グラフィクスとCAD 65－4
（1993．10．22）

知識ベースにもとづく三面図の矛盾解消

狩野均　梅沢顯　蒔田尚子　西原清一

筑波大学　電子情報工学系

知識ベースを用いて、三面図から3次元モデルを自動復元する方法を提案する。知識を立体の存在条件、面の接続条件、矛盾解消等の6つのグループに分類し、if-then ルールの形で示す。また、三面図の入力エラーを4種類に分類し、その修正方針を示す。

Removal of Contradictions in a Three Orthographic Views using Knowledge Base

Hitoshi Kanoh　Ken Umezawa　Naoko Makita　Seiichi Nishihara

Institute of Information Sciences and Electronics, University of Tsukuba

In this paper we propose a method to reconstruct a 3-D model of polyhedron from a three orthographic views using knowledge base. All knowledges are classified into 6 groups, that is, existing conditions of polyhedrons, connecting conditions of surfaces, removal of contradictions and so on. These knowledges are expressed by if-then rules. Also, input errors are classified into 4 groups. Policy to correct errors using knowlegdes is given.

1．まえがき

三面図から３次元モデルを自動復元（以下三図面理解と称す）する方法は、ＣＡＤの入力手段として多くの機関で研究されている[1-7]．著者らは以前、三面図から考えられるすべての面を復元し、これらが３次元物体として存在するための制約条件を満たす組みを探索する方法により、２次曲面を含む多面体を復元するシステムを開発実用化した[7,8]。また、線分の属性（実線／破線／鎖線）に関する入力エラーのため面図間に矛盾が生じた場合、線分属性を無視して探索空間を拡大し、線分属性以外は整合性のとれた３次元物体を生成する方法を開発した[9]。

本稿では、これらの開発過程において明らかになった三面図理解に関する知識を、ルールベースの形で表現する方法を示す。一般に３次元物体を２次元平面に射影するとき、情報の縮退がおこる。従って、三面図理解を正しく行うためには何らかの形で表現された知識が必要である。また、面図間の矛盾解消を効率的に行うためにも知識が有効である。

知識の表現方法としては、①プログラムの中に理解手順として表現する、②面・線・点の接続関係を制約ネットワークで表現する、③制約条件を論理式で表現する、④if-then 型の宣言的知識で与える方法などがある。本稿では、人が三面図を理解する方法にもとづくという立場から④を採用した。なお、この方法は、表現が分かりやすいこと、知識の追加・削除がやり易いこと、表現に柔軟性があること等の特徴がある。

以下では、まず三面図理解の概要と知識の分類について述べる。次に、知識の具体例を挙げ、これをルールベース化した例を示す。最後に、入力エラーの修正方針を述べ、修正のためのルールベースを示す。

