모바일 VR 을 위한 비대칭 코어에 최적화된 360 비디오 병렬처리

노현준, 류영일, 류은석
가천대학교
ggyo@gc.gachon.ac.kr, yiryu@gc.gachon.ac.kr, esryu@gachon.ac.kr

Optimizing 360 Video Parallel Processing for Asymmetric Core in Mobile VR

Hyun-Joon Roh, Yeongil Ryu, Eun-Seok Ryu
Gachon University

요 약

최근 초고화질 영상뿐만 아니라 360 비디오 콘텐츠의 보급이 확산되고 있다. 이미 대중적으로 보급된 스마트폰을 통해 누구나 쉽게 이 360 비디오 콘텐츠를 접할 수 있는데, 스마트폰의 성능은 제한적일 수 밖에 없다. 따라서 본 논문은 모바일 VR 에서 360 비디오 병렬처리를 수행할 때 보다 적합한 최적화 방법 2 가지를 소개한다. 이를 위해 전력 소모를 줄이는 장점으로 인해 모바일 디바이스에 많이 사용되는 비대칭 멀티코어의 특징을 활용한다. 두 방법 모두 공통적으로 각 코어의 성능비와 할당되는 작업량을 비례하게 하여 디코딩 작업의 시간을 줄이는 방법들이다. 첫 번째 방법은 영상을 타일로 분할할 때 각 코어의 성능비와 비례하게 분할하는 방법이다. 해당 기법을 적용하기 위해서, 비디오 크기별 연산 복잡도 분석 모델을 활용하여 사용한다. 제안하는 기법을 사용한 실험 결과, 평균적으로 약 25%의 디코딩 시간 개선을 보였다. 두 번째 방법은 타일로 분할된 영상의 각 복잡도 정도를 PU 의 양으로 추정하여, 각 코어의 성능비와 비례하게 코어에 할당하는 방법이다. 해당 기법을 사용하기 위해서, PU 의 양과 연산 복잡도 정도의 상관관계를 회귀분석하여 이를 이용한다. 제안하는 기법을 사용한 실험 결과, 약 9~16%의 디코딩 시간 개선을 보였다.

1. 서론

최근 비디오 시스템의 발달로 4K UHD(Ultra High Definition) 영상 콘텐츠 및 이를 제공하는 디바이스들의 보급도 활발하다. 이러한 4K UHD 영상은 지연 없이 실시간 영상재생을 위해 우수한 디바이스 성능이나 메서드가 요구된다. 이러한 실시간 처리를 위해서 JCT-VC(Joint Collaborative Team on Video Coding)에서는 2013 년에 HEVC(High Efficiency Video Coding) 표준을 제안할 당시 타일(Tile)등의 병렬처리 기법도 포함하였다. 또한 이제는 더 생생한 경험을 위해, 단순히 초고화질 영상뿐만 아니라 360 비디오 콘텐츠의 보급도 확산되고 있다. 360 비디오 콘텐츠는 일반적으로 VR(Virtual Reality)기기를 통해 인식하게 되는데, 스크린과 눈과의 거리가 매우 가까우므로 해상도가 낮으면 더 민감하게 반응한다. 따라서 360 비디오 역시 높은 화질의 영상을 사용해야 하고, VR 디바이스는 실시간 영상 처리를 위해 우수한 성능이 요구된다. 특히 모바일 VR 디바이스의 경우, 스마트폰의 성능에 전적으로 의존하기 때문에 연산 성능에 제한이 존재한다. 따라서 고성능 PC 등에 연결하여 사용하는 Tethered VR 들보다 더 수준 높은 최적화가 요구된다.

본 논문은 이 모바일 VR 디바이스를 위해 비대칭 코어의 구조를 이용한 최적화 방법들을 제안한다. 비대칭 코어란, 일부 코어는 성능이 더 좋고(이하 빅(Big)코어) 나머지 코어는 성능이 덜 좋은 코어들(이하 리틀(Little)코어)로 되어있는 것을 의미한다. 최근 출시되는 스마트폰은 거의 대부분 이 비대칭 멀티코어를 포함한 big.LITTLE 아키텍쳐로 구성되어 있다[1,2]. 스마트폰이 비대칭 코어를 사용하는 이유는 전력소모를 줄이기 위함이다. 단순한 작업의 경우 리틀코어만 사용하는 방식으로 불필요한 전력소모를 줄이는 것이다.

