千葉工業大学

修士学位論文

進行方向の計算回数削減による SFMを用いた人流シミュレーションの高速化

2023年3月

所属専攻　　：情報科学専攻

学生番号・氏名：2281011番

片寄 颯人

指導教員　　：前川 仁孝　教授

修士論文要旨

専　攻	学生番号	氏　　名
情報科学	2281011	片寄 颯人

論文題目

進行方向の計算回数削減による
SFM を用いた人流シミュレーションの高速化

キーワード　マルチエージェントシミュレーション，人流シミュレーション

論文要旨

本論文は，SFM(Social Force Model) を用いた人流シミュレーションを高速化するために，エージェントの進行方向の計算回数を削減する手法を提案する．SFM は，時間ステップごとに各エージェントの運動方程式を解くことで，人々の流れを解析する手法である．SFM の運動方程式は，目的地に向かう力，周囲のエージェントを避ける力，障害物を避ける力の合力を用いてエージェントの移動を決定する．SFM を用いた人流シミュレーションは，解析人数や壁などの障害物数が増えるほど，解析時間が膨大になるため，高速化が求められている．周囲のエージェントや障害物を避ける力は，影響範囲内のエージェントや障害物から距離に応じた大きさの力を受けるため，距離計算を用いた影響範囲内外の判定が必要である．人流シミュレーションのなかでも避難時の解析は，机や壁などの障害物が多い傾向がある．机や壁などの固定されている障害物や目的地は，解析中に座標が変化しない特徴があり，目的地までの向かうベクトルは，エージェントの座標に応じて決定するという特徴がある．そこで，本論文では，解析領域を格子状に分割し，格子領域ごとに進行方向をあらかじめ計算することで，解析中の進行方向の計算回数を削減する．評価の結果，提案手法は，従来のセル分割法に対して，許容できる誤差の範囲で解析時間が最大〇〇倍高速することを確認した．

千葉工業大学大学院情報科学研究科

◎要旨は 600 字程度

Summary of Master's Thesis

Course	Student No.	SURNAME, Firstname
Information and Computer Science	2281011	KATAYOSE Hayato

Title

Speedup of Pedestrian Simulation by Reduction of Direction Calculations using Grid Division

Keywords Multi Agent Simulation,Pedestrian Simulation,Social Force Model

Summary

hi, there.

The contents of English summary should be written in about 200 words.

目 次

図 目 次

表 目 次

第1章
はじめに

人が多く集まるイベントなどの場所では，想定よりも多くの人が集まることで，群集事故が発生する恐れがある．群集事故は，人が将棋倒しのように倒れる群衆雪崩や○○である．群衆事故を防止するには，急に狭くなる空間を作らないことや階段や出口を広くするなどが有効である（参考文献）．人の滞留や避難時間の予測に人流シミュレーションが用いられている．[(1)]，[(2)]，[(3)]，[(4)]．

人流シミュレーションは，コンピュータ上で人を運動方程式に基づくエージェントとして解析する手法である．人流シミュレーションのなかでも歩行者の動きの再現には，ネットワークモデルやフロアフィールドモデル，SocialForceModel(SFM) などが広く用いられている．ネットワークモデルは，○○である．ネットワークモデルは，災害時における都市部の避難シミュレーションなどの大域的な解析を高速に解析が可能であるが，群集事故の防止に必要な滞留などの解析ができないことが報告されている．このため，建物内などの滞留の解析には，フロアフィールドモデルやSFM がよく用いられている.（参考文献）フロアフィールドモデルは，〜〜の手法である．フロアフィールドモデルを用いた避難シミュレーションは，一つの格子が人の大きさに制約されるため，出口付近で見られる滞留 (アーチ現象) の解析精度が低いことが報告されいている．高い解析精度が必要な場合は，解析領域を二次元の連続座標として解析する SFM がよく用いられている（参考文献）．SFM は，社会心理学的な要素と物理学的な要素で成り立つ運動方程式をエージェントごとに計算することで，人流の動きを再現する手法である．SFM の運動方程式は，目的地に向かう力，周囲のエージェントを避ける力，障害物を避ける力の合力を算出し，エージェントの速度や進行方向を計算する．SFM の運動方程式の計算は，エージェントや障害物の数の増加に応じて解析時間が膨大になることから高速化が求められている．

一般的な SFM の高速化技法として，単位時間あたりの計算回数の増加や，モデルの単純化，エージェント間距離の計算回数の削減が行われている．SFM の単位時間あたりの計算回数の増加には，GPU(Graphics Processing Unit) や MPI(Message Passing Interface) を用いた並列処理を用いることが一般的である．SFM は，エージェントごとの運動方程式の計算に並列性があるため，高い並列性を得ることがで

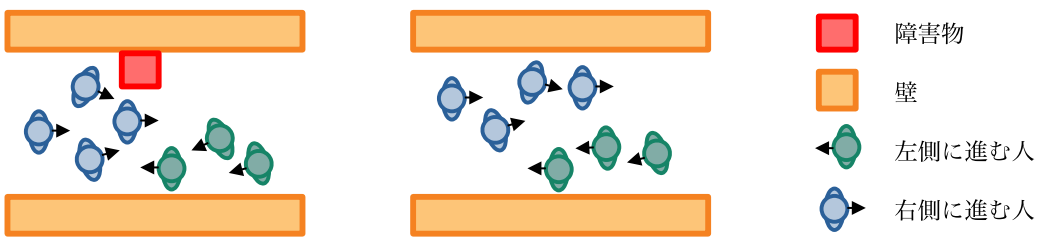
きる（参考文献 [GPU や MPI を使っている論文]）．GPU を用いた並列化手法は，エージェントごとに必要な進行方向の計算を複数スレッドで並列に計算する．モデルの単純化は，計算負荷が高い SFM の運動方程式の計算を一次元に簡易化することで，解析時間を削減する手法である（参考文献 [一次元化モデル]）．SFM の一次元化手法は，許容できる範囲の誤差で避難完了時間を解析できるが，出口付近などの滞留の再現度が低いことが報告されている（参考文献）．エージェント間距離の計算回数の削減には，影響半径の設定や，セル分割法が提案されている（参考文献）．影響半径は，エージェント間の距離が大きくなるほど相互作用力が大きくなることを利用し，相互作用力が 0 に近似可能な距離を設定する．影響半径を設定することで，影響半径外のエージェントに対する相互作用力が計算不要となる．セル分割法は，解析領域を格子状のセルに分割し，周囲のエージェントに対する影響範囲内外の判定をセル単位で実行する手法である．影響範囲内外の判定には，エージェント間距離の計算が必要となるため複数のエージェントに対する判定処理をまとめて実行することで，エージェント間距離の計算回数を削減する．建物内の避難時におけるシミュレーションは，壁や机といった障害物が多く，障害物を避ける力の計算回数が多い傾向がある．障害物を避ける力は，エージェントの座標と障害物の座標を用いて計算する．壁や机などの障害物は，解析中に座標が変化しない特徴がある．そこで，本論文は，解析領域を格子に分割し，格子領域ごとにあらかじめ進行方向を計算し，メモリ領域に保存することで，解析中の進行方向の計算回数を削減する．

