2G1-1

効率的なロボットプログラミング環境の実現に向けて

Towards Establishing an Effective Programming Environment for Robots

小林 一樹[*1] 北原 鉄朗[*1*2]
Kazuki Kobayashi Tesuro Kitahara

[*1]関西学院大学 Kwansei Gakuin University
[*2]科学技術振興機構戦略的創造研究推進事業 CrestMuse プロジェクト CrestMuse Project, CREST, JST

Software developers of robots simultaneously describe behavior rules and definitions of a sensor-based world model. They usually repeat writing a program and checking sensor values. It is complicated way for end-users to customize their robot. In this paper, we propose an effective programming environment that allows users to easily customize behavior of their robots. Our proposed method definitely distinguishes the text programming phase and the sensor-based world definition phase. It has the potential to reduce the amount of time required for robot programming.

1. はじめに

掃除ロボットや小型ヒューマノイドロボット，ペットロボットなどの家庭用ロボットが普及しはじめている．それに伴い，エンドユーザがロボットをカスタマイズするために，自らプログラミングを行う機会が増加している．このような状況に対し，エンドユーザが利用可能なロボットアプリケーション開発環境が提供されている．たとえば，Microsoft Robotics Studio や Tekkotsu[Touretzky 05] は個人での入手が容易なロボットアプリケーション開発環境である．

しかし，エンドユーザにとって，テキストプログラミングだけでなく，実際にロボットを動作させながらセンサ値を決める作業は煩雑である．たとえば,「壁を検出したとき右旋回する」という動作をロボットに実装しようとしたとき，距離センサの値がいくつになったときに壁を検出したとみなすのかは，実際にロボットを動作させつつ，そのときのセンサ値を観測した上で決定する必要がある．

本研究では，具体的なセンサ値を用いないテキストプログラミング手法を提案する．具体的なセンサ値は，テキストプログラミング後にロボットとのインタラクションを通して決定する．この手法により，ロボットの振る舞いに関する記述と外界の記述を分離できるため，効率的なプログラミングが可能になる．

2. ロボットプログラミングにおける問題

ここでは，ロボットプログラミングを 2 つのプロセスに分けて考える．ひとつは**ロボットの行動設計**であり，もうひとつは**センサ情報の解釈**である．ロボットの行動設計とは，ある条件を満足したときのロボットの振る舞いを記述することであり，センサ情報の解釈とは，ロボットがおかれている状況とセンサ情報との対応付けを指す．たとえば,「障害物を検出したとき」という条件を扱うには,「ロボットの先端に設置された赤外線センサの値が 700 以上の値をとったとき」とセンサ情報をあらかじめ解釈しておく必要がある．このとき，700 という数値は，現場においてロボットを対象となる障害物付近に配置し，値を参照して決定する必要がある．対象物表面の状態によって赤外線センサがとりうる値が異なるため，この作業を省略することは難しい．プログラマにとって大きな負担となるのは，図 1a のようにロボットの行動を設計する作業と，現場でセンサがとりうる値の解釈を並行して行わなくてはならない点である．特に，エンドユーザがロボットのカスタマイズする場合には，一人でこのような作業を行う必要があり，敷居が高い．

この問題を解決するひとつの方法は，あらゆる物体に対するセンサ特性をあらかじめ調査し，プログラムで利用可能にしておくことである．そうすれば，プログラマはセンサ情報の解釈を行うことなく，ロボットの行動設計に集中できる．しかし，あらゆる物体に対する特性を組み込むことは現実には困難である．

別のアプローチとして，図 1b のようにロボットの行動設計とセンサ情報の解釈を分離して行う方法が考えられる．つまり，ロボットの行動設計を完了した後に，センサ情報の解釈を行う．この方法では，プログラマの作業量は変わらないが，行動設計時にセンサ情報の解釈を行わなくてよいため，手順をシンプルにできる．本研究では，このアプローチを採用し，エンドユーザにとって効率的なロボットプログラミング環境を考える．

3. 未定義プログラミング

提案するロボットプログラミング手法は，ロボットの振る舞いをコンピュータ上で記述する**行動設計フェーズ**と，ロボットを実際に動作させながら外界の状態を記述する**センサ情報解釈フェーズ**を順番にたどる（図 1b）．外界に関する情報を未定義のままにし，テキストプログラミングを先に完了できるのが本手法の特徴である．

3.1 行動設計フェーズ

行動設計フェーズでは，ロボットを用いずにコンピュータ上でのプログラム開発を行う．ここでは，ルールベースでのロボットの振る舞い記述を前提とする．ロボットにタスクを行わせるためには，ロボットの状態や外界の状態を条件として，それらの条件を満足したときのロボットの動作を記述する．このとき，ロボットの状態や外界の状態は world オブジェクトとして定義し，具体的なセンサ値は記述しない．

連絡先: 小林一樹, 関西学院大学 理工学研究科
ヒューマンメディア研究センター,
〒 669–1337 兵庫県三田市学園 2 − 1 ,
Tel: 079-565-8300, E-mail: kby@ksc.kwansei.ac.jp

