初等関数に関するディジタル回路実装方式の基礎と応用

客野　一樹

システム情報工学研究科

筑波大学

2011年3月

目 次

図 目 次

表 目 次

第1章　序論

1.1　ディジタル回路における初等関数

現在，ディジタル回路において，ハードウェア記述言語による設計効率の向上と，プロセスの微細化に伴う集積度の向上が進んでいる．従来，ディジタル回路は回路図から設計する必要があったが、ハードウェア記述言語を用いれば，ソースコードを論理合成することでネットリストを生成可能なため，より抽象化されたレベルで設計が可能となる．加えて，集積度の向上により，従来と同じダイ・サイズに対して，より大規模な回路の実装が可能となるため，ディジタル回路の高機能化と，設計内容の高度化が加速している．特に，音声や 3D グラフィックス等のマルチメディアデータを扱う回路においては，圧縮率および描画性能の向上を目的としてアルゴリズムの高度化が加速しており，それに伴い，種々の初等関数の利用が拡大している．

例えば，ディジタル音響信号圧縮回路においては，圧縮率の向上を目的として，初等関数である正弦および余弦関数を基底とする周波数変換が広く用いられるようになってきており，MP3，AAC，あるいは OggVorbis は，窓単位に分割した入力信号を修正離散コサイン変換 (MDCT)[1]-[6] によって周波数変換し，周波数領域で符号化することで効率的な符号化を実現している．修正離散コサイン変換では直交性の前提として Princen-Bradley 条件 [1] を満たす窓関数が用いられるが，これも初等関数の合成関数である．窓幅を M とした場合，MDCT[1] の計算量は $O(M^2)$ であるが，DCT-IV や FFT の係数に変換することで，これを $O(M \log M)$ とする高速 MDCT が多数提案されている．例えば，文献 [2] は，MDCT の係数を FFT の係数に変換する事で高速化する方式を提案しており，文献 [3] および [4] は，FFT の係数への変換方式を改良することで，必要とする関数値の範囲を制限する方式を提案している．また，文献 [5] は，M が比較的小さい場合に漸化式を用いて実装する方式を，文献 [6] は，MDCT の対称性を利用し，ワークスペースを削減する方式をそれぞれ提案している．しかし，一般に，修正離散コサイン変換に用いられる正弦・余弦関数，および窓関数は，離散化・標本化されて関数テーブルとして ROM に格納されているため，窓幅の拡大に伴い，回路規模が大きくなっているという問題がある．

また，3D グラフィックス回路においては，集積度の向上に伴う演算能力の向上に比して，回路基板の実装に制約される転送速度の向上が追いついていないという問題があり，ポリゴンデータの圧縮への要求が高まっている．ポリゴンデータの圧縮においては，多数の歪み幾何圧縮 [21]-[26] が提案されている．文献 [21] および [22] は，頂点情報を整数精度へ量子化することで情報量を削減する方式を提案している．また，文献 [23] および [24] は，隣接三角形の情報を用いた平行四辺形予測による予測値との残差を符号化することで，圧縮効率を高める

方式を提案している．しかし，同方式は，データを大域座標系で一様に処理するため，予測残差の冗長性が 3 自由度に分散し，圧縮効率の劣化が生じていると考えられる．これに対して，文献 [25] および [26] は，隣接三角形の情報から基底系を構成し，残差を射影して圧縮効率を高める方式を提案している．特に，隣接する三角形からグラム-シュミット直交化によって生成された基底系による直交変換符号化 [26] によって効率的な符号化が可能であることが知られているが，グラム-シュミット直交化の実装にはベクトルノルムの逆数が必要であり，ここでも初等関数として平方根の逆数が利用されている．しかし，平方根の逆数はニュートン法などの反復法で実装され，精度に応じて多くの反復を必要とするため，ポリゴンデータ圧縮のような高いスループットが要求される応用に対して，十分な演算性能が得られていないという問題がある．

1.2 本論文の構成

本研究では，これらの問題に対し，正弦関数，余弦関数，窓関数，および平方根の逆数をより小さな回路規模で実装可能な方式を提案し，その修正離散コサイン変換およびグラム-シュミット直交化への応用について検討する．

第 2 章では，要素技術として，各初等関数とその一般的な実装を説明し，問題を明らかにする．一般に，正弦関数および余弦関数の実装においては，以下の 2 種類の方式が用いられている．

- 関数値をサンプリングし ROM テーブルに格納する方式
- 巡回型正弦波発生機構 [7]-[13] によって関数値を漸化式で計算する方式

巡回型正弦波発生機構はいずれも文献 [7] の漸化式に基づいており，文献 [8] は，正弦波の周期性と対称性が完全に成立する乗数 $\cos\theta$ を選択する方式を検討し，文献 [9] および [10] は，補助情報をテーブルとして与えることで低周波における精度改善法を提案している．文献 [11] および [12] は，乗数 $\cos\theta$ を十分小さく設定することで長周期の正弦波を，文献 [13] は，文献 [7] を加法定理で変形することで，複素正弦波を同時に生成する方式を提案している．しかし，ROM テーブルは精度に比例して回路規模が増大するという問題があり，巡回型正弦波発生機構はその演算に乗算を用いているため回路規模が大きいという問題がある．また，平方根の逆数演算機構にはニュートン法が用いられるが，浮動小数精度の乗算を必要としているため回路規模が大きく，精度に応じて多くの実行サイクル数を必要とするという問題がある．

第 3 章では，第 2 章で明らかにしたこれらの問題に対し，より小さな回路規模で実装可能な方式を提案する．まず，巡回型正弦波発生機構における乗算をビットシフトに置換することで，従来方式の $\frac{3}{4}$ 程度の回路規模で実装可能な方式を提案する．また，平方根の逆数を，仮数部に対する関数テーブルと，指数部に対するビットシフトで置換することで，反復が不要で，中から低の演算精度で従来方式の $\frac{1}{3}$ 程度の回路規模で実装可能な方式を提案する．提案方式の有効性は，各初等関数を FPGA 上に実装し，精度に対する回路規模および実行サイクル数の観点から従来方式と比較評価する．精度は，ライブラリ関数を真値とし，回路の出力

との SNR，もしくは平均二乗誤差で計測する．回路規模は FPGA のロジックの基本単位である SLICE 総数で評価する．SLICE 総数が少ないほど，小さな回路規模で実装が可能である．SLICE 総数は，回路を HDL で記述した後，論理合成することによって算出される．実行サイクル数は，アルゴリズムに依存して決まり，1 つの関数値を計算するために必要なクロックサイクル数として定義される．

第 4 章では，第 3 章で提案したビットシフトによる複素正弦波発生機構を修正離散コサイン変換に最適な形に改良し適用することで，回転子の回路規模を従来方式の $\frac{3}{4}$ 程度に削減が可能であることを示し，その有効性を提示する．また，窓関数に対し 2 次差分符号化を適用することで，窓関数の回路規模を従来方式の $\frac{1}{4}$ 程度に削減が可能であることを示す．その結果，修正離散コサイン変換回路全体の回路規模を従来方式の $\frac{3}{4}$ 程度に，逆変換回路全体の回路規模を $\frac{1}{2}$ から $\frac{3}{4}$ 程度に削減が可能であることを示す．

第 5 章では，第 3 章で提案した関数テーブルとビットシフトによる平方根の逆数演算機構をグラム-シュミット直交化に適用することで，基底の算出に必要な実行サイクル数を中から低の演算精度で従来方式の $\frac{1}{2}$ から $\frac{5}{8}$ 程度まで削減が可能であることを示し，その有効性を提示する．

最後に，第 6 章で本研究の論点をまとめ、結論を述べる．

第2章　要素技術

2.1　音声のディジタル表現

音声は音圧の時間変化として表現される．具体的に，時間を t としたとき，時間の関数 $x(t)$ として表現され，これを時間波形と呼ぶ．(図 2.1 参照) また，時間波形に対し，周波数変換を適用することで，周波数領域で表現することも可能である．具体的に，周波数を f としたとき，周波数の関数 $X(f)$ として表現され，これを周波数成分と呼ぶ．(図 2.2 参照)

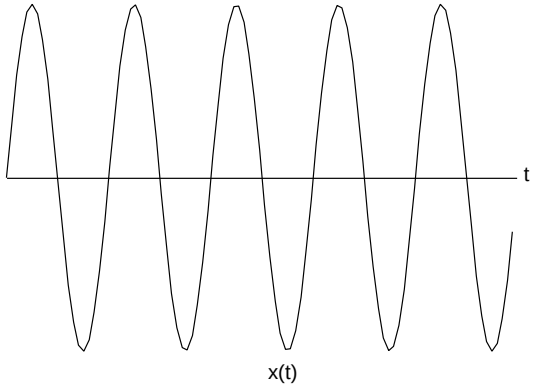

図 2.1: 時間波形 $x(t)$

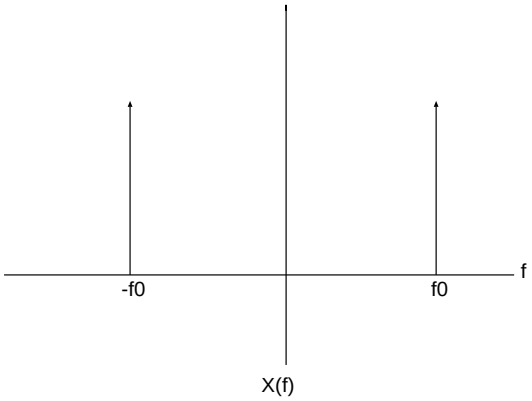

図 2.2: 周波数成分 $X(f)$

これらの関数値をディジタル媒体で扱う場合，t，f，$x(t)$，および $X(f)$ をそれぞれ離散化する必要がある．離散化した時間を m，離散化した周波数を n としたとき，離散化した時間波形を x_m，離散化した周波数成分を X_n と表現する．例えば，CD では，$x(t)$ を 16[bit] で，t を 44.1[kHz] で離散化している．

2.2 初等関数

本研究では，正弦関数，余弦関数，窓関数，および平方根の逆数について扱う．窓関数は，周波数変換の参照範囲の制限を行うために時間波形に乗ずる合成関数である．特に，修正離散コサイン変換では窓関数が直交性の前提となっており，離散化した時間 m の関数として窓関数を $w_m \in R^1$ と表現したとき，Princen-Bradley 条件 [1]

$$w_m^2 + w_{\frac{M}{2}+m}^2 = 1 \quad (m = 0, 1, ..., \frac{M}{2} - 1)$$

を満たす窓関数として MP3 では Sine 窓

$$w_m = \sin(\frac{\pi(m + 0.5)}{M}) \quad (m = 0, 1, ..., M - 1)$$

が，OggVorbis では Vorbis 窓

$$w_m = \sin(\frac{\pi}{2}\sin^2(\frac{\pi(m + 0.5)}{M})) \quad (m = 0, 1, ..., M - 1)$$

が，AAC では KBD 窓

$$w_{M-1-m} = w_m, \quad (m = 0, 1, ..., \frac{M}{2} - 1)$$

$$w_m = \sqrt{\frac{\Sigma_{j=0}^{m} w'_j}{\Sigma_{j=0}^{\frac{M}{2}} w'_j}}, \quad (m = 0, 1, ..., \frac{M}{2} - 1)$$

$$w'_m = I_0(\pi\alpha\sqrt{1.0 - (\frac{4m}{M} - 1.0)^2}) \quad (m = 0, 1, ..., \frac{M}{2})$$

がそれぞれ用いられている．ただし，M は窓幅である．尚，I_0 は 0 次ベッセル関数であり，α は窓関数の形状を決めるパラメータである．本論文では，$\alpha = 4.0$ について扱う．一般的な回路実装において，これらの関数値は一定間隔でサンプリングされ，q[bit] 固定小数精度で ROM に格納される．ただし，平方根の逆数については，必要とする ROM サイズが膨大になることから，後述する反復法によって実装される．

2.3 巡回型正弦波発生機構

離散化した時間を m としたとき，正弦関数値 $x_m \in R^1$ を再帰的に計算する方式として，種々の巡回型正弦波発生機構 [7]-[13] が検討されており，そのいずれも文献 [7] の漸化式

$$\begin{aligned} x_{m+1} &= 2\cos\theta x_m - x_{m-1}, \\ (x_1, x_0) &= (\sin\theta, 0) \end{aligned} \tag{2.1}$$

の一般解

$$x_m = \sin(m\theta)$$

に基づいている．(付録 A 参照) 具体的に，サンプリング周波数 s[Hz] によって離散化された基本周波数 f[Hz] の正弦関数値

$$\{\sin(\frac{2\pi f}{s}m) : m = 0, 1, 2, ...\}$$

は，初期値 $\sin\theta$ および乗数 $\cos\theta$ をそれぞれ q[bit] および d[bit] 固定小数精度で近似し，漸化式

$$\begin{aligned} x_{m+1} &= [2^d \cos(\frac{2\pi f}{s})]\frac{x_m}{2^{d-1}} - x_{m-1}, \\ x_1 &= [2^q \sin\theta] = [2^q \sin(\frac{2\pi f}{s})] \end{aligned} \tag{2.2}$$

を再帰的に反復することで得られる．$[\cdot]$ はガウス記号を示す．以降，本方式を漸化式方式と呼ぶ．文献 [7] を受けて，文献 [8] は，正弦波の周期性と対称性が完全に成立する乗数 $\cos\theta$ を選択する方式を検討し，文献 [9] および [10] は，補助情報をテーブルとして与えることで低周波における精度改善法を提案している．いずれの文献も，近似精度向上を目的とし，(2.2) に拘束条件を加える，もしくは別の機構を付加した方式であるため，回路規模においては (2.2) が最も小さくなることに注意する．

他方，文献 [11]，[12] は，乗数 $\cos\theta$ を十分小さく設定することで，長周期正弦波の発生機構を提案している．即ち，(2.1) において $(x_1, x_0) = (-\alpha, 0)$ と設定し，$\cos\theta \ll 1$ を仮定することで，

$$\begin{aligned} \phi &= 2\pi - 4\theta \ll 1, \\ \cos(\frac{1}{4}\phi) &= 1 + O(\phi^2) \simeq 1 \end{aligned}$$

とし，４反復毎の解

$$x_{4m} = \frac{\alpha}{\cos(\frac{1}{4}\phi)}\sin(m\phi) \simeq \alpha\sin(m\phi) \tag{2.3}$$

を用いる．また，文献 [13] は，(2.1) に対して加法定理を用いて，

$$\begin{aligned} x_{m+1} &= \frac{\alpha}{\sin\theta}\sin(m\theta + \theta) \\ &= \cos\theta\frac{\alpha}{\sin\theta}\sin(m\theta) + \alpha\cos(m\theta) \\ &= \cos\theta x_m + y_m \end{aligned}$$

を導き，初期値 $(x_1, x_0) = (\alpha, 0)$ による漸化式

$$\begin{aligned} x_{m+1} &= 2\cos\theta x_m - x_{m-1}, \\ y_m &= \cos\theta x_m - x_{m-1} \end{aligned} \tag{2.4}$$

を反復することで，複素正弦波

$$\begin{aligned} x_{m+1} &= \frac{\alpha}{\sin\theta}\sin((m+1)\theta), \\ y_m &= \alpha\cos(m\theta) \end{aligned} \tag{2.5}$$

を同時に生成する方式を提案している．上記の方式はいずれも乗算器を必要としているため，精度に依存し，回路規模が大きくなることに注意する．

2.4 ニュートン法

平方根の逆数の計算方式として，ニュートン法が広く用いられている．ニュートン法では，反復回数を i としたとき，漸化式

$$x_{i+1} = \frac{1}{2}x_i(3 - ax_i^2) : i = 0, 1, 2, ...$$

を初期値 $x_0 = 2^{-\frac{1}{2}e}$ で再帰的に反復することで，一般解

$$x_i = \frac{1}{\sqrt{a}}$$

を計算する．e は浮動小数 a の指数部である．上記の方式は浮動小数精度での加算および乗算を必要としているため回路規模が大きく，精度に依存し多くの反復が必要となることに注意する．

第3章　提案方式

3.1　正弦波発生機構

3.1.1　ビットシフトによる巡回型正弦波発生機構

本研究では，2^k によって表現できる乗数 $\cos\theta$ を用いることで，乗算器をビットシフトに置換し，回路規模を減ずる実装方式を検討する．具体的には，大きな $k > 0 \in Z^1$ に対して，

(A)　$\cos\theta = 1 - \frac{1}{2^k}, 0 < \theta \ll 1$
(B)　$\cos\theta = \sin\frac{\phi}{4} = \frac{1}{2^k} \ll 1, 0 < \phi \ll 1$