以下の章では，まず，ページフォーマットを示すために，第 2 章で「あああああ」を述べる．次に，第 3 章で，本スタイルファイルで定義したコマンドについて述べる．最後に，題 6 章でまとめる．

第2章
人流シミュレーション

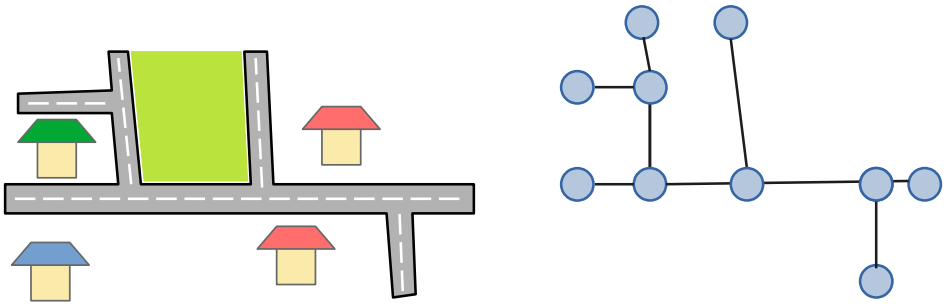
2.1　本章の概要

人流シミュレーションは，コンピュータ上で人の動きを再現する手法であり，図 2–1 に人流シミュレーションの例を示す．図 2–1 中の青色の丸は右側に進む人，緑色の丸は左側に進む人，黄色の四角は壁，青色の四角は障害物である．赤色の障害物は，自動販売機やゴミ箱などの移動が可能である設置物である．図 2–1 の例では，通路が赤色の障害物によって通路が狭くなっているため，人の滞留や混雑が起きているため，赤色の障害物を撤去することで滞留や混雑を防ぐことができる．図 2–1 のような混雑や滞留を発見するためには，実際に多くの人で実験する必要があるため，時間や費用がかかる．一方で，人流シミュレーションは，コンピュータ上で再現できることから，実際に多くの人を用いて実験するよりも，必要な時間や金額を抑えることが可能である．このように，人流シミュレーションの目的は，人の滞留や混雑が起きないように対策することである．このため，人流シミュレーションは，大規模なイベントを企画する企業や大規模や施設を設計，建築する建設業などで活用されている（参考文献）．人流シミュレーションを活用することで，事前に人の流れを予測することが可能になり，地震や火災などの有事のときに，非常灯や看板の配置，警備員の配置などを最適化できるため，適切な誘導が可能になる．人流シミュレーションを用いて人の動きを解析するためには，人の動きを再現するための歩行



(a)人の滞留が起きるとき　　(b)人が滞留が起きないとき

図 2–1 ： 人流シミュレーションの活用例



(a)解析する場所　　(b)ネットワークモデル

図 2–2 ： ネットワークモデルの例

者のモデル化 (歩行者モデル) が必要である．歩行者モデルは，求められる解析精度や解析規模に応じて使い分ける必要があるため，ネットワークモデルやセルオートマトン，SocialForceModel(SFM) などが提案されている．本章では，各歩行者モデルの使用用途や利点，欠点を用いて各手法の立ち位置について述べる．

2.2　ネットワークモデル

ネットワークモデルは，人々の移動や行動をネットワーク構造としてモデル化する手法である．図 2–2 にネットワークモデルの例を示す．図 2–2 中の (a) は解析対象であり，(b) はネットワークモデルである．図 2–2 中の (b) の青丸は交差点や道路の末端であり，ノードと呼ばれる．ネットワークモデルでは，ノード間に数十から数百人単位で動かすことで，人の動きを解析する．ネットワークモデルは，解析が高速であるが，モデルの準備に大きな作業が必要になるだけでなく，モデルの定義に知識や経験が必要になることが多い．ネットワークモデルを用いた解析では，都市間の人々の移動や，津波や地震などの災害時における都市の避難シミュレーションのような解析人数が多い解析に用いられることが多い（参考文献）ネットワークモデルで建物内の人の流れを解析する場合は，図??に示すように解析領域内にメッシュ状にノードを配置することで解析できる（参考文献）．一方で，建物内などの避難シミュレーションでは，人の流量が低下する原因や滞留の原因を調査することに用いられることが多いため，ネットワークモデルを用いた場合は，滞留などの再現ができない．このため，人の流量が低下する原因や滞留の原因を突き止めるために解析するときは，静的フロアモデルや SFM などが用いられることが一般的である．

2.3　静的フロアモデル

静的フロアモデルは，解析領域を格子状に分割し，

ーーーーーーーーーーーーー下記は下書きーーーーーーーーーーーーー

静的フロアモデルは，図 2–7 に示すようなフロアフィールドモデルの空間モデルを用いており，格子ごとに目的地までの距離を設定し，確率を用いてエージェントを移動させることで解析する手法である．図??に静的フロアフィールドモデルのイメージを示す．図??中の格子は解析領域，青丸はエージェント，青色の矢印はエージェントの移動可能な方向である．図??のように，静的フロアフィールドモデルは，図??に室内からの退出時におけるマンハッタン距離を用いた静的フロアフィールドの例を示す．図??中の格子は，各セルを示しており，セル中の数字は各セルの出口までのマンハッタン距離を示す．

