解 説

ロボット言語と制御ソフトウェア

Robot Languages and Robot Control Software

三宅 徳久* (株)日立製作所 機械研究所

Norihiisa Miyake* *Mechanical Engineering Research Laboratory, Hitachi Ltd.

1. はじめに

ロボットコントローラのソフトウェアは、コントローラの内部と外部のソフトウェアに大別される。すなわち、ロボットの動作を直接制御するためのシステムソフトウェアに相当するものと、ユーザがロボットの動作を記述するためのいわゆるロボット言語に相当するソフトウェアである。この両者は、それぞれ制御ソフトウェアとヒューマン・インターフェースとも言い替えることができる。

ロボットコントローラに代表される FA システム用コントローラの世界では、オープン化の動きが近年活発化してきている。オープン化の目的は、多くのメーカが共通した仕様のハード、ソフトを供給することで、ユーザに低コストでのシステム構築を可能とすると同時に、ユーザがコントローラ内部に手を加え、用途に合ったきめ細かな対応を可能とすることである。このためには、制御ソフトウェアのさまざまなレベルでヒューマン・インターフェースとしての整備が必要であり、従来のロボット言語とは異なったレベルでのユーザプログラミングを意識した関数化、コマンド化が重要となる。この意味において、コントローラ内部、外部のソフトウェアという分類は変化しつつあると言える。本稿では、以上を念頭に置きつつロボットコントローラのソフトウェア全般について概説する。

2. ロボットコントローラのソフトウェア

ロボットに作業を行わせるためには、作業工具の動作経路を制御するだけでなく、動作順序の制御、さらに周辺機器の監視や制御、あるいはその状態に依存した動作内容の選択制御が必要であり、場合によっては作業条件の制御なども必要となる。したがってコントローラには、これらの機能を実現するための制御ソフトウェアが搭載される。

また、ユーザの所望する作業内容に応じてこれらの制御

ソフトウェアに適切なパラメータを設定し、起動の順序や条件を指定することが必要であり、このためのヒューマン・インターフェースとして、いわゆるロボット言語が必要となる。コントローラ上にはロボット言語を解釈実行するためのインタプリタやコンパイラ、あるいはトランスレータなどが搭載され、言語を用いてユーザが作成したプログラム（作業ないしはタスクプログラムと呼ばれる）に基づいてロボットの運動制御が実行される。なお、通常はロボット言語の表面的なコマンドをトランスレータにより内部コードに変換して記憶し、作業実行時にはこの内部コードをインタプリタにより解釈、実行するものが多い。

3. ロボット言語

3.1 SLIM と STROLIC

ロボット言語とはロボットの動作内容、順序を記述するための言語であり、コンピュータのプログラミング言語と同様、標準化が望ましい。しかし、ロボットの機能やコントローラ内部のデータ構造などがメーカごとに異なるため、従来は言語の仕様も各社各様であった。これはロボットの普及を疎外する一つの要因でもあることから、統一された仕様のロボット言語を定めることが試みられて来た。

我が国では(社)日本ロボット工業会が中心となって標準ロボット言語の設計が行われ、組立ロボットを主対象に SLIM (Standard Language for Industrial Manipulators) と呼ばれる言語が 1992 年に、STROLIC (STandard RObot Language in Intermediate Code) と呼ばれる中間コードが 1995 年にそれぞれ JIS に制定された[1]~[3]。

SLIM は、計算機言語になじみの少ない現場技術者を意識するとともに、産業用ロボットに用いられている既存の言語との親和性などに留意し、BASIC 風の文法体系を持つ形で規格化された。このため、並列動作の記述能力や構造化などの点で限界もあるが、利用が容易であり規格化の意義は大きい。ただしデータの単位定義などが個々のコントローラに委ねられているため、この言語をサポートするコントローラ間でもそのままでは共通利用ができない点、

原稿受付 1996 年 8 月 1 日

キーワード: Robot Language, Control Software, Robot Controller, Motion Control, Software Architecture

*〒300 土浦市神立町 502

*Kandatsu-chou, Tsuchiura-shi, Ibaraki

注意が必要である。SLIM がユーザ記述言語であるのに対して、STROLIC はロボットの制御実行システムであるコントローラへの入力コードを規格化したもので、SLIM のオブジェクトコードに相当するものである。

