Vol.2012-MBL-64 No.2
Vol.2012-ITS-51 No.2
2012/11/15

モバイルエージェントプログラムの生成による格子状に接続されたユビキタスコンピュータ群の制御

國本　慎太郎[1,a)]　藤田　直生[1]　佐野　渉二[1]　寺田　努[1,2]　塚本　昌彦[1]

概要：筆者らの研究グループでは，格子状に配置したユビキタスコンピュータ群に対して，それぞれのユビキタスコンピュータに組み込まれた入出力デバイスをモバイルエージェントを用いて制御するシステムを構築してきた．モバイルエージェントを実行すべきコマンドの羅列であると捉え，モバイルエージェントの移動，並列処理，入出力制御などの機能をもつコマンドを作成した．本稿では，グローバルかつトポロジカルなプログラム言語からモバイルエージェントを生成することで，マクロな制御を可能にする手法を提案する．

キーワード：ユビキタスコンピューティング，モバイルエージェント，格子状ネットワーク

Controlling Ubiquitous Computers in Grid Topology by Generating Mobile Agents Programming

Kunimoto Shintaro[1,a)]　Fujita Naotaka[1]　Sano Shoji[1]　Terada Tsutomu[1,2]
Tsukamoto Masahiko[1]

Abstract: We have proposed a method using mobile agents to control I/O devices on ubiquitous computers in grid topology. By using a mobile agent program consisting of a set of commands, a user can perform various functions, such as migration, parallel processing, and I/O control by using these commands. In this paper, we propose the macroscopic control by generating mobile agents from global and topological program languages.

Keywords: Ubiquitous computing, Mobile agent, Grid network topology

1. はじめに

近年，情報機器の小型，軽量化，低価格化に伴い，小型のコンピュータが生活環境のいたるところに埋め込まれたユビキタスコンピューティングの実現が期待されている．ユビキタスコンピューティング環境では多数のコンピュータを制御することが望まれるため，個々のコンピュータへのプログラミングよりコンピュータ群全体へのプログラミングが求められる．筆者らの研究グループでは，格子状に配置したユビキタスコンピュータ群に対して，それぞれのユビキタスコンピュータに組み込まれた入出力デバイスをモバイルエージェントを用いて制御することで，ユビキタスコンピュータ群全体の入出力制御を行う手法を提案している [1]．モバイルエージェントを実行すべきコマンドの羅列であると捉え，モバイルエージェントの移動，並列処理，入出力制御などの機能をもつコマンドによって，並列処理や環境内のコンピュータ数の変化へ対応できるようにした．しかし上述のシステムでは，汎用的なふるまいを行うコマンドを多数作成し，それらを組み合わせたコマンドの羅列でさまざまなプログラミングを可能にしているた

[1] 神戸大学大学院工学研究科
Graduate School of Engineering, Kobe University, Kobe 657-8501, Japan
[2] 科学技術振興機構さきがけ
PRESTO, Japan Science and Technology Agency, Chiyoda, Tokyo 102-0075, Japan
[a)] s-kunimoto@stu.kobe-u.ac.jp

め，コマンドの数が多く，それゆえ，慣れていなければ覚えることが難しいという問題がある．

そこで本稿では，グローバルかつトポロジカルなプログラム言語からモバイルエージェントを生成することで，より容易にマクロな制御を可能にする手法を提案する．このことにより各コンピュータでの挙動を意識することなくマクロな視点でモバイルエージェントを用いたプログラムが作成できるようになり，また，プログラムがコンピュータ群全体でどのようなふるまいを行うかが把握しやすくなった．

以下，2 章で関連研究について説明し，3 章で我々が行ってきた研究概要について述べ，4 章でモバイルエージェントを生成する言語について説明し，最後に 5 章で本研究をまとめる．

2. 関連研究

ユビキタスコンピューティング環境を実現のため，さまざまな研究が行われている．寺田ら [2] はルール制御に基づくユビキタスコンピュータを提案し，それぞれのユビキタスコンピュータは組み込まれたルールにより入出力機器を制御する．MOTE[3] は，自発的にアドホックネットワークを形成し，マルチホップ機能を備えた小型デバイスであり，nesC と呼ばれる C 言語を拡張したプログラミング言語を用いて制御プログラムを記述する．しかし，これらは個々のコンピュータごとに制御プログラムを記述するため，複数のコンピュータにまたがる処理を行う場合，それぞれの制御プログラムで整合がとれるように記述する必要があり，労力と時間に関する開発コストが大きい．

