한국컴퓨터그래픽스학회

2024 학술대회

2024년 7월 9일 - 12일

소노벨 경주

KCGS 2024
Generative Imagination

H-ETC2: 모바일을 위한 CPU-GPU 하이브리드 텍스처 압축 기법

이현기 나재호
gusrl3204@gmail.com jaeho.nah@smu.ac.kr

상명대학교 컴퓨터과학과

논문 소개

- 본 발표에서는 Pacific Graphics 2023에서 기 발표된 연구 내용을 소개합니다.

- 논문: H. Lee and J.-H. Nah,
H-ETC2: Design of a CPU-GPU Hybrid ETC2 Encoder,
Computer Graphics Forum 42(7), *2023*

- 소스 코드: https://github.com/gusrlLee/HETC2

Pacific Graphics 2023
R. Chaine, Z. Deng, and M. H. Kim
(Guest Editors)

COMPUTER GRAPHICS *forum*
Volume 42 (2023), Number 7

H-ETC2: Design of a CPU-GPU Hybrid ETC2 Encoder

H. Lee[1] and J.-H. Nah[†1]

[1]Sangmyung University, South Korea

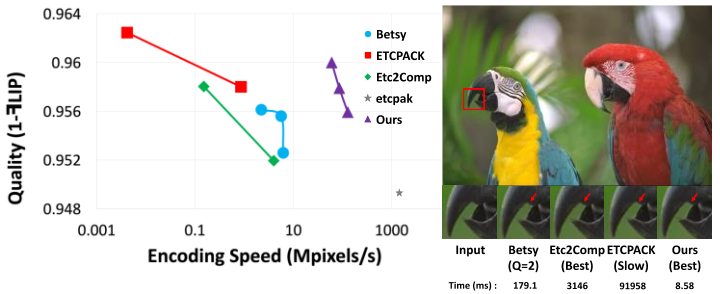


Figure 1: *Our H-ETC2 encoder is one to four orders of magnitude faster than previous high-quality ETC2 encoders, such as ETCPACK [Eri18], Etc2Comp [GB17], and Betsy [Gol22], while delivering a level of quality comparable to that of ETCPACK's slow mode. The right image is taken from kodim23, which is part of the Kodak Lossless True Color Image Suite. Please zoom in the image to distinguish the difference in quality.*

Abstract

This paper proposes a novel CPU-GPU hybrid encoding method based on the ETC2 format, commonly used on mobile platforms. Traditional texture compression techniques often face a trade-off between encoding speed and quality. For a better trade-off, our approach utilizes both the CPU and GPU. In a pipeline we designed, the CPU encoder identifies problematic pixel blocks during the encoding process, and the GPU encoder re-encodes them. Additionally, we carefully improve the base CPU and GPU encoders regarding encoding speed and quality. As a result, our encoder minimizes compression artifacts, increases encoding speed, or achieves both of these goals compared to previous high-quality offline ETC2 encoders.

CCS Concepts

• ***Computing methodologies*** → ***Image compression;***

1. Introduction

High-quality computer graphics are increasingly important in various fields, such as games, movies, and virtual/augmented reality. Texture mapping plays a crucial role in achieving this quality. Advancements in hardware and algorithms have made it possible to employ complex rendering algorithms with high-resolution textures in real-time applications. However, the utilization of multiple high-resolution textures necessitates greater memory capacity and bandwidth, which can result in decreased performance or increased power consumption. As a solution of the problems, textures are typically stored in memory as a compressed format [PP14].

† Corresponding author

KCGS 2024
Generative Imagination

한국컴퓨터그래픽스학회

2024 학술대회

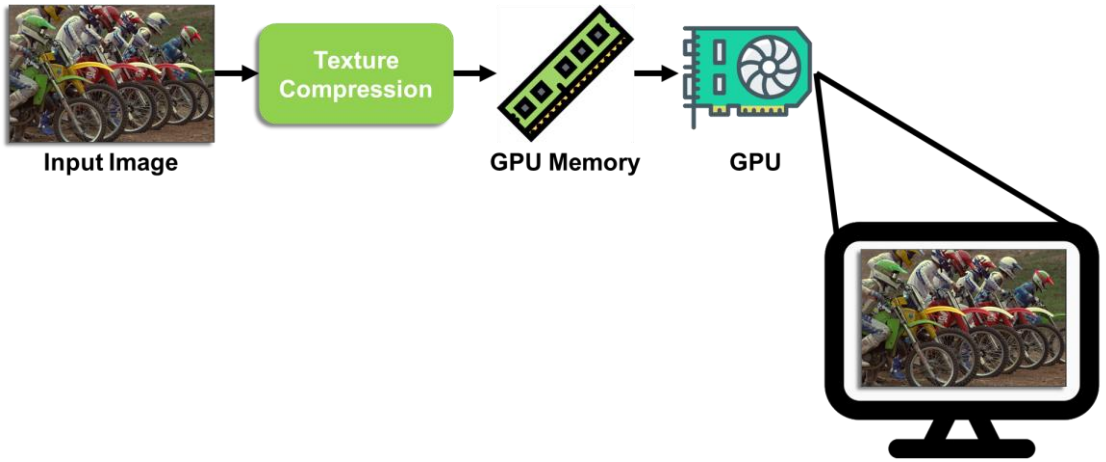
2024년 7월 9일 - 12일

소노벨 경주

서론 및 배경

텍스처 압축

- **최근 실시간 렌더링에서도 고품질 효과를 내기 위해, 많은 양의 고해상도 텍스처들이 사용되고 있음**



- **이러한 텍스처들은 메모리 사용량, 메모리 액세스량, 스토리지 사용량을 줄이기 위해 필수적으로 압축이 됨**

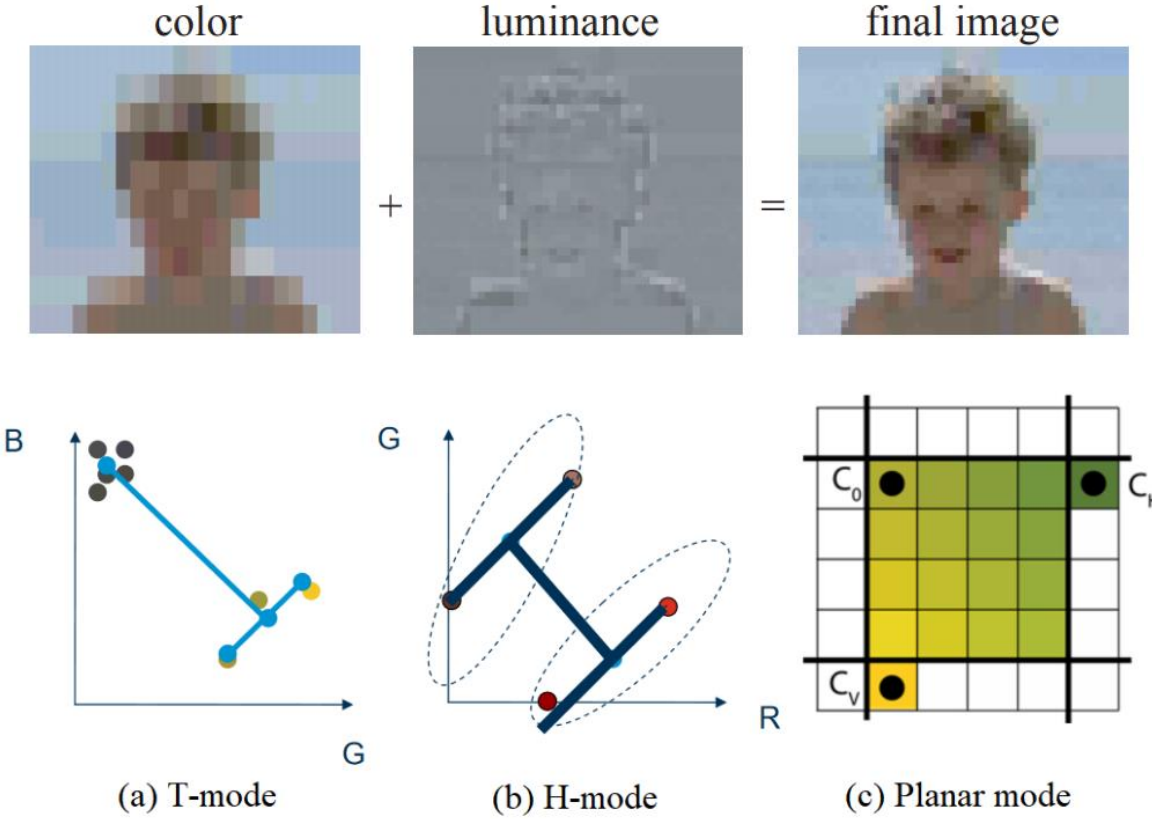
- **표준 텍스처 코덱**
  - 데스크탑: BC1~BC7 (MS)
  - 안드로이드: ETC1/2 (에릭슨), ASTC (ARM)
  - iOS: PVRTC (ImgTec), ASTC (ARM)
  - 공통적으로 실시간 decoding 및 random access를 지원하는 손실 압축 기법 사용

