コンピュータビジョン 42－1
（1986. 5. 22）

フレーム型物体表面モデルによる3Dビジョン

濱 利行　石塚 満
東京大学生産技術研究所

モデル表現として、構造記述及びモデル間の関係記述の両方を持つビジョンシステムについて検討し、プロトタイプを作成した。形状表現は、表面モデルを採用し、各モデルはオブジェクト指向システムであるCAMPSのクラスに対応させている。認識過程ではクラス間のISA関係を利用してより抽象的なモデルからマッチングをとっていき、順次拘束条件のきつい具体的なモデルへ至る方法（記号レベルでのcoarse-to-fine法）を採っている。

Frame-based 3D Vision System with a Object Surface Modeling.

Toshiyuki Hama　Mitsuru Ishizuka
Institute of Industrial Science, University of Tokyo

A 3D vision system using a modeling with class concept has been constructed. The whole system is built based on the object-oriented programing paradigm, which is very effective to implement a complex system incrementally. The models, which works as references in the understanding process, are represented as hierachical compositions of plane clusters, planes and line segments. For each hierachical level, the models are defined from coarse description to fine one utilizing the inheritance function of the object-oriented programing. This model description allows an efficient coarse-to-fine understanding of 3D object from input image. The implemented system also has a function to supplement incomplete edge-lines caused by an illumination condition or occlusion.

フレーム型物体表面モデルによる３Ｄビジョン

濱　利行　　石塚　満
東京大学生産技術研究所

1 はじめに

従来のビジョン・システムの多くでは、モデルが個別に表現され、表現としては各種の方法が提案されてきた。確かに複雑な物体の記述方式の確立及びそれに基づいて物体の同定ができることは重要なことである。しかしながら、モデルがそれぞれ個別のままであると、物体とモデルの同一性判断で終ってしまうことになる。コンピュータによるより深い理解を考えた場合、それぞれのモデル間の関係もモデル表現中に記述されていることが望ましいと考えた。

このような考えから従来どおりのモデルの構造記述（PARTOF関係）及びモデル間の関係記述（ISA関係）の両方をオブジェクト指向のクラス概念を用いて表現したモデルを持つビジョン・システムについて検討し、プロトタイプを作成した。

形状表現は、表面モデルにより表現し各モデルはオブジェクト指向システムであるCAMPSのクラスに対応させている。認識過程ではクラス間のISA関係を利用してより抽象的なモデルからマッチングをとっていき順次拘束条件のきつい具体的なモデルへ至る方法（記号レベルのcoarse-to-fine法）を採っている。

ビジョン・システムの特徴

(1) 立体モデル、平面モデルが独立したモデル体系である。
(2) それぞれのモデルは１つの形状概念を表現している。
(3) モデル間のISA関係による継承を利用して表現されている。
(4) モデル表現、推論機構、画像処理すべてがオブジェクト指向の枠組みで 記述されている。
(5) 処理結果、推論過程のウィンドウへの表示機能。

2 柔軟なモデル表現とクラス概念の親和性

画像理解におけるモデル表現の要件として必要であると思われる点をまとめると、

(1) １つの概念の対象となる物体の本質だけを表現に用い、個別の物体における差異をできるだけ反映させない。
(2) そのような表現を可能とする形状表現を採る。しかし、実際の画像とのマッチング過程を無視してはいけない。
(3) PARTOF関係を用い構造を明示的に表現する。
(4) ISA関係による概念間の関係記述により、理解を深める。

このような知識表現法としてはフレームがよく用いられている。フレームによる知識表現では、１つの概念に対して１つのフレームが対応し見とおしのよい知識表現法である。

モデル表現だけであればフレームの機能で十分であるが、フレームは階層的な知識ベースとして用いることが多く、受動的な知識表現である。しかし、画像理解ではマッチング過程で能動的な推論が必要となるので、そのような能動的手続きを含めたオブジェクト指向パラダイムが、画像理解には向いている。

しかし、もともとフレームとオブジェクト指向は異なる概念ではなく、オブジェクト指向は、フレームの階層性に手続きも含めたより広い概念である。フレームを実現したシステムであるFRLは、KRL,UNITを経てオブジェクト指向型言語LOOPSに至っている。

また、オブジェクト指向では、抽象的オブジェクトであるクラスと具体的オブジェクトであるインスタンスが、概念としてはっきり分れているので、抽象的モデル表現をクラスに対応させ画像上の特徴から得られた実体でモデルとマッチングしたものをインスタンスに対応させることができる。