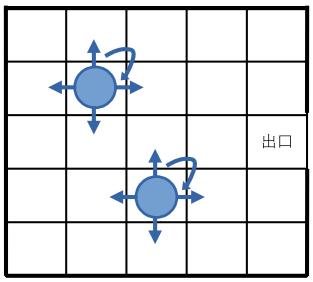


図 2–3 ： 静的フロアフィールドモデルのイメージ

6	5	4	3	2
5	4	3	2	1
4	3	2	1	出口 0
5	4	3	2	1
6	5	4	3	2

図 2–4 ： マンハッタン距離を用いた静的フロアフィールドモデルの例

静的フロアフィールドモデルは，図??に示すように，計算対象のエージェントの周囲のセルのなかから，出口までの距離が小さくなるようなセルを選択することで，出口までの解析が可能となる．静的フロアフィールドモデルの利点は，解析前に各格子の計算を事前にできるため，非常に高速な解析が可能である点である．一方で，静的フロアフィールドは，出口前に形成されるアーチ現象の再現度が低いことが知られている．図??に静的フロアフィールドモデルを用いた場合の出口前に形成されるアーチ現象の例を示す．図??中の〜〜〜である．静的フロアフィールドモデルは，図??のように，格子に一人のみ入ることができることから，動きが格子サイズに制約されるため，出口付近の再現度が低い．フロアフィールドモデルを用いた解析では，〇〇や△△，□□を用いることで，解析精度の向上が行われているが，格子サイズの成約から，精度の向上に上限がある．このため，高い解析精度が必要な場合は，SocialForceMoel(SFM) のような解析領域を連続座標で解析する手法が用いられ

図 2–5 ： アーチ現象の例

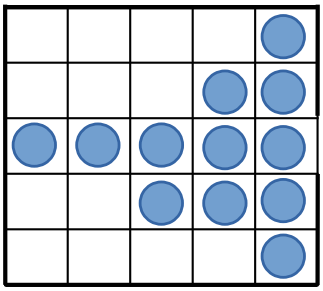


図 2–6 ： 静的フロアフィールドモデルを用いたアーチ現象

ることが多い

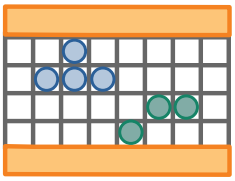


図 2–7 ： フロアフィールドモデルの例

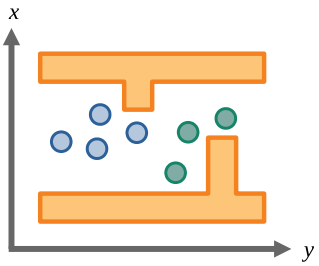


図 2–8 ： 二次元連続座標モデルの例

2.4　SocialForceModel(SFM)

SFM は，人間の社会心理学的な要素と物理的な力を結びつけた動力学モデルであり，近傍のエージェントや壁といった障害物から受ける力によってエージェントの進行方向や速度を解く．本手法は，心理的変数が組み込まれているため，災害時の避難シミュレーションによく用いられる [(5)(6)]．

SFM の解析空間は，二次元連続空間モデルが用いられる．二次元連続空間モデルは，解析領域を分割せずに (x, y) の連続した座標で解析する手法である．図 2–8 に図??の例を二次元連続座標で考えた例を示す．図 2–8 中の矢印は座標の x と y を示している．SFM は，図 2–7 のフロアフィールドモデルのように格子上のエージェント数に制限がなく，エージェントの位置を座標で考えるため，人流の再現度が高い．SFM におけるエージェント移動は，目的地へ進む力と他のエージェントから受ける力，壁などの障害物から受ける力を用いる運動方程式を用いて求める．式 (2–1) に

SFM の運動方程式を示す.

$$m_i \frac{dv_i}{dt} = m_i \frac{v_i^0(t)e_i^0(t) - v_i(t)}{t_i} + \sum_{j(\neq i)} f_{ij} + \sum_W f_{iW} \tag{2–1}$$

式 (2–1) 中の総和の記号 $\sum_{j(\neq i)} f_{ij}$ は，エージェント i 以外のすべてのエージェント j の総和をとることを意味する．同様に，$\sum_W f_{iW}$ は，すべての壁 W の総和をとることを意味する．式 (2–1) 中の m_i はエージェント i の体重，$v_i^0(t)$ はエージェントの希望速度，$e_i^0(t)$ は，目的地までの単位ベクトル，$v_i(t)$ は現在の速度ベクトル，t_i は時定数である．式 (2–1) の第一項はエージェントが目的地へ進む力，第二項は他のエージェントから受ける力 f_{ij}，第三項は壁などの障害物から受ける力 f_{iW} の合力である．f_{ij} と f_{iW} は，式 (2–2) と式 (2–3) を用いて導出する．

$$f_{ij} = \ \{A_i exp[\tfrac{r_{ij}-d_{ij}}{B_i}] + kg(r_{ij}-d_{ij})\}n_{ij} + \kappa g(r_{ij}-d_{ij})\Delta v_{ij}^t t_{ij} \tag{2–2}$$

$$f_{iW} = \{A_i exp[\frac{r_i - d_{iW}}{B_i}] + kg(r_i - d_{iW})\}n_{iW} + \kappa g(r_i - d_{iW})(v_i t_{iW})t_{iW} \tag{2–3}$$

表 2–1 に式 (2–2)，(2–3) 中の変数を示す．衝突時関数 $g(x)$ はエージェント同士や壁などに衝突したときに値をとる関数である．式 (2–4) に衝突時関数 $g(x)$ の条件式を示す.

