鹿児島工業高等専門学校
研究報告　46（2011）
19～28

JPEG XR ビューア第 2 版の試行

加治佐　清光†　　吉田　沙央里††

Feasibility Study of JPEG XR Viewer, Version 2

Kiyomitsu KAJISA and Saori YOSHIDA

JPEG XR (eXtended Range) is an international standard for compressing continuous-tone photographic still images. Its evaluation tool, Device Porting Kit, provided by Microsoft is a command-based tool. So, in 2010, we implemented JPEG XR Viewer which is a Windows-based application using WIC (Windows Imaging Component) supported by .NET Framework 3.0, 3.5, 4 and so on. Also, the 2nd version of the JPEG XR Viewer supports not only Gray16bpp (bits per pixel) and RGB48bpp but also RGBA64bpp (A: Alpha, transparency). This technical paper reports outlines of the JPEG XR Viewer and experimental results using the 2nd version of the JPEG XR Viewer.

Keywords：JPEG XR, JPEG, WIC, PSNR

1　まえがき

JPEG XR (eXtended Range)は 2009 年に制定された静止画像圧縮の国際標準[1)]である. JPEG XR は high dynamic range (HDR)の写真市場向け[2)]と云われている．JPEG XR の符号化技術は Microsoft 社の HD Photo[3), 4)]（以前は Windows Media Photo）に基づいており，各色成分 32bpp (bits per pixel)までのビット深度(bit depth)，ロスレスとロッシーの連続性，背景との透明度を示すアルファなどをその特徴とする．圧縮性能は JPEG2000 に非常に近く，同程度に JPEG を凌駕すると報告[2)]されているが，RGBA64bpp のビット深度などについては詳細な実験報告がない. なお，HD Photo は，Microsoft 社の Windows Vista や Windows 7 に標準搭載された .NET Framework 3.0 以降によりサポートされている．

JPEG XR の評価ツール[5)]は Microsoft 社により提供されているが, 一つの試験画像の PSNR (Peak Signal-to-Noise Ratio)を得るためには，コマンドプロンプト上で符号化，復号化，PSNR 評価の三つのコマンドを入力しなければならず，非効率的であった[6)]．そのため，平成 22 年度に Windows アプリケーション版の JPEG XR ビューアを試作した．特に，その第 2 版（**図 1** を参照）では，Gray16bpp と RGB48bpp のみならず, アルファを含む RGBA 64bpp にも対応可能とした．

以下，試作した JPEG XR ビューアの第 1 版と第 2 版の概要と，第 2 版による JPEG XR の評価について報告する．

2　JPEG XR ビューア第 1 版の概要

JPEG XR ビューア第 1 版には，著者の加治佐が試作し，その実行ファイルを平成 22 年度の卒業研究班に試作例として提示した版と，共著者の吉田らが卒業研究の課題として自力で試作した版[7)]がある．いずれも，24bpp のカラー画像と 8bpp のグレイスケール画像の BMP だけを入力原画像ファイルの対象としている．ここでは，前者についてその要点を報告する．

2.1　JPEG XR ビューアの特徴

JPEG XR ビューア第 1 版の実行画面の構成は**図 1** に示した第 2 版の実行画面の構成とほぼ同じである．第 2 版では，画面下部のアルファに関する三つの数値上下コントロールと最下部の二つの BMP 保存用のボタンを追加している．

JPEG XR ビューアの実行画面は，過去に研究報告として報告したウェーブレット画像変換(2004)[8)]と 2 値算術符号化(2007)[9)]の学習ソフト（シミュレータ）と同じく，メニュー主体でなく，必要な項目が画面上で全て設定でき, 結果が見えることを特徴としている．

なお，圧縮率の表現には種々の形式があるが，本報告では，圧縮率は，ヘッダ情報を含む入力ファイルサイズ（PNG の場合は画像サイズ）に対する出力ファイルサイズの割合(%)である（**図 1** の圧縮率を参照）．

†　情報工学科
††　(株)沖電気カスタマアドテック

図 1　JPEG XR ビューア第 2 版の実行画面の例

2.2　JPEG XR ビューアの開発環境

JPEG XR ビューア第 1 版と第 2 版の試作には Visual Studio 2008 と 2010 の C#を使用した．JPEG XR の符号器（エンコーダ）と復号器（デコーダ）の API は.NET Framework 3.0 以降で WIC (Windows Imaging Component)としてサポートされている．WIC はディジタル画像用の API をサポートするプラットフォームである．Windows Vista は.NET Framework 2.0 あるいは 3.0，Windows 7 は.NET Framework 3.0 あるいは 3.5 を標準搭載している．また，Visual Studio 2008 は.NET Framework 3.0 と 3.5，Visual Studio 2010 は.NET Framework 4.0 に対応しており，.NET Framework のバージョンの切替も可能である．

後述する JPEG XR ビューア第 2 版は，Visual Studio におけるプロジェクトプロパティの対象とするフレームワークを「.NET Framework 3.5」と設定して試作した．また，Visual Studio において後述する System.Windows.Media や BitmapSource をビルド可能とするには，プロジェクトの参照設定において.NET の PresentationCore.dll を参照追加する必要がある．また，WindowsBase.dll も参照追加した．

2.3　JPEG XR の符号化オプション

JPEG XR のエンコーダを定義する System.Windows.Media.Imaging.WmpBitmapEncoder クラスには 26 のプロパティ [10]がある．そのうち JPEG XR Viewer で使用した 12 のプロパティを**表 1** に示す．これらは，**図 1** に示した画面中のチェックボックスや数値上下コントロールに対応し，WmpBitmapEncoder を用いて符号化する際のパラメータとして使用される．なお，**表 1** の下部分のアルファに関する三つのプロパティは第 1 版では使用していない．

