

卒業論文

題名

マルチエージェントシステムを用いた避難シミュレーション
～社会情報学部棟での避難訓練の提案～

学籍番号

11601044

氏名

前田 瑠美

指導教員名

富山 慶典

平成 27 年 1 月 20 日 提出

概要

災害が発生した際、大学のような大人数が集まる場所では素早く避難できるかが生死をわける。しかし、私達が利用している社会情報学部棟（以下、社情棟）は6階建てのビル。にもかかわらず避難シューター等は備わっておらず、避難するためには全員が階段で1階に降りなければならない。そのため、ほかの大学施設に比べ階段や廊下での混雑が発生しやすく、避難に時間がかかることが予想される。災害時の最も初歩的な備えが避難訓練だ。本研究では、災害時に迅速かつ安全に避難を行うにはどのような避難訓練をいつ行うべきなのかを考え、提案する。

避難訓練の提案のために、シミュレーターを用いて災害時の社情棟を再現し、危険な時間帯や場所から行うべき避難訓練の時間帯や方法を考える。シミュレーションにはマルチエージェントシステムを用いた。今回のシミュレーションでは二つのモデルを使用する。第一の避難モデルは、距離のみを最優先して避難を行う「距離優先モデル」。このモデルでは、最も近い出口に人を避難誘導した際の動きを再現している。第二の避難モデルは混雑状況を把握し、人の少ないほうへ避難を行う「混雑回避モデル」。このモデルでは、人の混雑具合を判断した上で空いている方へ人を避難誘導した際の動きを再現している。それぞれのモデルを用いて、前期後期、各曜日、時間帯別でそれぞれ10回のシミュレーションを行う。その結果をモデル、前後期、曜日、コマをダミー変数、合計人数または各階の人数を独立変数（連続変数）、避難所要時間を従属変数として重回帰分析を行い、それぞれの要素と避難所要時間の関係性を求める。

その結果、モデル2はモデル1よりも避難にかかる時間が少なく、有用な避難方法であることが判明した。それに加え、2階、4階、6階の人数が避難所要時間に大きく関与していることも判明した。分析を進め実際に避難時間が長い時間帯の人数を確認してみると、2階に集まっている人数が極端に多いか、2階に多くの人が集まっているのに加え、4階と6階のどちらか、もしくは両方にも人数が集まっていた。

これらの分析結果から、モデル2のような避難方法を用いて、2階に極端に人が集まっているか、2階に多くの人が集まっており、なおかつ4階、6階にもそれなりの人が集まる時間帯に避難訓練を行うことを提案する。そうすれば、災害時に迅速かつ安全に避難をすることができるだろう。

目次

第1章	はじめに	- 2 -
1.1	研究目的	- 2 -
1.2	研究手段	- 2 -
1.2.1	実際の避難訓練の観察	- 2 -
1.2.2	シミュレーション～マルチエージェントシステム(MAS)～	- 3 -
1.3	先行研究	- 3 -
第2章	避難シミュレーションモデルの構築	- 4 -
2.1	マルチエージェントシミュレーター「artisoc」	- 4 -
2.2	社会情報学部棟内配置図	- 5 -
2.3	シミュレーションモデル作成	- 5 -
2.3.1	モデル概要	- 5 -
2.3.2	フロチャート	- 6 -
2.4	モデルで使用するエージェント	- 7 -
2.4.1	壁エージェント(wall)	- 7 -
2.4.2	人エージェント(people)	- 7 -
2.4.3	チェックポイントエージェント(check)	- 8 -
2.4.4	階段エージェント(step)	- 8 -
2.4.5	下降エージェント(down)	- 9 -
2.4.6	出口エージェント(exit・exitstep)	- 9 -
2.4.7	人数エージェント(count)	- 9 -
2.4.8	転換エージェント(change)	- 9 -
2.4.9	場所エージェント(U)	- 10 -
2.4.10	復帰エージェント(back)	- 10 -
第3章	シミュレーション実施	- 11 -
3.1	初期設定	- 11 -
3.2	それぞれのモデルの結果	- 14 -
3.2.1	モデル1:距離優先モデル	- 14 -
3.2.2	モデル2:混雑回避モデル	- 16 -
3.2.3	データ分析および考察	- 17 -
第4章	おわりに	- 24 -
4.1	有意義な避難訓練の提案	- 24 -
4.2	今後の課題	- 24 -
	謝辞	- 25 -
	参考文献	- 25 -
	付録	- 25 -

第1章 はじめに

1.1 研究目的

東日本大震災から今年で4年。その間にも、日本国内の様々な場所で地震が頻発している。私達のキャンパスがある群馬県は地盤などの影響により比較的に地震が少ない土地であるといわれているが、地震や火事などの災害が発生しない保証はない。

災害が発生した際、大学のような大人数が集まる場所では素早く避難できるかが重要になる。それに加え、私達社会情報学部が利用している社会情報学部棟(以下、社情棟)は6階建てのビルだ。にもかかわらず避難シューター等は備わっておらず、避難するためには全員が階段で1階に下りなければならない。そのため、ほかの学部甚至比階段や廊下での混雑が発生しやすく、避難に時間がかかることが予想される。

災害時の最も初歩的な備えが避難訓練だ。しかし、群馬大学荒牧キャンパスでは避難訓練を毎年一回だけしか行っておらず、避難訓練を受けたことのない生徒も多数存在する。そこで本研究では、災害時より迅速かつ安全に避難を行うにはどのような避難訓練をいつ行うべきなのかを提案する。

1.2 研究手段

有用な避難訓練の提案には、二つの手法が考えられる。一つ目は実際に避難訓練を観察し、問題点を発見した上でその原因を分析、改善策を提案すること。二つ目はコンピュータ(PC)を用いて災害時の社情棟を再現し、危険な時間帯や場所から行うべき避難訓練の時間帯や方法を提案することだ。

1.2.1 実際の避難訓練の観察

群馬大学荒牧キャンパスでは、毎年後期中旬の水曜日に避難訓練を実施している。今年度は10月22日(水)の午前10:00に地震を想定した避難訓練が実施された。参加者は傷害防止姿勢をとりつつ非常階段を使用してグラウンドへ避難を行った。社情棟では正面玄関が通行不能であると想定し、非常階段のみを利用して屋外へ避難した。避難の様子は図表1.2の通りだ。

実際に観察した避難訓練では順調に人が流れており、問題点を発見することができなかった。また、本研究の期間内では避難訓練が一回しか行われず、なおかつ避難訓練の妨害にならない場所からの観察しかできなかったため、分析ができるほどの情報を集めることができなかった。



図表 1

図表 2

1.2.2 シミュレーション～マルチエージェントシステム(MAS)～

実際の避難訓練の観察からは避難訓練を提案することができなかつたため、PC を用いてシミュレーションを行う方法を検討してみる。シミュレーションには MAS を用いた。

MAS とは複数のエージェントが共存し相互作用しあうシステムを指す。そして MAS におけるエージェントとは、自律的に判断し行動するシステム上の構成要素のことである。エージェント同士の相互作用によりシステム全体の流れが形成され、その流れが今度は逆にエージェントに影響を与え、エージェントの行動を変動させる。それにより、MAS では現実に近い状況を再現しシミュレーションを行うことができる。

このシステムを用いれば、実際には再現不可能な災害時の状況を PC 上に再現することができる。そのため、実際の避難経路の立案や避難時の問題発見、解決策の立案にも有効であるとされている。今回はこの MAS を用いて非常時の社情棟を再現し、人数による混雑具合の変化や人の流れを観察することで避難訓練の提案を行う。

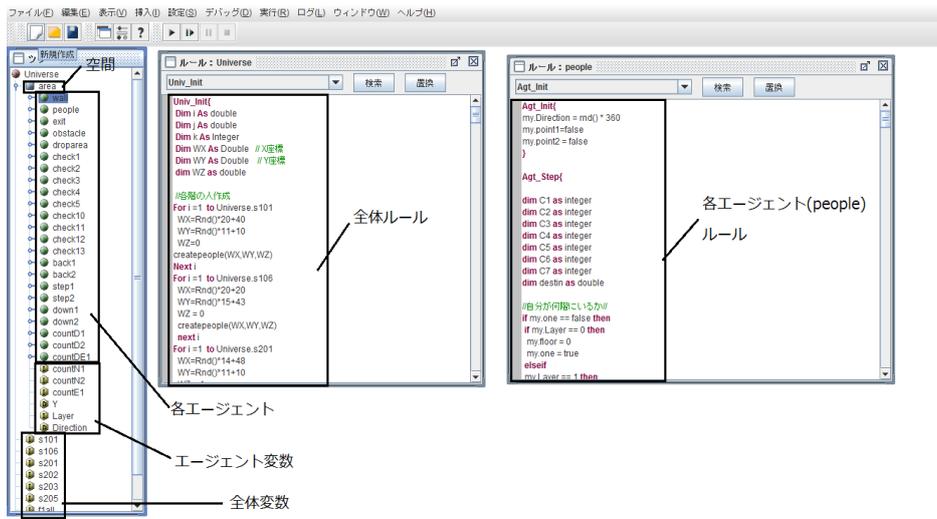
1.3 先行研究

MAS を用いた災害時の大学や商業施設から避難するシミュレーションについての論文は複数ある。大学キャンパスでの避難シミュレーションを行っている「マルチエージェントによる避難計画を踏まえた教室配置の検証～芝浦工業大学大宮キャンパス新 2 号館対象～」では、芝浦工業大学新 2 号館を対象とし、履修者数の予想を基に授業教室をどのように振り分ければ人の流れを円滑化でき、避難時さらには通常時の移動時間を短縮させることができるのかを検証している。今回のシミュレーションではこの論文で使用されている避難プログラムと兼田(2011)、および山影(2007)に掲載されていたプログラムを参考に、artisoc を利用した災害発生時の社情棟からの同時避難システムモデルを 2 パターン作成し、シミュレーションを行う。

第2章 避難シミュレーションモデルの構築

2.1 マルチエージェントシミュレーター「artisoc」

artisoc(アーティソック : artificial societies (=人工の社会))とは構造企画研究所が作成した分析支援ツールである。プログラム初心者でもマルチエージェントの手法により社会現象を分析・理解できるように設計されており、言語形式はvisual basicに近い。作成した「空間」に様々な「変数」を持った「エージェント」を配置し各エージェントまたは全体に「ルール」を与え、その「ルール」に沿って行動させることで仮想社会を形成することができる。



図表 3

artisoc における「空間」とは、「エージェント」が配置される場所であり、「エージェント」同士が相互作用しあう仮想空間である。

「エージェント」とは、シミュレーションの中で、一定の「ルール」に従って行動する存在であり、お互いに働きかけたり、影響を受けたりする。

「ルール」とは、「エージェント」の行動指針である。動かないエージェントによっては個別のルールを持たない場合もある。

「変数」とは、「エージェント」が持っている情報や状態を、数値や文字の形で格納しておく箱のようなものである。「変数」を持たないエージェントは存在しない。また、「ルール」と「変数」については各「エージェント」だけでなく空間全体にも設定することができる。

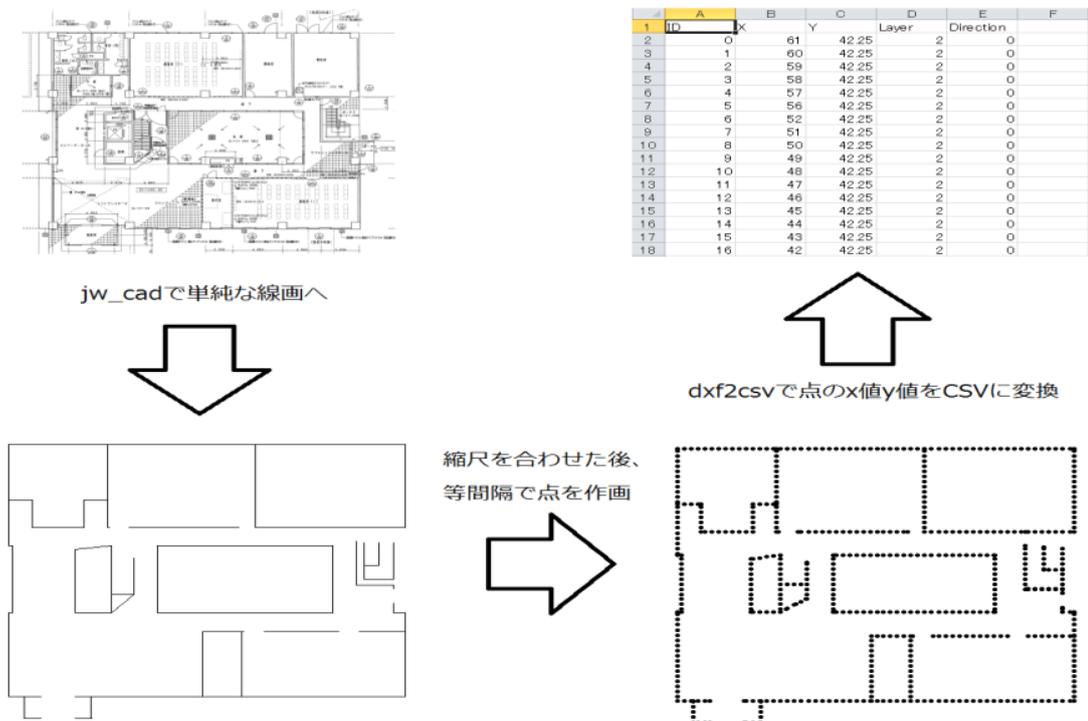
artisoc にはサンプルプログラムが数多くあり、構造計画研究所のホームページ(HP)上で自由にダウンロードすることができる。また、研究活動支援のため、artisoc を教育目的で使用する場合に限り、研究期間中の無償貸与が行われている。それと同時に、今までに無償貸与された artisoc を用いて行われたシミュレーションの内容やその結果についての論文も HP で自由に閲覧することができる。

2.2 社会情報学部棟内配置図

避難シミュレーションを行う場である社情棟を表現するために、壁の位置の X 値 Y 値を取得し、その位置に壁エージェントを配置する。

配置を再現するために使用したソフトは、cad 図面作成のためのフリーソフトである「Jw_cad」と、知り合いのシステムエンジニアの方に作ってもらった dxf データを csv 形式に変換する変換プログラム「dxftocsv」である。

作成手順は、まず Jw_cad を用いて実寸大で図面を作成し、jw 形式で保存をする。次に使用させて頂いた dxftocsv では座標の単位が 1mm だったが、シミュレーションで用いる単位は 0.5m のため、そちらにあわせて大きさを変更する。その後、線を点の集まりとして表現するために 1mm 間隔で点を作画し、線すべてを点へ変換し終わったら dxf 形式で保存、dxftocsv にて csv 形式に変換をする。そして最終的に、artisoc にて csv 形式で出力された座標データを壁エージェントの初期値として読み込ませた。



図表 4

2.3 シミュレーションモデル作成

今回のシミュレーションでは二つのモデルを使用する。その後、各モデルの最も危険な時間帯や強く影響する階層を調べるとともに、二つを比較することで避難誘導の仕方についても考える。

2.3.1 モデル概要

一つ目の避難モデルは、距離のみを最優先して避難を行う「距離優先モデル」(以下、モデル1)である。このモデルでは、最も近い出口に人を避難誘導した際の動きを再現している。そのため、人エージェント(以下、人)は自分に近いチェックポイントだけを目指して

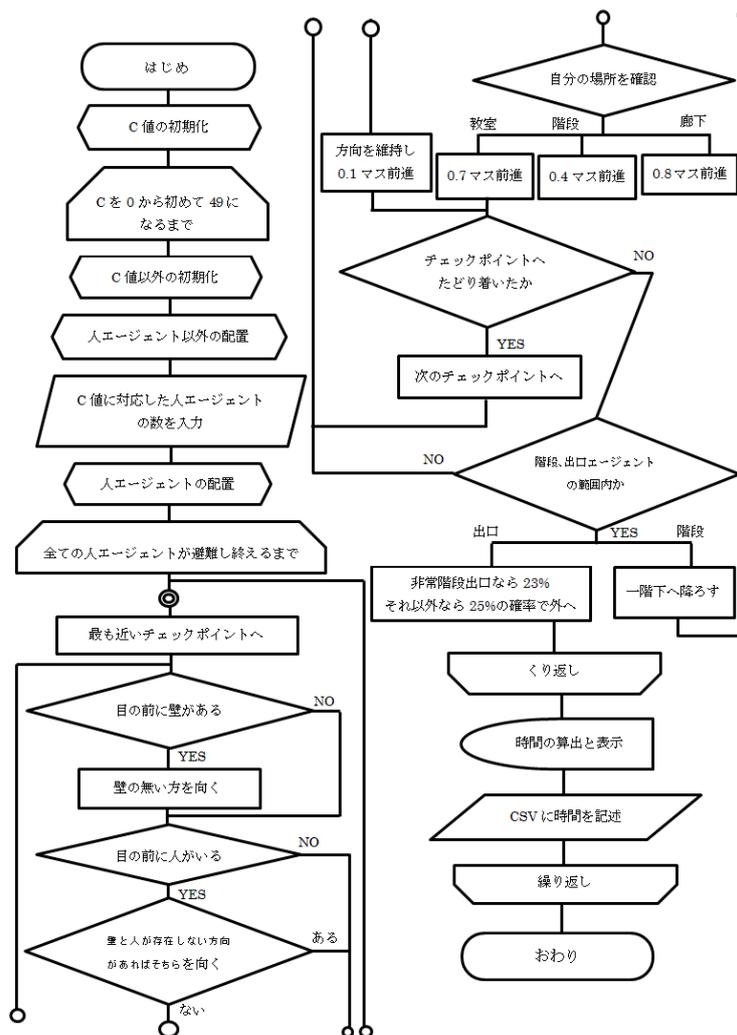
避難を行う。出入口や階段が混んでいて先に進めない状況になっていても、ほかに移動することなくそこにとどまり避難ができるまで待機する。

二つ目の避難モデルは、混雑状況を把握し人の少ない方へ避難を行う「混雑回避モデル」(以下、モデル 2)である。このモデルでは、人の混雑具合を判断した上で空いている方へ避難誘導した際の動きを再現している。そのため、人は階段付近や出入口に一定人数がたまっていた場合その階段、出入口を使用することをあきらめ、空いているもう片方を目指して移動する。人は教室から出た時点でどの階段、出入口が空いているかを判断し移動する。そのため、その場所についての時点でそこが混んでいたとしても、その場所で避難ができるまで待機する。また、一定の人数がその場にたまっていない、もしくは比較する場所の混雑具合が同じだった場合は、モデル 1 と同じように自分に近いチェックポイントを目指して避難を行う。

2.3.2 フロチャート

それぞれのモデルの大まかなフロチャートは図表 5.6 の通りだ。

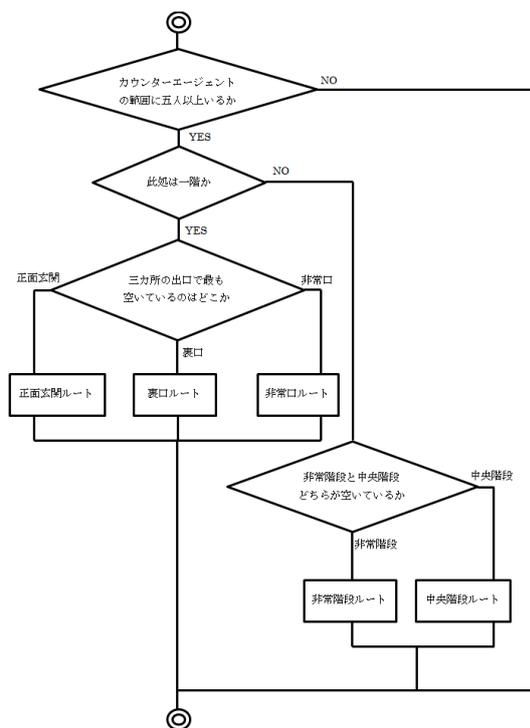
モデル 1：距離優先



図表 5

モデル 2：混雑回避

モデル 2 では、モデル 1 のフロチャート◎部分に図表 6 のフロチャートが加わったものである。それ以外は完全に同じのため、ここでは割愛する。



図表 6

2.4 モデルで使用するエージェント

artisoc では作成した「空間」に様々な特性を持った「エージェント」を配置することで実際の様子を再現し、一定数の step を繰り返しながらシミュレーションを行う。今回のシミュレーションでは城田(2011)のプログラムを参考に「壁エージェント」「人エージェント」「チェックポイントエージェント」「階段エージェント」「降下エージェント」「出口エージェント」「復帰エージェント」を作成し、配置する。モデル 2 ではこれらのエージェントに加え、「人数エージェント」「転換エージェント」「場所エージェント」を追加し、配置する。

2.4.1 壁エージェント(wall)

シミュレーションを行う場所の壁を表現しており、大学側より提供してもらった社情棟の配置図の通りに壁エージェントを設置することで社情棟内を再現している。エージェントの大きさは一定のため、壁の厚さは全て 0.5m(1 マス)となっている。壁エージェント自体はルールを持っていないが、人は壁エージェントを避けるルールを持っている。

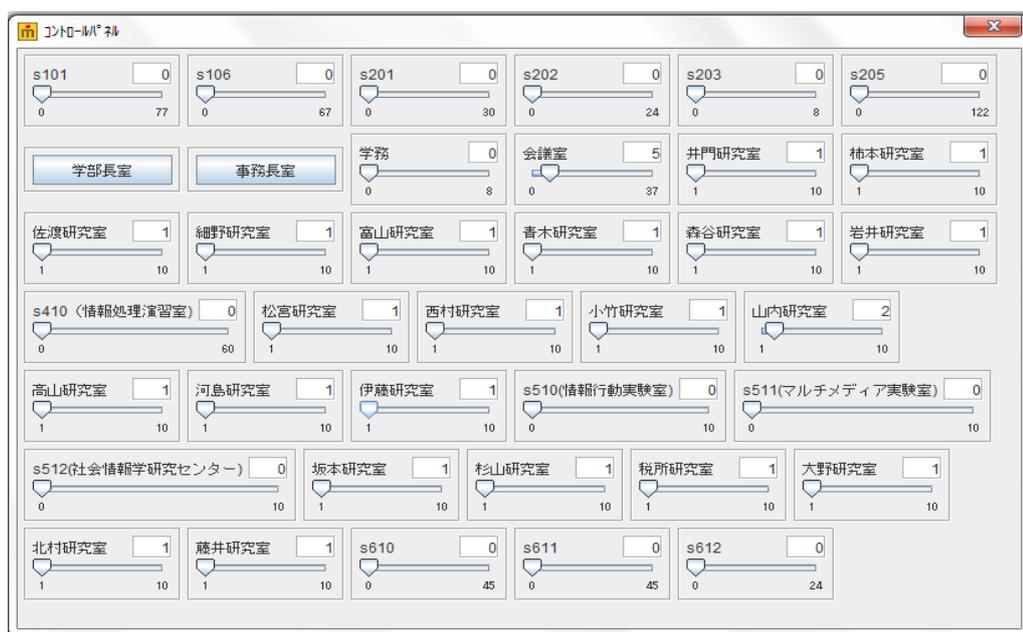
2.4.2 人エージェント(people)

壁や人を避けながら出口へ向かうエージェント。今年度の前期後期教室利用状況に合わ

せた人数を、教室内にランダムに配置する。人の歩行速度は城田(2011)より、廊下やエントランスでは 1.2m/s(0.8 マス/step)、教室では 1.0m/s(0.7 マス/step)、階段内では 0.6m/s(0.4 マス/step)としている。なお、今回のシミュレーションではモデルを単純化するために以下の教室の人数を一定にしている。

まず、教授の研究室には教授、または誰かが必ず一人存在しているとする。次に、事務室には固定人数(8 人)の事務員が常に存在し、事務長室や学部長室には常に事務長や学部長が存在しているとする。そして最後に、203 号室(F2 の下部一番左の部屋)の院生室には院生が常に 8 人存在しているとしてエージェントを配置する。

今回はあらかじめ準備しておいた教室使用人数配当データ(CSV 形式)を初期設定として読み込ませたが、直観的に人数を変更できるようにコントロールパネルを作成した。最大値はその教室の許容人数である。



図表 7

2.4.3 チェックポイントエージェント (check)

チェックポイントを通過したことを人に伝えるエージェント。入り組んだ建物内では、一直線に出口へ向かうことはできない。そのため、通路の要所にチェックポイントを設け、そこを通過しながら出口に向かわせる。曲がり角や出入口に設置し、人は自分に一番近いチェックポイントを目指して移動する。

2.4.4 階段エージェント (step)

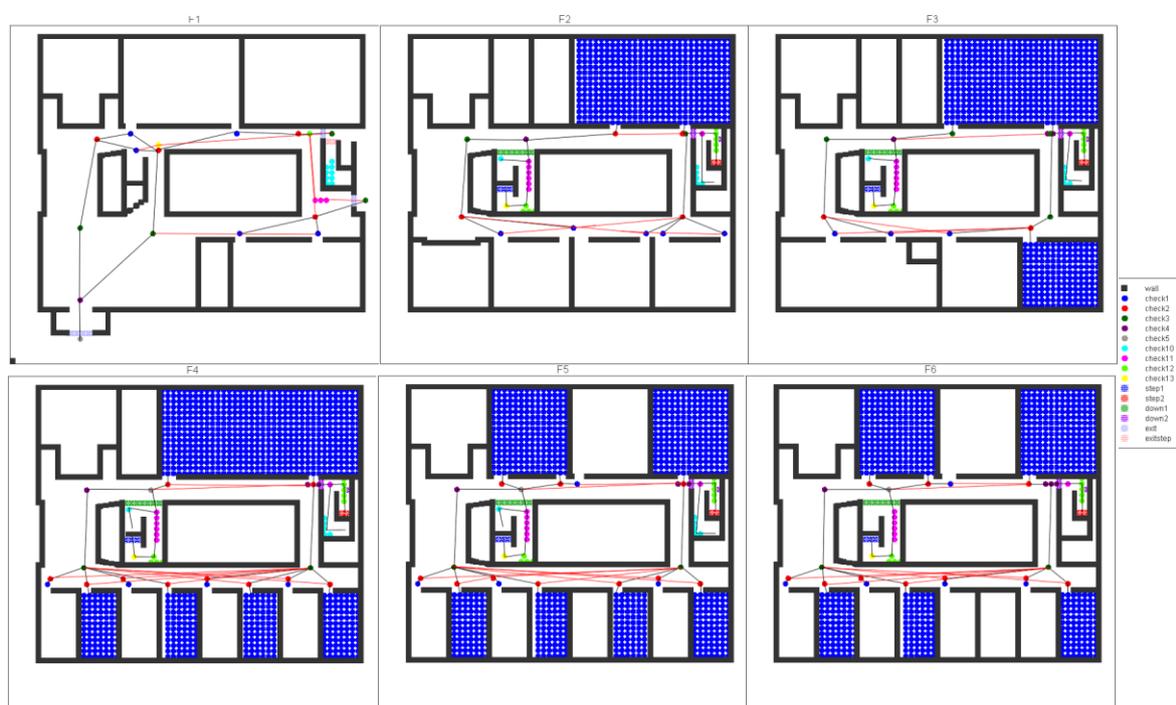
自分が存在する階に出口が存在しない場合、階段に集まってきた人を下の階へ移動させるエージェント。中央階段では下り階段の中央部に、非常階段では下り階段の最深部に設置している。

2.4.5 下降エージェント(down)

階段エージェントによって下に運ばれている人が途中で設置されているチェックポイントエージェントに接触し階段を出てしまわないように、一度階段に入った人は下に下り続けるルートを選択させるエージェント。階段入口へ設置する。なお、今回のシミュレーションではエレベーターが災害により停止し、使用不可であるとする。

2.4.6 出口エージェント(exit・exitstep)

正面出入口、裏出入口、非常口、非常階段出口に設置し、集まってきた人を外へ避難(人を消失)させるエージェント。出入口は流動係数により一秒間に出入される人数が決まっているため、今回は兼田(2011)より正面出入口、横出入口、非常口は1.5人/m²(1マスあたり0.25人/step)、非常階段出口は1.3人/m²(1マスあたり0.23人/step)としている。



図表 8 黒い線が形成されたルート、赤い線がモデル 2 で追加されたルート

2.4.7 人数エージェント(count)