という二つの場合を考える．(A) は，方式 [7] と同様に $x_1 = \sin\theta$ として解 $x_m = \sin(m\theta)$ を，(B) は，方式 [12] と同様に $x_1 = -1$ として解 $x_{4m} = \sin(m\phi)$ を利用する．両者とも回転角 θ および ϕ が十分に小さい場合，密に出力される正弦関数値を適切な間隔 σ でサンプリングすることで所望の正弦関数値が実現できる．

ここでシフトレジスタの精度を決める k と，k に依存して決定される出力正弦関数値の回転角 θ_k および ϕ_k の関係を調べてみる．$\cos\theta_k$ および $\sin(\frac{1}{4}\phi_k)$ をテイラー展開すると，

(A)　$\cos\theta_k = 1 + O(\theta_k^2)$
(B)　$\sin(\frac{1}{4}\phi_k) = \frac{1}{4}\phi_k + O(\phi_k^3)$

となるため，θ_k と $\theta_{k+\delta}$ および ϕ_k と $\phi_{k+\delta}$ の関係は，

(A)　$\theta_k \approx \theta_{k+\delta}$
(B)　$\phi_k : \phi_{k+\delta} \approx 1 : 2^{-\delta}$

となることに注意する．仮に，シフトレジスタの精度を δ[bit] 上げた場合，(B) においては回転角が $2^{-\delta}$ 倍されて出力の精度が向上するが，(A) において精度は改善されない．よって，本研究では (B) のみを改良方式として検討を進める．

改良方式では，まず k に対応する 1 周期分の反復回数 M を求め，f に依存した実数精度のサンプリング間隔 $\sigma = \frac{Mf}{s}$ を定める．(2.1) を実現する場合，初期値を d[bit]，σ を b[bit] 固定

小数精度で近似し，漸化式

$$\begin{aligned} x_{m+1} &= x_m \frac{1}{2^k} - x_{m-1}, \\ (x_1, x_0) &= (2^d, 0), \\ c_{m+1} &= c_m + \frac{[2^b\sigma] - 2^b[\sigma]}{4}, \\ c_0 &= 0 \end{aligned}$$

を再帰的に反復し，$4[\sigma]$ 反復毎に出力をサンプリングする．ただし，サンプリング間隔の誤差をリセットするため，$c_{m+1} > 2^{b-1}$ の場合のみ，c_{m+1} から 2^b を減じた後，値を出力することなく，x_{m+1}および c_{m+1} を 4 反復分更新する．以降，本方式を方式 [14] と呼ぶ.

方式 [14] は，4 反復毎の値をさらに σ でサンプリングするため，出力の精度を高くする程，σ が大きくなり，実行サイクル数が増加する欠点を持っている．この問題を解決するため，文献 [13] で提案された複素正弦波 $e^{i\theta}$ の発生機構に注目する．ここで，(2.5) を初期値 $(x_1, x_0) = (\alpha, 0)$ で反復した数列

$$\begin{aligned} x_m &: 0 \to \alpha \to \frac{\alpha\sin(2\theta)}{\sin(\theta)} \to \frac{\alpha\sin(3\theta)}{\sin(\theta)} \to ... \\ y_m &: \alpha \to \alpha\cos(\theta) \to \alpha\cos(2\theta) \to \alpha\cos(3\theta) \to ... \end{aligned}$$

において，$\cos\theta \ll 1$ を仮定し，θ を ϕ で置換すると，

$$\begin{aligned} x_m &: 0 \to \alpha \to \alpha\sin(\frac{\phi}{2}) \to -\alpha\cos(\frac{3\phi}{4}) \to ... \\ y_m &: \alpha \to \alpha\sin(\frac{\phi}{4}) \to -\alpha\cos(\frac{\phi}{2}) \to -\alpha\sin(\frac{3\phi}{4}) \to ... \end{aligned}$$

となることに注意する．上記の数列において，x_m および y_m の符号を適切に入れ換え，交互にサンプリングすることで，回転角 $\frac{\phi}{4}$ の正弦および余弦関数値が 1 反復毎に得られることが分かる.

したがって，$\cos\theta \ll 1$ として，(2.5) から，

$$\begin{aligned} x_{2m} &= (-1)^{m+1}\alpha\sin(\frac{\phi}{2}m), \\ y_{2m} &= (-1)^m\alpha\cos(\frac{\phi}{2}m), \\ x_{2m+1} &= (-1)^m\alpha\cos(\frac{\phi}{4}(2m+1)), \\ y_{2m+1} &= (-1)^m\alpha\sin(\frac{\phi}{4}(2m+1)) \end{aligned}$$

を導くことで，正弦および余弦関数値

$$Re(\alpha e^{i\frac{\phi}{4}m'}) = \begin{cases} (-1)^m y_{2m} & (m' = 2m) \\ (-1)^m x_{2m+1} & (m' = 2m+1) \end{cases}$$

$$Im(\alpha e^{i\frac{\phi}{4}m'}) = \begin{cases} (-1)^{m+1}x_{2m} & (m' = 2m) \\ (-1)^{m}y_{2m+1} & (m' = 2m+1) \end{cases}$$

が計算できる．

回路実装においては (2.4) の $2\cos\theta$ を 2^{-k} で置換する．具体的には，初期値を d[bit]，σ を b[bit] 固定小数精度で近似し，漸化式

$$\begin{aligned} x_{m+1} &= x_m \frac{1}{2^k} - x_{m-1}, \\ y_m &= x_m \frac{1}{2^{k-1}} - x_{m-1}, \\ (x_1, x_0) &= (2^d, 0), \\ c_{m+1} &= c_m + ([2^b\sigma] - 2^b[\sigma]), \\ c_0 &= 0 \end{aligned}$$

を再帰的に反復しつつ，x_m および y_m を適切に符号反転し，サンプリングする．また，サンプリング間隔のリセットは方式 [14] に準ずる．以降，本方式を方式 [15] と呼ぶ．

3.1.2 正弦波発生機構の実装方式

漸化式方式，方式 [14]，および方式 [15] による正弦波発生機構を VHDL で記述，FPGA 上へ実装し，近似精度に対する SLICE 総数と実行サイクル数を比較する．ただし，サンプリング周波数 s は 44.1[kHz]，生成する正弦波の周波数 f は 1[kHz] で設計した．SLICE 総数は VHDL のソースコードを論理合成することによって計測する．SLICE 総数が少ないほど，小さな回路規模で実装が可能である．実行サイクル数は，1 つの関数値を計算するために必要なクロックサイクル数である．シミュレータには ModelSimSE6.1c，論理合成には XilinksISE8.1i，FPGA には Spartan3 を使用した．近似精度は，ライブラリ関数を $L = 225000$ サンプルで用いた倍精度の正弦波 $s_m \in R^1$ を真値とし，回路の出力 $x_m \in R^1$ との SNR

$$\mathrm{SNR} = 10\log_{10}\frac{\sum_{m=0}^{L-1} s_m^2}{\sum_{m=0}^{L-1}(s_m - x_m)^2}\ [\mathrm{dB}]$$

で計測した．SNR の値が大きいほど波形ひずみが小さく，より高い品質の関数値が得られる．各方式の仕様を表 3.1，表 3.2，および表 3.3 に示す．尚，全ての方式において，反復に伴い誤差が累積するため，2π に対応する位置で初期化を行う．

3.1.3 正弦波発生機構の比較評価

各方式の実装結果を表 3.4，表 3.5，および表 3.6 に示す．

表 3.1: 正弦波における漸化式方式の仕様

d	q	$2\cos\theta$	x_1
11	13	8109	290
12	13	8109	581
13	13	8109	1163
15	13	8109	4652

表 3.2: 正弦波における方式 [14] の仕様

d	k	M	σ	b
11	8	3264	18.5034	14
14	10	12940	73.3560	14
15	11	25884	146.7347	14
19	12	51484	291.8594	14

表 3.3: 正弦波における方式 [15] の仕様

d	k	M	σ	b
11	6	807	18.5034	14
14	8	3223	73.3560	14
15	9	6441	146.7347	14
19	10	12871	291.8594	14

近似精度に対する回路規模　SNR に対する SLICE 総数を図 3.1 に示す．方式 [15] は，方式 [14] にレジスタ，符号反転器，およびセレクタを加えるため，SLICE 総数が 40 程度増加する．しかし，方式 [14] および方式 [15] は，あらゆる精度において，漸化式方式よりも少ない SLICE 総数で実現できる．

近似精度に対する実行サイクル数　各方式の SNR に対する実行サイクル数を図 3.2 に示す．漸化式方式は，1 サンプルを最小の 1[CLK] で出力する．他方，方式 [14] では，4 反復毎に得られる値をさらに σ でサンプリングするため，約 4σ の実行サイクル数を必要とする．よって，精度が高くなるほど σ が大きくなり，実行サイクル数が増加する欠点を持っている．しかし，72[dB] 程度の高精度出力においても，実行サイクル数は 1164 程度であり，クロック周波数 166[MHz] の許容限界 3764[CLK/sample] の半分以下で処理できる．このため，正弦波を発生する応用においては，提案方式は回路規模の観点から実用上の利点があると考えられる．方式 [15] は，55[dB] 以上で，方式 [14] と同等の精度が $\frac{1}{4}$ の実行サイクル数で実現できる．したがって，方式 [15] は，70[dB] 程度の高精度な正弦波であっても 13[MHz] 以上のクロック周波数で十分に実装できる．

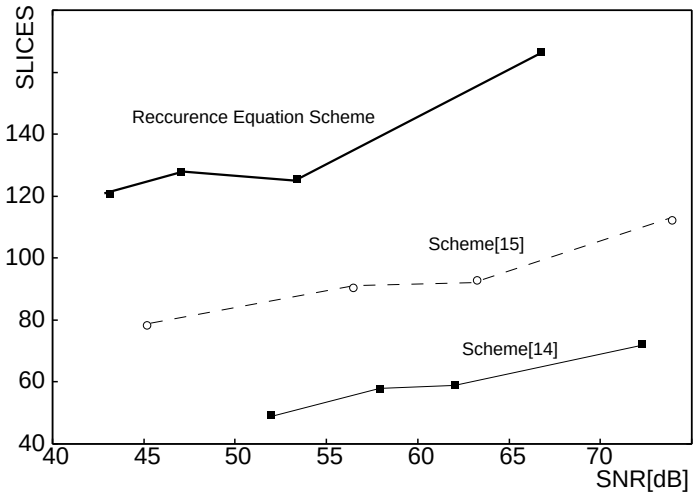

図 3.1: 正弦波における SNR に対する SLICE 総数 (縦軸：SLICE 総数，横軸：SNR[dB]，太線：漸化式方式，実線：方式 [14]，破線：方式 [15])

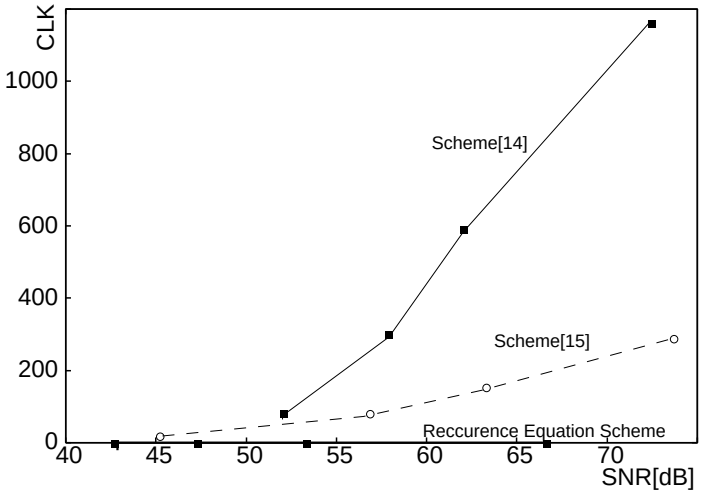

図 3.2: 正弦波における SNR に対する演算時間 (縦軸：CLK，横軸：SNR[dB]，太線：漸化式方式，実線：方式 [14]，破線：方式 [15])

表 3.4: 正弦波における漸化式方式の実装結果 (ROM 含む)

d	SNR[dB]	SLICES	FF	LUT	実行サイクル数
11	42.85	121	52	205	1
12	47.15	128	56	220	1
13	53.26	125	51	215	1
15	66.65	166	71	287	1

表 3.5: 正弦波における方式 [14] の実装結果 (ROM 含む)

k	SNR[dB]	SLICES	FF	LUT	実行サイクル数
8	52.02	49	53	49	74
10	57.91	58	62	61	292
11	62.03	59	64	59	584
12	72.37	72	74	61	1164

表 3.6: 正弦波における方式 [15] の実装結果 (ROM 含む)

k	SNR[dB]	SLICES	FF	LUT	実行サイクル数
6	45.38	79	86	116	18
8	56.51	91	97	128	73
9	63.06	92	101	134	146
10	73.80	113	121	154	291

3.1.4 複素正弦波発生機構の実装方式

正弦波発生機構と同一の実験条件で複素正弦波に関して方式 [13] と方式 [15] の比較を行う．方式 [13] は，(2.4) の乗数 $\cos\theta$ および $\sin\theta$ をそれぞれ q[bit] 固定小数精度で近似し，初期値 $(x_1, x_0) = (2^d, 0)$ から，乗算を含む漸化式

$$\begin{aligned} x_{m+1} &= [2^q \cos(\frac{2\pi f}{s})]\frac{x_m}{2^{q-1}} - x_{m-1}, \\ y_m &= [2^q \cos(\frac{2\pi f}{s})]\frac{x_m}{2^{q-2}} - x_{m-1} \end{aligned}$$

によって実装する．また，x_{m+1}，y_m，$\sin\theta x_m$ を順次計算することで乗算器を共有し回路規模を削減する．近似精度は，ライブラリ関数を $L = 225000$ サンプルで用いた倍精度の複素正弦波 $s_m \in C^1$ を真値とし，回路出力の実部との SNR，および虚部との SNR の平均で計測した．尚，各方式の仕様は表 3.7 および表 3.8 のように設定した．

3.1.5 複素正弦波発生機構の比較評価

各方式の実装結果を表 3.9 および表 3.10 に示す．

近似精度に対する回路規模　各方式の SNR に対する SLICE 総数を図 3.3 に示す．方式 [15] は乗算器が不要なため，いずれの精度においても，方式 [13] の $\frac{3}{4}$ 程度の SLICE 総数で実現できる．

表 3.7: 複素正弦波における漸化式方式 [13] の仕様

d	q	$2\cos\theta$	$\sin\theta$
9	13	8109	582
10	13	8109	582
12	13	8109	582
14	13	8109	582

表 3.8: 複素正弦波における方式 [15] の仕様

d	k	M	σ	b
7	13	1611	36.5306	14
8	14	3223	73.0839	14
9	15	6442	146.0771	14
11	18	25751	583.9229	14

近似精度に対する実行サイクル数　各方式の SNR に対する実行サイクル数を図 3.4 に示す．方式 [13] は，乗算器を共有しているため，1 サンプルを 3[CLK] で出力する．他方，方式 [15] では，方式 [13] と同様に精度が高くなるほど実行サイクル数が増加する性質を持つが，67[dB] 程度の高精度出力においても，実行サイクル数は 583 程度であり，26[MHz] 以上のクロック周波数で十分である．このため，複素正弦波を発生する応用においても，方式 [15] は回路規模の観点から実用上の利点があると考えられる．

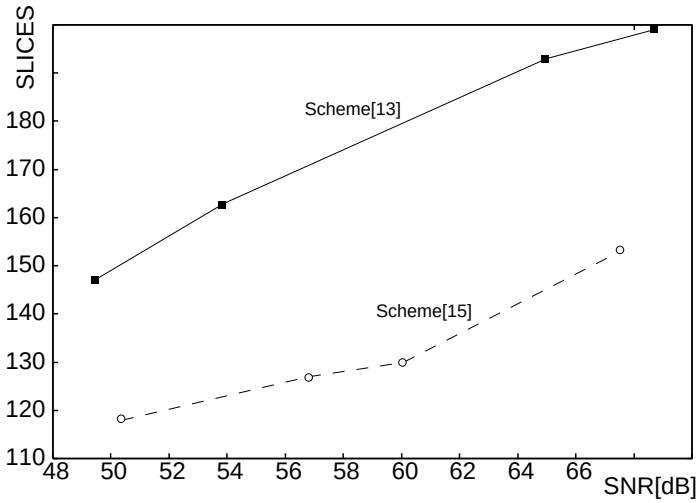

図 3.3: 複素正弦波における SNR に対する SLICE 総数 (縦軸：SLICE 総数，横軸：SNR[dB], 太線：方式 [13]，破線：方式 [15])

