

都市景観システムのための入力インタフェースの実現

- 地形入力 -

平本建志 加藤隆博 *間瀬実郎 河合和久 大岩元 *瀬口哲夫
(豊橋技術科学大学 情報工学系・建設工学系)

都市景観システムのための地形データを得る手段として、地形図の等高線から対話型の処理で地形データを抽出する方法を検討した。地形図の等高線には、道路、鉄道等の重ね書きがあり、1本の等高線全体を認識するのは困難である。そこで、前処理により、スキヤナから読み込んだ地形図を連続線(ブランチ)に分解する。標高値付与をブランチ単位に行うことで、入力ミスが少なくかつ効率的な地形データ抽出が可能となった。他の地形データ抽出方法との比較を交えて、本抽出方法の特徴、抽出実験結果、都市景観システムへの適合性について述べる。データ管理をオブジェクト指向で実現した。結果、河川、道路等の入力に応用できる設計である点についても言及する。

IMPLEMENTATION OF AN INPUT INTERFACE FOR A TOWNSCAPE SIMULATION SYSTEM - INPUT OF INFORMATION CONCERNING TERRAIN -

Takeshi HIRAMOTO Takahiro KATO *Jitsuro MASE
Kazuhisa KAWAI Hajime OHIWA *Tetsuo SEGUCHI

Dept. of Information and Computer Science,

*Dept. of Regional Planning

Toyohashi University of Technology

1-1, Hibarigaoka, Tempaku-cho, Toyohashi-shi, Aichi 441, Japan

We have investigated entering geographic data from printed maps and then processing it interactively as an input method for a townscape simulation system. The map is read by an image scanner and is preprocessed to identify branches on the map. The program then interacts with the user to assign altitudes to contour lines. This is an effective and relatively error-free method for reading geographic data. The advantages, disadvantages, and applicability of this system have been evaluated by comparison with other input methods. It is straightforward to extend the implemented object-oriented program to handle rivers and roads.

1. はじめに

CG (コンピュータ・グラフィックス) により、都市の景観を3次元で視覚的に表現することは、建築設計・都市設計の分野において、CADシステムとならんで主要なコンピュータの利用形態となろうとしている。CGによるプレゼンテーションは特に動画にした場合に、空間的な広がりや、マクロからミクロへの連続的な視点の移動等により、優れた訴求力を発揮する。建築設計・都市計画に携わる専門家に対してはもちろんのこと、特に、発注者や計画対象地域の住民という非専門家にとってたいへん分かりやすいプレゼンテーションが可能となる¹⁾。

しかし、CGによる都市景観シミュレーションを実現するためには、人手による、膨大な量の3次元都市データの入力が必要であり、このことが、都市景観シミュレーションにおける最大の問題といっても過言ではない。

このような現状を考慮し、我々は、多くの人手や専門的な知識を必要とせず、効率的に都市景観を構成する3次元情報を入力する方法について研究を行ってきた²⁾¹⁾³⁾¹⁴⁾¹⁵⁾¹⁶⁾。そして、その研究の一環として、都市景観システム「Swaco」の開発を行い、現在、建物と地形の入力が可能となっている。

本文では、都市景観システム「Swaco」における、地形図から対話形式で地形データを抽出する、地形データ入力について、他の方法との比較を交えて説明する。

2. 都市景観システム「Swaco」の概要

都市景観システムとは、建物や地形というさまざまな要素から構成される、都市の景観を、さまざまな視点から3次元で視覚的に表示する機能を備えたシステムである。建築設計・都市開発に携わるプランナーたちの支援を目的としたシステムである。

都市景観システム「Swaco」のシステム開発環境は、まず、使用機種に、EWS (エンジニアリング・ワークステーション) SUN 3/260を用いた。256色同時発色可能なディスプレイを使用している。開発言語はC++を使用した。この言語は従来のプログラミング言語Cにオブジェクト指向を盛り込んだものであり、Cの実用的な、処理速度、記述の自由度等を損なうことなく、オブジェクト指向の特徴である、モデルの階層表現、情報隠ぺい等を可能にした言語である。さらに、GUI (グラフィック・ユーザインタフェース) の実現のために、MITのXウィンドウ・システムを利用している¹⁷⁾¹⁸⁾¹⁹⁾¹¹⁰⁾¹¹¹⁾。

