



An Introduction for Panoramic Video Coding (case study)

Hot Period in Coding: 2017-2021

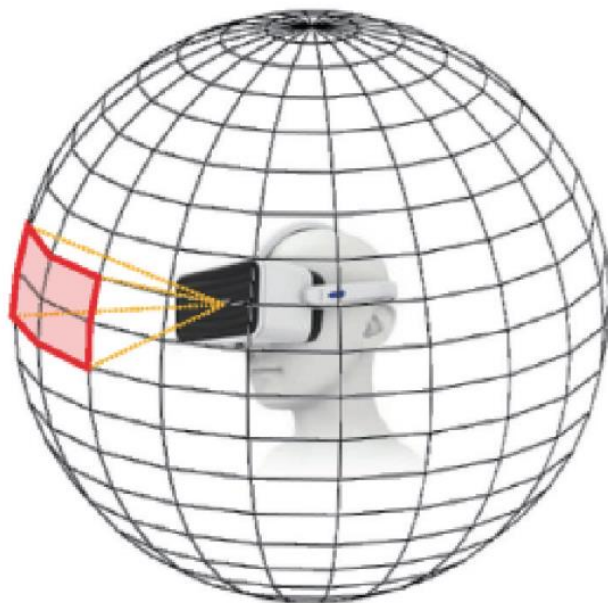
iVC Inter Group

2024.07.30

Panoramic Video



<https://www.bilibili.com/video/BV1nU4y1D7TR?t=18.2>



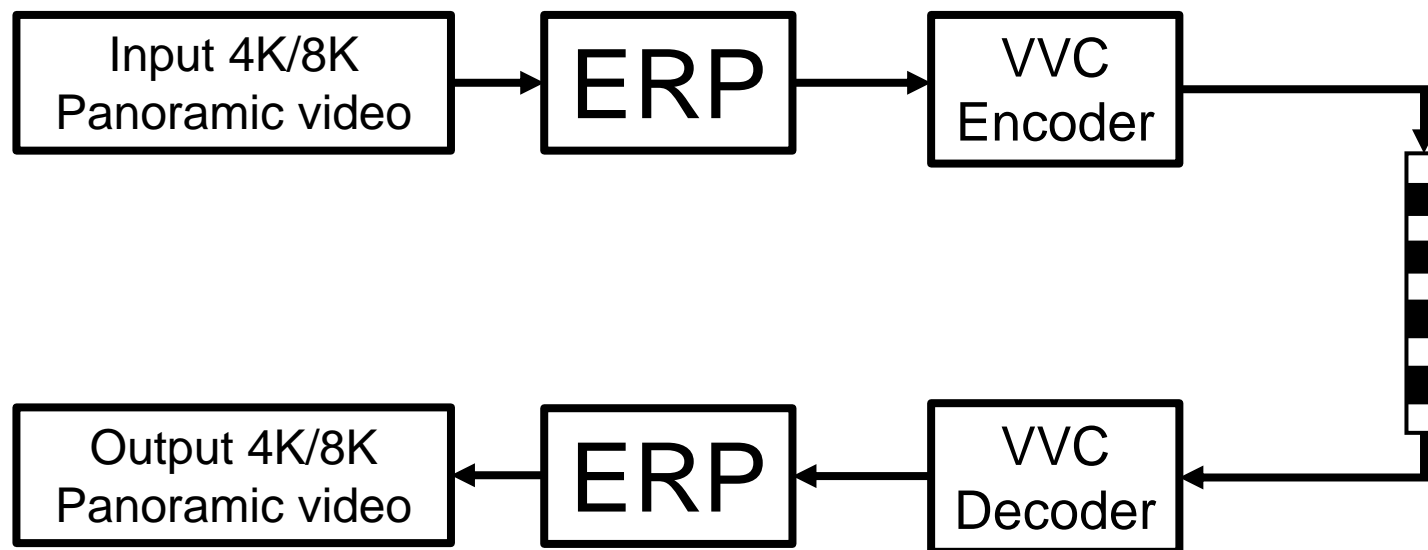
用户进行 VR 体验时的情景展示 (Lim 等, 2018) [1]

[1] Deep virtual reality image quality assessment with human perception guider for omnidirectional image. IEEE Transactions on Circuits and Systems for Video Technology

Standard For Panoramic Video

The horizontal wrap around motion compensation in the VVC is a 360-specific coding tool designed to improve the visual quality of reconstructed 360-degree video in the equi-rectangular (ERP) projection format. [1]

VVC旨在提高在等矩形(ERP)投影格式下重建360度视频的视觉质量。360-Lib是JVET小组设计并面向使用者公开的一个用于进行360全景视频研究的平台，关注于投影方案的评价。



[1] JVET-AB2002, Horizontal wrap around motion compensation



Projection For Panoramic Video

Index	Projection format
0	Equirectangular (ERP)
1	Cubemap (CMP)
2	Adjusted Equal-area (AEP)
3	Octahedron (OHP)
4	Viewport generation using rectilinear projection
5	Icosahedron (ISP)
6	Crasters Parabolic Projection for CPP-PSNR calculation
7	Truncated Square Pyramid (TSP)
8	Segmented Sphere Projection (SSP)
9	Adjusted Cubemap Projection (ACP)
10	Rotated Sphere Projection (RSP)
11	Equi-angular Cubemap Projection (EAC)
12	Equatorial Cylindrical Projection(ECP)

等距柱状投影

立方体投影

EAP-adjust (等积投影)

正八面体投影

直线投影

二十面体投影

抛物投影

金字塔棱台投影

条带投影

立方体投影-adjust

旋转球面投影

等角立方体投影

赤道圆柱投影

Projection For Panoramic Video

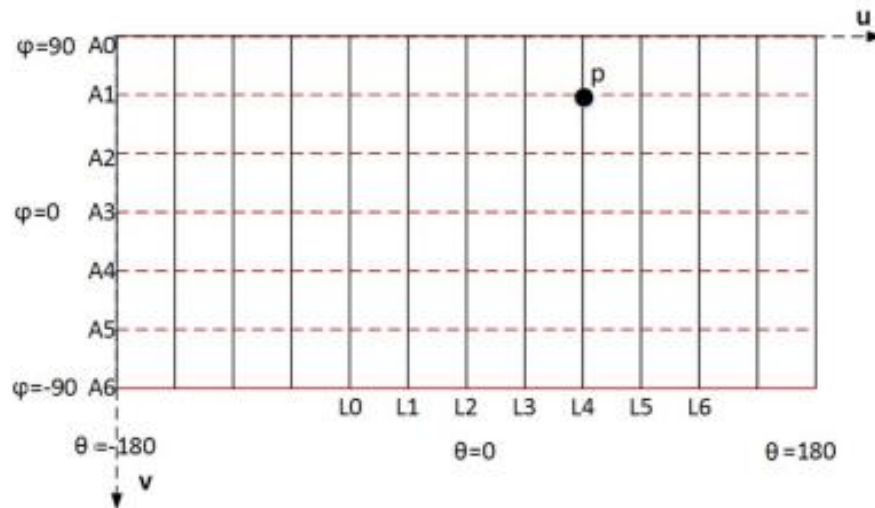
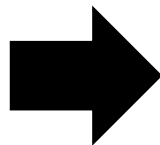
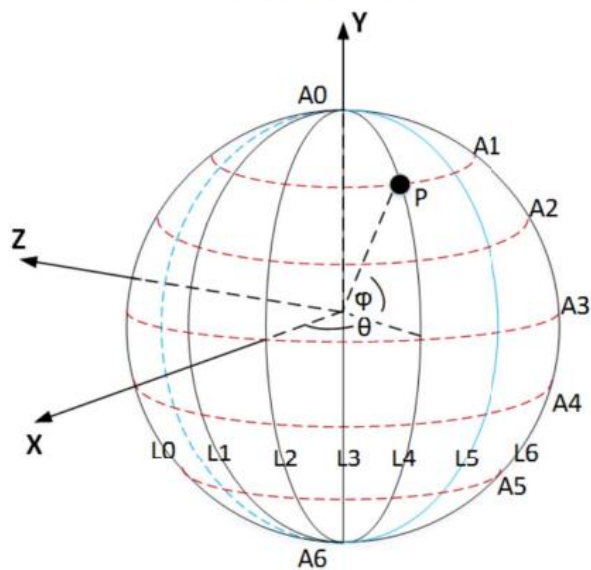
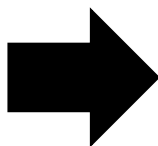


Equirectangular Projection, 等距柱状投影

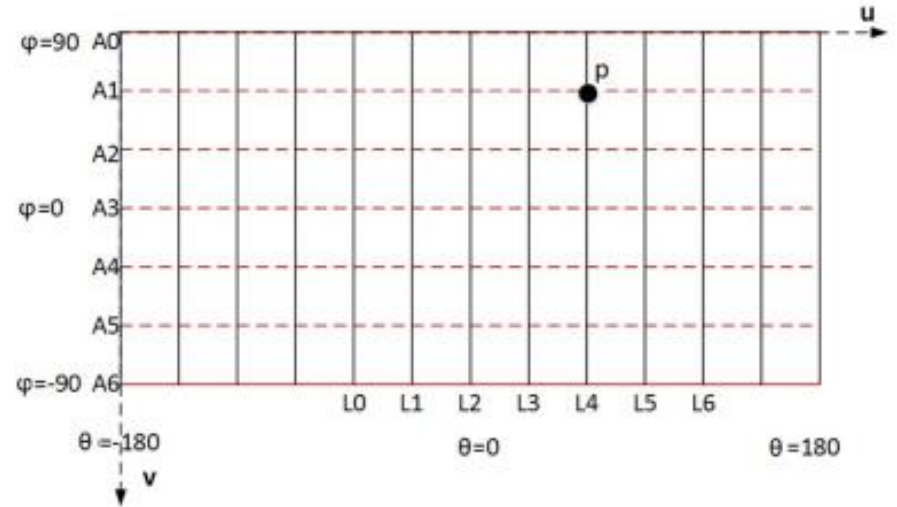
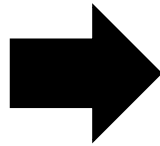
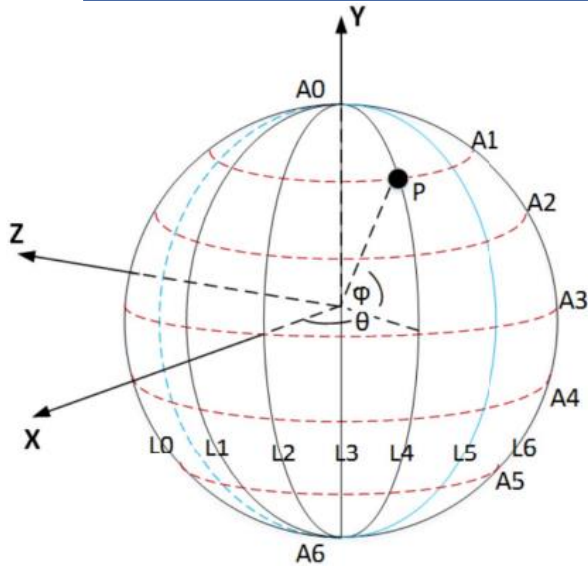


ERP

Projection For Panoramic Video



Projection For Panoramic Video



$$X = \cos(\theta) \cos(\phi) \quad (1)$$

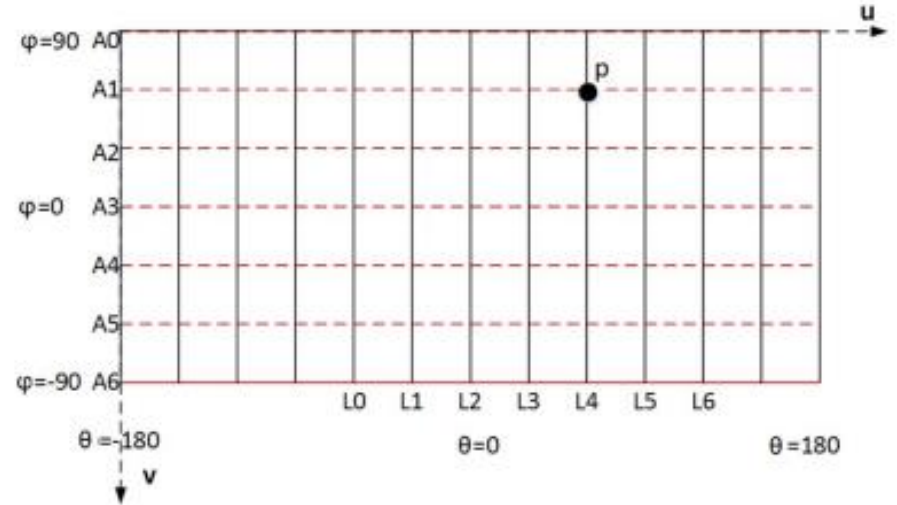
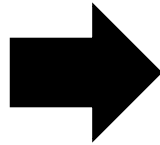
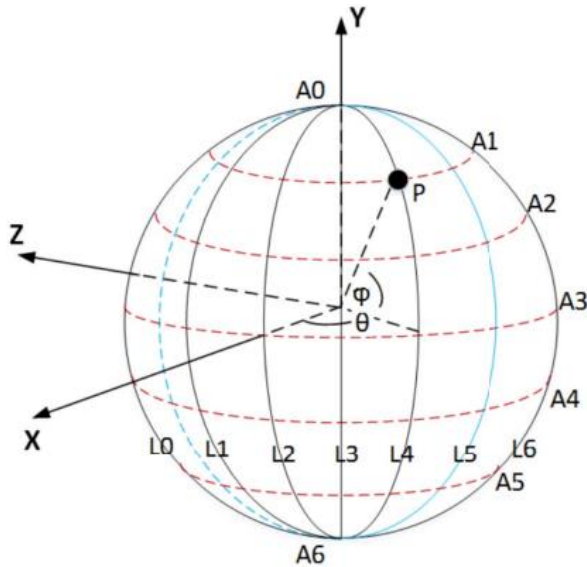
$$Y = \sin(\theta) \quad (2)$$

