３次元データの見やすい表示角度に関する研究

阿南正明
機械電子部

Study on Easily Understood Display Angle of 3D Range Data

Masaaki ANAN
Mechanics & Electronics Division

要旨

３次元形状データを画面に表示する際，どういう角度で表示するのが見やすいかを選定する方法について検討した．見やすい表示角度，見にくい表示角度の画面表示サンプルをとり，画面上のワイヤーフレームがからまり合う部分などに注目し，「見やすさ」を評価するための関数を試作した．最後に表示角度を自動指定するための，予備的な実験を行った．

１．　はじめに

３次元形状計測装置の出力や，ＣＡＤ上でデザインされた３次元データを扱う際，コンピュータ画面上にそれらのデータをまず表示する機能が必要であるが，３次元データはどのような向きで表示されるかによって，見かけの形が大きく変化し，その中からもっとも見やすい表示角度をユーザがマウスなどで操作して指定するのが一般的である．

対象となる３次元データがたくさんある場合（民芸品や機械部品のデータベースなど），このようなマウス操作を繰り返すのではなく，初期表示の時点で３次元図形の見やすい角度が，ある程度コンピュータ側で選定されることが望ましい．

本研究では，まず３次元図形の見やすいと思われる表示角度，わかりにくいと思われる表示角度の例をつくり，それぞれをワイヤーフレーム表示した場合の線分間の間隙について調べ，「見やすさ」をある程度数値化できるような評価関数の試作を検討した．

このような評価関数を使って，ｘ，ｙ，ｚの３軸まわりに図形を少しずつ回転しながら，それぞれの２次元画面を評価し，もっとも見やすい表示角度を自動選定することが可能であると思われる．（現状では実行速度に問題があるため，今回はｚ軸まわりの回転に対する実験のみを予備的に行った．）

２．　実験方法

２．１　実験用図形の表示

まず表示実験用の３次元形状データを作成した(Fig.1)．この図形は，６５本の線分からなる．

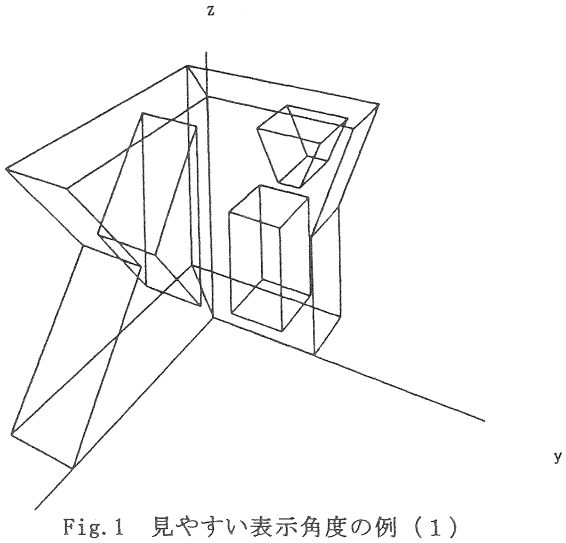


Fig.1　見やすい表示角度の例（１）

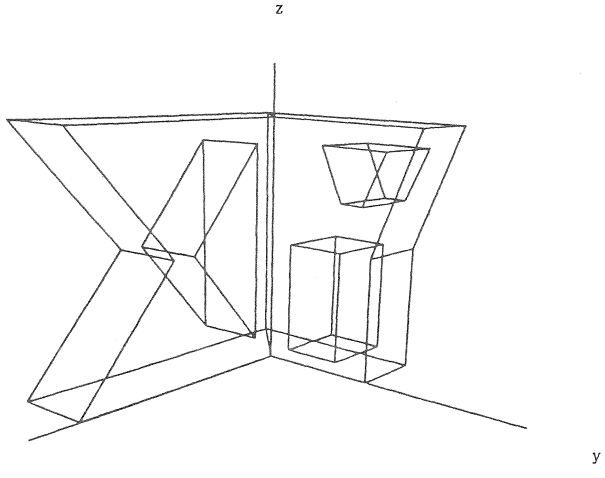


Fig.2　見やすい表示角度の例（２）

コンピュータ画面上にこのような図形を透視投影するために，以下の式を用いた．コンパイラは Windows 上の C++ を使用した．

$$s_x = F \cdot (P_x /(\sin\theta \cdot t_1) - \cos\theta \cdot t_2)$$
$$s_y = (F/\cos\phi) \cdot (\sin\phi - t_2)$$
ただし，
$$t_1 = P_x \cos\theta \sin\phi + P_y \sin\theta \sin\phi + P_z \cos\phi$$
$$t_2 = (P_x \cos\theta + P_y \sin\theta) /t_1$$
..... ①

