クライアント-サーバシステムにおけるポリゴン処理方式

5D－3

畑本 直樹　藤岡 誠　川辺 秀樹　武藤 信夫

NTT 情報通信研究所

1　はじめに

マッピングシステム等のベクトルデータを利用した画像処理において、領域を強調し見やすくするためにペイント表示が一般的に行なわれている。大容量データの集中管理に適したクライアント-サーバ(以下C/Sと表記)形態をとるシステムの場合、ポリゴンデータをそのまま処理する従来の方式では、C/S間データ転送時間やクライアントでの画像処理時間がかかるため、表示時間の短縮化を実現する高速なペイント処理方式が必要となる。本稿では、ポリゴンデータ自身をクライアントに転送せず、各ポリラインデータにペイント情報を付加し、表示時のクリッピング処理にてペイント対象のポリゴンデータを生成する方式を提案する。

2　前提条件

本稿では、C/Sの地図案内システムへの適用をねらいに、以下を前提とする。

1. ビューは背景に対して自由に設定できる
2. 強調表示のためのペイント領域を動的に制御できる
3. ペイント処理にはペイント代表点による標準的アルゴリズムを用いる
4. クライアントのメモリ使用量を節約する

3　ポリゴン処理方式

3.1　着眼点

ペイント処理では、ペイント領域を示すポリゴンデータは不可欠である。しかし、ポリゴンデータを境界を示すポリラインデータと重複して送ることはデータ転送量も多く、処理量も多い。したがって、ポリゴンデータはクライアントに送らない方式をねらいに、C/S間データ転送量やポリゴン処理量に着目し、以下の3項目について検討する。

1. ポリゴン表現方法
2. ペイント点の取得方法
3. クリッピング処理方法

3.2　方式検討

3.2.1　ポリゴン表現方法

ポリゴンデータを座標の集合として表現する方式は単純であるが、データ量が多くなってしまう。ポリラインにユニークな番号(ポリライン番号)を付与し、ポリゴンを構成する線分集合とポリライン番号の対応をとり、ポリライン番号のみを格納したポリゴンレコードを設ける方式がデータの縮小化等に有効である。ポリゴンレコードにポリライン番号を格納する方式では、ポリライン番号から座標への変換処理が必要となる。座標変換を処理するのをサーバ側とするか、クライアント側とするかで2案が考えられる。

3.2.2　ペイント点の取得方法

ビューを可変とするため、ポリゴンのペイントのための代表点を動的に取得する必要があり、以下の2案が考えられる。

1. ポリゴン方式:　クリッピングにより作成されたポリゴンから内部座標を求め、ペイント点とする
2. IN点方式:　ポリラインがビューに入る点(IN点と呼ぶ)から数画素分内側の点をペイント点とする
   本方式では、ポリラインの座標データをポリゴンと併用できる。IN点方式の概略を図1に示す。

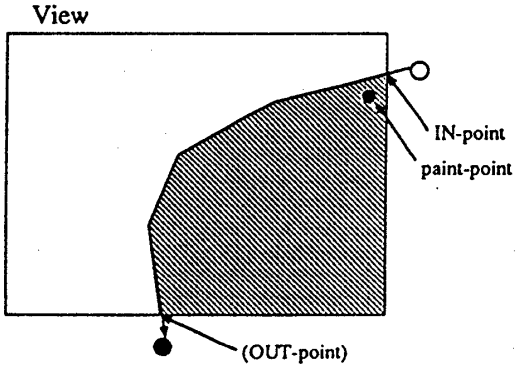

図1. IN点によるペイント処理

3.2.3　クリッピング処理方法

ポリゴンや背景となるポリラインがビューに入っているかを判断するクリッピング処理が必要となる。ペイント処理を行なうポリゴンのクリッピング処理はサーバ側で行なうかクライアント側で行なうか、2つの案がある。

3.3　方式比較評価

前項までで検討した方式の組合せにより複数の処理方式が考えられる。処理方式案を表1にまとめる。なお、表中、C及びSはそれぞれクライアント、サーバをあらわす。

Polygon processing for client-server style system
Naoki HATAMOTO, Makoto FUJIOKA, Hideki KAWABE, Nobuo MUTOH
NTT Information and Communication Systems Laboratories

表 1. ポリゴン処理方式案

案	ポリゴンレコード構成	座標変換処理	ペイント点取得	クリッピング処理
1	ポリライン番号	S	ポリゴン方式	S
2	座標	-	ポリゴン方式	S
3	ポリライン番号	-	IN 点方式	S
4	座標	-	IN 点方式	S
5	ポリライン番号	-	IN 点方式	C

