DYNAMIC CLOUD OFFLOADING FOR VIEW SYNTHESIS

Qian Li, Xin Jin, Zhanqi Liu and Qionghai Dai
Shenzhen Key Lab of Broadband Network and Multimedia,
Graduate School at Shenzhen, Tsinghua University, Shenzhen 518055, China

ABSTRACT

In this paper, a dynamic offloading model is proposed to minimize the energy consumption of mobile devices by exploiting cloud computational resources for view synthesis. The computational complexity of view synthesis, the processing capability of the cloud, the processing capability and the power consumption of the mobile are considered jointly into the model to provide an optimized solution. Several simulations based on parameters of real mobile devices demonstrate that the proposed method can save an average of 42.59%(4-partition case) and 46.40%(8-partition case) of total energy on different mobile devices and an average of 67.58%(4-partition case) and 69.76%(8-partition case) of total energy under different transmitting rates than the existing algorithms for view synthesis, respectively.

Index Terms—View synthesis, dynamic resources allocation, energy minimization, FTV

1. INTRODUCTION

In recent years, the emergence of mobile devices has led to the wide spread of mobile applications. Free viewpoint TV (FTV) [1] is a potential one among them, which allows viewers to watch a 3D scene from multiple perspectives.

The Moving Picture Experts Group (MPEG) started FTV standardization project in April 2007 and has adopted a system architecture of performing depth estimation and view synthesis with the data format Multiview Video plus Depth (MVD) [2]. The MVD based architecture can reduce the data size to be transmitted by generating the virtual views using depth image-based rendering (DIBR) [3]. In order to obtain virtual views with good quality, DIBR introduces Warping, Blending, Hole Filling and Boundary Noise Removing [3] to map the reference viewpoints to the virtual viewpoints. However, the process presents high computational complexity, which restricts the application of FTV especially on portable devices, such as smart phones, PCs etc.

As the rapid developing of multimedia cloud computing, it can provide a variety of computing and storage services for mobile devices. Some researchers propose to integrate cloud technology with free view point technology [4], but the introduction of hardware will result in much higher cost in equipment and inconvenience to the users of portable devices. B. Zhou *et al.* put forward a context sensitive offloading scheme. It did not consider the energy consumption during the mobile idle time, which may lead to an error in offloading [5]. K. Kumar *et al*. conducted a survey of cloud offloading methods and analyzed their applicability on different portable devices [6]. However, none of those methods is dynamic offloading method aimed at energy minimization for view synthesis on mobile devices.

In this paper, a dynamic offloading model is proposed to minimize the energy consumption of mobile devices by exploiting cloud computational resources for view synthesis. Exploiting the complexity of each partition in the current frame, predicted by the number of non-hole pixels in the previous frames, partitions in each frame are allocated dynamically between the cloud and the mobile based on the proposed dynamic offloading model to minimize energy consumption on the mobile device. Experiments performed on real mobile devices and a variety of video contents showed that the proposed approach can reduce up to 96.13%(4-partition case) and 97.07%(8-partition case) of total energy for the mobile.

The rest of the paper is organized as follows. In Section 2, the proposed cloud offloading system for view synthesis is introduced. In Section 3, the energy consumption of mobile is analyzed and the dynamic offloading model is described in detail. The experimental results are provided in Section 4 with conclusions in Section 5.

2. THE PROPOSED SYSTEM

The architecture of the proposed dynamic cloud offloading system for minimizing the energy consumption of view synthesis on mobile devices is shown in Fig. 1. The system mainly consists of *Data Partitioning, Workload Prediction*, *Dynamic Offloading Model, View Synthesis* and *Data Integrating*.

In *Data Partitioning*, each frame of the reference views can be partitioned into several parts according to the requirements. Each frame can be partitioned horizontally, vertically, or partitioned into tiles like that defined in 3D-HEVC [2]. Texture and depth from the left and right reference views of every frame in the video take the same partition scheme. Each partition group, including texture and depth partitions from the left and right reference views, is assigned to cloud or to mobile for view synthesis.

Workload Prediction calculates the workload in each partition based on the non-hole pixels' distribution in the warped image. It predicts the future workload by retrieving the number of non-hole pixels in each line of current frame.

In *Dynamic Offloading Model*, the complexity of view synthesis algorithm, the size of the video, the processing capability of the cloud, the processing capability and the power consumption of the mobile are considered jointly into the model to provide an optimized solution, which minimizes energy consumption on the mobile device by allocating the partitions to be processed dynamically between the cloud and the mobile.