なお、産業界で標準言語の普及が加速されないことの責めを言語仕様に負わせる意見もあるが、むしろ言語によるプログラミングと現場でのティーチングとの間のレベルの隔たりに問題があると思われる。言語を意識せずに使えるメニュー方式やアイコン方式による入力を、ティーチングと融合した形で提供することで使い勝手の向上が可能であり、このような GUI (Graphical User Interface) 層の工夫により言語の標準化を加速できるものと考えられる。

3.2 標準化に関する国際的活動[4]～[6]

ISO におけるロボット言語標準化の活動は、中間コード ICR (Intermediate Code for Robots) と高級ロボット言語 PLR (Programming Language for Robots) とを対象に進められた。ICR は規格化には至らなかったものの技術報告 TR 10562 として 1995 年にまとめられたが、PLR については標準化が中断されている。一方ドイツでは Pascal 型のロボット言語 IRL を 1993 年に国内規格化 (DIN 66312) している。ISO で審議された言語 PLR はこの IRL をベースとしたものであり、標準化は技術だけでなく政治の問題でもあるようである。

なお、ISO ワーキンググループの次の審議内容には、プログラミングおよび操作の GUI がとりあげられた。最近のロボット言語では GUI が積極的に取り入れられているが、これはシステムの記述能力とエンドユーザの使い易さとを両立する方法として有効な手段である。

3.3 今後の動向

ロボット言語の研究課題としては、上述のように GUI の取り入れ方、使い勝手と記述能力との両立などがあげられる。また近年のロボットコントローラは、ロボット単体だけではなく周辺機器を含めた FA セルレベルの統括制御機能を備えるようになって来っており、この場合の統一性のとれたプログラミング環境は研究課題である。

図 1 は、セル制御言語 (CCL) と呼ぶユーザプログラミング言語のエディタ画面の一例を示したものである[7][8]。CCL は、システムの制御によく用いられる SFC (Sequential Function Chart) と基本的に同一構造を持つペトリネットに若干の拡張を加えたもので、それぞれセルの運転モードの遷移、各運転モードにおける機器の動作を記述する 2 つの階層から成る。エディタ画面はペトリネットをグラフィカルに編集するウィンドウ、機器の動作コマンドを編集するウィンドウ、各状態の終了条件を編集するウィンドウから構成され、メニュー形式で容易にプログラミングを行うことができる。CCL エディタで編集された CCL プログラムは、トランスレータによってシーケンス

図 1 セル制御言語 (CCL) のエディタ画面例

制御言語のプログラムとロボット言語のプログラムに自動的に分離、変換される。この言語のターゲット・ハードウェアとしては、シーケンス制御言語は入出力制御プロセッサ、ロボット言語はメインプロセッサでそれぞれ相互に同期をとりながら並列に処理されるマルチ・プロセッサ構成を想定している。

なお、以上では動作レベルと呼ばれるロボット言語のみについて述べたが、作業レベル言語などについても種々の研究が行われていることは周知の通りである[2]。各言語の詳細については説明を省略するが、作業レベルの言語と GUI を指向したものとして、イメージ・ベースド・オペレーションと呼ぶ方式が提案されている[9]。これは、作業や動作の内容、環境モデルを階層的に定義するとともに、視覚センサを用いて、現実世界の状況を環境モデルに実時間で反映することによって作業のメンタルなイメージのレベルでロボットに指示を与えることを可能としたもので、この分野の研究では比較的初期のものである。

4. ロボット制御ソフトウェア

4.1 全体構成

次に、コントローラ内部にインプリメントされる制御ソフトウェアについて述べる。ロボットの運転モードとしては言語の入力や位置のティーチングを行う教示モード、プレイバック制御を行う自動運転モードが中心となるが、そのほかに作業工具の ON/OFF を行わずにロボットを低速で運転し、教示動作経路などのチェックを行うテスト運転モード、手動運転モード、ファイル操作やパラメータ設定のためのユーティリティ・モードなどが用意される。

コントローラのソフトウェアとしては、これらのモードを管理する処理、各モードの実行制御処理などに加えて、マンマシン系の表示や押ボタン操作の制御処理、異常監視処理、上位計算機との通信処理などのタスクが配置される。これらのタスクは、ロボットの動作中に実時間で並列して処理されなければならない。このため、全体のシステムはリアルタイム・マルチタスク・モニタ (OS) の管理の

もとで動作する。以下では、主として制御ソフトウェアの中核となる自動運転制御の内容について説明する。

4.2 自動運転制御

自動運転制御ソフトウェアは、図2に示す機能階層を持つ。すなわち、ロボット言語あるいはこれに対応する中間コードの形で記憶された作業プログラムの各ステップをインタープリタにより解釈し、入出力の制御、ロボット運動制御命令の実行(対応する制御処理の起動)などとともに、繰り返しや条件分岐などのフロー制御を行う。

