形状簡略化による3次元形状モデルの描画速度制御の一技法

新藤義昭† 片山滋友†
坂本康治† 松田郁夫†

3次元形状モデルをコンピュータグラフィックス技術によって可視化する際，計算機の性能に合わせて自動的に形状簡略化を行い，要求された描画時間を可能な限り保持する描画技法を提案する．形状モデルを基本立体プリミティブを組み合わせて構築する際，高品質メンバ，中品質メンバ，低品質メンバ，概略化メンバという四つのメンバを自動的に作成する．描画速度を高速化するため，低品質メンバは，あらかじめ設定された複数の視線候補ベクトルによる陰面消去を事前に行っておき，描画時には視線ベクトルと最も近い視線候補ベクトルを選択して可視面を抽出する．さらに概略化メンバは，3次元形状モデルの可視断面を2次元平面で近似する．各プリミティブは，メンバ決定用特徴量として3次元特徴量をもつ．描画の時点では，まず三つの特徴量を投影変換し，最長の長さを求める．この長さを，メンバしきい値テーブルと比較して，どの形状メンバを用いるかを決定する．描画ごとに実際の描画時間を計測し，要求された描画時間との差分に応じてメンバしきい値テーブルを切り替える．このフィードバック制御により，要求された描画速度を保持することができる．描画時間の変動を抑えるため，新たに視界に出現するプリミティブを直前に検出し，次回の描画時間を予測してメンバしきい値テーブルを素早く切り替える．

Display Method of 3-D Object for Keeping Display Time by Shape Simplification

Yoshiaki Shindo,† Shigetomo Katayama,† Koji Sakamoto† and Ikuo Matsuda†

We propose a display method for keeping user-specified time during visualization of 3-D objects, which is based on the automatic shape simplification according to machine performance. When modeling the object by using geometric primitives, 4 members are generated automatically, that are called High-Quality member, Middle-Quality member, Low-Quality member and Rough member. For saving display time of Low-Quality member, hidden surfaces from pre-defined view point vectors list are removed in advance, and only possibly visible surfaces are extracted by selecting the nearest vector to real view point vector from the list. Rough member is 2-D sections list approximating 3-D primitives. Each primitive includes 3-D identifying values which would be converted and transformed to the viewing position. Appropriate member is selected by comparing the maximum value among them with threshold table. Keeping display time is achieved by automatic selection of the appropriate threshold table, which is controlled by a feedback loop of the difference between target display time and previous display time. To restrain the fluctuation of display time, we propose a predictive feedback algorithm which detects newly-appeared primitives immediately and predicts the next display time.

1. はじめに

3次元形状モデルをコンピュータグラフィックス技術を用いて可視化することが各種の分野で盛んに行われている．初期のシステムでは，高品質の静止画像を時間をかけて描画することが主流であったが，近年のグラフィックスワークステーションの性能向上によって対話型の高速描画が可能になった[1)~4)]．一方，ワークステーションの互換性がかなり確保されたことから，同一仕様のワークステーションの性能格差が広がりはじめている．この性能格差が，可視化アプリケーションの描画速度に影響を与えるのは具合が悪いことが多い．視点の移動や形状モデルの編集などにおいては，描画速度はシステムの対話応答性を決定する要素であるため，操作性に大きな影響を与える．また，実時間の動画表示も行われているが，動画の場合は，映像品質を多少犠牲にしても描画速度を保持する必要がある．

† 日本工業大学工学部情報工学科
Department of Computer and Information Engineering, Nippon Institute of Technology

3次元形状モデルの構築や，アプリケーション開発に多大なコストがかかる点を考慮すると，動作環境の性能に個別対応した可視化アプリケーション開発は不経済である．ソフトウェア面で工夫を施し，動作環境の性能格差が描画速度ではなく，画像品質に反映するような描画技法が望ましい．すなわち，ワークステーションの性能を計測しながら，求める描画速度に近づくように画像品質を自動制御する必要がある．描画速度と画像品質のトレードオフについては，従来より形状モデルを簡略化する方法が提案されている[5),6)]．

文献5)には，視点と物体との距離に応じて形状簡略化を行いながら描画速度制御を行う方法が提案されている．この技法は，広大な空間に離散的に配置された建築物に対して，視点と物体との距離を簡略化の基準とし，多角柱の底面の画数を増減するものである．この簡略化距離パラメータを変更することにより描画速度を自動制御する方法を提案している．この方法では，遠方にある物体は，たとえ大きな物体でも簡略化されてしまうので，形状モデルの種類によっては画像の劣化が大きいと思われる．また，立体プリミティブとして多角柱しか対象としていない．

文献6)には，視線の向きによる面のグルーピング法，空間分割による立体のグルーピング法，形状の段階的簡略化法を併用した描画速度の高速化について述べられている．この技法では，B-Repsを用いた基本立体プリミティブに対して，一つの特徴量を与え形状簡略化の基準としている．さらに，視線ベクトルと面の法線ベクトルのクラス分けによる粗い陰面消去法を用いた簡略化が提案されている．しかし，3次元立体に対する特徴量が1次元の量なので，視点の高度や向きによる見え方の変化が考慮されていない．また描画速度の自動制御については言及されていない．

描画速度の自動制御については，**フィードバック方式**と**予測方式**（Predictive Algorithm）が知られている．文献5)で提案されている方式はフィードバック方式の一つである．予測方式は，事前にベンチマークテストで計算機の性能を測定した結果の統計値を用いて，描画前に最適な簡略化形状の組み合わせを求める方式である．文献7)は両方式の性能比較について述べたうえでフィードバック方式の欠点を指摘し，予測方式の優位性を主張している．しかし，適用範囲を「予測処理と描画処理を2個のCPUで並行処理する動作環境」に限っていること，計算機の動的な性能変動（走行タスク数の変化やネットワークからのアクセスなどによって引き起こされる，アプリケーション側からみた実質的な計算機性能の変動）を考慮していない点に問題が残る．また自動的な形状簡略化の方法や，比較対象としたフィードバック方式の簡略化形状の選択方法に関して詳しく検討していない．

本論文では，可視化対象物を建築物などの景観シミュレーションに特定せず，描画速度を保持するためのより適用範囲の広い形状簡略化法を提案する．ここで適用範囲とは，ビジネス，教育，プレゼンテーション，マンマシンインタフェース，さらには人工現実感システムなどのマルチメディアを支える基幹技術の一つとしてのコンピュータグラフィックス[4)]であり，映画や放送における特撮，芸術作品の創作などは対象としない．