In *View Synthesis*, both 1-D and 2-D view synthesis [7] techniques based on DIBR can be applied, which is determined by the arrangement of cameras and the location of virtual views. A partition of the virtual view will be generated after *View Synthesis*, which will be sent to *Data Integrating* for merging.

Data Integrating gathers the synthesized partitions from both the cloud and the mobile and combines the synthesized partitions into a complete virtual view for output.

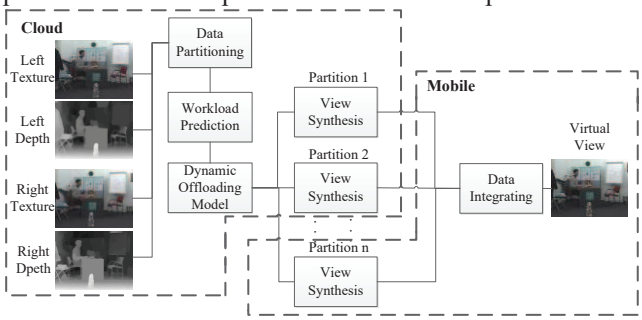


Fig. 1. The proposed parallel view synthesis system.

3. THE DYNAMIC OFFLOADING MODEL

In this section, we first analyse the energy consumption on the mobile device [8]. Then, a dynamic offloading model is proposed to minimize energy consumption on the mobile device by allocating the partitions dynamically between the cloud and the mobile.

3.1. Energy consumption analysis

The energy consumption of an application on a mobile device can be modeled by

$$E_{sum} = P_c \times \frac{C_{all}}{f} + P_{tr} \times \frac{D}{B}, \quad (1)$$

where P_c and P_{tr} denotes the power consumed during computing and data transmission, respectively; C_{all} is the total computations measured by the number of instructions; f is the processing speed of the mobile measured by the instructions per second; D is the size of data transmitted to or from the mobile; and B is the network bandwidth.

By exploiting cloud computational resources, the energy consumption of an application on the mobile device can be updated to be

$$E'_{sum} = P_c \times \frac{C_m}{f} + P_i \times \frac{C_c}{S} + P_{tr} \times \frac{D}{B}, \quad (2)$$

where P_i is the power consumption on the mobile when being idle; S is the processing speed of the cloud server measured by the instructions per second; C_c and C_m denote the total computations measured by the number of instructions on the cloud and on the mobile, respectively.

According to Eq. (2), the energy consumption of the application on the mobile depends on the complexity of the view synthesis algorithm, the computing capability of the mobile and the cloud, the bandwidth, and the size of transmitting data.

3.2. Offloading model

To minimize total energy consumption on the mobile, a dynamic offloading model is proposed to allocate the resources automatically, with considering the complexity of view synthesis [9], the processing capability of the cloud and the mobile, the size of transmitting data and the transmitting speed of the internet jointly. The model is defined as,

$$\vec{I} = \arg\min(P_c \times \frac{C_m}{f} + P_i \times \frac{C_c}{s} + P_{tr} \times \frac{D}{B}), \quad (3)$$

$$s.t.\ E_{sum} \le E'_{sum}$$

where $\vec{I}$ is a vector, $\vec{I} = (I_1, I_2, \cdots, I_n)$. If the value of $I_i (i = 1, 2, \cdots, n)$ is 0, the partition will be allocated to the mobile, otherwise it will be allocated to the cloud. n is the total number of partitions in one frame. The constraint means that the energy is saved by making use of the cloud computing resources. C_c and C_m denote the computations on the cloud and on the mobile respectively, which are calculated by

$$C_c = \vec{\psi} \times \vec{I}^T \times f_t, \quad (4)$$

$$C_m = \sum_{i=1}^{n} \psi_i \times f_t - C_c, \quad (5)$$

where f_t is the processing speed of the test machine measured by the instructions per second. $\vec{\psi}$ is a vector, $\vec{\psi} = (\psi_1, \psi_2, \cdots, \psi_n)$. $\psi_i (i = 1, 2, \cdots, n)$ is the workload of view synthesis for the i_{th} partition [9], ψ_i can be approximated by

$$\psi_i \approx \psi_{Fi} + \psi_{Mi}, \quad (6)$$

where ψ_{Fi} and ψ_{Mi} are workload of Forward Warping and Merging, respectively.