都市景観システム「Swaco」のデータ入力時には以下の2つのものが必要となる。

まず、平面的な位置情報の抽出を行うために、1/2,500程度の比較的大縮尺 (地形図では、例えば、1割る2,500を行った結果を用いて縮尺の大小が論じられる。つまり、1/2,500の地形図は1/50,000の地形図より大縮尺であるという) の地形図を用いる。この地形図は都市計画基本図と呼ばれ、市役所で青焼きとして入手可能である。このような、大縮尺の地形図を利用することとなった経緯については、3章で、「都市景観システムのための地形データとは」という観点から論じる。

また、地形図とは別にもう一つ、都市景観の構成要素の3次元的な情報を得るために、航空写真や現地調査の結果を用いる。これは、建物の回数、屋根の形状 (切妻、寄棟、入母屋等)、屋根の色というような、地形図だけでは分からない情報を得るためのものである。

現在、都市景観システム「Swaco」においては、地形と家屋の入力について実現できており、都市景観のさまざまな視点からの表示が可能である。

3. 地形データ入力のための原データ

都市景観システム「Swaco」の地形データ入力では、基本的に地形図に記述されている地形データだけを利用する (前述のように「Swaco」では入力時に、地形図と、航空写真または現地調査の結果、を必要とする)。ここでは、なぜ、データ入力のために地形図を利用し、さらに、1/2,500程度の大縮尺の地形図を用いることになったかについて説明する。

地形データを入手する方法としては、地形図を用いる方法以外にも、いくつかの方法が考えられる。

もっとも基本的な方法は、現地で地形の測量を行うことである。この方法は、もっとも正確な地形データを得る方法の一つであるが、比較的大規模なプランニングを行う場合ならともかく、小規模なものや候補地の検討段階では採算が合わない、あるいは、データ入手までの時間が掛かりすぎると考えられる。

次に考えられるのは、「国土数値情報」を国土地理院から購入することである。これは、国土の標高がメッシュ状に記述されたデータであるが、メッシュ間隔が比較的大きい点に問題がある。都市景観システムの対象としているユーザは、建築設計・都市開発に携わるプランナーである。したがって、基本的には人間が造成できるレベルの、細かい地形の形状が地形データとして要求される。また、視覚的に地形を表現する場合にも、人間の視点に立てばよ

り細かい地形の抑揚が見えてしかるべきである。以上の点から、「国土数値情報」ではデータの細かさが(サンプリング区間が広い)ため不足しているといえる。

また、最近では、「リモートセンシング」と呼ばれる人工衛星を利用するものが登場している。筆者は、この方法に関する知識が乏しいため正確な議論は避けるが、現状では、都市景観システムに用いるには、標高値の精度が不足していると思われる。少なくとも、低コストで身近から入手する事はできない、ということは事実である。

以上の点から、現状では、地形図の等高線を利用して地形データを入手することが、都市景観システム「Swaco」においては最適であると考えた。

さらに、さまざまな縮尺のものが存在する地形図のうち、都市景観システムに適した縮尺について検討した結果、「国土数値情報」の議論で述べたように細かい地形の形状が必要であるという点と、また、入手が容易であるという点から、2m間隔で等高線が記述されている1/2,500の都市計画基本図から等高線を抽出することとした。

4. 対話型地形データ入力

都市景観システム「Swaco」では、基本的に、対話型処理でデータ入力を行うものとしている。地形データについても同様で、対話型処理で地形図から地形データの抽出を行う。ここでは、地形データの抽出を対話型で行う理由を説明する。