起動された入出力制御処理、ロボット運動制御処理の実行系は、それぞれ外部入出力信号の監視と制御、ロボット本体や補軸の運動制御を行う。運動制御処理は、教示等により与えられた動作点のデータに基づいてロボット手先の作業工具の動作経路を生成し、これをロボットの各ジョイントに対する指令に分解してサーボ制御系に出力する。

サーボ制御系は運動制御処理から与えられる指令に基づき各アクチュエータの操作量を求め、駆動制御を行う。

ここで、各階層のソフトウェア間はそれぞれプロトコルを設けて通信することとなる。コントローラのオープン化を念頭におけば、第1章でも述べたように各階層を明確に分離し、相互間のコマンド・インターフェースをユーザが利用可能なように解放することが必要となる。

(1) インタープリタ系

ロボットシステムの動作はロボット言語に対応した中間コードで記憶されている。これを1つずつ読み出し、解釈して必要な制御処理を起動するのがインタープリタの機能である。システム構成機器としてはロボット本体のほかに補軸、各種の入出力、作業工具などがある。作業工具にはON/OFF式のハンドのような単純なものもあるが、溶接機などのようにロボットの動作中に連続的に制御する必要のある場合には汎用入出力制御とは別に言語側でコマンドを用意し、専用の制御処理を行わなければならない。また、センサ制御を行う場合や、溶接におけるウィーピング動作などについても同様の対応が必要となる。

インタープリタ系では、このように中間コードを解釈して対応する機器の制御処理を起動するが、各機器の制御処理ソフトウェアとのインターフェースについては、関数形式などとするにより他のソフトウェア・モジュールとの独立性、着脱性を高めることが可能である。特にロボットの運動に関する制御処理については、基本動作関数の整備がオープン化にとって不可欠となる[10]。

また、インタープリタ系では機器の制御処理の起動と同時に、入力の状態に応じて次に実行する言語中間コードのステップを決定する条件分岐などの処理も司っており、この意味でプレイバック制御における動作シーケンス制御系ないしはフロー制御系とも呼ぶことができる。

今後の課題としては、複数台ロボットの非同期独立動

図2 自動運転制御ソフトウェアの機能階層

作、同期動作、協調動作などを状況に応じて指定、実行可能とするためのソフトウェア構造やアルゴリズムの開発、システム全体の運転をユーザにプログラマブルとしつつ安全性を確保するための方策の検討などがあげられる。

(2) 運動制御系

運動制御系では、与えられた教示点間を直線、円弧などで補間してロボットの手先の動作経路を生成し、次いでこれをロボット各軸ジョイントのアクチュエータ駆動指令に分解、変換する処理を行う。このジョイント分解はいわゆる座標変換と呼ばれる演算であるが、産業用ロボットでは手先の変位からジョイント変位を求める位置レベルの変換を行っているものがほとんどであり、速度もしくはトルクのレベルで変換を行うものは力制御機能を持ったロボットなど、一部に限られているようである。各アクチュエータへの駆動指令はサーボ制御系に対して出力されるが、通常の場合サーボ制御系は運動制御を司るメインのプロセッサとは別のプロセッサとして構成されることが多く、コマンド形式のインターフェースが採用される[10]。

また運動制御系のソフトウェアは、その大半がロボットの動作に直接関係するため、上位のソフトウェアから関数形式で呼び出される場合、呼び出す側のタスクとは非同期で処理されるものが多い。したがってこれらの関数の授受と実行完了とを別個にセマフォなどを用いて管理する必要がある。また、動作速度などの正確な制御のため、サーボ系に駆動指令を発行するタイミングをタイマ割り込みなどを用いて管理する方法をとることが通常である。

この分野は、軌道生成や座標変換など、研究レベルと製品の技術レベルとのギャップが比較的少ない。しかし、機構の自由度配置や用途に対する汎用性を実現するための方法など、解決すべき課題も残されている。

(3) サーボ制御系

サーボ制御系では運動制御系とはコマンド形式で指令を

授受することが一般的であり、通常位置、速度、トルクなどの制御モードの指定が可能である。指令値の出力だけでなく各軸の実位置の入力、異常監視の有無などのコマンドも準備される。また、運動制御系の論理世界とアクチュエータの実世界との間のインターフェース部でもあるため、検出器の分解能などに依存したパラメータ設定が必要であり、より構造化が望まれる部分である。現状はハードウェアコストと制御演算速度との関係で使い勝手よりも性能を優先して作られる場合が多く、データの単位統一などを含めて各社の標準化も検討の余地があろう。