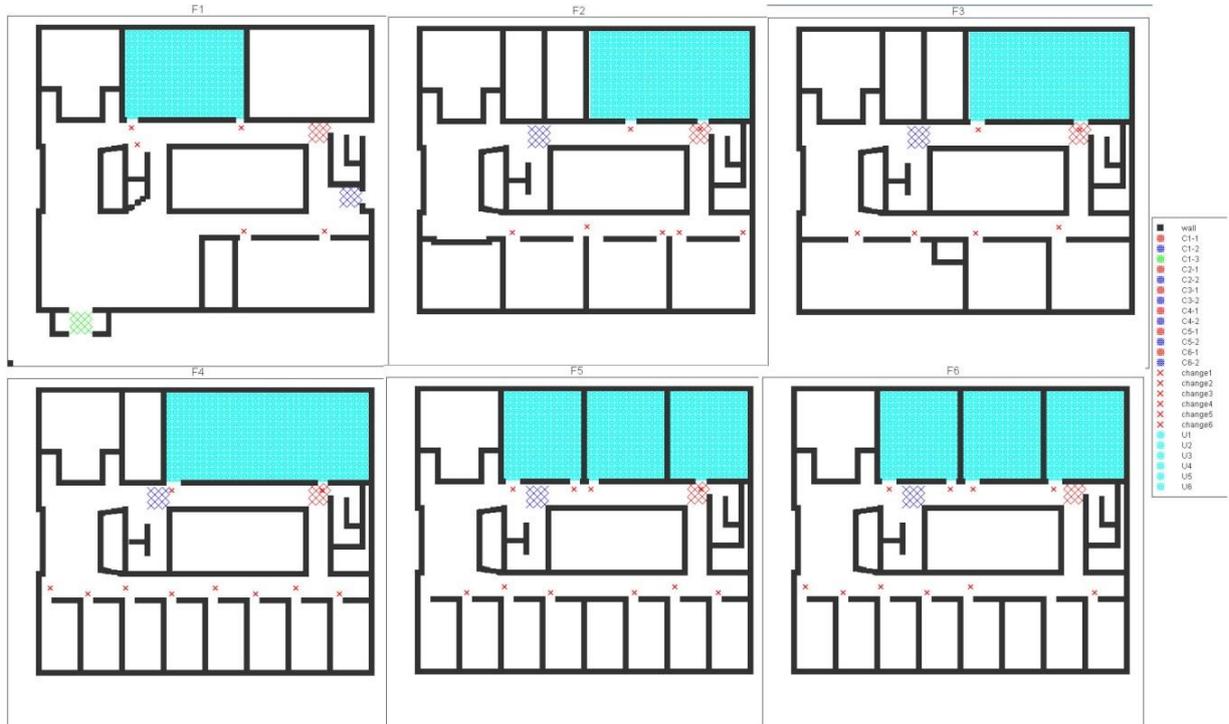
一つの出口や階段が混んでいた場合、もう一つの階段を選択するために特定の場所の人数を計測するエージェント。今回は各階の中央階段と非常階段の二カ所と、正面出入口と裏出入口、非常口の三カ所の入口付近に同じサイズのものを設置し、そこに存在する人数を計測する。

2.4.8 転換エージェント(change)

二つ、または三つの場所の人数を比べ、少ない方の階段や出入口を選択させるためのエージェント。各教室の出入口に設置する。人数エージェントにいた人が5人以上だった場合、転換エージェントは自分の階にある二つ(1階ならば三つ)の人数エージェントを比較し、空いている方の出入口、階段へのルートの人を選択させる。

2.4.9 場所エージェント(U)

人に自分が北側、南側(図表では上部か下部)どちらの教室にいるのかを認識させるエージェント。モデル2においてルートを新たに作成するために北側の教室内に設置した。



図表 9

2.4.10 復帰エージェント(back)

順路から外れてしまった人を元に戻すエージェント。周りの人に押され壁の中に入ってしまった人や、チェックポイントの通過ミスによりルートを逆走してしまった人を元に戻すため様々な場所に設置する。設置場所は図表 10 の通り。



図表 10

第3章 シミュレーション実施

3.1 初期設定

二つのモデルを使用し、前期後期、各曜日、時間帯でそれぞれ10回のシミュレーションを行う。今回は1マスを0.5m×0.5mとし、3stepを1秒として計測した。また、step数が2000を超えても避難が完了しなかった場合をエラーとし、再度同じ値でシミュレーションをやり直した。

初期値として用いた各教室の人数割り振りは図表 11. 12. 13. 14 の通りである。

前期		101	106	201	202	203	205	301	302	303	305-306	401	402	403	404	405	406	407	408	410
月	1コマ	39	29	0	0	8	0	1	1	8	0	1	1	1	1	1	1	1	1	0
月	2コマ	55	59	5	0	8	84	1	1	8	0	1	1	1	1	1	1	1	1	0
月	3コマ	0	64	0	4	8	67	1	1	8	0	1	1	1	1	1	1	1	1	0
月	4コマ	0	0	0	0	8	0	1	1	8	0	1	1	1	1	1	1	1	1	0
月	5コマ	0	47	0	0	8	89	1	1	8	0	1	1	1	1	1	1	1	1	0
火	1コマ	0	0	0	0	8	33	1	1	8	0	1	1	1	1	1	1	1	1	0
火	2コマ	0	0	0	2	8	91	1	1	8	0	1	1	1	1	1	1	1	1	17
火	3コマ	12	0	5	9	8	76	1	1	8	0	1	1	1	1	1	1	1	1	10
火	4コマ	24	0	8	9	8	103	1	1	8	0	1	1	1	1	1	1	1	1	0
火	5コマ	6	9	9	0	8	2	1	1	8	0	6	1	3	1	1	2	1	5	0
水	1コマ	0	0	0	0	8	146	1	1	8	0	1	1	1	1	1	1	1	1	0
水	2コマ	40	60	0	0	8	79	1	1	8	0	1	1	1	1	1	1	1	1	0
水	3コマ	0	0	0	0	8	0	1	1	8	0	1	1	1	1	1	1	1	1	60
水	4コマ	0	36	0	0	8	0	1	1	8	31	1	1	1	1	1	1	1	1	0
水	5コマ	0	0	0	0	8	0	1	1	8	0	1	1	1	1	1	1	1	1	0
木	1コマ	0	46	0	0	8	134	1	1	8	0	1	1	1	1	1	1	1	1	0
木	2コマ	31	40	0	0	8	97	1	1	8	0	1	1	1	1	1	1	1	1	0
木	3コマ	28	46	0	0	8	54	1	1	8	0	1	1	1	1	1	1	1	1	0
木	4コマ	60	0	28	0	8	100	1	1	8	0	1	1	1	1	1	1	1	1	27
木	5コマ	5	0	6	9	8	0	1	1	8	0	6	1	1	1	1	1	1	1	0
金	1コマ	0	0	0	0	8	0	1	1	8	0	1	1	1	1	1	1	1	1	0
金	2コマ	42	57	0	0	8	79	1	1	8	0	1	1	1	1	1	1	1	1	0
金	3コマ	0	57	0	5	8	115	1	1	8	0	1	1	1	1	1	1	1	1	0
金	4コマ	36	69	3	0	8	63	1	1	8	0	1	1	1	1	1	1	1	1	0
金	5コマ	0	0	3	2	8	53	1	1	8	0	1	1	1	1	1	1	1	1	0

図表 11

前期		501	502	503	505	506	507	508	510	511	512	601	604	605	606	607	608	610	611	612	合計人数
月	1コマ	1	1	1	1	1	1	1	0	0	0	1	1	1	1	1	1	0	0	0	107
月	2コマ	1	1	1	1	1	1	1	0	0	0	1	1	1	1	1	1	0	0	0	242
月	3コマ	1	1	1	1	1	1	1	0	0	0	1	1	1	1	1	1	0	0	0	174
月	4コマ	1	1	1	1	1	1	1	0	1	0	1	1	1	1	1	1	29	0	0	69
月	5コマ	1	1	1	1	1	1	1	0	0	0	1	1	1	1	1	1	0	0	0	175
火	1コマ	1	1	1	1	1	1	1	0	0	0	1	1	1	1	1	1	0	0	0	72
火	2コマ	1	1	1	1	1	1	1	0	0	0	1	1	1	1	1	1	0	0	0	149
火	3コマ	1	1	1	1	1	1	1	0	0	0	1	1	1	1	1	1	0	0	2	153
火	4コマ	1	1	1	1	1	1	1	0	0	0	1	1	1	1	1	1	9	0	7	199
火	5コマ	1	1	1	8	1	1	1	0	11	0	1	8	1	1	1	1	8	6	0	116
水	1コマ	1	1	1	1	1	1	1	0	0	0	1	1	1	1	1	1	0	0	0	185
水	2コマ	1	1	1	1	1	1	1	0	0	0	1	1	1	1	1	1	0	24	17	259
水	3コマ	1	1	1	1	1	1	1	0	0	0	1	1	1	1	1	1	0	0	0	99
水	4コマ	1	1	1	1	1	1	1	0	0	0	1	1	1	1	1	1	0	0	0	106
水	5コマ	1	1	1	1	1	1	1	0	0	0	1	1	1	1	1	1	0	0	0	39
木	1コマ	1	1	1	1	1	1	1	0	0	0	1	1	1	1	1	1	34	17	0	270
木	2コマ	1	1	1	1	1	1	1	0	0	0	1	1	1	1	1	1	0	0	0	207
木	3コマ	1	1	1	1	1	1	1	0	0	0	1	1	1	1	1	1	22	45	0	234
木	4コマ	1	1	1	1	1	1	1	0	0	0	1	1	1	1	1	1	5	15	0	274
木	5コマ	1	10	1	1	5	1	1	0	9	0	5	1	6	1	1	1	6	9	0	110
金	1コマ	1	1	1	1	1	1	1	0	0	0	1	1	1	1	1	1	0	0	0	39
金	2コマ	1	1	1	1	1	1	1	0	0	0	1	1	1	1	1	1	38	0	4	259
金	3コマ	1	1	1	1	1	1	1	0	0	0	1	1	1	1	1	1	12	28	0	256
金	4コマ	1	1	1	1	1	1	1	9	0	0	1	1	1	1	1	1	32	8	0	259
金	5コマ	1	1	1	1	1	1	1	0	0	0	1	1	1	1	1	1	0	16	0	113

図表 12

後期																				
		101	106	201	202	203	205	301	302	303	305-306	401	402	403	404	405	406	407	408	410
月	1コマ	0	49	0	0	8	0	1	1	8	0	1	1	1	1	1	1	1	1	0
月	2コマ	28	24	15	0	8	56	1	1	8	0	1	1	1	1	1	1	1	1	0
月	3コマ	0	0	0	0	8	143	1	1	8	0	1	1	1	1	1	1	1	1	50
月	4コマ	34	59	0	0	8	29	1	1	8	0	1	1	1	1	1	1	1	1	4
月	5コマ	50	59	0	8	8	43	1	1	8	0	1	1	1	1	1	1	1	1	0
火	1コマ	0	56	0	1	8	0	1	1	8	0	1	1	1	1	1	1	1	1	0
火	2コマ	0	0	0	6	8	56	1	1	8	0	1	1	1	1	1	1	1	1	14
火	3コマ	0	41	0	0	8	51	1	1	8	0	1	1	1	1	1	1	1	1	12
火	4コマ	0	0	8	10	8	30	1	1	8	0	1	1	1	1	1	1	1	1	11
火	5コマ	6	9	6	10	8	2	1	1	8	0	6	1	3	1	1	2	1	5	0
水	1コマ	0	0	0	0	8	119	1	1	8	0	1	1	1	1	1	1	1	1	0
水	2コマ	26	74	0	2	8	34	1	1	8	0	1	1	1	1	1	1	1	1	0
水	3コマ	0	0	0	0	8	113	1	1	8	0	1	1	1	1	1	1	1	1	0
水	4コマ	0	0	0	0	8	126	1	1	8	0	1	1	1	1	1	1	1	1	0
水	5コマ	0	0	0	0	8	0	1	1	8	0	1	1	1	1	1	1	1	1	0
木	1コマ	0	67	0	0	8	75	1	1	8	0	1	1	1	1	1	1	1	1	0
木	2コマ	34	35	3	13	8	54	1	1	8	0	1	1	1	1	1	1	1	1	0
木	3コマ	50	33	1	0	8	50	1	1	8	0	1	1	1	1	1	1	1	1	0
木	4コマ	29	26	0	0	8	84	1	1	8	31	1	1	1	1	1	1	1	1	57
木	5コマ	0	0	6	0	8	0	1	1	8	0	6	1	1	1	1	1	1	1	0
金	1コマ	0	0	0	0	8	0	1	1	8	0	1	1	1	1	1	1	1	1	0
金	2コマ	0	14	0	0	8	101	1	1	8	0	1	1	1	1	1	1	1	1	0
金	3コマ	30	58	0	6	8	0	1	1	8	0	1	1	1	1	1	1	1	1	37
金	4コマ	33	0	4	0	8	107	1	1	8	0	1	1	1	1	1	1	1	1	0
金	5コマ	0	0	4	0	8	46	1	1	8	0	1	1	1	1	1	1	1	1	0

図表 13

後期																						
		501	502	503	505	506	507	508	510	511	512	601	604	605	606	607	608	610	611	612	合計人数	
月	1コマ	1	1	1	1	1	1	1	0	0	0	1	1	1	1	1	1	0	0	0	88	
月	2コマ	1	1	1	1	1	1	1	0	0	0	1	1	1	1	1	1	19	0	0	181	
月	3コマ	1	1	1	1	1	1	1	0	0	0	1	1	1	1	1	1	10	20	0	262	
月	4コマ	1	1	1	1	1	1	1	0	1	0	1	1	1	1	1	1	5	0	0	171	
月	5コマ	1	1	1	1	1	1	1	0	0	0	1	1	1	1	1	1	0	0	0	199	
火	1コマ	1	1	1	1	1	1	1	0	0	0	1	1	1	1	1	1	0	0	0	96	
火	2コマ	1	1	1	1	1	1	1	0	0	0	1	1	1	1	1	1	0	0	0	115	
火	3コマ	1	1	1	1	1	1	1	0	0	0	1	1	1	1	1	1	0	0	0	143	
火	4コマ	1	1	1	1	1	1	1	0	0	0	1	1	1	1	1	1	8	39	0	145	
火	5コマ	1	1	1	8	1	1	1	2	7	0	1	8	1	1	1	1	7	9	0	123	
水	1コマ	1	1	1	1	1	1	1	0	0	0	1	1	1	1	1	1	0	0	0	158	
水	2コマ	1	1	1	1	1	1	1	0	0	0	1	1	1	1	1	1	13	0	1	189	
水	3コマ	1	1	1	1	1	1	1	0	0	0	1	1	1	1	1	1	0	0	0	152	
水	4コマ	1	1	1	1	1	1	1	0	0	0	1	1	1	1	1	1	0	0	0	165	
水	5コマ	1	1	1	1	1	1	1	0	0	0	1	1	1	1	1	1	0	0	0	39	
木	1コマ	1	1	1	1	1	1	1	0	0	0	1	1	1	1	1	1	0	0	0	181	
木	2コマ	1	1	1	1	1	1	1	0	0	0	1	1	1	1	1	1	7	29	0	214	
木	3コマ	1	1	1	1	1	1	1	0	0	0	1	1	1	1	1	1	0	0	23	196	
木	4コマ	1	1	1	1	1	1	1	0	0	0	1	1	1	1	1	1	16	0	0	282	
木	5コマ	1	10	6	1	5	1	1	12	9	0	5	1	6	1	1	1	6	9	0	113	
金	1コマ	1	1	1	1	1	1	1	0	0	0	1	1	1	1	1	1	0	0	0	39	
金	2コマ	1	1	1	1	1	1	1	0	0	0	1	1	1	1	1	1	14	0	0	168	
金	3コマ	1	1	1	1	1	1	1	0	0	0	1	1	1	1	1	1	0	0	0	170	
金	4コマ	1	1	1	1	1	1	1	0	0	0	1	1	1	1	1	1	0	34	0	217	
金	5コマ	1	1	1	1	1	1	1	0	0	0	1	1	1	1	1	1	0	0	0	89	

図表 14

前期後期共に、最も人数が多い時間帯は木曜日の4コマである。年間を通じてでは最も人数が多い時間帯は後期木曜日の4コマである。

また、各階層の人数と割合は図表 15.16 のようになった。

前期		一階	%	二階	%	三階	%	四階	%	五階	%	六階	%	合計人数
月曜日	1コマ	68	64%	8	7%	10	9%	8	7%	7	7%	6	6%	107
	2コマ	114	47%	97	40%	10	4%	8	3%	7	3%	6	2%	242
	3コマ	64	37%	79	45%	10	6%	8	5%	7	4%	6	3%	174
	4コマ	0	0%	8	12%	10	14%	8	12%	8	12%	35	51%	69
	5コマ	47	27%	97	55%	10	6%	8	5%	7	4%	6	3%	175
火曜日	1コマ	0	0%	41	57%	10	14%	8	11%	7	10%	6	8%	72
	2コマ	0	0%	101	68%	10	7%	25	17%	7	5%	6	4%	149
	3コマ	12	8%	98	64%	10	7%	18	12%	7	5%	8	5%	153
	4コマ	24	12%	128	64%	10	5%	8	4%	7	4%	22	11%	199
	5コマ	15	13%	19	16%	10	9%	20	17%	25	22%	27	23%	116
水曜日	1コマ	0	0%	154	83%	10	5%	8	4%	7	4%	6	3%	185
	2コマ	100	39%	87	34%	10	4%	8	3%	7	3%	47	18%	259
	3コマ	0	0%	8	8%	10	10%	68	69%	7	7%	6	6%	99
	4コマ	36	34%	8	8%	41	39%	8	8%	7	7%	6	6%	106
	5コマ	0	0%	8	21%	10	26%	8	21%	7	18%	6	15%	39
木曜日	1コマ	46	17%	142	53%	10	4%	8	3%	7	3%	57	21%	270
	2コマ	71	34%	105	51%	10	5%	8	4%	7	3%	6	3%	207
	3コマ	74	32%	62	26%	10	4%	8	3%	7	3%	73	31%	234
	4コマ	60	22%	136	50%	10	4%	35	13%	7	3%	26	9%	274
	5コマ	5	5%	23	21%	10	9%	13	12%	29	26%	30	27%	110
金曜日	1コマ	0	0%	8	21%	10	26%	8	21%	7	18%	6	15%	39
	2コマ	99	38%	87	34%	10	4%	8	3%	7	3%	48	19%	259
	3コマ	57	22%	128	50%	10	4%	8	3%	7	3%	46	18%	256
	4コマ	105	41%	74	29%	10	4%	8	3%	16	6%	46	18%	259
	5コマ	0	0%	66	58%	10	9%	8	7%	7	6%	22	19%	113

図表 15

後期		一階	%	二階	%	三階	%	四階	%	五階	%	六階	%	合計人数
月曜日	1コマ	49	56%	8	9%	10	11%	8	9%	7	8%	6	7%	88
	2コマ	52	29%	79	44%	10	6%	8	4%	7	4%	25	14%	181
	3コマ	0	0%	151	58%	10	4%	58	22%	7	3%	36	14%	262
	4コマ	93	54%	37	22%	10	6%	12	7%	8	5%	11	6%	171
	5コマ	109	55%	59	30%	10	5%	8	4%	7	4%	6	3%	199
火曜日	1コマ	56	58%	9	9%	10	10%	8	8%	7	7%	6	6%	96
	2コマ	0	0%	70	61%	10	9%	22	19%	7	6%	6	5%	115
	3コマ	41	29%	59	41%	10	7%	20	14%	7	5%	6	4%	143
	4コマ	0	0%	56	39%	10	7%	19	13%	7	5%	53	37%	145
	5コマ	15	12%	26	21%	10	8%	20	16%	23	19%	29	24%	123
水曜日	1コマ	0	0%	127	80%	10	6%	8	5%	7	4%	6	4%	158
	2コマ	100	53%	44	23%	10	5%	8	4%	7	4%	20	11%	189
	3コマ	0	0%	121	80%	10	7%	8	5%	7	5%	6	4%	152
	4コマ	0	0%	134	81%	10	6%	8	5%	7	4%	6	4%	165
	5コマ	0	0%	8	21%	10	26%	8	21%	7	18%	6	15%	39
木曜日	1コマ	67	37%	83	46%	10	6%	8	4%	7	4%	6	3%	181
	2コマ	69	32%	78	36%	10	5%	8	4%	7	3%	42	20%	214
	3コマ	83	42%	59	30%	10	5%	8	4%	7	4%	29	15%	196
	4コマ	55	20%	92	33%	41	15%	65	23%	7	2%	22	8%	282
	5コマ	0	0%	14	12%	10	9%	13	12%	46	41%	30	27%	113
金曜日	1コマ	0	0%	8	21%	10	26%	8	21%	7	18%	6	15%	39
	2コマ	14	8%	109	65%	10	6%	8	5%	7	4%	20	12%	168
	3コマ	88	52%	14	8%	10	6%	45	26%	7	4%	6	4%	170
	4コマ	33	15%	119	55%	10	5%	8	4%	7	3%	40	18%	217
	5コマ	0	0%	58	65%	10	11%	8	9%	7	8%	6	7%	89

図表 16

3.2 それぞれのモデルの結果

避難所要時間の平均値をモデルごとに求め、そこから最多時間の曜日、時間帯を求める。その後、モデル、前後期、曜日、コマをダミー変数、合計人数もしくは各階の人数を独立変数(連続変数)、避難所要時間を従属変数として重回帰分析を行い、それぞれの変数と避難所要時間の関係性を求める。また、今回のp値の有意水準は5%とする。統計分析には青木繁伸先生の作成した統計解析プログラムである「Blackbox」を使用させて頂いた。

3.2.1 モデル1:距離優先モデル

シミュレーションの結果は図表 17. 18. 19. 20 のようになった。

モデル1												単位:秒
前期		1	2	3	4	5	6	7	8	9	10	平均
月	1コマ	100	97	97	96	98	95	98	98	102	102	98.3
月	2コマ	208	208	199	197	205	200	198	203	204	200	202.2
月	3コマ	174	171	161	169	173	176	175	156	164	165	168.4
月	4コマ	150	144	154	140	149	154	154	149	143	144	148.1
月	5コマ	205	199	210	206	211	192	206	208	210	199	204.6
火	1コマ	104	105	102	96	98	97	104	99	112	98	101.5
火	2コマ	225	245	230	218	227	232	238	230	220	232	229.7
火	3コマ	217	223	219	217	219	226	223	216	212	227	219.9
火	4コマ	267	272	270	275	273	273	265	278	284	274	273.1
火	5コマ	161	144	141	154	155	148	139	145	157	146	149
水	1コマ	313	305	315	309	308	304	319	312	320	311	311.6
水	2コマ	225	221	216	221	225	206	217	214	223	217	218.5
水	3コマ	139	137	122	129	122	147	129	133	120	131	130.9
水	4コマ	97	104	98	102	109	97	109	101	103	100	102
水	5コマ	98	98	99	98	99	98	94	96	97	102	97.9
木	1コマ	284	297	280	289	287	278	288	280	292	292	286.7
木	2コマ	229	229	206	214	226	212	224	218	220	214	219.2
木	3コマ	210	209	207	204	202	210	221	197	208	196	206.4
木	4コマ	328	303	292	299	310	289	303	323	296	303	304.6
木	5コマ	146	137	144	153	136	145	138	143	138	141	142.1
金	1コマ	100	98	99	102	97	106	99	103	97	96	99.7
金	2コマ	188	179	193	187	191	186	202	190	188	197	190.1
金	3コマ	263	259	259	257	251	259	274	271	258	266	261.7
金	4コマ	180	176	180	182	185	186	177	174	172	186	179.8
金	5コマ	142	151	137	142	148	141	139	152	147	152	145.1

図表 17

モデル1・前期							単位:秒
平均値	1コマ	2コマ	3コマ	4コマ	5コマ	平均	
月	98.3	202.2	168.4	148.1	204.6	164.32	
火	101.5	229.7	219.9	273.1	149	194.64	
水	311.6	218.5	130.9	102	97.9	172.18	
木	286.7	219.2	206.4	304.6	142.1	231.8	
金	99.7	190.1	261.7	179.8	145.1	175.28	
平均	179.6	211.9	197.5	201.5	147.7	187.644	

図表 18

モデル1												単位:秒
後期		1	2	3	4	5	6	7	8	9	10	平均
月	1コマ	100	101	99	101	100	101	98	99	97	98	99.4
月	2コマ	172	171	170	170	167	175	167	173	167	175	170.7
月	3コマ	368	359	353	367	373	376	348	370	347	359	362
月	4コマ	106	108	103	102	101	103	103	103	116	104	104.9
月	5コマ	141	129	135	120	134	126	128	132	137	128	131
火	1コマ	100	97	100	98	101	96	98	100	97	99	98.6
火	2コマ	165	171	166	175	177	171	169	166	168	178	170.6
火	3コマ	156	133	154	145	153	158	150	144	145	144	148.2
火	4コマ	183	172	173	180	173	182	189	166	166	180	176.4
火	5コマ	157	142	142	152	142	142	149	146	149	142	146.3
水	1コマ	257	264	261	257	262	250	255	259	266	253	258.4
水	2コマ	125	113	114	119	118	121	119	114	116	118	117.7
水	3コマ	244	238	253	248	238	250	250	247	264	245	247.7
水	4コマ	269	271	274	267	266	272	270	274	274	272	270.9
水	5コマ	99	98	99	96	98	96	99	99	98	96	97.8
木	1コマ	186	177	187	177	173	177	182	180	175	176	179
木	2コマ	165	177	173	162	172	176	158	163	173	164	168.3
木	3コマ	176	171	172	190	174	175	182	181	178	178	177.7
木	4コマ	300	293	293	309	300	291	288	290	304	270	293.8
木	5コマ	146	141	146	144	141	142	144	151	135	142	143.2
金	1コマ	98	98	96	100	99	97	98	100	97	99	98.2
金	2コマ	222	227	215	229	228	223	224	231	232	219	225
金	3コマ	114	110	110	104	117	110	113	111	110	104	110.3
金	4コマ	241	241	244	248	236	261	240	240	250	238	243.9
金	5コマ	130	144	130	134	144	137	133	128	142	129	135.1

図表 19

モデル1・後期						単位:秒
平均値	1コマ	2コマ	3コマ	4コマ	5コマ	平均
月	99.4	170.7	362	104.9	131	173.6
火	98.6	170.6	148.2	176.4	146.3	148.02
水	258.4	117.7	247.7	270.9	97.8	198.5
木	179	168.3	177.7	293.8	143.2	192.4
金	98.2	225	110.3	243.9	135.1	162.5
平均	146.7	170.5	209.2	218	130.7	175.004

図表 20

平均避難所要時間が最も長いのは前期では水曜日の1コマ、後期では月曜の3コマであった。年間を通じてでは後期月曜の3コマが最も避難に時間を要している。

3.2.2 モデル2:混雑回避モデル

シミュレーションの結果は図表 21. 22. 23. 24 のようになった。

モデル2												単位:秒
前期		1	2	3	4	5	6	7	8	9	10	平均
月	1コマ	102	106	100	116	100	105	101	102	101	99	103.2
月	2コマ	129	122	138	122	132	131	127	133	119	125	127.8
月	3コマ	107	115	118	111	108	120	115	119	121	114	114.8
月	4コマ	134	137	132	125	124	131	128	147	141	133	133.2
月	5コマ	121	127	133	134	128	147	129	148	127	146	134
火	1コマ	103	100	104	102	105	101	98	99	103	98	101.3
火	2コマ	141	144	147	155	144	142	138	139	149	155	145.4
火	3コマ	153	150	132	142	156	137	143	146	132	156	144.7
火	4コマ	174	157	164	162	177	161	163	161	174	166	165.9
火	5コマ	151	154	149	147	139	143	154	143	149	145	147.4
水	1コマ	216	197	184	189	192	185	204	191	195	200	195.3
水	2コマ	148	152	153	157	153	155	151	152	153	153	152.7
水	3コマ	142	156	139	142	159	150	158	149	143	141	147.9
水	4コマ	110	108	110	105	108	115	107	108	107	118	109.6
水	5コマ	101	99	104	105	100	102	101	113	103	98	102.6
木	1コマ	213	219	211	205	211	216	210	210	216	191	210.2
木	2コマ	154	134	152	145	140	127	137	139	138	148	141.4
木	3コマ	192	189	179	186	192	186	207	191	176	192	189
木	4コマ	202	219	191	186	220	199	200	205	182	181	198.5
木	5コマ	155	148	143	151	150	146	155	156	149	144	149.7
金	1コマ	99	103	101	102	100	100	98	100	99	104	100.6
金	2コマ	154	152	151	148	144	149	145	149	142	145	147.9
金	3コマ	180	178	178	181	182	184	178	178	182	198	181.9
金	4コマ	148	154	145	163	149	141	152	160	156	151	151.9
金	5コマ	128	116	129	128	119	129	126	118	132	126	125.1

