2E4-4

動的環境におけるBDIベースのロボットの再プランニングによる再行動決定法

Replanning and Redetermining robot's action based on BDI architecture in the dynamical environment

藤田 恵[*1] Megumi Fujita　片山 寛子[*1] Hiroko Katayama　小島 侑子[*1] Yuko Ojima　新出 尚之[*2] Naoyuki Nide

[*1]奈良女子大学大学院 Graduate School, Nara Women's University　[*2]奈良女子大学 Nara Women's University

Today, robotics technique develops more and more than ever, but the method of equipping robots with intelligent agent is not being researched actively. We focus on low-price robots equipped with the BDI (Belief, Desire, Intention) agents in the real world and are studying the usefulness of this system.

We load a planner which is adaptable to the dynamical environment into the agent so that the agent can make a plan again when the robot can't control its action due to the error of its sensor's perception.

As a result, when the agent has wrong information due to the robot's sensor error, it can update its belief database and have proper information using this mechanism, so the robot can achieve its goal.

1. はじめに

近年、個人でも安価に入手できるロボットの登場に伴い、ロボットの潜在的な応用範囲はますます拡大している。しかし、現状のロボット制御に関する研究は特にロボットの動作自体の円滑さを主体としており、人工知能による知的な動作制御を行う試みは少ない。

人工知能による知的な制御システムによる制御の自動化や、自律的な行動によるロボットの独立化を図るためにも自律エージェントの搭載による制御を行うことは、これからのロボット制御に必要な手段であると思われる。

我々は、自律的で合理的なエージェントの一種である、BDIエージェントを搭載して、ロボット制御を行うことを目指している。

ここで、BDIエージェント [4] とは信念 (Belief)、願望 (Desire)、意図 (Intention) の 3 種類の心的パラメータを用いて、熟考を行い、エージェント自身の行動を意図に沿って構築するエージェントであり、人間の行動決定に基づく意志決定を模倣した形式で自身の行動を決定する。

意図の概念とは、人間における目標達成までの大まかな行動の方針にあたり、BDIエージェントはこの意図をある程度持続して保持することによって、エージェントの行動の一貫性を保つことを保証する。

我々は前回の実験 [11] で、このBDIエージェントを実際にロボットに搭載し、エージェントの挙動を実世界でのロボットに投影して行わせることにより、実世界ではこのエージェントはどれほどの有用性があり、また問題点があるのかを検証した。

ロボットに行わせた課題は、4×5 のグリッド空間で迷路を作成し、それを探索してオブジェクトを発見するというものである。

ロボットは最初は迷路の壁に関する情報を持たず、知覚によって壁の信念情報を追加していく。

この際の問題点としてロボット自体の行動の誤差で、正しく壁を認識できず、誤った信念が信念ベースに蓄積されることとなった。そしてそのまま行動を続けることにより、マップ上での行ける場所を探し尽くしてもオブジェクトに到達できなくなり、エージェントは目標達成ができず、ロボットの制御が不能になってしまうという状態に陥った。

本論文ではエージェントが目標を達成できなくなった場合に、動的環境に適応できるプランナ [10] を用いて再プランニングし、新たな行動戦略をエージェントに与えることで、この誤認識における信念の誤解による制御不能状態を解決する方法とその実装について述べる。

連絡先: 藤田 恵, 奈良女子大学大学院, 奈良市北魚屋西町, 0742-20-3555, saboten@ics.nara-wu.ac.jp

2. 本研究における設計方針及び実装

2.1 Jason と BDI アーキテクチャ

2.1.1 Jason とは

本研究では、BDIエージェントの実装に Jason を使用した。Jason は BDI エージェント記述用言語 AgentSpeak[5] のインタプリタである。

Jason は願望を達成するために信念ベースとプランライブラリを用いて意図を生成し、行動を行い、環境と相互作用することによって新しい知覚を得る。

これを繰り返し、エージェントは願望を達成しようとする。(図 1 参照)

この設計の基となるものが BDI アーキテクチャであり、次のような原理に基づく。

2.1.2 BDI アーキテクチャ

BDIエージェントのインタプリタは一般的に次のようなループで実装される [9]。

```
BDI-initerpreter
initialize-state();
do
    option := option-generator(event-queue, B, D, I);
    selected-options := deliberate(options, B, D, I);
    execute(I);
    get-new-external-events();
    drop-successful-attitudes(B, D, I);
    drop-impossible-attitudes(B, D, I);
until quit.
```