4.3 リアルタイム・マルチタスク・モニタ

以上に述べたソフトウェアは、ロボットの動作中に並行して実時間で処理する必要がある。したがって、これらのソフトウェアはタスクとして配置し、リアルタイム・マルチタスク・モニタの管理下で動作させる。なお、リアルタイム OS として市販されているものも多いが、ロボットの場合に要求されるリアルタイム性は厳しいため、性能、容量などの点から専用 OS を搭載する場合もある。

図3に制御ソフトウェアの全体構成例を示す。この場合のハードウェアは、メインプロセッサ、サーボプロセッサ、および I/O プロセッサから構成されている。メインプロセッサ上にはシステム全体の管理系の下にモード制御系、手動操作系、ユーティリティ制御系などのタスクが置かれ、これらの下に自動運転系、教示系、手動実行系などが配置される。自動運転系の下には、さらにインタープリタが位置する。各タスク間は、モニタのメールアドレスなどの機能を用いて通信する。またインタープリタから動作関数の起動には、システムコールなどの手段が用いられる。

4.4 今後の課題

個々の機能に関連する課題については既に述べた。全体的に見れば、ロボットコントローラのソフトウェアのうちロボットの動作制御に直接関連するものは、ステップ数で約2割程度であり、その他はマンマシン系ソフトウェアである。マンマシン系は、運転方案やインターロックなどの関係で構造の複雑化を免れない点も多く、整理が難しいと考えられるが、コントローラのソフトウェアで大半を占めるこの部分の体系化と整備は重要な課題である。

また、タスク間の通信、同期方法やタスク優先順位などの最適化、あるいはロボットとしての機能の特徴を考慮に入れた専用 OS の開発なども今後の研究課題であろう。

5. おわりに

ロボットコントローラのソフトウェアについて、全般にわたり概説した。内部ソフトウェアから外部ソフトウェアに至る広範な内容であり、表面的な記述となった項目も多い点、ご了承ください。

図3 制御ソフトウェアの全体構成例

なお、本稿の執筆にあたり、特にロボット言語に関する内容については(株)日立製作所生産技術研究所毛利峻治部長に種々お教え頂いた。ここに厚く御礼申し上げる。

参考文献

- [1] 日本工業規格, 産業用ロボットプログラム言語 SLIM, JIS B 8439, 日本規格協会, 1992.
- [2] 新井(編): 産業用ロボット言語 “SLIM”, 日本規格協会, 1994.
- [3] 日本工業規格, 産業用ロボット中間コード STROLIC, JIS B 8440, 日本規格協会, 1995.
- [4] 松元, 新井: “ロボット言語の現状と動向”, 日本ロボット学会誌, vol. 11, no. 1, pp. 36-39, 1993.
- [5] 松元: “ISO/TC 194/SC 2/WG 4 (産業用ロボットのプログラミング方式ならびに言語) フラン克福ルト会議報告”, ロボット, no. 106, pp. 8-14, 1995.
- [6] (社)日本機械工業連合会, (社)日本ロボット工業会: 平成6年度産業用ロボット国際標準化対策事業報告書, 1995.
- [7] 小山, 三宅: “FA セル制御言語の開発”, 第13回日本ロボット学会学術講演会講演論文集, pp. 1053-1054, 1995.
- [8] 小山, 三宅: “FA システムのための教示支援ツール”, 日本機械学会ロボティクス・メカトロニクス講演会 '96 講演論文集, no. 96-2, vol. A, pp. 375-376, 1996.
- [9] T. Hamada, K. Kamejima, et al.: “Image Based Operation—A Human-Robot Interaction Architecture for Intelligent Manufacturing,” Proc. IEEE IECON '89, pp. 556-562, 1989.
- [10] N. Miyake, et al.: “Design Concept of a New Robot Controller,” Proc. 20th Int. Symp. on Industrial Robots, pp. 777-784, 1989.

三宅徳久 (Nori-hisa Miyake)

1949年1月19日生。1974年早稲田大学大学院理工学研究科修士課程修了。同年(株)日立製作所入社。日立研究所、機械研究所にて産業用ロボットを中心とする各種ロボットの研究開発に従事。現在、同社機械研究所第4部主任研究員。1984-85年スタンフォード大学客員研究員。日本機械学会、計測自動制御学会などの会員。
(日本ロボット学会正会員)