ユビキタスコンピュータ群の制御の研究として，多数のコンピュータをあたかも 1 つのユビキタスコンピュータを扱うような記述で制御するマクロプログラミングがあり，多くの研究がされている [4], [5], [6]．Kairos[4] は，コンピュータ群に対して，複数のコンピュータに及ぶ処理やコンピュータ間のトポロジを用いた処理を 1 つのプログラムで記述できる．Regiment[5] は関数型プログラミングの概念を取り入れており，センサデータを扱った再帰的な処理を容易に行える．RuleCaster[6] は各コンピュータの動作をルール形式で記述してコンピュータ群を制御するシステムである．複数のコンピュータに及ぶ処理も 1 つのルールで定義でき，複数のルールを用いることで一度に多くの処理を行える．しかし，これらのシステムは個々のデバイスへのプログラミングの延長であり，各ユビキタスコンピュータの ID を指定しながら処理を記述する必要があったり，記述したプログラムから各ユビキタスコンピュータへのプログラムを生成する場合，ユビキタスコンピュータの数や配置に依存するため，ユビキタスコンピュータ数やネットワーク構造が変化する場合，プログラム自体を更新しなければならない．モバイルエージェントに関する研究も多数行われている [7], [8], [9], [10]．Agilla[7] ではプログラマはミドルウェアによって提供される API を利用してセンシングデータを取得するプログラムを記述する．これにより，プログラムを動的にノードに配備することができ，センシング情報に応じてプログラムを動的に配備可能である．Yu-Chee Tseng[8] らや Yingyue Xu[9] らは無線センサネットワークにおいてモバイルエージェントを用いてターゲットを追跡する手法を提案している．

格子状ネットワークについては，ノード間のリンクの有無を制御することで擬似的にマルチホップする形のテストベッドである ORBIT[11] など無線センサネットワークやメディア・アクセス制御の分野で主に研究されている．Chen Zhang ら [12] は，単純なアルゴリズムをローカルにホップさせていくことでセンサネットワーク中でのルックアップアルゴリズムを小さなエネルギーで動作させている．一般的に，格子状ネットワークのような規則的な構造は均一なノードのパターンを提供し，ローカライズされたプログラムはパターンを繰り返すことによってグローバルに展開することができる．更に，格子状ネットワークは，与点からのホップを数えることによりノードの位置を計算することを簡単にする．

3. モバイルエージェントを用いた制御システム

3.1 アプローチ

本研究では隣接するユビキタスコンピュータとの通信機能をもち，入出力機器の制御を行うユビキタスコンピュータが環境内に数百から数千個程度，**図 1** のように格子状に接続されている環境において，モバイルエージェントを用いてユビキタスコンピュータ群を制御する．モバイルエージェントは，コンピュータ間の移動性をもつプログラムで，あるコンピュータで実行している処理状態を記録し，他のコンピュータで引き続いて処理を行えるため，複数のコンピュータにまたがる処理をするプログラムも容易に作成できる．また，モバイルエージェントは非同期性をもつため，制御プログラムを更新する際，ユビキタスコンピュータのファームウェアを更新する必要がなく，新たなモバイルエージェントを実行すればよい．

格子状ネットワークのコンピュータ群を制御するためには，規則的な配置を活かし，個々のコンピュータで行う処理をグローバルに展開していくことで，コンピュータ群全体を制御することが有効である．本研究では，モバイルエージェントが各コンピュータを移動しながら同じパターンの処理を繰り返すことでコンピュータ群を制御できるよう，モバイルエージェントを実行すべきコマンドの羅列ととらえ，モバイルエージェントの移動や入出力制御など基本的な機能をもつコマンドを作成した．

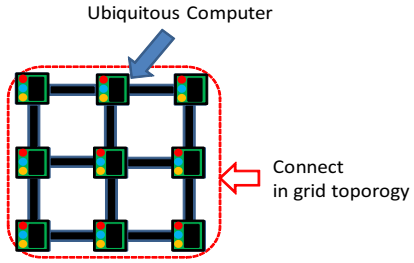


図 1 格子状ネットワーク
Fig. 1 Connected in grid topology.

表 1 進行コマンドと進路変更コマンド
Table 1 Migration commands.

F	進行方向にコマンド列を移動させる.
R	進行方向を右に変更する.
L	進行方向を左に変更する.
B	進行方向を後ろに変更する.

