Complexity Estimation for Load Balancing of 360-degree Intra Versatile Video Coding

Jose N. Filipe*, J. Carreira*†, Luis M. N. Tavora†, Sergio M. M. Faria*†, Antonio Navarro*‡, and Pedro A. A. Assuncao*†

*Instituto de Telecomunicações, Portugal,
†ESTG, Polytechnic of Leiria, Leiria, Portugal,
‡Universidade de Aveiro, Aveiro, Portugal
e-mails: jose.filipe@ieee.org, {jcarreira, sergio.faria, amado}@co.it.pt, luis.tavora@ipleiria.pt, navarro@av.it.pt

***Abstract*—The ever increasing demand for image and video content poses new requirements to support higher resolutions and richer representation formats, creating new challenges in coding algorithms. The forthcoming Versatile Video Coding (VVC) standard aims to increase the coding efficiency of existing algorithms and it is particularly suitable for Ultra-High Definition (UHD) resolutions and 360° video. However, since coding efficiency gains are obtained at the cost of increased complexity, fast computational approaches are needed to cope with real-time requirements, such as parallel processing. Thus, this work presents a contribution towards efficient parallel encoding of 360° video, based on coding complexity estimation and non-uniform data-level splitting (slice-based) for load balancing across multiple processors. A machine learning approach is proposed to estimate the complexity of intra coding VVC, using uncorrelated features, obtained through Principal Component Analysis (PCA) and Extremely Randomised Trees (ERT). Then, a complexity-balanced slice partition is devised, taking advantage of the clustered complexity inherent to Equirectangular Projection (ERP). It is shown that coding complexity is estimated with an accuracy of 92.25%, and the encoding time is reduced by 8.50%, when compared to the case where the 360° frames are evenly split.**

***Index Terms*—Video Coding Complexity, Load-balancing, VVC, Complexity Estimation**

I. Introduction

Image and video data accounts for the majority of all internet traffic, due to new applications and emerging services using mixed reality, 4K and 8K resolutions, cloud gaming, involving not only humans in the communication chains but also machines, such as self-driving vehicles, smart surveillance systems, etc, [1]. Such increasing demand for media content with visual information poses new requirements to support higher resolutions and richer representation formats, creating new challenges in coding algorithms, networks and user devices [2]. This is also driving standardisation efforts, specifically targeting 360° video, that require very high resolutions and complex compression algorithms to enable advanced services and applications [3].

The Joint Video Exploration Team (JVET) is currently developing the forthcoming video compression standard, named Versatile Video Coding (VVC), to face the challenging requirements posed by Ultra-High Definition (UHD) resolutions of these new visual representation formats. The forthcoming VVC standard is expected to be released by the end of 2020 [4], greatly increasing the coding efficiency of its predecessor High Efficiency Video Coding (HEVC). However, since such improvement is achieved at the cost of significant increase in computational complexity, fast computational methods are of utmost importance to ease adoption of the standard and to meet implementation constraints.

Data level parallel processing based on slice partitioning is a possible solution to deal with the high computational complexity of VVC and the huge amount of data used by UHD 360° video. In previous works related to HEVC, such as [5]–[8], this is achieved by splitting frames into slices of roughly equal complexity to evenly distribute the computational load across several processors. However, to the best of the authors knowledge, similar problems have not been addressed neither for VVC nor for 360° video, which has different characteristics from those of conventional planar video. A relevant observation in 360° video using Equirectangular Projection (ERP) is the inherent geometric characteristics of such projection, which typically result in regions with higher complexity near the equator, while lower complexity regions are clustered near the poles [9]. This leads to a significant imbalance of the computational load when data is evenly partitioned.

In this work, a computational complexity estimation model is proposed, based on Extremely Randomised Trees (ERT) [10], for slice-based splitting under a load balancing criterion. ERT is a variant of Random Forests, which achieves better performance in the presence of low dimensionality noisy data [10]. Moreover, Principal Component Analysis (PCA) [11] is used to not only achieving orthogonality, but also to reduce the dimensionality of the feature set that was found to be relevant for modelling the computational complexity of coding tree units (CTUs) through ERT. Then, the estimated CTU complexity is used to split 360° ERP images at the CTU-level, into slices of equal computational complexity, i.e., equal computational load. The splitting algorithm minimises the imbalance associated with each slice. By using unequal amount of CTUs in each slice, the proposed approach leads

This work was supported by Programa Operacional Regional do Centro, project ARoundVision CENTRO-01-0145-FEDER-030652 and by FCT/MCTES through national funds and when applicable co-funded EU funds under the project UIDB/EEA/50008/2020, Portugal.

