高次元アプローチによる一般無向グラフの対話的視覚化法

細部博史†

グラフ配置は，オブジェクト間の関係を表現するための情報視覚化技術である．一般無向グラフなどの複雑なグラフは，静的に配置することが困難であるため，対話的グラフ配置がしばしば重要となる．本論文では，一般無向グラフの対話的配置の新しいアプローチを提案する．本アプローチの基本アイデアは，高次元空間における静的なグラフ配置を用いて，ユーザの操作に応じて2次元グラフ配置を動的に決定するというものである．本アプローチに基づいて構築する手法は，以下の2つの特徴を持つ．(1) ユーザの操作に応じて，2次元グラフ配置をきわめて高速に計算する．(2) ユーザによるノードのドラッグ操作に合わせて，それに関連の深いノードを中心に移動を行う．本手法では，高次元グラフ配置を求めるために，固有ベクトル計算に基づく多次元尺度法を用い，2次元グラフ配置を得るために，制約解消によって決定される適切な2次元平面への射影を行うという方法を採用する．

A High-dimensional Approach to the Interactive Visualization of General Undirected Graphs

Hiroshi Hosobe†

Graph layout is an information visualization technology for illustrating relations between objects. Interactive graph layout is often important since it is difficult to statically lay out complex graphs such as general undirected graphs. In this paper, we propose a novel approach to the interactive layout of general undirected graphs. The basic idea behind our approach is to use static graph layouts in high-dimensional spaces to dynamically find two-dimensional layouts according to user interaction. The resulting method that we present exhibits the following two characteristics: (1) it efficiently updates two-dimensional graph layouts during user interaction; (2) it follows users' node dragging operations by actively moving other closely related nodes. Our method adopts eigenvector-based multidimensional scaling to compute high-dimensional graph layouts, and performs constraint satisfaction to determine appropriate two-dimensional planes onto which the high-dimensional layouts will be projected.

1. はじめに

対話的なアプリケーションやシステムにおいて，オブジェクト間の関係を表現するために，情報視覚化の技術が必要とされることが多い．**グラフ**はこのような構造や関係のための形式的な表現手段である．グラフはオブジェクトをノードとして，オブジェクト間の関係をエッジとして表す．グラフで表現された情報を視覚化するための技術としては，ノードとエッジの適切な位置を自動的に計算する**グラフ配置**あるいはグラフ描画と呼ばれる手法がさかんに研究されている[3),16),27)]．グラフ配置の手法は，グラフの構造に基づくグラフのクラスに応じて設計される．具体的なグラフのクラスとしては，木，有向グラフ，平面グラフ，**一般無向グラフ**などがある．

一般無向グラフは，エッジの方向が定められていないグラフであり，ネットワーク的な構造を持つ様々な情報を表現するために用いられている．一般無向グラフを扱うグラフ配置方式としては，グラフを仮想的な力学系に対応させてシミュレーションを実行することで安定配置を求める**力指向アプローチ**[3),16),27)]がよく知られており，数多くの研究が行われている．このような力指向アプローチに代表される一般無向グラフ配置法は，これまでに一定の成果を収めており，ノード数が数十程度の小さいグラフや，ノード数が100万個規模でもメッシュ状の構造を持つようなグラフに関しては，美しい配置を得ることが可能である．

その一方で，ある程度の大きさと複雑さを持つ一般無向グラフの配置法に関する研究は，まだ十分であるとはいえない．このようなグラフは，その高い一般性のために，静的な1つのグラフ配置によって，その構

† 国立情報学研究所
National Institute of Informatics

造を視覚化することが困難であると考えられる．しかし，このような一般無向グラフの配置法は必要である．特に近年，現実の文書や Web などの大規模データを分析処理することで，単語や文書などの間の関連や構造を抽出し，グラフとして視覚化する試みが数多く行われている☆[8),19),20),28)]．この種の技術の進展にともない，大きく複雑な一般無向グラフの配置法の必要性は今後ますます高まっていくと予想される．

ある程度の大きさと複雑さを持つ一般無向グラフの配置という課題に対する有力な手段として，ユーザの操作に基づく**対話的グラフ配置**[16)] がある．特に力指向アプローチは，仮想的力学シミュレーションにおいてユーザの操作を介入させることで，対話的グラフ配置に拡張可能である．このため，実際にこのような目的で利用されている[8),28)]．

本研究では，一般無向グラフの対話的配置という課題に対して，力指向とは異なる，新しいアプローチに基づく手法を提案する．本アプローチの基本アイデアは，**高次元空間**におけるグラフ配置をあらかじめ静的に求めておき，その高次元配置と，ユーザによるノードのドラッグ操作に応じて，2 次元グラフ配置を動的に決定するというものである．ここで高次元空間におけるグラフ配置とは，たとえば 1,000 個のノードからなるグラフが与えられたときに，各ノードに対して数百次元の座標を割り当てることで定められる配置であり，本手法の内部的なデータに相当するものである．一方，2 次元グラフ配置とは，コンピュータ画面上に表示するために用いられる平面的な配置である．本アプローチは，このような内部的な高次元配置を保存したままで，2 次元配置のみを変更することで，ユーザによる対話的配置を実現する．

本提案手法は以下の 2 つの特徴を持つ．

(1) 2 次元グラフ配置の計算がきわめて高速であり，ノード数が 1,000 以上のグラフに対しても数十ミリ秒で処理を完了することができる．このため，ユーザの対話的操作は実時間で実行可能である．

(2) ユーザによるノードのドラッグ操作に合わせて，それに関連の深いノードを中心に移動を行う．この性質を利用することで，たとえば少数個のノードをドラッグするだけでグラフの特定部分を強調することなども可能である．

☆ この種のデータではエッジが方向を持つことがあり，その場合は一般有向グラフが得られる．しかし，実際の配置の際に一般無向グラフとして処理し，エッジの方向は画面上で矢印として表現することも多い．

本手法では，内部的な高次元グラフ配置を求めるために，**固有ベクトル**計算に基づく**多次元尺度法**[12)] を用いてグラフ配置を求める方法[18)] を高次元に拡張して利用する．また，このようにして得られる高次元グラフ配置から 2 次元グラフ配置を求めるために，制約解消によって決定される適切な 2 次元平面への**射影**を行うという方法を採用する．

本論文は以下の構成からなる．まず 2 章で，本研究に関連するグラフ配置法について紹介する．次に 3 章で，本研究が基礎とするグラフ配置の従来手法を紹介する．4 章では，本研究で提案する対話的グラフ配置法について述べる．5 章で，本研究の提案手法に基づいて実装した試作システムについて概説し，6 章で，その評価実験の結果を与える．7 章では，本研究の提案手法について議論を行う．最後に 8 章で，本研究の結論と今後の課題を述べる．

2. 関 連 研 究

一般無向グラフの配置を求める代表的な方式として，力指向アプローチ[3),16),27)] がある．これは，グラフを仮想的な力学系に対応させてシミュレーションを実行することで，力学的に安定な配置を求めるアプローチである．Eades は，エッジにバネを対応させ，その引力と斥力によってノードの安定配置を求める，バネ埋込みという方法を提案した[5)]．Kamada らは，全ノード間のグラフ理論的距離を求め，バネを用いてノード間の距離をグラフ理論的距離に近付ける方法を提案した[15)]．文献 9) は，多次元（たとえば 4 次元）空間におけるグラフ配置を力指向アプローチによっていったん求め，それを 2 または 3 次元空間へ射影する手法を提案している．力指向アプローチを対話的な情報視覚化に適用した例として，論文などの文献間の関係を視覚化する DocSpace[28)] や，Web におけるコミュニティの関連構造を視覚化する Web Community Browser[8)] などがある．