3次元形状モデルの可視化アプリケーションの開発を容易にするためには，3次元形状モデルのモデリングから描画までを一貫してプログラムライブラリ化し，C言語等のプログラム開発環境からAPI☆関数として呼び出せる構造が望ましい．基本立体プリミティブによる形状モデルのモデリング，陰面消去，自動陰影付けだけではなく，形状簡略化，描画速度の自動制御などの処理もアプリケーションから隠蔽する方が効果的である．形状簡略化と描画速度の自動制御は複雑なプログラミングを要するので，アプリケーションプログラマに大きな負担がかかり，本来のアプリケーション開発に専念できない恐れがあるからである．従来から3次元グラフィックスライブラリとして，GKS-3D，PHIGS，PEX，OpenGLなどのライブラリが提案されているが[4),8)~11)]，形状簡略化と描画速度の自動制御を組み込んだものは提案されていない．

そこで，本研究では，形状簡略化と描画速度の自動制御を含んだ3次元グラフィックスライブラリを開発した．3次元形状モデルの表現法としては，サーフェイスポリゴン法を用いた．これは，グラフィックスワークステーションに装備されている高速描画ハードウェアとの親和性が最も高いからである[4)]．プリミティブとしてはポリゴン，基本立体などが考えられる．ポリゴンの集合体として記述する方法は，きめの細かい形状記述が可能な反面，記述データ量が膨大なためモデリングコストがかかること，対話形式で形状モデルを編集するシステムに適用しにくい，などの問題がある．球，円柱などの基本的な3次元立体を伸長，収縮，回転しながら組み合わせて記述する方法は，比較的少ないデータ量で複雑な形状を記述できること，対話形式による編集システムに適用しやすいなどの利点がある．

☆ Application Program Interfaceの頭文字．OS等のサービスを受けるための関数を意味する．

この方法は生成画像に高い芸術性を求めない限り，有効な手法の一つであると考えられる．

OpenGL においても，ポリゴンによる記述を基本機能としながらも，基本立体をプリミティブとする GLU (OpenGL Utility Library)，Open Inventor などが開発されている[11]．

このような観点から，本研究では，基本立体プリミティブとして，曲面を伴う5個の立体と五つの多面体，合わせて10個の基本立体を用いることとした．これらを組み合わせて形状モデルを作成する際，自動的に簡略化された形状メンバを作成する．形状メンバの構造については2章で述べる．

形状簡略化の方法として，3章では，あらかじめ設定された複数の視線候補ベクトルを用いて可視可能性の高い面だけを抽出する技法について，4章では，3次元形状モデルの可視断面を2次元平面で近似する技法ついて述べる．5章では形状メンバを選択する際の基準となる3次元特徴量について述べる．6章では描画速度の自動制御を行う技法について述べる．

2. 形状簡略化メンバの構造

アプリケーションプログラマは，基本立体プリミティブを回転，収縮，伸長しながら組み合わせて，3次元形状モデルを構築する．プリミティブとしては，曲面を伴う5個の立体（球，半球，円錐，円柱，円錐柱）と，五つの多面体（立方体，直方体，四角錘，三角錘，台形）を用いる．

曲面を伴う立体は，多数の微小な平面群によって多面体で近似する（**曲面近似面数**と呼ぶ）．この際，4種類の形状データメンバが自動的に作成される．それぞれ，**高品質メンバ**（メンバ0），**中品質メンバ**（メンバ1），**低品質メンバ**（メンバ2），**概略化メンバ**（メンバ3）と呼ぶ．

高品質メンバは簡略化を行わない形状メンバである．中品質メンバは曲面を伴う立体の曲面近似面数を少なくしたものである．この二つのメンバの差は，輪郭部分のエイリアスの差となってあらわれる．したがって曲面を伴わない多面体では差異はない．この二つのメンバは，曲面のスムースシェーディングのために，各頂点の曲面方向の法線ベクトル値をもつ．

低品質メンバは，あらかじめ設定された複数の視線候補ベクトルを用いて，事前に粗い陰面消去を施し，描画時には視線ベクトルと最も近い視線候補ベクトルを選択することによって，可視可能性の高い平面だけを抽出する．このメンバは，曲面のない多面体にも適用される．高品質メンバ，中品質メンバ，低品質メンバの曲面近似面数は，アプリケーションプログラマが任意に設定することができる．

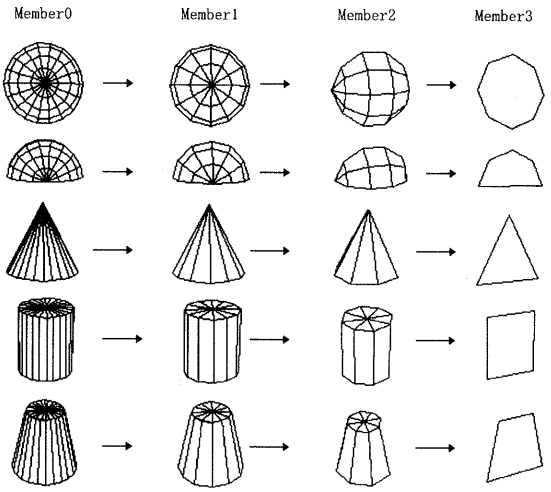


図1 形状簡略化メンバの構造
Fig. 1 Simplification members of primitives.

概略化メンバは，3次元形状モデルの可視断面を2次元平面で近似したものである．曲面を伴う五つの基本立体プリミティブの形状変化を**図1**に示す．

3. 視線候補ベクトルを用いた描画時間の短縮（低品質メンバ）

形状モデルの品質を落として描画時間の短縮を図るため，基本立体の重心点を原点とする直交座標系に，あらかじめ任意の視点候補を設定する．この座標系の原点から，視点候補へ向かう3次元のベクトルを**視線候補ベクトル**と呼ぶ．

視点候補の数および位置はアプリケーションプログラマが自由に設定できるが，明示宣言の無い場合には，**図2**に示すような18個の視点候補が自動的に設定される．

設定された n 個の視線候補ベクトルの配列 E（**視線候補ベクトル配列**と呼ぶ）を式 (1) のように表すことにする．

$$E = \{\vec{E}_0, \vec{E}_1, \vec{E}_2, \ldots, \vec{E}_{n-1}\} \tag{1}$$

立体を構成する平面群を作成する際，平面の法線ベクトル $\vec{S}$ と視線候補ベクトル配列の各要素との内積値を式 (2) によって求め，内積値配列 M を作る．ただし視線候補ベクトル配列の i 番目の要素を $\vec{E}[i]$，内積値配列の i 番目の要素を $M[i]$ と表す．

$$M[i] = \vec{S} \bullet \vec{E}[i] \tag{2}$$

ただし，$i = 0 \sim n-1$

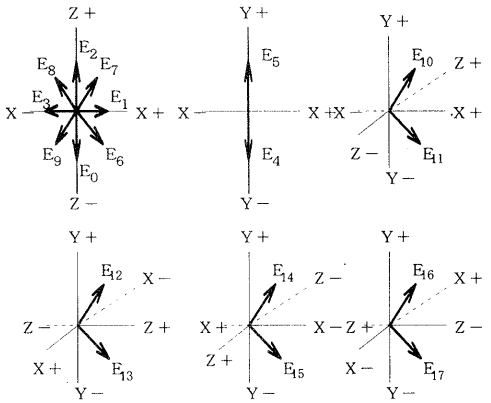


図2 視点候補と視線候補ベクトルの初期値
Fig. 2 Pre-defined view point vectors list.

