オブジェクト指向分析/設計に基づいた 組立ロボットのティーチング*

藤田 充**・岩間 憲三**

A Robot-Teaching Method for Assembling Parts
by Using Object-Oriented Analysis and Design*

Mitsuru FUJITA** and Kenzo IWAMA**

In assembly tasks, contact relationships between parts play an important role in cancellation of uncertainties. This paper describes a method of robot-teaching for assembly, especially mating of parts, by using object-oriented analysis and design. It explicitly uses an assembly strategy which is based on the state transition of contact relationships. The first part of this paper models the assembly strategy, where the model consists of 4 components; state, transitional-task, verifier and supervisor. The second part introduces object oriented analysis and a design technique for easy implementation of the model. The last part explains how to program and teach mating tasks of polyhedral blocks in a real system. Our method has several advantages: (1) Uncertainties on positions and orientations can be canceled when a robot plays back to instructed motions. (2) Modifications of source code in robot software can be localized in case of the addition of new mating functions because of object-oriented analysis and design of the software.

1. はじめに

ロボットによる組立作業では、対象部品を相手部品に対して高い精度で位置合わせしなければならない。もし十分な精度が実現できない場合には、目標作業を達成することができないばかりでなく、部品や装置に重大な損傷を及ぼす恐れもある。現実には、ロボットアームの位置決め誤差、部品供給や把握状態のばらつき、部品の寸法公差や変形など、位置に関して様々な不確定条件が含まれており、これに対処するための方法が必要となる。

位置に関する不確定さを取り除く有効な手段として、部品間の接触情報を利用する方法がある。これは、対象部品と相手部品の接触を参照状態とし、その状態から位置や姿勢を相対的に変化させることで、実質的な不確定さを吸収しようとするものである。起こり得る接触状態

を分類し、接触状態間の遷移動作を実現すれば、様々な接触状態を経由するような一般の組み付け作業にも適用可能であると考えられる。このため、部品間の接触状態遷移を利用した組立作業戦略について、これまで多くの研究がなされている^{1)~7)}。

しかし、接触状態を利用する組立については、教示方法の開発や、教示/再生を一連のシステムに組み上げていく時のシステムの柔軟性など、研究すべき多くの課題がある。とくに、組立作業では、部品どうしを接触させた状態で位置や姿勢を相対的に変化させなければならないため、力情報など位置以外の情報も教示する必要がある。そして、組立作業戦略が接触状態遷移の形で与えられた場合に対応した効率的な教示方法の開発が不可欠である。

接触を伴う作業の教示方法として、これまでにもいくつかの方法が提案されている。ロボットプログラム言語AL[®]では、力制御用のステートメントが用意されており、並進と回転で合計六つの力成分を指定することができる。しかしながら、プログラマが直観で力を指定しなければならないので、高度な作業に対するプログラミングは困難である。浅田らは、直接教示によって計測した

* 原稿受付 1994年4月22日

** (財)京都高度技術研究所 Advanced Software & Mechatronics Research Institute of Kyoto; Kyoto Research Park, 17 Chudoji Minamimachi, Shimogyo ward, Kyoto city, Kyoto 600, JAPAN

Key Words: robot-teaching, object-oriented analysis and design, assembly strategy, contact state, transitional task.

エンドエフェクタの変位と与えられた力からハイブリッド制御の制御モードを決定し、動作プログラムを生成する方式を開発した⁹⁾。この方法では、作業者の動作を直接計測することによって、位置と力の情報を獲得するため、力の調整を含むような高度な作業の教示も可能であるが、その反面、教示するオペレータの負担は大きい。また末広らは、多角形平面内組立作業における接触状態の解析から、組立作業を実現するためのスキルとして、「突き当て動作」、「辺合わせ動作」、「はめあい動作」の三つが重要であるとし、多面体対象物の場合について各々の動作を実現した¹⁰⁾。さらに植之原は、これら三つのスキルに基づいて物体間の接触を実現する基本動作コマンドを定義し、各基本動作を実現する制御方法および実行時の接触検知方法について検討している¹¹⁾。しかしながら、作業戦略から展開されるスキルの教示方法や教示プログラムの構成方法については論じていない。