$$Z = -\cos(\theta) \sin(\phi) \quad (3)$$

$$\phi = \tan^{-1}(-Z/X) \quad (4)$$

$$\theta = \sin^{-1}\left(Y/(X^2 + Y^2 + Z^2)^{1/2}\right) \quad (5)$$

Equirectangular Projection, 等距柱状投影



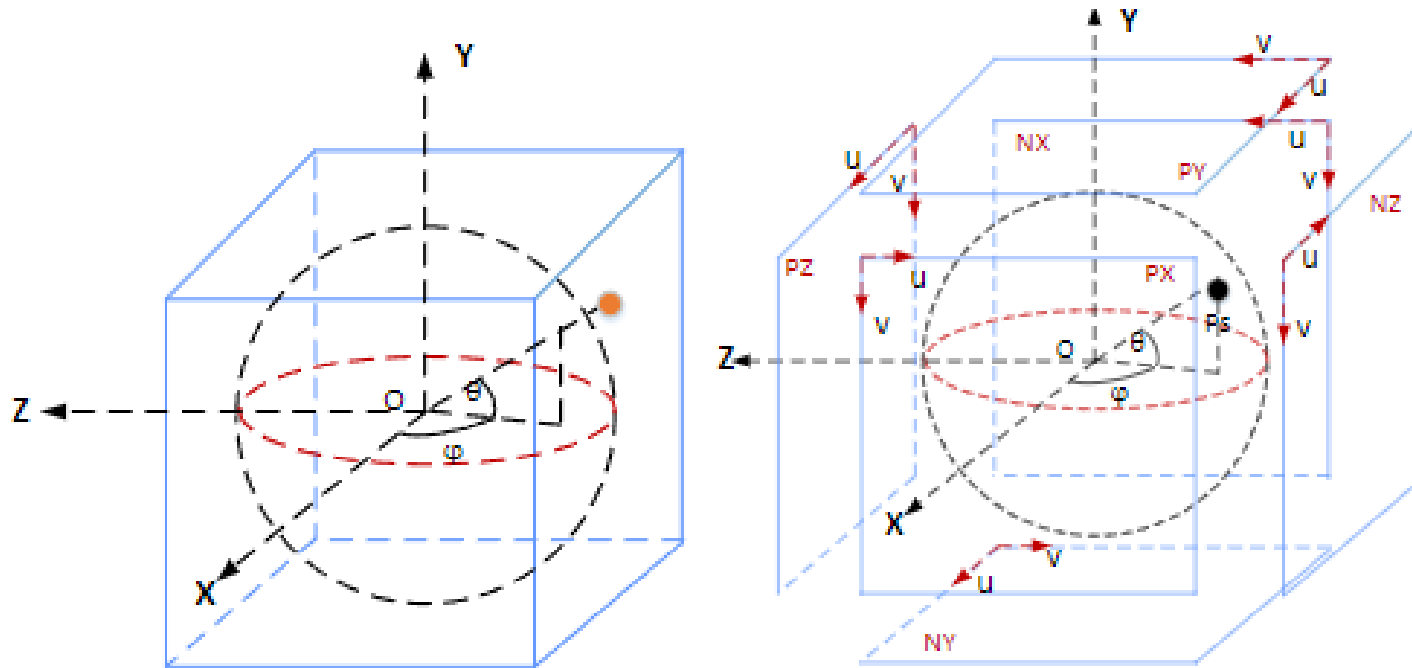
$$u = (m + 0.5) / W, 0 \leq m \leq W$$

$$v = (n + 0.5) / H, 0 \leq n \leq H$$

$$\phi = (u - 0.5) * (2 * \pi)$$

$$\theta = (0.5 - v) * \pi$$

Cubemap Projection, 立方体投影



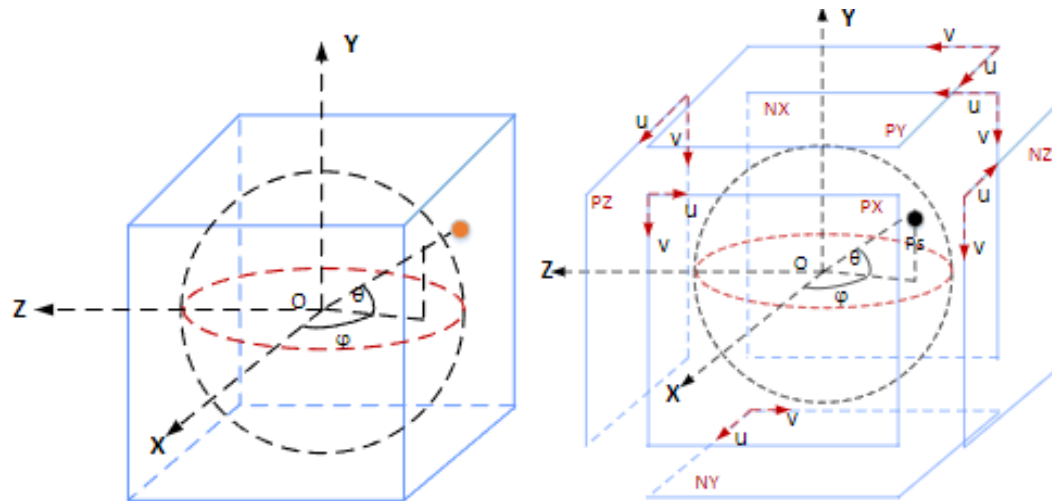
$$\begin{cases} u = \frac{2*(m+0.5)}{A} - 1, & 0 \leq m < A \\ v = \frac{2*(n+0.5)}{A} - 1, & 0 \leq n < A \end{cases}$$

[1] 360Libsoftwaremanual

[2] Convert 2:1 equirectangular panorama to cube map

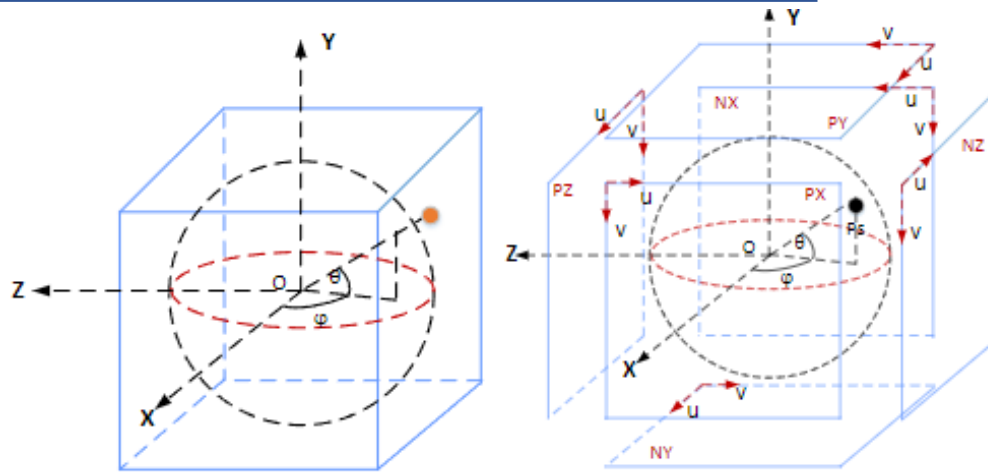
[3] 3D Graphics Rendering Cookbook A comprehensive guide to exploring rendering algorithms in modern OpenGL and Vulkan (Sergey Kosarevsky, Viktor Latypov)

Cubemap Projection, 立方体投影



正方体各面	face	X	Y	Z
前面(X轴正方向)	0	1.0	-v	-u
后面(X轴负方向)	1	-1.0	-v	u
上面(Y轴正方向)	2	u	1.0	v
下面(Y轴负方向)	3	u	-1.0	-v
右面(Z轴正方向)	4	u	-v	1.0
左面(Z轴负方向)	5	-u	-v	-1.0

Cubemap Projection, 立方体投影

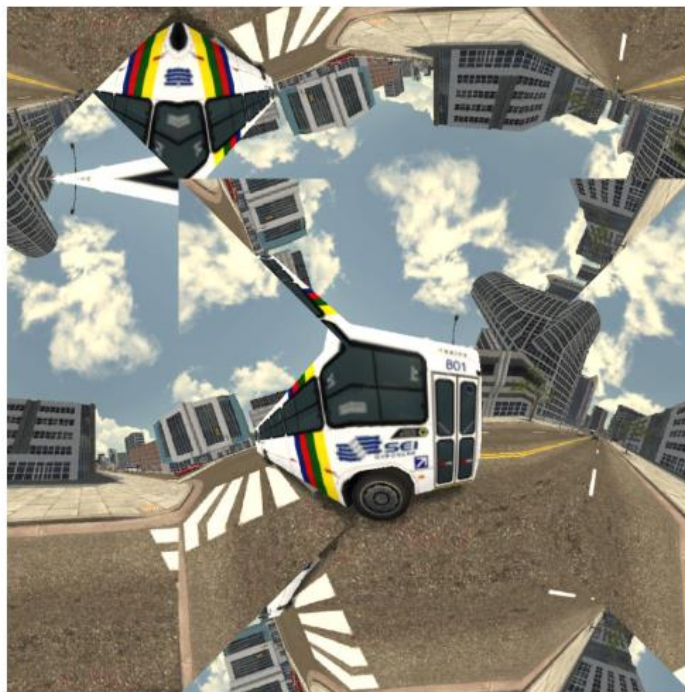


Condition	face	u	v
$ X \geq Y $ and $ X \geq Z $ and $X > 0$	0	$-Z/ X $	$-Y/ X $
$ X \geq Y $ and $ X \geq Z $ and $X < 0$	1	$Z/ X $	$-Y/ X $
$ Y \geq X $ and $ Y \geq Z $ and $Y > 0$	2	$X/ Y $	$Z/ Y $
$ Y \geq X $ and $ Y \geq Z $ and $Y < 0$	3	$X/ Y $	$-Z/ Y $
$ Z \geq X $ and $ Z \geq Y $ and $Z > 0$	4	$X/ Z $	$-Y/ Z $
$ Z \geq X $ and $ Z \geq Y $ and $Z < 0$	5	$-Z/ Z $	$-Y/ Z $