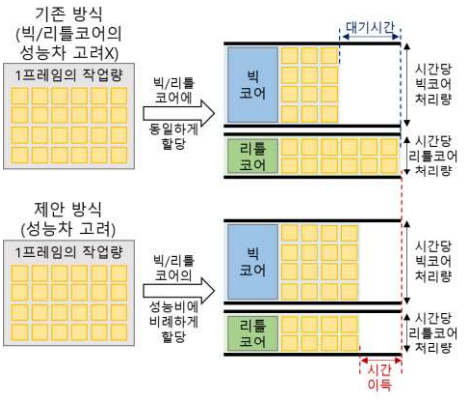

그림 1: 기존 비대칭코어 영상 병렬처리시 작업량 할당 방식과 제안하는 방식의 차이; 균등하게 작업량을 할당한 경우(위)와 최적화하여 불균등하게 작업량을 할당한 경우(아래)

위와 같은 이유로 코어마다 성능이 차이나는 빅/리틀코어에서 영상 병렬처리를 수행하는 경우, 동일한 작업량을 설정하게 되면 당연히 빅코어에서 작업이 먼저 끝나고 리틀코어는 더 많은 시간이 소요된다. 이를 방지하기

위해 빅/리틀코어의 성능비에 비례하게 작업을 할당하면, 거의 동시에 작업이 종료되고 전체적인 작업시간이 줄게 된다. 이를 그림으로 표현하면 그림 1 과 같다. 따라서 본 논문은 영상 병렬처리를 수행할 때 빅/리틀코어에 각각 적합한 작업량을 할당하는 기법 2 가지를 제안한다.

2. 관련연구

(1) 데이터의 양과 프레임의 타입을 복잡도 예측에 이용한 소비전력 감소 연구[3]: 해당 연구는 DVFS(Dynamic Voltage and Frequency Scaling)를 이용하여 전력감소를 유도하는 연구인데, 진행 과정 중에 영상의 복잡도 예측을 사용한다. DVFS 란 컴퓨터의 전압 변화로 클럭 속도를 조절하여, 상황에 따라 소비 전력을 낮추거나 성능을 향상시키는 기법이다. 이 연구에서는 DVFS 를 이용하여 예측되는 프레임의 연산 복잡도가 적을 때는 낮은 클럭 속도를, 연산 복잡도가 높은 경우 클럭 속도를 높게 설정하는 방식이다. 이 연구는 전력 효율성을 높이는 점에서 의미가 있지만 멀티 코어 시스템의 영상 병렬 처리 및 디코딩 시간 최적화는 고려하지 않는다.

(2) CTU 의 비트수를 기반으로 타일을 분할하여 병렬 디코딩을 수행하도록 하는 연구[4]: 해당 연구는 CTU(Coding Tree Unit)의 비트 수에 기반한 타일 분할 알고리즘을 제안한다. 타일 간의 디코딩 시간이 달라져서 대기시간이 발생하는 것을 최소화하기 위해, 각 타일의 총 비트 수를 균등하게 하는 방법을 사용한다. 이 연구는 본 논문의 제안과 많은 유사점을 가지고 있지만, 복잡성을 예측하기 위해 다른 방법을 사용하고, 비대칭 멀티 코어 환경을 고려하지 않는다.

3. 제안하는 비대칭 멀티코어를 위한 영상 병렬처리 기법들

해당 기법들을 적용하기 위해서는 우선적으로 비대칭 코어들의 성능비를 파악해야 한다. 각 코어 성능비를 정확하게 알아야 코어마다 할당할 복잡도 정도를 정확히 성능비와 비례하도록 할 수 있기 때문이다. 본 연구들에서는 빅/리틀코어가 동일한 디코딩 작업을 수행하게 하고, 디코딩 시간의 차이를 통해 성능비를 구한다.

