位相情報を用いたリスト構造による図形表現に関する一考察

梶 谷 哲 也*

A Study of Data Management Method Using Topological Information for Graphic Design Systems

Tetsuya Kajitani

要　旨　別稿[1]にて提案したデザイン支援装置の目的は，主に，生体に基づく形状デザインである。ここで，生体に基づくデザインの特徴は図形の形状（位相情報）は類似している一方で，デザインを適用する生体間の個体差が大きいことが挙げられる。その例は，被服構成学の体型別参考人体寸法[3]などでも明かである。

そこで，本紀要では，生体の特徴であるデザインの形状に関する位相情報は保存し，その一方で，生体間の個体差を簡便な手段で調整できる補間点間のデータ構造を新たに提案し，その具体的内容とそのデータ構造に基づく図形データの処理方式について考察した。

1. はじめに

これまでに，デザイン支援装置の主な使用目的[1]と，その基本的実現手法である補間関数について考察を加えた[2]。ここで，別稿[2]で問題となった閉曲線に関する補間関数は，閉曲線の始点と終点の曲率の連続性を保証するスプライス関数を用いれば解決する。したがって，2次元図形は，図1のように，まず描画する図形を複数の補間点に分解し，開曲線用の補間関数と閉曲線用の補間関数の2種類の補間関数を併用すれば描画できる。この時，複数の補間点間をむすぶ図形は，図1の一点破線のような折れ線図形となる。ところが，これまでに，これら複数の補間点間をむすぶ直線図形を対象とするデータ構造については考察されていない。

そこで，本紀要では，生体に基づくデザインの効率的支援を目的として，図形の形状に関する基本的な位相情報と，図形の大きさ（＝個体差）を分離して表現する補間点間のデータ構造を新たに提案し，その具体的な構造とその要素に対する処理方式について検討を加える。

2. 図形の基本情報

描画する図形を規定する要素は，形状そのものに関する情報と，その図形の大きさに関する情報の2つに分離することができる[5]。

ところが，一般的に，図形の情報はそれを構成する頂点（各補間点）のx, y座標の絶対量として表現することが多く，図形の形状の情報とその大きさの情報を独立して扱うことはできない。その他の表現方法としては極座標表現の絶対量として表現する場合がある。この極座標表現では，2点間を結ぶ直線を直線の傾きと，直線の長さの成分の2つに分解して表している。

同様に，本紀要で提案するデータ構造の要素は，図2のように，まず基準となる線分r_1を決定し，さらにその他の線分（例えば線分r_2）の長さを全てr_1との比率で表す一方で，線分の角度に関する情報は，1つ前の線分（例えば線分r_1）との相対角で表す。その結果，図形

