BRepからオクトツリーへの変換アルゴリズムとその評価†

登尾啓史†† 福田尚三†† 有本 卓††

オクトツリーは位置に関する階層構造をもつソリッドモデルであり，3次元空間そのものを物体の形状に従って分割することで作成される．オクトツリーではジオグラフィカルな情報が効率的に検索でき，レイ・トレーシングや干渉チェックなどが高速に行える．本論文では最も一般的に利用されているソリッドモデルであるBRep (Boundary representation) をオクトツリーに変換するアルゴリズムを提案する．このアルゴリズムの基本処理はある領域の8つの分割領域のBRepに対する内外・交差をその領域のパッチの情報だけから決定することであり2つの処理から構成される．第1はある領域のBRepのパッチを8つの分割領域に分配しBRepと交差する分割領域を決定する処理である．この処理はパッチを含む平面と交差する分割領域を選択すること（必要条件）とパッチと分割領域の X, Y, Z 方向への射影がおのおの交差する分割領域を選択すること（十分条件）で構成される．十分条件はBRepの稜を含む分割領域のみで調べる．第2はパッチが分配されなかったBRepが交差しない分割領域の内外判定である．領域のパッチの情報だけから分割領域の内外・交差が決定できることはすべての領域が独立に処理されることを意味し，このことはアルゴリズムを高速化するだけでなくその応用範囲も広げる．最後にアルゴリズムの計算手数とオクトツリーの記憶容量が，S (BRepの面積)，N (BRepの面数)，4^n (n：オクトツリーの作成レベル) に比例すると評価しそれを球のBRepデータで確かめる．

1. まえがき

コンピュータ・グラフィックス (CG) やCAD/CAMひいてはロボティックスの分野で，ジオグラフィカルな情報が効率的に検索できるオクトツリーという3次元モデルが研究されている[1)~3)]．そして，このオクトツリーを作成するための以下のような研究がある．

Franklinらは入力物体を近似した平行6面体の集合からオクトツリーを作成しているが，その平行6面体の集合を得るのに手間がかかる[4)]．Sametらは CSG-tree からビンツリー—オクトツリーによく似た3次元バイナリーツリー—の線形表現を作成するアルゴリズムを提案している[5)]．これも，そのCGS-treeの作成が大変である．Yauらは物体の断面を表現した2次元画像を重ね合わせてオクトツリーを作成している[6)]．この方法は物体の断面の2次元画像は容易に得られることから利用しやすいが，オクトツリー作成レベル n に関して 8^n に比例した計算手数を必要とする問題点がある．Tamminenらは BRep をビンツリーの線形表現へ変換するアルゴリズムを提案している[7)]．これはBRepという一般的な3次元モデルを対象としているだけに利用価値が高い．しかし，このアルゴリズムには，

1）BRepの面（パッチ）の集合に対応するビンツリーのノードが正確に作成されないので，BRepを近似したオクトツリーが作成できない．

2）まず，BRepの面（パッチ）の集合に対応するノード，続いてBRepの内部と外部に対応するノードを作成する2パス法なので，計算時間の点で効率が悪いだけでなくBRep全体のビンツリーしか作成できない．

といった問題点がある．

そこで本稿では，BRepからDF表現—オクトツリーの線形表現—への変換アルゴリズムを提案する．

このアルゴリズムは，

1）BRepの面（パッチ）に対応するオクトツリーのノードが正確に作成できる（定理1, 2による）．

2）BRepの面（パッチ）の集合に対応するノードとBRepの内部と外部に対応するノードを同時に作成する1パス法なので，高速であるばかりでなく限定した領域に属するBRepの一部に対応するオクトツリーも作成できる．このことから，提案するアルゴリズムはロボティックス[8),9)]，コンピュータ・ビジョン[10),11)]等の多くの応用分野に適用できる．

最後に，これまでほとんど行われていなかったアルゴリズムの計算手数の評価をBRepのパッチ数 N，表面積 S，オクトツリーの作成レベル n に関して行いその正当性を実験で確認する．

まず2章では，BRep，オクトツリー，DF表現を定義し，3章では，BRepをDF表現へ変換するアルゴ

† Conversion Algorithm from the Boundary Representation to the Octree and Its Complexity by HIROSHI NOBORIO, SHOZO FUKUDA and SUGURU ARIMOTO (Faculty of Engineering Science, Osaka University).

†† 大阪大学基礎工学部機械工学科

リズムを説明する．4章ではそのアルゴリズムの計算手数とオクトツリーのノード数を評価し，5章では実験により4章の評価式を検討する．

2. データ構造の定義

2.1 オクトツリー

オクトツリーは全体空間内のすべての物体（図1(a)）を位置に関して階層的に表現したソリッドモデル（図1(b)）である．オクトツリーでは物体が存在しない領域（"外側"キューブ）を白ノード，物体が占有する領域（"内側"キューブ）を黒ノード，物体が存在するが占有しない領域（"交差"キューブ）をミックスノードで表現する．

ミックスノード（キューブ C）は8つのサブ領域（子キューブ C_i）に分割され，それらはミックスノードの子ノードとして表される．なお，サブ領域と子ノードの対応は図2のように定義されている．

本研究では，オクトツリーの作成レベルを n とし，そのレベルのミックスノードは黒ノードで表現している．ここで，子ノードがすべて黒ノードであるようなミックスノードが現れた場合，それらの子ノードをミックスノードに統合しミックスノードを黒ノードに変更する．

このオクトツリーの利点は，

1）マスプロパティ（体積，面積，モーメント等の情報）が容易に計算できる．

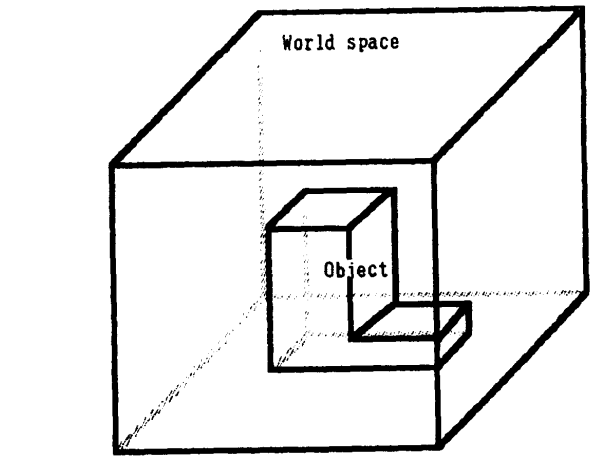
2）空間の8等分割により生成されたすべての領域を位置に関して階層的に保持しているので，任意の領域の属性が高速に検索できる．