グラフ配置を含む情報視覚化のために，多次元尺度法（MDS）[12),25)] がしばしば用いられる．MDS とは，オブジェクト間の親近性がデータとして与えられたときに，多次元空間におけるオブジェクトの配置を求める統計学的方法である．Kruskal らは，グラフ配置のために MDS を利用することを提案し，その例として固有ベクトル計算に基づく MDS を用いる手法を与えている[18)]（この手法については 3 章で詳しく述べる）．また文献 2) では，MDS と力指向アプローチを関連付ける試みが与えられている．MDS を情報視覚化に応用した例としては，3 次元空間におけるグラフ

配置によって大規模知識ベースの視覚化を可能にしたSemNet [7] や，2 次元と 3 次元を併用することで多次元データのブラウズ機能を向上した映画データベースシステム ZASH [21] などがある．

グラフ配置のために，固有ベクトルを利用する方法が近年注目されている．固有ベクトルとは，おおよその定義でいえば，正方行列 A に対して $A\boldsymbol{x} = \lambda\boldsymbol{x}$ を満たし，固有値と呼ばれるスカラー λ に対応する列ベクトル $\boldsymbol{x}$ のことである．文献 22) では，Laplace 行列の固有ベクトルを用いたグラフ配置法によって，サッカボール形のグラフ配置を得る例が示されている．これと同等の定式化に基づく ACE アルゴリズム[17] は，代数的マルチグリッド法によって固有ベクトルの計算を高速化することで，ノード数が数百万のグラフ配置を 1 分以内に計算できる．また文献 11) は，50 程度の比較的高次元のグラフ配置を，次元と同数のピボットと呼ばれるノードを基準として求めたうえで，固有ベクトル計算による主成分分析に基づいて 2 次元平面への射影を決定することで，ノード数が 10 万程度のグラフ配置を数秒で計算できる手法を提案している．ただし，これらの手法はグラフの概形を表現するためには有効であるが，局所的な形状の表現には不向きであり，実際の適用例も，メッシュ状の構造を持つグラフに限られている．

複雑な情報を効果的に視覚化する技術として，フィッシュアイがある．これは対象となる情報視覚化における興味のある部分を拡大強調しながら，それ以外の部分を簡略化することで，全体構造も同時に表示できるようにする技術であり，グラフに対しても適用されている[7),13),26),27)]．

3. 多次元グラフ配置法

本章では，本研究において使用する多次元グラフ配置のための従来手法を紹介する．

3.1 Torgerson の方法

最初に，多次元グラフ配置法の基礎として，Torgerson の方法[12] を述べる．この方法は，統計学の分野において計量的多次元尺度法としても知られた手法であり，すべてのオブジェクトの組合せに対して距離が与えられているときに，それらの距離を満たすようなオブジェクトの配置を求めるものである．

n 個のオブジェクトに対して，任意のオブジェクト i, j の間の距離 d_{ij} $(= d_{ji})$ が与えられ，距離の公理が満たされていると仮定する．まず，以下のように a_{ij} を定める．

$$a_{ij} = \frac{1}{2}\left(\frac{1}{n}\sum_k d_{ik}^2 + \frac{1}{n}\sum_k d_{kj}^2 - \frac{1}{n^2}\sum_k\sum_l d_{kl}^2 - d_{ij}^2\right)$$

次に n 次正方行列 $A = (a_{ij})$ を考えると，$a_{ij} = a_{ji}$ により，これは実対称行列となる．このとき，A は，対角行列 Λ と直交行列 X を用いて

$$X^T A X = \Lambda$$

のようにでき，すなわち対角化可能である．ここで，A の固有値を λ_k とし，λ_k に対応する固有ベクトルを $\boldsymbol{x}_k$ とすると，Λ と X を以下のように定めることができる．

$$\Lambda = \begin{bmatrix} \lambda_1 & & & 0 \\ & \lambda_2 & & \\ & & \ddots & \\ 0 & & & \lambda_n \end{bmatrix}$$

$$X = [\boldsymbol{x}_1\ \boldsymbol{x}_2\ \cdots\ \boldsymbol{x}_n]$$

さらに，対角行列

$$\Lambda^{1/2} = \begin{bmatrix} \sqrt{\lambda_1} & & & 0 \\ & \sqrt{\lambda_2} & & \\ & & \ddots & \\ 0 & & & \sqrt{\lambda_n} \end{bmatrix}$$

を用いて

$$P = X\Lambda^{1/2}$$

とおく．ここで，各 d_{ij} に対して理想的な値が与えられていて，各固有値 λ_k が非負となっていれば，P の第 i 行 $(p_{i1}, p_{i2}, \ldots, p_{in})$ は，オブジェクト i の n 次元実 Euclid 空間における座標と見なせる．ただし，現実のデータに対しては，負の値をとるような固有値が現れることが多い．

以上の方法で得られるオブジェクトの配置によって，すべての固有値が非負である場合に，与えられたオブジェクト間の距離が満たされることは，以下のように説明できる．まず，各オブジェクト i の位置を $\boldsymbol{p}_i = (p_{i1}, p_{i2}, \ldots, p_{in})$ とし（$\boldsymbol{p}_i$ は上述のとおり P の第 i 行に対応する），すべてのオブジェクトの重心が原点に一致する（すなわち，$(1/n)\sum_k \boldsymbol{p}_k = \boldsymbol{0}$ である）とする．このとき，各 a_{ij} は，$\boldsymbol{p}_i$ と $\boldsymbol{p}_j$ の内積に等しい．これは，任意の m に対して

$$\boldsymbol{p}_i \cdot \boldsymbol{p}_j = \left\{ (\boldsymbol{p}_i - \boldsymbol{p}_m) - \frac{1}{n} \sum_k (\boldsymbol{p}_k - \boldsymbol{p}_m) \right\} \cdot \left\{ (\boldsymbol{p}_j - \boldsymbol{p}_m) - \frac{1}{n} \sum_l (\boldsymbol{p}_l - \boldsymbol{p}_m) \right\}$$

であり，さらに，任意の i'，j' に対して，余弦定理により

$$(\boldsymbol{p}_{i'} - \boldsymbol{p}_m) \cdot (\boldsymbol{p}_{j'} - \boldsymbol{p}_m) = \frac{1}{2} \left(d^2_{i'm} + d^2_{j'm} - d^2_{i'j'} \right)$$

であることから示される．これにより，$A = PP^T$ が満たされる．また，$X^T A X = \Lambda$ より，$A = (X\Lambda^{1/2})(X\Lambda^{1/2})^T$ である．したがって，$P = X\Lambda^{1/2}$ が成立する．

通常の応用では，最大の固有値を少数個だけ用いて，それらに対応する座標成分のみを利用する．たとえば，λ_1 と λ_2 をそれぞれ 1，2 番目に大きい固有値であるとし，それらのみを用いるとする場合，各オブジェクト i の座標を (p_{i1}, p_{i2}) とするような 2 次元平面上の配置が得られる．

3.2 Torgerson の方法に基づくグラフ配置法

Kruskal らは，Torgerson の方法に基づく，連結な一般無向グラフの配置法（以下，TKS 法と呼ぶ）を提案した[18]．これは以下のように実現されている．

(1) 最初に，グラフの任意のノード間のグラフ理論的距離（最短路の長さ）を求める．
(2) 次に，グラフ理論的距離を用いて Torgerson の方法を実行し，2 次元平面上のノードの配置を計算する．

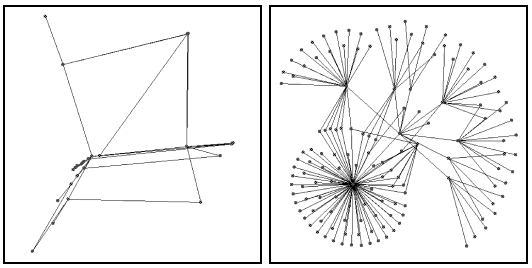
Kruskal らは 2 次元を想定していたが，Torgerson の方法の性質により，この手法は多次元グラフ配置へ容易に拡張可能である．