この内積値配列 M を各平面に記憶する．この内積値は，その平面が視点候補から可視かどうかを表す指標となる．低品質メンバを描画する際には，次に述べる処理手順で可視可能性の高い平面だけを抽出し，陰面消去アルゴリズムの負担を軽減する．

(1) 基本立体プリミティブの重心点から視点に向かう視線ベクトル $\vec{P}$ を求める．

(2) 視線候補ベクトル配列 E の各要素を描画時の座標系にアフィン変換して，ベクトル配列 F を作る．この配列を式 (3) のように表し，i 番目の要素を $F[i]$ と表す．

$$F = \{\vec{F}_0, \vec{F}_1, \vec{F}_2, \ldots \vec{F}_{n-1}\} \qquad (3)$$

(3) 視線ベクトル $\vec{P}$ と F の各要素との内積値配列 V を式 (4) によって計算する．

$$V[i] = \vec{P} \bullet \vec{F}[i] \qquad (4)$$
ただし，$i = 0 \sim n-1$

(4) $V[i]$ の値が最も大きい値となる i を求める．この i は，視線ベクトルに最も近い視線候補ベクトルの番号を表す．

(5) 形状メンバから平面を抽出する際，式 (5) が成立する時には描画対象からはずす．

$$M[i] < 0 \qquad (5)$$
内積値配列の値は，事前に計算済みなので，陰面可能性の高い面を高速消去することができる．

視線ベクトルと選択された視線候補ベクトルは完全に一致しているわけではないので，描画時には輪郭部の可視平面が欠落する恐れがある．また，透視投影の影響も考慮していない．しかし，不要な面が陰面消去アルゴリズムに入力されないため，描画速度の向上に寄与する．

4. 3次元形状の可視断面による近似（概略化メンバ）

概略化メンバは，3次元形状の可視断面を2次元平面で近似するものである．多数の微小平面を保持する立体が，単一平面で近似されるため描画速度向上に大きく寄与するが，画像の劣化も大きいため，微細化した物体の描画に使用する．

3次元形状の可視断面は，その物体を眺める視点の位置によって変化するので，単一の断面平面データを与えるだけでは，視点位置の変化によって断面形状が大きく狂ってしまう．たとえば，円錐の場合，上または下から見た場合には円に見え，横から見た場合には三角形に見える必要がある．そこで，各基本立体プリミティブごとに可視断面候補を自動的に作成する．**図3**に，曲面を伴う五つの基本立体についての可視断面候補を示す．立方体，直方体などの曲面を伴わない基本立体についても，同様に可視断面候補を作成する．

描画時には，可視断面候補の中から1面だけを選択して描画対象とする．このため，3章で述べた視点候補（図2）のうち，$E_0 \sim E_5$ までの6個の視点候補を設定し，断面候補ごとに，法線ベクトルと視線候補ベクトル配列の内積値配列Mを計算した後，次の処理を行って，1面のみ可視とする．

(1) $M[i]$ が最も大きい値となる i を求める．

(2) i 以外の要素にすべて -1（背面を表す）を代入する．

この結果，低品質メンバと同様の描画アルゴリズム（3章 (1)〜(5)）で，断面候補の中から1面だけを選択することができる．

5. 形状メンバの選択法

2章で述べたように，各基本立体プリミティブには，四つの形状メンバが自動作成される．描画時に各立体の形状メンバを自動的に選択するために，XYZ方向の3次元特徴量が設定される．各プリミティブに設定される3次元特徴量を**表1**に，プリミティブのデータ構造を**図4**に示す．

描画時にメンバを選択する方法を次に述べる．

(1) 3次元特徴量をアフィン変換して透視投影し，画面への投影量（画素数）を求める．

(2) 投影量の最大値 Vmax を求める．

(3) この値をメンバしきい値テーブル T と比較する．メンバしきい値テーブル T は，3個の数値要素をもつ配列であり，式 (6) のよう

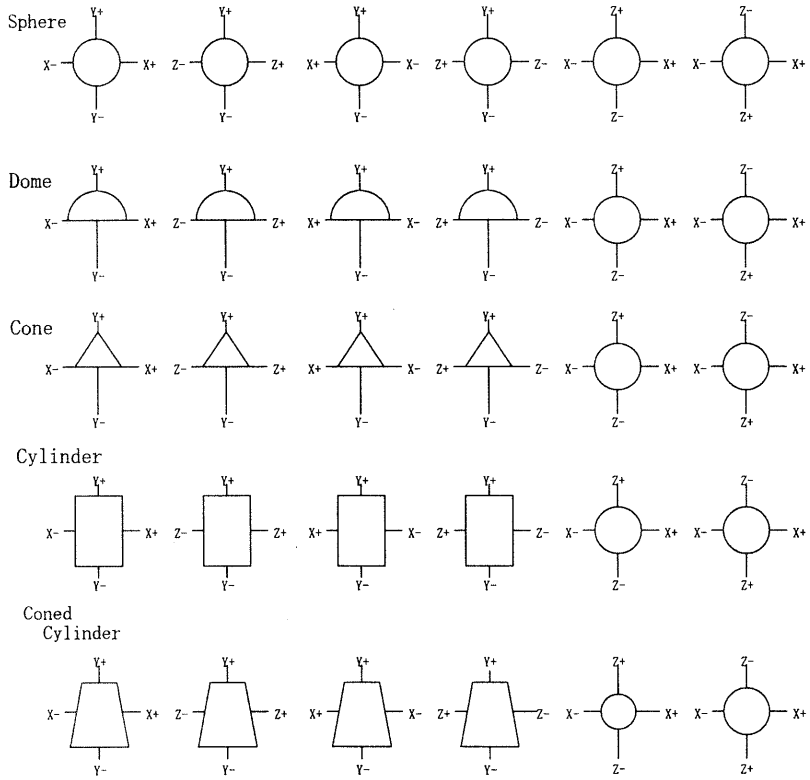


図3 可視断面候補
Fig. 3 Pre-defined visible sections list.

に表す.

$$T = \{T_h, T_n, T_a\} \qquad (6)$$

ただし, $T_h \geq T_n \geq T_a$

特徴量の最大値 Vmax とメンバしきい値テーブルの要素を次のように比較してメンバを決定する.

表1 基本立体プリミティブの3次元特徴量
Table 1 Identifying value of primitives.