図 1 において，Use Options あるいは Lossless が選択されていない場合は，ImageQuality だけが使用される．Use Options が選択された場合は，Image Quality は使用されず，その他のオプション用の設定値が使用される．

画面上の数値上下コントロールを設定時にどの値が何を意味するかが分かりやすいように，数値上下コントロール上にマウスポインタがある場合は，toolTip プロパティにて説明文（ツールヒント）が表示されるようにした．また，**表 1** で規定されたデフォルトの初期値は画面上に括弧（　）で表示した．

表１　WmpBitmapEncoder のプロパティ [10)]

UseCodecOptions	コーデックのオプションを使用	true: QualityLevel, OverlapLevel, SubsamplingLevel を ImageQualityLevel の代わりに使用. false: QualityLevel, OverlapLevel, SubsamplingLevel を ImageQualityLevel に基づいて設定.
Lossless	ロスレス	true: 無損失圧縮 false:有損失圧縮
ImageQualityLevel	イメージの品質レベル	範囲は 0～1.0 (lossless), 既定値は 0.9.（実装では 0.01～1.00 とした.）
QualityLevel	メインイメージの圧縮品質	範囲は 0～255, 既定値は 1. 値 1 が lossless, 値が高くなるにつれて圧縮率が高くなり, 画像品質が下がる.（実装では 1～255 とした.）
OverlapLevel	オーバーラップ処理のレベル	0: オーバーラップ処理は無効. 1: 1 レベルのオーバーラップ処理が有効. 4x4 ブロックのエンコード値を隣接するブロックの値に基づいて変更. 2: 2 レベルのオーバーラップ処理が有効. 最初のレベルの処理に加え, 16x16 マクロブロックのエンコード値を隣接するマクロブロックの値に基づいて変更.
SubsamplingLevel	サブサンプリングレベル	0: 4:0:0, クロマを破棄, ルミナンスは保持. 1: 4:2:0, クロマ解像度をルミナンス解像度の 1/4 に下げる. 2: 4:2:2, クロマ解像度をルミナンス解像度の 1/2 に下げる. 3: 4:4:4, クロマ解像度を保持.
HorizontalTileSlices	水平分割の数	値の範囲は 0～4095, 既定値は 0.
VerticalTileSlices	垂直分割の数	値の範囲は 0～4095, 既定値は 0.
FrequencyOrder	周波数順	true: 画像を周波数順に符号化. false: 画像をその空間的な向きにより符号化.
AlphaDataDiscardLevel	破棄するアルファ周波数データのレベル	CompressedDomainTranscode（圧縮ドメイン操作（画像データをデコードせずに行われる変換操作）を使用できるかどうかを示す値, 既定値は true）が true の場合に有効. 画像に planar または interleaved のアルファチャネルが含まれている場合にのみ有効. 0: 画像の周波数データを破棄しない. 1: FlexBits を破棄. 2: HighPass 周波数データ帯域を破棄. 3: HighPass と LowPass 周波数データ帯域を破棄. 4: アルファチャネルを完全に破棄.
AlphaQualityLevel	アルファチャネルの圧縮品質	InterleavedAlpha が false の場合のみ有効. 値の範囲は 0～255, 既定値は 1. 値 1 が無損失で, 値が増加するにつれて圧縮率が高くなり, 画像品質が下がる.
InterleavedAlpha	interleved チャネルとしてエンコードするかどうか	true: 画像を追加の interleaved アルファチャネルで符号化. false: planar アルファチャネルを使用.

2.4 PSNR (Peak Signal-to-Noise Ratio)

画像品質は,ピーク信号対雑音比(PSNR)により表現する．グレイスケール画像の場合，ピーク信号対雑音比は次式により算出される．

$$RMS = sqrt((\Sigma\Sigma \mid f_{i,j} - d_{i,j} \mid^2) / MN) \quad (1)$$

$$PSNR = 20\ log_{10}\ (255 / RMS) \quad (2)$$

ここで，f は座標(i, j) における原画像の画素値，d は座標(i, j) における復元画像の画素値，M, Nは縦横の画素数，$sqrt$ は平方根であり，RMS は二乗平均平方根誤差(Root Mean Square error)を示す．

RGB のカラー画像の場合は，式(1)において画素ごとに 3 色成分の絶対値誤差の総和を求め，総画素数 MNに代わり 3 倍の画素数で割れば，カラー画像としての RMS が求められる．すなわち，カラー画像の場合，ピーク信号対雑音比は次式により算出される．

$$RMS = sqrt((\Sigma\Sigma\Sigma \mid f_{i,j,k} - d_{i,j,k} \mid^2) / 3MN) \quad (3)$$

ここで，$k=0, 1, 2$ は RGB の 3 色成分である．

PSNR の算出にあたり，第 1 版では原画像と復元画像の画素値の取得に原画像と復元画像の表示用の Bitmap から GetPixel する手法を採用したが，高解像度の画像では処理時間が非常に長くなる欠点があり，これは第 2 版で改良した（**3.3** で後述する）．

(a) 原画像 Cathedral の窓枠　　(b) JPEG XR, 圧縮率 0.26%　　(c) JPEG, 圧縮率 0.57%

図２　目視検査のための拡大表示の例 (RGB24bpp)

2.5　拡大表示

画像品質の評価として定量的に表現できる PNSR に加えて，目視による実際の画像品質の評価は重要である．そのため，JPEG XR ビューアでは限られた画像表示領域の最大 256×256 画素内ではあるが，拡大・縮小表示機能を装備した．