3）オクトツリーのもとで利用されるアルゴリズムは単一処理を再帰的に用いるものが多いのでハードウェア化に適する[12]．

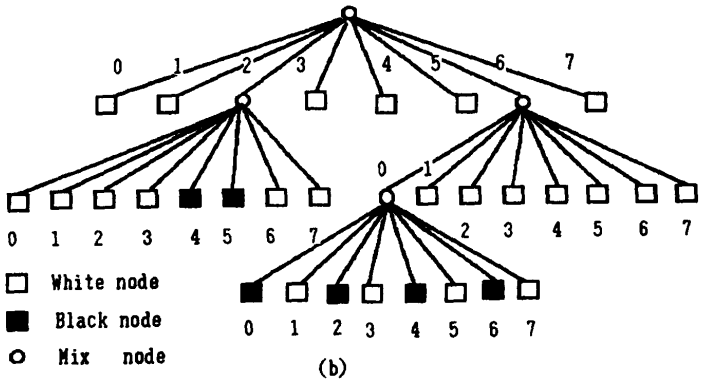
4）論理和，論理積などの演算が高速に行える．

ことである．

逆に，欠点は空間内のすべての物体の境界をキューブの集合で表現するために，BRep の表面積 S に比例，またオクトツリーの作成レベル n に関して 4^n に比例した記憶容量が必要なことである．



(a)



(b)

(00(00001100)000((10101010)0000000)0)

(c)

図1 全体空間(a)とそれに対応するオクトツリー(b)，DF-表現(c)
Fig. 1 World space (a) and its octree (b) and its DF-representation (c).

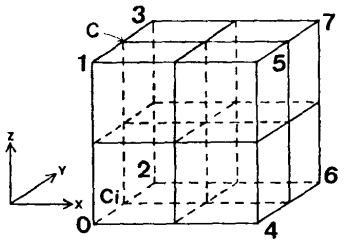


図2 8分割による位置の定義
Fig. 2 The order of octants.

2.2 DF 表 現

DF (Depth-First) 表現はオクトツリーの線形かつ

高圧縮な表現法[13]で，例えば図1(b)のオクトツリーは同図(c)のように表現される．ここで記号‘(’はミックスノードに入りツリーのレベルが1つ増加することを，記号‘)’はミックスノードから出てレベルが1つ減少することを意味する．また，記号‘1’は黒ノードを，記号‘0’は白ノードを表している．

2.3 BRep (Boundary Representation)

BRep はパッチ（面）の集合で物体を表現した最も一般的に利用されているソリッドモデルである[14]．本研究では，**図3**のように外向き法線方向に関して右ネジの順に並べた頂点の組でパッチ（面）を表している．また，本研究では BRep ならびにパッチをあらかじめ凸分割することにより，それらを凸形状と仮定している．

3. オクトツリー作成アルゴリズム

このアルゴリズムの基本処理は BRep と交差するキューブ（以後入力キューブと記す）内のパッチの情報だけから8つの子キューブを“内側”，“外側”，“交差”子キューブに分類することである．

全体空間に対応する“交差”キューブを最初の入力キューブとして次々と現れる“交差”キューブにこの基本処理を用いると全体空間の BRep に対応する DF 表現（オクトツリー）が作成される．また，限定した領域に属する BRep の一部に対応する DF 表現（オクトツリー）を作成するためには，その領域と全く交差しないキューブを“外側”と判断すればよい．

基本処理は以下の2つのルーチンから構成される．

1つめのルーチンは入力キューブ内のパッチ数によって MESH, PLURAL, SINGLE ルーチンから選択されるもので，そこでは入力キューブ内のパッチを8つの子キューブに重複を許して分配し“交差”子キューブ(パッチを1つでも含む子キューブ)を決定している．

2つめのルーチンは WRITE_DF ルーチンで，そこでは“非交差”子キューブ（パッチを全く含まない子キューブ）を“内側”，“外側”子キューブに分類している．

3.1 MESH ルーチン

このルーチンは，入力キューブが BRep のすべてのパッチを含む場合を扱う．

BRep は全体空間のどこに位置するかわからない．そこでここでは，BRep のすべてのパッチをひとまとめに考えそれを取り囲む最小の“交差”キューブを得ようと考える．

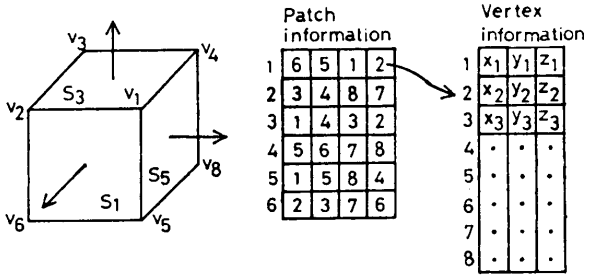


図3 境界表現
Fig. 3 Boundary representation (BRep).

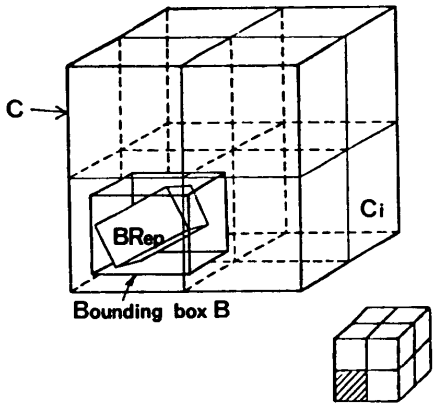


図4 MESH ルーチン
Fig. 4 MESH routine.

