Vol.2013-OS-127 No.5
Vol.2013-EMB-31 No.5
2013/12/3

自動掃除ロボットの自己適応化に向けて

工藤拓光[†1]　中川博之[†1]　清雄一[†1]　田原康之[†1]　大須賀昭彦[†1]

本研究では，自己適応システムのアプローチの１つとして知られる MAPE ループを組込みシステムに付与することで，組込みシステムの自己適応化を目指す．本論文では特に，アプリケーションを自動掃除ロボット Roomba の制御機構に組み込むことで，自動掃除ロボットの自己適応化を試みる． Roomba の動作によるオブジェクトの移動について検証をし，検証結果をもとに MAPE ループを導入したプロトタイプを実装した．実装したプロトタイプをもとに，MAPE ループを付与した組込みシステムの自己適応化の有用性を評価する．

Toward Self-Adaptation of the Automatic Cleaning Robot

TAKUMITSU KUDO[†1]　HIROYUKI NAKAGAWA[†1]　YUICHI SEI[†1]
YASUYUKI TAHARA[†1]　AKIHIKO OHSUGA[†1]

We study about the self-adaptive systems of the automatic cleaning robot. Our proposed system that moves various objects by the using of our application incorporated into the control mechanism of the Roomba is toward self-adaptation. In order to change the behavior of Roomba to match the characteristics of the object, we have used the MAPE loop methods. MAPE loop is one of the approaches of self-adaptive system. In this paper, we evaluate the effectiveness of the self-adaptation, by experiment to move various objects using the MAPE loop methods.

1. はじめに

近年，自律的な動作をする掃除ロボットなど，産業用ロボットを家電製品化したものが普及し，制御システムが搭載された機器を家庭で扱うことができるようになった．例えば，iRobot 社の Roomba[1]がこれに該当し，充電された状態で Clean ボタンを利用者が押すだけで，清掃を開始し，清掃の終了後には充電器に自動的に戻る．そのため利用者がこれまで床清掃に費やしていた手間や時間を大幅に削減することができるようになった．このような背景から，今後ますます制御システムが搭載された家電製品が普及し，さまざまな特徴を持った機器が増えていくことが予想される．

しかし，自律移動をする掃除ロボットなどをあらゆる局面で利用するにはまだまだ課題がある．例えば，利用者が床に置いてある小物類などをあらかじめ片づけておく必要があるなど，利用者自身が前もって環境整備をしないと掃除ロボットの床清掃に支障をきたす．そのため，現状では掃除ロボットの床清掃に対して利用者の手間が少なからず掛かっている．

そこで，このような利用者の手間に対する問題を解決するため，組込みソフトウェアの観点から，掃除ロボットの自律化を目指す．近年広く普及している，Android タブレット[2]などのスマートデバイスは，利用者が欲しい機能をアプリケーションとしてインストールして利用するなどのカスタマイズ性の高さが特徴である．Android のバージョン 3.1 以降のデバイスは USB ホスト機能が搭載されており，シリアル機器への接続も考慮され始めている．本論文では，環境の変化や掃除ロボット自体の機能では対処できない状況に対して，スマートデバイスを利用して自己適応性を付与することにより，自律性を向上させる．ここで，自己適応性[3],[4]とは，システムが自身の機能と目標を管理し，環境変化に応じて柔軟な振る舞いをする能力を持たせる性質のことである．

本研究では，自動掃除ロボットが目的とする床清掃の内容を変化させず，ロボットを自己適応化させる．そのため，掃除ロボットの制御機構にアクセスし，掃除ロボットがオブジェクトを移動させる動作をするシステムを，基本的な清掃のライフサイクルに組み込む．実装には Nexus7 2012 と Roomba を利用している．

本論文では，MAPE ループ[5]と呼ばれる自律的動作を実現する制御メカニズムを Android アプリケーションに組み込み，Roomba を自己適応化させる．また，MAPE ループを組込みシステムに付与することの有用性を評価する．

本論文は以下のように構成されている．2 章では提案手法で用いる MAPE ループの説明および自己適応化の定義，3 章では，本研究で扱う基本的なシステムの実装内容についての説明，特徴をもったオブジェクトを複数用意し，オブジェクトの移動について検証した結果を示す．4 章では失敗に対して自己適応を行うシナリオを明確化し，明確化されたシナリオをもとに，提案手法である MAPE ループを利用した自己適応化部分の実装内容の説明を行う．また，自己適応化部分の全体像で，本論文で実装した自己適応化の機能について明らかにする．5 章では自己適応化に関す