3.2 コマンド

コマンドはユビキタスコンピュータ間の通信量を下げるためにプログラムが短くなるよう大文字のアルファベットや記号で表している．プログラムは 1 文字ずつ処理され，大文字や記号でコマンドを特定する．また，アルファベットの小文字や「(」や「;」，「)」の記号を用いて，コマンドの機能の詳細を決定する．

以下，作成したコマンドの動作について説明する．ここで，n，m，lはそれぞれ 1Byte の符号なし整数，2Byte の符号なし整数，0 から 9 の整数，vはエージェント変数として g から w の 17 種類，ローカル変数として x，y，z の 3 種類，wは配列型のエージェント変数として a，b，c の 3 種類，ローカル変数として，d，e，f の 3 種類，Zは任意のコマンド列とする．

進行コマンドと進路変更コマンド

表 1 に進行コマンドと進路変更コマンドを示す．モバイルエージェントは進行方向として，前，後，左，右のいずれかの向きをもち，これらのコマンドにより，モバイルエージェントの移動を制御するためのコマンドである．

進行コマンド「F」は「F」より後ろのコマンド列を現在の進行方向先のユビキタスコンピュータに移動させるコマンドである．

進路変更コマンドの「R」，「L」，「B」は，モバイルエージェントの進行方向を現在の進行方向に対してそれぞれ右，左，後ろに変更させるコマンドである．

入出力制御コマンド

表 2 に入出力制御コマンドを示す．このコマンドはユビキタスコンピュータに搭載されている入出力機器の制御行うためのコマンドである．入力機器として，可視光センサ，赤外光センサ，出力機器として，フルカラー LED をユビキタスコンピュータに搭載することを想定して，コマンドを作成している．

表 2 入出力制御コマンド
Table 2 I/O control commands.

Ck	色を指定して，フルカラー LED を発光させる.
Mkv	赤外光センサ，可視光センサを k で指定して，そのセンサ値を v に代入する.
Mkw	赤外光センサ，可視光センサを k で指定して，そのセンサ値を w の最後尾のデータに代入する.

表 3 繰り返しコマンド
Table 3 Repetition commands.

$m(Z)$	Z を m 実行する.
$*(Z)$	Z を実行し続ける.

出力制御コマンド「C」はフルカラー LED を指定する色に発光させるコマンドである．Ck では k として r，g， b，m， y， c， w のいずれかを設定することで，それぞれ赤 (red)，緑 (green)，青 (blue)，紫 (magenta)，黄 (yellow)，シアン (cyan)，白 (white) に発光する．

入力制御コマンド「M」は可視光センサ，赤外光センサのセンサ値を変数に代入する．Mk_1v の k として，可視光 (visible) センサを用いるときは v，赤外光 (invisible) センサを用いるときは i を設定することで，そのセンサ値を変数 v に代入する．また，Mk_1w とすると，配列 w の最後尾のデータにセンサ値を代入する．

繰り返しコマンド

表 3 に繰り返しコマンドを示す．このコマンドはコマンド列の繰り返し処理を行うコマンドである．繰り返したいコマンドの入力時間，量を短縮し，繰り返し回数の設定も容易に行える．整数 m を用いて，$m(Z)$ とすると Z を m 回繰り返して実行し，「*」を用いて，$*(Z)$ とすると Z を繰り返して実行し続ける．本稿では，前者を有限繰り返しコマンド，後者を無限繰り返しコマンドと呼ぶ．

変数コマンド

表 4 に変数コマンドを示す．変数としては，モバイルエージェントが保持し，他のユビキタスコンピュータに移動してもそのモバイルエージェントが利用できるエージェント変数と，モバイルエージェントが保持せず，他のユビキタスコンピュータに移動しないローカル変数を設けた．

変数設定コマンド「S」は「Svn」，「Svv_1」，「S$v(Z)$」とすることで，変数 v の値をそれぞれ整数 n，変数 v_1，コマンド列 Z に設定する．「S$wl{:}n$」で配列 w の l 番目のデータ (以降，$w(l)$ と表記する) を n に設定する．また，「S$w{:}n$」とすると，配列 w の最後尾のデータに n を新たに追加する．「S$wv{:}v_1$」とすると，配列 w の v 番目のデータを変数 v_1 の値に設定し，「S$wl{:}w_1l_1$」とすると，$w(l)$ を $w_1(l_1)$ に設定する．

変数増減コマンド「I」，「D」は，「Ivn」で変数 v の値を n 増加させ，「Dvn」で変数 v の値を n 減少させる．

表 4 変数コマンド
Table 4 Variable commands.

Svn	v を n に設定する.
S$v(Z)$	v を Z に設定する
Swl:n	w の l 番目を n に設定する.
Ivn	v を n 増加させる.
Dvn	v を n 減少させる.
Uv	Uv を変数 v に書き換える.

表 5 分岐コマンド
Table 5 Branch commands.

$(Z_1;Z_2;\cdots;Z_i)$	コマンド列を移動可能な方向へ移動させる
E$vkn(Z_1;Z_2)$	v が kn で指定した条件を満たせば Z_1 を, 満たさなければ Z_2 を実行する.