立体	X 特徴量	Y 特徴量	Z 特徴量
球	球の直径	球の直径	球の直径
半球	半球の直径	半球の直径	半球の直径
円錐	底面円の直径	高さの半分	底面円の直径
円柱	底面円の直径	高さの半分	底面円の直径
円錐柱	底面円または上面円の直径（大きい方）	高さの半分	底面円または上面円の直径（大きい方）
立方体	1辺の長さ	1辺の長さ	1辺の長さ
直方体	底辺の長さ	高さの半分	底辺の長さ
四角錘	底辺の長さ	高さの半分	底辺の長さ
三角錘	底面三角形の底辺の長さ	高さの半分	底辺三角形の高さ
台形	上面または底面の辺の長さ（大きい方）	高さの半分	上面または底面の辺の長さ（大きい方）

if	(Vmax > T_h)	Member = 0
else if	(Vmax > T_n)	Member = 1
else if	(Vmax > T_a)	Member = 2
else		Member = 3

このメンバしきい値テーブルの値を変更すると，描画速度と画像品質をトレードオフすることができる.

この方法は，単に視点からの距離によってメンバが決定されるわけではないので，遠方の物体でも大きなものは簡略化されない．画面への投影量が小さいものから順に段階的に簡略化される.

6. 描画速度の自動制御

多くの対話型可視化システムの場合，画像品質を優先させたい場合と，描画速度を優先させたい場合がある．画像品質を優先する場合には，5章で述べたしきい値テーブルを1個だけシステムに与える．これにより，求める形状品質の画像が得られる.

描画速度を優先させたい場合には，フィードバック方式による描画速度制御を行う．フィードバック方式は，予測方式に対して以下のような利点がある.

(1) 計算機の描画アーキテクチャに依存しない.

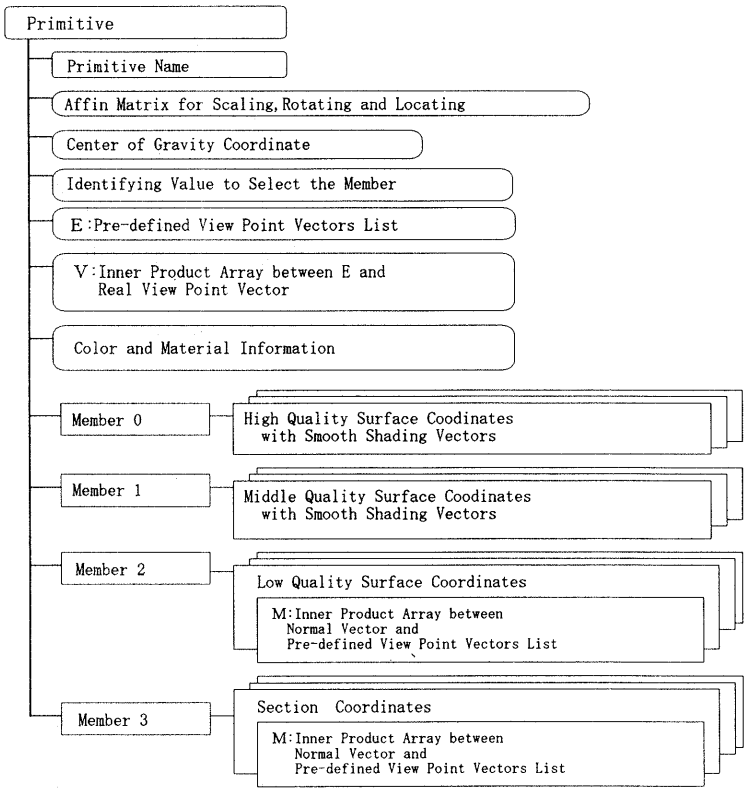


図4 基本立体プリミティブのデータ構造
Fig. 4 Data structure of primitives.

(2) 計算機の動作環境の変動による実質的な性能変動にも対応する.

しかし, 文献7) に指摘されている通り, 初めて視界に入った形状モデルに対する追従性能が悪いという欠点がある. これに対して, 予測方式は, 計算機の性能ベンチマークテストを行った時点での統計値を基礎データとして用いるため, 動作環境の影響を受けやすい. また, 組み合わせ問題の最適解を求める予測処理を行うため, 予測処理と描画処理が並行動作している環境でなければ描画時間の増大を招く.

本研究では, グラフィックスライブラリの適用範囲を広げるという観点から, フィードバック方式を基本モデルとする描画速度制御方式を開発した. まず, 単純なフィードバック方式に基づくモデルを用いて, 描画速度制御のアルゴリズムを述べる.

描画速度を優先する場合には, アプリケーションプログラマは要求描画速度と速度制御テーブルをシステムに与える. 速度制御テーブルは, 複数のしきい値テーブルを要素とする配列で, 式 (7) のように表す. 速度制御テーブルの構造を図5に示す.

$$TS = \{T_0, T_1, T_2, \ldots T_{n-1}\} \tag{7}$$

$$TS = \{T_0, T_1, T_2, \ldots\ldots T_{n-1}\}$$

$$T_i = \{T_{hi}, T_{ni}, T_{ai}\} \quad i = 0 \sim n-1$$

ただし $T_{hi} \leq T_{hj} (i < j)$
$T_{ni} \leq T_{nj} (i < j)$
$T_{ai} \leq T_{aj} (i < j)$

図5 速度制御テーブルの構造
Fig. 5 Structure of speed control table.

T_0 は最も高品質の画像が得られるように, T_{h0}, T_{n0}, T_{a0} には小さな値を設定する. T_1 は, これよりも少し大きな値を入れる. 以後, 同様に少しずつ大きな値を設定する. 描画速度を優先するモードになったときには, 速度制御テーブルの中の, どのしきい値テーブルを使用して形状メンバを決定するかを示すしきい値ポインタ P を用いる. しきい値ポインタの初期値は, $P = 0$, すなわち速度制御テーブルの先頭に格納されたしきい値テーブル T_0 をさし示す.

描画を行った際に実際に要した時間を計測する. この値を TRとする. 要求された描画時間を TU とする. 次に示すアルゴリズムに従って, TR と TU の差が TU の20%以上ある場合には, しきい値ポインタ

P を移動する．ただし，abs() 関数は絶対値を表す．

```
if (abs (TU - TR) > (TU/5)) {
// 変動率20%以上
  if (TR > TU) {
  // しきい値ポインタを進める
  p = p + 1;
  if (p > n) p = n;
  }
else {
  // しきい値ポインタを戻す
  p = p - 1;
  if (p < 0) p = 0;
  }
}
```

この結果，要求描画時間に近づくようにしきい値ポインタが移動し，最適なしきい値テーブルが自動的に選択される．

次に，速度制御テーブルの値の設定方法について述べる．速度制御テーブルの要素数や値の設定は形状モデルやアプリケーションの種類に依存する．最も単純な設定法は，一定の画素量を差分値とした線形テーブルを作成することであるが，無効な要素が多数登録される恐れがある．理想的な方法は，(1) 形状モデルを分析して最適な非線形差分値配列を求める．(2) アプリケーションをクラス分けして，求める速度調整範囲を得るための簡略化メンバの選択率を自動設定する，ことである．しかし，このアルゴリズムは複雑な組み合わせ問題となり，多様なケースに対応できる方法の開発には至っていない．そこで以下のような方法で値の設定を行っている．