地形図から等高線を自動的に抽出する研究は、すでにいくつかのものが行われている^{[12][13]}。しかし、いずれの方法も認識結果に問題を残しており、認識漏れや認識誤りを補うための人間の判断・指示による後処理が必要である。特に、本システムが地形データの抽出対象とする1/2,500の大縮尺の地形図では、

- ・単色刷りである
- ・青焼きのため画質が悪くスキャンした際にノイズが入る
- ・記述にフリーハンドに近い記述が行われている

というような特徴があり、認識漏れや認識誤りが増加することが予想され、人間による後処理の割合がきわめて大きくなると考えられる。

そこで、図面を対話型の処理を用いることで、より効率的な入力を実現しようとする研究がいくつか行われており^{[14][15]}、本研究においても、オペレータの入力を仰ぎながら徐々に地形データを作成し

ていく、対話型の処理がもっとも実用的なシステムを構築し得ると考え、対話型の地形データの抽出方法を探ることとした。

5. 地形データの入力方法

今回、都市景観システム「Swaco」の地形データ入力方法として、「線追跡」による抽出方法を開発した。ここでは、この「線追跡」による地形データ抽出方法と、われわれの研究室で先に開発した「等高線間帯の塗り込み」による抽出方法^{[2][13]}と、さらに、地形図から地形データを入力する一般的な方法と考えられる「標高値のポインティング」による方法について、それぞれの入力方法についてオペレータの入力作業を中心に説明する。

5.1 「標高値のポインティング」による地形データの抽出

「標高値のポインティング」による方法では、代表的な図面入力装置であるデジタイザを使用する。デジタイザに地形図を張り付けておき、等高線上をクリックしながらトレースしていくことで標高値を抽出するという、地形図から地形データを抽出する方法としては一般的な方法である。

オペレータが行う作業は、「キーボードから標高値を入力しておき、その標高値に対応する等高線を、ポインティングデバイス(デジタイザのカーソル)で、たどりながらクリックしていく」という動作の繰り返しになる。

この方法では、地形図は人間が参考にするだけでコンピュータには入力されておらず、完全にオペレータの手入力である。

5.2 「等高線間帯の塗り込み」による地形データの抽出^{[2][13]}

「等高線間帯の塗り込み」による方法は、イメージスキャナから読み込んだ画像を、標高値を定義した色で塗り分けることで標高値を抽出しようとするものである。地形図を標高値が定義してある色で塗り分けることにより、各座標での標高値は、その場所の色を調べることで求められる。

オペレータが行う作業は、「メニューの中から標高値を選んで、その標高値に対応する等高線間帯(等高線と等高線に囲まれた閉領域)をクリックする」ということの繰り返し作業になる。

図1を用いて、より具体的に入力作業について説明する。

オペレータは、これから塗り込みを行おうとする

等高線間帯の色（標高値を表現する）を、図1 (a) の様な色選択ボタンから選択する（この図では30m を選択している）。次に、図1 (b) の様に入力しようとする等高線間帯の中をクリックする。すると、図1 (c) の様に、ペイントルーチンが起動し等高線間帯に色が塗られる。以上の作業を各等高線間帯に対して行うことにより、図1 (d) の様な標高値を定義した色で塗り分けた地形図を作成する（つまり、標高値の抽出を行う）。

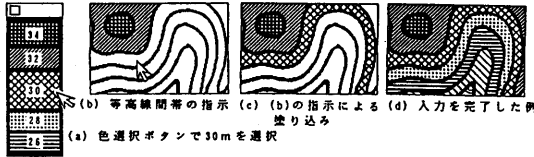


図1 「等高線間帯の塗り込み」による地形データ抽出

この方法では、特に都市部での地形データ抽出に問題が多く、都市景観システムへの適合性が低いため、次に示す「線追跡」による方法を改めて開発することになった。なお、それぞれの地形データ抽出法の問題点については、後ほどまとめて議論する。

5. 3 「線追跡」による地形データの抽出

「線追跡」による方法では、入力を開始する以前に前処理として、

- ・細線化処理 L16JL17
- ・線画像の分解処理

を行うことにより、画像をあらかじめノードとブランチに分解しておく（図2）。ここで、ノードとは、線画像中の端点や分岐点や孤立点であり、また、ブランチはノードとノードをつないでいる線分である。

