階層構造を有する複数縮尺地図ベクトルデータの一元的管理方式

6U－4

飯村 伊智郎　加藤 誠巳
熊本県立技術短期大学校　上智大学

1. まえがき

日本全国を対象とした道路網の経路案内システムのような大規模ベクトル地図データを描画するグラフィックス・システムにおいては，その描画時間が問題となる．また，地図を拡大表示する場合，より多くのノード情報が必要となるが，逆に縮小表示する場合には，ディスプレイの解像度を考慮すると，多くのノード情報は冗長となる[1]．一般に，カー・ナビゲーション・システム等で使用されているベクトル地図データは，縮尺毎に独立したデータを用意し，描画時に使い分けている．本稿では，これら複数個の縮尺のベクトル地図データを一元的に管理する方式を提案し，階層数が大きい場合，HD（大容量記憶装置）使用量の面で極めて有利であることを明らかにする．

2. 一元的管理方式

2.1. 概念

8 個のノードで構成される折れ線 L を考える．縮尺に応じて必要なノードのみを描画するために，これらのノードに対し以下の説明では，ノードを 4 段階に階層付けるものとする[2] [3]．その結果得られた折れ線 L の例を以下に示す．

$$L=\left\{P_1^1,P_1^2,P_2^1,P_2^2,P_1^3,P_1^4,P_2^3,P_3^1\right\}$$

ここで，P_j^i は，i 階層で描画される j 番目のノードであることを示す．次に，階層付けされたノードを元に，図 1 のような木を構築する．ここで，Q_j^i は，i 階層にある j 番目のダミーノードであり，$i+1$ 階層にあるノード及びダミーノードへのポインタである．

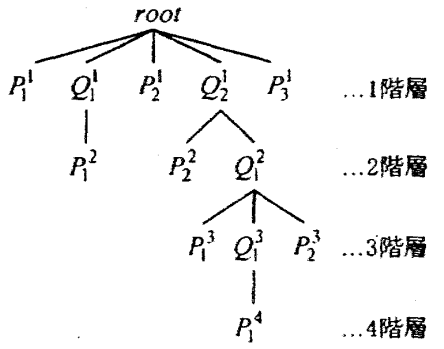

図 1　折れ線 L の木による表現

折れ線 L をこのように LISP[4]で使用されているリスト構造（木構造）で管理することにより，縮尺に応じて必要な階層のノードのみを容易に抽出して描画することが可能になる．すなわち，縮小表示をする場合，細部の情報は不要のため，1 階層のみを描画すればよく，また拡大表示をする場合，より多くの情報が必要となるので，より下の階層まで描画すればよい．

2.2. 評価

縮尺に関係なく常に全てのノードを描画する方式（方式 1），縮尺に応じて独立したデータを用意する方式（方式 2），そしてここで提案する方式（本方式）に対し，HD 使用量，実行時 RAM 使用量，そして描画時間を評価する．

条件：

- k 個のノードを持つ折れ線 L を n 階層に分割（n 階層まで描画すると最も詳細な折れ線が描ける）．
- k_i を i 階層までの描画に必要なノード数とする．
- d_i を i 階層目で必要となるダミーノード数とする（但し，$d_n=0$）．
- u_j^i を i 階層にある j 番目のダミーノードが持つノード及びダミーノードの数とする．この場合，k_i，d_i，u_j^i の間には次の関係式が成り立つ．

$$(k_{i+1}-k_i)+d_{i+1}=\sum_{j=1}^{d_i}u_j^i$$

HD 使用量：

HD 使用量は，格納すべきノード及びダミーノードの数で評価する．本方式の HD 使用量は，次式で与えられる．

$$k_1+\sum_{i=1}^{n-1}(k_{i+1}-k_i)+\sum_{i=1}^{n}d_i=k_n+\sum_{i=1}^{n}d_i$$

各方式の HD 使用量は，表 1 のようになる．

表 1　HD 使用量

方式 1	方式 2	本方式
k_n	$\sum_{i=1}^{n}k_i$	$k_n+\sum_{i=1}^{n}d_i$

実行（描画）時 RAM 使用量：

RAM 使用量は，m（$m\le n$）階層までの描画に必要なノード及びダミーノードの数で評価する．但し，ここでは，描画対象データはメッシュ分割等を行わずに，全て一括して取り扱う場合について考察している．本方式の RAM 使用量は，次式で与えられる．

$$k_1+\sum_{i=1}^{m-1}(k_{i+1}-k_i)+\sum_{i=1}^{m}d_i=k_m+\sum_{i=1}^{m}d_i$$

各方式の RAM 使用量は，表 2 のようになる．

A Unified Managing Method for Multiple Scale Vectorized Map Data having Hierarchical Structure
Ichiro IIMURA[1], Masami KATO[2]
[1]Kumamoto Pref. College of Technology, [2]Sophia University

表 2　RAM 使用量

方式 1	方式 2	本方式
k_n	k_m	$k_m + \sum_{i=1}^{m} d_i$

描画時間：

描画時間は，RAM 内のノード及びダミーノードを処理する時間と考えると，RAM 使用量に比例すると考えられる．

次に，$k=10{,}000$，$n=100$ と仮定し，階層が増す毎にノード数が線形に増加するケース L_1 と，指数的に増加するケース L_2 の 2 種類の折れ線を考える．図 2 は，2 種類の折れ線の階層 i とノード数 k_i の関係を示している．