(1) 一定の画素量を差分値とした線形のテーブルを作成する．
(2) このテーブルを用いて対象とする形状モデルを描画する．
(3) あらかじめ指定した視点移動シナリオに基づいて描画をくり返し各要素の描画時間と簡略化形状メンバの選択比率の平均値を求める．
(4) この結果によって，無効と思われる要素を手操作で削除する．

(1)～(4) までの処理がアプリケーション側で行えるように，速度制御テーブルの参照および登録，各要素の描画時間と形状メンバ選択率を問い合わせる API 関数を用意した．さらに，速度制御テーブルを作成するための対話型のソフトウェアツールを開発し，作業負担の軽減を図っている．これは，描画時間と簡略化メンバの選択率をグラフ表示しながら速度制御テーブルの編集を行うソフトウェアである．

簡略化メンバの選択比率については，3D 動画によるプレゼンテーションなど，視界近傍に多数の微細なプリミティブが存在する場合には概略化メンバが選択されやすいように，景観ウォークスルーやフライトシミュレータのように広大な空間の一部を投影する場合には，中品質メンバと低品質メンバが選択されやすいように設定すると，速度調整範囲を大きくすることができる．

この方式は，速度制御テーブルの要素数を増やすと，きめの細かい速度制御が可能になるが，新たに出現した形状モデルに対して追従するまで時間がかかる．逆に要素数を減らすと，追従性能は向上するが，速度制御の精度が落ちてしまう．また，単に差分量に応じた比率で速度制御ポインタの移動速度を早めると，ポインタのオーバシュートが頻繁に発生し，描画速度が安定しない．

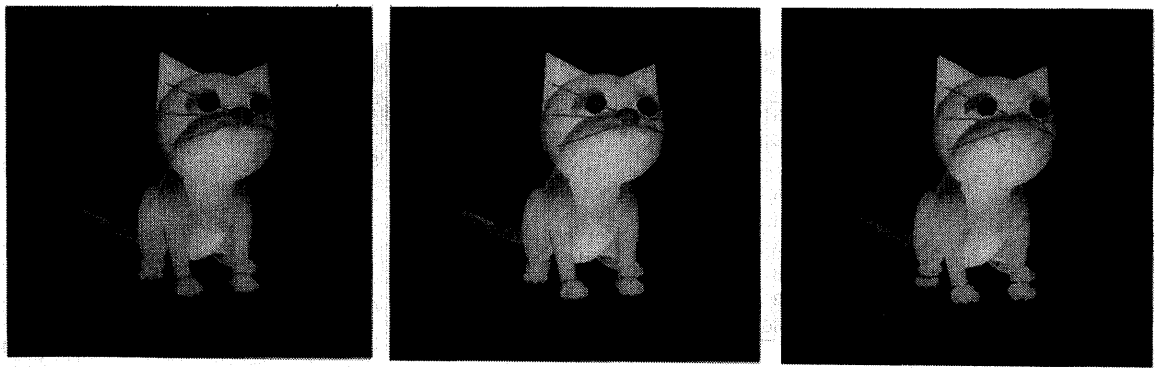
そこで，速度制御の精度を落とさずに追従性能を向上させるため，次回の描画時間の変動を予測し，速度制御ポインタを素早く移動させる予測型フィードバック方式を開発した．以下に予測方法を述べる．

フィードバック方式の速度制御の追従性能が悪化するのは，新たなプリミティブが突然視界に出現し，面数の多い形状メンバが選択された時である．以後，このようなプリミティブを出現プリミティブと呼ぶ．

出現プリミティブは，画面の端の部分から現れることに着目して，出現の瞬間だけ面数の少ない形状メンバで描画し，急激な対象面数の増加を緩和する．画面の端の部分は，操作者が注視している可能性が低いので，簡略化形状を用いることによる影響は少ない．

出現プリミティブは，次回の描画で，より面数の多い形状メンバが選択される可能性が高いことから，面数の差分を求める．視界内のすべての出現プリミティブに対する面数の差分の累積値を求め，次回の描画時間を予測する．出現プリミティブを低品質メンバで描画する場合のアルゴリズムを以下に詳しく述べる．

(1) 描画時に視界クリッピングによって描画対象からはずされたプリミティブにはマーキングしておく．
(2) プリミティブの形状メンバを選択する際，次の条件を満たす場合には，低品質メンバを選択する．
  (a) 前回の描画で選択されていなかった．
  (b) 特徴量によって本来選択されるべきメンバが高品質メンバまたは中品質メンバである．



(a) T_0 の画像 (b) T_4 の画像 (c) T_7 の画像

図6 実際に用いた猫のモデル
Fig. 6 Picture of cat model.

(3) (2)の条件を満たした場合は，本来選択されるべき形状メンバの面数 S_h，低品質メンバの面数 S_l より，面数の差分 S_d を求める．

(4) 描画中に，この差分 S_d の値を累積加算し，次回の描画で増加可能性の高い面の総数を予測する．

(5) 描画終了時の描画時間 TR と面数 S_c より，1面当たりの描画時間 TS を求め，TS と S_d の値より，次回の描画時間 TF を予測する．TF を求める式をまとめると，式(8)のようになる．

$$TF = TR + \frac{TR}{S_c} \times S_d \qquad (8)$$

ただし，$S_d = \sum(S_h - S_l)$

(6) S_d の値が0でない時には，視界内に大きな形状モデルが出現したことを表しているので，この時だけ速度制御ポインタの移動量を増やす．移動量 pstep は次のように設定した．

```
TD = abs (TU - TF)
if       (TD < (TU/4)) pstep=2;//25%
else if  (TD < (TU/2)) pstep=3;//50%
else if  (TD < TU)     pstep=4;//100%
else                   pstep=6;//Upper
```

速度制御ポインタの移動量の設定は経験値であるが，7章の実験結果でも示すとおり，追従性能がはるかに改善する．また，速度制御ポインタの移動量が増えるのは，新たなプリミティブが出現した瞬間だけなので，速度制御ポインタのオーバシュートが頻繁に発生することはない．このしきい値と移動量の値の設定は形状モデルの種類に依存する．現時点では，しきい値と移動量をシステムに登録するAPI関数を用意し，実際に描画を行いながら手操作で設定する仕様としている．

表2 実際に用いた猫の曲面近似面数
Table 2 Number of plane surfaces approximating curved surface.