3.1. 비대칭 멀티코어를 위한 타일분할시 각 코어들의 처리능력을 고려한 최적화 타일 분할 기법[5-7]

해당 제안하는 타일 분할 방식은 사전 연구[8]를 통하여 산출된 다중회귀분석 모델을 활용한다. 해당 회귀분석 모델은 영상의 해상도와 비디오 디코딩 연산 복잡도 사이의 상관관계를 나타낸다 이 회귀분석 모델을 이용하여 분할할 각 타일의 크기별 복잡도를 판단한다. 그 후, 빅/리틀코어의 성능비에 기반하여 각 코어에 할당할 디코딩 연산 복잡도 비를 산출하고, 이전에 언급한 회귀분석 모델을 활용하여 각 타일 크기를 확정하여 분할을 수행한다. 이후에는 적합한 크기의 타일들을 각 코어에 할당한다. 이와 같이 비대칭코어의 성능비를 고려하여 타일을 분할하면 그림 2 의 (b)와 같이, 타일이 불균등하게 분할된다. 그림 2 의 (b)의 경우 크기가 큰 2 개의 타일은 2 개의 빅코어에 각각 할당되며, 크기가 작은 4 개의 타일은 4 개의 리틀코어에 각각 할당되게 된다.

제안하는 방법을 사용하여 각 코어의 성능에 적합하도록 불균등하게 분할하게 되면, 모든 타일의 디코딩 타임이 거의 비슷하게 평준화되어 모든 타일의 디코딩이 유사한 시간에 종료되기 때문에, 균등하게 타일을 분할하였을 때보다 병렬처리 효율이 향상된다. 그림 3 은 위 모든 과정을 나타낸다.

(a)균등한 크기의 타일 분할

(b)각 코어의 처리능력을 고려한 타일 분할

그림 2: 서로 다른 방법으로 6 개의 타일로 분할된 PeopleOnStreet 의 모습

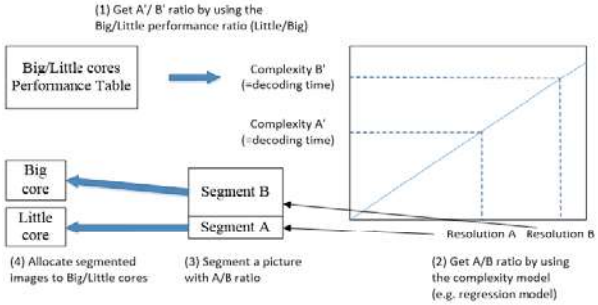

그림 3: 제안하는 비대칭 코어 처리능력 기반 타일 분할 방식

3.2. 비대칭 멀티코어를 위한 병렬 디코딩시 각 타일별 복잡도를 추정하여 성능비에 적합하게 각 코어에 할당하는 기법[9-11]

해당 제안하는 기법은 인코더에서 인코딩을 수행하며 측정되는 PU(Prediction Unit)수와 이를 기반으로 예측되는 복잡도 관련 회귀분석 결과를 전송해주면, 디코더는 이를 이용하여 최적화를 수행하는 방식이다. 그림 4 는 기존 방식과 제안 방식과의 차이를 보여준다. 기존 방식은 운영체제가 타일의 복잡 정도를 고려하지 않고 임의로 코어들에 할당하기 때문에 지연이 발생할 수 있지만, 제안하는 방식은 미리 복잡도 정도를 예측하여 코어들에 할당하기에 지연이 최소화 된다.