拡大・縮小表示機能の実現には種々の方法が考えられるが，JPEG XR ビューアでは，表示された原画像(bitmap1)，JPEG XR 復元画像(bitmap2)，あるいは JPEG 復元画像(bitmap3)のいずれかの表示画像をマウスでポイントした点を中心として左ボタンのクリックで縦横を順に 2 倍に拡大し，右ボタンのクリックで順に 1/2 倍に縮小する方式を採用した．縦横の 2 倍への拡大は，1 画素をそのまま 4 画素へコピーする方式とした．この方式により復元画像の各画素が原画像の各画素に比べてどのように画質劣化しているかが目視で確認できる．

詳細は割愛するが，具体的には，画像表示領域は最大 256×256 画素と小さいため，簡単ではあるが一般に処理速度が遅いと云われている方法である GetPixel と SetPixel メソッドを使用して，拡大・縮小の機能を実現した．

図２に目視検査のために画像の指定部分を拡大表示した例を示す．同図中の PSNR は，拡大表示された復元画像領域だけに対する PSNR であり，圧縮率は画像全体の圧縮率である．この例では，JPEG XR はより低い圧縮率ながら PSNR も目視結果も JPEG よりも良好であることが分かる．

3　JPEG XR ビューア第 2 版の概要

第 2 版では，JPEG XR の仕様に含まれる 48bpp などのビット深度とアルファに対応できるように改良した．そのため，入力原画像ファイルに，グレイスケールが 8bpp，カラーが 24bpp に限定される BMP に，16bpp のグレイスケール，24bpp と 48bpp の RGB フルカラーとアルファチャネルに対応する PNG を加えた．以下，それらに付随する第 2 版の要点について報告する．

3.1　処理の流れ

図3に JPEG XR ビューア第 2 版の処理の流れを示す．太線の枠が WIC で規定されている API 関数である．それらの API 関数の入出力となる BitmapSource は画素セットを特定のサイズと解像度で表す抽象クラス, MemoryStream はメモリ使用のストリームのクラス，Bitmap は画素データとその属性で構成するクラスである．第 2 版の特徴は，PSNR を算出する処理時間の短縮のため，Bitmap ではなく BitmapSource をデコーダの出力としている点である．BitmapSource は表示用の Bitmap へ直接に対応付けられないために，ここでは，MemoryStream へ変換して対応付けた．

この図の例では原画像に PNG を使用しているが，BMP も使用できる．以下，処理の流れを列記する．

- 入力原画像ファイルの BMP あるいは PNG を Bmp BitmapDecoder あるいは PngBitmapDecoder でデコードし，デコードした BitmapSource を Bit

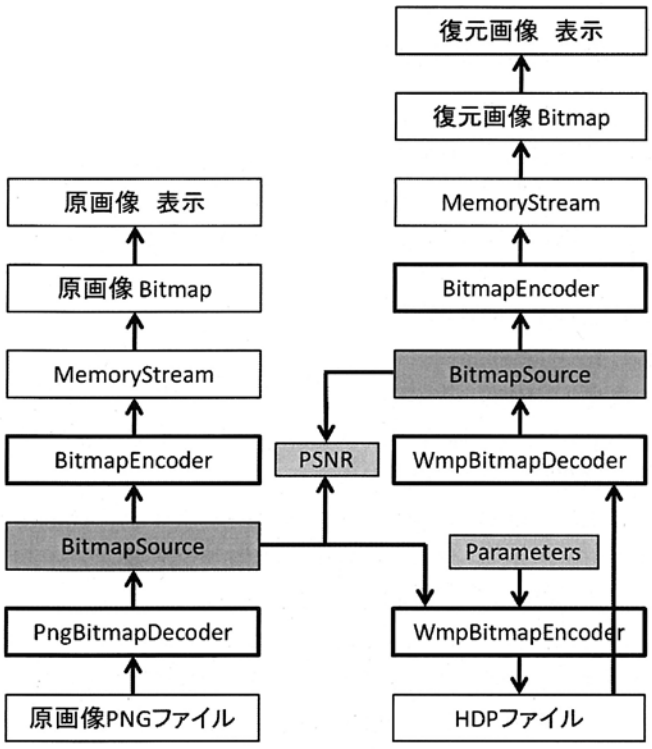


図３　JPEG XR ビューア第 2 版の処理の流れ

表２　主なピクセル形式

PixelFormat 列挙体 *1 for Bitmap		PixelFormats メンバ *2 for BitmapSource	
Format8bppIndexed	○	Bgr24	○
Format16bppGrayScale		Bgr32	
Format24bppRgb	○	Bgra32	○
Format32bppRgb	○	Bgr48	○
Format32bppArgb		Indexed8	○
Format32bppPArgb		Gray8	○
Format48bppRgb		Gray16	○
Format64bppArgb	○	Gray32Float	
Format64bppPArgb		Rgb24	
		Pbgra32	
		Rgb48	○
		Prgba64	
		Rgba64	○
		Rgb128Float	
		Rgba128Float	
		Prgba128Float	

*1: System.Drawing.Imaging.PixelFormat

*2: System.Windows.Media.PixelFormats

○: JPEG XR ビューア第 2 版で実際に使用

mapEncoder で MemoryStream へ一時保存し，MemoryStream を Bitmap へ貼り付け，原画像を画面表示する．

- デコードした BitmapSource を BitmapFrame を経由し，ビューアの画面上で設定されたパラメータを使用して，JPEG XR 用の WmpBitmapEncoder でエンコードし，JPEG XR の圧縮ファイルである HDP としてファイル保存する．
- 保存した HDP ファイルを JPEG XR 用 WmpBitmapDecoder でデコードし，デコードした BitmapSource を BmpBitmapEncoder で Memory Stream へ一時保存し，MemoryStream を Bitmap へ貼り付け，復元画像を画面表示する．
- 原画像の BitmapSource と復元画像の Bitmap Source 間で PSNR を算出する．