基本立体	高品質メンバ	中品質メンバ	低品質メンバ
球	128 面体	50 面体	32 面体
半球	80 面体	35 面体	24 面体
円錐	16 角錐	12 角錐	8 角錐
円柱	16 角柱	12 角柱	8 角柱
円錐柱	16 角柱	12 角柱	8 角柱

また，アプリケーションの種類によっては，出現プリミティブの表示を隠蔽したい場合もある．この場合には，視界クリッピング領域を，実際の画面よりも少し大きく設定する．この設定により，出現プリミティブは検出されるが，画面には表示されない．もちろん，クリッピング領域を不必要に大きくすると，描画対象面数が増加するので描画速度が低下するが，出現プリミティブによる描画速度の変動幅を抑えることができる．

7. 実験および考察

実験に使用した動作環境は，Pentium 90MHz Windows/NT Workstation である．開発したライブラリをダイナミックリンクライブラリとしてオペレーティングシステムの中に組み込み，実験プログラムはC++言語で記述した．

7.1 形状簡略化による形状容量と視覚効果の変化

まず，しきい値テーブルの値と可視面数の関係について考察する．実験を行うため，**図6**に示すような猫のモデルを作成した．高品質メンバ，中品質メンバ，低品質メンバに設定した曲面近似面数を**表2**に示す．

しきい値テーブル T の内容が $T = \{1,0,0\}$ の時は，すべての基本立体プリミティブの高品質メンバが

表3 実際に用いたしきい値テーブル
Table 3 Threshold table.

テーブル番号	値
T_0	$T=\{1,0,0\}$
T_1	$T=\{8,4,6\}$
T_2	$T=\{16,8,4\}$
T_3	$T=\{24,16,8\}$
T_4	$T=\{32,24,16\}$
T_5	$T=\{48,32,24\}$
T_6	$T=\{56,48,32\}$
T_7	$T=\{64,52,42\}$
T_8	$T=\{128,80,64\}$

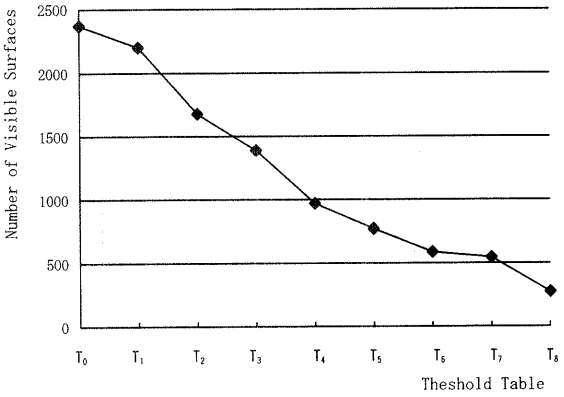


図7 しきい値テーブルと可視面数の関係
Fig. 7 Relation between visible surfaces and threshold table.

選択され，形状簡略化は行われない．図6(a)はこの場合の画像である．

しきい値テーブルの値を**表3**のように変化させた場合の可視面数の変化を**図7**に，形状簡略化メンバの選択状態を**図8**に示す．図7より，可視面数の調整範囲が5倍程度あることがわかる．図6(b)は$T=\{32,24,16\}$，図6(c)は$T=\{64,52,42\}$の場合の画像である．

画像の劣化を定量化して評価するために，形状保持率を求める．ここで，形状保持率とは，形状簡略化を行わない場合の画像を基本画像として，簡略化画像の変化しなかった画素数の割合を求めたものである．

ただし，基本画像の全画素に対する割合ではなく，基本画像の中に表示された形状モデルの画素を全体数とする．すなわち，基本画像の中に表示された形状モデルの画素数をN_a，対応する簡略化画像の画素のうちで値の異なる画素の個数を数えてN_sとすると，形状保持率Gは式(9)で表される．

$$G=(1-\frac{N_s}{N_a})\times 100 \qquad (9)$$

この式を用いて，T_0の画像を基本画像とし，T_1からT_8までの簡略化画像の形状保持率を求めたグラフを**図9**に示す．

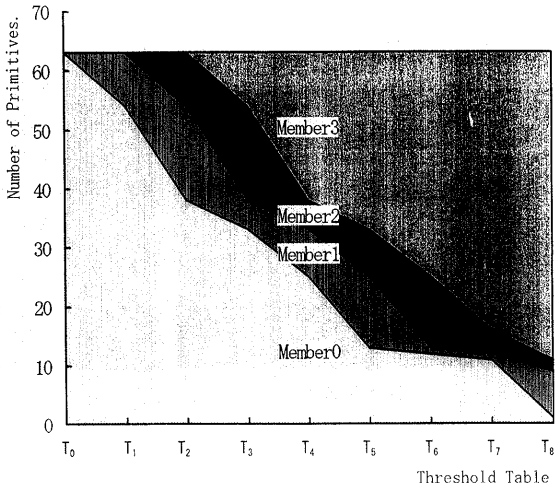


図8 形状簡略化メンバの選択状態
Fig. 8 Relation between selected members and threshold table.

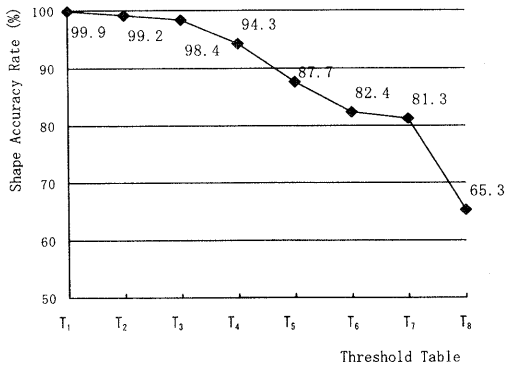


図9 簡略化画像の形状保持率
Fig. 9 Relation between shape accuracy rate and threshold table.

7.2 視線候補ベクトルによる低品質メンバの画像

7.1節の実験では，微細化された立体のみが低品質メンバとして選択されているので，視線候補ベクトルを用いた低品質メンバの画像を評価しにくい．そこで，しきい値テーブルに$T=\{1024,1024,0\}$という値を設定して低品質メンバのみで描画した画像を**図10**に示す．輪郭部の平面の欠落を観測するため，図1で示した視線候補うち，$E_6\sim E_9$の4点を削除し，14本の視線候補ベクトルを設定した．画像では，猫の腹部の輪郭平面が一部欠落しているのが観測できる．これは，視線ベクトルと選択した視線候補ベクトルのずれから生じたものである．低品質メンバは曲面のスムースシェーディングベクトルを保持していないので，多

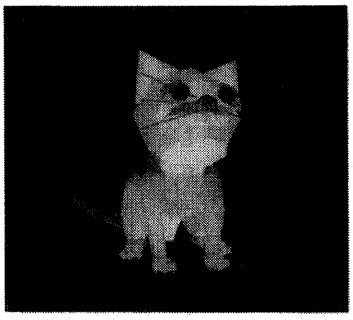


図10 低品質メンバによる画像
Fig. 10 Picture of Member 2.