図表 21

モデル2・前期						単位:秒
平均値	1コマ	2コマ	3コマ	4コマ	5コマ	平均
月	103.2	127.8	114.8	133.2	134	122.6
火	101.3	145.4	144.7	165.9	147.4	140.94
水	195.3	152.7	147.9	109.6	102.6	141.62
木	210.2	141.4	189	198.5	149.7	177.76
金	100.6	147.9	181.9	151.9	125.1	141.48
平均	142.1	143	155.7	151.8	131.8	144.88

図表 22

モデル2												単位:秒
後期		1	2	3	4	5	6	7	8	9	10	平均
月	1コマ	104	102	102	133	103	111	99	100	107	103	106.4
月	2コマ	147	150	141	131	133	131	137	152	142	135	139.9
月	3コマ	249	241	260	243	241	249	263	249	243	238	247.6
月	4コマ	115	104	102	112	106	107	110	107	104	121	108.8
月	5コマ	105	108	113	106	105	101	107	106	114	104	106.9
火	1コマ	102	101	104	103	106	100	101	99	101	104	102.1
火	2コマ	119	116	112	117	124	112	111	117	115	118	116.1
火	3コマ	106	114	102	114	106	110	109	111	107	112	109.1
火	4コマ	180	174	180	165	167	164	166	179	162	194	173.1
火	5コマ	162	152	162	155	164	147	148	160	146	144	154
水	1コマ	148	145	168	159	160	162	154	175	165	165	160.1
水	2コマ	117	118	116	123	116	124	121	115	112	127	118.9
水	3コマ	171	159	152	156	156	164	159	150	140	157	156.4
水	4コマ	158	174	181	171	172	153	174	176	166	165	169
水	5コマ	100	113	101	100	97	101	104	101	102	105	102.4
木	1コマ	120	123	120	122	114	132	115	115	127	140	122.8
木	2コマ	146	167	168	156	145	153	160	155	148	153	155.1
木	3コマ	119	120	120	122	122	118	124	131	114	123	121.3
木	4コマ	223	204	210	230	203	210	220	225	206	210	214.1
木	5コマ	169	169	166	158	171	160	169	162	160	168	165.2
金	1コマ	101	102	100	101	102	97	104	98	101	104	101
金	2コマ	147	154	138	150	155	160	152	155	160	150	152.1
金	3コマ	120	124	122	115	124	111	123	119	107	119	118.4
金	4コマ	168	166	169	166	168	163	161	163	165	165	165.4
金	5コマ	105	102	112	101	104	102	103	100	99	107	103.5

図表 23

モデル2・後期						単位:秒
平均値	1コマ	2コマ	3コマ	4コマ	5コマ	平均
月	106.4	139.9	247.6	108.8	106.9	141.92
火	102.1	116.1	109.1	173.1	154	130.88
水	160.1	118.9	156.4	169	102.4	141.36
木	122.8	155.1	121.3	214.1	165.2	155.7
金	101	152.1	118.4	165.4	103.5	128.08
平均	118.5	136.4	150.6	166.1	126.4	139.588

図表 24

平均避難所要時間が最も長いのは前期では木曜日の1コマ、後期では月曜の3コマである。年間を通じてでは後期月曜の3コマが最も時間を要している。

3.2.3 データ分析および考察

単純に人が多い時間帯の避難に時間がかかるのであれば、前期後期共に木曜日の4コマの避難所要時間が最長になるはずだ。しかし、どちらのモデルでも別の時間帯の避難所要時間が最長になっている。なぜこのようなことが起きているのかを把握するため、まずモデル、前後期、曜日、コマをダミー変数、合計人数を独立変数、避難所要時間を従属変数

として重回帰分析を行った。結果は図表 25 の通りだ。

**** 全体人数と避難時間の重回帰式 ****							
	偏回帰係数	標準誤差	t値	P値	標準化偏回帰係数	トレランス	分散拡大要因
model2	-39.09	2.400285	16.285563	0	-0.3413141	1	1
kouki	-2.543718	2.414213	1.0536428	0.2923	-0.02221046	0.9884952	1.011639
Tue	24.23921	3.890516	6.2303322	0	0.1693156	0.5947457	1.681391
Wed	29.27696	3.85286	7.5987606	0	0.2045053	0.606428	1.649
Thu	14.24572	3.922245	3.6320315	0.0003	0.09950913	0.5851622	1.708928
Fri	4.733469	3.79782	1.2463646	0.21293	0.0330642	0.6241328	1.602223
koma2	-25.73525	4.197613	6.1309239	0	-0.1797657	0.510906	1.957307
koma3	-4.422205	4.062107	1.0886481	0.27657	-0.03088997	0.5455606	1.832977
koma4	-1.141553	4.104547	0.278119	0.78098	-0.007973968	0.5343372	1.871478
koma5	-5.498597	3.805895	1.444758	0.14884	-0.03840877	0.6214873	1.609043
p.total	0.5946557	0.02397686	24.8012355	0	0.7016672	0.548766	1.82227
定数項	9.59781	5.122857	5.5377753	0			
t値の自由度: 988							
**** 分散分析表 ****							
要因	平方和	自由度	平均平方	F値	P値		
回帰	1856104	11	168736.7	117.1504	0		
残差	1423058	988	1440.343				
全体	3279162	999					
重相関係数: 0.75235							
決定係数(重相関係数の二乗): 0.56603							
自由度調整済み重相関係数の二乗: 0.56120							

図表 25

重相関係数が 0.7 であることから強い相関関係があることはわかるが、決定係数が 0.5 と低く、全体人数だけでは避難所要時間を正確に予測できていないことが判明した。そこで次に、各階層の人数を独立変数として分析を行った。結果は図表 26 の通りだ。

**** 各階人数と避難時間の重回帰式 ****							
	偏回帰係数	標準誤差	t値	P値	標準化偏回帰係数	トレランス	分散拡大要因
model2	-39.09	1.486268	26.3007718	0	-0.3413141	1	1
kouki	-2.82636	1.520283	1.8591017	0.06331	-0.02467834	0.9557531	1.046295
Tue	-11.22393	2.605353	4.3080257	0.00002	-0.07840133	0.5084886	1.966612
Wed	0.05722892	2.569836	0.0222695	0.98224	0.000399755	0.522641	1.913359
Thu	-3.718047	2.752314	1.3508803	0.17704	-0.02597128	0.4556364	2.194732
Fri	-13.94904	2.479575	5.6255755	0	-0.09743673	0.5613839	1.781312
koma2	-12.65113	2.638748	4.7943679	0	-0.08837059	0.4956997	2.01735
koma3	-8.650992	2.728822	3.1702292	0.00157	-0.06042887	0.4635151	2.157427
koma4	-5.416554	2.806557	1.9299635	0.0539	-0.03783568	0.4381941	2.282094
koma5	-12.14322	2.696736	4.5029335	0.00001	-0.08482279	0.4746106	2.106991
p1	-0.1666134	0.02539014	6.5621301	0	-0.1100757	0.5985198	1.670788
p2	0.9204752	0.0191831	47.9836576	0	0.7336385	0.7204335	1.388053
p3	0.5331547	0.153756	3.4675365	0.00055	0.05655866	0.6330169	1.579737
p4	0.9362413	0.06267353	14.9383837	0	0.2361731	0.6737793	1.484166
p5	1.003358	0.1449615	6.9215491	0	0.1224831	0.5378065	1.859405
p6	0.8070627	0.05434912	14.8496004	0	0.2449657	0.6188574	1.615881
定数項	5.28276	3.589118	6.5476806	0			
t値の自由度: 983							
**** 分散分析表 ****							
要因	平方和	自由度	平均平方	F値	P値		
回帰	2736302	16	171018.9	309.6775	0		
残差	542860	983	552.2483				
全体	3279162	999					
重相関係数: 0.91348							
決定係数(重相関係数の二乗): 0.83445							
自由度調整済み重相関係数の二乗: 0.83176							

図表 26

全体人数を独立変数として重回帰分析を行った結果よりも、重相関係数(0.9)、決定係数(0.8)共に増加し避難所要時間をより正確に予測できていることがわかる。このことから、全体人数よりも各階にいる人数の方が避難所要時間に強く影響を及ぼしていることが判明した。また、モデル2の偏回帰係数がマイナスであることからモデル2はモデル1よりも避難にかかる時間が少なく、有用な避難方法であることも判明した。それに加え、ほかのダミー変数の偏回帰係数から後期(-2.8)、金曜日(-13.9)、2コマ(-12.1)が避難所要時間に大きく関与しており、避難所要時間の増加には前期(2.8)、月曜日(0)、水曜日(0.05)、1コマ(0)が関与していることもわかった。

この結果に留意しつつさらに詳しい関連性を調べるために、モデルごと、学期ごとに結果を分割して再度重回帰分析を行った。結果は図表 27. 28. 29. 30 の通りだ。

***** モデル1前期・重回帰式 *****							
	偏回帰係数	標準誤差	t値	P値	標準化偏回帰係数	トレランス	分散拡大要因
Tue	-20.02324	3.575646	5.5998942	0	-0.1222823	0.3364296	2.97239
Wed	0.3782163	3.507123	0.1078423	0.91421	0.002309775	0.3497043	2.859559
Thu	-8.832737	3.672777	2.4049204	0.01695	-0.05394172	0.3188702	3.136073
Fri	-27.4155	3.088624	8.8762823	0	-0.167427	0.4508925	2.217824
koma2	9.793898	3.295234	2.9721398	0.00326	0.05981154	0.3961233	2.524466
koma3	3.377595	3.167525	1.06632	0.28737	0.02062705	0.4287094	2.332582
koma4	18.84156	3.232801	5.8282474	0	0.1150658	0.4115713	2.429713
koma5	-10.283	3.095843	3.3215504	0.00104	-0.0627985	0.448792	2.228204
p1	-0.3054869	0.03818457	8.0002696	0	-0.1787942	0.3211916	3.113407
p2	1.339207	0.02335291	57.3464543	0	0.9780367	0.5515245	1.813156
p3	-0.4282367	0.1977965	2.1650364	0.03139	-0.03971739	0.4766846	2.097823
p4	0.4685753	0.08900279	5.2647264	0	0.09273261	0.5170691	1.933977
p5	1.26994	0.2141096	5.931263	0	0.1089924	0.4750766	2.104924
p6	0.6682669	0.06431551	10.3904455	0	0.2021881	0.4236617	2.360374
定数項	3.97401	3.957548	1.2186954	0			
t値の自由度: 235							
***** 分散分析表 *****							
要因	平方和	自由度	平均平方	F値	P値		
回帰	1032077	14	73719.77	428.4705	0		
残差	40432.53	235	172.0533				
全体	1072509	249					
重相関係数: 0.98097							
決定係数(重相関係数の二乗): 0.96230							
自由度調整済み重相関係数の二乗: 0.96006							

図表 27

***** モデル1後期・重回帰式 *****							
	偏回帰係数	標準誤差	t値	P値	標準化偏回帰係数	トレランス	分散拡大要因
Tue	-8.627481	2.336839	3.6919453	0.00028	-0.04971024	0.4601794	2.173066
Wed	4.444943	2.450545	1.8138588	0.07097	0.02561109	0.4184652	2.389685
Thu	-4.924084	2.764123	1.7814273	0.07613	-0.02837183	0.3289047	3.040395
Fri	-8.224716	2.233088	3.6831136	0.00029	-0.04738957	0.5039334	1.984389
koma2	-25.72755	2.415923	10.6491605	0	-0.1482383	0.4305449	2.322638
koma3	-5.498051	2.662931	2.0646611	0.04005	-0.03167894	0.3543763	2.821859
koma4	-19.61082	2.787168	7.0361097	0	-0.1129946	0.3234882	3.091303
koma5	-18.90617	2.371192	7.9732803	0	-0.1089345	0.4469422	2.237426
p1	-0.2397576	0.02520521	9.5122235	0	-0.1287027	0.4557191	2.194334
p2	1.315939	0.0217969	60.3727805	0	0.8175179	0.4549842	2.197879
p3	1.647581	0.215565	7.6430825	0	0.1441703	0.2344735	4.264875
p4	0.9021332	0.07843618	11.5014945	0	0.2035157	0.2664533	3.753003
p5	0.8398467	0.1209243	6.9452267	0	0.09835819	0.4159683	2.404029
p6	0.7108604	0.0655227	10.84907	0	0.144521	0.4701449	2.127004
定数項	2.74781	3.606905	7.3965802	0			
t値の自由度: 235							
***** 分散分析表 *****							
要因	平方和	自由度	平均平方	F値	P値		
回帰	1181237	14	84374.09	839.3912	0		
残差	23621.78	235	100.5182				
全体	1204859	249					
重相関係数: 0.99015							
決定係数(重相関係数の二乗): 0.98039							
自由度調整済み重相関係数の二乗: 0.97923							

図表 28

***** モデル2前期・重回帰式 *****							
	偏回帰係数	標準誤差	t値	P値	標準化偏回帰係数	トレランス	分散拡大要因
Tue	-11.05387	2.48105	4.4553183	0.00001	-0.1394677	0.3364296	2.97239
Wed	4.142978	2.433504	1.7024745	0.08999	0.05227234	0.3497043	2.859559
Thu	2.345042	2.548447	0.920185	0.35842	0.02958762	0.3188702	3.136073
Fri	-11.19911	2.143118	5.2256178	0	-0.1413003	0.4508925	2.217824
koma2	-10.78596	2.286479	4.7172799	0	-0.1360875	0.3961233	2.524466
koma3	-6.085184	2.197865	2.7686796	0.00608	-0.07677734	0.4287094	2.332582
koma4	-4.079675	2.243159	1.818719	0.07023	-0.05147364	0.4115713	2.429713
koma5	-11.24283	2.148127	5.2337835	0	-0.1418519	0.448792	2.228204
p1	-0.1679789	0.0264953	6.3399499	0	-0.2031167	0.3211916	3.113407
p2	0.5065395	0.01620399	31.2601625	0	0.7642768	0.5515245	1.813156
p3	0.2383467	0.137246	1.7366391	0.08376	0.04567051	0.4766846	2.097823
p4	0.7185676	0.06175677	11.635446	0	0.2937992	0.5170691	1.933977
p5	1.155026	0.1485652	7.7745368	0	0.2048022	0.4750766	2.104924
p6	0.9017993	0.0446269	20.207526	0	0.5636966	0.4236617	2.360374
定数項	2.51489	2.746042	0.0486615	0			
t値の自由度: 235							
***** 分散分析表 *****							
要因	平方和	自由度	平均平方	F値	P値		
回帰	231803.7	14	16557.41	199.879	0		
残差	19466.72	235	82.83713				
全体	251270.4	249					
重相関係数: 0.96048							
決定係数(重相関係数の二乗): 0.92253							
自由度調整済み重相関係数の二乗: 0.91791							

図表 29

***** モデル2後期・重回帰式 *****							
	偏回帰係数	標準誤差	t値	P値	標準化偏回帰係数	トレランス	分散拡大要因
Tue	-11.90764	1.938508	6.142681	0	-0.1282089	0.4601794	2.173066
Wed	7.329226	2.032833	3.605425	0.00038	0.07891336	0.4184652	2.389685
Thu	-11.51782	2.292959	5.023125	0	-0.1240117	0.3289047	3.040395
Fri	-10.18912	1.852442	5.500371	0	-0.1097057	0.5039334	1.984389
koma2	-15.95814	2.004112	7.962696	0	-0.1718203	0.4305449	2.322638
koma3	-15.0379	2.209016	6.80751	0	-0.1619122	0.3543763	2.821859
koma4	-13.32783	2.312076	5.764445	0	-0.1435	0.3234882	3.091303
koma5	-15.0145	1.967006	7.633178	0	-0.1616603	0.4469422	2.237426
p1	-0.1139937	0.02090881	5.451947	0	-0.1143475	0.4557191	2.194334
p2	0.4312651	0.01808147	23.85122	0	0.5006523	0.4549842	2.197879
p3	0.6206617	0.1788204	3.470866	0.00062	0.101488	0.2344735	4.264875
p4	1.047983	0.06506619	16.10641	0	0.4417861	0.2664533	3.753003
p5	1.186344	0.1003119	11.82655	0	0.2596281	0.4159683	2.404029
p6	1.198399	0.0543539	22.04807	0	0.4552798	0.4701449	2.127004
定数項	7.30371	2.992083	5.836084	0			
t値の自由度: 235							
***** 分散分析表 *****							
要因	平方和	自由度	平均平方	F値	P値		
回帰	328789.4	14	23484.96	339.522	0		
残差	16255.13	235	69.17076				
全体	345044.6	249					
重相関係数: 0.97616							
決定係数(重相関係数の二乗): 0.95289							
自由度調整済み重相関係数の二乗: 0.95008							

図表 30

すべての決定係数が 0.9 以上であり、重相関係数に至っては全て 0.95 以上の値が出ている。そのことから、かなり正確に避難所要時間を予測できていることがわかる。また、各分析の偏回帰係数から水曜日、1 コマ目もしくは 4 コマが避難所要時間の増加に大きく関与していることが判明した。しかし、前期後期の水曜日、1 コマ目もしくは 4 コマ目は図表 18. 20. 22. 24 の最も避難所要時間のかかる時間帯にほぼ当てはまらなかった。

それぞれの結果の標準化偏回帰係数を見ると、ほかのダミー変数に比べ各階の人数が避難に強く影響を与えていることがわかる。それに加え、曜日とコマによって変化するのは人数だけのため、各階層の人数だけで重回帰分析を行いどの程度の影響があるのかを分析する。このとき、モデル 1 とモデル 2 では人数が全く同じになるため、モデルごとに結果を分割して分析を行う。結果は図表 31. 32 の通りだ。

**** モデル1・各階人数のみ・重回帰式 ****							
	偏回帰係数	標準誤差	t値	P値	標準化偏回帰係数	トレランス	分散拡大要因
p1	-0.1758556	0.02060569	8.5343235	0	-0.09815028	0.875398	1.142337
p2	1.369018	0.01757999	77.8736225	0	0.9217921	0.8263524	1.210137
p3	0.9127275	0.1277974	7.1419867	0	0.08179765	0.8826873	1.132904
p4	0.9173226	0.05374185	17.0690547	0	0.1954874	0.8827384	1.132838
p5	0.7496815	0.1170674	6.403847	0	0.07731269	0.7943845	1.258836
p6	0.5626998	0.04635076	12.1400346	0	0.1442874	0.819659	1.22002
定数項	3.42088	2.616552	0.416519	0			
t値の自由度: 493							
**** 分散分析表 ****							
要因	平方和	自由度	平均平方	F値	P値		
回帰	2166203	6	361033.9	1357.289	0		
残差	131136.2	493	265.9963				
全体	2297340	499					
重相関係数: 0.97104							
決定係数(重相関係数の二乗): 0.94292							
自由度調整済み重相関係数の二乗: 0.94222							

図表 31

**** モデル2・各階人数のみ・重回帰式 ****							
	偏回帰係数	標準誤差	t値	P値	標準化偏回帰係数	トレランス	分散拡大要因
p1	-0.1388337	0.01457774	9.52368	0	-0.151647	0.875398	1.142337
p2	0.4867651	0.01243717	39.13791	0	0.6414266	0.8263524	1.210137
p3	0.6102969	0.0904118	6.750192	0	0.1070396	0.8826873	1.132904
p4	0.8496608	0.03802031	22.34755	0	0.354361	0.8827384	1.132838
p5	0.9670516	0.08282069	11.67645	0	0.1951764	0.7943845	1.258836
p6	0.9126833	0.03279139	27.83301	0	0.4580107	0.819659	1.22002
定数項	8.24696	1.85111	6.86812	0			
t値の自由度: 493							
**** 分散分析表 ****							
要因	平方和	自由度	平均平方	F値	P値		
回帰	534181.7	6	89030.28	668.738	0		
残差	65633.96	493	133.1318				
全体	599815.6	499					
重相関係数: 0.94370							
決定係数(重相関係数の二乗): 0.89058							
自由度調整済み重相関係数の二乗: 0.88924							

図表 32

決定係数はモデル1では0.94、モデル2では0.89となっており、ダミー変数を含めた重回帰分析よりは値が低くなってしまっているが十分予測できていると言える。モデル1では、偏回帰係数は2階が最も大きく1.369であり、標準化偏回帰係数でも2階が0.92と最も高く、次いで4階(0.19)、6階(0.14)となっている。モデル2では、偏回帰係数は5階が最も大きく0.967であるが、標準化偏回帰係数では、2階(0.64)が最も大きく、次いで6階(0.45)、4階(0.35)となっている。

更に分析を進め実際に避難所要時間の平均が高い時間帯の各階層別人数(図表 15.16)を確認してみると、2階に集まっている人数が極端に多いか、2階に多くの人が集まっているのに加え、4階と6階のどちらか、もしくは両方にも人数が集まっていた。さらに、極端に平均避難所要時間が長い後期の月曜3コマは2、4、6階すべてに多くの人が集まっており、なおかつ避難しやすい1階には誰もいない状況になっていた。

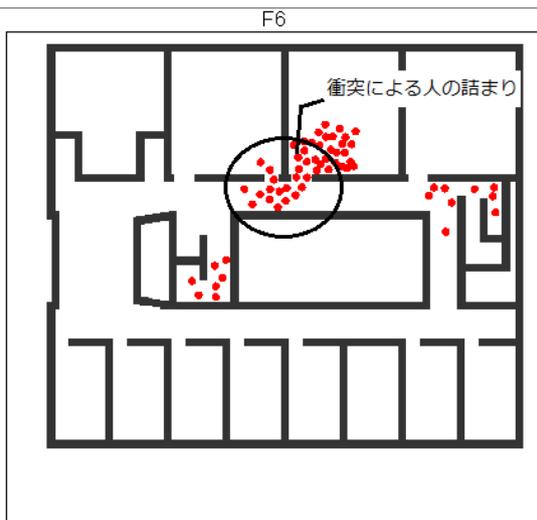
独立変数(合計人数、各階層の人数)の偏回帰係数は一人あたりの避難所要時間である。4

階や 6 階も一人あたりの避難所要時間は長いが、授業配分や教室の許容量の関係で人が極端には集まりづらい。それに比べ、2 階は社情棟最大の教室があり図表 15.16 に示した通り実際にも多くの人が集まっている。それに加え図表 8 を見てもわかるように、最も学生を収容できる 205 教室 (F2 の上部、大きめの教室) の出入口は両方とも非常口近く、距離優先で避難を行うとどうしても非常階段の方へ進んでしまう。そのためモデル 1 では非常階段入口付近で大きな混雑が発生していた (図表 33)。また、モデル 2 では社情棟は全体的に通路が狭いため、避難ルートを変更した人が廊下ですれ違う際に衝突が発生していた (図表 34.35)。

つまり、避難所要時間が人数に比例していない理由は特定の教室に多数の人が集まっているためである。これにより教室から廊下への移動や階段までの移動に時間がかかり、避難所要時間全体の延長をもたらしていた。



図表 33



図表 34



図表 35

第4章 おわりに

4.1 有意義な避難訓練の提案

重回帰分析によってそれぞれのダミー変数並びに独立変数の影響力を求めることができた。それらの結果から、効果的な避難訓練を行うには二つの条件が重要であるとわかった。第一に、モデル2のような混雑具合を判断し空いている方への避難誘導を行うこと。第二に、各階層の人数が特定の条件を満たすこと。具体的には、2階にいる人数が極端に多いとき、もしくは2階に多くの人が集まっており、なおかつ4階6階のどちらか、もしくは両方の階にある程度の人数が集まっている時である。つまり社情棟で今年度のような人数配置となる場合には、後期の月曜日3コマに混雑具合を加味した避難訓練を提案する。そうすれば、災害時に迅速かつ安全に避難をすることができるだろう。

4.2 今後の課題

今回は様々な場所で単純化を行い、あまり具体的な実験を行うことができなかった。例えば、今回のシミュレーションでは避難の阻害要因を壁と人だけに限定しているが本来ならば廊下に出ている机や仕切り板、傘立てなどの障害物についても考えなければならない。それに加え、今回は非常階段から地面についた時点や非常口から棟外へ出た時点を避難完了としたが、本来ならば非常階段は1階の非常口と合流する設計であり人の混雑が最も予想される場所である。しかし、現在では駐輪禁止にもかかわらず自転車が置かれていたり、台車やごみが散乱していたりと避難の阻害要因が数多く放置されている。それらのことを加味するならば、今回のシミュレーションは実際よりも素早く避難が行われていると考えられる。

しかし、本来ならば1階の人々は窓から避難することも可能であり、混雑具合の判断も教室を出た際だけではなく個人ごとに随時行われるため、シミュレーションよりも円滑な避難ができる可能性もある。

これらをプログラムに実装することでより現実的なシミュレーションモデルを作り、研究に用いることがシミュレーションモデル作成における今後の課題である。そして、避難阻害要因の排除や避難シューターの設置などを行い円滑な避難を行える環境をつくることが実際の避難における今後の課題である。

謝辞

この論文を書くにあたり沢山の方々にご協力をいただきました。真摯に論文の指導して下さい富山慶典先生。統計分析についてやさしくご指導して下さり、統計プログラムも提供して下さい青木繁伸先生。プログラムに関してご相談に乗って頂いた細野文雄先生。社会情報学部棟の図面や教室配当人数表をお貸しして下さい学務の皆様。作図のための変換プログラムを作成、提供して下さい北爪隆史様。影ながら支えてくれた家族、共に支え合った同じゼミの仲間たち。そして、シミュレーションプログラムである artisoc を無償で貸与して下さい構造計画研究所様。この場を借りて厚く御礼申し上げます。

参考文献

- 青木繁伸・” Black-Box --- data analysis on the WWW ---Blackbox” ・最終改定日 2002-12-11・〈<http://aoki2.si.gunma-u.ac.jp/BlackBox/BlackBox.html>〉 (2015-1-6)
- 兼田敏之、構造計画研究所構造工学部+名古屋工業大学兼田研究所・2011・
「artisoc で始める歩行者エージェントシミュレーション 原理・方法論から安全・賑わい空間のデザイン・マネジメントまで」・構造計画研究所
構造計画研究所・“MAS コミュニティー” ・最終改定日 2014-11-11・
〈<http://mas.kke.co.jp/index.php>〉・(2015-1-6)
- 城田拓耶・2011・「マルチエージェントによる日案計画を踏まえた教室配置の検証
～芝浦工業大学大宮キャンパス新二号館対象～」・「2010 年総合研究」
- 山影進・2007・「人工社会構築指南 artisoc によるマルチエージェント・
シミュレーション入門」・書籍工房早山

付録

今回のシミュレーションで用いたエージェントのルール(プログラム)を付録として添付する。いくつかのエージェントや歩行ルールは城田(2011)をそのまま使用したため、同じ変数名を使用している部分が多々ある。また、変数作成の都合上、英語のつづりを故意に間違えて使用している部分もある。