$$g(x) = \begin{cases} 1 & (x < 0) \\ 0 & otherwise \end{cases} \tag{2–4}$$

SFM の衝突時の計算は，条件式である式 (2–4) を用いることで，衝突時のみ計算できる．SFM を用いる人流シミュレーションのフローチャートを図 2–12 に示す．図 2–12 中の目的地へ進む力の計算は，式 (2–1) 中の第一項を用いて算出する．また，他のエージェントから受ける力の計算は，式 (2–2) を用いて算出する．そして，壁などの障害物から受ける力の計算は，式 (2–3) を用いて算出する．SFM を用いた人流シミュレーションは，図 2–12 に示すように，式 (2–1) の運動方程式を積分することで，新しい時間のエージェントの位置と速度を求めることができる．

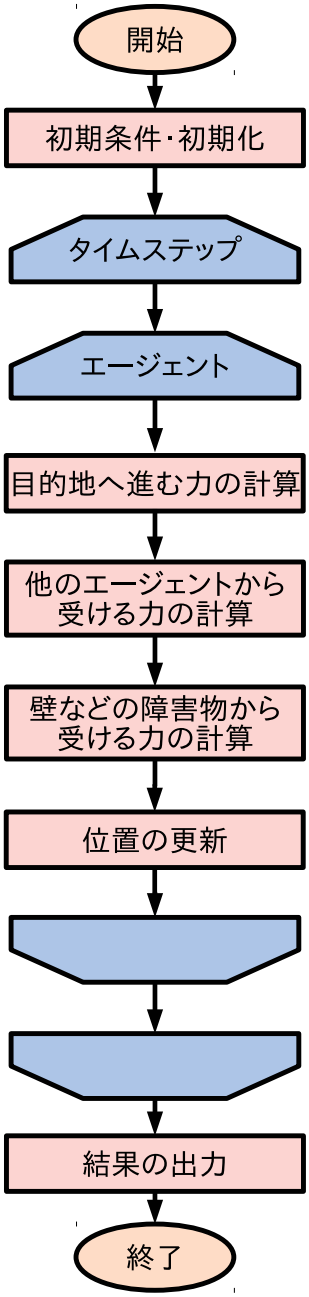


図 2–9 ： SFM を用いた人流シミュレーションのフローチャート

表 2–1 ： SFM のパラメータ

d_{ij}	エージェント間の距離
t_{ij}	エージェント i とエージェント j の衝突面の垂直ベクトル
n_{ij}	エージェント i とエージェント j の衝突面の法線ベクトル
r_i	エージェント i の体の半径
r_{ij}	エージェント i とエージェント j の体の半径の和
t_{iW}	エージェント i と壁 W の衝突面の垂直ベクトル
n_{iW}	エージェント i とエージェント W の衝突面の法線ベクトル
A_i	エージェント i のインタラクション作用
B_i	エージェント i の反発作用
k	衝突時の反発力係数
κ	衝突時の摩擦力係数
Δv_{ij}	エージェント i とエージェント j の接線速度の差
$g(x)$	衝突時関数

2.4.1　周囲のエージェントから受ける力

他のエージェントから受ける力は，解析領域全体に存在する他のエージェントから受ける．このため，SFM は，解析する人数が増えると他のエージェントから受ける力の計算時間が長くなる．他のエージェントから受ける力の計算負荷を削減するために，SFM を用いる人流シミュレーションでは，他のエージェントから受ける力を計算する範囲を限定することが多い [7][8]．他のエージェントから受ける力を計算する範囲を限定することで，遠くに存在するエージェントから受ける力を 0 に近似することができる．図 4–1 に他のエージェントから受ける力の計算範囲の例を示す．図 4–1 の赤丸は他のエージェントから受ける力を計算するエージェント，黒丸はエージェント 4 が計算するときの他のエージェント，オレンジ色の点線は他のエージェントから受ける力の範囲を示す．図 4–1 のエージェント 4 は，オレンジ色の点線内に存在するエージェント 0，1，2，6，7，10 の合計 5 人から力を受ける．本論文では，近くのエージェントから受ける力の範囲を限定する SFM を前提として述べる．近くのエージェントから受ける力の範囲は，図 4–1 のように，計算するエージェントの半径数メートルの範囲である．このため，SFM では，他のエージェントが近くのエージェントから受ける力の範囲に存在するか判定が必要である．この範囲に存在するかの判定は，エージェント i とエージェント j とのエージェント間の距離 d_{ij}

の算出が必要である．本論文では，式 (2–5) を用いてエージェント間の距離 d_{ij} を求め，他のエージェントから受ける力の範囲内であるかどうか判定する．

$$d_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \tag{2–5}$$

式 (2–5) 中の x_i と y_i はエージェント i の座標 (x_i, y_i)，x_j と y_j はエージェント j の座標 (x_j, y_j) である．エージェント i は，式 (2–5) で求めたエージェント距離 d_ij が他のエージェントから受ける力の範囲内であれば，エージェント j から式 (2–2) を用いて算出した力を受ける．他のエージェントから受ける力の範囲の半径を R としたとき，エージェント i の他のエージェント j が範囲内にいるかどうかの判定式を式 (2–6) に示す．

$$R \geq d_{ij} \tag{2–6}$$

エージェント i は，式 (2–6) の条件を満たす他のエージェント j から式 (2–2) で求まる力を受ける．

他のエージェントから受ける力の範囲を限定する SFM のフローチャートを図**??**に示す．図**??**のフローチャートでは，各エージェントに対しエージェント間の距離を計算し，範囲内であるか判定することで，他のエージェントから受ける力の範囲を限定している．SFM を用いる人流シミュレーションは，他のエージェントから受ける力の範囲を視野の範囲にすることで，視野を再現できる．

