faster. For the faster preset at the lowest quality, the speedup reaches about 180 times with a Bjontegaard Delta-Rate (BD-rate) gain of approximately 10% compared to HM-16.24. VVenC represents an immense step towards a real-time encoder. In Chapter 5, we have proposed a fast multi-rate encoding approach to further accelerate the VVenC encoder.

1.3 Video Coding of VVC

This section introduces the classical hybrid coding scheme utilized in VVC. Firstly, we present the general coding scheme for video codecs. As the main subject of this thesis, the partitioning scheme of VVC is then described and compared with previous codecs in Section 1.3.2. Subsequently, the intra coding and inter coding in VVC are briefly introduced. Following that, we present the RDO process to select the optimal combination of decisions for partitioning and inter/intra coding. Next, the temporal coding structure is depicted. Finally, the metrics used to evaluate coding quality are presented in the final part of this section.

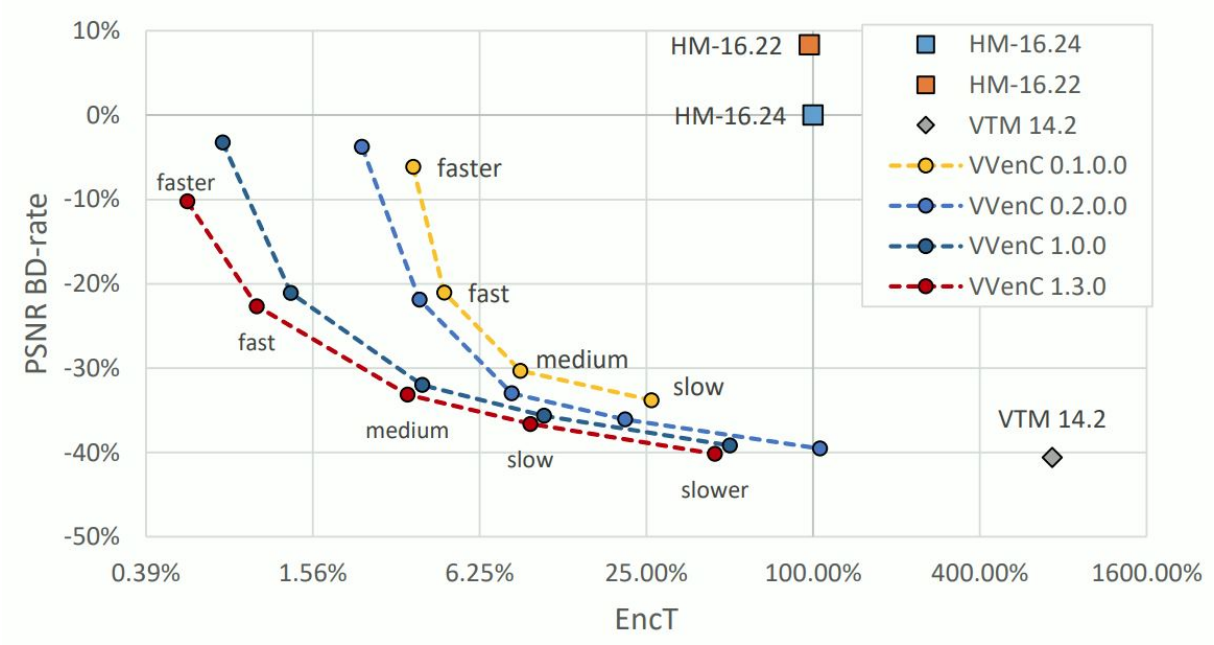


Figure 1.6 – Comparison of Runtime vs PSNR BD-rate.[26]

1.3.1 General Coding Scheme

Video codecs share the same fundamental coding scheme, as demonstrated in Figure 1.7, which can be summarized in five main steps.

1. Partitioning: Each frame of the input video is first divided into CTUs. Then each CTU is subdivided into multiple CUs. The subdivision of each CTU corresponds to a specific partition. During the RDO process, a large number of partitions are tested. The following steps are applied to each CU in various partitions. The process of partitioning is detailed in Section 1.3.2.

2. Prediction: Video coding incorporates two prediction techniques: inter prediction and intra prediction. Intra prediction utilizes neighboring pixels as references to exploit spatial redundancy in video signals. Conversely, inter prediction involves predicting pixels using blocks from other frames after motion estimation, known as motion compensation. Inter prediction leverages temporal redundancy. The residual values are computed by subtracting the original CU by the predicted CU. These values are then compressed in subsequent steps. The prediction process in VVC is presented in Section 1.3.3 and 1.3.4.

3. Transform and Quantization: Inside CUs, a robust correlation exists among neigh-

boring pixels, as well as among the residuals of these CUs. The process of transform consists of expressing the values in a novel base (*e.g.* Karhunen–Loève Transform (KLT), Discrete Sine Transform (DST), DCT [27]). The coefficients of the new base are the result of the transform. The primary goal of this transformation is to reduce redundancy in residual values and condense energy, resulting in more low-frequency values after the transform, thus facilitating quantization. VVC employs a variety of DST and DCT variations. Notably, the block dimension of the transform in VVC is more flexible, spanning from 4×4 to 128×128 block sizes, including rectangular blocks. Furthermore, VVC incorporates coding tools (namely MTS and Low Frequency Non-Separable Transform (LFNST) [28]) to optimize the transformation process.

Following the transform, the coefficients are quantified. Quantization involves dividing the coefficient by a quantization step and subsequently rounding the quotient. In VVC, dependent coding is used to exploit the redundancy of quantized values. Two possible quantized values can be coded with a single syntax element. From a perceptual standpoint, the quantization step represents a lossy compression technique, discarding negligible information.

4. Reconstruction: To restore the quantized coefficients to their original form, the process begins with an inverse quantization. Essentially, the coefficients are multiplied by the quantization step. The result is then inversely transformed to restore the original residuals. By adding these residuals to the prediction of the CU, we arrive at the reconstructed CU. In VVC, the in-loop filters consist of various components: Luma Mapping with Chroma Scaling (LMCS) [29], designed to enhance coding efficiency by altering the distribution of coded values; deblocking filters, which aim to mitigate blocking artifacts; SAO, intended to alleviate artifacts stemming from transform quantization; and ALFs, employed to refine the reconstructed values. These filters are applied sequentially to yield the decoded samples. These decoded images, stored in the buffer, also serve as reference points for inter prediction.

5. Entropy Coding: There exists statistical redundancy for syntax elements such as motion data, residual data, and data of coding modes. Therefore, we use the entropy coding for the syntax elements at the last step of the encoding process to reduce redundancy. As a lossless coding method, entropy coding improves the coding efficiency of the encoder. Context Adaptive Binary Arithmetic Coding (CABAC) [30] is integrated in VVC. After entropy coding, the encoded bitstream is produced.

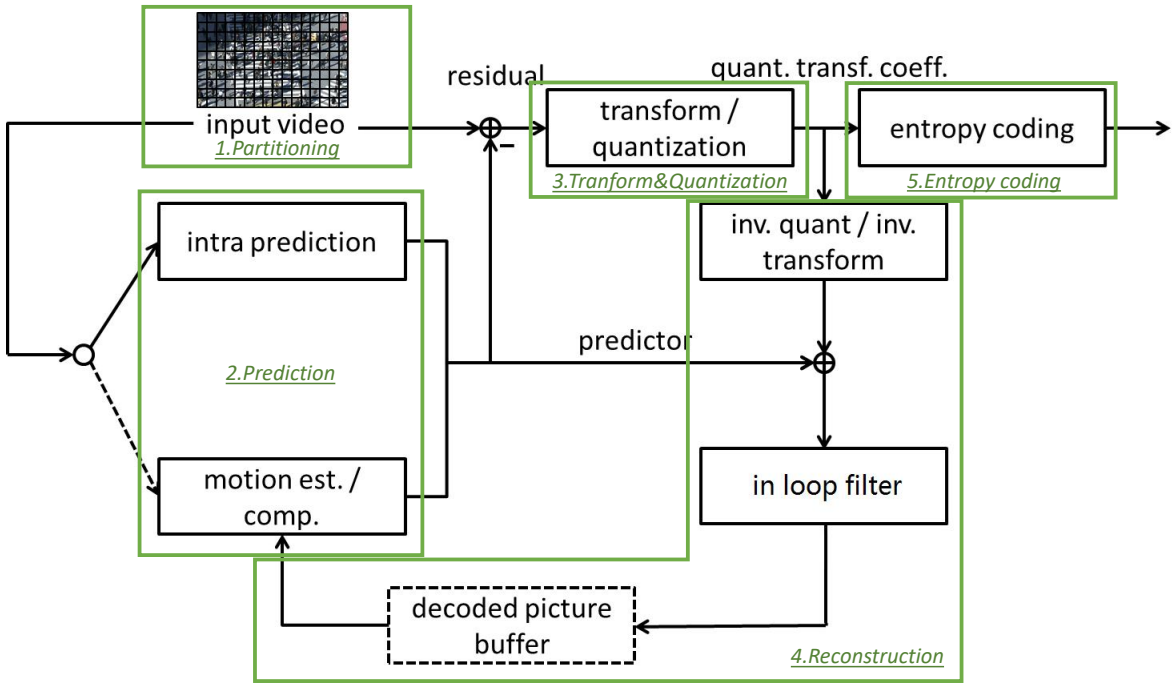


Figure 1.7 – Diagram of video coding system. [31]

1.3.2 Partitioning Scheme

The purpose of partitioning is to find the partition that suits the texture of the CTU region. If a region inside CTU is homogeneous or motionless compared to reference frames, it is highly advantageous to encode this region with a CU of the same shape and dimension. Partitioning involves searching among possible partitions for the most suitable one given the CTU texture at hand. The partitioning proceeds in a raster scan order on each frame at the CTU level, row by row, and from left to right. Inside CTU, the CUs of a partition are also processed in raster scan order, as demonstrated in Figure 1.8(a). The CU can be further split into Prediction Unit (PU)s for prediction. Additionally, the residual of the CU can be divided into Transform Unit (TU)s for applying different transforms to subpartitions of the CU. In the following section, the partitioning of AVC, HEVC, and VVC are separately presented.

1.3.2.1 Partitioning of previous codecs

In AVC, the term used for the equivalent of the CTU is MacroBlock (MB), and CU is referred to as a block. The size of a MB is fixed at 16×16. As illustrated in Figure 1.9, possible partitions differ between inter prediction and intra prediction. The inter partition

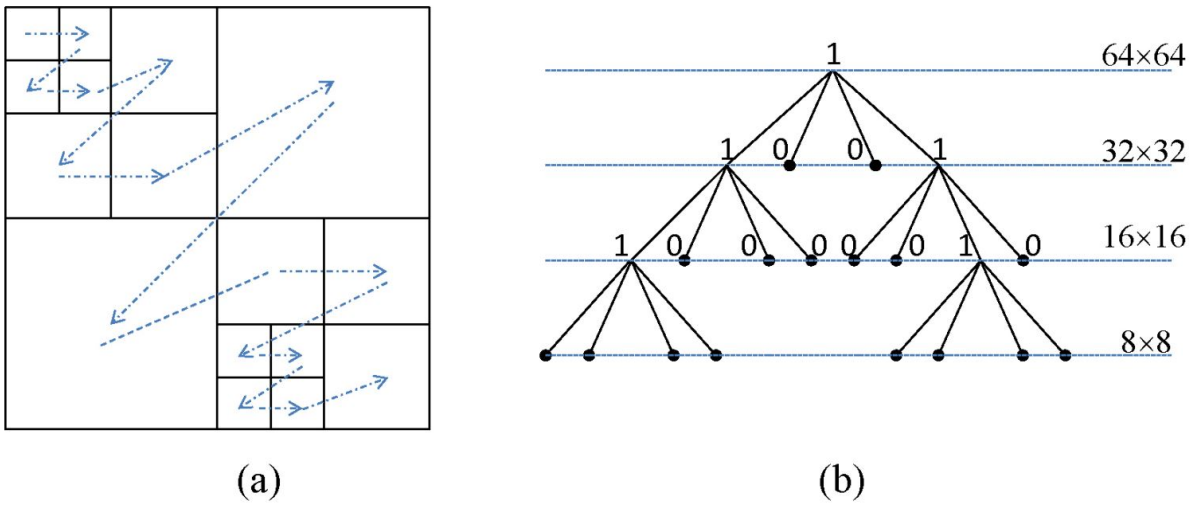


Figure 1.8 – Example of CTU Partition in HEVC [32] (a) CTU partition (b) Corresponding partition tree structure

adopts a more flexible scheme, where an MB can be predicted as one 16×16 block, two 16×8 or 8×16 blocks, or four 8×8 blocks. Each 8×8 sub-MB in MB can be divided vertically, horizontally, or into four 4×4 blocks. For intra prediction, a MB can be partitioned into 8×8 or 4×4 blocks. The possible sizes for the transform blocks are 4×4 and 8×8. While various non-square partitions exist, the block partitioning framework of AVC can be regarded as a three-level quadtree structure allowing blocks sizes from 4×4 to 16×16.

The partition in AVC has its limitation. The MB size of 16×16 is chosen to strike a balance between memory requirements and coding efficiency when AVC was standardized in the early 2000s. In recent years, with the emergence of high-resolution content, the MB size of 16×16 is insufficient to capture the increased spatial correlation in higher resolution content.

In HEVC, the CTU size is extended to 64×64 compared to the analogous term in AVC, enabling a more complex partition structure. Furthermore, the notions of CU, PU, and TU are introduced in HEVC. Each CU of size 2N×2N (where N is one of the values: 32, 16, or 8) can be divided into four smaller CUs, each with a size of N×N, which is defined as the QT split. By recursively applying the QT split to child CUs of the QT split, a QT partition tree is obtained. The maximum depth of the partition tree is 3 since the minimum size of the child node of the QT split is 8×8. Figure 1.8 illustrates an example of CTU partition in HEVC and its partition tree.

A CU can be split into one, two, or four PUs. HEVC defines two split types for intra-

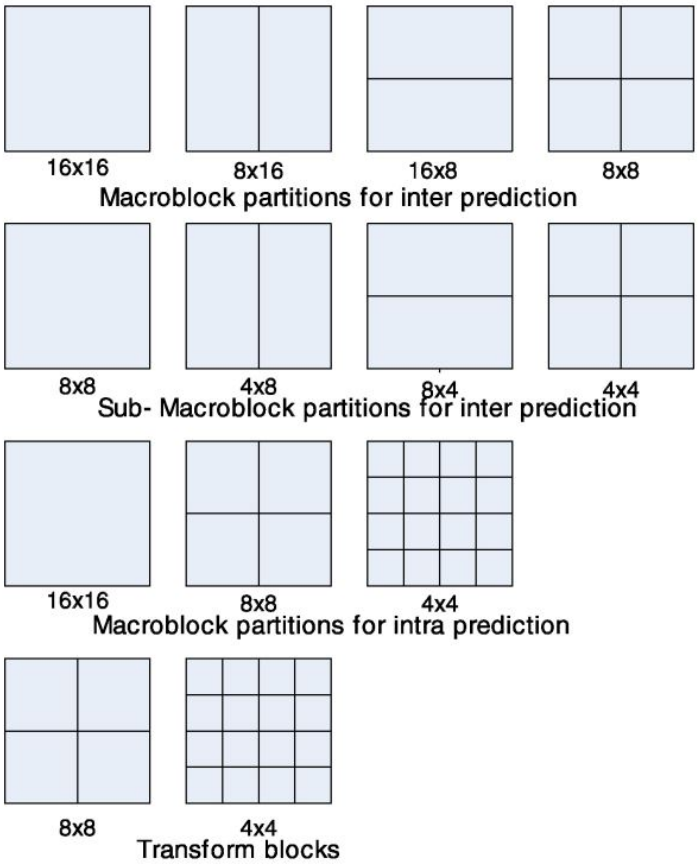


Figure 1.9 – Block Partitioning structure in AVC [32]

coded CUs and eight split types for inter-coded CUs, as shown in Figure 1.10. Unlike the CU, the PU may only be split once. Regarding the partition of TU, HEVC specifies four TU sizes: 4×4, 8×8, 16×16, and 32×32, for coding the prediction residual.

Compared to AVC, the QT partition structure in HEVC results in greater flexibility in terms of the sizes of coded blocks. With the partition of PUs, the predicted blocks could be rectangular in shape. Consequently, HEVC achieves better coding performance on high-resolution content at the cost of higher complexity, partly due to a more complex partitioning scheme.

1.3.2.2 Partitioning of VVC

In addition to the QT split of HEVC, VVC integrates the MT splits into the partitioning structure. Figure 1.11 demonstrates the adopted split types in VVC: Binary Tree (BT) split, including Vertical Binary Tree (VBT) split and Horizontal Binary Tree (HBT)

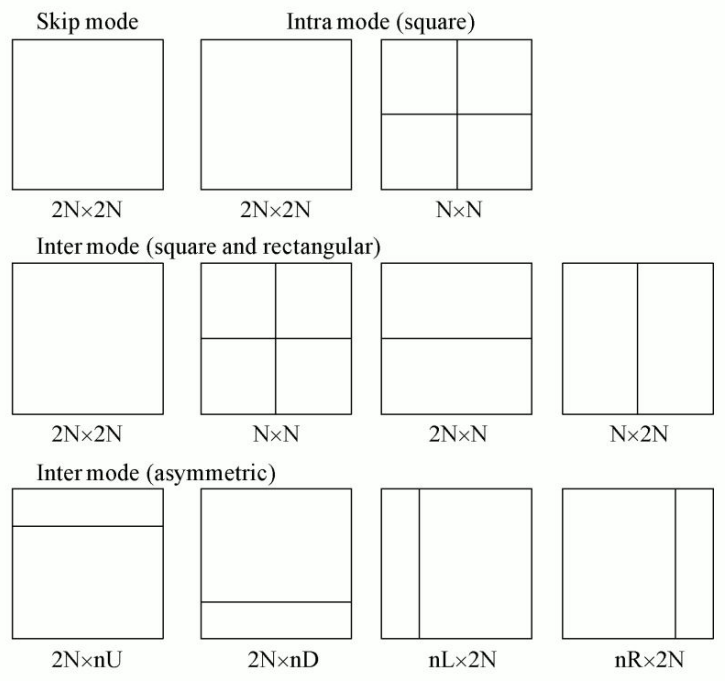


Figure 1.10 – Illustration of splitting CU to PUs in HEVC [32]

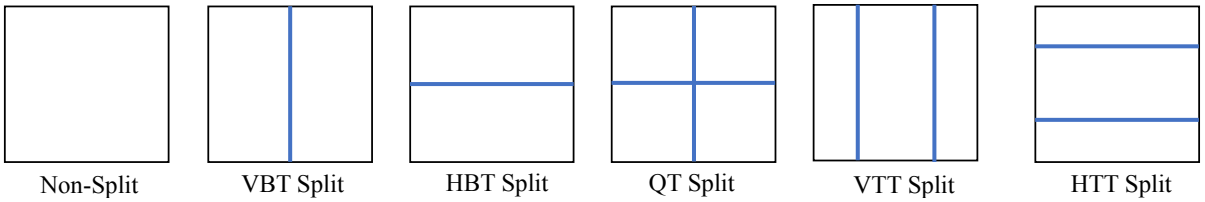
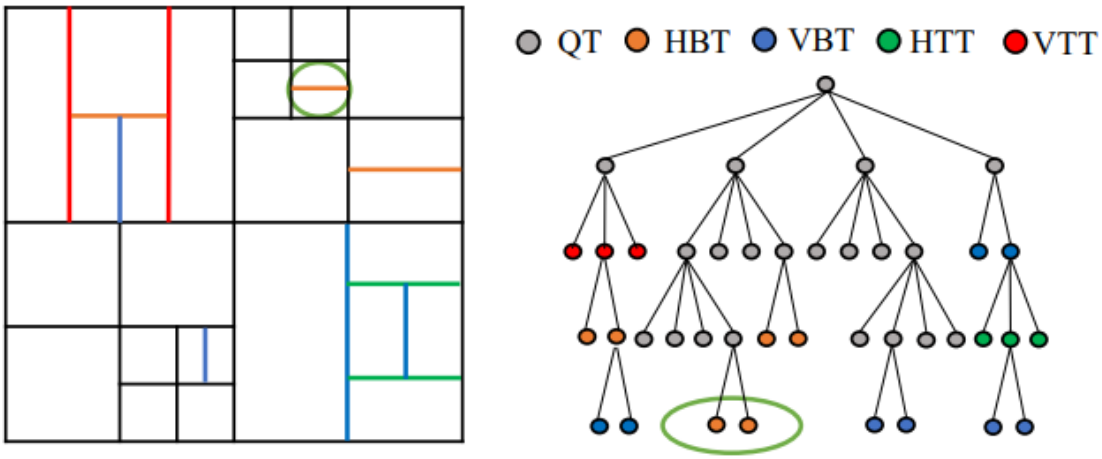


Figure 1.11 – Different split types in VVC

split, and Ternary-type Tree (TT) split, comprising Vertical Ternary Tree (VTT) split and Horizontal Ternary Tree (HTT) split. In the TT split case, CU is split in a ratio of 1:2:1 and the three children CU are a quarter, half, and quarter of the parent CU, respectively. The parent CU is horizontally or vertically divided into two CUs of equal size (*i.e.* half of the parent CU size). The novel partitioning structure in VVC is named QTMT. Figure 1.12 presents an example of QTMT partition and its partition tree. The encircled CUs of the partition in Figure 1.12(a) correspond to the encircled leaf nodes of the partition tree in Figure 1.12(b).

VVC employs a far more sophisticated partitioning structure than AVC and HEVC. There are 32 possible CU sizes for QTMT partition, including squared and rectangular CUs ranging from 128×128 to 4×4. A characteristic of QTMT partitioning is that the available split types depend on CU sizes. Figure 1.13 presents the number of available



(a) Partition of CTU (b) Partition tree

Figure 1.12 – Example of QTMT partition

split types including No Split (NS) per CU size for luma samples. The number of split options varies from 1 to 6 for different sizes of CU, making the partition at the CU level more complicated.

Another noteworthy characteristic of partitioning in VVC is that the CU, PU, and TU concepts are unified. More precisely, both prediction and transform are executed at the CU level without further splitting the current CU. The concepts of TU and PU are discarded in VVC. Consequently, the rectangular transform is available in VVC, which is an important improvement compared to AVC and HEVC.

Moreover, VVC allows for a CTU dual partition tree [33] for the luma component and the chroma component for intra-coded frames, suggesting that the partitioning and final partitions of luma and chroma are separated and distinct. For inter-coded frames, the luma partition is applied to chroma.

The above new features of partitioning in VVC comply with the increasing demand for coding at high resolutions (*e.g.*, UHD, QHD, FHD). At the same time, the coding complexity of VVC in terms of encoding time increases significantly, partly due to this novel partitioning structure.

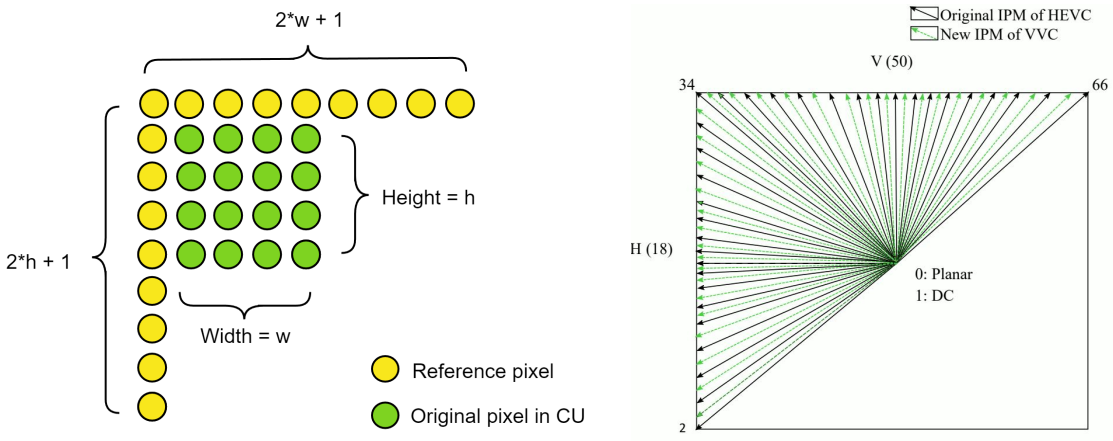
1.3.3 Intra Coding in VVC

Intra coding is used to reduce spatial redundancy within a frame by utilizing reference pixels. It operates at the CU level, referring to neighbouring pixels as demonstrated in Figure 1.14(a). Since encoding is processed in raster scan order, the reconstructed pixels

<table>
<tr><td></td><td colspan="7">Width</td></tr>
<tr><td rowspan="8">Height</td><td></td><td>128</td><td>64</td><td>32</td><td>16</td><td>8</td><td>4</td></tr>
<tr><td>128</td><td>4</td><td colspan="3">3</td><td colspan="2">-</td></tr>
<tr><td>64</td><td rowspan="3">3</td><td>6</td><td colspan="2">5</td><td rowspan="3">4</td><td rowspan="3">3</td></tr>
<tr><td>32</td><td rowspan="2">5</td><td>6</td><td>5</td></tr>
<tr><td>16</td><td>5</td><td>6</td></tr>
<tr><td>8</td><td rowspan="2">-</td><td colspan="3">4</td><td>3</td><td>2</td></tr>
<tr><td>4</td><td colspan="3">3</td><td>2</td><td>1</td></tr>
</table>

Figure 1.13 – Number of split types per CU in VVC for Luma

on the top row and left column of the CU are frequently available and can be used as reference pixels. Based on IPM, interpolations are performed on the reference pixels to predict the original pixels in CU. There are a total 67 IPMs available in VVC, an increase from the 33 modes in HEVC. These IPMs include 65 directional IPM, a DC mode, and a planar mode, as presented in Figure 1.14(b). Each directional IPM is associated with the propagation of neighboring reference samples at a specific angle, ranging from 45 degrees to -135 degrees. The DC mode predicts the current CU by calculating the average of the reference pixels, while the planar mode uses bilinear interpolation of reference pixels for prediction.



(a) CU and reference pixels (b) Intra Prediction Modes [34]

Figure 1.14 – Intra prediction of CU

Several intra coding tools are introduced in VVC. The ISP mode divides a luma intra-predicted block into 2 or 4 smaller subpartitions along one dimension, all of which are predicted using the same intra mode. Figure 1.15 illustrates the four possible lines of reference pixels introduced by the MRL coding mode. In addition, the Matrix-based Intra Prediction (MIP) modes [35] are used in VVC, which generates predictions using a weighted sum of neighboring downsampled reference samples. These weights are determined through machine learning methods.

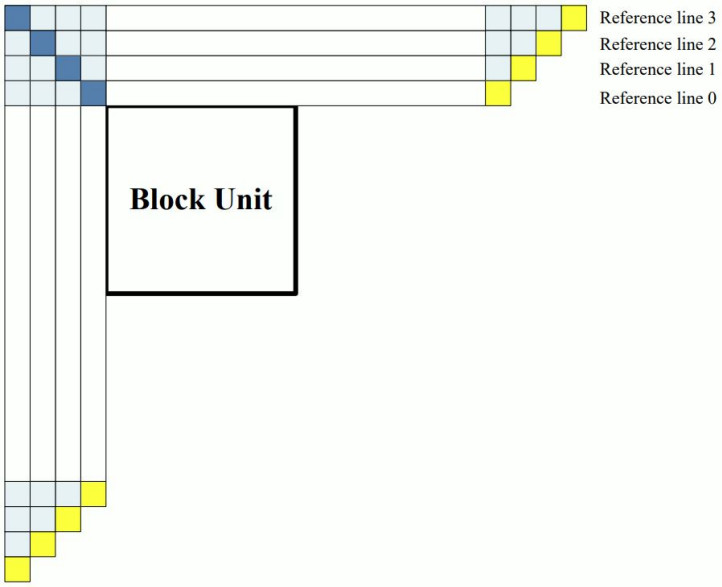


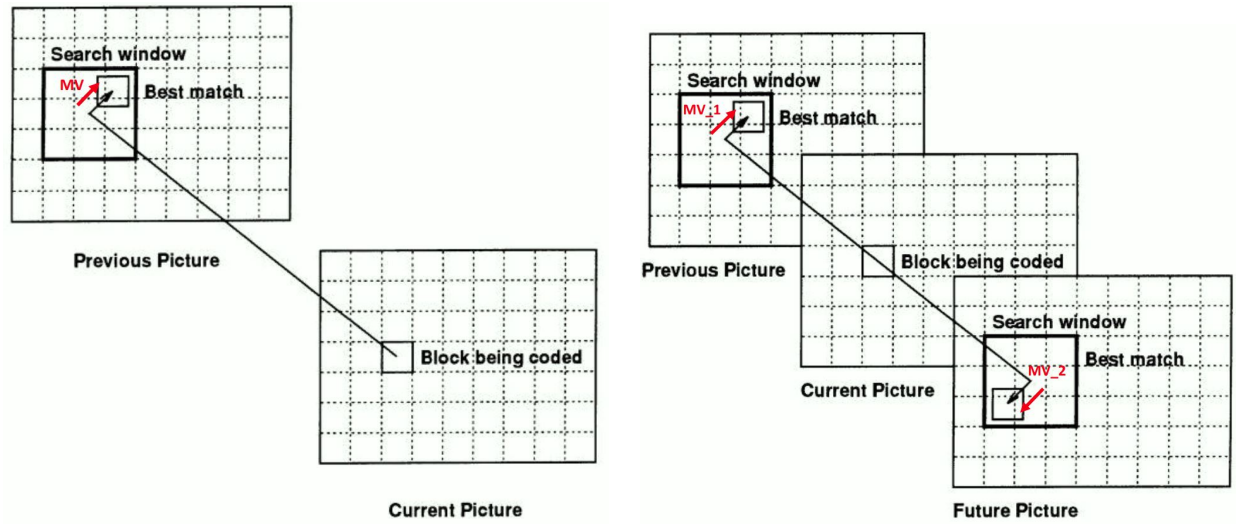
Figure 1.15 – Multiple Reference Line [31]

1.3.4 Inter Coding in VVC

To minimize temporal redundancy between multiple frames, inter coding involves predicting the current CU from reference frames, which are previously reconstructed frames stored in the decoded frames buffer. Inter prediction is achieved by identifying the most « similar » CU (*i.e.* reference CU) in the reference frame. The pixels of the reference CU are used to predict the current CU. We use the term Motion Vector (MV), which consists of horizontal and vertical offsets, to describe the coordinate shift between current CU and reference CU inside the frame. Examples of MV are provided in Figure 1.16. In the following part, we first present the motion estimation process, which is related to our work in Chapters 3 and 4. Then, we depict the inter coding modes involved in the statistical analysis in Annex.

