計測自動制御学会東北支部 第197回研究集会 (2001.10.13)
資料番号 197-7

車輪型自律移動ロボットの開発

Development of a autonomous mobile robot

○及川一美*，高氏秀則**，江丸貴紀**，土谷武士**，大久保重範*

○Kazumi Oikawa*，Hidenori Takauji**，Takanori Emaru**，
Takeshi Tsuchiya**，Shigenori Okubo*

*山形大学工学部, **北海道大学大学院工学研究科

*Faculty of Engineering, Yamagata University,
*Graduate School of Engineering, Hokkaido University

キーワード： 自律移動ロボット (Autonomous mobile robot)，行動決定手法 (Decision making)，
ロボットの開発 (Development of mobile robot)，行動規範型ロボット (Behavior-based Robotics)，
サブサンプション・アーキテクチャ (Subsumption Architecture)

連絡先： 〒992-8510　米沢市城南四丁目三番地十六号　山形大学 工学部 機械システム工学科
及川一美，Tel.: (0238)26-3246，Fax.: (0238)26-3246，E-mail: okazu@dip.yz.yamagata-u.ac.jp

1. はじめに

本研究の目的はロボットの利用を容易にすることである．今あるロボットのどれを見ても，ロボットを動かすためには工学的な知識が必要である．したがって，一般のユーザーに，と考えたとき，まだ敷居が高い．このことは，将来，介護福祉などの分野にロボットが家庭や福祉施設などで利用しようと考えたとき問題となると考える．今のロボットは主婦やケアワーカーが手軽に利用できるように考えられていない．しかし，ロボットは「誰にでも簡単に利用できる」事が理想である．

以上のことを念頭にして，今までシミュレーション上で議論を重ねてきたが，実機による議論もする必要があり，現在製作中である．今回の発表では製作中の移動ロボットの製作状況について述べると共に，行動決定手法について述べることにする．

2. ロボットの構成

移動ロボットは車輪型で独立二輪駆動方式とした．あまり制御に労力を割きたくないので，モータはステッピングモータを使うことにした．HDDを積んだまま移動するのは振動の問題や，バッテリーの問題から考えると好ましくないので，無線LANとNFSを使ってHDDを外し，更にFlush ROM Diskによって起動させることにより，完全なディスクレスの構成にすることにした．ロボットの基本的な構成は次のようなものを考えている．

- **CPUボード**
  PCI-555VRE（JDS）
- **CPU**
  Low Power Pentium 266MHz（Intel）
- **バックプレーン**
  PIC-09-5PC（JDS）

- **無線LAN** PCIA-11（corega）
- **Flush ROM Disk**
  MD-2800-D08（M-Systems）
- **ステッピングモータ**
  KH56QM2-951（日本サーボ）
- **モータドライバ**
  FSD 2B 2P12-01（日本サーボ）
- **OS**
  Vine Linux 2.1.5

Fig. 1　ロボットの基本構成

Fig. 1に現在，CPUボードなど構成中のロボット（と言えないですが）を示す．PCI-555VREは非常に曲者で理由は分からないがLILOで立ち上げることができず，立ち上げるまでに非常に手間取った．現在はSYSLINUXというブートローダでFDで起動している．

2.1　センサの選択

ロボットのセンサとして以下のようなものを考えている．

- **測距センサ**
  LRF，超音波センサなど
- **近距離センサ**
  PSDなど
- **タッチセンサ**
- **ランドマークセンサ**
- **ジャイロ**

LRFに関しては現在製作中で，以下のような装置を用いている．

- Video Blaster WebCam Plus (CREATIVE)
- LML-D12-635-5 (エフエムレーザテック)

Fig. 2　WebCam Plus とレーザ

WebCam Plusは安価で，性能もたいしたこともなくLRFとしてどこまで使えるのか疑問であるが，USBが使えるのでビデオボードも必要なく，LRF製作の入門としては手頃であると考える．

