工程の手戻りを最小限に

圧縮テクスチャ (PVRTC・DXTC・ETC)における傾向と対策

CEDEC2013公募セッションでのスライド資料です
http://cedec.cesa.or.jp/2013/program/VA/6801.html

Ver.2013-09-02

WebTechnology®

突然ですが、こんな経験はありませんか？


とある制作現場の実機評価版作成風景...


実機に出力してみたら
ロードは遅いし
パッケージは大きいし
困ったな・・・


カチャ
カチャ
カチャ
PGくん（プログラマー）　２７歳

カチッ
カチッ

・・・時間をおいて、
たぶん次の日とか


どれどれ
DCさん（ディレクター）34歳
ここはDCちゃんが調整してくれればいいんじゃないかな？
ドッキーン!!
どこから手を付けていいのかわからないよう。
あれ？　これをこうしてこれを分割して・・・繋げて
試行錯誤中・・・
カチッ
カチッ
ガクッ…
この辺・・・作り直しだぁ
www.comipo.com

工程の手戻りを最小限に

圧縮テクスチャ(PVRTC・DXTC・ETC)における傾向と対策

セッションの内容
「何故この画像はこんなに汚くなっちゃったの?」と聞かれて返答に窮したことはありませんか？
本セッションでは、代表的なテクスチャ圧縮のアルゴリズム概説とともに、「実務上失敗しやすいポイントとその回避方法」「許容できないノイズが発覚したときに手戻りを最小限に抑えるテクニック」をご紹介します。

受講スキル

- いざ圧縮してみたらぼろぼろの画像になってしまった経験者
- 圧縮テクスチャの作成指示、詰め込みの指示で困っている方

受講者が得られる知見

- 変質の原因を突き止め、他のスタッフへの説明や修正ができるようになります。
- 効率的に圧縮テクスチャの最終出力をコントロールすることができるようになります。

対象プラットフォーム

コンシューマゲーム機全般、スマートフォン、タブレット、PC
このセッションでは、これらで動作するアプリケーション全般を指して、**「アプリ」**と呼ぶことにします。

アジェンダ

- 圧縮テクスチャの必要性
- 圧縮テクスチャの注意点

- DXTCの傾向と対策
- PVRTCの傾向と対策
- ETCの傾向と対策
- 16bitカラーの検討

- 最後に～手戻りを最小限に

講師自己紹介

名前　**黒岡 聡亨**
ウェブテクノロジ　R&D部リーダー

昨年からSpriteStudioの開発チームに参加
PS1,Saturnの頃からプログラマーとしてゲーム開発現場を渡り歩く

Twitter @uribow_
わりとROM専

GitHub
https://github.com/Kurooka
https://github.com/SpriteStudio

講師自己紹介

名前：上田 堂弘
ウェブテクノロジ R&D部 所属

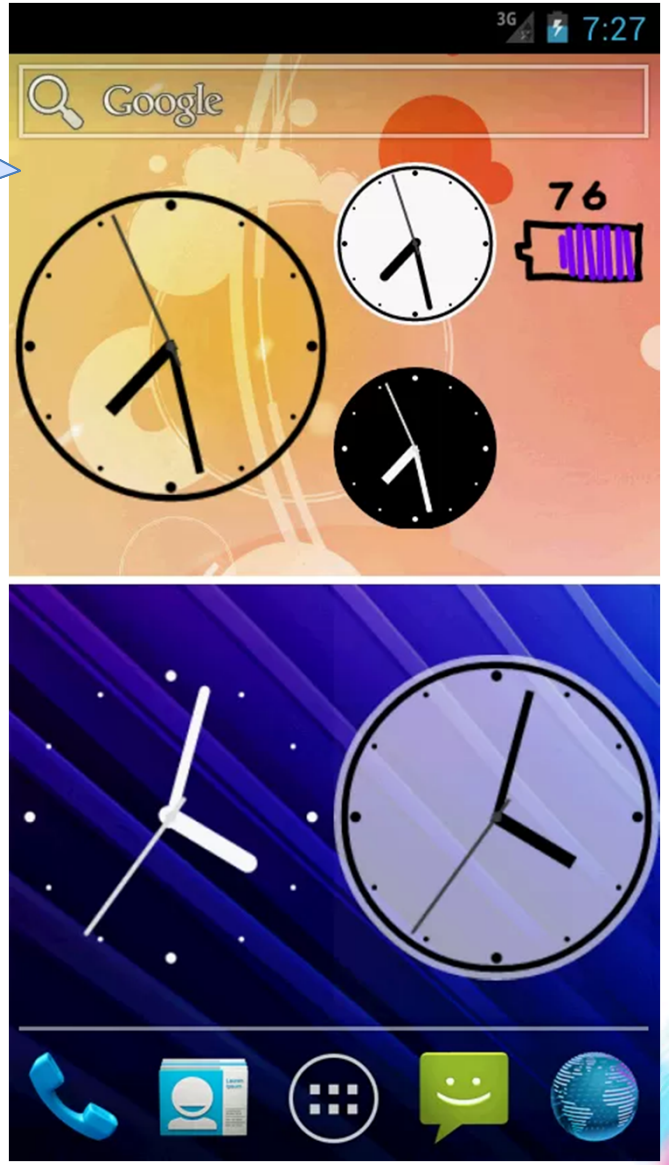
昨年までは東京工業大学の博士課程に所属
2013年、ウェブテクノロジに入社
普段の業務は測定装置などの開発担当