- **ETC2**
  - OpenGL ES 3.0 표준
  - 120억대 이상의 모바일 기기에서 지원

ETC1/2

- **iPACKMAN / ETC1** [Ström and Akenine-Möller, GH 2005]
  - 4x4 block 크기: 4x2 or 2x4 sub-blocks
  - 2개의 base color + per-pixel luminance
  - RGB 기준 6:1 압축률

- **ETC2** [Ström and Petersson, GH 2007] **& EAC**
  - 3개 추가 모드: T, H & planar
  - RGB에서 압축률 변화 없이
    Blocky & banding artifacts 감소
  - Alpha 채널 지원 (EAC) → 4:1 압축률



텍스처 인코딩

- **디코딩은 GPU 내에서 실시간 처리가 가능하지만, 인코딩은 보통 S/W로 오프라인 상에서 처리**

- **인코딩은 최적화 문제!**
  - 인코딩을 빠르게 끝내버리면 → 압축 품질 저하
  - 최고 품질에 도달하도록 검색 범위를 대폭 늘리면 → 인코딩 시간이 기하급수적으로 증가

- **텍스처 용량이 적다면 인코딩 시간이 큰 문제가 되진 않지만, 앱당 텍스처 용량은 계속해서 증가 중 (AAA 게임의 경우 최고 수십~수백 GB)**

Fallout 4's Ridiculously Huge, 58 GB HD Texture Pack Has Arrived

Paul Tassi Senior Contributor
News and opinion about video games, television, movies and the internet.

Follow

Feb 7, 2017, 10:58am EST

This article is more than 6 years old.

(Photo: Bethesda Softworks)

텍스처 인코딩

HELP! Texture Compression | ETC2

Help & Support

UFOMAMMOOT 2 Nov '18

Dear Community,

we got some problems with texture compression. After checking all texture compression checkboxes, the compression process starts and keeps loading for almost 2 days and nights. I mean we have quite a lot of textures, but they are not that big, so almost 1k and 2k. On some textures the Vram is already calculated but the loadingbar is still working and constantly processing. Lets say it's loading forever and not finishing properly.

Are the Textures loading all together or one by one?
If Vram is calculated the process should be done, so did it crashed?

We did some projects last year and texture compression works perfectly. So what happend?

Nov 2018

1 / 7

Dec 2018

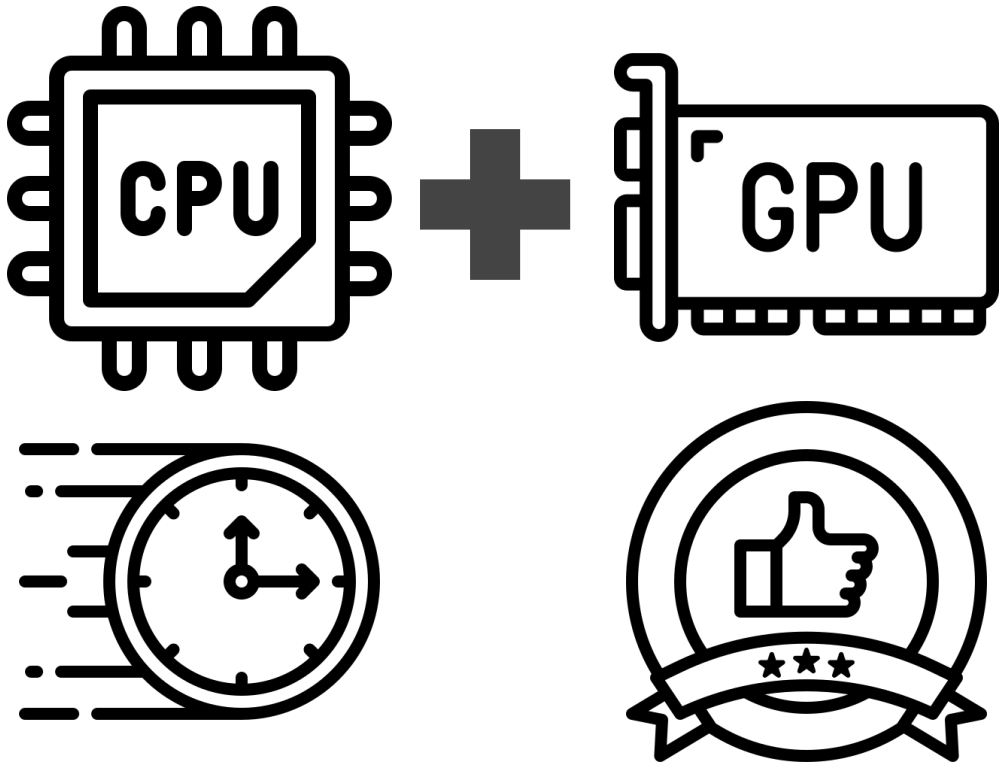
HELP! Texture Compression | ETC2 - Help & Support - PlayCanvas Discussion

H-ETC2 개요

- **연구 목표**
  - 인코딩은 최대한 빨리 끝내면서도
  - 압축 품질은 최종 S/W에 탑재 가능한 수준으로 끌어올릴 수 없을까?

- **속도와 품질이라는 서로 상충되는 목표를 달성하기 위해, CPU와 GPU를 함께 사용하는 새로운 인코딩 방법 제안**

- **H-ETC2 = Hybrid ETC2 encoder on CPUs and GPUs**
  - CPU 인코더 – 고속 압축 담당
  - GPU 인코더 – 고품질 압축 담당



관련 연구

QuickETC2: Fast ETC2 Texture Compression using Luma Differences

JAE-HO NAH, LG Electronics, South Korea

	Original	etcpak ETC1	etcpak ETC2 (p)	**Ours** ETC2 (p)	**Ours** ETC2	Etc2comp ETC2	ETCPACK ETC2	astcenc ASTC 6x6
Kodim05 (768×512)		0.14 ms	0.20 ms	0.14 ms	0.27 ms	105 ms	380 ms	43 ms
Kodim20 (768×512)		0.12 ms	0.20 ms	0.10 ms	0.14 ms	100 ms	392 ms	43 ms
Small-Char (512×512)		0.08 ms	0.14 ms	0.07 ms	0.11 ms	63 ms	223 ms	26 ms
ISCV2_u2_v4 (8192×8192)		10.8 ms	34.9 ms	12.3 ms	12.3 ms	14.1 s	24.3 s	2.7 s

Fig. 1. **Quality and performance comparison of our approach and other compressors (with their fastest settings).** Performance of *QuickETC2* in the partial ETC2 mode (planar only - ETC2 (p)) is comparable to that of *etcpak* [Taudul and Jungmann 2020] in the ETC1 mode, but its quality is similar to that of *etcpak* in the ETC2 (p) mode (see no banding artifacts on *ISCV2_u2_v4*). Ours in the full ETC2 mode provides much better edge handling and less color distortion than *etcpak* in the ETC2 (p) mode. Compared to *Etc2comp* [Google Inc. and Blue Shift Inc. 2017] and *ETCPACK* [Arm Limited 2016; Ericsson 2018], ours is two to three orders of magnitude faster. Compared to *astcenc* [Arm Limited. 2020] in the ASTC 6x6 mode, ours is two orders of magnitude faster and shows better color preservation (*Kodim05*) and less ringing artifacts (*Kodim20*). We obtained the timings on a desktop with an AMD Ryzen 7 3700X@3.6GHz 8-core (with hyper-threading) CPU. ©Kodak, UNC GAMMA Lab, and *fhernand*.

Compressed textures are indispensable in most 3D graphics applications to reduce memory traffic and increase performance. For higher-quality graphics, the number and size of textures in an application have continuously increased. Additionally, the ETC2 texture format, which is mandatory in OpenGL ES 3.0, OpenGL 4.3, and Android 4.3 (and later versions), requires more complex texture compression than the traditional ETC1 format. As a result, texture compression becomes more and more time-consuming.