このルーチンでは**図4**のように BRep を取り囲む最小直方体 B を定義し，その直方体と交差する子キューブを“交差”子キューブとする．それ以外の子キューブは“外側”子キューブなので記号‘0’を割りあてる．

3.2 PLURAL ルーチン

このルーチンは，入力キューブが2つ以上のパッチを含む，すなわち，BRep の稜を含む場合を扱う．このルーチンではそれらのパッチを8つの子キューブに重複を許して分配することで“交差”子キューブを決定している．

ここでは，あるパッチ P の分配方法を説明する．これを入力キューブ内のすべてのパッチに利用すると上述の“交差”子キューブが決定できる．パッチ P の分配は，BBOX, PLANE, PROJECTION 手続きをこの順に用いて行われる．

3.2.1 バウンディングボックス手続き (BBOX)

MESH ルーチンと同様な方法でパッチ P を取り囲む最小直方体 B を定義し，それと交差する子キューブを“交差”子キューブ，それ以外の子キューブを“非交差”子キューブとする．

面（パッチ）は凸形状なので，この手続きは PROJECTION 手続きを簡単化する．

3.2.2 平面手続き (PLANE)

パッチ P を含む平面 S が交差する子キューブを“交差”子キューブ，それ以外の子キューブを“非交差”子キューブとする．

パッチ P が含まれる平面 S の方程式を，

$$d(S,U) := \vec{n}\cdot(\vec{u}-\vec{u}_0).$$
$$= n_x * u_x + n_y * u_y + n_z * u_z - d.$$
$$(d := \vec{n}\cdot\vec{u}_0)$$

$\vec{n}$：平面 S の外向き法線ベクトル．

$\vec{u}_0$：平面 S 上の点の位置ベクトル．

$\vec{u}$：空間内の点 U の位置ベクトル．

と定義する．この関数は平面 S から点 U までの変位を表しており，その符号が負ならば点 U は平面 S の内側に，正ならば点 U は平面 S の外側に存在する．ここで平面 S の内側［外側］とは，全体空間を平面 S で分割したとき BRep が存在する［しない］領域のことである．

子キューブ C_i の位置（図2）を表す10進表現の添字 i の2進表現が $(r_0\ r_1\ r_2)_2$ であるとき，$\vec{i}=(r_0, r_1, r_2)$ なるベクトルを定義し添字 i に対応づける．

平面 S と交差する子キューブを見つけるため，まずキューブ C の中心点 G における $d(S,G)$ の正負を調べ，これより以下の処理を行う．

（i） $d(S,G)<0$ ［$d(S,G)>0$］のとき

中心点 G は平面 S の内側［外側］に存在する．内積 $\vec{n}\cdot\vec{i}$ が最大［最小］となる添字 i を k と表すと，子キューブ C_k の周辺で平面 S は子キューブと交差する．今，G を除く C_k の頂点 v_i において $d(S,v_i)>0$ ［$d(S,v_i)<0$］となると，頂点 v_i は平面 S の外側［内側］に存在し平面 S は線分 Gv_i と交差する．したがって，線分 Gv_i と隣接するすべての子キューブを“交差”子キューブ，それ以外の子キューブを“非交差”子キューブとする．

このとき，線分 Gv_i において交差が発見されれば，それと隣接する子キューブは平面 S と交差することがわかるので，頂点 v_i 以外の頂点のうちのいくつかは調べなくてもよくなる．例えば**図5**の場合，線分 Gv_1 において平面 S との交差が発見されたので頂点 v_2, v_6 は調べなくてよい．

この手続きと文献 7) のアルゴリズムの対応する処理を比較する．文献 7) の処理では子キューブごとに“交差”，“非交差”を調べているので，頂点の平面方程式への代入は最大 16 回必要となるが，この手続きでは8つの子キューブの“交差”，“非交差”をまとめて判定するので最大8回の代入となり半分の手数で分類できる．

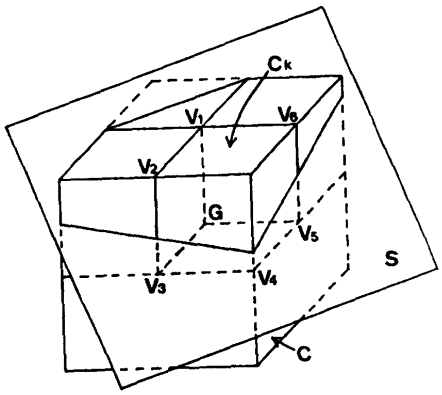


図 5 平面 S と子キューブの交差
Fig. 5 Intersection between plane S and subcube.

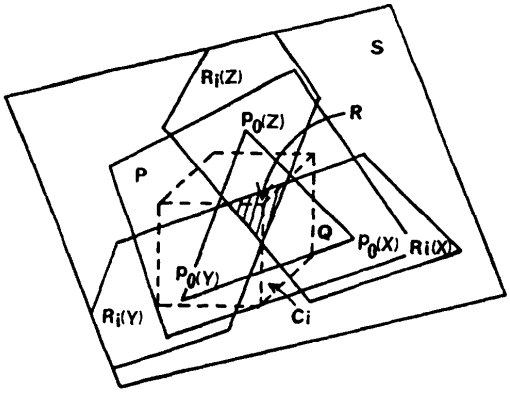


図 6 平面 S 上のキューブの射影領域
Fig. 6 Projecting region of cube onto the plane S.

（ii） $d(S,G)=0$ のとき

中心点 G は平面 S 上に存在する．この場合，$\vec{n}\cdot\vec{i}$ が最小［最大］となる子キューブは平面 S の“内側”［“外側”］子キューブとなり，その他の子キューブは“交差”子キューブとなる．

3.2.3 正射影手続き (PROJECTION)

ここでは，X, Y, Z 軸に垂直な3平面 $D(X), D(Y), D(Z)$ 上にパッチ P，子キューブ C_i の射影 $P(j)$, $C_i(j)$ $(j=X, Y, Z)$ をとり，その交差を調べることでパッチ P と子キューブ C_i の交差を正確に判断する．

まず記号を定義する（**図6**）．

R：子キューブ C_i と平面 S の交差領域 $(R=C_i\cap S)$．

$R_i(j)\cup R$：平面 $D(j)$ への射影が $C_i(j)$ となる平面 S

上の領域.

[3 D情報] (図7)

[非 交 差] $C_i \cap P = \phi$.

[部分交差] $C_i \cap S \supsetneq C_i \cap P \neq \phi$.

[完全交差] $C_i \cap S = C_i \cap P \neq \phi$.

[2 D情報] (図8)

[非 交 差] $C_i(j) \cap P(j) = \phi$.

[部分交差] $C_i(j) \supsetneq C_i(j) \cap P(j) \neq \phi$.