†1 電気通信大学大学院情報システム学研究科
Graduate School of Information Systems, University of Electro-Communications

る評価実験について述べる．6 章では評価実験をもとに提案手法の有用性や限界に関する考察を述べ，7 章では関連研究事例を複数挙げ，本研究の位置づけを示し，8 章でまとめと今後の課題について述べる．

2. 提案手法

本研究では，自己適応システムのアプローチの 1 つとして知られる MAPE ループを，組込みシステムに付与することで，組込みシステムの自己適応化を目指す．前章でも述べたとおり，本研究では，自動掃除ロボットが床清掃をするシステムを変化させず，自己適応化させる．本研究が目指す組込みシステムは，Roomba の清掃中に障害物となるオブジェクトを発見した際に清掃を中断し，オブジェクト移動の動作に切り替わり，オブジェクトの移動後にもとの清掃に戻る，という前提で実装を行っている．オブジェクト移動の動作は，様々な未知の特徴を持ったオブジェクトを移動させることを前提としているため，オブジェクトの移動に試行錯誤する過程で適切な動作が呼び出される必要がある．

本章では，2.1 節で MAPE ループの概要，2.2 節で目指すシステムの全体像，2.3 節で本論文の自己適応化の定義を示す．

2.1 利用する MAPE ループについて

自己適応システムにおける自らの振る舞いを決定するためのアプローチのうち，有効であるとされているコントロールループの 1 つとして MAPE ループが挙げられる．MAPE ループは自己適応の主要な機能の要素として，Monitor（監視），Analyze（分析），Plan（計画），Execute（実行）の 4 つのコンポーネントがあり，図 1 のように構成されている．

本研究で利用する MAPE ループは，Monitor で外部環境を監視し，Analyze で外部環境の様子を分析する．Plan で Analyze によって得た情報をもとに動作に関する計画を立て，Execute は Plan で選択された動作を実行する．

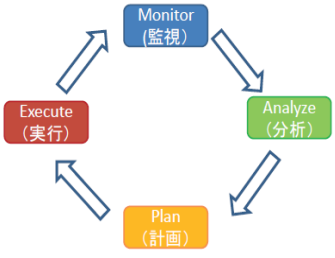

図 1　MAPE ループ

2.2 目指すシステムの全体像

図 2 にシステムの全体像を示す．清掃の開始と同時にオブジェクトを検知するために Monitor を開始する．オブジェクトが発見されず，正常に清掃することが出来る場合は，オブジェクト移動のツールを起動せずに，清掃の終了とともにシステムを終了する．オブジェクトを発見した場合は，Roomba の清掃を停止し，オブジェクトの移動をするため，清掃をしている Roomba の動作を一時的に停止させる．オブジェクト移動の基本的な動作を開始し，オブジェクトの移動を試みる過程で，移動時のオブジェクト追跡結果を記録し，オブジェクトの特徴を分析（Analyze）する．オブジェクトを運ぶ動作が終了せず，予期せぬ失敗が起こった際に，自己適応を開始し，Analyze で得た情報をもとに Plan でオブジェクトの特徴に適した動作計画を立て，自己適応後の動作（Execute）に取り入れ，自己適応後の動作を繰り返す．運ぶ動作終了後，自動清掃を再開してオブジェクトが見つからなければ清掃終了と同時にシステムが終了する．

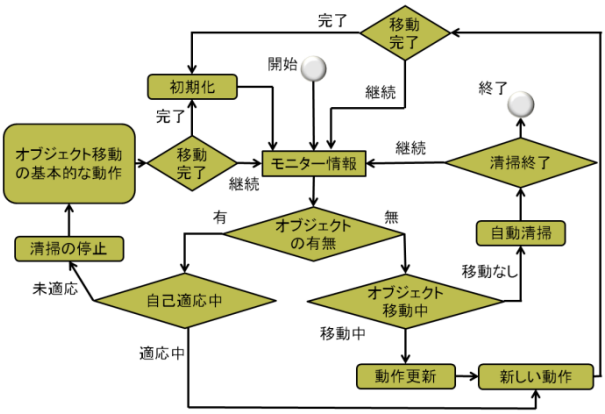

図 2　目指すシステムの全体像

2.3 掃除ロボットの自己適応化の実現方法

自律移動ロボットは予測不可能な事態に対処できるような自己適応能力の強化が不可欠である[6]．そのため本研究で提案する掃除ロボットの自己適応化とは，オブジェクト移動の基本的な動作ではうまく対処できない時に，状況を分析して新しい動作に切り替えることを指す．動作の切り替えは，オブジェクト移動時の予期せぬ失敗が発動契機となっており，本論文では，モニターからオブジェクトがフレームアウトすることを予期せぬ失敗として定義している．オブジェクトの特徴を，オブジェクト移動の基本的な動作で得たフィードバック情報をもとに分析し，新しい動作に切り替えることで，本研究の自己適応化が実現される．