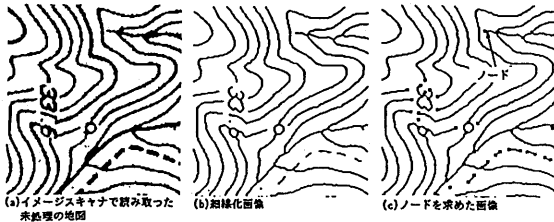


図2 「線追跡」による地形データ抽出（前処理）

図3 に示す画像パターンではノード4つとブランチ3つに分解されることになる。「線追跡」による方法は、それぞれのノードとブランチに属性として標高値を持たせることで、地形を表現しようという方法である。

オペレータが行う作業としては、「メニューの中

から標高値を選んで、その標高値に対応する等高線付近をクリックする」ということの繰り返し作業となる。

図4 を用いてより具体的に説明する。

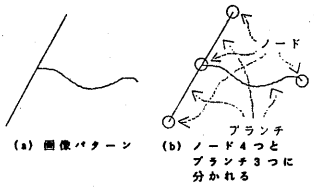


図3 ノードとブランチの分解例

オペレータはこれから入力しようとする等高線の標高値を、図4 (a) の様な標高値選択ボタンから選択する（この図では280m を選択している）。次に、図4 (b) の様に入力しようとする等高線の近傍をクリックする。すると、図4 (c) の様に分岐点間のブランチに標高値が付与される。図4 の例では、入力対象の等高線が、川と広葉樹の記号により、5つのブランチに分断されているため、それぞれのブランチの近傍で5回のポインティングが必要である。

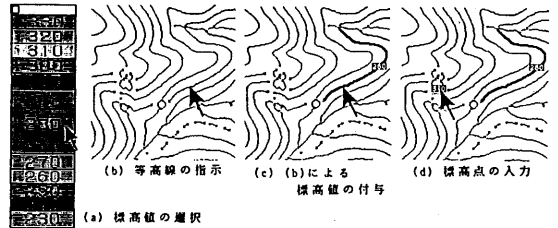


図4 「線追跡」による地形データ抽出（オペレータの作業）

また、図4 (d) では、標高値を入力したくても、属性を付与すべきノードもブランチも存在しない場合の入力方法を示している。図4 では、等高線中に標高値を表す331.6という文字が記述されている。そのため、標高値310m を付与するブランチが一部存在しない。このような場合のために、「線追跡」による方法では、新たなノードを追加する形で標高値310m の入力を実現している。

6. 各地形データ入力方法の特徴

ここでは、「標高値のポインティング」による方法、「等高線間帯の塗り込み」による方法、「線追跡」による方法のそれぞれについて特徴をまとめる。

- ・「標高値のポインティング」による方法の特徴

① この方法では、クリック回数により得られるデータ量が決定する。したがって、より細かく地形のデータを入力しようとする、それだけ多くのサンプリングが必要となる。言い替えれば、1本の等高線に対するクリック回数をそれだけ多く行う必要があるというこ

とである。

- ② 等高線の位置をより正確に入力しようとする場合、それだけ、入力位置を正確にポイントニングする必要がある。必然的に、ポイントニング・デバイスの位置合わせに時間をかける必要がでてくる。デジタイザのスクープを等高線上により正確に合わせようとする、オペレータのポイントニング作業が慎重に行われるのが普通である。
- ③ 以上のことから分かるように、入力を行うオペレータにより、地形データの質が大きく左右されると考えられる。