付 1-1 人エージェントプログラム (モデル 1)

```
Agt_Init{
my.Direction = rnd() * 360
my.point1=false
my.point2 = false
my.point3 = false
my.point4 = false
}
Agt_Step{
dim C1 as integer
dim C2 as integer
dim C3 as integer
dim C4 as integer
dim C5 as integer
dim C6 as integer
dim C7 as integer
dim C8 as integer
dim C9 as integer
dim C10 as integer
dim destin as double

//自分が何階にいるか//
if my.one == false then
  if my.Layer == 0 then
    my.floor = 0
    my.one = true
  elseif my.Layer == 1 then
    my.floor = 1
    my.one = true
  elseif my.Layer == 2 then
    my.floor = 2
    my.one = true
  elseif my.Layer == 3 then
    my.floor = 3
    my.one = true
  elseif my.Layer == 4 then
    my.floor = 4
    my.one = true
  elseif my.Layer == 5 then
    my.floor = 5
    my.one = true
  end if
end if

//一階の移動ルール//
if my.Layer == 0 then
  if my.down2 == true then
    if my.point10 == true then
      my.Direction = getdirection(my.X , my.Y , 56 , 45 , universe.area)
      destin = my.Direction
      avoidwallandpeople()
    elseif my.point1 == false then
      my.Direction = getdirection(my.X , my.Y , 56 , 33 , universe.area)
```

```

destin = my.Direction
  avoidwallandpeople()
  end if
elseif my.down2 == false then
  if my.point1 == false then
    C1 = MeasureDistance(my.X , my.Y , 21 , 41 , universe.area)
    C2 = MeasureDistance(my.X , my.Y , 39.9 , 41 , universe.area)
    C3 = MeasureDistance(my.X , my.Y , 40.8 , 23 , universe.area)
    C4 = MeasureDistance(my.X , my.Y , 54.5 , 23 , universe.area)
    C5 = MeasureDistance(my.X , my.Y , 21.8 , 38 , universe.area)
    C6 = MeasureDistance(my.X , my.Y , 57 , 33 , universe.area)
    if C1 < C2 and C1 < C3 and C1 < C4 and C1 < C5 and C1 < C6 then
      my.Direction = GetDirection(my.X , my.Y , 21 , 41 , universe.area)
    elseif C2 < C1 and C2 < C3 and C2 < C4 and C2 < C5 and C2 < C6 then
      my.Direction = GetDirection(my.X , my.Y , 39.9 , 41 , universe.area)
    elseif C3 < C1 and C3 < C2 and C3 < C4 and C3 < C5 and C3 < C6 then
      my.Direction = GetDirection(my.X , my.Y , 40.8 , 23 , universe.area)
    elseif C4 < C1 and C4 < C2 and C4 < C3 and C4 < C5 and C4 < C6 then
      my.Direction = GetDirection(my.X , my.Y , 54.5 , 23 , universe.area)
    elseif C5 < C1 and C5 < C2 and C5 < C3 and C5 < C4 and C5 < C6 then
      my.Direction = GetDirection(my.X , my.Y , 21.8 , 38 , universe.area)
    elseif C6 < C1 and 65 < C2 and C6 < C3 and C6 < C4 and C6 < C5 then
      my.Direction = GetDirection(my.X , my.Y , 57 , 33 , universe.area)
    end if
    destin = my.Direction
    avoidwallandpeople()
  elseif my.point2 == false then
    C1 = MeasureDistance(my.X , my.Y , 15 , 40 , universe.area)
    C2 = MeasureDistance(my.X , my.Y , 26 , 38 , universe.area)
    C3 = MeasureDistance(my.X , my.Y , 54 , 26 , universe.area)
    C4 = MeasureDistance(my.X , my.Y , 57 , 35 , universe.area)
    C5 = MeasureDistance(my.X , my.Y , 51 , 41 , universe.area)
    if C1 < C2 and C1 < C3 and C1 < C4 and C1 < C5 then
      my.Direction = GetDirection(my.X , my.Y , 15 , 40 , universe.area)
    elseif C2 < C1 and C2 < C3 and C2 < C4 and C2 < C5 then
      my.Direction = GetDirection(my.X , my.Y , 26 , 38 , universe.area)
    elseif C3 < C1 and C3 < C2 and C3 < C4 and C3 < C5 then
      my.Direction = GetDirection(my.X , my.Y , 54 , 26 , universe.area)
    elseif C4 < C1 and C4 < C2 and C4 < C3 and C4 < C5 then
      my.Direction = GetDirection(my.X , my.Y , 57 , 35 , universe.area)
    elseif C5 < C1 and C5 < C2 and C5 < C3 and C5 < C4 then
      my.Direction = GetDirection(my.X , my.Y , 51 , 41 , universe.area)
    end if
    destin = my.Direction
    avoidwallandpeople()
  elseif my.point3 == false then
    C1 = measureDistance(my.X , my.Y , 12 , 24 , universe.area)
    C2 = measureDistance(my.X , my.Y , 57 , 41 , universe.area)
    C3 = measureDistance(my.X , my.Y , 62 , 29 , universe.area)
    C4 = measureDistance(my.X , my.Y , 25 , 23 , universe.area)
    if C1 < C2 and C1 < C3 and C1 < C4 then
      my.Direction = getdirection(my.X , my.Y , 12 , 24 , universe.area)
    elseif C2 < C1 and C2 < C3 and C2 < C4 then

```

```

my.Direction = getdirection(my.X , my.Y , 57 , 41 , universe.area)
  elseif C3 < C1 and C3 < C2 and C3 < C4 then
    my.Direction = getdirection(my.X , my.Y , 62 , 29 , universe.area)
  elseif C4 < C1 and C4 < C2 and C4 < C3 then
    my.Direction = getdirection(my.X , my.Y , 25 , 23 , universe.area)
  end if
  destin = my.Direction
  avoidwallandpeople()
elseif my.point4 == false then
my.Direction = getdirection(my.X , my.Y , 12 , 11 , universe.area)
destin = my.Direction
avoidwallandpeople()
elseif my.point5 == false then
my.Direction = getdirection(my.X , my.Y , 12 , 3 , universe.area)
destin = my.Direction
avoidwallandpeople()
end if
end if
// 二階通常ルール//
elseif my.Layer == 1 then
  if my.down1 == false and my.down2 == false then
    if my.point1 == false then
      C1 = MeasureDistance(my.X , my.Y , 61 , 23 , universe.area)
      C2 = MeasureDistance(my.X , my.Y , 50 , 23 , universe.area)
      C3 = MeasureDistance(my.X , my.Y , 47 , 23 , universe.area)
      C4 = MeasureDistance(my.X , my.Y , 34 , 24 , universe.area)
      C5 = MeasureDistance(my.X , my.Y , 21 , 23 , universe.area)
      if C1 < C2 and C1 < C3 and C1 < C4 and C1 < C5 then
        my.Direction = GetDirection(my.X , my.Y , 61 , 23 , universe.area)
      elseif C2 < C1 and C2 < C3 and C2 < C4 and C2 < C5 then
        my.Direction = GetDirection(my.X , my.Y , 50 , 23 , universe.area)
      elseif C3 < C1 and C3 < C2 and C3 < C4 and C3 < C5 then
        my.Direction = GetDirection(my.X , my.Y , 47 , 23 , universe.area)
      elseif C4 < C1 and C4 < C2 and C4 < C3 and C4 < C5 then
        my.Direction = GetDirection(my.X , my.Y , 34 , 24 , universe.area)
      elseif C5 < C1 and C5 < C2 and C5 < C3 and C5 < C4 then
        my.Direction = GetDirection(my.X , my.Y , 21 , 23 , universe.area)
      end if
      destin = my.Direction
      avoidwallandpeople()
    elseif my.point2 == false then
      C1 = MeasureDistance(my.X , my.Y , 14 , 26 , universe.area)
      C2 = MeasureDistance(my.X , my.Y , 53.5 , 26 , universe.area)
      C3 = MeasureDistance(my.X , my.Y , 53.5 , 41 , universe.area)
      C4 = MeasureDistance(my.X , my.Y , 41.5 , 41 , universe.area)
      if C1 < C2 and C1 < C3 and C1 < C4 then
        my.Direction = GetDirection(my.X , my.Y , 14 , 26 , universe.area)
      elseif C2 < C1 and C2 < C3 and C2 < C4 then
        my.Direction = GetDirection(my.X , my.Y , 53.5 , 26 , universe.area)
      elseif C3 < C1 and C3 < C2 and C3 < C4 then
        my.Direction = GetDirection(my.X , my.Y , 53.5 , 41 , universe.area)
      elseif C4 < C1 and C4 < C2 and C4 < C3 then
        my.Direction = GetDirection(my.X , my.Y , 41.5 , 41 , universe.area)
      end if
    end if
  end if
end if

```

```

        end if
        destin = my.Direction
        avoidwallandpeople()
    elseif my.point3 == false then
        C1 = measureDistance(my.X , my.Y , 15 , 40 , universe.area)
        C2 = measureDistance(my.X , my.Y , 54 , 41 , universe.area)
        if C1 < C2 then
            my.Direction = getdirection(my.X , my.Y , 15 , 40 , universe.area)
        elseif C2 < C1 then
            my.Direction = getdirection(my.X , my.Y , 54 , 41 , universe.area)
        end if
        destin = my.Direction
        avoidwallandpeople()
    elseif my.point4 == false then
        C1 = measureDistance(my.X , my.Y , 25.5 , 40 , universe.area)
        C2 = measureDistance(my.X , my.Y , 59.5 , 41 , universe.area)
        if C1 < C2 then
            my.Direction = getdirection(my.X , my.Y , 25.5 , 40 , universe.area)
        elseif C2 < C1 then
            my.Direction = getdirection(my.X , my.Y , 59.5 , 41 , universe.area)
        end if
        destin = my.Direction
        avoidwallandpeople()
    elseif my.point4 == true then
        my.Direction = getdirection(my.X , my.Y , 27 , 28 , universe.area)
        destin = my.Direction
        avoidwallandpeople()
    end if
//非常階段//
    elseif my.down2 == true then
        if my.point12 == true then
            my.Direction = getdirection(my.X , my.Y , 60 , 36 , universe.area)
            destin = my.Direction
            avoidwallandpeople()
        elseif my.point11 == true then
            my.Direction = getdirection(my.X , my.Y , 60 , 41 , universe.area)
            destin = my.Direction
            avoidwallandpeople()
        elseif my.point10 == true then
            my.Direction = getdirection(my.X , my.Y , 56.5 , 42 , universe.area)
            destin = my.Direction
            avoidwallandpeople()
        elseif my.point1 == true then
            my.Direction = getdirection(my.X , my.Y , 58 , 41 , universe.area)
            destin = my.Direction
            avoidwallandpeople()
        elseif my.point1 == false then
            my.Direction = getdirection(my.X , my.Y , 56 , 33 , universe.area)
            destin = my.Direction
            avoidwallandpeople()
        end if
//階段//
    elseif my.down1 == true then

```

```

if my.point13 == true then
    my.Direction = getdirection(my.X , my.Y , 22 , 31 , universe.area)
    destin = my.Direction
    avoidwallandpeople()
elseif my.point12 == true then
    my.Direction = getdirection(my.X , my.Y , 22 , 28 , universe.area)
    destin = my.Direction
    avoidwallandpeople()
elseif my.point11 == true then
    my.Direction = getdirection(my.X , my.Y , 26 , 27 , universe.area)
    destin = my.Direction
    avoidwallandpeople()
elseif my.point10 == true then
    my.Direction = getdirection(my.X , my.Y , 27 , 36 , universe.area)
    destin = my.Direction
    avoidwallandpeople()
elseif my.point1 == false and my.point10 == false then
    my.Direction = getdirection(my.X , my.Y , 21 , 36 , universe.area)
    destin = my.Direction
    avoidwallandpeople()
elseif my.point11 == false then
    my.Direction = getdirection(my.X , my.Y , 26 , 36 , universe.area)
    destin = my.Direction
    avoidwallandpeople()
end if
end if

// 三階通常ルール//
elseif my.Layer == 2 then
    if my.down1 == false and my.down2 == false then
        if my.point1 == false then
            C1 = MeasureDistance(my.X , my.Y , 15 , 23 , universe.area)
            C2 = MeasureDistance(my.X , my.Y , 25 , 23 , universe.area)
            C3 = MeasureDistance(my.X , my.Y , 35.5 , 23 , universe.area)
            if C1 < C2 and C1 < C3 then
                my.Direction = GetDirection(my.X , my.Y , 15 , 23 , universe.area)
            elseif C2 < C3 and C2 < C3 then
                my.Direction = GetDirection(my.X , my.Y , 25 , 23 , universe.area)
            elseif C3 < C1 and C3 < C2 then
                my.Direction = GetDirection(my.X , my.Y , 35.5 , 23 , universe.area)
            end if
            destin = my.Direction
            avoidwallandpeople()
        elseif my.point2 == false and my.point3 == false then
            C1 = MeasureDistance(my.X , my.Y , 13 , 26 , universe.area)
            C2 = MeasureDistance(my.X , my.Y , 50 , 24 , universe.area)
            C3 = MeasureDistance(my.X , my.Y , 36 , 41 , universe.area)
            C4 = MeasureDistance(my.X , my.Y , 53.5 , 41 , universe.area)
            if C1 < C2 and C1 < C3 and C1 < C4 then
                my.Direction = GetDirection(my.X , my.Y , 13 , 26 , universe.area)
            elseif C2 < C1 and C2 < C3 and C2 < C4 then
                my.Direction = GetDirection(my.X , my.Y , 50 , 24 , universe.area)
            elseif C3 < C1 and C3 < C2 and C3 < C4 then

```

```

my.Direction = GetDirection(my.X , my.Y , 36, 41 , universe.area)
elseif C4 < C1 and C4 < C2 and C4 < C3 then
my.Direction = GetDirection(my.X , my.Y , 53.5 , 41 , universe.area)
end if
destin = my.Direction
avoidwallandpeople()
elseif my.point3 == false then
C1 = measureDistance(my.X , my.Y , 13.5 , 40 , universe.area)
C2 = measureDistance(my.X , my.Y , 53.5 , 27 , universe.area)
if C1 < C2 then
my.Direction = getdirection(my.X , my.Y , 13.5 , 40 , universe.area)
elseif C2 < C1 then
my.Direction = getdirection(my.X , my.Y , 53.5 , 27 , universe.area)
end if
destin = my.Direction
avoidwallandpeople()
elseif my.point4 == false then
C1 = measureDistance(my.X , my.Y , 25.5 , 40 , universe.area)
C2 = measureDistance(my.X , my.Y , 53.5 , 41 , universe.area)
if C1 < C2 then
my.Direction = getdirection(my.X , my.Y , 25.5 , 40 , universe.area)
elseif C2 < C1 then
my.Direction = getdirection(my.X , my.Y , 53.5 , 41 , universe.area)
end if
destin = my.Direction
avoidwallandpeople()
elseif my.point4 == true then
C1 = measureDistance(my.X , my.Y , 26 , 28 , universe.area)
C2 = measureDistance(my.X , my.Y , 59.5 , 41 , universe.area)
if C1 < C2 then
my.Direction = getdirection(my.X , my.Y , 26 , 28 , universe.area)
elseif C2 < C1 then
my.Direction = getdirection(my.X , my.Y , 59.5 , 41 , universe.area)
end if
destin = my.Direction
avoidwallandpeople()
end if
//非常階段//
elseif my.down2 == true then
if my.point12 == true then
my.Direction = getdirection(my.X , my.Y , 60 , 36 , universe.area)
destin = my.Direction
avoidwallandpeople()
elseif my.point11 == true then
my.Direction = getdirection(my.X , my.Y , 60 , 41 , universe.area)
destin = my.Direction
avoidwallandpeople()
elseif my.point10 == true then
my.Direction = getdirection(my.X , my.Y , 56.5 , 42 , universe.area)
destin = my.Direction
avoidwallandpeople()
elseif my.point1 == true then
my.Direction = getdirection(my.X , my.Y , 58 , 41 , universe.area)

```

```

    destin = my.Direction
    avoidwallandpeople()
    elseif my.point1 == false then
    my.Direction = getdirection(my.X , my.Y , 56 , 33 , universe.area)
    destin = my.Direction
    avoidwallandpeople()
    end if
//階段//
elseif my.down1 == true then
if my.point13 == true then
    my.Direction = getdirection(my.X , my.Y , 22 , 31 , universe.area)
    destin = my.Direction
    avoidwallandpeople()
    elseif my.point12 == true then
    my.Direction = getdirection(my.X , my.Y , 22 , 28 , universe.area)
    destin = my.Direction
    avoidwallandpeople()
    elseif my.point11 == true then
    my.Direction = getdirection(my.X , my.Y , 26 , 27 , universe.area)
    destin = my.Direction
    avoidwallandpeople()
    elseif my.point10 == true then
    my.Direction = getdirection(my.X , my.Y , 27 , 36 , universe.area)
    destin = my.Direction
    avoidwallandpeople()
    elseif my.point1 == false and my.point10 == false then
    my.Direction = getdirection(my.X , my.Y , 21 , 36 , universe.area)
    destin = my.Direction
    avoidwallandpeople()
    elseif my.point11 == false then
    my.Direction = getdirection(my.X , my.Y , 26 , 36 , universe.area)
    destin = my.Direction
    avoidwallandpeople()
    end if
end if

// 四階通常ルール//
elseif my.Layer == 3 then
if my.down1 == false and my.down2 == false then
    if my.point1 == false then
    C1 = MeasureDistance(my.X , my.Y , 6.5 , 23 , universe.area)
    C2 = MeasureDistance(my.X , my.Y , 21 , 23 , universe.area)
    C3 = MeasureDistance(my.X , my.Y , 35 , 23 , universe.area)
    C4 = MeasureDistance(my.X , my.Y , 49.5 , 23 , universe.area)
    if C1 < C2 and C1 < C3 and C1 < C4 then
    my.Direction = GetDirection(my.X , my.Y , 6.5 , 23 , universe.area)
    elseif C2 < C3 and C2 < C3 and C2 < C4 then
    my.Direction = GetDirection(my.X , my.Y , 21 , 23 , universe.area)
    elseif C3 < C1 and C3 < C2 and C3 < C4 then
    my.Direction = GetDirection(my.X , my.Y , 35 , 23 , universe.area)
    elseif C4 < C1 and C4 < C2 and C4 < C3 then
    my.Direction = GetDirection(my.X , my.Y , 49.5 , 23 , universe.area)
    end if

```

```

    destin = my.Direction
    avoidwallandpeople()
elseif my.point2 == false then
C1 = MeasureDistance(my.X , my.Y , 7 , 24 , universe.area)
C2 = MeasureDistance(my.X , my.Y , 13.5 , 23 , universe.area)
C3 = MeasureDistance(my.X , my.Y , 20 , 24 , universe.area)
C4 = MeasureDistance(my.X , my.Y , 28 , 23 , universe.area)
C5 = MeasureDistance(my.X , my.Y , 35.5 , 24 , universe.area)
C6 = MeasureDistance(my.X , my.Y , 42.5 , 23 , universe.area)
C7 = MeasureDistance(my.X , my.Y , 49.5 , 24 , universe.area)
C8 = MeasureDistance(my.X , my.Y , 57 , 23 , universe.area)
C9 = MeasureDistance(my.X , my.Y , 28 , 41 , universe.area)
C10 = MeasureDistance(my.X , my.Y , 54 , 41 , universe.area)
    if C1 < C2 and C1 < C3 and C1 < C4 and C1 < C5 and C1 < C6
    and C1 < C7 and C1 < C8 and C1 < C9 and C1 < C10 then
        my.Direction = GetDirection(my.X , my.Y , 7 , 24 , universe.area)
    elseif C2 < C1 and C2 < C3 and C2 < C4 and C2 < C5 and C2 < C6
    and C2 < C7 and C2 < C8 and C2 < C9 and C2 < C10 then
        my.Direction = GetDirection(my.X , my.Y , 13.5 , 23 , universe.area)
    elseif C3 < C1 and C3 < C2 and C3 < C4 and C3 < C5 and C3 < C6
    and C3 < C7 and C3 < C8 and C3 < C9 and C3 < C10 then
        my.Direction = GetDirection(my.X , my.Y , 20 , 24 , universe.area)
    elseif C4 < C1 and C4 < C2 and C4 < C3 and C4 < C5 and C4 < C6
    and C4 < C7 and C4 < C8 and C4 < C9 and C4 < C10 then
        my.Direction = GetDirection(my.X , my.Y , 28 , 23 , universe.area)
    elseif C5 < C1 and C5 < C2 and C5 < C3 and C5 < C4 and C5 < C6
    and C5 < C7 and C5 < C8 and C5 < C9 and C5 < C10 then
        my.Direction = GetDirection(my.X , my.Y , 35.5 , 24 , universe.area)
    elseif C6 < C1 and C6 < C2 and C6 < C3 and C6 < C4 and C6 < C5
    and C6 < C7 and C6 < C8 and C6 < C9 and C6 < C10 then
        my.Direction = GetDirection(my.X , my.Y , 42.5 , 23 , universe.area)
    elseif C7 < C1 and C7 < C2 and C7 < C3 and C7 < C4 and C7 < C5
    and C7 < C6 and C7 < C8 and C7 < C9 and C7 < C10 then
        my.Direction = GetDirection(my.X , my.Y , 49.5 , 24 , universe.area)
    elseif C8 < C1 and C8 < C2 and C8 < C3 and C8 < C4 and C8 < C5
    and C8 < C6 and C8 < C7 and C8 < C9 and C8 < C10 then
        my.Direction = GetDirection(my.X , my.Y , 57 , 23 , universe.area)
    elseif C9 < C1 and C9 < C2 and C9 < C3 and C9 < C4 and C9 < C5
    and C9 < C6 and C9 < C7 and C9 < C8 and C9 < C10 then
        my.Direction = GetDirection(my.X , my.Y , 28 , 41 , universe.area)
    elseif C10 < C1 and C10 < C2 and C10 < C3 and C10 < C4 and C10 < C5
    and C10 < C6 and C10 < C7 and C10 < C8 and C10 < C9 then
        my.Direction = GetDirection(my.X , my.Y , 54 , 41 , universe.area)
    end if
    destin = my.Direction
    avoidwallandpeople()
elseif my.point3 == false and my.point4 == false and my.point5 == false then
C1 = measureDistance(my.X , my.Y , 13 , 26 , universe.area)
C2 = measureDistance(my.X , my.Y , 53.5 , 26 , universe.area)
C3 = measureDistance(my.X , my.Y , 25.5 , 40 , universe.area)
C4 = measureDistance(my.X , my.Y , 54 , 41 , universe.area)
    if C1 < C2 and C1 < C3 and C1 < C4 then
        my.Direction = getdirection(my.X , my.Y , 13 , 26 , universe.area)

```

```

elseif C2 < C1 and C2 < C3 and C2 < C4 then
my.Direction = getdirection(my.X , my.Y , 53.5 , 26 , universe.area)
elseif C3 < C1 and C3 < C2 and C3 < C4 then
my.Direction = getdirection(my.X , my.Y , 25.5 , 40 , universe.area)
elseif C4 < C1 and C4 < C2 and C4 < C3 then
my.Direction = getdirection(my.X , my.Y , 54 , 41 , universe.area)
end if
destin = my.Direction
avoidwallandpeople()
elseif my.point4 == false and my.point3 == true then
C1 = measureDistance(my.X , my.Y , 13.5 , 40 , universe.area)
C2 = measureDistance(my.X , my.Y , 54 , 41 , universe.area)
if C1 < C2 then
my.Direction = getdirection(my.X , my.Y , 13.5 , 40 , universe.area)
elseif C2 < C1 then
my.Direction = getdirection(my.X , my.Y , 54 , 41 , universe.area)
end if
destin = my.Direction
avoidwallandpeople()
elseif my.point5 == false and my.point4 == true then
C1 = measureDistance(my.X , my.Y , 25.5 , 40 , universe.area)
C2 = measureDistance(my.X , my.Y , 59.5 , 41 , universe.area)
if C1 < C2 then
my.Direction = getdirection(my.X , my.Y , 25.5 , 40 , universe.area)
elseif C2 < C1 then
my.Direction = getdirection(my.X , my.Y , 59.5 , 41 , universe.area)
end if
destin = my.Direction
avoidwallandpeople()
elseif my.point5 == true then
my.Direction = getdirection(my.X , my.Y , 26 , 36 , universe.area)
destin = my.Direction
avoidwallandpeople()
end if
//非常階段//
elseif my.down2 == true then
if my.point12 == true then
my.Direction = getdirection(my.X , my.Y , 60 , 36 , universe.area)
destin = my.Direction
avoidwallandpeople()
elseif my.point11 == true then
my.Direction = getdirection(my.X , my.Y , 60 , 41 , universe.area)
destin = my.Direction
avoidwallandpeople()
elseif my.point10 == true then
my.Direction = getdirection(my.X , my.Y , 56.5 , 42 , universe.area)
destin = my.Direction
avoidwallandpeople()
elseif my.point1 == true then
my.Direction = getdirection(my.X , my.Y , 58 , 41 , universe.area)
destin = my.Direction
avoidwallandpeople()
elseif my.point1 == false then

```

```

my.Direction = getdirection(my.X , my.Y , 56 , 33 , universe.area)
destin = my.Direction
avoidwallandpeople()
end if
//階段//
elseif my.down1 == true then
if my.point13 == true then
my.Direction = getdirection(my.X , my.Y , 22 , 31 , universe.area)
destin = my.Direction
avoidwallandpeople()
elseif my.point12 == true then
my.Direction = getdirection(my.X , my.Y , 22 , 28 , universe.area)
destin = my.Direction
avoidwallandpeople()
elseif my.point11 == true then
my.Direction = getdirection(my.X , my.Y , 26 , 27 , universe.area)
destin = my.Direction
avoidwallandpeople()
elseif my.point10 == true then
my.Direction = getdirection(my.X , my.Y , 27 , 36 , universe.area)
destin = my.Direction
avoidwallandpeople()
elseif my.point1 == false and my.point10 == false then
my.Direction = getdirection(my.X , my.Y , 21 , 36 , universe.area)
destin = my.Direction
avoidwallandpeople()
elseif my.point11 == false then
my.Direction = getdirection(my.X , my.Y , 26 , 36 , universe.area)
destin = my.Direction
avoidwallandpeople()
end if
end if
// 五階通常ルール//
elseif my.Layer == 4 then
if my.down1 == false and my.down2 == false then
if my.point1 == false then
C1 = MeasureDistance(my.X , my.Y , 6.5 , 23 , universe.area)
C2 = MeasureDistance(my.X , my.Y , 21 , 23 , universe.area)
C3 = MeasureDistance(my.X , my.Y , 49.5 , 23 , universe.area)
C4 = MeasureDistance(my.X , my.Y , 35 , 41 , universe.area)
if C1 < C2 and C1 < C3 and C1 < C4 then
my.Direction = GetDirection(my.X , my.Y , 6.5 , 23 , universe.area)
elseif
C2 < C3 and C2 < C3 and C2 < C4 then
my.Direction = GetDirection(my.X , my.Y , 21 , 23 , universe.area)
elseif
C3 < C1 and C3 < C2 and C3 < C4 then
my.Direction = GetDirection(my.X , my.Y , 49.5 , 23 , universe.area)
elseif
C4 < C1 and C4 < C2 and C4 < C3 then
my.Direction = GetDirection(my.X , my.Y , 35 , 41 , universe.area)
end if
destin = my.Direction