本論文では、組立作業の表わし方として、部品間の接触状態と状態間の遷移を導入し、教示および実行の双方にこれを適用する。さらに、この表現方法をオブジェクト指向分析/設計手法に基づいて詳細化し実装する。これにより、組立作業における部品間の不確定さを除去し、かつ教示や実行プログラムの再利用性、拡張性を向上させることができる。実用的な観点からは、プラスチック成型品に代表されるような部品で、不確定性が比較的大きく、クリアランスの厳しくないような組立を想定している。本論文では、まず、部品間の接触状態遷移を利用した組立作業戦略を教示するための「組立作業教示モデル」を提案する。つぎに、組立作業教示モデルに対してオブジェクト指向の概念を導入して実現するための方法を述べる。さらに、この方法を実験システムへ適用した例を示し、その有用性について説明する。

2. 接触状態遷移を利用した組立作業教示モデル

2.1 組立作業における接触状態遷移表現の有効性

組立作業では、部品相互の距離や接触の種類によって目標動作が変化していると考えられることができる。たとえば、ロボットが対象部品を把握する場合の目標動作は、手先から対象部品までの距離が長いときには手先を対象部品に接近させることであり、対象部品までの距離が短いときには微小動作して把握させることである。また、ブロックを穴に挿入する場合においても、ブロックと穴の接触状態により、並進・回転のような微小動作の種類や、力制御・位置制御をするための作業軸や条件など、動作の特性が異なる。

このように、組立作業における目標動作は作業の過程で刻々と変化している。組立作業をロボットによって実現させるためのアプローチは、目標動作が連続的に変化していくとみなすか、あるいは離散的に変化していくとみなすかによって、二つの方法に分けることができる。

前者の代表的な例として、学習制御がある。学習制御では、複雑な作業を忠実に再現できる反面、教示方法としては普通は直接教示が用いられるため、教示者の負担が大きい。また、一般には学習結果から特性を抽出できないので、教示後に微調整したり過去の教示データを同種の異なった作業へ再利用することが困難である。

一方、目標動作が離散的に変化していくと考えるアプローチとして、接触状態遷移による方法がある。

Fig. 1 に示すように、作業過程で発生する部品間の接触状態の関係に着目すると、各々は一つの作業状態として考えることができる。各接触状態に固有の遷移動作を割り付けて接触状態の遷移を計画に基づいて実現することができれば、目標作業を達成することが可能である。この方法は、部品間の接触状態とそれに割り付けられる遷移動作によって組立作業を表現するので、作業戦略を記述するのに適している。また、状態変化を確認しながら作業を実行していくため、作業中に生じる位置の不確定さを吸収することができる。さらに、同種の遷移動作については、教示データを再利用できる場合もある。反面、学習制御のようにきめの細かい教示は困難であると考えられるが、この場合には細かな動作が要求される遷移動作のみ部分的に学習制御を用いることも可能である。

こうしたことから筆者らは、ロボットの組立作業を表現する方法として、接触状態遷移による方法が有効であると考えた。2.2 では、部品間の接触状態遷移を利用し

Fig. 1 Assembly strategy by using contact states between two parts

た組立作業戦略を教示するための「組立作業教示モデル」を提案する。

2.2 組立作業教示モデル

組立作業手順が部品間の接触状態遷移を利用した作業戦略の形で与えられる場合における組立作業教示モデルを Fig. 2 に示す。組立作業教示モデルには、次の四つの構成要素がある。ここで、接触状態は、すでに発表されている表わし方と、考え方において同じである（例えば参考文献 6), 7)）。

state:

- 部品間の接触状態を表現する。

transitional-task:

- 接触状態に割り付けられる遷移動作を表現する。
- ロボットコントローラとのインターフェイスにより、動作の教示/再生を行う。

verifier:

- 現在の部品間の接触状態を検証する。接触状態が望んでいる状態でない時には、失敗とする。

supervisor:

- ユーザインターフェイスを介して、ユーザからの要求を受け付ける。
- 組立作業戦略を解釈する。
- 上記三種類の構成要素の生成を含めシステム全体の管理を行う。

組立作業教示モデルのインスタンスは、次のように生

成されて利用される。

(1) 作業戦略の教示

ユーザは、部品間の接触状態を利用した作業戦略を **supervisor** に与える。ここで作業戦略には、作業過程で発生する部品間の接触状態と接触状態間を結ぶ遷移動作に関する情報が含まれているものとする。**supervisor** では、与えられた作業戦略を解釈し、各々の接触状態に対して **state** を、接触状態間を結ぶ遷移動作に対して **transitional-task** を生成する。