[完全交差] $C_i(j) = C_i(j) \cap P(j) \neq \phi$.

3 D情報と2 D情報の間には以下の関係がある.

[定理1]

すべての平面 $D(j)$ において $C_i(j) \cap P(j) \neq \phi$
$\leftrightarrow C_i \cap P \neq \phi$.

[証明] ($\Rightarrow$) すべての平面 $D(j)$ において $C_i(j) \cap P(j) \neq \phi$ となることは，パッチ P 上に点 $p_0(j) \in R_i(j) \cup R\ (j = X, Y, Z)$ が存在することを意味する．すでに，PLANE 手続きの処理を行って領域 $R \neq \phi$ を確かめており，3点 $p_0(j)\ (j = X, Y, Z)$ から構成された領域 Q は領域 R と明らかに交差をもつ．ここで，パッチ P は凸形状としているので領域 Q はその内部に存在する (図6)．以上のことから，$C_i \cap P \neq \phi$ が成立する．

($\Leftarrow$) 点 $p \in C_i \cap P \neq \phi$ の平面 $D(j)$ への射影をとると，点 $p_1(j) \in C_i(j) \cap P(j)$ が必ず存在するので，すべての平面 $D(j)$ において $p_1(j) \in C_i(j) \cap P(j) \neq \phi$ が成立する．□

[定理2]

ある平面 $D(j)$ において $C_i(j) \cap P(j) = \phi$
$\leftrightarrow C_i \cap P = \phi$ が成り立つ.

[証明] ($\Rightarrow$) 定理1における ($\Leftarrow$) の証明の対偶.

($\Leftarrow$) 定理1における ($\Rightarrow$) の証明の対偶. □

文献 7) では，$|\vec{n} \cdot \vec{j}|$ ($\vec{j}$: X, Y, Z 軸上の単位ベクトルでそれぞれ X, Y, Z 軸に対応づけられている) が最大となる平面 $D(j)$ 上の情報しか利用していないが，BRep を正確に表現したオクトツリーを作成するためには，3つの平面 $D(X)$, $D(Y)$, $D(Z)$ 上の情報をすべて利用しなければならないことが定理 1, 2 から理解できる.

平面 $D(j)$ における射影 $C_i(j)$ と射影 $P(j)$ の "交差", "非交差" の判定方法について説明する.

まず，射影 $P(j)$ を構成するエッジの1つを E_i，そのエッジを含む直線を L_i とすると，その直線により平面 $D(j)$ は2つの領域に分けられる．このとき，射影 $P(j)$ を含む領域を内側，そうでない領域を外側とする.

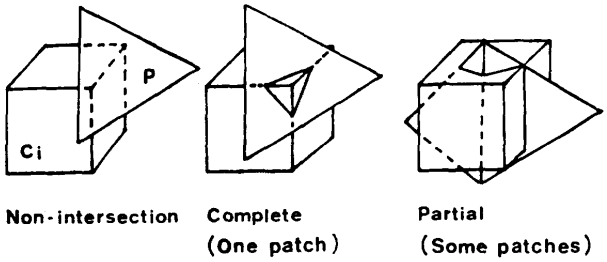


図 7 パッチとキューブの交差パターン
Fig. 7 Three dimensional intersecting pattern between patch and cube.

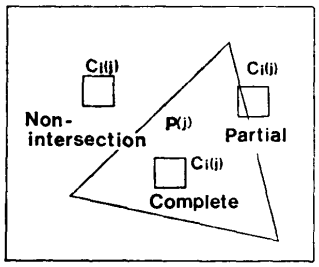


図 8 正射影平面でのパッチとキューブの交差パターン
Fig. 8 Two dimensional intersecting pattern between patch and cube.

既に BBOX 手続きを行っているので，ここではパッチ P を囲む最小直方体 B の射影 $B(j)$ 内に射影 $C_i(j)$ は存在するものと考えればよい．また，パッチ P は凸形状としていることから射影 $B(j)$ の周囲上にない射影 $P(j)$ の角はすべて鈍角となる.

これらのことから，ある直線に関して射影 $C_i(j)$ が外側なら射影 $P(j)$ と射影 $C_i(j)$ は "非交差"，すべての直線に関して射影 $C_i(j)$ が内側なら射影 $P(j)$ と射影 $C_i(j)$ は "完全交差"，それ以外の射影 $C_i(j)$ は射影 $P(j)$ と "部分交差" となる (図9参照).

本研究では，図10のようにキューブ C の平面 $D(i)$ への射影を4分割して領域 $R_h\ (h = 0, \cdots, 3)$ を定義し，それらと射影 $P(j)$ の "交差"，"非交差" をまとめて判断して効率を上げている．射影 $C_i(j)\ (i = 0, \cdots, 7)$ は領域 R_h のいずれかに対応するので，射影 $P(j)$ と領域 R_h の "交差"，"非交差" から射影 $P(j)$ と射影 $C_i(j)$ の "交差"，"非交差" が判断できる.

ここで，2つのベクトル $\vec{n}_i, \vec{h}$ を定義する．$\vec{n}_i$ は直線 L_i に垂直で外向きのベクトル，$\vec{h}$ は領域 R_h の位置を表す 10 進表現の添字 h の2進表現 $(r_0\, r_1)_2$ から定義されるベクトル (r_0, r_1) であり添字 h に対応さ

せる．

まず，ある直線 L_i に関する4つの領域 R_k の“交差”，“非交差”の判定について説明する．直線 L_i に関するキューブ C の射影の中心点 G の内外・交差から以下の処理を選択する．

（i） 中心点 G が内側［外側］の場合：$\vec{n}_i \cdot \vec{h}$ が最大［最小］になる位置 h を k とおくと，直線 L_i は領域 R_k と交差し領域 R_k は“部分交差”となる．次に，領域 R_k を構成する頂点のうち点 G と隣接する頂点 v（v_0 そして v_1）の直線 L_i に関する内外を判定する．もし，頂点 v が外側［内側］であれば，直線 L_i は線分 Gv と交差するので線分 Gv と接する領域 R_k も“部分交差”となる．

最後に，中心点 G が内側［外側］のとき，それ以外の領域は“完全交差”［“非交差”］となる．

（ii） 中心点 G と交差する場合：$\vec{n}_i \cdot \vec{h}$ が最大になる位置 h の領域 R_k は“非交差”，最小になる位置 h の領域 R_k は“完全交差”，あとの2つの領域は“部分交差”となる．