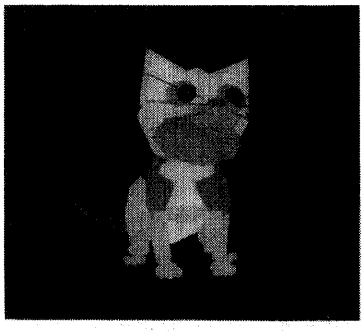


図11 概略化メンバによる画像
Fig. 11 Picture of Member 3.

面体がフラットシェーディングされる．このため，単に可視面数の減少だけではなく自動陰影付けの負荷も軽減されるため，描画の高速化が期待できるが，形状品質は大幅に低下する．この時の可視面数は596，形状保持率は42.4%であった．可視面数は形状を簡略化しない場合に比べて26%に削減されている．

7.3 概略化メンバによる画像

すべての基本立体が概略化された場合の画像は，立体が断面平面化してしまうので，3次元表示とはいいがたい．しかし，その画像を評価するために，あえてしきい値テーブル $T = \{2048, 2048, 2048\}$ を与えて描画した結果を図**11**に示す．この場合の可視面数は63となり大幅に減少する（3%）が，画像劣化も大きく，形状保持率は25.88%となった．

7.4 速度制御テーブルによる描画速度の自動制御

速度制御テーブルによる描画速度の自動制御の実験を行った．使用した速度制御テーブルを図**12**に示す．

新たに視界に入った物体に対する描画速度の追従性能を計測するため，図**13**に示すようなシーンを設定し，視点1から視点2まで，視点を並行移動した場合の描画速度の計測を行った．視点1の視界では，猫のモデルは見えていないが，視点2の視界では猫のモデルが出現する．

$$TS = \{T_0, T_1, T_2, T_3, T_4, T_5, T_6, T_7, T_8, T_9, T_{10}, T_{11}, T_{12}, T_{13}, T_{14}, T_{15}, T_{16}, T_{17}, T_{18}, T_{19}, T_{20}, T_{21}, T_{22}, T_{23}\}$$

$$\begin{aligned}
T_0 &= \{1, 0, 0\} & T_{12} &= \{128, 80, 64\} \\
T_1 &= \{8, 4, 0\} & T_{13} &= \{128, 100, 80\} \\
T_2 &= \{16, 8, 4\} & T_{14} &= \{256, 128, 80\} \\
T_3 &= \{24, 16, 8\} & T_{15} &= \{256, 128, 128\} \\
T_4 &= \{32, 24, 16\} & T_{16} &= \{256, 200, 160\} \\
T_5 &= \{48, 32, 16\} & T_{17} &= \{512, 256, 128\} \\
T_6 &= \{48, 32, 24\} & T_{18} &= \{512, 400, 256\} \\
T_7 &= \{64, 48, 24\} & T_{19} &= \{768, 512, 400\} \\
T_8 &= \{64, 48, 32\} & T_{20} &= \{1024, 768, 512\} \\
T_9 &= \{80, 64, 32\} & T_{21} &= \{1512, 1024, 768\} \\
T_{10} &= \{80, 64, 48\} & T_{22} &= \{2048, 1024, 768\} \\
T_{11} &= \{128, 80, 48\} & T_{23} &= \{2048, 2048, 1024\}
\end{aligned}$$

図12 実験で用いた速度制御テーブル
Fig. 12 Speed control table.

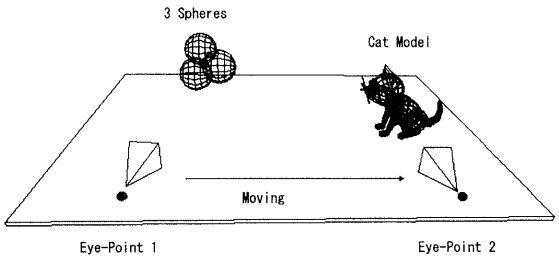


図13 実験で用いたシーン
Fig. 13 Scene of experiment.

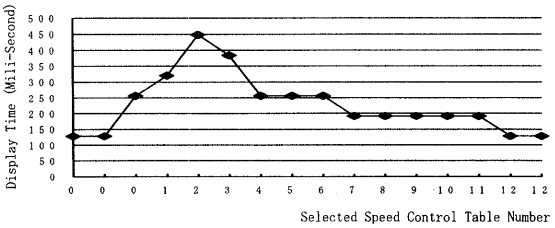


図14 単純なフィードバック制御
Fig. 14 Simple feedback algorithm.

要求描画時間150ミリ秒を与えて，単純なフィードバック方式で追従した場合の経過を図**14**に示す．突然視界に登場した猫のために，描画速度が大きく低下し，追従に時間がかかっていることがわかる．実際に視点をマウスを用いて移動した時の操作感は，猫が何かに引っ張られているような感覚である．

次に，予測型フィードバック方式で追従した場合の経過を図**15**に示す．図15 (a) は出現プリミティブを検出して低品質メンバで描画した場合の経過である．視界に猫が出現した時点で描画速度が低下するが，低下率は単純なフィードバック方式よりも少ない．また，速度制御ポインタが大きく移動して，素早く追従していることがわかる．

図15 (b) は，出現プリミティブを概略化メンバで描

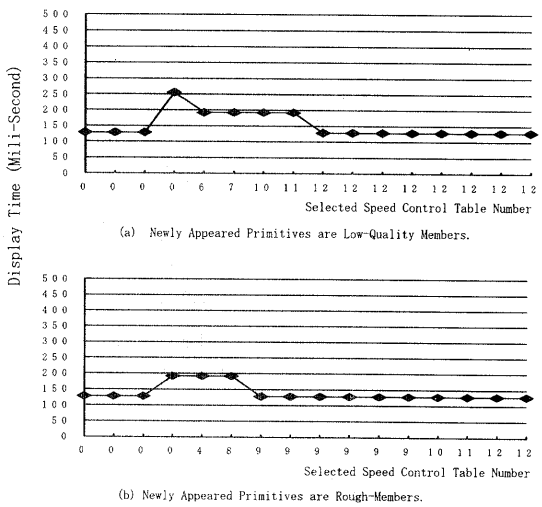


図15 予測型フィードバック制御
Fig. 15 Predictive feedback algorithm.