978-1-7281-8099-1/20/$31.00 ©2020 IEEE

to faster computation of 360° video coding when compared to split images into slices with the same size.

The paper is organised as follows: Section II describes the proposed method, including the features (Section II-A) used for complexity modelling (Section II-B) and the frame splitting method (Section II-C). Section III presents the results, and conclusions are drawn in Section IV.

II. Proposed method

In order to take advantage of the inherent complexity patterns of the various spherical projections, namely ERP projection, we propose a non-uniform frame splitting scheme to obtain horizontal stripes of different height but equal complexity. The coding complexity of each CTU is first estimated using a machine learning model and then, an optimisation scheme is used to split the frame into slices of roughly equal complexity. In the following sections, the feature extraction process, the machine learning model training and testing, as well as the optimisation scheme for the frame splitting process, are described.

A. Feature Set

To estimate the coding complexity of CTUs, the prediction model requires multiple features with the following specific requirements:

- must be informative with respect to the computational load required by intra coding modes of 360° video;
- must be computed from the video signal itself and not requiring information from the video encoder;
- must be uncorrelated to a given extent.

To accomplish the above requirements, a set of features are defined for the proposed model as listed in Table I. Features f_1 to f_9 are known to be informative in regard to coding complexity. They are indicators of different types of spatial information related to coding complexity. Some of them, have also been found relevant in modelling coding decision processes to accelerate state of the art encoders [12], [13]. Feature f_{10} is indicative of the intra coding mode that might be chosen for a given CTU. For different purposes, this feature has also been used in [14].

TABLE I: Features used in the complexity prediction model.

ID	Features
f_1	Luminance mean
f_2	Luminance standard deviation (std.)
f_3	Mean of Sobel filtered luminance in x direction
f_4	Std. of Sobel filtered luminance in x direction
f_5	Mean of Sobel filtered luminance in y direction
f_6	Std. of Sobel filtered luminance in y direction
f_7	Entropy of square root of sum of squares of Sobel filtered luminance in x and y directions
f_8	Std. of spatial information [15]
f_9	Root mean square of spatial information [15]
f_{10}	Angle of average Sobel gradient vector of luminance
f_{11}	Latitude of the centre point of the CTU (ϕ)
f_{12}	$\sec(\phi)$

Due to the intrinsic geometric characteristics of the ERP format, 360° video frames tend to present more redundancy in regions near the poles rather than near the equator. In other words, regions near the poles tend to be composed by CTUs with lower frequency content, while CTUs near the equator tend to have higher spatial frequency components [9]. Thus, to some extent the latitude of a given CTU is informative with respect to its complexity and for that reason included in the feature set as f_{11}.

The last feature, f_{12}, is the secant of the latitude f_{11}. This is related to the sampling density variation that causes the intrinsic spatial distortions of the ERP projection. The transformation between the plane coordinates (x, y) and the spherical coordinates (λ, ϕ), where ϕ is the latitude and λ is the longitude, is given by (1) [16]. In general the radius of the sphere, R, is assumed to be 1, thus it is omitted in the expression.

$$\begin{aligned} x &= \lambda, \quad -\pi < \lambda \leq \pi \\ y &= \phi, \quad -\frac{\pi}{2} < \phi < \frac{\pi}{2} \end{aligned} \tag{1}$$

The variation of the sampling density with the latitude ϕ can be explained as follows. Considering a plane intersecting the sphere at the equator, such intersection is a circle with the same radius as the sphere (R). If this plane, parallel to the equator, is moved towards any pole the radius r of the circle spawned by the intersection decreases, which can be expressed by (2).

$$r(\phi) = \cos(\phi) \tag{2}$$

To convert an image from the spherical to the planar domain, the perimeter of each of these circles is sampled. In order to get a rectangular shape, the same number of pixels must be sampled from each perimeter. However, according to (2), the perimeter decreases with the latitude, thus the sampling density must increase as moving closer to the poles. The non-linear increasing of the sampling density is function of the inverse of the radius, originating the intrinsic distortion of the ERP projection. Taking the sampling density of the equator as reference, the relative sampling density s_ϕ at any latitude ϕ is given by (3).

$$s_\phi = \frac{1}{\cos(\phi)} = \sec(\phi) \tag{3}$$

Essentially, (3) shows that the sampling density increases with the latitude, thus it implicitly indicates that the redundancy is higher in CTUs located closer to the poles, which has an impact on coding complexity and therefore justifies the use of f_{12} as a feature in the ERT model.