(2) 遷移動作に対する教示

(1) によって組立の作業戦略が教示された時点では、生成された **transitional-task** は動作に必要な教示情報をもっていない。ユーザが **supervisor** に対して教示開始を要求すると、**supervisor** は生成した **transitional-task** に対して順に教示指令メッセージを送信する。**transitional-task** では、内部の教示用プログラムを起動することにより、作業実行時に必要な教示情報を獲得・蓄積する。なお、ロボットに対する教示方法は、教示時に実際のロボットや作業環境を利用するオンライン教示と、それらを計算機システム内のモデルで実現するオフライン教示に大別されるが、遷移動作の教示にどちらを採用するかは、**transitional-task** 内の教示プログラムに依存することになる。ここで、実際に実装したシステムでは、教示情報としてハンド先端部を移動させるための並進方向ベクトルおよび回転方向ベクトルを採用した。

Fig. 2 Teaching model for assembly operations

(3) 組立作業の再生

上記 (1), (2) による教示後, ユーザは **supervisor** に対して作業の再生を要求する. **supervisor** は, 接触状態を作業戦略にしたがって遷移させるため, 現在状態に相当する **transitional-task** に対して, 再生指令メッセージを送信する. **transitional-task** では, 内部の再生用プログラムが起動することにより, すでに蓄積されている教示情報を利用して, 教示結果を再生する. 一つの **transitional-task** が終了するごとに, **supervisor** では, 計画どおりの接触状態であるかを **verifier** に問い合わせることによって確認する. 異常がなければ, 次の接触状態へ遷移させるため, 上記と同様に次の **transitional-task** に対して, 再生指令メッセージを送信する. 異常があった場合には, **verifier** は, 現在の接触状態がどの **state** に対応しているかを調べるが, 判定が不可能な場合にはエラーとして処理される.

3. オブジェクト指向プログラミングによる組立作業教示モデルの実現

3.1 オブジェクト指向プログラミングの導入

2. で提案した組立作業教示モデルの構成要素のうち, **state** と **transitional-task** は, 与えられた作業戦略に含まれる接触状態の数だけ生成される. そこで, **state** を要素とする集合を C_{state} , **transitional-task** を要素とする集合を C_{trans} として, これらの集合の特性を考えてみる.

C_{state} の特性は, 部品間の接触状態をシステム内でユニークに表現することである. この特性は, 扱う部品の形状や材質などの範囲から接触状態を表現するためのデータ構造を定めて, これに必要な操作を付け加えることによって実現できる. C_{state} の各要素の特性は, このデータ構造にセットされた具体値によって決まる.

一方, 集合 C_{trans} の特性は, 作業戦略における遷移動作をユニークに表現することである. この特性は, 遷移動作が利用する情報のデータ構造を定めて, これに教示/再生時に必要な操作を付け加えることによって実現できる. ところが, これらのデータ構造や操作の詳細は, 遷移動作の種類によって異なる場合もある. たとえば, 部品間の接触状態遷移として, 非接触状態から接触状態へ遷移させる場合と, ある接触状態から別の接触状態へ遷移させる場合とでは, 遷移動作におけるロボットアームの制御方法や力覚のセンシング目的は違ったものになる. 前者の場合は非接触状態での動作なので位置制御のみが用いられるが, 後者の場合には接触状態を維持した動作となるので, 力制御あるいは力と位置のハイブリッド制御が用いられる. したがって, C_{trans} は, 遷移動作の種類

によって, さらにいくつかの部分集合に分類され, 各要素の特性は, それが属する部分集合のデータ構造にセットされた具体値によって決まる.

さて, 集合の特性を実現する手段として, オブジェクト指向分析/設計がある¹²⁾. オブジェクト指向分析/設計は, システムが取り扱う対象(オブジェクト)を, 構造化された抽象データ型の集合として実現しながらソフトウェアシステムを構築する手法である. ここで, 構造化された抽象データ型の集合をクラス, 集合に属する個々の要素をインスタンスと呼んでいる. この手法では, システムはデータ構造に基づいてモジュール化されるため, ソフトウェアの再利用性や拡張性を著しく向上させることができる.