To accelerate ETC2 compression, we introduce two new compression techniques, named QuickETC2. The first technique is an early compression-mode decision scheme. Instead of testing all ETC1/2 modes to compress a texel block, we select proper modes for each block by exploiting the luma difference of the block to reduce unnecessary compression overhead. The second technique is a fast luma-based T- and H-mode compression method. When clustering each texel into two groups, we replace the 3D RGB space with the 1D luma space and quickly find the two groups that have the minimum luma differences. We also selectively perform the T- or H-mode and reduce its distance candidates, according to the luma differences of each group. We have implemented both techniques with AVX2 intrinsics to exploit SIMD parallelism. According to our experiments, QuickETC2 can compress more than 2000 1K×1K-sized images per second on an octa-core CPU.

Author's address: Jae-Ho Nah, LG Electronics, 19, Yangjae-daero 11-gil, Seocho-gu, Seoul, 06772, South Korea, nahjaeho@gmail.com.

0730-0301/2020/12-ART270 $15.00
https://doi.org/10.1145/3414685.3417787

CCS Concepts: • **Computing methodologies → Image compression.**

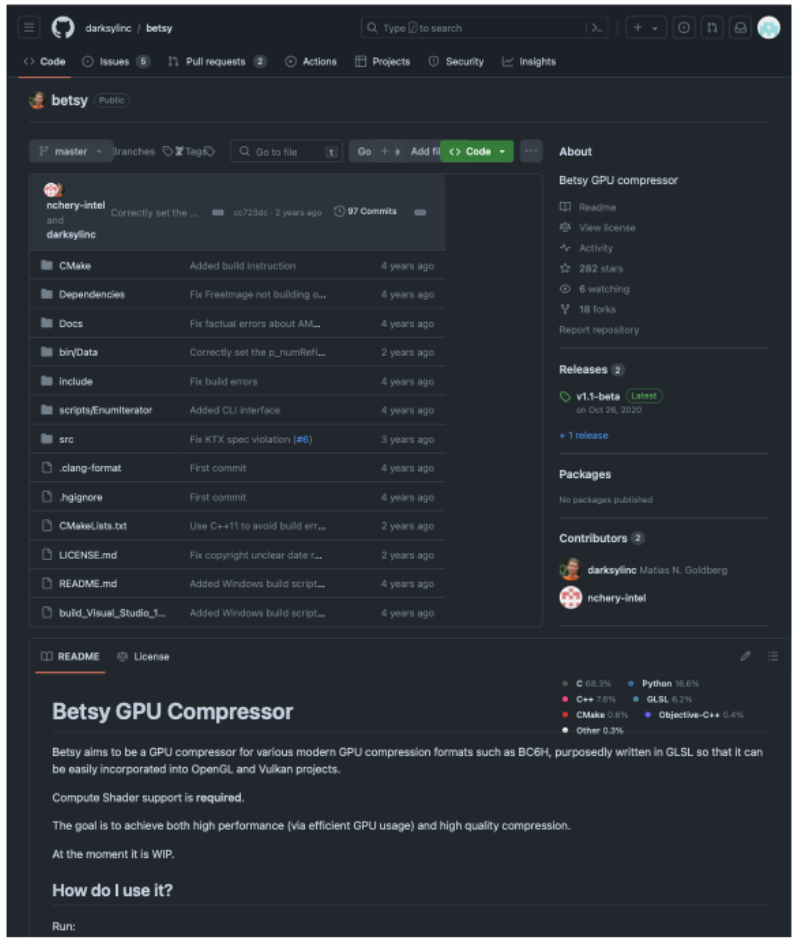
Additional Key Words and Phrases: texture compression, ETC2

ACM Trans. Graph., Vol. 39, No. 6, Article 270. Publication date: December 2020.

- **멀티쓰레딩과 SIMD를 이용하는 고속 CPU 인코더인 etcpak를 개선한 기법**
- **두 가지 최적화 방법 제안**
  - Early compression-mode decision
  - Luma-based T-/H-mode compression
- **논문 발표 이후 etcpak 정식 버전(v1.0)[Taudul, 2022]에 통합**
- **H-ETC2 CPU 인코딩 파트의 기반 코드**

QuickETC2 [Nah, TOG 2020]

관련 연구



- **OpenGL 기반의 오픈 소스 GPU 텍스처 인코더**
- **ETC2의 각 모드(T/H/P)의 인코딩 방법을 개선**
  - 고품질을 지향하면서 성능 향상
- **오픈 소스 게임 엔진인 고도(Godot)에 포함됨**
- **H-ETC2 GPU 인코딩 파트의 기반 코드**

Betsy [Goldberge, 2022]