・「等高線間帯の塗り込み」による方法の特徴

- ① きちんとした形で等高線間帯が存在する場合、つまり、等高線に跡切れが存在して等高線間帯が閉領域になっていなかったり、あるいは、他の要素（道路、家屋、鉄道、文字など）の存在や、等高線と等高線が非常に接近して、等高線間帯が複数の閉領域に分割されていなければ、ひとつの等高線間帯に対して一回のポイントニングしか必要としない。
- ② 色を塗り込むことで入力を進めるため作業状況を把握しやすい。山岳部中心の画像
- ③ 等高線が分断されていた場合、隣の等高線間帯に塗り込みが行われ、入力ミスが発生する（この場合、塗り込みを行った色を元の色に戻し、等高線の途切れている部分を修正し、再び塗り込みを行うことになる）
- ④ ただし、入力ミスは視覚的に非常に分かりやすい。

・「線追跡」による方法の特徴

- ① きちんとした形で存在する等高線、つまり、跡切れや、他の要素（道路、家屋、隣の等高線）との接触や重なりがなく、閉曲線として存在する等高線に対しては、ひとつの等高線に対して一回のポイントニングしか必要としない。
- ② 入力を、ブランチ単位に行う、つまり、分岐点までにとどめるため、等高線が他の要素と接触していたり重なりあっても、入力ミスが起こりにくい。
- ③ オペレータがブランチあるいはブランチに標高値を付与する場合には、

そのノードあるいはブランチの近傍をクリックすればよいと、正確な座標のポイントニングを必要としない。したがって、疲労感も少なく素早いポイントニングが可能と考えられる。

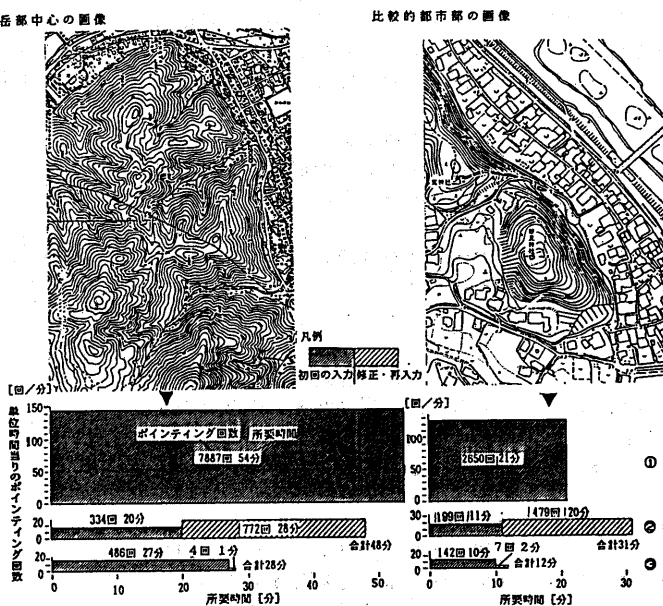
- ④ 等高線の画像上での座標は、ノードとブランチに分解した時点でコンピュータが求めており、オペレータには近傍の座標の入力しか要求しないため、オペレータによる地形データの質の変化が少ない。

7. 地形データの抽出実験と都市景観システムへの適合性

各入力方法で、実際に地形データの抽出実験を行った結果を図5に示す。

山岳部中心の画像と、家屋や道路といった情報と等高線がある程度混在した比較的都市部の画像についてそれぞれ抽出作業を行った。

「標高値のポイントニング」による方法では、特徴①で述べたように非常にポイントニング回数が多いことが分かる。単位時間あたりのポイントニング



このグラフは、縦軸に単位時間当たりのポイントニング回数、横軸に所要時間をとっている。したがって、グラフの面積は総ポイントニング回数にあたる。

- ①：「標高値のポイントニング」による地形データ抽出
- ②：「等高線間帯の塗り込み」による地形データ抽出
- ③：「線追跡」による地形データ抽出

図5 地形データの抽出実験結果

回数の多さは、デジタイザがポインティングという操作に非常に適した装置であることや、ポインティングを行う際のデジタイザのカーソルの移動量が少ないことによると考えられる。

対して、「等高線間帯の塗り込み」による方法では、ポインティング回数そのものは、特徴①で述べた理由から減少している。しかしながら、特徴③で述べた、等高線の分断による入力ミスが多く、そのための、修正・再入力が必要とする時間が非常に多く掛かっていることが分かる。山岳部中心の画像では「標高値のポインティング」による方法より処理時間が短縮されたものの、都市部の画像では逆に処理時間が増加している。