オブジェクト指向分析/設計に基づいたプログラミング作業は, オブジェクト指向プログラミングと呼ばれる. オブジェクト指向プログラミングでは, パッケージ(データ構造とそのデータ構造に対する操作をひとまとめたもの)によるカプセル化機能や継承(すでに定義されたクラスを拡張または制限したものとして新しいクラスを宣言すること)機能によって, 上記で述べた構築手法が実現可能となる.

ロボットによる組立作業は未だ研究途上にあり, 近い将来, より優れた作業戦略や制御則が開発されるものと予想される. したがって組立ロボットのソフトウェアは, これらの新しい技術が開発された時に, 容易に移行できるものである必要がある.

筆者らは, 組立作業教示モデルを効率的にシステムへと適用するには, 現時点のソフトウェア技術では, オブジェクト指向プログラミングのもつソフトウェアの再利用性や拡張性といった特徴が最良であると考えた. そこで, 組立作業記述モデルにオブジェクト指向の概念を導入して, その実現方法の検討を行った.

3.2 組立作業記述モデルの実現方法

2.2 で述べた組立作業記述モデルにオブジェクト指向の概念を導入して, 教示システムを実現する方法を述べる. なお, ここでの議論は次の前提条件に基づいている.

- 対象部品と相手部品は多面体で剛体である. ただし, 実行時に力の加わり方によって変形する場合も, 接触状態の分類が維持できる(つまり, 稜線が変形によって面にならない)限りにおいて, この前提に入れる. さらに, 曲面から成る部品についても, 接触状態が離散的に記述できる限りにおいては, この前提に入れる.
- ロボットアームの自由度は6(位置:3, 姿勢:3)である.
- ロボットアーム手首部分には6軸の力覚センサが取

り付けられており、対象部品が受ける力を観測できる。

3.2.1 オブジェクトモデリング

オブジェクト指向分析/設計では、対象とするシステムを分析した後に、詳細な設計が行われていく。分析の段階で、クラスとクラス間の関連を抽出・整理する作業は、特にオブジェクトモデリングと呼ばれる。

2.2で述べた組立作業教示モデルに対して Rumbaugh の記法¹³⁾を用いて、オブジェクトモデリングした結果を Fig. 3 に示す。クラスの特長 (feature) は、インスタンスのデータ項目として機能する属性 (attribute) と、インスタンスの様々な振舞いを実現するための操作項目として機能するメソッド (method) によって決定される。クラス間の関連 (association) は、クラスとクラスを実線で結ぶことによって示す。また関連を表わす実線の端にある黒丸は、一方のクラスに属する一つのインスタンスに対して、もう一方のクラスに属する複数のインスタンスが結びついていることを意味している。図において、各クラスは組立作業教示モデルの構成要素に対応させている。インスタンスに着目すると、組立作業戦略は複数の接触状態と遷移動作で表現されるので、**supervisor** インスタンスは、**state** インスタンスと **transitional-task** インスタンスに対して、1:多で関連付けられる。遷移動作は接触状態に割り付けられた形で与えられるので、**state** インスタンスと **transitional-task** インスタ

ンスの関連は 1:1 である。

クラス **state** は、属性として部品間の接触状態を表わすデータ構造 `state_of_contact` を持っている。メソッドには、インスタンスから `state_of_contact` の値を取り出すための `get` と、`state_of_contact` の値を設定する `set` がある。

クラス **transitional-task** は、属性として遷移動作の教示情報のデータ構造 `teaching_data` を持っている。メソッドには、このインスタンスに対して教示動作を実行する `teach` と、教示結果に基づいて動作を再生する `perform` がある。

クラス **verifier** は、特に属性を持っていない。メソッドには、現在状態が指定の状態であるかどうかを検証するための `verify_current_state` と、現在状態を識別するための `find_current_state` がある。

クラス **supervisor** は、属性として作業戦略 `strategy` と現在状態 `current_state` を持っている。メソッドは、組立作業教示モデルのライフサイクルに対応しており、組立作業戦略の教示を行うための `set_strategy`、戦略における遷移動作に対して教示を行うための `teach`、再生を行うための `perform` がある。

3.2.2 クラス state の実現方法

クラス **state** の属性である `state_of_contact` は、部品間の接触状態を意味しており、プログラミングの段階では、適切なデータ構造によって定義されなければなら