KCGS 2024
Generative Imagination

한국컴퓨터그래픽스학회

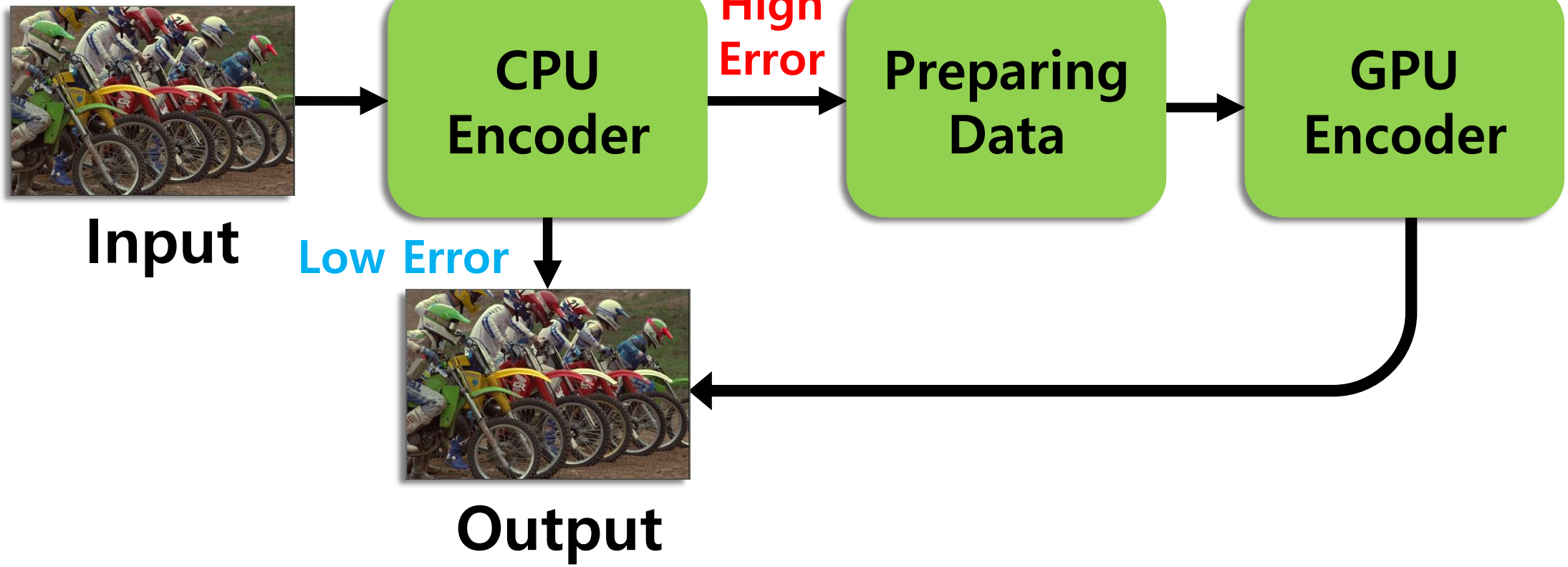
2024 학술대회

2024년 7월 9일 - 12일

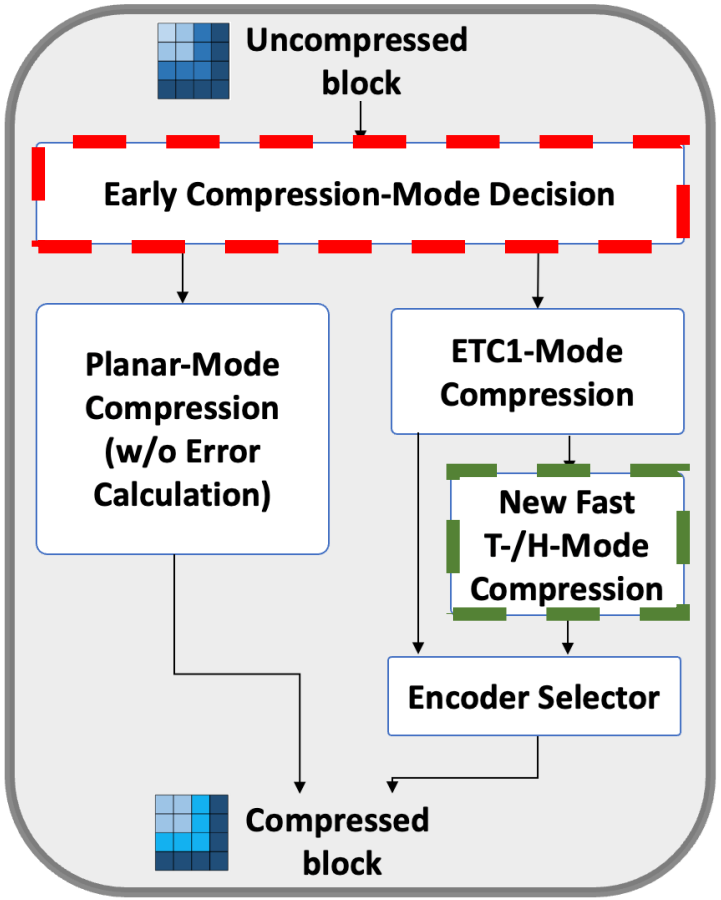
소노벨 경주

H-ETC2의 구현 내용

SYSTEM OVERVIEW

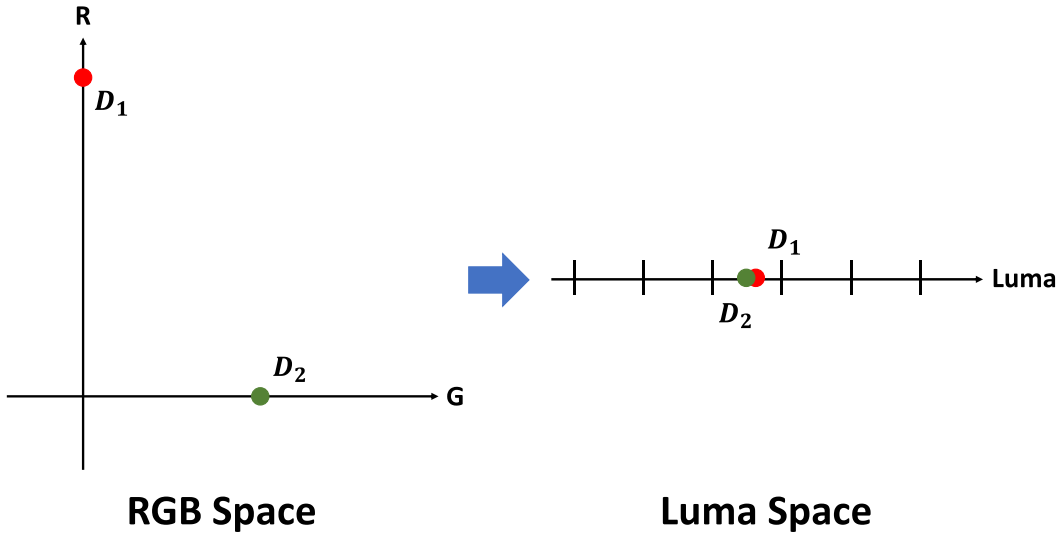


QUICKETC2 인코딩



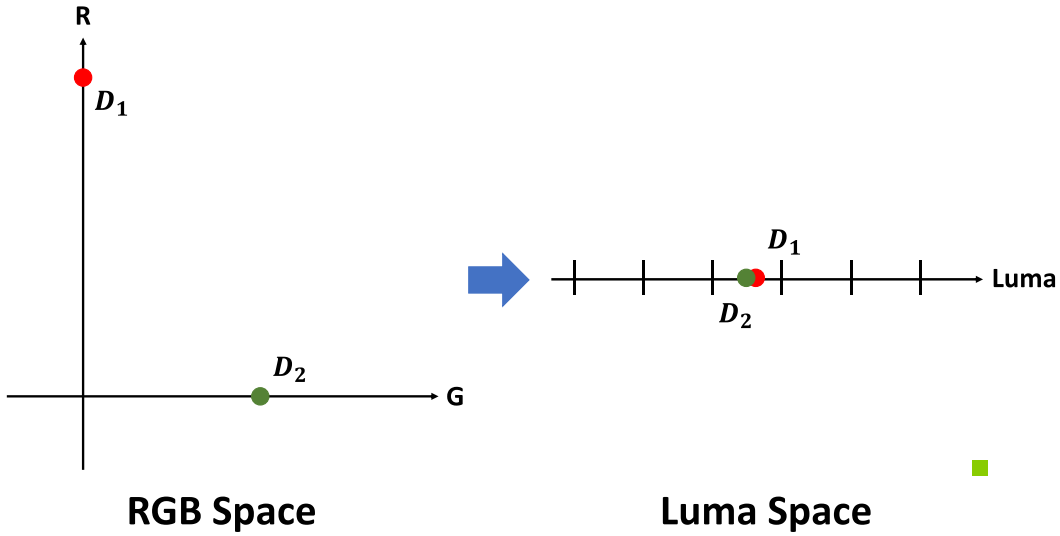
- **Early Compression-Mode Decision**
  - 블록 내 RGB data를 luma로 변경한 후,
    이 luma의 차이값에 따라 미리 압축 모드를 지정
    (ETC1, T-mode, H-mode, Planar mode)
- **New Fast T-/H-Mode Compression**
  - 변환된 luma 값을 이용하여 T/H 모드 압축 수행
    (3차원을 1차원으로 축소하여 고속 실행 가능)
- **ETC2 내 모든 모드를 지원하는 인코더 중 최고 속도 달성**

LUMA 공간의 문제점



- **한 가지 상황 가정**
  - RGB Channel 을 가진 D_1 = (255, 0, 0)
  - RGB Channel 을 가진 D_2 = (0, 128, 9)
- **RGB Space → Luma Space**
  - $luma = 0.3 \times R + 0.59 \times G + 0.11 \times B$
  - $D_{1(luma)} = 76.5, \ D_{2(luma)} = 76.509$
- **왼쪽의 그림과 같이 서로 다른 색상인데도, luma space 에서는 서로 비슷한 값을 가짐**
  - 압축 artifact 의 원인

LUMA 공간의 문제점



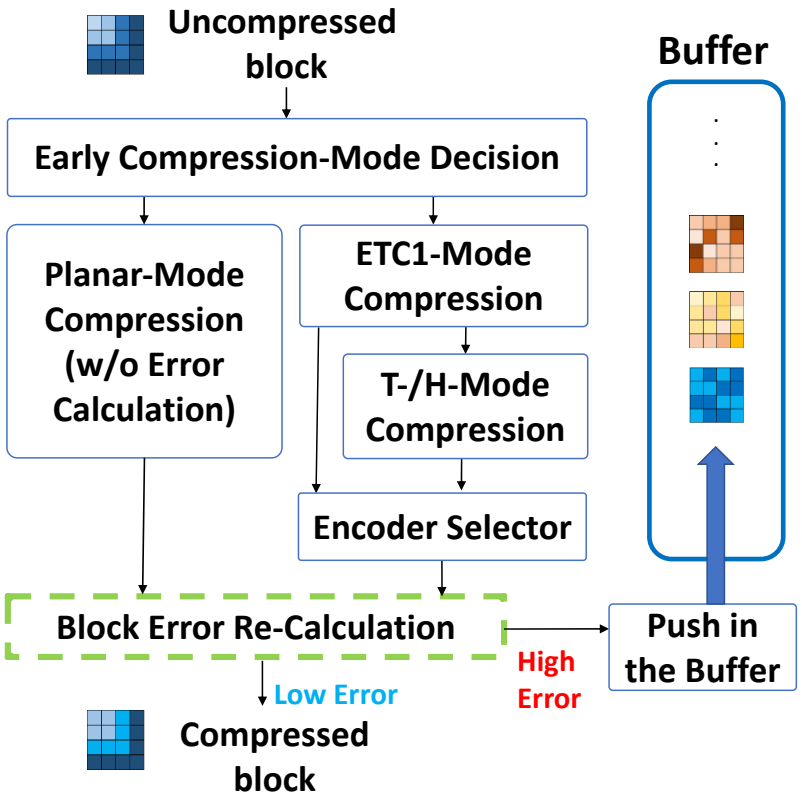
Re-calculation error metric

$$error = \sum_{i=0}^{N-1} \max\left(\left|\bar{x}_{i,r} - x_{i,r}\right|, \left|\bar{x}_{i,g} - x_{i,g}\right|, \left|\bar{x}_{i,b} - x_{i,b}\right|\right)^2$$