以上のようにオブジェクト指向パラダイムの中に画像理解で必要なモデル表現、推論手続き、画像から得られる具体的実体がうまくあてはまる。

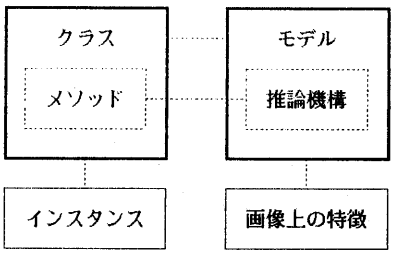

図1 クラス概念とモデル

3 オブジェクト指向システムＣＡＭＰＳ

CAMPS(Classes and Methods Programing System)は、lisp環境下でオブジェクト指向のプログラムを実現するために作成した。CAMPSでは、フレームのデータ構造をオブジェクトとして扱っている。フレーム表現はFRL[1]をその他はFlavor[2]、LOOPS[3]を参考とした。

CAMPSの特徴は次の通りである。

(1) オブジェクトは、クラスとインスタンスだけである。Smalltalkや、LOOPSのようなメタクラスは無い。

(2) マルチプル・インヘリタンス機構がある。

(3) クラス変数、インスタンス変数にあたるクラス・スロット、インスタンス・スロットを持つ。

(4) インスタンス・スロットに対する付加手続きによって、データ指向型のプログラミングも可能である。

CAMPSは、ワークステーションRidgeのUNIX環境の下にfranz lispで記述されている。

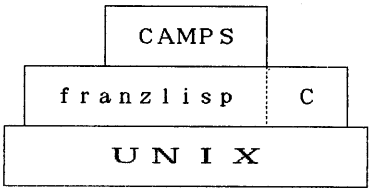

図2 CAMPSの構成

3.1 クラス定義

クラスの定義にはlisp関数defclassを用いる。

定義の形式

```
(defclass class (superclass_list)
    (class_slot_list)
    (instance_slot1  (facet1  value1)
                     (facet2  value2)
                      ..............)
    ..............................
    ($class_slot1                value1)
    ..............................  )
```

スーパークラス

マルチプル・インヘリタンス機構を持つのでスーパークラスは複数定義可能である。しかし、スーパークラスはすべて既に定義されたクラスでなければならない。複数定義する場合、スーパークラスの記述の順番はスロットの継承、メソッドの探索に重要な意味を持つ（左側優先、縦型探索）。

スロットのオプション

(*) インスタンス・スロット

インスタンス生成時に値が埋る必要のあるインスタンス・スロットにつける。このスロットは、インスタンス生成時に、デフォルト値、初期値が無い場合は付加手続き:if_neededにより値を得る。

($) クラス・スロット

クラス・スロットに初期値を与えたい場合、インスタンス・スロットと区別するためこのオプションを付けて初期値を定義する。

インスタンス・スロット用ファセット

インスタンス・スロットには次の６種類のファセットを使用できる。

:type	スロット値がアトムかリストかの指定（指定を省略するとアトムとみなす）
:default	デフォルト値
:if_needed	スロット値参照時の付加手続き
:if_filled	スロット値代入時の付加手続き
:if_added	スロット値追加時の付加手続き（リスト型のみ）
:if_removed	スロット値削除時の付加手続き

付加手続きを指定するファセットは、値としてメソッド名をとるが:if_neededを除いて、操作を受けた値を引数としてメソッドに渡すため指定されたメソッドは１つの引数を取るメソッドでなければならない。

3.2 メソッド定義

メソッドには、クラス・メソッドとインスタンス・メソッドがある。定義には、それぞれlisp関数defcmethod, defmethodを用いる。

定義の形式

クラス・メソッド

```
(defcmethod class method (argument)
    (local_variable_list)
      lisp functions
      ..............................)
```

インスタンス・メソッド

```
(defmethod  class method (argument)
    (local_variable_list)
      lisp functions
      ..............................)
```