ゲームは好き
梅干しが嫌い

圧縮テクスチャの必要性

圧縮テクスチャの必要性

画像が少なく、
描画パフォーマンスを
要求しないアプリ



**圧縮テクスチャ
不要**

圧縮テクスチャの必要性

3Dゲームはもちろん、
2Dでも画像が多く
描画パフォーマンスを
要求するアプリなど

https://play.google.com/store/apps/details?id=com.eamobile.nfshp_na_wf&hl=ja

https://itunes.apple.com/jp/app/pazuru-doragonzu/id493470467?mt=8

圧縮テクスチャ必須

圧縮テクスチャの必要性

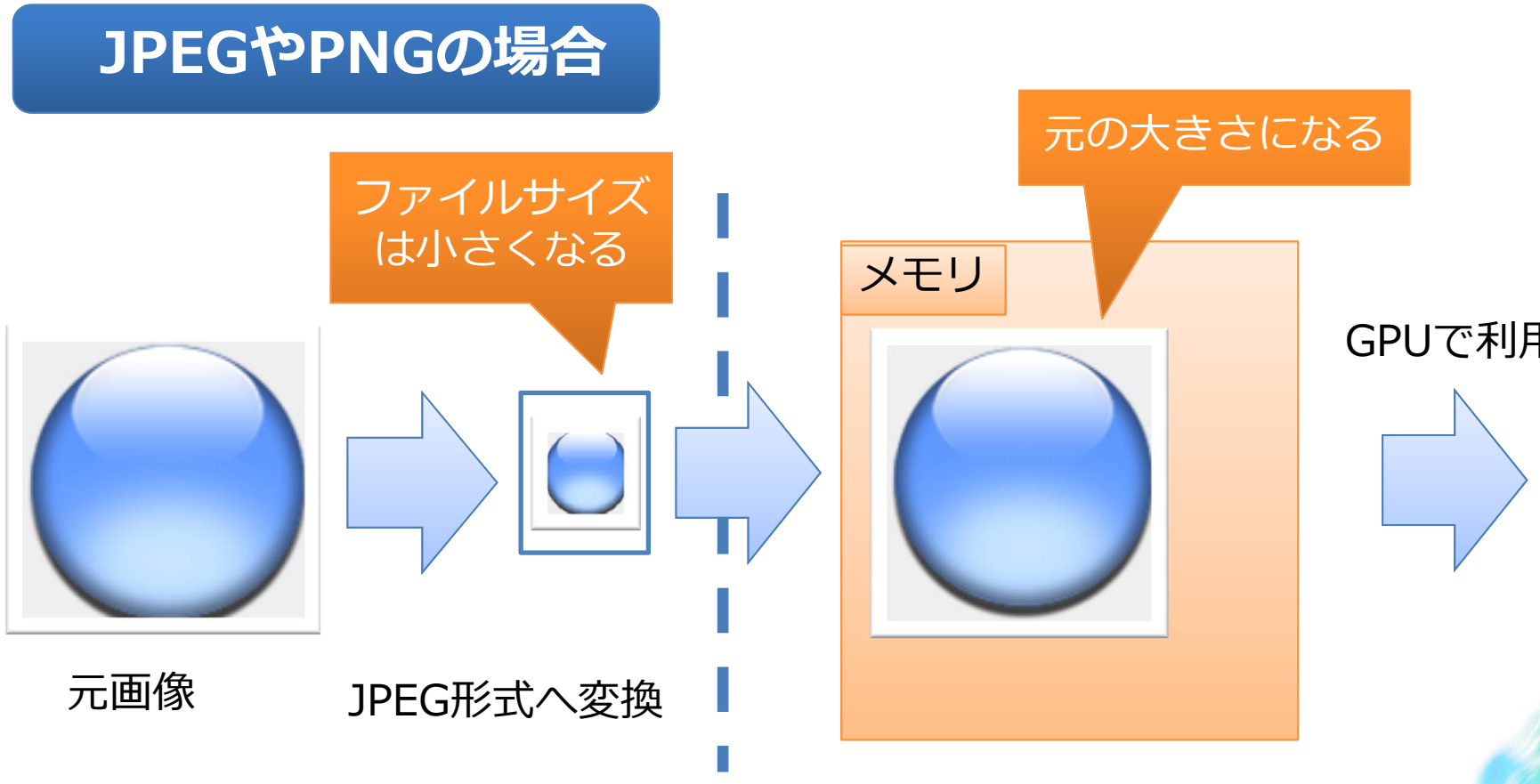
圧縮テクスチャとは

- **本来は、3Dポリゴンの表面に貼る模様を表現するための画像フォーマット**
- 現行ゲーム機、スマートフォン、PCでは、どの機種でも圧縮テクスチャが使えます
- PVRTC, DXTC, ETCなど、いくつかの形式があり、プラットフォームによって使える形式が異なります
- 圧縮率は、フルカラー画像と比べて1/4, 1/8, 1/16 程度