$\bar{x}$: compressed pixel
x : original pixel

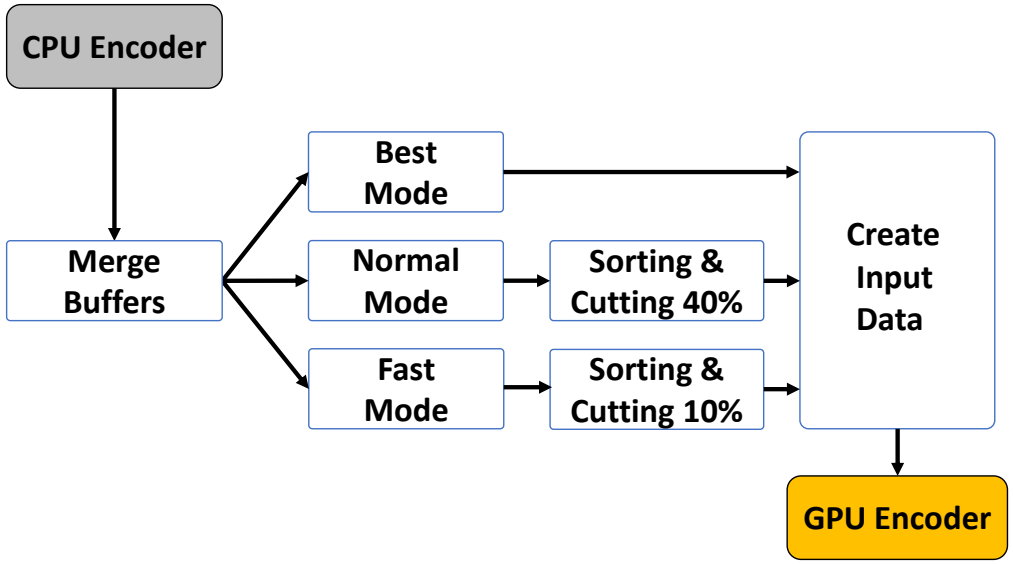
- **보수적으로 각 RGB channel의 오류를 확인하기 위해 위와 같은 식으로 error 를 계산**
- **계산된 error가 threshold T의 값을 넘어가면 문제가 있을 수 있는(problematic) pixel block으로 판단**
  - 여기서 사용된 T값은 ASTC encoder[Smith, 2018]의 medium quality 에 해당하는 *dblimit* 값 (PSNR 35.68 dB)을 기준으로 함

QUICKETC2의 수정



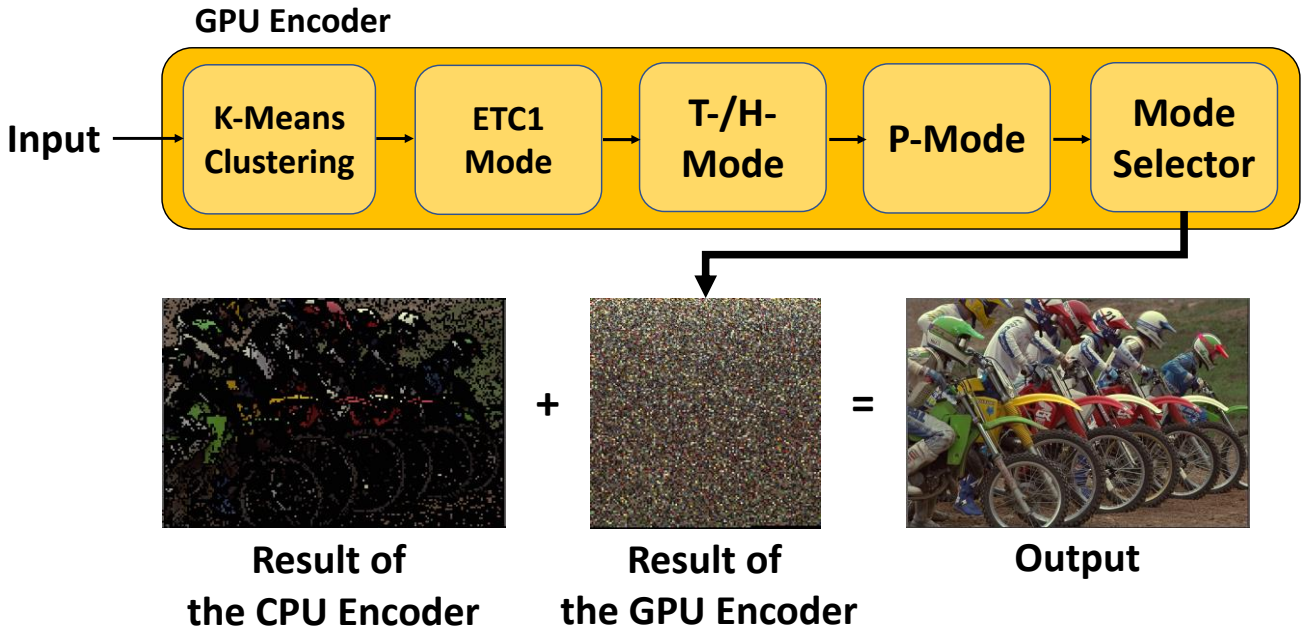
- **QuickETC2 코드에 Block Error Re-Calculation 과정 추가**
- **Result → high error?**
  - 해당 블록을 thread 의 local buffer에 저장
- **Result → low error?**
  - 인코딩 완료된 블록을 최종 output에 저장

GPU 입력 데이터 준비



- **인코딩이 끝나면,**
  각 thread local buffer의 결과를 하나로 merge
- **사용자가 원하는 모드를 선택하여**
  속도 및 품질 개선 정도를 조절
  - Etc2comp[Google Inc. and Blue Shift Inc., 2017]
    에서 아이디어를 얻음
- **Best mode**
  - Problematic pixel block을 전부 수정
- **Normal mode**
  - 에러를 기준으로 sorting 후
    problematic pixel block 전체의 40%만 수정
- **Fast mode**
  - 에러를 기준으로 sorting 후
    problematic pixel block 전체의 10%만 수정

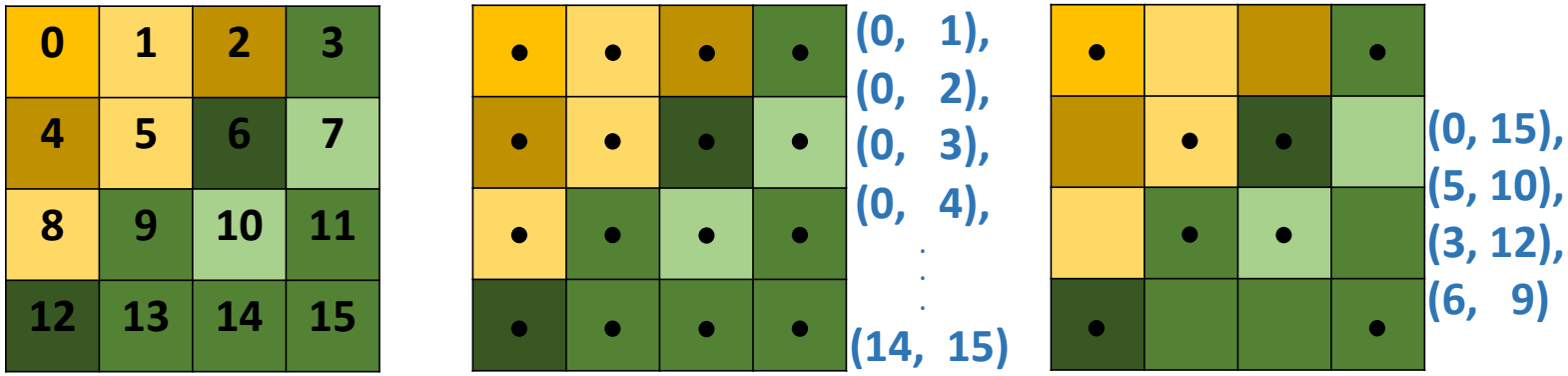
GPU 인코더 디자인



- **Betsy 를 기반으로 작성**
- **품질 향상을 위한 수정 사항**
  - 셰이더 내 quantization 버그를 수정 (max 285 → 255)
  - iPACKMAN을 참고하여 perceptual error metric을 적용 (에러 계산시 각 RGB 채널의 비중을 29:60:11으로 설정)
- **수정 결과 artifact를 상당부분 개선**

→ 하지만 여전히 느린 인코딩 속도!