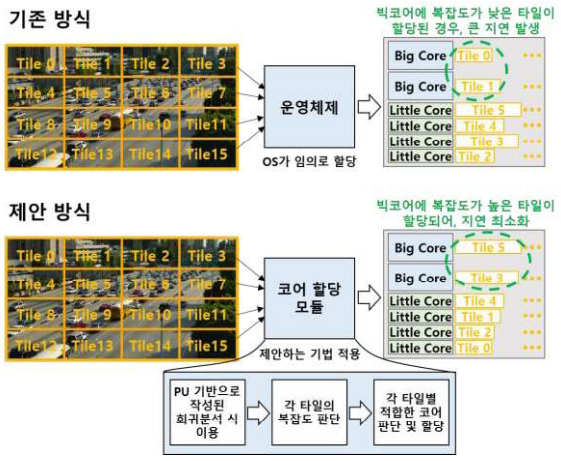

그림 4: 기존 타일 할당 방식(위)과 제안하는 코어 할당 시스템(아래)의 구조도

해당 기법 적용을 위해서는 각 타일의 복잡도 예측을 제대로 하는 것이 매우 중요하다. 비대칭 코어의 성능비와 복잡도 정도가 비례하도록 수행하는 것이 해당 기법의 핵심인데, 그 복잡도 정도가 잘못 예측된 것이면 성능개선을 기대하기 힘들기 때문이다. 본 기법에서는 HEVC 예측 유닛인 PU 의 분할된 정도(Partitioning)를 이용하는 방법을 사용한다. PU 는 인/디코딩 중 움직임 보정 작업에 사용되는데, ARM 프로세서의 경우 해당 작업이 디코딩 시간의 약 43%나 차지한다[31]. 따라서 영상의 복잡도가 높을수록 PU 의 양이 많을 것이라고 예상되어 PU 양과 디코딩 시간을 측정하여 상관관계를 구한 결과, 실제로 그림 5 와 같이 높은 상관관계를 보인다. 이런 실험결과를 토대로, 인코더 측에서는 PU 의 양과 디코딩 시간 데이터를 이용하여 회귀분석 식을 도출하여 디코더 측에 보내주게 되고, 이를 복잡도 예측에 사용한다.

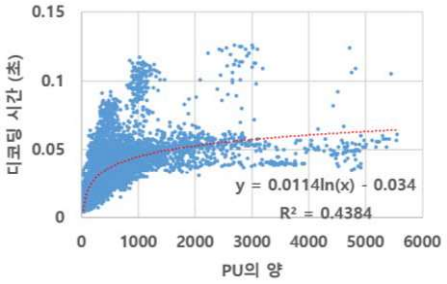

그림 5. PU 양과 디코딩 시간과의 상관관계 (시퀀스: Cactus, QP: 22, 타일의 수: 16 개)

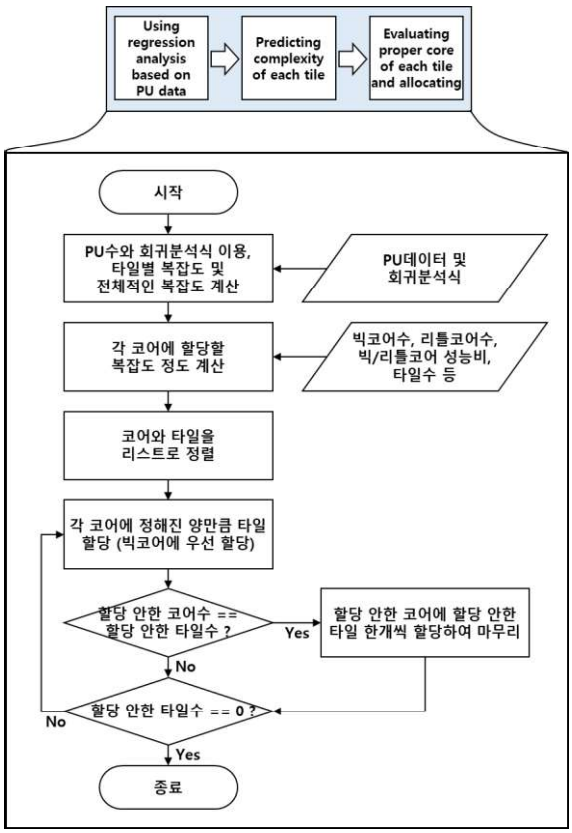

그림 6. 전체적인 각 타일별 최적 코어 판단 작업 절차

디코더에서 회귀분석 관련 데이터를 전송 받으면, 전송 받은 데이터를 토대로 최적 코어를(빅/리틀코어 할당 여부) 판단한다. 최적 코어 판단 작업에는 PU 관련 데이터, 타일의 개수, 빅/리틀코어의 개수, 빅/리틀코어의 성능비, 알파값 등이 사용된다. 여기서 알파값이란, 빅/리틀코어의 성능비를 도출하고 타일들을 이와 비례하게 할당하려고 하는데 특정 타일의 복잡도 정도가 애매하여, 이를 빅코어에 할당하면 더 좋은 결과를 얻을 수 있음에도 리틀코어에 할당하게 되어 성능이 떨어지는 경우를 막기 위해 빅코어에 우선 할당을 수행하도록 생성한 변수이다. 여러 테스트 시퀀스들을 사용하여 평균적으로 좋은 결과를 보인 0.2 를 초기 알파값으로 설정하였다.