3.2　ピクセル形式

前述の BmpBitmapDecoder あるいは PngBitmapDecoder でデコードする際は，BitmapCreateOptions.PreservePixelFormat により，原画像のピクセル形式を保持するようにする．デコードした BitmapSource の Format プロパティとして割り当てられる主なピクセル形式を**表2**の右欄に示す．また，画像表示するための Bitmap の PixelFormat プロパティとして割り当てられる主なピクセル形式を**表2**の左欄に示す．

例えば，RGB24bpp の BMP を入力すると，Bitmap Source と Bitmap のピクセル形式は，Bgr24 と Format 24bppRgb となる．RGB48bpp の PNG を入力すると，それぞれ Bgr48 と Format64bppArgb，アルファ付きの RGBA64 の PNG を入力すると，Rgba64 と Format 64bppArgb となる．

表2中の○印は JPEG XR ビューア第 2 版で実際に実験で使用したピクセル形式である．**表2の**右欄中で末尾に Float の付くピクセル形式は浮動小数点数で画像データを表現した場合で，JPEG XR ではサポートされているが，今回の試験画像としては使用していない．

入力原画像ファイルがデータ圧縮された PNG の場合，圧縮率の算出には，ここではヘッダ部を除く画像サイズを使用した．具体的には，BitmapSource.Format.BitsPerPixel からビット深度を定め，画像サイズを求めた．

3.3　PSNR (Peak Signal-to-Noise Ratio)

16 ビットデータ対応の PSNR を算出するプログラムの試作の前に，まず，上位バイトと下位バイトの位置の確認を行った．実験で使用した参考文献 11)のダウンロード用の試験画像は PGM (Portable Gray Map)と PPM (Portable Pixel Map)であったので，確

認にはこれらの書式を使用した.

試験画像 PGM の 16bpp のデータは Big Endian で上位バイト，下位バイトの順（例えば，0x44, 0x2A の順）に格納されており，BitmapSource.CopyPixels で格納する配列には下位アドレスより Little Endian で下位バイト，上位バイトの順（例えば，0x2A, 0x44 の順）に格納される.

同様に, 試験画像 PPM の 48bpp のデータは R, G, B の順でそれぞれ Big Endian の 2 バイト（例えば，R: 0x56, 0x29; G: 0x42, 0x26; B: 0x1F, 0x82 の順）が，BitmapSource.CopyPixels により，それぞれの色成分が Little Endian で格納される（例えば，0x29, 0x56, 0x26, 0x42, 0x82, 0x1F の順）.

JPEG XR ビューア第 1 版は 8 ビットのグレイスケール画像(8bpp)と 8 ビットの色成分の RGB カラー画像(24bpp)に対応していたが，JPEG XR ビューア第 2 版は 16 ビットのグレイスケール画像(16bpp)と 16 ビットの色成分の RGB カラー画像(48bpp)に対応する. また，PSNR は 8 ビットおよび 16 ビットの色成分のアルファ付きの RGBA カラー画像（32bpp，64bpp）にも対応している. PSNR 用の bpp も, BitmapSource. Format. BitsPerPixel プロパティにより判定する.

具体的には，RMS の算出には，8bpp と 16bpp の場合は前述の式(1)を，24bpp，32bpp，48bpp および 64bpp の場合は式(2)を適用した. また，16bpp, 48bpp および 64bpp の場合は，BitmapSource. CopyPixels で格納した配列を上述の 2 バイトでアクセスする方式とし，式(2)において信号 255 ではなく信号 65535 に対する PSNR を算出する.

3.4　アルファ

背景画面との透明度を制御するアルファ機能は，JPEG ではサポートされていないが，GIF，PNG，および JPEG XR ではサポートされている. さらに，JPEG XR では，**表 1** に示したように，WmpBitmap Encoder のプロパティとして，AlphaDataDiscard Level, AlphaQualityLevel, および InterleavedAlpha の三つが規定されている.

JPEG XR ビューア第 2 版の試行では，これらの三つのプロパティの設定を組み込んだが，現時点では，これらのプロパティを変化させた実験では妥当な実験結果を得ることができなかった. そのため，次の「**4** 第 2 版による JPEG XR の評価」では，オプションの設定を除くアルファ成分を含む画像の評価に留める.

4　第 2 版による JPEG XR の評価

ここでは，試行中ではあるが，JPEG XR ビューア第 2 版を用いて得られた評価結果について報告する. アルファを除く試験画像にはコントラストの高い高解像度の画像 Cathedral[11]（**図 1** を参照）を使用した.

4.1　JPEG との比較

4.1.1　RGB24bpp

まず，JPEG と JPEG XR で共通に扱える RGB 24bpp のカラー画像 Cathedral.BMP を使用して実験した結果について報告する.

JPEG では Quality を 100～5, JPEG XR では Image Quality を 0.99～0.01 の範囲で変化させた場合の圧縮率と PSNR の関係を**図 4** に示す.

