動画像解析に適したウィンドウ透視可能なマルチウィンドウシステム

茅野　功

A Multi-window System with Translucentizing the Windows Suitable for Video Analysis

Isao KAYANO

キーワード：マルチウィンドウ，ユーザインタフェース，透視化，動画像解析

概　要

本稿では，既に筆者等が開発をしたスムーズ操作が可能なマルチウィンドウシステムを応用した，操作の対象となるウィンドウ（以下，操作対象ウィンドウ）以外のウィンドウを透視化することのできるマルチウィンドウシステムについて，その合成法を中心に解説した．本合成方式は，操作対象ウィンドウ以外のウィンドウを描画時合成原理に基づいて合成し格納しているメモリと，新たに設けた背景画像を格納するメモリの出力に，輝度の低下や透視化処理を行うハードウェアを付加することにより，ホストへの負荷をほとんど与えることなく所望の処理を高速に画面に反映することができる．本システムを応用することにより，従来では表示の困難であった動画像上へのウィンドウ透視化が可能になり，特に医療分野で多用される動画像解析システムへの適用が有用である．

1．まえがき

コンピュータの性能の向上･用途の多様化に伴って，画面上に多数のウィンドウを同時に表示したまま作業を進めるユーザが増えている．これに伴って，作業対象のウィンドウの識別性が低下してしまったり，壁紙に配置されるGUIアイコンが認識できなくなるといった問題が生じている．

前者の問題は，作業対象のウィンドウの輝度が，その周辺に表示されているウィンドウの輝度に比して低い場合に顕著になると思われる．そこで，このような問題を解消する１つの方法として，作業対象外のウィンドウの輝度を低下する方法が提案されている[1]．また，後者の問題を解消する方法として，作業対象以外のウィンドウを透視化する方法が提案されている[2]．しかし，これらの方法では，いずれもソフトウェア的な手法が用いられているため，作業対象のウィンドウが頻繁に変更されるような場合，CPUに多大な負担を強いることになる．

臨床の分野においては，血管内血流速度解析や膝関節運動解析など，動画像を利用した運動解析等の演算処理がコンピュータを用いて盛んに行われている．最新の高速運動解析システムを図１に示す[3]．近年のこのようなシステムでは，ビデオカメラによって撮影された動画像からその運動を解析し，その結果をグラフ化して表示することができる．しかしこの結果の表示は，解析が全て終了してから行うか，あるいは，解析過程を表示するために動画像ウィンドウとは別のウィンドウ（子ウィンドウ）を作成し，そこに行うしかない．つまり，動画像の撮影からリアルタイムで解析過程を

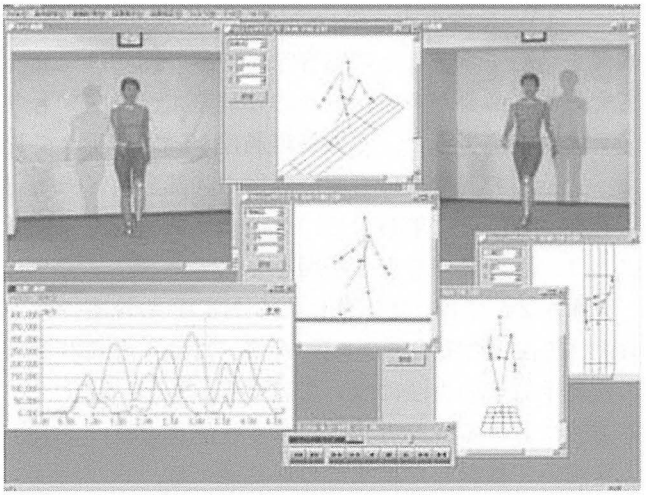

図１　高速運動解析システムの画面例

（平成16年10月４日受理）
川崎医療短期大学　臨床工学科
Department of Medical Engineering, Kawasaki College of Allied Health Professions

動画像ウィンドウ上に表示することはできないのである．これを行うためには，動画像と解析結果画像のそれぞれを演算処理して透視画像を作成しなければならないことから，画像動画像の表示及び運動解析のためにCPUに多大な負荷をかけ，ソフトウェア処理の大きな透視化画像の生成が画像表示に間に合わないためである．撮影した動画像ウィンドウ上でリアルタイムに運動解析結果を透過して表示することができれば，動画像上の部位と結果を一目で判断することができるため，解析結果のより的確な判断を行うことができ，非常に有用なシステムになると考えられる．