ポリゴンレコードに座標を直接格納する方式(案2,4)はポリラインレコードがポリラインデータの数の2倍以上となり、例えば、地図での行政界の場合では、データ量が膨大となる。データ量の観点からポリライン番号を格納する方式が優れている。

ペイント代表点取得については、ポリゴン方式(案1,2)は、どのような場合でもペイント処理が可能である一方、ポリライン番号から座標に変換する必要があり、処理が複雑で、メモリを多く必要とし、送信データ量が多い。それに比べ、IN 点方式を採用した場合(案3,4,5)、ポリラインデータを使うため、座標変換処理が不要となり、処理が簡単で、しかも、送信データ量を抑えることができる。

クリッピング処理については、IN 点方式を採用することにより、ポリゴンのクリッピング処理は必要ない。また、ペイント対象のポリラインはクリッピング処理時にペイント代表点取得が可能なため、クライアントのビューを意識したクリッピング処理をサーバ側で行なう必要はない。

以上から、案5では、ポリゴンレコードに格納されているポリライン番号を座標に変換する処理と、ポリゴン座標データの送信が不要となるため、バッファメモリ量の削減、処理時間の短縮、データ送信時間の短縮、といった問題を解決することが可能となる。

3.4 アルゴリズム

前項で採用した案5のアルゴリズム及び C/S での機能構成を以下に示す。

サーバ側では、ペイント対象のポリゴンレコードから得られるポリライン番号から、送信ポリラインレコードを比較、一致するレコードに対しフラグを付与し、データを送信する。

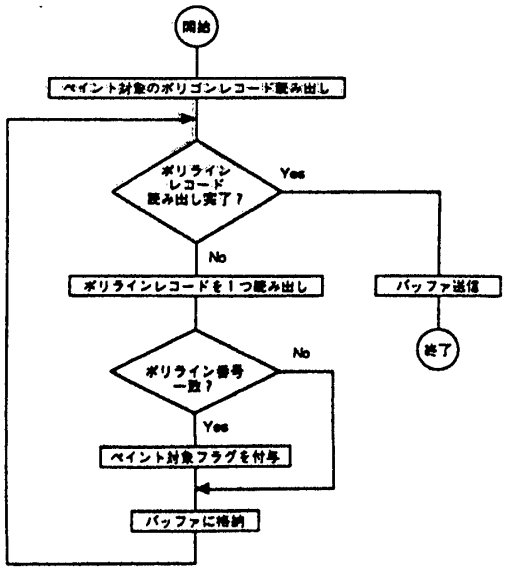

図2. サーバ側でのポリゴン処理

クライアント側では、送信されたデータのうち、ペイント対象のポリラインについて IN 点を算出、記憶しておき、描画処理が完了した後 IN 点でペイント処理を行なう。

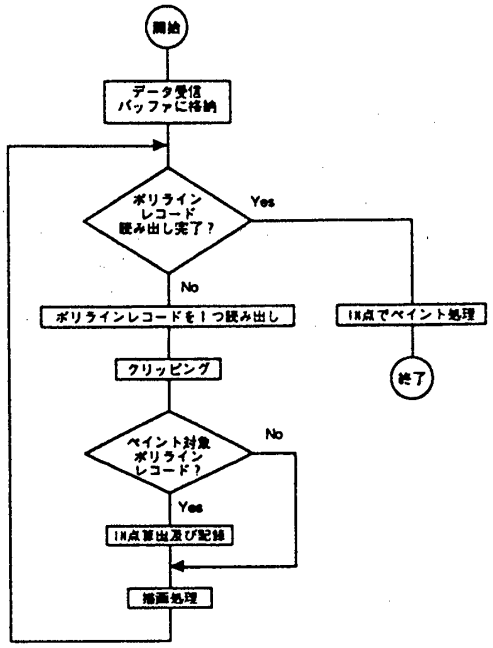

図3. クライアント側でのポリゴン処理

4 実装

本方式を実装した C/S 方式の地図案内システムをワークステーション(NEC UP/4800 シリーズ)を地図サーバ、パーソナルコンピュータ(NEC PC-9800 シリーズ)を地図表示クライアントとして実現した。本システムにおいては、海・湖、ゴルフ場などの他、行政区域の強調に動的にペイント領域を制御する本方式を適用した。データ量としては約80%、ペイント領域用のメモリ量としては67%の効率化を図ることができた。

5 おわりに

本稿では、C/S 形態において、サーバからポリゴンデータ自身をクライアントに転送せず、各ラインデータにペイント情報を付加し、表示時のクリッピング処理にてペイント対象のポリゴンデータを生成する方式を提案した。通信データ量の縮小化と処理の簡略化により、メモリ量が少ないクライアントでも十分なレスポンスが得られ、他の業務プログラムとの連携を容易にした。今後は、比較検証を行ない、本方式の有効性を実証する予定である。