RGB24bpp のカラー画像で最高の高画質の復元画像を得たい場合，JPEG では Quality=100 で圧縮率が 25.79%，PSNR が 41.2dB の復元画像しか得ることができないが，JPEG XR では圧縮率が 51.48%でロスレス符号化が可能である. 他方，JPEG XR では Image Quality を 0.99～0.90 の範囲で変化させることにより，

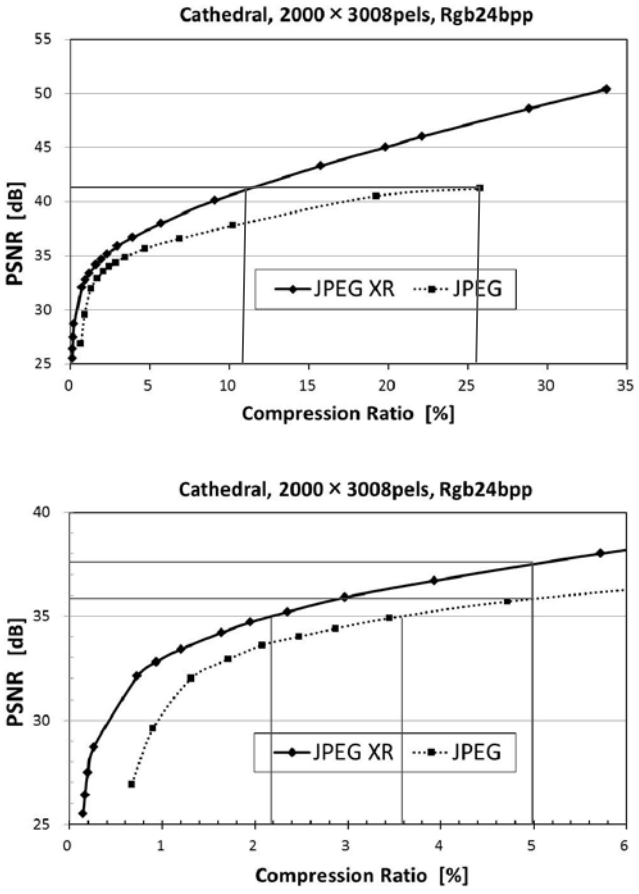


図 4　RGB24bpp の圧縮率と PSNR

PSNR が 50.4～40.1dB の JPEG 復元画像よりも高画質な復元画像を得ることができる．また，JPEG で最高の 41.2dB(圧縮率は 25.79%)と同じ PSNR は JPEG XR では 11.36%と半分以下の圧縮率で可能である．

圧縮率 5%以上ではいずれも圧縮率に比例して PSNR が増加する．圧縮率に対する PSNR は JPEG よりも JPEG XR の方が良く，同じ圧縮率ではより高画質の復元画像を得ることができる．

図４において圧縮率が同じ 5%の場合，JPEG の PSNR は約 37.5dB であるが，JPEG XR の PSNR は約 36dB と，約 1.5dB のより高画質な復元画像が得られる．また，PSNR が同じ 35dB の場合，JPEG の圧縮率は約 3.6%であるが, JPEG XR の圧縮率は約 2.2%と，約 1.4%ほど圧縮率が低い．これは，圧縮率の別の表現で，一般に JPEG XR の圧縮率は JPEG よりも約 2 倍良いと云われる根拠であると推察される．

上記の実験結果により, 圧縮率対 PSNR の性能の観点からは，JPEG XR は JPEG よりも高い性能が得られることが分かる．

4.1.2 Gray8bpp

次に, JPEG と JPEG XR で共通に扱える Gray8bpp のグレイスケール画像 Cathedral.BMP を使用して実験した結果について報告する．

JPEG では Quality を 100～1, JPEG XR では Image Quality を 0.99～0.01 の範囲で変化させた場合の圧縮率と PSNR の関係を**図５**に示す．

全般的に同じ圧縮率では JPEG より JPEG XR の方が高画質の復元画像が得られる．圧縮率が同じ 10%の場合，JPEG の PSNR より JPEG XR の PSNR は約 2dB だけ良い. また, PSNR が同じ 35dB の場合, JPEG の圧縮率は約 3.5%であるが，JPEG XR の圧縮率は約 2%と，約 1.5%ほど圧縮率が低い．これは上述の RGB 24bpp の場合と同様の結果である．

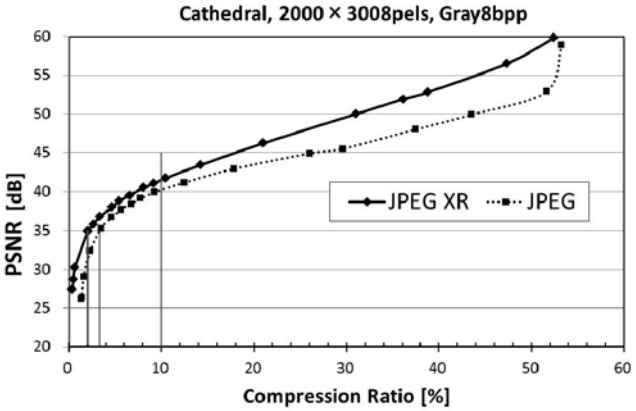
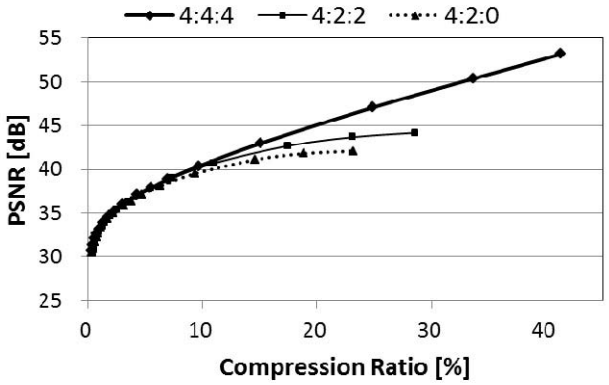