メソッド本体の評価

メソッドの本体のlisp関数列は普通のlisp関数定義の場合と同様に上から順に評価され、最後に評価された値がメッセージの送り手に返される。

ｓｅｌｆ

メソッド定義内ではselfは特別なアトムであり、メッセージを受けたオブジェクト自身を示す。

局所変数

メソッド内で使用する局所変数は引数の後に定義する。初期値の与え方は、一般のlispのletの場合と同じである。

3.3 メッセージ転送

メソッドの駆動のためにはlisp関数>>,>>>,^^,^^^を用いる。

形式

クラス・メソッド

```
(>>> 'class method [ 'arg1 ...])
(^^^ 'class method [ 'arg1 ...])
```

インスタンス・メソッド

```
(>>  'class method [ 'arg1 ...])
(^^  'class method [ 'arg2 ...])
```

>>>,>>と^^^,^^の違い

クラス・メソッド、インスタンス・メソッドそれぞれにメッセージ転送用の関数が２つずつ用意されている。>>>,>>は、通常のメッセージ転送であり、メソッド探索はメッセージの受け手の属するクラスから始められる。これに対して、^^^,^^によってメッセージ転送を行うと、下位のクラスのメソッド定義によって隠蔽されている上位のクラスの同名のメソッドが駆動される。Smalltalkでは、superにあたるメソッド探索である。

3.4 インスタンス生成

一般のインスタンス生成は、CAMPSの最上位クラスであるobjectに定義されているクラス・メソッドnewを用いる。

形式

```
(>>> 'class new
      [ 'instancename (facet1 value1)...])
```

インスタンス名、初期値のオプション

newにオプションの引数を与えることで、インスタンス名、インスタンス・スロットに対する初期値の設定を行うことができる。インスタンス名の指定を省略した場合には、クラス名に数字の付いたアトムがインスタンス名として自動的に付られる。

4 物体表面モデルの記述

4.1 立体モデルと平面モデル

CAMPSのクラスを用いて画像理解のためのモデルを表現するのであるが、まず形状表現の方法を選択しなければならない。一般円筒のような体積指向型の形状表現は表現能力の点では十分であるがマッチング過程が間接的になり難しい。また、三次元のモデルを導入する以前にマルチプル２Ｄによる形状表現を用いていた[4]こともあって表面モデルによって形状を表現することにした。

表面モデルを用いた場合、曲面が入ると抽象的表現が難しいだけでなく表面モデルそのものの限界もあり、認識対象は多面体だけに限定することにする。

３Ｄ物体のモデルが１つの立体の概念を表現できるようにするためには、その構成要素である平面のモデルも１つの平面の概念を表現していなければならない。そのためには平面のモデルは立体のモデルから独立して表現されている必要がある。実際のモデル表現では立体モデル、平面モデルの形状を次のようにして表現する。

立体モデル

立体モデルは、平面の種類と接続関係によって形状が記述されているが、多面体という限られた世界の中でも記述可能な形状は多種多様で、ISA関係で分類するには対象が広すぎる。そこで、凸多面体を基本モデルとしてISA関係の中で（インヘリタンス機能を用いて）表現することにし、基本モデルの簡単な組合わせでできる凹部を含んだ多面体は個別に複合モデルとして表現することにする。

ISA関係の分類は、平面の接続関係、平面の種類を基準とする。

・接続状態による分類

接続状態が同じもので１つのクラスとする。このクラスをbasicモデルと呼び、basicモデルにだけ平面の接続関係を記述しサブクラスに継承させる。

・平面の種類による分類

basicモデルをこの基準でサブクラスに分類する。このクラスをspecificモデルと呼び、接続状態はbasicモデルから継承し、specificモデルには平面の種類だけを記述する。

平面モデル

平面モデルは構成要素が直線の境界だけであるのでISA関係による分類は立体モデルのように対象を限定しなくてもよい。接続状態は一意的であるので、境界の数とそれらの関係により分類する。

・境界数による分類

境界の数が同じものを１つのクラスとする。立体モデルの場合と同様にこのクラスをbasicモデルと呼び、basicモデルにだけ接続関係を記述してサブクラスに継承させる。

・境界の関係による分類

境界間の関係として平行、等長を用いる。basicモデルをこの基準でサブクラスに分類する。このクラスをspecificモデルと呼び平行、等長の関係だけを記述する。

4.2 平面モデルのクラスによる記述形式

立体モデルではファセットにより形状を記述するが平面モデルはクラス・スロットにまとめて記述する。そのため境界スロットには平面スロットのようなファセットは用いない。

モデル表現用クラス・スロット

type	basicモデル、specificモデルの区別。
requestslot	境界スロットを左回りに記述。
request	境界スロット間の接続状態、関係記述。

