3G2-05

包摂アーキテクチャ上の協調学習を用いた群ロボットによる作業効率の改善

Improvemnt of operating efficiency by multi-robots using collaborative learning on subsumption architecture

伊藤芳子[*1]　長谷川修[*2]
Ito Yoshiko　Hasegawa Osamu

[*1]東京工業大学　総合理工学研究科　知能システム科学専攻
Department of Computational Intelligence and System Science,Tokyo Institute of Technology

[*2]東京工業大学大学院理工学研究科像情報工学研究施設/[*3]科学技術振興機構　さきがけ研究 21
Imaging Science and Engineering Lab.,Tokyo Institute of Thechnology / PRESTO,JST

In this research,we consider the rest of robots' energy, the change of the number of robots and the change of the kind of tasks. In such situation,we intend to create the architecture that robots learn not to fail in the behavior , to use less energy and to do more task in obedience to the task.

1. はじめに

近年、複数台のロボットに分散・協調的に作業を行わせる研究が盛んであるが、それらの多くは一定のタスクを効率的に処理するための協調学習手法やアルゴリズムの研究に主眼が置かれ、タスクやロボット数が変化する状況にも対応可能なアルゴリズムはあまり報告されていない [平野 2000],[Mataric 1997]。

本研究では、個々のロボットにエネルギーの自律的な補給機能を与え、また群ロボットとしてタスクやロボット数の変動に対する学習・適応性を与えることにより、全体として環境の変化に対する頑健性を持たせるせることを目標とする。またタスクの処理にあたっては、様々な環境下でロボット全体のエネルギー消費最小、タスク達成効率最大となることを目指す。

2. アルゴリズムの概要

作業フィールド内にランダムに生じるタスクを処理するため、包摂アーキテクチャ上の協調学習を用いる。包摂アーキテクチャは群ロボットの基本的な挙動の制御に利用し、これによりロボットは原則として停止することなく動作する。

包摂アーキテクチャ上のレベルには、上位に「平衡状態レベル」、「タスク処理レベル」を導入する。「平衡状態レベル」により各ロボットは作業フィールド内でタスク発生に備え予め均等に散らばるようになる。しかしその分、全体としての消費エネルギーは多くなる。「タスク処理レベル」では、タスク発生後にどのロボットがタスク処理に向かうかが決定される。

「平衡状態レベル」における各ロボットの位置は、作業フィールドの壁および他のロボットからの反力と、床からの仮想的な摩擦力により計算される。ここで、これらの項目間には作業フィールド内のロボットの台数に依存したウエイトが存在し、ロボットには各状況において適切なウエイトを協調学習させることを考える。また「平衡状態レベル」、「タスク処理レベル」の起動によりタスク達成効率が改善するか否かはタスクに依存しており、この点においても協調学習を導入する。

本稿では、このうちタスクに応じてロボットが振る舞いを変えることの有効性をシミュレーションにより検証する。

連絡先: 伊藤芳子:〒 226-8503 横浜市緑区長津田町 4259 東京工業大学 像情報工学研究施設 R2-52 長谷川研究室，045-924-(5180 Tel，5175 Fax)，ito@isl.titech.ac.jp

3. 問題設定

作業フィールド：ロボットは正方形の作業フィールド内で活動する。正方形の一辺に充電場を設定し、エネルギー残量が一定値を下回ったロボットはここでエネルギーを補給する。

ロボット：ロボットには以下の仮定をおく。

- ロボットはすべて均質である
- 各ロボットはロボット間反力の計算のため周囲に一定のポテンシャルを持つ
- 移動距離に応じてエネルギーを消費する
- タスクの発生は全ロボットが認識可能
- ロボット同士は互いに通信可能

タスク：タスクは作業フィールド内にランダムに発生する。タスクが発生するとその場所に指定された台数が集まり、設定されたエネルギーを消費する。

4. 動作計画手法

包摂アーキテクチャ：包摂アーキテクチャはのシンプルな処理メカニズムを多層に重ね、高レベルの層が低レベルの層を包摂する機構である。本研究では以下の４層のレベルを設定する。