2．三面図理解の概要と知識の分類

2.1 三面図理解の概要

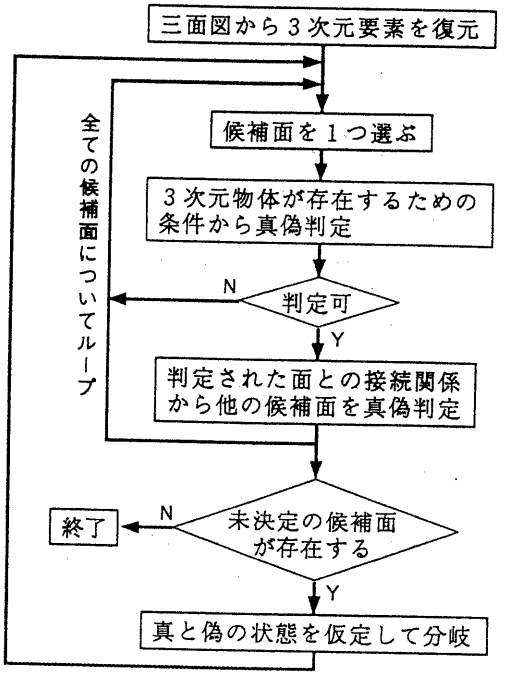

図１　三面図理解手順

入力された三面図にエラーがない場合の処理の流れを図１に示す。また、各処理の概要を以下に示す。

(ｽﾃｯﾌﾟ1) 三面図から頂点、稜線、面などの３次元要素を可能な限り復元する。

(ｽﾃｯﾌﾟ2) 一つの面を、目的の３次元物体を構成する候補面として取り出す。

(ｽﾃｯﾌﾟ3) ３次元物体が存在するための条件（後述）から、候補面の真／偽を判定する。

(ｽﾃｯﾌﾟ4) すでに決定された面との関係から候補面の真／偽を判定する。

(ｽﾃｯﾌﾟ5) 候補面が真または偽であると仮定して探索を継続する。

2.2 知識の分類

本システムでは、知識の利用目的ならびに上記の理解手順との関係から、知識を次の６つのグループに分類することとした。

(1) 存在条件

３次元物体が存在するための最も基本的な条件、

すなわち

「３次元空間における任意の閉曲線は、
３次元物体の表面と偶数回交差する」

から導かれる知識。

(2) 接続条件

一つの面が決定されたとき、この面との接続関係等から、別の面の真／偽を決定するための知識。

(3) 分岐条件

候補面の真／偽判定ができない場合に、真と偽のどちらを仮定した方が探索が効率的に行えるかを判定するための知識。

(4) 優先順位

候補面を取り出す順番を決めるための知識。

(5) 多義解釈

復元される立体に複数解（多義解釈）があるとき、自然な解を優先的に復元するための知識。

(6) 矛盾解消

入力エラーが原因で面図間に矛盾が生じたとき、可能ならこれを解消するための知識。

3．ルールベース

表１　知識の例

①各単純領域の対応面はその領域が輪郭を境界成分として含まない場合は必ず偶数個である。すなわち、未決定面が１つで、既に真と決定されている面が奇数個なら、その面は真である。
②破線を境界とする領域に対応する面はその破線位置において面を隠ぺいする視点により近い他の面が存在する。つまり、破線に対応する稜線を覆い隠す面が存在する。

3.1 面の特徴を表す指標の設定

2.2節の存在条件グループに属す知識の例を表１に示す。この知識をルール化するために、候補面の特徴を表す指標を設定した（表２）。例えば表２の１行目は、"候補面は" という変数が"真"または"偽"の値をとるという意味である。以下に述べるルールは、全てこの指標を用いて記述する。なお、本システムでは現在約７０個の指標を用いている。

表２　指標の例

①候補面は　[真，偽]
②領域が輪郭を境界成分として　[含む，含まない]
③対応面の状態未決定面数が　[1，その他]
④対応面の真の面数が　[偶数，奇数]
⑤候補面の接続面数が　[0，1，2，3，その他]
⑥候補面の接続同一面が　[存在する，存在しない]
⑦候補面の一稜線の投影要素が　[実線，破線，補助線]
⑧候補面の接続面で候補面を覆い隠す面が　[存在する，存在しない]

3.2 ルールの書式

本システムにおける知識の表現形式は、①個々のルールが作り易くわかり易い、②追加変更が容易である、③推論メカニズムが簡単である、④処理スピードが早い、という理由からif-then ルールとした。ルールの一般的な書式は、図面理解に応用することを考慮して次のように決めた。

ルール（G，N，
　[if，　A_1，A_2，・・・・・，A_n]，
　[then，B_1，B_2，・・・・・，B_m]）．・・・(1)

(1)式で、A_i（i=1・・・n）、B_j（j=1・・・m）はそれぞれ独立した事実であり、次のいづれかの形とする。

A_i ： ・Xは［Y_1またはY_2…またはY_k］であるという形の事実。
・prologの項。
・C言語の関数。

B_j ： ・XはYであるという形の事実。
・Prologの項。
・C言語の関数。

このとき(1)式は、

$$A_1 \cap A_2 \cap \cdots \cap A_n \Rightarrow B_1 \cap B_2 \cap \cdots \cap B_m$$