ここで、B は信念、D は願望、I は意図である。

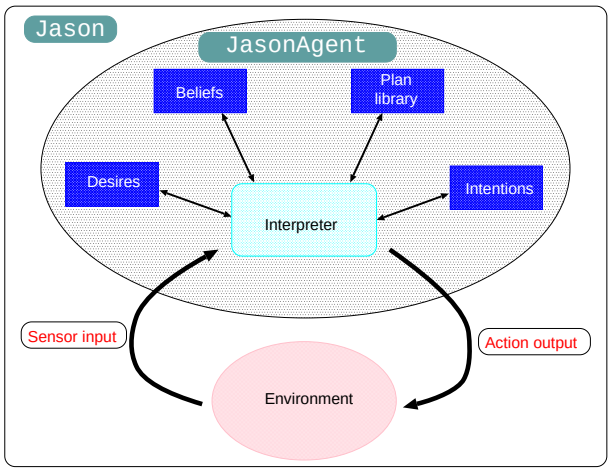

図 1: Jason の模式図

初期条件を与えられると、このループの中で event-queue を読み、行動のリスト、option を返す。次に deliberate を用いて、最適な行動を option から取り出し、それを意図構造体 I に加える。そして、意図 I が原始的な論理式であれば execute で実行する。この後 get-new-external-events により外部の変化を受け取る。ここで、成功した目標と意図を捨てて意図と願望を修正し、次に不可能であった目標や実現不可能な意図を捨ててまた修正し整合性を保つ。これを繰り返すことにより最終的に実現したい願望を達成するという仕組である。

2.1.3 実世界上での問題点

本研究では、この概念に基づく Jason インタプリタを使用した際、Jason のシミュレーション環境では実世界におけるロボットのセンサの誤差を反映することができず、行動の失敗後の対処をすることができなかった。そこで対処法として、次に挙げるプランナと Jason の統合により問題解決を図った。

2.2 動的な環境に対応するプランナ

動的な環境に対応するプランナを搭載することにより、センサの誤認識による誤った信念を訂正し、動的にその場に応じた戦略を与えられるように設計を行い、これを実装した。

具体的には、Jason の環境シミュレータファイルが Java で記述されている点に着目し、環境ファイルの中で動的な環境に対応してプランを生成する、プランナと呼ばれるプロセスを生成し、これにより再プランニングを行い、ロボットのセンサの誤差による誤認識による願望達成不能状態を脱した。

2.2.1 プランナの設計

我々のプランナは階層的プランナで知られる SHOP2[8] というプランナを改造したものであり、プランを組むエンジンとして SHOP2 を使用している。

SHOP2 はプランの組み方が、初期状態から達成すべき目標であるゴールまでのプランをすべて組むことにより、動的環境に適応できない。従って、一度大まかな粒度でプランの分解を中断し、それをサブゴールとして蓄積しておき、エージェントにはそのサブゴールを一つ与え、さらに分解してタスクを生成し、それを実行し、達成できた場合には次のサブゴールを分解し、達成できなかった場合にはその時点の状態を元に再プランニングを行うという設計にした。

2.2.2 エージェントからのプランナの起動

今回の実験では、エージェントプログラムが起動すると同時にプランナのプロセスが生成され、プランナはゴールまでの一連のサブゴールを生成し、その最初のサブゴールを分解してできあがったタスクをエージェントの願望として与えるよう設計した。

その後エージェントはタスクを達成するために意図を生成し、それに従って行動をしていくが、この間、ロボットのセンサ誤差による誤認識が起こり、願望達成不能状態に陥ると、エージェントはプランナに現在の状態を渡し、プランナに新たな行動戦略を与えてもらうようになっている。