変数書き換えコマンド「U」は「Uv」とすると，Uvを変数 v の値に書き換えるコマンドである．変数 v に n を設定していれば，「Uv」は n に書き換えられ，変数 v にコマンド列「Z」を設定していれば 「Uv」は Z に書き換えられる．

並列処理コマンド並列処理コマンドはモバイルエージェントをコピーして分岐させることにより複数のコンピュータにおける並列処理を可能にするためのコマンドである．「$[Z_1;Z_2;\cdots;Z_i]$ Z_{i+1}」により Z_1Z_{i+1},Z_2Z_{i+1},$\cdots$，Z_iZ_{i+1} を順に実行する．

分岐コマンド

表 5 に分岐コマンドを示す．方向分岐コマンドは，進路変更コマンドを用いて指定した進行方向先にユビキタスコンピュータが接続されているかを判断し，接続されている場合，進行する行うコマンドである．方向分岐コマンドを用いた「$(Z_1;Z_2;\cdots;Z_i)$」では，現在の進行方向から Z_1 中で設定された進行方向にユビキタスコンピュータが接続されているかを調べ，その方向に進行可能なら Z_1 を実行し，進行不可能なら Z_2 が同様に実行可能かを調べる．Z_2 が実行可能であれば，Z_2 を実行し，進行不可能なら Z_3 を評価して実行できれば実行する．以下，同様にして，どれか 1 つを実行するか，Z_i まで実行できなかったときに処理を終了する．

変数分岐コマンド「E」は，E$vkn(Z_1;Z_2)$ とすることで，変数 v の値と n の値を k で設定した条件で評価し，評価結果に応じた処理を実行するコマンドである．k は e，n，g，l，b のいずれかを設定する．k が e，n，g，l の場合は，それぞれ v が n と等しい (equal)，v が n と等しくない (not equal)，v が n より大きい (greater)，v が n より小さい (less) ことを示す．b のときは「Evbn_1:$n_2(Z_1;Z_2)$」により，v が n_1 より大きく，n_2 より小さい (between) ことを満たせば Z_1，満たさなければ Z_2 を実行する．

その他のコマンド

表 6 にその他のコマンドを示す．待機コマンド「W」は，待機時間を設定するコマンドである．例えば，Z_1WmZ_2 のコマンド列の処理を行う際，Z_1 を処理した後，Wm で m を設定することで，$m \times 100$ms 間待機してから Z_2 の処理を行う．

表 6 その他のコマンド
Table 6 Miscellaneous commands.

Wm	次のコマンドの処理を行うまで $m \times 100$ms 待機する.
Pm	LED の発光時間を調整する.
:	コマンドを区切る.
K	エージェントの強制終了を行う.

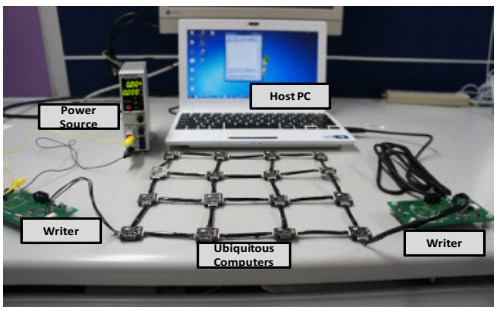


図 2 システム構成
Fig. 2 The system configuration.

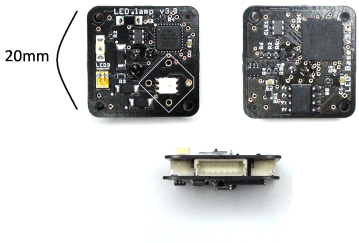


図 3 制御対象の小型デバイス
Fig. 3 Our developed ubiquitous computer.

点滅コマンド「P」は，LED の発光時間を調節するコマンドである．例えば，出力制御コマンドと組み合わせて，CkPmZ_1 のコマンド列の処理を行う際，LED が $m \times 100$ms 間発光させ，消える．その状態のまま $m \times 100$ms 間待機したあと Z_1 の処理を行う．

区切りコマンド「:」は，複数の数字が並ぶときにそれらを区別するためのコマンドである．例えば，v を n に設定し，Z を m 回繰り返すようなコマンド列は「S$vnm(Z)$」となり，nm を一つの整数 t とした S$vt(Z)$ と区別することができない．そのため区切りコマンドを用いて「Svn:$m(Z)$」とすると，意図した動作を行わせられる．

終了コマンド「K」は，それ以降のコマンド列を消去し，モバイルエージェントを消滅させるためのコマンドである．KZ とすると K 以降のコマンド列 Z は実行されない．

3.3 実行環境と制御対象

図 2 にシステム構成を示す．制御対象であるユビキタスコンピュータ，ユビキタスコンピュータの接続線，ユビキ

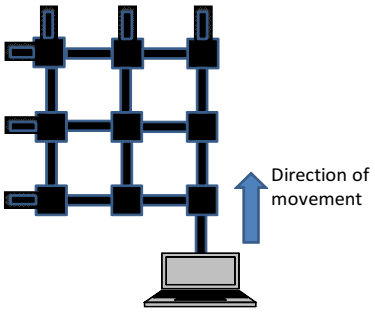


図 4 モバイルエージェントの向き
Fig. 4 Direction of mobile agent movement.

