5K-2

新スキャンラインアルゴリズム

秋葉 澄伸　山本 強　青木 由直
(北海道大学)

1．まえがき

スキャンラインアルゴリズムは，コンピュータグラフィックスのレンダリング手法として最もポピュラーなものである．アルゴリズムにはいくつかのバリエーションがあるが，いずれも完成されたものである．筆者らは，従来のアルゴリズムの問題点を分析し，さらに効率のよいアルゴリズムの提案をおこなう．実際に処理系をつくり，その優位性をしめす．

2．従来のスキャンラインアルゴリズム

基本となるのは，隠面消去の問題をスキャン平面上の線分(セグメント)の交差問題に置き換え，走査線ごとに処理するものである．つまり，三次元空間上の２次元平面の問題を，２次元平面上の１次元直線へと還元することである．
セグメント群の優先判定法は．．．
[1] スキャンライン単位のZ-buffer法
[2] 前後関係によってソーティングを施し，優先順にセグメントをフレームバッファに書き込む方法(Fuches)
[3] 走査線をセグメントの端点で分割したサンプルスパンを単位として，アクティブなセグメントのリストを作る方法(Bouknight, Romney)
[4] スパンを２分法によりサブスパンに分割して，サブスパン内が一様となるまで２分割を繰り返す方法(Watkins)
などが古くから提案されている．

原理が単純で安直に用いられる[1]は，フルサイズのZ-bufferを用いたほうが明かに高速であるし，スキャンラインの利点をスポイルしている．スキャンラインアルゴリズムの特徴は，ライン間コヒーレンシを利用できることである．この特徴を活かしたものは[2][3][4]の方法である．[2]はソーティングが難しく，フレームバッファへのアクセスが冗漫となり，プリミティブに相貫が生じた場合に対応しにくい．ソーティングにおおきく依存しているため，モデルが複雑になると計算オーダが膨大になる．

3．提案するアルゴリズム

○スキャンライン方向に，セグメントの左始点についてバブルソートを行ない，ライン間コヒーレントを有効に利用する．([3]なども利用している．)
○優先判定は２個のセグメントのペアーについて行なう．([3]や[4]はダイナミックなセグメントのソートリストが必要となり，複数のセグメントの間での比較となる．)セグメントの区間が交差しているものみ優先関係の判定を行なう．([2]ではその選択がしにくく，全要素が比較の対象となる．)
○優先関係は，まずＭＩＮＭＡＸ法を用いる．そのあと線分の始点終点から状態判別関数をよぶ．交点や深さは，判別関数により交差することがわかるまで計算しない．(プリミティブに相貫がない場合には深さの計算は行なわれない．)判別関数は各セグメントの始点と終点のみから判別式を計算する．([3]は多くのサンプルスパン境界で，エッジの深さの計算が必要となる．)
○サンプルスパンを単一方向にインクリメンタルに処理を進める．フレームバッファに対するアクセスは１回となり最低限ですむ．([2]はフレームバッファにたいするアスセスが冗漫となる．)
○メモリーはセグメントのセットとスタックだけである．リニアーなメモリー環境なので，記憶容量ぎりぎりまでのデータを処理できる．([2][4]はダイナミックなメモリー管理や広域のリスト操作が必要となり，ページスワッピングなどが頻繁になる．)

本アルゴリズムでは，注目しているセグメントと区間が重なり，優先するセグメントがあればそちらに注目して再帰的に処理する．最も優先するセグメントをサンプルスパン単位でインクリメンタルにトレースする．セグメントの位置関係は再帰呼び出しによるスタックと，処理関数内の状態遷移がもっていることになる．
[3]や[4]の位置関係はアクティブセグメントリストが持ち，直感的に分かりやすい．しかし[4]では，サブスパンへの分割のたびにリストをそっくりコピーしなければならない．

下の l_edge=0; func(0); を評価する事により１走査線が作画できる．

```
q[MAX];  /*ソートされたセグメントのセット*/
l_edge;  /*注目している区間の左はじ*/

func(n) int n ; /* 関数本体 */
{
  for( i=n+1 ; i<MAX ; i++ ) {
      if( q[n],q[i]のセグメントが重ならない ) break ;
      switch( q[n]とq[i]の状態が ) {
        case q[n]優先 : continue ;
        case q[i]優先 :
          l_edgeからq[i].左までをq[n]でレンダリング ;
          l_edge := q[i].左 ;
          func(i) ; /* 再帰呼び出し */
          if( l_edge < p[i].右 ) continue ; else return ;
        case セグメントが交差 :
          交点を計算して区間を分割し再計算 ; break ;
      }
      l_edgeからq[i].左 までをq[n]でレンダリング ;
      l_edge := q[i].左 ;
      func(i) ; /* 再帰呼び出し */
      if( l_edge < q[n].右 ) continue ; else return ;
  }
  l_edgeからq[n].右までをq[n]でレンダリング ;
  l_edge := q[n].右 ;
} /* func */  /* 一部簡略化 */
```

5．まとめ

高速なスキャンラインアルゴリズムを提案し試作した．i80286を用いて１万ポリゴンのデータを640×480画素に数分で処理する．
数倍から数十倍以上の速度改善が得られているが，今後 Bouknight や Watkins のアルゴリズムに対する客観的な速度比較をおこないたい．

6．参考文献

1)Th. Ottmann,"Computational Geometry: Selected Algorithms and Paradigms",EUROCAL'85,April 1985
2)中前,西田,"３次元コンピュータグラフィックス",昭晃堂

Another Scan-line Algorithm

Suminobu AKIBA , Tsuyoshi YAMAMOTO , Yoshinao AOKI

Hokkaido Univ.