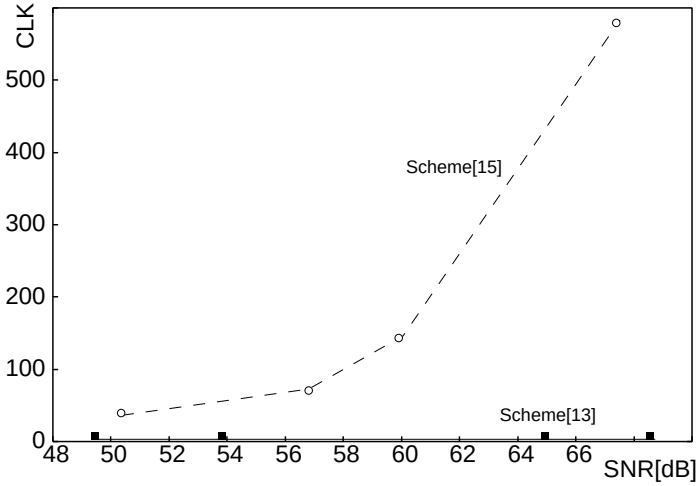

図 3.4: 複素正弦波における SNR に対する演算時間 (縦軸：CLK，横軸：SNR[dB]，太線：方式 [13]，破線：方式 [15])

表 3.9: 複素正弦波における漸化式方式 [13] の実装結果 (ROM 含む)

d	SNR[dB]	SLICES	FF	LUT	実行サイクル数
9	49.44	147	101	252	3
10	53.90	163	109	269	3
12	65.00	193	133	334	3
14	68.73	199	135	334	3

表 3.10: 複素正弦波における方式 [15] の実装結果 (ROM 含む)

k	SNR[dB]	SLICES	FF	LUT	実行サイクル数
7	50.38	118	107	138	36
8	56.81	127	114	147	73
9	60.07	130	119	152	146
11	67.51	153	137	176	583

3.2 平方根の逆数演算機構

3.2.1 ビットシフトとテーブル参照による平方根の逆数の計算

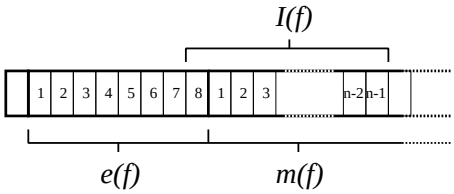

図 3.5: IEEE754 形式における浮動小数の形式

本研究では，実数を IEEE754 形式で扱う．(図 3.5 参照）仮数 $m \in [2^0, 2^1)$ および指数 $\epsilon \in Z$ で表現した 2 進小数 $f = 2^\epsilon \times m$ を考える．ここで，ϵ が奇数の場合は，$\epsilon = 2e + 1, M = 2m$, 偶数の場合は $\epsilon = 2e, M = m$ とすると，ノルムの逆数は,

$$\frac{1}{\sqrt{f}} \quad = \quad 2^{-e} \times \frac{1}{\sqrt{M}} \tag{3.1}$$

となる．ここで，$M \in [2^0, 2^2)$ の上位 n ビットを取り出し，整数 $\{0, 1, ..., 2^n - 1\}$ と対応づけると，関数台 $[2^0, 2^2)$ を 2^n 分割した関数テーブル

$$\{\frac{1}{\sqrt{M_i}} : i = 0, 1, ..., 2^n - 1\}$$

が得られる．よって，M の上位ビットによって関数値を引き，2^{-e} をビットシフトで実行することで平方根と除算を省くことができる．

3.2.2 ビットシフトとテーブル参照による平方根の逆数演算機構

以下，実数 f の指数部を $e(f)$，仮数部を $m(f)$ と表記し，図 3.5 のように指数部の下位 1 ビットから仮数部の上位 $n-1$ ビットまでの値を $I(f)$ とおく．また，関数テーブルの値を $\{T_i | i = 0, 1, .., 2^n - 1\}$ とする．尚，$e(f)$ は実際の指数値に 127 のバイアスが加えられていることに注意する．

$M \in [2^0, 2^2)$ は指数部の下位 1 ビットと仮数部から決定されるため,

$$\{T_{I(M)} = M^{-\frac{1}{2}} | I(M) = 0, 1, .., 2^n - 1\}$$

として関数テーブルを構築する．ただし，入力の丸め誤差を抑えるため，$m(M)$ の左から n ビット目は常に 1 とする．また，指数部はバイアスによって下位 1 ビットが反転するため,

$$e(M) = \begin{cases} 128 & (i < 2^{n-1}) \\ 127 & (i \geq 2^{n-1}) \end{cases}$$

とする．尚，$M^{-\frac{1}{2}} \in (2^{-1}, 2^0)$ から，T_i の指数部は 126 と一意に定まるため，テーブルにおいては仮数部の値のみを保持する．使用する ROM の構成を表 3.11 に示す.

3.1 において，e は ϵ の算術右シフトによって得られるので，$r \simeq \frac{1}{\sqrt{f}}$ の指数部と仮数部は，以下によって求められる．

$$\begin{aligned} e(r) &= 126 - \{(e(f) - 127) \gg 1\}, \\ m(r) &= m(T_{I(f)}) \end{aligned}$$

ただし，$\gg$ は算術右シフトとする．

表 3.11: ビットシフトとテーブル参照による平方根の逆数演算機構の ROM の構成

アドレスビット数	仮数ビット数	テーブルサイズ [byte]
3	23	23
5	23	92
7	23	368
9	23	1472
11	23	5888

3.2.3 平方根の逆数演算機構の比較評価

従来方式および提案方式について，VerilogHDL で記述，FPGA 上に実装し，近似精度に対する SLICE 総数と実行サイクル数を比較する．ターゲット FPGA として Spartan3 を用い，乗算器および ROM は共に LUT で実装した．論理合成は ISE7.1 で行った．ISE の設定を表 3.12 に示す．近似精度は，区間 $(0, 1]$ 上の乱数 $r \in R^1$ を $L = 1000$ 組使用し，ライブラリ関数によって計算した平方根の逆数

$$s_j = \frac{1}{\sqrt{r_j}} \in R^1 : j = 0, 1, ..., L-1$$

を真値とし，従来方式および提案方式で計算した値 x_j との平均二乗誤差

$$\mathrm{MSE} = \frac{1}{L} \sum_{j=0}^{L-1} (s_j - x_j)^2$$

で計測した．MSE の値が小さいほど，誤差が少なく，より高い品質の関数値が得られる．

従来方式のデータフローを図 3.6 に，使用した演算器の仕様を表 3.13 に示す．従来方式においては各項の並列化が困難であるため，浮動小数加算器と浮動小数乗算器を各 1 つ使用し，クロックサイクル毎に入力を切り替えることで，最小の回路規模となるように構成した．これより，ニュートン法 1 反復につき 5 サイクルで出力が得られる．

提案方式のデータフローを図 3.7 に示す. 提案方式では，入力した 32bit 浮動小数の仮数部の上位ビットを ROM テーブルに入力し，並行してアドレスシフトによって指数部を計算，ROM テーブルの出力仮数部と結合することで，平方根の逆数を 1 サイクル単位で計算する．

表 3.14 および表 3.15 に，従来方式および提案方式の実装結果を示す.

回路規模の評価　各方式の平均二乗誤差に対する回路規模を図 3.8 に示す．従来方式は，いずれの精度においても，反復回数が変化するだけであり，回路規模に変化はない．加算器と乗算器を単独で合成したものよりも，従来方式の SLICE 総数が小さくなるのは，最適化によって LUT の使用効率が向上したものと考えられる．対して，提案方式は，必要とされる精度に応じて回路規模が変化する．高精度においては，ROM サイズが増大し，従来方式よりも回路規模が大きくなるが，中から低の演算精度においては，提案方式は従来方式に比べ，$\frac{1}{3}$ 程度の回路規模で実装できる．

実行サイクル数の評価の評価　各方式の平均二乗誤差に対する実行サイクル数を図 3.9 に示す．提案方式は，従来方式に比べ，高精度で $\frac{1}{15}$ 以下，低精度で $\frac{1}{5}$ 以下の実行サイクル数で実装できる．そのため，3D ポリゴン圧縮等，高いスループットが要求される応用において提案方式は有効であると考えられる．

総合評価　提案方式は，中から低の演算精度において，従来方式の $\frac{1}{3}$ 程度の回路規模かつ $\frac{1}{5}$ 以下の実行サイクル数で実装できる．そのため，精度が要求される科学技術計算用途ではなく，コンピュータアミューズメント等，ある程度の歪が許容される応用に適用されるのが望ましいと考えられる．

表 3.12: ISE の設定

オプション	パラメータ
OptimizationGoal	Area
OptimizationEffort	Normal
ROM Extraction	None
Multiplier style	LUT

表 3.13: 浮動小数加算器および浮動小数乗算器の仕様

	入出力ビット数	SLICES	FF	LUT	実行サイクル数
加算器	32	949	122	1653	2
乗算器	32	422	93	771	1

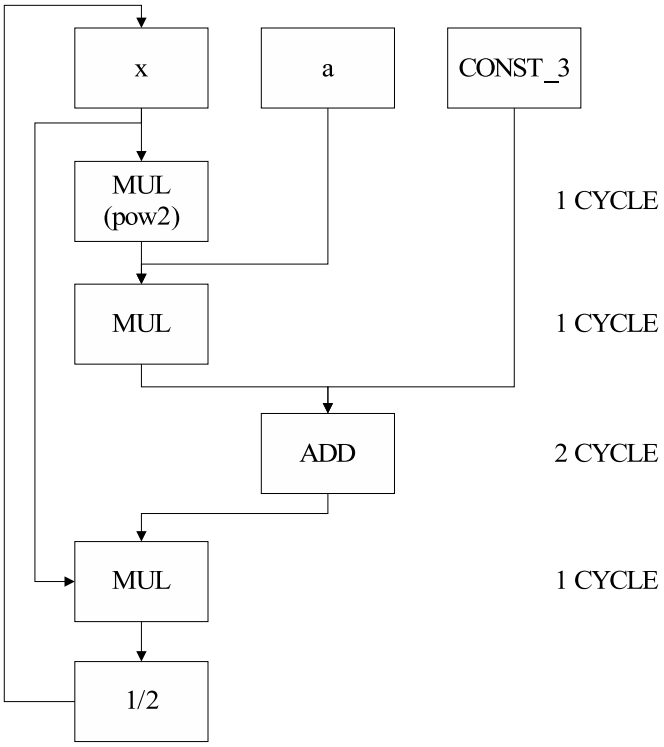

図 3.6: ニュートン法による平方根の逆数演算機構

表 3.14: ニュートン法による平方根の逆数演算機構の実装結果

反復回数	平均二乗誤差	SLICES	FF	LUT	実行サイクル数
1	6.2×10^{-1}	1214	322	2154	5
2	3.3×10^{-2}	1214	322	2154	10
3	4.2×10^{-4}	1214	322	2154	15
4	2.4×10^{-7}	1214	322	2154	20

表 3.15: ビットシフトと参照テーブルによる平方根の逆数演算機構の実装結果

アドレスビット数	平均二乗誤差	SLICES	FF	LUT	実行サイクル数
3	3.7×10^{-1}	41	64	50	1
5	4.9×10^{-2}	51	64	75	1
7	7.9×10^{-3}	133	64	250	1
9	2.0×10^{-6}	422	68	800	1
11	9.3×10^{-8}	1376	75	2596	1

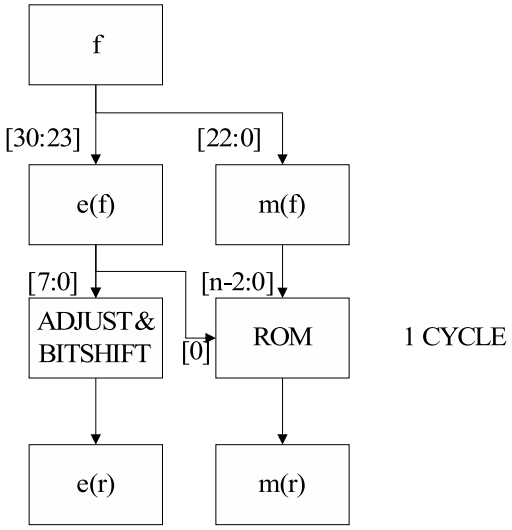

図 3.7: ビットシフトと参照テーブルによる平方根の逆数演算機構

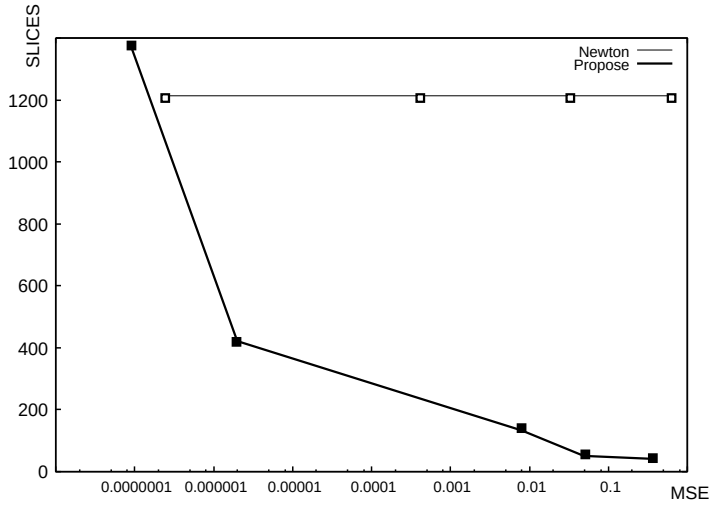

図 3.8: 平方根の逆数演算機構における平均二乗誤差に対する回路規模 (縦軸：SLICE 総数，横軸：平均二乗誤差，細線：従来方式，太線：提案方式）

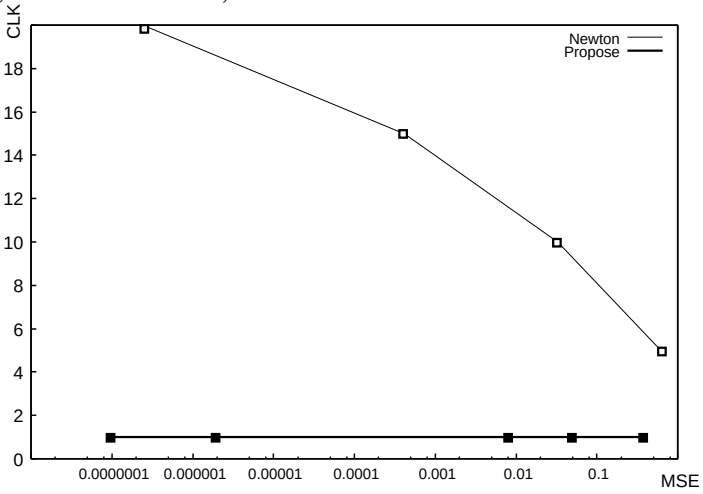

図 3.9: 平方根の逆数演算機構における平均二乗誤差に対する演算時間 (縦軸：CLK，横軸：平均二乗誤差，細線：従来方式，太線：提案方式）

第4章　応用例：修正離散コサイン変換

4.1　導入

1980 年代以降，磁気テープ，レコード等のアナログ媒体に代えて，CD，DVD 等のディジタル媒体が広く普及し，それに応じて種々のディジタル音響信号圧縮方式が利用されている．近年，主流となっている MP3，AAC，および OggVorbis は，いずれも窓単位に分割した入力信号を周波数変換し，周波数成分を聴覚心理に応じて適応的に符号化する歪み圧縮方式である．MP3 を用いれば，16[bit/sample] の信号を 1.45[bit/sample](128kBps) まで，AAC および OggVorbis を用いれば，16[bit/sample] の信号を 1.08[bit/sample](96kBps) まで減ずることが可能である．音声波形には符号ビットが存在することを考えると，これらの圧縮率は極めて高い．従来の ADPCM の限界であった 4[bit/sample] の壁を超えられたのは，周波数領域における符号化の恩恵である．