でも、JPEGやPNGのほうがはるかに圧縮率が高いんじゃないの？という疑問の声が…

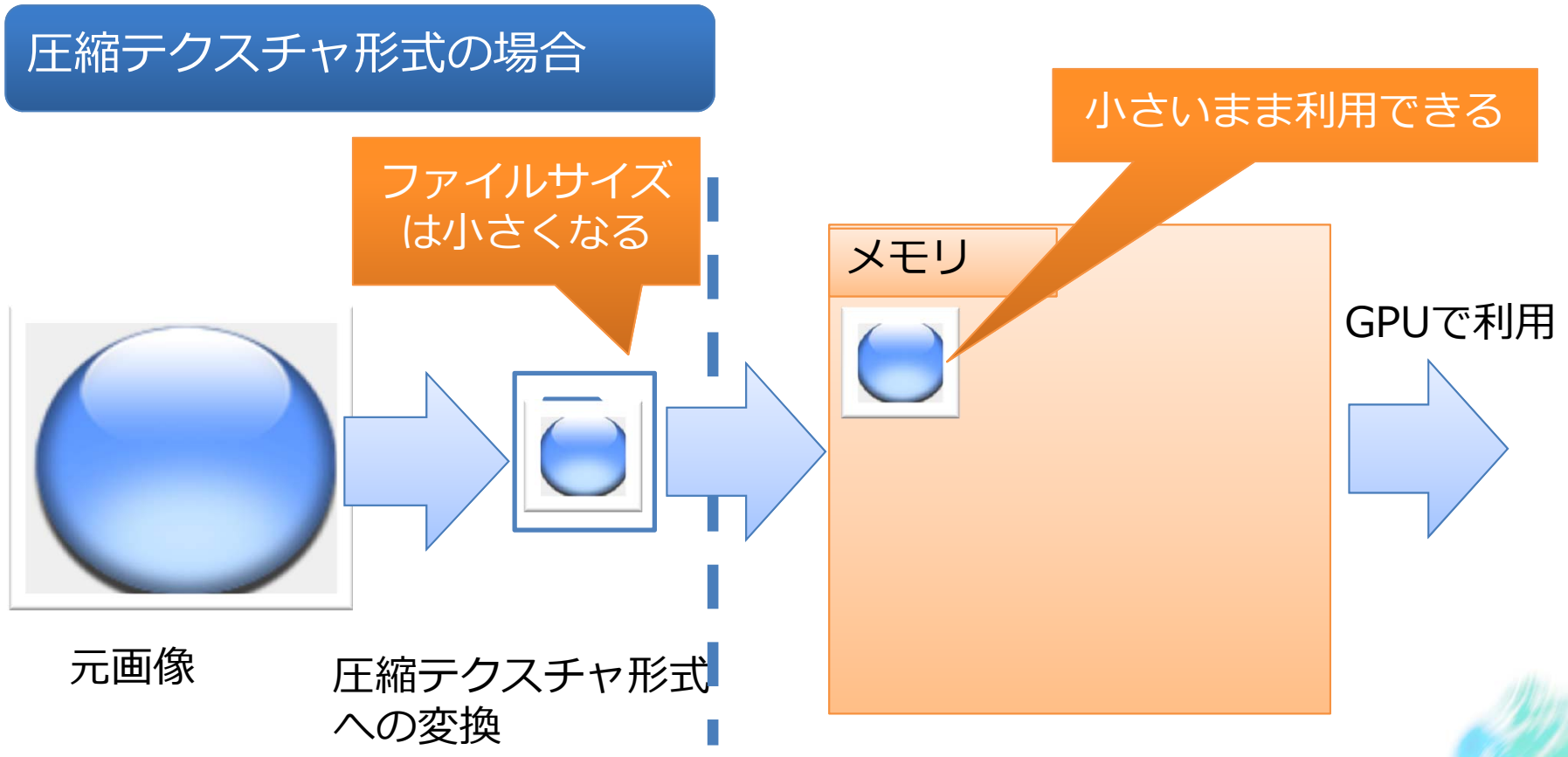
圧縮テクスチャの必要性

JPEGやPNGのほうがはるかに圧縮率が高いのに圧縮テクスチャを使う理由…



圧縮テクスチャの必要性

JPEGやPNGのほうがはるかに圧縮率が高いのに圧縮テクスチャを使う理由…



圧縮テクスチャの必要性

圧縮テクスチャを使うメリット

- 描画パフォーマンス向上
- 実行時のメモリ容量削減
  - 1つのシーンに沢山のコンテンツが入る
  - 安定動作にも有利である
- アプリ容量の削減
- 消費電力削減

画像容量が小さくなるとあらゆる点で有利！

圧縮テクスチャの注意点

その1
プラットフォーム毎に使える形式が異なります

プラットフォーム毎に使える形式が異なります

テクスチャ形式	据置ゲーム機 PC	携帯ゲーム機	iOS	Android
32bitダイレクトカラー RGBA8888	○	○	○	○
16bitダイレクトカラー RGBA4444,RGB5551,RGB565	○	○	○	○
8bitインデックスカラー	×	△	×	×
DXTC(S3TC)	◎	△	×	△
PVRTC	×	△	◎	△
ETC	×	△	×	◎
その他	×	△	×	△