B. Complexity Prediction Model

Given the number and diversity of features related to the computational complexity of VVC intra coding, a machine learning approach is best suited to devise a prediction model for such complexity. From preliminary studies, Extremely Randomised Trees combined with PCA were found to be capable of providing good accuracy for this purpose.

1) Principal Component Analysis: The PCA [11] is used to reduce the dimensionality of the original feature space and, at the same time, to maximise the orthogonality between the features in the transformed feature space. The new set of features obtained through the PCA process are the principal components, which are computed by applying a linear transform to the original features. Such principal components correspond to largest eigenvalues of the co-variance matrix of features. Then a reduced set of principal components can be used to reconstruct most of the original data with maximum variance, thus keeping most of its information. The orthogonality between components ensures decorrelation in the transformed feature space. In this paper, PCA was used on the set of all 12 features described in Section II-A and reducing the dimensionality of the actual feature set to 5 components.

2) Extremely Randomised Trees: To some extent, the advantages of ERT over Random Forests, arise from the method used to chose attributes and cut-points while cutting a tree node. While in Random Forest this is done by computing the local optimum cut-point for each feature, using Information Gain for example, in ERT, a set of cut-points are randomly generated for each feature. Then, the cut-point from the randomly generated set that yields the best accuracy is selected. Furthermore, while Random Forests use a Bootstrap Aggregating approach [17] to split the training dataset into smaller training sets to train each decision tree individually, in ERT each decision tree is trained over the entire training dataset.

In this research, a set of 5 PCA features are computed from the original set of 12 and given to the ERT model to classify each CTU into one of 8 uniformly distributed complexity classes. Class 0 means that the CTU accounts for 0 to 12.5% of the complexity of the CTU that took the longest time to encode in the sequence, while a classification of 8 means that the CTU accounts for 87.5 to 100% of the complexity of the CTU that took the longest time.

3) Training: The set of features described in Section II-A, which are computed for each CTU, are used to compute the 5 principal components trough PCA, that in turn will be used to train the ERT model.

Three UHD 360° video sequences (*Skateboard*, *Harbor* and *Kite Flite*) from [18] are intra coded and used in the training process. A total of $14,400$ CTUs are used between the training and testing processes. Taking the whole set of CTUs of these 3 sequences, 75% are randomly selected for the training dataset while the remaining 25% are used for testing. In order to deal with the imbalance of the dataset, mis-classification is penalised according to the inverse of each class relative frequency, as suggested in [19].

With this training setup, an accuracy of 92.25% was achieved on the test dataset. Fig. 1 shows the confusion matrix computed for a balanced sub-set of the test dataset. The y axis represents the actual complexity of a CTU while the prediction given by the proposed model is the x axis. The matrix is normalised, thus it would be an identity matrix if the model was 100% accurate. Yellow colours, as well as circles with bigger radius, represent higher occurrences, while blue colour, as well as smaller circles, represent less occurrences. Squares with no circles represent an occurrence of 0%.

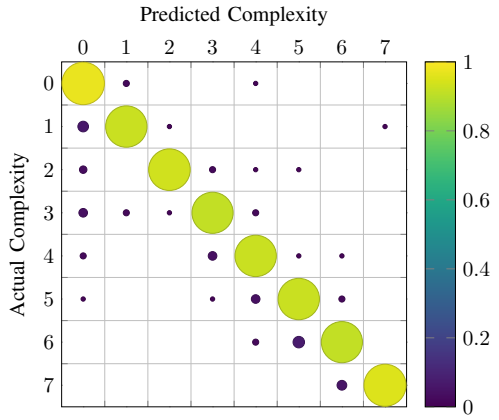

Fig. 1: Confusion matrix of the complexity prediction model.

As expected, the majority of the test cases fall on the main diagonal of the confusion matrix. All the eight elements of the main diagonal present an occurrence above 90%. Furthermore, it is also possible to notice that the mis-identified cases fall mostly close to the main diagonal (within a ± 1 interval). In this specific application, the idea is to have a rough estimation of the overall distribution of the complexity within a frame *ergo*, the small number of mis-classifications that fall within this interval do not represent a major problem.