- レベル 0：障害物回避
  距離センサ入力の値が一定値以上なら回避行動をとる。
- レベル 1：充電場で充電する
  エネルギー残量が一定値以下になると充電場に行って充電する。
- レベル 2：タスクの処理
  タスクが発生していれば以下に述べる "$Value$" 値を算出する。"$Value$" 値を他のロボットと通信して比較し、これが大きい順に必要とされる台数だけタスク処理に向かう。
- レベル 3：平衡状態に向かう
  壁からの反力・他のロボットからの反力・摩擦力を受け、力学的に平衡に達するように移動する。

以上により個々のロボットは原則としていかなる状況でも停止することなく作業し続けることができる。

"$Value$" **値の算出：**レベル２における "$Value$" 値には以下の２種類を設定する。

- 方法 1:タスク発生場所までの距離 d からタスク処理に使用可能なエネルギー量を求める。その値がタスク処理に必要なエネルギーを上回るロボットの $Value = -d$ とし、この値の大きなものからタスク処理に向かうとする。

- 方法 2:タスク発生場所までの距離 d を $Value = -d$ とし、この値の大きなものからタスク処理に向かうとする。

平衡状態の算出：包摂アーキテクチャ上のレベル 3 に導入する平衡状態の算出のための反力は以下により計算する。

- 壁からの反力
  ロボットの左右上下の壁からの距離をそれぞれ d_{left}、d_{right}、d_{up}、d_{down} とすると、壁から受ける反力の x 成分 f_{wall_x}、y 成分 f_{wall_y} は

$$f_{wall_x} = d_{right} - d_{left}$$

$$f_{wall_y} = d_{up} - d_{down}$$

- 他のロボットからの反力
  ロボット i が他のロボット j となす角を θ_j、距離を d_j とすると、受ける反力の x 成分 $f_{robot_{jx}}$、y 成分 $f_{robot_{jy}}$ はガウス関数状に近似して以下とする。

$$f_{robot_{jx}} = \alpha \times \exp\left(\frac{-d_j}{\alpha^2\sigma^2}\right) \times \cos\theta_j$$

$$f_{robot_{jy}} = \alpha \times \exp\left(\frac{-d_j}{\alpha^2\sigma^2}\right) \times \sin\theta_j$$

  ここで α、σ は実験的に定めた定数である。これを自身以外のすべてのロボットについて求め、合計したものを他のロボットからの反力とする。

以上の反力に摩擦力 f_{react} を加え、ロボット i に働く力の x 成分 f_{i_x}、y 成分 f_{i_y} を以下のように求める。

$$f_{i_{(x,y)}} = a \times f_{wall_{(x,y)}} + b \times \sum_{j \neq i}^{j} f_{robot_{j(x,y)}} - c \times f_{react_{(x,y)}} \quad (1)$$

協調学習の導入：式 (1) の変数 a、b、c の値をロボット台数に応じて協調学習することでそれに対するロバスト性を保障する。

5. シミュレーション実験の結果と考察

以上のアルゴリズムの有用性を確かめるため、[Khepare Simulator version 2.0] によりシミュレーションを行った。

- **タスク処理に必要なロボット数と平衡状態のパラメータ**

タスク処理に必要とされるロボットの台数が 2 台、8 台の場合を設定し、式 (1) において、$(a, b) = (3, 2)$、$(a, b) = (5, 1)$ としてシミュレーションを行った。図 1 に結果を示す。評価（縦軸）は「効率　=　(タスク達成量)/(全消費エネルギー)」にて行った。図 1 で、2 台と 8 台の場合ではよい評価を与えるパラメータが異なっている。これはフィールド内に存在するロボットの台数と、各ロボット間の反力の間に密接な関係があるためと考える。よってここで協調学習を行うことにより、効率の改善が見込めると考える。

