

## 自律搬送分散エージェントシステムのための ソフトウェア開発手法に関する一考察

村本 宜之<sup>1</sup> 伊東 靖英<sup>2</sup> 藤田 聡<sup>2</sup>

1. 広島大学大学院工学研究科 2. 広島大学工学部

近年、工場やビルなどで自動搬送システム (AGV: Automatic Guided Vehicle) の導入が進められつつある。AGV は動作において環境要因の影響を受けやすく、制御ソフトウェアには環境変化への高い適応性を要求される。本稿では、マルチエージェントシステムの実装方法を用いてこの問題を解決する手法を提案する。ここでは、1次元 AGV システムを用いて、その実現方法について考察するとともに、その効果を実験的に確かめた。

### A study of the technique of making softwares for autonomous distributed transport agent systems

Takayuki Muramoto<sup>1</sup> Yasuhide Ito<sup>2</sup> Satoshi Fujita<sup>2</sup>

1. Graduate School of Engineering, Hiroshima University  
2. Faculty of Engineering, Hiroshima University

Recently, more and more AGVs are being installed in factories, buildings, and so on. Control softwares for AGVs need high adaptivity for the change of environments, because AGVs are subject to environments. In this paper, we suggest some methods to solve this problem using implementation method of multi-agent systems. And we study the method and show the effect experimentally in an one-dimension AGV system.

## 1 はじめに

近年、工場の組立てラインやインテリジェントビル内のオフィス、あるいは病院やホテルなどの各種業務の現場において、自動搬送システム (AGV: Automatic Guided Vehicle) の導入が急速に進められつつある。その動機と目的は、多品種・大量資材の効率的運搬・配送、人為的ミスの軽減や作業効率の向上などである。こうした背景において AGV 制御に関する理論的な研究が行われている。[1][2]

本稿の目的は、AGV の制御をおこなうためのソフトウェアに対して強く求められる“環境の変化に対する柔軟な適応性”に着目し、そのような、いわゆる発展可能性をもつソフトウェアを開発するための具体的な方法論を提案・評価することである。

一般に、AGV のような自動搬送系を制御するための最適戦略は、その使用環境によって大きく異なると考えられる。AGV の制御戦略に関わる要因には、一般に次のような項目があると考えられる。

- 時間に関すること:  
処理の定常性 (決まった時刻に実行される処理の有無)、時間帯による処理内容の変化の有無、突発的

な処理要求の有無、およびその予測可能性、処理に要求されるリアルタイム性の強さ、など

- 空間に関すること:  
空間の広がり、搬送経路上の障害物の有無、およびその形状やサイズ、単位体積あたりに配置される搬送車の数、など
- 処理の質に関すること:  
処理内容の均質性、あるいはその多様性、など

このような要因に対して、できるだけ多くの可能性をあらかじめ想定しておくことは設計者の責任ではある。しかし、システム運用時に実際に発生する可能性のある要因をあらかじめ「すべて」列挙しておくことは不可能であり、したがって AGV のための制御ソフトウェアには、本質的に、環境や仕様の変化に対する高い適応性が求められることになる。

以上のような発展的な適応機能を実現するための手法として、本研究ではマルチエージェントシステム [3][4] を用いたソフトウェアの実装方法に着目する。具体的には、システムの出荷時に、基本的な機能と典型的な戦略のいくつかを複数のエージェントを用いて実装・提供しておき、運用時の機能拡張は、1) システム内への

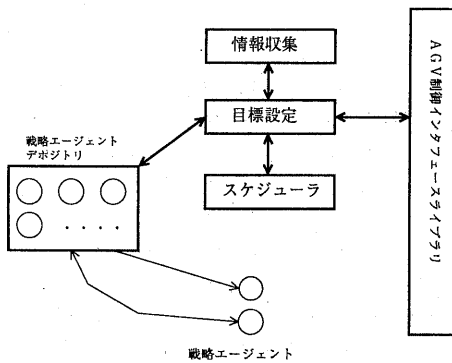


図 1: AGV のためのソフトウェアの基本機能

新しいエージェントの追加と 2) 既存エージェントの新しいエージェントへの置き換えとによって実現するという方針をとる。このようなマルチエージェントを用いた制御ソフトウェアは、各搬送車ごとに個別にインストールされる。すべての搬送車を集中的に管理するための中央コントローラは、用いてもよいし用いなくてもよい。本研究では、AGV システムのハードウェアコンポーネントを制御するソフトウェアは利用可能な形であらかじめ用意されているものと仮定する。