次に，射影 $P(j)$ を構成するエッジに対して定義されたすべての直線に上述の処理を行ったとき，ある直線に関して“非交差”となる領域 R_k は射影 $P(j)$ と“非交差”，すべての直線に関して“完全交差”となる領域 R_k は射影 $P(j)$ と“完全交差”，それ以外の領域は射影 $P(j)$ と“部分交差”となる．

これによって，平面 $D(j)$ における射影 $C_i(j)$ と射影 $P(j)$ の“交差”，“非交差”が判定できる．

3つの手続きで得た子キューブ C_i の3つの“交差”，“非交差”情報のうち1つでも“非交差”であれば，子キューブ C_i とパッチ P は“非交差”，そうでなければ子キューブ C_i とパッチ P は“交差”となりパッチ P を子キューブ C_i に分配する．このことにより，あるパッチ P の8つの子キューブへの重複を許した分配が実現される．

3.3 SINGLE ルーチン

このルーチンは，入力キューブが1つのパッチのみを含む場合を扱う．

この場合，パッチ P と子キューブ C_i の交差は平面 S と子キューブ C_i の交差となるので，PLANE 手続きのみで“交差”子キューブが決定できる．

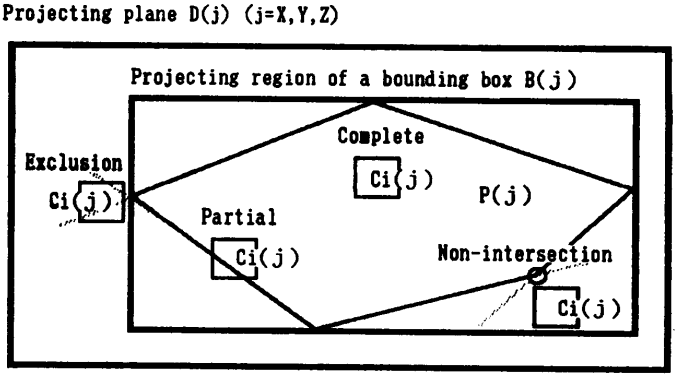


図 9 PROJECTION 手続きにおけるバウンディングボックスの利用
Fig. 9 Use of the bounding box in the PROJECTION procedure.

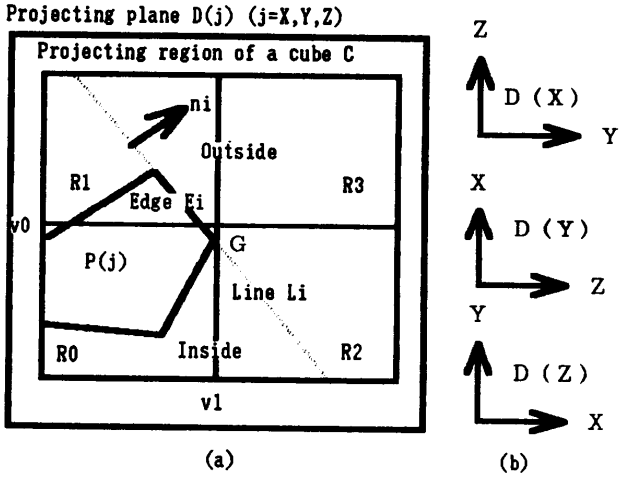


図 10 4つの領域の内外・交差判定と正射影平面の選択（b）
Fig. 10 Judgement of the four regions into inside, outside, or intersection (a) and selection of a projecting plane (b).

本研究では，これまで述べてきた3つのルーチン MESH, PLURAL, SINGLE を以下のように用いている．まず，始めの入力キューブを MESH ルーチンで処理する．MESH ルーチンで“交差”子キューブが1個現れたときは(図4参照)，それを MESH ルーチンで続けて処理する．MESH ルーチンで“交差”子キューブが2個以上現れたときは，それらを MESH ルーチンではなく PLURAL ルーチンで処理する．

また，“交差”キューブに分配されたパッチの個数を数えておくと，それが入力キューブになったときその個数により PLURAL, SINGLE ルーチンを選択できる．

3.4 WRITE_DF ルーチン

このルーチンでは，入力キューブ内の“非交差”子キューブの内外を判断し，その DF 表現 ‘0’, ‘1’ を作

成する．

入力キューブ内の“非交差”子キューブは以下の2つの性質をもっている．

性質1 BRep は凸形状としているので，ある平面 S でキューブの中心点 G が外側であれば中心点 G は BRep の外側，すべての平面 S で中心点 G が内側であれば中心点 G は BRep の内側，そうでなければ中心点 G は BRep と交差する．□

性質2 パッチ（面）が中心点を通らない限り，“非交差”子キューブの内外は，中心点 G の BRep に関する内外と一致する．□

したがって，“非交差”子キューブの DF 表現は次のように作成される．中心点 G を PLANE 手続きで調べたとき，ある平面 S で外側の場合には記号 '0' を，また，すべての平面 S で内側の場合には記号 '1' をすべての“非交差”子キューブに割り当てる．そうでない場合には手続き PLANE（ii）の内外情報を利用して“非交差”子キューブに記号 '0'，'1' を割り当てる．

4. アルゴリズムの計算手数の評価

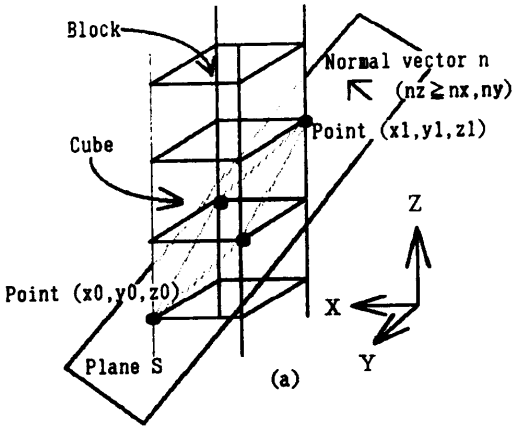
提案したアルゴリズムの計算手数は，それが処理した“交差”キューブの個数を調べることで評価できる．オクトツリーの各レベルでの“交差”キューブの個数を調べるため定理 1，2 を利用する．