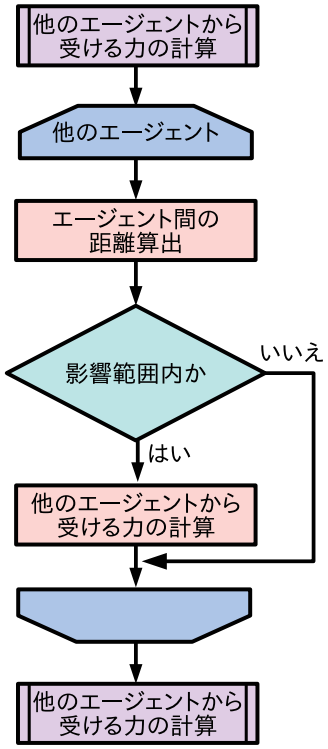


図 2–10 ： SFM における周囲のエージェントから受ける力の計算

2.4.2　周囲の壁から受ける力

周囲の壁から受ける力は，エージェントの周囲の障害物を避けるために受ける力である．SFM を用いた人流シミュレーションは，壁や机などの障害物を粒子として計算することが一般的である（参考文献）．図??に壁を粒子化した例を示す．図??中の◯◯色の丸はエージェント，△△色の丸は壁粒子である．図??のように，〜〜である．

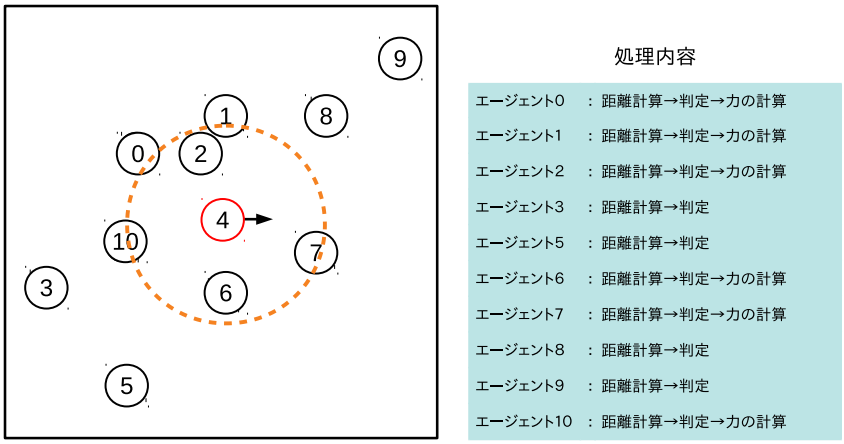


図 2–11 ： 他のエージェントから受ける力の範囲を限定するときの例

2.4.3　経由地の設定

SFM は，エージェントと目的地の間に障害物が存在するとスタック現象が発生することが報告されている（参考文献）．スタック現象は，エージェントが動かなくなる現象のことであり，障害物から受ける力と目的地に向かう力の関係により生じる．図 2–13 に SFM でスタック現象が生じる例を示す．図 2–13 中の緑色の丸はエージェント，矢印はエージェントの進行方向，オレンジ色の四角は机などの障害物である．図 2–13 中のエージェント A とエージェント B は解析領域右上の出口に向かうため，エージェント A が机などの障害物に向かって進む．この場合は，エージェント A が机に向かって進み続けることや机の上を歩くなどの想定しない動きをすることがある．このため，障害物が多く存在するような解析では，出口 (目的地) だけでなく，目的地までの道のりを示す経由地を設定することで，エージェントのスタック現象や想定しない動きを防ぐことができる．図 2–14 に経由地を設定する例を示す．図 2–14 中の緑色の丸はエージェント，四角は障害物，青色の四角は経由地，赤色の四角は目的地を示す．図 2–14 の例では，エージェント A は，経由地を通ったあとに目的地に進むため，図 2–13 のように机に進むことが防げる．教室などの障害物多い解析では，図 2–15 に示すように，複数の経由地を設定する必要がある．図 2–15 の例では，エージェントは一番近くの経由地から目的地までの道のりを辿る．目的地までの道のりを決定する手法は，ダイクストラ法などのグラフ理論で用いられる手法が使われることが多い（参考文献）．ダイクストラ法は，〇〇である．

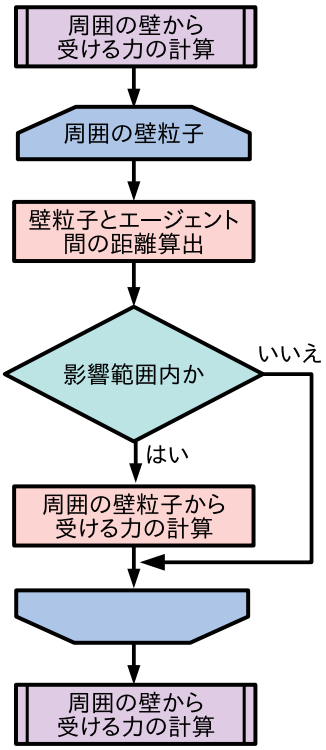


図 2–12 ： SFM における周囲の壁から受ける力の計算

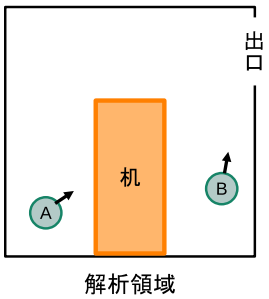


図 2–13 ： SFM でスタック現象が起きる例

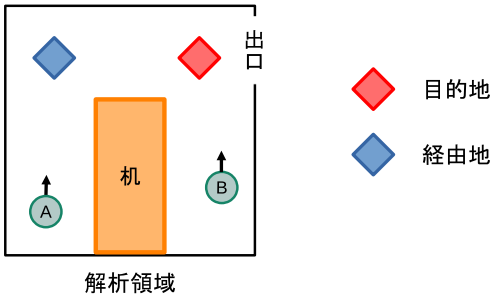


図 2–14 ： 経由地を設定する SFM の例

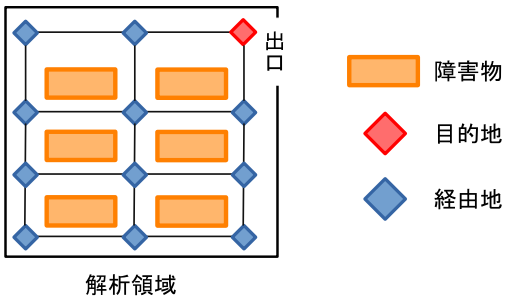


図 2–15 ： 経由地を設定する SFM の例

2.5　本章のまとめ

第3章
一般的なSFMの高速技法

3.1　本章の概要

SFM を用いた人流シミュレーションは，解析人数が多くなるほど計算負荷が膨大になるため，解析に時間がかかる．SFM の解析時間を削減するために，モデルの単純化（参考文献）やエージェント間距離の計算回数削減手法（参考文献），単位時間あたりの計算回数の増加手法 （参考文献），経路選択時の判定回数の削減手法 (参考文献) などが提案されている．本章では，SFM の各高速化手法について述べる．