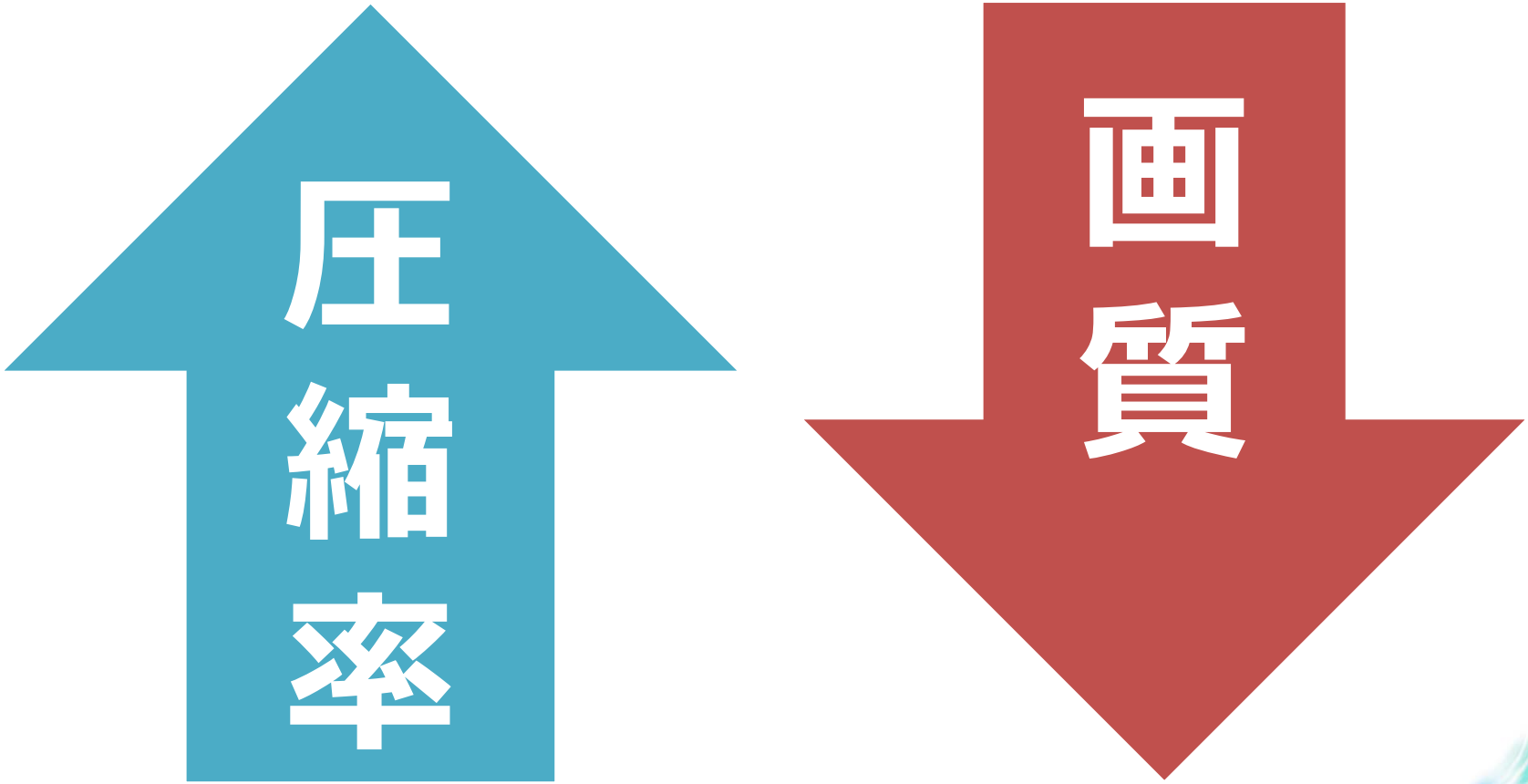
○ サポート　◎ 主要
△ 機種によりサポート
× 非サポート

圧縮テクスチャの注意点

その2
圧縮には画質劣化がつきものです

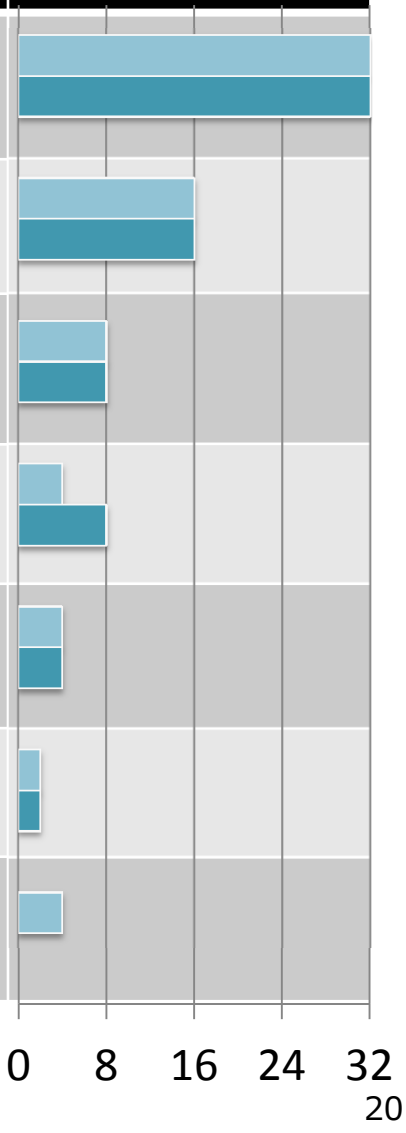
圧縮には画質劣化がつきものです

一般論として、非可逆圧縮では、画像の圧縮率を高くすると画質は低下します



テクスチャ形式ごとのbppと圧縮比

テクスチャ形式	アルファなし	アルファあり	サイズ
24bit / 32bitダイレクトカラー RGB888, RGBA8888	**32bpp**	**32bpp †**	
16bitダイレクトカラー RGB565,RGBA4444,RGB5551	16bpp	16bpp	
8bitインデックスカラー	8bpp	8bpp	
DXTC(S3TC)	**4bpp**	**8bpp**	
PVRTC 4bpp	**4bpp**	**4bpp**	
PVRTC 2bpp	**2bpp**	**2bpp**	
ETC	**4bpp**	**使用不可**	



bpp：bit per pixel
† VRAM上では32bit必要

圧縮には画質劣化がつきものです

無問題

要注意

圧縮テクスチャの必要性 – まとめ

- **圧縮テクスチャを使うことのメリットは大きい**
  - ✓ 描画パフォーマンス向上
  - ✓ 実行時のメモリ容量削減
  - ✓ アプリ容量の削減
  - ✓ 消費電力削減
- **プラットフォーム毎に使える形式が異なる**
- **アニメキャラやUIパーツのような縁取りがクッキリした画像の圧縮は苦手**
  - ✓ 2D表示では、画質が劣化すると特に目立つ

デザイン見本が出来上がったら、使用する圧縮テクスチャ形式で圧縮結果を確認する

Attention

Attention！

ここから先は各圧縮形式について個別に説明していきますが、圧縮にまつわるトラブル、疑問を掛けあい形式で進めさせていただきます。

DXTCの傾向と対策

DXTCとは？（基本情報）

WebTechnology®
OPTPiX imésta

DirectX Texture Compression

S3TC（S3 Texture Compression）の拡張

フォーマット: DXT1, DXT2, DXT3, DXT4, DXT5

圧縮単位: 4×4ピクセル

DXTCとは？　（補足情報）

- 昔懐かしいGPUメーカーS3社が開発した圧縮形式のため、S3TC(S3 Texture Compression)という別名で呼ばれることも
- DirectXで標準採用されたため、広く使われるようになった
- **据置ゲーム機やPCで広く使われている**
- Androidでも、一部の機種(Tegra搭載機等)で使用可能
- **ツールもTIPSも豊富だが、ツールにより出力にばらつきがある。**

