フォール トトレランスを考慮した人間とロボットの協調系の構築

Human-robot Cooperative System with Fault Tolerance

望月恒治、下倉雅行、上野敦志、武田英明、西田豊明
Koji Mochizuki, Masayuki Shimokura, Atsushi Ueno, Takeda Hideaki, Toyoaki Nishida

奈良先端科学技術大学院大学 情報科学研究科
Graduate School of Information Science, Nara Institute of science and technology

Abstract: Disadvantage of cooperative system with multiple robots is that fault of any robots may cause unexpected effects to the whole system. In this paper, we propose a method to solve the above problem. If the robot doing a task happens a trouble, the task is succeeded to other robots as possible or continued by the robot which is restored afterword. On this method, status in the environment is managed by Environment Agent. Restoration is supported by Confirm Agent. The method is adopted to the exvironment in which various kinds of autonomous robots and people coexist. When one of robots has a trouble, the system generates a new plan without the troubled robot or a request to human to repair the robot.

1 はじめに

近年、ビルの窓拭きロボットのようにロボットが身近に見られるようになった。今後、生活環境にもさまざまな、複数のロボットを導入する事が考えられる。

そのように、複数のロボットが稼働している中、あるロボットが何らかの原因により停止した場合に、その影響でシステム全体が停止する事になれば、非常に効率が悪い。しかし、その場に同じタスクをこなす事のできるロボットが存在するならば、そのロボットに停止した仕事を引き継がさせれば効率が良くなる。また、ロボットを復旧した際、系に戻すのに再構成をしなくても済むのであれば、全システムを一度停止してから組み直さなくてもよくなるので、人の手を煩わす事も少なくなる。

本研究の目的は、システム内の個々のロボットに障害が発生しても、システム全体としてはタスクを可能な限り遂行できるシステムを構築する事にある。もう一つの目的は、障害が発生したロボットがシステム全体を停止する事なく、自己あるいはシステム中の他のエージェントによって、障害から復旧し、自然にシステムに戻ることができる事にある。

この様な環境を構築するために、人間にロボットの復旧依頼を行い、そして復旧されたかどうかを確認するためのエージェントを新たにシステムに付加する事が有効であると思われる。このことを実証するため、当研究室で構築されたKappaシステムのエージェントに変更を加え、実験を行った。

2 フォール トトレラントな協調系

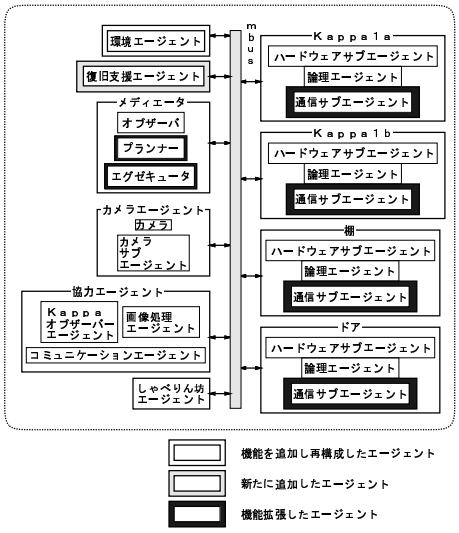
ここでいう、「フォール トトレランス」とは、1つ目に、あるロボットに何らかの障害が発生した場合でも、他のロボットによって可能な限りタスクを継続して実行できる、という事である。2つめに、障害が発生したロボットは、その障害を人間あるいは環境中の別のエージェントに伝え、一刻も早く障害を取り除いてもらい、再び元の環境に戻る事ができる事を、指している。

3　システムの概要

今回の研究では、Kappa システムに機能の追加、拡張を行った。Kappa システムとは、当研究室で開発されている複数のロボットによる、協調システムである [1]。本研究を開始する時点で実現されていた協調作業は、異種のロボットと人間が混在する中、人間からの非同期に要求される作業をロボットによる柔軟な協調実行が可能であった [2]。また、人間からのジェスチャーによる意思の伝達方法が考案されている [3]。

環境エージェントはもともとウォッチャというエージェントから再構成されたものである。また、復旧支援エージェントは本研究で新たに作成されたエージェントである。通信サブエージェントは機能の拡張を行ってある。