3.3 Kamada らの方法との関連および比較

前項で述べた TKS 法と，2 章で述べた Kamada らの方法[15]（以下，KK 法と呼ぶ）の間には深い関連がある．なぜなら，両者とも，ノード間の距離がグラフ理論的距離となるようなノードの配置を求めようとしているからである．最大の相異点は，前者が必要な次元数の空間を用いるのに対して，後者が 2 次元平面のみを用いる点である．

通常の TKS 法によって生成された 2 次元グラフ配置は，元のグラフの大まかな構造のみを表現していると考えられる．一方，そのグラフにおける，より局所的な構造は，他の次元で表現されているために，2 次元グラフ配置から読み取ることができないと考えられる．

これに対して KK 法では，2 次元グラフ配置においてグラフの局所的な形状がよく表現される．しかしその一方で，グラフの概形が崩れてしまって，本来は離れているはずのノードどうしが近くに配置されてしまうことがある．その原因として，グラフ理論的距離の大きいノード間のバネを弱くしていることと，実際の計算の際に局所解に陥りやすいことがあげられる．



(a) (b)

図 1 (a) TKS 法と (b) KK 法による AT&T グラフ ug_263（ノード数 141，エッジ数 177）の配置
Fig. 1 Layouts of the AT&T graph ug_263 (with 141 nodes and 177 edges) obtained by the (a) TKS and (b) KK methods.

TKS 法と KK 法の具体的な比較のために，同一のグラフに対して適用して得られた 2 次元配置の例を**図 1** に示す．ここで扱ったグラフは，AT&T グラフ☆ ug_263 と呼ばれる，141 個のノードと 177 本のエッジからなる一般無向グラフである．

4. 提 案 手 法

本章では，高次元のグラフ配置を利用した対話的な 2 次元グラフ配置法を提案する．

4.1 2 次元グラフ配置の計算

本研究で提案する手法は，高次元空間におけるグラフ配置を 2 次元平面に射影することで，2 次元グラフ配置を計算する．処理の対象として連結な一般無向グラフを扱い，エッジはノード間の直線として表すものとする．

高次元空間におけるグラフ配置の計算には，前章で紹介した TKS 法を採用する．ただし本提案手法では，すべての正の固有値に対応する座標成分を用いるものとする．一般に，TKS 法のようにグラフ理論的距離を用いて Torgerson の方法を実行した場合，多くの固有値が正になるため，結果として得られるグラフ配置の次元は高いものとなる．以下では，固有値 λ_1，λ_2，... を大きさの順に並べ，正の固有値の個数を d とし，$d \geq 2$ であると仮定する．このとき，

☆ http://graphdrawing.org/data/index.html

$$\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_d > 0$$

であり，高次元空間における各ノード i の位置は

$$\boldsymbol{p}_i = (p_{i1}, p_{i2}, \ldots, p_{id})$$

である．

このような d 次元グラフ配置を以下のようにして 2 次元平面へ射影する（以下，この 2 次元平面を射影平面と呼ぶ）．射影平面として，2 つの正規直交な d 次元ベクトル $\boldsymbol{e}_1$，$\boldsymbol{e}_2$ によって張られる平面を考える．これらを用いれば，ノード i の 2 次元座標は

$$(\boldsymbol{p}_i \cdot \boldsymbol{e}_1, \boldsymbol{p}_i \cdot \boldsymbol{e}_2)$$

として求められる．

最初の 2 次元グラフ配置を求めるために用いる $\boldsymbol{e}_1$，$\boldsymbol{e}_2$ の初期値は以下のようにする．

$$\begin{aligned} \boldsymbol{e}_1 &= \boldsymbol{f}_1/\|\boldsymbol{f}_1\| \\ \boldsymbol{e}_2 &= \boldsymbol{f}_2/\|\boldsymbol{f}_2\| \end{aligned}$$

ただし，$\boldsymbol{f}_1$，$\boldsymbol{f}_2$ は，以下のように定められる d 次元ベクトルである．

$$\begin{aligned} \boldsymbol{f}_1 &= \left(\lambda_1^\delta, 0, \lambda_3^\delta, 0, \ldots\right) \\ \boldsymbol{f}_2 &= \left(0, \lambda_2^\delta, 0, \lambda_4^\delta, \ldots\right) \end{aligned}$$

ここで，δ は座標成分の寄与度を調節するためのパラメータである．このようにしたとき，$\boldsymbol{e}_1$ と $\boldsymbol{e}_2$ は正規直交になる．

この定義では，一般に，パラメータ δ が大きくなるほど，グラフの初期 2 次元配置の計算における高次元の座標成分の寄与度が低くなっていく．特に $\lambda_1 > \lambda_3$ かつ $\lambda_2 > \lambda_4$ である場合，$\delta \to \infty$ のときの極限は，TKS 法による 2 次元グラフ配置と一致する．なお，δ の適切な値を決定することは一般には困難であるので，デフォルトとして $\delta = 1/2$ を用いるものとする．

4.2 2 次元グラフ配置の更新

次に，2 次元グラフ配置をユーザが対話的に更新できるようにする手法を提案する．本手法は，射影平面を移動することで実現され，高次元空間におけるグラフ配置を変更する必要がないため，きわめて高速に 2 次元グラフ配置を更新できるという特徴がある．以下では，ユーザの入力として，1 つのノードをドラッグする場合を扱うものとする．

本手法の基本アイデアは，新しい射影平面を定めるために，特定の部分空間の中で，元の射影平面を回転移動するというものである．そのような部分空間としては，元の射影平面と，ドラッグされるノードの位置ベクトルによって張られる 3 次元空間を用いる．そして，この回転移動を決定するために，射影平面を張るベクトルが満たすべき制約を構成したうえで，制約解消を行う．

以下に本手法の詳細を述べる．最初に，入力となる定数を定義する．射影平面を張る現在のベクトルを $\boldsymbol{e}_1$，$\boldsymbol{e}_2$ とし，正規直交であるとする．また，ユーザがノード i をドラッグしているとし，射影平面におけるその元の座標を (x_i, y_i)，新しい座標を (x_i', y_i') とする．このとき，定義により $(x_i, y_i) = (\boldsymbol{p}_i \cdot \boldsymbol{e}_1, \boldsymbol{p}_i \cdot \boldsymbol{e}_2)$ が成立する．また，$\|(x_i, y_i)\| < \|\boldsymbol{p}_i\|$ かつ $\|(x_i', y_i')\| < \|\boldsymbol{p}_i\|$ であると仮定する（これらの条件については後に詳しく述べる）．

また，$\boldsymbol{p}_i$ を $\boldsymbol{e}_1$，$\boldsymbol{e}_2$ によって正規直交化して得られるベクトルを $\boldsymbol{e}_3$ とする．これは以下の式により求められる．

$$\boldsymbol{e}_3 = \boldsymbol{f}_3/\|\boldsymbol{f}_3\|$$

ただし，$\boldsymbol{f}_3$ は，以下のように定められる d 次元ベクトルである．

$$\boldsymbol{f}_3 = \boldsymbol{p}_i - x_i\boldsymbol{e}_1 - y_i\boldsymbol{e}_2$$

このとき，$\|\boldsymbol{f}_3\| = \sqrt{\|\boldsymbol{p}_i\| - x_i^2 - y_i^2}$ であり，$\|(x_i, y_i)\| < \|\boldsymbol{p}_i\|$ より，$\|\boldsymbol{f}_3\| > 0$ が成立する．