関係としてはCONT（右回り接続）、PARA（平行）、SAME（等長）を用いる。

requestスロットの記述形式

```
スロット値＝（条件１　条件２　・・・・）
      条件＝（関係１　関係２　・・・・）
      関係＝（CONT  slot1  slot2）｜
            （PARA  slot1  slot2）｜
            （SAME  slot1  slot2）
```

なお、条件はＯＲ、関係はＡＮＤの関係にある。CONTはbasicモデルにのみ使用する。

4.3 立体モデルのクラスによる記述形式

立体モデルでは平面の種類、接続状態をファセットによって記述する。平面スロットは次の形式を採る。

```
（平面slot　　（:class　平面class）
              （:link　（接続平面slot））
```

モデル表現用ファセット

:class　そのスロットを埋めるべき平面モデルのクラスを記述する。

:link　その平面を中心として右回りに接続している平面スロットのリストを記述する。（basicモデルのみ）

モデル表現用クラス・スロット

type　basicモデル、specificモデルの区別。

requestslot　平面スロットのリストを記述。

equivalent　互いに対称な位置関係にある平面スロットの組を記述する。モデル表現にとって本質的ではないが無駄な仮説の生成を防ぐ。

```
(defclass 6kaku (plane)
        (request type requestslot)
        (hasa               (:if_filled nil)
                            (:if_added nil))
        (b1      )
        (b2      )
        (b3      )
        (b4      )
        (b5      )
        (b6      )
        ($type              basic)
        ($requestslot (b1 b2 b3 b4 b5 b6))
        ($request           (((CONT b2 b1)
                              (CONT b3 b2)
                              (CONT b4 b3)
                              (CONT b5 b4)
                              (CONT b6 b5)
                              (CONT b1 b6)))))
```

図3 平面モデル記述例

```
(defclass generic_3kaku_chu (chu)
        (type equivalent requestslot)
        (p1                 (:class 3kaku)
                            (:link (p2 p3 p4)))
        (p2                 (:class 4kaku)
                            (:link (p1 p4 p5 p3)))
        (p3                 (:class 4kaku)
                            (:link (p1 p2 p5 p4)))
        (p4                 (:class 4kaku)
                            (:link (p1 p3 p5 p2)))
        (p5                 (:class 3kaku)
                            (:link (p4 p3 p2)))
        ($equivalent        ((p1 p5) (p2 p3 p4)))
        ($requestslot       (p1 p2 p3 p4 p5))
        ($type              basic))
```

図4 立体モデル記述例

5 画像理解用クラス構成

画像理解に必要なクラスとしてimage,line,model,cluster,plane,boundary等をモデル表現クラス以外にCAMPSに定義している。図5にそれらの構成を示す。

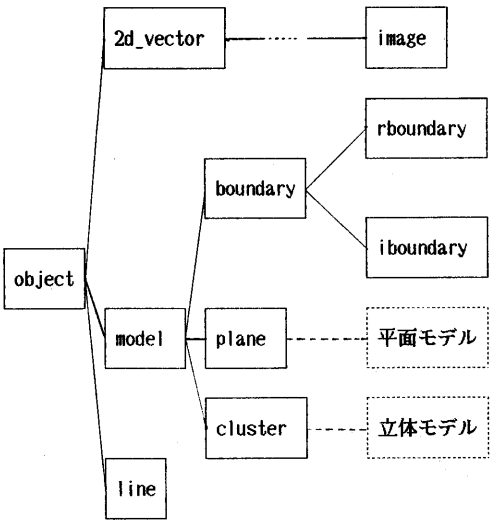

図5 画像理解用クラス構成

image

imageは、２次元ベクタのサブクラスとして定義された画像データを持つオブジェクトである。

主な機能

・直線成分抽出(filter,trace)

・処理結果の表示(draw_image,draw_edge,draw_line)

line

imageでの画像処理の結果得られる直線成分を表現するオブジェクト。

主な機能

・左右濃度差、長さの範囲指定にによる直線インスタンスの取り出し(stronge　r,longer)

・直線の左右に対応する境界インスタンスの生成(candidate)

・直線のウィンドウへの表示(draw,wdraw)

model

画像理解のモデル表現、推論機構を持つクラスの最上位クラス。このクラス固有のスロットはない。

主な機能

・立体インスタンス、平面インスタンスに対するクラスの問い合せ(is,isnot)

cluster

立体のモデル表現クラスの最上位クラス

主な機能

・核構造を持つ立体インスタンスの生成と仮説の立体インスタンスの生成(sea　rch,match,slotmatch ...)