本研究で追加されたエージェントを含む、全体の構成を下に示す。



4　フォールトトレランスのためのエージェント

4.1　通信サブエージェント

今まで使われていた通信サブエージェントに、状態管理機能を追加した。状態管理の方法としては、知識表現としていくつか定義されているもの (表 1) を用いる。

知識表現	状態の意味
(from A)	A から出発した
(to A)	A に向かっている
(at A)	A にいる
(carry B)	B を持っている
(in-front-of C)	C の前にいる
(task D)	D というタスクを実行中
(open)	ドアが開いている
(close)	ドアが閉まっている

表 1: 状態管理のための知識表現

例えばロボットが Wait Point の W1 から W4 へ移動している時で何ものを持っていないというような状態では,

```
((from w1)(to w4)
(carry nothing)(task move))
```

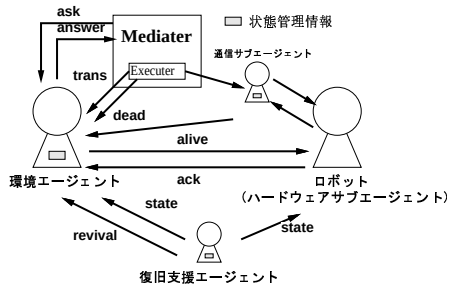
という形で管理される. この情報は移動によって位置情報が書き換えられ、物を持ってるとなればそれに対応する情報が書き換えられる。人間とのコミュニケーションを行う場所等の特別な位置ではその情報も書き加えられる。

4.2　環境エージェント

環境エージェントは、これまでの研究で用いられてきたウォッチャの機能を全て引き継ぎ、フォールトトレランスを実現するために機能の拡張を行った。このエージェントのの役割は、ロボットの動く環境に関する情報の管理である。すなわち、各ロボットの状態、タスクの内容、物の所在地である。また、カメラエージェントや行動型エージェントに依頼されたタスクに応じて、メディエータを起動する。タスクの実行状態や使用されているエージェントについても、このエージェントが管理している。

環境エージェントの状態管理ポリシーは、“ロボットに動きがある毎に、Executer から状態遷移メッセージ (trans) を環境エージェントに発信する” である。また、古い情報の更新には、各状態遷移情報に付け加えられているタイムス

タンプの情報を利用する。



ここでの状態管理は、通信サブエージェントの管理方法とは異なり、この様なメッセージがやり取りされる。

```
(trans (kappa1a (from w7)(to w14)
(carry nothing)(task move)))
```

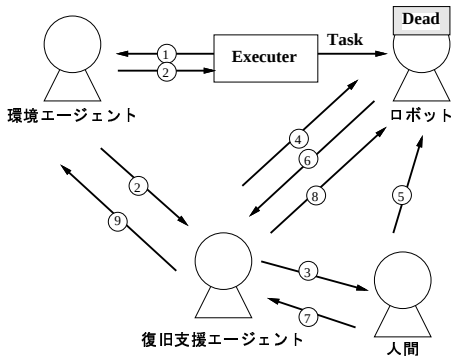
この意味は、「kappa1a が Wait Point W7 から W14 へ何も持たずに移動している」という事を示している。

メッセージ	内容
trans	状態遷移
ask	状態問合せ
answer	状態返答
alive	生存問合せ
ack	返答
dead	障害発生
revival	復旧通知
state	復旧後の状態

表 2: 状態管理メッセージ

4.3 復旧支援エージェント

このエージェントは、ロボットからの反応が無くなった時に、すなわち、なんらかの障害が発生したと思われる時に、起動される。ロボットの復旧を人間に促すメッセージを流し、復旧されるまで監視している。このエージェントは障害が発生したと思われるエージェント1つに対して1つ起動し、復旧したさいにはそのロボットの現在位置とタスク内容を人間から教えてもらい、それらの情報を障害から復旧したロボットの通信サブエージェントと環境エージェントに引き渡して、仕事を終える。



このエージェントは自分だけで復旧作業を行わずに、復旧支援作業のみを行う。人間はこのエージェントが示してくれる、応答の無くなったロボットの位置情報を基に復旧作業を行い、復旧支援エージェントに復旧したロボットの場所を伝える。復旧支援エージェントはその位置情報と、状態情報をロボットと環境エージェントに知らせ、作業を終える。