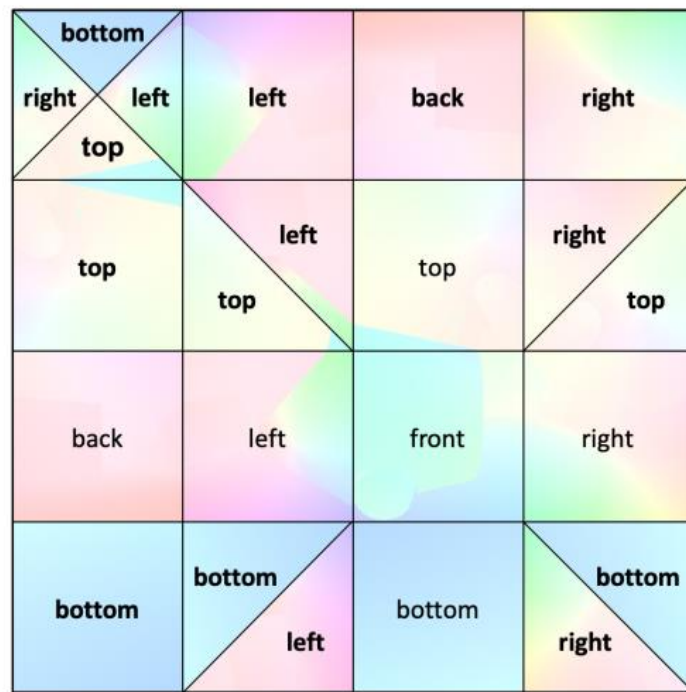
Cubemap Projection, 立方体投影



(a) Cube-map projection and optical flow

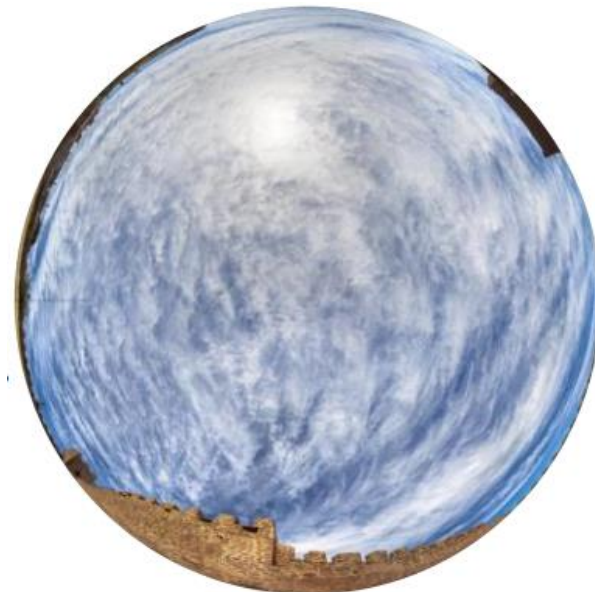


(b) Cube-padding projection

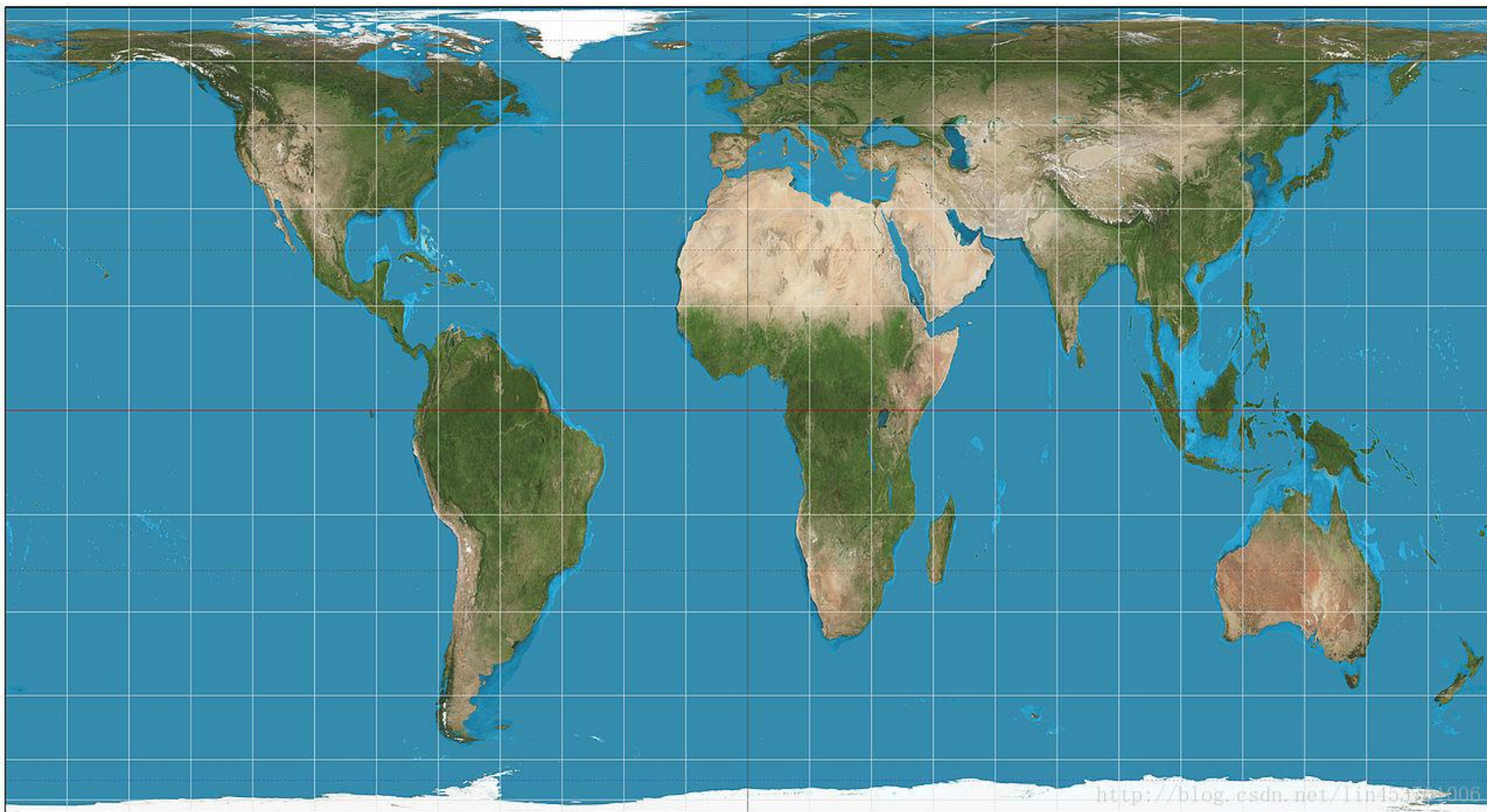


(c) Cube-padding optical-flow

Cubemap Projection, 立方体投影

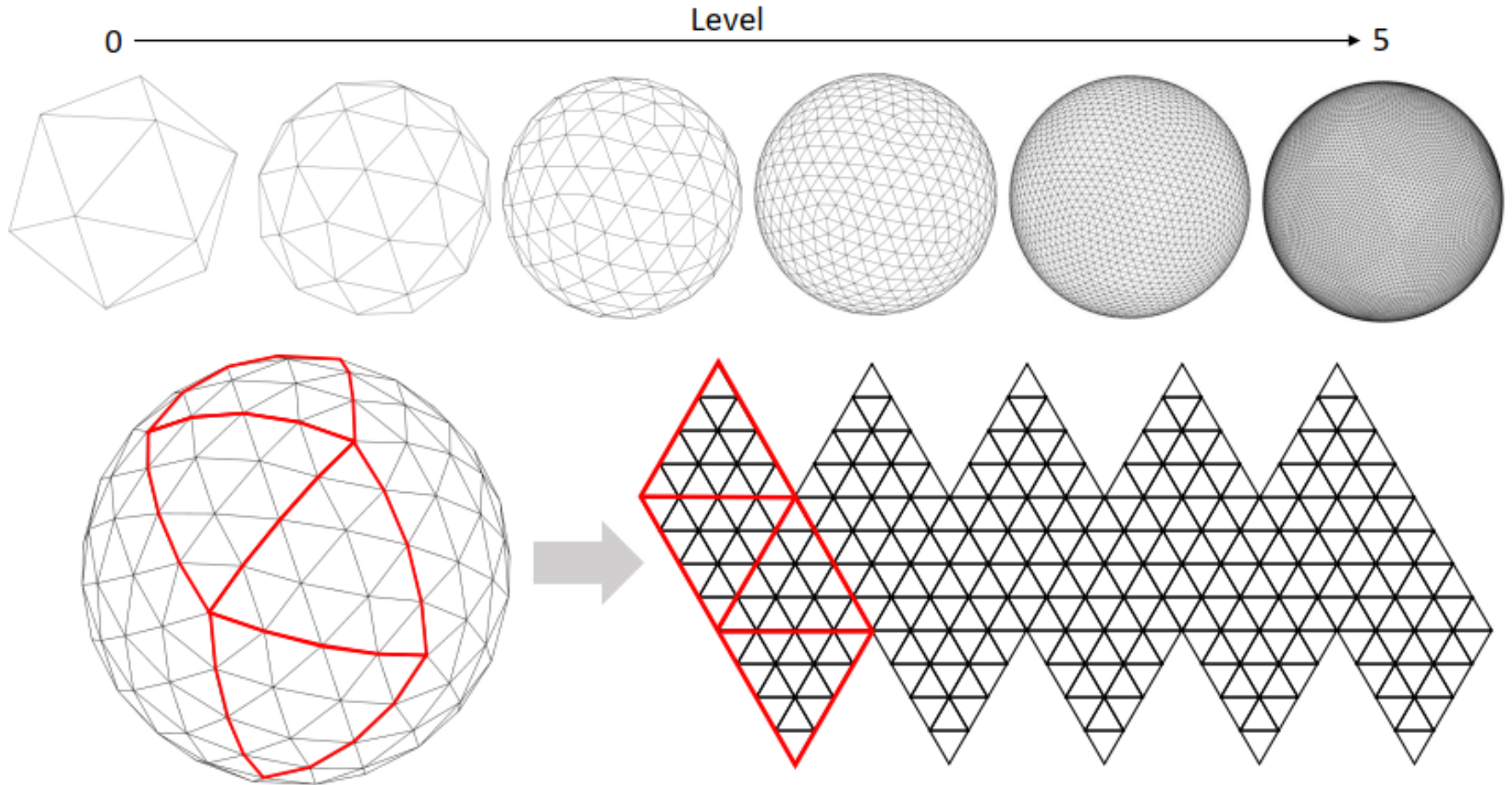


Cylindrical Equal-area Projection, 圆柱等积



圆柱等面积投影，将经线映射为等间隔的垂直线，纬线映射为水平线（非等间距）。可以想象为，将球面映射到一个圆柱面上并将圆柱面展开。

Icosahedron Projection (二十面体投影)



[1] SphereSR: 360 Image Super-Resolution with Arbitrary Projection via Continuous Spherical Image Representation

Paper List (Red in Today, Black for Next Time)



- Advanced Spherical Motion Model and Local Padding for 360° Video Compression (TIP-2019, Li Li)
- Motion-Plane-Adaptive Inter Prediction in 360-degree Video Coding (TCSVT-2023)
- A Geodesic Translation Model for Spherical Video Compression (TIP-2023)
- End-to-End Optimized 360° Image Compression (TIP-2022)
- LAU-Net: Latitude Adaptive Upscaling Network for Omnidirectional Image Super-resolution (CVPR-2021)
- Omnidirectional Image Super-Resolution with Transformer (CVPR-2022)
- Deep 360° Optical Flow Estimation Based on Multi-Projection Fusion (ECCV-2022)
- SphereSR: 360° Image Super-Resolution with Arbitrary Projection via Continuous Spherical Image Representation (CVPR-2023)

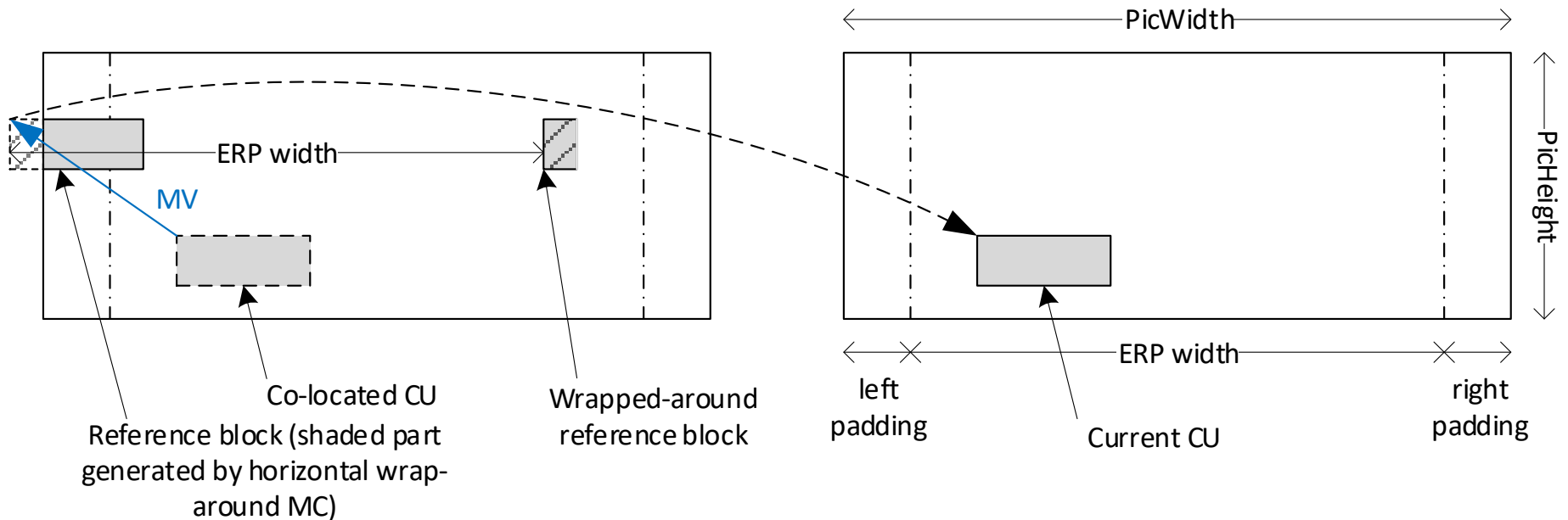


Padding for Panoramic Video Coding

Local Padding for 360° Video Compression

Reference picture

Current picture



推荐阅读的最新文章:

[1] Padding-Aware Learned Image Compression, ISCAS 2023

[2] AHG10: Improvement of Input Video Padding in VTM, JVET-AD0129

Local Padding for 360° Video Compression



(a)



(b)

Fig. 13. A comparison of the proposed padding and the original padding in HEVC. (a) Proposed padding. (b) Original padding in HEVC.

[1] Advanced Spherical Motion Model and Local Padding for 360° Video Compression (TIP-2019, Li Li)

Local 3D Padding in Spherical Video Coding

局部三维填充法的本质是将其他面中的**像素投影到当前面的延伸处**，以保证纹理的精确连续性。如果当前块的MV指向人脸边界之外，我们仍然可以得到一个**纹理连续的预测块**。对于不同类型的多面体投影格式，其主要区别在于**面数和面间角度的不同**。



Fig. 14. A typical example of the local 3D padding.

Local Padding for 360° Video Compression

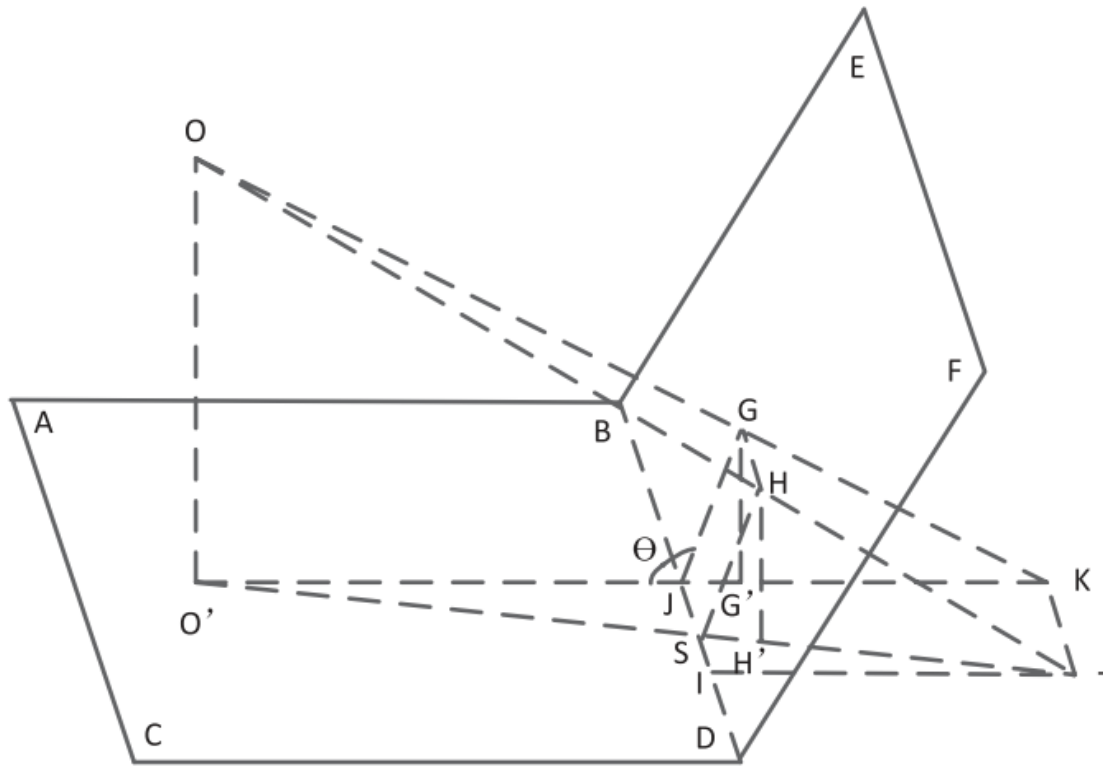
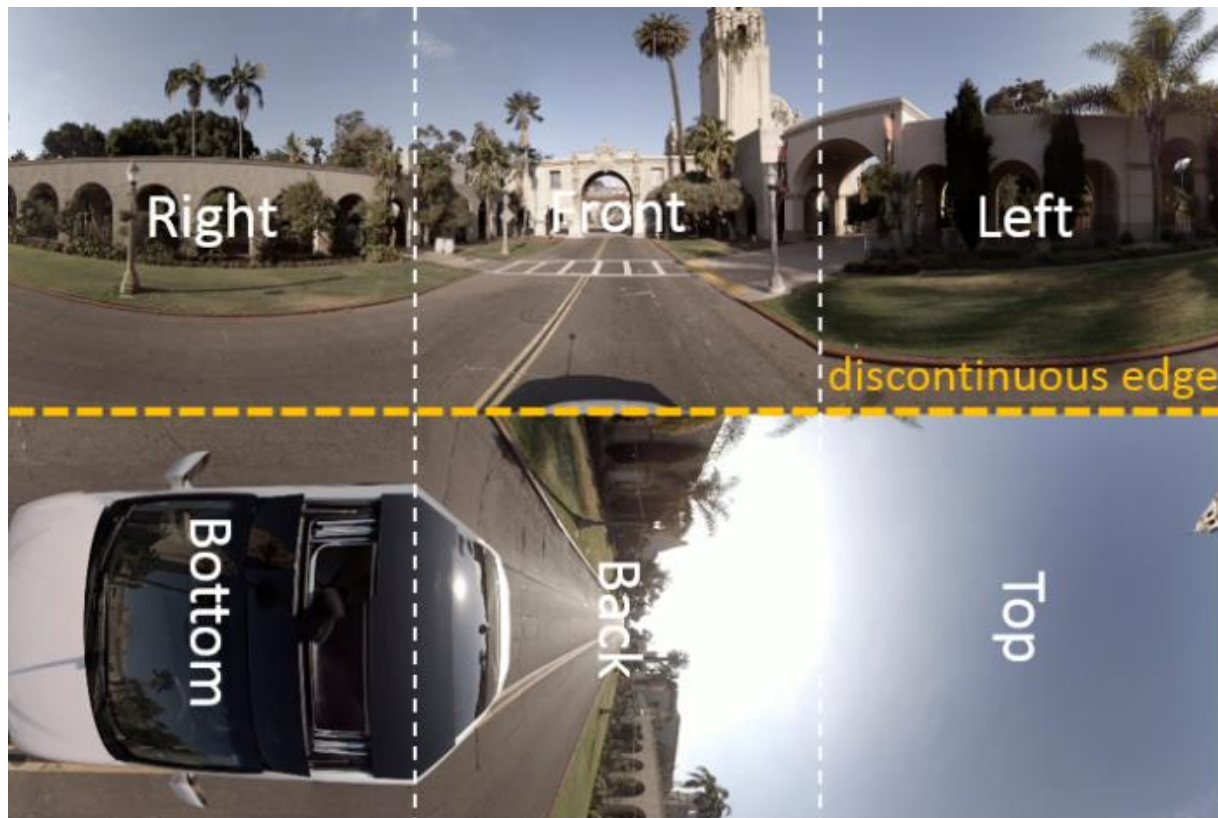


Fig. 7. A general example of the local 3D padding.