1.3.4.1 Motion Estimation

Motion estimation is the process of determining motion vectors that point to the best reference CU. The search for motion vectors occurs within a search window, as shown in Figure 1.16.



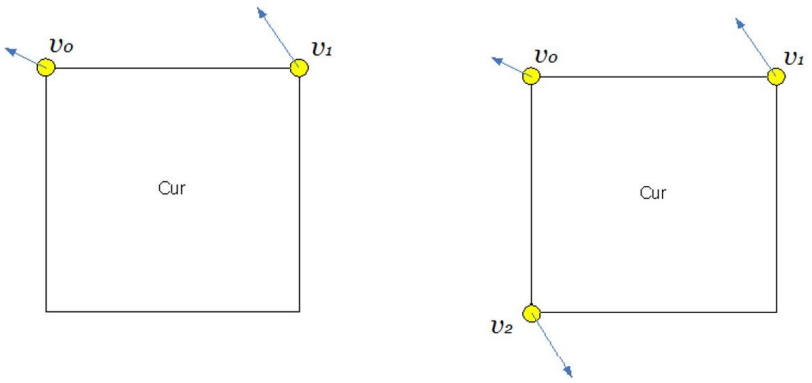
(a) Motion estimation for uni-directional prediction

(b) Motion estimation for bi-directional prediction

Figure 1.16 – Motion estimation [36]

There are two types of inter prediction: uni-directional prediction and bi-directional prediction. Figure 1.16 illustrates these two types. The motion estimation for unidirectional prediction is based on one reference CU from one previous reference frame, whereas two MVs are needed for bidirectional prediction. One of the reference frames comes from previous frames before the current frame, and the other is from frames after the current frame in terms of display order. For bidirectional prediction, the final prediction in VVC is generally calculated as the weighted sum of two predictions from two directions.

Besides the aforementioned traditional translational MV, the affine motion model is integrated to efficiently represent complex motions such as rotation, resizing, and shearing. In VVC, the affine motion model takes 4 or 6 parameters defined by 2 or 3 motion vectors, as illustrated in Figure 1.17. The CU is then divided into sub-blocks, and the motion vector associated with each sub-block is calculated using the parameters of the affine model.



(a) 4 parameter affine model (b) 6 parameter affine model

Figure 1.17 – Affine motion model [31]

1.3.4.2 Inter Coding Modes

After the motion estimation, motion compensation is executed. This represents the process of producing residuals of a CU by subtracting the inter prediction of the CU from the original CU. Subsequently, the residuals and motion vector information need to be transmitted. Depending on the transmission of MV and residuals, three coding modes are available: Advanced Motion Vector Prediction (AMVP) [37] mode, Merge mode, and Skip mode. For the sake of simplification, we refer to AMVP as the Reg mode for the rest of the document.

Before transmitting MVs, a candidate list of MVs is constructed based on the spatial and temporal neighboring CUs by exploiting the correlations of MVs between them. Four types of inter prediction data can be signaled, including the reference frame index (*i.e.* Ref Frame Idx), the candidate index of the best MV (*i.e.* MV Cand Idx), the difference between the best candidate and the MV determined by motion estimation (*i.e.* Motion Vector Difference (MVD)), and residuals. Table 1.3 presents the signaled data types for Reg, Merge and Skip.

Table 1.3 – Data types to transmit for motion data coding

	Ref Frame Idx	MV Cand Idx	MVD	Residuals
Reg	✓	✓	✓	✓
Merge	X	✓	X	✓
Skip	X	✓	X	X

In addition to the Affine mode and the Intra mode, two novel coding modes are available in VVC. For CUs coded in merge mode, Combined Intra-Inter Prediction (CIIP) [38]

combines the inter prediction and the intra prediction to form a final prediction. The Geometric Partitioning Mode, denoted Geo, is designed to better predict moving objects in video. In Figure 1.18, CU is split into two parts by a straight partitioning boundary. Uni-directional prediction is separately conducted on each part of the Geo partition. Additionally, Geo is conventionally coded with Merge mode.

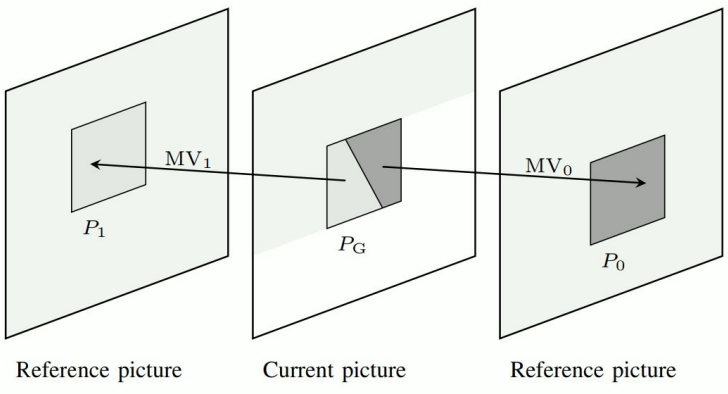


Figure 1.18 – Geometric Partitioning Mode [39]

For inter coding configuration, coding modes of Reg, Merge and Skip could combine with Affine, CIIP, and Geo, which results in a total of 10 coding modes: *Intra*, *Reg*, *Merge*, *Skip*, *Affine*, *AffineMerge*, *AffineSkip*, *GeoMerge*, *GeoSkip*, *CiipMerge*. Statistics of these coding modes are further collected and analyzed in Annex.

1.3.5 RDO

During the encoding process, there are many decisions for the encoder to make. These decisions include:

— the best partition for each CTU
— the best coding modes for each CU of the partition
— the best setting of coding tools for each coding mode
— ...

An immense search space is constructed based on the combinations of these decisions, represented by the coding parameter set. The RDO process consists of finding the optimal set of coding parameters by minimizing the Rate-Distortion cost (RD-cost) J in the search space. The RD-cost is defined as follows:

$$J(s) = D(s) + \lambda * R(s) \tag{1.2}$$

where the D, R, and s represent the distortion, bitrate and coding parameter set, respectively. λ is the Lagrange multiplier. RD-cost measures the cost considering not only the coding loss but also the bitrate consumed.

The trade-off between the coding loss and the bitrate is regulated by λ, which is controlled by the encoding parameter Quantization Parameter (QP). The relation between λ and QP is expressed in Equation 1.3.

$$\lambda = 0.85 \times 2^{(QP-12)/3} \tag{1.3}$$

With larger QP values, a larger weight is placed on the bitrate. Therefore, sets of coding parameters with low bitrates are more likely to be selected in the RDO process. The video is thus encoded at a low bitrate. In [40], the encoding of each sequence is executed at QP 22, 27, 32, 37.

In VTM, the distortion is measured by Sum of Absolute Differences (SAD) or Sum of Absolute Transform Differences (SATD) of residuals. By encoding the syntax elements produced by the coding parameter set with CABAC, its bitrate is obtained. During the RDO process, the encoder performs a nearly exhaustive search in the search space by computing the RD-cost for each set of coding parameters s. Although the search space is previously constrained, the RDO process is computationally costly. The core of fast encoding methods/algorithms is to accelerate the RDO process by removing less probable sets of encoding parameters in the search space. As a result, fast methods based on partitioning are effective approaches to speed up RDO, which are addressed in Sections 3, 4 and 5.

1.3.6 Temporal Coding Structure

The previous part of this chapter gives an overview of how coding is processed at the CU level. In this section, we present the temporal coding structure at the frame level. Firstly, frame types used in video coding are introduced. Secondly, we provide details of the GOP structure, composed of multiple frames with different frame types. Finally, the coding configuration that defines the frequently used GOP structure is detailed.

1.3.6.1 Slice and Frame Types

In video coding, frames are divided into slices, allowing concurrent parallel encoding of each slice. In this thesis, parallel encoding is not involved. Thus, we assume that the

frame contains only one single slice. The term *slice* is therefore replaced by *frame*.

Video frames are encoded using different algorithms that correspond to different levels of data compression. These algorithms for video frames are defined as frame types. As shown in Figure 1.19, there are three major frame types: Intra-coded picture (*i.e.* I-frame), Predicted picture (*i.e.* P-frame), and Bidirectional predicted picture (*i.e.* B-frame).

- ***I-frame***:
  I-frame is independently coded without reference frames. Only intra prediction is allowed during the coding of I-frame, which indicates that only spatial redundancy is exploited. Compared to other frame types, I-frame is therefore the least compressed frame type.
- ***P-frame***:
  The coding of P-frame depends on the previous frame in coding order. Both intra prediction and inter prediction are used in the coding of the P-frame. The P-frame is more compressed than the I-frame.
- ***B-frame***:
  The B-frame can use both temporal backward (previous) and forward (future) frames as reference frames. The B-frame is the most compressed among the three types. Nevertheless, the coding of B-frames is the most computationally expensive.

1.3.6.2 GOP Structure

A GOP defines a group of successive frames containing I-frames, B-frames and P-frames. In video compression, each video is coded with successive GOPs, each of which is independently encoded and decoded. When compressing the video with a GOP structure, the encoding/decoding order of frames can be different from the display order of frames (*i.e.* Picture Order Count (POC)).

A GOP structure generally starts with an I frame, which is followed by several B-frames and P-frames. The distance between two I-frames is equal to the GOP size or a multiple of the GOP size. Figure 1.19 presents an example of a GOP structure of size 12. Generally, the GOP size has a great impact on the quality of the coded video. Coding with a larger GOP size results in better quality at a given bitrate. In the following sections, the GOP size is set to 32 for the evaluation of methods.

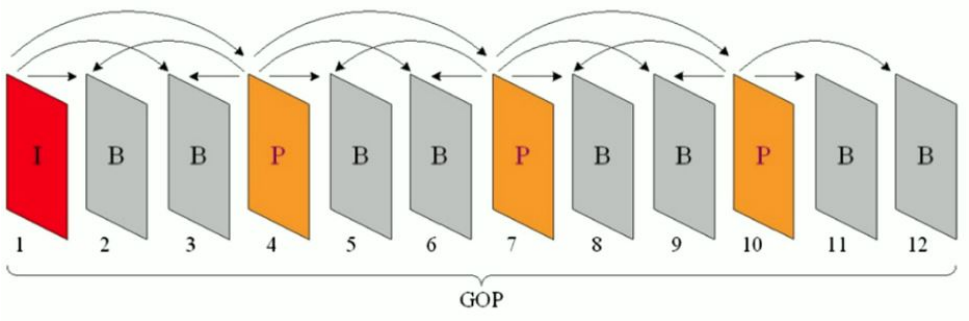


Figure 1.19 – Example of GOP structure [41]

1.3.6.3 Coding Configuration

Coding configuration specifies a set of coding parameters such as the GOP structure, coding tool configuration, motion estimation configuration, *etc.* Three example coding configurations are proposed by Common Test Condition (CTC) [40], namely the All-Intra (AI) configuration, Low-Delay (LD) configuration, and RA configuration. These coding configurations are used for different application scenarios.

- ***AI configuration***:

  For the AI configuration, all frames are separately intra-coded. Since there is no dependence between coded frames, GOP is not utilized in the AI configuration. Therefore, the frames are encoded in POC order. The coding of the I-frame consumes much more bitrate than other frame types. Thus, the AI configuration is rarely deployed in scenarios such as streaming.

- ***LD configuration***:

  To improve coding efficiency, both B-frames and I-frames are incorporated into the GOP of the LD configuration. To achieve lower latency for video streaming, a smaller GOP size and a simpler frame dependency compared to the RA configuration are used in the LD configuration. In VTM, there are two standard LD configurations: Low-Delay B (LDB) and Low-Delay P (LDP). The LDB configuration includes I-frames and B-frames, while the LDP configuration contains I-frames and P-frames. Both of these configurations have a GOP size of 8. Figure 1.20 shows the GOP structure of LDP in VTM. The number inside the box represents the POC of the frame. LD configurations are commonly employed in video conferencing due to their superior coding efficiency compared to the AI configuration and their lower

delay compared to the RA configuration.

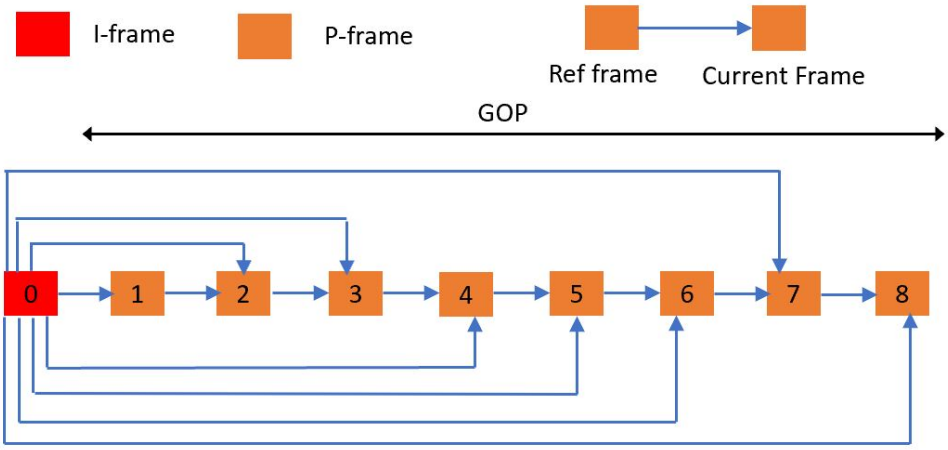


Figure 1.20 – LDP configuration of VTM

■ ***RA configuration***:

RA represents the ability to access and decode a specific frame or a portion of video without decoding all previous frames. RA configurations include I-frames and B-frames. I-frames serve as random access points where the RA functionality is available. A hierarchical GOP structure is adopted in the RA configuration with the introduction of Temporal Layer (TL). Frames at lower TLs are first encoded and consequently more referenced by future frames at higher TLs. In VTM, the two default RA configurations are RAGOP16 and RAGOP32. The GOP sizes for RAGOP16 and RAGOP32 are 16 and 32, respectively. The intra period for both configurations is 32, indicating that a random access point is set for every 32 frames. The number of TLs for these two configurations is 5. Figure 1.21 illustrates the GOP structure of RAGOP16.
RA configuration achieves better coding performance compared to the AI and LD configurations. Consequently, it is more widely employed in over-the-top and broadcasting services. However, this efficiency gain is accompanied by an increase in encoding complexity. Consequently, we focus on analyzing and reducing the complexity of encoding with the RA configuration. RAGOP32 is the configuration used in the Chapter 3, 4, and 5.

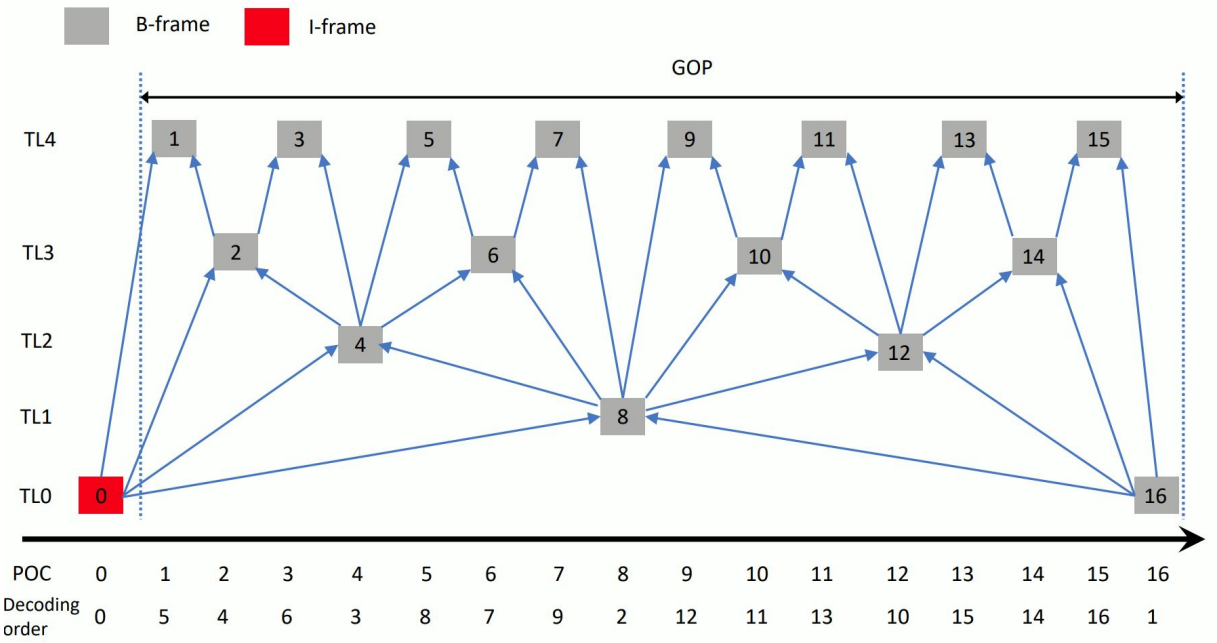


Figure 1.21 – Random Access of GOP size 16

1.3.7 Quality Evaluation Metrics

In this thesis, we utilize the objective metric BD-rate and the subjective metric Video Multi-method Assessment Fusion (VMAF) to assess the encoding quality of video sequences. In Chapter 3 and 4, we refer to BD-rate as a quality metric. In Chapter 5, we compute BD-rate and VMAF to evaluate our method.

1.3.7.1 Bjontegaard Delta metric

The Bjontegaard Delta (BD) metric consists of two evaluation criteria: BD-rate and Bjøntegaard Delta PSNR (BD-PSNR). It serves as a metric to compare the overall coding performance of two distinct coding algorithms. These algorithms may belong to different video codecs or modified versions of the reference codec. Similar to RD-cost mentioned in Section 1.3.5, the BD metric is based on two metrics: Peak Signal-to-Noise Ratio (PSNR), measuring the loss of the reconstructed video signal, and the bitrate consumed by the transmission of the encoded video signal.

PSNR is derived from Mean Square Error (MSE). MSE represents the mean square error, which is the mean energy of the difference between the original image and the degraded image. It is defined as follows:

$$MSE = \frac{1}{m \times n} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} [I(i,j) - K(i,j)]^2 \tag{1.4}$$

where m, n, I, and K represent the width of the image, the height of the image, the original image and the degraded image, respectively. As the expression of PSNR is the logarithm of MSE, its unit is decibel (dB). The PSNR calculates the ratio of the energy of the peak signal to the average energy of the noise as:

$$PSNR = 10 \log_{10} \frac{MaxValue^2}{MSE} = 10 \log_{10} \frac{(2^{BitDepth} - 1)^2}{MSE} \tag{1.5}$$

The *MaxValue* here is the maximum pixel value. If the *BitDepth* is equal to 8, the *MaxValue* would be 255. A less degraded image has a smaller MSE value, thus a larger PSNR value. The other parameter involved in the BD metric is the bitrate, referring to the amount of data used by the encoded video file in units of time. Its unit is usually Kbps. The size of the encoded file can be expressed as the product of the bitrate and the video length in seconds.

The Rate Distortion (RD) curve is plotted with PSNR and bitrate. As demonstrated in Figure 1.22, each curve is interpolated at four points obtained by four encodings of different qualities. The figure on the left illustrates the calculation of BD-PSNR, where the PSNR difference is averaged over the green region denoted as $\triangle S$ within the bitrate range $\triangle R$. It signifies the change in PSNR at the same bitrate. Similarly, for BD-rate, the average bitrate difference in percentage is calculated in the region $\triangle S'$, which represents the mean gain/loss in terms of bitrate at the same PSNR quality.

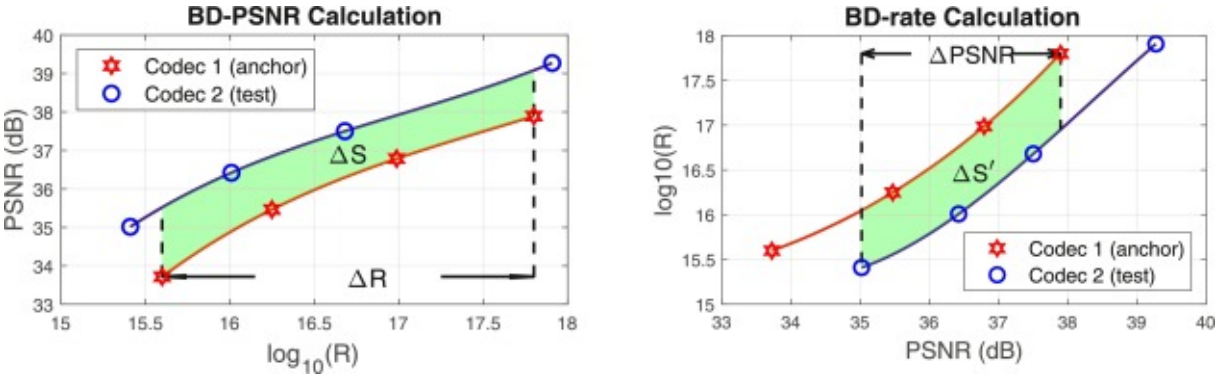


Figure 1.22 – RD curves for BD metric

In this thesis, we use BD-rate for the result evaluation procedure defined in the CTC standard. The four points of the RD curve are encodings at QP 22, 27, 32, 37. A negative

BD-rate value represents that the coding performance is improved. Conversely, a positive BD-rate value indicates that the coding performance is degraded. In Chapter 3 and 4, we intend to accelerate the VTM encoder with our machine learning-based approach. Therefore, the degradation of encoding is unavoidable. In this case, a slight positive BD-rate value is expected, which suggests that the acceleration approaches proposed in this thesis have largely accelerated the encoding with negligible coding efficiency loss.

1.3.7.2 VMAF

VMAF [42], developed by Netflix, is a perceptual video quality metric that integrates human vision modeling with machine learning techniques. More precisely, it individually assesses multiple elementary metrics and combine them into a final VMAF score with the help of a trained Support Vector Machine (SVM). VMAF includes the evaluation of 38 features, encompassing visual defects, detail losses, temporal differences between frames, *etc.*

VMAF is proven to be highly correlated with subjective video quality evaluations and has been widely used in various video quality assessment tasks. In Chapter 5, we use VMAF to evaluate our multi-rate approach, as perceptual quality in coding is crucial in real-time scenarios such as steaming.

Chapter 2

Fast Partitioning and Multi-Rate Encoding: Related Work

In this chapter, the state of the art of research fields related to this thesis are presented. In the first section, we summarize the related work for Chapter 3 and 4. The first section includes the fast partitioning methods for various codecs. The second section is dedicated to existing multi-rate coding approaches. The last section contains related work of our contribution in Annex, listing previous analysis of complexity of VVC.

2.1 Fast Partitioning Methods

In this section, fast partitioning methods for previous codecs and VVC are introduced. In general, these methods can be categorized into two groups: machine learning-driven methods and heuristic methods. From another perspective, we can classify these methods into two types: inter fast partitioning methods and intra fast partitioning methods.

2.1.1 Partitioning Acceleration for Previous Standards

There is a handful of acceleration methods dedicated to the complexity reduction of previous standards such as HEVC [43], VP9 [44], etc. The HEVC standard deploys a quadtree structure, which means that the partition of the image into CTUs could be depicted as a QT depth map. In [45], Aolin *et al.* firstly propose the depth map representation of HEVC block partitions and develop a CNN based fast partitioning method. From another perspective, Chen *et al.* in [46] rely on a low-complexity CNN to predict the split decision at the CU level such that redundant RDO calculations are omitted. Additionally, Shen *et al.* in [47] train separately SVM classifiers on all CU sizes to decide if the partition of CU should be early terminated. Others, such as in [48], [49], utilize decision trees to early terminate the partition search.

Apart from the aforementioned machine learning-driven methods, there are some heuristic methods [15]-[18] that use intermediate information during the encoding process for early termination. More precisely, the authors in [50] apply edge filters on luminance samples of CU to determine if the current CU should be split or not. Correa *et al.* [51] propose an acceleration method by exploiting the correlation of QT depth of co-located CTU of adjacent frames. Another QT depth related approach is developed in [52]. Based on the homogeneity of the texture, adaptive thresholds are applied on QT depth of a CTU to achieve early termination. In [53], Chiang proposes a threshold decision scheme based on SATD to early terminate the QT partitioning. VP9 developed by Google is a competitor to HEVC. It exploits the possibility of QT split and BT split at each level of partition search. In [54], Paul *et al.* present a multi-level merge map as a novel representation of the quadtree plus binary tree (QTBT) partition. They design a hierarchical CNN to predict the merge decision map at each level and utilize the predicted maps from the bottom up to accelerate partition search.

2.1.2 Partitioning Acceleration for VVC

Studies concerning partition search speed-up of VVC can be divided into two categories: heuristic approaches and machine learning ones. Some heuristic methods make full use of pixel-related statistics, such as gradient, variance, etc. Fan *et al.* [55] developed a fast hybrid QTMT partitioning algorithm based on early termination at a CU level based on pixel variance. They also propose a early termination of the MT partitioning based on gradients and a partition selection method based on variance. In [56], an early termination algorithm based on directional gradients is used to accelerate the intra partitioning in VVC. The method skips the RDO checks of unnecessary split types. Other heuristic methods employ coding information to simplify the partitioning process. Saldanha *et al.* [57] designed strategies to avoid checking MT splits according to the variance and the best ISP [58] mode chosen for the current CU.

Numerous machine learning methods have also been proposed to speed up the RDO search of partitioning in VVC. For both intra and inter partition acceleration, these methods could be further categorized into two classes: CNN-based approaches and Decision Tree (DT)-based approaches.

For intra partition acceleration, Galpin *et al.* [59] firstly utilize a CNN model to predict boundaries of all 4x4 sub-blocks inside a CTU for an experimental software of JVET - Joint Exploration Model (JEM). In [60], Tissier *et al.* adopt this method in

VVC. Another work [61] by Wu *et al.* consists of merging the boundaries of 4x4 blocks into split boundaries of different lengths per partition level. Then a Multi-Stage Exit CNN (MSE-CNN) [62] is proposed by Li *et al.* intending to determine the possible splits and early exit by inference of trained subnet for each CU of different sizes. In [63], Feng et al. propose a fast partitioning method by predicting a QT depth map, multiple MT depth maps, and multiple MT direction maps with a CNN. For each MT partitioning level, the MT depth map is utilized to determine the selection between the BT split and the TT split, with the assistance of the MT direction map guiding the choice between horizontal or vertical splitting. Regarding the DT-based approaches, Mário *et al.* [64] propose a fast block partitioning method in which various Light Gradient Boosting Machine (LGBM) classifiers are trained separately on different CU sizes to predict the possible splits.

Relatively less contributions for inter-partitioning have been proposed for VVC. For DT-based approaches, the work proposed by Amestoy *et al.* [65] uses a cascade of split-type skipping schemes consisting of different binary skip decisions predicted by trained Random Forest (RF)s. This method has been later improved by Kulupana *et al.* [66], by combining it with hand-crafted rule-based early termination for the TT splits. In the CNN-based category of methods, the work presented in [67] proposes a multi-branch CNN to perform a binary classification of the “Partition" or “Non-partition" at the CU level, in order to early terminate the partitioning. In this paper, different models are trained for different CU sizes. Moreover, Yeo *et al.* propose a method in [68], where a variant of Branch Convolutional Neural Network (B-CNN) [69], called multi-level tree CNN (MLT-CNN), is used to predict the partitioning mode of the CTU. The key feature of this CNN is that its outputs correspond to different decisions at different levels of a split mode tree. In [70], Tissier *et al.* use the MobileNetV2 structure fed with current CTU and reference CTUs to predict the boundaries of all 4x4 subblocks. Then LGBM models are trained to predict the possible splits for each CU sizes.