5 障害発生と復旧

5.1 障害の発生判定

障害が発生した場合には、状態の更新がされない、メッセージを送っても反応が無い、という状態になる。これを利用し、本研究では障害発生の判定を行っている。状況としては、以下の2つを想定している。

問い合わせに対して反応が無い 移動中のロボットに対して状態の問い合わせがあり、前回の問い合わせからある程度の時間が経っていた場合、その移動中のロボットに状態の問い合わせを行う。問い合わせに対して更新情報が送られてこない場合、再度問い合わせ、それでも反応が無い場合は移動中に障害が発生した可能性があると環境エージェントは判断し、そのロボットに対して確認メッセージ(alive)を送る。障害が実際には発生していない場合はロボットから返答メッセージ(ack)が返されるが、障害発生時には返答が無い。これを利用して判定を行う。

障害発生メッセージの送信　実行中にExecuterから障害発生のメッセージ(dead)が環境エージェントへ送られてくる場合である。Executerがメッセージを送るのは、やはりロボットからの反応が無い場合である。

5.2　障害からの復旧

ロボットに障害が発生した場合、速やかな復旧を行わなければならない。このため、本研究で開発したシステムでは以下の方法を取る。

1. 障害の発生したロボットの状態を環境エージェントから復旧支援エージェントに送る。
2. そのロボットを動かすために起動していたメディエータを終了させ、ロボットのいる経路を通行不能とする。
3. タスクの引き継ぎを試みる。引き継ぎができた場合は以下の処理を行わない。
4. 復旧支援エージェントはロボットの復旧をチェックする。
5. 復旧支援エージェントから復旧メッセージが届いたら、そのロボットをタスク実行可能エージェントとし、状態情報を元に戻す。前に実行していたタスクを継続するようにメディエータを起動して、復旧処理を終了する。

ロボットの復旧後、復旧支援エージェントから復旧した旨のメッセージ(revival)が送られてくる。そして、人に教えてもらった現在位置を追加した状態情報が復旧支援エージェントから送られてくる。これをもとに、環境エージェントは状態情報を書き換える。

6　実験、評価

本研究の実験として、実際に発生する可能性のある障害を疑似的に発生させて、その上で安定して動作することを確認した。障害としては通信が届きにくい場所があり、その場所で通信が途切れてしまった事を想定した。

今までのシステムでは、一つのロボットが停止してしまったり、メッセージを一つ取りこぼすだけで全てのシステムに影響を及ぼし、全体が停止する事がよく発生していた。

しかし、本研究で構築したシステムではこれらに対応できるようになった。これにより、障害発生時に障害が発生したロボットのみに影響をとどめる事ができ、ロボットが復旧すればシステムに戻る事が可能になった。また、障害の復旧を人間に行ってもらう場合に、障害発生地点の大まかな場所を知らせる事ができるので、復旧支援者にとって非常に探しやすいと思われる。また、ロボットが停止している経路のみを通行停止にすればよいので、通路を塞がれる事によるプランニングの失敗が少なくなる。

7　まとめ

本研究では、複数のロボットが動く中であるロボットに障害が発生した場合に、復旧作業を人間と協調し、その障害が発生したロボットが実行していたタスクを継続させる事のできる協調環境を構築した。

本研究で開発した状態管理の手法を使うことにより、障害発生の位置と状態を明確に伝える事ができ、復旧時に状態を復元する事も可能になった。

参考文献

[1] 武田英明、小林展英、松原慶幸、西田豊明 :人間 機械の共生のための知識的環境の提案と実現の試み、1997年度電子通信情報学会 情報・システム ソサアティ大会 併催シンポジウム 「ソフトウエアエージェントとその応用」 シンポジウム講演論文集、pages 147-154,1997

[2] 松原慶幸、上野敦志、沢田篤史、武田英明、西田豊明 :人を含んだ実世界エージェントの協調系におけるプランニングと実行、人工知能学会人工知能基礎論研究会 SIG-FAI-9603, pages 12-17, 1997.

[3] 小林展英、上野敦志、沢田篤史、武田英明、西田豊明 :人間を含んだ実世界エージェントの協調のための人間–ロボット間のコミュニケーション、人工知能学会人工知能基礎論研究会SIG-FAI-9603, pages 18-23, 1997