以上のような前提のもとで、本研究で対象とする制御ソフトウェアが提供すべき基本的な機能は、大きく分けて以下の 3 つにまとめられるであろう。

- 入力された大量の情報から必要な情報を抽出するための情報分類・獲得機能。
- 獲得した外界の状況 (のモデル) に応じて自分の動作のゴールやサブゴールを設定するための目標設定機能。
- 設定されたサブゴール群にしたがって、適切な動作をおこなうためのスケジュール機能。

## 2 システムの基本構成

本稿では、研究の第一ステップとして、一次元方向に移動可能な複数の自律搬送エージェントからなる搬送システムを対象とする。上下次元方向に移動する搬送システムの典型例であるエレベータシステムとの類似性から、本稿では各搬送エージェントのことをエレベータと呼ぶ。

## 2.1 概要

本稿で対象とするシステムは、中央コントローラ (以下、マネージャと呼ぶ)、固定された軌道上を一次元的に移動するエレベータ (群)、複数の停止可能位置 (以下、ポートと呼ぶ) に設置された呼び出しボタン、および、それらを結ぶ情報ネットワークから構成される。各ポートには 1 から  $n$  ( $\geq 2$ ) までのシリアル番号がそれぞれ割り当てられており、番号  $i$  のポートから番号  $j$  ( $\neq i$ ) のポートへ移動したい乗客は、 $j < i$  であれば下向きの呼び出しボタンを、 $j > i$  であれば上向きの呼び出しボタンを押すことでシステムに対して配車要求を送ることができる。配車要求は上下いずれかの方向のみを示す形でなされ、行き先を直接指定する形の要求を出すことはできない。行き先の指示は、乗車後に、エレベータ内部に用意された行き先ポート指示ボタンを操作することによっておこなわれる。また本稿で対象とする搬送システムはオムニバス方式で運行される。すなわち、ある決められた定員の範囲内であれば、異なるポートからの乗降を自由におこなうことができるものとする。

呼び出しボタンから入力された情報は、すべてマネージャに集められ、集中的に管理される。またシステム全体に関する静的なパラメータや各エレベータの運行状況もマネージャによって維持・管理される。マネージャによって管理される情報は具体的には以下の通りである。

- エレベータの台数と各エレベータの基本性能に関する諸元 (現段階では台数を固定し、エレベータはすべて同一性能をもつものとする)
- 各エレベータの停止可能ポートの集合 (現段階ではすべて均一; 将来的には、急行や快速などの搬送車もシステム構成に含めて考える予定である)
- エレベータのドアのオープン時間
- 各呼出ボタンの押下状況
- 各エレベータの現在位置、速度、進行方向 (上下、あるいは停止)
- 各エレベータ内にある行き先ポートボタンの押下情報
- 各エレベータ内にいる乗客の総重量
- すべてのボタンの押下状況と各エレベータの総重量の各時刻ごとの履歴 (履歴情報の利用方法については 4 節参照のこと)