3.2　モデルの簡易化

SFM の簡易化手法は，SFM の計算負荷を削減するために，エージェント同士や壁や机などの障害物から受ける力，進行方向を単純化する手法である．SFM の簡易化手法の一つに一次元歩行者モデルがある（参考文献）．一次元歩行者モデルは，エージェントの動きを x や y のみにする手法である．図 3–1 に避難シミュレーション時の SFM と一次元歩行者モデルの例を示す．図 3–1 中の (a) は SFM などの二次元連続空間モデルを示し，(b) は一次元連続歩行者モデルの例を示す．一次元歩行者モデルは，図 3–1 のように，エージェントの動きを算出する式を一次元に変更することで，計算負荷を削減できる．本手法は，人の流れ (流量) を解析する場合では，高速かつ許容できる誤差の範囲で解析できることが報告されている（参考文献）．一方で，一次元でエージェントの動きを再現するため，人の押し合いや図 3–6 のようなアーチ現象などを再現できない．このため，人の押し合いやアーチ現象を再現したい場合には，一次元歩行者モデルなどのモデルを簡易化しない高速化技法を用いることが望ましい．

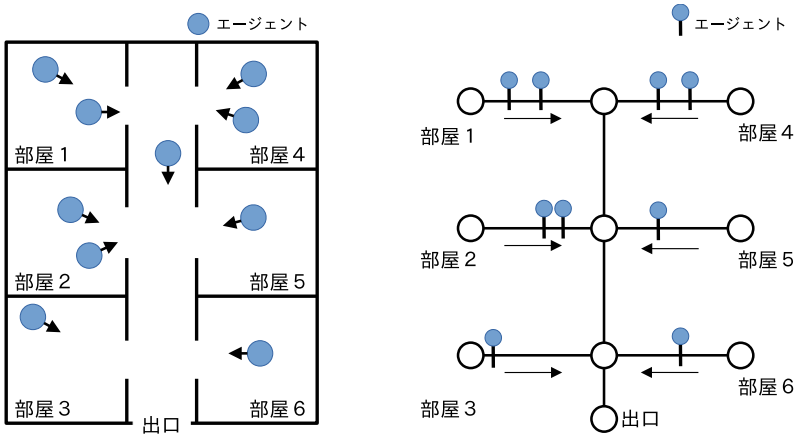


図 3–1 ： 一次元モデルの例

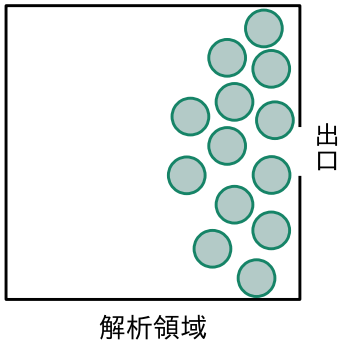


図 3–2 ： アーチ現象の例

3.3　エージェント間の距離の計算回数削減

SFM は，解析人数が増加するほど周囲のエージェントから受ける力の計算に必要な周囲のエージェントが影響範囲内外かの判定の回数が増加する．周囲のエージェントが影響範囲内外かの判定は，ルートなどを用いてエージェント間距離 d_{ij} を求める必要があるため，特にエージェント間距離 d_{ij} の計算に時間がかかる．このため，SFM を用いた人流シミュレーションの解析時間を削減するためには，エージェント間距離 d_{ij} の計算回数を削減することが有効である．エージェント間距離の計算回数削減手法にセル分割法や視野パラメータを用いた削減手法がある．

3.3.1　セル分割法

セル分割法は，水や空気などの動きを解析できる MPS 法 [9] や銀河系などの圧縮性流体に用いられる SPH 法 [10] などの粒子法によく用いられている．粒子法は，近傍の粒子と相互作用する力を計算し，粒子の行動を決定する．このため，セル分割法は，粒子法と同じように近傍のエージェントとの相互作用力を計算する SFM に対しても用いられる．本手法は，解析領域を格子上のセルに分割し，計算するエージェントの存在するセルと近傍のセルに存在するエージェントに対して他のエージェントから受ける力の範囲であるか判定し，範囲内であれば他のエージェントから受ける力を計算する手法である．図 3–3 にセル分割法を用いる SFM の例を示す．図 3–3 中の赤丸は他のエージェントから受ける力の計算をするエージェント，黒丸は他のエージェント，四角は解析領域を格子状に分割したセルである．この例では，エージェント 4 の行動を更新する際に青色のセル内に存在するエージェントのみを参照するため，エージェント番号 3，5，9 の計算を削減できる．このとき，視野を用いる SFM は，速度計算するエージェントの進行方向前方に存在するエージェント情報を用いて計算するため，エージェントの進行方向後方のセルに存在するエージェント情報は不要になる，このため，視野を用いる SFM では，視野を考慮し，参照するセルを視野範囲に近づけることで，計算回数を減らすことができる．