* 本学講師　人工知能

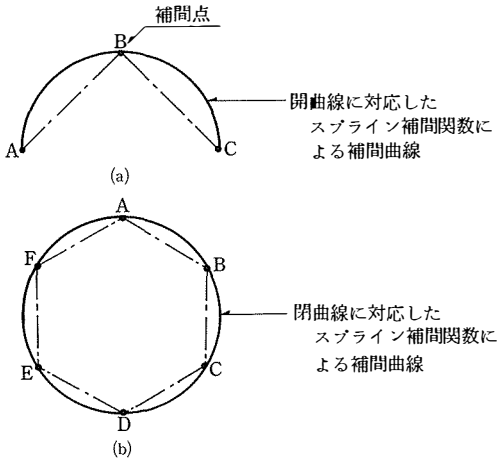

図 1　補間関数による開曲線と閉曲線の描画

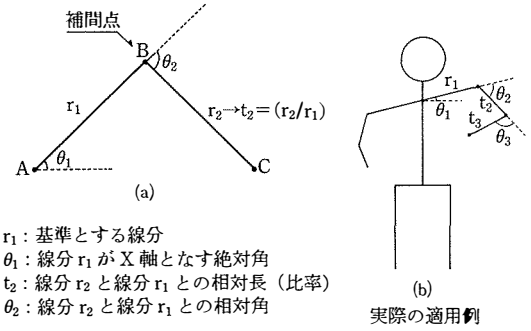

図 2　補間点の位置情報の表現

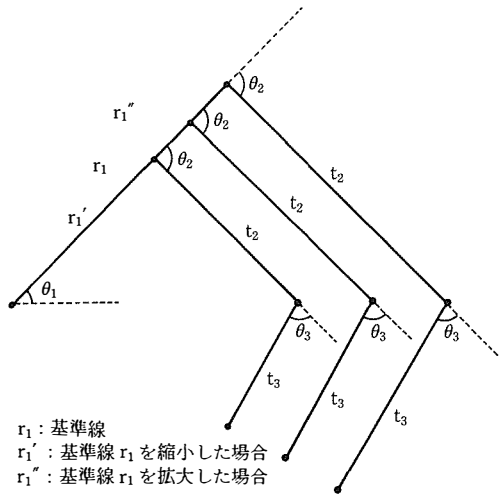

図 3　線分 $\mathbf{r_1}$ を基準とした図形の相対的な拡大，縮小

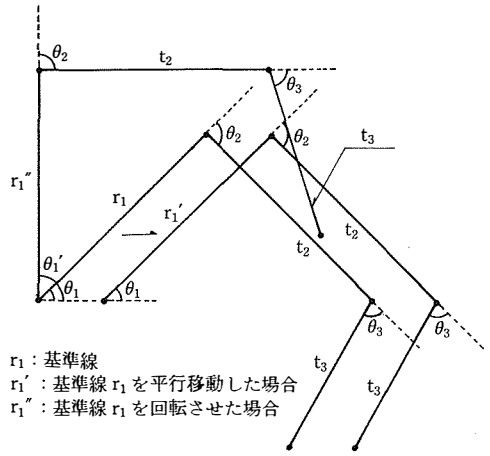

図 4　線分 $\mathbf{r_1}$ を基準とした図形の移動，回転

の形状そのものに関する情報と，その図形の大きさに関する情報は独立する。

従って，本方式で図形を表現した場合，まず，図 3 のような r_1 を基準とした図形の相対的な拡大，縮小処理は，基準となる r_1 の絶対長を変更することだけでその他の補間点(図形)も一元的に決定される。さらに，図 4 のように r_1 を基準とした図形の移動，回転処理も基準となる r_1 の絶対位置を変更したり，回転（角度を変更）させることだけで同様に決定される。これらは，図形の形状（位相）情報を r_1 の絶対長や絶対角と独立させていることによる。

3.　補間点とリスト構造

2 次元で描画された図形を，2. で述べたような一連の補間データで表現するためには，まず，描画する図形によって適用する補間関数が異なるために，図形が開曲線なのか閉曲線なのかの情報が必要となる。次に，2. で述べた位相情報を基に表現された補間点それぞれの情報が必要となる。最後に，これらの情報をもった補間点相互の連結の仕方やその順序を記述した情報が必要となる。

ここで，連結（連続）する多数のデータを記述する手法の一つとしてリスト構造と呼ばれる

データ構造がある[4]。これらのリスト構造にも，

- 記号処理を前提としたもの
- データベースを前提としたもの
- 図形処理を前提としたもの

といった特徴を持ったものがある。本紀要では，これらのリスト構造のうち図形処理を前提とするCOAL[6]を基本とした。

COALのようなフルポインタリング構造のリスト構造では，図形を構成する各線分は図5のように表される。この図において，各ポインターはデータの順序(並び)を規定し，さらに，これらのポインターの始端と終端とが連続している。このようなデータ構造をリング構造と呼ぶ。ここでリング構造内のデータ検索を高速で行うためには，リングの中の各データに対して，双方向ポインターと呼ばれる，次の項目へのポインター（前進ポインター），1つ手前のデータへのポインター（後進ポインター）とリングスタートポイント（親ノード）への直接ポインター（スタートポインター）を持たせる必要がある。以上のようなリスト構造をフルポインタリング構造（full-pointer ring structure）と呼ぶ[4]。

ここで，今回提案するリスト構造は，フルポインタリング構造もつ2層構造のリスト構造とした。これは，通常，1つのデザイン図形は複数の部分図形から構成されるため，まず，部分図形の基準線相互を双方向ポインターで結び部分図形単位の検索，変更に対応し，さらに，その結ばれた各基準線は，図5のようなフルポインタリング構造のノードで部分図形を各々記述することで，部分図形単位の変更にそれぞれ対応することを目的としている。具体例として，図6は図形を構成する基準線分が2本（r_1, r_1'）があり，それらによって表される部分図形が各々1つある場合の例を表している。ここで，図形全体を表すリスト構造として，まず，基準線分r_1, r_1'の情報を記述した親ノードを相互に双方向ポインターで結ぶ。さらに，各親ノード（例えばr_1）で規定される残りの補間点の情報（部分図形）は，それぞれの親ノードを起点とした双方向ポインターと直接ポインターとで結ばれたノードによって表す。なお，前述のようにフルポインタリング構造を採用したことにより，リスト構造の各要素に対する処理は高速で行う事が可能となる。これは，図形を構成する各点（補間点）のデータに対して処理を行う時点で，そのたびに処理すべきデータまでポインターを1つ1つたどって検索する必要がないことによる。