Local Padding Local Reference Padding Padding Storage

[1]Advanced Spherical Motion Model and Local Padding for 360° Video Compression (TIP-2019, Li Li)

Loop Filter Disabled Across Virtual Boundaries



Compared to using two tiles, one for each set of continuous faces, and to **disable in-loop filtering operations across tiles**, this technique is more flexible as it does not require the face size to be a multiple of the CTU size.

The virtual boundary could also be used in **Gradual Decoding Refresh (GDR)** or **Progressive Intra Refresh (PIR)** which is a technique to limit the large bitrate variations.



Motion Modeling for Panoramic Video Coding



Motion-Plane-Adaptive Inter Prediction in 360-Degree Video Coding

Andy Regensky, Christian Herglotz, *Member, IEEE*, and André Kaup, *Fellow, IEEE*

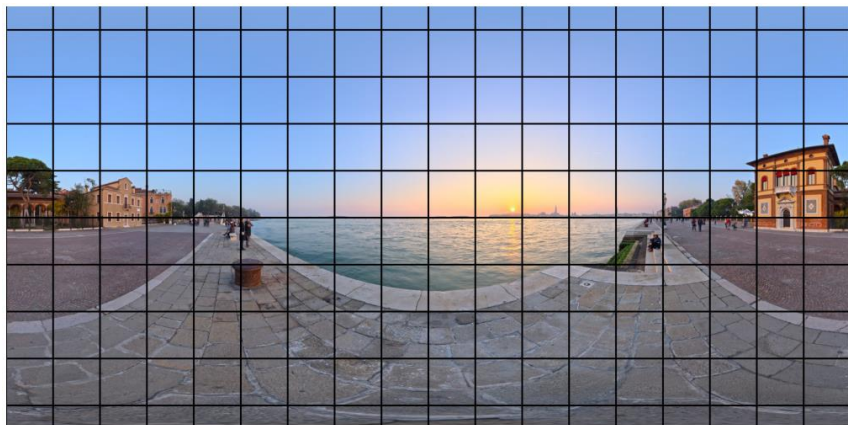


(b) sphere mapping (front)



(c) sphere mapping (back)

Introduction



(a) ERP-projected 360-degree image



(b) sphere mapping (front)



(c) sphere mapping (back)

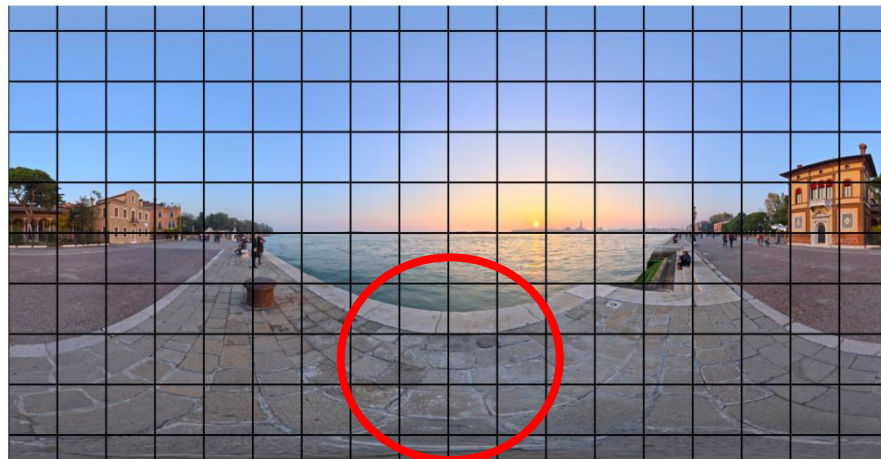
In this paper, we propose a motion-plane-adaptive inter prediction technique (MPA) for 360-degree video that takes the spherical characteristics of 360-degree video into account. Based on the known projection format of the video, MPA allows to **perform inter prediction on different motion planes** in 3D space **instead of having to work on the - in theory arbitrarily mapped - 2D image** representation directly. We furthermore **derive a motion-plane-adaptive motion vector prediction technique (MPA-MVP)** that allows to translate motion information **between different motion planes and motion models.**

传统360的编码方法  3D的编码方法

Key Question



The key question of sphere video coding



(a) ERP-projected 360-degree image

Typically, a 2D representation of the 360-degree video is required to allow compression using existing video coding techniques such as the H.264/AVC, the H.265/HEVC or the H.266/VVC, video coding standards. Fig(a) shows an example of a 360-degree image mapped to the 2D image plane through an **equirectangular projection (ERP)**. While this is one of the most common 360-degree projection formats, there exist a plethora of other formats including various variations of cubemap projections, segmented and rotated sphere projections, or octa- and isocahedron projections to name only a few [5].

While the black lines of constant azimuthal (vertical) and polar (horizontal) angles form a block structure in the ERP-projected image, this is not the case in the spherical domain. It is clearly visible that the different blocks become increasingly distorted with a higher distance to the equator.

Idea



In this work, we propose a novel *motion-plane-adaptive* inter prediction technique (MPA) for 360-degree video that allows to **perform inter prediction on different motion planes in 3D space**. Any motion on these planes is modeled purely using horizontal and vertical shifts while the motion planes themselves can be oriented freely in 3D space. In this way, MPA takes both the spherical characteristics of 360-degree video and the translational nature of most camera and object motion into account. MPA thus is able to more accurately reproduce **the resulting pixel shifts in the 2D projection domain than classical translational techniques are able to**. Due to their narrow field of view, such 3D space considerations are not necessary for conventional perspective video. To further improve the performance of MPA and make it compatible to existing inter prediction techniques, we additionally derive an efficient method to transfer motion information between different motion planes and motion models.



1 如何利用投影来在2D上进行更准确的3D帧间预测



2 不改变投影方式，如何完成3D的帧间预测



3 如何将2D的帧间预测在3D上更好的发挥

Loop Filter Disabled Across Virtual Boundaries



- 1 An overview over related approaches to improving 360-degree video coding is given.
- 2 Briefly recapitulates the traditional inter prediction procedure,
- 3 Introduce the proposed MPA. Within this section, the projection functions required for motion-plane adaptivity are introduced including **a generalized formulation of the perspective projection, the motion-plane-adaptive motion model** is presented, an **adapted motion vector prediction method** is derived.

Related work:

- Y. Wang, L. Li, D. Liu, F. Wu, and W. Gao, "A New Motion Model for Panoramic Video Coding," in ICIP, Sep 2017, pp. 1407–1411.
- Y. Wang, D. Liu, S. Ma, F. Wu, and W. Gao, "Spherical Coordinates Transform-Based Motion Model for Panoramic Video Coding," IEEE JETCAS., vol. 9, no. 1, pp. 98–109, Mar 2019.
- L. Li, Z. Li, M. Budagavi, and H. Li, "Projection Based Advanced Motion Model for Cubic Mapping for 360-Degree Video," in ICIP, Sep 2017, pp. 1427–1431.
- L. Li, Z. Li, X. Ma, H. Yang, and H. Li, "Advanced Spherical Motion Model and Local Padding for 360° Video Compression," TIP, vol. 28, no. 5, pp. 2342–2356, May 2019.

Related Work

Wang et al. propose a 3D translational motion model, where **all pixels in a regarded block on the sphere are shifted in 3D space** according to a 3D motion vector derived from the original 2D motion vector.

A similar approach is followed by *Li et al.* in [12],[13], where the 3D motion vector is derived based on the assumption that two neighboring blocks adhere to the same motion in 3D space (local padding and frame padding).



Y.F Wang 's Method



(a)



(b)

Assumption: Motion in sphere is translation.

L.Li 's Method



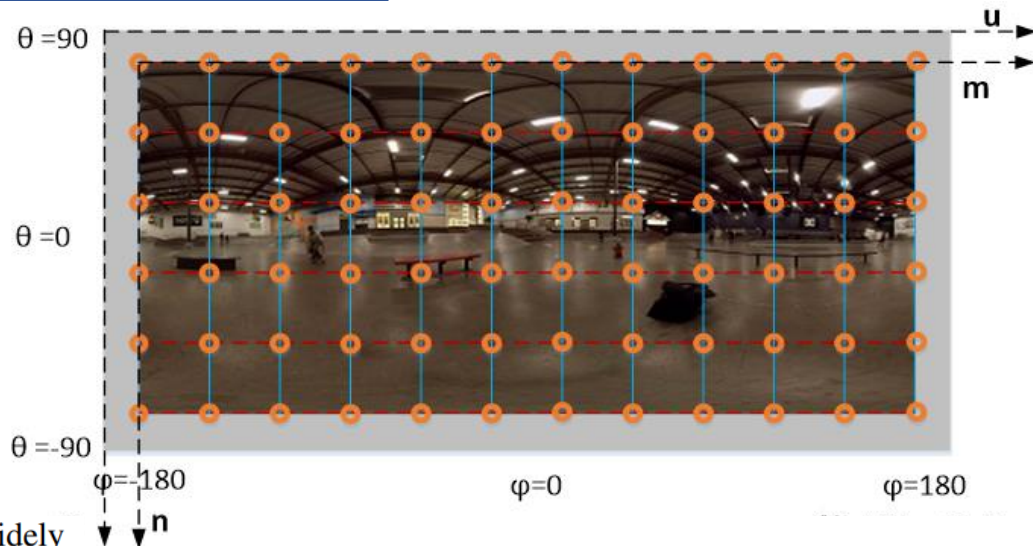
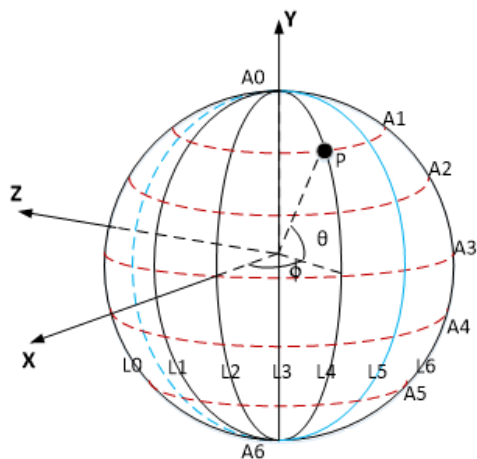
Projection ERP

As MPA is based on the known mappings between the 2D image plane and the 3D space representations of a 360-degree video, a general formulation of these mappings in the form of projection functions is required. Any valid projection function $\xi : \mathcal{S} \rightarrow \mathbb{R}^2$ is invertible and describes the relation between a 3D space coordinate $\mathbf{s} = (x, y, z)^T \in \mathcal{S}$ on the unit sphere and the corresponding pixel coordinate $\mathbf{p} = (u, v)^T \in \mathbb{R}^2$ on the 2D image plane, where $\mathcal{S} = \{\mathbf{s} \in \mathbb{R}^3 \mid \|\mathbf{s}\|_2 = 1\}$ describes the set of all coordinates on the unit sphere. The inverse projection function $\xi^{-1} : \mathbb{R}^2 \rightarrow \mathcal{S}$ maps the 2D image plane coordinate back to the unit sphere in 3D space.



任何的sphere投影都是一种可逆的过程

Projection ERP



The equirectangular projection ξ_{erp} is a popular and widely applied example of a general 360-degree projection. It maps the polar angle $\theta \in [0, \pi]$ to the vertical v -axis and the azimuthal angle $\varphi \in [0, 2\pi]$ to the horizontal u -axis of the 2D image plane. For projecting a 3D space coordinate s on the unit sphere to the 2D image plane, its spherical angles (θ, φ) according to Fig. 2 need to be obtained first through

$$\theta = \arccos(z), \quad (4)$$

$$\varphi = \arctan2(y, x), \quad (5)$$

where $\arctan2$ describes the four-quadrant arctangent. The spherical angles are then projected to the 2D image plane yielding the pixel coordinate $p_{erp} = (u_{erp}, v_{erp})^T \in \mathbb{R}^2$ as

$$u_{erp} = \frac{\varphi}{2\pi} \cdot U, \quad (6)$$

$$v_{erp} = \frac{\theta}{\pi} \cdot V, \quad (7)$$

where U describes the width and V the height of the 2D image plane in pixels. Typically, $U = 2V$ in case of the equirectangular projection as the azimuthal angle φ has twice the angular range compared to the polar angle θ . The equirectangular projection function ξ_{erp} combines steps (4)-(7) in a concise expression.

Projection ERP



For the inverse equirectangular projection ξ_{erp}^{-1} , the pixel coordinate on the 2D image plane is projected back to the spherical domain through

$$\varphi = \frac{u_{\text{erp}}}{U} \cdot 2\pi, \quad (8)$$

$$\theta = \frac{v_{\text{erp}}}{V} \cdot \pi, \quad (9)$$

before the final pixel coordinate on the unit sphere is obtained as

$$x = \sin(\theta) \cos(\varphi), \quad (10)$$

$$y = \sin(\theta) \sin(\varphi), \quad (11)$$

$$z = \cos(\theta). \quad (12)$$

Projection Generalized Perspective Projection



The orientation of the applied 3D coordinate system (x, y, z) is visualized in black in Fig. 2. Thereby, y is oriented horizontally, z is oriented vertically and x is oriented perpendicular to y and z . The default camera is positioned at the origin and oriented in negative x -direction. The rotated blue coordinate system (x', y', z') is an intermediate system for the generalized perspective projection, which will be introduced later.

For MPA, two projection functions are important. First, the employed 360-degree projection ξ_o of the given video, and second, the perspective projection ξ_p for representing motion on the desired motion planes in 3D space.

只需要坐标系的转换

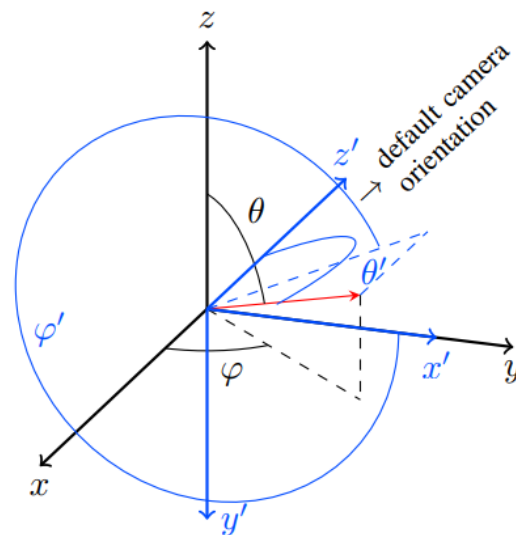


Fig. 2. The employed 3D coordinate systems. The black coordinate system (x, y, z) describes the main system orientation where y is oriented horizontally, z is oriented vertically, and x is oriented perpendicular to y and z . The default camera is positioned at the origin and oriented in negative x -direction. θ and φ denote the corresponding polar and angular angles in spherical coordinates. The blue coordinate system (x', y', z') describes an intermediate system used for the perspective projection where the virtual perspective camera is oriented in positive z' -direction. The corresponding polar and angular angles θ' and φ' are given in blue.