ない。そこで部品間の接触状態を表現するための方法について考える。いま、扱う部品は多面体で、かつ剛体であるという仮定から、安定的な接触状態における接触箇所は、多面体の構成要素である面 (Face)、稜線 (Edge)、頂点 (Vertex) のいずれかである。接触状態は次のように表現する。

【接触状態の表現法】

対象部品 a の構成要素の全体集合 A を

$$A = \{A_F \cup A_E \cup A_V\} \quad (1)$$

$$= \{a_1, a_2, \dots, a_l\} \quad (2)$$

とする。ここで、 A_F 、 A_E 、 A_V はそれぞれ、a の面の集合、a の稜線の集合、a の頂点の集合である。同様に相手部品 b の構成要素の全体集合 B を

$$B = \{B_F \cup B_E \cup B_V\} \quad (3)$$

$$= \{b_1, b_2, \dots, b_m\} \quad (4)$$

とする。ここで、 B_F 、 B_E 、 B_V はそれぞれ

Fig. 3 Object modeling of the teaching model for assembly operations

れ, b の面の集合, b の稜線の集合, b の頂点の集合である。

いま, a と b が n 箇所 ($n \geq 0$) で接触しているものとする。ここで, k ($0 < k \leq n$) 番目の接触 c_k が, A の i ($0 < i \leq l$) 番目の構成要素と B の j ($0 < j \leq m$) 番目の構成要素による接触である場合, この接触関係 c_k を

$$c_k = a_i \cdot b_j \quad (5)$$

で表わす。接触状態 C は, n 箇所における接触関係をリスト構造にして, 次のように表現する。

$$C = (c_1, c_2, \dots, c_n) \quad (6)$$

上記の表現法により, 例えば Fig. 4 で (b), (c) の接触状態はそれぞれ, $(a_2 \cdot b_9)$, $(a_3 \cdot b_8, a_1 \cdot b_2)$ で表わすことができる。また, 非接触の状態はヌルリストすなわち, $C = ()$ となる。

Fig. 4 Examples of contact expression

3.2.3 クラス transitional-task の実現方法

クラス **transitional-task** を実現するためには, 属性 `teaching_data` に教示情報を蓄積するための操作を `teach` メソッドとして定義し, `teaching_data` を利用して遷移動作を再生するための操作を `perform` メソッドとして定義する。ところが, 3.1 で述べたように, 遷移動作の詳細は, 動作の種類によって異なる場合もある。

そこでクラス **transitional-task** をさらに部分集合に分類して特性を詳細化する。この場合, 上位のクラス **transitional-task** は, 単に特性を記述する上で最低限必要な属性と共通化されたメソッドインターフェイス

だけを持つ「抽象クラス」として機能する。オブジェクト指向分析/設計では, この抽象化された上位のクラスをスーパークラス, 詳細化する下位のクラスをサブクラスと呼んでいる。サブクラスはスーパークラスの特性を継承しているので, サブクラスの構築はサブクラスに固有の部分だけを追加すればよく, 拡張性がある。

つぎに, クラス **transitional-task** のサブクラスを検討するために, 接触状態に割り付けられる遷移動作を分類する。分類の観点は, 相手部品からの干渉の種類と, 移動の種類である。

まず, 相手部品からの干渉に着目すると, 次の四つに分類できる。

(1) 自由空間動作

自由空間における動作。

(2) 接触維持動作

接触を維持した状態での動作。

(3) 接触達成動作

自由空間から接触状態へ突入する動作。

(4) 接触離脱動作

接触状態から自由空間へ離脱する動作。

つぎに, 移動の種類に着目して次の三つに分類した。

(1) 並進動作

相手部品に対する姿勢関係を一定に保って並進する動作。

(2) 回転動作

相手部品に対する位置関係を一定に保って回転する動作。

(3) 並進回転動作

相手部品に対する位置, 姿勢関係を変える動作。

ここで, 回転と並進回転の区別は, 部品の間の接触点に着目して行う。

上記の分類にしたがって定義するサブクラス名を Table 1 に示す。

各サブクラスの定義は, オブジェクト指向言語を用いて次の手順で行う。

Table 1 Derived transitional-task classes

	translation	rotation	translation and rotation
motion in free space	translate	rotate	move
motion for contact	translate-until-contact	rotate-until-contact	move-until-contact
motion for keeping contact	translate-with-contact	rotate-with-contact	move-with-contact
motion of breaking contact	translate-from-contact	rotate-from-contact	move-from-contact