C. Complexity-based slice partitioning

The CTU complexity predicted with the ERT model is used to split a frame into slices with roughly the same total complexity, in order to efficiently distribute the computation load through a predefined number of processing cores. This can be achieved in VVC using rectangular slices [20], and then modifying the encoder so that a compression process is spawned for each slice.

The proposed algorithm divides a given frame into n slices of approximately equal complexity. Considering C_T as the total complexity of a frame to be split into n slices, c_j the complexity of slice j, and $F_t = \{c_1, ..., c_j, ...c_n\}$ defining a given set of n slices from the same frame, this is done by finding the set F_{opt} that minimises the sum of the squares of the load imbalance, as given by (4).

$$F_{opt} = \arg\min_{F_t} \sum_{j \in F_t}^{n} \left(\frac{c_j - C_T/n}{C_T/n} \right)^2 \tag{4}$$

III. Results

As previously mentioned, complexity of ERP frames along the vertical axis is clustered, with regions of higher complexity near the equator and regions with lower complexity near the pole. To take advantage of this characteristic, each frame is split horizontally into slices of roughly equal complexity, in order to allow efficient parallel encoding. To evaluate the

efficiency of the proposed method in double-core and quad-core processing platforms, the 360° frames were split into two and four slices. All test sequences were encoded with the reference software VTM 4.0.1, using All Intra configuration, Next Profile, and QP=22. The encoding times were measured CTU by CTU. Five sequences from [18] were used (*Skateboard*, *Harbor*, *KiteFlite*, *Trolley*, and *Chairlift*), two of them were not used in the training/testing processes (*Trolley* and *Chairlift*).

Table II shows the average maximum imbalance of each frame, for all sequences used in the evaluation. For an arbitrary slice j, the imbalance I is computed by using (5),

$$I(j) = \frac{c_j - C_T/n}{C_T/n} \cdot 100\% \tag{5}$$

The imbalance indicates by how much a given slice deviates from the ideal computational complexity of a frame divided in n slices. This deviation is given in percentage. For instance, an imbalance of 0% means that the slice matches the ideal complexity of C_T/n, while an imbalance of 5% means that the slice complexity is 5% above C_T/n.

In Table II, the imbalance of the proposed method and even split are shown, for the cases of splitting the 360° frames in either 2 or 4 slices. For the case of 2 slices, the proposed method clearly outperforms the uniform splitting method, in the *Skateboard* and *Harbor* sequences, confirming the unequal distribution of the computational complexity in ERP images. In the *KiteFlite* sequence, both the proposed method and uniform splitting case yield the same result. This is due to the fact that the optimal slice-cutting point in latitude coincides exactly with the horizontal geometric mean of the frames. Finally, in sequences *Trolley* and *Chairlift*, the proposed method also outperforms evenly split slices, albeit by a smaller margin. The results obtained for these last two sequences can be explained by the fact that ERP projection is almost horizontally symmetric around the equator. This means that if there are not diverse scene characteristics (such as high frequency regions) in one of the hemispheres, the cutting point tends to be near the equator in the case where 2 slices are considered. However, when 4 slices are used, this is no longer the case and the unequal distribution of the complexity is much more noticeable. This is confirmed by the fact that the proposed method consistently achieves much lower imbalance than straightforward evenly distributed slices of the same size.

TABLE II: Average maximum imbalance (%).

Sequences	Proposed Method		Uniform Splitting	
	2 Slices	4 Slices	2 Slices	4 Slices
Skateboard	3.37	13.07	20.70	53.57
Harbor	0.39	14.57	37.55	77.35
Kiteflite	2.69	7.49	2.69	42.24
Trolley	3.21	9.95	4.97	26.85
Chairlift	3.13	6.00	5.67	15.80

Fig. 2 shows the simulated gains in processing time, by using the proposed complexity estimation model and balanced slice partitioning as described in Section II, using horizontal

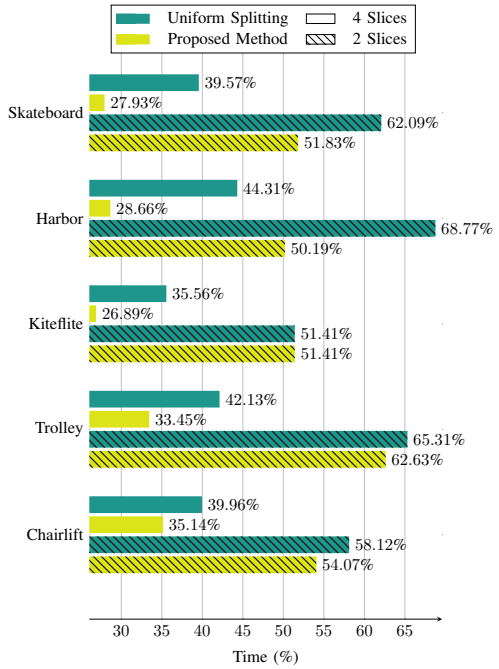

Fig. 2: VVC processing time for 360°video sequences (%).