Projection Generalized Perspective Projection



The inverse projection then entails the following steps. First, the polar coordinates (r_p, φ') need to be calculated from the pixel coordinate \mathbf{p}_p

$$r_p = \sqrt{u_p^2 + v_p^2}, \quad (19)$$

$$\varphi' = \arctan2(v_p, u_p). \quad (20)$$

The corresponding incident angle is then obtained using

$$\theta' = \begin{cases} \arctan(r_p/f) & \text{if } b_{vip} = 0, \\ \pi - \arctan(r_p/f) & \text{if } b_{vip} = 1. \end{cases} \quad (21)$$

Finally, the pixel coordinate \mathbf{s} in the spherical domain results in

$$x = -\cos(\theta'), \quad (22)$$

$$y = \sin(\theta') \cos(\varphi'), \quad (23)$$

$$z = -\sin(\theta') \sin(\varphi'). \quad (24)$$

转换方式

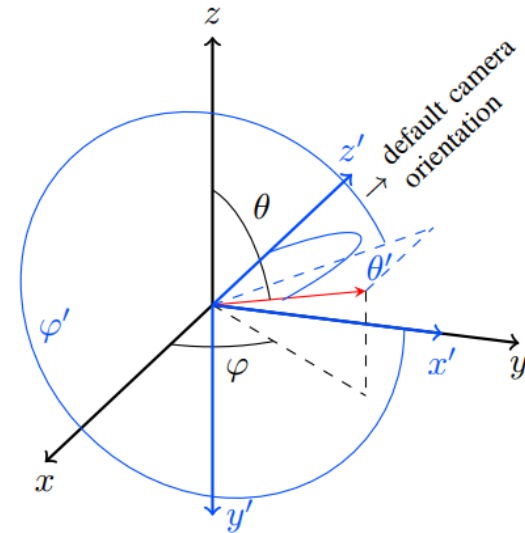
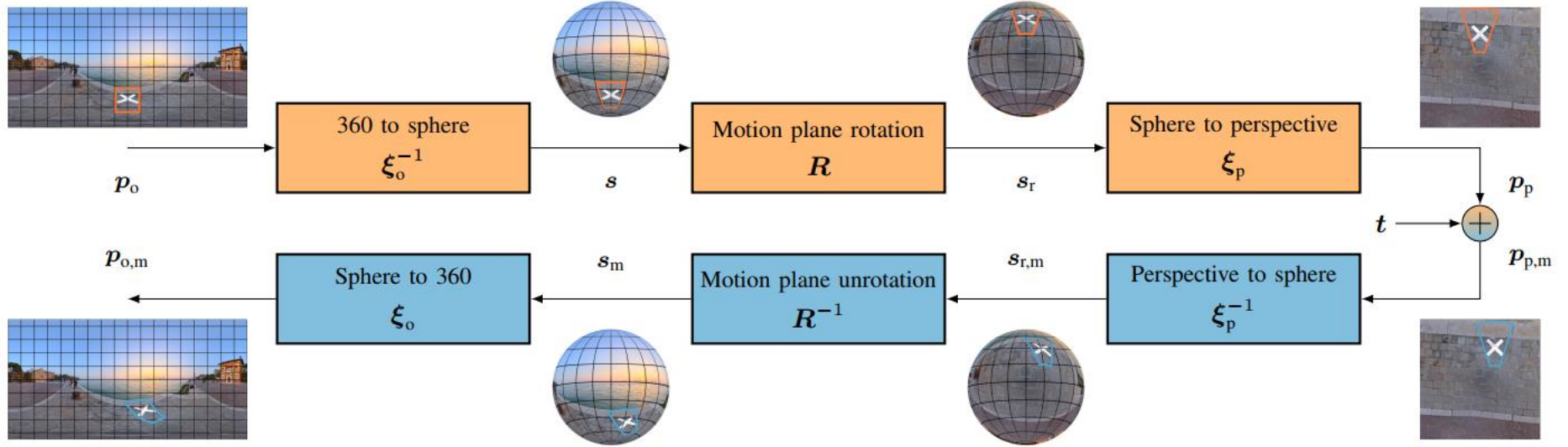


Fig. 2. The employed 3D coordinate systems. The black coordinate system (x, y, z) describes the main system orientation where y is oriented horizontally, z is oriented vertically, and x is oriented perpendicular to y and z . The default camera is positioned at the origin and oriented in negative x -direction. θ and φ denote the corresponding polar and angular angles in spherical coordinates. The blue coordinate system (x', y', z') describes an intermediate system used for the perspective projection where the virtual perspective camera is oriented in positive z' -direction. The corresponding polar and angular angles θ' and φ' are given in blue.

Motion Model



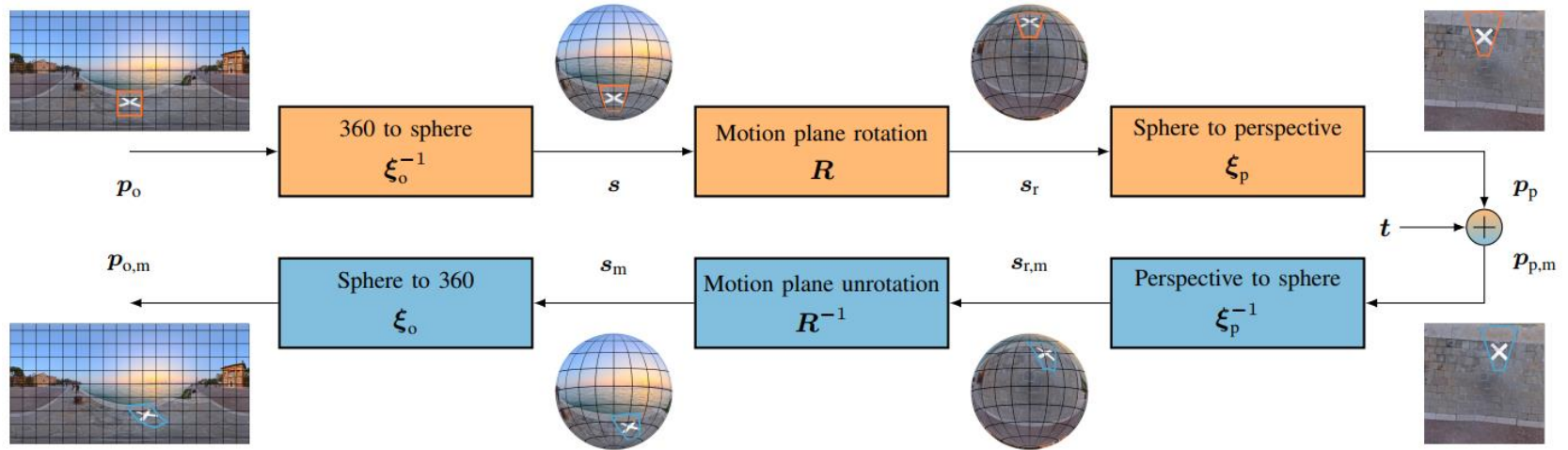
Based on the generalized perspective projection function ξ_p and a given 360-degree projection function ξ_o such as the equirectangular projection function ξ_{erp} described in Section IV-A, the motion-plane-adaptive motion model is derived as follows.

In a first step, the original 360-degree pixel coordinate p_o (in, e.g., the ERP domain) is projected to the pixel coordinate p_p on the desired motion plane using

$$p_p = \xi_p (R\xi_o^{-1}(p_o)), \quad (25)$$

where ξ_o^{-1} projects the original 360-degree pixel coordinate p_o to the unit sphere, the motion plane rotation matrix $R \in \mathbb{R}^{3 \times 3}$ rotates the pixel coordinate on the unit sphere according to the desired motion plane orientation, and ξ_p then projects the pixel coordinate onto the motion plane.

Motion Model



In a second step, the translational motion according to the motion vector t is performed on the obtained motion plane yielding the moved pixel coordinate on the motion plane

$$p_{p,m} = p_p + t. \quad (26)$$

In the final third step, the moved pixel coordinate $p_{p,m}$ on the motion plane is projected back to the original 360-degree format to obtain the moved pixel coordinate in the 360-degree projection

$$p_{o,m} = \xi_o (R^{-1} \xi_p^{-1} (p_{p,m})), \quad (27)$$

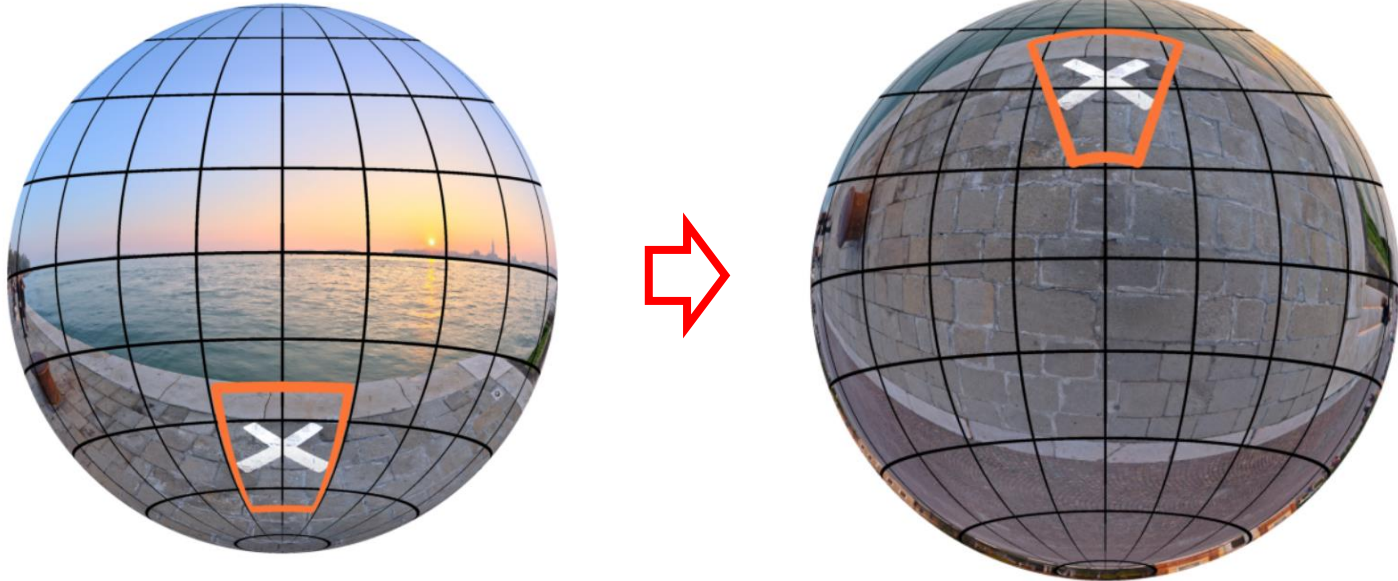
where $R^{-1} = R^T$.

Putting steps (25)-(27) together, the overall motion model m_{mpa} is defined as

$$m_{mpa}(p_o, t, R) = \xi_o (R^{-1} \xi_p^{-1} (\xi_p (R \xi_o^{-1} (p_o)) + t)). \quad (28)$$

A schematic representation of the described motion model is shown in Fig. 4. The figure also visualizes block motion for an exemplary block in an ERP-projected 360-degree image, where it is clearly visible that the proposed motion model is able to accurately replicate the distortions of the block in the ERP domain resulting from a translational motion on the street surface.

Motion Estimation

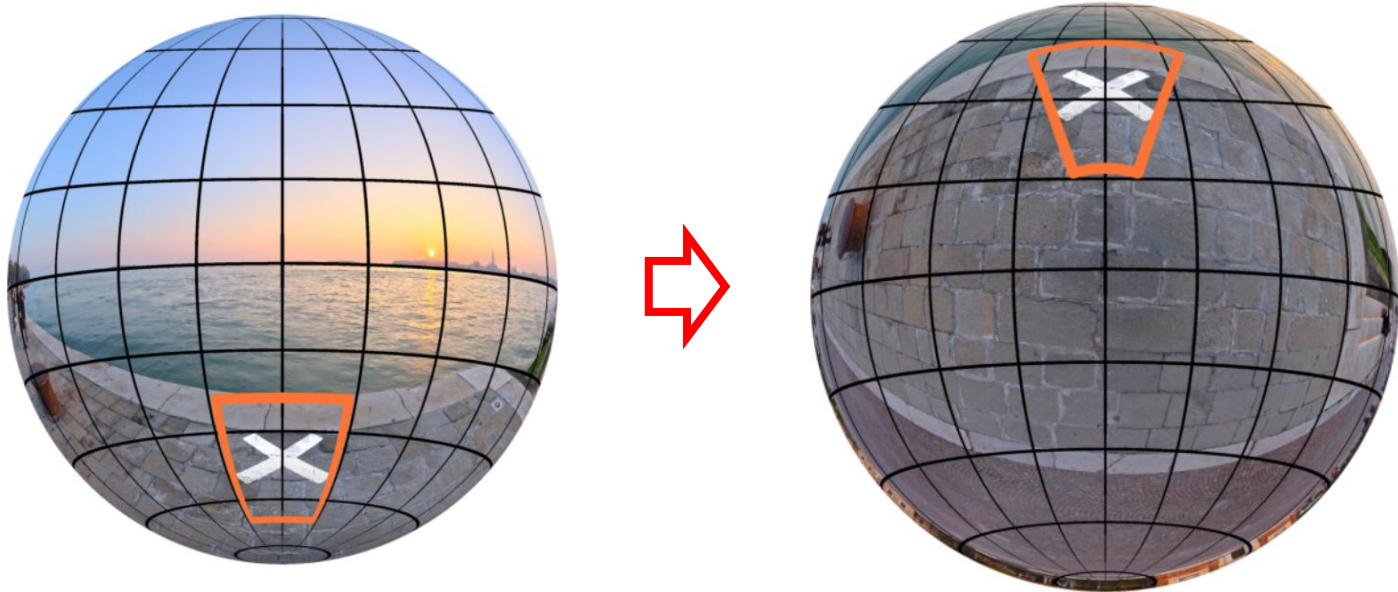


Rotations by multiples of 90 degrees around one or more axes can be formulated through simple transpositions of the 3D space coordinates, such that a suitably defined set of motion planes allows to considerably speed up the calculations for the inherent coordinate rotations. We thus formulate a limited set of three rotation matrices leading to the motion planes


- front/back: No rotation,
- left/right: $\pi/2$ around z -axis,
- top/bottom: $\pi/2$ around y -axis.

A potential encoder can then select the best matching motion plane through rate-distortion optimization. Please note, however, that in general, the motion-plane-adaptive motion model is not limited to these motion planes and could employ arbitrary motion plane rotation matrices R .

Motion Estimation



However, in the context of MPA, the motion information at the different candidate positions could be represented on motion planes differing from the investigated motion plane for the currently regarded block. Hence, a method to efficiently translate motion information, i.e., motion vectors, between different motion planes is required.