2.2 Multi-rate Coding Methods

In streaming applications, where video content is encoded at multiple bitrates, fast multi-rate encoding methods are employed. These methods typically designate a single representation as the reference representation, encoding it first, and utilizing its information to speed up the encoding process for the remaining representations, known as dependent representations. Schroeder *et al.* [71] proposed a multi-rate encoding approach

in which the video is first encoded at a high quality representation, and its information is reused to accelerate the encoding of lower quality representations. They leverage the fact that each CTU achieves its highest partitioning depth in the highest quality representation. Consequently, by determining the partitioning depth of CTUs in the highest quality representation (d_m), it is possible to skip the partitioning of co-located CTUs in lower quality representations beyond d_m. This leads to significant time savings during the encoding process. Çetinkaya *et al.* [72] proposed an approach in which the lower quality representation is initially encoded, and for each CTU with a partitioning depth of (d_m) in the reference representation, depths lower than d_m are skipped during the RDO search. In this method, CNNs is used to extract additional features, improving the performance of the partitioning depth decisions. Amirpour *et al.* [73] investigate the trade-off between time savings and quality drop by considering different representations as a reference representation. They concluded that utilizing a middle-quality representation can effectively strike a balance between these factors, optimizing the trade-off. Menon *et al.* [74] presented an extensive investigation of various multi-rate schemes and introduced innovative heuristics aimed at constraining the RDO process across different representations. Building upon these heuristics, they proposed three multi-encoding schemes that leverage encoder analysis sharing across various representations. These schemes were designed to optimize: *(i)* highest compression efficiency, *(ii)* the optimal trade-off between compression efficiency and encoding time savings and *(iii)* maximum encoding time savings.

2.3 Statistical Analysis of VVC

As VVC was finalized in 2020, there has been relatively less work done in the realm of statistical analysis compared to previous video codecs. Several articles have contributed to the statistical analysis of VVC. Toipwala *et al.* in [75] first compare the compression performance and subjective evaluation of VVC with HEVC, AV1 and Essential Video Coding (EVC). Then Tissier *et al.* explore in [8] the opportunities for complexity reduction of VVC intra-encoder. In their research, block partition, intra prediction mode and the selection of transform are considered. In [76], a detailed complexity analysis based on VVC intra prediction tools and CU sizes has been performed. Pakdaman *et al.* in [77] have broken down the encoding process into encoding modules such as motion estimation, intra prediction, entropy coding, etc. and then analyzed the complexity partition of mod-

ules in multiple encoding configurations. [78] reviews complexity aspects of the different modules of the VVC standard and provide a complexity breakdown of these modules in a more precise way. In [79], VVC and HEVC are compared in terms of rate-distortion and complexity analysis.

2.4 Conclusion

In this chapter, we have reviewed the state of the art in fast partitioning and multi-rate encoding of VVC and previous codecs.

Regarding fast partitioning methods for inter-coding in VVC, most machine learning methods are based on decision trees or random forests. Initially, these methods mainly focus on split decisions at the CU level, thereby disregarding the interconnections between sequential splits occurring at different hierarchical depths. Furthermore, the training and assessment of these models are vulnerable to the issue of overfitting, due to the fact that these models are trained on a selective subset of videos drawn from the test set. In response to these limitations, we introduce our CNN-based approach in Chapter 4. Our model is designed to conform to the partitioning structure in VVC while considering the interdependence between split decisions at different depths. As a result, our proposed method outperforms state-of-the-art methods.

Regarding multi-rate encoding, the existing literature pertains to HEVC. However, it's important to note that there is an absence of research in this area concerning the relatively recent codec, VVC. In Chapter 5, we develop a multi-rate encoding method based on fast partitioning. This marks a pioneering effort in the field of VVC multi-rate encoding.

Chapter 3

Light-weight CNN-based VVC Inter Partitioning Acceleration

3.1 Introduction

The novel QTMT partitioning, as detailed in Section 1.3.2.2, is the primary contributor to the increase in VVC codec complexity compared to previous codecs. As mentioned in Section 2.1.2, papers based on machine learning generally utilize CNN or RF to accelerate the partitioning process of VVC. In [45], the concept of the QT depth map is introduced and predicted by a CNN to efficiently accelerate the intra partitioning in HEVC. In this chapter, we propose a fast partitioning approach based on the QT depth map, specifically designed for inter partitioning in VVC. This domain has remained relatively underexplored when compared to its intra partitioning counterpart.

The contribution presented in this chapter was proposed after the standardization phase of VVC. As outlined in Section 1.2.2.2.1, our method relies on the VTM10, which is a relatively complete and stable implementation of VVC. Additionally, we have reimplemented our method in VTM6, VTM8, and VTM11 to facilitate comparisons with other state-of-the-art approaches.

In this chapter, we propose a lightweight CNN-based inter partitioning acceleration method for VVC. Given that the partitioning syntax of VVC can be specified as a series of QT-then-MT splits, the proposed method stores the QT depth that each 8×8 block has been associated. The QT depth grid is then learned by a designed CNN from a large dataset of encoded videos, and is integrated into the VTM with the help of a threshold to control the trade-off between complexity reduction and coding performance. Our method outperforms existing CNN-based approaches while achieving results comparable to those of RF-based methods.

The upcoming sections of this chapter are structured as follows:

— In Section 3.2, we elaborate on our approach, including the introduction of the

QT depth map representation, the partitioning search process in the VTM, the architecture and training for our CNN model, and the associated acceleration algorithm.

— Section 3.3 presents the results of our experiments and a comparative analysis of our method with state-of-the-art solutions.

— Section 3.4 provides an ablation study on input features of the CNN model.

— Finally, Section 3.5 concludes this chapter.

3.2 Proposed Method

To address the complexity issue of VVC, we propose a CNN-based scheme for predicting the split modes in inter-coding by skipping unnecessary RDO checks. RF- or CNN-based solutions have already been proposed for this problem. However, the potential of acceleration of these methods listed in Section 2.1.2 is limited. Indeed, in [67], the encoder can only save complexity on CUs classified as "No Partition" while the CNN proposed in [68] determines the split modes merely at a CTU level. With RF-based solutions in [65], 18 classifiers are involved. In contrast, the proposed solution is based on one unique model trained on complete CTUs which predicts a depth level on the partition tree. Skipping split modes at multiple CU levels allows us to save a higher number of intermediate RDO checks.

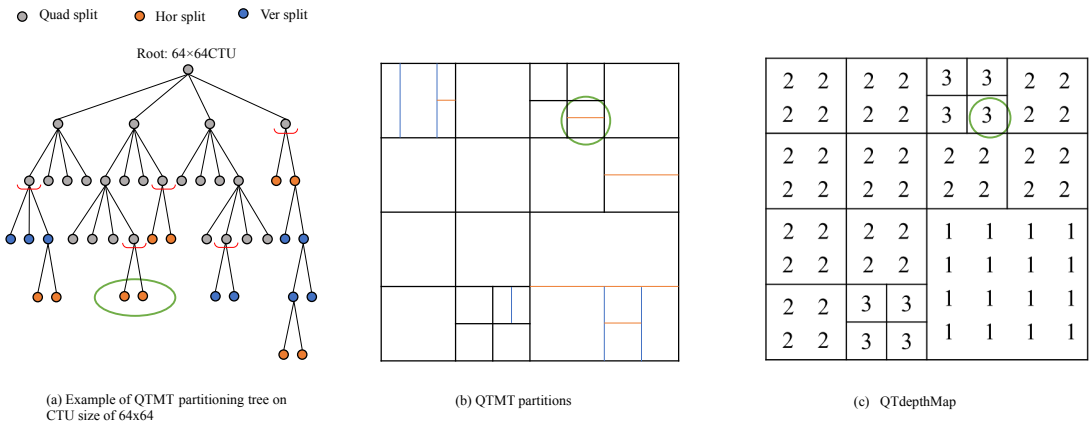


Figure 3.1 – Example of QTMT partitioning structure, partition and QTdepthMap for a 64×64 CTU

3.2.1 Quad-tree Depth Map Representation

Figure 3.1-(a) shows an example of QTMT partition tree, with its actual partition demonstrated in Figure 3.1-(b). Without loss of generality and for the sake of simplicity, we represent this example for CTU size 64×64, instead of 128×128. As mentioned earlier, once the first MT split is chosen, the QT split becomes forbidden for the following nodes (*i.e.* QT-then-MT scheme). In Figure 3.1-(a), the red horizontal arc-shaped curves indicate the border between the last QT split and the first MT split (if any).

Considering the QT-then-MT scheme in VVC, one can represent partitioning series for reaching any arbitrary CU, represented as leaf nodes in Figure 3.1-(a), as N consecutive QT splits, followed by M consecutive MT splits. As each QT split divides a CU into four smaller sub-CUss with exactly half of its size in width and height, it is in contrary with MT splits, where the size of sub-CUs highly depends on the MT split type that is selected. Therefore, in the proposed method, we use the values of N that are associated to leaf nodes (*i.e.* final partitions) and form a grid called Quad Tree depth Map (QTdepthMap). Figure 3.1-(c) shows how this map is computed for the above partitioning example. It is worth mentioning that as the example in this figure is simplified with a 64×64 CTU, QT depth values fall in $\{0, 1, 2, 3\}$, while in the default configuration of the VTM, a CTU of size 128×128 would result in QT depth values in $\{0, 1, 2, 3, 4\}$.

3.2.2 Baseline Partitioning Search in VTM

The proposed inter partitioning acceleration is designed on top of the existing algorithm of the VTM. This algorithm performs a nearly exhaustive search on possible ways of partitioning a CTU, except that it incorporates a handful of hand-crafted conditional shortcuts. Therefore, this work can be considered as an additional shortcut which relies on CNN.

Figure 3.2 illustrates a simplified tree representation showing all the split types checked during the partitioning process of a CTU. The green arrows indicate the path to a part of the final decision for the CTU, while the black arrows represent other paths that have been checked, but were not included in the final decision. Particularly, it can be observed that several RDO checks have been unnecessarily carried out at QT depths of 0, 1 and 2. The complexity of these unnecessary RDO checks can be reduced if the final QT depth can be accurately predicted. Preliminary tests show that the QTdepthMap representation is potentially an adequate solution for obtaining such estimation at the CTU level.

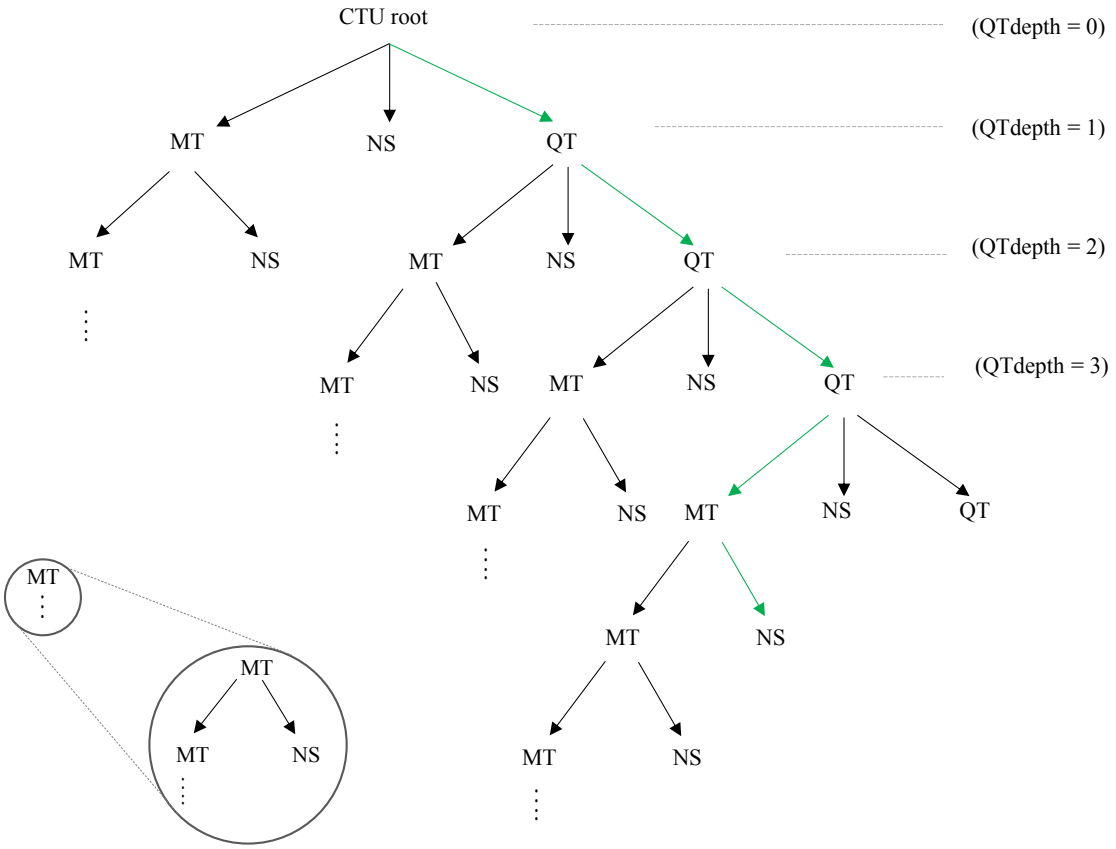


Figure 3.2 – Example of partition path with MT referring to VBT, HBT, HTT and VTT types.

3.2.3 Network Architecture for QT Depth Map Prediction

The CNN proposed in this contribution, called Multi-Branch Multi-Pooling CNN (MBMP-CNN), is depicted in Figure 3.3. This network structure fuses inputs of different types. Its multi-pooling layer is designed to extract features at different scales corresponding to CU dimensions at different QT depths.

Different from the traditional intra partitioning problem, features reflecting temporal correlation between frames are vital for predicting the inter partition. Consequently, a residual block of the current CTU is computed by motion-compensated prediction from the nearest frame and is fed into the network in addition to the luma component. In multiple papers [67, 68, 80], block residuals are fed into the CNN to give information on the similarity between the current and the reference block.

Another important feature used in [67] and [65] is the motion field which is composed

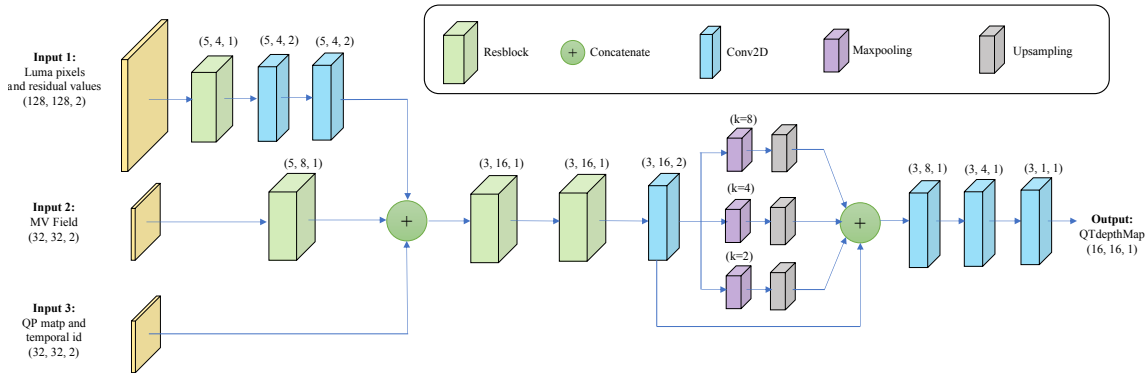


Figure 3.3 – Structure of proposed MBMP-CNN. The vector of three elements on top of Resblocks and Conv2D layers represents kernel size, number of filters and stride respectively. Value "k" denotes pooling size for Maxpooling layers.

by motion vectors calculated on each 4x4 sub-block in reference to the closest frame. As mentioned in [65], the motion field is correlated with the optimal partition. According to our test on the first 64 frames of all CTC sequences in RAGOP32 configuration, generating the motion field for each CTU takes on average only 0.1% of the encoding time of VTM10. Since the motion vector for each 4×4 grid contains a vertical motion value and a horizontal motion value, the motion field is of the shape 32×32×2.

The QP and temporal ID also impact the partitioning process. Generally, a larger temporal ID and a smaller QP result in a finer partition. For this branch, we have simply padded the QP value to 32×32 matrix and concatenated it with the similarly padded temporal ID matrix to form another input of shape 32×32×2.

Tensors of same width and height corresponding to multiple input branches are concatenated before being fed into the main branch of the network architecture.The main branch begins with three residual blocks [81] followed by the multi-pooling layer introduced in [45]. The different kernel sizes are consistent with the CU sizes at different QT depths. Finally, three convolutional layers shrink the tensor dimensions, which leads to the predicted QTdepthMap. The output of the CNN is composed of the QTdepths on different 8×8 sub-blocks. Each value equals to the QT depth of the CU containing the corresponding sub-block, as described earlier in Figure 3.1-(c). The inference within the VTM is performed by the CNN for each CTU before entering the RDO decision loop. By doing so, the encoder can use the acceleration algorithm presented in the next section to speed up the partition search.

3.2.4 Partitioning Acceleration Algorithm

Algorithm 1 MT and NS early skipping

Input: QTdepthMap; $\text{QTdepth}_{\text{cur}}$, CU; Th; Size_{CU}; Pos_{CU}, in CTU,
Output: SkipMT_NS: Boolean to decide whether to skip
MT and NS split types or not

1: Compute the $\text{QTdepth}_{\text{average}}$ based on Size_{CU}, Pos_{CU} and QTdepthMap
2: **if** $\text{QTdepth}_{\text{average}} \geq (\text{QTdepth}_{\text{cur}} + \text{Th})$ **then**
3: SkipMT_NS = True
4: **else**
5: SkipMT_NS = False
6: **end if**

We have merged the proposed QTdepthMap prediction with the original partition search process, with the goal of skipping the RDO of unnecessary MT and NS splits. The algorithm for early skipping is depicted in Algorithm 1 and the overall algorithm is described by the flowchart in Figure 4.6.

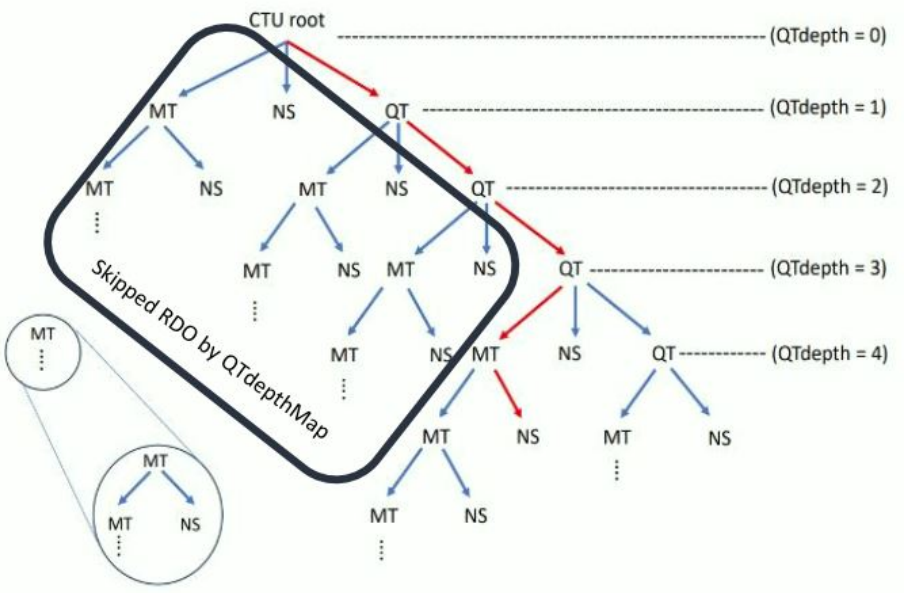


Figure 3.4 – Example of acceleration by QTdepthMap

The main idea of the proposed acceleration algorithm is to determine if we should evaluate the MT and NS split modes or not for the CU partitioning. The average of *QTdepth* values is computed for each CU based on predicted QTdepthMap before the RDO check of split modes. We are using the mean value for the current CU since there

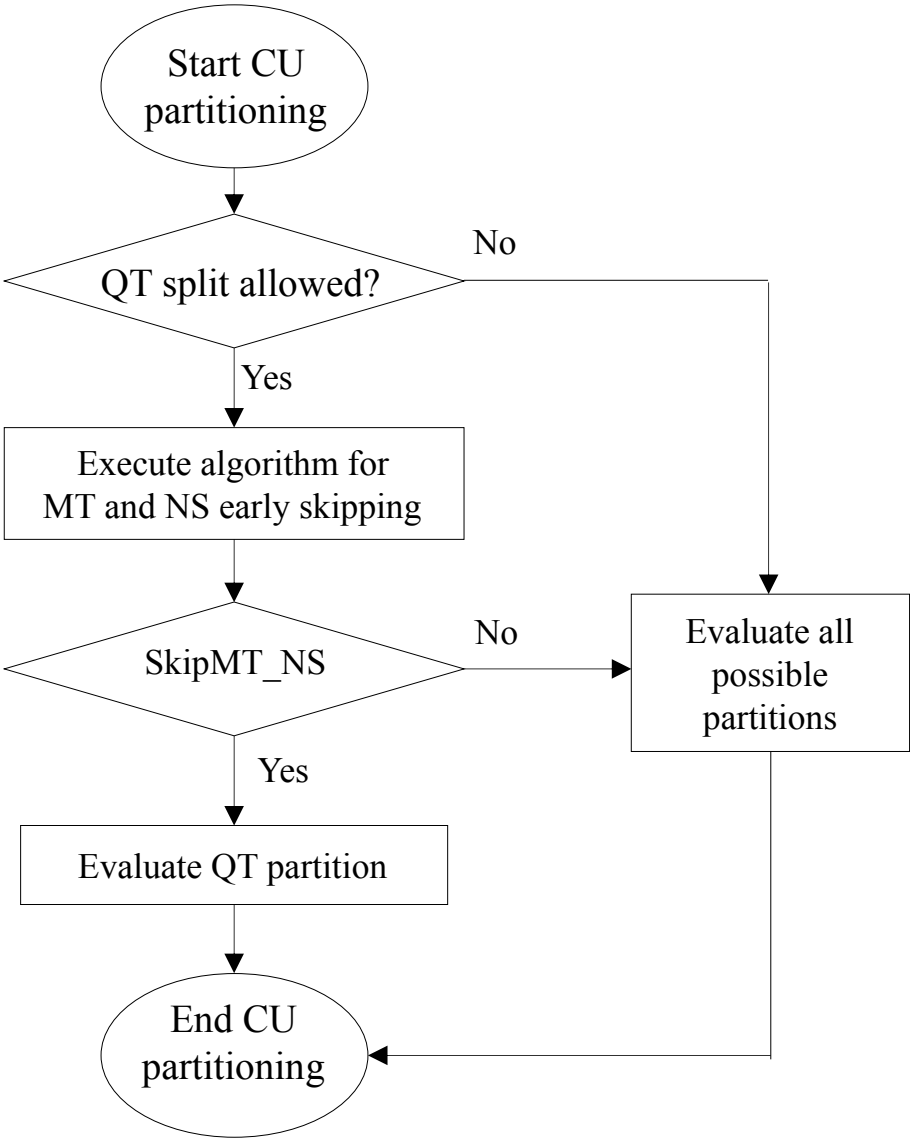


Figure 3.5 – Flow chart of proposed acceleration method

may exist outlier values in the predicted QTdepthMap and we can thus minimize the error brought by these values compared with the maximum or minimum value methods mentioned in [45]. If the average *QTdepth* is larger than the sum of *QTdepth* of current CU and the threshold, it indicates that the evaluations of MT and NS splits are skippable. Otherwise, all possible splits are checked as usual, which means that there is no complexity reduction in that case. We illustrate an example of acceleration through the prediction of QTdepthMap in Fig. 3.4. The circled CU in Fig. 3.1 can be achieved by employing the partitioning outlined by the red path in Fig. 3.4. With the prediction of QTdepthMap,

the QTdepth of the CU can be determined as 3 before starting the partitioning process. Consequently, the RDO check of partitions in the black box can be skipped.

To prove the feasibility and effectiveness of this method, a ground truth - or oracle - test has been conducted before the training and implementation of CNN. This oracle mode allows us to get the possible upper limit in terms of complexity saving. The ground truth test consists of implementing and applying the method with pre-obtained ground truth of QTdepthMap. Experiments show that the ground truth test can speed up the encoding by 39% with merely 0.61% loss on BD-rate compared with the standard VTM10 with the RAGOP32 configuration. Apart from the oracle test, another test measuring the overhead of CNN inference has been done. Our proposed CNN comprising nearly 20k parameters increases the encoding time by only 0.21% when integrated in VTM10.

3.2.5 Dataset and Training Details

800 sequences of different resolutions, namely 240p, 540p, 720p and 1080p, have been encoded using the VTM10 with the RAGOP32 configuration. From this set, 600 sequences of 64 frames come from the database BVI-DVC [82], while the remaining 200 sequences have been selected from the Youtube UGC dataset [83]. We have collected the CTU-level features, mentioned in previous section, together with its QTdepthMap. 200k CTU samples have been randomly chosen for each resolution. The model has been trained using Keras on GTX1080ti with AMD 3700x processor. We used the L1 loss function and Adam optimizer [84]. The learning rate has been set to $1e^{-3}$ for the first 20 epochs, then decreased by 10% every 10 epochs. The batch size for training is 200. The dataset is split into 80% of training data and 20% of validation data. The Mean Absolute Error (MAE) is used as validation metric.

3.3 Experimental Results and Analyses

The proposed method has been implemented in various VTM versions using the frugally-deep library [85] to load the trained model and to perform the inference on the CPU. The tests with the CTC sequences have been carried out on a Linux machine with Intel Xeon E5-2697 v4. These experiments are conducted on the first 64 frames of CTC sequences with the RAGOP32 configuration on four QP values of 22, 27, 32, 37. Two main metrics have been used to assess performance: BD-rate [86] and Time Saving (TS).

Table 3.1 – Performance of the proposed method in comparison with reference CNN-based methods

Class	Sequence	Pan [67] (VTM6)		Yeo [68] (VTM11)		Proposed (Th=0, VTM6)		Proposed (Th=0.1, VTM11)	
		BD-rate (%)	TS (%)	BD-rate (%)	TS (%)	BD-rate (%)	TS (%)	BD-rate (%)	TS (%)
A (3840×2160)	Tango2	4.03	38.56	2.70	25.95	2.93	38.28	1.77	26.92
	FoodMarket4	1.74	46.12	NA	NA	1.70	39.88	1.11	28.26
	Campfire	3.17	38.23	0.58	7.36	0.90	26.98	0.59	18.80
	CatRobot1	6.45	36.84	2.58	25.68	2.99	37.61	2.16	27.65
	DaylightRoad2	5.63	35.47	NA	NA	2.88	35.41	2.37	25.50
	ParkRunning3	2.10	26.45	NA	NA	1.29	35.67	0.88	28.35
	Average	**3.85**	**36.46**	**1.95**	**16.99**	**2.12**	**35.88**	**1.47**	**26.05**
B (1920×1080)	MarketPlace	4.33	33.64	1.33	23.25	2.14	37.44	1.51	27.34
	RitualDance	3.55	34.17	0.92	16.35	2.08	36.69	1.25	26.74
	Cactus	5.72	29.36	0.82	13.80	1.19	30.80	0.90	23.61
	BasketballDrive	3.30	37.28	0.99	15.65	2.19	30.46	1.24	24.85
	BQTerrace	1.90	20.21	0.67	13.54	0.33	22.14	0.39	13.08
	Average	**3.76**	**30.27**	**0.95**	**16.19**	**1.59**	**31.79**	**1.06**	**23.34**
C (832×480)	BasketballDrill	2.29	29.23	0.19	7.43	0.34	21.45	0.27	15.63
	BQMall	2.69	27.48	0.12	3.99	0.56	23.54	0.31	18.35
	PartyScene	2.22	20.80	-0.10	2.88	0.09	19.46	0.07	12.93
	RaceHorses	3.02	26.39	0.15	3.16	0.57	26.70	0.53	21.66
	Average	**2.56**	**25.77**	**0.09**	**4.05**	**0.39**	**22.94**	**0.29**	**17.29**
D (416×240)	BasketballPass	1.85	26.97	0.06	3.37	0.11	13.10	0.08	11.38
	BQSquare	1.61	14.86	0.09	3.54	0.03	8.32	-0.01	2.25
	BlowingBubbles	3.03	22.15	0.12	3.59	0.25	10.51	0.24	8.63
	RaceHorses	2.92	24.20	0.16	0.70	0.37	17.00	0.28	12.55
	Average	**2.35**	**21.53**	**0.11**	**2.34**	**0.19**	**12.34**	**0.15**	**8.82**
E (1280×720)	FourPeople	2.31	33.77	NA	NA	0.97	31.24	0.47	17.66
	Johnny	3.53	35.22	NA	NA	1.16	27.63	0.72	10.56
	KristenAndSara	2.58	36.50	NA	NA	1.33	28.25	1.22	13.27
	Average	**2.81**	**35.15**	**NA**	**NA**	**1.15**	**29.18**	**0.80**	**14.23**
Total average		**3.18**	**30.63**	**0.71**	**7.12**	**1.20**	**27.85**	**0.83**	**19.34**

The formula for computing TS is given in Equation 4.6. T_{Test} indicates the encoding time of the proposed method and T_{VTM} indicates for the encoding time of encoder in the same condition.

$$TS = \frac{1}{4} \sum_{q \in \{22,27,32,37\}} \frac{T_{VTM}(q) - T_{\text{Test}}(q)}{T_{VTM}(q)} \tag{3.1}$$

Before presenting the results, it is important to note that any complexity performance comparison between different versions of the VTM is biased and naturally unfair. This is due to the fact that in the latest versions of VTM, particularly since VTM10, a significant effort has been put by the JVET community to accelerate the VTM software by optimizing its decision engine. As a result, the later the version of VTM is, the less its encoding run time for a given test set is, hence the more difficult it is to accelerate [25].