画した場合の経過である．概略化メンバは，1個のプリミティブが1個の断面で近似されているので，負荷が少ない．猫が出現しても描画時間が急激に変化しないため，安定した追従性能を示した．マウスによる視点移動の操作感覚は，図15の(a)，(b)ともに大きな感覚の差異はなく，滑らかな感覚であった．

8. むすび

3次元形状モデルをコンピュータグラフィックス技術によって可視化する際，計算機の性能に合わせて自動的に形状簡略化を行い，要求された描画時間を可能な限り保持する描画技法について提案した．これをまとめると次の7点になる．

(1) 基本立体プリミティブごとに高品質メンバ，中品質メンバ，低品質メンバ，概略化メンバを自動的に作成する．

(2) 中品質メンバは高品質メンバよりも少ない面数で曲面を近似したデータである．

(3) 低品質メンバは視線候補ベクトルを伴うデータで，あらかじめ設定された視点候補での可視面抽出を事前に計算しておく．描画時には視線ベクトルに最も近い視線候補ベクトルを1個選んで可視面を抽出することにより，描画速度を高速化する．このメンバは描画時間は高速であるが，輪郭部の欠落が有り得る．

(4) 概略化メンバは，基本立体の可視断面を2次元平面で近似する．視点位置によって最適な可視断面を選択するため，6方向の視線候補ベクトルを用いる．

(5) 各立体プリミティブは，XYZ方向の3次元特徴量をもつ．この三つの特徴量を投影変換した結果の最大値をメンバ決定のための基準値とする．この基準値を，メンバしきい値テーブルと比較することにより，メンバを自動選択する．この結果，視点と物体との距離ではなく，プリミティブの画面への投影量が小さいものから自動的に簡略化されていく．

(6) 描画速度をフィードバック方式によって自動制御するため，メンバしきい値テーブルの配列を作り，実際の描画時間を計測しながらメンバしきい値テーブルを切り替える．

(7) 描画速度制御の追従性能を向上させるため，出現プリミティブを検出し，面数の少ない形状メンバで描画して描画速度の急変を抑えるとともに，次回選択される可能性の高い形状メンバとの面数の差分を求める．視界内のすべての出現プリミティブの面数の差分を累積して，次回の描画時間を予測する．この予測値より，メンバしきい値テーブルを素早く切り替える．

上記の機能を含んだ3次元グラフィックスライブラリを開発し有効性を確認した．本技法は，視点と物体との距離を元にした形状簡略化ではなく，画面への投影画素量を基準とした形状簡略化なので，景観シミュレーションに限定されることなく，広範囲な3次元視覚化アプリケーションに適用できる．

今後の課題は，速度制御テーブルの自動作成，速度制御ポインタの移動量としきい値の最適値を求めるアルゴリズムの開発，より効率的なデータ構造や動的な簡略化形状メンバの作成によるメモリ消費量の削減などである．

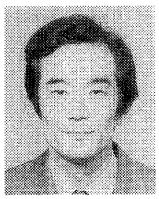
本研究では，本技法を含む3次元グラフィックスライブラリ全体を開発して評価実験を行った．しかし本技法は，3次元描画を行うハードウェア環境，陰面消去アルゴリズム，自動陰影づけのアルゴリズムなどに依存しないので，OpenGLやPEXなどを利用してインプリメントする方法も考えられる．この場合，既存のライブラリでは，投影変換機能が内部に隠蔽されているため，3次元特徴量の算出および視線候補ベクトルの選択処理を工夫する必要がある．これらの詳細については，機会をあらためて報告したいと考えている．

参考文献

1) 穂坂 衛：コンピュータグラフィックスの現状と動向，情報処理，Vol.29, No.10, pp.1068–1074

(1988).
2) 中前栄八郎：三次元コンピュータグラフィックス技法，情報処理，Vol.29, No.10, pp.1082–1089 (1988).
3) 国井利泰：CGの最新動向，電子情報通信学会，Vol.74, No.4, pp.381–385 (1991).
4) 福永，藤田，古賀：三次元グラフィックスの動向と技術課題，情報処理，Vol.34, No.7, pp.902–908 (1993).
5) 加藤伸子，岡崎彰夫：形状簡略化に基づく3次元オブジェクト空間の最適高速表示，信学論 (D-II), Vol.J76-D-II, No.8, pp.1712–1721 (1993).
6) 北嶋克寛，遊佐洋子：リアルタイム景観シミュレータのための形状のグルーピングと多重表現に基づく描画時間の短縮，信学論 (D-II), Vol.J77-D-II, No.2, pp.311–320 (1994).
7) Funkhouser,T. and Sequin, C.: Adaptive Display Algorithm for Interactive Frame Rates During Visualization of Complex Environments, *Proceedings of Siggraph '93*, pp.247–254 (1993).
8) 川合：コンピュータグラフィックスにおける標準化とその動向，情報処理，Vol.29, No.10, pp.1214–1222 (1988).
9) Rost, R.J. et al.: PEX: A Network-Transparent 3D Graphics System, *IEEE Computer Graphics & Applications*, pp.14–26 (1989).
10) Sung, H.C.K. and Kubitz, W.: A Critical Evaluation of PEX, *IEEE Computer Graphics & Applications*, pp.65–75 (1990).
11) Neider, J., Davis, T. and Woo, M.: *OpenGL Programming Guide,The Official Guide to Learning OpenGL, Release 1*, Addison-Wesley Publishing Company (1993).

(平成7年2月23日受付)
(平成7年7月 7 日採録)

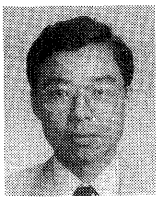


新藤 義昭（正会員）

昭和29年生．昭和51年電気通信大学卒業．同年富士通（株）入社．平成5年より日本工業大学工学部電気電子工学科専任講師．平成7年同大学情報工学科専任講師．オペレーティングシステム，コンピュータグラフィックス，画像処理等の研究に従事．「パソコンネットワーク実用読本」（ナツメ社），「グラフィックスプログラミング入門」（技術評論社）．電子情報通信学会会員．

片山 滋友（正会員）

1969年東京工業大学工業教員養成所電気工学科卒業．同年日本工業大学工学部電気工学科助手．1980年同学科専任講師．1989年電気電子工学科助教授．1995年情報工学科教授．精密小型モータと摺動材料，電動車椅子の制御，パソコン利用一斉教育システム，CAIにおける入力・提示技術に関する研究などに従事．電気学会，電子情報通信学会各会員．



坂本 康治（正会員）

昭和23年生．昭和46年茨城大学工学部電気工学科卒業．同年電子技術総合研究所入所．平成6年4月日本工業大学電気電子工学科教授．平成7年4月同大学情報工学科教授．この間昭和60年より1年間，フランス国立情報ならびに自動化研究所（INRIA）客員研究員．メモリシステム，VLSI CAD，ネットワーク，コンピュータ・グラフィックスの研究に従事．工学博士．電子情報通信学会会員．

松田 郁夫（正会員）

昭和7年生．昭和36年電気通信大学卒業．同年通産省工業技術院電子技術総合研究所（旧電気試験所）入所．最適化制御，システム・シミュレーション，応用システム分析，意志決定支援システム等に興味を持つ．昭和61年より日本工業大学工学部電気電子工学科教授．平成7年同大学情報工学科教授．工学博士．電子情報通信学会，人工知能学会，IEEE各会員．昭和56年電気学会論文賞受賞．