3. オブジェクト移動の動作の実装・検証

本研究の実装には，物理的にオブジェクトを移動するための装置としてロボット掃除機 Roomba 500 Series を利用した．移動させる対象のオブジェクトの監視，および所定の動作が実装されたアプリケーションを組み込むための装置として，Android のタブレット端末 Nexus7 2012 を利用した．USB ホストケーブル及び，アイロボットルンバコミュニケーションケーブルをそれぞれの機器に接続して装置（図 3）を作成した．

Vol.2013-OS-127 No.5
Vol.2013-EMB-31 No.5
2013/12/3

図 3　作成した装置

3.1 オブジェクト認識

最初に，オブジェクトを Monitor で色認識できるようにするため，図 4 のようにオブジェクトを赤色に統一した．赤色でないオブジェクトには赤色のビニールテープを巻きつける処置を行い，赤色と判定できるようにした．色認識は，コールバックさせたカメラプレビュー映像を Bitmap 形式の画像データに変換し，各ピクセルの色判定によって赤色を認識できるという内容で実装した．

図 4　移動させる対象のオブジェクト

3.2 Roomba の動作

オブジェクト移動の基本的な動作で，Roomba の動作によってオブジェクトがどのように移動するかを検証するため, オブジェクト移動の基本的な動作メソッドを実装した．iRobot Roomba 500 Open Interface (OI) Specification[7]という Roomba を制御するための仕様書を利用した．Roomba と Android タブレットの接続を確認したのち Android から特定のバイトコードを送信する機能を実装した．Roomba の動作メソッドは，一定距離直進する（Go straight），左に回転し一定距離直進し元の向きに戻る（Turn left），右に回転し一定距離直進し元の向きに戻る（Turn right）の 3 項目を実装した．図 5 で示す通り，Bitmap 画像で赤色が検出された座標によってそれぞれの動作が呼び出されるように設計した．

図 5　Bitmap 画像の座標と動作の対応関係

オブジェクト移動の基本的な動作の全体像を図 6 に示す．図 6 は, 図 2 の一部を詳細化したアクティビティ図である．図 5 の通り，オブジェクトと Roomba の位置関係を把握し動作を行った後，再び位置関係を補足するといった処理を繰り返し行う．

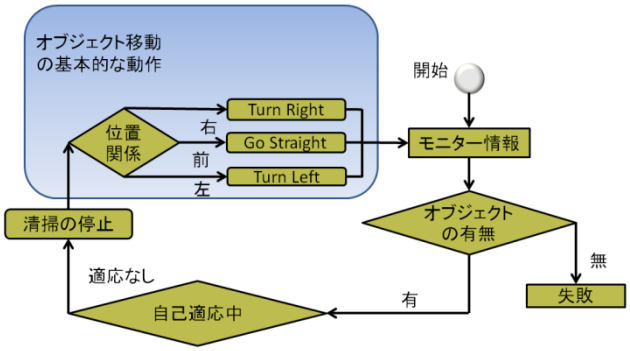

図 6　オブジェクト移動の基本的な動作
（図 2 の一部を詳細化したもの）

3.3 オブジェクト移動の検証

オブジェクトの特徴が原因で予期せぬ失敗をするシナリオの発見，および本研究で扱う自己適応の内容を定義するため，実装したオブジェクト移動の基本的な動作メソッドを用いて様々な種類のオブジェクトを移動させる検証を行った．この検証は，オブジェクト移動の基本的な動作に対して，どのようにオブジェクトが移動するかを観察するために設けており，自律移動ロボットの基本的な動作である直進・左折・右折のみでオブジェクトを移動させる検証は妥当と考える．Roomba によって動かす対象とするオブジェクトを，見た目の形状と大きさを考慮して 4 種類（図 4 参照）用意した．オブジェクトの重さや重心の位置を考慮して，空き缶とコップに関しては 2 パターン，ボールに関しては 1 パターン，ペットボトルに関しては 3 パターンの計 8 パターンで検証を行った．Roomba が移動させる 4 種類のオブジェクトとそれぞれの状態について表 1 に示す．