```

```

avoidwallandpeople()
elseif my.point2 == false then
C1 = MeasureDistance(my.X , my.Y , 7 , 24 , universe.area)
C2 = MeasureDistance(my.X , my.Y , 13.5 , 23 , universe.area)
C3 = MeasureDistance(my.X , my.Y , 20 , 24 , universe.area)
C4 = MeasureDistance(my.X , my.Y , 28 , 23 , universe.area)
C5 = MeasureDistance(my.X , my.Y , 42.5 , 23 , universe.area)
C6 = MeasureDistance(my.X , my.Y , 49.5 , 24 , universe.area)
C7 = MeasureDistance(my.X , my.Y , 57 , 23 , universe.area)
C8 = MeasureDistance(my.X , my.Y , 21.5 , 41 , universe.area)
C9 = MeasureDistance(my.X , my.Y , 32 , 41 , universe.area)
C10 = MeasureDistance(my.X , my.Y , 54 , 41 , universe.area)
  if C1 < C2 and C1 < C3 and C1 < C4 and C1 < C5 and C1 < C6
  and C1 < C7 and C1 < C8 and C1 < C9 and C1 < C10 then
    my.Direction = GetDirection(my.X , my.Y , 7 , 24 , universe.area)
  elseif C2 < C1 and C2 < C3 and C2 < C4 and C2 < C5 and C2 < C6
  and C2 < C7 and C2 < C8 and C2 < C9 and C2 < C10 then
    my.Direction = GetDirection(my.X , my.Y , 13.5 , 23 , universe.area)
  elseif C3 < C1 and C3 < C2 and C3 < C4 and C3 < C5 and C3 < C6
  and C3 < C7 and C3 < C8 and C3 < C9 and C3 < C10 then
    my.Direction = GetDirection(my.X , my.Y , 20 , 24 , universe.area)
  elseif C4 < C1 and C4 < C2 and C4 < C3 and C4 < C5 and C4 < C6
  and C4 < C7 and C4 < C8 and C4 < C9 and C4 < C10 then
    my.Direction = GetDirection(my.X , my.Y , 28 , 23 , universe.area)
  elseif C5 < C1 and C5 < C2 and C5 < C3 and C5 < C4 and C5 < C6
  and C5 < C7 and C5 < C8 and C5 < C9 and C5 < C10 then
    my.Direction = GetDirection(my.X , my.Y , 42.5 , 23 , universe.area)
  elseif C6 < C1 and C6 < C2 and C6 < C3 and C6 < C4 and C6 < C5
  and C6 < C7 and C6 < C8 and C6 < C9 and C6 < C10 then
    my.Direction = GetDirection(my.X , my.Y , 49.5 , 24 , universe.area)
  elseif C7 < C1 and C7 < C2 and C7 < C3 and C7 < C4 and C7 < C5
  and C7 < C6 and C7 < C8 and C7 < C9 and C7 < C10 then
    my.Direction = GetDirection(my.X , my.Y , 57 , 23 , universe.area)
  elseif C8 < C1 and C8 < C2 and C8 < C3 and C8 < C4 and C8 < C5
  and C8 < C6 and C8 < C7 and C8 < C9 and C8 < C10 then
    my.Direction = GetDirection(my.X , my.Y , 21.5 , 41 , universe.area)
  elseif C9 < C1 and C9 < C2 and C9 < C3 and C9 < C4 and C9 < C5
  and C9 < C6 and C9 < C7 and C9 < C8 and C9 < C10 then
    my.Direction = GetDirection(my.X , my.Y , 32 , 41 , universe.area)
  elseif C10 < C1 and C10 < C2 and C10 < C3 and C10 < C4 and C10 < C5
  and C10 < C6 and C10 < C7 and C10 < C8 and C10 < C9 then
    my.Direction = GetDirection(my.X , my.Y , 54 , 41 , universe.area)
  end if
  destin = my.Direction
avoidwallandpeople()
elseif my.point3 == false and my.point4 == false and my.point5 == false then
C1 = measureDistance(my.X , my.Y , 13 , 26 , universe.area)
C2 = measureDistance(my.X , my.Y , 53.5 , 26 , universe.area)
C3 = measureDistance(my.X , my.Y , 25.5 , 40 , universe.area)
C4 = measureDistance(my.X , my.Y , 12 , 4 , universe.area)
  if C1 < C2 and C1 < C3 and C1 < C4 then
    my.Direction = getdirection(my.X , my.Y , 13 , 26 , universe.area)
  elseif C2 < C1 and C2 < C3 and C2 < C4 then

```

```

my.Direction = getdirection(my.X , my.Y , 53.5 , 26 , universe.area)
elseif C3 < C1 and C3 < C2 and C3 < C4 then
my.Direction = getdirection(my.X , my.Y , 25.5 , 40 , universe.area)
elseif C4 < C1 and C4 < C2 and C4 < C3 then
my.Direction = getdirection(my.X , my.Y , 12 , 4 , universe.area)
end if
destin = my.Direction
avoidwallandpeople()
elseif my.point4 == false and my.point3 == true then
C1 = measureDistance(my.X , my.Y , 13.5 , 40 , universe.area)
C2 = measureDistance(my.X , my.Y , 54 , 41 , universe.area)
if C1 < C2 then
my.Direction = getdirection(my.X , my.Y , 13.5 , 40 , universe.area)
elseif C2 < C1 then
my.Direction = getdirection(my.X , my.Y , 54 , 41 , universe.area)
end if
destin = my.Direction
avoidwallandpeople()
elseif my.point5 == false and my.point4 == true then
C1 = measureDistance(my.X , my.Y , 25.5 , 40 , universe.area)
C2 = measureDistance(my.X , my.Y , 59.5 , 41 , universe.area)
if C1 < C2 then
my.Direction = getdirection(my.X , my.Y , 25.5 , 40 , universe.area)
elseif C2 < C1 then
my.Direction = getdirection(my.X , my.Y , 59.5 , 41 , universe.area)
end if
destin = my.Direction
avoidwallandpeople()
elseif my.point5 == true then
my.Direction = getdirection(my.X , my.Y , 26 , 36 , universe.area)
destin = my.Direction
avoidwallandpeople()
end if
//非常階段//
elseif my.down2 == true then
if my.point12 == true then
my.Direction = getdirection(my.X , my.Y , 60 , 36 , universe.area)
destin = my.Direction
avoidwallandpeople()
elseif my.point11 == true then
my.Direction = getdirection(my.X , my.Y , 60 , 41 , universe.area)
destin = my.Direction
avoidwallandpeople()
elseif my.point10 == true then
my.Direction = getdirection(my.X , my.Y , 56.5 , 42 , universe.area)
destin = my.Direction
avoidwallandpeople()
elseif my.point1 == true then
my.Direction = getdirection(my.X , my.Y , 58 , 41 , universe.area)
destin = my.Direction
avoidwallandpeople()
elseif my.point1 == false then
my.Direction = getdirection(my.X , my.Y , 56 , 33 , universe.area)

```

```

    destin = my.Direction
    avoidwallandpeople()
end if
//階段//
elseif my.down1 == true then
if my.point13 == true then
    my.Direction = getdirection(my.X , my.Y , 22 , 31 , universe.area)
    destin = my.Direction
    avoidwallandpeople()
    elseif my.point12 == true then
    my.Direction = getdirection(my.X , my.Y , 22 , 28 , universe.area)
    destin = my.Direction
    avoidwallandpeople()
    elseif my.point11 == true then
    my.Direction = getdirection(my.X , my.Y , 26 , 27 , universe.area)
    destin = my.Direction
    avoidwallandpeople()
    elseif my.point10 == true then
    my.Direction = getdirection(my.X , my.Y , 27 , 36 , universe.area)
    destin = my.Direction
    avoidwallandpeople()
    elseif my.point1 == false and my.point10 == false then
    my.Direction = getdirection(my.X , my.Y , 21 , 36 , universe.area)
    destin = my.Direction
    avoidwallandpeople()
    elseif my.point11 == false then
    my.Direction = getdirection(my.X , my.Y , 26 , 36 , universe.area)
    destin = my.Direction
    avoidwallandpeople()
    end if
end if
// 六階行動ルール//
elseif my.Layer == 5 then
if my.down1 == false and my.down2 == false then
    if my.point1 == false then
    C1 = MeasureDistance(my.X , my.Y , 6.5 , 23 , universe.area)
    C2 = MeasureDistance(my.X , my.Y , 21 , 23 , universe.area)
    C3 = MeasureDistance(my.X , my.Y , 35 , 23 , universe.area)
    C4 = MeasureDistance(my.X , my.Y , 36 , 41 , universe.area)
    if C1 < C2 and C1 < C3 and C1 < C4 then
    my.Direction = GetDirection(my.X , my.Y , 6.5 , 23 , universe.area)
    elseif C2 < C1 and C2 < C3 and C2 < C4 then
    my.Direction = GetDirection(my.X , my.Y , 21 , 23 , universe.area)
    elseif C3 < C1 and C3 < C2 and C3 < C4 then
    my.Direction = GetDirection(my.X , my.Y , 35 , 23 , universe.area)
    elseif C4 < C1 and C4 < C2 and C4 < C3 then
    my.Direction = GetDirection(my.X , my.Y , 36 , 41 , universe.area)
    end if
    destin = my.Direction
    avoidwallandpeople()
    elseif my.point2 == false then
    C1 = MeasureDistance(my.X , my.Y , 7 , 24 , universe.area)
    C2 = MeasureDistance(my.X , my.Y , 13.5 , 23 , universe.area)

```

```

C3 = MeasureDistance(my.X , my.Y , 20 , 24 , universe.area)
C4 = MeasureDistance(my.X , my.Y , 28 , 23 , universe.area)
C5 = MeasureDistance(my.X , my.Y , 57 , 23 , universe.area)
C6 = MeasureDistance(my.X , my.Y , 35 , 24 , universe.area)
C7 = MeasureDistance(my.X , my.Y , 21.5 , 41 , universe.area)
C8 = MeasureDistance(my.X , my.Y , 32 , 41 , universe.area)
C9 = MeasureDistance(my.X , my.Y , 50 , 41 , universe.area)
  if C1 < C2 and C1 < C3 and C1 < C4 and C1 < C5 and C1 < C6
  and C1 < C7 and C1 < C8 and C1 < C9 then
  my.Direction = GetDirection(my.X , my.Y , 7 , 24 , universe.area)
  elseif C2 < C1 and C2 < C3 and C2 < C4 and C2 < C5 and C2 < C6
  and C2 < C7 and C2 < C8 and C2 < C9 then
  my.Direction = GetDirection(my.X , my.Y , 13.5 , 23 , universe.area)
  elseif C3 < C1 and C3 < C2 and C3 < C4 and C3 < C5 and C3 < C6
  and C3 < C7 and C3 < C8 and C3 < C9 then
  my.Direction = GetDirection(my.X , my.Y , 20 , 24 , universe.area)
  elseif C4 < C1 and C4 < C2 and C4 < C3 and C4 < C5 and C4 < C6
  and C4 < C7 and C4 < C8 and C4 < C9 then
  my.Direction = GetDirection(my.X , my.Y , 28 , 23 , universe.area)
  elseif C5 < C1 and C5 < C2 and C5 < C3 and C5 < C4 and C5 < C6
  and C5 < C7 and C5 < C8 and C5 < C9 then
  my.Direction = GetDirection(my.X , my.Y , 57 , 23 , universe.area)
  elseif C6 < C1 and C6 < C2 and C6 < C3 and C6 < C4 and C6 < C5
  and C6 < C7 and C6 < C8 and C6 < C9 then
  my.Direction = GetDirection(my.X , my.Y , 35 , 24 , universe.area)
  elseif C7 < C1 and C7 < C2 and C7 < C3 and C7 < C4 and C7 < C5
  and C7 < C6 and C7 < C8 and C7 < C9 then
  my.Direction = GetDirection(my.X , my.Y , 21.5 , 41 , universe.area)
  elseif C8 < C1 and C8 < C2 and C8 < C3 and C8 < C4 and C8 < C5
  and C8 < C6 and C8 < C7 and C8 < C9 then
  my.Direction = GetDirection(my.X , my.Y , 32 , 41 , universe.area)
  elseif C9 < C1 and C9 < C2 and C9 < C3 and C9 < C4 and C9 < C5
  and C9 < C6 and C9 < C7 and C9 < C8 then
  my.Direction = GetDirection(my.X , my.Y , 50 , 41 , universe.area)
  end if
  destin = my.Direction
  avoidwallandpeople()
elseif my.point3 == false and my.point4 == false and my.point5 == false then
C1 = measureDistance(my.X , my.Y , 13 , 26 , universe.area)
C2 = measureDistance(my.X , my.Y , 53.5 , 26 , universe.area)
C3 = measureDistance(my.X , my.Y , 25 , 40 , universe.area)
C4 = measureDistance(my.X , my.Y , 54 , 41 , universe.area)
  if C1 < C2 and C1 < C3 and C1 < C4 then
  my.Direction = getdirection(my.X , my.Y , 13 , 26 , universe.area)
  elseif C2 < C1 and C2 < C3 and C2 < C4 then
  my.Direction = getdirection(my.X , my.Y , 53.5 , 26 , universe.area)
  elseif C3 < C1 and C3 < C2 and C3 < C4 then
  my.Direction = getdirection(my.X , my.Y , 25 , 40 , universe.area)
  elseif C4 < C1 and C4 < C2 and C4 < C3 then
  my.Direction = getdirection(my.X , my.Y , 54 , 41 , universe.area)
  end if
  destin = my.Direction
  avoidwallandpeople()

```

```

elseif my.point4 == false and my.point3 == true then
    C1 = measureDistance(my.X , my.Y , 13.5 , 40 , universe.area)
    C2 = measureDistance(my.X , my.Y , 54 , 41 , universe.area)
    if C1 < C2 then
        my.Direction = getdirection(my.X , my.Y , 13.5 , 40 , universe.area)
    elseif C2 < C1 then
        my.Direction = getdirection(my.X , my.Y , 54 , 41 , universe.area)
    end if
    destin = my.Direction
    avoidwallandpeople()
elseif my.point5 == false and my.point4 == true then
    C1 = measureDistance(my.X , my.Y , 25.5 , 40 , universe.area)
    C2 = measureDistance(my.X , my.Y , 59.5 , 41 , universe.area)
    if C1 < C2 then
        my.Direction = getdirection(my.X , my.Y , 25.5 , 40 , universe.area)
    elseif C2 < C1 then
        my.Direction = getdirection(my.X , my.Y , 59.5 , 41 , universe.area)
    end if
    destin = my.Direction
    avoidwallandpeople()
elseif my.point5 == true then
    my.Direction = getdirection(my.X , my.Y , 26 , 36 , universe.area)
    destin = my.Direction
    avoidwallandpeople()
end if
//非常階段//
elseif my.down2 == true then
    if my.point12 == true then
        my.Direction = getdirection(my.X , my.Y , 60 , 36 , universe.area)
        destin = my.Direction
        avoidwallandpeople()
    elseif my.point11 == true then
        my.Direction = getdirection(my.X , my.Y , 60 , 41 , universe.area)
        destin = my.Direction
        avoidwallandpeople()
    elseif my.point1 == true then
        my.Direction = getdirection(my.X , my.Y , 58 , 41 , universe.area)
        destin = my.Direction
        avoidwallandpeople()
    end if
//階段//
elseif my.down1 == true then
    if my.point13 == true then
        my.Direction = getdirection(my.X , my.Y , 22 , 31 , universe.area)
        destin = my.Direction
        avoidwallandpeople()
    elseif my.point12 == true then
        my.Direction = getdirection(my.X , my.Y , 22 , 28 , universe.area)
        destin = my.Direction
        avoidwallandpeople()
    elseif my.point11 == true then
        my.Direction = getdirection(my.X , my.Y , 26 , 27 , universe.area)
        destin = my.Direction
    end if
end if

```

```

        avoidwallandpeople()
    elseif my.point11 == false then
        my.Direction = getdirection(my.X , my.Y ,26, 36 , universe.area)
        destin = my.Direction
        avoidwallandpeople()
    end if
end if
end if

```

//階段を降る//

```

If My.stapN ==True Then
    My.point1=False
    My.point2=False
    My.point3=False
    My.point4=False
    My.point5=False
    my.point10=false
    my.point11=false
    my.point12=false
    my.point13=false
    MoveLayerSpace(-1)
    my.X = 21 + rnd()*2
    My.Y = 33 + rnd()
    My.stapN=False
Elseif My.stapE==True Then
    My.point1=False
    My.point2=False
    My.point3=False
    My.point4=False
    My.point5=False
    my.point10=false
    my.point11=false
    my.point12=false
    my.point13=false
    MoveLayerSpace(-1)
    my.X = 59 + rnd()
    My.Y = 32 + rnd()
    My.stapE=False
End if
}

```

//場所によって歩行スピードを変える関数//

```

function speedchanger() as integer{
dim speed as integer
speed = 0
if my.Layer == 0 then
    If my.X >= 39 and my.X <= 62 and my.Y >= 10 and my.Y <= 21 Then//教室 101
        Forward(0.7)
    elseif my.X >= 20 and my.X <= 41 and my.Y >= 43 and my.Y <= 58 Then//教室 106
        Forward(0.7)
    elseif my.X >= 20 and my.X <= 23 and my.Y >= 32 and my.Y <= 36 Then//階段 N
        Forward(0.4)
    elseif my.X >= 56 and my.X <= 62 and my.Y >= 35 and my.Y <= 39 Then//階段 E

```

```

    Forward(0.4)
    else
    Forward(0.8)
    end if
elseif my.Layer == 1 then
    If my.X >= 20 and my.X <= 62 and my.Y >= 10 and my.Y <= 21 Then//教室 203-201
    forward(0.7)
elseif my.X >= 34 and my.X <= 62 and my.Y >= 43 and my.Y <= 58 Then//教室 205
    forward(0.7)
elseif my.X >= 20 and my.X <= 27 and my.Y >= 30 and my.Y <= 35 Then//階段 N
    Forward(0.4)
elseif my.X >= 56 and my.X <= 62 and my.Y >= 35 and my.Y <= 39 Then//階段 E
    Forward(0.4)
    else
    Forward(0.8)
    end if
elseif my.Layer == 2 then
    If my.X >= 6 and my.X <= 62 and my.Y >= 10 and my.Y <= 21 Then//教室 301-303
    forward(0.7)
elseif my.X >= 34 and my.X <= 62 and my.Y >= 43 and my.Y <= 58 Then//教室 305-306
    forward(0.7)
elseif my.X >= 20 and my.X <= 27 and my.Y >= 30 and my.Y <= 35 Then//階段 N
    Forward(0.4)
elseif my.X >= 56 and my.X <= 62 and my.Y >= 35 and my.Y <= 39 Then//階段 E
    Forward(0.4)
    else
    Forward(0.8)
    end if
elseif my.Layer == 3 then
    If my.X >= 6 and my.X <= 62 and my.Y >= 10 and my.Y <= 21 Then//教室 401-408
    forward(0.7)
elseif my.X >= 27 and my.X <= 62 and my.Y >= 43 and my.Y <= 58 Then//教室 410
    forward(0.7)
elseif my.X >= 20 and my.X <= 27 and my.Y >= 30 and my.Y <= 35 Then//階段 N
    Forward(0.4)
elseif my.X >= 56 and my.X <= 62 and my.Y >= 35 and my.Y <= 39 Then//階段 E
    Forward(0.4)
    else
    Forward(0.8)
    end if
elseif my.Layer == 4 then
    If my.X >= 6 and my.X <= 62 and my.Y >= 10 and my.Y <= 21 Then//教室 501-508
    forward(0.7)
elseif my.X >= 20 and my.X <= 62 and my.Y >= 43 and my.Y <= 58 Then//教室 510-512
    forward(0.7)
elseif my.X >= 20 and my.X <= 27 and my.Y >= 30 and my.Y <= 35 Then//階段 N
    Forward(0.4)
elseif my.X >= 56 and my.X <= 62 and my.Y >= 35 and my.Y <= 39 Then//階段 E
    Forward(0.4)
    else
    Forward(0.8)
    end if
elseif my.Layer == 5 then

```

```

    If my.X >= 6 and my.X <= 62 and my.Y >= 10 and my.Y <= 21 Then//教室 501-508
        forward(0.7)
    elseif my.X >= 20 and my.X <= 62 and my.Y >= 43 and my.Y <= 58 Then//教室 510-512
        forward(0.7)
    elseif my.X >= 20 and my.X <= 27 and my.Y >= 30 and my.Y <= 35 Then//階段 N
        Forward(0.4)
    elseif my.X >= 56 and my.X <= 62 and my.Y >= 35 and my.Y <= 39 Then//階段 E
        Forward(0.4)
    else
        Forward(0.8)
    end if
end if
return(speed)
}

```

//前確認//

```

Function WatchAhead(dist As Double, range As Double, type As Agttype) As Integer{
Dim neighbor As Agtset
Dim target As Integer
Forward(dist)
MakeOneAgtsetAroundOwn(neighbor, range, type, False)
target = CountAgtset(neighbor)
Forward(-dist)
Return(target)
}

```

//壁や人を避ける関数の引用

```

Function AvoidWallAndPeople() As Integer{
Dim deg As Double
Dim destin As Double
deg=0
destin=0
// まず壁の無い方向を向く
If WatchAhead(1, 1, Universe.area.wall) > 0 Then
    deg = (2 * Round(Rnd()) - 1) * 90
    Turn(deg)
    If WatchAhead(1, 1, Universe.area.wall) > 0 Then
        Turn(-deg * 2)
    End if
End if
// 前方に人がいるか
If WatchAhead(1, 1, Universe.area.people) > 0 Then
    deg = (2 * Round(Rnd()) - 1) * 45
    Turn(deg)
    If WatchAhead(1, 1, Universe.area.people) > 0 Then
        Turn(-deg*2)
        If WatchAhead(1, 1, Universe.area.people) > 0 Then
            Turn(-deg/2)
            Forward(0.05)
        Else
            If WatchAhead(1, 1, Universe.area.wall) > 0 Then
                Turn(-deg/2)
                Forward(0.05)
            End if
        End if
    End if
End if
}

```

```

    Else
    speedchanger()
    End if
End if
Else
If WatchAhead(1, 1, Universe.area.wall) > 0 Then
    Turn(-deg/2)
    If WatchAhead(1, 1, Universe.area.people) > 0 Then
        Turn(-deg/2)
        Forward(0.05)
    Else
        If WatchAhead(1, 1, Universe.area.wall) > 0 Then
            Turn(-deg/2)
            Forward(0.05)
        Else
            speedchanger()
        End if
    End if
Else
    speedchanger()
End if
End if
Else
speedchanger()
end if
Return(destin)
}

```

付 1-2 人エージェントプログラム(モデル 2)

※//階段を降る// 以降のプログラムはモデル 1 と同じのため省略

```
Agt_Init{
my.Direction = rnd() * 360
my.point1=false
my.point2 = false
my.point3 = false
my.point4 = false
}
Agt_Step{
dim C1 as integer
dim C2 as integer
dim C3 as integer
dim C4 as integer
dim C5 as integer
dim C6 as integer
dim C7 as integer
dim C8 as integer
dim C9 as integer
dim C10 as integer
dim destin as double

//自分が何階にいるか//
if my.one == false then
  if my.Layer == 0 then
    my.floor = 0
    my.one = true
  elseif my.Layer == 1 then
    my.floor = 1
    my.one = true
  elseif my.Layer == 2 then
    my.floor = 2
    my.one = true
  elseif my.Layer == 3 then
    my.floor = 3
    my.one = true
  elseif my.Layer == 4 then
    my.floor = 4
    my.one = true
  elseif my.Layer == 5 then
    my.floor = 5
    my.one = true
  end if
end if
//一階通常ルール//
if my.Layer == 0 then
  if my.down2 == true then
    if my.point10 == true then
      my.Direction = getdirection(my.X , my.Y , 57 , 41 , universe.area)
      destin = my.Direction
      avoidwallandpeople()
    elseif my.point1 == false then
      my.Direction = getdirection(my.X , my.Y , 57 , 33 , universe.area)
```

```

    destin = my.Direction
    avoidwallandpeople()
end if
elseif my.down2 == false then
    if my.change == 0 then
        if my.point1 == false then
            C1 = MeasureDistance(my.X , my.Y , 21 , 41 , universe.area)
            C2 = MeasureDistance(my.X , my.Y , 39.9 , 41 , universe.area)
            C3 = MeasureDistance(my.X , my.Y , 40.8 , 23 , universe.area)
            C4 = MeasureDistance(my.X , my.Y , 54.5 , 23 , universe.area)
            C5 = MeasureDistance(my.X , my.Y , 21.8 , 38 , universe.area)
            C6 = MeasureDistance(my.X , my.Y , 57 , 33 , universe.area)
            if C1 < C2 and C1 < C3 and C1 < C4 and C1 < C5 and C1 < C6 then
                my.Direction = GetDirection(my.X , my.Y , 21 , 41 , universe.area)
            elseif C2 < C1 and C2 < C3 and C2 < C4 and C2 < C5 and C2 < C6 then
                my.Direction = GetDirection(my.X , my.Y , 39.9 , 41 , universe.area)
            elseif C3 < C1 and C3 < C2 and C3 < C4 and C3 < C5 and C3 < C6 then
                my.Direction = GetDirection(my.X , my.Y , 40.8 , 23 , universe.area)
            elseif C4 < C1 and C4 < C2 and C4 < C3 and C4 < C5 and C4 < C6 then
                my.Direction = GetDirection(my.X , my.Y , 54.5 , 23 , universe.area)
            elseif C5 < C1 and C5 < C2 and C5 < C3 and C5 < C4 and C5 < C6 then
                my.Direction = GetDirection(my.X , my.Y , 21.8 , 38 , universe.area)
            elseif C6 < C1 and 65 < C2 and C6 < C3 and C6 < C4 and C6 < C5 then
                my.Direction = GetDirection(my.X , my.Y , 57 , 33 , universe.area)
            end if
            destin = my.Direction
            avoidwallandpeople()
        elseif my.point2 == false then
            C1 = MeasureDistance(my.X , my.Y , 15 , 40 , universe.area)
            C2 = MeasureDistance(my.X , my.Y , 26 , 38 , universe.area)
            C3 = MeasureDistance(my.X , my.Y , 54 , 26 , universe.area)
            C4 = MeasureDistance(my.X , my.Y , 57 , 35 , universe.area)
            C5 = MeasureDistance(my.X , my.Y , 51 , 41 , universe.area)
            if C1 < C2 and C1 < C3 and C1 < C4 and C1 < C5 then
                my.Direction = GetDirection(my.X , my.Y , 15 , 40 , universe.area)
            elseif C2 < C1 and C2 < C3 and C2 < C4 and C2 < C5 then
                my.Direction = GetDirection(my.X , my.Y , 26 , 38 , universe.area)
            elseif C3 < C1 and C3 < C2 and C3 < C4 and C3 < C5 then
                my.Direction = GetDirection(my.X , my.Y , 54 , 26 , universe.area)
            elseif C4 < C1 and C4 < C2 and C4 < C3 and C4 < C5 then
                my.Direction = GetDirection(my.X , my.Y , 57 , 35 , universe.area)
            elseif C5 < C1 and C5 < C2 and C5 < C3 and C5 < C4 then
                my.Direction = GetDirection(my.X , my.Y , 51 , 41 , universe.area)
            end if
            destin = my.Direction
            avoidwallandpeople()
        elseif my.point3 == false then
            C1 = measureDistance(my.X , my.Y , 12 , 24 , universe.area)
            C2 = measureDistance(my.X , my.Y , 57 , 41 , universe.area)
            C3 = measureDistance(my.X , my.Y , 62 , 29 , universe.area)
            C4 = measureDistance(my.X , my.Y , 25 , 23 , universe.area)
            if C1 < C2 and C1 < C3 and C1 < C4 then
                my.Direction = getdirection(my.X , my.Y , 12 , 24 , universe.area)
            end if
        end if
    end if
end if

```

```

elseif C2 < C1 and C2 < C3 and C2 < C4 then
my.Direction = getdirection(my.X , my.Y , 57 , 41 , universe.area)
elseif C3 < C1 and C3 < C2 and C3 < C4 then
my.Direction = getdirection(my.X , my.Y , 62 , 29 , universe.area)
elseif C4 < C1 and C4 < C2 and C4 < C3 then
my.Direction = getdirection(my.X , my.Y , 25 , 23 , universe.area)
end if
destin = my.Direction
avoidwallandpeople()
elseif my.point4 == false then
my.Direction = getdirection(my.X , my.Y , 12 , 11 , universe.area)
destin = my.Direction
avoidwallandpeople()
elseif my.point5 == false then
my.Direction = getdirection(my.X , my.Y , 12 , 3 , universe.area)
destin = my.Direction
avoidwallandpeople()
end if
//正面玄関避難//
elseif my.change == 1 then
if my.U1 == false then
if my.point2 == false and my.point3 == false then
C1 = MeasureDistance(my.X , my.Y , 15 , 40 , universe.area)
C2 = MeasureDistance(my.X , my.Y , 26 , 38 , universe.area)
C3 = MeasureDistance(my.X , my.Y , 25 , 23 , universe.area)
if C1 < C2 and C1 < C3 then
my.Direction = GetDirection(my.X , my.Y , 15 , 40 , universe.area)
elseif C2 < C1 and C2 < C3 then
my.Direction = GetDirection(my.X , my.Y , 26 , 38 , universe.area)
elseif C3 < C1 and C3 < C2 then
my.Direction = GetDirection(my.X , my.Y , 25 , 23 , universe.area)
end if
destin = my.Direction
avoidwallandpeople()
elseif my.point3 == false then
C1 = measureDistance(my.X , my.Y , 12 , 24 , universe.area)
C2 = measureDistance(my.X , my.Y , 25 , 23 , universe.area)
if C1 < C2 then
my.Direction = getdirection(my.X , my.Y , 12 , 24 , universe.area)
elseif C2 < C1 then
my.Direction = getdirection(my.X , my.Y , 25 , 23 , universe.area)
end if
destin = my.Direction
avoidwallandpeople()
elseif my.point4 == false then
my.Direction = getdirection(my.X , my.Y , 12 , 11 , universe.area)
destin = my.Direction
avoidwallandpeople()
elseif my.point5 == false then
my.Direction = getdirection(my.X , my.Y , 12 , 3 , universe.area)
destin = my.Direction
avoidwallandpeople()
end if