・仮説の立体インスタンスの検証(verify,verify_dummy,visible? ...)

・立体インスタンスの表示(draw,wdraw)

plane

平面のモデル表現クラスの最上位クラス。

主な機能
・境界追跡による平面の完成(search,insearch,outsearch ...)
・不完全な平面の検証(match,match_dummy,cand_match..)
・平面インスタンスの表示(draw,wdraw,draw_member)

boundary
境界表すクラスの最上位クラス
主な機能
・接続可能性のある境界インスタンスの追跡(incontinue,outcontinue)
・境界間の関係の検証(cont,pararell,same)
・境界インスタンスの表示(draw,wdraw)

rboundary
直線インスタンスから生成される境界を表すクラス
主な機能
・隣り合う平面の取り出し(neigh_inpalne,neigh_outplane)

iboundary
rboundaryのインスタンスを連結してできる境界、平面の検証用のダミー境界を表すクラス
主な機能
・検証用ダミー境界の生成(make_dummy,dummy_link)

6 画像理解過程

画像理解の過程は、図6に示すように直線抽出、平面推論、立体推論、仮説検証の４つの過程からなる。また画像はCAMPSに取込まれる前に圧縮、平滑化の処理をする。

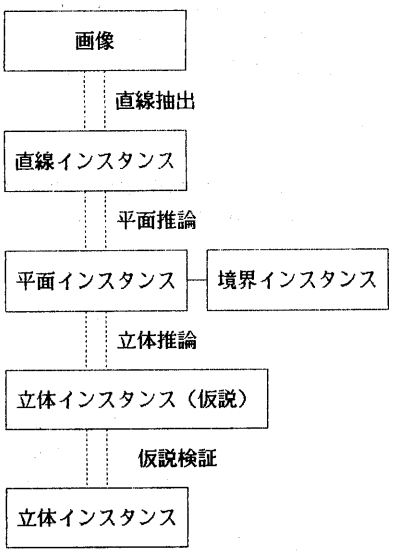

図6 画像理解過程

6.1 平面推論過程

この過程では境界の追跡により平面を形成する。完全な平面が形成されると平面モデルとのマッチングをとり形状を判断する。

境界の追跡による平面の形成
(1) 追跡対象の境界インスタンス生成

lineのインスタンスには雑音によってできたものも含まれるので、左右の濃度差、長さが、gray_diff,length(planeｸﾗｽ)の値より大きいlineのインスタンスをもとにして追跡対象の境界インスタンスを生成する。

(2) 境界インスタンスの追跡

追跡されていない境界インスタンスを１つ取り出しこれを含む平面として１つの平面インスタンスを生成する。この平面インスタンスはsearchのメッセージを受け取るとhasaの先頭にある境界インスタンスに接続境界を探すメッセージ(incontinue,outcontinue)を追跡が終了するまで送り続けて平面を完成させていく。

・接続境界の条件
1.濃度差が閾値gdiff(boundaryｸﾗｽ)以下である。
2.距離が閾値cont_radius(boundaryｸﾗｽ)以内である。
3.既にその平面に含まれる境界インスタンスと隣り合う境界インスタンス　prohibit(planeｲﾝｽﾀﾝｽ)に含まれていない。

・接続状態の分類
incont　回転角度が負（凸部）
colinear　回転角度が０に近い（連続）
outcont　回転角度が正（凹部）

境界インスタンスは平面領域がその向きに対して右側に位置するように生成されるので接続境界とのなす角度が負の場合は凸部、正の場合は凹部であることがわかる。１つの平面インスタンスに凸部と凹部を含めてしまうと形状判断が複雑となるので、凸部、凹部は分離する。

また、画像処理によって得られる直線成分は不完全であったり雑音を含んでいたりするので、接続境界の条件を満たすものはすべて可能性として受理する。

・追跡の停止条件
1.接続境界が見つからない
2.接続境界が既にその平面に含まれている
3.接続境界が追跡済みで他の平面に含まれている
(3の場合には接続境界が含まれている平面インスタンスに自分自身を連結して１つの平面とする(inlink,outlink))