各エレベータは、マネージャとの情報交換によって自身の動作に関して必要な情報を得ることができる (必要なすべての情報は必ず一度マネージャに集められるた

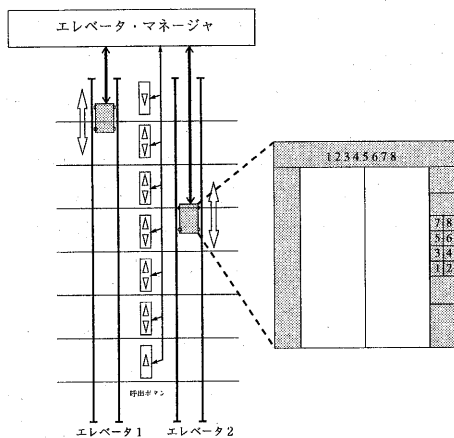


図 2: 対象とする AGV システムのシステム構成

め)。しかし、エレベータ数が大きくなった場合やより迅速な応答が望まれる場合などには、エレベータどうしが個別に通信や交渉をおこなって自身の動作を決定した方がより効率的になる場合も数多く存在すると考えられる。本アーキテクチャではそのような状況も想定し、エレベータ間で(マネージャを介することなく)情報交換をおこなうための通信チャンネルも設定してある。

## 2.2 エレベータの状態遷移

各エレベータは以下の3つの状態をもつ。なお状態間の遷移は、エレベータごと独立かつ非同期的におこなわれることに注意されたい。

**待機モード:** ドアを閉じて停止している状態(停止は必ずポートの位置でおこなわれる)

**乗降モード:** ドアを開けて停止している状態

**走行モード:** ドアを閉じて移動している状態

各状態間の遷移は、乗客によるボタン操作の他に、エレベータ自身の移動に伴う状態変化(目標ポートへの到着)やタイマー割り込み(ドアの自動開閉)などのイベントによって引き起こされる。時刻  $t$  において呼び出しボタンの押されているポート集合を  $Call(t) \subseteq \{1, 2, \dots, n\}$  とする。エレベータ  $j$  が対応すべき呼び出し  $Call_j(t)$  は何らかの集中的あるいは分散的な戦略によって適宜決定される(後述のように  $Call_j(t)$  は動的に変化し、しかもその変化は、エレベータ  $j$  自身によって制御されるときは限らない)。時刻  $t$  においてエレベータ  $j$  の行き

先ポートボタンによって指定されているポート集合を  $Dest_j(t)$  と記す。

エレベータ  $j$  における各状態間の遷移は、以下のようにおこなわれる。

### A. 待機モードからの遷移

対応するイベント:  $Call_j(t)$  が空集合から非空集合に変化したとき

**Step 1:** 現在エレベータのいるポートを  $i_1$  とし、 $Call_j(t) = \{i_2\}$  とする。

**Step 2:** もし  $i_1 = i_2$  ならば「乗降モード」に移り、そうでないならば目標ポートを  $i_2$  に設定し「走行モード」に移る(走行モードにおける目標ポートは、進行方向にある現在位置に最も近い  $Call_j(t) \cup Dest_j(t)$  の要素につねに更新される)。

### B. 乗降モードからの遷移

対応するイベント: ドアが閉じ終わったとき

**Case 1:** もし進行方向のポートの中に  $Call_j(t) \cup Dest_j(t)$  の要素があれば、その中の現在位置から最も近いポートを目標ポートとして「走行モード」に移る。

**Case 2:** もし Case 1 に該当せず、しかも進行方向と反対方向にあるポートの中に  $Call_j(t) \cup Dest_j(t)$  の要素があれば、方向を反転し、 $Call_j(t) \cup Dest_j(t)$  の中の現在位置から最も近いポートを目標ポートとして「走行モード」に移る。

**Case 3:** もし Case 1, Case 2 のいずれにも該当しなければ(つまり  $Call_j(t) \cup Dest_j(t) = \emptyset$  ならば)、「待機モード」に移る。

### C. 走行モードからの遷移

対応するイベント: 目標ポート  $i_2$  へ到着したとき

**Step 1:** もし  $i_2 \in Dest_j(t)$  ならば  $i_2$  を  $Dest_j(t)$  から消去する。また  $i_2 \in Call(t)$  ならば  $i_2$  を  $Dest(t)$  から消去する ( $Dest_j$  ではないことに注意)。

**Step 2:** もし進行方向のポートの中に  $Call_j(t) \cup Dest_j(t)$  の要素があれば、方向はそのまま Step 5. に進む。

**Step 3:** もし進行方向のポートの中に  $Call_j(t) \cup Dest_j(t)$  の要素が存在せず、しかも進行方向と反対方向にあるポートの中に  $Call_j(t) \cup Dest_j(t)$  の要素があれば、方向を反転して Step 5. に進む。

