A NEW MOTION MODEL FOR PANORAMIC VIDEO CODING

Yefei Wang[1], *Li Li*[2], *Dong Liu*[1]*, *Feng Wu*[1], *Wen Gao*[3]

[1]CAS Key Laboratory of Technology in Geo-Spatial Information Processing and Application System, University of Science and Technology of China, Hefei 230027, China

[2]University of Missouri-Kansas City, 5100 Rockhill Road, Kansas City, MO 64111, USA

[3]Peking University, Beijing 100871, China

ABSTRACT

Virtual reality (VR) has been a hot topic in both research and industry, calling for more efficient compression of panoramic videos. Currently, panoramic video is played as if it is spherical, but such video is actually mapped to planar video, e.g. using equirectangular projection, before compression and transmission. The projection causes deformation and thus makes the traditional translational motion model not efficient. In this paper, we propose a new motion model based on spherical coordinates transform to compensate for the deformation in panoramic videos. Our model requires no additional motion vector but rather derives pixel-wise 2D motion vectors from a block-level 3D motion vector. Our experimental results show the significant bits saving achieved by the new model, which leads to as high as 8.0% BD-rate on the test sequences.

Index Terms— Equirectangular projection, Motion model, Panoramic video, Spherical coordinates, Video coding.

1. INTRODUCTION

Virtual reality (VR) has been a hot topic recently [1], which calls for more efficient compression of panoramic videos. The panoramic video is currently played as if it is spherical, e.g. in head-mounted displays, but for ease of compression and transmission, such panoramic video is actually mapped to planar video, e.g. using equirectangular projection [2]. Note that the mapping can cause large deformation of the video content, for example, the equirectangular projection produces significant deformation, especially in the pole area, as shown in Fig. 1. Therefore, it is necessary to take such deformation into account when compressing the planar video of panoramic content. In this paper, we specifically consider how to compensate for the deformation in the motion estimation (ME) and motion compensation (MC) processes in video coding.

* Corresponding author (email: dongeliu@ustc.edu.cn). This work was supported by the National Program on Key Basic Research Projects (973 Program) under Grant 2015CB351803, by the Natural Science Foundation of China (NSFC) under Grants 61390512, 61331017, and 61632001, and by the Fundamental Research Funds for the Central Universities under Grant WK3490000001.

Fig. 1. The equirectangular projection causes very large deformation in the pole area.

Almost all of the existing video coding standards, including H.264 [3] and HEVC [4], adopt block-based ME and MC, that is, the motion vectors (MVs) of all the pixels inside the same block (prediction unit, PU, in HEVC) are all the same. These methods are efficient to remove temporal redundancy if the real motion is only translation in a plane without change of depth. However, the motion in real-world videos can be very diverse, such as rotation and zooming. There have been some techniques that utilize more complex motion models. For example, Li *et al.* proposed a four-parameter affine motion model to handle rotation and zooming, which achieves good performance on normal videos [5]. However, when the affine motion model is used for panoramic video, it does not provide significant gain, since the deformation caused by equirectangular projection is too complex to be described by affine motion model [6].

Recently, some researches have been done in fisheye video coding to handle the deformation. In [7], an elastic motion model is proposed to describe the deformation in fisheye video, an elastic reference frame is generated for MC and the warping parameters need to be encoded. In [8], an equidistant warping model is used, and in [9], the calibration information is used, both to handle the deformation in fisheye video depending on the camera parameters.

In this paper, we propose a new motion model based on spherical coordinates transform for panoramic video coding. Our key idea is that, each block (PU in HEVC) has one MV to be encoded, but the pixels inside this block have differ-

ent 2D MVs during MC. The derivation of pixel-wise MVs from block-level MV is achieved by using spherical coordinates and estimating relative depth. Compared to previous work [6, 7, 8, 9], our method do not require additional MVs, and is independent of camera parameters. We verify our proposed method on equirectangular projected videos, but the basic idea of our method can be extended to the other kinds of projection as well.

The remainder of this paper is organized as follows. In Section 2, we present the motion model based on spherical coordinates. In Section 3, we discuss the details of ME and MC with the new motion model. Experiment results are given in Section 4, followed by conclusions in Section 5.