図５　Gray8bpp の圧縮率と PSNR



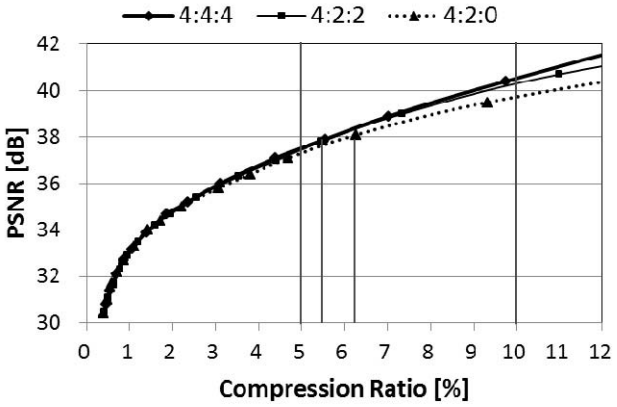


図６　RGB24bpp の圧縮率と PSNR

4.2 サブサンプリング

RGB 24bpp のカラー画像 Cathedral.BMP を使用してサブサンプリングを変化させた場合の圧縮率と PSNR の関係を**図６**に示す．ここでは，Use(Codec) Options を選択し，サブサンプリングと MainQuality 以外はデフォルトのままとした．

1/10 以上の圧縮率では，色差成分を間引く 4:2:2 と 4:2:0 は 40～44dB で飽和する．つまり，圧縮率を高めても，45dB 以上の高画質な復元画像を得ることはできない. 他方, 4:4:4 の場合は, 圧縮率は犠牲になるが，ほぼリニアに 45dB 以上の高画質な復元画像を得ることができる．

1/20 の圧縮率の場合，PSNR の差はほとんどなく，色差成分を間引いても，画像品質はほとんど同じである．4:2:2 や 4:2:0 を使用する効果は見られない．1/10 の圧縮率の場合も，4:4:4 は 4:2:0 より約 1dB ほど画像品質が良いが，4:2:2 や 4:2:0 を使用する効果は見られない．

38dB の画像品質の復元画像を得るには，4:2:0 では

圧縮率は約 6.2%だが, 4:4:4 では圧縮率が約 5.5%で得られる．つまり，色差成分を間引く効果は画像品質の評価に PSNR を用いた今回の実験では観測できない．

今回の報告と同様な実験結果は，卒業研究での第 1 版と RGB 24bpp のカラー画像 lena を使用した実験でも観測された [7]．サブサンプリングは本来，人間の視覚には敏感でない色差成分を間引いて圧縮率を改善することが目的である．期待通りの実験結果が得られないこの問題については今後の課題としたい．

4.3　16 ビットのグレイスケール画像

Gray16bpp のグレイスケール画像 Cathedral.PNG を使用して Image Quality を 0.99～0.01 の範囲で変化させた場合の圧縮率と PSNR の関係を，上述の Gray8bpp と比較して，**図 7** に示す．圧縮率 5%以上ではいずれも圧縮率に比例して PSNR が増加する．圧縮率に対する PSNR は Gray8bpp よりも Gray16bpp の方が極めて良く，同じ圧縮率では高画質の復元画像を得ることができる．ImageQuality = 0.99 の設定で Gray8bpp は 59.84dB の画像品質しか得ることができないが，Gray16bpp は 100.8dB の高画質の復元画像を得ることができる．逆に，ロスレス符号化の場合，Gray16bpp の圧縮率は 75.41%であるが，Gray8bpp の圧縮率は 51.14%と小さく圧縮できる．

図 7 の下のグラフから，圧縮率が同じ 10%の場合，Gray8bpp の PSNR に対して Gray16bpp の PSNR は約 4dB のより高画質な復元画像が得られることが分かる．また，PSNR が同じ 38dB の場合，Gray8bpp の約 4.6%の圧縮率に対して，Gray16bpp の圧縮率は約 2.2%と，約半分の圧縮率で済むことが分かる．したがって，圧縮率対 PSNR の性能の観点からは，Gray16bpp は Gray8bpp よりも高い性能が得られると云える．しかし，**図 7** の下のグラフから明らかなように，Gray16bpp では Image Quality = 0.01 での PSNR = 36.6dB が下限であり, それ以下の低画質な画像を得ることはできないという欠点が現在のエンコーダ（WmpBitmapEncoder）のパラメータ設定に存在する．

特記事項として，Windows 7 (64 ビット版)上の Visual Studio 2010 で開発した JPEG XR ビューア第 2 版の実行プログラムを Windows 7 (64 ビット版と 32 ビット版)上で使用すると，Gray16bpp の画像に対してのみ動作しなかった（ImageQuality を変化させても圧縮率と PSNR は異常な不変値であった）が，Windows Vista (32 ビット版) 上では Gray16bpp の画像に対しても期待通りに動作した．現時点では，原因は不明である．ここでは，上記の Gray16bpp に関する実験データだけは Windows Vista (32 ビット版) 上で得た実験データ（それ以外は 64 ビット版の Windows 7 上で得た実験データ）であることを付記しておく. この問題に関しては今後の検討が必要である．