**Step 4:** Step 2, 3 で該当するポートがない場合は、進行方向を停止にする。

**Step 5:** ドアを開け「乗降モード」に移る。

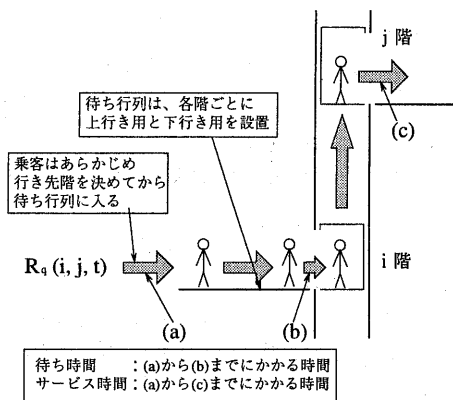


図 3: 乗客の流れのモデル化

### 3 乗客の流れのモデル化

#### 3.1 乗客の基本動作

本システムにおける各乗客の基本動作は以下の通りである。

1. 行き先ポートを決め、行き先ポートに応じた待ち行列の最後尾に並ぶ。
2. エレベータの到着を待つ。
3. エレベータが到着したら、待ち行列の先頭の人から順番に(定員いっぱいになるまで)エレベータの中に乗り込む。
4. エレベータに乗ったら、自分の行きたいポート(以下、個人目標ポートという)に対応する行き先ポートボタンを押す。
5. ドアが閉まり、エレベータが発車する。
6. エレベータが個人目標ポートに到着すると、エレベータから降りる。

次に評価値としてサービス時間と待ち時間を定義する。サービス時間とは、行き先ポートを決めて待ち行列の最後尾に並んでから、個人目標ポートに到達してエレベータを降りるまでの時間である。また待ち時間とは、行き先ポートを決めて待ち行列の最後尾に並んでから、エレベータに乗り込むまでの時間である。図 3 に乗客と待ち行列のモデルを示す。

#### 3.2 到着頻度分布関数

本稿では、乗客の流れを、待ち行列理論等で用いられる到着頻度分布によってモデル化する。したがって、あるポートから別のポートに移動する乗客の到着頻度

に関する情報は、到着頻度分布関数の形で与えられることになる。ただし、個々の乗客に独自の目標ポートを選択させるため、ここでは出発ポートと到着ポートのすべての組合せについて到着頻度分布を個別に設定することにする。時刻  $t$  におけるポート  $i$  からポート  $j$  へ向かう乗客の到着頻度を  $R_q(i, j, t)$  であらわす。ここで、ポート  $i$  からポート  $j$  に移動したい乗客は、 $i < j$  であれば上行き用の待ち行列に、 $i > j$  であれば下行き用の待ち行列に入ることに注意されたい。

## 4 環境の変化の自律的な学習

エージェントベースのアーキテクチャを採用することにより、各々のエレベータが環境の変化を自律的に学習し、その結果を以後の運行に効果的に反映させることが比較的容易に行なえるようになるものと期待される。ここでは学習対象として、各ポートの利用に関する時間的・空間的な傾向に着目する。学習によって獲得された要求のパターンに応じて制御戦略をうまく切替えることにより、各待ち行列の混雑の緩和と各乗客に対するサービス時間・待ち時間の短縮が実現される。

### 4.1 入力情報からの頻度変化の推定

本稿で対象としているシステムが直接得ることのできる情報は、呼び出しや行き先指定のボタンの押下履歴と各エレベータの重量変化のみである。すなわち、頻度分布の変化そのものを直接知ることはできない。

現在、ある一定時間間隔の押下履歴と重量履歴とから、その時間間隔の頻度分布のパターンを推定するための具体的なメカニズムについて検討を進めているところである。現在のところ、ニューラルネットワーク[5]を利用した学習法や、Genetic Algorithm を利用した Classifier System[6]、あるいは強化学習等の適用を検討している。