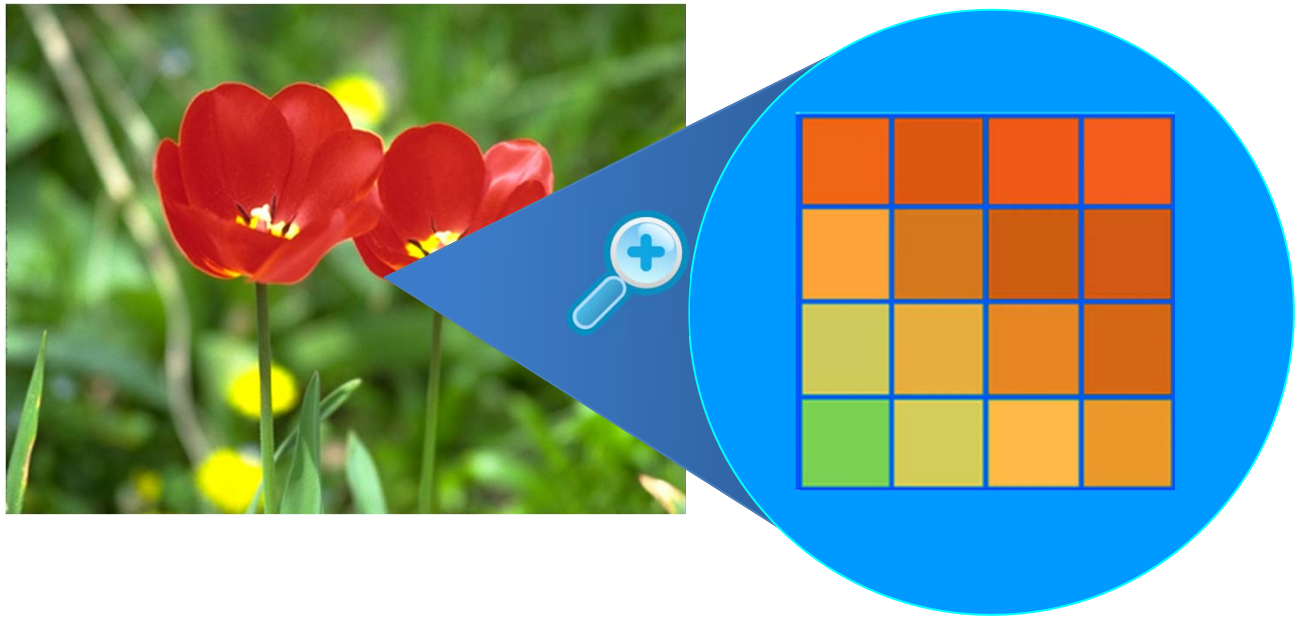
DXTCの対策

トラブル　その1
この灰色の物はなんですか？

DXTC 謎の灰色が出る件

DXTCの理論 (1) 圧縮の最小単位

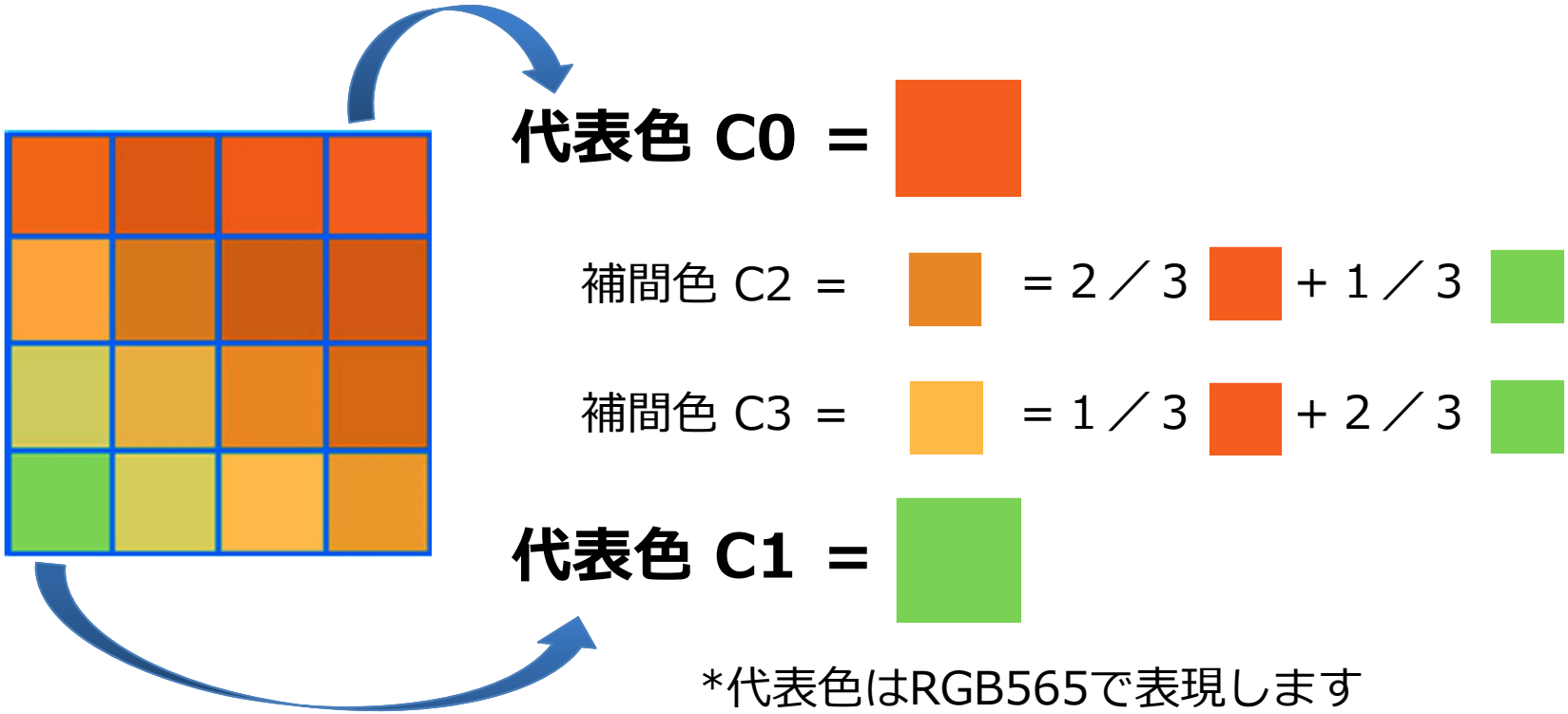
４×４ピクセルを圧縮の最小単位（１ブロック)とする



１ブロック
（４×４ピクセル）

DXTCの理論 (2) 代表色を決める

代表色を2色選び、それらの補間色を求める



*代表色はRGB565で表現します
*ブロックに無い色でも代表色に使用できます

DXTCの理論 (3) 色を置き換える

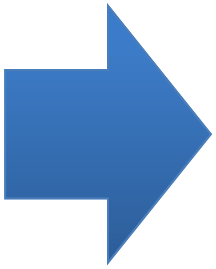
元のピクセルの色を置き換える

代表色 C0 = 0

補間色 C2 = 2

補間色 C3 = 3

代表色 C1 = 1



2	0	0	0
3	2	0	0
1	3	2	0
1	1	3	2

色にインデックス番号(0～3)を割り当て
16個の配列データとして保存する

DXTCの理論 (4) ブロック内のデータ

インデックスと二つの色の情報のみになる