入力パラメータは，

$E = (E_x, E_y, E_z)$	：視点の位置
$X = (X_x, X_y, X_z)$	：物体の注目点（例えば頂点）
F	：画面と視点の距離
θ	：視線の水平回転角
ϕ	：視線の上下回転角
(s_x, s_y)	：画面上での注目点Ｘの位置

$P = (P_x, P_y, P_z) = X - E$ と置いた．

２．２　評価関数の作成

この図形をさまざまな角度から表示し，比較的見やすいと思われる表示角度（Fig.1, Fig.2）と，比較的見にくいと思われる表示角度（Fig.3, Fig.4, Fig.5）のサンプル画面を保存した．

これに対して３種類の評価関数を次のように定義した．

画面上に（画面の左端から右端まで）高さ y における１本の水平線 L_y を引き，画面上にすでに描画されている線分との交点を左から順に P_{y1}, P_{y2}, P_{y3} .... $P_{yN(y)}$ と書く．また交点の間隙 $D_{yi} = P_{yi+1} - P_{yi}$ と書く．水平線 L_y を画面の上端から下端まで順次動かすものとして，以下の値をそれぞれ求める．

【評価関数Ａ】$= \Sigma (D_{yi} - \overline{D_{yi}})^2/(N(y)-1)$
：この値を画面のｙ軸に対して平均する．

【評価関数Ｂ】＝（$D_{yi} < s$ である D_{yi} の個数）
／（全画面中の D_{yi} の個数）

【評価関数Ｃ】＝（$D_{yi} < s$ である D_{yi} が２つ以上隣り合う箇所の個数）
／（全画面中の D_{yi} の個数）

ただし，　ｓ＝画面幅／100 と設定した．実際には同様の処理を垂直線に対しても行い，両者の平均値を評価関

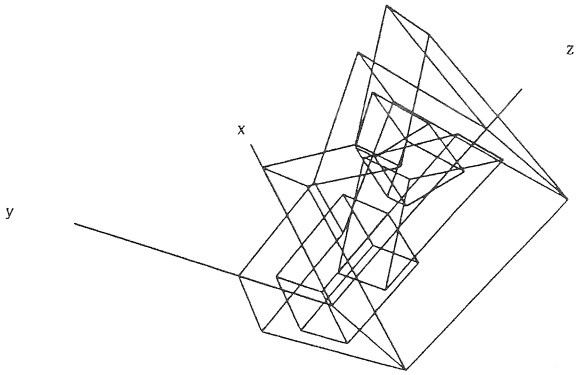


Fig.3　見にくい表示角度の例（１）

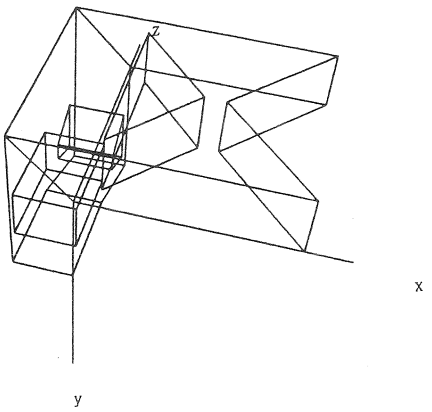


Fig.4　見にくい表示角度の例（２）

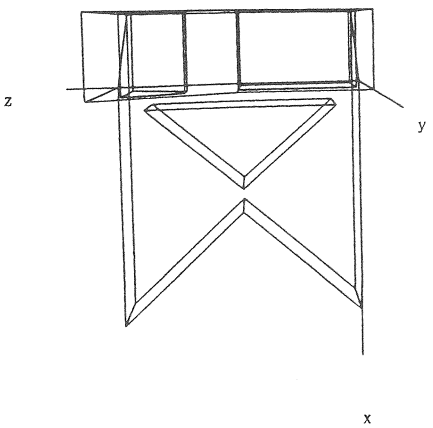


Fig.5　見にくい表示角度の例（３）

数の値とした．各評価関数は以下のような仮定に基づいている．

評価関数Ａ：見やすい図形は線分の間隙 Dyi が均等に分布している．従って平均値との２乗誤差をとればその値は小さくなる．
評価関数Ｂ：見やすい図形は微少な間隙 Dyi が少ない．
評価関数Ｃ：見やすい図形は微少な間隙 Dyi が２つ以上連続するような箇所が，少ない．

２．３　評価関数のテストと自動化の予備実験

Fig.1 ～ Fig.5 のサンプル画面に対して，評価関数Ａ～Ｃを適用し，どのような値が出るかを調べた．