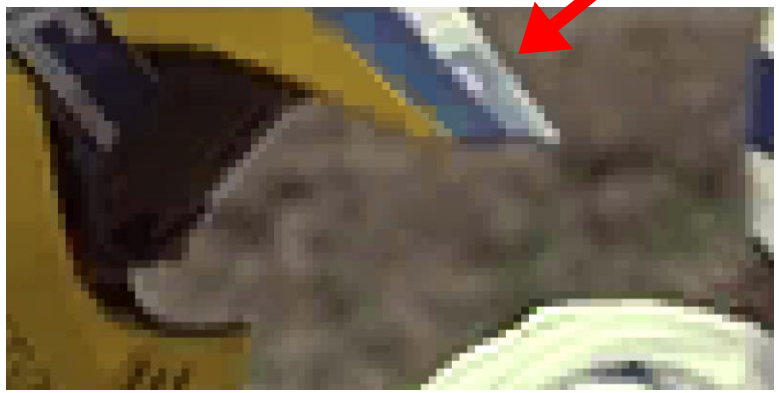
선택적 압축 방법



- **K-means clustering의 초기 픽셀 쌍 후보 개수의 감소 → T/H 모드의 연산량을 대폭 절감**
  - THUMB[Pettersson and Ström, SL 2005]의 selective compression method에서 영감
  - 4x4 블록 내 총 120개($=_{16}C_2$) 후보를 대각선 부분의 4개로 감소
  - T/H 모드가 ETC1 모드에 비해 대각선으로 나누어진 클러스터를 더 잘 다룬다는 점과, 개별 파티션 내의 픽셀들은 공간적 일관성을 나타낸다는 점에 착안
- **별다른 품질 저하 없이 GPU 인코딩 파트의 성능이 6배 향상**

개선 단계별 결과

kodim05



Original
Betsy

+ Fix
quantization
errors

+ Apply
the perceptual
error metric

개선 단계별 결과

kodim05

+ Selective compression method

+ CPU-GPU hybrid compression (Best mode)

Uncompressed

KCGS 2024
Generative Imagination

한국컴퓨터그래픽스학회

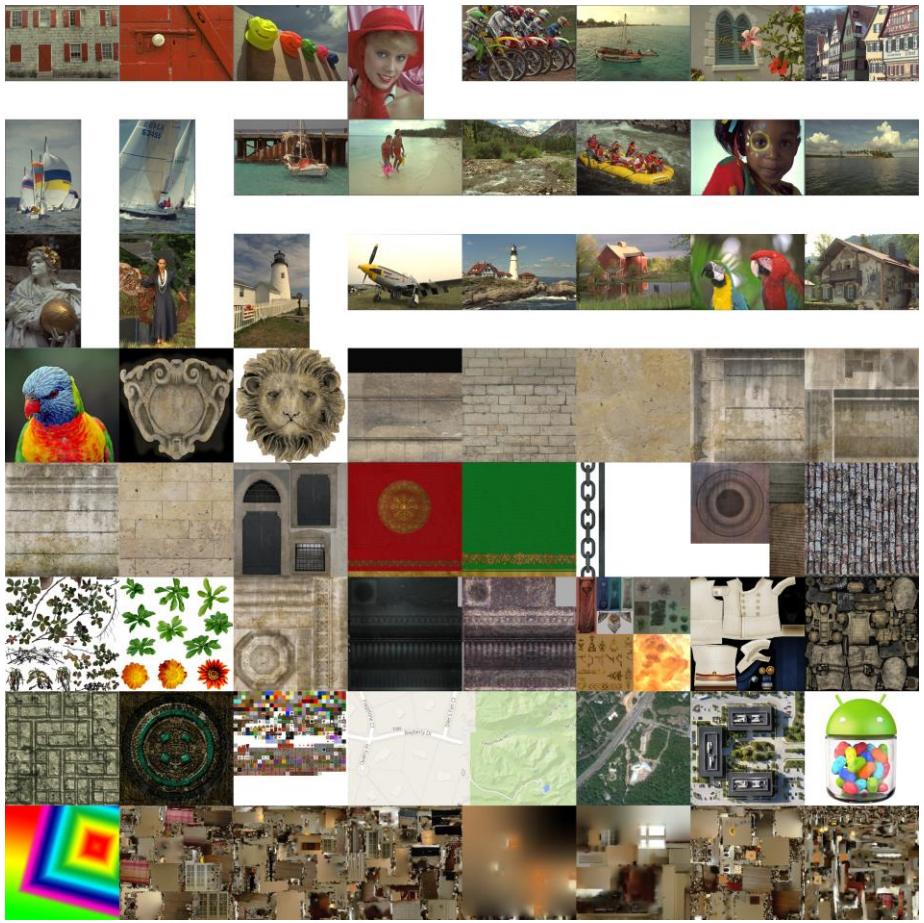
2024 학술대회

2024년 7월 9일 - 12일

소노벨 경주

실험 및 결과

테스트 이미지



- **QuickETC2 [Nah, TOG2020] 테스트 셋**
  - 55 RGB + 9 RGBA textures
  - Size : 256 X 256 ~ 8192 X 8192
- **Photos (No. 1-25)**
  - kodak Lossless True Color Image Suite & Lorikeet
- **Game textures (No. 26-51)**
  - Crytek Sponza, FasTC & Vokseli Spawn (Minecraft)
- **GIS maps (No. 52-55)**
  - Google Maps & Cesium
- **Synthesized images (No. 56-57)**
- **Captured images for 3D reconstruction (No. 58-64)**
  - Bedroom