2	0	0	0
3	2	0	0
1	3	2	0
1	1	3	2

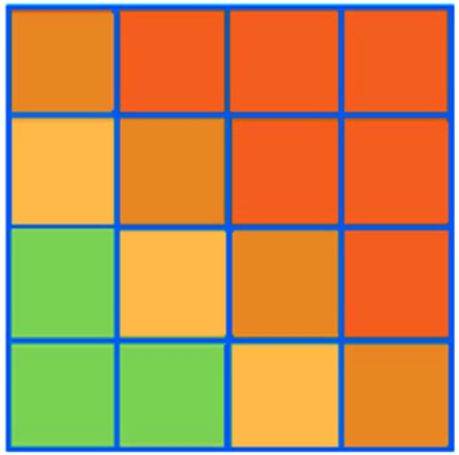
インデックス2ビット
× 16ピクセル
= **32 ビット**

代表色 C0 =

代表色 C1 =

16ビット
(RGB565)
× 2色
= **32 ビット**

復元



1ブロックあたり 64 ビット

＊ DXT2～5のアルファチャネルはこれと異なります。

DXTCのここがポイント！！

代表色 C0 = 

補間色 C2 =  = 2／3  + 1／3 

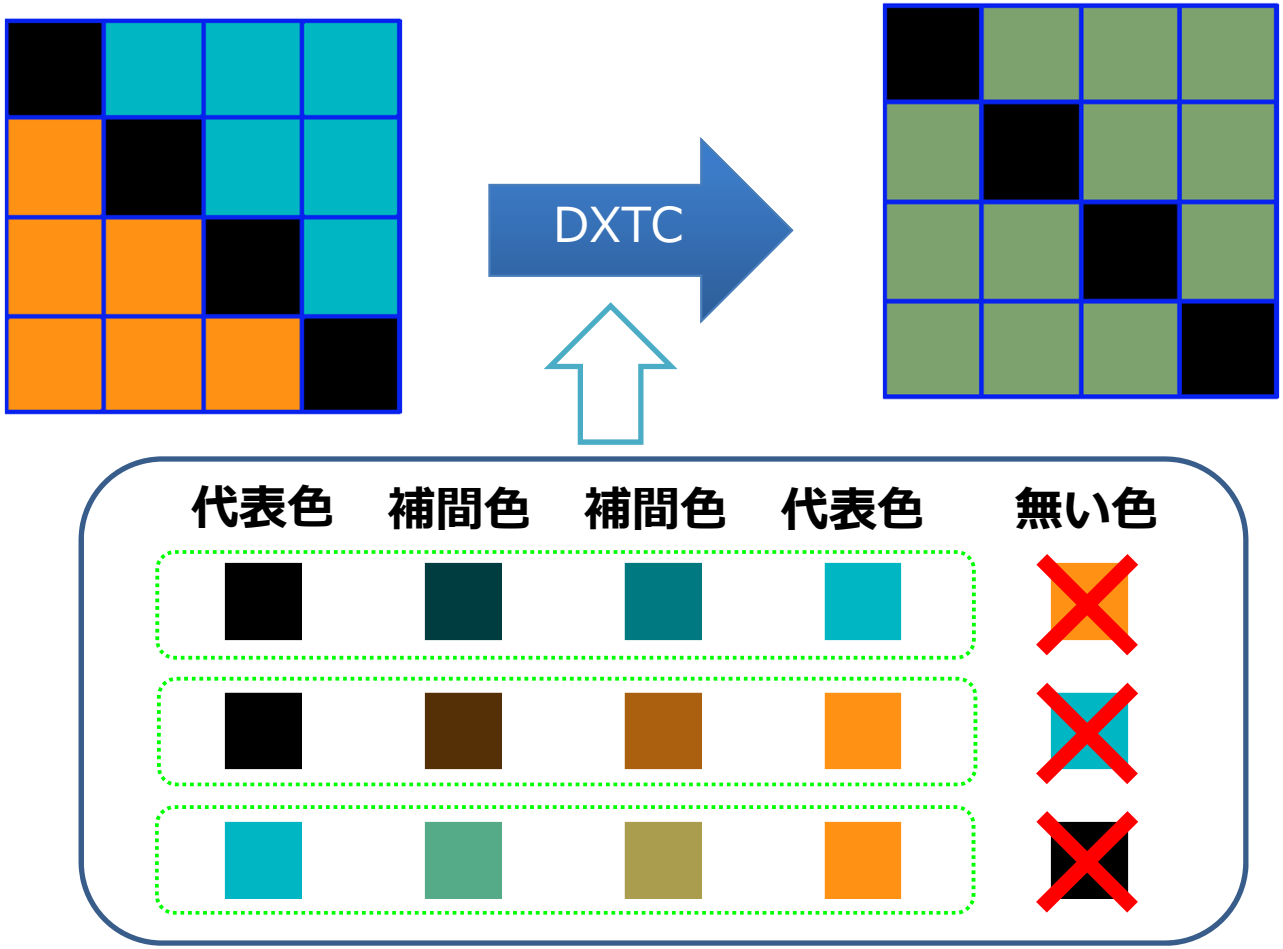
補間色 C3 =  = 1／3  + 2／3 

代表色 C1 = 

補間色は二つの代表色から自動的に
生成されて、選ぶことはできない

ブロックノイズの原因

- １ブロックに傾向の違う３色が混ざっている場合

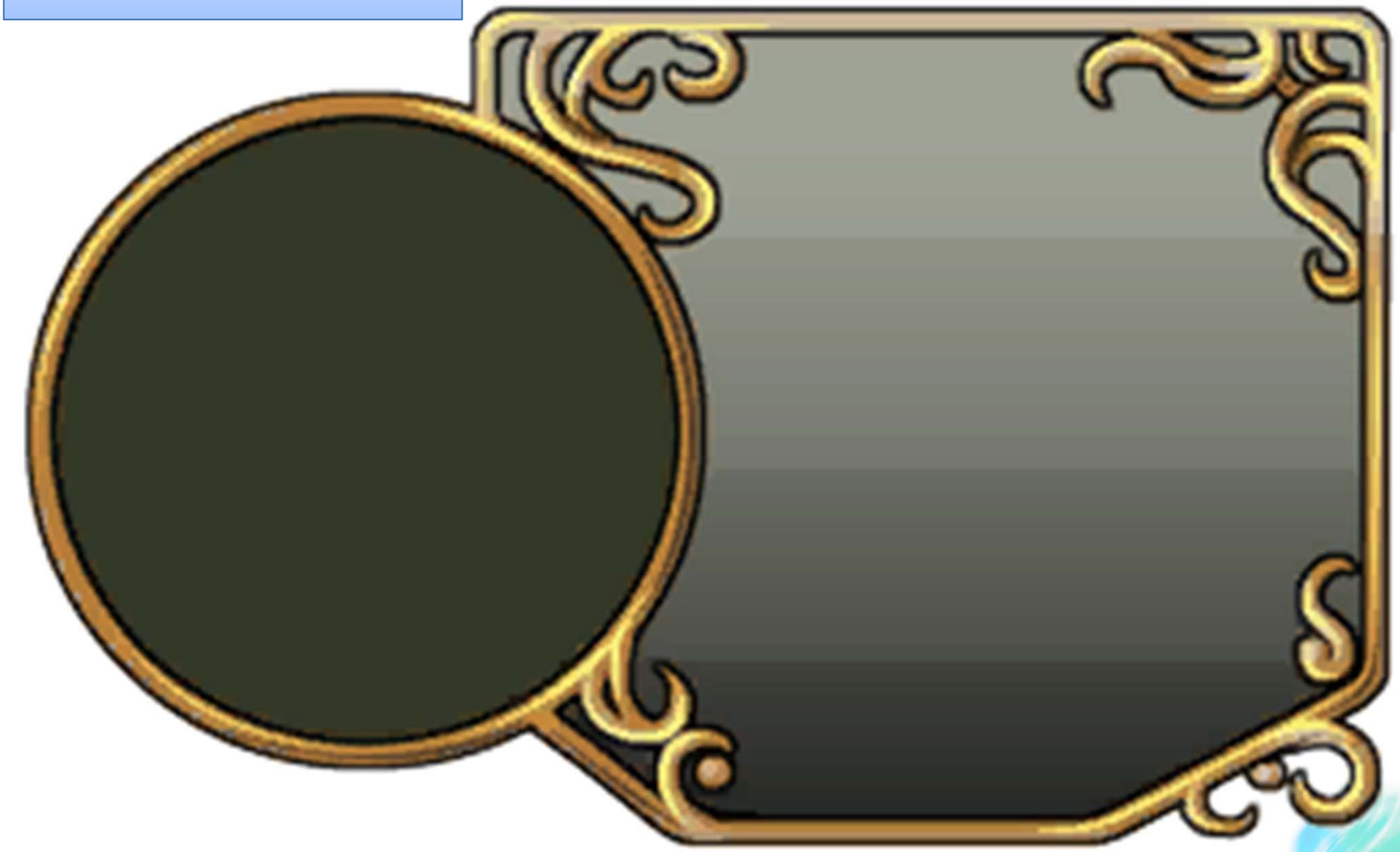


輪郭線の太さでブロックノイズをコントロール

DXTCの対策

その2
コンバート時の出力を注意しよう

DXT?