2.2　インタフェース

メインのCPUボードとセンサやモータなどとのインタフェースとしてPICの利用を考えている．構成としては**Fig. 3**を考えている．

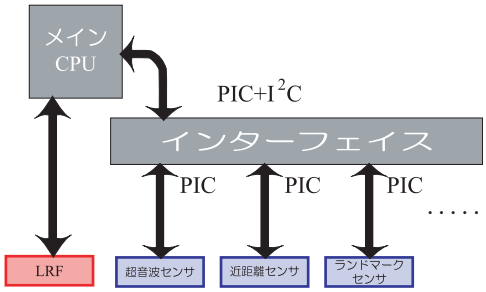

Fig. 3　インタフェース

PICは安価でその割りには多機能である．元々周辺機器とのインタフェース用に作られているので，シリアル通信などの機能も持っている．プログラムで自由に動作を変えることが可能であるの

が非常に魅力である．ただ，メモリが少ないことや高速な動作は期待することができないので，インタフェースとしてどこまでできるのか，今後調べてPICを採用するかを決めたい．

3. 行動決定手法

上記のように多種多様なセンサを装備して環境を認識しようとしたときに，センサ間の矛盾が問題になる．その矛盾を解消して正確に環境を認識することはセンサフュージョンの問題であり，そう簡単なことではない．BrooksはSA[1)]を用いてこの問題の解決を図った．本研究においてもSAを利用していた[2)]が，反射的な行動が基本であるこのSAを，計画的な行動と共に使おうとすると，SAの構成が非常に難しくなった．

今回は「状況」という概念を用いて，状況毎にSAを用意し，状況の変化に応じてSAを切り替えることで，上記の問題の解決を図った．

3.1 SAの問題点

SAはセンサの状態によって要素行動が発火し，同時に発火した行動の中から優先順位によって選択される．センサの状態によって行動が引き起こされるため，状況が異なってもセンサの状態が同じであれば同じ行動が引き起こされる．そのため，合目的にある障害物に向かって欲しいのに，反射的な行動がその障害物を避けてしまうような，矛盾が起こり，それを解消するためにパラメータを調整や，優先順位に工夫を凝らしてみたが，それは非常に難しい問題であった．

そこで，状況によってSAを別々に用意することを考えた．例えば，人が移動するときのことを想像して欲しい．普通に歩いている場合背中に何か接触を感じるとそれは障害物であり接触がなくなる方向へ行動が生まれる．しかし，車に乗っている場合はどうだろうか．背中に接触を感じてもそれは椅子であり障害物ではない．したがって，椅子を避けるような行動は生まれない．このように状況が異なれば同じ刺激を受けても異なる行動が生まれる．状況によってSAを切り替えることで，状況ごとのパラメータを決めるだけで良くなり，他の状況との兼ね合いを考える必要がなくなる．したがって，パラメータの決定に悩まされることが軽減されることが予想できる．ここではそのようなアルゴリズムを提案したい．

3.2 状況遷移

シミュレーションではあるが移動ロボットの状況として，**Fig. 4**を用意した．楕円で囲まれたものが状況を意味し，矢印が遷移の方向を意味している．状況は後述のイベントによって遷移される．そして，この状況毎にSAが用意され状況毎の異なる行動が現われる．

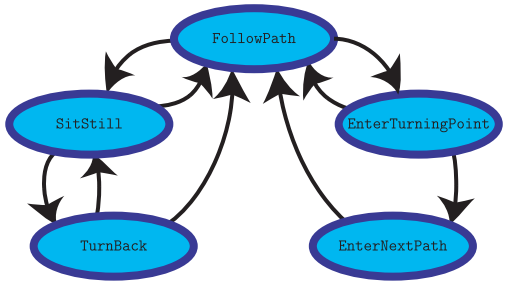

Fig. 4 状況遷移

3.3 イベント駆動

SAではセンサの状態で行動を発火させていたのに対して，本手法では「イベント」を採用した．センサの状態によって様々なイベントが発行される．イベントは状況を遷移させることと，行動を発火させることに用いられる．イベントを採用したのは，センサと行動決定をそれぞれオブジェクト化し切り離したかったためで，依存関係をなくすことで拡張性をよくすると共に，行動決定をシンプルにするのが目的である．