次に，射影平面を張る新しいベクトルを $\boldsymbol{e}_1'$，$\boldsymbol{e}_2'$ とし，これらは $\boldsymbol{e}_1$，$\boldsymbol{e}_2$，$\boldsymbol{e}_3$ からなる 3 次元空間内のベクトルであるとする．このとき，これらは 6 個の変数 α_1，α_2，α_3，β_1，β_2，β_3 を用いて以下のように表せる．

$$\begin{aligned} \boldsymbol{e}_1' &= \alpha_1\boldsymbol{e}_1 + \alpha_2\boldsymbol{e}_2 + \alpha_3\boldsymbol{e}_3 \\ \boldsymbol{e}_2' &= \beta_1\boldsymbol{e}_1 + \beta_2\boldsymbol{e}_2 + \beta_3\boldsymbol{e}_3 \end{aligned}$$

元の射影平面から新しい射影平面への回転移動の軸をベクトル $\boldsymbol{r}$ とすると，これは 2 個の変数 γ_1，γ_2 を用いて次のように表せる．

$$\boldsymbol{r} = \gamma_1\boldsymbol{e}_1 + \gamma_2\boldsymbol{e}_2$$

以上の定数と変数を用いて，以下の 8 つの制約を構成する．

$$\|\boldsymbol{e}_1'\| = 1 \quad (1)$$

$$\|\boldsymbol{e}_2'\| = 1 \quad (2)$$

$$\boldsymbol{e}_1' \cdot \boldsymbol{e}_2' = 0 \quad (3)$$

$$\|\boldsymbol{r}\| = 1 \quad (4)$$

$$\boldsymbol{r} \cdot \boldsymbol{e}_1' = \boldsymbol{r} \cdot \boldsymbol{e}_1 \quad (5)$$

$$\boldsymbol{r} \cdot \boldsymbol{e}_2' = \boldsymbol{r} \cdot \boldsymbol{e}_2 \quad (6)$$

$$\boldsymbol{p}_i \cdot \boldsymbol{e}_1' = x_i' \quad (7)$$

$$\boldsymbol{p}_i \cdot \boldsymbol{e}_2' = y_i' \quad (8)$$

制約 (1)〜(3) は，$\boldsymbol{e}_1'$，$\boldsymbol{e}_2'$ が正規直交であることを意味する．(4)〜(6) は，$\boldsymbol{r}$ が単位ベクトルであり，$\boldsymbol{e}_1'$，$\boldsymbol{e}_2'$ が，$\boldsymbol{r}$ を軸として $\boldsymbol{e}_1$，$\boldsymbol{e}_2$ をそれぞれ回転したものであることを意味する．(7)，(8) は，$\boldsymbol{p}_i$ を新しい射影平面に射影したときの座標が (x_i', y_i') であることを意味する．**図 2** はこれらのベクトルを 3 次元的に図

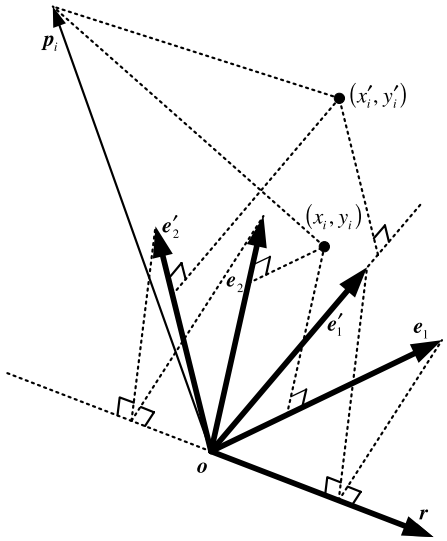


図 2 射影平面の更新
Fig. 2 Updating the projection plane.

示したものである．

これら 8 つの制約は以下のように容易に解くことができる．まず，(1)〜(3) は α_1，α_2，α_3，β_1，β_2，β_3 に関する 2 次等式制約となる．

$$\begin{aligned} \alpha_1^2 + \alpha_2^2 + \alpha_3^2 &= 1 \\ \beta_1^2 + \beta_2^2 + \beta_3^2 &= 1 \\ \alpha_1\beta_1 + \alpha_2\beta_2 + \alpha_3\beta_3 &= 0 \end{aligned}$$

(4)〜(6) は α_1，α_2，β_1，β_2，γ_1，γ_2 に関する制約となり，さらに γ_1，γ_2 を消去することで，以下の 2 次等式制約が得られる．

$$(\alpha_1 - 1)(\beta_2 - 1) - \alpha_2\beta_1 = 0$$

また，(7)，(8) より，α_3，β_3 を以下のように α_1，α_2，β_1，β_2 に関する 1 次式で表現できる．

$$\begin{aligned} \alpha_3 &= (x'_i - x_i\alpha_1 - y_i\alpha_2)/\|\boldsymbol{f}_3\| \\ \beta_3 &= (y'_i - x_i\beta_1 - y_i\beta_2)/\|\boldsymbol{f}_3\| \end{aligned}$$

以上により，実際には α_1，α_2，β_1，β_2 を変数として，4 つの 2 次等式制約の系を解けばよいことになる．この制約系は，Newton 法のような基本的な連立非線形方程式の数値解法によって高速に解消できる．

以上において，$\|(x_i, y_i)\| < \|\boldsymbol{p}_i\|$ と $\|(x'_i, y'_i)\| < \|\boldsymbol{p}_i\|$ という 2 つの条件を仮定した．ここで，$\|(x_i, y_i)\| < \|\boldsymbol{p}_i\|$ は，$\boldsymbol{p}_i$ が $\boldsymbol{e}_1$，$\boldsymbol{e}_2$ に対して線形独立であることを保証するための条件であり，射影平面が 3 次元的に回転できるようにするために必要である．この条件が成立しないのは $\|(x_i, y_i)\| = \|\boldsymbol{p}_i\|$ である場合のみであり，これは稀であるので，このようなノードのドラッグを禁止すれば，この条件を保証できる．一方，$\|(x'_i, y'_i)\| < \|\boldsymbol{p}_i\|$ は，新しい射影平面が実際に存在し，同じノードに対するドラッグを続けてできるようにするための条件である．この条件を保証するには，ある小さい正の定数 τ（たとえば $\tau = 10^{-3}$）を導入し，$\|(x'_i, y'_i)\| > (1 - \tau)\|\boldsymbol{p}_i\|$ である場合に，$\{(1 - \tau)\|\boldsymbol{p}_i\|/\|(x'_i, y'_i)\|\}(x'_i, y'_i)$ を改めて (x'_i, y'_i) とすればよい．

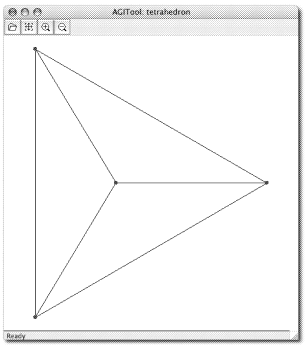


図 3 試作システム AGI
Fig. 3 The prototype system AGI.