また，図形を回転させながら，同様に評価関数を適用し，「見やすい」角度の選定を行わせた．本来なら，ｘ，ｙ，ｚの３軸まわりで１度きざみの回転をするなどして，その結果得られる 46,656,000 枚の画面の中から最も見やすいものを選ぶべきである．しかし現状ではプログラムの実行時間に問題があり，今回は予備的に，ｚ軸周りの回転を２度刻みで行い，その結果得られる 180 枚の画面から，選定を行った．

３．　結果と考察

３．１　実験結果

各サンプル画面に対する，各評価関数の値を Table 1 に示す．

評価関数Ａに関しては，期待したような結果は得られなかった．評価関数Ｂに関しては，例えば２５％以下のものを「見やすい」と考えれば，見やすさの判定ができる．評価関数Ｃに関しては，例えば５％以下のものを採用すれば，見やすさの判定ができる．またＢよりもＣのほうが値の差が明確である．

Table 1　評価関数の値

サンプル	主観評価	評価関数 A	評価関数 B[%]	評価関数 C[%]
Fig. 1	○	142.0	17,0	3.0
Fig. 2	○	1580.0	23.8	1.7
Fig. 3	×	502.5	28.6	8.0
Fig. 4	×	820.4	28.4	9.6
Fig. 5	×	1998.2	40.7	19.2

そこで評価関数としてはＣを用いることにした．

ｚ＝０～ 360 度の範囲で２度ごとの回転をデータに加えながら，評価関数Ｃに見やすいと思われる画面を選定させた．値が小さかったものの中から，上位５枚を Fig.6 に示す．

３．２　処理速度について

実験システムの実行速度を改善するために，次のような点を検討課題としたい．

○ 現在は画面をなめる（ピクセルゲット）方式のコーディングを行っているが，これを線分の情報を蓄え，直接交点を計算するような方式に改良する．

○ １度きざみで少しずつ回転するのではなく，おおざっぱな回転を先に行い，極小値近辺を集中的に調べるような処理に変更する．

○ 画面の空白部分を処理の初期段階から除外する．

３．３　評価方法について

球や，立方体の稜など，もともと極めて単純な図形を入力した場合，評価関数Ｃが常に０となり，表示角度によらず「見やすい」と判断されることが予想される．実際このような図形は人間の目で見ても，どの角度が見やすいかは一概に判定しにくい．

また，細かい３角パッチを張り巡らした曲面図形のように，評価関数Ｃが常に高い値を保つケースも予想される．

このため，あらかじめ図形の輪郭抽出を行うなど，個々の状況に応じた別処理について検討の必要がある．

３．４　評価関数について

評価関数Ａ，Ｂ，Ｃの中で，Ｃがもっとも有効であることがわかったが，特に評価関数Ａが，期待通りの結果にならなかった理由として，次の２点が考えられる．

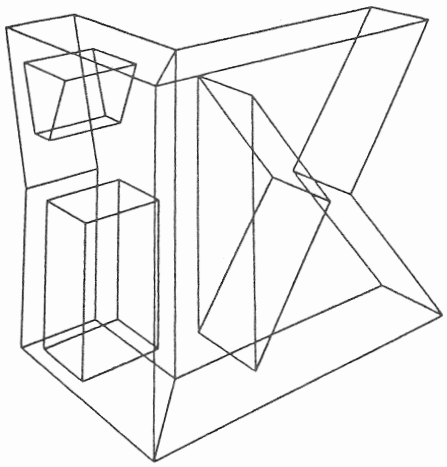
①平均値に対する２乗誤差を求めているため，平均値そのものが極めて小さい場合は，実際には画面上のワイヤーフレームが複雑にからみあっていても，小さな値しか出さない．