まず，$|\vec{n}\cdot\vec{j}|$ が最大になる平面 $D(j)$ へパッチ P の正射影をとりその写像を P' とする．パッチ P の面積，周囲長を S_P，L_P とすると，写像 P' の面積 S_P'，周囲長 L_P' は $S_P'=S_P*\cos\theta$，$L_P'=L_P*\cos\theta$ ($\cos\theta=|\vec{n}\cdot\vec{j}|/|\vec{n}|*|\vec{j}|$，$\cos\theta$ は定数）で与えられる．

ここで，あるレベルのキューブの射影（ブロック）の面積を S_C，一辺の長さを L_C とすると，そのレベルで写像 P' と“完全交差”になるブロックの個数は最大 S_P'/S_C 個，“部分交差”になるブロックの個数は最大 $\sqrt{2}L_P'/L_C+2*\alpha$ 個（α は射像 P' を構成するエッジ数）となる（付録1）．また，ブロック上で平面 S（パッチ P）と交差するキューブは最大3個である（**図 11**，付録 2）．

平面手続きまでの処理手数を C_1'，正射影手続きの処理手数を C_2' とし，パッチ P_k に関するアルゴリズムの計算手数を C_k，オクトツリーのノード数を N_k とする．

全体空間を構成する面の面積を S'，辺の長さを L' とすると，レベル i の S_C，L_C はおのおの $S_C=S'/4^i$，$L_C=L'/2^i$ となり，



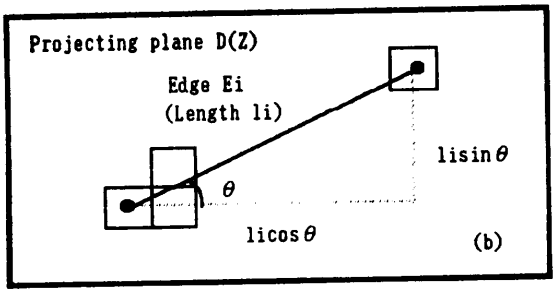


図 11 平面 S と交差するキューブの個数
Fig. 11 Number of cubes intersecting the plane S.

$$C_k\leqq 3\sum_{i=0}^{n-1}\{C_1'S_P'4^i/S'+\sqrt{2}C_2'L_P'2^i/L'+2C_2'\alpha\}$$
$$=C_1S_P\cdot 4^n/S'+3\sqrt{2}C_2L_P\cdot 2^n/L'+6C_2'n\alpha.$$
$$N_k\cong 24\sum_{i=0}^{n-1}S_P'4^i/S'$$
$$=8C_3S_P\cdot 4^n/S'.\quad (C_1, C_2, C_3: \text{定数})$$

アルゴリズムの総計算手数を Cost，オクトツリーの総ノード数を Storage とすると，

$$\text{Cost}\leqq\sum_{k=1}^{N}C_k$$
$$=C_1S\cdot 4^n/S'+3\sqrt{2}C_2L\cdot 2^n/L'+6C_2'Nn\alpha.$$
$$\text{Storage}\cong\sum_{k=1}^{N}N_k$$
$$\leqq 8C_3S*4^n/S'.$$

となる．BRep の表面積を S，総辺長を L，総パッチ数を N と記述している．

アルゴリズムの計算手数とオクトツリーの記憶容量はともに 4^n，S そして N に比例した値となる．ここで，表面積 S を一定にしたとき L が $\sqrt{N}$ に比例した値となるので，Nの値が大きくないときアルゴリズムの計算手数はその影響を受ける．

5. 実験結果

この章では実験によりパラメータ S, N, n に関するアルゴリズムの計算時間とオクトツリーのノード数を調べる．プログラムはC言語で書かれており，球をBRep（パッチは四角形）で近似したものをデータとしVAX 11/750 の Unix 上で実行した．

ここで，単位 Unit は全体空間の一辺の長さを直径とした球を，また 1/8 Unit はその 1/2 を直径とした球をもとにした BRep を表す．レベル n は全体空間を 8^n 等分割した解像度で BRep を表現したことを意味している．

図12, 13, 14のグラフから4章の評価式が正当であることが確認できる．また，図12(a)のオクトツリー作成レベル n が6と7の結果から，提案したアルゴリズムが文献 7) のアルゴリズムよりも高速なことがわかる．

6. むすび

最も一般的に利用されるソリッドモデルである

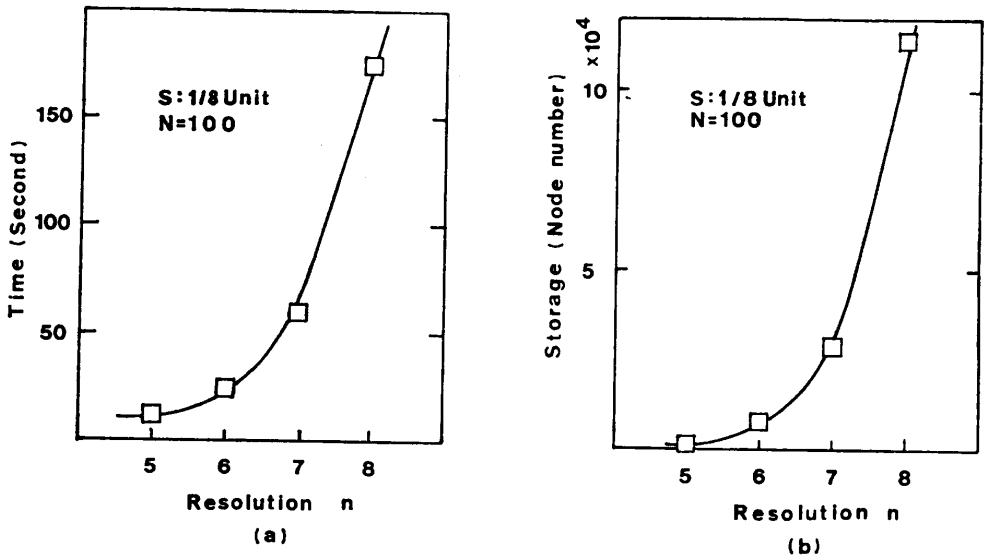


図 12 オクトツリー作成レベル(n)に関する(a)アルゴリズムの計算時間，(b)オクトツリーのノード数
Fig. 12 (a) Complexity of the algorithm, (b) number of nodes in the octree as a function of resolution (n).