Vol.2013-OS-127 No.5
Vol.2013-EMB-31 No.5
2013/12/3

表 1　移動させるオブジェクトの種類とパターン

パターン	オブジェクト	状態
パターン 1	空き缶	倒れている
パターン 2		潰れている
パターン 3	コップ	倒れている
パターン 4		立っている
パターン 5	ボール	――――――
パターン 6	ペットボトル	空の状態
パターン 7		半分まで水
パターン 8		満タンまで水

本論文の検証では，オブジェクト移動の内容を検証の対象としているため，Roomba の清掃からオブジェクト移動に切り替える部分については扱わない．オブジェクトを Roomba の前面に同じ向きに配置し，Android アプリケーションを起動させることを検証開始時の条件とした．オブジェクトの特徴のみではなく，オブジェクトの接地面の床の材質によって，接地面の摩擦の大きさが変わるため，摩擦の大きさが異なる床（フローリングおよび絨毯）で検証を実施した．5m の直線距離でオブジェクトを見失わずに移動させられるか，それぞれのシチュエーション毎に 5 回ずつ検証した．検証失敗時にオブジェクトを見失った原因についても記録した．

表 2　結果の記録方法について

結果	移動の成否	オブジェクトの特徴
○	成功	――――――
左	失敗	左にずれる
右	失敗	右にずれる
転	失敗	前方に転がる
乗	失敗	乗り上げられる

3.4 検証結果

検証の結果を表 3，表 4 に示す．

表 3　オブジェクトごとの検証結果　（フローリング）

パターン	1 回目	2 回目	3 回目	4 回目	5 回目
パターン 1	転	○	転	転	○
パターン 2	乗	○	乗	○	○
パターン 3	右	右	右	右	転
パターン 4	○	○	○	転	○
パターン 5	転	転	転	転	転
パターン 6	右	右	○	左	右
パターン 7	○	○	○	右	○
パターン 8	○	○	○	○	○

表 4　　オブジェクトごとの検証結果　（　絨　毯　）

パターン	1 回目	2 回目	3 回目	4 回目	5 回目
パターン 1	○	○	○	○	○
パターン 2	乗	乗	乗	○	乗
パターン 3	右	右	右	右	右
パターン 4	○	○	○	○	○
パターン 5	○	○	○	○	右
パターン 6	右	○	○	右	右
パターン 7	○	○	○	○	○
パターン 8	○	○	○	○	○

検証結果より，以下の項目について言及できる．

(A) 潰れた空き缶は，絨毯では乗り上げられてしまう．
(B) 倒れたコップは，水平方向にずれる特徴がある．
(C) ボールは，フローリングでは転がってしまう．
(D) 中身が空のペットボトルは，失敗しやすい．

4 章では(A)，(B)，(C)についてのオブジェクトの特徴に着目し，オブジェクトの移動に失敗した際の自己適応を行うアルゴリズムの実装を行ったことについて記述を行う．

4. 自己適応化部分の実装

4.1 失敗の定義

本論文ではオブジェクトがスクリーンからフレームアウトした時に，監視できなくなり失敗することを自己適応の発動契機とした．失敗するオブジェクトの特徴は 3 章の検証結果から

I.　左にずれる特徴
II.　右にずれる特徴
III.　前方に転がる特徴
IV.　乗り上げられる特徴

と定義でき，認識範囲から外れた様子を，図 7 のように表わすことができる．

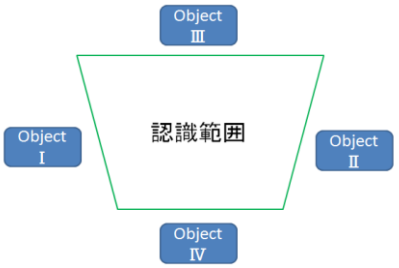

図 7　認識範囲から外れたオブジェクトの特徴

オブジェクトのそれぞれの特徴を定義すると，Object Ⅰは左にずれる特徴をもつオブジェクト，Object Ⅱは右にずれる特徴をもつオブジェクト，Object Ⅲは前方に転がる特徴をもつオブジェクト，ObjectⅣは乗り上げられる特徴をもつオブジェクトである．

4.2 オブジェクトの特徴についての判定方法

Android アプリケーションでオブジェクトの特徴を判定するため，画像認識の段階でオブジェクトを追跡する部分を実装した．アプリケーションの起動後，6 ブロックに分割した画面（図 8 参照）で一定数のフレーム間隔を空け赤色検出を行い，検出箇所（A～F）にオブジェクトが出現した回数を N（N_A～N_F）とし，N の値を利用してオブジェクトの特徴を判定した．