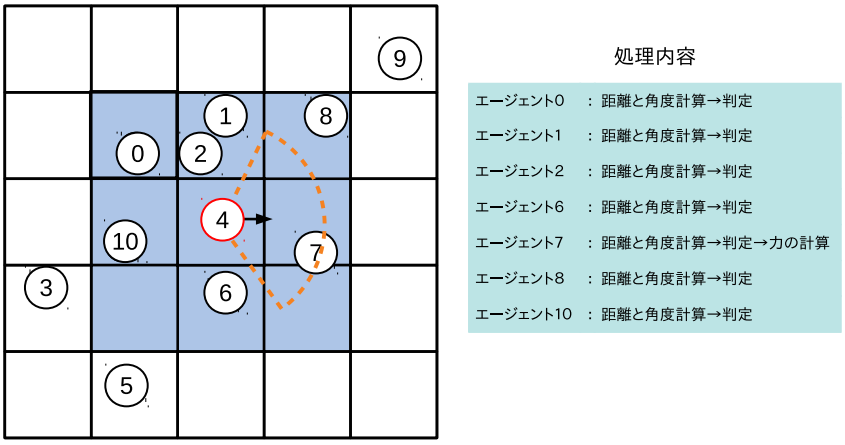


図 3–3 ： セル分割法を用いた例

3.3.2　視野パラメータを用いた削減手法

視野パラメータを用いた SFM では,

3.4　並列性の抽出

3.4.1　エージェントごとの並列性を用いた手法

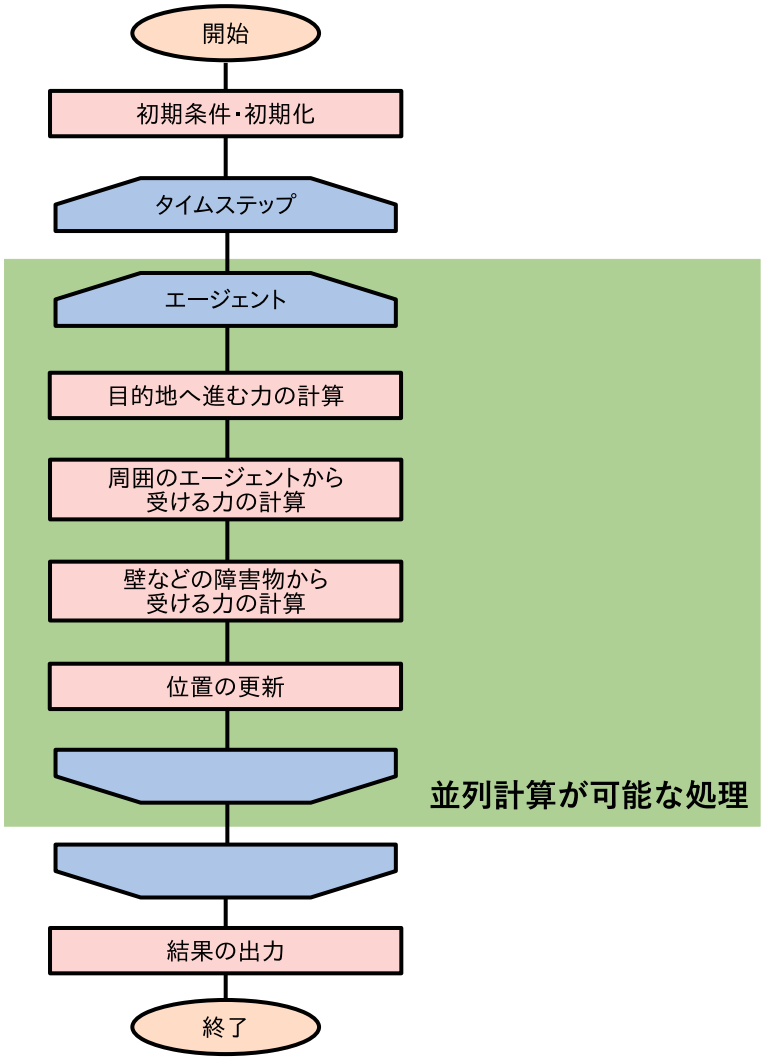


図 3–4 ： SFM の並列化可能な処理

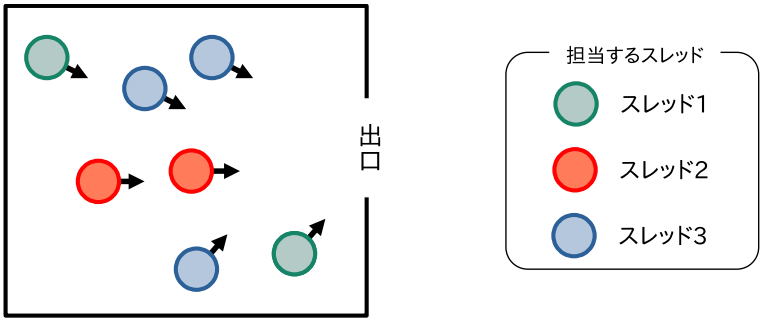


図 3–5 ： 3 スレッドでの並列化の例

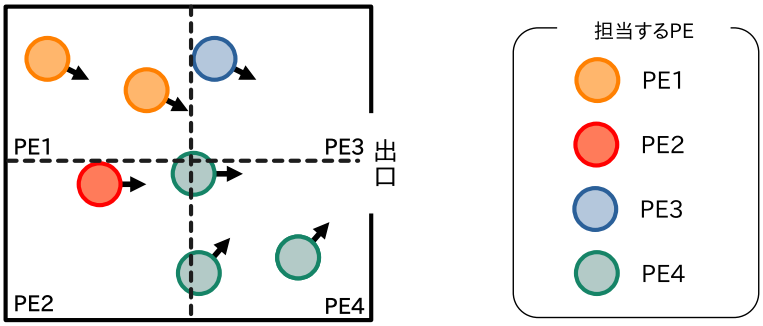


図 3–6 ： 領域分割の例

3.4.2　解析領域ごとの並列性を用いた手法

aaAa

3.5　経路選択時の判定回数削減

aa

3.5.1　経路選択の単純化

aa

3.5.2　経路選択手法の〜〜

aa

3.6　本章のまとめ

本章についてまとめる予定である.