タスコンピュータの書き込み機（ライター），シリアル通信用アプリケーションは筆者らの研究グループで作成したものである．ホスト PC とシリアル通信を行うため，USB ケーブルとライターを接続し，ライターと格子状のユビキタスコンピュータ群を接続する．小型デバイスにバッテリは搭載していないため，電源を同じく別のライターに接続してライターから格子状のユビキタスコンピュータ群と接続する．PC から送信すると，最初に PC とつながっているユビキタスコンピュータへ送られ，そこから処理が始まる．このとき，**図 4** のようにユビキタスコンピュータが下の通信ポートからモバイルエージェントを受信するとき，モバイルエージェントは上方向の向きを持つ．同様に PC からモバイルエージェントを受信する通信ポートが上，左，右のときはそれぞれ下，右，左の進行方向をもつ．制御対象は**図 3** に筆者らの研究グループで開発した小型デバイスを用いる．大きさは 20mm 四方で，マイコンには Atmel 社製ワンチップマイコンである AVR を使用し，1MB のフラッシュメモリ，フルカラー LED，可視光センサ，赤外光センサを搭載し，通信線で隣接するユビキタスコンピュータと 4 方向までシリアル通信ができる．ユビキタスコンピュータにはコマンド列の処理を行うために，コマンドを解釈して実行するエンジンを格納させている．

4. モバイルエージェントの生成によるコンピュータ群制御

本研究では，汎用的なふるまいを行うコマンドを多数作成し，それらを組み合わせたコマンドの羅列でさまざまなプログラミングを可能にしている．コマンド羅列を用いることで，どのコンピュータでどのコマンドが処理されるかが把握しやすい．しかしながら，コマンドの数が多いため慣れていなければ覚えることが難しかったり，コンピュータ間の通信量や負荷を下げるためにプログラムが短くなるよう文字や記号の羅列であらわされるため，プログラムがコンピュータ群全体でどのようなふるまいを行うかが分かりにくいという問題がある．そこで本稿では，モバイルエージェントを生成するグローバルかつトポロジカルなプログラム言語を作成した．そこからモバイルエージェントが自動で生成されることで，個々のコンピュータを意識せず，モバイルエージェントを用いたマクロな制御が可能になる．

表 7 コマンド生成言語

コマンド生成言語	生成されるコマンド
Forward	F
Right，Left，Back	R，L，B
Color	C
v = Invisivle.Sensor	Miv
v = Visivle.Sensor	Mvv
Repeat{ X }，Repeat n{ X }	*(Z),n(Z)
Set v = n，v++，v–	Svn，Iv，Dv
Parallel{ X_1 and X_2}	[Z_1;Z_2]
Branch{ X_1 or X_2 }	(Z_1;Z_2)
if(v==n){ X_1} else{ X_2}	Even(Z_1;Z_2)
if(v!=n){ X_1} else{ X_2}	Evnn(Z_1;Z_2)
if(v<n){ X_1} else{ X_2}	Evln(Z_1;Z_2)
if(v=>n){ X_1} else{ X_2}	Evgn(Z_1;Z_2)
if(v>n_1&&v<n_2){ X_1} else{ X_2}	Evbn_1:n_2(Z_1;Z_2)
Kill	K
delay(m)	Wm
Point(m)	Pm

はじめにコマンド生成言語について説明する．これらはひとつの関数からひとつのコマンドを生成し，それらを一般的に使われる関数やより直観的な言語で表すことでモバイルエージェントを用いたプログラムを書きやすくした．作成したコマンド生成言語を**表 7** に示す．ここで X は本稿で作成した任意の言語で，Z は X により生成されたコマンド列とする．

次にモバイルエージェントを生成する関数について説明する．これらは単体のコマンドではなく，複数のコマンドからなるモバイルエージェントが生成される．これらの言語は Processing[13] を参考にして作成した．Processing であらかじめ用意されている関数から制御対象となるユビキタスコンピュータの位置や接続関係を利用したモバイルエージェントを自動で生成するように拡張した．このことにより個々のコンピュータを意識せずにマクロな視点からプログラムを作成することができ，プログラムがコンピュータ群全体でどのようなふるまいを行うかが把握しやすくなった．ここで X は本稿で作成した任意の言語で，Z は X により生成されたコマンド列，x，y は任意の正の整数とする．