そこで筆者等は，画像演算に伴うCPUの負担を軽減するために，ウィンドウ画像の輝度の低下及び透視化処理上をハードウェア的に実現したマルチウィンドウシステム（以下，単に本方式と呼ぶ）を開発した．

筆者等はこれまでに，すべてのウィンドウをウィンドウ操作の対象となるウィンドウ（TWと略記する）とそれ以外のウィンドウ（NTWと略記する）に分類し，NTWについてはソフトウェア的手法[4,5]で合成した上で，この合成画像とTWの画像とをハードウェア的手法[6,7]で高速合成するという原理に基づいたマルチウィンドウシステムを提案している[8]．このシステムにおいて，TW及びNTWを，それぞれ，上述した作業対象ウィンドウ及び作業対象外ウィンドウに対応させると，作業対象ウィンドウとそれ以外のウィンドウの画像に対して独立に処理を施すことができることになる．そこで，この点に着目して，NTWに対して輝度の低下処理あるいは透視化処理を施すことによって，CPUの高速化にほとんど配慮することなく高速に輝度の低下あるいは透視化を行い，またその解除が可能なマルチウィンドウ合成方式を提案した[9]．この方法によれば，TWを変更する場合には，TW及びNTWの画像を格納するメモリの内容を，作業の開始に先立って変更する必要があるが，この処理をすべてソフトウェア的に実施すると，その処理過程が画面に現れてしまう．そこで，処理に必要な画像の大部分がメモリ内に存在することに着目して，2つのメモリ間でDMAにより，この処理を行う手順を与えている．

以下，**2**．では，本方式によるNTWの輝度の低下および透視化の実現方法を述べ，**3**．では本方式のTWの変更手続きの方法について述べる．また，**4**．では，本方式における従来方法に対する性能比を試作システムを用いて検証した結果について述べる．

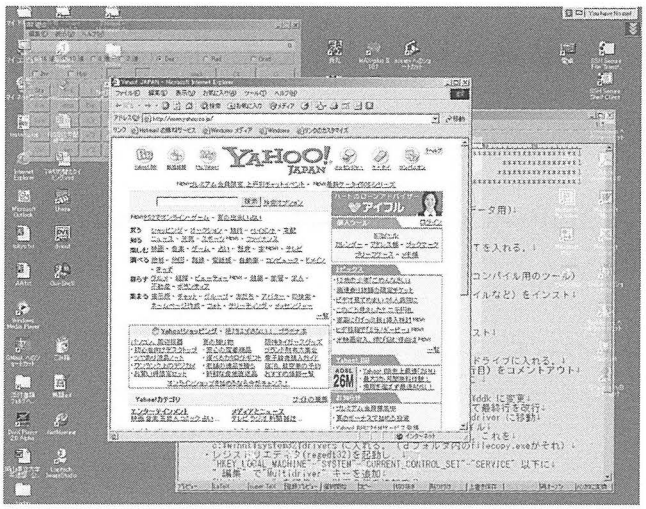

図2　透視化した画面の一例

2　マルチウィンドウの合成原理

2．1　操作対象ウィンドウの表示優先順位と透視化

本方式のマルチウィンドウシステムのベースとなる文献[8]のシステムでは，TWの表示優先順位は任意に設定できる．しかし，本方式では，ユーザがTWとNTWとを容易に識別できるマルチウィンドウシステムの実現を目的としているので，TWは最前面に表示することが望ましい．そこで以下，TWの表示優先順位は常に最上位に固定することとする．

ウィンドウの透視化は，一般に，あるウィンドウの領域内において，その下に隠された別のウィンドウが可視状態になるように，そのウィンドウを透かすことを意味する．前述した壁紙に配置されるGUIアイコンの認識を実現するためには，すべてのウィンドウを透視化すべきである．しかし，このままではTWの識別性が低下する．そこで，これらのトレードオフとして，TWは透視化せず，NTWのみを透視化することとする．さらに，NTWの透視化については，GUIアイコンの認識を実現するという視点に立って，すべてのNTWの可視領域内において背景画像が可視状態になるようにするものとする．透視化の一例を図2に示す．