5. 実　　装

前章で提案した対話的グラフ配置法に基づいて，試作システム AGI☆を実装した．プログラムの記述は C++ で行い，固有ベクトル計算を含む線形計算のために，数値計算パッケージ ATLAS 3.4.1 および LAPACK 3.0 を使用した．浮動小数点数の計算には単精度を用い，ATLAS の内部で SIMD 演算命令を利用している．また，グラフ理論的距離の計算には Floyd のアルゴリズム[14] を使用した．グラフィカルユーザインタフェースの実現には wxWidgets を用いた．

図 3 は，AGI システムの実行画面である．AGI では，グラフを GraphML 形式で記述したファイルを読み込むことで，その 2 次元配置を表示することができる．またユーザはノードをドラッグすることで，グラフ配置を対話的に更新することが可能である．

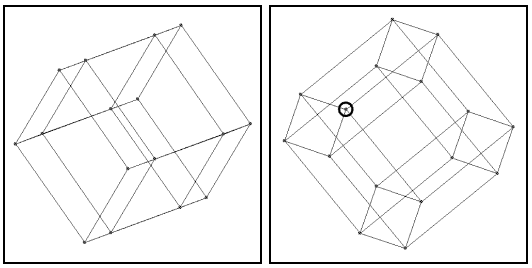
6. 実　　験

本章では，本研究の提案手法の評価のために行った実験の結果について述べる．

6.1 実験環境

本研究では，前章で述べた試作システム AGI を用いて，実験を行った．いずれの実験においても，4 章で述べたグラフの初期 2 次元配置を求めるためのパラメータとして，$\delta = 1/2$ を用いている．また試作シ

☆ AGI は，“Active Graph Interface” の略である．



(a) (b)

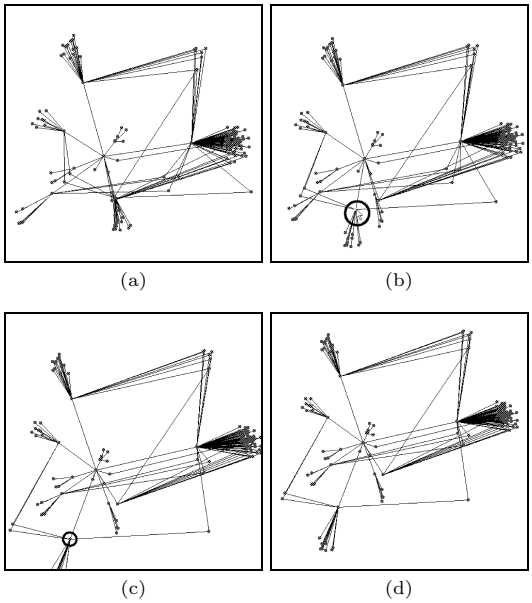
図 4 4 次元ハイパーキューブ型構造のグラフの対話的配置
Fig. 4 Interactive layout of the graph with the four-dimensional hypercubic structure.

ステムは，–O3 オプションを与えた GCC 3.3 でプログラムをコンパイルし，Mac OS X 10.3 で動作する 2 GHz の PowerPC G5☆上で実行した．

6.2 対話的配置に関する予備実験

最初に，対話的グラフ配置に関する予備的な実験として，本提案手法を 3 つの例に対して適用することで得られた結果を与える．第 1 の例は，4 次元ハイパーキューブ型の構造を持つグラフの配置である．このグラフにおけるノードの個数は 16，エッジの本数は 32 である．本提案手法によって得られた，このグラフの初期配置を**図 4** (a) に示す．次にノードを 1 個ドラッグすることで得られた配置を図 4 (b) に示す（丸印の付けられたノードがドラッグされたものであり，これは図 4 (a) で上から 2 番目にあったノードを左下に向けて移動したものである）．図 4 (b) のように配置を更新することで，この構造が，4 つの四角形の対応する頂点どうしを接続したもの，あるいは 2 つの直方体の対応する頂点どうしを接続したものに相当することが分かりやすくなる．なお，このグラフの初期配置と，ドラッグによる配置更新の計算時間は，ともに 10 ミリ秒未満であった．

次の例は，3 章で用いた AT&T グラフ ug_263 の配置である．このグラフの初期配置を**図 5** (a) に示す．このグラフの内部的な次元数（4 章における d）は 125 である．図 1 (a) で示した TKS 法による配置との比較からも分かるように，TKS 法でノードの高次元座標成分として隠されていたグラフの局所的な形状が，本提案手法の 2 次元配置では表現されている．次にグラフの対話的配置の例として，1 つのノードをドラッグした場合を示す．図 5 (b) はユーザがノードをドラッグしている途中を示したものであり，図 5 (c) はドラッグを終えた直後の状態である（ドラッグされたノードを丸印で示す）．さらに画面上での縮尺を調整したものを図 5 (d) に示す．この例で示されるように，本提案手法では，ユーザによるノードのドラッグに合わせて，それに関連の深いノードの移動を中心にグラフ配置の更新が行われる．以上の処理に要した時間を**表 1** に示す．



(a) (b) (c) (d)

図 5 AT&T グラフ ug_263（ノード数 141，エッジ数 177）の対話的配置
Fig. 5 Interactive layout of the AT&T graph ug_263 (with 141 nodes and 177 edges).

最後の例は，1,104 個のノードと 3,231 本のエッジからなる AT&T グラフ ug_380 の配置である．**図 6** (a) は本提案手法によるこのグラフの初期配置であり，その内部的な次元数は 697 である．この配置の右下側にあるノードを 1 つドラッグし，画面上での縮尺を調整して得られた配置を図 6 (b) に示し，さらに 2 個のノードを 1 つずつドラッグして得られた配置を図 6 (c) および (d) に示す（ドラッグされたノードは丸印で示されている）．このように，本手法は，グラフの一部を拡大強調して見るような目的でも利用することが可能である．最後に，これらの処理に要した時間を表 1 に与える．この結果が示すように，本手法では，初期配置に時間がかかるようなグラフを扱う場合でも，配置の更新処理はきわめて高速に実行できる．

6.3 現実のデータへの適用

次に，現実のデータに基づくグラフに対して，本提案手法を適用した実験の結果を与える．具体的なグラ

☆ 実行に用いた計算機は 2 基のプロセッサを搭載するものであるが，本実験ではオペレーティングシステムの設定変更により 1 基のプロセッサのみで動作させた．

表 1 本研究で提案した対話的グラフ配置法の実行時間
Table 1 Running times of our interactive graph layout method.

グラフ	ノード数	エッジ数	内部的な次元数	初期配置の全計算時間（そのうちグラフ理論的距離と固有ベクトルの計算時間）	配置更新 1 回分の計算時間
ug_263	141	177	125	30 ミリ秒（10 ミリ秒，20 ミリ秒）	10 ミリ秒未満
ug_380	1,104	3,231	697	12.6 秒（5.7 秒，6.8 秒）	10 ミリ秒
修正 erdos1graph	463	1,547	259	830 ミリ秒（180 ミリ秒，640 ミリ秒）	10 ミリ秒未満
torus64x16	1,024	2,048	116	29.5 秒（23.6 秒，5.6 秒）	10 ミリ秒
folded_grid40	1,600	3,122	154	43.9 秒（28.0 秒，15.6 秒）	10 ミリ秒

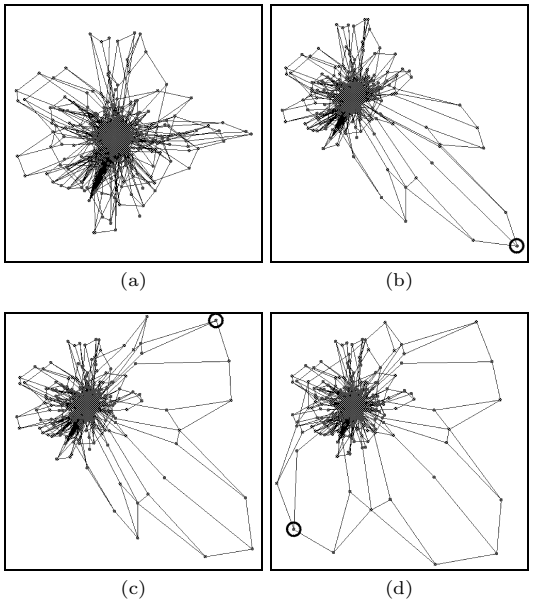


図 6 AT&T グラフ ug_380（ノード数 1,104，エッジ数 3,231）の対話的配置
Fig. 6 Interactive layout of the AT&T graph ug_380 (with 1,104 nodes and 3,231 edges).