partitioning. The bars with the diagonal line overlaid show the case where the frames are split into 2 slices, while the clean bars show the case where the frames are split into 4 slices. These results compare the total time spent to encode each video sequence, given by the sum of the encoding time used by all frames, for three different simulations:

1) encoding each frame as a single thread (this is the reference to calculate the percentage shown in the figure);
2) encoding each frame in parallel using either 2 or 4 slices evenly distributed;
3) encoding each frame in parallel using either 2 or 4 slices of unequal size as given by the proposed method.

The percentage shown in Fig. 2 is computed according to (6), where t_{ref} is the time obtained from 1), while t_p is the encoding time obtained from the parallel processing simulations, obtained from 2) and 3). As mentioned before, all processing times used in these simulations were taken from actually running the VVC encoder for the given sequences, *i.e.*, the splitting method described in Section II-C, applied to the predicted complexities obtained with the method presented in Section II-B.

$$T_r = \frac{t_p}{t_{ref}} \cdot 100\% \tag{6}$$

From Fig. 2, it is possible to observe that using 2 slices of equal size, for instance in the *Harbor* sequence, takes about 68.77% of the time it took to be encoded sequentially, reaching a complexity reduction of about 41.23%. However, the proposed method (yellow) achieves faster encoding, taking only 50.19% of the time in the case of splitting in 4 slices and around 28.66% of time in the case of splitting the frame in 2

slices. In fact, the proposed method outperforms the uniform splitting in all tests, which demonstrates that it is better to distribute 360° video data unevenly across different processors rather than agnostic processing of the same amount of data in each one.

It is also worthwhile to notice that the theoretical optimal encoding time reduction would lead to 50% and 25% of the time, for the cases where the frames are divided into 2 slices and 4 slices, respectively. Therefore, the proposed method is on average 4.03% above the theoretical limit for the case where the frames are divided into 2 slices and 5.42% for the case where the frames are divided into 4 slices. This deviations are mainly due to the fact that the minimum granularity that can be used for determining the cutting point between slices is a row of CTUs. The proposed method adds a complexity overhead of about 2% on average, computed using (6), to the time required for encoding the same test sequences with the reference algorithm. Furthermore, hypothetical coding efficiency loss due to constrained intra prediction within slice boundaries is negligible, as one can infer from previous studies [21]. This was confirmed through a single experiment by encoding the Harbor test sequence. Uniform frame splitting in 2, 3 and 4 horizontal slices results in a coding efficiency loss of 0.49%, 0.46% and 0.88% respectively, measured using Bjontegaard Delta Rate. The magnitude of such losses is below 1% and thus are deemed negligible.

IV. Conclusions

This work addresses the problem of load-balancing for parallel coding of 360° video using the forthcoming VVC. A machine learning model is proposed to estimate the computational complexity on a CTU basis and then splitting intra frames of 360° video into slices of different size but equal target complexity, for parallel encoding. Namely, a machine learning model, using PCA for dimensionality reduction and decorrelation of features followed by ERT for classification, is proposed to predict the complexity of each CTU, achieving an average accuracy of 92.25%. By splitting 360° frames into slices with the same estimated computational complexity results in faster encoding than simply dividing each frame into an equal number of CTUs (i.e., same amount of data).

Besides the inherent differences among the visual content, this method also takes advantage of the clustered complexity of ERP images along the latitude. It is also shown that the proposed method presents equal or less imbalance than uniform splitting for all sequences tested. Furthermore, it is shown that the encoding time can be reduced, on average, by 8.50%, when compared to the case where the frames are evenly split. It is also concluded that the encoding time reduction is only slightly above the optimum limit (4.72% on average, i.e., it is $4,72\%$ higher then 50% and 25% of the complexity, when two and four slices are used respectively) due to the coarse granularity resulting from the CTU size. The proposed approach can be further expanded to include features related to the temporal dimension and higher video data dimensions including temporal segments of limited duration, allowing for optimisation of frame partitioning in case of inter-frame coding.