2．2　マルチウィンドウ合成原理

表示されるn個のウィンドウをW_1～W_nで表す．添え字は，ウィンドウが生成された時に割り当てられる各ウィンドウ固有の値であり，単にウィンドウ番号（表示優先順位ではないことに注意する）と呼ぶ．ウィンドウ番号はウィンドウが消滅するまで値が変化することはなく，ウィンドウが生成される毎に1から順番に生成される．また，背景画像も1つのウィンドウとみ

なし，このウィンドウはW_0とする．ユーザは，W_1～W_nのうちいずれか1つをTWに指定し，それに対してウィンドウ操作あるいは作業の実行を指令する．したがって，TWに指定されなかった残りのウィンドウ（$n-1$個）によってNTWの画像を構成する．

本方式のマルチウィンドウ合成原理を図3（$n=3$）に示す．ただし，W_1がTW，W_2とW_3がNTWである．BM_{TW}，BM_{NTW}及びBM_{BG}は，それぞれ，TWの完全な画像，NTWを合成した画像及び背景画像を格納するためのメモリである．これらのメモリはいずれも表示装置1画面分に対応するアドレス空間をもち，その深さは画像の色と同じ深さである．また，FM_{TW}及びNTW_{NUM}は，それぞれ，TWの完全な領域及びNTWを構成する各ウィンドウの可視領域を，領域内部であれば対応するウィンドウ番号，領域外であれば0（背景ウィンドウの番号）として格納するためのメモリ（画像メモリと同じアドレス空間と深さ）である．ただし，NTW_{NUM}については，NTWの和領域以外にのみ0を格納する．以下，FM_{TW}とNTW_{NUM}のことを領域メモリといい，それらに格納される情報のことを領域情報と呼ぶ．OPC及びISは，それぞれ，NTWの画像に対して輝度の低下や透視化を行うための演算回路及び2-1マルチプレクサである．

マルチウィンドウを表示する際，BM_{NTW}上のNTWの画像及びBM_{BG}上の背景画像は，表示装置のフレーム周期毎に，0番地（画面左上に対応する）から表示装置の走査に同期して，それぞれ，シリアル信号i_{NTW}及びi_{BG}として読み出される．これと同様に，NTW_{NUM}の領域情報も0番地からシリアル信号f_{NTW}として読み出される．OPCにおいては，各画素毎に，i_{NTW}とi_{BG}に対して輝度の低下や透視化のための演算を施し，その結果をシリアル信号i_{NTW}'として出力する．輝度の低下はi_{NTW}のR，G，Bの各成分毎に乗算を行うことで実現し，透視化はi_{NTW}とi_{BG}について加重加算を行うことで実現する（詳細は**2．3**参照）．これと同期して，BM_{TW}上の完全画像およびFM_{TW}上の領域情報も，それぞれ，シリアル信号i_{TW}およびf_{TW}として読み出される．ただし，これらの読み出し開始アドレスは，ウィンドウ操作にともなう表示画面上でのウィンドウ位置に応じて左最上位から移動する．最後に，ISにおいては，$f_{NTW}=0$の場合i_{NTW}'を，$f_{NTW}\neq 0$の場合i_{TW}を選択・出力し，表示装置にマルチウィンドウ画像を表示する．

2．3 輝度の低下と透視化の実現法

輝度の低下と透視化を実現する図3のOPCにおける演算原理を図4に示す．

table Aには，NTWと1対1に対応したエントリが設けられており，各エントリには，輝度の低下率または透視化を行う際のNTWの画像に対する重み（乗率）が格納される．また，table Bには，table Aと同様のエントリが設けられており，各エントリには透視化を行う際の背景画像に対する重みが格納される．Multiplier A及びBは乗算器であり，Adderは加算器である．また，MUX 1とMUX 2はいずれも2-1マルチプレクサである．

NTWの輝度を低下させる場合，画面の走査に同期して出力されるi_{NTW}の各画素毎に，f_{NUM}に基づいて，table Aから読み出された輝度低下率を乗じることにより輝度を低下させ，MUX 1及びMUX 2を介して出力される．