 Motion estimation in different planes

Motion Estimation for Different Plane

However, in the context of MPA, the motion information at the different candidate positions could be represented on motion planes differing from the investigated motion plane for the currently regarded block. Hence, a method to efficiently translate motion information, i.e., motion vectors, between different motion planes is required.

According to Fig. 5, let's assume a motion vector t_s and a motion plane rotation matrix R_s are obtained at a source candidate pixel coordinate p_s . To transform the motion vector to a different target motion plane described by the rotation matrix R_t , we need to find a motion vector t_t on the target motion plane that results in an identical pixel shift in the 360-degree domain as the motion vector t_s on the source motion plane, i.e.,

$$m_{\text{mpa}}(p_s, t_s, R_s) \stackrel{!}{=} m_{\text{mpa}}(p_s, t_t, R_t). \quad (29)$$

The source pixel coordinate p_s is hereby used as an anchor for the motion plane translation. By inserting (28) into (29), we can solve for the required motion vector on the target motion plane as

$$t_t = \xi_p (R_t R_s^{-1} \xi_p^{-1} (\xi_p (R_s \xi_o^{-1}(p_s)) + t_s)) - \xi_p (R_t \xi_o^{-1}(p_s)).$$

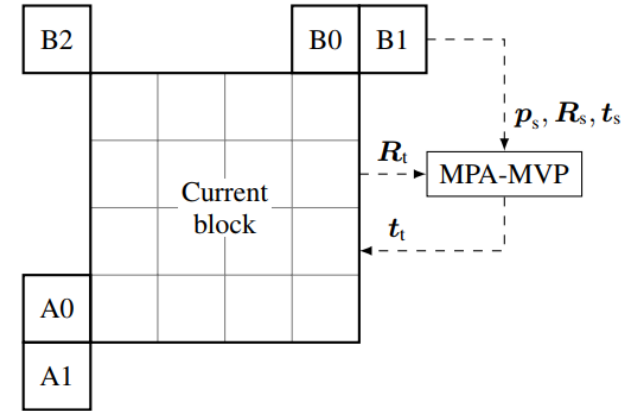


Fig. 5. Spatial motion vector predictor candidates in the H.266/VVC video coding standard [24] with a schematic illustration of MPA-MVP for candidate position B1 at pixel coordinate p_s . With R_s describing the motion plane of the candidate position and R_t describing the motion plane of the current block, MPA-MVP translates the motion vector t_s from B1 to the corresponding motion vector predictor t_t for the current block.

Motion Estimation for Different Plane

$$t_t = \xi_p (R_t R_s^{-1} \xi_p^{-1} (\xi_p (R_s \xi_o^{-1} (p_s)) + t_s)) - \xi_p (R_t \xi_o^{-1} (p_s)).$$

Thereby, the first row explains, where the candidate pixel coordinate moved by t_s on the source motion plane lies on the target motion plane, and the second row explains, where the "unmoved" candidate pixel coordinate lies on the target motion plane. Fig. 5 visualizes the general procedure of the described motion-plane-adaptive motion vector prediction (MPA-MVP) for an exemplary candidate position B1.

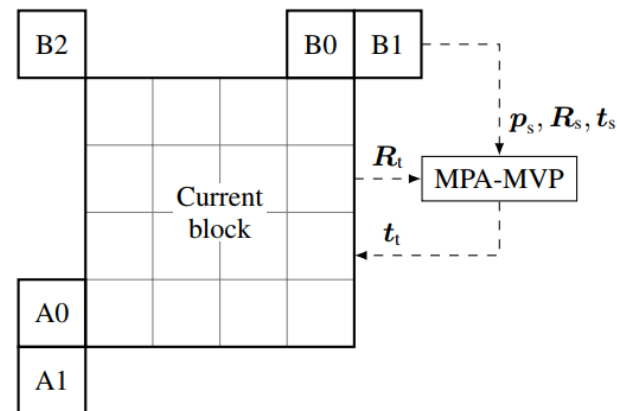


Fig. 5. Spatial motion vector predictor candidates in the H.266/VVC video coding standard [24] with a schematic illustration of MPA-MVP for candidate position B1 at pixel coordinate p_s . With R_s describing the motion plane of the candidate position and R_t describing the motion plane of the current block, MPA-MVP translates the motion vector t_s from B1 to the corresponding motion vector predictor t_t for the current block.

Motion Estimation for translational Plane

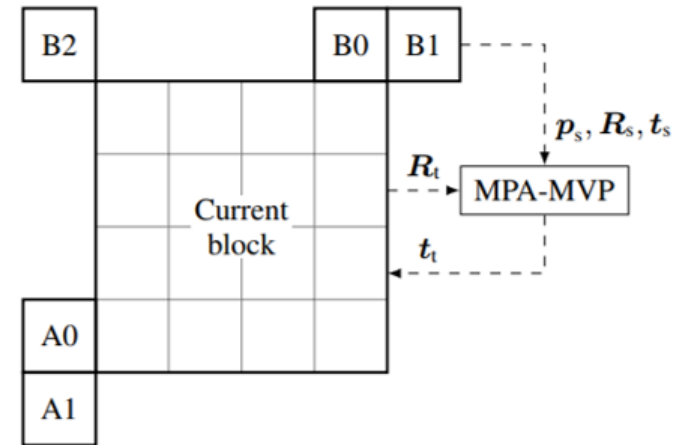
In case the motion-plane-adaptive motion model is the source model, an equivalent motion vector t_t for the classical translational motion model can be derived based on the motion vector t_s on the source motion plane as

$$t_t = \xi_o (R_s^{-1} \xi_p^{-1} (\xi_p (R_s \xi_o^{-1}(p_s)) + t_s)) - p_s. \quad (32)$$

On the other hand, in case the classical translational motion model is the source model, an equivalent motion vector t_t for the motion-plane-adaptive motion model can be derived based on the source motion vector t_s and the desired target motion plane as

$$t_t = \xi_p (R_t \xi_o^{-1}(p_s + t_s)) - \xi_p (R_t \xi_o^{-1}(p_s)). \quad (33)$$

The key differences between (32) and (33) are the domain in which the source motion vector is applied, and the domain in which the target motion vector is calculated. In (32), the source motion vector is applied on the source motion plane and the target motion vector is calculated in the 360-degree domain, whereas in (33), the source motion vector is applied in the 360-degree domain and the target motion vector is calculated on the target motion plane.



Motion Vector Coding

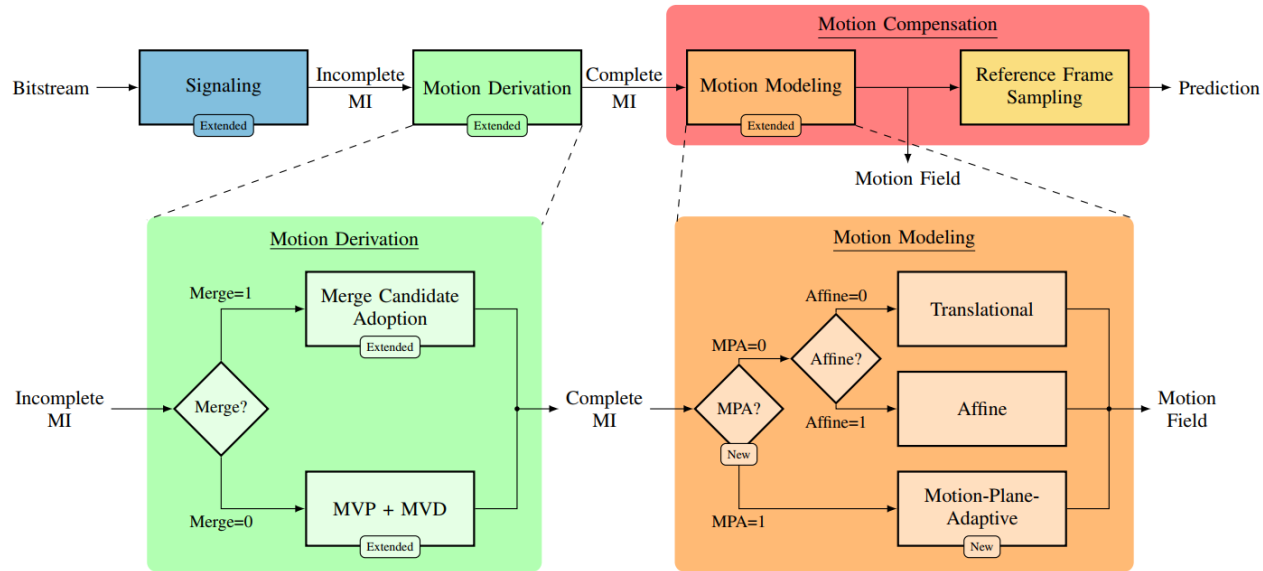


Fig. 6. Schematic representation of the decoder-side inter prediction pipeline in the H.266/VVC video coding standard including the proposed MPA tool. Components that need to be extended or added with respect to the original H.266/VVC inter prediction pipeline are labeled explicitly. For details on the performed extensions and additions, please see the text.

If a merge mode is signaled or the current CU is coded using an affine motion model, no adaptations to the signaling procedure are required for MPA. If no merge mode is signaled and the current CU is not coded using an affine motion model, the integration of MPA requires additional information to be signaled. The MPA-specific information (MPA flag, motion plane index) is signaled right after the affine flag and starts with a first bin encoding the MPA flag. If this is true, a

second bin denotes whether the front/back motion plane is used, and, if this is not the case, a third bin denotes whether the left/right or the top/bottom motion plane is used. Similar to the existing motion information (MI) in the H.266/VVC video coding standard, the described information is signaled using the context-based binary arithmetic coder (CABAC) [27] with dedicated context models. In case of bi-prediction, i.e., two predictions with different MI are averaged to form an overall

Motion Compensation



To reduce the computational complexity of the motion-plane-adaptive motion modeling, the motion model is executed on 4×4 subblocks similar to the realization of the newly introduced 4-parameter and 6-parameter affine motion models [28], [29] as shown in Fig. 7. Hence, the resulting pixel shift needs to be calculated only once per 4×4 subblock and all pixels within a subblock share the same pixel shift. This greatly speeds up the motion modeling procedure.

The precise pixel position within each subblock, at which the motion-plane-adaptive motion model is evaluated, can in principle be chosen arbitrarily. While intuitively, one would choose the center position of the subblock, our experiments showed that better results are obtained using one of the actual pixel positions surrounding the floating point center position.

For further evaluations, we use the pixel position in the second row and second column of each subblock as shown in blue in Fig. 7.

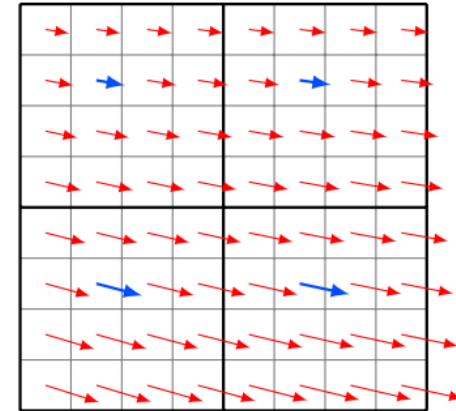


Fig. 7. Exemplary 4×4 subblock partitioning in MPA. Without subblock partitioning, the resulting pixel shift has to be calculated for each pixel individually (all arrows). With subblock partitioning, the pixel shift resulting from the motion-plane-adaptive motion model is only calculated for one pixel position within each 4×4 subblock (blue arrows).



Other Task for Panoramic Video



End-to-End Optimized 360° Image Compression

Mu Li^{ID}, Jinxing Li^{ID}, *Member, IEEE*, Shuhang Gu^{ID}, Feng Wu, *Fellow, IEEE*,
and David Zhang^{ID}, *Life Fellow, IEEE*

Introduction

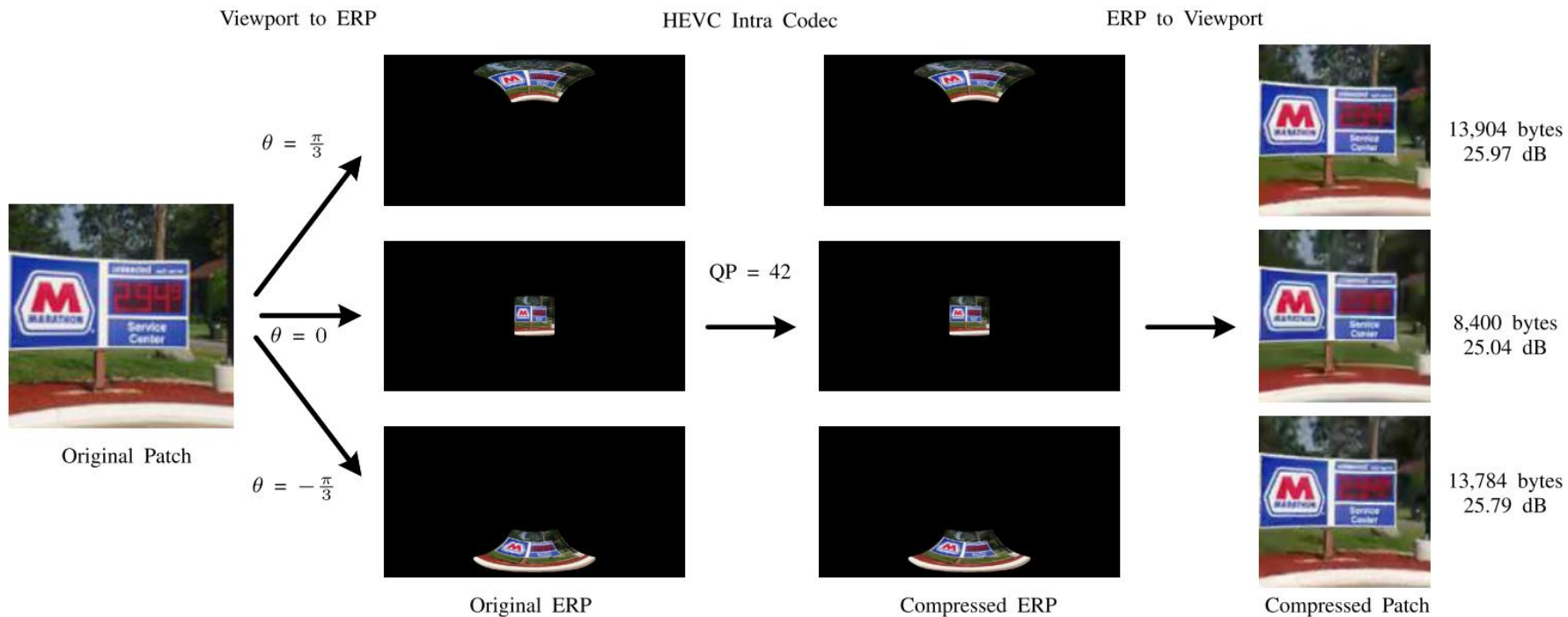


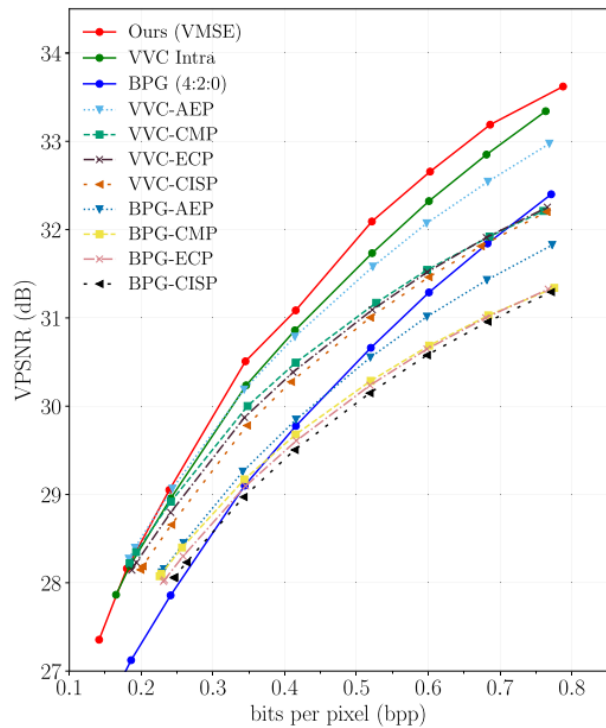
Fig. 2. Illustration of the unbalance sampling problem of equirectangular projection. We first project the same image patch to different latitudes of the ERP images (padded with zeros), and compress them by the HEVC intra coding with the identical hyperparameter. The performance is given in the format of bytes/peak signal-to-noise ratio (PSNR in dB).