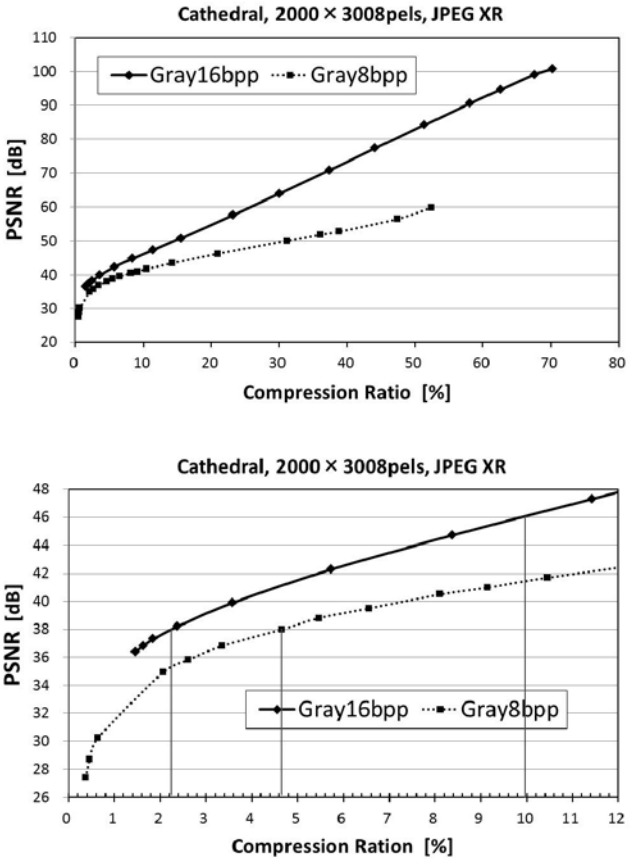


図 7　Gray16bpp の圧縮率と PSNR

4.4　48 ビットのカラー画像

Image Quality を 0.99～0.01 の範囲で変化させた場合の圧縮率と PSNR の関係を**図 8** に示す．圧縮率 5%以上ではいずれも圧縮率に比例して PSNR が増加する．圧縮率に対する PSNR は RGB24bpp よりも RGB48bpp の方が良く, 同じ圧縮率では高画質の復元画像を得ることができる．ImageQuality = 0.99 の設定で RGB24bpp は 50.4dB の画像品質しか得ることができないが，RGB48bpp は 82.8dB の高画質の復元画像を得ることができる．しかし，ロスレス符号化の圧縮率で比較すると，RGB24bpp の 51.48%に比べて RGB48bpp は 76.66%と非常に圧縮率が悪い．

上述の実験結果は Gray16bpp の実験結果と同じ結果である．ビット深度が深くなると圧縮率が悪くなる原因は現時点では不明である．

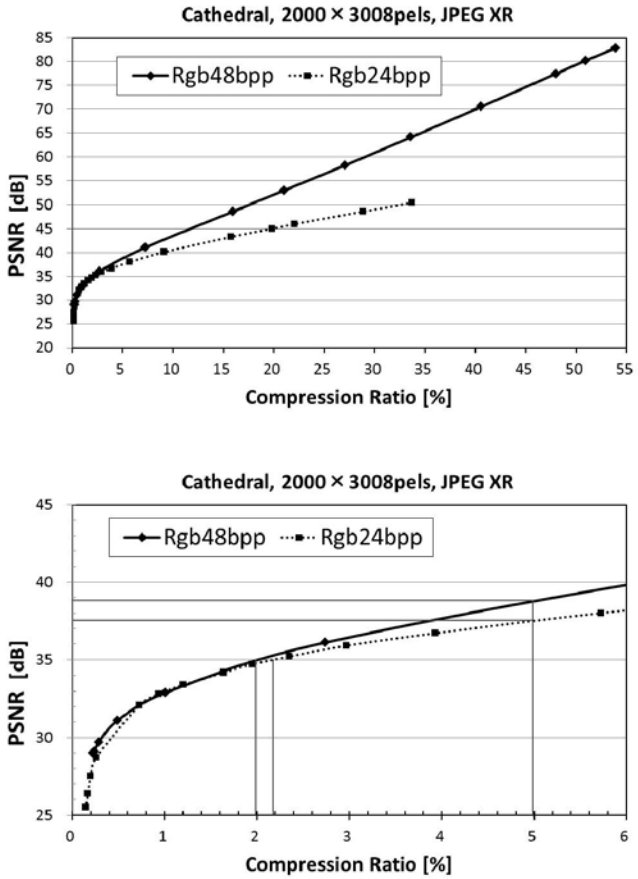


図８　RGB48bpp の圧縮率と PSNR

図８の下のグラフにおいて，圧縮率が同じ 5%の場合，RGB48bpp では RGB24bpp よりも PSNR が約 1dB だけ良好な復元画像が得られるが大差はない．また，PSNR が同じ 35dB の場合，RGB24bpp の圧縮率に対して RGB48bpp の圧縮率は約 0.2%だけ低いが，大差はない．したがって，通常用いられる圧縮率の領域においては，圧縮率対 PSNR の性能の観点からは大差はないと云える．この実験結果は，前述の Gray16 bpp の場合とは異なる．

4.5　アルファ

アルファの試験画像には，“The official test-suit for PNG”サイト[12]にある transparency 用の 13 枚の試験画像（32×32 画素）から 3 枚の試験画像を使用した（**表３**と**図９**を参照）．

この 3 枚の試験画像に対する圧縮率と PSNR の関係を**図１０**に示す．アルファなしの RGB24bpp に比べて，アルファ付きの RGBA32bpp は圧縮率対 PSNR の性能が悪い．つまり，アルファチャネルを加えることにより，3 色成分の圧縮よりも効率が悪くなることを意味する．しかし，アルファなしの RGB24bpp に

表３　アルファの PNG 試験画像[12]

RGB24bpp	tp0n2c08.png	not transparent
RGBA32bpp	tbrn2c08.png	transparent + red background
RGBA64bpp	tbgn2c16.png	transparent + green background

tp0n2c08.png　(a)　RGB24bpp
tbrn2c08.png　(b) RGBA32bpp
tbgn2c16.png　(c) RGBA64bpp

図９　アルファの PNG 試験画像（32×32 画素）

比べて，アルファ付きの RGBA64bpp の圧縮率対 PSNR の性能は非常に良い．アルファなしの RGBA48 bpp の試験画像がないために，正確な比較はできていないが，各色成分が 16bpp の場合は，アルファを加えることによる圧縮率対 PSNR の性能への悪影響よりも，16bpp の各色成分を使用することによる圧縮率対 PSNR の性能への寄与度が全体の圧縮率対 PSNR の性能を良くしていると推察される．