他方，NTWの画像を背景画像に透視化する場合，

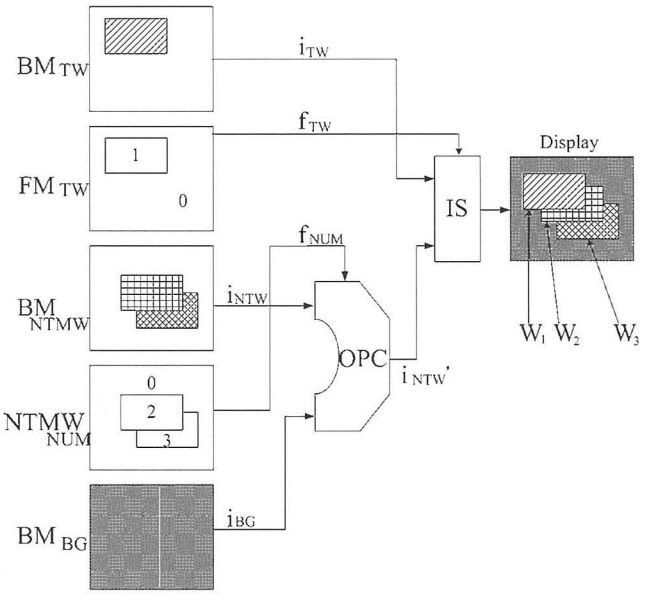

図3　本方式によるマルチウィンドウの合成原理

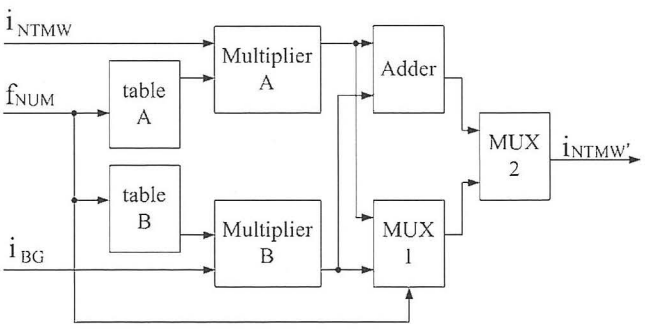

図4　輝度の低下と透視化の実現原理

まず，画面の走査に同期して i_{BG} の各画素毎に，f_{NUM} に基づいて，table B から読み出された重みを乗じることにより背景画像の輝度を低下させる．これと並行して，輝度を低下させる場合と同様に，i_{NTW} の輝度も低下させる．この際，同一エントリにおける table A と table B の値の関係は，table B の値（乗率）を α とする場合，table A の値は $(1-\alpha)$ となる．Multiplier A及びBから出力されたそれぞれのシリアル画像情報は，Adder により加算され，MUX 1 及び MUX 2 を介して透視化された画像として出力される．

NTW の輝度を低下させる場合，NTW の画像を背景画像に透過する場合のいずれについても，BM_{NTW} 上の NTW の画像及び BM_{BG} 上の背景画像自体は変化しない．したがって，輝度の低下または透過化を解除した場合や，TW が変更された場合にも高速に対応することができることに注意する．

3 TW の切り替え

TW を変更する場合には，2つの画像メモリ BM_{TW} と BM_{NTW} の内容，及び，2つの形状メモリ FM_{TW} と NTW_{NUM} の内容を変更する必要がある．そこでここでは，変更前における各メモリの内容を極力利用し，変更途上における表示画面のちらつきを完全に抑制しつつ，新たな TW と NTW の画像を生成する．

3.1 切り替え方針

まず以下の記号を定義する．

$I^{C(h)}$：W_h の完全な画像

$I^{V(h)}$：W_h の可視領域内の画像

$I^{B(h)}$：TW の切替え前における W_h の仮想可視領域内の画像

$I^{A(h)}$：TW の切替え後における W_h の仮想可視領域内の画像

$R^{C(h)}$：W_h の完全な領域

$R^{V(h)}$：W_h の可視領域

$R^{B(h)}$：TW の切替え前における W_h の仮想可視領域

$R^{A(h)}$：TW の切替え後における W_h の仮想可視領域

T^h ：W_h の輝度テーブルの内容

切り替え前および切り替え後の TW を，それぞれ，W_i，W_j とする．ここで，h はターゲットウィンドウ切替え前におけるウィンドウの優先順位を示す．したがって，ターゲットウィンドウ切替えが終了すると，切り替え後の TW は自動的に $h=1$ になり，切り替え前の TW は $h=2$ となる．切り替え前の TW における NTW は，その相対的な優先順位を保ったまま $h=3$ 以降に順次変更になる．しかし，本システムにおける各ウィンドウには，ウィンドウ番号として作成から消去までの間それぞれ固有の番号が割り当てられており，ここで定義する h はターゲットウィンドウ切替え手順を説明するため仮想番号であることに注意する．