2. THE PROPOSED MOTION MODEL

2.1. Spherical coordinates transform and equirectangular projection

Conversion between the 3D Cartesian coordinates and the spherical coordinates is performed by:

$$\begin{aligned} x &= R\cos\phi\sin\theta \\ y &= R\sin\phi\sin\theta \\ z &= R\cos\theta \end{aligned} \tag{1}$$

Note that θ is the latitude, and ϕ is the longitude. R is the distance from the origin, which can be regarded as the depth in panoramic videos. (θ, ϕ) have the following relation to the frame-level coordinates (u, v), according to the equirectangular projection:

$$\theta = \pi\frac{v}{H}, \phi = 2\pi\frac{u}{W} \tag{2}$$

where W and H are width and height of the panoramic video. Therefore, we can calculate the (θ, ϕ) of each pixel but the depth R is unknown. We now propose a method to estimate (relative) depth based on MVs.

2.2. Depth estimation

To start with, we have three assumptions:
1. The depths in a small local region are the same. Indeed, we assume the depths of all the pixels inside a PU are the same. This assumption is valid if the PU is small enough.
2. The motions of all the pixels inside a PU, relative to the camera (i.e. the origin of the spherical coordinates), are the same. Again, this assumption is valid as long as the PU is small enough.
3. The motions of two neighboring PUs relative to the camera are also the same.

Based on these assumptions, we can use the MV of current PU and the MV of one of its neighboring PUs to estimate the depth R.

Fig. 2 shows two frames in a panoramic video, the current frame and the reference. There are two neighboring PUs in the current frame. The center point of the current PU is $\mathbf{c_1} = (\mathbf{u_{1c}}, \mathbf{v_{1c}})$, and the MV is $\mathbf{mv_1}$. The center of the neighboring PU is $\mathbf{c_2} = (\mathbf{u_{2c}}, \mathbf{v_{2c}})$ and the MV is $\mathbf{mv_2}$. We then calculate the locations in the reference frame corresponding to $\mathbf{c_1}$, $\mathbf{c_2}$, i.e. $\mathbf{a_1} = (\mathbf{u_1}, \mathbf{v_1})$ and $\mathbf{a_2} = (\mathbf{u_2}, \mathbf{v_2})$, according to $\mathbf{mv_1}$ and $\mathbf{mv_2}$.

We then convert the coordinates of $\mathbf{a_1}$, $\mathbf{a_2}$, $\mathbf{c_1}$ and $\mathbf{c_2}$ into 3D space. As the depths are unknown, we assume them to be r_{a1}, r_{a2}, r_{c1} and r_{c2} (shown in Fig. 3). Specifically, the 3D coordinates of these 4 points, denoted by $\mathbf{a_1^{3D}}$, $\mathbf{a_2^{3D}}$, $\mathbf{c_1^{3D}}$ and $\mathbf{c_2^{3D}}$, are:

$$\begin{aligned} \mathbf{a_1^{3D}} &= \begin{bmatrix} r_{a1}\cos\phi_{a1}\sin\theta_{a1} & r_{a1}\sin\phi_{a1}\sin\theta_{a1} & r_{a1}\cos\theta_{a1} \end{bmatrix}^{\mathbf{T}} \\ \mathbf{a_2^{3D}} &= \begin{bmatrix} r_{a2}\cos\phi_{a2}\sin\theta_{a2} & r_{a2}\sin\phi_{a2}\sin\theta_{a2} & r_{a2}\cos\theta_{a2} \end{bmatrix}^{\mathbf{T}} \\ \mathbf{c_1^{3D}} &= \begin{bmatrix} r_{c1}\cos\phi_{c1}\sin\theta_{c1} & r_{c1}\sin\phi_{c1}\sin\theta_{c1} & r_{c1}\cos\theta_{c1} \end{bmatrix}^{\mathbf{T}} \\ \mathbf{c_2^{3D}} &= \begin{bmatrix} r_{c2}\cos\phi_{c2}\sin\theta_{c2} & r_{c2}\sin\phi_{c2}\sin\theta_{c2} & r_{c2}\cos\theta_{c2} \end{bmatrix}^{\mathbf{T}} \end{aligned} \tag{3}$$

where ϕ_{ci} is the longitude of $\mathbf{c_i}$ and ϕ_{ai} is the longitude of $\mathbf{a_i}$ (i is 1 or 2). θ_{ci} and θ_{ai} are the latitudes. Using $\mathbf{a_1^{3D}}$, $\mathbf{a_2^{3D}}$, $\mathbf{c_1^{3D}}$ and $\mathbf{c_2^{3D}}$, we can calculate two 3D MVs in the 3D space, denoted by $\mathbf{mv_1^{3D}}$ and $\mathbf{mv_2^{3D}}$:

$$\mathbf{mv_1^{3D}} = \mathbf{a_1^{3D}} - \mathbf{c_1^{3D}}, \mathbf{mv_2^{3D}} = \mathbf{a_2^{3D}} - \mathbf{c_2^{3D}} \tag{4}$$