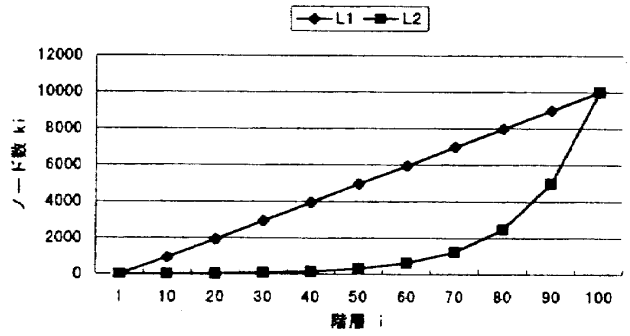

図 2　階層 i とノード数 k_i の関係

このような 2 つの折れ線に対し前述の 3 つの方式を適用した場合の HD 使用量は，表 3 のようになる．但し，u_j^i は i，j に関係なく $u_j^i \equiv u=5$ とした場合について考察している．

表 3　HD 使用量

折れ線	方式 1	方式 2	本方式
L_1	10,000	500,545	12,568
L_2	10,000	148,287	12,547

また，表 3 は階層数 n が 100 の場合であるが，階層数 n を変化させた場合の HD 使用量の変化を図 3 に示す．

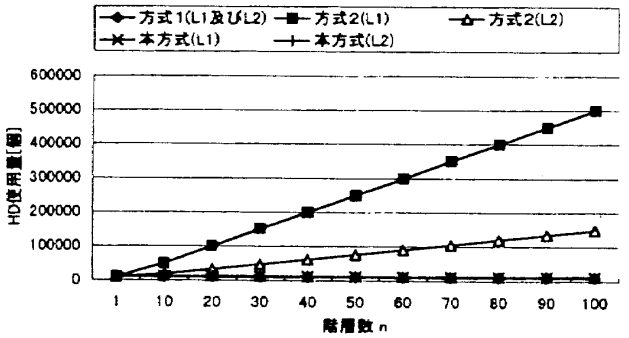

図 3　階層数 n と HD 使用量の関係

次に，階層数 n を 100 とし，m 階層までの描画で用いる RAM 使用量を図 4 に示す．（但し，$u=5$ とする）．

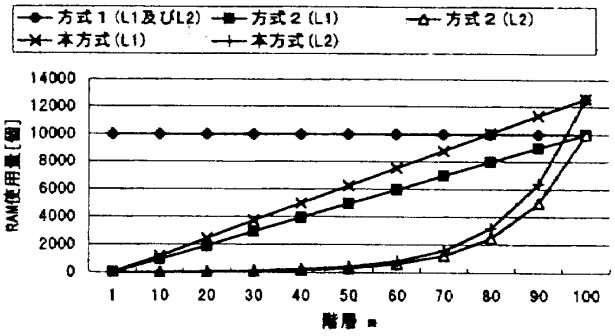

図 4　階層 m と RAM 使用量の関係

本方式は，$m=100$ の場合，実行時の RAM 使用量が方式 2 に比べ約 1.3 倍になるものの（図 4），HD 使用量を L_1 で約 1/40，L_2 で約 1/12 に減らすことができる（表 3）．

3. 実際のデータへの適用例

ここで対象としたデータは，日本デジタル道路地図協会の全国デジタル道路地図データベースから，佐渡島，種子島，屋久島の海岸線 3 種類と，大垂水峠，ヤビツ峠の道路区間 2 種類を抽出したものである．

これら 5 種類のデータに対して，本方式を $n=5$ として適用した結果を図 5 に示す．この場合，5 階層と階層分けが少ないにもかかわらず，本方式の方が方式 2 よりも有利であることがわかる．

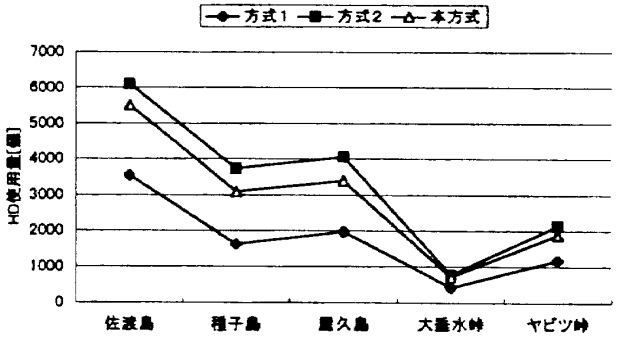

図 5　HD 使用量

4. むすび

階層構造を有する複数縮尺地図ベクトルデータの一元的管理方式とその有利性について述べた．今後は，日本全国を対象とした大規模なベクトル地図データに対し本方式を適用し評価していく予定である．

参考文献

[1] H. Hoppe: "Progressive Meshes", Computer Graphics (SIGGRAPH '96 Proceedings), pp. 99-108(1996-08).

[2] 加藤，大西，石田: "ディジタル地図における縮尺に応じたノード間引き手法"，情処第 46 回(平成 5 年前期)全大，8Q-8(1993-03).

[3] 加藤，石田，能見，倉川: "ディジタル地図におけるノードのランク付けに関する検討"，情処第 50 回(平成 7 年前期)全大，3T-10(1995-03).

[4] P.H.Winston, B.K.P.Horn 著，白井，安部訳:「LISP」，培風館，p.385，昭和 57 年．