また，BM_{TW} 及び FM_{TW} の読み出し開始アドレスは，BM_{NTW} 及び NTW_{NUM} のそれに対して相対的に異なる可能性がある．そこで，$R^{C(h)}$，$R^{V(h)}$，$R^{B(h)}$ および

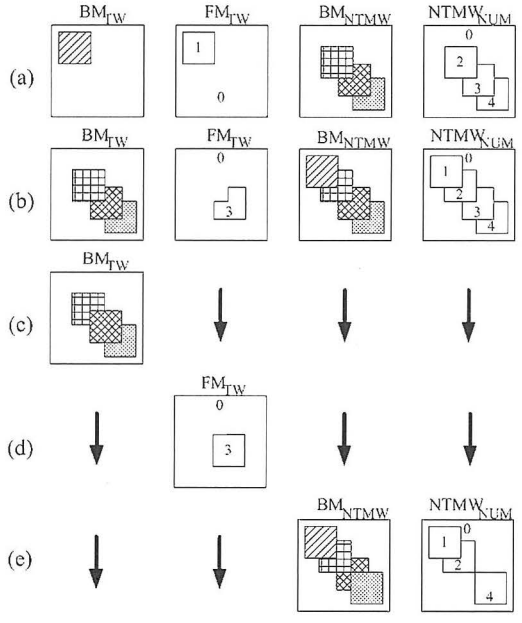

図5 TW の切り替え過程における各メモリの内容の例

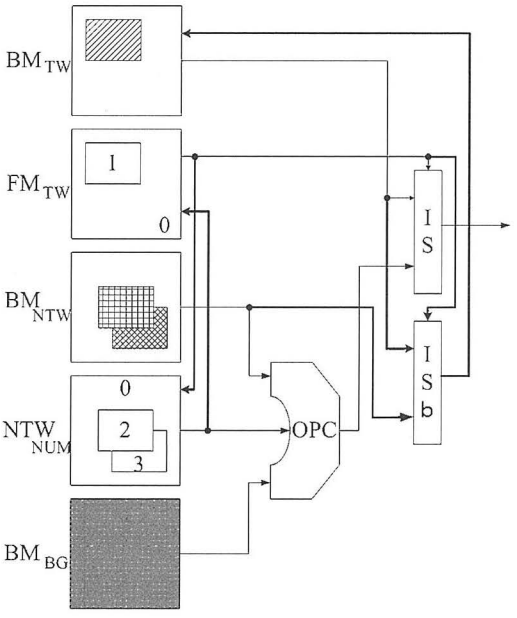

図6 TW の切り替え機能を付加したハードウェア構造

$R^{A(h)}$の領域の位置関係を明確にするため，切り替え前のBM_{NTW}における領域を基準とする．

3．2 切り替え手順

TWの切り替え手順を以下に示す．

(1) 切り替えの要求が発生した直後のフレーム走査期間（以下，走査期間という）に，DMAにより（1-1）から（1-3）を並列に実行する．

（1-1）輝度の低下や透視化を施していないマルチウィンドウ画像をフレーム走査にしたがってBM_{TW}に上書きする．

（1-2）FM_{TW}から読み出される領域のうち，$R^{C(i)}$の領域をNTW_{NUM}に上書きする．

（1-3）NTW_{NUM}から読み出される領域のうち，$R^{B(j)}$の領域をFM_{TW}に上書きし，その他の領域には0を上書きする．

(2) 手順(1)の終了直後の垂直帰線期間にBM_{TW}及びBM_{NTW}を交換し，$O_{IS}=i_{TW}$となるようにISの制御を固定する．

(3) 手順(2)終了後に，（3-1）から（3-3）を実行する．ただし，実行順序に制約はない．

（3-1）$R^{C(j)}-R^{B(j)}$の画像をBM_{TW}に上書きする．

（3-2）$R^{C(j)}-R^{B(j)}$の情報をFM_{TW}に上書きする．

（3-3）T^jの情報を書き換える．ただし，TW切り替えにともなってすべてのウィンドウ輝度を元に戻す場合は，$T^h(1 \leqq h \leqq n)$の情報をクリアする．