Forward Warping warps the reference partition to the virtual viewpoint pixel by pixel according to the 3D warping formula. So, its workload is proportional to the partition size, which is given by

$$\psi_{Fi} = 2\alpha_1 wh, \quad (7)$$

where multiplying by 2 represents warping from the left and right views simultaneously. w and h denote the width and

height of each partition, respectively. α_1 denotes the warping workload per pixel, which can be calculated by

$$\alpha_1 \approx \frac{\sum_N T_{Fi}}{N \cdot 2wh}, \tag{8}$$

where N is the total number of processed partition, T_{Fi} is the execution time of Warping during processing the i_{th} partition. Since α_1 is algorithm and platform dependent, α_1 is retrieved and updated dynamically with view synthesis for a stable result on a specific platform.

In Merging, the target viewpoint is generated by blending the two partitions warped from the left and right reference views pixel by pixel. Since it processes the pixel in the non-hole regions, the workload of Merging, ψ_{Mi}, is proportional to the total number of pixels in the non-hole regions. So, it is given by

$$\psi_{Mi} \approx \alpha_2(\lambda 2wh - \Theta), \tag{9}$$

where λ equals to 1 or 2 corresponding to mapping to integer pixel or half pixel precision, respectively. Θ denotes the sum of hole pixels in two warped depth maps, which is illustrated in [9]. α_2 denotes the workload of blending of a pixel in non-hole regions, which can be calculated by

$$\alpha_2 \approx \frac{\sum_N T_{Mi}}{\sum_N (\lambda 2wh - \Theta)}, \tag{10}$$

where T_{Mi} is the execution time of Merging during processing the i_{th} partition.

For every frame of an input video, the data transmitted from the cloud to the mobile can be calculated by

$$D = D_m + D_c, \tag{11}$$

for the partition view synthesised on the mobile, the data transmitted from the cloud to the mobile is D_m; for partition view synthesised on the cloud, the corresponding transmitting data is D_c. Therefore, D_m and D_c can be calculated by

$$D_m = \frac{WH(\gamma+1)\times 2}{n} \times (n - \sum_{i=1}^{n} I_i) \times 8, \tag{12}$$

$$D_c = \frac{WH(3\gamma+2)}{n} \times \sum_{i=1}^{n} I_i \times 8, \tag{13}$$

where W and H denote the width and height of a frame; γ ,1 and 8 are the parameter converting the size of the video into bits , γ equals to 1.5, 2 and 3 for videos using color space YUV 420, YUV 422 and YUV 444, respectively. Added 1 in Eq. (12) represents adding the data size of Y component of depth video; multiplying by 2 in Eq. (12) means the partitions from the left and right reference views. In Eq. (13), multiplying by 3 represents the virtual synthesized view partition together with texture view from the right and left view; added 2 means Y component of depth video from the right and left view.

For a given platform, whatever the input video is, $\vec{I}$ can be calculated by the proposed dynamic offloading model, which decides every partition whether being offloaded to the cloud or not.

4. EXPERIMENTAL RESULTS

In order to demonstrate the effectiveness of the proposed model, we design several experiments considering the impact of different videos, different mobile devices and different transmitting rates of networks.

4.1. The saved energy in different platforms

Table 1. Test sequences and viewpoints.

Seq.	Res.	No. of Frames	Input Views	Syn. View
Bookarrival [10]	1024×768	100	10, 8	9
Champagne_tower[11]	1280×960	100	39, 41	40
Newspaper [10]	1024×768	100	4, 6	5
PoznanStreet [12]	1920×1080	100	3, 5	4
Kendo[10]	1024×768	100	5,3	4
Lovebird1[10]	1024×768	100	6,8	7

In our experiment, six video sequences (the first 100 frames) with representative features, as listed in Table 1, are selected for testing. The input reference viewpoints and the virtual viewpoint needed to be synthesized are also listed. For simplicity, the name of each sequence is abbreviated by the first three characters in the following.

To prove that the proposed method is applicable to different platforms, we obtain realistic parameters of four mobiles. The related power parameters of different mobiles are shown in Table 2. The parameters of HP iPAQ PDA come from [8], others are calculated by referring to the specifications. The transmitting rate of network in this experiment is 60Mbps and the speedup of the cloud is 160. The performance of proposed model is evaluated under 4 partitions (n=4) and 8 partitions (n=8) cases. For simplicity, all frames are partitioned horizontally in experiment. Comparing with performing view synthesis directly on the mobile without offloading, Table 3 shows the percentage of saved energy on different mobile devices of different input sequences. In Several simulations based on parameters of real mobile devices, 94.95% (n=4) and 96.19% (n=8) of total energy can be saved at maximum. For different mobile devices, an average of 42.59% (n=4) and 46.40% (n=8) total energy are saved for different sequences. For the same sequence, the amount of saved energy is mainly related to the processing speed of the mobile devices. Since the processing speed of HP iPAQ

Table 2. The parameters of mobile device.