シミュレーションで用意したイベントの一例を以下に載せる．

- noOpenSpace

  開空間が無いときに発行するイベント

- oneOpenSpace

  開空間が1つ存在する時に発行するイベント

- openSpaceInFront

  前方に開空間があるときに発行するイベント

- existObstacle

  障害物が存在するときに発行するイベント

- detectLandmark

  ランドマークを発見した時に発行するイベント

- detectTurningPoint

  転回点用ランドマークを発見した時に発行するイベント

3.4 状況毎のSA

状況はイベントによって遷移され，状況毎にSAが用意されている．SAの要素行動もイベントによって発火される．SAで決定された要素行動は適切なセンサの値を用いて行動を生成する．

以下に状況の一例を載せる．

state `FollowPath` 枝路に沿って移動する状況

noOpenSpace

`SitStill`へ状況遷移

oneOpenSpace

`followPath`を発火

twoOpenSpaces

`followPath`を発火

manyOpenSpaces

`followPath`を発火

noOpenSpaceInFront

`SitStill`を発火

existObstacle

`avoid`を発火

detectTurningPoint

`EnterTurningPoint`へ状況遷移

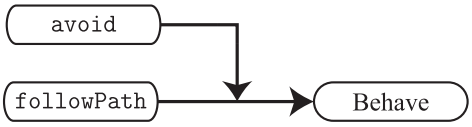

Fig. 5　state `FollowPath` におけるSA

state `EnterTurningPoint` 転回点に進入する状況

twoOpenSpaces

`enter`を発火

manyOpenSpaces

`EnterNextPath`へ状況遷移

existObstacle

`avoid_TP`を発火

noLandmark

`FollowPath`へ状況遷移

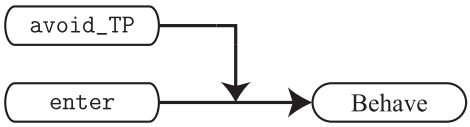

Fig. 6　state `EnterTurningPoint` におけるSA

基本はSAなのでパラメータの決定は試行錯誤であるが，このように状況に分け状況ごとにパラメータを決定できるので，以前よりは非常に楽に決定することができた．また，以前の方法では困難を極めた目的の枝路への進入が可能になった．以前の手法では状態の違いで状況も区別しなければ

ならなく，障害物を避ける場合と目的の枝路へ進入する場合を奇麗に分けることが困難であったからである．先程のSAの構成の内容を見て頂いてお分かりだと思うが，非常に簡単な構成である．実際にはもう少し複雑ではあるが，それでも以前に比べると簡単な構成である．以前の方法はここで紹介するのも嫌になるくらい複雑であった．今回のように問題が奇麗に状況を分けることができるのであれば，SAを使って反射的な行動と意志的な行動を混在させるには，本手法は構成が楽になるという意味で意義があると考える．

4. おわりに

開発中の移動ロボットの構成について述べた．イベント駆動型の行動決定手法について述べた．今後は実機を完成させ，この行動決定手法を適用して有効性を確認すると共に，本来の目的であるコミュニケーション手段, 特に手書き地図を用いたロボットとのコミュニケーションついて議論を進めたい．

参考文献

1) Rodney A.Brooks: “A Robust Layered Control System For A Mobile Robot,” IEEE Journal of Robotics and Automation, vol. RA-2, no. 1, pp. 14–23, 1986.

2) 及川一美，土谷武士：“手書き地図を用いた通路状環境のオフライン教示”，日本ロボット学会誌，vol. 17，no. 5，pp. 704–713，1999.