一例として AAC[19] の符号化および復号化の処理の流れを図 4.1 に示す．AAC の符号化では，入力された音声波形を窓単位に分割し，Pre-Processing ステージでゲインコントールを行った後，FilterBank ステージで周波数成分に変換する．各窓は，窓間の不整合によるノイズを抑制するため，一つ前の窓と半分ずつオーバラップした形で分割する．従って，FilterBank ステージは，単純なフーリエ変換を使用することができない．これは，単純なフーリエ変換では，オーバラップに相当する数だけ情報量が増加してしまうためである．そこで，窓をオーバラップした場合でも情報量が増加しない周波数変換として，修正離散コサイン変換が用いられる．フーリエ変換では N の音声波形に対して N の周波数成分が出力されるが，修正離散コサイン変換では $\frac{N}{2}$ の周波数成分しか出力されない．修正離散コサイン変換ではオーバラップを前提として直交性を担保しており，結果として，音響信号全体を通した音声波形と周波数成分の数は一致する．PerceptualModel ステージでは，FilterBank ステージによる周波数変換と平行し，聴覚心理モデルによって聴覚特性に応じたマスキング値を決定し，後段のツールに供給する．聴覚心理モデルでは，最小可聴限界である ATH(Absolute threshold of hearing)，SpreadingFunction のたたみ込みによって構成される同時マスキング，および時間方向のマスキングであるポストマスキングの三つのモデルから周波数帯域ごとの許容歪を求める．TNS(Temporal Noise Shaping) ステージでは，周波数成分に線形予測フィルタを適用することで，量子化雑音を時空間に分散させる．TNS は ATRAC3[20] で用いられる時空間でのゲイン制御を周波数領域で行うことと等価であり，強いアタック音から生じるプリエコーを抑制できる．Intensity/Coupling ステージでは，チャンネルカップリングおよびインテンシティステレオによって，チャンネル間の冗長性を削減する．Prediction ステージでは，過去フレームを利用した予測符号化を行うこと

で，フレーム間の冗長性を削減する．M/S ステージでは，各チャンネルの信号を，和信号と差信号に分解することで，チャンネル間の冗長性を削減する．ScaleFactors ステージでは，各周波数帯域ごとに，代表サンプルとなるスケールファクタを決定する．Quantizer ステージでは，各周波数帯域内の周波数成分を，スケールファクタに対する倍率の形式で表現し量子化する．NoiselessCoding ステージでは，量子化された各周波数成分へのエントロピー符号化を施す．ScaleFactors ステージ以降は，量子化雑音を出来る限り許容歪以下に抑えるために，各帯域へのビット割り当てを変更しながら IterationLoop を繰り返し，最適解を探索する．このように，符号化では，聴覚心理モデルによって周波数バンドごとの許容歪を求め，量子化雑音を許容歪以下に抑えることで，主観音質を保ったまま符号化効率を向上させている．

AAC の復号は，符号化の逆操作となる．まず，NoiselessDecoding ステージでビットストリームから周波数成分を復号し，InverseQuantizer ステージで逆量子化を行う．ScaleFactors ステージでは逆量子化された周波数成分にスケールファクタを乗じる．M/S ステージでは和信号と差信号からチャンネル別の周波数成分に復号する．Prediction ステージでは過去フレームとの予測符号を復号する．Intensity/Coupling ステージではステレオ処理を復号する．TNS ステージでは線形予測フィルタによってフレーム内の予測符号を復号する．FilterBank ステージでは復号された周波数成分から周波数逆変換によって音声波形を復元する．最後に Post-Processing としてゲインコントロールを行い，最終的な音声波形を得る．

以上は AAC の例であるが，MP3，OggVorbis とも，FilterBank で周波数成分に変換した後，量子化，エントロピー符号化をする流れは共通である．例えば，OggVorbis では，スケールファクタではなくスペクトル包絡の概形近似である Floor を構成し，Floor に対する各周波数成分の倍率をベクトル量子化することで符号化効率を高めている．

回路実装において，FilterBank は修正離散コサイン変換 [1] を高速フーリエ変換に変形した高速修正離散コサイン変換・逆変換 [2]-[6] によって実装され，回転子として正弦・余弦関数値を，窓関数として初等関数の合成関数値を用いる．AAC および OggVorbis では処理の単位となる窓幅を MP3 の 1152 から 2048 に拡大することで圧縮率を向上させているが，一般に，これらの関数値は離散化・標本化されて関数テーブルとして ROM に格納されているため，窓幅の拡大に伴い，回路規模が大きくなっている．特に，修正離散コサイン変換回路全体において，これらの関数テーブルは $\frac{1}{2}$ から $\frac{3}{4}$ の規模を占めている．回路規模は LSI のコストと消費電力に影響するため，特に iPod などに代表されるモバイル音楽プレイヤーにおいて，その小規模化が望まれている．

この問題に関して，本研究では，第 3 章で提案したビットシフトによる複素正弦波発生機構を，修正離散コサイン変換に最適な形に変形し，適用することで，回転子の回路規模を従来方式の $\frac{1}{2}$ から $\frac{7}{8}$ 程度に削減する方式を提案する．また，2 次差分予測符号化を利用することで，窓関数の回路規模を従来方式の $\frac{1}{4}$ 程度に削減する方式を提案する．その結果，修正離散コサイン変換回路全体を従来方式の $\frac{3}{4}$ 程度の回路規模に，逆変換回路全体を $\frac{1}{2}$ から $\frac{3}{4}$ 程度の回路規模に削減が可能であることを示す．

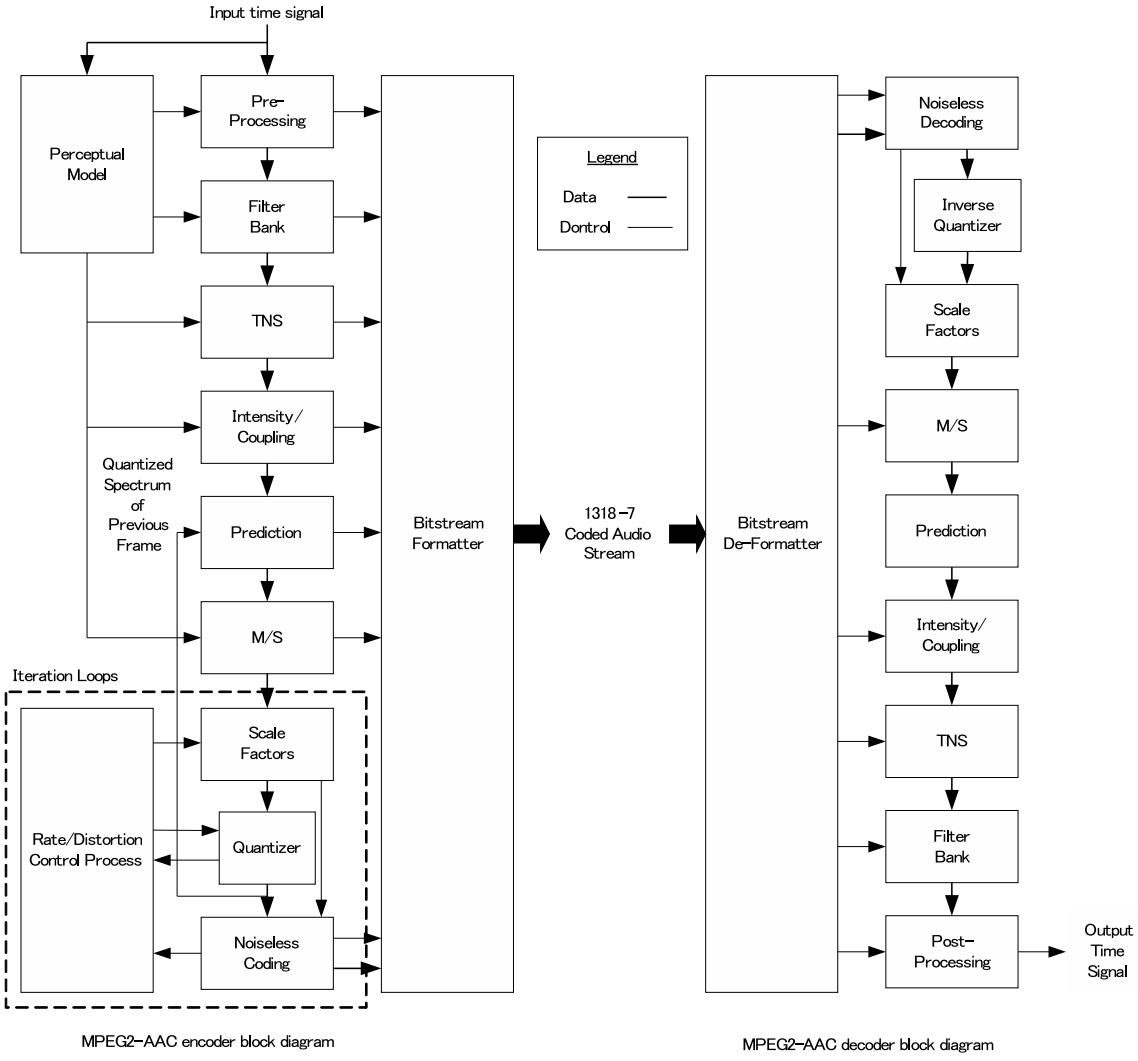

図 4.1: MPEG-2 AAC における符号化および復号化の処理の流れ [19]

4.2 要素技術

4.2.1 周波数変換

フーリエ変換は，時間波形 f_m から周波数成分 F_n に変換する処理であり，

$$\begin{aligned} F_n &= \sum_{m=0}^{N-1} f_m W_N^{mn}, \qquad (n = 0, 1, ..., N-1) \\ W_N^k &= e^{-i\frac{2\pi k}{N}} \end{aligned} \tag{4.1}$$

で定義される[1]．上記において，N は窓幅，W_N^k は回転子を示す．フーリエ変換の計算量は $O(N^2)$ であるが，これを $O(N \log N)$ とする高速フーリエ変換 (FFT) が多数提案されている．本論文では，(4.1) を変形して得られる基数 2 の Cooley-Tukey 型周波数間引き FFT

$$\begin{aligned} F_{2n} &= \sum_{m=0}^{\frac{N}{2}-1} (f_m + f_{\frac{N}{2}+m}) W_{\frac{N}{2}}^{mn}, \\ F_{2n+1} &= \sum_{m=0}^{\frac{N}{2}-1} (f_m - f_{\frac{N}{2}+m}) W_N^m W_{\frac{N}{2}}^{mn} \end{aligned}$$

を用いる．

修正離散コサイン変換 (MDCT)[1] は時間波形 x_m から周波数成分 X_n に変換する処理であり，

$$\begin{aligned} X_n &= \sum_{m=0}^{M-1} w_m x_m \cos(\frac{\pi}{2M}(2n+1)(2m+1+\frac{M}{2})), \\ (n &= 0, 1, ..., \frac{M}{2}-1) \end{aligned}$$

で定義される．ただし，M は窓幅，w_m は窓関数値である．本論文では，現在のコーデックで主流になっている $M = 2048$ について扱う．

MDCT[1] の計算量は $O(M^2)$ であるが，DCT-IV や FFT の係数に変換することで，これを $O(M \log M)$ とする高速 MDCT が多数提案されている [2]-[6]. 例えば，文献 [2] は，$2M$[sample] の RAM と $2M$[sample] の ROM により，MDCT の係数を FFT の係数に変換する事で高速化する方式を提案しており，文献 [3] および [4] は，FFT の係数への変換方式を改良することで，M[sample] の RAM と $\frac{5M}{4}$[sample] の ROM で実装する方式を提案している．また，文献 [5] は，M が比較的小さい場合に漸化式を用いて実装する方式を，文献 [6] は，MDCT の対称性を利用し，$\frac{M}{2}$[sample] の RAM と M[sample] の ROM を用いて実装する方式をそれぞれ提案している．本研究では，文献 [6] の計算式

$$\begin{aligned} &X_{2n} - iX_{\frac{M}{2}-2n-1} = \\ &\sum_{m=0}^{\frac{M}{4}-1} \{(y_{2m} - y_{M-2m-1}) + i(y_{\frac{M}{2}-2m-1} - y_{\frac{M}{2}+2m})\} W_M^{m+\frac{1}{8}} W_{\frac{M}{4}}^{mn} W_M^{n+\frac{1}{8}} \end{aligned} \tag{4.2}$$

[1]本論文では”i”で虚数単位を表す.

を用いる．ここで，(4.2) においては，FFT に対応する回転子 $W_{\frac{M}{4}}^{mn}$ と，前後の処理に対応する回転子 $W_M^{m+\frac{1}{8}}$，$W_M^{n+\frac{1}{8}}$ の乗数が異なり，回転子 W_M^m によって共有化できないため，2 種類の回転子が必要となることに注意する．この問題は，(4.2) を

$$X_{2n} - iX_{\frac{M}{2}-2n-1} = \sum_{m=0}^{\frac{M}{4}-1} \{(y_{2m} - y_{M-2m-1}) + i(y_{\frac{M}{2}-2m-1} - y_{\frac{M}{2}+2m})\} W_M^m W_M^{\frac{1}{8}} W_{\frac{M}{4}}^{mn} W_M^n W_M^{\frac{1}{8}}$$

と変形し，2 種類の乗数を一致させることで解決できる．

修正離散コサイン逆変換 (IMDCT) は周波数成分 X_m から時間波形 y_n を復元する処理であり，

$$\begin{aligned} y_m &= w_m x_m, \\ x_m &= \sum_{n=0}^{\frac{M}{2}-1} X_n \cos(\frac{2\pi}{M}(m + \frac{M}{4} + \frac{1}{2})(n + \frac{1}{2})) \\ (m &= 0, 1, ..., M-1) \end{aligned}$$

で定義される．ただし，M は窓幅，w_n は窓関数値である．尚，IMDCT には対称性

$$\begin{aligned} x_{\frac{M}{2}-1-2m} &= -x_{2m}, \quad (m < M/4) \\ x_{M-1-2(m-\frac{M}{4})} &= x_{2m} \quad (m \geq M/4) \\ (m &= 0, 1, ..., \frac{M}{2} - 1) \end{aligned}$$

が成立するため，x_{2m} が計算できれば十分である．

IMDCT の計算量は $O(M^2)$ であるが，DCT-IV や FFT の係数に変換することで，これを $O(M \log M)$ とする高速 MDCT が多数提案されている [2]-[6]．本論文では，文献 [6] の計算式

$$x_{2m} + ix_{2m+\frac{M}{2}} = \sum_{n=0}^{\frac{M}{4}-1} (-1)^n (X_{\frac{M}{2}-2n-1} - iX_{2n}) W_M^n W_M^{\frac{1}{8}} W_{\frac{M}{4}}^{nm} W_M^m W_M^{\frac{1}{8}} W_8^{-1}$$

を用いる．

4.2.2 関数テーブルの実装

一般に，回転子および窓関数値は，サンプリングされ，q[bit] 固定小数で ROM に格納される．その際，対称性

$$\begin{aligned} Im(W_M^{\frac{M}{2}-1-m}) &= Im(W_M^{m+1}), \quad (m = 0, 1, ..., \frac{M-4}{4}) \\ Im(W_M^{M-1-m}) &= -Im(W_M^{m+1}), \quad (m = 0, 1, ..., \frac{2M-4}{4}) \end{aligned}$$

$$
\begin{aligned}
Re(W_M^m) = Im(W_M^{m+\frac{M}{4}}), \quad (m &= 0, 1, ..., \frac{3M-4}{4}) \\
Re(W_M^{m+\frac{3M}{4}}) = Im(W_M^m), \quad (m &= 0, 1, ..., \frac{M-4}{4}) \\
w_m = w_{M-1-m} \quad (m &= 0, 1, ..., \frac{2M-4}{2})
\end{aligned}
$$

を利用することで，必要 ROM 容量をそれぞれ $q(\frac{M}{4}+1)$[bit] および $q\frac{M}{2}$[bit] まで削減することができる．

4.3 提案方式

4.3.1 回転子テーブルの削減

窓幅 2048 の MDCT および IMDCT における処理段階を

- 前後処理：$p = 0$
- FFT 処理：$p \in \{2, 3, \ldots, 10\}$

とする．各処理では，p に応じて区間 $[0, 2\pi]$ を 2^{11-p} 点で離散化した複素正弦関数値を 1[CLK] に 1 サンプルずつ順に利用する．上述したように方式 [15] は，1 サンプルの出力に複数実行サイクルを要するため，回転子の速度要件を満たすことができない．そこで，方式 [15] において，$\phi = 4\frac{2\pi}{2^{11-p}}$ とし，

$$
\cos\theta \approx \frac{1}{2^{16-p}}(2^7 + 2^6 + 2^3 + 1)
$$

なる近似を用いる．即ち，ROM の代用として，初期値 $(x_1, x_0) = (2^q, 0)$ における漸化式

$$
\begin{aligned}
x_{m+1} &= 2\cos\theta x_m - x_{m-1}, \\
y_m &= \cos\theta x_m - x_{m-1}, \\
\cos\theta &= \frac{1}{2^{16-p}}(2^7 + 2^6 + 2^3 + 1)
\end{aligned}
$$