(4) 手順(3)終了直後の垂直帰線期間に，ISの制御を通常の制御に戻す．

(5) $I^{A(h)}$ $(j \leqq h \leqq n)$の画像の画素のうち，$R^{A(h)}(j \leqq h \leqq n)-R^{B(h)}(j \leqq h \leqq n)$の画素と，背景画像の画素のうち$R^{B(j)}-\bigcup_{1 \leqq h \leqq n;h \neq j}R^{A(h)}$の画像とを$BM_{NTW}$上の対応する領域に上書きする．

(6) $I^{A(h)}(j \leqq h \leqq n)$のウィンドウ番号情報のうち，$R^{A(h)}(j \leqq h \leqq n)-R^{B(h)}(j \leqq h \leqq n)$の情報と，背景番号情報のうち$R^{B(j)}-\bigcup_{1 \leqq h \leqq n;h \neq j}R^{A(h)}$の番号情報とを$NTW_{NUM}$上の対応する領域に上書きする．

ここで，TWの切り替え過程における各メモリの内容の一例を図5に示す．ここで，TWがウィンドウ番号1から3に切り替わる過程について，同図(b)は手順(1)が完了した時点，(c)は手順(3-1)が完了した時点，(d)が手順（3-2）が完了した時点，(e)が手順(5)と(6)が完了した時点での各メモリの内容を示している．同図中の矢印は変化がないことを示している．

3．3 切り替え手順の実現

TWの切り替え機能を付加したシステムの構造を図6に示す．TWの切り替えのために付加した部分を太線で示している．ISbは2-1マルチプレクサであり，輝度の低下や透視化を施さないマルチウィンドウ画像を生成しており，これにより，3．2の手順（1-1）が実現される．また，FM_{TW}からNTW_{NUM}への布線およびNTW_{NUM}からFM_{TW}への布線は，それぞれ，手順（1-2）および（1-3）を実現している．

表1　試作システムの概要

構成要素	使用パーツ
FPGA	ALTERA EPF 10K50 RC 240-4
使用ゲート数	約42000
PCI Interface	PLX PCI 9052
Bitmap Memory	三菱 M5M482256J-8×2

4．試作システムに対する評価

本方式に基づいたハードウェアを，FPGAとビットマップメモリを使用してPCIボードの作成を行い，これをWindows NT 4.0上のパーソナルコンピュータに実装した試作システムを使用して，操作対象外のウィンドウ透視化処理に必要な実行時間の比較を行った．比較対象としてはWindows系OSで採用されている描画時合成方式とする．試作システムの概要を表1に示す．また，試作システムの概観を図7に示す．

512×480の画像の輝度を任意に低下させ，その画像が画面出力されるまでの処理時間について，2種のパーソナルコンピュータを用いて測定を行った．この測定のために使用したコンピュータのスペックを表2に示す．また，結果を表3に示す．

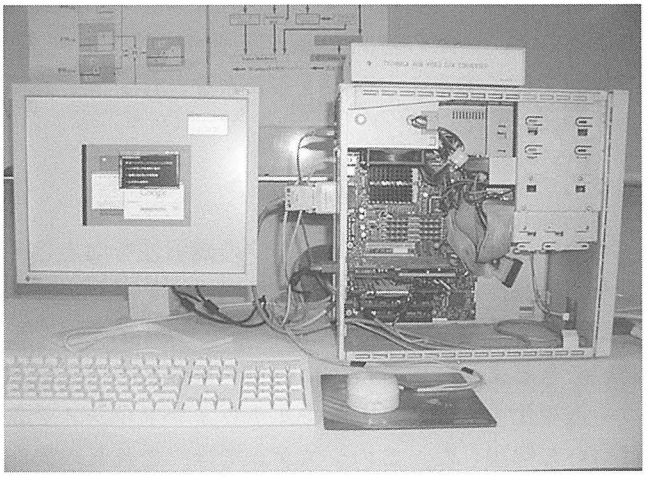

図7　試作システムの概観

表2 使用マシンスペック

マシン	CPU	MEMORY
A	Intel Pen-4 2.53 GHz	2 GB
B	Intel Pen-Pro 200 MHz	96 MB

表3 各合成方式のウィンドウ透視化処理時間

合成方式	マシンA [ms]	マシンB [ms]
従来方式	1093	8132
本方式	16.6 (max)	16.6 (max)
性能比	103.1	767.2