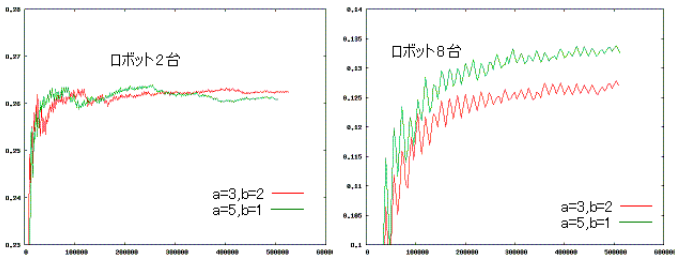

図 1: ロボット数が 2 台（左）、8 台（右）の場合の結果比較

- **”$Value$” 値および平衡状態の導入の有無についての評価**

評価は前項と同じく「効率　=　(タスク達成量)/(全消費エネルギー)」とした。実験では、ロボットの設定とタスクの種類を表 1 のように設定した。図 2 に結果を示す。

表 1 および図 2 において、タスクが頻発する場合、平衡状態の導入により効率が改善した。これは次々とタスクが発生する状況では、平衡状態に移行することによるタスク処理の向上率が消費エネルギーの効率を上回るためと考えられる。一方で、タスクが散発する場合は平衡状態を導入しないほうが良かった。タスクが頻発し、かつ必要台数が多い時は、”$Value$” 値は（方法 1）を採る方が良かった。これはロボットのエネルギー切れでタスク処理ロボットが交代する回数が増えるためと考えられる。一方必要台数が少ない場合は、タスク発生場所に近いロボットが交代で処理した方が効率がよいことが示唆された。よってタスクが頻発するときは平衡状態に移行するほうが望ましく、台数が多いときは ”$Value$” 値は (方法 1) をとるのがよいと考えられる。

以上より、”$Value$” 値の計算、平衡状態の導入を考慮にいれること、およびそれらを協調学習することの有用性が示されたと考える。

表 1: ロボットの動作設定とタスクの種類

ロボットの動作設定			タスクの種類		
	”$Value$”	平衡		台数	発生頻度
Case1	方法 1	導入	*Task1*	1	多い
Case2	方法 2	導入	*Task2*	1	少ない
Case3	方法 1	導入なし	*Task3*	3	多い
Case4	方法 2	導入なし	*Task4*	3	少ない

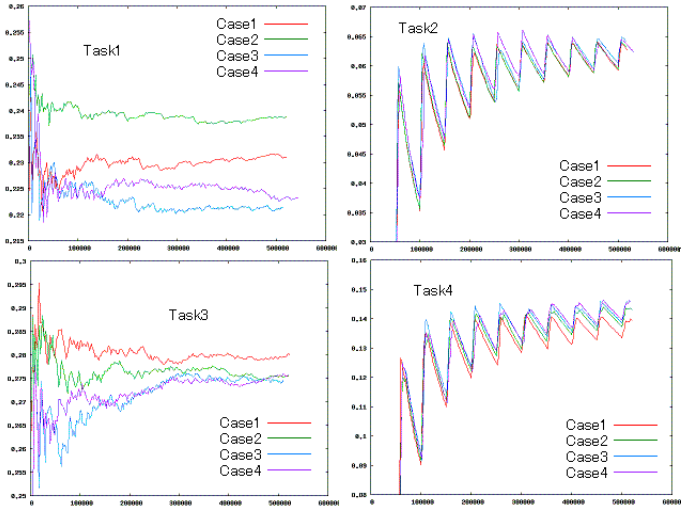

図 2: ロボットの動作とタスクの種類の実験結果（左上図から右下図方向に表 1 の *Task1* から *Task4* が対応）

参考文献

[平野 2000] 平野智一: 未知環境における移動ロボット群の経路学習, 日本機械学会論文集 (C 編), pp168-175(2000)

[Mataric 1997] Mataric,M.J.:”Reward Functions for Accelerated Learning”, The 11th Int’l Conf on Machine Learning,pp181-189(1997)