### 4.2 頻度に応じた最適戦略の変化

前節の手法によって、観測されたデータから頻度分布の変化がうまくとらえられたとして、その後で問題になるのは、その情報を具体的にどのようにいかにせよいのかという問題である。本研究では、1節でも述べたように、あらかじめいくつかの戦略を用意しておき、それらを環境の変化に応じて適宜切替えるという方針をとる。

次節では、最初のバージョンで用意されることになるいくつかの制御戦略の具体的な頻度パターンに対す

る性能を実験的に評価する。

## 5 実験

以下の実験では、エレベータ台数を2に、ポート数を8にそれぞれ固定する。その他のパラメータの値は表1のとおりである。

表1: 実験で設定されたパラメータの値

1台当たりの定員	30名
最高速度	120m/分
停止時間	6秒
初期位置	1階
待ち行列の長さの上限	600名

### 5.1 制御戦略

実験で対象とした制御戦略は以下の3つである。

**戦略1:** 呼び出し要求は、要求を出したポートの方向を進行方向とするエレベータの中でもっともそのポートに近いものに割り当てる。

**戦略2:** 各エレベータに対して担当ポートを静的に割り当てておき、呼び出し要求は、その要求を出したポートを担当するエレベータに自動的にわりあてられる(ここでは目標ポート1~4を担当するものと目標ポート5~8を担当するものを設ける。)

**戦略3:** 総重量を監視し、現在の総重量のもっとも軽いエレベータに要求を割り当てる。

### 5.2 到着頻度分布のパターン

実験では到着頻度分布の偏りのパターンとして、以下の4パターンに着目する。

**パターン1:** 到着頻度分布が任意の2ポート間ですべて等しい場合。

**パターン2:** ポート  $i_0$  からほかのポートへの移動要求が高い頻度であられるとき(出勤時間のモデル化)。

**パターン3:** すべてのポートからある特定のポート  $i_0$  への移動要求が高い頻度であられるとき(退社時間のモデル化)。

**パターン4:** ある特定の2ポート  $i_1, i_2$  間の移動要求が高い頻度であられるとき(映画館のあるビルのモデル化)。

各パターンの時間帯別到着頻度は、モデルとした状況を表現するよう、表2のように設定した。

表2: 各パターンにおける時間帯別到着頻度

パターン1		パターン2		パターン3	
t(分)	$R_q(i, j, t)$ (人/分)	t(分)	$R_q(1, j, t)$ (人/分)	t(分)	$R_q(i, 1, t)$ (人/分)
0~60	2.0	0~10	0.05	0~10	0.05
		10~50	7.5	10~50	7.5
		50~60	0.05	50~60	0.05

パターン4		
t(分)	$R_q(1, 6, t)$ (人/分)	$R_q(2, 6, t)$ (人/分)
0~10	0.05	0.05
10~30	0.05	20
30~60	20	20

### 5.3 結果

実験結果から、前記の乗客頻度パターンに対する制御戦略ごとの待ち時間の経時変化をプロットしたグラフを示す。図4がパターン1(均一分布モデル)に対するもの、図5がパターン2(出勤時間帯モデル)、図6がパターン3(退社時間帯モデル)、そして図7がパターン4(映画館がある場合のモデル)である。

実験結果のグラフには各々のパターンにおいて実験上最も意味があるような観測対象を図示した。

パターン1では上行きと下行きで傾向に大きな差はないこと、パターン2では需要の大半が1階からであること、パターン3では需要の大半が下行きであること、パターン4では1階からの乗客が増加した後の2階からの利用に着目する必要があるというようにそれぞれで最も意味のある観測対象が異なるためである。

これらの図から、それぞれの戦略に得手不得手があること、特に戦略2においてその傾向が顕著であった。戦略1と3については、ここで示した実験環境においては退社時間帯モデルを除き大きな差は見られなかった。このことは、需要のパターンを正確に読み取り、正しい制御戦略を選択することで大きな効果を上げることができる反面、制御戦略の選択を誤ると、逆に大きな混乱を招くことになる。