表3中の max は，処理に必要な最大時間を表す．表3からわかるように，操作対象外のウィンドウの輝度を低下させるために必要な処理時間は，本方式を用いることにより，劇的に高速化することができると言える．描画時合成方式では，輝度低下を行うためのウィンドウに対して，1画素毎の画像演算処理が必要となる．これをすべてソフトウェアにより演算を行う必要があるため，その演算量が多くなり，現在の高スペックなパーソナルコンピュータを用いても多大な時間を要する．また，これらの演算は，透視化率の変化に伴って演算処理を逐次行う必要がある．

一方，本方式の合成原理におけるウィンドウ操作対象外の透視化処理は，**2.3** で述べたように，すべてハードウェアを用いて行われるため，CPU 等に負担をかけることなく，高速に処理を行うことができる．システムからの輝度低下の要求信号をハードウェアが検知すると，図3の OPC 内部の乗率テーブルを更新し，次のフレームの開始から輝度を低下したウィンドウ画像が出力される．これにより，コンピュータの演算能力によらず，一定の時間 (max 16.6ms：1フレーム) で処理を終了することができることを試作システムにより確認をしている．

5. あとがき

本稿では，操作対象ウィンドウの識別性を向上させるために操作対象ウィンドウ以外のウィンドウの輝度を低下させる機能と，壁紙に配置した GUI アイコンを認識できるようにするために操作対象以外のウィンドウを透視化する機能をもつマルチウィンドウシステムについて解説した．このシステムでは，これらの機能をハードウェア的に実現するとともに，もとの画像を保持しているので，操作対象ウィンドウ，輝度の低下率等を変更処理を CPU の負担を極力抑えつつ，高速に実行できる．

また本方式では，ウィンドウの透視化処理を CPU への負荷を掛けることなく実行することができるため，CPU 負荷の高い医療用動画像等を表示している状態においてもその動画像ウィンドウに対する透視化処理を行うことができる．先述した動画像解析アプリケーションを本システムに導入することにより，リアルタイムに動画像上に解析結果を重ねることができ，より解析能力に優れたシステムの実現ができる可能性があることを示唆している．

現在筆者等は，本マルチウィンドウ合成方式にさらなる改良を加え，操作対象ウィンドウと操作対象以外のウィンドウ同士の透視化等，様々な画面構成が可能なマルチウィンドウ合成方式の考案を行っている．また，本方式を Linux ベース上のパーソナルコンピュータに実装し，本方式による CPU 及び周辺ハードウェアにかかる負荷の割合等の詳細な動作の検証を行う予定である．

参考文献

1) 宮里 勉：“マルチウィンドウ表示装置”，日本国特許庁，公開特許公報(A)，特許公開2001—147760，Apr. 2001.
2) 本田政則 他：“マルチウィンドウ表示方法および装置”，日本国特許庁，公開特許公報(A)，特許公開平7—104724，May. 1995.
3) 株式会社ライブラリー：“2/3次元運動解析システム Carrot”，型式 LB 640 WD 3S，http：//www.library-inc.co.jp
4) T. Brunhoff：“Pleasing The eye”, UNIX REVIEW, vol 7，no. 10，pp. 65—72，Oct. 1990.
5) B.A. Myers：“A complete and efficient implementation of covered Windows”, IEEE Comp., vol. 19，no. 9，pp. 57-67，Sept. 1986.
6) 中村太一，菅原昌平，前嶋 司：“マルチウィンドウ表示装置”，信学論(D)，vol. J71-D，no. 10，pp. 2069—2077，Oct. 1988.
7) 前嶋 司，中村太一，菅原昌平：“マルチウィンドウシステムにおける仮想化機能の高性能化”，信学論(D-I)，vol. J72-D-I，no. 9，pp. 642—651，Sept. 1989.
8) 佐藤洋一郎，横平徳美，籠谷裕人，岡本卓爾，茅野 功：“描画時合成方式と表示時合成方式の併用によるスムーズ操作が可能なマルチウィンドウシステム”，信学論(D-I)，vol. J86-D-I，no. 9，pp. 650—660，Sept. 2003.
9) 茅野 功，田邊勝也：“操作対象ウィンドウの識別性を向上したマルチウィンドウシステム”，第5回 IEEE HISS 論文集，pp. 164—167，Dec. 2003.