Our CNN has been trained on dataset generated from VTM10 and been implemented

into VTM10. We have obtained encoding time acceleration from 17% to 30% for only 0.37% to 1.18% BD-rate loss in RAGOP32. Table 3.1 shows the encoding efficiency in terms of BD-rate increase as well as the percentage of TS. In this table, the proposed method is compared with two CNN-based inter partitioning methods of [67] and [68]. Two values of 0 and 0.1 are used as the internal threshold of the proposed method. It is noteworthy that, as described in Algorithm 1, a larger threshold can give a higher acceleration as more partitioning modes are skipped. As can be seen in this table, the proposed method offers the best trade-off between time saving and performance drop. More precisely, we have achieved a triple acceleration with the same loss when comparing to [68]. At the same time, if we compared our result with [67], we obtain a larger acceleration with a significantly lower loss. Finally, it is important to note that we have reimplemented our method in the VTM6 and VTM11 respectively for a fair comparison with reference papers in Table 3.1.

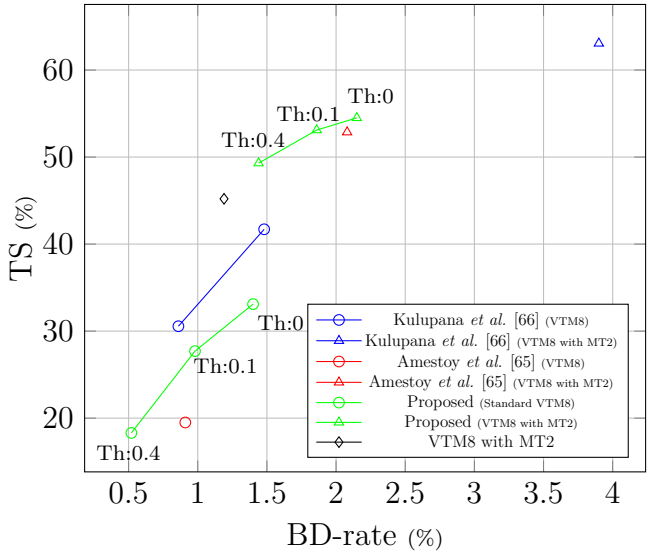


Figure 3.6 – Complexity gains versus BD rate loss in comparison with RF based methods.

The two other acceleration methods for inter partitioning of VVC, notably the RF-based works presented in [65] and [66], generate their results differently. The test set used in these papers is different from JVET CTC. Also, the encoder configuration is modified. In fact, the maximum of the MT split depth has been reduced from its default value 3 to 2 in [65] and [66]. We refer to this setup as VTM8 with MT2 in the following. Fig. 3.6 presents a performance comparison of RF solutions with our method implemented in VTM8 and VTM8 with MT2. For a fair comparison, the results in this figure are

calculated for all sequences CTC included in the test set of [66]. The label “Th" indicate the value of the threshold.

Interestingly, the proposed method achieves similar results to these of the RF based methods. This is remarkable as our method requires a unique network to predict the QTdepthMap for any QP value and any frame position in the GOP, and the learning has been performed for a different VTM version from the one used for testing.

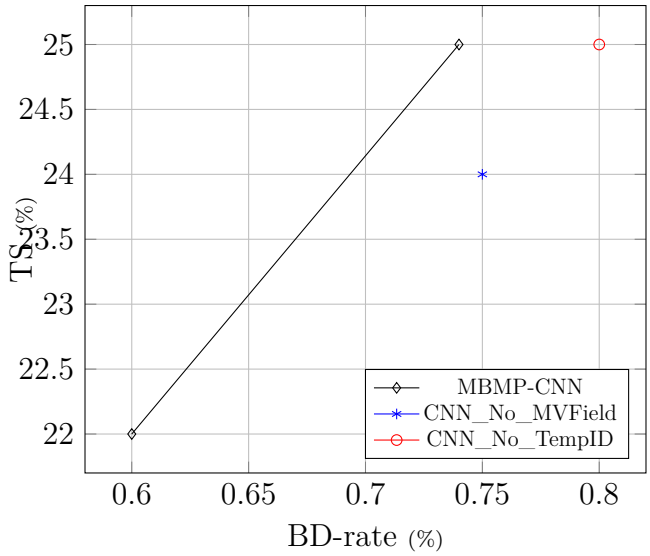


Figure 3.7 – Complexity gains versus BD rate loss for CNN models in ablation study

3.4 Ablation Study

The CTU pixel value and QP value are essential for the CNN model. In this work, we include additional input features such as residual value, tempID value, and MV Field. To assess the impact of these three features, we conducted an ablation study. Three CNN models, named *CNN_No_Resi*, *CNN_No_MVField* and *CNN_No_TempID*, were trained separately on the same dataset. Compared to our purposed model, *CNN_No_Resi* shares the same CNN structure except that the residual input feature is removed, and a similar approach is taken for *CNN_No_MVField* and *CNN_No_TempID*. We then evaluated these models on CTC sequences. The performances of these three models are compared to our *MBMP-CNN* in Fig. 3.7. The performance of *CNN_No_Resi* is 61% TS with a 25.25% BD-rate increase, indicating that the model malfunctions without the residual value as an input feature. Fig. 3.7 shows that the performance of the CNN model

is worse when removing MV Field and tempID features. Consequently, we can conclude that the residual value is crucial for predicting the QTdepth, and MV Field and tempID are valuable features for the CNN model.

3.5 Conclusion

In this chapter, we have proposed a lightweight CNN-based CU partition acceleration algorithm specialised for inter coding in VVC. The proposed MBMP-CNN takes inter related features as input and predicts a QTdepthMap. An early termination algorithm is proposed using the predicted QTdepthMap, to reduce the complexity of the MT and NS partitions. This scalable method succeeds in speeding up the VTM10 efficiently from 17% to 30% for only 0.37% to 1.18% BD-rate increase in the RAGOP32 configuration. Additionally, we have built a large-scale dataset for inter partitioning of VVC which could be beneficial to other related works.

Chapter 4

CNN-based Prediction of Partition Path for VVC Fast Inter Partitioning Using Motion Fields

4.1 Introduction

In Chapter 3, we achieved a maximum acceleration of 30% in inter coding for VTM10. Utilizing the QTdepthMap allowed us to omit the RDO checks for the MT partitions at specific levels of the partitioning tree. Consequently, RDO checks for MT partitions at other levels and QT partitions were not considered. To achieve a more substantial acceleration in this contribution, we aim to speed up the RDO check for both QT partitions and MT partitions across all levels of the partitioning tree. Therefore, the method proposed in this chapter offers a comprehensive approach to accelerating VVC partitioning.

Generally, fast partitioning approaches, described in Section 2.1.2, aim to reduce the search space of potential partitions. Therefore, accurately predicting the subset of partitions is of crucial importance. Heuristic methods proposed for fast intra partitioning of VVC [55, 87] heavily depend on handcrafted features to determine whether to check a partition. These methods are fast and simple to implement but lack accuracy for two reasons. Firstly, the features are computed locally on the CU and/or sub-CUs, which fails to provide a synthesized view of the entire CTU. Secondly, these features, including variances, gradients, and coding information, are low dimensional and do not adequately capture the complexity of CTU.

One approach to improve the accuracy of partition prediction involves increasing the dimension of the extracted features. This is the case with the methods based on RF [65] or DT [66], which use over 20 features from a given CU and its sub-CUs. As a result, decisions made by these methods remain confined to the local context of CU, without considering the entirety of CTU. Rather than relying on local information, a more effective selection

of subsets of partitions should be based on global features computed on the entire CTU. This can be accomplished through the utilization of CNN-based methods.

Several approaches [67, 68, 88] use CNN to partially accelerate the partition search process. For instance, in [67], Pan *et al.* propose a multi-branch CNN to perform a binary classification of the "Partition" or "Non-partition" at the CU level. In [68], the split type at the CTU level is predicted, whereas the partitions of its sub-CUs are not determined. In our contribution in Chapter 3, we employ a CNN to estimate an 8×8 grid map of QT depth, which is used to discard a portion of the MT splits. These methods cover only a part of the partition search space, while the partition search is conducted exhaustively on the remainder. These methods could be referred as partial partitioning acceleration methods by CNN.

A complete partitioning acceleration of inter coding by CNN is proposed in [70]. A vector containing probabilities of the existence of split boundaries in the partition is predicted similarly to [60]. This method is fast in the sense that a single vector is computed for each CTU. Nevertheless, it is observed in [60], that the predictions are more accurate at higher levels of the partitioning tree. Hence, they propose improving the decisions by adding 16 trained DTs to process the CNN output, introducing additional complexity to the method.

In the MT partitioning, both binary and ternary (with sub-CUs of two different sizes) splits are available. Consequently, CUs at a specific depth in the tree do not correspond to the same size and shape, introducing dependence between the MT splits along the partition path. This dependence partly explains the decrease in partition prediction accuracy as the depth of the partitioning tree increases, as observed in recent studies [60, 70] presented in the previous section.

More precisely, since the size and shape of a CU depend not only on its depth in the tree but also on consecutive MT splits, depth alone is insufficient to define a partition. Therefore, we propose making decisions on MT partitioning in a hierarchical manner, considering their dependence on the partition path. We also introduce a one-shot approach for QT partitioning which precedes MT partitioning, since there is a one-to-one correspondence between the QT depth and the CU size at that particular depth.

Hence, our overall proposition involves predicting the partition path, which includes a one-shot prediction for the QT partitioning, followed by a hierarchical prediction for the MT partitioning. Additionally, to further improve the accuracy of partition prediction, we suggest basing the partition decision not only on pixel values and residual values but

also on motion vector fields, as these fields exhibit a strong correlation with partitioning [65]. Similarly to previous work, our method is based on VTM10. Furthermore, we have re-implemented our method in VTM6 for comparison with state-of-the-art methods.

Our two main contributions are as follows:

— We propose a novel partition-path-based representation of the QTMT partition at the CTU level as a map of QT depth plus three maps of MT split well adapted to the sophisticated partitioning scheme in VVC.

— We design a U-Net-based CNN model taking multi-scale fields of motion vectors as input to effectively predict QT depth map as well as split decisions at different MT levels.

We also have other contributions such as:

— We build MVF-Inter [1], a large-scale dataset for the inter QTMT partition of VVC, which could facilitate research in this field.

— We propose a fine tuned loss function for this complex multi-branch multi-class classification problem.

— We develop a fast partition scheme effectively exploiting the prediction of a CNN model in a way that the most possible splits are determined at each partition level.

— We design a specific threshold-based selection approach to match with the partition scheme, which allows us to realize a large range of trade-offs between complexity and compression efficiency.

The remainder of this chapter is organized as follows. In Section 4.2, we provide an overview of the proposed method. In this section, we present a novel representation for QTMT partition and illustrate the structure of the CNN model. In the end, we provide details on the corresponding acceleration algorithm and the training of the CNN model. In Section 4.3, we evaluate the prediction accuracy of our CNN model. Additionally, we compare our results with the state-of-the-art of RF and CNN based methods, respectively.

1. Our dataset MVF-Inter is available at `https://1drv.ms/f/s!Aoi4nbmFu71Hgx9FJphdskXfgIVo?e=fXrs0o`

This section also includes a complexity analysis of our method. Finally, in Section 4.4, we draw conclusions and provide perspectives for future work. Our source code is available at `https://github.com/Simon123123/MSMVF_VVC_TIP2023.git`.

4.2 Proposed Method

In this work, we use the concept of partition path to depict the partition of a CU. The partition path refers to the sequence of splits applied to obtain a CU during the partitioning. In the RDO process of partitioning, numerous partition paths included in the partition search space are checked and the optimal one leading to the final partition is selected. Figure 3.2 in Chapter 3 illustrates a simplified tree representation showing all possible partition paths checked for a CTU. The green arrows indicate the selected partition path with the lowest RD-cost [89], while the black arrows represent other paths that have been tested by RDO, but were not selected.

Specifically, within the QTMT partition, it is important to note that QT splits are prohibited for the child nodes of a MT split. Consequently, the search for optimal partition path in VVC can be conceptualized as a sequential two-step decision-making process, comprising a sequence of QT splits followed by a series of MT splits.

4.2.1 Novel Representation of QTMT Partition

Based on the partition structure QTMT and the partition path of VVC presented in the previous section, we introduce in this section a novel representation of QTMT partition by partition path. In 4.2.1.1, we explain the motivation for this new representation. The partition path representation is illustrated in 4.2.1.2.

4.2.1.1 Motivation

Previous partition representations at the CTU level have typically used binary vectors to depict split boundaries. In [59, 60, 70], the authors intend to predict the split boundary of each 4×4 sub-block in CTU. Lately, Wu *et al.* improve this representation in [61] by proposing hierarchical boundaries. This adaptation is designed to better align with the QTMT partition pattern. In this work, binary labels for split boundaries of varying lengths are predicted. Collectively, these methods provide a geometric representation of the partition.

The limitations of the geometric representation mainly lie in two aspects. Firstly, it is an implicit representation of the partitioning process, requiring conversions from boundary vectors to split decisions. In the case of [59], conversions are carried out by computing the average probability at the location of the specific split. [70] and [60] convert boundary vectors to split decisions by DT models separately trained for different CU sizes. Secondly, different partition paths could be deduced from a particular partition presented in a geometric way. For example, as demonstrated in Figure 4.1, the partition defined by the split boundaries can lead to three distinct partition paths. These partition paths correspond to different coding performances and are individually tested in the RDO process. This multiplicity of partition paths of the geometric representation limits the acceleration potential of the method.

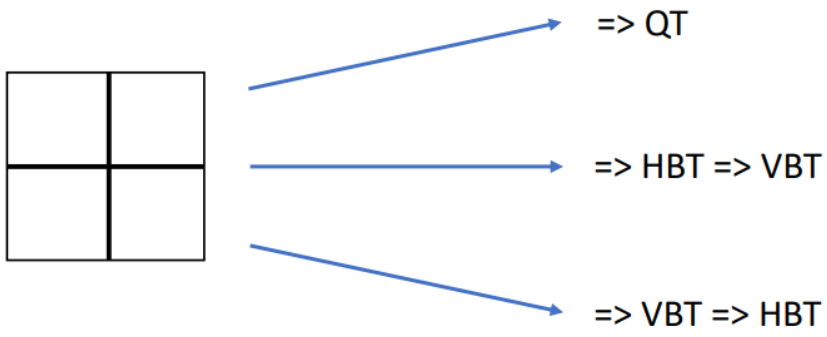


Figure 4.1 – Possible partition paths for a final partition given by split boundaries

To address the above limitations, we introduced a novel representation based on the partition path. Our representation comprises the QT depth map and the MT split maps. Firstly, the split decisions at each depth can be directly deduced from either the QT depth map or the split map. This eliminates the need for decision trees, reducing method overhead, and simplifying implementation. Secondly, it corresponds to a unique partition path, maximizing the potential for complexity reduction.

4.2.1.2 QT Depth Map and MT Split Maps

Considering that the maximum number of QT splits and MT splits is typically set to 4 and 3 in VTM, any partition can be effectively described by a QT depth map (*i.e.* QTdepthMap) along with three MT split maps (*i.e.* Multi-type Tree split Map (MTsplitMap)) in sequence. Each element within the QTdepthMap and MTsplitMap

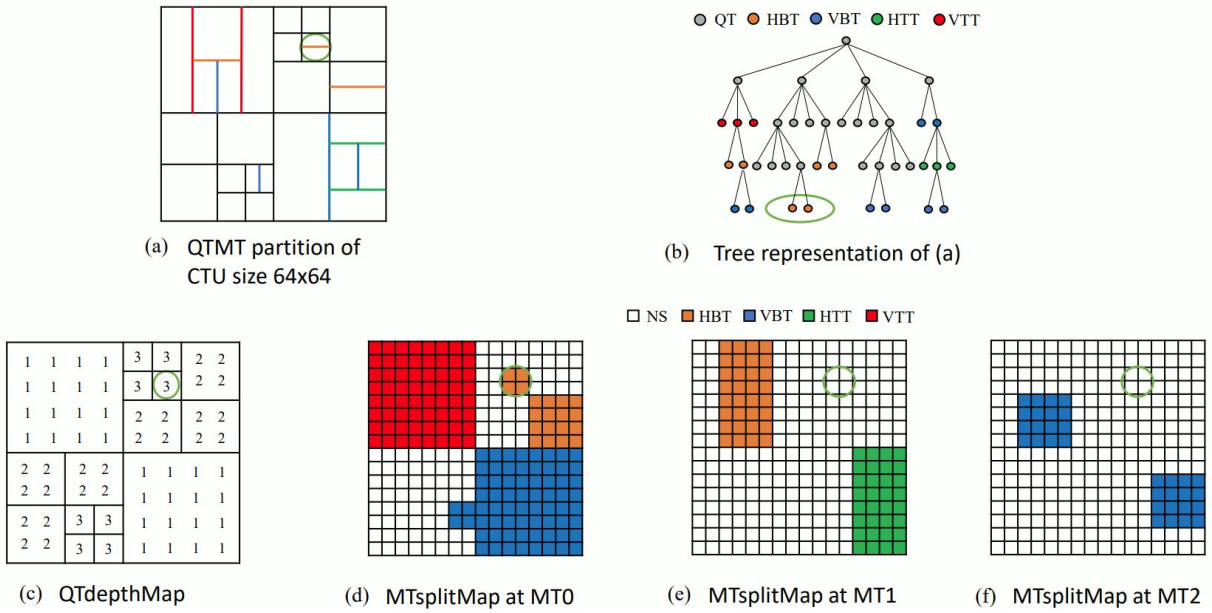


Figure 4.2 – Example of QTMT partition, tree representation, QTdepthMap and MTsplitMaps of CTU size of 64×64.

corresponds to an 8×8 and 4×4 area, which aligns with the dimensions of the smallest sub-CUs for QT split and MT split in VTM.

A detailed example of our partition representation is shown in Figure 4.2. To keep it simple and without loss of generality, we represent this example for a CTU size of 64×64. In this figure, (a) shows an instance of QTMT partition with its corresponding tree representation shown in (b). (c)-(f) illustrate the QTdepthMap and MTsplitMaps generated from this partition. Given that the CTU size in this example is 64×64, the sizes of QTdepthMap and MTsplitMap are 8×8 and 16×16, respectively. The QTdepthMap in (c) consists of QT depth values ranging from 0 to 4, while each element in MTsplitMap in (d)-(f) represents the split decision among five options: NS, HBT, VBT, HTT and VTT. This representation depicts a distinct partition path for every CU within the partition. To provide an example, consider the CU highlighted in the green circle in Figure 4.2. Its partition path can be expressed as three QT splits (QT depth 3), followed by a HBT split and two NS decisions.

4.2.2 CNN-based Prediction of Partition Path

Predicting the optimal partition is equivalent to predicting the optimal partition path. In VTM, the size of CTU is set to 128×128 by default, consequently yielding QTdepthMap

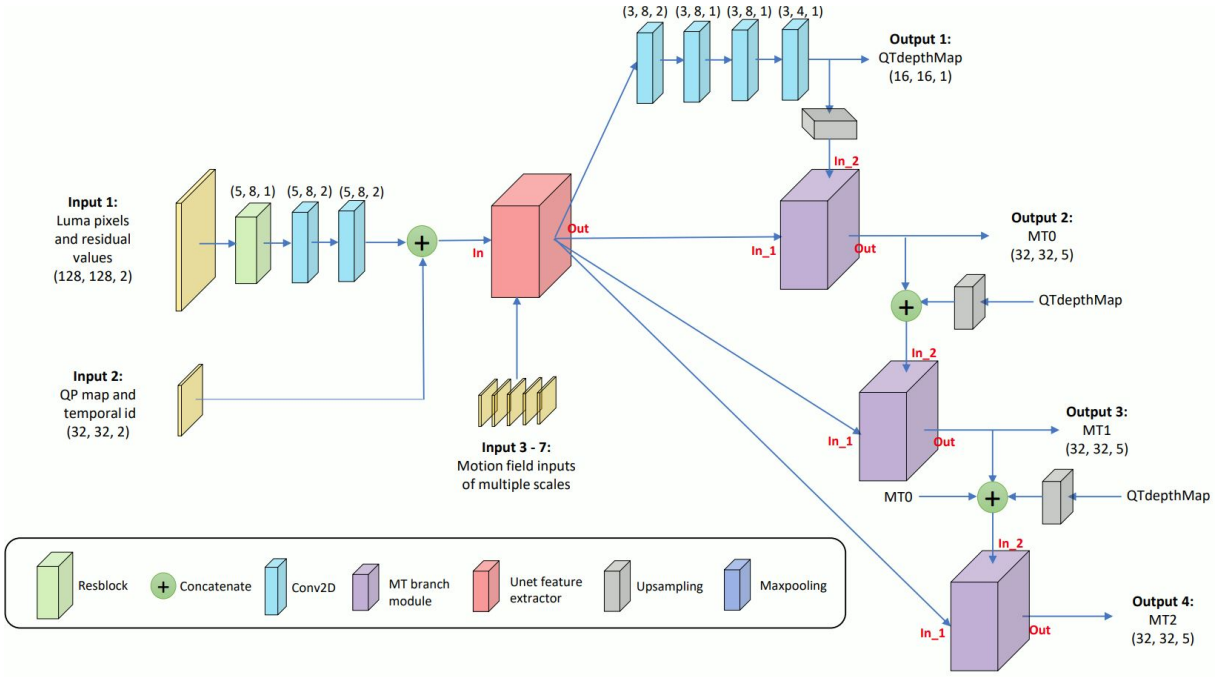


Figure 4.3 – Multi-Scale Motion Vector Field CNN. The vector above Resblock and Conv2D represents (kernel size, number of filters, stride).

and MTsplitMap dimensions of 16×16 and 32×32, respectively. The representation of partition path can be predicted by a multi-branch CNN, where one branch infers the QTdepthMap of regression values with dimension 16×16×1, while the other three branches produce the MTsplitMap. Each element of MTsplitMap is classified into one of five classes, corresponding to five split types, resulting in three MT outputs with dimensions of 32×32×5. We have handled the classification of MT splits as an image segmentation problem based on 4×4 sub-blocks. Accordingly, we adopted the classical U-Net structure [90] to design our CNN model to address this segmentation-like task.

In this section, the U-Net structure is briefly introduced. Then we present the structure of the proposed CNN in Section 4.2.2.2. Afterwards, we list its input features and explain the reasons for choosing them in Section 4.2.2.3.

4.2.2.1 U-Net

The U-Net structure is derived from Fully Convolutional Network (FCN)[91]. It consists of an encoder part which is composed of a sequence of convolutional layers plus maxpooling layers. Then this part is followed by a decoder part in which the maxpooling layers are replaced by upsampling layers. In addition, skip connections concatenate

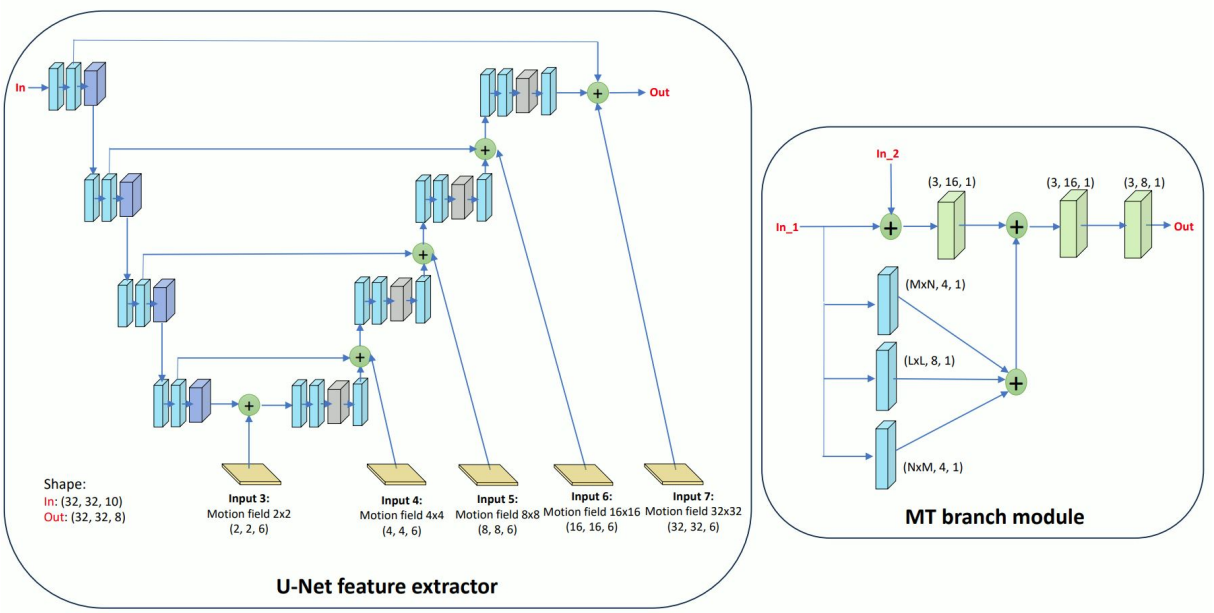


Figure 4.4 – U-Net feature extractor and MT branch module

feature maps from the encoder and decoder with the same dimension. The U-Net and its variations have been widely applied to image segmentation tasks.

4.2.2.2 MS-MVF-CNN Structure

The CNN structure proposed in this paper, named Multi-Scale Motion Vector Field CNN (MS-MVF-CNN), is depicted in Figure 4.3. The proposed CNN has 7 inputs and 4 outputs. After two convolutional layers with stride, the tensor of Input 1 is downsampled to dimension 32×32×8 and then concatenated with the Input 2. The merged input is then fed to the U-Net feature extractor demonstrated in Figure 4.4. Regarding the design of this module, we are referring to the classical structure of U-Net depicted in [90]. Specifically, we concatenate the upsampled feature map in the decoder part of U-Net, the feature map copied from the encoder part with the motion vector field of the same scale. At the decoding part, the feature map is gradually expanded and merged with normalized motion field of 2×2×6, 4×4×6, 8×8×6, 16×16×6, and 32×32×6. As a result, the U-Net feature extractor outputs a feature map of dimension 32×32×8, combining pixels features with motion estimation features.

Since the split at each level depends on previous splits, we employ a hierarchical multi-branch prediction mechanism. QTdepthMap is predicted after shrinking the features

extracted from U-Net by four convolutional layers. For MT branches, we designed the MT branch module presented in Figure 4.4. Two inputs of this module are the extracted features of U-Net and outputs from previous partition levels. We utilize the asymmetric kernel structure to process the extracted features. This structure is originally proposed by [46] in HEVC to pay attention to near-horizontal and near-vertical textures for predicting split decision of intra coding by CNN. We adopt this structure to exploit the horizontal and vertical information contained in Multi-Scale Motion Vector Field (MS-MVF). The MT branch module contains branches of kernel size MxN, LxL, and NxM. The values of (M, N, L) are set as (5, 7, 9) for branch MT0, (3, 5, 7) for branch MT1, and (1, 3, 3) for branch MT2. On deeper MT levels, splits are made on smaller CUs. Thus, smaller kernel sizes are applied to extract finer features. After the asymmetric kernels, the feature map is then concatenated with outputs from previous levels. In the end, the merged feature maps are given to two residual blocks [81] before yielding classification results of MT branches. No activation is applied to the fully connected output layer of the QT depth branch. The output layer of the MT branch is with softmax activation.

4.2.2.3 Input Features

This network structure takes three different types of input. The involved inputs are presented below:

4.2.2.3.1 Original and Residual CTU

In Figure 4.3, Input 1, with dimensions of 128×128×2, is created by merging the original CTU with the residual CTU. The original luma pixels carry the texture details of the CTU, while the residual CTU is generated through motion compensation of the original CTU based on the nearest frame.

Several studies [67, 68, 80] have adopted a method in which both the original CTU values and the residual of CTU are fed to a CNN. Combining the original and residual values as input allows CNN to assess the similarity between current CTU and reference CTU. This combined input offers features that reflect the temporal correlation between frames which is a crucial factor in inter partition prediction.

4.2.2.3.2 QP and Temporal ID

The Input 2, as illustrated in Figure 4.3, has dimensions of 32×32×2, consisting of two separate 32×32 matrices. These matrices are assigned specific values: one holds the QP value, while the other contains the temporal identifier. This temporal identifier in VVC, similar to its usage in HEVC, signifies a picture's position within a hierarchical temporal prediction structure, controlling temporal scalability [92].