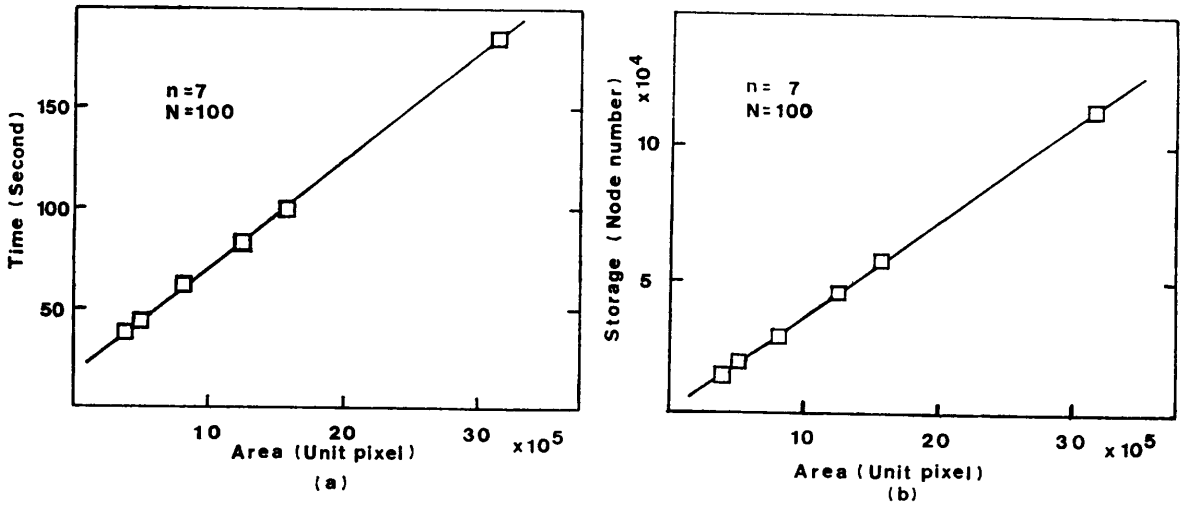


図 13 BRep の表面積(S)に関する(a)アルゴリズムの計算時間，(b)オクトツリーのノード数
Fig. 13 (a) Complexity of the algorithm, (b) number of nodes in the octree as a function of the area (S).

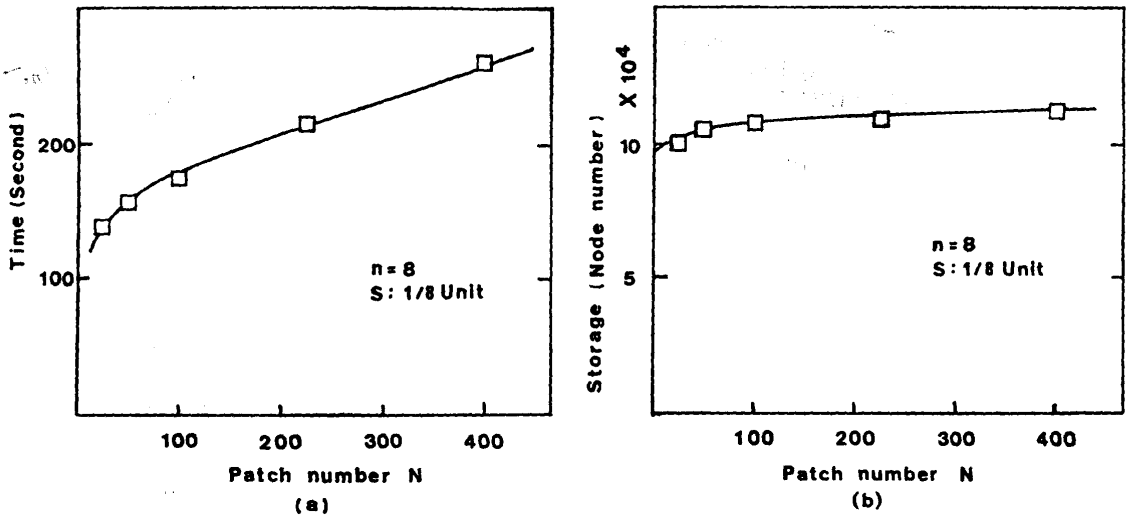


図 14 BRep の面数(N)に関する(a)アルゴリズムの計算時間，(b)オクトツリーのノード数
Fig. 14 (a) Complexity of the algorithm, (b) number of nodes in the octree as a function of the patch number (N).

BRep から DF 表現（オクトツリー）を作成する高速なアルゴリズムを提案した．

領域（キューブ）ごとに独立に DF 表現（オクトツリー）が作成できるという性質より，限定した領域に属する BRep の一部からでもパッチの情報さえわかっていればその領域内の BRep を表現した DF 表現（オクトツリー）は容易に作成できる．

DF 表現（オクトツリー）の作成が領域ごとに動的に選択できるこの性質から

1）ロボティックス：ロボット動作をグラフィックス・シミュレータで確認するときに必要な干渉チェック[8),9)]

2）コンピュータ・ビジョン：錐体相貫法により画像から環境を入力する場合のように，複数の多面体の共通領域の作成[10),11)]

などの用途にこのアルゴリズムはそのまま適用できる．

今回の実験では，“交差” キューブを縦形探索で選択しそれを「逐次処理」している．しかし，前述のキューブの処理の独立性から “交差” キューブを横形探索で選択し，オクトツリーのレベルごとにそれらを「並列処理」すると一層の高速化がはかれる．

最後に，BRep が凸形状でない場合，まず前処理でそれを凸分割し，次に，生じた “架空” の面（パッチ）を “真” の面（パッチ）と区別して一“架空” の面のみを含むキューブは，“交差” キューブではなく “内側” キューブと考えて一アルゴリズムで処理すると，BRep が凸形状の場合と同様に効率良くその DF 表現（オクトツリー）が作成できる．