の解

$$
Re(2^q e^{i\frac{2\pi}{2^{11-p}}m'}) = \begin{cases} (-1)^m y_{2m} & (m' = 2m) \\ (-1)^m x_{2m+1} & (m' = 2m+1) \end{cases}
$$

$$
Im(2^q e^{i\frac{2\pi}{2^{11-p}}m'}) = \begin{cases} (-1)^{m+1} x_{2m} & (m' = 2m) \\ (-1)^m y_{2m+1} & (m' = 2m+1) \end{cases}
$$

を利用する．特に，実装においては，

$$
\begin{aligned}
\sin(\theta + \frac{\pi}{4}) &= \cos(\theta), \cos(\theta + \frac{\pi}{4}) = \sin(\theta), \\
\sin(\theta + \frac{\pi}{2}) &= \cos(\theta), \cos(\theta + \frac{\pi}{2}) = -\sin(\theta)
\end{aligned}
$$

と変換することで精度を改善する．ただし，$\cos\theta \ll 1$ の仮定が破れて十分な精度を得られないため，$p \geq 7$ では，q[bit] 固定小数でサンプリングした 16 サンプルの関数テーブル ROM によって実装する．

4.3.2 窓関数テーブルの削減

本論文では，窓関数値 w_m の 1 次差分

$$\delta_m(1) = w_{m+1} - w_m \quad (m = 0, 1, 2, ..., \frac{M}{2} - 2)$$

及び 2 次差分

$$\delta_m(2) = \delta_{m+1}(1) - \delta_m(1) \quad (m = 0, 1, 2, ..., \frac{M}{2} - 3)$$

を考える．$q = 18$[bit] で離散化した Vorbis 窓における $\delta(1)$ 及び $\delta(2)$ の頻度分布を図 4.2 および図 4.3 に示す．

$\delta(1)$ は幅広い値域に分散するが，$\delta(2)$ は区間 $[-2, 2]$ に制限されることに注意する．この性質は，$q \in \{8, 9, ..., 15\}$ の Sine 窓，Vorbis 窓，および KBD 窓で成立するため，初期値 w_0，$\delta_0(1)$ 及び係数列 $\delta_m(2)$ を ROM に格納し，

$$w_{m+1} = w_m + \delta_m(1), \quad (m = 0, 1, 2, ..., \frac{M}{2} - 2) \tag{4.3}$$

$$\delta_{m+1}(1) = \delta_m(1) + \delta_m(2) \quad (m = 0, 1, 2, ..., \frac{M}{2} - 3) \tag{4.4}$$

を反復することで，窓関数値が計算可能となる．

4.4 比較評価

テーブル方式および提案方式について，修正離散コサイン変換および逆変換回路を FPGA 上にハードウェア実装し，計算精度および SLICE 総数を計測した．回路記述言語は VerilogHDL，ターゲット FPGA として Spartan3 を選択した．in-place 演算のための RAM は，16[bit]*$\frac{M}{4}$[sample] の BlockRAM を，実部および虚部の合計 2 つ利用した．ROM は，SLICE 総数で評価するため，LUT で実装した．計算精度は，性質の異なる複数のディジタル信号を各方式で符号化し実数精度の IMDCT で復号化した結果，および実数精度の MDCT で符号化し各方式で復号した結果と原曲との SNR で計測した．ディジタル信号の詳細および具体的な仕様を表 4.1，表 4.2 および表 4.3 に示す．

修正離散コサイン変換回路の評価　各方式の各ディジタル信号および各窓関数に対する SNR を表 4.4 および表 4.5 に，各方式の実装結果を表 4.6，および表 4.7 に示す．

3 種類の窓関数に対してテーブル方式および提案方式を適用した場合の平均 SNR に対する SLICE 総数を図 4.4 に示す．窓関数の種類に関わらず，提案方式はテーブル方式に比べ，高精

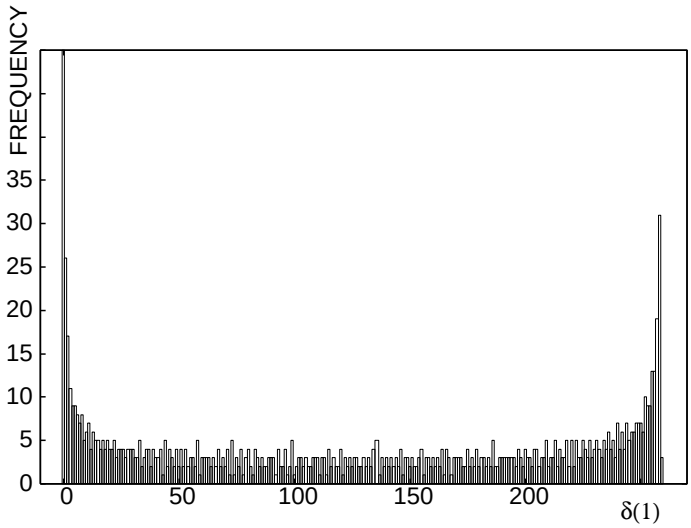

図 4.2: $\delta(1)$ の頻度分布 (縦軸：頻度，横軸：$\delta(1)$)

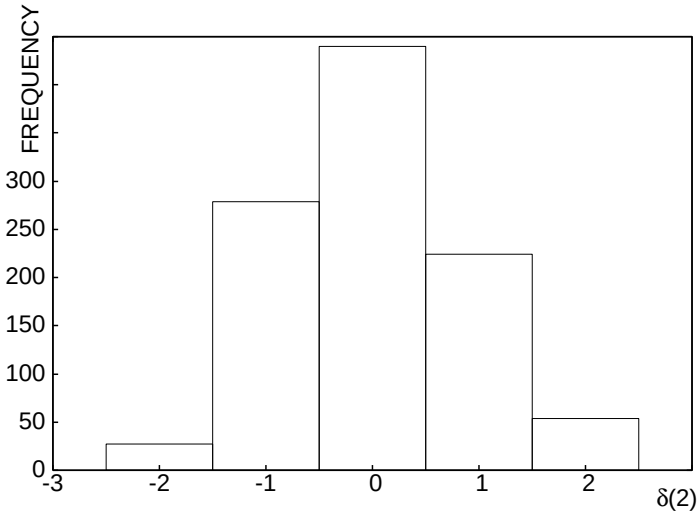

図 4.3: $\delta(2)$ の頻度分布 (縦軸：頻度，横軸：$\delta(2)$)

表 4.1: 複数のディジタル信号

名称	構成楽器	サンプル数
オーケストラ	ヴァイオリン・シンバル等	749700
テクノ	シンセサイザー・ドラム等	970200
ボイス	女性音声	132300
ピアノ	ピアノ	970200

表 4.2: テーブル方式の仕様

q[bit]	回転子 [bit]	窓関数 [bit]
12	12*513=6156	12*1024=12288
13	13*513=6669	13*1024=13312
14	14*513=7182	14*1024=14336
15	15*513=7695	15*1024=15360

表 4.3: 提案方式の仕様

q[bit]	回転子 [bit]	窓関数 [bit]
12	12*17=204	3*1024=3072
13	13*17=221	3*1024=3072
14	14*17=238	3*1024=3072
15	15*17=255	3*1024=3072

度で 79%，低精度で 83% 程度の SLICE 総数で実現できる．以降，3 種類の窓関数の SNR および SLICE 総数を平均した値で評価する．

回転子および窓関数のそれぞれについてテーブル方式および提案方式を適用した場合の平均 SNR に対する平均 SLICE 総数を図 4.5 に示す．提案方式を回転子のみに適用した場合，提案方式はテーブル方式に比べ，高精度で 91%，低精度で 93% 程度の SLICE 総数で実現できる．提案方式を窓関数のみに適用した場合，提案方式はテーブル方式に比べ，高精度で 79%，低精度で 83% 程度の SLICE 総数で実現できる．

提案方式を窓関数および回転子に適用した場合，提案方式はテーブル方式に比べ，高精度で 77%，低精度で 83% 程度の SLICE 総数で実現できる．提案方式を窓関数および回転子に適用した場合，各々に適用した場合に比べ，SLICE 総数の減少率が低下している．これは，ROM テーブルとして使用する SLICE の総量が減少したため，漸化式の計算に転用できる FF の総量も減少し，使用率の低い SLICE を新たに確保しなければならなかったためだと考える．

修正離散コサイン逆変換回路の評価　各方式の各ディジタル信号に対する SNR を表 4.8 および表 4.9 に，各方式の実装結果を表 4.10 および表 4.11 に示す．ディジタル信号の種類に関わらず提案方式は，テーブル方式より高い SNR を実現している．

回転子における平均 SNR に対する SLICE 総数を図 4.6 に示す．提案方式は，高精度の場合にテーブル方式の $\frac{1}{2}$ の SLICE 総数で実現できる．他方，低精度においては，加算器やレジスタの大きさが相対的に増し，テーブル方式の $\frac{7}{8}$ 程度の SLICE 総数に留まる．

窓関数における量子化ビット数に対する SLICE 総数を図 4.7 に示す．あらゆる精度において提案方式は，テーブル方式の $\frac{1}{4}$ 程度の SLICE 総数で実現できる．テーブル方式の SLICE 総数が高精度な領域において急増する理由は，アドレス 1 つに対応する ROM のデータ量が，1 つの SLICE に格納できるビット数を越え，隣接する SLICE を新たに確保したためだと考えら

図 4.4: MDCT における 3 種類の窓関数に対してテーブル方式および提案方式を適用した場合の平均 SNR に対する SLICE 総数 (縦軸：SLICE 総数，横軸：SNR[dB]，実線：テーブル方式，破線：提案方式，○：Sine 窓，△：Vorbis 窓，□：KBD 窓)

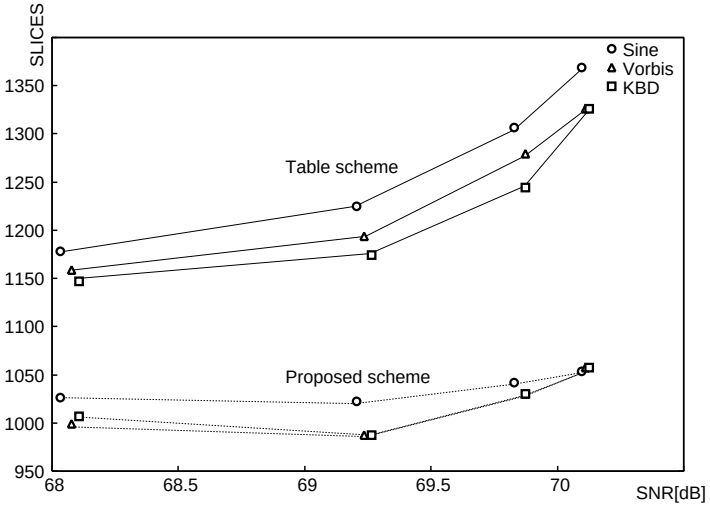

図 4.5: MDCT における回転子および窓関数のそれぞれについてテーブル方式および提案方式を適用した場合の平均 SNR に対する平均 SLICE 総数 (縦軸：SLICE 総数，横軸：SNR[dB]，(A) 回転子および窓関数をテーブル方式で実装，(B) 回転子を提案方式，窓関数をテーブル方式で実装，(C) 回転子をテーブル方式，窓関数を提案方式で実装，(D) 回転子および窓関数を提案方式で実装)

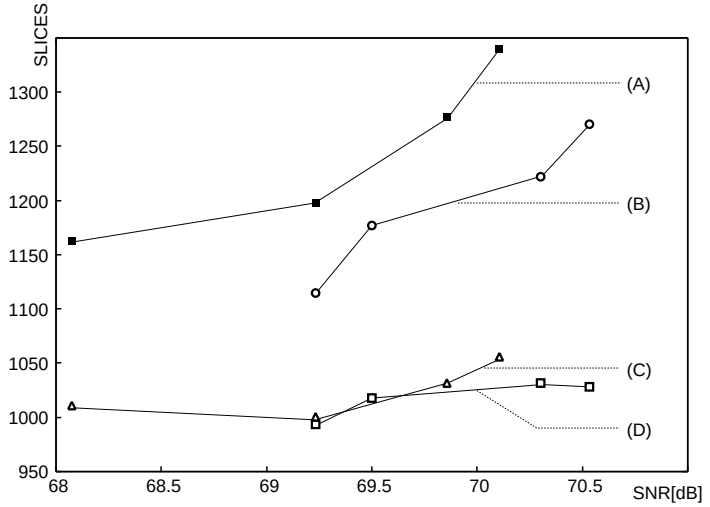

表 4.4: MDCT における回転子をテーブル方式で実装した場合の SNR[dB]

窓関数	q[bit]	オーケストラ	テクノ	ボイス	平均
Sine	12	68.26	66.61	69.24	68.04
	13	69.29	68.20	70.11	69.20
	14	69.83	69.08	70.58	69.83
	15	70.06	69.46	70.76	70.09
Vorbis	12	68.30	66.67	69.28	68.08
	13	69.33	68.24	70.15	69.24
	14	69.86	69.13	70.61	69.87
	15	70.08	69.50	70.78	70.12
KBD	12	68.32	66.70	69.32	68.11
	13	69.34	68.26	70.18	69.26
	14	69.87	69.13	70.61	69.87
	15	70.09	69.50	70.79	70.13

表 4.5: MDCT における回転子を提案方式で実装した場合の SNR[dB]

窓関数	q[bit]	オーケストラ	テクノ	ボイス	平均
Sine	11	69.55	67.94	70.13	69.21
	12	69.61	68.58	70.24	69.48
	13	70.29	69.68	70.88	70.28
	14	70.51	69.95	71.06	70.51
Vorbis	11	69.56	67.97	70.15	69.23
	12	69.64	68.61	70.25	69.50
	13	70.32	69.72	70.90	70.31
	14	70.53	70.00	71.08	70.54
KBD	11	69.57	68.00	70.19	69.25
	12	69.65	68.64	70.30	69.53
	13	70.33	69.73	70.91	70.32
	14	70.52	70.00	71.09	70.54

表 4.6: MDCT における回転子をテーブル方式で実装し窓関数をテーブル方式および提案方式で実装した場合の平均 SNR に対する SLICE 総数

窓関数	q[bit]	平均 SNR[dB]	テーブル方式	提案方式
Sine	12	68.03	1177	1026
	13	69.20	1225	1020
	14	69.83	1304	1040
	15	70.09	1366	1052
Vorbis	12	68.08	1159	996
	13	69.24	1194	986
	14	69.87	1277	1028
	15	70.12	1327	1054
KBD	12	68.11	1150	1006
	13	69.26	1176	987
	14	69.87	1246	1029
	15	70.13	1326	1055
平均	12	68.08	1162	1009
	13	69.23	1198	998
	14	69.86	1276	1032
	15	70.11	1340	1054

表 4.7: MDCT における回転子を提案方式で実装し窓関数をテーブル方式および提案方式で実装した場合の平均 SNR に対する SLICE 総数

窓関数	q[bit]	平均 SNR[dB]	テーブル方式	提案方式
Sine	11	69.21	1139	986
	12	69.48	1196	1030
	13	70.28	1246	1051
	14	70.51	1291	1038
Vorbis	11	69.23	1104	999
	12	69.50	1170	998
	13	70.31	1217	1013
	14	70.54	1261	1023
KBD	11	69.25	1096	990
	12	69.53	1166	1027
	13	70.32	1208	1027
	14	70.54	1254	1023
平均	11	69.23	1113	992
	12	69.50	1177	1018
	13	70.31	1223	1030
	14	70.53	1269	1028

れる．FPGA では配線上の制約が存在するため，SLICE 中の余ったリソースが再利用されにくく，階段状に回路規模が増加する傾向がある．

IMDCT 回路全体における平均 SNR に対する SLICE 総数を図 4.8 に示す．提案方式はテーブル方式に比べ，高精度で $\frac{1}{2}$，低精度で $\frac{3}{4}$ 程度の SLICE 総数で実現できる．

表 4.8: IMDCT におけるテーブル方式の SNR[dB]

q[bit]	オーケストラ	テクノ	ボイス	ピアノ	平均
12	66.45	65.48	67.32	69.64	67.22
13	67.01	66.48	67.91	69.89	67.82
14	67.33	67.00	68.20	70.05	68.15
15	67.50	67.22	68.34	70.19	68.31