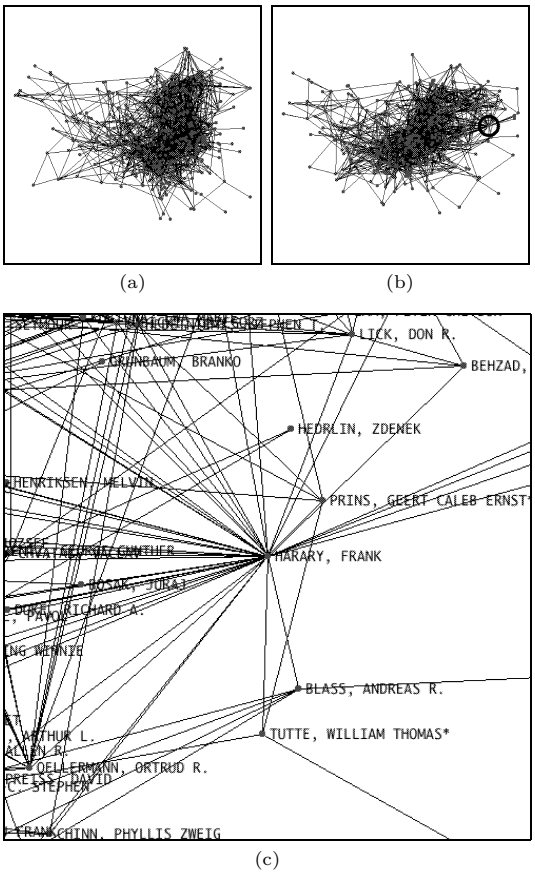


図 7 ソーシャルネットワークを表す現実のデータへの適用
Fig. 7 An application to a real-world example showing a social network.

フとして，Erdös Number プロジェクト☆で作成された，erdos1graph と呼ばれるグラフを修正したものを用いた．このプロジェクトでは，1996 年に他界した数学者 Paul Erdös を起点とした，論文の共著関係に関するデータを収集している．erdos1graph は，Erdös が執筆した論文の共著者をノードとするグラフであり（Erdös 自身は含まれない），ノード間のエッジは，それらに対応する人物同士が共著で論文を執筆したことがある（Erdös が共著者でない場合も含む）ことを示している．したがって，このグラフは，現実世界における人間関係のソーシャルネットワークを表すものであるといえる．なお，本実験では，連結グラフを得るために，元の erdos1graph から，最大の連結成分以外に属する 46 個のノードと 4 本のエッジを除いて得られるグラフ（以下，修正 erdos1graph と呼ぶ）を用いている．修正 erdos1graph は，463 個のノードと 1,547 本のエッジからなる．

図 7 (a) は，本提案手法による修正 erdos1graph の初期配置である．次に，このグラフ配置のほぼ中央に位置する，多数のエッジに接続されたノードを右側にドラッグすることで得られた配置を図 7 (b) に示す（丸印は，ドラッグされたノードを示している）．これ

☆ http://www.oakland.edu/enp/

らの処理に要した時間を表 1 に与える．

図 7 (c) は，ドラッグされたノードを中心として拡大表示し，ノードに対応する人物の名前を付加したものである．ドラッグされたノードは，グラフ理論の研究者 Frank Harary に対応するもので，Erdös の共著者の中での共著関係が 2 番目に多い（すなわち，`erdos1graph` で 2 番目に多くのエッジを持つ）人物である．このノードの周辺には，本提案手法の性質により，このノードに関連の深いノードが配置されていると考えられる．その 1 つである図 7 (c) の下側にあるノードは，古典的なグラフ配置法を提案したことでも知られる William T. Tutte[3] に対応するものである．Tutte は，Erdös の共著者の中では 4 人としか共著関係がなく，Harary はそのうちの 1 人である．このために，Tutte のノードが，Harary の近くに配置されたと考えられる．

6.4 従来手法における典型的な例への適用

最後に，一般無向グラフ配置のための従来手法の評価で用いられる種々の典型的な例に対して，本研究の提案手法を適用した実験の結果を与える．本実験で扱うグラフは，大きく 2 つのグループに分かれる．第 1 のグループは，たかだか数十個のノードを持つ小さいグラフからなり，古典的な力指向アプローチで扱われることの多いものである．具体的には，**図 8** (a) に示される木構造，図 8 (b)，(c)，(d) に示される対称グラフ，図 8 (e)，(f) に示される非対称グラフを扱った（図 8 (b)，(c)，(d)，(e)，(f) は，KK 法に関する文献 15) より引用した）．一方，第 2 のグループは，1,000 個以上のノードを持つメッシュ構造のグラフからなり，最近の一般無向グラフ配置手法[10),11),17)] で，例として扱われているものである．具体的には，**図 9** (a) に示される，64×16 の大きさのトーラス型グラフ（以下，`torus64x16` と呼ぶ）と，図 9 (b) に示される，対角線上で向き合ったノードどうしを接続した 40×40 の大きさのグリッド（以下，`folded_grid40` と呼ぶ）を扱った．図 8 および 図 9 の左側の列に，第 1 グループのグラフについては，KK 法によって得られた配置を，第 2 グループのグラフについては，TKS 法を低次元（`torus64x16` では 4 次元，`folded_grid40` では 2 次元）で実行して得られた配置を示す．

図 8 および 図 9 の中央の列に示されたグラフ配置は，本研究の提案手法によって得られた初期配置である．一方，図 8 および 図 9 の右側の列にある配置は，本提案手法で，少数個のノードをドラッグすることで得られたグラフ配置である（ドラッグされたノードを丸印で示す）．表 1 に，`torus64x16` と `folded_grid40` の処理に要した時間を与える．

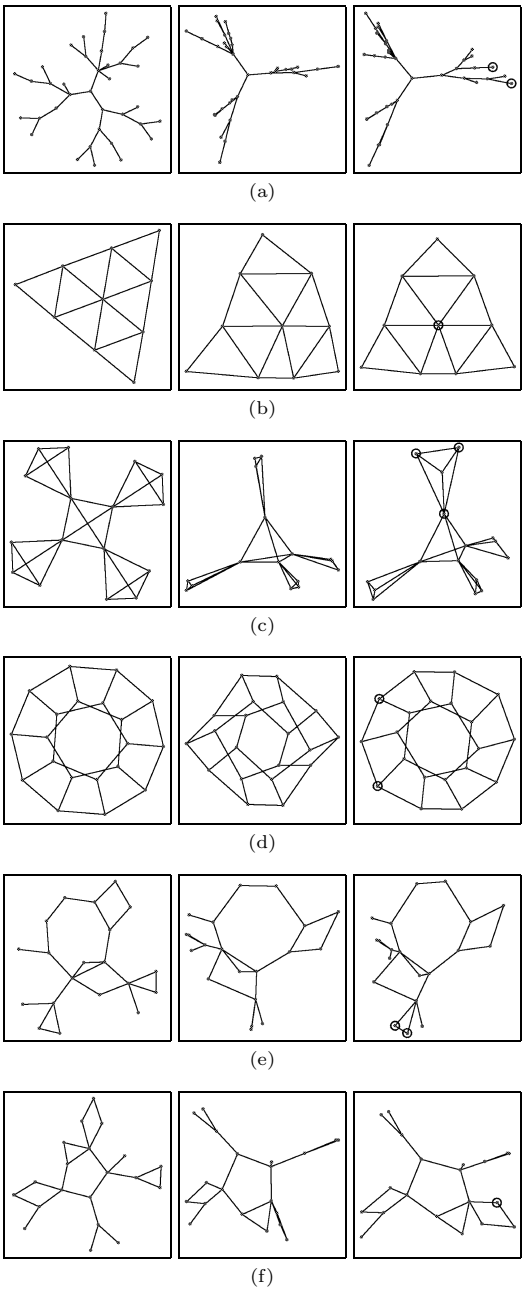


図 8 古典的な力指向アプローチにおける典型的な例への適用
Fig. 8 Applications to typical examples for the classical force-directed approach.