謝辞 本研究に関して，日頃から貴重な御助言をいただく，三菱電機(株) 竹垣盛一氏，大井忠氏に感謝いたします．

参 考 文 献

1) Jackins, C. L. and Tanimoto, S. L.: Oct-trees and Their Use in Representing Three Dimensional Objects, *Comput. Gr. Image Process.*, Vol. 14, No. 3, pp. 249-270 (1980).
2) Jackins, C. L. and Tanimoto, S. L.: Quadtrees, Octrees, and K-trees: A Generalized Approach to Recursive Decomposition of Euclidean Space, *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. PAMI-5, No. 5, pp. 533-539 (1983).
3) Meagher, D.: Geometric Modeling Using Octree Encoding, *Comput. Gr. Image Process.*, Vol. 19, No. 2, pp. 129-147 (1982).
4) Franklin, W. R. and Akman, V.: Building an Octree from a Set of Parallele-pipeds, *IEEE Computer Graphics and Applications*, Vol. 5, No. 10, pp. 58-64 (1985).
5) Samet, H. and Tamminen, M.: Bintrees, CSG Trees and Time, *Comput. Graph.* (*Proc. SIGGRAPH '85 Conf.*), Vol. 19, No. 3, pp. 121-130 (1985).
6) Yau, M. and Srihari, S. N.: A Hierarchical Data Structure for Multidimensional Images, *Comm. ACM*, Vol. 26, No. 7, pp. 504-515

(1983).
7) Tamminen, M. and Samet, H.: Efficient Octree Conversion by Connectivity Labeling, *Comput. Graph. (Proc. SIGGRAPH '84 Conf.)*, Vol. 18, No. 3, pp. 43-51 (1984).
8) Noborio, H., Fukuda, S. and Arimoto, S.: A New Interference Check Algorithm Using Octree, *the 1987 IEEE International Conference on Robotics and Automation*, Raleigh, U. S. A. (1987).
9) 登尾，福田，有本：オクトツリーを用いた高速干渉チェック法，日本ロボット学会誌，Vol. 5, No. 3, pp. 21-30 (1987).
10) 登尾，福田，有本：錐体相貫法を用いた2次元画像からのオクトツリーの作成，情報処理学会第46回コンピュータ・ビジョン研究会資料，pp. 49-58, 1月 (1987).
11) Hong, Tsai-Hong and Shneier, M. O.: Describing a Robot's Work Space Using a Sequence of Views from a Moving Camera, *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. PAMI-7, No. 6, pp. 721-726 (1985).
12) Meagher, D.: The Solids Engine: A Processor for Interactive Solid Modeling, *Proc. NICOGRAPH '84 Conf.* (1984).
13) Kawaguchi, E. and Endo, T.: On a Method of Binary Picture Representation and Its Application to Compression, *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. PAMI-2, pp. 27-35 (1980).
14) Requicha, A. A. G. and Voelcker, H. B.: Solid Modeling: Current Status and Research Directions, *IEEE Computer Graphics and Applications*, Vol. 3, No. 7, pp. 25-37 (1983).
15) Lee, Y. T. and Requicha, A. A. G.: Algorithms for Computing the Volume and Other Integral Properties of Solid. I and II, *Comm. ACM*, Vol. 25, No. 9, pp. 635-650 (1982).

付 録 1

(長さ l_i の線分 E_i と交差するブロック数)

$\leqq \lceil l_i \cdot (\cos\theta + \sin\theta)/Lc \rceil + 1$

$\leqq l_i \cdot (\cos\theta + \sin\theta)/Lc + 2$

$= \sqrt{2}\, l_i \cdot \sin(\theta + \pi/4)/Lc + 2$

$\leqq \sqrt{2}\, l_i/Lc + 2$

(θ: 線分 E_i を含む直線 L_i が座標軸となす小さい方の角度)

したがって

(射影 P' が交差するブロック数)

$\leqq \sum_i (\sqrt{2}\, l_i/Lc + 2) = \sqrt{2}\, L_{P'}/Lc + 2\alpha.$

(α: 射影 P' を構成するエッジ数) □

付 録 2

平面の方程式を $f = n_x \cdot x + n_y \cdot y + n_z \cdot z - d$ ($n_z \geqq n_x, n_y$)，ブロックの対角に位置する2頂点を (x_0, y_0), (x_1, y_1) とする．

$n_x \cdot x_0 + n_y \cdot y_0 + n_z \cdot z_0 - d = 0.$

$z_0 = (d - n_x \cdot x_0 - n_y \cdot y_0)/n_z.$

$n_x \cdot x_1 + n_y \cdot y_1 + n_z \cdot z_1 - d = 0.$

$z_1 = (d - n_x \cdot x_1 - n_y \cdot y_1)/n_z.$

$0 \leqq |z_1 - z_0|$

$= |(x_0 - x_1) \cdot n_x + (y_0 - y_1) \cdot n_y|/|n_z|.$

$\leqq (|x_0 - x_1| \cdot |n_x| + |y_0 - y_1| \cdot |n_y|)/|n_z|$

$= Lc \cdot (|n_x| + |n_y|)/|n_z|.$

$0 \leqq |z_1 - z_0|/Lc \leqq (|n_x| + |n_y|)/|n_z| \leqq 2.$

$1 \leqq$ (交差キューブ数) $= |z_1 - z_0|/Lc + 1 \leqq 3.$ □

(昭和61年11月17日受付)
(昭和62年9月9日採録)

登尾 啓史 (正会員)

昭和33年生．昭和57年静岡大学工学部情報工学科卒業．昭和59年同大学院修士課程修了．昭和62年大阪大学大学院基礎工学研究科物理系専攻機械工学分野後期(博士)課程修了．同年同学機械工学科助手．大阪大学工学博士．ロボティックス，コンピュータビジョン，コンピュータグラフィックスの研究に従事．電子情報通信学会，日本ロボット学会，IEEE 各会員．

福田 尚三

昭和38年生．昭和61年大阪大学基礎工学部機械工学科卒業．現在，同大学大学院在学中．ロボティックスの研究に従事．人工知能，並列処理システムなどに興味をもつ．

有本 卓 (正会員)

昭和11年生．昭和34年京都大学理学部卒業．同年沖電気(株)入社．電子計算機の開発に従事．37年東京大学工学部助手，42年講師，43年大阪大学基礎工学部助教授，48年教授，42年工学博士．この間，情報理論，制御理論，ディジタル信号処理の高速アルゴリズムに従事する一方，ロボット工学に興味をもち，そのインテリジェント化の研究開発を行っている．電子情報通信学会，計測自動制御学会，日本機械学会，ロボット学会各会員．IEEE の Fellow 会員．