Then we set $\mathbf{mv_1^{3D}}$ and $\mathbf{mv_2^{3D}}$ to be equal according to our assumption. The equation is not able to solve because there are four unknown depths. We set $r_{a1} = 1$ so as to solve the other three depths. In other words, we are calculating relative depths since the absolute depths are not used in the motion compensations stage (discussed in Section 3.1). Thus, we need to solve:

$$\mathbf{\Phi r} = \mathbf{a} \tag{5}$$

where

$$\begin{aligned} \mathbf{\Phi} &= \begin{bmatrix} \cos\phi_{a2}\sin\theta_{a2} & \cos\phi_{c1}\sin\theta_{c1} & \cos\phi_{c2}\sin\theta_{c2} \\ \sin\phi_{a2}\sin\theta_{a2} & \sin\phi_{c1}\sin\theta_{c1} & \sin\phi_{c2}\sin\theta_{c2} \\ \cos\theta_{a2} & \cos\theta_{c1} & \cos\theta_{c2} \end{bmatrix} \\ \mathbf{r} &= \begin{bmatrix} r_{a2} & r_{c1} & r_{c2} \end{bmatrix}^{\mathbf{T}} \\ \mathbf{a} &= \begin{bmatrix} \cos\phi_{a1}\sin\theta_{a1} & \sin\phi_{a1}\sin\theta_{a1} & \cos\theta_{a1} \end{bmatrix}^{\mathbf{T}} \end{aligned} \tag{6}$$

Then

$$\mathbf{r} = \mathbf{\Phi^{-1}a} \tag{7}$$

gives out the results of r_{a2}, r_{c1} and r_{c2}. We set $r_{a2} = r_{c1} = r_{c2} = 1$ in case $\mathbf{\Phi}$ is not invertible.

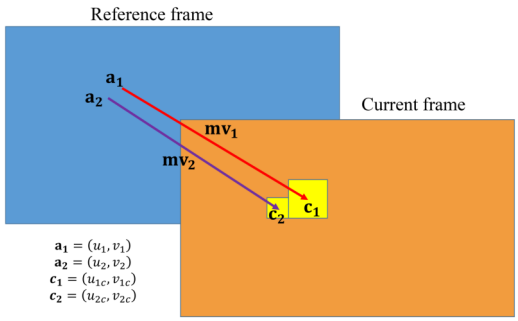

Fig. 2. Two adjacent frames in a panoramic video, showing two neighboring PUs and their block-level MVs.

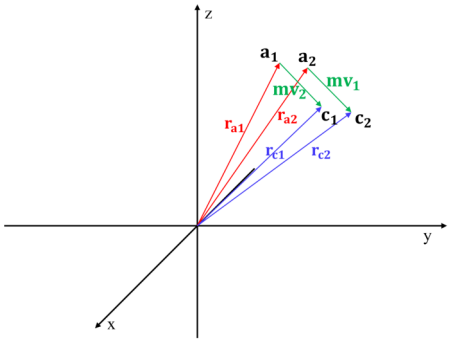

Fig. 3. The center points of the neighboring two PUs (shown in Fig. 2) displayed in 3D space.

3. MOTION ESTIMATION AND COMPENSATION WITH THE NEW MOTION MODEL

3.1. Motion compensation

Now we can perform motion compensation for the current PU. Firstly, we calculate $\mathbf{mv_1^{3D}}$ by (4), since the (relative) depths are available. For a pixel $\mathbf{p}$ in the current PU, we transform its coordinates (u, v) into spherical coordinates, denoted by $\mathbf{p^{3D}}$:

$$\mathbf{p^{3D}} = \begin{bmatrix} r_{c1}\cos\phi_p \sin\theta_p & r_{c1}\sin\phi_p \sin\theta_p & r_{c1}\cos\theta_p \end{bmatrix}^{\mathbf{T}} \tag{8}$$

Note that we use the depth r_{c1} due to our assumption. Then, we would get the original 3D coordinates of $\mathbf{p}$ in the reference frame by adding the MV:

$$\mathbf{p_{org}^{3D}} = \begin{bmatrix} x_{org} & y_{org} & z_{org} \end{bmatrix}^{\mathbf{T}} = \mathbf{p^{3D}} + \mathbf{mv_1^{3D}} \tag{9}$$