전체적인 최적 코어 판단 작업은 먼저 현재 프레임의 전체 타일들을 복잡도 내림차순으로 리스트로 나열하고, 각 타일들의 복잡도 정도(PU 관련 데이터로 판단)를 종합한 후, 빅/리틀코어의 수와 각 코어들의 성능비를 반영하여 각 코어에 어느 정도 복잡도를 할당할지 설정한다. 이러한 판단 작업을 통해 각 코어에 어떤 타일을 할당할지 정해지게 되고, 전체적으로 작업량과 각 코어의 성능비가 최대한 비례하게 되어 최적화된 디코딩을 수행한다. 이를 그림으로 표현하면 그림 6 과 같다.

4. 실험환경 구축 및 실험결과

4.1. 비대칭 멀티코어를 위한 타일분할시 각 코어들의 처리능력을 고려한 최적화 타일 분할 기법의 실험결과

해당 실험은 갤럭시 S7 Edge 디바이스에서 수행하였고, 사용한 테스트 시퀀스와 코딩 옵션은 각각 표 1, 표 2 와 같다. 실험결과, 표 3 과 같이 기존방식 대비 평균 25%의 디코딩 시간 개선을 보인다.

표 1: 해당 실험에 사용한 테스트 시퀀스 정보

Test sequences	Resolution	Frame length	Frame rate
PeopleOnStreet	3840×2160	150	30
Traffic	3840×2048	300	30

표 2: 해당 실험에 사용한 코딩 옵션 정보

Coding option	Parameter
Coding structure	Random Access (RA) All Intra (AI) Low-Delay B(LDB)
QP	22, 27, 32, 37
Number of Tiles	6 (3×2)

표 3: 해당 제안하는 기법을 사용한 실험 결과

Test sequences	QP	Decoding time gain (%)		
		RA	LDB	AI
PeopleOnStreet	22	23.51	23.91	23.12
	27	26.42	25.69	23.37
	32	27.84	27.12	24.59
	37	27.73	28.72	24.86
Traffic	22	26.31	25.17	19.02
	27	22.96	27.73	22.01
	32	27.91	28.44	19.87
	37	23.42	29.65	15.54

4.2. 비대칭 멀티코어를 위한 병렬 디코딩시 각 타일별 복잡도를 추정하여 성능비에 적합하게 각 코어에 할당하는 기법의 실험결과

해당 실험은 갤럭시 S8 디바이스에서 수행하였고, 사용한 테스트 시퀀스와 실험환경은 각각 표 4, 표 5 와 같다. 실험결과, 표 6 과 같이 기존방식 대비 약 9~16%의 디코딩 시간 개선을 보인다.

표 4: 해당 실험에 사용한 테스트 시퀀스 정보

Test sequences	Resolution	Frame length	Frame rate
Traffic	3840×2048	300	30
Cactus	1920×1080	500	50

표 5: 해당 실험에 사용한 코딩 옵션 정보

Coding option	Parameter
Coding structure	Random Access (RA)
QP	22, 27, 32, 37
Number of Tiles	16 (4×4)

표 6: 해당 제안하는 기법을 사용한 실험 결과

Test sequences	QP	Decoding time gain (%)
Traffic	22	10.95
	27	9.44
	32	9.76
	37	9.27
Cactus	22	9.61
	27	9.31
	32	11.75
	37	16.63