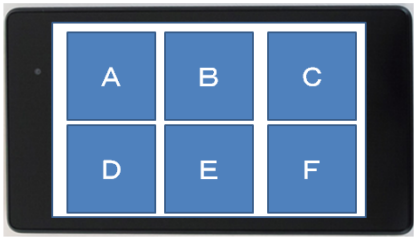

図 8　オブジェクト追跡時の赤色検出箇所

Roomba がオブジェクトの移動に失敗し，自己適応後の動作を開始する条件を図 9 に示す．Roomba の動作が呼ばれていないことを Bitmap が更新されるフレーム数でカウントし，カウント数が一定の値を上回った際に自己適応後の動作を開始する．オブジェクト検出箇所に該当する引数を，自己適応後の動作を呼び出す判断材料になるように実装を行った．条件式については以下のとおりである．

I. D にオブジェクトが存在している回数 N が F の場合よりも大きい．（$N_D > N_F$）
II. F にオブジェクトが存在している回数 N が D の場合よりも大きい．（$N_F > N_D$）
III. A+B+C にオブジェクトが存在している回数 N が D+E+F の場合よりも大きい．（$N_{A+B+C} > N_{D+E+F}$）
IV. N_E が最も大きい．（$N_E > N_A$ && $N_E > N_B$…&& $N_E > N_F$）

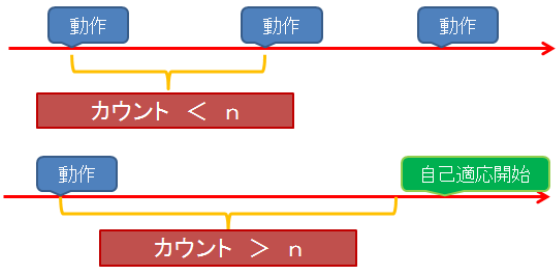

図 9　自己適応後の動作の発動契機

4.3 Roomba の自己適応後の動作

MAPE ループを組込みシステムに付与させることの有用性を評価するため，自己適応後の動作の選択肢を限定し，具体的な動作の内容を実装した．自己適応後の動作の内容については以下のとおりである．

I. オブジェクトを Roomba の右前に配置させるために，Turn left 実行時に，直進距離を長くとる．
II. オブジェクトを Roomba の左前に配置させるために，Turn right 実行時に，直進距離を長くとる．
III. オブジェクトと Roomba が連動して動作する．
IV. 1 回に進む距離を短くし，速度を上げる（押し出す）．

Object Ⅰ，Object Ⅱ，ObjectⅣはスクリーンのフレーム内にオブジェクトを戻す処置をするため，自己適応後の動作に入る前に Roomba が後退する処理を挿入した．

4.4 システムの全体像

自己適応化部分の実装に際して，MAPE ループでどのような処理を行うかを以下のように定義した．本論文で用いる MAPE ループの構成パターンは図 10 のとおりである．

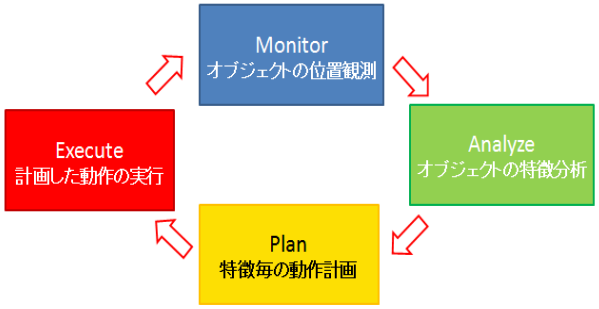

図 10　本論文の MAPE ループ

実装したプロトタイプを図 11 で示す．オブジェクト移動の基本的な動作の繰り返しの過程でオブジェクトの特徴を Monitor で追跡すると同時に分析（Analyze）し，分析結果をもとに自己適応後の動作に関するプランニング（Plan）を行う．自己適応後の動作（Execute）は移動の基本的な動作と同様に，オブジェクトと Roomba の位置関係を取得しながら繰り返す．

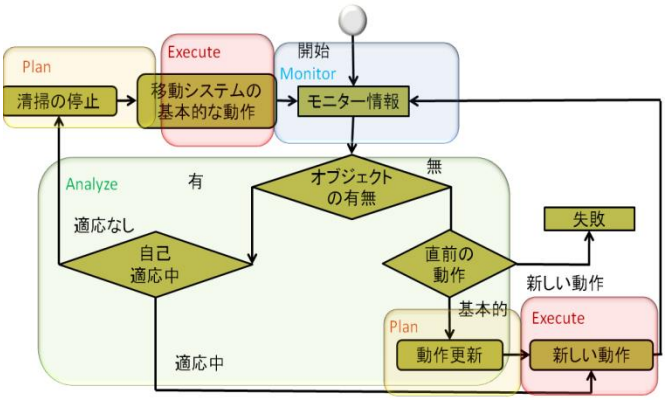

図 11　プロトタイプのアクティビティ図

Vol.2013-OS-127 No.5
Vol.2013-EMB-31 No.5
2013/12/3

5. 評価実験

本章では，自己適応化した Roomba の動作に関して

- オブジェクト移動の失敗を判断できるか.
- 提案した条件式が適切であるか.
- 提案した動作が適切であるか.