表４に PNG ファイルと JPEG XR による圧縮率の比較を示す．同表中の圧縮率は，左欄から順に，PNG ファイルのサイズから算出した圧縮率，JPEG XR によるロスレス圧縮の圧縮率，および PSNR が 35dB 近傍での JPEG XR による圧縮率である．PNG ファイルの圧縮率は，アルファチャネルを一つの色成分とした画像サイズに対する PNG ファイルのサイズの割合として算出した．

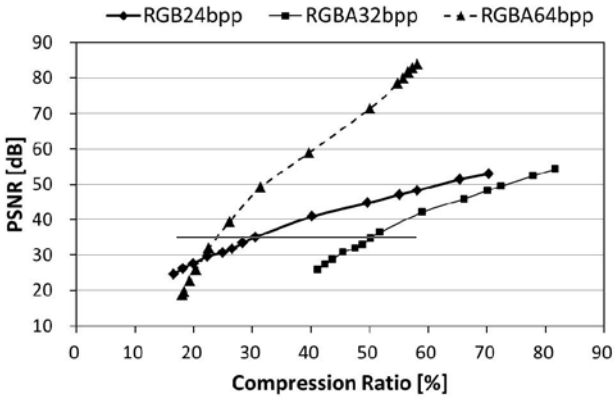


図１０　PNG 試験画像の圧縮率と PSNR

表４　PNG ファイルと JPEG XR による圧縮率の比較

	PNG ファイル	JPEG XR			
	ロスレス圧縮			ロッシー圧縮	
試験画像	[Bytes]	圧縮率 [%]			PSNR[dB]
RGB24bpp	1311	42.7	74.2	30.6	35.1
RGBA32bpp	1347	32.9	74.8	50.8	35.1
RGBA64bpp	1994	24.3	69.3	24.4	35.5

ロスレス圧縮の場合を比較すると，PNG ファイルが bpp の増加につれて，42.7%, 32.9%, 24.3%と圧縮率が良くなるのに対し，JPEG XR によるロスレスの圧縮率は約 7 割と非常に悪い．これは，JPEG XR がアルファチャネルを独立したチャネルとしてのみ扱っているためではないかと推察される．

次に，ロスレス圧縮の PNG と，PSNR を 35dB 近傍とした JPEG XR によるロッシー圧縮の場合を比較すると, JPEG XR の圧縮率は, アルファなしの RGB24 bpp では約 12%低くなるが, アルファ付きの RGBA32 bpp では逆に約 18%高くなり, アルファ付きの RGBA 64bpp ではほぼ同じ圧縮率となる（**図１０**中の 35dB のラインも参照）.

以上より，使用した試験画像の場合，画像品質を劣化させることにより JPEG XR でもアルファ付き画像の圧縮率を低くすることはできるが，高画像品質のアルファ付き画像を使用する際は PNG を使用した方が良いことになる．自然画像については今後の検討が必要である．

5　むすび

以上，試作した JPEG XR ビューアの第 1 版と第 2 版の概要と，第 2 版による JPEG XR の評価について報告した. JPEG XR ビューア第 2 版で, BitmapSource クラスを使用することにより高速に PSNR を算出することが可能となり, 入力原画像ファイルとして PNG をサポートすることにより Gray16bpp，RGB48bpp および RGBA64bpp を評価することが可能となった. また, JPEG との比較評価の結果, JPEG よりも JPEG XR が優位であることなどが確認できた.

しかし，試行的な評価用ビューアとして，あるいは試行的な評価の結果，本文でも述べたようにいくつかの不可解な点や未達成の点が残った．最後に，これらを今後の課題として列記する．

- サブサンプリング（色差成分の間引き）の効果
- Gray16bpp の Windows 7 (64 ビット版と 32 ビット版)での誤動作
- Gray16bpp と RGB48bpp の悪いロスレス圧縮率
- アルファ付き RGBA32bpp, RGBA64bpp の圧縮率
- アルファに関する三つのプロパティの評価
- Tile Slices と Overlap の効果

謝辞

本研究は平成 22 年度校内研究助成金（一般研究）を受けて行われたことを記して謝意を表します.

参考文献

1) ISO/IEC FCD 29199-2 (JPEG XR image coding - Specification), Sep. 2008.

2) F. Dufaux. et. al, “The JPEG XR Image Coding Standard”, IEEE Signal Processing Magazine, vol.26, num.6, pp.195-199, 204-204, Nov. 2009.

3) HD Photo - Photographic Still Image File Format - Feature Specification, Version 1.0, Microsoft, 2006.

4) HD Photo - Photographic Still Image File Format - Bitstream Specification, Version 1.0, Microsoft, 2006.

5) HD Photo - Photographic Still Image File Format - Device Porting Kit Specification, Version 1.0, Micro soft, 2006.

6) 馬場園智貴, 平田祐也, “JPEG XR の機能調査と評価”, 卒業研究論文，鹿児島高専情報工学科，2009.

7) 池田あずさ, 上小川愛倫, 吉田沙央里, “WIC (Windows Imaging Component)を用いた JPEG XR の評価”，卒業研究論文，鹿児島高専情報工学科，2010.

8) 加治佐清光，“ウェーブレット画像変換の学習ソフトの試作”，鹿児島工業高等専門学校研究報告，第 39 号, pp.1-10, 2004.

9) 加治佐清光，“学習ソフト：2 値算術符号化シミュレータ BACS の試作”，鹿児島工業高等専門学校研究報告, 第 42 号, pp.9-16, 2007.

10) Visual Studio 2008 / 2010 用 MSDN ライブラリ, Microsoft，2008 / 2010.

11) Sachin Garg, The New Test Images – Image Compression Benchmark, http://www.imagecompression.info/test_images/ (Mar. 2011)

12) W. Schaik, The official test-suit for PNG, http://www.schaik.com/pngsuite/ (Mar. 2011)