

1. はじめに

多数の自律的な物体が同一空間内を移動するとき、物体同士の衝突回避が問題となる。この問題について従来ヒューリスティックな回避の実現のためファジィ推論などを用いる方法が提案されてきた<sup>1)</sup>。しかしメンバーシップ関数などを人間が設定しなければならないため柔軟性に欠けている。柔軟性の向上のためには外部環境の応答により行動を修正する学習機能が有効であると考えられる。

本研究では学習機能を持つ回避アルゴリズムの構築のために、LA (学習オートマトン) を用いた方法を提案する。

2. LAの概要

一般にLAは  $\{x, W, a, p, A, G\}$  の6要素で構成される<sup>2)</sup>。  $x$  は入力集合で、LAに対する環境  $E$  の応答を構成する。  $W = \{w_1, w_2, \dots, w_s\}$  は内部状態の集合。  $a = \{a_1, a_2, \dots, a_r\}$  は行動の集合で、LAは各時刻においてひとつの行動を選択する。  $p(n) = \{p_1(n), p_2(n), \dots, p_r(n)\}$  は  $0 \leq p_k(n) \leq 1$ 、 $\sum_k p_k(n) = 1$  を満たす離散時刻  $n$  での確率の集合。  $A$  は確率集合を更新するアルゴリズム。  $G$  は内部状態から相当する行動を導く出力関数を示す。

LAは  $G$  より内部状態から行動を選択し環境  $E$  に送る。  $E$  はペナルティー確率  $C(n)$  より行動に対する応答をLAに返す。LAは応答を受けて  $A$  より確率集合を更新する (Fig. 1)。

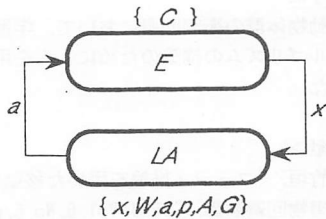


Fig. 1 Learning Automaton and Environment

3. 問題設定

問題の簡略化のため、移動物体は円形でステアリングのみを操作でき、速度は一定とする。また各物体の目標点は固定し、移動空間内に静的障害物は存在しないとする。

これらの仮定の下に自律移動物体が他の移動物体を回避しながら目標点に到達するモデルをFig. 2に示す。ここで物体の個数を  $m$  とし、各物体を  $M_1, \dots, M_m$  で表すことにする。さらに  $s_i(n)$ 、 $e_i$ 、 $d_i(n)$  を  $M_i$  の絶対座標系での位置ベクトル、目標点ベクトル、移動方向ベクトルとする ( $d_i$  は単位ベクトル)。

$o_i$  は  $M_i$  から見た目標方向単位ベクトルで、

$$o_i(n+1) = \frac{e_i - s_i(n)}{|e_i - s_i(n)|} \quad (1)$$

となる。  $l_{ij}(n) = |s_i(n) - s_j(n)|$  は  $M_i$  と  $M_j$  との距離、  $k_i(n) = |s_i(n) - e_i|$  は  $M_i$  と目標点との距離を示す ( $j = 1, \dots, m, j \neq i$ )。  $h_i$  は  $M_i$  の探索範囲円の半径、  $a_i(n)$  は  $M_i$  の回避角である。

もし  $M_i$  において、すべての  $l_{ij}(n)$  が  $h_i$  よりも大きければ  $d_i(n) = o_i(n)$  となるが、そうでない場合は回避アルゴリズムにより適切な  $a_i(n)$  を求めなければならない。

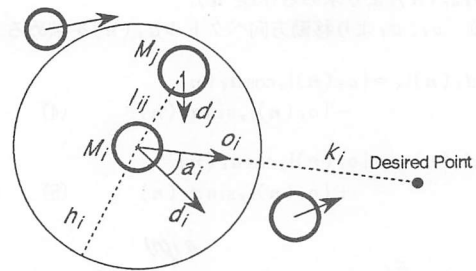


Fig. 2 Model of mobile machines

4. 回避アルゴリズムにおけるLAの応用

LAを用いた回避アルゴリズムは大きく分けると、入力から状態の導出、行動の決定、応答の決定、出力関数の更新、となる。

4-1. 状態の導出

①  $M_i$  に対し、  $l_{ij} \leq h_i$  である  $M_j$  の相対位置方向  $\theta_{ij}(n)$ 、相対速度方向  $\phi_{ij}(n)$  を目標方向ベクトルを基準に求める。ただし  $d_j(n)$  は未知なため  $M_j$  は直進すると仮定し、  $d_j(n-1)$  を用いる。

$$\theta_{ij}(n) = \cos^{-1} \left\{ \frac{o_i(n) \cdot (s_j(n) - s_i(n))}{l_{ij}(n)} \right\} \quad (2)$$

$$\phi_{ij}(n) = \cos^{-1} \{ o_i \cdot d_j(n-1) \} \quad (3)$$

②  $\theta_{ij}(n)$ ,  $\phi_{ij}(n)$ より $M_i$ に対する $M_j$ の状況認識を行う (Fig. 3 (a)).