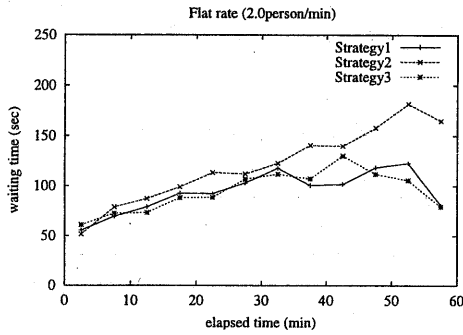


図 4: 均一分布モデル

● 上行き全体での待ち時間の平均

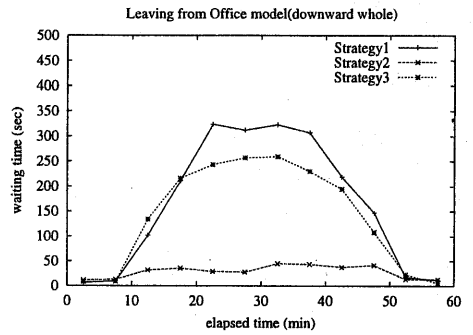


図 6: 退社時間帯モデル

● 下行き全体での待ち時間の平均

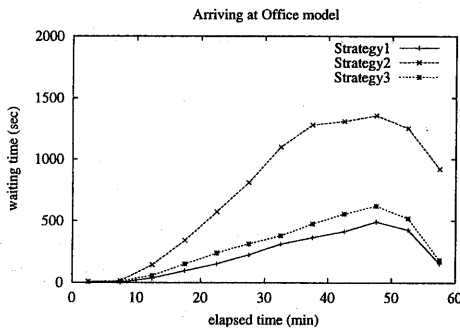


図 5: 出勤時間帯モデル

● 1階上行きでの待ち時間の平均

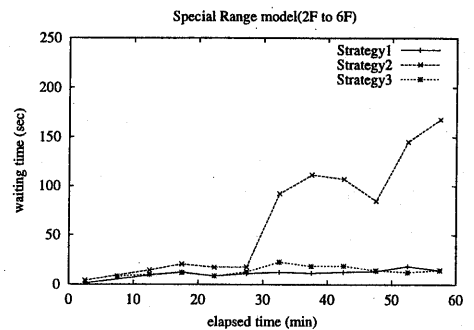


図 7: 映画館がある場合のモデル

● 2階から6階へ向かう乗客の待ち時間の平均

## 6 おわりに

実験の結果、需要の変化に対し、制御戦略を切替えることで AGV システムのより効果的運用が可能になることを示すことができた。なお、今回の実験は需要水準が高いものについての値なので、より低い需要水準における制御戦略についての実験や需要水準の高い区間を変更しての実験が必要となる。

今後は、正しい制御戦略の選択を行うために、正確な需要パターンの読み取りに必要な情報(利用状況、所要時間など)を学習し、需要動向が異なるいくつかの場面を組み合わせることで自動的に切替え、どのような場面においても AGV システムを効果的に運用できるようにシステムの開発をしていきたい。

## 参考文献

- [1] “AGV システムにおける最悪移動完了時間”, 佐々木淳・増山繁・山川栄樹, 電子情報通信学会技術報告, COMP94-10, SS94-4, pp.25-32, 1994-5.
- [2] “分散型の AGV システムにおける情報受信範囲の理論的解析”, 佐々木淳・増山繁・山川栄樹, システム制御情報学会, Vol.10, No.4, pp.211-222, 1997.
- [3] “An Introduction to Software Agents”, Jeffrey M. Bradshaw, “Software Agents”, AAAI/MIT Press, pp.3-46, 1997.
- [4] “Software Agents”, Michael R. Genesereth and Steven P. Ketchpel, Comm. ACM, vol.37, no.7, 1994.
- [5] “ニューラルネットの基礎と応用”, 馬場則夫・小島史男・小澤誠一, 共立出版, 1994.
- [6] “遺伝的アルゴリズム”, 北野宏明, 産業図書, 1993.