Device	f(MHz)	P_c (W)	P_i (W)	P_{tr} (W)
HP iPAQ PDA	400	0.900	0.300	1.300
Samsung S6	1536	0.183	0.031	0.750
IPhone 6S	1843	0.150	0.025	0.825
HTC 10	3072	0.400	0.023	0.981

Table 3. The percentage of saved energy in different platforms(%).

Device	Boo.		Cha.		New.		Poz.		Ken.		Lov.		Average	
	n=4	n=8	n=4	n=8	n=4	n=8	n=4	n=8	n=4	n=8	n=4	n=8	n=4	n=8
HP iPAQ PDA	86.64	87.41	90.77	91.89	83.65	85.87	94.95	96.19	84.48	85.76	86.71	88.22	87.87	89.22
Samsung S6	27.74	30.44	41.62	46.12	21.00	26.26	61.36	69.11	22.87	25.97	28.48	32.86	33.85	38.46
IPhone 6S	14.01	16.48	27.21	31.76	8.03	12.78	48.31	57.46	9.66	12.41	14.69	18.74	20.32	24.93
HTC 10	22.08	24.71	35.85	40.42	15.58	20.64	56.36	64.74	17.37	20.36	22.80	27.10	28.34	33.00
Average	37.62	39.76	48.86	52.55	32.07	36.39	65.25	71.88	33.60	36.13	38.17	41.73	42.59	46.40

Table 4. The percentage of saved energy in different networks(%).

Bandwidth (Mbps)	Boo.		Cha.		New.		Poz.		Ken.		Lov.		Average	
	n=4	n=8	n=4	n=8	n=4	n=8	n=4	n=8	n=4	n=8	n=4	n=8	n=4	n=8
2	0.19	2.52	10.06	14.07	0.00	1.3	30.05	39.95	0.00	0.76	0.15	3.28	6.74	10.31
20	66.56	68.59	76.04	78.66	61.02	65.41	86.29	89.56	62.63	65.18	67.13	70.33	69.95	72.96
40	80.74	82.06	86.72	88.29	77.00	79.98	92.67	94.47	78.11	79.82	81.11	83.18	82.76	84.63
60	86.64	87.41	90.77	91.89	83.65	85.87	94.95	96.19	84.48	85.76	86.71	88.22	87.87	89.22
80	89.51	90.28	92.91	93.77	87.53	89.07	96.13	97.07	87.96	88.97	89.73	90.92	90.63	91.68
Average	64.73	66.17	71.3	73.34	61.84	64.33	80.02	83.45	62.64	64.10	65.00	67.19	67.58	69.76

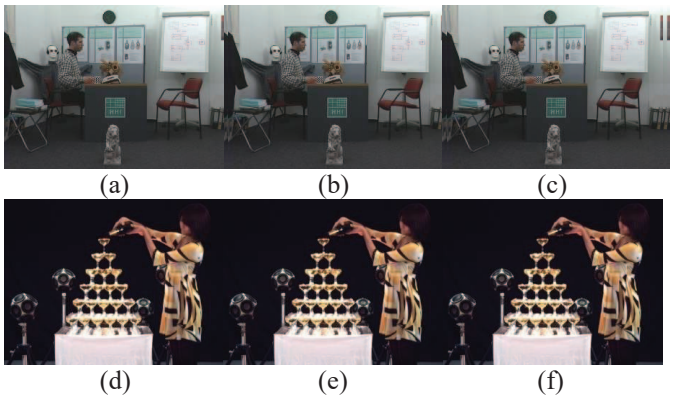
Table 5. Synthesis quality.

	Boo.		Cha.		New.		Poz.		Ken.		Lov.		Average	
	n=4	n=8	n=4	n=8	n=4	n=8	n=4	n=8	n=4	n=8	n=4	n=8	n=4	n=8
∆ PSNR(dB)	0.002	0.007	-0.003	-0.004	0.002	0.004	0.000	0.001	0.003	0.005	0.000	0.002	0.001	0.003

PDA is only 400MHz, which is much less than the other three devices, offloading partitions to cloud for view synthesis could save more power.

4.2. The saved energy in different networks

Different transmitting rates are used to investigate the impact of different networks. Considering that the network bandwidth between the cloud and mobile device is 2.8Mbps for 3G and 100Mbps for 4G, we tested the bandwidth from 2Mbps to 80 Mbps to cover more transmitting rates. The experiment is conducted on HP iPAQ PDA and the speed up of the cloud is 160. As shown in Table 4, for different bandwidths, an average of 67.58% (n=4) and 69.76% (n=8) total energy are saved for different sequences. For the bandwidth of 2Mbps, since transmitting data consumes much time and power, partitions are rarely allocated to the cloud and the percentage of the saved energy is low. If the bandwidth is high enough, most of the partitions will be allocated to the cloud, which contributes to extremely high percentage of saved energy. Moreover, the higher the transmitting rate of the network is, the more energy the mobile saved.