表 4.9: IMDCT における提案方式の SNR[dB]

q[bit]	オーケストラ	テクノ	ボイス	ピアノ	平均
12	66.95	66.26	67.77	70.13	67.78
13	67.44	67.02	68.23	70.33	68.26
14	67.63	67.25	68.40	70.39	68.42
15	67.70	67.28	68.45	70.47	68.48

表 4.10: IMDCT におけるテーブル方式の平均 SNR に対する SLICE 総数

q[bit]	平均 SNR[dB]	TOTAL	内回転子	内窓関数
12	67.22	1515	233	470
13	67.82	1645	306	521
14	68.15	1761	418	516
15	68.31	2260	454	1215

表 4.11: IMDCT における提案方式の平均 SNR に対する SLICE 総数

q[bit]	平均 SNR[dB]	TOTAL	内回転子 (内 ROM)	内窓関数
12	67.78	1133	203(19)	127
13	68.26	1173	219(18)	138
14	68.42	1187	223(19)	144
15	68.48	1238	247(23)	148

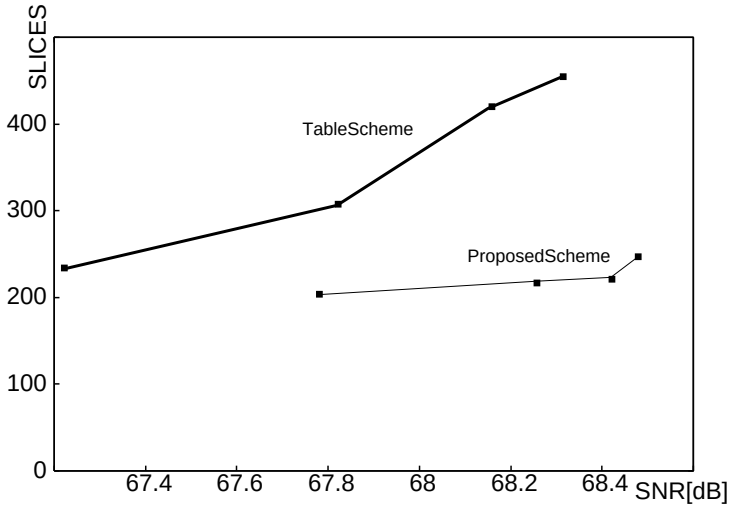

図 4.6: IMDCT の回転子における平均 SNR に対する SLICE 総数 (縦軸：SLICE 総数，横軸：SNR[dB]，太線：テーブル方式，細線：提案方式)

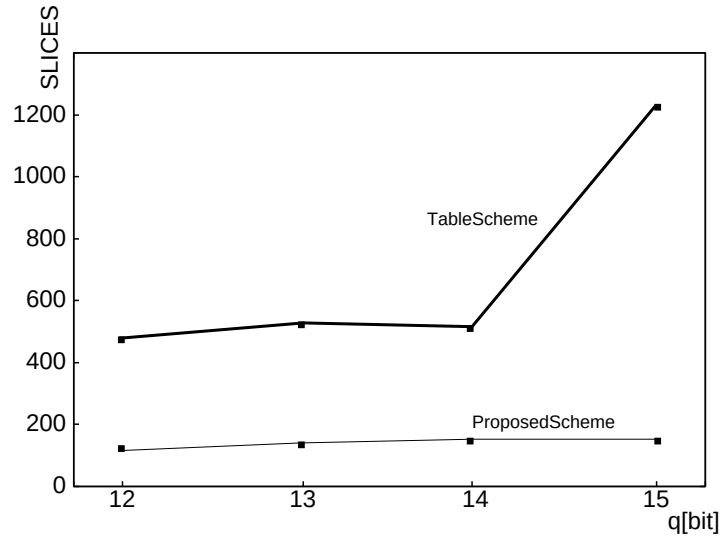

図 4.7: IMDCT の窓関数における量子化ビット数に対する SLICE 総数 (縦軸：SLICE 総数，量子化ビット数：q[bit]，太線：テーブル方式，細線：提案方式)

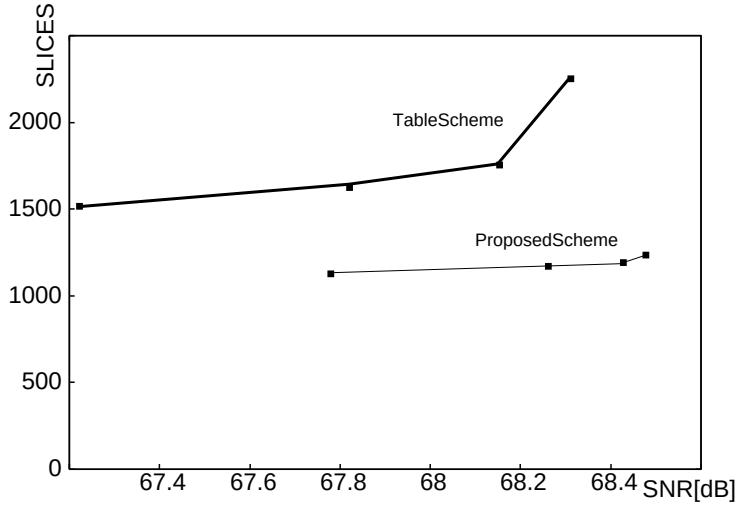

図 4.8: IMDCT 回路全体における平均 SNR に対する SLICE 総数 (縦軸：SLICE 総数，横軸：SNR[dB]，太線：テーブル方式，細線：提案方式)

第5章　応用例：グラム-シュミット直交化

5.1　導入

3D グラフィックス，特にリアルタイムレンダリングにおいて，ポリゴンは重要かつ一般的な方法論である．その表現の多様性と数学的枠組みの利便性から，科学技術計算における可視化，CAD・CAM の工業分野のみならず，コンピュータゲーム等のアミューズメントに広く応用されている．加えてポリゴンは，OpenGL や DirectX 等の 3D グラフィックス API における中心的手法であり，世界中のグラフィックスハードウェアに最も多く実装されている．近年，コンピュータアミューズメント分野における 3D グラフィックスの発展は特に著しく，グラフィック専用プロセッサの普及や OpenGL ES 等の組み込み機器向け API の策定にも後押しされ，携帯ゲーム機や携帯電話等の小型端末でも大量のポリゴンを用いたコンテンツが増加している．しかし，これらの組み込み機器においては，データ伝送用のバス幅は依然として狭く，パイプラインにおける伝送速度がポリゴンの高度利用の障害となっている．

この問題を解消するため，ポリゴンデータの転送コストを下げる様々な圧縮方式が検討されており，位相情報を保存する圧縮方式としては，

1. 座標や法線に関する幾何圧縮 [21]-[26]

2. 頂点の接続情報に関する位相圧縮 [21],[22],[23]

等がある．(2) はデータ損失のない無歪み圧縮であるが，(1) は，量子化・予測符号化・エントロピー符号化等の要素技術の組み合わせで構成されており，無歪みと歪み圧縮がある．科学技術用データおよび CAD・CAM データに対しては，無歪み圧縮の適用が不可欠であるが，コンピュータアミューズメントに用いられるマルチメディアデータにおいては，次の 2 点から，歪み幾何圧縮が有効と考えられる．

(a) 画像データや音響信号と同様の主観的利用であるため，モデルにおける凹凸等の曲面上の微細構造は，テクスチャにより 2 次元画像として扱われる．このためポリゴンそのものには，曲面の連続性や滑らかさの幾何学的冗長性が多く含まれている．

(b) ポリゴンデータのほとんどは浮動小数で記述される数値情報であり，無歪み圧縮では高い圧縮効率が得られない．

上記の観点から多数の歪み幾何圧縮が提案されている．文献 [21]，[22] は，頂点情報を整数精度へ量子化することで情報量を削減する方式を提案している．また，文献 [23]，[24] は，隣接三角形の情報を用いた平行四辺形予測による予測値との残差を符号化することで，圧縮効率を高める方式を提案している．しかし，同方式は，データを大域座標系で一様に処理す

るため，予測残差の冗長性が 3 自由度に分散し，圧縮効率の劣化が生じていると考えられる．これに対して，文献 [25] あるいは [26] は，隣接三角形の情報から基底系を構成し，残差を射影して圧縮効率を高める方式を提案している．

しかし，上記方式をハードウェア実装する場合，グラム-シュミット直交化の計算が平方根の逆数を用いたノルムの正規化処理を含むため，精度に応じて，反復法であるニュートン法に起因した多くの実行サイクル数を必要とするという問題がある．3D データ圧縮においては頂点毎に基底系を算出するため，正規化処理の負荷は無視できない．

そこで，本研究では，第 3 章で提案したビットシフトとテーブル参照による平方根の逆数演算機構をグラム-シュミット直交化に適用することで，より少ない実行サイクル数で基底を算出することが可能な方式を提案する．

5.2 要素技術

5.2.1 グラム-シュミット直交化

グラム-シュミット直交化は，線形独立なベクトルの組が与えられたとき，そこから正規直交系を生成する方法である．具体的に，線形独立なベクトル $x, y \in R^3$ が与えられた場合，正規直交基底 $w_x, w_y \in R^3$ は

$$\begin{aligned} w_x &= \frac{x}{||x||} \in R^3, \\ w_y &= \frac{y-(y\cdot w_x)w_x}{||y-(y\cdot w_x)w_x||} \in R^3 \end{aligned}$$

と計算できる．

5.2.2 グラム-シュミット直交化による直交変換符号化

文献 [21] は，ポリゴンデータの一般的応用において 32[bit] 浮動小数による頂点座標の記述が冗長である事を指摘し，整数精度への量子化が有効となることを報告している．また，文献 [22] は，頂点座標を量子化した後，位相情報をランレングス符号化することで，圧縮効率を高める方式を提案している．さらに，文献 [23]，[24] は，全頂点を量子化した後，隣接する 2 つの三角形が平行四辺形を成すと仮定し，一定の順路で辺を巡りながら頂点 $\{p_n \in R^3\}$ を差分ベクトルとして予測符号化する方式を提案している．例えば，図 5.1 における三角形 $\{p_n, p_{n+1}, p_{n+2}\}$ から p_{n+3} の予測値

$$p'_{n+3} = p_{n+2} + p_{n+1} - p_n \in R^3$$

を生成し，残差ベクトル

$$d = p_{n+3} - p'_{n+3} \in R^3$$

を符号化対象とする．

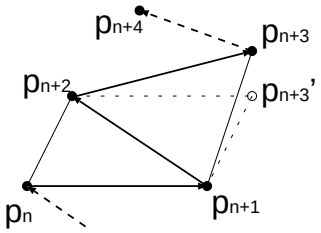

図 5.1: 平行四辺形予測

他方，文献 [25] は，隣接三角形 $\{p_n, p_{n+1}, p_{n+2}\}$ から非直交基底系

$$\begin{aligned} w_x &= p_{n+1} - p_n \in R^3, \\ w_y &= p_{n+2} - p_n \in R^3, \\ w_z &= \frac{w_x \times w_y}{||w_x \times w_y||}\sqrt{||w_x||||w_y||} \in R^3 \end{aligned} \tag{5.1}$$

を構成し，差分ベクトル

$$d = p_{n+3} - p_n \in R^3$$

を座標 (5.1) に関する 3 成分へ変換し，符号化対象とする方式を提案している.（図 5.2 参照）

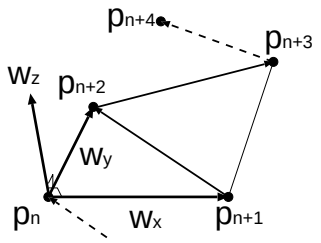

図 5.2: 文献 [25] の基底系

また，文献 [26] は， 2 辺

$$\begin{aligned} x &= p_{n+1} - p_{n+2} \in R^3, \\ y &= p_{n+1} - p_n \in R^3 \end{aligned}$$

をグラム-シュミット直交化と外積によって正規直交化した基底系

$$\begin{aligned} w_x &= \frac{x}{||x||} \in R^3, \\ w_y &= \frac{y - (y \cdot w_x)w_x}{||y - (y \cdot w_x)w_x||} \in R^3, \\ w_z &= w_x \times w_y \in R^3 \end{aligned} \tag{5.2}$$

を構成し，差分ベクトル

$$d = p_{n+3} - \frac{p_{n+1} + p_{n+2}}{2} \in R^3$$

を座標 (5.2) に関する 3 成分へ変換し，符号化対象とする方式を提案している．（図 5.3 参照）文献 [26] は，高い圧縮率が実現できるが，三角形と共面となる $\{w_x, w_y\}$ の正規化処理に逆数 $||x||^{-1}$ および $||y-(y\cdot w_x)w_x||^{-1}$ が必要となり，平方根の逆数演算が必要となる．

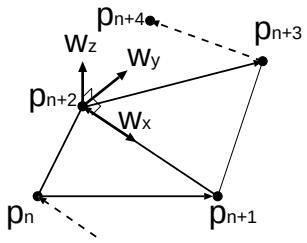

図 5.3: 文献 [26] の基底系

5.2.3 ニュートン法によるグラム-シュミット直交化

3 次元ベクトル $x, y \in R^3$ を入力とし，基底 $w_x, w_y, w_z \in R^3$ を計算する回路を考える．w_x および w_y はグラム-シュミット直交化によって計算し，両者を外積することで w_z を求める．実行サイクル数を削減するため，浮動小数加算器 3 つと浮動小数乗算器 3 つを並列に用い，サイクルごとに入力を切り替えて使用する．w_x のデータフローを図 5.4 に示す．平方根の逆数演算を除けば，三次元ベクトルの正規化は 6 サイクルで計算が可能である．w_y のデータフローを図 5.5 に示す．w_y の中には最終段で正規化を含むが，平方根の逆数演算を除けば，14 サイクルで計算が可能である．w_z のデータフローを図 5.6 に示す．外積演算には 4 サイクルが必要となる．

ニュートン法の実装においては，浮動小数加算器および浮動小数乗算器をグラム-シュミット直交化と共有する．これは，グラム-シュミット直交化は各計算に強い順序依存性があるため，ニュートン法とその他の演算の並列化が困難であるためである．ニュートン法は，精度に応じて多くの反復が必要であるため，グラム-シュミット直交化のスループットが低下するという問題がある．

5.3 提案方式

5.3.1 ビットシフトと参照テーブルによるグラム-シュミット直交化

従来方式におけるニュートン法による平方根の逆数演算機構を，ビットシフトと参照テーブルによる平方根の逆数演算機構で置換する．提案方式は，1 サイクルで平方根の逆数を計算可能なため，高いスループットを得ることができる．

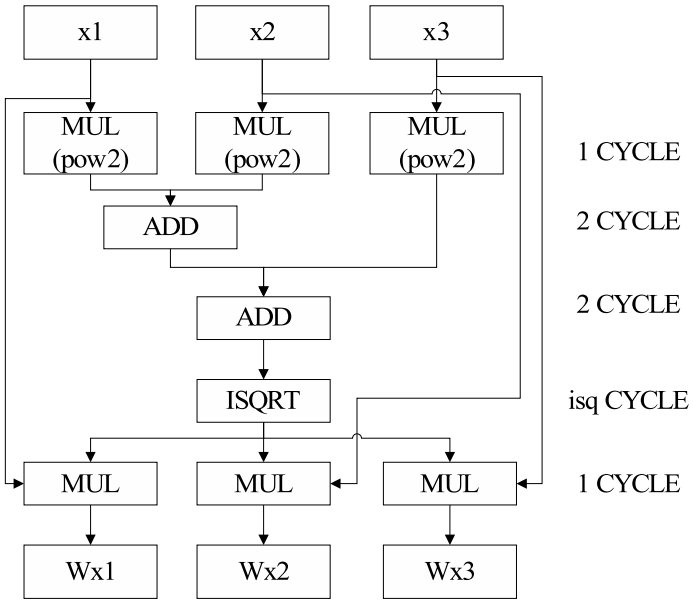

図 5.4: w_x の計算

5.4 比較評価

従来方式および提案方式について，平方根の逆数演算機構と同条件で FPGA 上に実装し，近似精度に対する SLICE 総数を比較する．精度は，1000 組の乱数ベクトルについて，ライブラリ関数による基底を真値として，各方式による基底との平均二乗誤差 (MSE) で計算した．表 5.1 および 5.2 に，従来方式および提案方式の実装結果を示す．