DXTCのモード使い分け

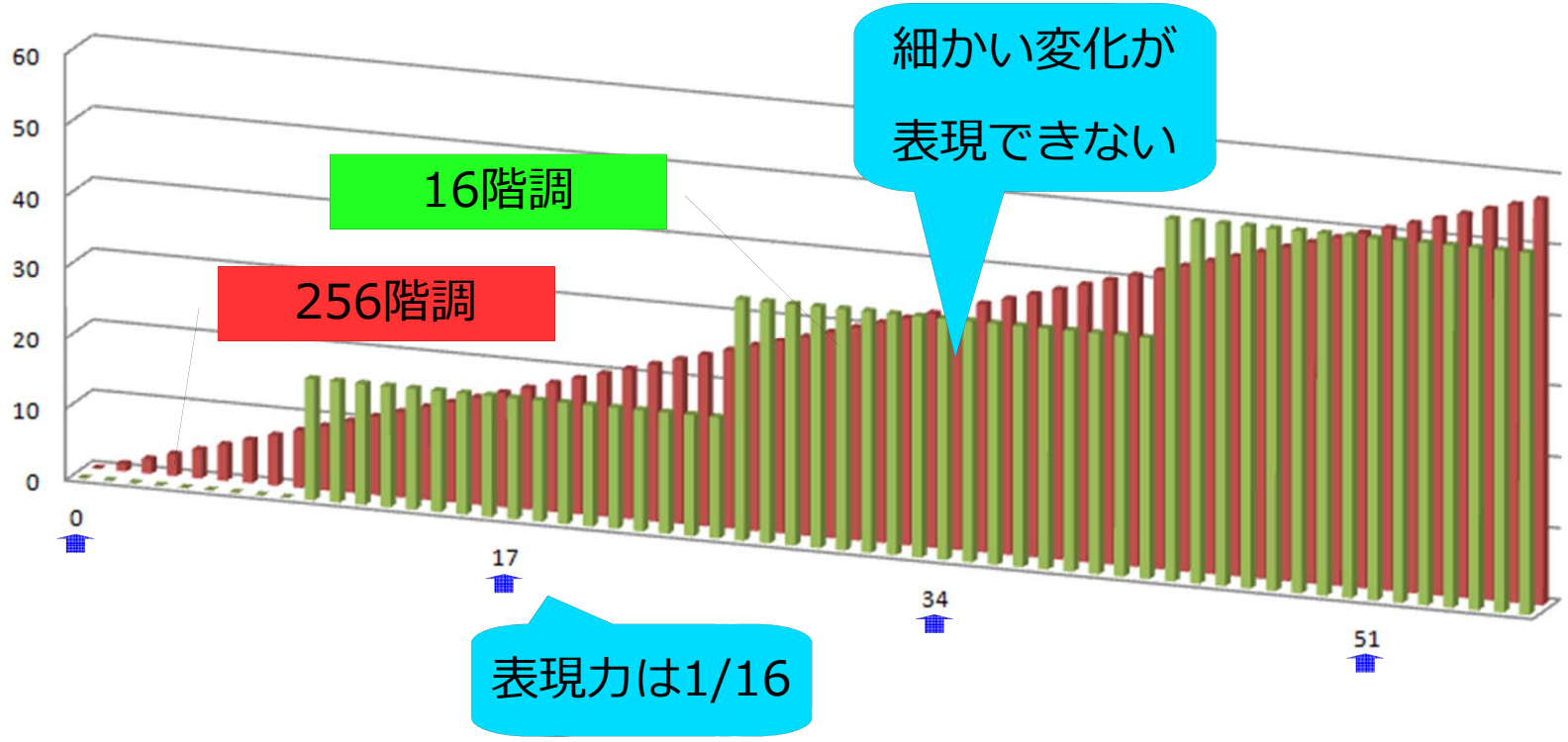
基本的にアルファ無し画像はDXT1、
アルファ有り画像はDXT5でOK

名前	RGBチャンネル	アルファ チャンネル	圧縮率*	乗算済み アルファ
DXT1 (BC1)	代表２値＋ 補間２値or１値	未使用or 1ビット(抜き)	1/8**	
DXT2	代表２値＋ 補間２値	４ビット(16階調) 固定アルファ値	1/4	○
DXT3 (BC2)	代表２値＋ 補間２値	４ビット(16階調) 固定アルファ値	1/4	
DXT4	代表２値＋ 補間２値	補間(8階調) アルファ値	1/4	○
DXT5 (BC3)	代表２値＋ 補間２値	補間(8階調) アルファ値	1/4	