平面インスタンスの分類

追跡の終了した平面インスタンスは次のように分類されている。

凹凸(trace)による分類
in　凸平面あるいはその一部分
out 凹平面の一部分あるいは立体、穴の外周

完成度(complete)による分類
complete　境界で閉領域を囲むもの

imcomp1　境界の回転角の総和(tangle)が180°より小さいもの
incomp2　境界の回転角の総和(tangle)が180°に近いもの
incomp3　境界の回転角の総和(tangle)が180°より大きいもの
（incomp3は凹部を含む平面の一部である可能性がある）

以下では、completeを完全平面、それ以外を不完全平面と呼ぶ。

形状判断

平面インスタンスの形状判断は完全平面インスタンスに対してのみ行う。不完全平面インスタンスは立体モデルからの情報なしでは独自に判断することはできないので、保留しておく。

完全平面の形状判断は比較的容易である。平面モデルのbasicモデルは境界の数によって分類されているので、完全平面インスタンスでは対応するbasicモデルが一意的に決まる。basicモデルのインスタンスとなった平面インスタンスは、basicモデルのサブクラスであるspecificモデルとのマッチングを取りながらできるだけ特殊なモデルのインスタンスへと変更されていき(specify)どのモデルのインスタンスとなったかで形状が決まる。

6.2 立体推論過程

この過程では完全平面の接続状態とbasicモデルとのマッチングにより、仮説となるbasicモデルのインスタンスを生成して行く。

核構造を持つ立体インスタンスの生成

立体の仮説を生成するのには、ほぼ確実であると思われる完全平面を対象とする。完全平面インスタンスから１つを選び、それと隣接する完全平面インスタンスを順次取り出して行く。この互いに隣接し合った完全平面インスタンスを核構造と呼ぶ。

核構造とbasicモデルとのマッチング

核構造を持った立体インスタンス（以下略して核インスタンス）とbasicモデルとのマッチングをとり、成功した場合には仮説として核インスタンスの複製を作りbasicモデルのインスタンスに変更する。そして、核構造の完全平面インスタンスを対応する平面スロットに代入する(make_hypo)。

6.3 仮説検証過程

この過程では仮説インスタンスの満たされていない平面スロットに対する検証を行う。立体のモデルは３次元的な接続関係によって記述されているため、仮説インスタンスの平面スロットはすべてが満たされることはない。そのため見え方の検査をし、その平面スロットを満たす平面インスタンスが存在すべきかどうかを調べ、そのあとで満たすべき平面インスタンスの生成をする。

見え方のチェック

平面スロットが満たされていなくても隣接する平面スロットが満たされていればそのスロットを満たすべき平面インスタンスが持つ境界インスタンスがわかる。この境界インスタンスからその平面が見えるべきかを判断する(visible?)。

・境界インスタンスが連続している時

境界インスタンスには向きがあり、この向きにそって境界をたどった時、見　えている平面では回転角が負(visible)であり、見えていない平面では回転　角が正(invisible)である。

・境界インスタンスが連続していない時

判断することはできない(unknown)。

平面モデルのインスタンス生成

完全平面インスタンスはすべて核構造となっているので、不完全平面インスタンスを対象として指定された平面モデルのインスタンスを生成する。不完全平面インスタンスの境界は正しく追跡されているとは限らないのですべての可能性を試みる必要がある。

ダミー境界よる検証

画像撮影時の光線の具合によって直線成分が不完全である場合がある。この時は対応する境界インスタンスが存在しないので仮説インスタンスはいつまでも満たされない。そこで、平面スロットが満たされるべきであるのに平面モデルのインスタンスが生成できない時は、ダミーの境界を１つの平面につき１つだけ挿入して(make_dummy)検証をする。

次の場合はダミー境界は挿入しない。

1.引こうとする部分に交差する直線インスタンスがある
2.引こうとする両端の境界インスタンスのどちらかが隠されている時

これ以外の場合はダミー境界の挿入により検証を続ける。また、ダミー境界以外がすべて確かな境界の時、ダミーが引けなければ仮説は否定される。

検証結果の分類

完全平面インスタンスが少ない場合（不完全な画像）には、多くの仮説が生成される。また、仮説の否定は見え方がinvisibleの時とダミー境界挿入の時だけであるので、検証後も可能性として多くの仮説が残る。そこで、検証後の完成度(complete)で差別化する。