Move(x,y) Move はコンピュータ群全体での位置を利用して特定のコンピュータへの移動を容易にする関数である．本研究では方向は相対的とし，Move 関数は現在の向きへ x 回移動し，左へ y 回移動するモバイルエージェントを生成するようにした．そのため，Move(x,y) は**表 8** に示すよ

表 8 モバイルエージェント生成言語

関数	生成されるモバイルエージェント
Move(x,y)	x(F)Ly(F)
Move(-x,-y)	Bx(F)Ry(F)
Move(x,0)	x(F)
Move(0,y)	Ly(F)
Rect(x,y){ X }	2(x(ZF)y(ZF))
Line(x,y){ X } ※ x=y	$Z$$x$(FL F$Z$R)
Line(x,0){ X }	$Z$$x$(F Z)
Line(0,y){ X }	$Z$$y$(LF Z)
Circle(x){ X }	4(x(ZF)LF)

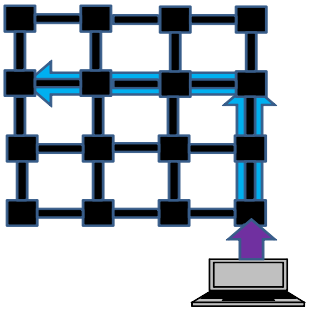


図 5 Move(2,3) の移動経路
Fig. 5 Migration path of Move(2,3)

うに進行，進路変更コマンドと繰り返しコマンドを組み合わせたモバイルエージェントを生成し，例えば Move(2,3) ならば，

“2(F)L3(F)”

を生成し，また，負の値はそれぞれ反対方向を表すようにし，例えば Move(-2,-3) ならば，

“B2(F)R3(F)”

現在の向きとは反対方向に x 回，右へ y 回移動するモバイルエージェントが生成される．

Rect(x,y){ X }

Rect は高さ x，幅 y の四角形をコンピュータ群で描写する関数である．現在の向きへ x 回の移動と，左へ y 回の移動を 2 回繰り返すことで四角形が描写される．Move と同様に負の値はそれぞれ反対方向へ移動し，四角形を描写する．本研究では Rect(x,y) は表 8 に示すように進行，進路変更

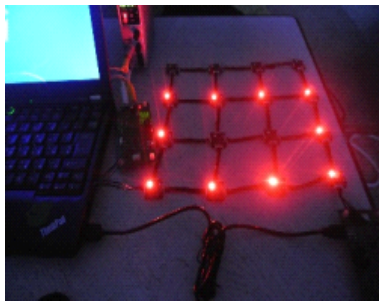
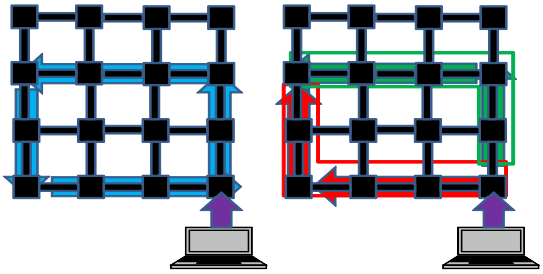


図 6 Rect(2,3){ Color Red} の実行結果
Fig. 6 Results of Rect(2,3){ Color Red}



(a) 実装した経路　(b) 分裂させた場合の経路

図 7 Rect(2,3) の移動経路
Fig. 7 Migration path of Rect(2,3)

コマンドと繰り返しコマンドを組み合わせたモバイルエージェントを生成する．また，{ } には任意の言語 X を入れることができ，例えば，Rect(2,3){ Color Red } は，

“2(2(CrF)L3(CrF)L)”

を生成する．

Rect によって生成されたモバイルエージェントを実行すると**図 7(a)** ように周回するような経路をたどりその経路上のコンピュータを制御できる．四角形を描画する方法としてはモバイルエージェントを分裂させた**図 7(b)** のような経路もあり，この処理を達成するモバイルエージェントとしては並列処理コマンドを用いた以下が考えられる．

“[2(CrF)L3(CrF);L3(CrF)R2(CrF)]”

しかし，本稿で周回経路を移動するモバイルエージェントを生成させる理由として，このエージェントは実行した後，処理を始めるコンピュータの位置に戻るというメリットがある．このことにより，環境内でのデータ収集，例えば，経路上にあるコンピュータのセンサの値を取得しホスト PC に自動で戻ってくるということなどが容易にできる．また，通信量は両者とも変わらない．しかし，本稿で実装した方法は一つのモバイルエージェントが周回移動するため，分裂させた場合よりも伝搬遅延が多くなってしまいコンピュータ群での処理の達成時間は遅くなってしまう．