Contribution

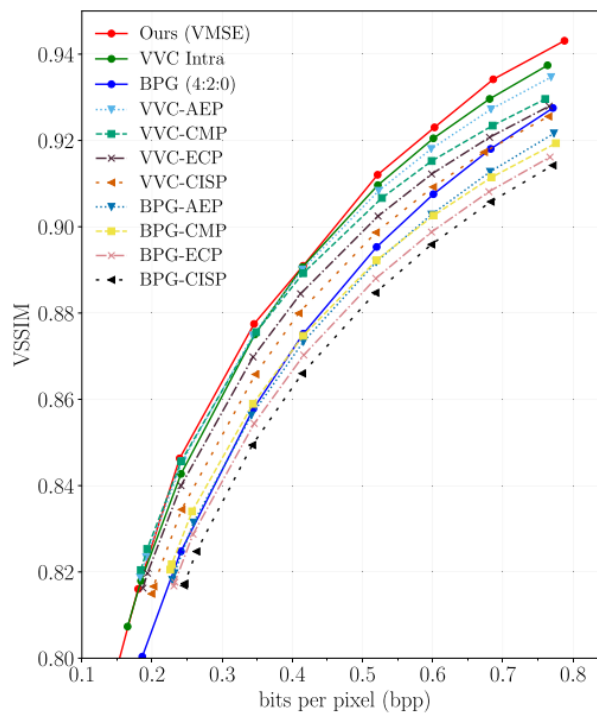


- We introduce a general regional adaptive rate loss by considering both the structure and entropy of codes in different locations. Based on the regional adaptive rate loss, we propose latitude adaptive constraints to guide the code allocation in different latitudes to tackle the unbalanced sampling problem of ERP images. The proposed rate-controlling scheme can serve as a general regional resampling solution, *i.e.*, channel resampling, for learned 360° image compression methods.
- The viewport-based distortion loss and the spherical padding are proposed to avoid the deformation problem in ERP image quality assessment.
- We collect a large high-quality 360° image dataset for 360° image compression, which can be further adopted in omnidirectional image super-resolution and denoising.

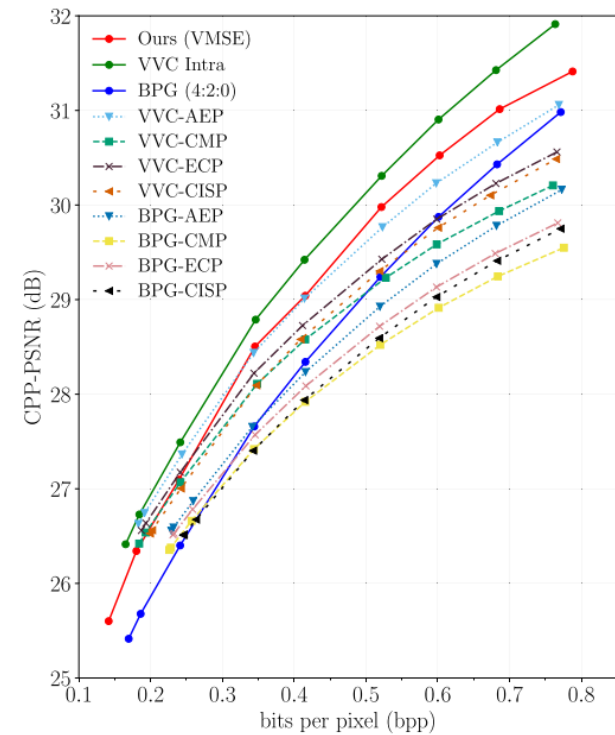
Performance



(a)




(b)



(c)



Deep 360° Optical Flow Estimation Based on Multi-Projection Fusion

Yiheng Li¹, Connelly Barnes², Kun Huang¹, and Fang-Lue Zhang^{1,*} 

¹ School of Engineering and Computer Science, Victoria University of Wellington
{Yiheng.Li, Kun.Huang, Fanglue.Zhang}@vuw.ac.nz

² Adobe Research, Seattle, US
ConnellyBarnes@gmail.com

Introduction

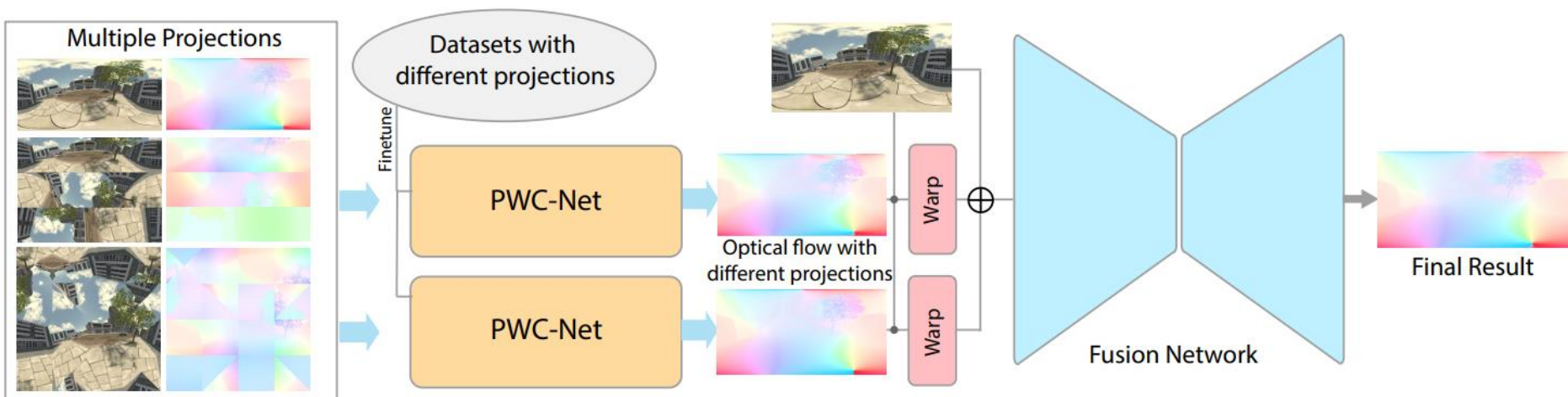


Fig. 1. Our multi-projection fusion method for estimating 360° optical flow

SD/EPE	PWC-Net	RAFT	TanImg	FlowNet2	Ours
City100	3.86/9.84	12.10/21.57	1.27/3.69	2.72/10.85	0.82/1.79
EFT100	6.26/15.64	20.21/29.64	4.61/8.06	4.91/14.88	2.64/5.01
Average	5.06/12.74	16.15/25.61	2.94/5.88	3.81/12.87	1.73/3.40

Table 5. Comparison with other methods using Spherical Distance (SD) / EPE.

Contribution



In summary, this paper makes the following contributions: 1. We introduce a fusion framework for learning 360° optical flow, which learns to fuse the complementary optical flow prediction results generated using different projections; 2. We apply common map projection techniques to make two spherical image projection methods, namely tri-cylinder projection and cube padding projection that are complementary to equirectangular projection in terms of distorted regions and performance under fusion; 3. We build a novel 360° optical flow dataset with ground truth optical flow data with various camera paths and dynamic objects.



LAU-Net: Latitude Adaptive Upscaling Network for Omnidirectional Image Super-resolution

Xin Deng^{1, *}, Hao Wang^{2, *}, Mai Xu^{2, †}, Yichen Guo², Yuhang Song³, Li Yang²

¹School of Cyber Science and Technology, Beihang University, Beijing, China

²School of Electronic and Information Engineering, Beihang University, Beijing, China

³Department of Computer Science, University of Oxford, UK

{cindydeng, wang_hao, MaiXu, ycguo, LiYang2018}@buaa.edu.cn,

Yuhang.Song@some.ox.ac.uk

Introduction

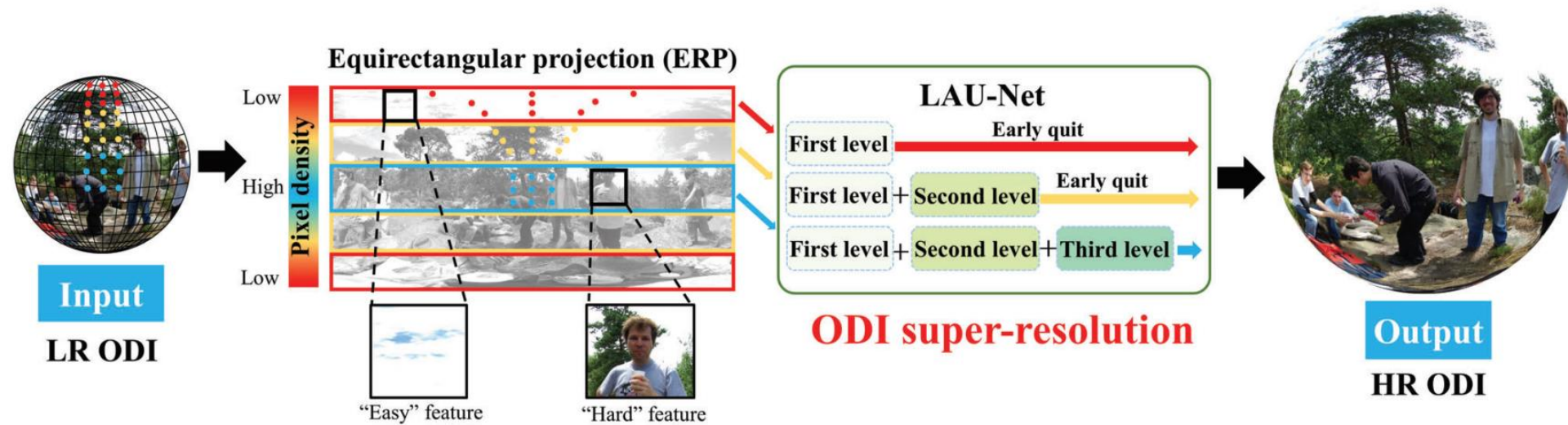


Figure 1. The basic framework of our method for omnidirectional image super-resolution.

Table 1. The average and standard deviation of WS-PSNR (dB) and WS-SSIM results of different methods. The red values indicate the best and the blue values indicate the second best results.

Method	8×				16×			
	ODI-SR		SUN 360 Panorama		ODI-SR		SUN 360 Panorama	
	WS-PSNR	WS-SSIM	WS-PSNR	WS-SSIM	WS-PSNR	WS-SSIM	WS-PSNR	WS-SSIM
Bicubic	19.64±2.96	0.5908±0.0834	19.72±3.15	0.5403±0.0862	17.12±3.06	0.4332±0.0845	17.56±3.06	0.4638±0.0848
SRCNN	20.08±1.65	0.6112±0.0712	19.46±1.83	0.5701±0.0819	18.08±2.03	0.4501±0.0806	17.95±2.12	0.4684±0.0813
VDSR	20.61±1.74	0.6195±0.0796	19.93±1.91	0.5953±0.0798	18.24±2.35	0.4996±0.0824	18.21±2.47	0.4867±0.0829
LapSRN	20.72±1.89	0.6214±0.0823	20.05±2.51	0.5998±0.0816	18.45±2.54	0.5161±0.0861	18.46±2.53	0.5068±0.0841
MemNet	21.73±1.84	0.6284±0.0802	21.08±2.35	0.6015±0.0875	20.03±2.68	0.5411±0.0822	19.88±2.13	0.5401±0.0830
MSRN	22.29±1.86	0.6315±0.0815	21.34±2.43	0.6002±0.0918	20.05±3.02	0.5416±0.0968	19.87±3.27	0.5316±0.0976
EDSR	23.97±1.74	0.6417±0.0724	22.46±2.32	0.6341±0.0861	21.12±2.58	0.5698±0.0829	21.06±2.49	0.5645±0.0864
D-DBPN	24.15±1.72	0.6573±0.0758	23.70±2.25	0.6421±0.0858	21.25±2.42	0.5714±0.0831	21.08±2.45	0.5646±0.0918
RCAN	24.26±1.68	0.6628±0.0714	23.88±2.02	0.6542±0.0824	21.94±1.75	0.5824±0.0815	21.74±2.28	0.5742±0.0892
EBRN	24.29±1.72	0.6656±0.0698	23.89±2.04	0.6598±0.0832	21.86±1.68	0.5809±0.0792	21.78±2.12	0.5794±0.0842
360-SS	21.65±1.91	0.6417±0.0865	21.48±2.56	0.6352±0.0872	19.65±2.44	0.5431±0.0868	19.62±2.96	0.5308±0.0879
LAU-Net	24.36±1.73	0.6801±0.0736	24.02±2.13	0.6708±0.0801	22.07±1.74	0.5901±0.0812	21.82±2.36	0.5824±0.0865

Contribution



The omnidirectional images (ODIs) are usually at low-resolution, due to the constraints of collection, storage and transmission. The traditional two-dimensional (2D) image super-resolution methods are not effective for spherical ODIs, because ODIs tend to have non-uniformly distributed pixel density and varying texture complexity across latitudes. In this work, we propose a novel latitude adaptive upscaling network (LAU-Net) for ODI super-resolution, which allows pixels at different latitudes to adopt distinct upscaling factors. Specifically, we introduce a Laplacian multi-level separation architecture to split an ODI into different latitude bands, and hierarchically upscale them with different factors. In addition, we propose a deep reinforcement learning scheme with a latitude adaptive reward, in order to automatically select optimal upscaling factors for different latitude bands. To the best of our knowledge, LAU-Net is the first attempt to consider the latitude difference for ODI super-resolution. Extensive results demonstrate that our LAU-Net significantly advances the super-resolution performance for ODIs. Codes are available at <https://github.com/wangh-allen/LAU-Net>.