- 手順1 クラス **transitional-task** をオブジェクト指向言語の機能を用いて継承する。
- 手順2 サブクラスが必要とする教示情報のデータ構造体を属性として追加定義する。
- 手順3 `teaching_data` に教示情報を蓄積するための `teach` メソッドと、`teaching_data` に蓄積された教示情報にしたがって遷移動作を再現するための `perform` メソッドをプログラマルーチンとして記述する。

3.3 クラス **verifier** の実現方法

クラス **verifier** では、組立作業の再生時に接触状態の問い合わせに対して応答する機能を実現するため、次の二つのメソッドを定義する。

(1) `verify_current_state`

(2) `find_current_state`

ある遷移動作を完了した後、どの接触状態に遷移するかについては、与えられた作業戦略情報に含まれている。したがって、通常は(1)の問い合わせによって、組立作業が教示内容どおりに進行していることが確認できれば

よい。しかし再生時に何らかの異常が発生して現在状態がわからなくなった場合には、(2)の問い合わせを用いて現在状態を判定しなければならない。一般には、(1)と比べて(2)の処理の方が困難であり、もし現在状態をユニークに識別することができなかった場合には、組立作業の再生は「失敗」として処理される。

なお、多面体部品間の接触状態を識別するためには、力覚や視覚などをアクティブにセンシングする技術と、そのセンシングデータから状態を判定するための技術が要求される。これらの技術に関しては、筆者らによるものを含めて、これまでいくつかの研究がなされている^{14),15)}。

3.4 クラス **supervisor** の実現方法

クラス **supervisor** では、組立作業教示モデルのライフサイクルに対応したメソッドを実現する。

(1) `set-strategy` メソッドの処理

`set-strategy` メソッドがコールされた時の処理フローを Fig.5 (a) に示す。図に示すように、**supervisor** では、与えられた組立作業戦略を解釈した後、作業戦略

Fig. 5 Flow chart for implementation of supervisor

に対応して、**state** インスタンスと **transitional-task** インスタンスを生成する。さらに、**state** インスタンスと **transitional-task** インスタンスを双方向に検索できるように **strategy** のテーブルを作成する。

(2) teach メソッドの処理

teach メソッドがコールされた時の処理フローを Fig. 5 (b) に示す。図に示すように、**supervisor** では、まず、スタートの接触状態を表現する **state** インスタンスを属性 **current_state** の値としてセットする。つぎに、属性 **strategy** のテーブルから、対応する **transitional-task** インスタンスを検索し、そのインスタンスに対して **teach** メッセージを送信する。メッセージを受信した **transitional-task** インスタンスでは、教示実行プログラムが起動される。**supervisor** では、教示終了信号が到着するまで待機する。信号が到着すると属性 **current_state** の値として次の **state** インスタンスをセットし、そのインスタンスに対して **teach** メッセージを送信する。この後、教示すべき遷移動作がなくなるまで、同様な処理が行われる。

(3) perform メソッドの処理

perform メソッドがコールされた時の処理フローを Fig. 5 (c) に示す。**teach** メソッドの場合と同様に、**supervisor** では、まず、スタートの接触状態を表現する **state** インスタンスを属性 **current_state** の値としてセットする。つぎに、属性 **strategy** のテーブルから、対応する **transitional-task** インスタンスを検索し、そのインスタンスに対して **perform** メッセージを送信する。メッセージを受信した **transitional-task** インスタンスでは、再生実行プログラムが起動される。**supervisor** では、再生実行終了信号が到着するまで待機する。信号が到着すると、現在の接触状態を検証するため、**verifier** インスタンスに対して **verify_current_state** メッセージを送信する。検証の結果、異常があれば、**verifier** インスタンスに対して今度は **recognize_current_state** メッセージを送信し、現在の接触状態を判定する。検証に異常がなければ、属性 **current_state**

state の値として次の **state** インスタンスをセットし、そのインスタンスに対して **perform** メッセージを送信する。この後、接触状態が目標状態になるまで、同様な処理が行われる。