を確認するために評価実験を行った.

5.1 実験手順

3 章で実際に用いたオブジェクトのうち, 図 7 で Roomba の自己適応によって対処すべきオブジェクトを定義したため，Object Ⅰや Object Ⅱの特徴をもつ硬質プラスチックのコップ，ObjectⅢの特徴をもつボール，ObjectⅣの特徴をもつ潰れた空き缶を，移動させる対象のオブジェクトとして用いた．実験環境も 3 章と同様にフローリングと絨毯のフロアを用意した.

「条件式が適切であるか（絨毯)」の検証については失敗後の自己適応後の動作の内容を 5 回ずつ観察し，条件式が適切であるかを判断した. 別の動作が呼ばれた場合には，呼び出された動作を記録した.「提案した動作が適切であるか」の検証については, 自己適応後の動作を検証するため，自己適応後に，動作開始直後のオブジェクトの特徴に対して適切な動作が呼ばれたことを確認したのち，動作が 10 ステップ行われたことを確認できた場合を成功とみなし，検証開始地点が異なることから進んだ距離については言及しないこととした．それぞれ 5 回ずつ検証を行った.

ObjectⅢに該当するボールには，絨毯の検証で成功していたため，「条件式が適切であるか（絨毯)」の検証をしなかった．また動作テストの段階で，オブジェクトと連動した動作の実現が困難であることから「提案した動作が適切であるか」の検証をしなかった.

5.2 実験結果

オブジェクトの移動に関する自己適応の結果を以下に示す．Roomba がオブジェクトの移動に失敗したことの判断については，全て成功していた．それぞれのオブジェクトは 4.1 節で定義したものである．条件式や提案した動作の適切さを評価するにあたり，成功した確率が 80%以上（5 回の検証のうち 4 回以上成功する）の場合，適切であると判断している.

表 5　条件式が適切であるか.　（フローリング）

	1回目	2回目	3回目	4回目	5回目
Object Ⅰ	○	○	○	Ⅲ	Ⅲ
Object Ⅱ	Ⅳ	○	○	○	Ⅲ
ObjectⅢ	○	Ⅰ	○	○	Ⅰ
ObjectⅣ	Ⅱ	○	○	○	○

表 6　条件式が適切であるか.　（絨毯）

	1回目	2回目	3回目	4回目	5回目
Object Ⅰ	○	○	Ⅳ	○	Ⅳ
Object Ⅱ	○	○	○	Ⅰ	Ⅳ
ObjectⅣ	○	○	○	○	○

表 7　提案した動作が適切であるか.　（フローリング）

	1回目	2回目	3回目	4回目	5回目
Object Ⅰ	○	○	転	○	転
Object Ⅱ	転	○	○	転	転
ObjectⅣ	○	乗	○	○	○

表 8　提案した動作が適切であるか.　（絨毯）

	1回目	2回目	3回目	4回目	5回目
Object Ⅰ	○	○	○	○	○
Object Ⅱ	○	左	○	○	○
ObjectⅣ	乗	乗	乗	乗	乗

結果に関する評価は，以下の通りである．最初に，オブジェクト移動の失敗を判断できるかについては，オブジェクトの移動に失敗したことを判断して自己適応の動作を開始することが，検証した全てのオブジェクトで確認できたため，判断できるといえる.

次に，提案した条件式の適切さについては，ObjectⅣを判断する時以外には適切ではないといえる．Object Ⅰと Object Ⅱは, Roomba が動作してオブジェクトを移動させる途中で，オブジェクトが回転して特徴が入れ替わることがあった．また，回転してオブジェクトの底面が Roomba の進行方向と一直線上になった時に，条件式の N_E が大幅に加算されてしまうことによって ObjectⅣの自己適応後の動作が呼び出される結果が出た．Monitor の段階でオブジェクトを見失った瞬間を補足する手法が必要である．スクリーンの外枠を頻繁に監視する手法で，フレームアウトした位置を補足することによって，オブジェクトの特徴を分析する時の失敗は回避できると考える.