```

```

elseif my.U1 == true then
  if my.point2 == false then
    C1 = MeasureDistance(my.X , my.Y , 15 , 40 , universe.area)
    C2 = MeasureDistance(my.X , my.Y , 26 , 38 , universe.area)
    if C1 < C2 then
      my.Direction = GetDirection(my.X , my.Y , 15 , 40 , universe.area)
    elseif C2 < C1 then
      my.Direction = GetDirection(my.X , my.Y , 26 , 38 , universe.area)
    end if
    destin = my.Direction
    avoidwallandpeople()
    elseif my.point3 == false then
      C1 = measureDistance(my.X , my.Y , 12 , 24 , universe.area)
      C2 = measureDistance(my.X , my.Y , 25 , 23 , universe.area)
      if C1 < C2 then
        my.Direction = getdirection(my.X , my.Y , 12 , 24 , universe.area)
      elseif C2 < C1 then
        my.Direction = getdirection(my.X , my.Y , 25 , 23 , universe.area)
      end if
      destin = my.Direction
      avoidwallandpeople()
    elseif my.point4 == false then
      my.Direction = getdirection(my.X , my.Y , 12 , 11 , universe.area)
      destin = my.Direction
      avoidwallandpeople()
    elseif my.point5 == false then
      my.Direction = getdirection(my.X , my.Y , 12 , 3 , universe.area)
      destin = my.Direction
      avoidwallandpeople()
    end if
  end if
//裏口避難//
elseif my.change == 2 then
  if my.U1 == false then
    if my.point2 == false and my.point13 == false then
      C1 = measureDistance(my.X , my.Y , 54 , 26 , universe.area)
      C2 = measureDistance(my.X , my.Y , 51 , 41 , universe.area)
      C3 = measureDistance(my.X , my.Y , 26 , 39 , universe.area)
      if C1 < C2 and C1 < C3 then
        my.Direction = getdirection(my.X , my.Y , 54 , 26 , universe.area)
      elseif C2 < C1 and C2 < C3 then
        my.Direction = getdirection(my.X , my.Y , 51 , 41 , universe.area)
      elseif C3 < C1 and C3 < C2 then
        my.Direction = getdirection(my.X , my.Y , 26 , 39 , universe.area)
      end if
      destin = my.Direction
      avoidwallandpeople()
    elseif my.point11 == false and my.point12 == false then
      C1 = measureDistance(my.X , my.Y , 54 , 29 , universe.area)
      C2 = measureDistance(my.X , my.Y , 53 , 41 , universe.area)
      if C1 < C2 then
        my.Direction = getdirection(my.X , my.Y , 54 , 29 , universe.area)
      elseif C2 < C1 then

```

```

    my.Direction = getdirection(my.X , my.Y , 53 , 41 , universe.area)
  end if
  destin = my.Direction
  avoidwallandpeople()
  elseif my.point11 == false then
  my.Direction = getdirection(my.X , my.Y , 54 , 29 , universe.area)
  destin = my.Direction
  avoidwallandpeople()
  elseif my.point3 == false then
  my.Direction = getdirection(my.X , my.Y , 63 , 27 , universe.area)
  destin = my.Direction
  avoidwallandpeople()
  end if
elseif my.U1 == true then
  if my.point2 == false then
  my.Direction = getdirection(my.X , my.Y , 51 , 41 , universe.area)
  destin = my.Direction
  avoidwallandpeople()
  elseif my.point11== false then
  my.Direction = getdirection(my.X , my.Y , 54 , 29 , universe.area)
  destin = my.Direction
  avoidwallandpeople()
  elseif my.point3 == false then
  my.Direction = getdirection(my.X , my.Y , 63 , 27 , universe.area)
  destin = my.Direction
  avoidwallandpeople()
  end if
end if
//非常口避難//
elseif my.change == 3 then
  if my.U1 == false then
    if my.point2 == false and my.point13 == false then
      C1 = measureDistance(my.X , my.Y , 54 , 26 , universe.area)
      C2 = measureDistance(my.X , my.Y , 51 , 41 , universe.area)
      C3 = measureDistance(my.X , my.Y , 26 , 39 , universe.area)
      if C1 < C2 and C1 < C3 then
        my.Direction = getdirection(my.X , my.Y , 54 , 26 , universe.area)
      elseif C2 < C1 and C2 < C3 then
        my.Direction = getdirection(my.X , my.Y , 51 , 41 , universe.area)
      elseif C3 < C1 and C3 < C2 then
        my.Direction = getdirection(my.X , my.Y , 26 , 39 , universe.area)
      end if
      destin = my.Direction
      avoidwallandpeople()
    elseif my.point12 == false then
      my.Direction = getdirection(my.X , my.Y , 53 , 41 , universe.area)
      destin = my.Direction
      avoidwallandpeople()
    elseif my.point3 == false then
      my.Direction = getdirection(my.X , my.Y , 56 , 41 , universe.area)
      destin = my.Direction
      avoidwallandpeople()
    end if
  end if

```

```

elseif my.U1 == true then
  if my.point2 == false then
    my.Direction = getdirection(my.X , my.Y , 51 , 41 , universe.area)
    destin = my.Direction
    avoidwallandpeople()
    elseif my.point3 == false then
      my.Direction = getdirection(my.X , my.Y , 56 , 41 , universe.area)
      destin = my.Direction
      avoidwallandpeople()
    end if
  end if
end if
end if
// 二階通常ルール//
elseif my.Layer == 1 then
  if my.down1 == false and my.down2 == false then
    if my.change == 0 then
      if my.point1 == false then
        C1 = MeasureDistance(my.X , my.Y , 61 , 23 , universe.area)
        C2 = MeasureDistance(my.X , my.Y , 50 , 23 , universe.area)
        C3 = MeasureDistance(my.X , my.Y , 47 , 23 , universe.area)
        C4 = MeasureDistance(my.X , my.Y , 34 , 24 , universe.area)
        C5 = MeasureDistance(my.X , my.Y , 21 , 23 , universe.area)
        if C1 < C2 and C1 < C3 and C1 < C4 and C1 < C5 then
          my.Direction = GetDirection(my.X , my.Y , 61 , 23 , universe.area)
        elseif C2 < C1 and C2 < C3 and C2 < C4 and C2 < C5 then
          my.Direction = GetDirection(my.X , my.Y , 50 , 23 , universe.area)
        elseif C3 < C1 and C3 < C2 and C3 < C4 and C3 < C5 then
          my.Direction = GetDirection(my.X , my.Y , 47 , 23 , universe.area)
        elseif C4 < C1 and C4 < C2 and C4 < C3 and C4 < C5 then
          my.Direction = GetDirection(my.X , my.Y , 34 , 24 , universe.area)
        elseif C5 < C1 and C5 < C2 and C5 < C3 and C5 < C4 then
          my.Direction = GetDirection(my.X , my.Y , 21 , 23 , universe.area)
        end if
        destin = my.Direction
        avoidwallandpeople()
      elseif my.point2 == false then
        C1 = MeasureDistance(my.X , my.Y , 14 , 26 , universe.area)
        C2 = MeasureDistance(my.X , my.Y , 53.5 , 26 , universe.area)
        C3 = MeasureDistance(my.X , my.Y , 53.5 , 41 , universe.area)
        C4 = MeasureDistance(my.X , my.Y , 41.5 , 41 , universe.area)
        if C1 < C2 and C1 < C3 and C1 < C4 then
          my.Direction = GetDirection(my.X , my.Y , 14 , 26 , universe.area)
        elseif C2 < C1 and C2 < C3 and C2 < C4 then
          my.Direction = GetDirection(my.X , my.Y , 53.5 , 26 , universe.area)
        elseif C3 < C1 and C3 < C2 and C3 < C4 then
          my.Direction = GetDirection(my.X , my.Y , 53.5 , 41 , universe.area)
        elseif C4 < C1 and C4 < C2 and C4 < C3 then
          my.Direction = GetDirection(my.X , my.Y , 41.5 , 41 , universe.area)
        end if
        destin = my.Direction
        avoidwallandpeople()
      elseif my.point3 == false then

```

```

C1 = measureDistance(my.X , my.Y , 15 , 40 , universe.area)
C2 = measureDistance(my.X , my.Y , 54 , 41 , universe.area)
  if C1 < C2  then
    my.Direction = getdirection(my.X , my.Y , 15 , 40 , universe.area)
  elseif C2 < C1  then
    my.Direction = getdirection(my.X , my.Y , 54 , 41 , universe.area)
  end if
  destin = my.Direction
  avoidwallandpeople()
elseif my.point4 == false  then
C1 = measureDistance(my.X , my.Y , 25.5 , 40 , universe.area)
C2 = measureDistance(my.X , my.Y , 59.5 , 41 , universe.area)
  if C1 < C2  then
    my.Direction = getdirection(my.X , my.Y , 25.5 , 40 , universe.area)
  elseif C2 < C1  then
    my.Direction = getdirection(my.X , my.Y , 59.5 , 41 , universe.area)
  end if
  destin = my.Direction
  avoidwallandpeople()
elseif my.point4 == true  then
my.Direction = getdirection(my.X , my.Y , 27 , 28 , universe.area)
destin = my.Direction
avoidwallandpeople()
end if
//二階非常階段混雑時ルール//
elseif my.change == 1  then
  if my.U2 == false  then
    if my.point2 == false  then
      my.Direction = GetDirection(my.X , my.Y , 14 , 26 , universe.area)
      destin = my.Direction
      avoidwallandpeople()
    elseif my.point3 == false  then
      my.Direction = getdirection(my.X , my.Y , 15 , 40 , universe.area)
      destin = my.Direction
      avoidwallandpeople()
    elseif my.point4 == false  then
      my.Direction = getdirection(my.X , my.Y , 25.5 , 40 , universe.area)
      destin = my.Direction
      avoidwallandpeople()
    elseif my.point4 == true  then
      my.Direction = getdirection(my.X , my.Y , 26 , 28 , universe.area)
      destin = my.Direction
      avoidwallandpeople()
    end if
  elseif my.U2 == true  then
    if my.point4 == false  then
      my.Direction = getdirection(my.X , my.Y , 25.5 , 40 , universe.area)
      destin = my.Direction
      avoidwallandpeople()
    elseif my.point4 == true  then
      my.Direction = getdirection(my.X , my.Y , 26 , 28 , universe.area)
      destin = my.Direction
      avoidwallandpeople()
    end if
  end if
end if

```

```

        end if
    end if
//二階中央階段混雑時ルール//
elseif my.change == 2 then
    if my.U2 == false then
        if my.point2 == false then
            my.Direction = GetDirection(my.X , my.Y , 53.5 , 26 , universe.area)
            destin = my.Direction
            avoidwallandpeople()
            elseif my.point3 == false then
                my.Direction = getdirection(my.X , my.Y , 54 , 41 , universe.area)
                destin = my.Direction
                avoidwallandpeople()
            elseif my.point4 == false then
                my.Direction = getdirection(my.X , my.Y , 59.5 , 41 , universe.area)
                destin = my.Direction
                avoidwallandpeople()
            end if
        elseif my.U2 == true then
            if my.point3 == false then
                my.Direction = getdirection(my.X , my.Y , 54 , 41 , universe.area)
                destin = my.Direction
                avoidwallandpeople()
            elseif my.point4 == false then
                my.Direction = getdirection(my.X , my.Y , 59.5 , 41 , universe.area)
                destin = my.Direction
                avoidwallandpeople()
            end if
        end if
    end if
//非常階段//
elseif my.down2 == true then
    if my.point12 == true then
        my.Direction = getdirection(my.X , my.Y , 60 , 36 , universe.area)
        destin = my.Direction
        avoidwallandpeople()
    elseif my.point11 == true then
        my.Direction = getdirection(my.X , my.Y , 60 , 41 , universe.area)
        destin = my.Direction
        avoidwallandpeople()
    elseif my.point10 == true then
        my.Direction = getdirection(my.X , my.Y , 56.5 , 42 , universe.area)
        destin = my.Direction
        avoidwallandpeople()
    elseif my.point1 == true then
        my.Direction = getdirection(my.X , my.Y , 58 , 41 , universe.area)
        destin = my.Direction
        avoidwallandpeople()
    elseif my.point1 == false then
        my.Direction = getdirection(my.X , my.Y , 56 , 33 , universe.area)
        destin = my.Direction
        avoidwallandpeople()
    end if
end if

```

```

//階段//
elseif my.down1 == true then
if my.point13 == true then
my.Direction = getdirection(my.X , my.Y , 22 , 31 , universe.area)
destin = my.Direction
avoidwallandpeople()
elseif my.point12 == true then
my.Direction = getdirection(my.X , my.Y , 22 , 28 , universe.area)
destin = my.Direction
avoidwallandpeople()
elseif my.point11 == true then
my.Direction = getdirection(my.X , my.Y , 26 , 27 , universe.area)
destin = my.Direction
avoidwallandpeople()
elseif my.point10 == true then
my.Direction = getdirection(my.X , my.Y , 27 , 36 , universe.area)
destin = my.Direction
avoidwallandpeople()
elseif my.point1 == false and my.point10 == false then
my.Direction = getdirection(my.X , my.Y , 21 , 36 , universe.area)
destin = my.Direction
avoidwallandpeople()
elseif my.point11 == false then
my.Direction = getdirection(my.X , my.Y , 26 , 36 , universe.area)
destin = my.Direction
avoidwallandpeople()
end if
end if

// 三階通常ルール//
elseif my.Layer == 2 then
if my.down1 == false and my.down2 == false then
if my.change == 0 then
if my.point1 == false then
C1 = MeasureDistance(my.X , my.Y , 15 , 23 , universe.area)
C2 = MeasureDistance(my.X , my.Y , 25 , 23 , universe.area)
C3 = MeasureDistance(my.X , my.Y , 35.5 , 23 , universe.area)
if C1 < C2 and C1 < C3 then
my.Direction = GetDirection(my.X , my.Y , 15 , 23 , universe.area)
elseif C2 < C3 and C2 < C3 then
my.Direction = GetDirection(my.X , my.Y , 25 , 23 , universe.area)
elseif C3 < C1 and C3 < C2 then
my.Direction = GetDirection(my.X , my.Y , 35.5 , 23 , universe.area)
end if
destin = my.Direction
avoidwallandpeople()
elseif my.point2 == false and my.point3 == false then
C1 = MeasureDistance(my.X , my.Y , 13 , 26 , universe.area)
C2 = MeasureDistance(my.X , my.Y , 50 , 24 , universe.area)
C3 = MeasureDistance(my.X , my.Y , 36 , 41 , universe.area)
C4 = MeasureDistance(my.X , my.Y , 53.5 , 41 , universe.area)
if C1 < C2 and C1 < C3 and C1 < C4 then
my.Direction = GetDirection(my.X , my.Y , 13 , 26 , universe.area)

```

```

elseif C2 < C1 and C2 < C3 and C2 < C4 then
my.Direction = GetDirection(my.X , my.Y , 50 , 24 , universe.area)
elseif C3 < C1 and C3 < C2 and C3 < C4 then
my.Direction = GetDirection(my.X , my.Y , 36 , 41 , universe.area)
elseif C4 < C1 and C4 < C2 and C4 < C3 then
my.Direction = GetDirection(my.X , my.Y , 53.5 , 41 , universe.area)
end if
destin = my.Direction
avoidwallandpeople()
elseif my.point3 == false then
C1 = measureDistance(my.X , my.Y , 13.5 , 40 , universe.area)
C2 = measureDistance(my.X , my.Y , 53.5 , 27 , universe.area)
if C1 < C2 then
my.Direction = getdirection(my.X , my.Y , 13.5 , 40 , universe.area)
elseif C2 < C1 then
my.Direction = getdirection(my.X , my.Y , 53.5 , 27 , universe.area)
end if
destin = my.Direction
avoidwallandpeople()
elseif my.point4 == false then
C1 = measureDistance(my.X , my.Y , 25.5 , 40 , universe.area)
C2 = measureDistance(my.X , my.Y , 53.5 , 41 , universe.area)
if C1 < C2 then
my.Direction = getdirection(my.X , my.Y , 25.5 , 40 , universe.area)
elseif C2 < C1 then
my.Direction = getdirection(my.X , my.Y , 53.5 , 41 , universe.area)
end if
destin = my.Direction
avoidwallandpeople()
elseif my.point4 == true then
C1 = measureDistance(my.X , my.Y , 26 , 28 , universe.area)
C2 = measureDistance(my.X , my.Y , 59.5 , 41 , universe.area)
if C1 < C2 then
my.Direction = getdirection(my.X , my.Y , 26 , 28 , universe.area)
elseif C2 < C1 then
my.Direction = getdirection(my.X , my.Y , 59.5 , 41 , universe.area)
end if
destin = my.Direction
avoidwallandpeople()
end if
//三階非常階段混雑時ルール//
elseif my.change == 1 then
if my.U3 == false then
if my.point2 == false then
my.Direction = GetDirection(my.X , my.Y , 14 , 26 , universe.area)
destin = my.Direction
avoidwallandpeople()
elseif my.point3 == false then
my.Direction = getdirection(my.X , my.Y , 13.5 , 40 , universe.area)
destin = my.Direction
avoidwallandpeople()
elseif my.point4 == false then
my.Direction = getdirection(my.X , my.Y , 25.5 , 40 , universe.area)

```

```

    destin = my.Direction
    avoidwallandpeople()
    elseif my.point4 == true then
    my.Direction = getdirection(my.X , my.Y , 26 , 28 , universe.area)
    destin = my.Direction
    avoidwallandpeople()
    end if
elseif my.U3 == true then
    if my.point3 == false then
    my.Direction = GetDirection(my.X , my.Y , 36 , 41 , universe.area)
    destin = my.Direction
    avoidwallandpeople()
    elseif my.point4 == false then
    my.Direction = getdirection(my.X , my.Y , 25.5 , 40 , universe.area)
    destin = my.Direction
    avoidwallandpeople()
    elseif my.point4 == true then
    my.Direction = getdirection(my.X , my.Y , 26 , 28 , universe.area)
    destin = my.Direction
    avoidwallandpeople()
    end if
end if

//三階中央階段混雑時ルール//
elseif my.change == 2 then
    if my.U3 == false then
        if my.point2 == false then
            my.Direction = GetDirection(my.X , my.Y , 50 , 24 , universe.area)
            destin = my.Direction
            avoidwallandpeople()
        elseif my.point3 == false then
            my.Direction = getdirection(my.X , my.Y , 53.5 , 27 , universe.area)
            destin = my.Direction
            avoidwallandpeople()
        elseif my.point4 == false then
            my.Direction = getdirection(my.X , my.Y , 53.5 , 41 , universe.area)
            destin = my.Direction
            avoidwallandpeople()
        elseif my.point4 == true then
            my.Direction = getdirection(my.X , my.Y , 59.5 , 41 , universe.area)
            destin = my.Direction
            avoidwallandpeople()
        end if
    elseif my.U3 == true then
        if my.point3 == false then
            my.Direction = GetDirection(my.X , my.Y , 53.5 , 41 , universe.area)
            destin = my.Direction
            avoidwallandpeople()
        elseif my.point4 == false then
            my.Direction = getdirection(my.X , my.Y , 54 , 41 , universe.area)
            destin = my.Direction
            avoidwallandpeople()
        elseif my.point4 == true then

```

```

        my.Direction = getdirection(my.X , my.Y , 59.5 , 41 , universe.area)
        destin = my.Direction
        avoidwallandpeople()
    end if
end if
end if
//非常階段//
elseif my.down2 == true then
    if my.point12 == true then
        my.Direction = getdirection(my.X , my.Y , 60 , 36 , universe.area)
        destin = my.Direction
        avoidwallandpeople()
    elseif my.point11 == true then
        my.Direction = getdirection(my.X , my.Y , 60 , 41 , universe.area)
        destin = my.Direction
        avoidwallandpeople()
    elseif my.point10 == true then
        my.Direction = getdirection(my.X , my.Y , 56.5 , 42 , universe.area)
        destin = my.Direction
        avoidwallandpeople()
    elseif my.point1 == true then
        my.Direction = getdirection(my.X , my.Y , 58 , 41 , universe.area)
        destin = my.Direction
        avoidwallandpeople()
    elseif my.point1 == false then
        my.Direction = getdirection(my.X , my.Y , 56 , 33 , universe.area)
        destin = my.Direction
        avoidwallandpeople()
    end if
//階段//
elseif my.down1 == true then
if my.point13 == true then
    my.Direction = getdirection(my.X , my.Y , 22 , 31 , universe.area)
    destin = my.Direction
    avoidwallandpeople()
elseif my.point12 == true then
    my.Direction = getdirection(my.X , my.Y , 22 , 28 , universe.area)
    destin = my.Direction
    avoidwallandpeople()
elseif my.point11 == true then
    my.Direction = getdirection(my.X , my.Y , 26 , 27 , universe.area)
    destin = my.Direction
    avoidwallandpeople()
elseif my.point10 == true then
    my.Direction = getdirection(my.X , my.Y , 27 , 36 , universe.area)
    destin = my.Direction
    avoidwallandpeople()
elseif my.point1 == false and my.point10 == false then
    my.Direction = getdirection(my.X , my.Y , 21 , 36 , universe.area)
    destin = my.Direction
    avoidwallandpeople()
elseif my.point11 == false then
    my.Direction = getdirection(my.X , my.Y , 26 , 36 , universe.area)

```

```

    destin = my.Direction
    avoidwallandpeople()
end if
end if

// 四階通常ルール//
elseif my.Layer == 3 then
if my.down1 == false and my.down2 == false then
if my.change == 0 then
if my.point1 == false then
C1 = MeasureDistance(my.X , my.Y , 6.5 , 23 , universe.area)
C2 = MeasureDistance(my.X , my.Y , 21 , 23 , universe.area)
C3 = MeasureDistance(my.X , my.Y , 35 , 23 , universe.area)
C4 = MeasureDistance(my.X , my.Y , 49.5 , 23 , universe.area)
if C1 < C2 and C1 < C3 and C1 < C4 then
my.Direction = GetDirection(my.X , my.Y , 6.5 , 23 , universe.area)
elseif C2 < C3 and C2 < C3 and C2 < C4 then
my.Direction = GetDirection(my.X , my.Y , 21 , 23 , universe.area)
elseif C3 < C1 and C3 < C2 and C3 < C4 then
my.Direction = GetDirection(my.X , my.Y , 35 , 23 , universe.area)
elseif C4 < C1 and C4 < C2 and C4 < C3 then
my.Direction = GetDirection(my.X , my.Y , 49.5 , 23 , universe.area)
end if
destin = my.Direction
avoidwallandpeople()
elseif my.point2 == false then
C1 = MeasureDistance(my.X , my.Y , 7 , 24 , universe.area)
C2 = MeasureDistance(my.X , my.Y , 13.5 , 23 , universe.area)
C3 = MeasureDistance(my.X , my.Y , 20 , 24 , universe.area)
C4 = MeasureDistance(my.X , my.Y , 28 , 23 , universe.area)
C5 = MeasureDistance(my.X , my.Y , 35.5 , 24 , universe.area)
C6 = MeasureDistance(my.X , my.Y , 42.5 , 23 , universe.area)
C7 = MeasureDistance(my.X , my.Y , 49.5 , 24 , universe.area)
C8 = MeasureDistance(my.X , my.Y , 57 , 23 , universe.area)
C9 = MeasureDistance(my.X , my.Y , 28 , 41 , universe.area)
C10 = MeasureDistance(my.X , my.Y , 54 , 41 , universe.area)
if C1 < C2 and C1 < C3 and C1 < C4 and C1 < C5 and C1 < C6
and C1 < C7 and C1 < C8 and C1 < C9 and C1 < C10 then
my.Direction = GetDirection(my.X , my.Y , 7 , 24 , universe.area)
elseif C2 < C1 and C2 < C3 and C2 < C4 and C2 < C5 and C2 < C6
and C2 < C7 and C2 < C8 and C2 < C9 and C2 < C10 then
my.Direction = GetDirection(my.X , my.Y , 13.5 , 23 , universe.area)
elseif C3 < C1 and C3 < C2 and C3 < C4 and C3 < C5 and C3 < C6
and C3 < C7 and C3 < C8 and C3 < C9 and C3 < C10 then
my.Direction = GetDirection(my.X , my.Y , 20 , 24 , universe.area)
elseif C4 < C1 and C4 < C2 and C4 < C3 and C4 < C5 and C4 < C6
and C4 < C7 and C4 < C8 and C4 < C9 and C4 < C10 then
my.Direction = GetDirection(my.X , my.Y , 28 , 23 , universe.area)
elseif C5 < C1 and C5 < C2 and C5 < C3 and C5 < C4 and C5 < C6
and C5 < C7 and C5 < C8 and C5 < C9 and C5 < C10 then
my.Direction = GetDirection(my.X , my.Y , 35.5 , 24 , universe.area)
elseif C6 < C1 and C6 < C2 and C6 < C3 and C6 < C4 and C6 < C5
and C6 < C7 and C6 < C8 and C6 < C9 and C6 < C10 then

```

```

my.Direction = GetDirection(my.X , my.Y , 42.5 , 23 , universe.area)
elseif C7 < C1 and C7 < C2 and C7 < C3 and C7 < C4 and C7 < C5
and C7 < C6 and C7 < C8 and C7 < C9 and C7 < C10 then
my.Direction = GetDirection(my.X , my.Y , 49.5 , 24 , universe.area)
elseif C8 < C1 and C8 < C2 and C8 < C3 and C8 < C4 and C8 < C5
and C8 < C6 and C8 < C7 and C8 < C9 and C8 < C10 then
my.Direction = GetDirection(my.X , my.Y , 57 , 23 , universe.area)
elseif C9 < C1 and C9 < C2 and C9 < C3 and C9 < C4 and C9 < C5
and C9 < C6 and C9 < C7 and C9 < C8 and C9 < C10 then
my.Direction = GetDirection(my.X , my.Y , 28 , 41 , universe.area)
elseif C10 < C1 and C10 < C2 and C10 < C3 and C10 < C4 and C10 < C5
and C10 < C6 and C10 < C7 and C10 < C8 and C10 < C9 then
my.Direction = GetDirection(my.X , my.Y , 54 , 41 , universe.area)
end if
destin = my.Direction
avoidwallandpeople()
elseif my.point3 == false and my.point4 == false and my.point5 == false then
C1 = measureDistance(my.X , my.Y , 13 , 26 , universe.area)
C2 = measureDistance(my.X , my.Y , 53.5 , 26 , universe.area)
C3 = measureDistance(my.X , my.Y , 25.5 , 40 , universe.area)
C4 = measureDistance(my.X , my.Y , 54 , 41 , universe.area)
if C1 < C2 and C1 < C3 and C1 < C4 then
my.Direction = getdirection(my.X , my.Y , 13 , 26 , universe.area)
elseif C2 < C1 and C2 < C3 and C2 < C4 then
my.Direction = getdirection(my.X , my.Y , 53.5 , 26 , universe.area)
elseif C3 < C1 and C3 < C2 and C3 < C4 then
my.Direction = getdirection(my.X , my.Y , 25.5 , 40 , universe.area)
elseif C4 < C1 and C4 < C2 and C4 < C3 then
my.Direction = getdirection(my.X , my.Y , 54 , 41 , universe.area)
end if
destin = my.Direction
avoidwallandpeople()
elseif my.point4 == false and my.point3 == true then
C1 = measureDistance(my.X , my.Y , 13.5 , 40 , universe.area)
C2 = measureDistance(my.X , my.Y , 54 , 41 , universe.area)
if C1 < C2 then
my.Direction = getdirection(my.X , my.Y , 13.5 , 40 , universe.area)
elseif C2 < C1 then
my.Direction = getdirection(my.X , my.Y , 54 , 41 , universe.area)
end if
destin = my.Direction
avoidwallandpeople()
elseif my.point5 == false and my.point4 == true then
C1 = measureDistance(my.X , my.Y , 25.5 , 40 , universe.area)
C2 = measureDistance(my.X , my.Y , 59.5 , 41 , universe.area)
if C1 < C2 then
my.Direction = getdirection(my.X , my.Y , 25.5 , 40 , universe.area)
elseif C2 < C1 then
my.Direction = getdirection(my.X , my.Y , 59.5 , 41 , universe.area)
end if
destin = my.Direction
avoidwallandpeople()
elseif my.point5 == true then