We specifically utilize the QP value and temporal identifier as input features since inter partitioning depends on them. In essence, a higher temporal layer identifier or a lower value of QP tends to result in finer partitions, as outlined in [68]. Instead of developing separate models for each parameter instance, our approach focuses on training a model with adaptability to varying values of QP and temporal identifier.

4.2.2.3.3 Multi-Scale Motion Vector Field

In this paper, we have introduced a CNN model based on a novel input feature called MS-MVF. Our MS-MVF at five scales is presented as Input 3-7 in Figure 4.4. To compute MS-MVF, we divide the 128×128 CTU into multiple scale sub-blocks ranging from 4×4 pixels to 64×64 pixels, and perform motion estimation on these sub-blocks. Each motion vector of sub-block comprises a vertical and horizontal motion value, along with the associated SAD cost value as the third element. By concatenating elements pointing to reference frame of L0 with those of L1, each sub-block corresponds to 6 elements in the motion vector field. For example, the motion vector field input for 8x8-pixel scale has dimensions of 16×16×6.

A significant challenge in inter partition prediction is the large motion search space, which spans up to 6 regions of 384×384 pixels across different reference frames in the RAGOP32 configuration. State-of-the-art methods typically employ motion fields or pixels from reference frames as input features for machine learning models. Notably, in [65] and [67], a crucial feature used is the motion field, which comprises motion vectors calculated for each 4×4 sub-block referring to the nearest frame. As mentioned in [65], this motion field is strongly correlated with the optimal partition. In a different approach, Tissier *et al.* in [70] opt to utilize two reference CTUs in the nearest frames.

The choice of using MS-MVF as the CNN input, instead of motion fields and reference pixels, is based on the following reasons. First, the MS-MVF contains crucial motion information for the current CTU, which is essential for both inter prediction and inter partitioning. This information can be interpreted more effectively by the CNN model

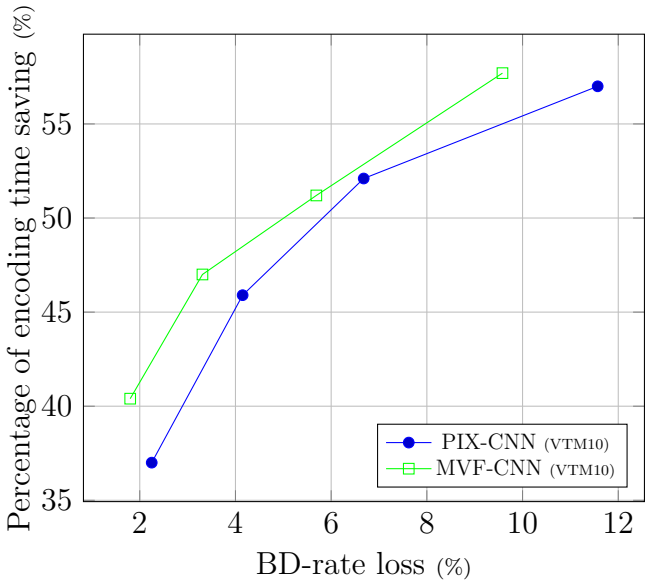


Figure 4.5 – Comparison of performances between PIX-CNN and MVF-CNN.

compared to using reference pixels as CNN input. Second, the multi-scale nature of the MS-MVF aligns well with the multi-level structure of U-Net and can leverage this structure effectively. Essentially, MS-MVF represents motion features at various resolutions, allowing for the combination with features extracted from CTU pixels at the same resolution scale.

To demonstrate the effectiveness of our MS-MVF input, we conducted an experiment involving the training of two CNN models. The only distinction between these models is their input: the first model, PIX-CNN, takes the pixels of two reference CTUs as input, while the second model, MVF-CNN, utilizes our proposed MS-MVF as input. Both models share the same architecture as in Figure 4.3. The training dataset comprises 250k samples randomly selected from the RAGOP32 encoding of 200 sequences with a resolution of 540p from [82]. Performance evaluations in Figure 4.5 are based on Class C sequences of CTC. The results consistently show that MVF-CNN outperforms PIX-CNN at all four data points, which justifies the advantages of using the MS-MVF input over pixel input.

Based on our evaluation conducted on the first 64 frames of all CTC sequences using the RAGOP32 configuration, the computation of MS-MVF for each CTU consumes, on average, a mere 0.52% of the encoding time in VTM10. Importantly, the generation of MS-MVF introduces only minimal encoding overhead, making it a task that can be readily

preprocessed or parallelized.

4.2.3 Proposed CNN-based Acceleration Method

After the prediction by our trained CNN model, we obtain one QTdepthMap and three MTsplitMaps per CTU. The predicted QTdepthMap is composed of floating-point values. The predicted MTsplitMaps comprise probabilities of five split types for each 4×4 sub-block within the CTU. In this section, we elucidate the post-processing of the CNN prediction, with the aim of achieving a wide range of acceleration-loss trade-off.

Algorithm 2 MT splits early skipping

Input:
QTdepthMap; MTsplitMap; Thm; $\text{QTdepth}_{\text{cur}}$, CU; Size_{CU}; Pos_{CU}
Output:
SkipMT: Boolean to decide whether to skip MT split types or not.
CandSplit: Candidate list of splits for RDO check

Compute the average $\text{QTdepth}_{\text{pred}}$ based on Size_{CU}, Pos_{CU} and QTdepthMap
if round($\text{QTdepth}_{\text{pred}}$) > $\text{QTdepth}_{\text{cur}}$ **and** QT is possible for current CU **then**
    SkipMT = True
    CandSplit = {NS, QT}
else
    SkipMT = False
    CandSplit = {NS}
    for sp in {BTH, BTV, TTH, TTV} **do**
        Compute average Proba_{sp} based on Size_{CU}, Pos_{CU} and MTsplitMap
        if Proba_{sp} > Thm **then**
            CandSplit append split sp
        end if
    end for
end if

Decision errors at low partitioning depth can result in large loss of BD-rate. Based on the predictions of our CNN, selecting the best single partition path, equivalent to choosing the best split at each MT level, will not be optimal or scalable. Our approach involves generating candidate lists at each level, which means that multiple partition paths are chosen for the RDO test. This approach of creating candidate lists at various levels is designed to achieve satisfying trade-off between acceleration and coding loss while assuring the scalabilty of method.

The acceleration algorithm is precisely described in Algorithm. 2 and Figure 4.6. We introduce two parameters *Thm* and *QTskip* to regulate the acceleration-loss trade-off. Specifically, *Thm* is the threshold for the split probability. *QTskip* represents whether we should accelerate RDO of QT splits or not. Increasing the *Thm* value and setting *QTskip* to true will lead to greater acceleration at the cost of increased coding loss.

Regarding the algorithm applied at the CU level, Algorithm. 2 is first executed. This algorithm produces two outputs: the *SkipMT* variable and the *CandSplit* list, both of which are subsequently utilized in the flowchart in Figure 4.6. To start with, the mean $QTdepth_{pred}$ of current CU is calculated based on the corresponding area in QTdepthMap. If the rounded $QTdepth_{pred}$ is larger than the QT depth of the CU and QT split is feasible, the current CU should be split by QT. Consequently, all MT splits are excluded from the *CandSplit* list and *SkipMT* is set to true. Otherwise, the mean probability of each available split is computed on the corresponding MTsplitMap in a similar way to that of the QTdepth. Then *CandSplit* is filled by splits with $Proba_{sp}$ larger than the threshold *Thm*. In this case, the value assigned to *SkipMT* is False.

In the flowchart of Figure 4.6, if the *SkipMT* is true after the execution of Algorithm 2, we directly check the *CandSplit*. In this scenario, the encoder conducts RDO of CU and splits CU with QT because *CandSplit* contains only NS and QT. If *SkipMT* is false, then we will verify if NS is the only choice in *CandList*. If this is the case, we will add the MT split with the highest probability to the list. Next, if QT split is not allowed for CU due to CU shape or shortcuts, we go directly to the check of *CandSplit*. If the QT split is feasible, we refer to *QTskip* to determine whether to add QT to the *CandList* or not. Setting the *QTskip* to true signifies that we will always check QT if possible. This is for rectifying the potential error of predicting a $QTdepth_{pred}$ value smaller than the actual ground truth value. However, it comes at the expense of sacrificing some acceleration. Finally, we execute RDO on CU and partition it by split types in the *CandSplit* list. The partition search then repeats for the next CU, and the algorithm described above is applied anew.

Our inter partitioning acceleration method is designed on top of the partitioning algorithm of VTM which performs a nearly exhaustive search on possible partition paths of a CTU, except that it incorporates a handful of handcrafted conditional shortcuts. Therefore, this work can be considered as a CNN-based shortcut to reduce the search space of partition paths.

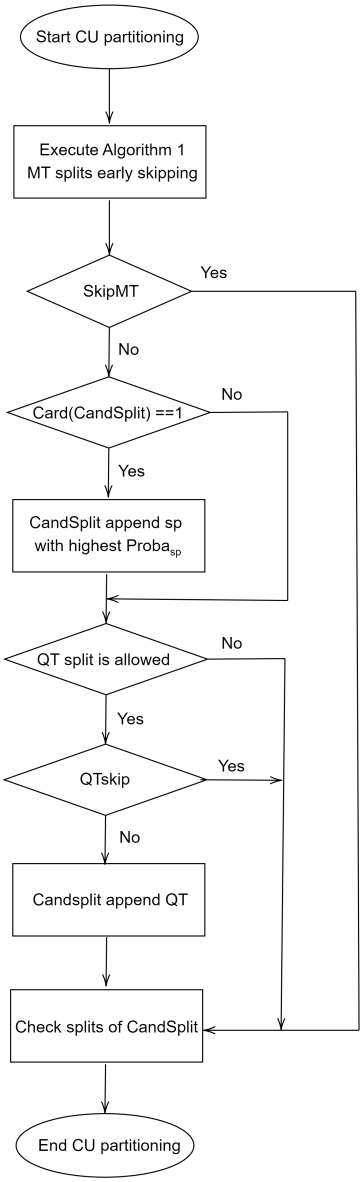


Figure 4.6 – Flowchart of acceleration algorithm

4.2.4 Training of MS-MVF-CNN

To effectively train our CNN model, we have designed a hybrid loss function and created a large-scale dataset named MVF-Inter[1]. First, we will explain how this loss function is determined in Section 4.2.4.1. Then Section 4.2.4.2 describes training details and the generation of the dataset.

4.2.4.1 Loss Function

The outputs of MS-MVF-CNN contain one regression output as well as three classification outputs. Therefore, a hybrid loss function is developed in our case. We choose the category cross-entropy for classification loss and mean square error for regression loss as follows:

$$L = a\frac{1}{n_q}\sum_{i=1}^{n_q}(d_i - \hat{d}_i)^2 - (1-a)(\sum_{b=1}^{n_b}\sum_{i=1}^{n_m}\sum_{s=1}^{n_s} w_{b,s} y_{b,i,s}\log(\hat{y}_{b,i,s})) \tag{4.1}$$

Here, we have $n_q = 256$, $n_b = 3$, $n_m = 1024$ and $n_s = 5$, representing the number of elements in QTdepthMap, the number of MT branches, the number of elements in MTsplitMap, and the number of split types, respectively. In this equation, d_i denotes the ground-truth QT depth value, while $\hat{d}_i$ represents the predicted QT depth value. Additionally, $\hat{y}_{b,i,s}$ is used to denote the predicted probability of split type s for the i-th element of the MT decision map at the b-th MT branch. Similarly, $y_{b,i,s}$ signifies the ground-truth label for the same case. Notably, we introduce a parameter a, which falls within the range [0, 1], in Equation 4.1 to fine-tune the relative weights of the regression loss and classification loss.

The split types are distributed unbalancedly at different MT depths as illustrated in Figure 4.7. To counteract this imbalance, we introduce class weights for split type s on MT branch b, denoted as $w_{b,s}$. The definition of these weights is as follows:

$$w_{b,s} = \frac{\lambda_s p_{b,s=ns}}{p_{b,s}} \tag{4.2}$$

where $p_{b,s}$ represents the percentage of split type s within MT branch b. For each branch b, $\frac{p_{\mathrm{b,s=ns}}}{p_{\mathrm{b,s}}}$ can be interpreted as the inverse percentage of the split type s normalized by the inverse percentage of the NS split. In [33], a series of tests were performed to evaluate the coding gain and increase of complexity associated with the BT and TT splits individually as demonstrated in Table 4.1.

Table 4.1 – Settings of split type in VTM9 under RA [33]

	BT split	TT split	BD-rate	Encoding Time
Anchor Setting	X	X	-	-
Setting 1	✓	X	-8.26%	337%
Setting 2	X	✓	-10.22%	732%

When comparing Setting 1 and Setting 2 to the anchor configuration, it's observed that Setting 1 and Setting 2 exhibit similar BD-rate gains, but the encoding time in Setting 2 is twice that of Setting 1. These tests suggest that BT split performs better in terms of the trade-off between complexity and coding gain compared to TT split. Thus, placing greater importance on the prediction of the BT split can result in a better acceleration-loss trade-off. To achieve this, the ratio between the proportion of NS and proportion of split s is computed for MT branch b. The class weight $w_{b,s}$ in Equation 4.2 is formulated as the product of this ratio and λ_s which is another weight added to prioritize the split type s.

After fine-tuning the model, we find that the best performance is achieved with a value of 0.8 for a and λ_s set to 2 for BT splits and 1 for TT splits and NS.

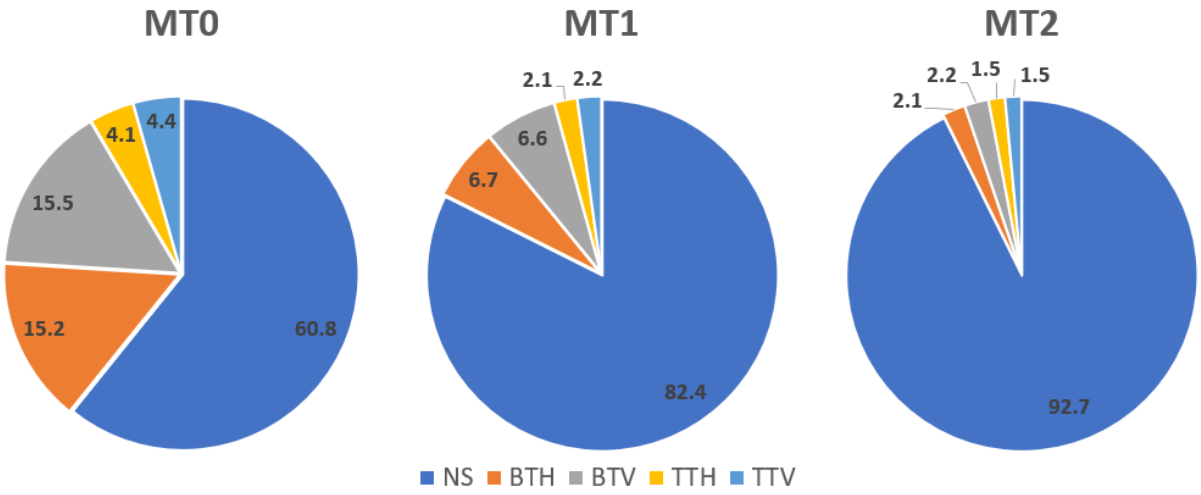


Figure 4.7 – Distribution of split types for MT0, MT1, MT2

4.2.4.2 Dataset Generation and Training Details

Constructing a large-scale inter-partition dataset is more challenging than that of intra partition because the former needs to encode a substantial number of video sequences, while the latter could be done by encoding images. To the best of our knowledge, there exists no prior work focused on developing an inter-partition dataset.

Our MVF-Inter[1] dataset involved the encoding of 800 sequences from [82] and an additional 28 sequences of 600 frames in 720p resolution extracted from [83]. Sequences of [82] cover resolutions of 240p, 540p, 1080p, and 4k, with 200 videos of 64 frames for

each resolution. We have encoded all these videos with the VTM10[93] encoder in the RAGOP32 configuration with QP 22, 27, 32, and 37. We randomly selected a total of 820k CTU partition samples, equally distributed per resolution and QP, with 120k samples reserved as a validation set.

Each sample of our dataset contains the following components of each CTU: pixel values, residual values, motion vector fields at five scales, QP value, temporal ID value, QTdepthMap with depths ranging from 0 to 4, and MTsplitMap for MT0, MT1, and MT2. MTsplitMap labels are encoded as VTT (0), VBT (1), NS (2), HBT (3), and HTT (4).

In terms of training details, we employed the Adam optimizer [84] to train the model. The initial learning rate was set to $1e^{-3}$ and was exponentially decreased by 3% every 5 epochs. The batch size set for training is 400.

4.3 Experimental Results and Analyses

In this section, we present the results of our experiments and provide an in-depth analysis of the results. To begin, in Section 4.3.1, we assess the precision of the prediction of our CNN model. Subsequently, comparisons with the RF and CNN based approaches are made in Section 4.3.2. Finally, the complexity analysis of our framework is carried out in Section 4.3.3.

4.3.1 Prediction Accuracy Evaluation

At the CU level, our algorithm can be broken down into two decisions: the decision of *SkipMT* and the decision of *CandSplit* list. To evaluate the precision of decisions based on our model's output, we have performed the encoding where both the ground truth partitioning and the CNN output were collected. The analysis is done on the first 64 frames of all CTC sequences excluding class D with QP 22, 27, 32, 37. The accuracy of these decisions presented in Table 4.2 and Figure 4.8 are calculated by averaging four QPs and various test sequences.

There is no need to make a *SkipMT* decision on QT depth 4 since the partitioning is forced to proceed to MT splits with the maximum of QT depth reached. The accuracy of *SkipMT* decision is independently measured on QT depth from 0 to 3. If the current CU requires further splitting of QT and *SkipMT* is equal to False, then this decision of

SkipMT is classified as False Negative (FN). The proportion of True Positives (TP), FN, True Negatives (TN), False Positives (FP) and their corresponding Precision (Prec) and Recall (Rec) are shown in Table 4.2. Precision, recall, and F1 score are calculated as follows:

$$Precision_{\text{QTdepth}} = \frac{TP_{\text{QTdepth}}}{TP_{\text{QTdepth}} + FP_{\text{QTdepth}}} \tag{4.3}$$

$$Recall_{\text{QTdepth}} = \frac{TP_{\text{QTdepth}}}{TP_{\text{QTdepth}} + FN_{\text{QTdepth}}} \tag{4.4}$$

$$F1score_{\text{QTdepth}} = 2\frac{Precision_{\text{QTdepth}} Recall_{\text{QTdepth}}}{Precision_{\text{QTdepth}} + Recall_{\text{QTdepth}}} \tag{4.5}$$

Generally, our model exhibits strong performance at QT depths ranging from 0 to 2, as depicted in Table 4.2. Both precision and F1 score decrease as QT depth increases. At QT depth 3, the precision and F1 score drop to 25% and 40%, respectively, suggesting that the *SkipMT* decision at this level is less reliable. These observations could be explained by two reasons:

First of all, the scale of decision-making diminishes as the QT depth increases. More explicitly, the *SkipMT* decision at QT depth 0 is made at the CTU scale by computing the mean of 256 values from the QTdepthMap. Nevertheless, the decision at QT depth 3 relies only on 4 values from the QTdepthMap within the 16×16 CU. Consequently, decisions at smaller scales are less resilient to incorrectly predicted QTdepthMap values, resulting in lower overall accuracy at higher QT depths.

Secondly, decisions at higher QT depths are noticeably more imbalanced than those at lower QT depths. Positive cases of ground truth at QT depth 3 represent only 0.02% , while the proportion of positive cases is 49.65% at QT depth 0. In conclusion, the model is trained in such a way that it tends to make negative *SkipMT* decision at larger QT depths. This explains the decline in precision as the QT depth increases.

Table 4.2 – Table of confusion for *SkipMt* (Unit: %)

	TP	FN	TN	FP	Prec	Rec	F1score
QT depth 0	41.84	7.81	45.83	4.52	90.3	84.3	87.2
QT depth 1	19.53	0.58	72.57	7.32	72.7	97.1	83.1
QT depth 2	2.69	0.08	94.67	2.57	51.1	97.1	67.0
QT depth 3	0.02	0	99.92	0.06	25.0	100.0	40.0

In Figure 4.8, the accuracy of the *CandSplit* list decision is determined by whether the list contains the ground truth split at the MT level. We calculate and draw separate accuracy curves for MT0, MT1 and MT2 separately by varying the threshold *Thm*. As *Thm* increases, the size of the *CandSplit* list decreases, leading to decreasing precision. Once *Thm* reaches a certain value, the accuracy stabilizes because the *CandList* is constant, containing only the MT split type with the highest probability and NS. It's worth noting that the minimum accuracy of the MT increases with the MT depth. This is mainly due to the fact that NS is more frequent at larger MT depths, as illustrated in the pie chart in Figure 4.7. Since our *CandSplit* list consistently includes NS, the accuracy tends to be relatively higher at larger MT depths.

In general, our model achieves a satisfactory F1 score for QT depths 0, 1 and 2 regarding the *SkipMT* decision. As for the *CandSplit* list decision, our algorithm maintains an accuracy exceeding 65% while adjusting the value of *Thm* at various MT levels. These performance evaluations justifies the high accuracy of the decisions made by our method during the partition search process in VVC.

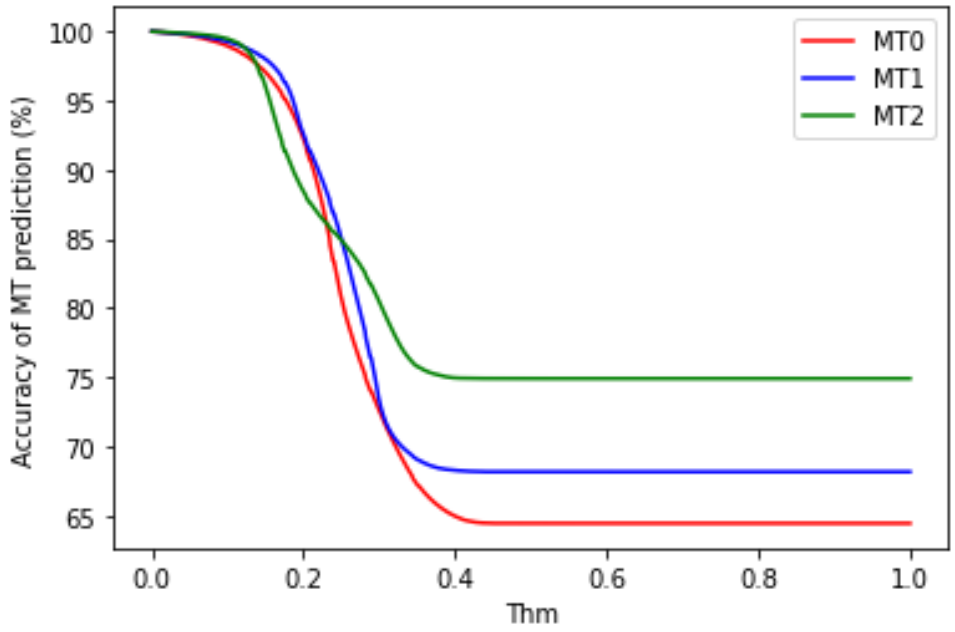


Figure 4.8 – Curves of accuracy and *Thm* for MT0, MT1, MT2

4.3.2 Comparison with the State of the Art

The proposed method has been implemented in the VTM10.0 encoder using the Frugally deep library [85] for CPU-based inference in real time. Encodings of CTC sequences are performed on a Linux machine with the Intel Xeon E5-2697 v4 in a single-threaded manner. To show the effectiveness of our method in the latest version of VTM, we conducted experiments using VTM21, as represented by the black curve in Figure 4.9. These experiments were conducted on the first 64 frames of CTC sequences with the RAGOP32 configuration on four QPs values of 22, 27, 32, 37.

Two metrics were used to assess the performance: BD-rate [86] and Time Saving (TS). The formula for computing TS is provided in Equation 4.6. Here, T_{Test} denotes the encoding time of the proposed method, while T_{VTM} represents the encoding time of the original VTM10 under the same conditions. The average BD-rate loss and TS are computed as the arithmetic mean and geometric mean, respectively, on four QPs values over CTC sequences as defined in [40]. In addition, sequences of class D are excluded when computing the overall average performance.

$$TS = \frac{1}{4} \sum_{q \in \{22,27,32,37\}} \frac{T_{VTM}(q) - T_{\text{Test}}(q)}{T_{VTM}(q)} \tag{4.6}$$

The acceleration performances obtained from the state-of-the-art RF-based methods could not be directly compared with our performance. There are two main reasons for this. First of all, the results of [65] and [66] are based on VTM5.0 and VTM8.0, respectively. The differences of encoder complexity among various VTM versions are not negligible as highlighted in [25], which makes it less valid to directly compare our performances with theirs. Secondly, the training dataset was generated from a subset of CTC sequences, and the results were not obtained from the entire CTC. This approach results in possible overfitting and reduces the credibility of their results. As a result, comparing our results obtained on the entire CTC with their results is not fair.

[66] is an extended and specialized work for VVC based on [65]. We have reproduced the result of [66] in VTM10 to perform an unbiased comparison between our method and RF-based method in [66]. First of all, we created a non-CTC dataset for training. Table 4.3 presents details on the composition of sequences for the dataset. For the 720p resolution, sequences are selected from [83] and sequences for other resolutions are from [82]. In the end, we generated a large dataset with $3.7e^7$ samples for the training of 17 Hor/Ver classifiers as well as $2.5e^6$ samples for the training of 4 QT/MTT classifiers.

After generating the dataset, we trained, pruned and integrated the RF classifiers in VTM10.0. This was done in a manner consistent with the original article, including the implementation of the early termination rule for TT [2].

Table 4.3 – Breakdown of sequences used to train RFs of [66]

	Resolution				
Number of videos	240p	480p	720p	1080p	4k
	50	13	10	10	5

We reproduce the result of the medium and fast speed preset of [66] in VTM10. It should be noted that the maximum MT depth is limited to 2 for the fast preset. We plot the curve of BD-rate loss and TS of our method by gradually adjusting the threshold *Thm* and *QTskip* to build six settings. The curves obtained are shown in Figure 4.9. For example, the label (T, 0.125) signifies that in this particular setting, *QTskip* and *Thm* are assigned the values True and 0.125, respectively. Our method can achieve a scalable acceleration varying from 16.5% to 60.2% with BD-rate loss ranging from 0.44% to 4.59%. Compared to the fast preset, the setting (T, 0.175) produces the same acceleration with a 0.84% lower BD-rate loss. Similarly, the setting (T, 0) reaches the same BD-rate loss while providing a 17% higher speed-up compared to the medium preset. In summary, our method generally outperforms the state-of-the-art RF-based method. It is worth mentioning that the results in VTM21 are obtained by implementing our CNN model, which was originally trained on VTM10. Consequently, it is expected to show reduced performance compared to the results in VTM10. Nonetheless, our method remains applicable and effective in the latest version of VTM.