以上の結果から分かるように、「等高線間帯の塗り込み」による方法では、等高線の分断に弱く、特に、都市景観システムが必要とする都市部の地形データ抽出には不向きであった。

本システムの入力方法である「線追跡」による方法では、ポインティング回数は特徴①で述べたように、「等高線間帯の塗り込み」による方法と同等に少なく、また、特徴②で述べたように、入力ミスを補う修正・再入力に必要とする時間も非常に少なく、「標高値のポインティング」による方法のほぼ半分の時間で入力が行われている。特に、入力ミスが増加しがちな都市部の画像に対しても、良好な結果を得ている。

以上の結果から、「線追跡」による地形データの抽出方法は、都市景観システムの地形データ入力方法として実用に耐え得ると考えられる。

8. 地形データ入力部のクラス設計

ここでは、都市景観システム「Swaco」の実現に際し行った、地形データ入力部のクラス設計について述べる。

今回行ったクラス設計は、図面入力を対話型で進めるためのものである。対話型図面入力のためのクラス設計を、地形図の等高線入力に応用したと理解してもらおうとよい。

8.1 画像の一括管理

地形データ入力部のクラス設計では、まず、画像全体をクラス「画像」が一括して管理するものとした。いいかえると、図面の各構成要素を全てクラス「画像」が管理する。

また、画像の各構成要素は、全てクラス「図形」の導出クラスで表現する。つまり、画像全体はクラス「図形」の導出クラスの集合として表現されるこ

とになる。

画像を構成する各構成要素全てを、クラス「図形」の導出クラスとすることにより、クラス「画像」は、個々の構成要素を、道路のグループ、等高線のグループ、河川のグループというように、別々に管理する必要がなくなり、全てクラス「図形」のグループとして一括して管理することが可能となっている（図6）。

また、クラス「図形」は、各構成要素の基本クラスとなる。クラス「図形」は特にデータをもっておらず、クラス「画像」が利用するメンバ関数（メソッド）が記述されている。例えば、メンバ関数の一つに「ある座標からの距離を返す」というものがあるとすると、この場合、具体的な距離計算のアルゴリズムは各構成要素により異なるため、クラス「図形」にはなんら記述されておらず（つまり、仮想関数）、各構成要素を表現する導出クラスごとにそのアルゴリズムは記述される。つまり、クラス「図形」には、クラス「画像」が必要とするメンバ関数（実際には仮想関数）の一覧が定義されている。

クラス「画像」は、クラス「図形」に定義されているメンバ関数を呼ぶ形で全ての処理が記述されている。したがって、クラス「図形」が実際にどのような導出クラスであろうとクラス「画像」の処理にはなんら影響を与えない。

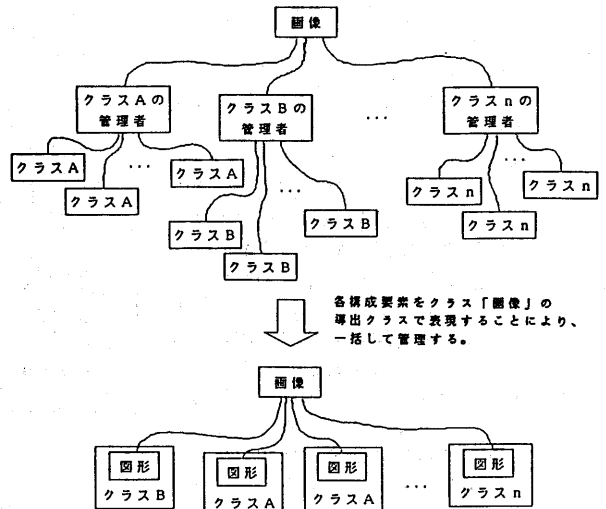


図6 クラス「画像」による一括管理

8.2 初期状態

5.3ですです述べてのように、入力を開始する以前に前処理として、

- ・細線化処理
- ・線画像の分解処理

を行うことにより、画像をあらかじめノードとブランチに分解する(図2)。ノードは、ノード自身の座標と、そのノードに付属しているブランチへのリンク情報で表現され、また、一方のブランチは、始点と終点の座標と、始点から終点への画素の並びで表現される。