という意味である。

また、Gは2.2節で述べたグループ名、Nはグループ内の通し番号である。

以上のようにすると、表1の知識は次のように簡潔に表現できる。

```
ルール （ 存在条件, 5,
[if,
 [領域が輪郭を境界成分として [含まない]],
 [対応面の状態未決定面 が [ 1 ]],
 [対応面の真の面数が [ 奇数 ]]],
[then,
 [候補面は [ 真 ]]]).

ルール （ 存在条件, 6,
[if,
 [候補面の接続面数が [ 2,3 ]],
 [候補面の接続同一面が [ 存在しない ]],
 [候補面の一稜線の投影要素が [ 破線 ]],
 [候補面の一稜線を共有し候補面を覆い隠す面が
                    [ 存在する ]]],
[then,
 [候補面は [ 偽 ]]]).
```

3.3 ルールベース

本研究では現在約４０個のルールを構築した。ルールはPrologの項で表現し、ワークエリアはPrologのデータベースを利用した。また、指標の計算部分はC言語で記述されている。なお、推論エンジンは、現在Prologで開発中である。

以上のように記述されたルールベースの例を付録に示す。

4．ルールによる矛盾の解消

入力エラーを面図間の矛盾解消という観点から分類することについては前報[9]で述べた。ここでは結果のみ示す（図2）。なお、本稿では、検出できても修正できない矛盾は扱わない。

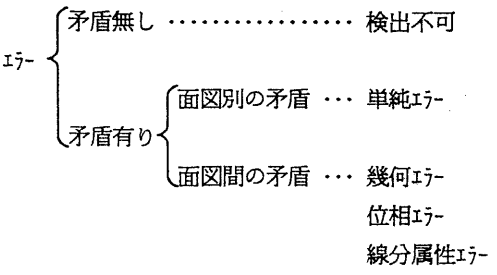

図2　三面図に発生するエラーの分類

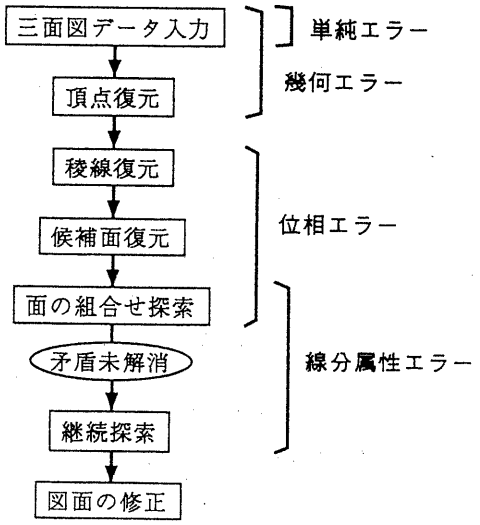

図3　エラーを含む三面図理解手順

図3は、入力エラーを含む三面図の理解手順中にエラーの検出処理を行う箇所を示したものである。図3で、最初の４つの処理が、図1の一番目

の処理に対応する。また、５番目の「面の組合せ探索」は、図１の２番目以降の処理をまとめたものである。

単純エラーからは、孤立点、線分のはみ出し、領域が形成されないなどの矛盾が生じるが、データ入力時点で、線分の閉包などの条件から修正できる。幾何エラーからは、面図間の座標値のずれが生じるが、これは、頂点を復元する時点で修正されるべきと考える。

面の組合せ探索のとき検出されるのは、位相エラー（面図間の線分の不対応、線分の過不足、結線間違い等）と、線分属性エラー（実線／破線／鎖線の書き間違い）である。前者に対しては[存在条件グループ]と[接続条件グループ]の知識(2.2節の(1)と(2))を用いて修正できる。また、後者に対しては[矛盾解消グループ]の知識で対処できる。