References

[1] Cisco, "White paper: Cisco Annual Internet Report (2018–2023)," Cisco, Tech. Rep., 2020.

[2] M. Wien, J. M. Boyce, T. Stockhammer, and W. Peng, "Guest editorial immersive video coding and transmission," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 9, no. 1, pp. 1–4, Mar. 2019.

[3] R. Skupin, Y. Sanchez, Y. . Wang, M. M. Hannuksela, J. Boyce, and M. Wien, "Standardization status of 360 degree video coding and delivery," in *IEEE Visual Communications and Image Processing (VCIP)*, Dec. 2017, pp. 1–4.

[4] B. Bross, J. Chen, and S. Liu, "JVET-M1001: Versatile Video Coding (Draft 5)," Joint Video Experts Team (JVET), 14th Meeting: Geneva, SW, Tech. Rep., Mar. 2019.

[5] P. K. Papadopoulos, M. Koziri, and T. Loukopoulos, "A fast heuristic for tile partitioning and processor assignment in HEVC," in *25th IEEE International Conference on Image Processing (ICIP)*, Oct. 2018, pp. 4143–4147.

[6] Y. Ahn, T. Hwang, D. Sim, and W. Han, "Complexity model based load-balancing algorithm for parallel tools of HEVC," in *Visual Communications and Image Processing (VCIP)*, Nov. 2013, pp. 1–5.

[7] M. Koziri, P. K. Papadopoulos, N. Tziritas, A. N. Dadaliaris, T. Loukopoulos, S. U. Khan, and C. Xu, "Adaptive tile parallelization for fast video encoding in HEVC," in *IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)*, Dec. 2016, pp. 738–743.

[8] M. Koziri, P. Papadopoulos, N. Tziritas, A. N. Dadaliaris, T. Loukopoulos, and S. U. Khan, "Slice-based parallelization in HEVC encoding: Realizing the potential through efficient load balancing," in *IEEE 18th International Workshop on Multimedia Signal Processing (MMSP)*, Sep. 2016, pp. 1–6.

[9] B. Ray, J. Jung, and M. Larabi, "A low-complexity video encoder for equirectangular projected 360 video content," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Apr. 2018, pp. 1723–1727.

[10] P. Geurts, D. Ernst, and L. Wehenkel, "Extremely randomized trees," *Mach. Learn.*, vol. 63, no. 1, p. 3–42, Apr. 2006. [Online]. Available: https://doi.org/10.1007/s10994-006-6226-1

[11] I. Jolliffe, *Principal Component Analysis*. Springer Verlag, 1986.

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

[13] T. Amestoy, A. Mercat, W. Hamidouche, C. Bergeron, and D. Menard, "Random forest oriented fast QTBT frame partitioning," in *2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2019, pp. 1837–1841.

[14] M. Jamali and S. Coulombe, "Fast HEVC intra mode decision based on rdo cost prediction," *IEEE Transactions on Broadcasting*, vol. 65, no. 1, pp. 109–122, Mar. 2019.

[15] H. Yu and S. Winkler, "Image complexity and spatial information," in *Fifth International Workshop on Quality of Multimedia Experience (QoMEX)*, July 2013, pp. 12–17.

[16] J. P. Snyder, "Map projections: A working manual," U.S. Government Printing Office, Tech. Rep., 1987.

[17] L. Breiman, "Bagging predictors," *Mach. Learn.*, vol. 24, no. 2, p. 123–140, Aug. 1996. [Online]. Available: https://doi.org/10.1023/A:1018054314350

[18] P. Hanhart, J. Boyce, K. Choi, and J.-L. Lin, "L1012: JVET common test conditions and evaluation procedures for 360°video," Joint Video Experts Team (JVET), 12th Meeting: Macau, CH, Tech. Rep., Oct. 2018.

[19] C. Chen and L. Breiman, "Using random forest to learn imbalanced data," *University of California*, Jan. 2004.

[20] M. Coban, V. Seregin, and M. Karczewicz, "AHG12: On rectangular slices," Joint Video Experts Team (JVET), 15th Meeting: Gothenburg, SE, Tech. Rep., Jul. 2019.

[21] K. Misra, A. Segall, M. Horowitz, S. Xu, A. Fuldseth, and M. Zhou, "An overview of tiles in HEVC," *IEEE Journal of Selected Topics in Signal Processing*, vol. 7, no. 6, pp. 969–977, 2013.