③ ①, ②を $j=1, \dots, m (j \neq i)$ まで繰り返す.

④ 状態 $f_i(n)$ を求める (Fig. 3 (b)).

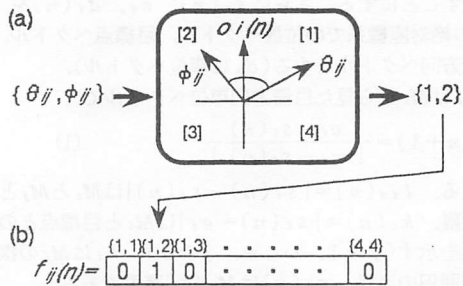


Fig. 3 Decision of state

#### 4-2. 行動の決定

①  $f_i(n)$ に等しい状態 $f_{iu}$ に対し, 相当する回避角 $a_i(n)$ を確率集合 $P_{iu}(n) = \{p_{iu1}(n), \dots, p_{iur}(n)\}$ より求める (Fig. 4).

②  $o_i, a_i$ より移動方向ベクトル $d_i(n)$ を求める.

$$\{d_i(n)\}_x = \{o_i(n)\}_x \cos a_i(n) - \{o_i(n)\}_y \sin a_i(n) \quad (4)$$

$$\{d_i(n)\}_y = \{o_i(n)\}_y \cos a_i(n) + \{o_i(n)\}_x \sin a_i(n) \quad (5)$$

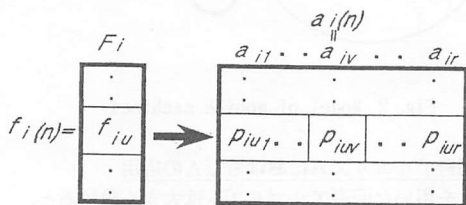


Fig. 4 Decision of action

#### 4-3. 応答の決定

各移動物体は定められた $d_i(n)$ より, 時刻 $n+1$ の $M_i$ の位置ベクトルは以下ようになる.

$$s_i(n+1) = s_i(n) + v_i * d_i(n) \quad (6)$$

ここで $v_i$ は $M_i$ の速度である.

この行動における環境の応答を以下のようにして決定する. 移動物体 $M_i$ と $M_j$ において, ( $j \neq i$ )両物体間の距離 $l_{ij}(n+1)$ を求め $l_{ij}(n)$ と比較する.

$$l_{ij}(n+1) \leq l_{ij}(n), \text{ and } \leq h_i \rightarrow \beta_j(n) = 1 \quad (7)$$

$$\text{otherwise} \rightarrow \beta_j(n) = 0 \quad (8)$$

移動物体 $M_i$ とその目標点において,  $k_i(n+1)$ を求め $k_i(n)$ と比較する.

$$k_i(n+1) \geq k_i(n) \rightarrow \beta_i(n) = 1 \quad (9)$$

$$k_i(n+1) < k_i(n) \rightarrow \beta_i(n) = 0 \quad (10)$$

$l_{ij}$ が以前より小さければ危険が増大するので $\beta_j$ は"ペナルティー"を示す1となる.  $k_i$ は逆に大きければ目標点からさらに遠ざかるので $\beta_i = 1$ となる.

これより $M_i$ の行動に対する評価を示す環境の応答を以下のように決定する.

$$x_i(n) = \frac{1}{2} \left\{ \frac{1}{m-1} \left( \sum_{k \neq i}^m \beta_k(n) \right) + \beta_i(n) \right\} \quad (11)$$

#### 4-4. 出力関数の更新

応答 $x_i(n)$ は $x_i \in [0, 1]$ なので確率集合の更新に $SLR-I$ スキームを用いる<sup>3)</sup>. これよりペナルティ $-x_i$ の期待値を最小にする様な行動に収束させることができる. 具体的には $\lambda$ を(0, 1)内の定数とすると次式のようになる.

$$\text{if } a_i(n) \neq a_{ik} \\ p_{ik}(n) = p_{ik}(n) - \lambda (1 - x_i(n)) \quad (12)$$

$$\text{if } a_i(n) = a_{ik} \\ p_{ik}(n) = p_{ik}(n) + \lambda (1 - x_i(n)) \quad (13)$$

#### 5. おわりに

自律移動物体群の衝突回避において, 学習機能を持つ回避アルゴリズムの構築のためにLAを用いた方法を提案した.

#### <参考文献>

- 1) 前田, 竹垣, "フェジィ推論を用いた移動ロボットの動的障害物回避制御, "JRSJ, Vol. 6, No. 6, pp. 50~54, Des. 1988
- 2) R. Mirchandaney, J. Stankovic, "Using Stochastic Learning Automaton for Job Scheduling in Distributed Processing Systems, "J. Parallel and Distributed Computing 3, pp. 572~552, 1986.
- 3) K. S. Narendra, M. A. L. Thathachar, Learning Automata, Prentice Hall, 1990.