回路規模の評価　各方式の平均二乗誤差に対する回路規模を図 5.7 に示す．低精度においては，ニュートン法の制御回路よりも ROM テーブルの方が回路規模が小さくなるため，提案方式は従来方式よりも小さな回路規模で実現できる．中精度においては，従来方式と提案方式はほぼ同等の回路規模で実現可能であるが，従来方式におけるニュートン法の演算器をグラム-シュミット直交化と共有できるため，平方根の逆数演算機構単独で合成した場合に比べ，より低い精度で回路規模が交差している．高精度においては，ROM テーブルの回路規模が大きくなるため，従来方式の方が小さな回路規模で実現可能である．

実行サイクル数の評価　各方式の平均二乗誤差に対する実行サイクル数を図 5.8 に示す．提案方式は，従来方式に比べ，中精度で $\frac{1}{2}$ 程度，低精度で $\frac{5}{8}$ 程度の実行サイクル数で演算が可能である．

総合評価　中から低の演算精度において，提案方式は従来方式と同等の回路規模で，従来方式の $\frac{1}{2}$ から $\frac{5}{8}$ 程度の実行サイクル数で実装可能である．従って，ある程度の歪が許容され，ス

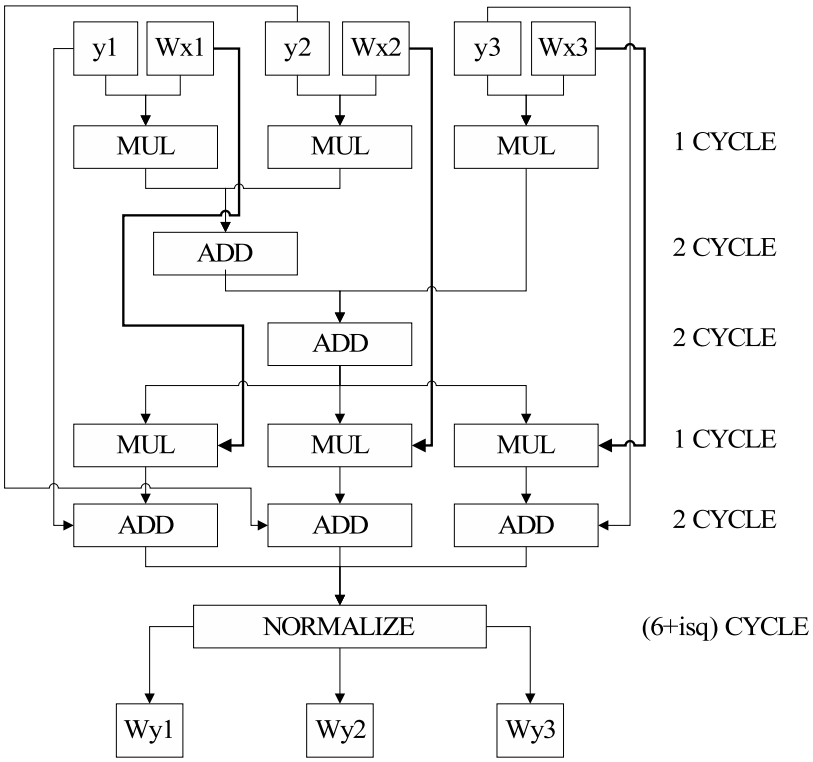

図 5.5: w_y の計算

ループットが重要となる 3D ポリゴン圧縮への応用において，提案方式は有効であると考える．

表 5.1: ニュートン法によるグラム-シュミット直交化の実装結果

反復回数	平均二乗誤差	SLICES	FF	LUT	実行サイクル数
1	8.9×10^{-2}	4122	1747	7142	34
2	1.5×10^{-2}	4122	1747	7142	44
3	4.5×10^{-4}	4122	1747	7142	54
4	1.8×10^{-7}	4122	1747	7142	64

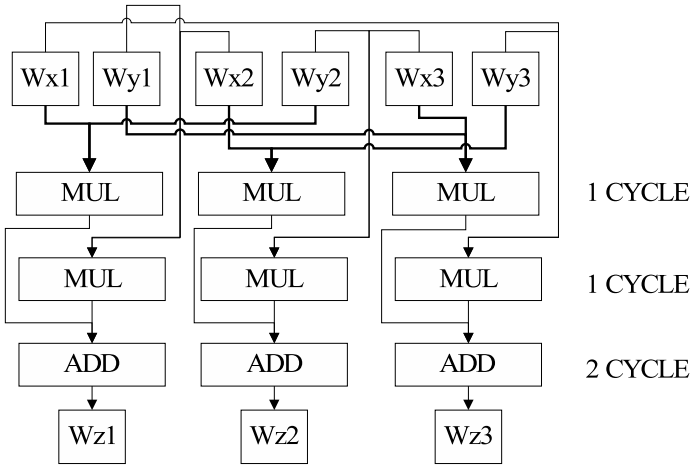

図 5.6: w_z の計算

表 5.2: ビットシフトと参照テーブルによるグラム-シュミット直交化の実装結果

アドレスビット数	平均二乗誤差	SLICES	FF	LUT	実行サイクル数
5	3.1×10^{-2}	3948	1611	6849	26
7	7.8×10^{-3}	4026	1611	6999	26
9	1.4×10^{-3}	4272	1615	7449	26
11	7.9×10^{-4}	5180	1630	9150	26

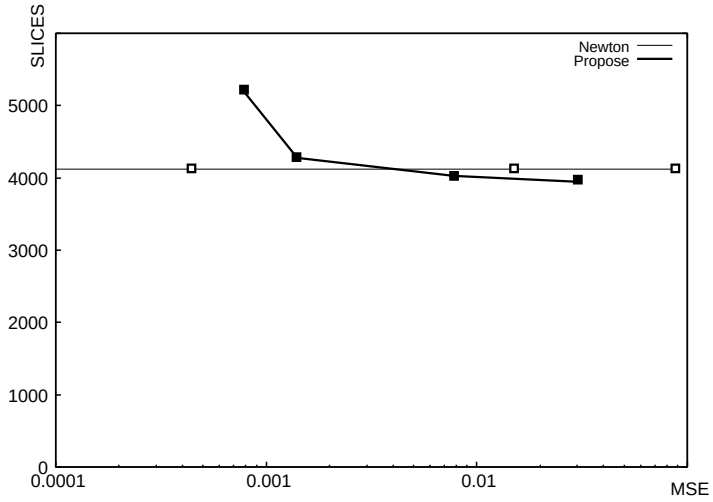

図 5.7: グラム-シュミット直交化の平均二乗誤差に対する回路規模 (縦軸：SLICE 総数，横軸：平均二乗誤差，細線：従来方式，太線：提案方式）

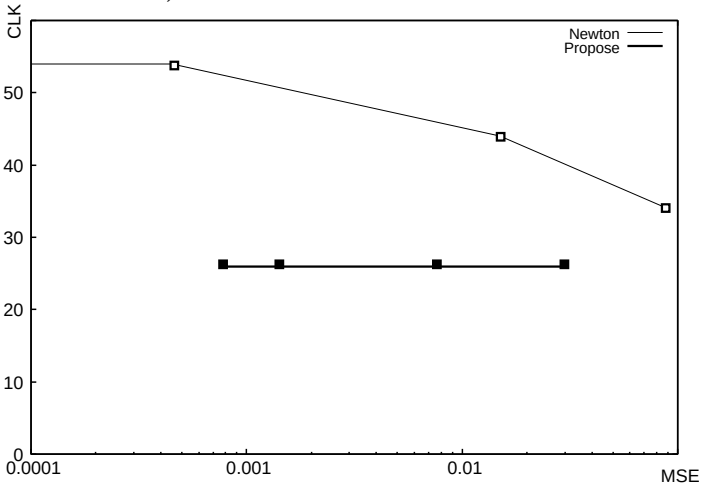

図 5.8: グラム-シュミット直交化の平均二乗誤差に対する演算時間 (縦軸：CLK，横軸：平均二乗誤差，細線：従来方式，太線：提案方式）

第6章　結論

本論文は，正弦関数，余弦関数，窓関数，および平方根の逆数の実装について，より小さな回路規模で実装可能な方式を提案し，その応用を検討した．第 2 章では，各初等関数およびその一般的な実装について解説し，巡回型正弦波発生機構における乗算に起因した回路規模の増加と，ニュートン法における浮動小数の乗算に起因した回路規模と実行サイクル数の増加という二つの問題を明らかにした．第 3 章では，これらの問題に対し，ビットシフトと参照テーブルを用いた各初等関数の小規模な実装を提案した．第 4 章では，第 3 章で提案したビットシフトによる複素正弦波発生機構を修正離散コサイン変換に最適な形に変形し適用することで，音声符号化および復号化における周波数変換回路の回路規模を削減可能なことを示した．第 5 章では，第 3 章で提案した参照テーブルとビットシフトによる平方根の逆数演算機構をグラム-シュミット直交化の正規化処理に適用することで，3D ポリゴン圧縮におけるスループットを向上させることが可能であることを示した．本論文の論点は以下の通りである．

- 初等関数の小規模な実装の提案（第 3 章)

  巡回型正弦波発生機構における $\cos\theta$ を $\frac{1}{2^k}$ で近似し，乗算をビットシフトに置換することで，正弦関数および余弦関数を従来方式の $\frac{1}{2}$ 程度の回路規模で実装可能な方式を提案した．また，複素正弦波発生機構においても，同様に乗算をビットシフトに置換し，出力を交互に符号反転しながらサンプリングすることで，従来方式の $\frac{3}{4}$ 程度の回路規模で実装可能な方式を提案した．また，平方根の逆数演算機構において，入力される浮動小数を指数部と仮数部に分解し，指数部の最下位ビットと仮数部の上位ビットをアドレスとするテーブル参照と，指数部のビットシフトによる出力とを結合することによって，平方根の逆数を中から低の演算精度で従来方式の $\frac{1}{3}$ 程度の回路規模で実装可能な方式を提案した．

- 修正離散コサイン変換への応用（第 4 章)

  修正離散コサイン変換の高速実装における前後処理，およびフーリエ変換で用いられる回転子に対し，第 3 章で提案したビットシフトによる複素正弦波発生機構を窓幅 2048 に最適な形に変形し適用することで，回転子の回路規模を従来方式の $\frac{1}{2}$ から $\frac{7}{8}$ 程度に削減が可能であることを示した．提案方式は $\cos\theta = \frac{1}{2^{16-p}}(2^7 + 2^6 + 2^3 + 1)$ として乗算を加減算とビットシフトに置換することで，シフト量 p を制御するだけで 32 から 2048 サンプルの複素正弦波を 1 サンプル単位で出力可能である．また，窓関数に対し，2 次差分符号化を適用することで，Sine 窓，Vorbis 窓，および KBD 窓の回路規模を従来方式の $\frac{1}{4}$ 程度に削減が可能であることを示した．これら 2 方式を修正離散コサイン変換お

よび逆変換に適用することで，修正離散コサイン変換回路全体を従来方式の $\frac{3}{4}$ 程度の，逆変換回路全体を $\frac{1}{2}$ から $\frac{3}{4}$ 程度の回路規模で実装可能なことを示した．

- 3D ポリゴン圧縮回路への応用（第 5 章)
  3D ポリゴン圧縮におけるグラム-シュミット直交化の正規化処理に対し，第 3 章で提案した参照テーブルとビットシフトによる平方根の逆数演算器を適用することで，基底の算出に必要な実行サイクル数を中から低の演算精度で $\frac{1}{2}$ から $\frac{5}{8}$ 程度まで削減可能なことを示した．

本研究で示したように，必要とする精度および実行サイクル数を定め，漸化式，ビットシフト，ROM テーブル，および差分符号を適切に組み合わせ，最適化することで，初等関数をより小さく，あるいはより高速に実装をすることが可能である．これらの回路規模および実行サイクル数の改善は，ディジタル音響信号圧縮および 3D ポリゴン圧縮の低コスト化と低消費電力化をより促進させると考えられる．

付 録A　巡回型正弦波発生機構の導出

最も単純な 2 係数族 2 次元線形差分方程式

$$\begin{pmatrix} x_{n+1} \\ x_n \end{pmatrix} = \begin{pmatrix} 2a & b \\ 1 & 0 \end{pmatrix} \begin{pmatrix} x_n \\ x_{n-1} \end{pmatrix} \tag{A.1}$$

の一般解が，正弦関数となる場合，次の 2 条件が成立する．

領域保存条件　(A.1) の線型作用素は領域保存

$$|\det \begin{pmatrix} 2a & b \\ 1 & 0 \end{pmatrix}| = |b| = 1$$

でなければならない．仮に，(A.1) が非保存（散逸）である場合，軌道は発散もしくは原点に収束する．

発振条件　(A.1) が振動的であるためには，固有値が複素共役でなければならない．したがって，特性方程式

$$\det \begin{pmatrix} \lambda - 2a & -b \\ -1 & \lambda \end{pmatrix} = \lambda^2 - 2a\lambda - b = 0$$

の判別式から $a^2 + b < 0$ となり，少なくとも $b < 0$ となる．

上記の 2 つの条件から，$b = -1$ となり，

$$-1 < a = \cos\theta < 1 \tag{A.2}$$

が (A.1) の調和振動の発振条件となる．最終的に (A.2) から 1 係数族差分力学系

$$\begin{pmatrix} x_{n+1} \\ x_n \end{pmatrix} = \begin{pmatrix} 2\cos\theta & -1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} x_n \\ x_{n-1} \end{pmatrix}$$

が定まる．上式の線形作用素の標準形は，回転行列

$$\begin{pmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{pmatrix}$$

となり，一般解は

$$\begin{pmatrix} x_{n+1} \\ x_n \end{pmatrix} = \begin{pmatrix} \frac{\sin(n+1)\theta}{\sin\theta} & \frac{\sin n\theta}{\sin\theta} \\ \frac{\sin n\theta}{\sin\theta} & \frac{\sin(n-1)\theta}{\sin\theta} \end{pmatrix} \begin{pmatrix} x_1 \\ x_0 \end{pmatrix}$$

である．

付 録B　ビットシフトと参照テーブルによる平方根の逆数演算機構のソフトウェアへの応用

B.1　導入

本論文で述べた提案方式は，ハードウェアのみならずソフトウェアに適用することも可能である．例えば，汎用 CPU において除算および平方根計算は極めて低速であり，Intel Pentium4 における IA-32 命令であれば，加減算の 11 倍ものクロックサイクル数が必要である [27]．従って，頂点毎に基底系を算出する場合，ソフトウェアにおいても正規化処理の負荷は無視できず，平方根計算命令を提案方式で置換することでパフォーマンスを向上させることが可能である．

これらの観点から，付録 B では，グラム-シュミット直交化を用いたポリゴンデータの復号処理を IA-32 命令によって基底算出をする従来方式と，ビットシフトと参照テーブルによる平方根の逆数演算機構によって基底算出をする提案方式とを，ソフトウェアで実装，比較・評価し，その有効性を示す．

B.2　グラム-シュミット直交化を用いたポリゴンデータの符号化

符号化は次の手順で行う．

1. 三角形ストリップから頂点 $p_n \in R^3$ を読み込み，$\{p_{n-3}, p_{n-2}, p_{n-1}\}$ から座標系 (5.2) による直交変換を求める．このとき，従来方式または提案方式により正規化を行う．
2. 差分ベクトル $d = p_n - \frac{p_{n-1}+p_{n-2}}{2} \in R^3$ を変換する．
3. 変換後の 3 成分を量子化ステップ 2^c により線形量子化する．
4. 量子化後の 3 成分を符号部 1[bit]，指数部 8[bit]，仮数部 $(a-c)$[bit] で格納する．ただし，a は指数値を示す．
5. 逆量子化を行い 3 成分を復号し，p_n を更新する．
6. 頂点が終了するまで上記を反復する．

尚，$\{p_{n-3}, p_{n-2}, p_{n-1}\}$ が縮退している場合は，事前に計算された直交変換を用いる．次に (4) の形式で格納された頂点情報を読み出し，上記の (1)，(2)，(5) を反復することで三角形ストリップを復号する．

B.3 実験環境

実験モデルとして，Stanford Computer Graphics Laboratory[28] において公開されている，“Dragon”，“Stanford Bunny”，“Armadillo” の 3 つを使用する．(表 B.1 参照) これらの位相情報は三角形リスト形式で記述されているため，TriStripper[29] を用い，一定の順路で頂点を重複なく巡る三角形ストリップ形式に変換する．

表 B.1: 3 種類のモデルデータ

モデル名	頂点数	ストリップ長
Dragon	433652	1284359
Stanford Bunny	34834	90972
Armadillo	172974	493400