Line(x,y){ X }

Line は現在の場所と，そこから進行方向へ x，左へ y の位置にあるコンピュータを結ぶ線上にあるコンピュータを制御する関数である．Line は現在の位置から上下左右の 4 方向と対角線方向のみを実装していて，x，y は上下左右に線

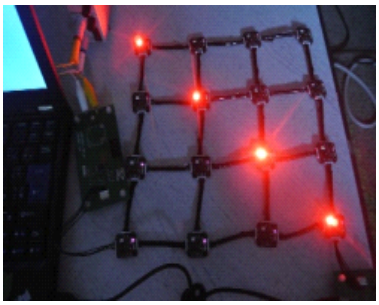
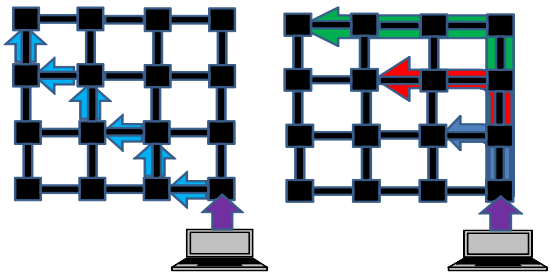


図 8 Line(3,3){ Color Red} の実行結果
Fig. 8 Result of Line(3,3){ Color Red}



(a) 実装した経路　(b) 分裂させた場合の経路

図 9 Line(3,3) の移動経路
Fig. 9 Migration path of Line(3,3)

を描写するなら x=0 または y=0 で，対角線上に描写するなら x=y の必要があり，これらはは表 8 に示すように進行，進路変更コマンドと繰り返しコマンドを組み合わせた別々のモバイルエージェントを生成する．また，同様に { } には任意の関数 X を入れることができ，Line(3,3){ Color Red } とすると，

“Cr3(FLFCrR)”

を生成する．これを実行すると**図 8** のように現在の場所から進行方向へ 3，そこから左へ 3 の位置にあるコンピュータを結ぶ線上にあるコンピュータの LED を赤色で描写する．Line によって生成されたモバイルエージェントを実行すると**図 9(a)** ような経路をたどりその経路上のコンピュータを制御できる．線を描画する方法としてはモバイルエージェントを分裂させた**図 9(b)** のような経路もあり，この処理を達成するモバイルエージェントとしては並列処理コマンドを用いた以下が考えられる．

“[2(3(F)L);2(2(F)L);2(FL);]Cr”

しかしこの方法だと，複数のエージェントを生成するため，最初のコンピュータ (図 9(b) だとホスト PC と直接接続しているコンピュータ) にかかる負荷が大きくなってしまい，線の長さが長くなるとそれだけ多くのエージェントを生成する必要があるため実行にかかる時間が多くなると考えられる．

Circle(x){ X }

Circle は現在の向きに対して，現在の場所を右下としてそこから直径 x の円をコンピュータ群で描写する関数である．

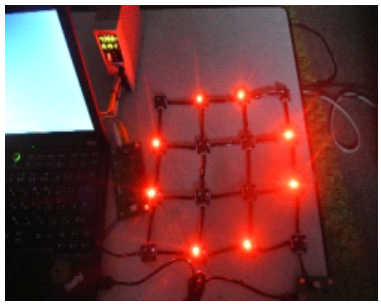


図 10 Circle(2){ Color Red} の実行結果
Fig. 10 Result of Circle(2){ Color Red}

本研究では Circle(x) は表 8 に示すように進行，進路変更コマンドと繰り返しコマンドを組み合わせたモバイルエージェントを生成する．同様に { } には任意の言語 X を入れることができ，Circle(2){ Color Red } は，

“4(2(CrF)LF)”

を生成する．これを実行すると**図 10** に示すように直径 2 の赤色の円形をコンピュータ群上で描写する．

Circle によって生成されたモバイルエージェントを実行すると Rect と同様な周回経路をたどりその経路上のコンピュータを制御できる．円を描画する方法としては Rect 同様モバイルエージェントを分裂させた経路も考えられるが，エージェントは実行した後，処理を始めるコンピュータの位置に戻るというメリットを優先し周回する方法を実装した．