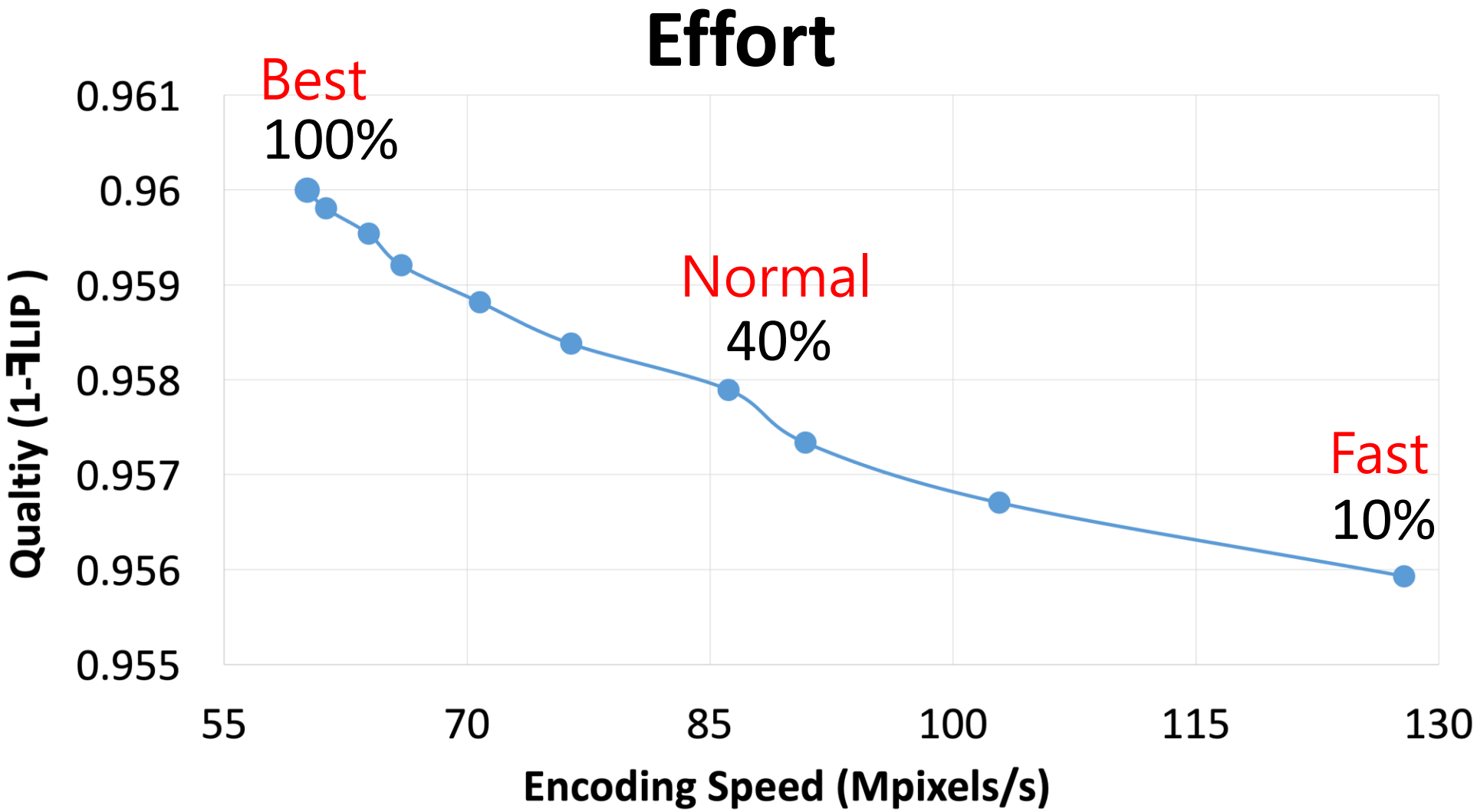
HW & SW 셋업

- **테스트 H/W**
  - Intel Core i5-12400 CPU, 32GB RAM, NVIDIA GeForce RTX 3060, 1TB SSD

- **평가 메트릭: ꓘLIP [Andersson et al., HPG 2020], Mpixels/s**
  - 낮은 ꓘLIP 값 → 높은 품질, 높은 Mpixels/s → 빠른 속도
    (1M pixels/s는 1024x1024 크기 텍스처를 1초에 인코딩 가능하는 의미 – 파일 I/O 및 디코딩 시간 제외)

- **사용된 인코더**
  - etcpak 1.0 (QuickETC2)
  - Betsy – quality parameter를 0, 1, 2로 설정
  - Etc2comp – fast & best 모드
  - ETCPACK – fast & slow 모드
  - H-ETC2 – fast, normal & best 모드

압축 모드 기준점 실험


Effort
Best
100%
Normal
40%
Fast
10%
0.961
0.96
0.959
0.958
0.957
0.956
0.955
Qualtiy (1-FLIP)
55
70
85
100
115
130
Encoding Speed (Mpixels/s)

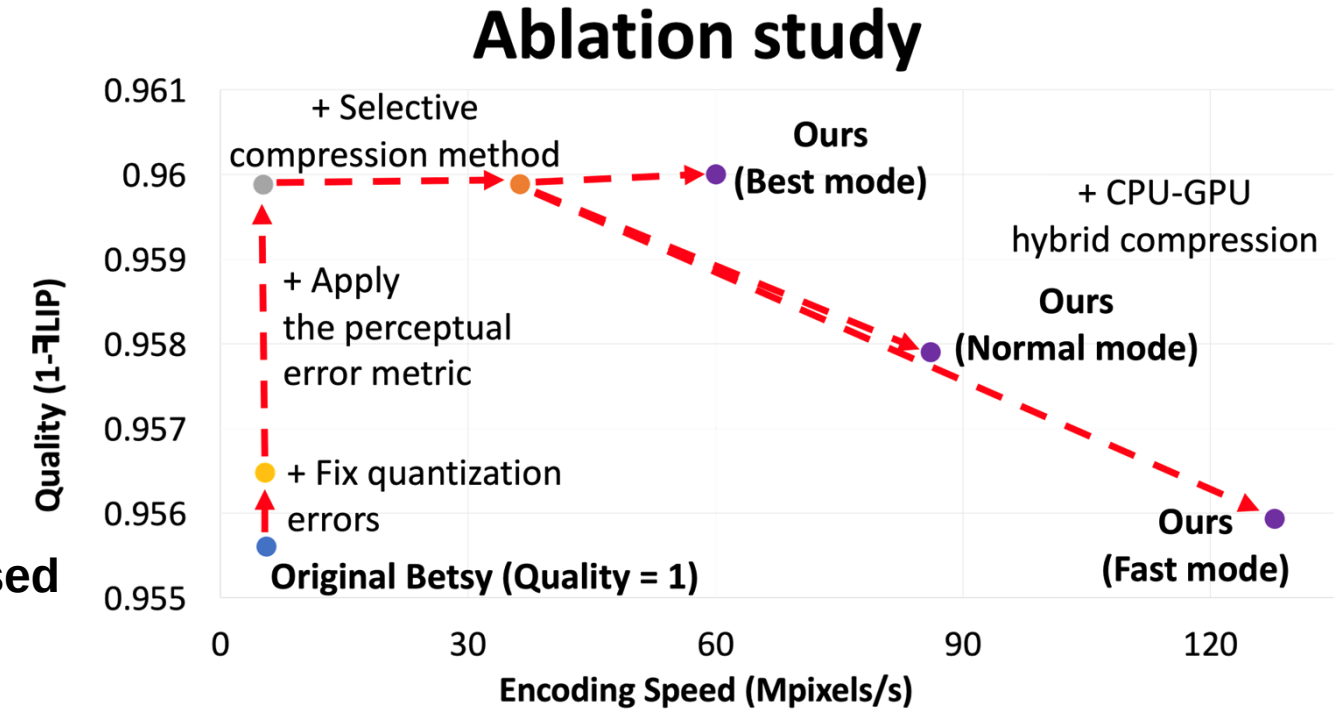
GPU 파트 절제 연구

kodim05

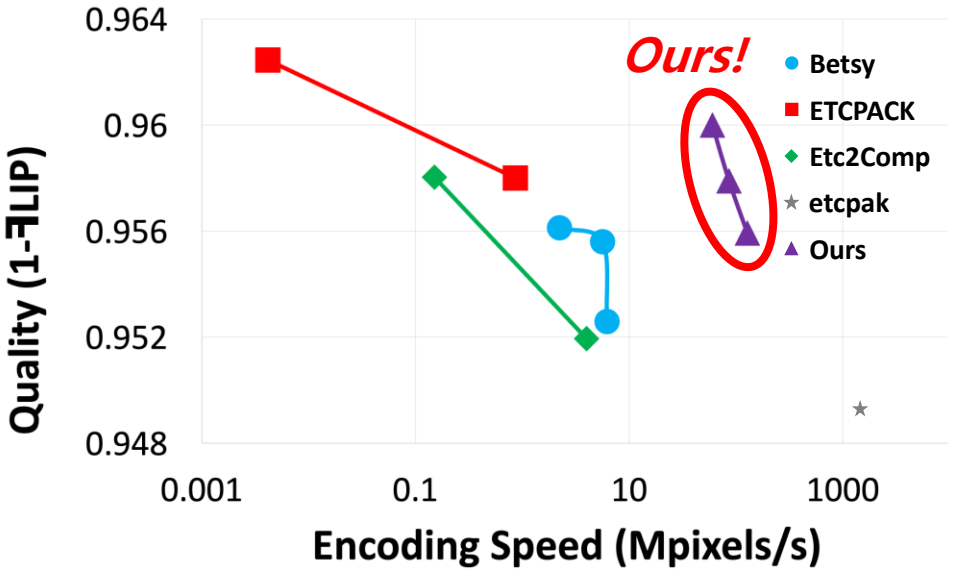
Original Betsy

Ours

Uncompressed



결과 (정량적 비교)