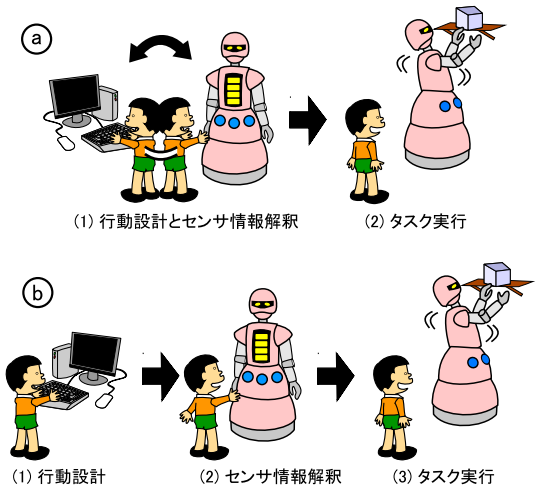

図 1: タスクプログラミングの流れ

```
1   while (1)
2     robot.goForward;
3     if (world.nearObject==true) {
4       if (world.nearTrashCan==true) {
5         robot.insertGarbage;
6       } else {
7         robot.turnRight;
8       }
9     }
10  }
```

図 2: ゴミ捨てプログラムの例

たとえば，ゴミ箱にゴミを捨てるタスクは図 2 のように記述することができる．この例では，ロボットは常に前進を行う．ロボットが 1 障害物を検出したとき，それがゴミ箱であればゴミをその中に入れ，ゴミ箱でなければ，右に旋回する．図 2 のプログラムでは，robot オブジェクトと world オブジェクトを用いてロボットの振る舞いを記述している．robot オブジェクトのメソッド goForward，turnRight と，world オブジェクトの nearObject に関しては汎用性が高いため，あらかじめ動作が定義されているものとする．しかし，insertGarbage と nearTrashCan については未定義とする．よって，プログラムは完成しているものの未定義な状態と行動があるため，このままではロボットはタスクを実行できない．この未定義なものに関しては，次のセンサ情報解釈フェーズにて行う．

3.2 センサ情報解釈フェーズ

センサ情報解釈フェーズでは，ロボットを用いてテキストプログラミングで未定義であった変数を，具体的なセンサ値やモータ駆動時間に基き定義する．このフェーズではテキストプログラミングは行わず，ロボットとのインタラクションを通して定義を行う．

上記のゴミ捨てプログラムの例では，insertGarbage と nearTrashCan が未定であった．この状態でロボットを動作させたとき，図 2 の 3 行目までは実行可能である．そのため，ロボットは障害物を検出するまで前進することができる．しかし，4 行目で nearTrashCan が未定義であるため，タスク実行を停止し，ユーザに問い合わせを行う．最もシンプルな問い合わせ方法は，音声で「nearTrashCan はどのような状態ですか？」と発話することである．ユーザはこの問いかけに対し，ロボットをゴミ箱の近くに移動し，ロボット上に配置された確認ボタンを押すことで，そのときのセンサ値を登録する．音声ではなく，ユーザが携帯端末を利用することで，より詳細なセンサ情報を与えることもできる．ロボットが複数のセンサを備えているとすれば，その中のいくつかのセンサ値だけを用いることで，他の状態との区別を明確にできる可能性がある．携帯端末とロボットで情報通信を行い，具体的にどのセンサを用いるのかを指定することで，より精度の高い定義が可能だと考えられる．

また，insertGarbage の定義に関しても同様に，ロボットからの問い合わせがあったときに，具体的な動作を教示する．教示方法は，ユーザが直接ロボットの身体を動かす方法と，携帯情報端末を用いて専用の制御インタフェースから遠隔操作する方法が考えられる．ロボットは未定義変数や未定義メソッドがなくなるまでユーザに対して問い合わせを行う．行動設計フェーズで記述したプログラムが大規模になれば，問い合わせの回数が増加する問題があるが，以前教示した変数を記録しておき，それらを再利用することで対処する．

4. 考察

提案手法では，テキストプログラミングを採用しているため，エンドユーザにプログラミングスキルを要求する問題がある．プログラムを記述することなく，実行例を示すことでロボットにタスクを学習させるアプローチ [Kuniyoshi 94, Friedrich 96] があるが，ユーザの多様な要求に答えるためには実装コストが大きい．そのため，本研究ではソフトウェア開発において利用され，普及している既存の手法を採用した．

外界の状態の定義方法に関して，本研究ではロボットのセンサ値に基づいて記述する方法を採用した．他の手法としては，環境中にカラーマーカーを貼り付けたり，RFID タグを設置することで定義することも可能である．定義の正確さという面では，環境に手を加える方が有利であるが，本研究では人工的な環境を用いない方針をとった．これは，家庭内での利用を考えたとき，ユーザの生活空間を積極的に侵食することは好ましくないと考えたためである．

5. まとめ

本研究ではセンサ値に基づく外界の記述を未定義のままテキストプログラミングを行い，実行段階で必要に応じてセンサ値を参照しながら外界を定義していく効率的なプログラミング方法を提案した．今後，提案手法を実装したプログラミング環境を開発するとともに，移動ロボットを用いた評価実験を行う予定である．

参考文献

[Friedrich 96] Friedrich, H., Münch, S., Dillmann, R., Bocionek, S., and Sassin, M.: Robot programming by Demonstration (RPD): Supporting the induction by human interaction, *Machine Learning*, Vol. 23, No. 2, pp. 163–189 (1996)

[Kuniyoshi 94] Kuniyoshi, Y., Inaba, M., and Inoue, H.: Learning by watching: extracting reusable task knowledge from visual observation of human performance, *IEEE Transactions on Robotics and Automation*, Vol. 10, No. 6, pp. 799–822 (1994)

[Touretzky 05] Touretzky, D. S. and Tira-Thompson, E. J.: Tekkotsu: A framework for AIBO cognitive robotics, in *Proc. of National Conference on Artificial Intelligence* (2005)