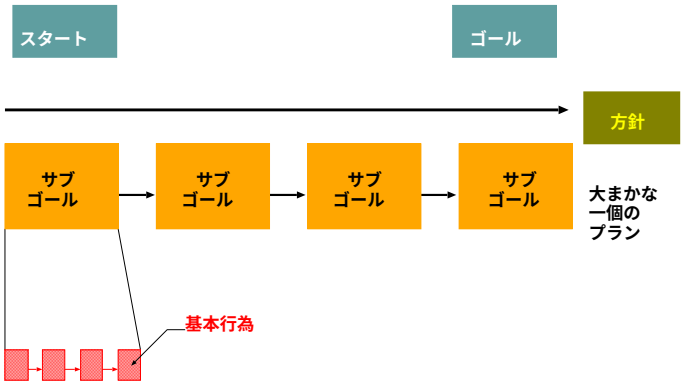

図 2: プランナの模型図

2.3 Python 制御プログラムと本研究で使用したロボット

今回の実験では、エージェントから与えられたロボットへの命令を、Python で記述されたロボット制御プログラムを介して伝達を行い、ロボット制御を行った。

なお、使用したロボットは LEGO 社の MINDSTORMS NXT[3] である。

NXT の構造としては、前輪 2 つと後輪 1 つのタイヤによる移動ロボットになっており、NXT の正面に超音波センサがあり、これにより距離判定を行い、壁の認識のための知覚情報として取り扱う。

また、NXT は Python による既存の制御ライブラリが存在しているので、これを利用してロボット制御プログラムを作成した。

Jason からは「1 マス前進」、「右を向く」、「壁を知覚」などの高位レベルの命令が送られるが、NXT の制御にはモータの回転数などの低レベル命令を用いる。そこで、Jason のエージェントプログラムから高位レベルの命令を Python 制御プログラムに TCP ソケットを介し伝達し、Python 制御プログラム内で低レベル制御命令に変換して、Bluetooth を介して NXT に伝達し、NXT は動作を行うという設計方針で実装を行った。

また知覚情報はロボットのセンサから与えられるものであり、これをロボット制御プログラムに伝達することにより、エージェントの知識に変換し、エージェントが認識を行うように設計した。

これらを実装し、今回迷路を探索させる実験をおこなった。

この設計は図 3 のようになっている。

3. 実験

実験は図 4 のような 4×5 の 2 次元グリッド空間でおこなった。

エージェントは最初、グリッド空間で表されたマップの枠しか知識がなく、ロボットはマップのマス目単位で行動して前方

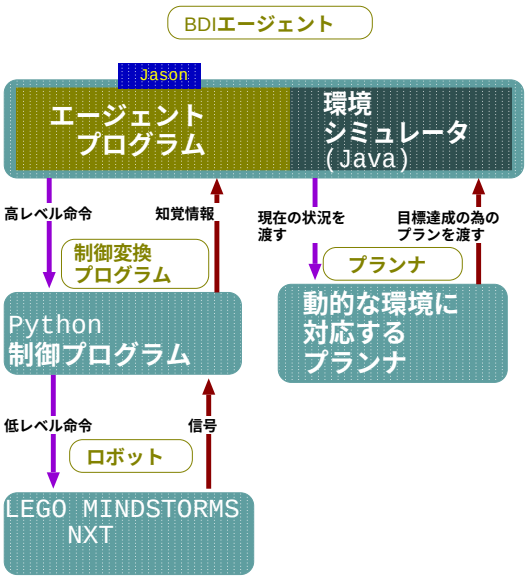

図 3: 設計の全体図

の壁の知覚を行うことにより、エージェントの信念ベースに壁の信念が追加されていく。

今回の実験では 4. 節に示すように、人為的に誤認識を発生させることによって、意図的にエージェントが願望達成をできない状態にし、この状態で先述のメカニズムがいかに働くかを検証した。

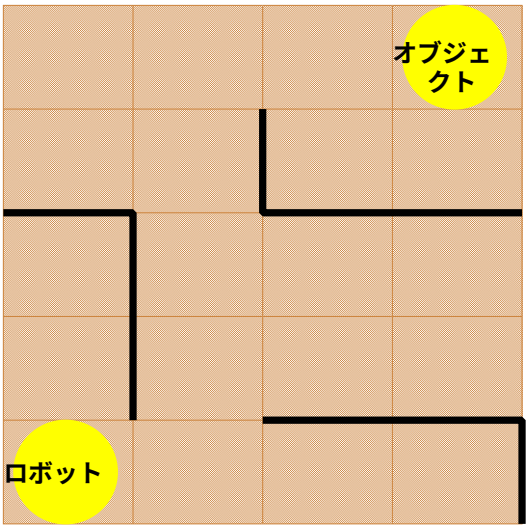

図 4: 実験でのマップ

4. 結果

実験の結果としては、図 5 のようになった。

ロボットは図 4 の迷路を探索していきオブジェクトの探索を始めた。

最初の探索方法は、壁を確認して壁が存在すればエージェントの信念ベースに追加、なければ信念ベースには何も追加されず、その知識を元に探索を進め、その信念が絶対であるという search モードで行われた。