Regarding CNN-based approaches, we compared our method with [67] and [70] in Table 4.4. The VTM version of [67] is VTM6. Thus we reimplement our method and integrate our model trained on VTM10 into VTM6 for a fair comparison within the same context. In Table 4.4, the reimplementation in VTM6 labeled as (T, 0, VTM6) reaches a slightly larger acceleration with only one-third of BD-rate loss compared to [67]. For [70], their VTM version is the same as ours, allowing for direct comparisons. Encoding with $Thm = 0.125$ yields a 40.6% reduction in the encoding time, which is similar to the acceleration achieved by the C2 configuration in [70], but with only half of its BD-rate loss. Furthermore, our method with *Thm* set to 0.2 outperformed their C3 configuration,

2. The code and dataset of reproduction is available at `https://github.com/Simon123123/vtm10_fast_dt_inter_partition_pcs2021.git`

Table 4.4 – Performance of the proposed method in comparison with reference CNN-based methods

Class	Sequence	Pan [67] (VTM6)		Tissier[70] (C2)		Tissier[70] (C3)		Ours (T, 0, VTM6)		Ours (T, 0.125, VTM10)		Ours (T, 0.2, VTM10)	
		BD-rate (%)	TS (%)	BD-rate (%)	TS (%)	BD-rate (%)	TS (%)	BD-rate (%)	TS (%)	BD-rate (%)	TS (%)	BD-rate (%)	TS (%)
A (4k)	Tango2	4.03	38.56	-	-	-	-	1.35	32.3	1.84	43.7	3.08	57.5
	FoodMarket4	1.74	46.12	-	-	-	-	0.75	29.4	0.85	55.1	1.13	53.6
	Campfire	3.17	38.23	-	-	-	-	1.49	40.7	1.83	48.5	3.22	63.2
	CatRobot1	6.45	36.84	-	-	-	-	1.31	36.5	1.45	42.6	2.67	57.6
	DaylightRoad2	5.63	35.47	-	-	-	-	1.57	39.4	2.00	45.6	3.94	57.9
	ParkRunning3	2.10	26.45	-	-	-	-	0.99	42.6	0.98	45.9	1.93	59.8
	Average	**3.85**	**36.46**	**1.84**	**47.7**	**3.06**	**59.7**	**1.25**	**37.0**	**1.49**	**45.3**	**2.66**	**58.4**
B (1080p)	MarketPlace	4.33	33.64	-	-	-	-	0.99	37.6	1.48	46.3	2.78	57.7
	RitualDance	3.55	34.17	-	-	-	-	1.75	39.9	1.91	49.4	3.91	61.8
	Cactus	5.72	29.36	-	-	-	-	1.05	37.8	1.30	44.8	2.45	58.3
	BasketballDrive	3.30	37.28	-	-	-	-	1.34	39.6	1.95	49.7	3.65	63.4
	BQTerrace	1.90	20.21	-	-	-	-	0.99	32.6	1.18	39.8	2.23	52.2
	Average	**3.76**	**30.27**	**2.21**	**46.5**	**3.09**	**58.2**	**1.22**	**37.5**	**1.56**	**46.1**	**3.00**	**58.9**
C (480p)	BasketballDrill	2.29	29.23	-	-	-	-	1.04	26.9	1.08	30.3	2.60	39.7
	BQMall	2.69	27.48	-	-	-	-	1.20	29.1	1.18	32.2	2.71	39.7
	PartyScene	2.22	20.80	-	-	-	-	0.78	31.5	0.86	33.3	2.25	43.3
	RaceHorses	3.02	26.39	-	-	-	-	0.96	32.8	1.09	34.6	2.94	45.6
	Average	**2.56**	**25.77**	**3.20**	**43.1**	**3.79**	**53.8**	**0.99**	**30.1**	**1.05**	**32.6**	**2.63**	**42.1**
D (240p)	BasketballPass	1.85	26.97	-	-	-	-	0.76	19.0	0.85	22.2	1.72	25.1
	BQSquare	1.61	14.86	-	-	-	-	0.50	17.6	0.54	19.6	1.38	22.5
	BlowingBubbles	3.03	22.15	-	-	-	-	0.33	17.2	0.45	18.9	1.12	23.4
	RaceHorses	2.92	24.20	-	-	-	-	1.04	22.8	0.85	24.4	2.11	31.2
	Average	**2.35**	**21.53**	**3.02**	**36.8**	**3.26**	**45.2**	**0.66**	**19.2**	**0.67**	**21.0**	**1.58**	**25.6**
E (720p)	FourPeople	2.31	33.77	-	-	-	-	0.95	29.5	0.90	34.3	1.65	41.2
	Johnny	3.53	35.22	-	-	-	-	0.93	22.1	1.13	27.6	2.01	32.8
	KristenAndSara	2.58	36.50	-	-	-	-	1.00	24.3	1.11	30.3	1.73	36.4
	Average	**2.81**	**35.15**	**1.45**	**38.7**	**2.2**	**49.6**	**0.96**	**25.4**	**1.04**	**30.8**	**1.79**	**36.9**
Total average		**3.18**	**30.63**	**2.33**	**43.4**	**3.12**	**54.3**	**1.14**	**33.8**	**1.34**	**40.6**	**2.60**	**52.2**

achieving a 0.52% lower BD-rate loss at the same level of acceleration. In conclusion, our method consistently outperforms all state-of-the-art methods.

It is important to note that the level of acceleration can vary depending on different sequence classes (e.g. resolution), which is consistent with other CNN-based methods. As discussed in [33], CTUs that exceed the picture boundary are referred to as partial CTUs. These partial CTUs require a different partition search scheme compared to regular CTUs. Consequently, the encoding of partial CTUs are not accelerated since the CNN-based approaches are not applicable to them. Generally, the proportion of the frame region occupied by partial CTUs is larger for lower resolutions, resulting in less acceleration when fast partitioning approaches are used on smaller resolutions. This could partially explain the limited acceleration observed in class D which was excluded from the overall performance calculation. More specifically, our method tends to perform better on higher resolutions (e.g. class A and class B) while achieving less acceleration than state-of-the-art methods on lower resolutions (e.g. class C, class D and class E). Investigating and improving this aspect could be a focus of future work.

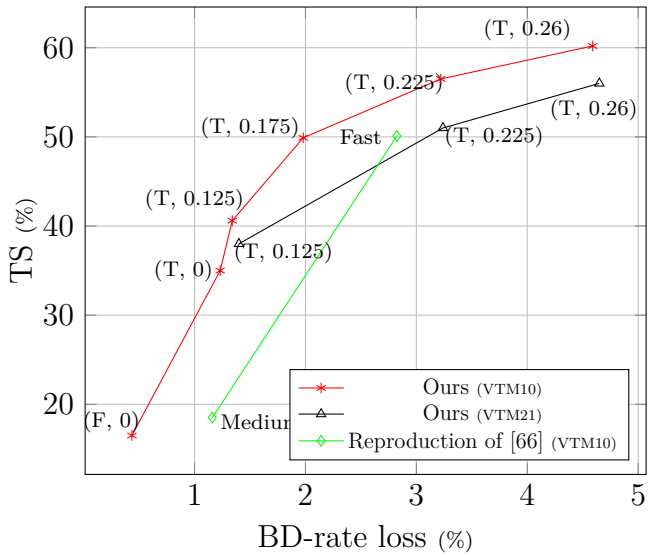


Figure 4.9 – Comparison of performances between proposed method and reproduction of [66].

4.3.3 Complexity Analysis

Machine learning-based fast partitioning methods may not be suitable for alternative implementations of the same codec. For example, VVenc [94] is a fast implementation of VVC. In the All Intra configuration, VTM10.0 is reported to be 27 times more complex compared to VVenc with fast preset, as mentioned in [95]. The overall complexity of the CNN-based method presented in [63] accounts for only 2.34% of the encoding time of the VTM10 encoder. However, when this method is implemented in VVenc without any adjustments, its overhead increases to about 67% of the encoding time with the fast preset, which means that this method is not directly applicable to VVenc. Consequently, it is crucial to develop a lightweight method to ensure its applicability across different implementations. Furthermore, lightweight methods do not require parallel execution, enhancing the cost-effectiveness of such solutions.

Table 4.5 – Overhead of our method (Unit: %)

	240p	480p	720p	1080p	4k	Average
CNN	0.23	0.37	0.99	0.90	0.84	0.60
Preprocess	0.24	0.41	1.15	0.81	0.86	0.62

As a result, we conducted a complexity analysis of our method to compare it with

the state of the art. The overhead of a machine learning-based method typically consists of three components: preprocessing time, inference time, and postprocessing time. The post-processing of our method is integrated into the VVC partitioning process and introduces minimal overhead to the encoding process. However, preprocessing is necessary to compute the MS-MVF as model input. Table 4.5 provides the complexity of the preprocessing and the inference of CNN related to the encoding of the anchor VTM10. The last column corresponds to the geometric average of complexity for sequences from class A to E (including class D). Based on experimental results, the CNN inference time on a CPU accounts for only 0.60% of the total encoding time. Our approach consumes only 1.21% of the total encoding time, underscoring its lightweight nature.

Another important metric for evaluating the complexity of the model is its floating point operations (FLOPs). Our model has a FLOPs value of $1.12e^6$. In comparison, the FLOPs of the model in [62] is approximately $1.1e^9$[61]. [61] employs a pruned ResNet-18 as the backbone with $9e^7$ FLOPs, and [70] utilizes the pretrained MobileNetV2 with $3.14e^8$ FLOPs. Our model is hundreds of times lighter than these methods. The lightweight nature of our proposed approach facilitates its adaptation to faster encoders.

4.4 Conclusion

In this study, we propose a machine learning-based method to accelerate VVC inter partitioning. Our method leverages a novel representation of the partition structure QTMT based on the partition path, which consists of QTdepthMap and MTsplitMaps. Our work is structured as follows. Firstly, we have created a large-scale inter-partition dataset. Secondly, we train a novel U-Net-based model that takes MS-MVF as input to predict the partition paths of CTU. Thirdly, we develop a scalable acceleration algorithm based on thresholds to utilize the output of the model. Finally, we speed up the VTM10 encoder under RAGOP32 configuration by 16.5%∼60.2% with a BD-rate loss of 0.44%∼4.59%. This performance surpasses state-of-the-art methods in terms of coding efficiency and complexity trade-off. Notably, our method is one of the most lightweight methods in the field, making it adaptable to faster codecs.

For future work, we intend to investigate how video resolution influences partitioning acceleration, aiming to boost the speed-up of our method on lower resolutions. Furthermore, there is still acceleration potential lying in the selection of inter coding modes at the CU level, as discussed in [96]. An extension of our approach could be the incorporation

of fast inter coding mode selection algorithm into our method to further accelerate the inter coding process.

Chapter 5

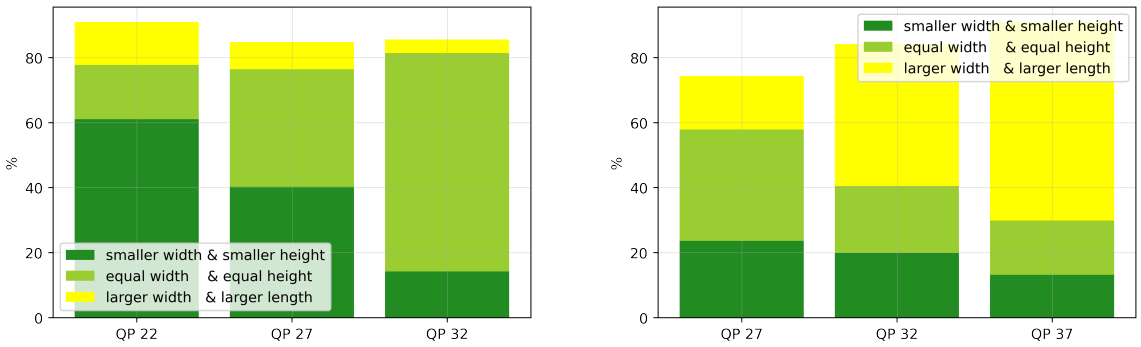
Preparing VVC for Streaming: A Fast Multi-Rate Encoding Approach

5.1 Introduction

In Chapters 3 and 4, we introduce two CNN-based fast partitioning methods for VVC, both of which are integrated into VTM. In this chapter, our focus shifts to the application of VVC in streaming scenarios. As we adopt more advanced video codecs to deliver higher quality videos, one common challenge is the increased computational time associated with these codecs. This challenge becomes particularly critical in video streaming applications, where we must encode the same content at multiple bitrates to accommodate varying network conditions experienced by users. Consequently, various fast multi-rate encoding methods have been proposed to utilize the encoding at a reference bitrate to accelerate encodings at other bitrates. While these methods, discussed in Section 2.2, have demonstrated significant performance in the context of HEVC, the exploration of fast multi-rate encoding techniques in VVC remains relatively unexplored.

In this chapter, we aim to bridge this gap by applying fast multi-rate encoding to VVenC 1.7.0, a fast and efficient software implementation of VVC. By leveraging this approach, we investigate the potential benefits and performance improvements of fast multi-rate encoding in the VVC domain.

The rest of this chapter is organized as follows. Section 5.2 outlines our proposed fast multi-rate encoding approach specifically tailored for VVC. The experimental results are presented in Section 5.3. Finally, Section 5.4 concludes this chapter, summarizing the key findings and contributions.



(a) Reference = QP 37

(b) Reference = QP 22

Figure 5.1 – Average percentage of the CUs with CU height and width larger, equal to, or smaller than the reference encoding.

5.2 Proposed Method

5.2.1 Multi-rate Encoding Opportunities

Multi-rate VVC representations (*i.e.* encoding) of a video content can carry a significant level of redundancies. To demonstrate this, we encoded 18 sequences with RA coding configurations at QPs of 22, 27, 32, 37. These sequences include six from class A of the VVC CTC [40] and 12 from the Inter4K dataset [97]. To select 12 sequences out of the 1000 sequences available in the Inter4K dataset, we compared their spatial and temporal complexities using Video Complexity Analyzer (VCA) [98]. Based on this comparison, we divided them into four clusters as follows: low spatial and low temporal complexity, low spatial complexity and high temporal complexity, high spatial complexity and low temporal complexity, and high spatial complexity and high temporal complexity. From each cluster, we then selected three sequences randomly for further evaluation (sequences 10, 47, 56, 176, 186, 339, 497, 500, 606, 646, 666, 754). QP 22 and 37 were used as reference for comparisons of CU sizes in non-reference QPs. Figure 5.1 (a) and (b) show, respectively, what percentage of coded co-located CUs have *both* CU width and height smaller than the reference of QP 37 or larger than the reference of QP 22.

When the reference encoding is set to QP 37, it is observed that a smaller portion of CUs in the dependent encodings have larger width and height sizes compared to their co-located CUs in the reference encoding (yellow bars in Figure 5.1(a)). The closer the QP of the dependent encoding to the reference encoding, the larger the portion of CUs that have the same size as the co-located CU in the reference encoding. Therefore, when

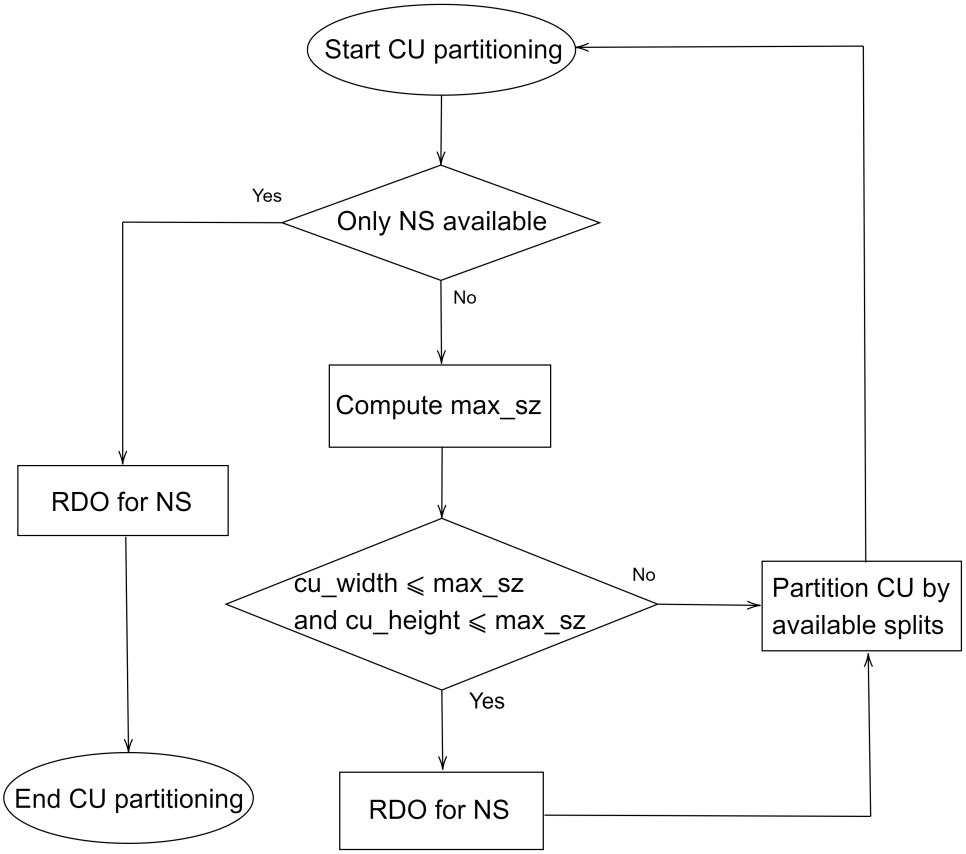


Figure 5.2 – Diagram of CU size based fast partitioning.

the reference encoding is set to QP 37, RDO can be skipped for co-located CUs that have a width and height larger than the reference CU. On the other hand, when the reference encoding is set to QP 22, a smaller portion of CUs in the dependent encoding have smaller width and height sizes compared to their co-located CUs in the reference encoding (dark green bars in Figure 5.1(b)). Therefore, when the reference encoding is set to QP 22, RDO can be skipped for co-located CUs that have a smaller width and height than the reference CU. Please note that the cumulative percentages of CUs do not reach 100% because the cases where CUs have smaller widths but larger heights or larger widths but smaller heights have not been considered in the evaluations.

5.2.2 A Fast Multi-rate Encoding Scheme for VVC

Based on the experiments presented previously, a method is proposed to exploit the redundancy in the CU sizes with respect to a reference-coded bitrate. Selecting the appropriate reference encoding for reusing its information to accelerate the encoding of

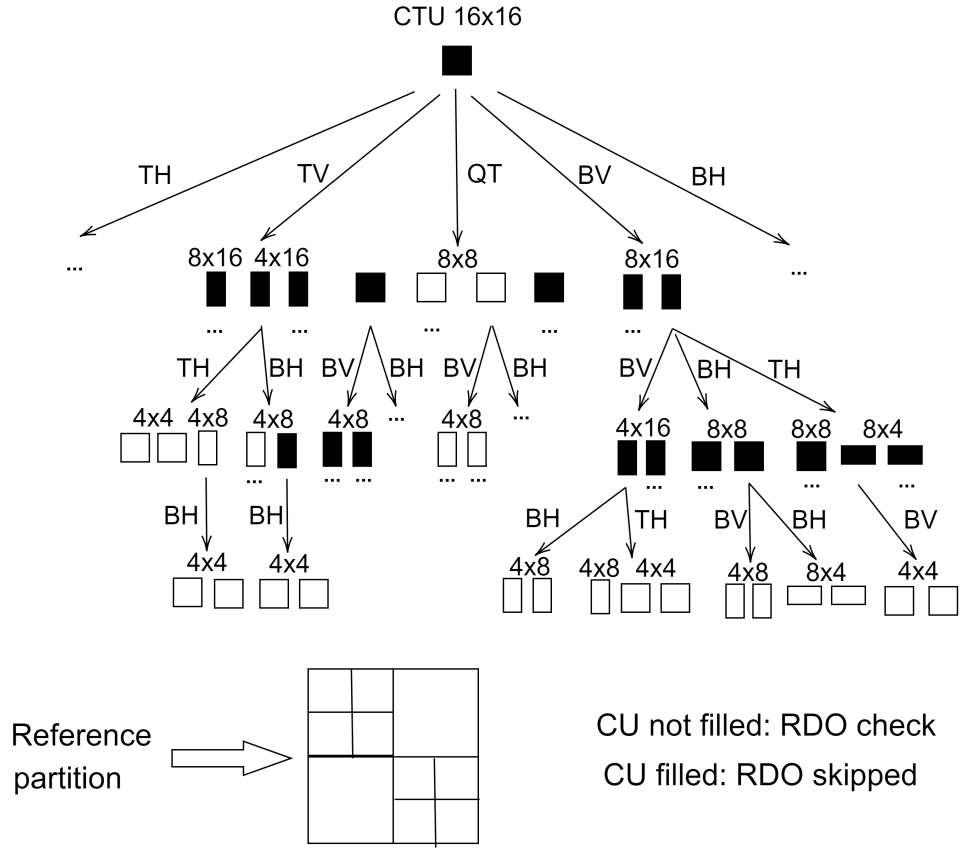
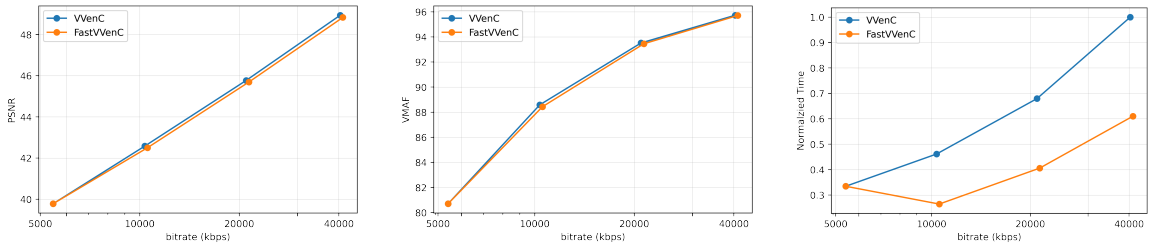


Figure 5.3 – Fast approach involved partition search.

dependent representations is indeed a challenging task and can vary depending on the specific application. In the context of live video streaming, it is commonly observed that the highest quality representation requires the longest encoding time, followed by lower quality representations as the quality decreases. Given the significant impact of encoding time on the latency of live video streaming, it is advantageous to select a representation with a lower encoding time (lower quality) as a reference for accelerating the encoding of representations with higher encoding time (higher quality). By doing so, the maximum and overall encoding process can be expedited, reducing the latency and enhancing the potential for transitioning towards real-time streaming with VVC. Therefore, in this chapter, we propose selecting the representation with the lowest encoding time (QP 37) as the reference encoding. By reusing its information, we aim to accelerate the encoding process of the dependent representations (QP 32, QP 27, and QP 22).

When a reference CTU of 128×128 is encoded at QP 37, its size map is calculated to contain the maximum width and height of CUs in the reference partition. Since the



(a) (b) (c)

Figure 5.4 – The comparison between the standalone VVenC and the proposed fast multi-rate encoding.

smallest CU size is 4×4, each 4×4 sub-block is assigned the maximum width and height of the CU it belongs to. This process results in a map of size 32×32 for each CTU. With a reference encoding of QP 37 and a reference size map of 32×32, the algorithm for accelerating the encoding of co-located dependent CUs is illustrated in Figure 5.2.

Firstly, possible split types of current CU are investigated. If no split types other than NS are available, the fast partitioning approach will not be applied to current CU. The encoding of current CU is directly processed and the CU partitioning is terminated. If other split types are possible, we obtain *max_sz* in Figure 5.2 by getting the maximum value of elements of the size map inside current CU. If the width and height of the current CU are both smaller or equal to *max_sz*, we do the RDO for current CU. Otherwise, the encoding of CU is skipped and the CU is further split to sub-CUs. A part of RDOs of CU is leaped by applying our fast partitioning method.

To better illustrate the effect of the acceleration achieved by our algorithm, we present the partition search results with our fast approach applied in Figure 5.3. To simplify the partition tree, we have set the CTU size to 16×16 and the QT split is disabled for 8×8 CU. Meantime, the maximum partitioning depth is 3. For better visualization, repetitive branches are not presented in the figure. CUs with RDO skipped are filled with black. If we proceed with a larger CTU size, the proportion of skipped CUs would be larger.

5.3 Experimental Results and Analyses

To evaluate the proposed algorithm, we performed encodings using the VVenC with both the standalone and modified versions of the encoder on the same set of sequences and with the same encoding configuration as mentioned in Section 5.2.2. For each encoding, we recorded the bitrate, and encoding time, and computed objective metrics,

including PSNR and VMAF. Figure 5.4 shows the PSNR vs. bitrate, VMAF vs. bitrate, and normalized time vs. bitrate for the video sequence 754 from Inter4K dataset as an example. It is observed that the rate-distortion performance remains approximately the same while the encoding time is significantly reduced for the dependent representations. From Figure 5.4-(c), it can be concluded that not only is the overall encoding time of all representations reduced, but also the maximum encoding time is decreased, which is equivalent to the encoding latency. This is highly beneficial in live video streaming as it ensures efficient and timely processing of the video data. In this example, using the proposed method, the maximum encoding time is reduced by 40%, as can be seen in Figure 5.4-(c). The average BD-PSNR and BD-VMAF are -0.21 and -0.43, respectively, indicating that the proposed fast multi-rate encoding method causes a slight decrease in video quality compared to the standalone VVenC at equivalent bitrates. At the same time, the average increase in bitrate in equivalent quality is 5.11% and 4.82% using PSNR and VMAF as objective metrics, respectively. However, the reduction in video quality is minimal, and the benefits of significantly reduced encoding time make the proposed method a valuable option for various applications, especially in live video streaming scenarios where real-time performance is crucial. The proposed fast multi-rate encoding results in an average encoding time reduction of 40% for all dependent representations. Additionally, it achieves an average maximum encoding time reduction of 39%, indicating that the encoding latency is reduced by 39% when using this approach.

5.4 Conclusion

This chapter proposes a fast multi-rate encoding approach for VVC aimed at reducing the encoding time for video streaming applications. The approach involves encoding a reference representation using a standalone VVC encoder. The information obtained from the reference representation is then utilized to accelerate the encoding process for the dependent representations. To ensure compatibility with parallel processing, the representation with minimal encoding time is selected as the reference encoding. This reference representation is then utilized to reduce the encoding time of the longer encoding time representations, aiming not only to reduce the overall encoding time but also to minimize the maximum encoding time, which is equivalent to encoding latency. When a CTU is encoded in the reference representation, the RDO process is skipped in the co-located CTUs in dependent representations for CUs that have larger width and height

compared to the reference CTU. The experimental results demonstrate that the proposed method achieves an average 40% encoding time reduction, while the quality drop, on average, is only 0.43 VMAF when using VVenC, an optimized VVC implementation.

Chapter 6

Conclusion

6.1 Summary

The latest VVC standard finalized in 2020 outperforms its predecessor HEVC by around 40% bitrate reduction for the same quality. Nevertheless, the gain of coding performance is at the expense of largely increased encoding complexity. For example, with the RA configuration, the encoding time of VVC is 8 times longer than that of HEVC. The substantial increase in the complexity of the coding poses a challenge to the widespread deployment of the VVC codec. This increase in complexity is partly due to the adoption of a complex partition structure. In the literature, numerous fast partitioning methods have been suggested, showcasing their effectiveness in reducing encoding complexity. Moreover, the acceleration of inter coding is overlooked when compared to that of intra coding. As a result, the focus of this thesis is directed towards reducing the complexity of inter coding in VVC through the utilization of fast partitioning methods.

In this context, this thesis introduces three fast partitioning methods that involve different implementations of VVC. Two CNN-based fast partitioning methods are introduced in Chapters 3 and 4. In Chapter 3, our first contribution involves utilizing a lightweight CNN to predict the QT depth map to remove a part of MT partitions from the search space of partitioning. Experiments shows that this work can reduce the complexity by one third with slight coding loss, outperforming other CNN-based methods. Our second contribution, presented in Chapter 4 significantly improves our first contribution. A novel CNN structure and acceleration algorithm are proposed in this work. In addition to the QT depth map, MT decisions at various levels are predicted at the same time, corresponding to a full and explicit representation of QTMT partition of VVC. Higher acceleration is reached than our first contribution since more partitions are removed from the search space owing to the complete representation. Its acceleration performance exceeds the existing stat-of-the-art methods. For a non machine learning contribution in Chapter 5, we target a more specific application scenario. In steaming scenario, the same content

need to be encoded at multiple bitrates (*i.e.* representations). By utilizing the partitioning structure of the reference representation, we accelerate the encoding process of the remaining representations, known as dependent representations. Through the utilization of our proposed rapid partitioning technique, an acceleration up to 40% is achieved in VVenC, accompanied by a minor decline in encoding quality. The last stage of this thesis is a statistical analysis presented in Annex, which offers a global view of the complexity distribution of inter coding.

With results superior to the state of the art, first two contributions enrich the research in the field of fast inter coding of VVC by machine learning method. Furthermore, our last contribution is the first work on fast multi-rate encoding for VVC. Nonetheless, opportunities for further improvement of these contributions are yet to be explored, as detailed in the following section.

6.2 Perspectives

Even though our proposed approaches significantly accelerate the inter coding process of VVC, the resulting complexity reduction is still insufficient for achieving real-time encoding. Therefore, it is conceivable to combine the fast partitioning method with other fast coding algorithms.

6.2.1 Machine-learning-based fast partitioning

The nature of fast partitioning method is to reduce the number of CUs checked in RDO, whereas numbers of coding modes checked for each CU is untouched. Our work in Annex gives an overview of inter coding modes in RDO. As a result, an interesting extension of fast partitioning method for inter coding is to incorporate the fast inter coding mode selection algorithm into our method to further speed up the inter coding process.

6.2.2 Fast multi-rate encoding

In Chapter 5, the partition information of encoding at lower bitrate is utilized to speed up the the encoding at higher bitrate. The correlation of other coding information between different representations are not exploited in this work. Hence, we have the potential to achieve even more substantial acceleration by using the coded partition in

conjunction with other coding information (such as coded motion vectors and coded inter coding modes).

Annex: Statistical Analysis of Inter Coding

Introduction

As stated in the Introduction of this manuscript, it is vital to develop acceleration methods for VVC to largely reduce the encoding complexity while preserving the majority of encoding efficiency. To design acceleration methods, it is essential to understand the complexity distribution and statistics of encoding decisions inside a VVC encoder (*e.g.* VTM). The papers mentioned in Section 2.4 present a complexity analysis for VVC. However, this contribution is the first to provide an analysis of CU sizes and decisions on coding modes for inter coding in VVC.

In this chapter, a statistical analysis of the RDO process in inter coding of VTM 15.0 is presented. The main focus is on the statistics of two factors: CU sizes and inter coding modes. The goal is to provide useful information for related works aiming at speeding up inter coding in VVC. In Section A.2, all statistical observations are presented, which are later analyzed and concluded in Section A.3.