Then we inversely perform the spherical coordinates transform, using $r_{a1} = 1$, we can get the original $\mathbf{p}$ in the reference frame, denoted by $\mathbf{p_{org}}$:

$$\mathbf{p_{org}} = \begin{bmatrix} u_{org} & v_{org} \end{bmatrix} \tag{10}$$

where

$$u_{org} = \begin{cases} \arccos\left(\dfrac{x_{org}}{\sqrt{x_{org}^2 + y_{org}^2}}\right)\dfrac{W}{2\pi} & y_{org} > 0 \\ W - \arccos\left(\dfrac{x_{org}}{\sqrt{x_{org}^2 + y_{org}^2}}\right)\dfrac{W}{2\pi} & y_{org} \le 0 \end{cases}$$
$$v_{org} = \arccos\left(\frac{z_{org}}{r_{a1}}\right)\frac{H}{\pi} \tag{11}$$

From the above equations, it is clear that the absolute value of r_{a1} does not matter in the MC process as it is cancelled by division. Thus, the actual MV for the point $\mathbf{p}$ is $\mathbf{p_{org}} - \mathbf{p}$.

Note that u_{org} and v_{org} are floating-point numbers, and we quantize them to the accuracy of 1/64 pixel. We then adopt the DCT-based interpolation filters (DCTIF) with the accuracy of 1/64 pixel to calculate the pixel value at $\mathbf{p_{org}}$.

One special note is that the new model is used only for luma component. For chroma components, the traditional block-based model is used since the chroma components are quite smooth.

3.2. MV prediction and motion estimation

For MV predictive mode, MVP (motion vector prediction) candidates are generated the same as in HEVC, but we add our model into the ME process. That is, one PU would perform MV search twice, one with the traditional motion model, the other with our new motion model. To reduce the encoding time, we save the MVs and other motion information we need during the first round of ME, and reuse these MVs as the starting MVs for the second round of ME. When doing the second round, we narrow the search range down to 3×3 for integer pixel search. The motion vector accuracy is up to 1/4. If the starting $MV = (\Delta u, \Delta v)$ satisfies the following condition:

$$|\Delta u| + |\Delta v| \le 4 \tag{12}$$

we would skip integer pixel search and perform fractional pixel search only in the second round of ME. One special case is that, when starting $MV = (0, 0)$, the second round would be skipped.

When doing ME for the current PU, we first search its neighboring PUs, the same as that in HEVC (shown in Fig. 4). If the neighboring MV is available, we calculate the Euclidean distance between the current MV and the neighboring MVs, and select the one with the minimun distance. If the reference frame of the neighboring MV is not the same with the current PU, we will scale the neighboring MV before calculating distance. After getting the nearest neighboring MV, we finally save all the information we need for MC, including MVs and PUs.

Table 1. BD-rate results of our new motion model compared to HEVC anchor

Sequence	QP=27–42				QP=22–37			
	Y	U	V	Y SSIM	Y	U	V	Y SSIM
Hangpai_1	-7.7%	-1.5%	-7.5%	-7.6%	-6.0%	-2.9%	-5.5%	-7.1%
Hangpai_2	-8.0%	4.2%	-6.2%	-8.0%	-6.4%	-1.1%	-5.4%	-7.2%
Hangpai_3	-1.3%	0.3%	-0.1%	-1.4%	-1.0%	-0.7%	1.2%	-0.8%
Fengjing_1	0.2%	0.0%	0.1%	0.2%	0.0%	0.1%	0.0%	0.1%
Fengjing_3	-0.6%	1.2%	0.9%	-0.1%	-0.7%	0.3%	0.3%	-0.6%

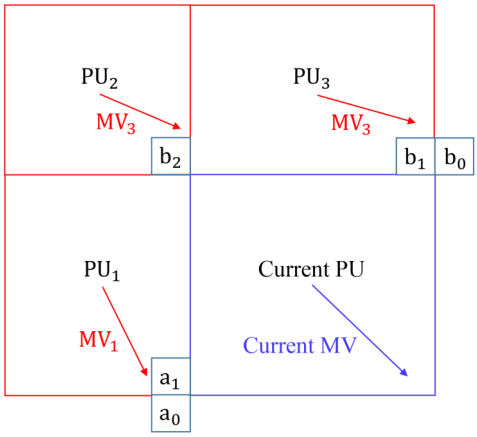

Fig. 4. The current PU and neighboring PUs.

The MC process during MV search is the same as have been discussed in Section 3.1. When MC is done, sum of absolute transformed difference (SATD) is calculated to decide the best MV for the current PU.

3.3. Merge mode

Like HEVC, we also define merge mode for our model, but we do not simply reuse the MV. Take Fig. 4 for example, if current PU uses a_1 as its best merge candidate, the current MV is equal to MV_1. However, in our motion model, we do not hope them to be equal, since then the nearest neighboring MV is always MV_1.