3.1 リスト構造の詳細

今回提案した図6のリスト構造を構成する親ノード，ノードの詳細な内容を図7のように規定した。

一般に，複雑なデザイン図形は，複数の部分

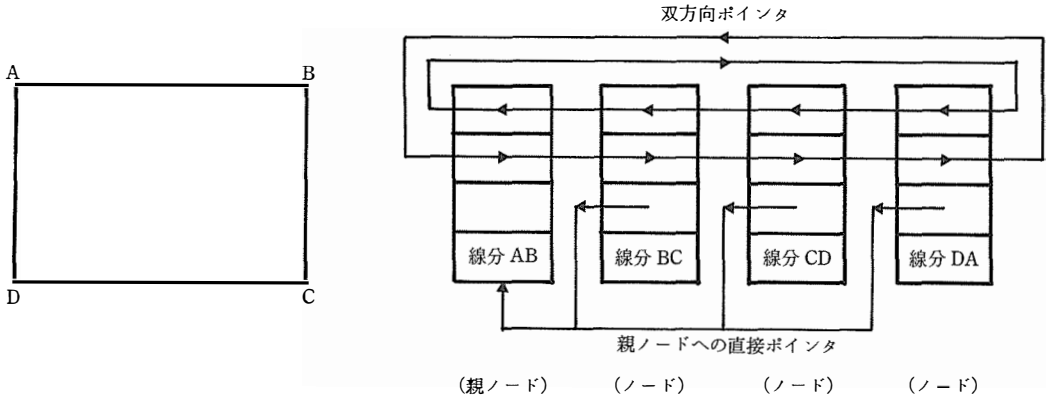

図5　フルポインタリング構造で表現された図形情報

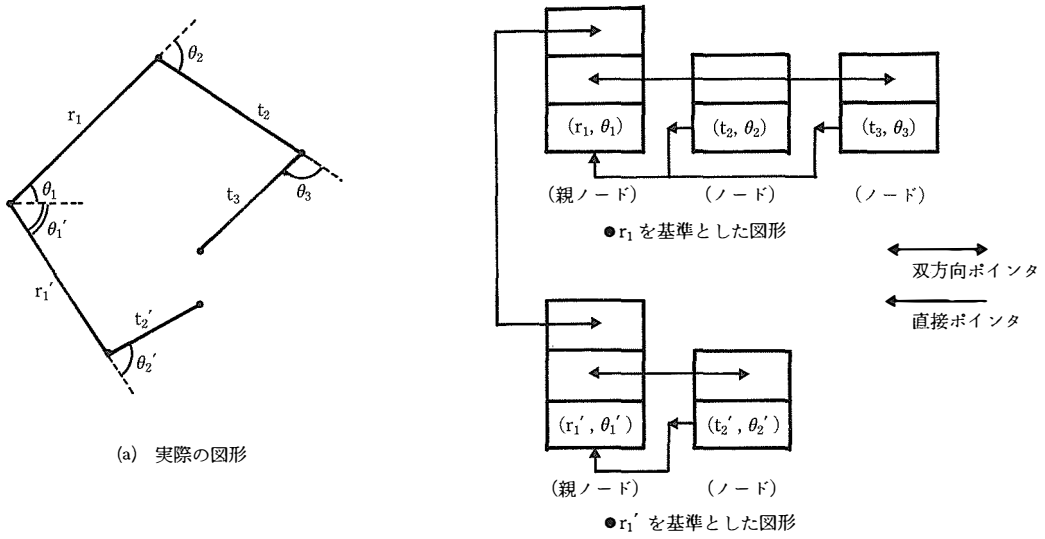

図 6　位相情報をもとにした図形表現

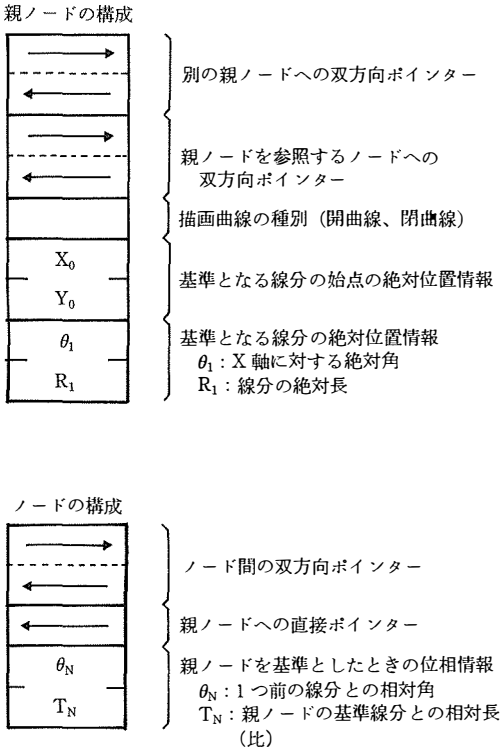

図 7　親ノードとノードの詳細

図形から構成されていることが一般的でるため，複数の親ノードを持つことになる。そこで，図 6 のように部分図形の基準である親ノードは他の親ノードとの双方向ポインターで相互に結ばれて全体の図形を構成する。また，それぞれの親ノードは複数のノードから参照されるため，それらのノードへの双方向ポイ ンタを持つ，次に，複数のノードで表す図形が開曲線か閉曲線かによって適用する補間関数を変更するために，曲線の情報を各親ノードごとに記述する。加えて，親ノードだけは，表現する線分の 2 次元空間の絶対位置を知る必要があるために，基準となる線分（例えば図 6 の r_1）の始点の X, Y 座標の値を各々記述する。最後に，線分の始点に基ずいた線分の絶対角度と絶対長を記述する。但し，線分の角度に関する情報は，X 軸を 0 度として反時計回りをプラスの角度として表現する。