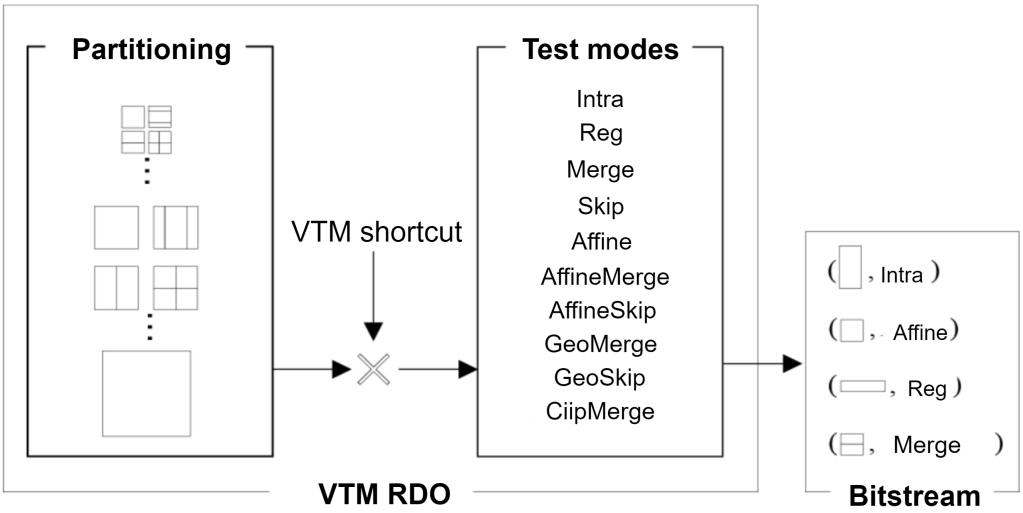


Figure 1 – High-level view of the RDO process involving partitioning, test modes and possible VTM shortcuts.

Statistical Analysis

As depicted in Section 1.3.5, the RDO process involves searching for the best combination of coding decisions that achieve the best trade-off between bit rate and distortion. With trivial coding decisions (*e.g.* choice of transform, MV representation mode) ignored,

RDO consists of finding the optimal combination of coding modes and CU sizes. Consequently, in this chapter, we analyze the statistics of both CU sizes and coding modes collectively.

As presented in Figure 1, various CU sizes are the result of partitioning in the RDO process. In VVC, a CU size is authorised if its widths and heights are any power of two between 4 and 128, except for sizes 128×4, 128×8, 128×16 and 128×32. There are in total 32 possible CU sizes in VVC as shown in Figure 1.13.

The hash-based inter prediction and palette modes are coding modes disabled in the JVET CTC. Therefore, they are not considered in the analysis of this paper. For each CU, ten coding modes are available according to Section 1.3.4.2. These modes include: *Intra*, *Reg*, *Merge*, *Skip*, *Affine*, *AffineMerge*, *AffineSkip*, *GeoMerge*, *GeoSkip*, *CiipMerge*.

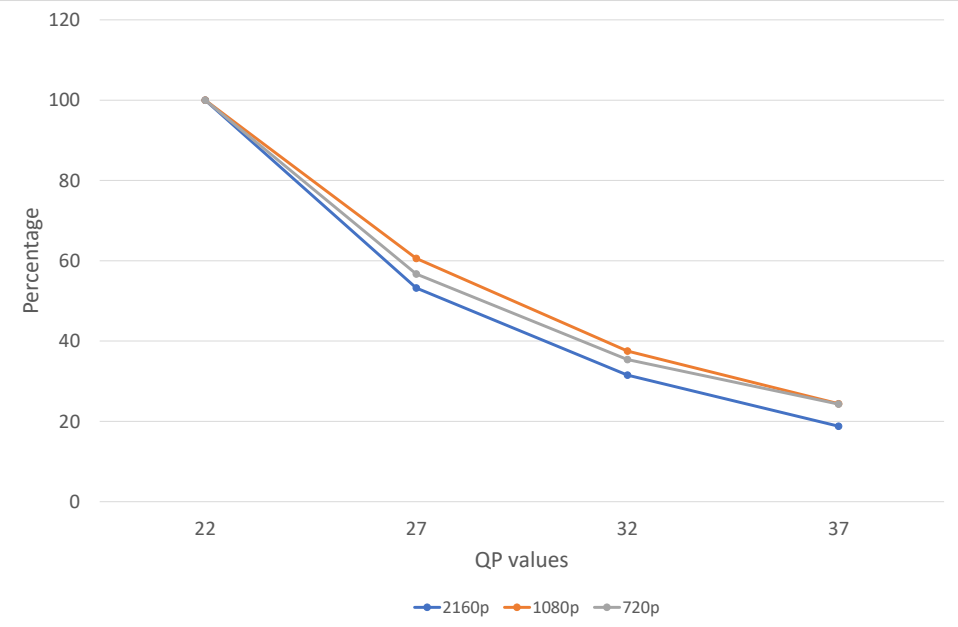


Figure 2 – Encoding time in different QPs comparing to QP 22

Although VVC is computationally expensive, the JVET group has already adopted diverse shortcuts or conditional early exits as presented in [99] for the VTM. We have deactivated existing shortcuts in VTM 15.0 and evaluated its performance. As a result, the complexity increases by 138%. Furthermore, the performance of the tested encoder (*i.e.* without shortcuts) is 0.76% better than that of the reference encoder (*i.e.* with shortcuts), in terms of BD-rate. This trade-off might be interpreted as an indicator that the shortcuts in VTM are efficient in terms of identifying useless tests and partitioning

depths. Many of shortcuts are based on history of the tested split modes. Nonetheless, aspects of CU sizes and coding modes are overlooked.

Our main purpose in the following analysis is to find CU sizes and / or coding modes with relatively high complexity occupation and low selection rate in the RDO process. From the perspective of encoder acceleration, CU sizes or coding modes with higher complexity portion and significantly lower selection rate are more favorable to the design of acceleration rules based on block size/coding mode. The size or mode with larger complexity portion has more potential in accelerating. Lower selection rate indicates it is less likely to make wrong decisions when skipping the RDO of current CU size or mode.

All our experiments and analyses are performed on the first 64 frames of the CTC sequences in the RAGOP32 configuration, in which intra frames are excluded. Exceptionally, Figure 2 is based on sequences in Class A, B, E of CTC.

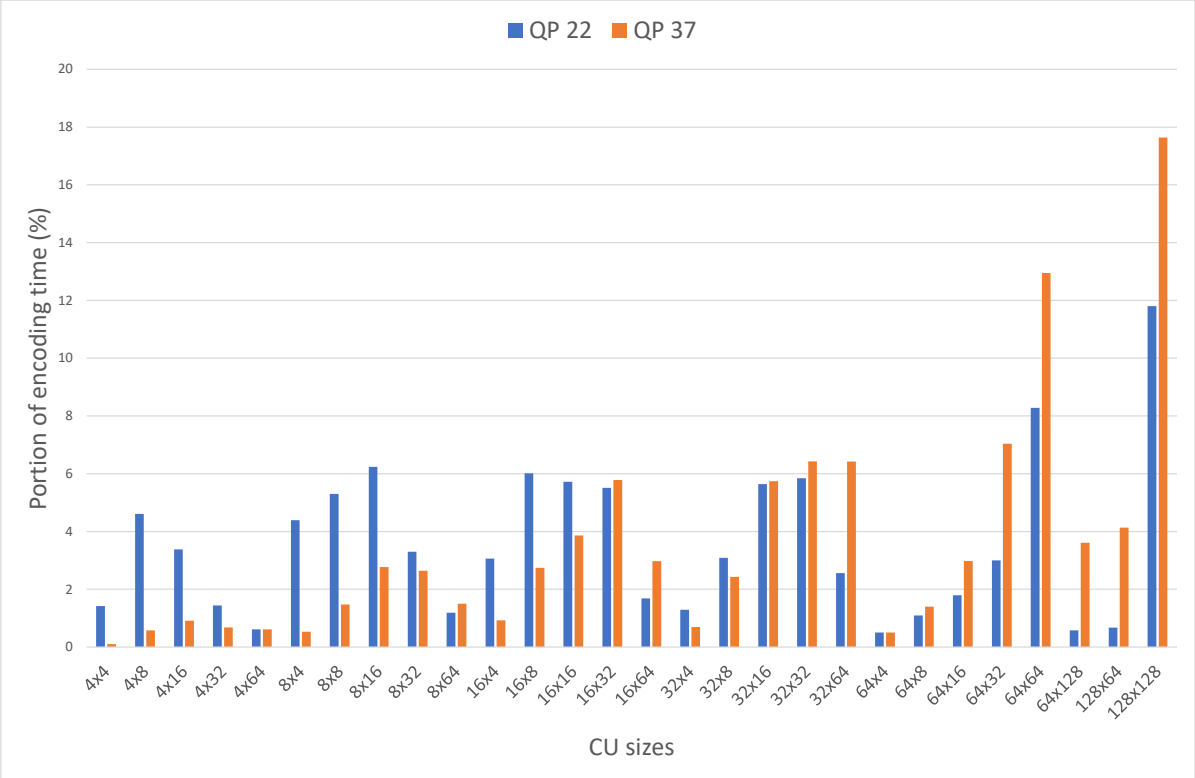


Figure 3 – Complexity distribution for block sizes in QP 22 and QP 37

The encoding complexity for Chrominance channel only accounts for a small part comparing to Luminance channel. Thus we focus on Luminance channel in the remaining of the paper. From a high-level view, the encoding complexity of VTM depends significantly

on the selected QP. Particularly, larger QP values tend to have faster encoding with the VTM. Figure 2 is obtained by measuring the encoding times of sequences of resolution 2160p, 1080p, and 720p in QP 22, 27, 32 and 37. Then the average ratio is calculated between the encoding time of each QP and that of QP 22 is calculated. It shows that the encoding time at QP 22 could be five times as much as at QP 37.

Figure 3 shows the percentage distribution of the encoding time spent on different block sizes in QP 22 and QP 37. In addition to the fact that the overall encoding time is higher for QP 22, it can be observed that a relatively higher portion of the time in QP 22 is passed on smaller block sizes. This could be explained in part by the existing shortcuts in VTM disallowing excessively small blocks in QP 37. We could declare that larger block sizes are in general more crucial to speeding up the partitioning process, especially block 64x64 and 128x128 which take in total from 20% complexity in QP 22 to 30% in QP 37.

In another test, the selection percentages of different block sizes are computed. This metric is defined as the ratio between the total number of times it is selected and the total number of times a block size is tested. Figure 4 shows the values of this metric in QP 22 and 37. As we can see from this figure, larger CU sizes correspond to larger selection rates compared with smaller blocks. Another noteworthy phenomenon is that the selection rate of 128x128 increases dramatically from 14% in QP 22 to 37% QP 37.

QP 22

H\W	4	8	16	32	64	128
4	0.25	0.4	0.49	0.4	0.27	NA
8	0.5	1	0.84	0.86	0.64	NA
16	0.77	1.15	1.64	1.58	2.21	NA
32	0.78	1.48	2.03	3.91	4.64	NA
64	0.48	0.9	1.71	3.89	11	5.61
128	NA	NA	NA	NA	5.87	15.73

QP 37

H\W	4	8	16	32	64	128
4	0.25	0.29	0.42	0.38	0.14	NA
8	0.35	0.61	0.7	0.8	1.35	NA
16	0.68	0.97	1.31	1.33	4.96	NA
32	0.73	1.47	1.67	3.11	4.24	NA
64	0.47	1.49	2.73	3.84	8.5	4.51
128	NA	NA	NA	NA	5.25	41.5

Figure 4 – Selection rate for different block sizes

Combining the above figures, it is observed that CU sizes such as 16x8, 8x16, and 16x16 are sizes with low selection rate and high complexity. For example, 16x16 CUs have the same level of complexity, while its selection rate is half of 32x32 CUs in QP 22.

To take one step further in the statistical analysis of inter coding, we present how different inter coding modes are involved in RDO search. The first experiment in Figure 5 presents the distribution of the encoding time at inter coding mode level in QP 22 and QP 37.

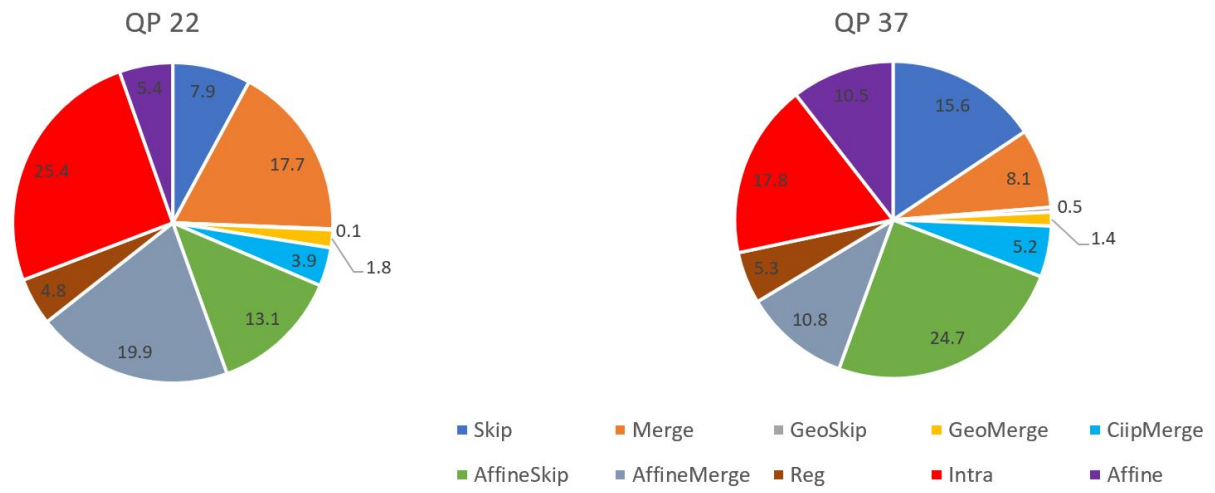


Figure 5 – Pie chart of complexities of inter coding modes

In general, *Intra*, *AffineMerge*, *AffineSkip*, *Merge*, and *Skip* are main contributors to the encoding time. Figure 6 provides selection rates as the ratio between number of selected inter coding modes and number of tested modes. We could observe that the three modes, namely, *AffineMerge*, *AffineSkip*, and *Merge* have relatively low selection rates, even though they collectively account for nearly half of the complexity.

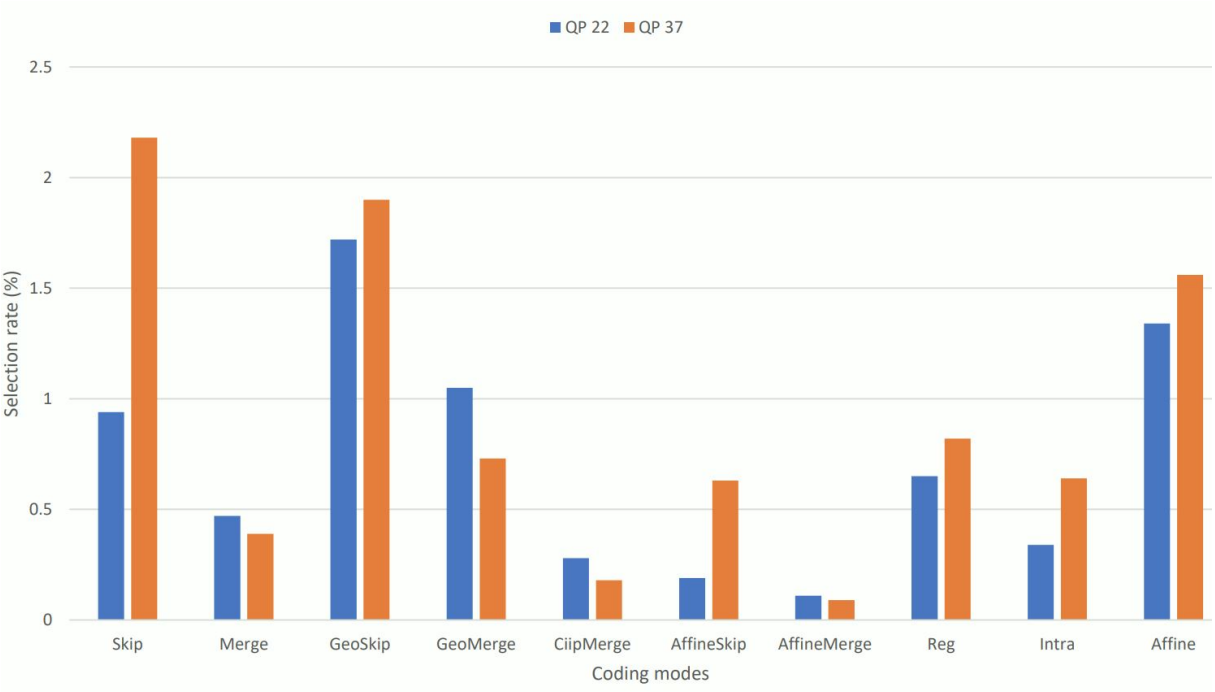


Figure 6 – Selection rate of inter coding modes

Figure 7 shows the distribution of inter modes for encoded CUs of different sizes. The fact that the aforementioned three coding modes are less chosen could also be proved by this figure. We find that the number of *Skip* is dominant for most CU sizes and that the number of these three modes is relatively small, which is consistent with Figure 6. From Figure 7 we also observe that smaller CUs tend to be encoded with intra mode. Another remark is that the skip modes (*i.e.* *Skip*, *AffineSkip*, and *GeoSkip*) are more frequently selected for larger CUs. It is probably because the residual of larger CUs is more expensive to be encoded. In addition, *Merge* mode have higher chance to be selected in smaller QP which is in contrast to *AffineSkip* and *GeoSkip*. For some CU sizes, we could make shortcuts or conditions for early termination for RDO of inter modes which are rarely selected to speed up the encoding process, such as *AffineMerge* mode with merely 1.9% selected for bloc 128x128.

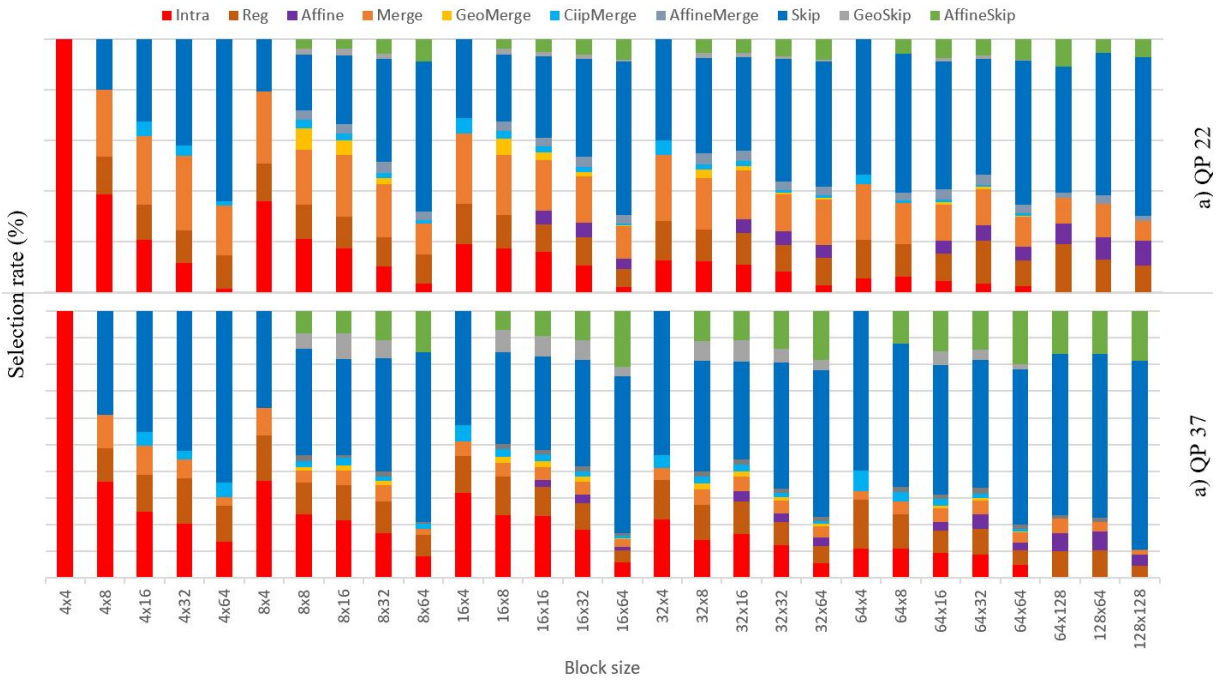


Figure 7 – Stacked chart of selected inter modes in different block sizes

Conclusion

In this study, complexity analysis of CU sizes and inter coding modes has been combined with selection rate analysis. From the perspective of CU size, CU sizes with high

complexity generally correspond to a high selection rate. CU sizes 128x128 and 64x64 are responsible for one third of the complexity. In addition, CU sizes like 16x8, 8x16, 16x16 exhibit relatively low selection rate while requiring a significant share of the overall complexity. Therefore, these CU sizes are relevant targets for acceleration algorithms. From coding mode perspective, *AffineMerge*, *AffineSkip*, and *Merge* tend to be less likely to be selected. Thus, shortcuts dedicated to adaptively skip these coding modes might be promising. Shortcuts on coding modes and partitioning acceleration method are in different scopes. The former focus on reducing number of CU for RDO. The latter speeds up RDO for CU of certain sizes. The combination of these two could lead to a larger speed-up of encoding.

Personal Publications

[1] Y. Liu, M. Abdoli, T. Guionnet, C. Guillemot, and A. Roumy, « Statistical Analysis of Inter Coding in VVC Test Model (VTM) », *in*: *2022 IEEE International Conference on Image Processing (ICIP)*, 2022, pp. 3456–3459, DOI: 10.1109/ICIP46576.2022.9897595.

[2] Y. Liu, M. Abdoli, T. Guionnet, C. Guillemot, and A. Roumy, « Light-weight cnn-based vvc inter partitioning acceleration », *in*: *2022 IEEE 14th Image, Video, and Multidimensional Signal Processing Workshop (IVMSP)*, IEEE, 2022, pp. 1–5.

[3] Y. Liu, H. Amirpour, M. Abdoli, C. Timmerer, and T. Guionnet, « Preparing VVC for Streaming: A Fast Multi-Rate Encoding Approach », *in*: *2023 IEEE Visual Communications and Image Processing (VCIP)*, IEEE, 2023.

BIBLIOGRAPHY

[1] sandvine, *The Global Internet Phenomena Report January 2023*, 2023, URL: https://www.sandvine.com/phenomena (visited on 01/09/2023).

[2] B. Bross, Y. Wang, Y. Ye, S. Liu, J. Chen, G. Sullivan, and J. Ohm, « Overview of the Versatile Video Coding (VVC) Standard and its Applications », *in*: *IEEE Transactions on Circuits and Systems for Video Technology* 31.*10* (2021), pp. 3736–3764, DOI: 10.1109/TCSVT.2021.3101953.

[3] Cisco, *Cisco Annual Internet Report (2018–2023) White Paper*, 2020, URL: https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.html (visited on 03/09/2020).

[4] Ericsson, *Ericsson Mobility Report*, 2023, URL: https://www.ericsson.com/en/reports-and-papers/mobility-report (visited on 01/09/2023).

[5] J. Urban, *Time spent for online video*, 2017, URL: https://www.statista.com/statistics/611707/online-video-time-spent/ (visited on 03/15/2023).

[6] J. Urban, *Energy-efficient Cloud Computing Technologies and Policies for an Eco-friendly Cloud Market*, 2020, URL: https://digital-strategy.ec.europa.eu/en/library/energy-efficient-cloud-computing-technologies-and-policies-eco-friendly-cloud-market (visited on 11/09/2020).

[7] Netflix, *Environmental Social Governance Report 2022*, 2022, URL: https://downloads.ctfassets.net/4cd45et68cgf/7rnC6zK537cM8zAGrXA90E/3c654a2d0023a4dac26a20b2fff3
Netflix_2022-ESG-Report-FINAL.pdf (visited on 01/09/2022).

[8] A. Tissier, A. Mercat, T. Amestoy, W. Hamidouche, J. Vanne, and D. Menard, « Complexity reduction opportunities in the future VVC intra encoder », *in*: *2019 IEEE 21st International Workshop on Multimedia Signal Processing (MMSP)*, IEEE, 2019, pp. 1–6.

[9] *Global number of digital video viewers 2019-2023*, 2023.

[10] *Time Spent Watching Online Video Expanding To 100 Minutes Daily, Ad Budgets Set To Follow*, 2019.

[11] J. Urban, *How Chroma Subsampling Works*, 2017, URL: https://blog.biamp.com/how-chroma-subsampling-works/ (visited on 09/14/2017).

[12] M. Communication, *Happy World Standards Day 2020 - Protecting the planet with standards*, 2020, URL: https://multimediacommunication.blogspot.com/2020/10/happy-world-standards-day-2020.html (visited on 10/14/2020).

[13] Coconut, *The State of Video Codecs in 2023: A Comprehensive Overview*, 2023, URL: https://www.coconut.co/articles/the-state-of-video-codecs-in-2023-a-comprehensive-overview#:~:text=H.264%2FAVC%20(Advanced,with%20various%20devices%20and%20platforms..

[14] J.-R. Ohm, G. J. Sullivan, H. Schwarz, T. K. Tan, and T. Wiegand, « Comparison of the Coding Efficiency of Video Coding Standards—Including High Efficiency Video Coding (HEVC) », *in*: *IEEE Transactions on Circuits and Systems for Video Technology* 22.*12* (2012), pp. 1669–1684, DOI: 10.1109/TCSVT.2012.2221192.

[15] M. Wien, *Versatile Video Coding – Video Compression beyond HEVC: Coding Tools for SDR and 360° Video*, 2023, URL: https://www.slideshare.net/foerderverein/versatile-video-coding-video-compression-beyond-hevc-coding-tools-for-sdr-and-360-video.

[16] S. De-Luxán-Hernández, V. George, J. Ma, T. Nguyen, H. Schwarz, D. Marpe, and T. Wiegand, « An Intra Subpartition Coding Mode for VVC », *in*: *2019 IEEE International Conference on Image Processing (ICIP)*, 2019, pp. 1203–1207, DOI: 10.1109/ICIP.2019.8803777.

[17] J. Li, B. Li, J. Xu, and R. Xiong, « Intra Prediction Using Multiple Reference Lines for Video Coding », *in*: Apr. 2017, pp. 221–230, DOI: 10.1109/DCC.2017.59.

[18] R. R. Schultz and R. L. Stevenson, « Extraction of high-resolution frames from video sequences », *in*: *IEEE Transactions on Image Processing* 5.*6* (1996), pp. 996–1011, DOI: 10.1109/83.503915.

[19] A. Alshin, E. Alshina, and T. Lee, « Bi-directional optical flow for improving motion compensation », *in*: *28th Picture Coding Symposium*, 2010, pp. 422–425, DOI: 10.1109/PCS.2010.5702525.

[20] T. Nguyen, B. Bross, P. Keydel, H. Schwarz, D. Marpe, and T. Wiegand, « Extended Transform Skip Mode and Fast Multiple Transform Set Selection in VVC », *in*: *2019 Picture Coding Symposium (PCS)*, 2019, pp. 1–5, DOI: 10.1109/PCS48520.2019.8954540.

[21] M. Karczewicz et al., « VVC In-Loop Filters », *in*: *IEEE Transactions on Circuits and Systems for Video Technology* 31.*10* (2021), pp. 3907–3925, DOI: 10.1109/TCSVT.2021.3072297.

[22] *Versatile Test Model*, 2022.

[23] J. Chen and E. Alshina, *Algorithm Description for Versatile Video Coding and Test Model 1 (VTM 1)*, document MPEG-N17670, MPEG, 2018.

[24] Y. Y. Adrian Browne and S. H. Kim, *Algorithm Description for Versatile Video Coding and Test Model 20 (VTM 20)*, document JVET-AD2002, JVET, 2023.

[25] G. Sullivan, *Versatile Video Coding (VVC) Delivers: Coding Efficiency and Beyond, DCC*, 2021, URL: https://www.cs.brandeis.edu/~dcc/Programs/Program2021KeynoteSlides-Sullivan.pdf (visited on 03/25/2021).

[26] *VVenC Fraunhofer Versatile Video Encoder*, 2021.

[27] N. Ahmed, T. Natarajan, and K. Rao, « Discrete Cosine Transform », *in*: *IEEE Transactions on Computers* C-23.*1* (1974), pp. 90–93, DOI: 10.1109/T-C.1974.223784.

[28] M. Koo, M. Salehifar, J. Lim, and S.-H. Kim, « Low Frequency Non-Separable Transform (LFNST) », *in*: *2019 Picture Coding Symposium (PCS)*, 2019, pp. 1–5, DOI: 10.1109/PCS48520.2019.8954507.

[29] T. Lu et al., « Luma Mapping with Chroma Scaling in Versatile Video Coding », *in*: *2020 Data Compression Conference (DCC)*, 2020, pp. 193–202, DOI: 10.1109/DCC47342.2020.00027.