5. 결론

본 논문은 모바일 VR 에서 360 비디오 병렬처리를 수행할 때 보다 적합한 최적화 방법 2 가지를 소개한다. 이를 위해 전력 소모를 줄이는 장점으로 인해 모바일 디바이스에 많이 사용되는 비대칭 멀티코어의 특징을 활용한다. 두 방법 모두 공통적으로 각 코어의 성능비와 할당되는 작업량을 비례하게 하여 디코딩 작업의 시간을 줄이는 방법들이다. 첫 번째 방법은 영상을 타일로 분할할 때 각 코어의 성능비와 비례하게 분할하는 방법이다. 해당 기법을 적용하기 위해서, 비디오 크기별 연산 복잡도 분석 모델을 활용하여 사용한다. 제안하는 기법을 사용한 실험 결과, 평균적으로 약 25%의 디코딩 시간 개선을 보였다. 두 번째 방법은 타일로 분할된 영상의 각 복잡도 정도를 PU 의 양으로 추정하여, 각 코어의 성능비와 비례하게 코어에 할당하는 방법이다. 해당 기법을 사용하기 위해서, PU 의 양과 연산 복잡도 정도의 상관관계를 회귀분석하여 이를 이용한다. 제안하는 기법을 사용한 실험 결과, 약 9~16%의 디코딩 시간 개선을 보였다.

Acknowledgement

이 논문은 2018 년도 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업임(No. NRF-2015R1C1A1A02037743)

References

[1] D. Koufaty, D. Reddy, S. Hahn, “Bias scheduling in heterogeneous multi-core architectures,” in Proc. the 5th ACM European conference on Computer systems, pp. 125-138. Mar. 2010

[2] M. Pricopi, T.S. Muthukaruppan, V. Venkataramani, T. Mitra, S. Vishin, “Power-performance modeling on asymmetric multi-cores,” in Proc. 2013 International Conference on Compilers, Architecture and Synthesis for Embedded Systems (CASES), pp. 1-10, Sep. 2013

[3] H. Ahn, S. Jeong, “Power-minimizing dvfs algorithm using estimation of video frame decoding complexity,” The Journal of Korean Institute of Communications and Information Sciences, vol.38, no.1, pp.46-53, Jan. 2013

[4] H. Baik, H. Song, “A complexity-based adaptive tile partitioning algorithm for hevc decoder parallelization,” in Proc. IEEE International Conference on Image Processing (ICIP) 2015, pp. 4298-4302, Sep. 2015

[5] Y. Ryu, H.J. Roh, E.S. Ryu, “Tile Partitioning-based HEVC Parallel Decoding Optimization for Asymmetric Multicore Processor,” Journal of The Korean Institute Of Information Scientists and Engineers (JOK), vol. 43. no. 9, pp. 1060-1065, Sep. 2016

[6] Y. Ryu, E.S. Ryu, “Video on Mobile CPU: UHD Video Parallel Decoding for Asymmetric Multicores,” in Proc. of the 8th ACM International Conference on Multimedia System (MMSys 2017), pp. 229-232, Jun. 2017

[7] Y. Ryu, E.S. Ryu, “Fast SW Decoding: HEVC Tile-based Parallel Decoding on Asymmetric Mobile Cores”, IEEE Visual Communications and Image Processing 2017 (VCIP2017), 2017. (Demo Session)

[8] H.J. Roh, Y. Ryu, E.S. Ryu, “Video Decoding Complexity Analysis Based on HEVC Resolution,” in Proc. of Korea Information Processing Society (KIPS) Fall Conference, Oct. 2015

[9] H.J. Roh, Y. Ryu, E.S. Ryu, “Novel HEVC Parallel Processing Method for Asymmetric Multicores Based on the Prediction Complexity,” in Proc. The Fifth International Multi-Conference on Engineering and Technology Innovation 2016 (IMETI2016), pp. 26, Oct. 2016

[10] H.J. Roh, Y. Ryu, S.D. Kim, S.S. Lee, E.S. Ryu, “Prediction Complexity-Based Tile Allocation for Asymmetric Multicores in HEVC Parallel Processing,” in Proc. of Korean Institute of Information Scientists and Engineers Winter Conference, pp.1063-1065, Dec. 2016

[11] H.J. Roh, S.W. Han, E.S. Ryu, “Prediction Complexity-Based HEVC Parallel Processing for Asymmetric Multicores,” Multimedia Tools and Applications (MTAP), vol. 76, no. 23, pp. 25271-25284, Dec. 2017