*32ビット非圧縮ARGB画像からの圧縮率
**24ビット非圧縮RGB画像からの圧縮率は1/6

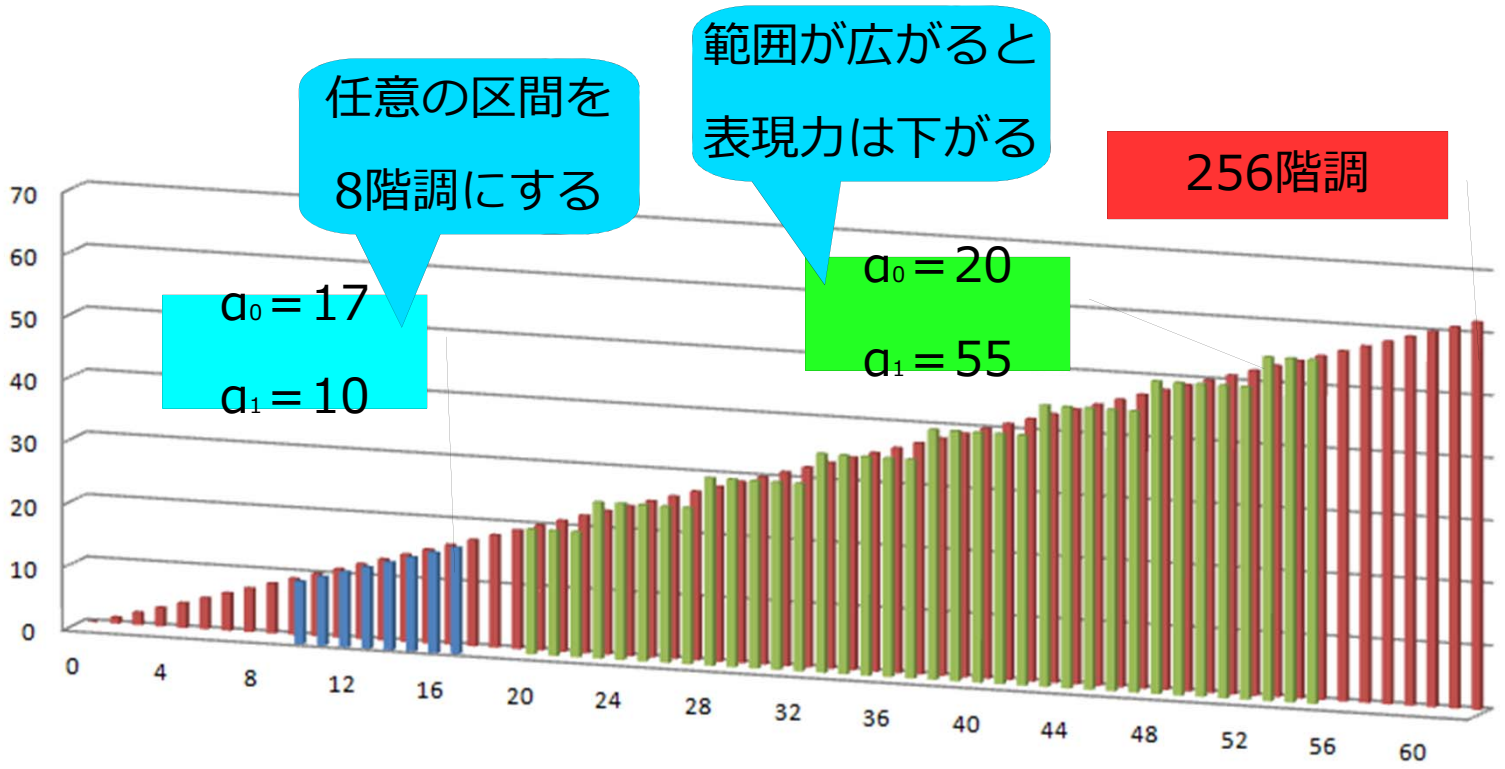
DXT3でのアルファチャンネルの扱い

アルファの値を256階調から16階調に落とす



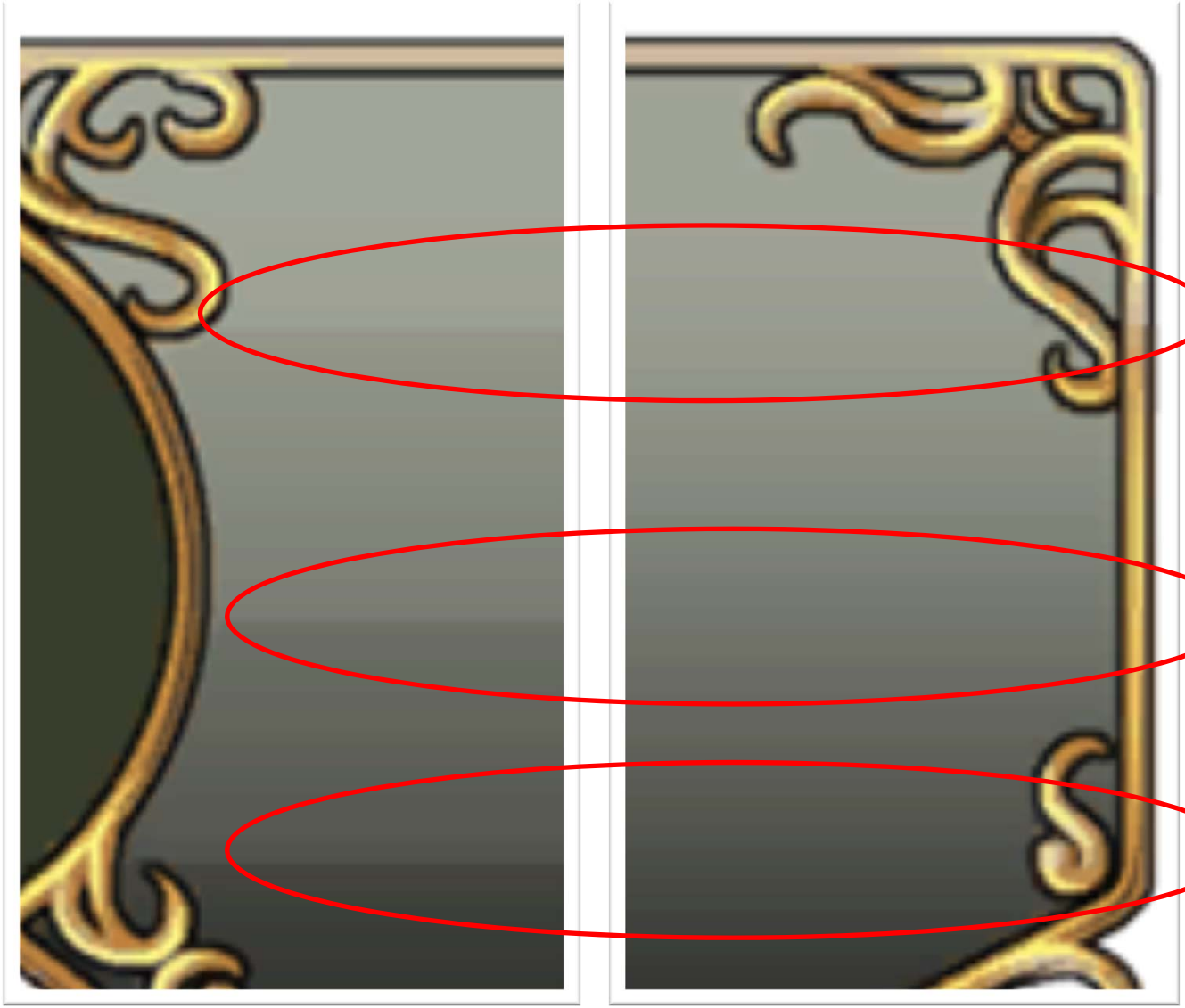
DXT5でのアルファチャンネルの扱い

任意のアルファの範囲を8階調にして使うことができる



DXT3

DXT5



DXTCにおける傾向と対策まとめ

輪郭線周辺でブロックノイズが出やすい

- 可能なら輪郭線の太さを調整する

DXTCは各モードで圧縮結果に違いがでる

- 画像毎に適切なモードを選ぶこと
  （アルファあり画像はDXT3ではなく
  DXT5で圧縮する）