SphereSR: 360° Image Super-Resolution with Arbitrary Projection via Continuous Spherical Image Representation

Youngho Yoon, Inchul Chung, Lin Wang*, and Kuk-Jin Yoon
Visual Intelligence Lab., KAIST, Korea
{dudgh1732, inchul1221, wanglin, kjyoon}@kaist.ac.kr

Introduction

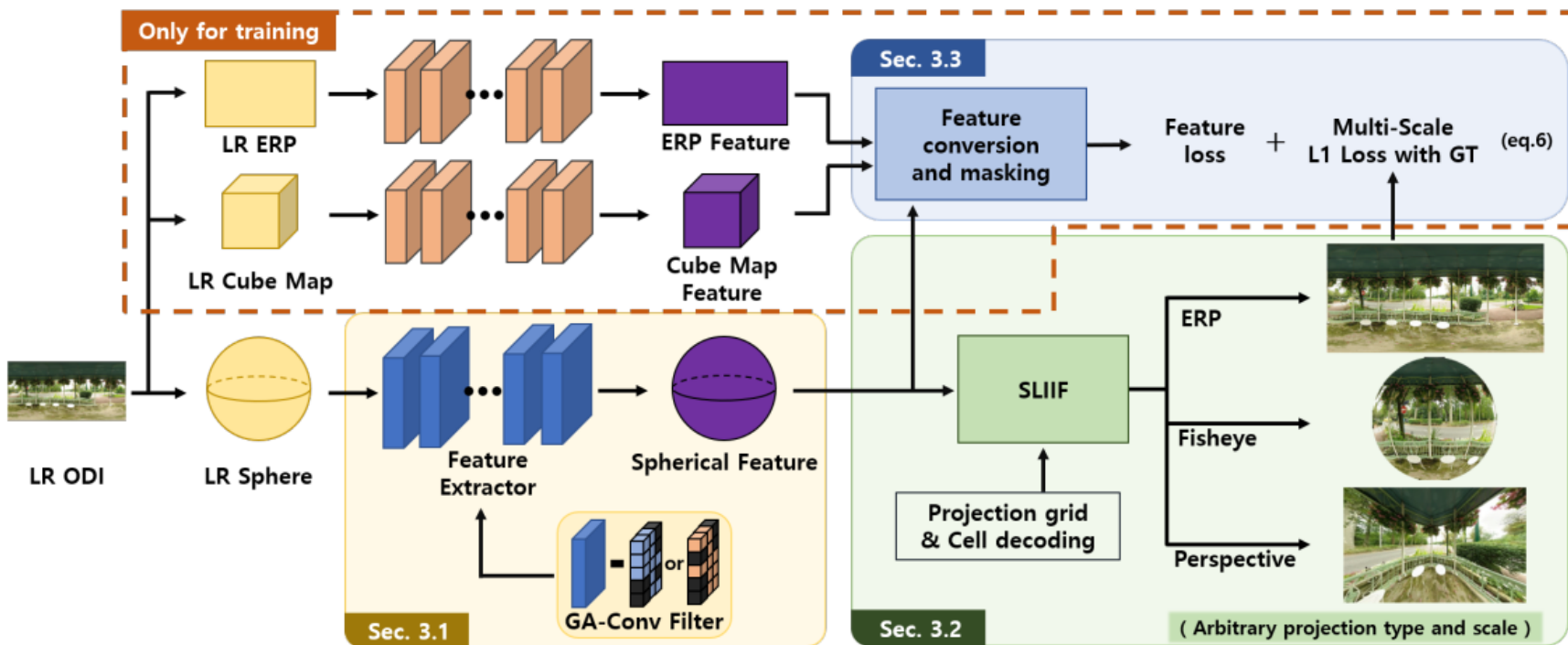


Figure 2. Overall framework of the proposed SphereSR.

Contribution



In this paper, as shown in Fig. 1, we propose a novel framework, called SphereSR, with the goal of super-resolving an LR 360° image to an HR image *with an arbitrary projection type via continuous spherical image representation*. First, we propose a feature extraction module that represents spherical data based on icosahedron and efficiently extracts features on a spherical surface composed of uniform faces (Sec. 3.1). As such, we solve the ERP image distortion problem and resolve the pixel density difference according to the latitude. Second, we propose a spherical local implicit image function (SLIIF) that can predict RGB values at arbitrary coordinates on a sphere feature map, inspired by LIIF [7] (Sec. 3.2). SLIIF works on *triangular faces*, buttressed by position embedding based on normal plane polar coordinates to obtain relative coordinates on a sphere. Therefore, our method tackles pixel-misalignment issue when the image is projected onto another ODI projection. As a result, SphereSR can predict RGB values for any SR scale parameters. Additionally, to train SphereSR, we introduce a feature loss that measures the similarity between two projection types, leading to a considerable performance enhancement (Sec. 3.3). Extensive experiments on various benchmark datasets show that our method significantly surpasses existing methods.



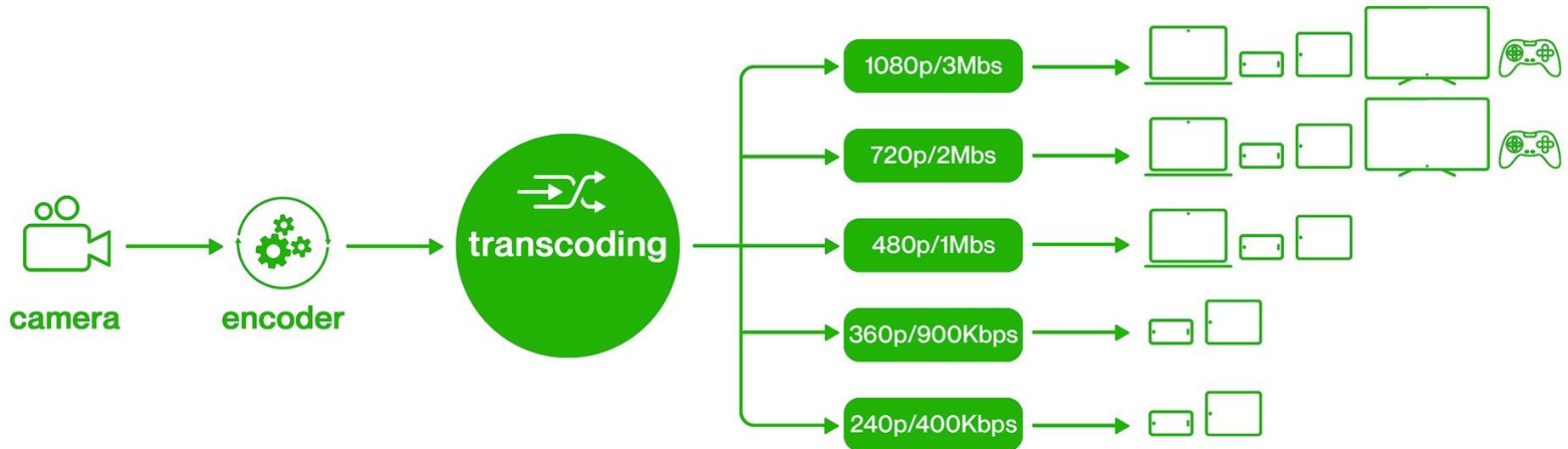
An Introduction for Transcoding (case study)

Hot Period in Coding: 2007-2014

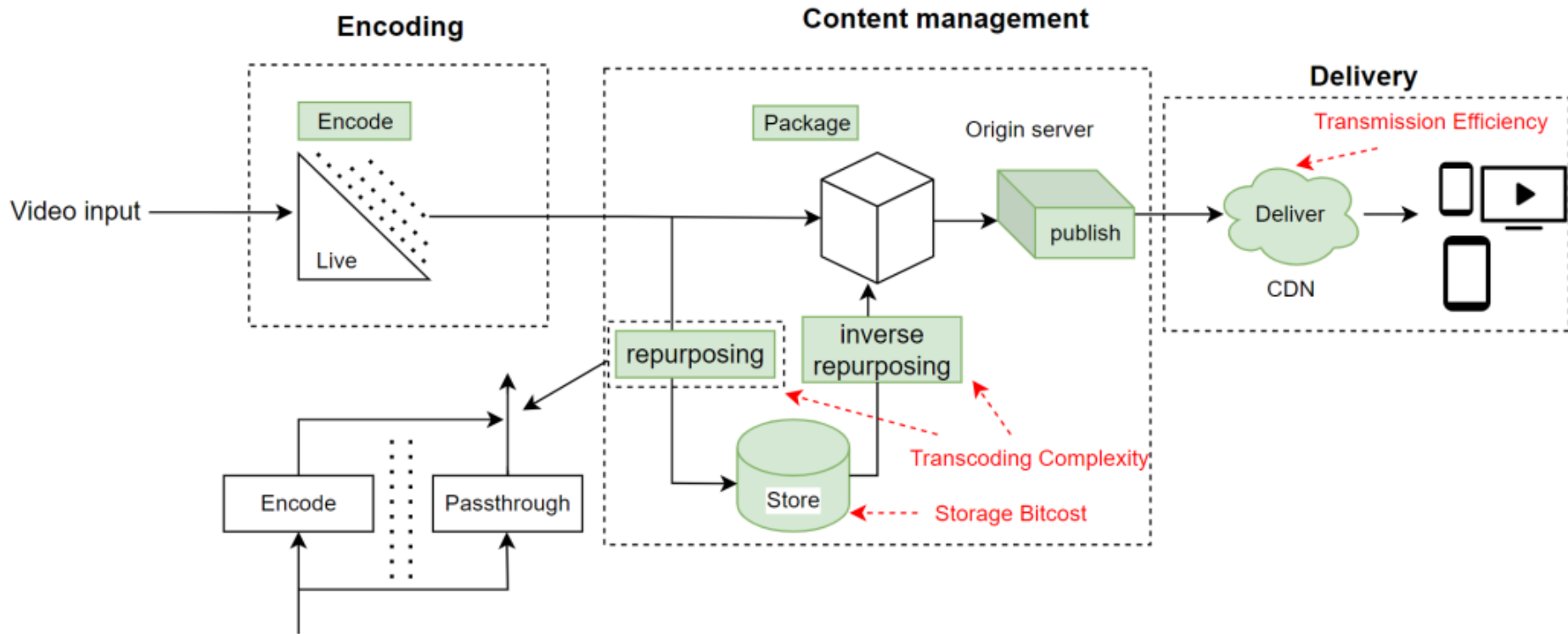
iVC Inter Group

2024.07.30

Framework



Basic Scheme



Overview

Basic Scheme



Method	Performance		
	Storage	Transcoding Complexity	Transmission Efficiency
Simulcast [22]	Highest	Lowest	Highest
Full Transcoding [22]	Lowest	Highest	Lowest
Guided Transcoding [26], [27]	Low	Low	Lowest
Guided Transcoding using Deflation and Inflation [17]	Low	Low	Highest

Comparison

Guided Transcoding

Deflation

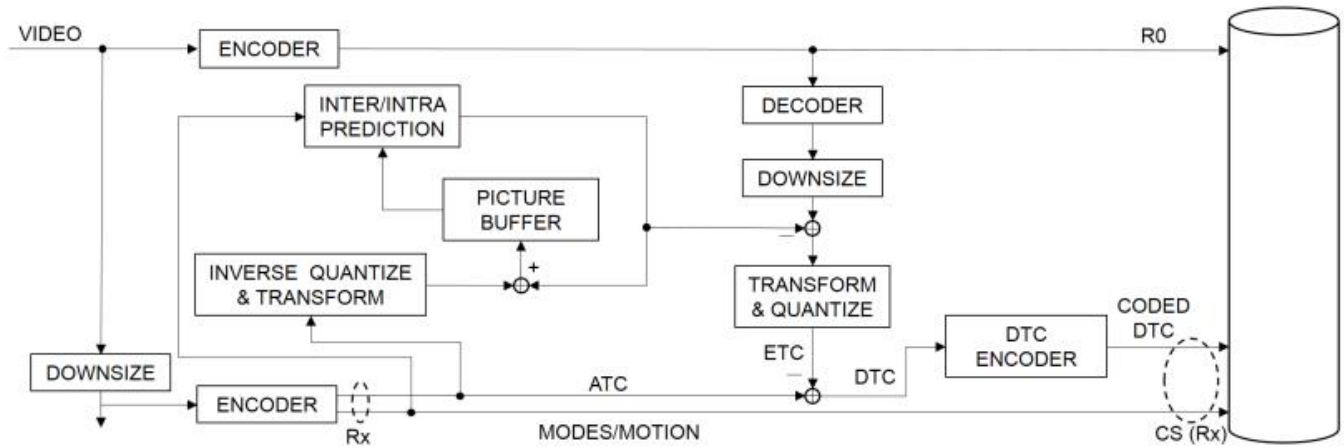


Figure 5: Schematic overview of deflation

Inflation

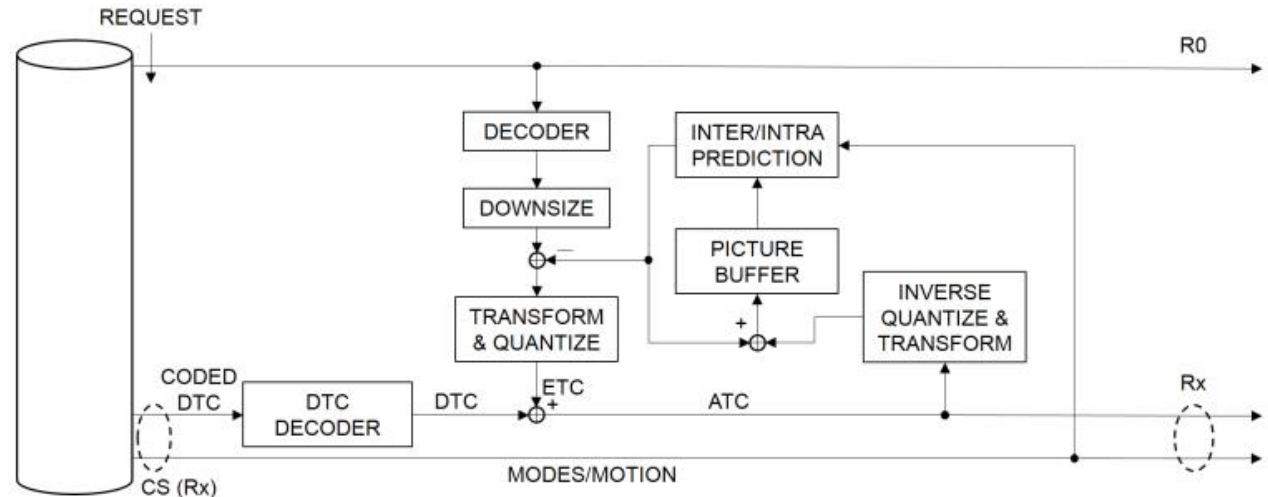


Figure 6: Schematic overview of inflation