最後に，提案した動作が適切であるかについて，絨毯での Object Ⅰと Object Ⅱおよび，フローリングでの ObjectⅣについては，適切であると判断できる．フローリングで倒れたコップを検証した際に,コップが前方に転がる特徴を有していることから，横移動を強化する自己適応後の動作では対処できず，オブジェクトを移動させる途中でスクリーンからフレームアウトした．そのため，本論文で提案した動作が，適切であるかを判断するためには，Object Ⅰ及び Object Ⅱの特徴のみを再現できるオブジェクトで検証する必要がある．絨毯での ObjectⅣについては，自己適応後

に Roomba が速度を上げる動作をしても，オブジェクトを押し出せずにそのまま乗り上げてしまう．Roomba がオブジェクトを移動させる際に利用するフロントバンパーとObjectⅣが，ほとんど接触しないためである．Roomba がObjectⅣに乗り上げないようにするためには，自己適応後の動作の内容を変更し，フロントバンパーとオブジェクトを接触させずに運ぶことが出来る動作を定義する必要がある．

6. 考察

評価実験の結果をもとに考察を行う．まず，ObjectⅠ，ObjectⅡに該当するオブジェクトにコップを選定していたが，前方へ転がるという ObjectⅢの特徴も持ちあわせている．このようにさまざまな特徴を持ったオブジェクトに対して有効な自己適応の動作を提案する必要がある．

本論文の実装では，Roomba の動作に関する自己適応化を，基本的な移動からプロトタイプに変更する 1 回のみを動作の更新で定義していたため，オブジェクトの特徴が変化することに対処できなかった．そのため，動作の自己適応化を基本的な動作からプロトタイプに変化させるだけでなく，プロトタイプの動作がうまくいかない場合についても動作を変更する必要がある．本論文の実装内容は，Plan であらかじめ実装した動作メソッドを呼び出している．この場合，自己適応後の動作が決まっていることから，未知の特徴をもったオブジェクトに対して有効ではない動作をする可能性がある．したがって，再プランニングする機構および学習アルゴリズムを実装し，Roomba が未知のオブジェクトを移動させながら試行錯誤ができるようなメカニズムが必要となる．

床が絨毯の場合での ObjectⅣに関しては，自己適応前と自己適応後のオブジェクト移動において，共に成功しなかった．Roomba 前方のフロントバンパーで ObjectⅣを前に押し出せないことが原因として挙げられる．この場合，フロントバンパーを用いずに，前に押し出す動作を作り出す必要がある．Monitor だけで，このようなオブジェクトの特徴を判定することは困難である．しかし Roomba の場合では, 清掃でサイドブラシを利用し, ゴミを回収している．これらを利用してオブジェクトを移動するなど，様々な動作を Plan で計画することによって, オブジェクトの移動の手段が充実化する可能性がある．

今回の実装部分の有用性として，オブジェクトの移動が失敗した際に，自己適応のシナリオ通りに Roomba の動作が実現できたため，有用性があるといえる．しかし，システムが自ら動作について計画を立てる Plan のフェーズにおいて，条件分岐による動作メソッドの呼び出しのみで処理が完結していることに起因し，自己適応後の動作でもオブジェクトが運べない結果が出た．学習アルゴリズムや Monitor の段階での監視能力の強化などが求められており，実現においては課題が多いといえる．

7. 関連研究

本研究と同様に，未知のオブジェクトの移動に貢献した自律ロボットの協調動作によるオブジェクト移動に関する研究事例および，本研究と同様に自己適応システムを適用した，Roomba の協調動作について紹介する．紹介した事例をもとに本研究の位置づけを明確化する．

7.1 自律ロボットの協調によるオブジェクト移動

UAV と呼ばれるオブジェクトの状態監視を担当するロボットと, UGV と呼ばれるオブジェクトの移動を担当するロボットの協調動作について扱っている研究がある[8],[9]. UAV は飛行する性質をもったロボットであり，UGV が地上で観測できないオブジェクトに関する情報について，上空からオブジェクトが配置されている状況（PUSH 型，PULL 型）[9]を，画像認識を利用して判断し，複数の UGV のうち，そのオブジェクトの移動に適した UGV に通信をする．通信する際に UAV は UGV に対してオブジェクトの移動に適した動作方法をナビゲートする．

この研究では，オブジェクトの移動方法の導出結果によって，オブジェクトがうまく移動できなかった際の再プランニングする機構が存在しない．本研究は，今後，学習アルゴリズムの導入によって再プランニングする機能を実装する予定である．また，本研究では協調動作について扱っていないが，この研究と本研究を組み合わせる事により実用的なシステムが構築できるのではないかと考えられる．