②２乗誤差であるから，平均値よりも大きな値についても累積することになるが，Fig.1 に見るように画面上の線分間に大きな間隙があっても，とくに「見にくくなる」というような事はない．

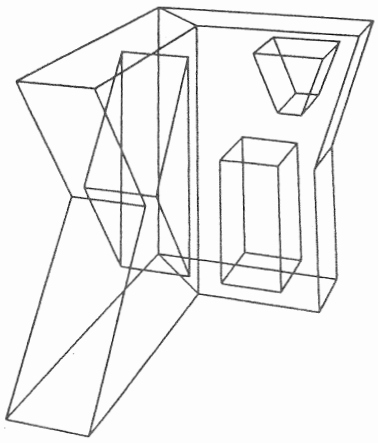
４．　まとめ

ワイヤーフレーム表示された３次元データのもっとも見やすい表示角度を自動的に選定するため，画面の見やすさを評価する関数をいくつか作成し，テストした．その結果，線分の間隙が極めて小さくなる箇所が２個以上連続して現れるものをカウントすることで，画面の「見やすさ」を評価できるという見通しが得られた．

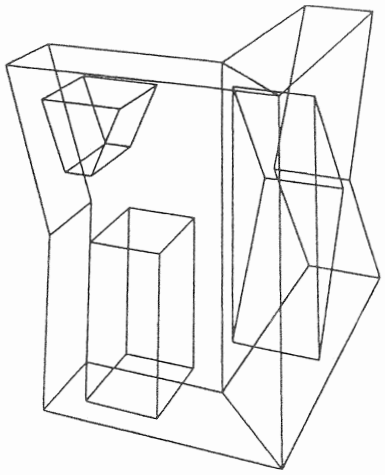
実行速度の改善，ｘ，ｙ，ｚ３軸に対する回転と表示角度の自動選定，任意図形への対応など，今後ひきつづき検討する．



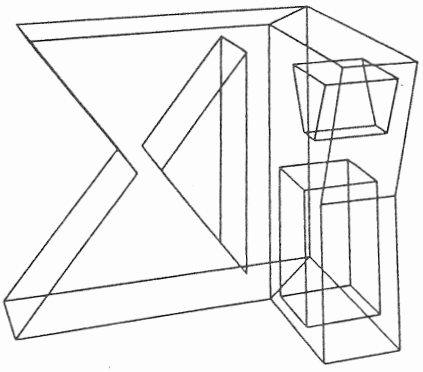
① z=148[degree]，評価関数Ｃ=0.7[%]



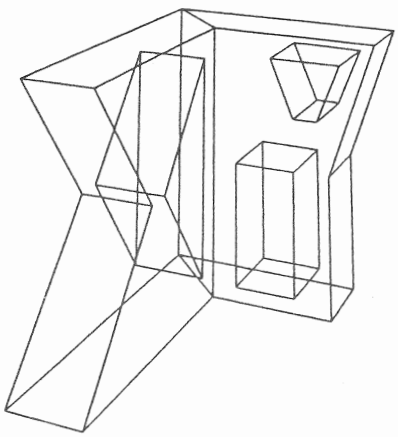
② z=0[degree]，評価関数Ｃ=0.9[%]



③ z=178[degree]，評価関数Ｃ=0.9[%]



④ z=308[degree]，評価関数Ｃ=0.9[%]



⑤ z=356[degree]，評価関数Ｃ=0.9[%]

Fig.6　評価関数Ｃによって「見やすい」と判断された表示角度（上位５枚）

＊ｚ軸まわり　０～360度，２度刻みで回転したものから選定．（他のパラメータは固定）