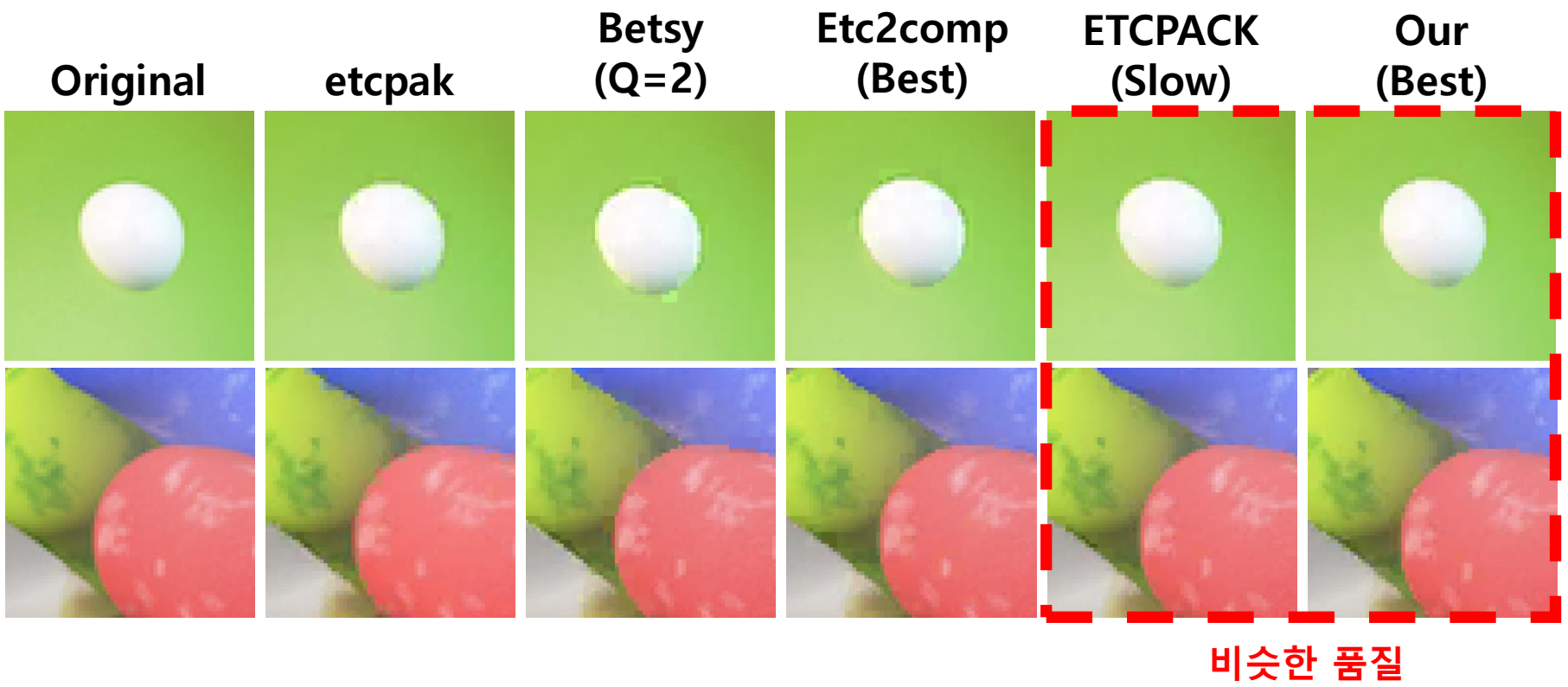


Compressor	Mode	ꟻLIP	Mpixels/s
etcpak		0.0506	1350.82
Betsy	Q=0	0.0474	6.20
	Q=1	0.0444	5.63
	Q=2	0.0438	2.22
Etc2Comp	Fast	0.0480	3.97
	Best	0.0419	0.15
ETCPACK	Fast	0.0419	0.85
	Slow	0.0375	0.0041
H-ETC2 (ours)	Fast	0.0440	127.87
	Normal	0.0421	86.15
	Best	0.0400	60.14

Unity 기준
← Fast 모드
← Best 모드
← Normal 모드

결과 (정성적 비교)

Jelly



결과 (정성적 비교)

kodim23

Original | etcpak | Betsy (Q=2) | Etc2comp (Best) | ETCPACK (Slow) | Our (Best)

비슷한 품질

KCGS 2024
Generative Imagination

한국컴퓨터그래픽스학회

2024 학술대회

2024년 7월 9일 - 12일

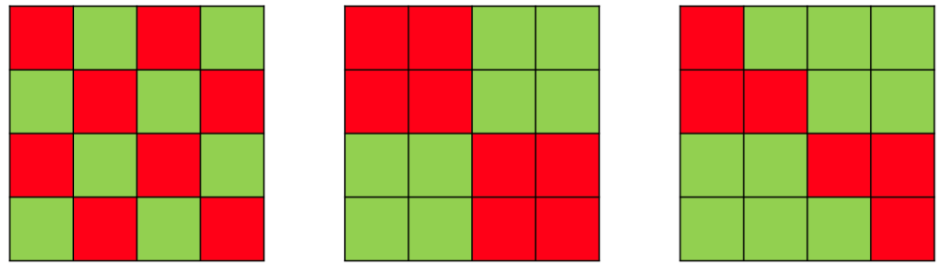
소노벨 경주

결론 및 참고문헌

결론 및 한계

- **결론**
  - CPU-GPU Hybrid 기법을 결합한 ETC2 인코딩 파이프라인 제안
  - 품질과 속도 간 trade-off를 기존 연구 대비 개선
    (Unity의 normal 및 best 모드와 비교시, 성능이 수~수십배 빠르면서도 품질도 함께 향상)

- **본 연구의 한계점**
  - 선택적 압축 방법은 특정 픽셀 패턴을 잘 다룰 수 없음
  - GPU 인코딩 파트가 전체 처리 시간 좌우
    (etcpak CPU 인코더에 비해 상대적으로 느림)



- **향후 연구 주제**
  - CPU 인코딩 파트에 새로운 알고리즘[Nah, CAG 2023]을 적용하여 GPU 부하 감소
  - CPU-GPU 인터페이스 개선을 통해 H/W 이용률 증가
  - 다른 표준 코덱(BC7, ASTC 등)에 유사한 방법 적용

- [Andersson et al., HPG 2020] Pontus Andersson, Jim Nilsson, Tomas Akenine-Möller, Magnus Oskarsson, Kalle Åström, and Mark D. Fairchild. 2020**. ꟻLIP: A Difference Evaluator for Alternating Images**. Proc. ACM Comput. Graph. Interact. Tech. 3, 2, Article 15. https://doi.org/10.1145/3406183
- [Taudul, 2022] Bartosz Taudul. 2022**. etcpak: the fastest ETC compressor on the planet**. URL: https://github.com/wolfpld/etcpak
- [Ericsson, 2018] Ericsson. 2018. **ETCPACK**. URL: https://github.com/Ericsson/ETCPACK
- [Google Inc. and Blue Shift Inc., 2017] Google Inc., Blue Shift Inc. 2017. **Etc2Comp - texture to ETC2 compressor**. URL: https://github.com/google/etc2comp
- [Goldberge, 2022] Goldberge M. N. 2022**. Betsy GPU compressor**. URL: https://github.com/darksylinc/betsy
- [Nah, TOG 2020] Jae-Ho Nah. 2020. **QuickETC2: Fast ETC2 texture compression using Luma differences**. ACM Trans. Graph. 39, 6, Article 270. https://doi.org/10.1145/3414685.3417787
- [Nah, CAG 2023] Jae-Ho Nah. 2023. **QuickETC2-HQ: Improved ETC2 encoding techniques for real-time, high-quality texture compression**. Comput. & Graph. 116, pp. 308-316. https://doi.org/10.1016/j.cag.2023.08.032
- [Ström and Petersson, GH 2007] Jacob Ström and Martin Pettersson. 2007. **ETC2: texture compression using invalid combinations**. In Proceedings of the ACM SIGGRAPH/EUROGRAPHICS Symposium on Graphics hardware, pp. 49-54. https://dl.acm.org/doi/10.5555/1280094.1280102
- [Ström and Akenine-Möller, GH 2005] Jacob Ström and Tomas Akenine-Möller. 2005**. iPACKMAN: high-quality, low-complexity texture compression for mobile phones**. In Proceedings of the ACM SIGGRAPH/EUROGRAPHICS conference on Graphics hardware, pp. 63-70. https://doi.org/10.1145/1071866.1071877
- [Pettersson and Ström, SL 2005] Martin Pettersson and Jacob Ström. 2005. **Texture Compression: THUMB—Two Hues Using Modified Brightness**. In Proceedings of Sigrad, Lund, pp. 7-12. URL: https://ep.liu.se/ecp/016/002/ecp01602.pdf
- [Smith, 2018] Stacy Smith. 2018. **Adaptive scalable texture compression**. In GPU Pro 360 Guide to Mobile Devices, pp. 153-166. AK Peters/CRC Press. URL: https://armkeil.blob.core.windows.net/developer/Files/pdf/graphics-and-multimedia/Stacy_ASTC_white%20paper.pdf