```

```

my.Direction = getdirection(my.X , my.Y , 26 , 36 , universe.area)
destin = my.Direction
avoidwallandpeople()
end if
//四階非常階段混雑時ルール//
elseif my.change == 1 then
if my.U4 == false then
if my.point3 == false then
my.Direction = GetDirection(my.X , my.Y , 14 , 26 , universe.area)
destin = my.Direction
avoidwallandpeople()
elseif my.point4 == false then
C1 = measureDistance(my.X , my.Y , 13.5 , 40 , universe.area)
C2 = measureDistance(my.X , my.Y , 54 , 41 , universe.area)
if C1 < C2 then
my.Direction = getdirection(my.X , my.Y , 13.5 , 40 , universe.area)
elseif C2 < C1 then
my.Direction = getdirection(my.X , my.Y , 54 , 41 , universe.area)
end if
destin = my.Direction
avoidwallandpeople()
elseif my.point5 == false then
my.Direction = getdirection(my.X , my.Y , 25.5 , 40 , universe.area)
destin = my.Direction
avoidwallandpeople()
elseif my.point5 == true then
my.Direction = getdirection(my.X , my.Y , 26 , 28 , universe.area)
destin = my.Direction
avoidwallandpeople()
end if
elseif my.U4 == true then
if my.point2 == false then
my.Direction = GetDirection(my.X , my.Y , 28.5 , 41 , universe.area)
destin = my.Direction
avoidwallandpeople()
elseif my.point5 == false then
my.Direction = getdirection(my.X , my.Y , 25.5 , 40 , universe.area)
destin = my.Direction
avoidwallandpeople()
elseif my.point5 == true then
my.Direction = getdirection(my.X , my.Y , 26 , 28 , universe.area)
destin = my.Direction
avoidwallandpeople()
end if
end if
//四階中央階段混雑時ルール//
elseif my.change == 2 then
if my.U4 == false then
if my.point3 == false then
my.Direction = getdirection(my.X , my.Y , 53.5 , 27 , universe.area)
destin = my.Direction
avoidwallandpeople()
elseif my.point4 == false then

```

```

C1 = measureDistance(my.X , my.Y , 13.5 , 40 , universe.area)
C2 = measureDistance(my.X , my.Y ,54, 41 , universe.area)
  if C1 < C2  then
my.Direction = getdirection(my.X , my.Y , 13.5 , 40 , universe.area)
  elseif C2 < C1  then
my.Direction = getdirection(my.X , my.Y , 54 , 41 , universe.area)
  end if
  destin = my.Direction
  avoidwallandpeople()
  elseif my.point4 == true then
my.Direction = getdirection(my.X , my.Y , 59.5 , 41 , universe.area)
  destin = my.Direction
  avoidwallandpeople()
  end if
elseif my.U4 == true then
  if my.point2 == false  then
my.Direction = GetDirection(my.X , my.Y , 53.5 , 41 , universe.area)
  destin = my.Direction
  avoidwallandpeople()
  elseif my.point4 == false  then
my.Direction = getdirection(my.X , my.Y , 54 , 41 , universe.area)
  destin = my.Direction
  avoidwallandpeople()
  elseif my.point4 == true then
my.Direction = getdirection(my.X , my.Y , 59.5 , 41 , universe.area)
  destin = my.Direction
  avoidwallandpeople()
  end if
end if
end if
//非常階段//
elseif my.down2 == true then
  if my.point12 == true then
my.Direction = getdirection(my.X , my.Y , 60 , 36 , universe.area)
  destin = my.Direction
  avoidwallandpeople()
  elseif my.point11 == true then
my.Direction = getdirection(my.X , my.Y , 60 , 41 , universe.area)
  destin = my.Direction
  avoidwallandpeople()
  elseif my.point10 == true then
my.Direction = getdirection(my.X , my.Y , 56.5 , 42 , universe.area)
  destin = my.Direction
  avoidwallandpeople()
  elseif my.point1 == true then
my.Direction = getdirection(my.X , my.Y , 58 , 41 , universe.area)
  destin = my.Direction
  avoidwallandpeople()
  elseif my.point1 == false then
my.Direction = getdirection(my.X , my.Y , 56 , 33 , universe.area)
  destin = my.Direction
  avoidwallandpeople()
end if

```

```

//階段//
elseif my.down1 == true then
if my.point13 == true then
my.Direction = getdirection(my.X , my.Y , 22 , 31 , universe.area)
destin = my.Direction
avoidwallandpeople()
elseif my.point12 == true then
my.Direction = getdirection(my.X , my.Y , 22 , 28 , universe.area)
destin = my.Direction
avoidwallandpeople()
elseif my.point11 == true then
my.Direction = getdirection(my.X , my.Y , 26 , 27 , universe.area)
destin = my.Direction
avoidwallandpeople()
elseif my.point10 == true then
my.Direction = getdirection(my.X , my.Y , 27 , 36 , universe.area)
destin = my.Direction
avoidwallandpeople()
elseif my.point1 == false and my.point10 == false then
my.Direction = getdirection(my.X , my.Y , 21 , 36 , universe.area)
destin = my.Direction
avoidwallandpeople()
elseif my.point11 == false then
my.Direction = getdirection(my.X , my.Y , 26 , 36 , universe.area)
destin = my.Direction
avoidwallandpeople()
end if
end if
// 五階通常ルール//
elseif my.Layer == 4 then
if my.down1 == false and my.down2 == false then
if my.change == 0 then
if my.point1 == false then
C1 = MeasureDistance(my.X , my.Y , 6.5 , 23 , universe.area)
C2 = MeasureDistance(my.X , my.Y , 21 , 23 , universe.area)
C3 = MeasureDistance(my.X , my.Y , 49.5 , 23 , universe.area)
C4 = MeasureDistance(my.X , my.Y , 35 , 41 , universe.area)
if C1 < C2 and C1 < C3 and C1 < C4 then
my.Direction = GetDirection(my.X , my.Y , 6.5 , 23 , universe.area)
elseif
C2 < C3 and C2 < C3 and C2 < C4 then
my.Direction = GetDirection(my.X , my.Y , 21 , 23 , universe.area)
elseif
C3 < C1 and C3 < C2 and C3 < C4 then
my.Direction = GetDirection(my.X , my.Y , 49.5 , 23 , universe.area)
elseif
C4 < C1 and C4 < C2 and C4 < C3 then
my.Direction = GetDirection(my.X , my.Y , 35 , 41 , universe.area)
end if
destin = my.Direction
avoidwallandpeople()
elseif my.point2 == false then
C1 = MeasureDistance(my.X , my.Y , 7 , 24 , universe.area)

```

```

C2 = MeasureDistance(my.X , my.Y , 13.5 , 23 , universe.area)
C3 = MeasureDistance(my.X , my.Y , 20 , 24 , universe.area)
C4 = MeasureDistance(my.X , my.Y , 28 , 23 , universe.area)
C5 = MeasureDistance(my.X , my.Y , 42.5 , 23 , universe.area)
C6 = MeasureDistance(my.X , my.Y , 49.5 , 24 , universe.area)
C7 = MeasureDistance(my.X , my.Y , 57 , 23 , universe.area)
C8 = MeasureDistance(my.X , my.Y , 21.5 , 41 , universe.area)
C9 = MeasureDistance(my.X , my.Y , 32 , 41 , universe.area)
C10 = MeasureDistance(my.X , my.Y , 54 , 41 , universe.area)
  if C1 < C2 and C1 < C3 and C1 < C4 and C1 < C5 and C1 < C6
    and C1 < C7 and C1 < C8 and C1 < C9 and C1 < C10 then
      my.Direction = GetDirection(my.X , my.Y , 7 , 24 , universe.area)
    elseif C2 < C1 and C2 < C3 and C2 < C4 and C2 < C5 and C2 < C6
      and C2 < C7 and C2 < C8 and C2 < C9 and C2 < C10 then
        my.Direction = GetDirection(my.X , my.Y , 13.5 , 23 , universe.area)
      elseif C3 < C1 and C3 < C2 and C3 < C4 and C3 < C5 and C3 < C6
        and C3 < C7 and C3 < C8 and C3 < C9 and C3 < C10 then
          my.Direction = GetDirection(my.X , my.Y , 20 , 24 , universe.area)
        elseif C4 < C1 and C4 < C2 and C4 < C3 and C4 < C5 and C4 < C6
          and C4 < C7 and C4 < C8 and C4 < C9 and C4 < C10 then
            my.Direction = GetDirection(my.X , my.Y , 28 , 23 , universe.area)
          elseif C5 < C1 and C5 < C2 and C5 < C3 and C5 < C4 and C5 < C6
            and C5 < C7 and C5 < C8 and C5 < C9 and C5 < C10 then
              my.Direction = GetDirection(my.X , my.Y , 42.5 , 23 , universe.area)
            elseif C6 < C1 and C6 < C2 and C6 < C3 and C6 < C4 and C6 < C5
              and C6 < C7 and C6 < C8 and C6 < C9 and C6 < C10 then
                my.Direction = GetDirection(my.X , my.Y , 49.5 , 24 , universe.area)
              elseif C7 < C1 and C7 < C2 and C7 < C3 and C7 < C4 and C7 < C5
                and C7 < C6 and C7 < C8 and C7 < C9 and C7 < C10 then
                  my.Direction = GetDirection(my.X , my.Y , 57 , 23 , universe.area)
                elseif C8 < C1 and C8 < C2 and C8 < C3 and C8 < C4 and C8 < C5
                  and C8 < C6 and C8 < C7 and C8 < C9 and C8 < C10 then
                    my.Direction = GetDirection(my.X , my.Y , 21.5 , 41 , universe.area)
                  elseif C9 < C1 and C9 < C2 and C9 < C3 and C9 < C4 and C9 < C5
                    and C9 < C6 and C9 < C7 and C9 < C8 and C9 < C10 then
                      my.Direction = GetDirection(my.X , my.Y , 32 , 41 , universe.area)
                    elseif C10 < C1 and C10 < C2 and C10 < C3 and C10 < C4 and C10 < C5
                      and C10 < C6 and C10 < C7 and C10 < C8 and C10 < C9 then
                        my.Direction = GetDirection(my.X , my.Y , 54 , 41 , universe.area)
                      end if
                destin = my.Direction
                avoidwallandpeople()
elseif my.point3 == false and my.point4 == false and my.point5 == false then
C1 = measureDistance(my.X , my.Y , 13 , 26 , universe.area)
C2 = measureDistance(my.X , my.Y , 53.5 , 26 , universe.area)
C3 = measureDistance(my.X , my.Y , 25.5 , 40 , universe.area)
C4 = measureDistance(my.X , my.Y , 12 , 4 , universe.area)
  if C1 < C2 and C1 < C3 and C1 < C4 then
    my.Direction = getdirection(my.X , my.Y , 13 , 26 , universe.area)
  elseif C2 < C1 and C2 < C3 and C2 < C4 then
    my.Direction = getdirection(my.X , my.Y , 53.5 , 26 , universe.area)
  elseif C3 < C1 and C3 < C2 and C3 < C4 then
    my.Direction = getdirection(my.X , my.Y , 25.5 , 40 , universe.area)

```

```

elseif C4 < C1 and C4 < C2 and C4 < C3 then
my.Direction = getdirection(my.X , my.Y , 12 , 4 , universe.area)
end if
destin = my.Direction
avoidwallandpeople()
elseif my.point4 == false and my.point3 == true then
C1 = measureDistance(my.X , my.Y , 13.5 , 40 , universe.area)
C2 = measureDistance(my.X , my.Y , 54 , 41 , universe.area)
if C1 < C2 then
my.Direction = getdirection(my.X , my.Y , 13.5 , 40 , universe.area)
elseif C2 < C1 then
my.Direction = getdirection(my.X , my.Y , 54 , 41 , universe.area)
end if
destin = my.Direction
avoidwallandpeople()
elseif my.point5 == false and my.point4 == true then
C1 = measureDistance(my.X , my.Y , 25.5 , 40 , universe.area)
C2 = measureDistance(my.X , my.Y , 59.5 , 41 , universe.area)
if C1 < C2 then
my.Direction = getdirection(my.X , my.Y , 25.5 , 40 , universe.area)
elseif C2 < C1 then
my.Direction = getdirection(my.X , my.Y , 59.5 , 41 , universe.area)
end if
destin = my.Direction
avoidwallandpeople()
elseif my.point5 == true then
my.Direction = getdirection(my.X , my.Y , 26 , 36 , universe.area)
destin = my.Direction
avoidwallandpeople()
end if

```

//五階非常階段混雑時ルール//

```

elseif my.change == 1 then
if my.U5 == false then
if my.point3 == false then
my.Direction = GetDirection(my.X , my.Y , 14 , 26 , universe.area)
destin = my.Direction
avoidwallandpeople()
elseif my.point4 == false then
C1 = measureDistance(my.X , my.Y , 13.5 , 40 , universe.area)
C2 = measureDistance(my.X , my.Y , 54 , 41 , universe.area)
if C1 < C2 then
my.Direction = getdirection(my.X , my.Y , 13.5 , 40 , universe.area)
elseif C2 < C1 then
my.Direction = getdirection(my.X , my.Y , 54 , 41 , universe.area)
end if
destin = my.Direction
avoidwallandpeople()
elseif my.point5 == false then
my.Direction = getdirection(my.X , my.Y , 25.5 , 40 , universe.area)
destin = my.Direction
avoidwallandpeople()
elseif my.point5 == true then

```

```

    my.Direction = getdirection(my.X , my.Y , 26 , 28 , universe.area)
    destin = my.Direction
    avoidwallandpeople()
    end if
elseif my.U5 == true then
    if my.point5 == false then
        my.Direction = getdirection(my.X , my.Y , 25.5 , 40 , universe.area)
        destin = my.Direction
        avoidwallandpeople()
        elseif my.point5 == true then
            my.Direction = getdirection(my.X , my.Y , 26 , 28 , universe.area)
            destin = my.Direction
            avoidwallandpeople()
        end if
    end if
//五階中央階段混雑時ルール//
elseif my.change == 2 then
    if my.U5 == false then
        if my.point3 == false then
            my.Direction = getdirection(my.X , my.Y , 53.5 , 27 , universe.area)
            destin = my.Direction
            avoidwallandpeople()
            elseif my.point4 == false then
                C1 = measureDistance(my.X , my.Y , 13.5 , 40 , universe.area)
                C2 = measureDistance(my.X , my.Y , 54 , 41 , universe.area)
                if C1 < C2 then
                    my.Direction = getdirection(my.X , my.Y , 13.5 , 40 , universe.area)
                elseif C2 < C1 then
                    my.Direction = getdirection(my.X , my.Y , 54 , 41 , universe.area)
                end if
                destin = my.Direction
                avoidwallandpeople()
                elseif my.point4 == true then
                    my.Direction = getdirection(my.X , my.Y , 59.5 , 41 , universe.area)
                    destin = my.Direction
                    avoidwallandpeople()
                end if
            end if
        elseif my.U5 == true then
            if my.point2 == false then
                my.Direction = GetDirection(my.X , my.Y , 53.5 , 41 , universe.area)
                destin = my.Direction
                avoidwallandpeople()
                elseif my.point4 == false then
                    my.Direction = getdirection(my.X , my.Y , 54 , 41 , universe.area)
                    destin = my.Direction
                    avoidwallandpeople()
                elseif my.point4 == true then
                    my.Direction = getdirection(my.X , my.Y , 59.5 , 41 , universe.area)
                    destin = my.Direction
                    avoidwallandpeople()
                end if
            end if
        end if
    end if
end if

```

```

//非常階段//
elseif my.down2 == true then
  if my.point12 == true then
    my.Direction = getdirection(my.X , my.Y , 60 , 36 , universe.area)
    destin = my.Direction
    avoidwallandpeople()
  elseif my.point11 == true then
    my.Direction = getdirection(my.X , my.Y , 60 , 41 , universe.area)
    destin = my.Direction
    avoidwallandpeople()
  elseif my.point10 == true then
    my.Direction = getdirection(my.X , my.Y , 56.5 , 42 , universe.area)
    destin = my.Direction
    avoidwallandpeople()
  elseif my.point1 == true then
    my.Direction = getdirection(my.X , my.Y , 58 , 41 , universe.area)
    destin = my.Direction
    avoidwallandpeople()
  elseif my.point1 == false then
    my.Direction = getdirection(my.X , my.Y , 56 , 33 , universe.area)
    destin = my.Direction
    avoidwallandpeople()
  end if
//階段//
elseif my.down1 == true then
if my.point13 == true then
  my.Direction = getdirection(my.X , my.Y , 22 , 31 , universe.area)
  destin = my.Direction
  avoidwallandpeople()
elseif my.point12 == true then
  my.Direction = getdirection(my.X , my.Y , 22 , 28 , universe.area)
  destin = my.Direction
  avoidwallandpeople()
elseif my.point11 == true then
  my.Direction = getdirection(my.X , my.Y , 26 , 27 , universe.area)
  destin = my.Direction
  avoidwallandpeople()
elseif my.point10 == true then
  my.Direction = getdirection(my.X , my.Y , 27 , 36 , universe.area)
  destin = my.Direction
  avoidwallandpeople()
elseif my.point1 == false and my.point10 == false then
  my.Direction = getdirection(my.X , my.Y , 21 , 36 , universe.area)
  destin = my.Direction
  avoidwallandpeople()
elseif my.point11 == false then
  my.Direction = getdirection(my.X , my.Y , 26 , 36 , universe.area)
  destin = my.Direction
  avoidwallandpeople()
end if
end if
// 六階行動ルール//
elseif my.Layer == 5 then

```

```

if my.down1 == false and my.down2 == false then
  if my.change == 0 then
    if my.point1 == false then
      C1 = MeasureDistance(my.X , my.Y , 6.5 , 23 , universe.area)
      C2 = MeasureDistance(my.X , my.Y , 21 , 23 , universe.area)
      C3 = MeasureDistance(my.X , my.Y , 35 , 23 , universe.area)
      C4 = MeasureDistance(my.X , my.Y , 36 , 41 , universe.area)
      if C1 < C2 and C1 < C3 and C1 < C4 then
        my.Direction = GetDirection(my.X , my.Y , 6.5 , 23 , universe.area)
      elseif C2 < C1 and C2 < C3 and C2 < C4 then
        my.Direction = GetDirection(my.X , my.Y , 21 , 23 , universe.area)
      elseif C3 < C1 and C3 < C2 and C3 < C4 then
        my.Direction = GetDirection(my.X , my.Y , 35 , 23 , universe.area)
      elseif C4 < C1 and C4 < C2 and C4 < C3 then
        my.Direction = GetDirection(my.X , my.Y , 36 , 41 , universe.area)
      end if
      destin = my.Direction
      avoidwallandpeople()
    elseif my.point2 == false then
      C1 = MeasureDistance(my.X , my.Y , 7 , 24 , universe.area)
      C2 = MeasureDistance(my.X , my.Y , 13.5 , 23 , universe.area)
      C3 = MeasureDistance(my.X , my.Y , 20 , 24 , universe.area)
      C4 = MeasureDistance(my.X , my.Y , 28 , 23 , universe.area)
      C5 = MeasureDistance(my.X , my.Y , 57 , 23 , universe.area)
      C6 = MeasureDistance(my.X , my.Y , 35 , 24 , universe.area)
      C7 = MeasureDistance(my.X , my.Y , 21.5 , 41 , universe.area)
      C8 = MeasureDistance(my.X , my.Y , 32 , 41 , universe.area)
      C9 = MeasureDistance(my.X , my.Y , 50 , 41 , universe.area)
      if C1 < C2 and C1 < C3 and C1 < C4 and C1 < C5 and C1 < C6
        and C1 < C7 and C1 < C8 and C1 < C9 then
        my.Direction = GetDirection(my.X , my.Y , 7 , 24 , universe.area)
      elseif C2 < C1 and C2 < C3 and C2 < C4 and C2 < C5 and C2 < C6
        and C2 < C7 and C2 < C8 and C2 < C9 then
        my.Direction = GetDirection(my.X , my.Y , 13.5 , 23 , universe.area)
      elseif C3 < C1 and C3 < C2 and C3 < C4 and C3 < C5 and C3 < C6
        and C3 < C7 and C3 < C8 and C3 < C9 then
        my.Direction = GetDirection(my.X , my.Y , 20 , 24 , universe.area)
      elseif C4 < C1 and C4 < C2 and C4 < C3 and C4 < C5 and C4 < C6
        and C4 < C7 and C4 < C8 and C4 < C9 then
        my.Direction = GetDirection(my.X , my.Y , 28 , 23 , universe.area)
      elseif C5 < C1 and C5 < C2 and C5 < C3 and C5 < C4 and C5 < C6
        and C5 < C7 and C5 < C8 and C5 < C9 then
        my.Direction = GetDirection(my.X , my.Y , 57 , 23 , universe.area)
      elseif C6 < C1 and C6 < C2 and C6 < C3 and C6 < C4 and C6 < C5
        and C6 < C7 and C6 < C8 and C6 < C9 then
        my.Direction = GetDirection(my.X , my.Y , 35 , 24 , universe.area)
      elseif C7 < C1 and C7 < C2 and C7 < C3 and C7 < C4 and C7 < C5
        and C7 < C6 and C7 < C8 and C7 < C9 then
        my.Direction = GetDirection(my.X , my.Y , 21.5 , 41 , universe.area)
      elseif C8 < C1 and C8 < C2 and C8 < C3 and C8 < C4 and C8 < C5
        and C8 < C6 and C8 < C7 and C8 < C9 then
        my.Direction = GetDirection(my.X , my.Y , 32 , 41 , universe.area)
      elseif C9 < C1 and C9 < C2 and C9 < C3 and C9 < C4 and C9 < C5

```

```

and C9 < C6 and C9 < C7 and C9 < C8 then
my.Direction = GetDirection(my.X , my.Y , 50 , 41 , universe.area)
end if
destin = my.Direction
avoidwallandpeople()
elseif my.point3 == false and my.point4 == false and my.point5 == false then
C1 = measureDistance(my.X , my.Y , 13 , 26 , universe.area)
C2 = measureDistance(my.X , my.Y , 53.5 , 26 , universe.area)
C3 = measureDistance(my.X , my.Y , 25 , 40 , universe.area)
C4 = measureDistance(my.X , my.Y , 54 , 41 , universe.area)
if C1 < C2 and C1 < C3 and C1 < C4 then
my.Direction = getdirection(my.X , my.Y , 13 , 26 , universe.area)
elseif C2 < C1 and C2 < C3 and C2 < C4 then
my.Direction = getdirection(my.X , my.Y , 53.5 , 26 , universe.area)
elseif C3 < C1 and C3 < C2 and C3 < C4 then
my.Direction = getdirection(my.X , my.Y , 25 , 40 , universe.area)
elseif C4 < C1 and C4 < C2 and C4 < C3 then
my.Direction = getdirection(my.X , my.Y , 54 , 41 , universe.area)
end if
destin = my.Direction
avoidwallandpeople()
elseif my.point4 == false and my.point3 == true then
C1 = measureDistance(my.X , my.Y , 13.5 , 40 , universe.area)
C2 = measureDistance(my.X , my.Y , 54 , 41 , universe.area)
if C1 < C2 then
my.Direction = getdirection(my.X , my.Y , 13.5 , 40 , universe.area)
elseif C2 < C1 then
my.Direction = getdirection(my.X , my.Y , 54 , 41 , universe.area)
end if
destin = my.Direction
avoidwallandpeople()
elseif my.point5 == false and my.point4 == true then
C1 = measureDistance(my.X , my.Y , 25.5 , 40 , universe.area)
C2 = measureDistance(my.X , my.Y , 59.5 , 41 , universe.area)
if C1 < C2 then
my.Direction = getdirection(my.X , my.Y , 25.5 , 40 , universe.area)
elseif C2 < C1 then
my.Direction = getdirection(my.X , my.Y , 59.5 , 41 , universe.area)
end if
destin = my.Direction
avoidwallandpeople()
elseif my.point5 == true then
my.Direction = getdirection(my.X , my.Y , 26 , 36 , universe.area)
destin = my.Direction
avoidwallandpeople()
end if
//六階非常階段混雑時ルール//
elseif my.change == 1 then
if my.U6 == false then
if my.point3 == false then
my.Direction = GetDirection(my.X , my.Y , 14 , 26 , universe.area)
destin = my.Direction
avoidwallandpeople()

```

```

elseif my.point4 == false then
C1 = measureDistance(my.X , my.Y , 13.5 , 40 , universe.area)
C2 = measureDistance(my.X , my.Y ,54, 41 , universe.area)
if C1 < C2 then
my.Direction = getdirection(my.X , my.Y , 13.5 , 40 , universe.area)
elseif C2 < C1 then
my.Direction = getdirection(my.X , my.Y , 54 , 41 , universe.area)
end if
destin = my.Direction
avoidwallandpeople()
elseif my.point5 == false then
my.Direction = getdirection(my.X , my.Y , 25.5 , 40 , universe.area)
destin = my.Direction
avoidwallandpeople()
elseif my.point5 == true then
my.Direction = getdirection(my.X , my.Y , 26 , 28 , universe.area)
destin = my.Direction
avoidwallandpeople()
end if
elseif my.U6 == true then
if my.point5 == false then
my.Direction = getdirection(my.X , my.Y , 25.5 , 40 , universe.area)
destin = my.Direction
avoidwallandpeople()
elseif my.point5 == true then
my.Direction = getdirection(my.X , my.Y , 26 , 28 , universe.area)
destin = my.Direction
avoidwallandpeople()
end if
end if
//六階中央階段混雑時ルール//
elseif my.change == 2 then
if my.U6 == false then
if my.point3 == false then
my.Direction = getdirection(my.X , my.Y , 53.5 , 27 , universe.area)
destin = my.Direction
avoidwallandpeople()
elseif my.point4 == false then
C1 = measureDistance(my.X , my.Y , 13.5 , 40 , universe.area)
C2 = measureDistance(my.X , my.Y ,54, 41 , universe.area)
if C1 < C2 then
my.Direction = getdirection(my.X , my.Y , 13.5 , 40 , universe.area)
elseif C2 < C1 then
my.Direction = getdirection(my.X , my.Y , 54 , 41 , universe.area)
end if
destin = my.Direction
avoidwallandpeople()
elseif my.point4 == true then
my.Direction = getdirection(my.X , my.Y , 59.5 , 41 , universe.area)
destin = my.Direction
avoidwallandpeople()
end if
elseif my.U6 == true then

```

```

    if my.point2 == false then
        my.Direction = GetDirection(my.X , my.Y , 50 , 41 , universe.area)
        destin = my.Direction
        avoidwallandpeople()
    elseif my.point4 == false then
        my.Direction = getdirection(my.X , my.Y , 53.5 , 41 , universe.area)
        destin = my.Direction
        avoidwallandpeople()
    elseif my.point4 == true then
        my.Direction = getdirection(my.X , my.Y , 59.5 , 41 , universe.area)
        destin = my.Direction
        avoidwallandpeople()
    end if
end if
end if
//非常阶段//
elseif my.down2 == true then
    if my.point12 == true then
        my.Direction = getdirection(my.X , my.Y , 60 , 36 , universe.area)
        destin = my.Direction
        avoidwallandpeople()
    elseif my.point11 == true then
        my.Direction = getdirection(my.X , my.Y , 60 , 41 , universe.area)
        destin = my.Direction
        avoidwallandpeople()
    elseif my.point1 == true then
        my.Direction = getdirection(my.X , my.Y , 58 , 41 , universe.area)
        destin = my.Direction
        avoidwallandpeople()
    end if
//阶段//
elseif my.down1 == true then
    if my.point13 == true then
        my.Direction = getdirection(my.X , my.Y , 22 , 31 , universe.area)
        destin = my.Direction
        avoidwallandpeople()
    elseif my.point12 == true then
        my.Direction = getdirection(my.X , my.Y , 22 , 28 , universe.area)
        destin = my.Direction
        avoidwallandpeople()
    elseif my.point11 == true then
        my.Direction = getdirection(my.X , my.Y , 26 , 27 , universe.area)
        destin = my.Direction
        avoidwallandpeople()
    elseif my.point11 == false and my.point1 == true then
        my.Direction = getdirection(my.X , my.Y , 26 , 36 , universe.area)
        destin = my.Direction
        avoidwallandpeople()
    end if
end if
end if