[30] D. Marpe, H. Schwarz, and T. Wiegand, « Context-based adaptive binary arithmetic coding in the H.264/AVC video compression standard », *in*: *IEEE Transactions on Circuits and Systems for Video Technology* 13.*7* (2003), pp. 620–636, DOI: 10.1109/TCSVT.2003.815173.

[31] Y. Y. Jianle Chen and S. H. Kim, *Algorithm Description for Versatile Video Coding and Test Model 6 (VTM 6)*, document JVET-O2002, JVET, 2019.

[32] I.-K. Kim, J. Min, T. Lee, W.-J. Han, and J. Park, « Block Partitioning Structure in the HEVC Standard », *in*: *IEEE Transactions on Circuits and Systems for Video Technology* 22.*12* (2012), pp. 1697–1706, DOI: 10.1109/TCSVT.2012.2223011.

[33] Y. Huang, J. An, H. Huang, X. Li, S. Hsiang, K. Zhang, H. Gao, J. Ma, and O. Chubach, « Block Partitioning Structure in the VVC Standard », *in*: *IEEE Transactions on Circuits and Systems for Video Technology* 31.*10* (2021), pp. 3818–3833, DOI: 10.1109/TCSVT.2021.3088134.

[34] N. Zouidi, A. Kessentini, W. Hamidouche, N. Masmoudi, and D. Menard, « Multi-task Learning Based Intra-Mode Decision Framework for Versatile Video Coding », *in*: *Electronics* 11.*23* (2022), ISSN: 2079-9292, DOI: 10.3390/electronics11234001, URL: https://www.mdpi.com/2079-9292/11/23/4001.

[35] M. Schäfer, B. Stallenberger, J. Pfaff, P. Helle, H. Schwarz, D. Marpe, and T. Wiegand, « Efficient Fixed-Point Implementation Of Matrix-Based Intra Prediction », *in*: *2020 IEEE International Conference on Image Processing (ICIP)*, 2020, pp. 3364–3368, DOI: 10.1109/ICIP40778.2020.9190883.

[36] A. j. Tabatabai, R. S. Jasinschi, and T. Veen, « Motion Estimation Methods for Video Compression—A Review », *in*: *Journal of the Franklin Institute* 335.*8* (1998), pp. 1411–1441, ISSN: 0016-0032, DOI: https://doi.org/10.1016/S0016-0032(98)00007-6, URL: https://www.sciencedirect.com/science/article/pii/S0016003298000076.

[37] W.-J. Chien, L. Zhang, M. Winken, X. Li, R.-L. Liao, H. Gao, C.-W. Hsu, H. Liu, and C.-C. Chen, « Motion Vector Coding and Block Merging in the Versatile Video Coding Standard », *in*: *IEEE Transactions on Circuits and Systems for Video Technology* 31.*10* (2021), pp. 3848–3861, DOI: 10.1109/TCSVT.2021.3101212.

[38] X. Jin, K. Ngan, and G. Zhu, « COMBINED INTER-INTRA PREDICTION FOR HIGH DEFINITION VIDEO CODING », *in*: (Jan. 2007).

[39] H. Gao, S. Esenlik, E. Alshina, and E. Steinbach, « Geometric Partitioning Mode in Versatile Video Coding: Algorithm Review and Analysis », *in*: *IEEE Transactions on Circuits and Systems for Video Technology* 31.*9* (2021), pp. 3603–3617, DOI: 10.1109/TCSVT.2020.3040291.

[40] J. Boyce, K. Suehring, X. Li, and V. Seregin, *JVET common test conditions and software reference configurations*, document JVET-J1010, JVET, July 2018.

[41] Q. Dong, G. Yang, and N. Zhu, « A MCEA based passive forensics scheme for detecting frame-based video tampering », *in*: *Digital Investigation* 9 (Nov. 2012), pp. 151–159, DOI: 10.1016/j.diin.2012.07.002.

[42] Z. Li, A. Aaron, I. Katsavounidis, A. Moorthy, and M. Manohara, « Toward a practical perceptual video quality metric », *in*: *The Netflix Tech Blog* 6 (2016), p. 2.

[43] G. Sullivan, J. Ohm, W. Han, and T. Wiegand, « Overview of the high efficiency video coding (HEVC) standard », *in*: *IEEE Transactions on circuits and systems for video technology* 22.*12* (2012), pp. 1649–1668.

[44] D. Mukherjee, J. Bankoski, A. Grange, J. Han, J. Koleszar, P. Wilkins, Y. Xu, and R. Bultje, « The latest open-source video codec VP9-an overview and preliminary results », *in*: *2013 Picture Coding Symposium (PCS)*, IEEE, 2013, pp. 390–393.

[45] A. Feng, C. Gao, L. Li, D. Liu, and F. Wu, « Cnn-Based Depth Map Prediction for Fast Block Partitioning in HEVC Intra Coding », *in*: *2021 IEEE International Conference on Multimedia and Expo (ICME)*, 2021, pp. 1–6, DOI: 10.1109/ICME51207.2021.9428069.

[46] Z. Chen, J. Shi, and W. Li, « Learned fast HEVC intra coding », *in*: *IEEE Transactions on Image Processing* 29 (2020), pp. 5431–5446.

[47] X. Shen and L. Yu, « CU splitting early termination based on weighted SVM », *in*: *EURASIP journal on image and video processing* 2013 (2013), pp. 1–11.

[48] F. Duanmu, Z. Ma, and Y. Wang, « Fast mode and partition decision using machine learning for intra-frame coding in HEVC screen content coding extension », *in*: *IEEE Journal on Emerging and Selected Topics in Circuits and Systems* 6.*4* (2016), pp. 517–531.

[49] N. Westland, A. Dias, and M. Mrak, « Decision trees for complexity reduction in video compression », *in*: *2019 IEEE International Conference on Image Processing (ICIP)*, IEEE, 2019, pp. 2666–2670.

[50] B. Min and R. Cheung, « A fast CU size decision algorithm for the HEVC intra encoder », *in*: *IEEE Transactions on Circuits and Systems for Video Technology* 25.*5* (2014), pp. 892–896.

[51] G. Corrêa, P. Assuncao, L. Agostini, and L. da Silva Cruz, « Complexity control of high efficiency video encoders for power-constrained devices », *in*: *IEEE Transactions on Consumer Electronics* 57.*4* (2011), pp. 1866–1874.

[52] L. Shen, Z. Zhang, and Z. Liu, « Effective CU size decision for HEVC intracoding », *in*: *IEEE Transactions on Image Processing* 23.*10* (2014), pp. 4232–4241.

[53] P. Chiang and T. Chang, « Fast zero block detection and early CU termination for HEVC video coding », *in*: *2013 IEEE International Symposium on Circuits and Systems (ISCAS)*, IEEE, 2013, pp. 1640–1643.

[54] S. Paul, A. Norkin, and A. Bovik, « Speeding up VP9 intra encoder with hierarchical deep learning-based partition prediction », *in*: *IEEE Transactions on Image Processing* 29 (2020), pp. 8134–8148.

[55] Y. Fan, J. Chen, H. Sun, J. Katto, and M. Jing, « A Fast QTMT Partition Decision Strategy for VVC Intra Prediction », *in*: *IEEE Access* 8 (2020), pp. 107900–107911, DOI: 10.1109/ACCESS.2020.3000565.

[56] J. Cui, T. Zhang, C. Gu, X. Zhang, and S. Ma, « Gradient-Based Early Termination of CU Partition in VVC Intra Coding », *in*: *2020 Data Compression Conference (DCC)*, 2020, pp. 103–112, DOI: 10.1109/DCC47342.2020.00018.

[57] M. Saldanha, G. Sanchez, C. Marcon, and L. Agostini, « Fast Partitioning Decision Scheme for Versatile Video Coding Intra-Frame Prediction », *in*: *2020 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2020, pp. 1–5, DOI: 10.1109/ISCAS45731.2020.9180980.

[58] S. De-Luxán-Hernández et al., « An intra subpartition coding mode for VVC », *in*: *IEEE Int. Conf. on Image Processing (ICIP)*, 2019, pp. 1203–1207.

[59] F. Galpin, F. Racapé, S. Jaiswal, P. Bordes, F. Le Léannec, and E. François, « CNN-based driving of block partitioning for intra slices encoding », *in*: *2019 Data Compression Conference (DCC)*, IEEE, 2019, pp. 162–171.

[60] A. Tissier, W. Hamidouche, S. Mdalsi, J. Vanne, F. Galpin, and D. Menard, « Machine Learning based Efficient QT-MTT Partitioning Scheme for VVC Intra Encoders », *in*: *IEEE Transactions on Circuits and Systems for Video Technology* (2023).

[61] S. Wu, J. Shi, and Z. Chen, « HG-FCN: Hierarchical grid fully convolutional network for fast VVC intra coding », *in*: *IEEE Transactions on Circuits and Systems for Video Technology* 32.*8* (2022), pp. 5638–5649.

[62] T. Li, M. Xu, R. Tang, Y. Chen, and Q. Xing, « DeepQTMT: A deep learning approach for fast QTMT-based CU partition of intra-mode VVC », *in*: *IEEE Transactions on Image Processing* 30 (2021), pp. 5377–5390.

[63] A. Feng, K. Liu, D. Liu, L. Li, and F. Wu, « Partition Map Prediction for Fast Block Partitioning in VVC Intra-Frame Coding », *in*: *IEEE Transactions on Image Processing* 32 (2023), pp. 2237–2251, DOI: 10.1109/TIP.2023.3266165.

[64] M. Saldanha, G. Sanchez, C. Marcon, and L. Agostini, « Configurable fast block partitioning for VVC intra coding using light gradient boosting machine », *in*: *IEEE Transactions on Circuits and Systems for Video Technology* 32.*6* (2021), pp. 3947–3960.

[65] T. Amestoy, A. Mercat, W. Hamidouche, D. Menard, and C. Bergeron, « Tunable VVC Frame Partitioning Based on Lightweight Machine Learning », *in*: *IEEE Transactions on Image Processing* 29 (2020), pp. 1313–1328, DOI: 10.1109/TIP.2019.2938670.

[66] G. Kulupana, V. Kumar M, and S. Blasi, « Fast Versatile Video Coding using Specialised Decision Trees », *in*: *2021 Picture Coding Symposium (PCS)*, 2021, pp. 1–5, DOI: 10.1109/PCS50896.2021.9477461.

[67] Z. Pan, P. Zhang, B. Peng, N. Ling, and J. Lei, « A CNN-Based Fast Inter Coding Method for VVC », *in*: *IEEE Signal Processing Letters* 28 (2021), pp. 1260–1264, DOI: 10.1109/LSP.2021.3086692.

[68] W. Yeo and B. Kim, « CNN-based Fast Split Mode Decision Algorithm for Versatile Video Coding (VVC) Inter Prediction », *in*: *Journal of Multimedia Information System* 8.*3* (2021), pp. 147–158.

[69] X. Zhu and M. Bain, « B-CNN: Branch Convolutional Neural Network for Hierarchical Classification », *in*: (Sept. 2017).

[70] A. Tissier, W. Hamidouche, J. Vanne, and D. Menard, « Machine Learning Based Efficient Qt-Mtt Partitioning for VVC Inter Coding », *in*: *2022 IEEE International Conference on Image Processing (ICIP)*, IEEE, 2022, pp. 1401–1405.

[71] D. Schroeder, P. Rehm, and E. Steinbach, « Block structure reuse for multi-rate high efficiency video coding », *in*: *2015 IEEE International Conference on Image Processing (ICIP)*, Quebec City, QC, Canada: IEEE, Sept. 2015, pp. 3972–

3976, ISBN: 978-1-4799-8339-1, DOI: 10.1109/ICIP.2015.7351551, URL: http://ieeexplore.ieee.org/document/7351551/ (visited on 05/15/2023).

[72] E. Cetinkaya, H. Amirpour, C. Timmerer, and M. Ghanbari, « FaME-ML: Fast Multirate Encoding for HTTP Adaptive Streaming Using Machine Learning », *in*: *2020 IEEE International Conference on Visual Communications and Image Processing (VCIP)*, Macau, China: IEEE, Dec. 2020, pp. 87–90, ISBN: 978-1-72818-068-7, DOI: 10.1109/VCIP49819.2020.9301850, URL: https://ieeexplore.ieee.org/document/9301850/ (visited on 05/15/2023).

[73] H. Amirpour, E. Çetinkaya, C. Timmerer, and M. Ghanbari, « Towards Optimal Multirate Encoding for HTTP Adaptive Streaming », en, *in*: *MultiMedia Modeling*, ed. by J. Lokoč, T. Skopal, K. Schoeffmann, V. Mezaris, X. Li, S. Vrochidis, and I. Patras, vol. 12572, Series Title: Lecture Notes in Computer Science, Cham: Springer International Publishing, 2021, pp. 469–480, ISBN: 978-3-030-67831-9 978-3-030-67832-6, DOI: 10.1007/978-3-030-67832-6_38, URL: https://link.springer.com/10.1007/978-3-030-67832-6_38 (visited on 05/15/2023).

[74] V. V. Menon, H. Amirpour, M. Ghanbari, and C. Timmerer, « EMES: Efficient Multi-Encoding Schemes for HEVC-based Adaptive Bitrate Streaming », en, *in*: *ACM Transactions on Multimedia Computing, Communications, and Applications* (Dec. 2022), p. 3575659, ISSN: 1551-6857, 1551-6865, DOI: 10.1145/3575659, URL: https://dl.acm.org/doi/10.1145/3575659 (visited on 05/15/2023).

[75] P. Topiwala, M. P. Krishnan, and W. Dai, « Performance comparison of VVC, AV1, and HEVC on 8-bit and 10-bit content », *in*: *Optical Engineering + Applications*, 2018, URL: https://api.semanticscholar.org/CorpusID:69820243.

[76] M. Saldanha, G. Sanchez, C. Marcon, and L. Agostini, « Complexity analysis of VVC intra coding », *in*: *2020 IEEE International Conference on Image Processing (ICIP)*, IEEE, 2020, pp. 3119–3123.

[77] F. Pakdaman, M. A. Adelimanesh, M. Gabbouj, and M. R. Hashemi, « Complexity Analysis Of Next-Generation VVC Encoding And Decoding », *in*: *2020 IEEE International Conference on Image Processing (ICIP)*, IEEE, 2020, pp. 3134–3138.

[78] F. Bossen, K. Sühring, A. Wieckowski, and S. Liu, « VVC complexity and software implementation analysis », *in*: *IEEE Transactions on Circuits and Systems for Video Technology* 31.*10* (2021), pp. 3765–3778.

[79] A. Mercat, A. Mäkinen, J. Sainio, A. Lemmetti, M. Viitanen, and J. Vanne, « Comparative Rate-Distortion-Complexity Analysis of VVC and HEVC Video Codecs », *in*: *IEEE Access* 9 (2021), pp. 67813–67828.

[80] Z. Wang, S. Wang, X. Zhang, S. Wang, and S. Ma, « Fast QTBT Partitioning Decision for Interframe Coding with Convolution Neural Network », *in*: *2018 25th IEEE International Conference on Image Processing (ICIP)*, 2018, pp. 2550–2554, DOI: 10.1109/ICIP.2018.8451258.

[81] K. He, X. Zhang, S. Ren, and J. Sun, « Deep residual learning for image recognition », *in*: *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[82] D. Ma, F. Zhang, and D. Bull, « BVI-DVC: A training database for deep video compression », *in*: *IEEE Transactions on Multimedia* 24 (2021), pp. 3847–3858.

[83] Y. Wang, S. Inguva, and B. Adsumilli, « YouTube UGC dataset for video compression research », *in*: *2019 IEEE 21st International Workshop on Multimedia Signal Processing (MMSP)*, IEEE, 2019, pp. 1–5.

[84] « Adam: A method for stochastic optimization, author=Kingma, D.P. and others », *in*: *arXiv preprint arXiv:1412.6980* (2014).

[85] T. Hermann, *Frugally-Deep*, 2018.

[86] G. Bjontegaard, « Calculation of average PSNR differences between RD-curves », *in*: *VCEG-M33* (2001).

[87] T. Fu, H. Zhang, F. Mu, and H. Chen, « Fast CU Partitioning Algorithm for H.266/VVC Intra-Frame Coding », *in*: *2019 IEEE International Conference on Multimedia and Expo (ICME)*, 2019, pp. 55–60, DOI: 10.1109/ICME.2019.00018.

[88] Y. Liu, M. Abdoli, T. Guionnet, C. Guillemot, and A. Roumy, « Light-weight cnn-based vvc inter partitioning acceleration », *in*: *2022 IEEE 14th Image, Video, and Multidimensional Signal Processing Workshop (IVMSP)*, IEEE, 2022, pp. 1–5.

[89] G. Sullivan and T. Wiegand, « Rate-distortion optimization for video compression », *in*: *IEEE Signal Processing Magazine* 15.*6* (1998), pp. 74–90, DOI: 10.1109/79.733497.

[90] O. Ronneberger, P. Fischer, and T. Brox, « U-net: Convolutional networks for biomedical image segmentation », *in*: *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*, Springer, 2015, pp. 234–241.

[91] J. Long, E. Shelhamer, and T. Darrell, « Fully convolutional networks for semantic segmentation », *in*: *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3431–3440.

[92] Y. Wang et al., « The high-level syntax of the versatile video coding (VVC) standard », *in*: *IEEE Transactions on Circuits and Systems for Video Technology* 31.*10* (2021), pp. 3779–3800.

[93] Y. Y. J. Chen and S. Kim, *Algorithm Description for Versatile Video Coding and Test Model 10 (VTM 10)*, document JVET-S2002, JVET, 2020.

[94] A. Wieckowski et al., « Vvenc: An Open And Optimized Vvc Encoder Implementation », *in*: *2021 IEEE International Conference on Multimedia Expo Workshops (ICMEW)*, 2021, pp. 1–2, DOI: 10.1109/ICMEW53276.2021.9455944.

[95] I. Taabane, D. Menard, A. Mansouri, and A. Ahaitouf, « Machine Learning Based Fast QTMTT Partitioning Strategy for VVenC Encoder in Intra Coding », *in*: *Electronics* 12.*6* (2023), ISSN: 2079-9292, DOI: 10.3390/electronics12061338, URL: https://www.mdpi.com/2079-9292/12/6/1338.

[96] Y. Liu, M. Abdoli, T. Guionnet, C. Guillemot, and A. Roumy, « Statistical Analysis of Inter Coding in VVC Test Model (VTM) », *in*: *2022 IEEE International Conference on Image Processing (ICIP)*, 2022, pp. 3456–3459, DOI: 10.1109/ICIP46576.2022.9897595.

[97] A. Stergiou and R. Poppe, « AdaPool: Exponential Adaptive Pooling for Information-Retaining Downsampling », *in*: *arXiv preprint* (2021).

[98] V. V. Menon, C. Feldmann, H. Amirpour, M. Ghanbari, and C. Timmerer, « VCA: video complexity analyzer », *in*: *Proceedings of the 13th ACM Multimedia Systems Conference*, MMSys '22, June 2022, pp. 259–264, ISBN: 978-1-4503-9283-9, DOI: 10.1145/3524273.3532896, URL: https://doi.org/10.1145/3524273.3532896 (visited on 08/22/2022).

[99] A. Wieckowski, J. Ma, H. Schwarz, D. Marpe, and T. Wiegand, « Fast partitioning decision strategies for the upcoming versatile video coding (VVC) standard », *in*: *2019 IEEE International Conference on Image Processing (ICIP)*, IEEE, 2019, pp. 4130–4134.

Appendices

Appendix A

List of Acronyms

List of Acronyms

AI All-Intra. 46, 47

ALF Adaptive Loop Filter. 29, 33

AME Affine Motion Estimation. 29

AMVP Advanced Motion Vector Prediction. 42

AV1 AOMedia Video 1. 28, 54

AVC Advanced Video Coding. 17, 27–29, 34–38, 136

B-CNN Branch Convolutional Neural Network. 53

BD Bjontegaard Delta. 48, 49, 132

BD-PSNR Bjøntegaard Delta PSNR. 48, 49

BD-rate Bjontegaard Delta-Rate. 31, 48–50, 63, 65, 79, 91, 106

BDOF Bi-Directional Optical Flow. 29

BT Binary Tree. 36, 52, 53, 82, 83

CABAC Context Adaptive Binary Arithmetic Coding. 33, 44

CIIP Combined Intra-Inter Prediction. 42, 43

CNN Convolutional Neural Network. 12, 13, 19, 20, 22, 51–60, 63–67, 69, 70, 74–81, 84, 88–91, 93, 101

CTC Common Test Condition. 46, 49, 60, 63, 65, 66, 78, 84, 87, 106, 107

CTU Coding Tree Unit. 12, 20, 32, 34, 35, 38, 43, 51–54, 57–60, 63, 66, 68–71, 73, 76–80, 84, 85, 89, 91, 96–99, 132

CU Coding Unit. 19, 21, 22, 32–44, 51–55, 57–63, 68, 69, 71–73, 76, 80, 84, 85, 91, 94–98, 102, 105–108, 110, 111, 132, 133

dB decibel. 49

DCT Discrete Cosine Transform. 27, 33

DST Discrete Sine Transform. 33

DT Decision Tree. 52, 53, 68, 69, 72

EVC Essential Video Coding. 54

FCN Fully Convolutional Network. 74

FHD Full High Definition. 24, 38

FLOPs floating point operations. 91

FPS Frames per Second. 25

GOP Group Of Picture. 26, 44–47, 132

HBT Horizontal Binary Tree. 36, 59, 73, 84, 132

HD High Definition. 23, 24, 26, 27

HEVC High Efficiency Video Coding. 17, 18, 22, 23, 26–29, 34–39, 51, 52, 54–56, 76, 77, 93, 101, 136

HM HEVC Test Model. 26, 30, 31, 132

HTT Horizontal Ternary Tree. 37, 59, 73, 84, 132

IEC International Electrotechnical Commission. 23

IPM Intra Prediction Mode. 29, 39

ISO International Organization for Standardization. 23

ISO/IEC International Organization for Standardization / International Electrotechnical Commission. 27

ISP Intra SubPartition. 29, 40, 52

ITU-T International Telecommunication Union - Telecommunication Standardization Sector. 23, 27

JEM Joint Exploration Model. 52

JVET Joint Video Exploration Team. 23, 30, 52, 64, 65, 106

KLT Karhunen–Loève Transform. 33

LD Low-Delay. 46, 47

LD Low Definition. 24

LDB Low-Delay B. 46

LDP Low-Delay P. 46, 47, 132

LFNST Low Frequency Non-Separable Transform. 33

LGBM Light Gradient Boosting Machine. 53

LMCS Luma Mapping with Chroma Scaling. 33

MAE Mean Absolute Error. 63

MB MacroBlock. 34, 35

MBMP-CNN Multi-Branch Multi-Pooling CNN. 59, 67

MIP Matrix-based Intra Prediction. 40

MLT-CNN multi-level tree CNN. 53

MPEG Moving Picture Experts Group. 23

MRL Multi Reference Line. 29, 40

MS-MVF Multi-Scale Motion Vector Field. 76–78, 91

MS-MVF-CNN Multi-Scale Motion Vector Field CNN. 75, 82

MSE Mean Square Error. 48, 49

MSE-CNN Multi-Stage Exit CNN. 53

MT Multi-type Tree. 13, 20, 36, 52, 53, 58, 61, 62, 65, 67–74, 76, 79, 80, 82–84, 86, 88, 101

MTS Multiple Transform Set. 29, 33

MTsplitMap Multi-type Tree split Map. 72–74, 82, 84

MV Motion Vector. 40–42, 105

MVD Motion Vector Difference. 42

NS No Split. 38, 61, 62, 67, 73, 80, 82–84, 86, 97

POC Picture Order Count. 45, 46

PSNR Peak Signal-to-Noise Ratio. 48, 49, 98

PU Prediction Unit. 34–36, 38

QHD Quad High Definition. 24, 38

QP Quantization Parameter. 44, 49, 60, 63, 66, 77, 84, 87, 94–97, 108, 110

QT Quadtree. 13, 20, 29, 35, 36, 51–53, 56–60, 63, 68–73, 76, 80, 84, 85, 97, 101

QTBT quadtree plus binary tree. 52

QTdepthMap Quad Tree depth Map. 58, 60–63, 67, 68, 72–74, 80, 82, 84, 85

QTMT QuadTree with nested Multi-type Tree. 29, 37, 38, 52, 56, 58, 70, 71, 73, 91, 101, 132

RA RandomAccess. 22, 46, 47, 94, 101

RAGOP32 RandomAccess Group Of Picture 32. 20, 60, 63, 65, 67, 77, 78, 84, 87, 91, 107

RD Rate Distortion. 49

RD-cost Rate-Distortion cost. 43, 44, 48, 71

RDO Rate-Distortion Optimization. 12, 22, 23, 26, 31, 32, 43, 44, 51, 52, 54, 57, 58, 60, 61, 63, 68, 71, 72, 79, 80, 95, 97, 98, 102, 105–108, 110, 111

RF Random Forest. 53, 56, 57, 65, 66, 68, 70, 84, 87, 88, 133, 136

SAD Sum of Absolute Differences. 44, 77

SAO Sample Adaptive Offset. 29, 33

SATD Sum of Absolute Transform Differences. 44, 52

SD Standard Definition. 24

SVM Support Vector Machine. 50, 51

TL Temporal Layer. 47

TS Time Saving. 63–65, 87, 88

TT Ternary-type Tree. 37, 53, 82, 83, 88

TU Transform Unit. 34–36, 38

UHD Ultra High Definition. 24, 26, 27, 38

VBT Vertical Binary Tree. 36, 59, 73, 84, 132

VCA Video Complexity Analyzer. 94

VMAF Video Multi-method Assessment Fusion. 48, 50, 98, 99

VTM VVC Test Model. 20, 26, 30, 31, 44, 46, 47, 50, 56–58, 60, 63–65, 67, 68, 70, 72, 73, 80, 84, 87, 88, 90, 91, 93, 105–108, 132, 133

VTT Vertical Ternary Tree. 37, 59, 73, 84, 132

VVC Versatile Video Coding. 11, 12, 15, 17–23, 26–34, 36–42, 51–58, 65, 67, 68, 70, 71, 77, 86, 87, 90, 91, 93, 94, 96, 98, 99, 101, 102, 105, 106, 136

VVdeC Versatile Video Decoder. 31

VVenC Versatile Video Encoder. 31, 93, 97–99, 102

Appendix B

List of Figures

List of Figures

Appendix C

List of Tables

List of Tables

COLLEGE DOCTORAL BRETAGNE
MATHS, TELECOMS
INFORMATIQUE, SIGNAL
SYSTEMES, ELECTRONIQUE

Université de Rennes

Titre : Apprentissage pour l'encodage vidéo nouvelle génération

Mot clés : VVC, Codage inter, Réduction de la complexité, CNN, Partitionnement

Résumé : L'encodage vidéo avec le dernier codec Versatile Video Coding (VVC) requiert d'importantes ressources de calcul. Malgré son impact sur le temps d'encodage global, peu d'études portent sur l'accélération de l'encodage inter. Cette thèse se concentre ainsi sur ce sujet, en proposant des approches de partitionnement rapide.

Notre première contribution consiste à utiliser un CNN léger pour réduire l'espace de recherche de partitionnement. En estimant la carte de profondeur des décisions de partitionnement QT, ce CNN nous permet d'élaguer l'espace de recherche de l'arbre MT initial. Les expériences montrent que ce travail peut réduire d'un tiers la complexité, pour une perte légère en efficacité de codage.

Dans la deuxième partie, nous améliorons la première contribution en proposant une nouvelle structure CNN associé à un algorithme d'accélération. La carte de profondeur QT et les décisions MT sont prédites simultanément, réduisant davantage l'espace de recherche. Le compromis efficacité et accélération d'encodage obtenu surpasse l'état de l'art.

Dans la dernière partie, nous proposons une méthode de partitionnement rapide multi-débit pour les scénarios de streaming. Les données collectées à partir d'encodages à bas débits sont exploités pour accélérer les encodages à débits supérieurs. 40% de la complexité est reduite en appliquant notre approche à l'encodeur VVenc, avec une perte raisonnable.

Title: Learning for new generation video coders

Keywords: VVC, Inter coding, Complexity reduction, CNN, Partitioning

Abstract:

Video encoding with the latest video codec Versatile Video Coding (VVC) requires significant computational resources. Despite its impact on the overall encoding time, few studies focus on accelerating inter-frame encoding. This thesis, therefore, targets this topic by proposing fast partitioning approaches.

Our first contribution involves utilizing a lightweight CNN to reduce the search space of partitioning. By predicting the QT depth map, this CNN allows us to prune the search space of the MT partitions. Experiments show that this work can reduce complexity by a third, with slight coding loss.

In the second part, we intend to improve the first contribution by proposing a novel CNN structure associated with an acceleration algorithm. The QT depth map and the MT decisions are predicted simultaneously, further reducing the search space. The trade-off between efficiency and encoding acceleration of our method outperforms the stat of the art.

In the final part, we propose a multi-rate fast partitioning process for streaming scenarios. Partition data collected from low-bitrate encodings are leveraged to accelerate high-bitrate encodings. 40% reduction in complexity is achieved by applying our approach to the VVenc encoder, with a reasonable loss.