上述のようにマクロな言語からモバイルエージェントを生成することで，ユビキタスコンピュータが互いに連携しながらコンピュータ群を制御することができた．また，個々のコンピュータを意識せずにマクロな視点からプログラミングができるため，プログラムがコンピュータ群全体でどのようなふるまいを行うかが把握しやすくなった．生成されるモバイルエージェントとして本稿では負荷を軽減させることや，例えばセンサデータ収集のしやすさなどユビキタス環境やセンサネットワークへの応用も視野に入れた経路を移動するようなモバイルエージェントを選択した．しかし，コンピュータ群全体での伝搬速度やコンピュータ群内での断線なども考慮すると今後もさまざまな生成方法と定量的に比較する必要がある．

5. ま と め

本研究では，格子状に配置したユビキタスコンピュータ群に対して，それぞれのユビキタスコンピュータに組み込まれた入出力デバイスを制御するためのプログラミングを想定し，モバイルエージェントによる制御を行った．ユビキタスコンピュータにモバイルエージェントを用いるため，進行方向に移動，並列処理，入出力制御機能などをもつコマンドを多数作成した．本稿ではモバイルエージェントを生成する言語を作成することで，各コンピュータでの

挙動を意識することなくマクロな視点でモバイルエージェントを生成でき，プログラムがコンピュータ群全体でどのようなふるまいを行うかが把握しやすくなった．今後としては生成方法の定量的な評価や，新しい言語の作成や拡張を行いユビキタス環境で求められる処理を短いマクロな言語で実現することがあげられる．また，状況によってユーザーが生成されるモバイルエージェントを自由に選択できるようにシステムの拡張をしていく．

謝辞

本研究の一部は，文部科学省科学研究費補助金基盤研究 (A)(20240009, 23240010) によるものである．ここに記して謝意を表す．

参考文献

[1] 國本慎太郎，藤田直生，佐野渉二，寺田 努，塚本昌彦：格子状に接続されたユビキタスコンピュータ群のモバイルエージェントを用いた制御手法，マルチメディア，分散，協調とモバイルシンポジウム (DICOMO2011)，pp. 741-748 2011.

[2] T. Terada, M. Tsukamoto, K. Hayakawa, T. Yoshihisa, Y. Kishino, A. Kashitani, and S. Nishio: Ubiquitous Chip: a Rule-based I/O Control Device for Ubiquitous Computing, *Proceeding of the International Conference on Pervasive Computing (Pervasive 2004)*, pp. 238–253, 2004.

[3] B. Warneke, M. Last, B. Liebowitz, and K. Pister: Smart Dust: Communicating with a Cubic-Millimeter Computer, *Proceedings of the IEEE Computer Magazine*, Vol. 34, Issue 1, pp. 44-51, 2001.

[4] R. Gummadi, O. Gnawali, and R. Govindan: Macro-Programming Wireless Sensor Networks Using Kairos, *Proceedings. of the International Conference on Distributed Computing in Sensor Systems (DCOSS2005)*, pp. 126–140, 2005.

[5] R. Newton, G. Morrisett, and M. Welsh: The Regiment Macroprogramming System, *Proceedings of 6th International Conference on Information Processing in Sensor Networks (IPSN2007)*, pp. 489–498, 2007.

[6] B. Urs and K. Gerd: RuleCaster: A Macroprogramming System for Sensor Networks, *Proceedings of the 21th Annual ACM Conference on Object-Oriented Programming, Systems, Languages, and Applications (OOPSLA2006)*, 2006.

[7] C.-L. Fok, G.-C. Roman and C. Lu: Mobile agent middleware for sensor networks: an application case study, *Proceedings of the 6th International Conference on Information Processing in Sensor Networks (IPSN2005)*, pp. 382–387 2005.

[8] Y.-C. Tseng, S.g-P. Kuo, H.-W. Lee and C.-F. Huang: Location Tracking in a Wireless Sensor Network by Mobile Agents and Its Data Fusion Strategies, *Proceedings of the Computer Journal*, Vol. 47, Issue 4, pp. 448–460, 2004.

[9] Y. Xua, and H. Qi: Mobile agent migration modeling and design for target tracking in wireless sensor networks, *Ad Hoc Networks*, Vol. 6, Issue 1, pp. 1–16, 2008.

[10] S. Ilarria, E. Menaa, and A. Illarramendib: Using cooperative mobile agents to monitor distributed and dynamic environments, *Information Sciences* , Vol. 178, Issue 9, pp. 2105–2127, 2008.

[11] D. Raychaudhuri: Overview of the ORBIT radio grid testbed for evaluation of next-generation wireless network protocols,*Proceedings of the Wireless Communications and Networking Conference (WCNC2005)*, Vol. 3, pp. 1664-1669, 2005.

[12] C. Zhang and T. Herman: Localization in Wireless Sensor Grids, *Computers and Their Applications*, pp. 388–393, 2006.

[13] Processing, http://www.processing.org/.