５．おわりに

本稿では、三面図理解のための知識を６つのグループに分類して、ルールベースの形で示した。また、ルールを用いた入力エラーの検出方針を示した。エラーの検出に関しては、線分を入力した時点で対話形式で矛盾をチェックする方法が最も有効であると考えるが、本研究では、既存の２次元ＣＡＤデータを対象として検討している。今後は、ルールを追加・改良し、ルールベースによる三面図理解システムを構築する予定である。

参考文献

[1]西田,張,西原：面の組合せ探索による三面図の解釈,人工知能学会第３回全国大会,437-440 (1989).

[2]Kim,Inoue,Nishihara:Heuristic Understanding of Three Orthographic Views, JIP, 15, 4 (1992).

[3]千田：三面図からもとの立体の自動復元-円柱部分を含む立体への適用-,情報処理学会論文誌, 1122-1128 (1991).

[4]宮本,小林,伊藤：外接による三面図からの曲面物体の合成,人工知能学会第４回全国大会,407-410(1990).

[5]横山,河上：三面図から曲面を含む立体の自動生成, 日本機会学会論文誌56巻526号,174-179 (1990)

[6]Sakurai,H. : Solid Model Input Through Orthographic Views, CG17.3,243-252(1989).

[7]井上,金,西原：代数曲面を含む三面図の解釈,情報処理学会研究会,グラフィクスとＣＡＤ CG61-2 (1993).

[8]Computer Today, ７月号(1993)

[9]梅沢,菊池,狩野,西原：曲面を含む三面図の矛盾の検出と理解, 情報処理学会研究会, グラフィクスとＣＡＤ, CG63-7 (1993).

付録：ルールベースの例

```
ルール （ 存在条件, ５,
[if,
 [領域が輪郭を境界成分として [含まない]],
 [対応面の状態未決定面 が [ １ ]],
 [対応面の真の面数が [ 奇数 ]]],
[then,
 [候補面は [ 真 ]]]).

ルール （ 存在条件, ６,
[if,
 [候補面の接続面数が [ ２,３ ]],
 [候補面の接続同一面が [ 存在しない ]],
 [候補面の一稜線の投影要素が [ 破線 ]],
 [候補面の一稜線を共有し候補面を覆い隠す面が
                        [ 存在する ]]],
[then,
 [候補面は [ 偽 ]]]).

ルール （ 接続条件, １,
[if,
```

[決定面の状態が [真]],
[接続面数が [2]]],
[then,
[未決定接続面の状態は [真]]]).

ルール (接続条件, 6,
[if,
[決定面の状態が [真]],
[対応面数が [2]],
[領域が輪郭を境界成分として [含む]]],
[then,
[対応面は [真]]]).

ルール (分岐条件, 1,
[if,
[候補面に交差面が [あり]]],
[then,
[先に探索するのは [偽]]]).

ルール (優先順位, 1,
[if,
[領域の種類が [単純領域]],
[対応面の状態未決定面数が [4以上]],
[領域を構成する線分のすべての対応稜線が
[未使用]],
[最前面の状態が [未決定]],
[最前面の状態を [決定不可能]]],
[then,
[対応面の状態決定を [行わない]]]).

ルール (分岐条件, 4,
[if,
[候補面の位置が [最前]],
[二番面の投影領域数が [3]]],
[then,
[先に探索するのは [偽]]]).

ルール (矛盾解消, 2,
[if,
[最前面が [真]],
[最前面の接続面が [真]],
[最前面と接続面が [滑接する]],
[2面間の稜線が [シルエット稜線]]],
[then,
[2面間の稜線の対応線分の属性は [実線]]]).

ルール (矛盾解消, 5,
[if,
[線分の対応稜線の接続面が全て [決定済]],
[全ての対応稜線の接続面が [滑接している]],
[接稜線が [1本以上]]],
[then,
[線分の属性は [一点鎖線]]]).