・完成度による分類

complete　必要な平面スロットがすべて満たされている
occluded　平面スロットは満たされたが一部を隠された平面インスタンスを持つ
dummy　平面スロットは満たされたがダミー境界を挿入した
incomplete　満たされない平面スロットが残っている
illegal　仮説が否定された

この分類の結果、上から確からしいとして順に結果が表示される。

7 結論

本論文では、画像理解における物体のモデル表現はどうあるべきかを中心テーマとして,実際に作成したビジョン・システムとその画像理解のプロセスについて述べた。

モデルの形状記述に平面だけの表面モデルを用いていること、単眼視であるため絶対量による記述を省いている等の理由により形状表現能力には欠けるが、オブジェクト指向パラダイムでのクラス間の継承関係によるモデル記述は概念レベルでのモデル記述には有効であると考える。また、立体モデル、平面モデルの２つのモデル表現体系を独立させたので立体モデルの記述の際に表面を個別に記述する必要がなく見とおしのよいモデル表現となったと考える。

ISA関係はACRONYM[5]のモータの例のように、形状そのものではなく事物間の関係として用いるのは適当である。本システムでは、ISA関係を形状概念そのものの記述に適用し、記号レベルのcoarse-to-fine法を推論により実現したが、対象を多面体に限定しても現実の対象にはISA関係を利用してうまく表現できない物体が多数存在する。

深いモデルとしてのソリッドモデルの導入、理解の手がかりもモデル中に記述し推論を制御するなどとともに、今後の課題である。

[参考文献]

[1] R.Roberts I.Goldstein "The FRL PRIMER" AI Memo No.408 MIT Jul 1977

[2] D.Weinreb "Flavors:Message Passing in the Lisp Machine" AI Memo No.602 MIT Nov 1980

[3] D.Bobrow M.Stefik "The LOOPS Manual" Xerox PARC 1982

[4] M.Numao M.Ishizuka "A Frame-like Knowledge Representation System for Computer Vision" 7th ICPR Montreal 1984

[5] R.Brooks "Symbolick Reasoning Among 3-D Models and 2-D Images" Artifitial Inteligence Vol.17 No.1-3 pp285-348 1981

[6] 濱、石塚 「フレームの階層性を用いたビジョンシステム」 情報処理31回全国大会

[7] 濱、石塚 「知識型３Ｄビジョンのための物体表面モデルのフレーム型表現」 情報処理32回全国大会

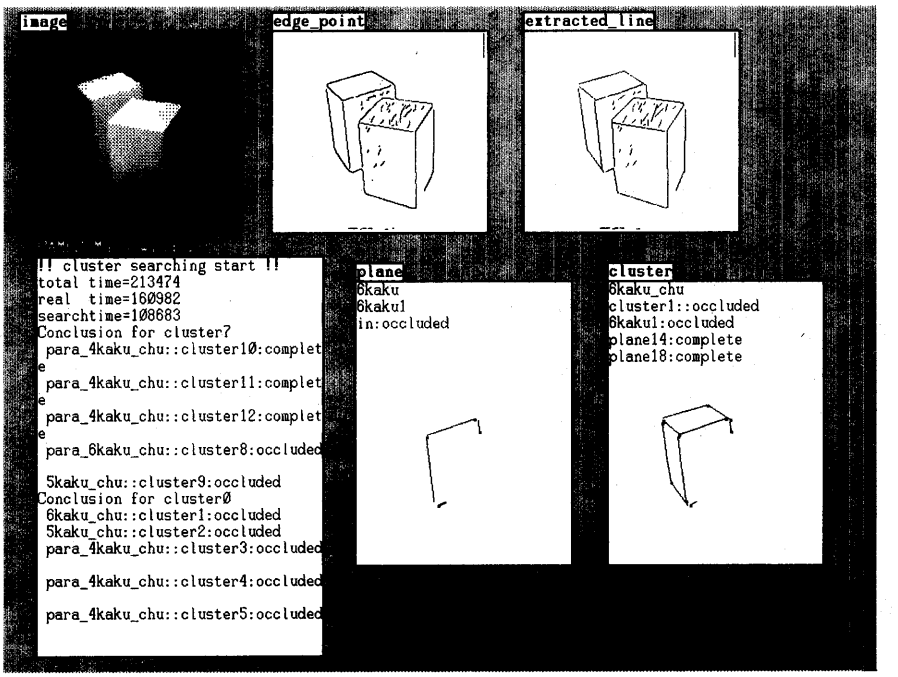

図7 認識結果の表示