4. システムの実装

これまで述べた方法にしたがって、多面体ブロック組み付け作業を目的とした教示システムを実装中である。ここでは、その教示システムの構成と実装方法について説明する。

4.1 システム構成

実験システムの構成図を Fig. 6 に示す。ロボットアームは、ダイキン工業 (株) 製のダイレクトドライブ 6 軸垂直多関節型 R1680 である。アームの手首部分には、接触情報を得るために、日立建機 (株) 製の 6 軸力覚センサ LSA 6010 A を取り付けている。コントローラは、VME bus システムで構成されており、CPU ボードには MVME-167 A (CPU: 68040) を用いている。また、開発環境および GUI (Graphical User Interface) を実現するため、SUN ワークステーションをホストにしている。ホストとコントローラは、Ethernet によりネットワーク接続されており、互いに通信することが可能である。システムの実装は、コントローラ側とホスト側で

Fig. 6 System configuration

切り分けて行う。

コントローラ側の OS には、リアルタイム OS である VxWorks 5.0 を用いている。VxWork の NFS (Network File System), RPC (Remote Procedure Call) などの機能や標準ライブラリを用いて、ロボットアームの制御に必要なライブラリを構築し、その上で **transitional-task** や **verifier** を実現するのに必要な機能を C 言語の関数として実装している。

一方ホスト側では、UNIX の上でオブジェクト指向言語を用いて **state**, **transitional-task**, **verifier**, **supervisor** の各クラスを定義する。また、ユーザインターフェイスを提供するため、X-window システムによる GUI の実装を行っている。

state, **transitional-task**, **verifier**, **supervisor** の各インスタンスは、実行時にホスト側で生成される。**transitional-task** インスタンスと **verifier** インスタンスのメソッドは、ホスト側から、コントローラ側の対応する関数を RPC することによって実現される。

以上のシステムに対して、これまでに、コントローラ側の **transitional-task** とホスト側の GUI の実装を行った。

4.2 Transitional-task の実装

ここでは、実装したコントローラ側の **transitional-task** のうち、**translate-until-contact** クラスと **rotate-with-contact** について、**teach** メソッドおよび **perform** メソッドによる教示・再生方法を説明する。なお、ここで述べる位置および姿勢は、ロボットの基準座標系からみたハンド座標系の原点位置のことを指している。

4.2.1 Translate-until-contact の実装

• **teach** メソッドの実行手順

手順1 システムは、現在の位置 P_c を読み取る。

手順2 ユーザは、ペンダントを操作して並進軌道上の経由点 P_o へと移動させる。

手順3 システムは、 P_c と P_o から、並進動作の方向ベクトル K_p を計算し、これを教示情報として蓄積する。

• **perform** メソッドの実行手順

手順1 現在の位置から、姿勢を一定に保った状態で、 K_p の方向へ一定速度で移動させる。

手順2 このとき、力覚センサの値をモニタしながら、6 軸成分のいずれかがあらかじめ定めておいたしきい値を越えた時点で接触したとみなして停止させる。

4.2.2 Rotate-with-contact の実装

• **teach** メソッドの実行手順

手順1 システムは、現在の姿勢 R_c を読み取る。

手順2 ユーザは、ペンダントを操作して回転軌道上の経由点 R_o へと移動させる。

手順3 システムは R_c と R_o から、等価回転軸 K_r を計算し、これを教示情報として蓄積する。ここで等価回転軸とは、姿勢 R_c を回転軸 K_r のまわりに θ だけ回転させることによって R_o になるような K_r のことである。

• **perform** メソッドの実行手順

手順1 現在の姿勢から、 K_r を等価回転軸として、一定速度で回転させる。

手順2 このとき、力覚センサの値から、姿勢による重力の影響を除去して相手部品から受ける抗力を観測し、これが一定となるように位置を抗力方向に補正する。

手順3 力覚センサの値をモニタしながら、モーメント成分のいずれかがあらかじめ定めておいたしきい値を越えた時点で、状態変化とみなして停止させる。

4.3 ソフトウェアの変更・追加に対する考察

本システムにおいて、ソフトウェアを部分的に変更する場合と追加する場合の方法を、**transitional-task** を例として考える。

(1) 属性はそのまま、メソッドのみを変更する場合
コントローラ側で対応する **transitional-task** の関数を変更する。

(2) 属性とメソッドのみを変更する場合
ホスト側で該当クラスの属性に関する定義を変更し、コントローラ側で対応する **transitional-task** の関数を変更する。