そして探索をする過程において、実験者が通路の一部に壁を意図的に設置し、誤認識を人為的に行わせた。

この結果、ロボットは間違った信念を持ったまま探索を進めて、結果的に探索できる場所が存在しないことを認識し、この結果を元にプランナに問い合わせ、他の戦略である、research モードという自身の信念を疑いながら探索を行うというモードに変更した。

これにより、もう一度すべての壁を探索し始め、現在の信念と照し合せながら行動を行い、間違いを発見して新たな通路を発見すると、その場所の信念を訂正し、その方向へ移動した。そしてその先の未開拓なエリアを探索するという結果となった。

また二箇所通路を塞ぐことをしたときも、探索モードがプランナに問い合わせることにより変更され、全ての方向を探索し始め、塞がれていない通路を見つけ出し、制御不能状態になることを免れることができた。

これらの知見から誤認識における問題を解決することができたと言える。

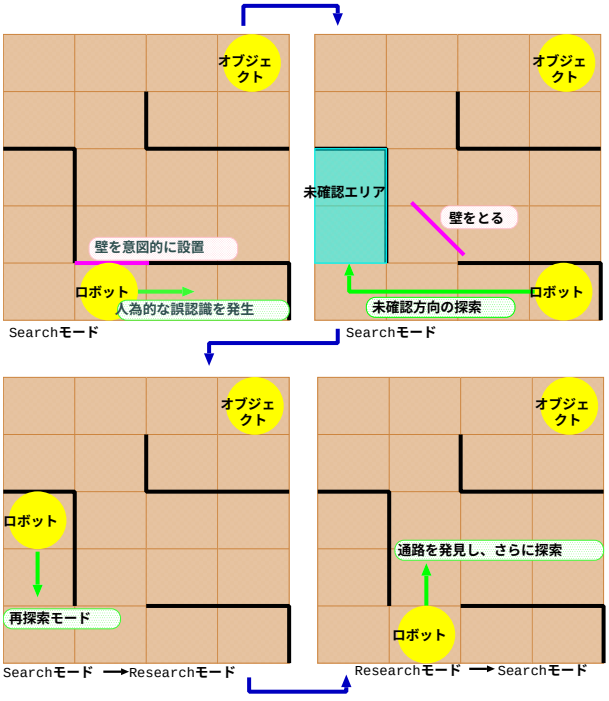

図 5: 実験の結果

5. 考察

今回の実験において、ロボットは誤認識を行ったとしても、動的な環境に適応できるプランナによって再プランニングを行うことにより、仮に何度も誤認識を行っても正しい知識を得ることが可能となった。

実世界上では、環境が動的であることによる行動の失敗や誤認識による信念の誤りはつきものであり、これに柔軟に対応するためにプランナを搭載することは有益であると言える。

現状では、プランナの役割としては誤認識を行って行動ができなくなってしまった場合にのみ使用しており、今後の課題としてはより複雑なプランニングを要する場合に今回の手法を適用し、有用性を検証したい。

また、現時点では現実世界をグリッドワールドとみなすことにより、ロボットの行動が直角方向のみの回転や前進といった単純な行動パターンしか表現できず、知的な振舞と言える行動をロボットに行わせることは難しい。従ってよりエージェント側のシミュレータを充実させ、より複雑なロボットの行動に対応させることにより、BDI エージェントを搭載したことの有用性を明確にすることが必要であると考えられる。

6. 関連研究

BDI エージェントによる NXT の制御例としては、[7] や [6] などがある。

[7] では NXT を NXT-G[1][2] というソフトで制御しており、BDI エージェントは .NET framework を搭載した Visual Studio で構築している。

NXT-G は Windows のみでしか使用できないので、開発環境に制限がかかってしまう。これに伴い、本研究における開発は UNIX システム上で行っているが、OS 依存のライブラリ等を使用せず、利用環境の制約がかからないようになっている。

また、[7] や [6] はロボットの制御方法が光センサを活用することにより、ライントレースを行って動作をするようになっているが、現実の応用ではあらかじめ設置したマーカーを使ってライントレースを利用できるとは限らない。