7.2 自己適応システムを採用した Roomba の協調動作

ロボット工学分野におけるモデル駆動開発について扱っている研究がある[10]．システムの自己適応における適応計画（Plan）の自動生成を，3 階層のアーキテクチャを用いた手法を用いて行うことを提案している．提案手法の有用性を示すために，Roomba の協調動作について扱っている．Desktop PC で適応計画を自動生成し，Roomba に無線で通信を行い，複数台の Roomba が予期せぬシナリオに対して自己適応ができることを証明した．

本研究との差異として，この研究では Plan の自動生成の面で優れた研究であるが，Roomba の動作によるオブジェクトの移動に関して扱っている内容ではない．また，この研究では，システム内部の動作処理の失敗について主に言及していることから，本論文のシナリオであるオフラインの外部環境に対して自己適応を行っている点で差異がある．

8. おわりに

本論文では，自律動作する掃除ロボットにおける現状の課題について着目し，床に置かれたオブジェクトの移動に対して，MAPE ループを利用した自己適応化を掃除ロボットに組み込むことで，提案手法の有用性を示した．3 章で実装をしたオブジェクト移動の基本的な動作から，4 章で

実装した動作の自己適応化によって，様々な特徴をもったオブジェクトを移動できるようになったことが本論文の貢献である．

しかし，本論文は自己適応後の動作メソッドがあらかじめ実装されていることから，オブジェクトの特徴が変化した際に自己適応後の動作が切り替わらないなどの課題がある．今後，掃除ロボットの自己適応化に向けて，Plan のフェーズで再プランニングする機構の実装，Roomba が試行錯誤してオブジェクトを移動させるツールの実装が不可欠である．また，本論文では Monitor のフェーズでオブジェクトの追跡に失敗した際に自己適応を行っているが，現在ではオブジェクト追跡の手法が高精度化している．そのため設定した失敗のシナリオでは不十分であり，より完成度の高い失敗のシナリオが求められている．

今後の研究として，Roomba が未知のオブジェクトを移動できるようにするため，Roomba が様々な動作を試行錯誤して，オブジェクトを移動させるような学習システムを Plan のフェーズで実装する．また，本論文の失敗シナリオである，スクリーンからのフレームアウトだけでなく，予期せぬ失敗に関するシナリオを充実化させ，あらゆるシステムの失敗に対処できるような組込みシステムの自己適応化を目指す．

謝辞 本研究はJSPS科研費24300005，23500039，25730038の助成を受けたものです．

参考文献

[1] iRobot 社， Roomba， http://www.irobot-jp.com/
[2] Android タブレット， android， http://www.android.com/
[3] Rogerio de Lemos, Holger Giese, Hausi A. Muller, Mary Shaw, et al., .: Software Engineering for Self-Adaptive Systems: A Second Research Roadmap, Dagstuhl Seminar Proceedings 2011, pp.1-16, 2011
[4] Betty H.C Cheng, Rogerio de Lemos, Holger Giese, Paola Inveradi, Jeff Magee, et al., .: Software Engineering for Self-Adaptive Systems: A Research Roadmap, Dagstuhl Seminar 2008, pp. 1-26, 2008
[5] Kephart, J.O., Chess, D.M.: The Vision of autonomic computing. Computer 36(1), pp. 41-50, 2003.
[6] John C.Georgas, Richard N.Taylor : Policy-Based Self-Adaptive Architectures: A Feasibility Study in the Robotics Domain: Proceeding of the 2008 international workshop on Software engineering for adaptive and self-managing systems, pp. 105-122, 2008
[7] iRobot Roomba 500 Open Interface (OI) Specification http://www.irobot.lv/uploaded_files/File/iRobot_Roomba_500_Open_Interface_Spec.pdf
[8] A.Kelly, A.Stentz, and O.Amidi: ”Toward Reliable Off Road Autonomous Vehicles Operating in Challenging Envoironments“ The International Journal of Robotics Research Vol.25, No.5-6, pp.449-483, 2006
[9] Shigeo Nakamura, Hiroyuki Nakagawa, Yasuyuki Tahara, Akihiko Ohsuga.: Toward solving an obstacle problem by the cooperation of UAVs and UGVs Proceeding of the 28th Annual ACM Symposium on Applied Computing pp. 77-82, 2013
[10] Hossein Tajalli, Joshua Garcia, George Edwards, Nenad Medvidovic.: PLASMA: a plan-based layered architecture for software model-driven adaptation: 25th IEEE/ACM International Conference on Automated Software Engineering 2010, pp467-476, 2010