(3) 新しい **transitional-task** を追加する場合
ホスト側で新しくクラス定義を行い、コントローラ側で対応する **transitional-task** の関数を追加する。既にある **transitional-task** クラスを継承できる場合には、変更量は少なく済む。

今回の実装では、コントローラ側にはオブジェクト指向言語を用いてないが、分析/設計段階でオブジェクト指向を導入したため、ソフトウェアの変更・追加をかなり局所化することができたといえる。

5. おわりに

本論文では、組立ロボットに対する新しい教示方法として、作業戦略が部品間の接触状態遷移の形で与えられた場合における組立作業教示モデルを提案した。さらに、このモデルをオブジェクト指向分析/設計手法を用いて実現する方法について述べた。

この教示方法の特長としては、次のことがいえる。

- (1) 部品間の接触状態遷移を利用した作業戦略を実現するための教示手段を提供している。
- (2) 作業中の位置に関する不確定さに対処できる。
- (3) オブジェクト指向プログラミングで実現しているため、教示や実行プログラムの変更を局所化することができる。

今後は、実験システムに対する実装を進めて、詳細な評価を行っていく予定である。

参 考 文 献

- 1) T. Lozano-Perez, M. Mason and R. Taylor : Automatic Synthesis of Fine-Motion Strategies for Robots ; The International Journal of Robotics Research, Vol. 3, No. 1, pp. 3~24 (1984)
- 2) A. Giraud and D. Sidobre : Contact Manipulations and Geometric Reasoning ; Geometry and Robotics, Lecture Notes in Computer Science, Springer-Verlag, pp. 361~376 (1988)
- 3) 平井, 浅田, 得丸 : 組立作業における物体の接触状態遷移に関する運動学的生成とそのネットワーク表現の自動生成 ; 計測自動制御学会論文誌, Vol. 24, No. 4, pp. 406~413 (1988)
- 4) J. B. McCarragher and H. Asada : A Discrete Event Approach to the Control of Robotic Assembly Tasks ; Proc. 1993 IEEE Int. Conf. on Robotics and Automation, pp. 331~336 (1993)
- 5) 吉川, 横小路, 余 : 拘束状態遷移難度に基づくロボットによる組立作業の計画法 ; システム制御情報学会論文誌, Vol. 5, No. 7, pp. 283~293 (1992)
- 6) S. R. Desai and A. R. Volz : Identification and Verification of Termination Conditions in Fine Motion in Presence of Sensor Errors and Geometric Uncertainties ; Proc. 1989 IEEE Int. Conf. on Robotics and Automation, pp. 800~807 (1989)
- 7) S. Hirai and K. Iwata : Recognition of Contact State Based on Geometric Model ; Proc. 1992 IEEE Int. Conf. on Robotics and Automation, pp. 1507~1512 (1992)
- 8) J. Craig (三浦, 下山 (訳)) : ロボティクス, 共立出版 (1991)
- 9) 浅田, 出海 : 作業者の動作計測によるハイブリッド制御のための作業教示とプログラム生成 ; 日本ロボット学会誌, Vol. 5, No. 6, pp. 452~458 (1987)
- 10) 末広, 高瀬 : スキルに基づくマニピュレーションシステム ; 日本ロボット学会誌, Vol. 8, No. 5, pp. 551~562 (1990)
- 11) 植之原 : 接触状態の認識と実現方法に関する研究 ; 第10回日本ロボット学会学術講演会予稿集, pp. 1005~1008 (1992)
- 12) B. Meyer (仁木 (監訳)) : オブジェクト指向入門, アスキー出版局 (1990)
- 13) J. Rumbaugh et al. : Object-Oriented Modeling and Design, Prentice-Hall (1991)
- 14) 藤田, 岩間, 宮口 : 力覚センサを利用した対象物と環境の接触状態推定方法 ; 第10回日本ロボット学会学術講演会予稿集, pp. 1027~1030 (1992)
- 15) 井手, 宮崎, 安 : 力学情報を用いたはめあい作業の学習 ; 日本ロボット学会誌, Vol. 10, No. 1, pp. 118~128 (1992)