他方，親ノードを参照する各ノードは，まず，同じ親ノードを参照しているノード間の双方向ポイ ンターと，親ノードへの直接ポインタ ーからなる。特に，親ノードへの直接ポインタ ーは，各ノードから直接親ノードを検索するこ

とを可能とするために後述するリスト構造で表現されたデータの処理速度を向上させる。最後に，基準となる親ノードの線分の絶対長との相対的な大きさの比率と1つ前のノードによって表された線分との相対的な傾きとを記述する。

4. 図形処理とリスト構造

2次元図形の処理には，種々のものがあるが基本的な図形処理の内容を列挙すると以下のようなものが挙げられる。

- 図形の移動
- 図形の拡大，縮小
- 図形の追加
- 図形の削除
- 図形のコピー（複写）
- 図形の分割

以下に，それぞれの図形処理時のリスト構造とその要素に対する処理手順と内容についての具体例を挙げた後，一般的な処理手順について述べる。

ここで，図形の移動は図4のように，その図形の基準となっている親ノードの基準線分の始点情報または，基準線分のX軸に対する絶対角を変更することで実現できる。同様に，図形の拡大，縮小についても図3のように基準となっている親ノードの基準線分の絶対長を変更することで実現できる。

4.1 図形の追加

既存の図形に，新たな図形を追加する場合には，図8のような処理が必要となる。

まず，第1に，追加する線分の位置を決定する（図8の場合は，点Cの場所に線分EFGを追加する）。第2に，追加線分の要素で絶対値表現であるr_4とr_1との相対比率表現t_4'に変換し，さらにX軸に対する傾きである角θ_4も線分t_2（r_2）からの相対角に変換する。ここでt_4'と相対角θ_4'は，