実験では，各モデルの復号に要する時間及び復号結果を用いて比較を行う．復号後のポリゴンデータの歪み [30] は，

$$\begin{aligned} \mathrm{error} &= \sum_{\mathrm{k}=1}^{\mathrm{N}} (\mathrm{V_k} - \mathrm{V'_k})^2, \\ \mathrm{SNR} &= 10 \log_{10} \frac{\sum_{\mathrm{k}=1}^{\mathrm{N}} (\mathrm{V_k} - \mathrm{center})^2}{\mathrm{error}} \ [\mathrm{dB}] \end{aligned}$$

を用いて評価する．尚，N は頂点数，$\mathrm{V_k} \in R^3$ は真の，$\mathrm{V'_k} \in R^3$ は復号後の k 番目の頂点座標，center は全頂点の重心を示す．SNR は，V_k の各次元で計算した後，平均化することで 1 次元化する．

グラム-シュミット直交化においては，一対の 3 次元ベクトルからグラム-シュミット直交化によって第一と第二基底を求め，両者を外積し第三基底を求める．関数テーブルの各々の値は，IEEE754 方式による単精度仮数部 23[bit] で表し，テーブルサイズは

$$\{23 \times 2^n [\mathrm{byte}] \mid \mathrm{n} = 0, 1, 2, 3\}$$

の 4 種類を用いた．また，実験環境として，Windows XP Professional SP2，Intel Core2 Duo 1.86[GHz] $\times$ 2, 0.99[GB] RAM を用いた．

B.4 比較・評価

“Stanford Bunny” を用いてテーブルサイズを $n \in \{0, 1, 2, 3\}$ とした場合の SNR の変化を図 B.1 に，復号時間の変化を図 B.2 に示す．テーブルサイズが増加する程，関数近似の精度が向上し，SNR も向上することが分かる．歪みの改善は，$n = 2$ でほぼ飽和するため，92[byte] の

関数テーブルで十分な精度が得られることが分かる．他方，復号時間はテーブルサイズに依存せず，15〜16[ms] の間に収まる．尚，上記の 2 つの性質は，“Dragon” 及び “Armadillo” においても，同様に観察された．

最終的に，テーブルサイズを 92[byte] に固定した場合の復号時間と SNR を表 B.2 に示す．復号時間を見ると，提案方式は従来方式に比べ 50%〜65%程度に短縮できている．SNR も，“Dragon” の $c = -13$ においては 1.0[dB] の劣化が見られるものの，その他の誤差は僅かであり，また，“Stanford Bunny” と “Armadillo” においては従来方式とほぼ同等の SNR が得られていることが分かる．

表 B.2: 復号時間と SNR

Dragon

c	復号時間 [ms]		SNR[dB]	
	従来方式	提案方式	従来方式	提案方式
-13	406	234	55.8	54.8
-12	375	235	50.0	49.6
-11	375	234	43.8	43.6
-10	375	234	38.0	37.7
-9	375	250	32.1	31.2

Stanford Bunny

c	復号時間 [ms]		SNR[dB]	
	従来方式	提案方式	従来方式	提案方式
-13	31	16	57.4	57.3
-12	31	15	51.4	51.2
-11	31	16	44.9	44.9
-10	32	15	37.2	37.2
-9	31	16	31.0	31.1

Armadillo

c	復号時間 [ms]		SNR[dB]	
	従来方式	提案方式	従来方式	提案方式
-3	156	94	56.0	56.0
-2	157	94	48.8	48.8
-1	140	94	41.9	41.9
0	141	78	35.9	35.9
1	141	79	30.1	30.1

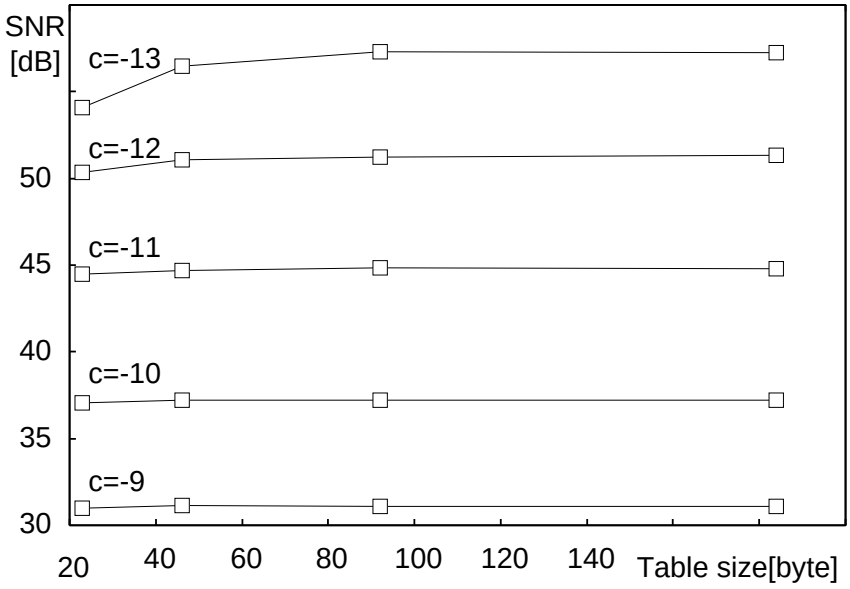

図 B.1: 関数テーブルサイズと SNR(横軸：テーブルサイズ [byte]，縦軸：SNR[dB]，節点：左から $n = 0, 1, 2, 3$)

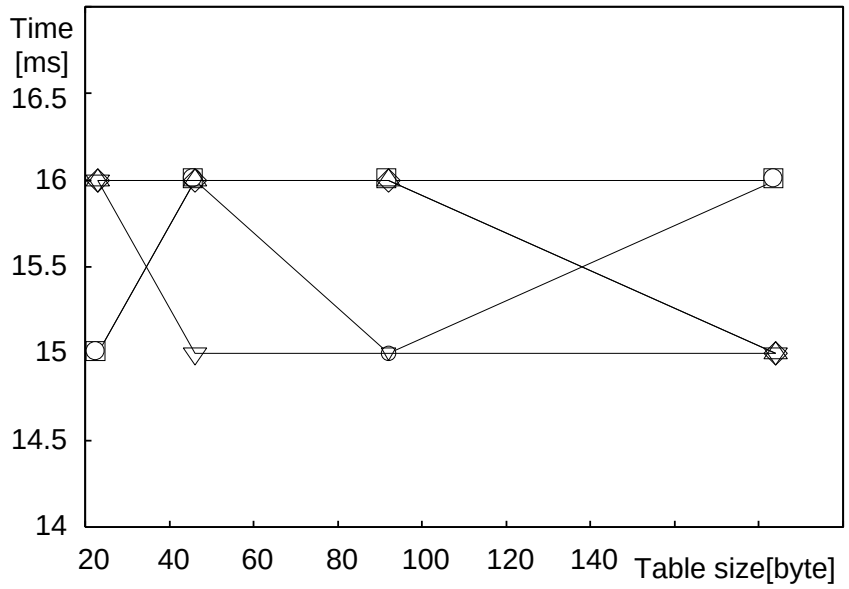

図 B.2: 関数テーブルサイズと復号時間 (横軸：テーブルサイズ [byte]，縦軸：復号時間 [ms]，節点：左から $n = 0, 1, 2, 3$，□：$c = -13$，○：$c = -12$，△：$c = -11$，▽：$c = -10$，◇：$c = -9$)

B.5　結論

付録 B では，ポリゴンモデル圧縮方式の一つである直交変換符号化における平方根除算に対し，第 3 章で提案した関数テーブルとビットシフトによる平方根の逆数演算機構を適用することで，少ない SNR の劣化の下でソフトウェアでの復号時間を 50〜65% に短縮できることを報告した.

付 録C　研究業績一覧

C.1　本論文に関する研究業績

1. 客野一樹，徳永隆治,
“巡回型ディジタル正弦波発生機構に関する考察 : ビットシフトによる回路規模の低減”,
電子情報通信学会論文誌 (A)，vol.J90-A，no.2，pp.165-169，Feb．2007.

2. 客野一樹，徳永隆治,
“ビットシフトによる巡回型ディジタル正弦波発生機構の性能改善 : 実行サイクル数の削減”,
電子情報通信学会論文誌 (A)，vol.J90-A，no.4，pp.330-334，Apl．2007.

3. 客野一樹，徳永隆治,
“音響信号デコーダにおける関数テーブルの省メモリ化”,
電子情報通信学会論文誌 (A)，vol.J90-A，no.10，pp.735-741，Oct．2007.

4. 客野一樹，徳永隆治,
“音響信号エンコーダにおける関数テーブルの省メモリ化”,
電子情報通信学会論文誌 (A)，vol.J91-A，no.9，pp.909-914，Oct．2008.

5. 長野寛，客野一樹，松本健太，徳永隆治,
“ポリゴンデータ幾何圧縮のための局所変換の実装方式”,
電子情報通信学会論文誌 (A)，vol.J93-A，no.3，pp.207-211，Oct．2010.

C.2　その他の研究業績

1. 星月優佑，田中英智，客野一樹，徳永隆治,
“ブロック中心値補正および直流成分保存型フィルタによる交流成分予測の改良”,
電子情報通信学会論文誌 (A)，vol.J92-A，No.1，pp.62-66，Jan．2009.

2. 森屋和喜，星月優佑，小島悠貴，客野一樹，徳永隆治,
“交流成分予測に基づく平均値保存型画像フィルターの構成”,

電子情報通信学会論文誌 (A)，vol.J92-A，No.9，pp.644-650，Sept．2009.

3. 小島悠貴，森屋和喜，星月優佑，客野一樹，徳永隆治，
“交流成分予測に基づく平均値保存型画像フィルターの改良：１係数族による解析”，
電子情報通信学会論文誌 (A)，vol.J92-A，No.12，pp.999-1008，Dec．2009.

4. 星月優佑，田中英智，客野一樹，徳永隆治，
“交流成分予測に基づくブロック型無歪み画像圧縮方式の構成”，
電子情報通信学会論文誌 (A)，vol.J93-A，No.3，pp.222-228，Mar．2010.

5. 森屋和喜，小島悠貴，客野一樹，徳永隆治，
“交流成分予測に基づく平均値保存型画像フィルタの改良：オーバー・アンダーシュートに起因する歪の除去”，
電子情報通信学会論文誌 (A)，vol.J93-A，No.6，pp.433-439，Jun．2010.

参考文献

[1] J.Princen, A.Johnson, A.Bradley, “Subband / Transform Coding Using Filter Bank Design Based on Time Domain Aliasing Cancellation”, Proceedings of IEEE ICASSP, pp.2161-2164 (1987)

[2] M.Bosi, Richard E.Goldberg, “An Introduction to digital audio coding and standards”, Kluwer Academic Publishers (2002)

[3] Th.Sporer, Kh.Brandenbarg, B.Edler, “The use of multilate filter banks for coding of high quality digital audio”, 6th EUSIPCO, Vol.1, pp.211-214 (1992)

[4] H.S.Malvar, “Extended lapped transforms:fast algorithms and applications”, Proceedings of IEEE ICASSP, pp.1797-1800 (1991)

[5] H.C.Chiang, J.C.Liu, “Regressive Implementations for the Forward and Inverse MDCT in MPEG Audio Coding”, IEEE Signal Processing Letters, Vol.3, No.4 (1996).

[6] P.Duhamel, Y.Mahieux, J.P.Petit, “A fast algorithm for the implementation of filter banks on ’time domain aliasing cancellation”’, Proceedings of IEEE ICASSP, pp.2209-2212 (1991).

[7] A.B.Gumacos, “Digital multiple tone generator”, U.S.Patent no.3649821 (1972)

[8] K.Fruno, S.K.Mitra, K.Hirano, Y.Ito, “Design of Digital Oscillator with Absolute Periodicity”, IEEE Transaction on Aerospace and Electronic Systems, vol.AES-11, no.6, pp.1286-1299 (1975)

[9] M.M.Ai-Ibrahim, “Efficient digital oscillator with continuous phase and uniform frequency spacing”, IEEE International Conference on Electronics, Circuits and Systems, vol.3, no.14-17, pp.1129-1132 (2003)

[10] S.Menon, G.Cho, N.Soderstand, “An Improved Numerically Controlled Digital Oscillator”, IEEE Pacific Rim Conference on Communications, Computers and Signal Processing, vol.28-30, pp.1040-1044 (2003)

[11] C.S.Turner, “Recursive Discrete-Time Sinusoidal Oscillator”, IEEE Signal Processing Magazine, vol.20, Issue.3, pp.103-111 (2003)

[12] J.Smith，P.Cook，“The second digital waveguide oscillator”，Proceedings of International Computer Music Conference San Josse，pp.150-153 (1992)

[13] N.J.Fliega，“Complex digital oscillators and FSK modulators”，IEEE Transactions Signal Processing，vol.40，no.2，pp.333-342 (1992)

[14] 客野一樹，徳永隆治，“巡回型ディジタル正弦波発生機構に関する考察 : ビットシフトによる回路規模の低減”，電子情報通信学会論文誌 (A)，vol.J90-A，no.2，pp.165-169，Feb.2007.

[15] 客野一樹，徳永隆治，“ビットシフトによる巡回型ディジタル正弦波発生機構の性能改善 : 実行サイクル数の削減”，電子情報通信学会論文誌 (A)，vol.J90-A，no.4，pp.330-334，Apl.2007.

[16] 客野一樹，徳永隆治，“音響信号デコーダにおける関数テーブルの省メモリ化”，電子情報通信学会論文誌 (A)，vol.J90-A，no.10，pp.735-741，Oct.2007.

[17] 客野一樹，徳永隆治，“音響信号エンコーダにおける関数テーブルの省メモリ化”，電子情報通信学会論文誌 (A)，vol.J91-A，no.9，pp.909-914，Oct.2008.

[18] 長野寛，客野一樹，松本健太，徳永隆治，“ポリゴンデータ幾何圧縮のための局所変換の実装方式”，電子情報通信学会論文誌 (A)，vol.J93-A，no.3，pp.207-211，Oct.2010.

[19] M Bosi，K. Brandenburg，S. Quackenbush，L. Fielder，K.Akagiri，H. Fuchs，M. Dietz，J.Herre，G.Davidson，and Y. Oikawa，“ISO/IEC MPEG-2 Advanced Audio Coding”，Journal of the Audio Engineering Society，vol.45，pp.789-812 (1997)

[20] ソニー株式会社，“信号符号化方法及び装置、信号復号化方法および装置、並びに情報記録媒体及び情報伝送方法”，特開平 8-237132

[21] M. Deering，“Geometry Compression”，SIGGRAPH 95，pp.13-22 (1995)

[22] G. Taubin，J. Rossignac，“Geometric Compression Through Topological Surgery”，ACM Transactions on Graphics，vol. 17，no. 2，pp.84-115 (1998)

[23] C. Tauma，C. Gotsman，“Triangle Mesh Compression”，Graphics Interface ’98 Conference Proceedings，pp.26-34 (1998)

[24] M. Isenburg，P. Alliez，“Compressing Polygon Mesh Geometry with Parallelogram Prediction”，Proceedings of Visualization 2002，pp.141-146 (2002)

[25] E. Lee，H. Ko，“Vertex data compression for triangular Meshes”，In Pacific Graphics ’00 Conference Proceedings，pp.383-392 (2002)

[26] S. Gumhold，R. Amjoun，“High Order Prediction for Geometry Compression”，International Conference on Shape Modelling And Applications，pp.59-66 (2003)

[27] インテル株式会社，“IA-32 命令のレイテンシ とスループット”，
http://download.intel.com/jp/developer/jpdoc/ia32.pdf

[28] Stanford University，“Stanford Computer Graphics Laboratory”，
http://graphics.stanford.edu/

[29] GPSnoopy’s Development Arena , “Tristripper”, http://users.pandora.be/tfautre/softdev/tristripper

[30] J. Li，C.C. Kuo, “Progressive coding of 3d graphics models”，Proceedings of the IEEE, Special Issue on Multimedia Signal Processing，vol. 86，pp.1052-1063 (1998)

謝辞

本研究を行うにあたって，研究を行う機会を与えて頂き，多くのご指導，ご助言を頂いたシステム情報工学研究科准教授の徳永隆治先生に深い感謝の意を表します．さらに，本論文の審査にあたって貴重な助言を頂いた筑波大学システム情報工学研究科の山本幹雄教授，工藤博幸教授，久野誉人教授，河辺徹准教授に深謝し御礼申し上げます．最後に，本研究を進めるにあたって惜しみない協力をして頂いた，研究室の皆様にも改めて感謝の意を表します．