(a) (b) (c)

(d) (e) (f)

Fig. 2. A frame of synthesized view of Boo: (a) without partition; (b) 4 partitions (c) 8 partitions; A frame of synthesized view of Cha: (d) without partition; (e) 4 partitions (f) 8 partitions,

4.3. Objective quality assessment

The performance of view synthesis with cloud offloading, is compared with that without offloading. 4-partition (n=4) and 8-partition (n=8) cases are evaluated. As shown in the Table 5, only 0.001dB (n=4) and 0.003dB (n=8) degradation in the synthesis quality is introduced, which is negligible to real applications and presents no influence on the visual quality. The slight quality degradation is mainly caused by the reduction in the pixel correlations around the partition boundaries.

4.4. Subjective quality assessment

Fig. 2 compares the subjective quality for Bookarrival and Champagne_tower in three cases, namely no partition, 4-partition (n=4) and 8-partition (n=8) cases. As shown in Fig. 2, no visual quality difference can be detected in the synthesized views of all the three cases.

5. CONCLUSIONS

In this paper, a dynamic offloading model is proposed to minimize the energy consumption of mobile devices by exploiting cloud computational resources for view synthesis. Simulation results based on parameters of real mobile devices demonstrate that our method can save an average of 42.59% (n=4) and 46.40% (n=8) total energy on different mobile devices and an average of 69.58% (n=4) and 69.76% (n=8) total energy under different transmitting rates than the existing algorithms for view synthesis. Moreover, both the subjective and objective quality of the synthesized view are not degraded, which is very beneficial to fast view synthesis on mobile device.

6. ACKNOWLEDGMENT

This work was supported in part by the project of NSFC61371138 and NSF project of Guangdong 2014A030313733, China.

7. REFERENCES

[1] M. Tanimoto, M.P. Tehrani, T. Fujii, and T. Yendo, “Free-Viewpoint TV,” *Signal Processing Magazine*, *IEEE*, vol. 28, no. 3, pp. 67-76, 2011.

[2] G. Tech, K. Wegner, Y. Chen, and S. Yea, “3D-HEVC test model 5,” in *JCT-3V Doc. JTC3V-E1005, 5th Meeting*, 2013.

[3] C. Fehn, “A 3D-TV approach using depth-image-based rendering (DIBR),” *Proc. of VIIP*, vol. 3, pp. 84-88, 2003.

[4] Ya-Chun Li, I-Ju Liao, Hua-Pu Cheng and Wei-Tsong Lee, "A cloud computing framework of free view point real-time monitor system working on mobile devices," Intelligent Signal Processing and Communication Systems (ISPACS), 2010 International Symposium on, Chengdu, 2010, pp. 1-4.

[5] B. Zhou, A. V. Dastjerdi, R. N. Calheiros, S. N. Srirama and R. Buyya, "A Context Sensitive Offloading Scheme for Mobile Cloud Computing Service," 2015 IEEE 8th International Conference on Cloud Computing, New York City, NY, 2015, pp. 869-876.

[6] K. Kumar, J. Liu, Y. H. Lu, et al. "A Survey of Computation Offloading for Mobile Systems," Mobile Networks and Applications, vol. 18, no. 1, pp. 129-140, 2013

[7] View Synthesis Reference Software (VSRS 3.5) in Tech. Rep. ISO/IEC JTC1/SC29/WG11, 2010.

[8] K. Kumar and Y. H. Lu, "Cloud Computing for Mobile Users: Can Offloading Computation Save Energy?" in Computer, vol. 43, no. 4, pp. 51-56, April 2010.

[9] Z. Liu, X. Jin, C. Li and Q. Dai, "A workload balanced parallel view synthesis for FTV," 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), South Brisbane, QLD, 2015, pp. 1558-1562.

[10] http://sp.cs.tut.fi/mobile3dtv/stereo-video/.

[11] MPEG-FTV Test Sequence, available at: http://www.fujii.nuee.nagoya-u.ac.jp/~fukushima/mpegftv/.

[12] D. Rusanovskyy, K. Müller, and A. Vetro, “Common test conditions of 3DV core experiments,” in JCT-3V Doc. JCT3V-D1100, 4th meeting, 2013.