この前処理により、画像全体をノードとブランチの集合に変換できたことになる。

そこで、クラス「画像」の入力を行う以前の初期状態として、図7の様に、各ノードとブランチの情報を、クラス「図形」の導出クラスであるクラス「図形ノード」、「図形ブランチ」という形で持たせることで、画像を表現することとした。

8.3 地形データ入力への応用

本システムでは、地形データを入力する目的で使用するので、あらかじめ、各クラス「図形ノード」と「図形ブランチ」を、標高値用の変数を追加した導出クラスであるクラス「等高線ノード」と「等高線ブランチ」に変換した(図8)。

クラス「等高線ノード」、「等高線ブランチ」の標高値用変数に標高値を書き込む形で地形データの入力が進められる。

8.4 他の図面入力への応用

上述のように、クラス「画像」はクラス「図形」に対する処理として記述されているため、新たな導出クラスの追加が容易である。

図9の例では、地形データの入力と同様に、河川、道路の抽出を行う場合を表している。図7に示す初期状態から、ユーザの指示を仰ぎながら入力を進め、例えば、あるブランチに対して、河川であるという指示が行われた場合には、クラス「図形ブランチ」であったものをクラス「河川ブランチ」に変換することで図面の入力が進められる訳である。

「河川ブランチ」がクラス「画像」に追加されたとしても、クラス「画像」からはクラス「図形」として扱われることに変わりなく、画像全体の管理者であるクラス「画像」の変更は、この場合も必要としない。

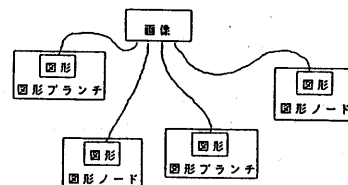


図7 クラス「画像」の初期状態

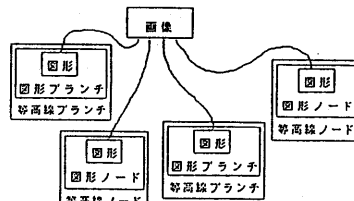


図8 地形データ入力用のクラス構成

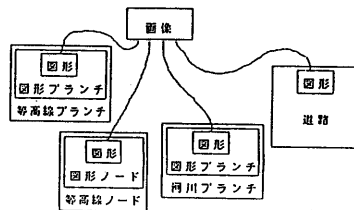


図9 河川、道路等の入力への応用

以上のように、非常に拡張性が高いクラス構成となっており、将来的に、道路や、河川等のクラスを追加し、地形データと同様に管理することが可能であると考えられる。

また、地形図以外の図面、例えば、製図図面の読み取りに利用し、ブランチの属性として、直線、折線(連続直線)、円、円弧、自由曲線、長方形、文字等の属性を付与する作業を行い、最終的に、図形ブランチの持つ情報を利用して、ベクター化したCADデータを生成する、というような応用も考えられる。

9. まとめ

本研究では、都市景観システムのための地形データ入力方法として、「線追跡」による方法を開発した。この方法により、簡単に入手可能な都市計画基本図より、都市景観システムに必要とされるレベルの地形データを、デジタイザを用いた「標高値のポイントイング」による方法の半分の作業時間で、得ることが可能となった。特に、オペレータの疲労度という点を考慮すると、有用な手法であると考えられる。