$$t_4' = r_4 / r_1 \quad (1)$$

$$\theta_4' = \theta_4 - (\theta_1 + \theta_2) \quad (2)$$

同様に，線分r_4に対する相対比率で表現され

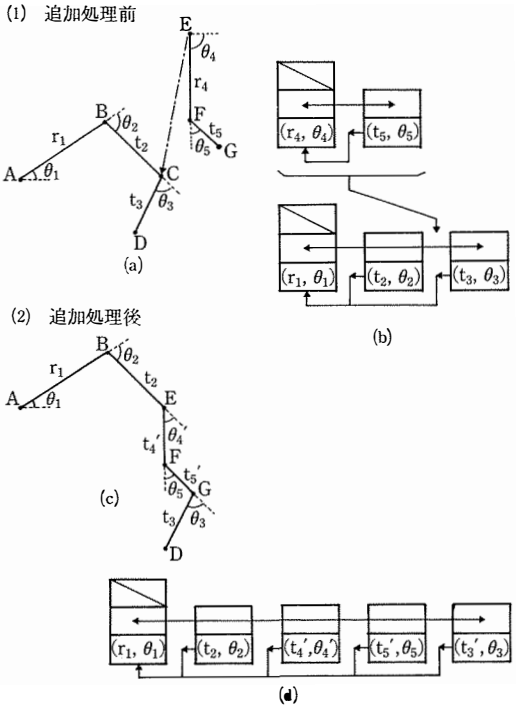

図8 図形の追加処理

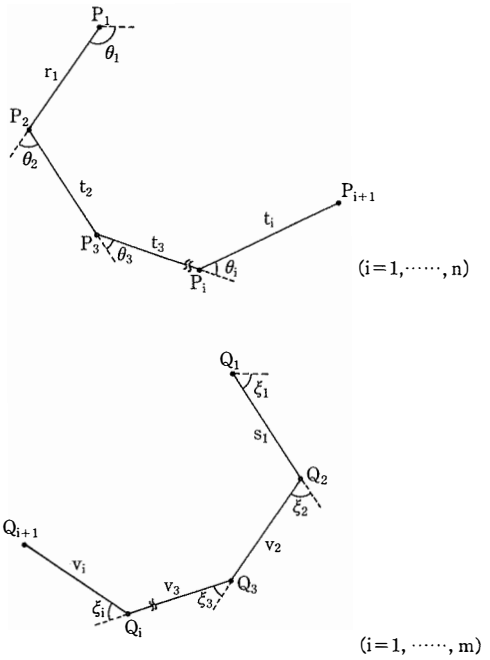

図8.1 一般化した線分と追加線分

ていた線分比率 t_5 も r_1 との比率 t_5' に変換する。ここで t_5' は，

$$t_5' = (t_5 \times r_4)/r_1 \quad (3)$$

最後に，図 8 の(d)のように 1 つの図形のリスト構造として表現する。

一般に，図 8.1 のように，追加される図形と追加する図形とを以下のように定める。

図形を追加される図形，

線分 P_i　$(i=1, \cdots, n+1)$

基準線　(r_1, θ_1)

その他の線分　(t_i, θ_i)　$(i=2, \cdots, n)$

追加する図形も同様に，

線分 Q_i　$(i=1, \cdots, m+1)$

基準線　(s_1, ξ_1)

その他の線分　(v_i, ξ_i)　$(i=2, \cdots, m)$

ここで，図形を追加処理する頂点を P_j とすると，図形追加後は以下のようになる

線分 P_i　$(i=1, \cdots, j-1, j+1, \cdots, n+m+1)$

基準線　(r_1, θ_1)

その他の線分は，(r_1, θ_1)，(t_2, θ_2)，$\cdots$，(t_{j-1}, θ_{j-1})，(t_j', θ_j')，(t_{j+1}', θ_{j+1})，$\cdots$，(t_{j+m}', θ_{j+1})，$(t_{j+m+1}, \theta_{j+m+1})$，$\cdots$，$t_{m+n}, \theta_{m+n})$

ここで，

基準線 s_1 は，

$$t_j' = s_1/r_1 \quad (4)$$

基準角 ξ_1 は，

$$\theta_j' = \xi_1 - (\Sigma_{i=1,j}\theta_i) \quad (5)$$

また，

$$t_{j+i}' = (v_i \times s_1)/r_1 \quad (i=2, \cdots, m) \quad (6)$$

$$t_i \rightarrow t_{m+i} \quad (i=j+1, \cdots, n) \quad (7)$$

$$\theta_i \rightarrow \theta_{m+i} \quad (i=j+1, \cdots, n) \quad (8)$$

4.2　図形の削除

既存の図形から，図形の一部を削除する場合には，図 9 のような処理が必要となる。まず第 1 に，削除する線分の位置を決定する（図 9 では，線分 BCD を削除する）。第 2 に，削除されて残る線分要素で線分 AB に対する相対角 θ_4 を修正する。