この実験結果が示すように，本提案手法では，特に初期配置において，KK 法や TKS 法のような従来手法に比べて，全体的な構造が歪んでいたり，局所的な構造が見えにくくなっていたりするなどの制限がある．これは，本提案手法で，グラフの 2 次元配置に関する

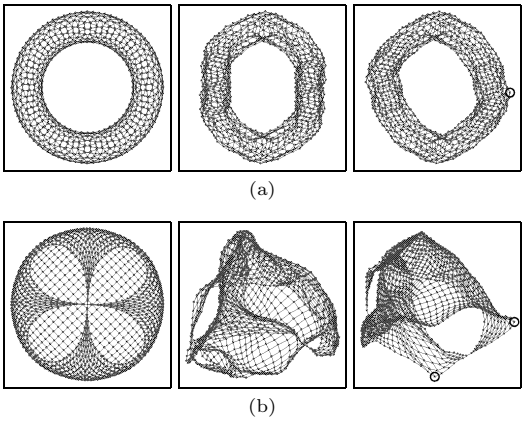


図 9 最近の一般無向グラフ配置手法における典型的な例への適用
Fig. 9 Applications to typical examples for recent general undirected graph layout methods.

自動的な最適化処理を導入していないことによるものである．しかし，本提案手法では対話的なグラフ配置の更新が可能であり，実験結果が示すように，局所的な構造の修正に対して有効であることが多い．特に，`folded_grid40` の例でも示されているように，1,000 個以上のノードからなるグラフを実時間で対話的に配置できるという点は，従来手法にはない利点である．このように，本提案手法は，従来手法と置き換わるものではなく，異なった役割を持つものであるといえる．すなわち，従来手法は，1 つの図における美しい配置を得るのに適しているのに対して，本提案手法は，対話的配置が必要とされるような，ある程度の大きさと複雑さを持つグラフの視覚化に適している．

7. 議　　論

本章では，本研究の提案手法について議論を行う．

7.1 従来手法との比較

一般に，対話的なグラフの視覚化手法は，美しいグラフ配置を求めるためのものと，グラフの構造を調べるためのものの 2 つに分類でき，本研究の提案手法は後者に属すると考えられる．後者の手法としては，特に，Cone Tree[13),24)] などの 3 次元表示を利用するものと，2 章でも述べたフィッシュアイ[7),13),26),27)] に基づくものがさかんに研究されている．これらの手法の詳細については，文献 13) によるサーベイを参照されたい．

通常のフィッシュアイに基づく対話的グラフ視覚化手法では，グラフの 2 次元配置に関する部分的な強調表示を行うことで視覚化が実現される．このため，本研究の提案手法のように，グラフの全体的な 2 次元配置が大きく変更されるようなことは起こらない．したがって，フィッシュアイの対象となるグラフは，巨大ではあるが，基本的に 2 次元平面上で静的に美しく配置可能なものであるといえる．

3 次元表示を用いたグラフの視覚化は，一般無向グラフに対しても適用されることが多い[1),4),6)]．3 次元空間においては，2 次元平面上よりもグラフ配置の自由度が高く，さらに，視点の移動によってグラフの見え方を変更することができるからである．この意味では，本研究の提案手法は，3 次元表示の視点変更に基づくグラフの視覚化を高次元に一般化したものであるといえる．高次元への一般化には，本質的な利点があり，これは以下のように説明できる．内部的なグラフ配置の次元数を d とし，各ノード i $(1 \leq i \leq n)$ の位置を $\boldsymbol{p}_i$ とする．このとき，2 次元グラフ配置における各ノード i の位置を (x_i, y_i) に合わせたいという状況を考える．射影平面を張る d 次元ベクトルを $\boldsymbol{e}_1$，$\boldsymbol{e}_2$ とすると，これらは以下の制約を満たす必要がある．

$$\begin{aligned} \boldsymbol{p}_i \cdot \boldsymbol{e}_1 &= x_i \ (1 \leq i \leq n) \\ \boldsymbol{p}_i \cdot \boldsymbol{e}_2 &= y_i \ (1 \leq i \leq n) \\ \|\boldsymbol{e}_1\| &= 1 \\ \|\boldsymbol{e}_2\| &= 1 \\ \boldsymbol{e}_1 \cdot \boldsymbol{e}_2 &= 0 \end{aligned}$$

ここで，$\boldsymbol{e}_1$，$\boldsymbol{e}_2$ が未知であることから，変数の個数は $2d$ である．一方，制約の個数は，$2n+3$ である．$d \leq n$ より $2d < 2n+3$ であるため，制約過多となって解が存在しない可能性があるが，一般には，制約の個数が変わらなければ，変数が多いほど，解の存在の可能性が高くなる．したがって，内部的なグラフ配置の次元が高いほど，より多様な 2 次元グラフ配置を表現できると考えられる．

1 章でも述べたように，一般無向グラフの対話的配置を実現するために，力指向アプローチを拡張し，仮想的力学シミュレーションでユーザの操作を介入できるようにすることも多い[8),28)]．このような力指向アプローチによる通常の手法と比較したとき，本研究の提案手法は，2 次元グラフ配置の更新が高速であるという利点を持つ．このことは，以下のように説明できる．本提案手法が 2 次元グラフ配置を更新する際に時間を要するのは，新しい射影平面を求める制約解消の部分と，新しい射影平面に対する各ノードの射影を計算する部分の 2 つである．前者については，対象となるグラフの大きさにかかわらず，変数の個数と制約の個数が一定となるため，通常はほぼ一定の時間内に処理を行うことができる（制約解消は反復法に基づくが，問題が単純であるために，反復回数は通常，それほど

多くならない)．また後者については，ノードの個数を n，内部的な次元数を d としたとき，時間計算量が $O(dn)$ となり，この処理は，単純なベクトル計算であるため，最近のプロセッサできわめて高速に実行可能である．一方，力指向アプローチでは，ノード間に働く力を計算し，ノードの配置を改善する反復処理が必要であり，通常は，各反復において $O(n^2)$ の時間を要するうえに，配置が収束するまでに，ある程度の回数の反復を行う必要がある．力指向アプローチに近似的手法を導入することで，各反復における計算量を改善する手法[23]も提案されているが，本研究における提案手法と同程度の速度を達成することは容易でないと考えられる．

7.2 提案手法における制限

本研究では，入力として与えられた一般無向グラフが連結であることを仮定した．グラフが連結でない場合については，グラフ理論的距離の計算結果に無限大が含まれるために判定が可能であり，このことを利用すれば，グラフの各連結成分を別々に配置できる．連結でないグラフを 1 つにまとめて配置したい場合には，グラフ理論的距離が無限大となっているノード間の距離を適当な値に置き換えて計算する方法が考えられる．

本研究の提案手法では，ノード数が増えるにつれて，2 次元グラフ配置の中心付近のノードの密度が増大する傾向がある．このような状況でグラフの特定部分を強調表示したい場合，その部分の適当なノードを外側に向けてドラッグする方法が有効である．しかし，その部分が高次元空間において実際に中心に近い場合には，2 次元配置で中心から遠ざけることができない．このような問題に対処するためには，注目しているノードからグラフ理論的距離の遠いノードを半透明表示するなどの補佐的な表示技術を追加する必要があると考えられる．

本提案手法を用いてグラフの中のエッジの密度が高い部分の対話的配置を行う場合，本手法に特有の問題が生じる．本手法は，ドラッグされているノードに関連の深い他のノードも同時に移動しようとする特徴があるが，その副作用として，密度の高い部分では，過去にドラッグされたノードまで大きく移動してしまうという問題である．この問題を防ぐための方法として，過去にドラッグされたノードについてはその位置を可能な限り変えないようにすることが考えられる．これを実現するには，射影平面を決定するための制約の構成方法を拡張し，複数のノードの位置を基準とすることができるようにする必要がある．