また、地形データ入力部をオブジェクト指向で設計したことにより、将来的に、道路や河川などの入力にも応用できるものとなった。

参考文献

- [1] 笹田 剛史：「建築設計・都市計画におけるコンピュータグラフィックス」、情報処理 Vol. 29 No. 10 pp. 1177-1180 (1988)
- [2] 間瀬 実朗、瀬口 哲夫：「等高線塗り分けによる地形データ入力支援システムの研究」、都市計画論文集 No. 23, pp. 457-462 (1988)、日本都市計画学会
- [3] 間瀬 実朗、瀬口 哲夫、河合 和久、大岩 元：「3次元地図データベースの入力方法の1考察 - 2次元ラスタ画像による地形・建物データの入力 -」、情報処理学会研究報告、情報学基礎89-FI-16 (情報処理学会) 16-4 (1989)
- [4] 間瀬 実朗、瀬口 哲夫：「都市景観シミュレーションのための建物形状データ入力支援システムの研究」、都市計画論文集 No. 24, pp. 421-426 (1989)、日本都市計画学会
- [5] 間瀬 実朗、瀬口 哲夫：「都市景観シミュレーションのための建物形状データ入力支援システムの研究 - 外壁開口の表現 -」、都市計画論文集 No. 25, pp. 265-270 (1990)、日本都市計画学会
- [6] 間瀬 実朗、瀬口 哲夫、加藤 隆博、平本 建志、河合 和久、大岩 元：「都市景観システムのための地図情報入力インタフェースについて」、第6回ヒューマン・インタフェース・シンポジウム論文集 pp. 313-320 (1990)、計測自動制御学会 ヒューマン・インタフェース部会
- [7] 辺見 悦子：実用 X-Window、日刊工業新聞社
- [8] 木下 凌一：X-Window Ver. 11 プログラミング、日刊工業新聞社
- [9] Oliver Jones(著)、西村 亨(監修)、三浦 明美、ドキュメントシステム(訳)：X Window ハンドブック、アスキー出版局
- [10] B. ストラウストラップ(著)、齊藤 信男(訳)：プログラミング言語C++、トッパン
- [11] R. S. ウイナー、L. J. ロビンソン(著)、前川 守(訳)：C++:オブジェクト指向プログラミング、トッパン
- [12] 森 正寿、瀬戸 浩昭、中村 彰：「地形図における3次元情報の自動抽出とその応用」、情報処理 Vol. 29 No. 3 pp. 221-232 (1988)
- [13] 山田 豊通、鈴木 智、星野 肇夫、小杉 信：「地図認識入力システムMARI S - 概要 -」、情報処理学会第33会(昭和61年後期)全国大会 4N-1 pp. 1479-1480 (1986)
- [13] 志沢 雅彦、鈴木 智、山田 豊通：「地図認識入力システムMARI S - 前処理 -」、情報処理学会第33会(昭和61年後期)全国大会 4N-2 pp. 1481-1482 (1986)
- [13] 鈴木 智、志沢 雅彦、山田 豊通：「地図認識入力システムMARI S - ベクトル化処理 -」、情報処理学会第33会(昭和61年後期)全国大会 4N-3 pp. 1483-1484 (1986)
- [13] 鈴木 智、星野 肇夫、山田 豊通：「地図認識入力システムMARI S - 自動認識処理 -」、情報処理学会第33会(昭和61年後期)全国大会 4N-4 pp. 1485-1486 (1986)
- [13] 鈴木 智、金子 透、山田 豊通：「地図認識入力システムMARI S - 知的会話処理 -」、情報処理学会第33会(昭和61年後期)全国大会 4N-5 pp. 1487-1488 (1986)
- [14] 山川 修三、小田 泰充：「会話型図面入力システム」、日経CG No. 7 PP. 120-130 (1987)、日経マグローウヒル社
- [15] 大沢 裕、滝嶋 康弘、坂内 正夫：「会話的な認識による信頼性の向上を図った地図自動入力システム」、信学会 D-II Vol. J72-D-II No. 4 pp. 545-554 (1989)
- [16] 安居院 猛、中島 正之、長尾 智晴：Turbo Pascal 画像処理の実際、工学社
- [17] 田村 秀行：「多面的画像処理とそのソフトウェア・システムに関する研究」、電子技術総合研究所研究報告 第835号 (1983)