ここで θ_4' は，

$$\theta_4' = \theta_2 + \theta_3 + \theta_4 \quad (9)$$

最後に，図 9 の(d)のようにリスト構造を再構成する。

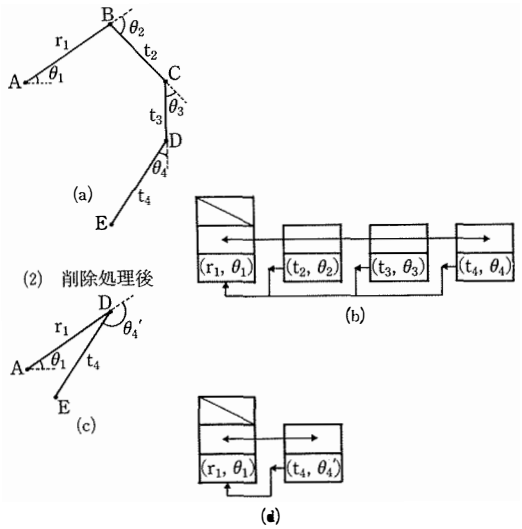

図 9　図形の削除処理

一般に，図形を削除する線分を以下のように定める。

線分 P_i　$(i=1, \cdots, n+1)$

基準線　(r_1, θ_1)

その他の線分　(t_i, θ_i)　$(i=2, \cdots, n)$

ここで，削除線分を p_{j+k}

（但し，$1<j<n, k=1, \cdots, h : h<n$）とする。

すると，図形を削除した後の図形は，

線分 P_i　$(i=1, \cdots, n+1-h)$

基準線　(r_1, θ_1)

その他の線分は，(r_1, θ_1)，(t_2, θ_2)，$\cdots$，(t_{j-1}, θ_{j-1})，(t_{j+h}, θ_{j+h}')，$(t_{j+h+1}, \theta_{j+h+1})$，$\cdots$，$(t_{n-h}, \theta_{n-h})$

ここで，$\theta_{j+h}' = \Sigma_{k=1,h}\theta_{j+k}$ (10)

4.3　図形のコピー

既存の図形のコピーを作成する場合，図10のような処理が必要となる。ここで，線分 ABCD の絶対位置は，線分 AB のみで決定される。そこで，図形のコピー処理は，新たな親ノード（線分 A′B′）を新たに決定し，既存の親ノード（線分 AB）を参照しているノードの情報を，すべて新たな親ノードを参照するようにポインター構造とその要素を複写して新たな