We design a different merge method for our model. When we have a merge candidate (for example MV_1 in Fig. 4), we do not set current MV equal to MV_1, but rather search all the other MV candidates to choose the MV, whose Euclidean distance from MV_1 is minimum (for example MV_3 in Fig. 4). Then we can use MV_1, MV_3, PU_1 and PU_3 to estimate depths as discussed in Section 2, after that, we set:

$$r_c = \frac{r_{c1} + r_{c2}}{2} \tag{13}$$

And when performing MC for Merge mode, the current MV is set to:

$$MV = \frac{MV_1 + MV_2}{2} \tag{14}$$

3.4. Syntax

The syntax of our model is very simple. We add a binary flag into the bitstream to indicate whether the new model is used or not. The flag is defined on CU level, though our model is working on PU level, according to empirical results. If the flag is true, all PUs inside the CU will use the new model. Otherwise, all of them will use the traditional model. There are two cases when we omit this flag. One is that all the MVs inside the CU are equal to zero, and the other is at the largest CU (LCU) level.

4. EXPERIMENTAL RESULTS

We implemented our algorithms on HEVC reference software, and tested using five panoramic videos, each of which has 96 frames at the resolution of 1024×512 (not shown here due to space limitation). The test condition is low-delay B (LDB). Table 1 shows the results at both high bit rates (QP = 22, 27, 32, 37) and low bit rates (QP = 27, 32, 37, 42). It can be observed that our new motion model achieves significant bits saving on some panoramic videos. Especially for Hangpai_1 and Hangpai_2 at low bit rates, the gain of luma component reaches 7.7% and 8.0%, respectively. For the other videos the improvement is limited. Note that Hangpai_1 and Hangpai_2 are captured by aerial photography with heavy camera motion but little object motion, which fits our model better. The other videos either have few motion (Fengjing_1 and Fengjing_3) or the motion is very erratic (lake water is moving in Hangpai_3). Note that the BD-rate measured by PSNR or SSIM (shown in Table 1 as Y and Y SSIM) is consistent for most cases. Also, the BD-rate at high bit rates is less than at low bit rates, possibly due to the more fine-granularity of block partition that helps the translational model to perform better.

5. CONCLUSION

In this paper, we propose a new motion model based on spherical coordinates transform for panoramic video coding. We achieve pixel-wise MVs in MC by transmitting a block-level MV. Experimental results show that our motion model provides significant bits saving. In the future, we will extend the model to the other kinds of projections for panoramic videos.

6. REFERENCES

[1] Joseph Isaac, "Step into a new world - virtual reality (vr)," https://www.completegate.com/2016070154/blog/virtual-reality-explained, 2016.

[2] John P Snyder, *Flattening the earth: Two thousand years of map projections*, University of Chicago Press, 1997.

[3] Thomas Wiegand, Gary J Sullivan, Gisle Bjontegaard, and Ajay Luthra, "Overview of the H.264/AVC video coding standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 7, pp. 560–576, 2003.

[4] Gary J Sullivan, Jens Ohm, Woo-Jin Han, and Thomas Wiegand, "Overview of the high efficiency video coding (HEVC) standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, no. 12, pp. 1649–1668, 2012.

[5] Li Li, Houqiang Li, Zhuoyi Lv, and Haitao Yang, "An affine motion compensation framework for high efficiency video coding," in *IEEE International Symposium on Circuits and Systems (ISCAS)*. IEEE, 2015, pp. 525–528.

[6] Zheng Jiali, Zhang Yongdong, Shen Yanfei, and Ni Guangnan, "Panoramic video coding using affine motion compensated prediction," *Multimedia Content Analysis and Mining*, pp. 112–121, 2007.

[7] Ashek Ahmmed, Miska M Hannuksela, and Moncef Gabbouj, "Fisheye video coding using elastic motion compensated reference frames," in *IEEE International Conference on Image Processing (ICIP)*. IEEE, 2016, pp. 2027–2031.

[8] Guoxin Jin, Ankur Saxena, and Madhukar Budagavi, "Motion estimation and compensation for fisheye warped video," in *IEEE International Conference on Image Processing (ICIP)*. IEEE, 2015, pp. 2751–2755.

[9] Andrea Eichenseer, Michel Bätz, and André Kaup, "Motion estimation for fisheye video sequences combining perspective projection with camera calibration information," in *IEEE International Conference on Image Processing (ICIP)*. IEEE, 2016, pp. 4493–4497.