我々の研究ではロボットの動作により現実性を持たすために、タイヤで回転してマス目空間を移動しながら壁を知覚して動作を行うという実装にした。

さらに [7]、[6] は BDI エージェントの観点から見て、与えられた願望に対して取得する意図が決定的であり、特に [6] は願望に対してひとつの意図を対応させており、その意図もひとつの原始的行動のみで構成されている。

しかし、意図とは本来エージェントの行動の方針として扱われるべきであり、その中身は大まかなプランから構成される行動列から形成されるものである。従って、願望に対しての意図が固定されていると柔軟な行動決定ができない。

これに対し、今回の我々の実験では、Jason プラットフォーム上で BDI エージェントに特化したプログラミング言語 AgentSpeak を用いてエージェントを実装することで、目的を達成するための意図を願望に対し固定するのではなく、現在の環境に関する信念を用いた熟考で決定でき、また必要に応じて意図の再考慮を行うなど、より洗練された意図を保持することができる。

さらに、動的環境に対応できるプランナを取り付けたことにより、実世界における誤差に対しても有効な行動戦略に変更することができ、より意図の決定に柔軟性を持たせることができるようになっている。

7. むすび

今回の実験においては、実世界におけるロボット上でのエージェントの在り方を確認するという意味で、現実世界の誤差や失敗を取扱い、これに対応可能とするエージェントの設計及び実装を行った。

今後の課題は、プランナの設計の見直しを図り、より現実世界に対応がとれるようなプランニングシステムの構築を行う予定である。

現状のプランナのシステムとしては、サブゴールの順序が定まっており、先頭のサブゴールから順次タスクとしてエージェントに渡されることになっている。そして、そのタスクが達成されるとサブゴールは消去されることになっている。

しかし、なんらかの外部環境の影響で、すでに達成されたサブゴールにあたるプランをもう一度与えることが必要な場合、このシステムではその状況を理解することができない状態にある。従って、現在のサブゴールの在り方の見直しをする必要がある。

また、シミュレーション環境を充実させることにより、現状での単純な行動からより複雑な行動をシミュレータの段階で行わせることにより、現実世界で生じる摩擦からくる誤差などの問題に柔軟に対応できるようにすることを目指す。

参考文献

[1] Lego mindstorms nxt. `http://www.nebomusic.net/NXT-G-DriveASquare.html`.

[2] Nxt tutorial. `http://www.ortop.org/NXT_Tutorial/index.html`.

[3] Lego mindstorms nxt: Welcome to the next generation. `http://www.tik.ee.ethz.ch/tik/education/lectures/PPS/mindstorms/sa_nxt/index.php?page=print`, 2006.

[4] Rao Anand S and Michael P. Georgeff. Modeling rational agents within a bdi-architecture. *Proc, of International Conference on Principles of Knowlegde Representation and Reasoning*, 1991.

[5] Rafael H. Bordini, Jomi Fred Hübner, and Michael Wooldridge. *Programming multi-agent systems in AgentSpeak using Jason*. WILEY, 2007.

[6] Soren Andreas Juul, Kaspar Henrik Moss Lyngsie, Michael Vandborg, Kim Fiedler Vestergaard, Torsteinn Sævar Hjartarson, and René Bach Gustafson. Agent programming — robot traffic. 2008.

[7] Anders Lemke, Johan Laidlaw, and Lars Zilmer-Pedersen. Developing multi-agent lego robotics. 2007.

[8] Dana Nau, Tsz-Chiu Au, Okhtay Ilghami, Ugur Kuter, J. William Murdock, Dan Wu, and Fusun Yaman. Shop2: An htn planning system. *Journal of Artificial Intelligence Research*, 20:379–404, 2003.

[9] Munindar P. Singh, Anand S. Rao, and Michael P. Georgeff. Formal method in dai. *Multiagent systems: amodern approach to distributed artificial intelligence*, 1999.

[10] 藤田 恵 and 新出 尚之. 動的な環境に対応する BDI エージェントのためのプランナの設計と実装. *Joint Agent Workshops and Symposium 2008*, 2008.

[11] 藤田 恵, 小島 侑子, 片山 寛子, and 新出 尚之. 実世界での BDI ベースのロボットの柔軟な行為決定の実現に向けて. *Joint Agent Workshops and Symposium 2009*, 2009.