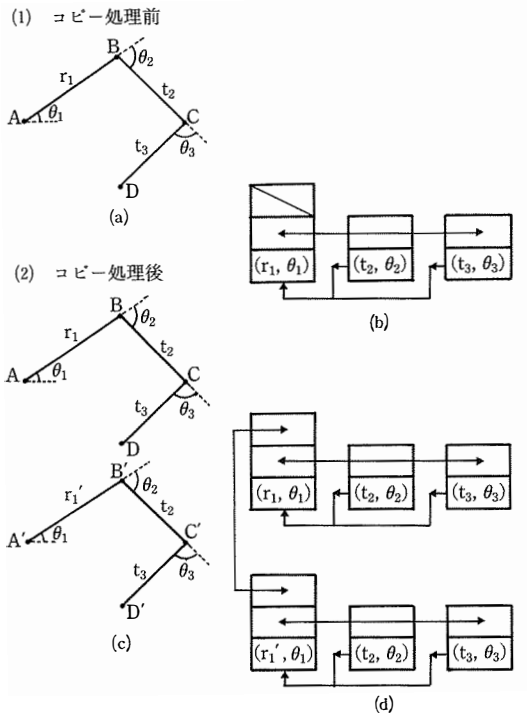

図10 図形のコピー処理

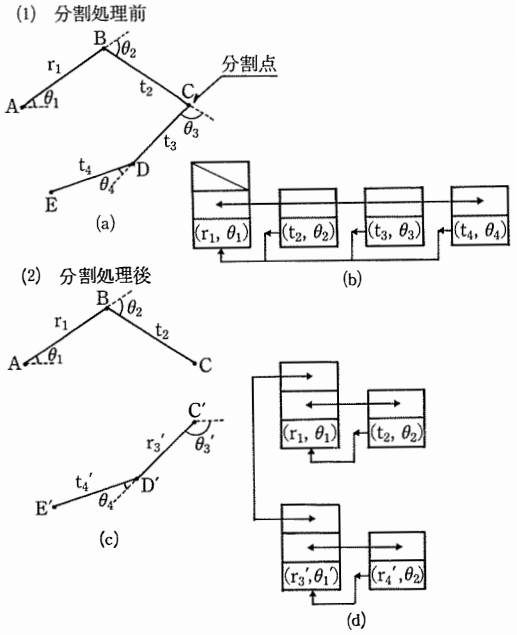

図11 図形の分割処理

リスト構造を作成する。

一般に，コピーする線分を以下のように定める。

線分 P_i （i=1, …, n+1）

基準線（r_1, θ_1）

その他の線分（t_i, θ_i）（i=2, …, n）

ここで，線分 P_i を基準線（r_1', θ_1）を基にして作成した場合

コピー後の図形は，

線分 P_i' （i=1, …, n+1）

基準線（r_1', θ_1）

但し，r_1' の変更内容は，r_1 の始点情報のみとする。

その他の線分は，（t_i, θ_i）（i=2, …, n）

4.4 図形の役割

既存の図形を分割する場合，図11のような処理が必要となる。まず第1に，分割する線分の分割点（図11では点C）を定める。第2に，分割した結果，新たに基準となる親ノード（線分C′, D′）の始点X, Y座標と絶対長 r_3' と基準角 θ_3' を求める。

ここで，r_3', θ_3' は，

$$r_3' = t_3 \times r_1 \quad (11)$$

$$\theta_3' = \theta_1 + \theta_2 + \theta_3 \quad (12)$$

第3に，新たな親ノードの基準長 r_3' に基づく線分の相対比率 t_4' を求める。

ここで，t_4' は，

$$t_4' = (r_1 \times t_4) / r_3' \quad (13)$$

最後に，図11の(d)のように新ためて2つのリスト構造を再構成する。

一般に，分割する線分を以下のように定める。

線分 P_i （i=1, …, n+1）

基準線（r_1, θ_1）

その他の線分は，（t_i, θ_i）（i=2, …, n）

ここで，線分 P_i の分割点を頂点 P_j

（但し，1<j<n）として，線分 P_i と R_i に分割した場合は，

線分1：P_i （i=1, …, j）

基準線（r_1, θ_1）

その他の線分は，（t_i, θ_i）（i=2, …, j−1）

線分2：R_i （i=j, …, n+1）

基準線（r_j, θ_j'）
その他の線分は，（t_1', θ_i）（$i=j+1, \cdots, n$）
ここで，

$$r_j = t_j \times r_1 \quad (14)$$

$$\theta_j' = \Sigma_{k=1,j}\theta_k \quad (15)$$

$$t_i' = (t_i \times r_1)/r_j \quad (i=j+1, \cdots, n) \quad (16)$$

5. 結　　論

生体に基づいたデザイン活動を効率良く支援するために，滑らかな補間関数の適用を前提とした補間点間相互のデータ構造を新に提案した。

次に，図形処理（データ処理）に伴うリスト構造とその要素に対する具体的な処理内容，手順について述べた後に，一般的処理手順について検討した。

ただし，本紀要で提案した図形を構成する補間点間のデータ構造（フルポインタリング構造のリスト構造）は，親ノード間と各補間点のデータ間とで双方向ポインターを用い，さらに，親ノードと各ノード間で直接ポインターを用いた結果，データ処理は高速で行えるが，その一方で図形そのものを記述するために必要となるデータ量に対して，リスト構造全体を記述ために必要となる記憶容量が大きくなる。従って，今後は，図形データの処理速度を下げずに，より少ないポインター等で同等のデータを記述する検討が必要となる。

また，同時に，デザイン活動をより効率化する目的で，図形を構成する複数の部分図形を部品化し，ライブラリ化してその後のデザイン活動に有効利用するための最適な手段の検討も必要である。

参考文献，資料

1）梶谷哲也，若林絵里子，楊　国林：近傍系を用いた柄合わせ，文化女子大学研究紀要21，p. 387–392，（1990. 1）
2）梶谷哲也：滑らかな補間関数の検討，文化女子大学研究紀要22，p. 213–220，（1991．1）
3）文化女子大学被服構成学研究室：被服構成学理論編，文化出版局，（1990）
4）浦　昭二：データ構造，共立出版株式会社，（1988）
5）山口富士夫：形状処理工学Ⅲ，日刊工業新聞社，（1989）
6）W. R. Sutherland: "The COAL language and data structure", Appendix C, Lincoln Laboratory Technical Report 405,（1965）