```

付 チェックポイントエージェント 1

```
Agt_Init{  
}  
Agt_Step{  
  Dim surrou As Agtset  
  Dim ok As Agt  
  MakeoneAgtsetAroundOwn(surrou,1,Universe.area.people,False)  
  If CountAgtSet(surrou)>0 Then  
    ok=GetAgt(surrou,int(Rnd()*CountAgtset(surrou)))  
    ok.point1 = True  
  End if  
}
```

付 チェックポイントエージェント 2

```
Agt_Init{  
}  
Agt_Step{  
  Dim surrou As Agtset  
  Dim ok As Agt  
  MakeoneAgtsetAroundOwn(surrou,1,Universe.area.people,False)  
  If CountAgtSet(surrou)>0 Then  
    ok=GetAgt(surrou,int(Rnd()*CountAgtset(surrou)))  
    ok.point2 = True  
  End if  
}
```

付 チェックポイントエージェント 3

```
Agt_Init{  
}  
Agt_Step{  
  Dim surrou As Agtset  
  Dim ok As Agt  
  MakeoneAgtsetAroundOwn(surrou,2,Universe.area.people,False)  
  If CountAgtSet(surrou)>0 Then  
    ok=GetAgt(surrou,int(Rnd()*CountAgtset(surrou)))  
    ok.point3 = True  
  End if  
}
```

付 チェックポイントエージェント 4

```
Agt_Init{  
}  
Agt_Step{  
  Dim surrou As Agtset  
  Dim ok As Agt  
  MakeoneAgtsetAroundOwn(surrou,1,Universe.area.people,False)  
  If CountAgtSet(surrou)>0 Then  
    ok=GetAgt(surrou,int(Rnd()*CountAgtset(surrou)))  
    ok.point4 = True  
  End if  
}
```

付 チェックポイントエージェント 5

```
Agt_Init{
}
Agt_Step{
  Dim surrou As Agtset
  Dim ok As Agt
  MakeoneAgtsetAroundOwn(surrou,1,Universe.area.people,False)
  If CountAgtSet(surrou)>0 Then
    ok=GetAgt(surrou,int(Rnd()*CountAgtset(surrou)))
    ok.point5 = True
  End if
}
```

付 チェックポイントエージェント 10

```
Agt_Init{
}
Agt_Step{
  Dim surrou As Agtset
  Dim ok As Agt
  MakeoneAgtsetAroundOwn(surrou,1,Universe.area.people,False)
  If CountAgtSet(surrou)>0 Then
    ok=GetAgt(surrou,int(Rnd()*CountAgtset(surrou)))
    ok.point10 = True
  End if
}
```

付 チェックポイントエージェント 11

```
Agt_Init{
}
Agt_Step{
  Dim surrou As Agtset
  Dim ok As Agt
  MakeoneAgtsetAroundOwn(surrou,1,Universe.area.people,False)
  If CountAgtSet(surrou)>0 Then
    ok=GetAgt(surrou,int(Rnd()*CountAgtset(surrou)))
    ok.point11 = True
  End if
}
```

付 チェックポイントエージェント 12

```
Agt_Init{
}
Agt_Step{
  Dim surrou As Agtset
  Dim ok As Agt
  MakeoneAgtsetAroundOwn(surrou,1,Universe.area.people,False)
  If CountAgtSet(surrou)>0 Then
    ok=GetAgt(surrou,int(Rnd()*CountAgtset(surrou)))
    ok.point12 = True
  End if
}
```

付 チェックポイントエージェント 13

```
Agt_Init{
}
Agt_Step{
  Dim surrou As Agtset
  Dim ok As Agt
  MakeoneAgtsetAroundOwn(surrou,1,Universe.area.people,False)
  If CountAgtSet(surrou)>0 Then
    ok=GetAgt(surrou,int(Rnd()*CountAgtset(surrou)))
    ok.point13 = True
  End if
}
```

付 階段エージェント 1

```
Agt_Init{
}
Agt_Step{
  Dim surrou As Agtset
  Dim pass As Agt
  MakeoneAgtsetAroundOwn(surrou,2,Universe.area.people,False)
  If CountAgtSet(surrou)>0 Then
    pass=GetAgt(Surrou,int(Rnd()*CountAgtset(surrou)))
    If pass.Layer==1 or pass.layer == 2 or pass.layer == 3 or pass.layer == 4 or pass.layer == 5 Then
      pass.stapN =True
    End if
  End if
}
```

付 階段エージェント 2

```
Agt_Init{
}
Agt_Step{
  Dim surrou As Agtset
  Dim pass As Agt
  MakeoneAgtsetAroundOwn(surrou,2,Universe.area.people,False)
  If CountAgtSet(surrou)>0 Then
    pass=GetAgt(Surrou,int(Rnd()*CountAgtset(surrou)))
    If pass.Layer==1 or pass.layer == 2 or pass.layer == 3 or pass.layer == 4 or pass.layer == 5 Then
      pass.stapE =True
    End if
  End if
}
```

付 降下エージェント 1

```
Agt_Init{
}
Agt_Step{
Dim surrou As Agtset
Dim pass As Agt
  MakeoneAgtsetAroundOwn(surrou,1,Universe.area.people,False)
  If CountAgtSet(surrou)>0 Then
    pass=GetAgt(Surrou,int(Rnd()*CountAgtset(surrou)))
    pass.down1=True
    //階避難完了処理
    If pass.floor ==1 Then
      pass.floor =6
      universe.f2all = universe.f2all - 1
    elseif
      pass.floor == 2 then
      pass.floor = 6
      universe.f3all = universe.f3all - 1
    elseif
      pass.floor == 3 then
      pass.floor = 6
      universe.f4all = universe.f4all - 1
    elseif
      pass.floor == 4 then
      pass.floor = 6
      universe.f5all = universe.f5all - 1
    elseif
      pass.floor == 5 then
      pass.floor = 6
      universe.f6all = universe.f6all - 1
    End if
  End if
}
```

付 降下エージェント 2

```
Agt_Init{
}
Agt_Step{
Dim surrou As Agtset
Dim pass As Agt
  MakeoneAgtsetAroundOwn(surrou,1,Universe.area.people,False)
  If CountAgtSet(surrou)>0 Then
    pass=GetAgt(Surrou,int(Rnd()*CountAgtset(surrou)))
    if pass.down2 == false then
      pass.down2=True
      //階避難完了処理
    If pass.floor ==1 Then
      pass.floor =6
      universe.f2all = universe.f2all - 1
    elseif
      pass.floor == 2 then
      pass.floor = 6
      universe.f3all = universe.f3all - 1
```

```
elseif
pass.floor == 3 then
pass.floor = 6
universe.f4all = universe.f4all - 1
elseif
pass.floor == 4 then
pass.floor = 6
universe.f5all = universe.f5all - 1
elseif
pass.floor == 5 then
pass.floor = 6
universe.f6all = universe.f6all - 1
end if
End if
End if
```

付 出口エージェント

```
Agt_Init{
}
Agt_Step{
dim surrou as agtset
dim escape as agt
if Rnd() < 0.25 then
  makeoneAgtSetAroundOwn(surrou,2,universe.area.people, false)
  if countAgtSet(surrou) > 0 then
    escape = getAgt(surrou,int(rnd()*countagtset(surrou)))
    if escape.floor == 0 then
      escape.floor = 6
      universe.flall = universe.flall - 1
    end if
    delagt(escape)
  end if
end if
}
```

付 階段出口エージェント

```
Agt_Init{
}
Agt_Step{
dim surrou as agtset
dim escape as agt
if Rnd() < 0.23 then
  makeoneAgtSetAroundOwn(surrou,2,universe.area.people, false)
  if countAgtSet(surrou) > 0 then
    escape = getAgt(surrou,int(rnd()*countagtset(surrou)))
    if escape.floor == 0 then
      escape.floor = 6
      universe.flall = universe.flall - 1
    end if
    delagt(escape)
  end if
end if
}
```

付 人数エージェント 1-1

```
Agt_Init{  
}  
Agt_Step{  
Dim surround As Agtset  
MakeoneAgtsetAroundOwncell(surround,2,Universe.area.people,False)  
Universe.area.cauntF11=CountAgtSet(surround)  
}
```

付 人数エージェント 1-2

```
Agt_Init{  
}  
Agt_Step{  
Dim surround As Agtset  
MakeoneAgtsetAroundOwncell(surround,2,Universe.area.people,False)  
Universe.area.cauntF12=CountAgtSet(surround)  
}
```

付 人数エージェント 1-3

```
Agt_Init{  
}  
Agt_Step{  
Dim surround As Agtset  
MakeoneAgtsetAroundOwncell(surround,2,Universe.area.people,False)  
Universe.area.cauntF13=CountAgtSet(surround)  
}
```

付 人数エージェント人 2-1

```
Agt_Init{  
}  
Agt_Step{  
Dim surround As Agtset  
MakeoneAgtsetAroundOwncell(surround,2,Universe.area.people,False)  
Universe.area.cauntF21=CountAgtSet(surround)  
}
```

付 人数エージェント 2-2

```
Agt_Init{  
}  
Agt_Step{  
Dim surround As Agtset  
MakeoneAgtsetAroundOwncell(surround,2,Universe.area.people,False)  
Universe.area.cauntF22=CountAgtSet(surround)  
}
```

付 人数エージェント 3-1

```
Agt_Init{  
}  
Agt_Step{  
Dim surround As Agtset  
MakeoneAgtsetAroundOwncell(surround,2,Universe.area.people,False)  
Universe.area.cauntF31=CountAgtSet(surround)  
}
```

付 人数エージェント 3-2

```
Agt_Init{  
}  
Agt_Step{  
Dim surround As Agtset  
MakeoneAgtsetAroundOwncell(surround,2,Universe.area.people,False)  
Universe.area.cauntF32=CountAgtSet(surround)  
}
```

付 人数エージェント 4-1

```
Agt_Init{  
}  
Agt_Step{  
Dim surround As Agtset  
MakeoneAgtsetAroundOwncell(surround,2,Universe.area.people,False)  
Universe.area.cauntF41=CountAgtSet(surround)  
}
```

付 人数エージェント 4-2

```
Agt_Init{  
}  
Agt_Step{  
Dim surround As Agtset  
MakeoneAgtsetAroundOwncell(surround,2,Universe.area.people,False)  
Universe.area.cauntF42=CountAgtSet(surround)  
}
```

付 人数エージェント 5-1

```
Agt_Init{  
}  
Agt_Step{  
Dim surround As Agtset  
MakeoneAgtsetAroundOwncell(surround,2,Universe.area.people,False)  
Universe.area.cauntF51=CountAgtSet(surround)  
}
```

付 人数エージェント 5-2

```
Agt_Init{  
}  
Agt_Step{  
Dim surround As Agtset  
MakeoneAgtsetAroundOwncell(surround,2,Universe.area.people,False)  
Universe.area.cauntF52=CountAgtSet(surround)  
}
```

付 人数エージェント 6-1

```
Agt_Init{  
}  
Agt_Step{  
Dim surround As Agtset  
MakeoneAgtsetAroundOwncell(surround,2,Universe.area.people,False)  
Universe.area.cauntF61=CountAgtSet(surround)  
}
```

付 人数エージェント 6-2

```
Agt_Init{  
}  
Agt_Step{  
  Dim surround As Agtset  
  MakeoneAgtsetAroundOwncell(surround,2,Universe.area.people,False)  
  Universe.area.cauntF62=CountAgtSet(surround)  
}
```

付 転換エージェント 1

```
Agt_Init{
}
Agt_Step{
dim surrou as agtset
dim c as agt
if Universe.area.cauntF11 >= 5 or Universe.area.cauntF12 >= 5
or Universe.area.cauntF13 >= 3 then
makeoneAgtSetAroundOwn(surrou,1,universe.area.people , false)
if countagtset(surrou) > 0 then
c = getagt(surrou,int(rnd()*countagtset(surrou)))
c.change = 0
if c.change == 0 then
if universe.area.cauntF13 < universe.area.cauntF11 and
universe.area.cauntF13 < universe.area.cauntF12 then
c.change = 1
elseif universe.area.cauntF12 < universe.area.cauntF11 and
universe.area.cauntF12 < universe.area.cauntF13 then
c.change = 2
elseif universe.area.cauntF11 < universe.area.cauntF12 and
universe.area.cauntF11 < universe.area.cauntF13 then
c.change = 3
end if
end if
end if
end if
}
```

付 転換エージェント 2

```
Agt_Init{
}
Agt_Step{
dim surrou as agtset
dim c as agt
if Universe.area.cauntF21 >= 5 or Universe.area.cauntF22 >= 5 then
makeoneAgtSetAroundOwn(surrou,1,universe.area.people , false)
if countagtset(surrou) > 0 then
c = getagt(surrou,int(rnd()*countagtset(surrou)))
if c.change == 0 then
if universe.area.cauntF22 < universe.area.cauntF21 then
c.change = 1
elseif
universe.area.cauntF21 < universe.area.cauntF22 then
c.change = 2
end if
end if
end if
end if
}
```

付 転換エージェント 3

```
Agt_Init{
}
Agt_Step{
dim surrou as agtset
dim c as agt
if Universe.area.cauntF31 >= 5 or Universe.area.cauntF32 >= 5 then
makeoneAgtSetAroundOwn(surrou,1,universe.area.people , false)
if countagtset(surrou) > 0 then
c = getagt(surrou,int(rnd()*countagtset(surrou)))
if c.change == 0 then
if universe.area.cauntF32 < universe.area.cauntF31 then
c.change = 1
elseif
universe.area.cauntF31 < universe.area.cauntF32 then
c.change = 2
end if
end if
end if
end if
}
```

付 転換エージェント 4

```
Agt_Init{
}
Agt_Step{
dim surrou as agtset
dim c as agt
if Universe.area.cauntF41 >= 5 or Universe.area.cauntF42 >= 5 then
makeoneAgtSetAroundOwn(surrou,1,universe.area.people , false)
if countagtset(surrou) > 0 then
c = getagt(surrou,int(rnd()*countagtset(surrou)))
if c.change == 0 then
if universe.area.cauntF42 < universe.area.cauntF41 then
c.change = 1
elseif
universe.area.cauntF41 < universe.area.cauntF42 then
c.change = 2
end if
end if
end if
end if
}
```

付 転換エージェント 5

```
Agt_Init{
}
Agt_Step{
dim surrou as agtset
dim c as agt
if Universe.area.cauntF51 >= 5 or Universe.area.cauntF52 >= 5 then
makeoneAgtSetAroundOwn(surrou,1,universe.area.people , false)
if countagtset(surrou) > 0 then
```

```

c = getagt(surrou,int(rnd()*countagtset(surrou)))
if c.change == 0 then
  if universe.area.cauntF52 < universe.area.cauntF51 then
    c.change = 1
  elseif
    universe.area.cauntF51 < universe.area.cauntF52 then
    c.change = 2
  end if
end if
end if
end if
}

```

付 轉換エージェント 6

```

Agt_Init{
}
Agt_Step{
dim surrou as agtset
dim c as agt
if Universe.area.cauntF61 >= 5 or Universe.area.cauntF62 >= 5 then
makeoneAgtSetAroundOwn(surrou,1,universe.area.people , false)
if countagtset(surrou) > 0 then
c = getagt(surrou,int(rnd()*countagtset(surrou)))
if c.change == 0 then
  if universe.area.cauntF62 < universe.area.cauntF61 then
    c.change = 1
  elseif
    universe.area.cauntF61 < universe.area.cauntF62 then
    c.change = 2
  end if
end if
end if
end if
}

```

付 場所エージェント 1

```
Agt_Init{
}
Agt_Step{
Dim surrou As Agtset
Dim ok As Agt
  MakeoneAgtsetAroundOwn(surrou,1,Universe.area.people,False)
  If CountAgtSet(surrou)>0 Then
    ok=GetAgt(surrou,int(Rnd()*CountAgtset(surrou)))
    ok.U1 = True
  End if
}
```

付 場所エージェント 2

```
Agt_Init{
}
Agt_Step{
Dim surrou As Agtset
Dim ok As Agt
  MakeoneAgtsetAroundOwn(surrou,1,Universe.area.people,False)
  If CountAgtSet(surrou)>0 Then
    ok=GetAgt(surrou,int(Rnd()*CountAgtset(surrou)))
    ok.U2 = True
  End if
}
```

付 場所エージェント 3

```
Agt_Init{
}
Agt_Step{
Dim surrou As Agtset
Dim ok As Agt
  MakeoneAgtsetAroundOwn(surrou,1,Universe.area.people,False)
  If CountAgtSet(surrou)>0 Then
    ok=GetAgt(surrou,int(Rnd()*CountAgtset(surrou)))
    ok.U3 = True
  End if
}
```

付 場所エージェント 4

```
Agt_Init{
}
Agt_Step{
Dim surrou As Agtset
Dim ok As Agt
  MakeoneAgtsetAroundOwn(surrou,1,Universe.area.people,False)
  If CountAgtSet(surrou)>0 Then
    ok=GetAgt(surrou,int(Rnd()*CountAgtset(surrou)))
    ok.U4 = True
  End if
}
```

付 場所エージェント 5

```
Agt_Init{  
}  
Agt_Step{  
  Dim surrou As Agtset  
  Dim ok As Agt  
  MakeoneAgtsetAroundOwn(surrou,1,Universe.area.people,False)  
  If CountAgtSet(surrou)>0 Then  
    ok=GetAgt(surrou,int(Rnd()*CountAgtset(surrou)))  
    ok.U5 = True  
  End if  
}
```

付 場所エージェント 6

```
Agt_Init{  
}  
Agt_Step{  
  Dim surrou As Agtset  
  Dim ok As Agt  
  MakeoneAgtsetAroundOwn(surrou,1,Universe.area.people,False)  
  If CountAgtSet(surrou)>0 Then  
    ok=GetAgt(surrou,int(Rnd()*CountAgtset(surrou)))  
    ok.U6 = True  
  End if  
}
```

付 復帰エージェント 1

```
Agt_Init{
}
Agt_Step{
dim surrou as agtset
dim out as agt
makeoneagtsetarowndown(surrou,1,universe.area.people,false)
if countagtset(surrou) > 0 then
out = getagt(surrou , int(rnd()*countagtset(surrou)))
if out.point2 == true then
out.Y = 42 - rnd()*1
else
out.X = 42 - rnd()*2
end if
end if
}
```

付 復帰エージェント 2

```
Agt_Init{
}
Agt_Step{
dim surrou as agtset
dim out as agt
makeoneagtsetarowndown(surrou,1,universe.area.people,false)
if countagtset(surrou) > 0 then
out = getagt(surrou , int(rnd()*countagtset(surrou)))
if out.down2 == true then
out.X = 55 + rnd()*3
out.Y = 41
end if
end if
}
```

付 復帰エージェント 3

```
Agt_Init{
}
Agt_Step{
dim surrou as agtset
dim out as agt
makeoneagtsetarowndown(surrou,1,universe.area.people,false)
if countagtset(surrou) > 0 then
out = getagt(surrou , int(rnd()*countagtset(surrou)))
if out.point3== true or out.point4 == true then
out.X = 53 + rnd()*1
out.Y = 40 + rnd()*1
end if
end if
}
```

付 復帰エージェント 4

```
Agt_Init{
}
Agt_Step{
dim surrou as agtset
```

```

dim out as agt
makeoneagtsetarowndown(surrou,1,universe.area.people,false)
if countagtset(surrou) > 0 then
out = getagt(surrou , int(rnd()*countagtset(surrou)))
if out.point11== true then
out.X = 57 + rnd()*1
out.Y = 40 + rnd()*1
end if
end if
}

```

付 復帰エージェント 5

```

Agt_Init{
}
Agt_Step{
dim surrou as agtset
dim out as agt
makeoneagtsetarowndown(surrou,1,universe.area.people,false)
if countagtset(surrou) > 0 then
out = getagt(surrou , int(rnd()*countagtset(surrou)))
if out.down2== true then
out.Y = 40 + rnd()
end if
end if
}

```

付 復帰エージェント 6

```

Agt_Init{
}
Agt_Step{
dim surrou as agtset
dim out as agt
makeoneagtsetarowndown(surrou,1,universe.area.people,false)
if countagtset(surrou) > 0 then
out = getagt(surrou , int(rnd()*countagtset(surrou)))
if out.down1== true then
out.X = 24 + rnd()
out.Y = 35 + rnd()*2
end if
end if
}

```

付 復帰エージェント 7

```

Agt_Init{
}
Agt_Step{
dim surrou as agtset
dim out as agt
makeoneagtsetarowndown(surrou,1,universe.area.people,false)
if countagtset(surrou) > 0 then
out = getagt(surrou , int(rnd()*countagtset(surrou)))
out.Y = 41 - rnd()
end if
}

```

```
}
```

付 復帰エージェント 8

```
Agt_Init{  
}  
Agt_Step{  
  dim surrou as agtset  
  dim out as agt  
  makeoneagtsetarountdown(surrou,1,universe.area.people,false)  
  if countagtset(surrou) > 0 then  
    out = getagt(surrou , int(rnd()*countagtset(surrou)))  
    if out.point11== true or out.point12 == true then  
      out.X = 28 - rnd()  
    end if  
  end if  
}
```

付 復帰エージェント 9

```
Agt_Init{  
}  
Agt_Step{  
  dim surrou as agtset  
  dim out as agt  
  makeoneagtsetarountdown(surrou,1,universe.area.people,false)  
  if countagtset(surrou) > 0 then  
    out = getagt(surrou , int(rnd()*countagtset(surrou)))  
    if out.point2== true then  
      out.X = 25 + rnd()  
    end if  
  end if  
}
```

付 復帰エージェント 10

```
Agt_Init{  
}  
Agt_Step{  
  dim surrou as agtset  
  dim out as agt  
  makeoneagtsetarountdown(surrou,1,universe.area.people,false)  
  if countagtset(surrou) > 0 then  
    out = getagt(surrou , int(rnd()*countagtset(surrou)))  
    if out.point1== false then  
      out.X = 28 - rnd()*2  
      out.Y = 18 - rnd()*2  
    end if  
  end if  
}
```

付 復帰エージェント 11

```
Agt_Init{  
}  
Agt_Step{  
  dim surrou as agtset
```

```

dim out as agt
makeoneagtsetarowdown(surrou,1,universe.area.people,false)
if countagtset(surrou) > 0 then
out = getagt(surrou , int(rnd()*countagtset(surrou)))
if out.point1== false then
out.X = 31 - rnd()*2
end if
end if
}

```

付 復帰エージェント 12

```

Agt_Init{
}
Agt_Step{
dim surrou as agtset
dim out as agt
makeoneagtsetarowdown(surrou,1,universe.area.people,false)
if countagtset(surrou) > 0 then
out = getagt(surrou , int(rnd()*countagtset(surrou)))
if out.point1== true then
out.Y = 22 + rnd()*1
end if
end if
}

```

付 復帰エージェント 13

```

Agt_Init{
}
Agt_Step{
dim surrou as agtset
dim out as agt
makeoneagtsetarowdown(surrou,1,universe.area.people,false)
if countagtset(surrou) > 0 then
out = getagt(surrou , int(rnd()*countagtset(surrou)))
if out.point2== true then
out.Y = 22 + rnd()*1
end if
end if
}

```

付 復帰エージェント 14

```

Agt_Init{
}
Agt_Step{
dim surrou as agtset
dim out as agt
makeoneagtsetarowdown(surrou,1,universe.area.people,false)
if countagtset(surrou) > 0 then
out = getagt(surrou , int(rnd()*countagtset(surrou)))
if out.point2== true then
out.Y = 42 - rnd()*1
end if
end if
}

```

```
}
```

付 復帰エージェント 15

```
Agt_Init{  
}  
Agt_Step{  
  dim surrou as agtset  
  dim out as agt  
  makeoneagtsetarowndown(surrou,1,universe.area.people,false)  
  if countagtset(surrou) > 0 then  
    out = getagt(surrou , int(rnd()*countagtset(surrou)))  
    if out.point11== true then  
      out.X = 24 + rnd()*1  
    end if  
  end if  
}
```

付 復帰エージェント 16

```
Agt_Init{  
}  
Agt_Step{  
  dim surrou as agtset  
  dim out as agt  
  makeoneagtsetarowndown(surrou,1,universe.area.people,false)  
  if countagtset(surrou) > 0 then  
    out = getagt(surrou , int(rnd()*countagtset(surrou)))  
    out.X = 60 - rnd()*1  
  end if  
}
```

付 復帰エージェント 17

```
Agt_Init{  
}  
Agt_Step{  
  dim surrou as agtset  
  dim out as agt  
  makeoneagtsetarowndown(surrou,1,universe.area.people,false)  
  if countagtset(surrou) > 0 then  
    out = getagt(surrou , int(rnd()*countagtset(surrou)))  
    if out.point1== true then  
      out.Y = 42 - rnd()*1  
    end if  
  end if  
}
```

付 復帰エージェント 18

```
Agt_Init{  
}  
Agt_Step{  
  dim surrou as agtset  
  dim out as agt  
  makeoneagtsetarowndown(surrou,1,universe.area.people,false)  
  if countagtset(surrou) > 0 then  
    out = getagt(surrou , int(rnd()*countagtset(surrou)))
```

```
out.X = 52 + rnd()
end if
}
```

付 復帰エージェント 19

```
Agt_Init{
}
Agt_Step{
dim surrou as agtset
dim out as agt
makeoneagtsetarounndown(surrou,1,universe.area.people,false)
if countagtset(surrou) > 0 then
out = getagt(surrou , int(rnd()*countagtset(surrou)))
out.X = 61 - rnd()
end if
}
```

付 復帰エージェント 20

```
Agt_Init{
}
Agt_Step{
dim surrou as agtset
dim out as agt
makeoneagtsetarounndown(surrou,1,universe.area.people,false)
if countagtset(surrou) > 0 then
out = getagt(surrou , int(rnd()*countagtset(surrou)))
out.X = out.X - rnd()*2
end if
}
```

付 復帰エージェント 21

```
Agt_Init{
}
Agt_Step{
dim surrou as agtset
dim out as agt
makeoneagtsetarounndown(surrou,1,universe.area.people,false)
if countagtset(surrou) > 0 then
out = getagt(surrou , int(rnd()*countagtset(surrou)))
out.X = 20 + rnd()*1
end if
}
```

付 復帰エージェント 22

```
Agt_Init{
}
Agt_Step{
dim surrou as agtset
dim out as agt
makeoneagtsetarounndown(surrou,1,universe.area.people,false)
if countagtset(surrou) > 0 then
out = getagt(surrou , int(rnd()*countagtset(surrou)))
if out.point1 == false then
out.X = 56 + rnd()*1
end if
}
```

```
end if
end if
```

付 復帰エージェント 23

```
Agt_Init{
}
Agt_Step{
dim surrou as agtset
dim out as agt
makeoneagtsetarowndown(surrou,1,universe.area.people,false)
if countagtset(surrou) > 0 then
out = getagt(surrou , int(rnd()*countagtset(surrou)))
out.Y = 32 + rnd()*1
end if
}
```

付 復帰エージェント 24(モデル 2 のみ)

```
Agt_Init{
}
Agt_Step{
dim surrou as agtset
dim out as agt
makeoneagtsetarowndown(surrou,1,universe.area.people,false)
if countagtset(surrou) > 0 then
out = getagt(surrou , int(rnd()*countagtset(surrou)))
if out.change == 1 and out.point3 == true then
my.X = my.X - 12
my.Y = 39
end if
end if
}
```