本研究では，高次元グラフ配置を求めるための手法として TKS 法を採用したが，他の手法によっても高次元グラフ配置を求めることは可能である．ただし，高次元グラフ配置を得ることのできる手法が，本研究の目的で利用可能であるとは限らない．なぜなら，射影平面を移動したときの 2 次元グラフ配置の変形が有効に作用しない可能性があるためである☆．しかし，本研究の利用性をさらに向上させるために，高次元グラフ配置の計算を高速化することは必要であると考えられるので，本研究のアプローチに適合する他の高次元グラフ配置法を追及することは重要な課題である．

8. おわりに

本研究では，一般無向グラフの対話的配置のための新しいアプローチとして，高次元空間におけるグラフ配置を用いる手法を提案した．本手法は，2 次元グラフ配置をきわめて高速に計算し，ユーザの操作に合わせて 2 次元グラフ配置を変形するという特徴を持つ．

今後の研究課題として，まず，グラフの高次元配置の計算を高速化することがあげられる．そのために，TKS 法以外の方法によって得られる高次元グラフ配置が，本研究の目的に適合するかどうかの検討を行う．また，本研究において試作したシステムを改良し，その表示と操作の機能を拡充することも予定している．

謝辞 本研究の一部は栢森情報科学振興財団の助成による．

☆ 2 章で紹介した Laplace 行列の固有ベクトルを用いる方法[17),22)]は，TKS 法の場合と同様のアイデアを適用したとしてもうまく働かないことをすでに確認している．

参考文献

1) Bruß, I. and Frick, A.: Fast Interactive 3-D Graph Visualization, *Graph Drawing—GD'95*, LNCS, Vol.1027, pp.99–110, Springer (1996).
2) de Leeuw, J. and Michailides, G.: Graph Layout Techniques and Multidimensional Data Analysis, Preprint 248, Dept. Stat., UCLA (1999).
3) Di Battista, G., Eades, P., Tamassia, R. and Tollis, I.G.: *Graph Drawing: Algorithms for the Visualization of Graphs*, Prentice Hall (1999).
4) Dwyer, T. and Eckersley, P.: WilmaScope—An Interactive 3D Graph Visualisation System, *Graph Drawing—GD2001*, LNCS, Vol.2265, pp.442–443, Springer (2002).
5) Eades, P.: A Heuristic for Graph Drawing, *Congressus Numerantium*, Vol.42, pp.149–160 (1984).

6) Eades, P., Houle, M.E. and Webber, R.: Finding the Best Viewpoints for Three-Dimensional Graph Drawings, *Graph Drawing—GD'97*, LNCS, Vol.1353, pp.87–98, Springer (1997).
7) Fairchild, K.M., Poltrock, S.E. and Furnas, G.W.: SemNet: Three-Dimensional Graphic Representation of Large Knowledge Bases, *Cognitive Science and Its Applications for Human-Computer Interaction*, Lawrence Erlbaum, pp.201–233 (1988).
8) 福地健太郎，豊田正史，喜連川優：Web Community Browser: 大規模 Web グラフ探索ツールとそのインタラクション技術，WISS2002 論文集，日本ソフトウェア科学会 (2002).
9) Gajer, P., Goodrich, M.T. and Kobourov, S.G.: A Multi-Dimensional Approach to Force-Directed Layouts of Large Graphs, *Graph Drawing—GD2000*, LNCS, Vol.1984, pp.211–221, Springer (2000).
10) Harel, D. and Koren, Y.: A Fast Multi-scale Method for Drawing Large Graphs, *Graph Drawing—GD2000*, LNCS, Vol.1984, pp.183–196, Springer (2000).
11) Harel, D. and Koren, Y.: Graph Drawing by High-Dimensional Embedding, *Graph Drawing—GD2002*, LNCS, Vol.2528, pp.207–219, Springer (2002).
12) 林知己夫，飽戸 弘（編）：多次元尺度解析法：その有効性と問題点，サイエンス社 (1976).
13) Herman, I., Melançon, G. and Marshall, M.S.: Graph Visualization and Navigation in Information Visualization: A Survey, *IEEE Trans. Visual. Comput. Gr.*, Vol.6, No.1, pp.24–43 (2000).
14) 石畑 清：アルゴリズムとデータ構造，岩波書店 (1989).
15) Kamada, T. and Kawai, S.: An Algorithm for Drawing General Undirected Graphs, *Inf. Process. Lett.*, Vol.31, No.1, pp.7–15 (1989).
16) Kaufmann, M. and Wagner, D. (Eds.): *Drawing Graphs: Methods and Models*, LNCS, Vol.2025, Springer (2001).
17) Koren, Y., Carmel, L. and Harel, D.: ACE: A Fast Multiscale Eigenvectors Computation for Drawing Huge Graphs, *Proc. IEEE InfoVis*, pp.137–144 (2002).
18) Kruskal, J.B. and Seery, J.B.: Designing Network Diagrams, *Proc. 1st General Conf. on Social Graphics*, pp.22–50, U.S. Dept. Census (1980).
19) Murata, T.: Machine Discovery Based on the Co-occurrence of References in a Search Engine, *Discovery Science—DS'99*, LNAI, Vol.1721, pp.220–229, Springer (1999).
20) Niwa, Y., Nishioka, S., Iwayama, M., Takano, A. and Nitta, Y.: Topic Graph Generation for Query Navigation: Use of Frequency Classes for Topic Extraction, *Proc. Natural Language Processing Pacific Rim Symp.*, pp.95–100 (1997).
21) Orimo, E. and Koike, H.: ZASH: A Browsing System for Multi-Dimensional Data, *Proc. IEEE VL*, pp.288–295 (1999).
22) Pisanski, T. and Shawe-Taylor, J.: Characterizing Graph Drawing with Eigenvectors, *J. Chem. Inf. Comput. Sci.*, Vol.40, No.3, pp.567–571 (2000).
23) Quigley, A.J. and Eades, P.: FADE: Graph Drawing, Clustering, and Visual Abstraction, *Graph Drawing—GD2000*, LNCS, Vol.1984, pp.197–210, Springer (2000).
24) Robertson, G.G., Mackinlay, J.D. and Card, S.K.: Cone Trees: Animated 3D Visualizations of Hierarchical Information, *Proc. ACM CHI*, pp.189–194 (1991).
25) 齋藤堯幸（編）：多次元尺度構成法，朝倉書店 (1980).
26) Sarkar, M. and Brown, M.H.: Graphical Fisheye Views, *Comm. ACM*, Vol.37, No.12, pp.73–83 (1994).
27) 杉山公造：グラフ自動描画法とその応用，コロナ社 (1993).
28) 舘村純一：DocSpace: 文献空間のインタラクティブ視覚化，インタラクティブシステムとソフトウェア IV—日本ソフトウェア科学会 WISS'96，pp.11–20 (1996).

(平成 16 年 10 月 18 日受付)
(平成 17 年 5 月 9 日採録)

細部 博史（正会員）

1969 年生．1993 年東京大学理学部情報科学科卒業．1995 年同大学大学院理学系研究科情報科学専攻修士課程修了．1998 年同専攻博士課程修了．博士（理学）．日本学術振興会特別研究員-PD，文部省学術情報センター助手，国立情報学研究所助手を経て，2004 年より国立情報学研究所助教授．2005 年仏ナント大学計算機科学研究所客員教員．制約プログラミング，ユーザインタフェース，情報視覚化，対話型グラフィクス等に興味を持つ．2003 年日本ソフトウェア科学会高橋奨励賞受賞．日本ソフトウェア科学会，ACM 各会員．