第4章
エージェント間距離の削減手法

4.1　本章の概要

SFM の運動方程式では，エージェント間距離が大きいほどエージェント間に働く相互作用力が小さくなる．このため，SFM では，影響半径 R_c を用いてエージェント間距離の計算回数を削減するのが一般的である．図 4–1 に，エージェント 4 の影響半径の例を示す． 図 4–1 中の◯はエージェントであり，オレンジ色の点線で囲まれた領域がエージェント 4 の影響範囲である．影響半径 R_c は影響範囲の半径であり，影響範囲外から受ける力を 0 とすることで，運動方程式を計算する際に演算を省略することができる．一方，視野を用いた SFM では，視野範囲外の相互作用力を 0 とする．図 4–1 の例で，エージェント 4 が図中矢印の方向に移動する際には，エージェント 4 の視野は，図中の緑色の領域のように設定され，視野 $\subseteq$ 影響範囲のような関係となる．

SFM で用いられるエージェント間距離の計算回数削減手法は，影響範囲が円形であることを前提としており，影響範囲が視野のように扇形の場合を想定していない．例えば，図 4–1 のエージェント配置では，半径 R_c 内の白い領域に存在する 5 つのエージェントが視野外にあるにも関わらず，これらのエージェントに対するエージェント間距離の計算が必要となる．人の視野角 θ は π 以下であるため，視野を用いた

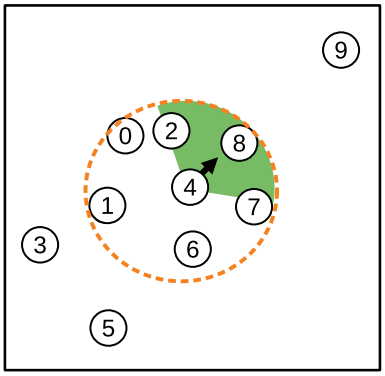


図 4–1 ： 影響範囲の例

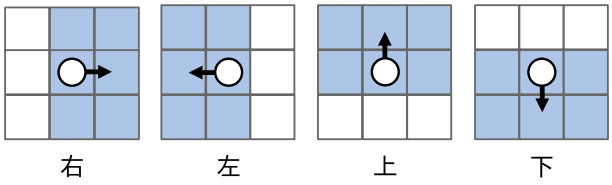


図 4–2 ： 進行方向ごとの近似領域

SFM に一般的な SFM のエージェント間距離の計算回数削減手法を用いると，影響範囲を円形に絞り込みをした上で，扇形に絞り込みを行う操作が必要となる．このため，視野範囲外となる半径 R_c 内のエージェントが多く存在するほど，相互作用力を 0 として計算するエージェントに対するエージェント間距離の計算回数が増える．そこで，本論文では，視野形状である扇形に近似した領域を設定し，近似領域外のエージェントに対するエージェント間距離の計算回数を削減することで，視野を用いた SFM を高速化する．

4.2　近似領域の選択方法

提案手法では，近似領域の算出に必要な計算時間を最小限に抑えるために，長方形で視野範囲を近似する．これに合わせて，視野範囲はエージェントの進行方向前方に存在するため，エージェントの進行方向を長方形の辺数に合わせて上下左右の 4 パターンに分類し，エージェントの進行方向に応じた近似領域を設定する．図 4–2 に上下左右の 4 パターンの近似領域を示す．図 4–2 中の◯はエージェント，矢印は進行方向，格子はセル分割法のセル，青い四角は，各進行方向ごとの近似領域である．図 4–2 の近似領域の選択方法は，エージェントの座標や進行方向などといった複数の選択方法が考えられる．エージェントの進行方向の分類方法によりシミュレーション時間が変化する可能性があるため，本論文では，既存手法であるセル分割法を含む 6 パターン実装し，その有効性を評価する．提案手法であるパターン 2〜6 は，いずれの手法もパターン 1 のセル分割法の考え方を基にしており，時間ステップごとに影響範囲の近似領域を設定した上で，近似領域内で影響範囲内となるエージェントを相互作用力を計算する対象として設定する．

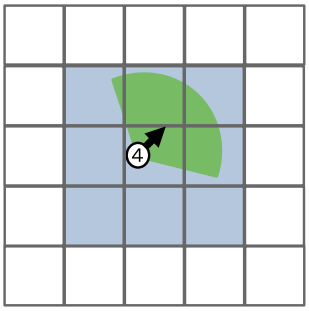


図 4–3 ： パターン 1 を用いたエージェント 4 の近似領域

表 4–1 ： パターン 2，3 の進行方向判定条件

	右	左	上	下
条件 1	$\frac{1}{\sqrt{2}} < e_x \leq 1$	$-1 \leq e_x < \frac{-1}{\sqrt{2}}$	$\frac{-1}{\sqrt{2}} < e_x < \frac{1}{\sqrt{2}}$	$\frac{-1}{2} < e_x < \frac{1}{2}$
条件 2	$\frac{-1}{2} < e_y < \frac{1}{2}$	$\frac{-1}{2} < e_y < \frac{1}{2}$	$\frac{1}{\sqrt{2}} < e_y \leq 1$	$-1 \leq e_y < \frac{-1}{\sqrt{2}}$

4.2.1　パターン 1（セル分割法）

パターン 1 は，一般的なセル分割法 [11]，[12] によるエージェント間距離の計算回数を削減する．図 4–3 に図 4–1 中のエージェント 4 にパターン 1 を用いて近似領域を設定した例を示す．セル分割法は，図 4–3 のように解析領域を格子状に分割し，影響半径 R_c の円内に重なるセルを近似領域とする手法である．本手法は，分割するセルのサイズを影響半径と同じ距離に設定することで，近似領域をエージェントの近傍 9 セルに限定することができる．本例では，エージェント 4 の近傍 9 セルは黄色のセルであり，それ以外のセルに属するエージェント 3，5，9 に対するエージェント間距離の計算を削減できる．なお，セル分割法は相互作用力を計算する範囲を円形で想定した手法であるため，近似領域が正方形となることから，パターン 1 の実装では進行方向の推測は行わない．また，本論文では，セル分割法のなかでもメモリ使用量を抑えることができると知られている連結リスト法 [13]，[14] を用いる．

4.2.2　パターン 2

パターン 2 は，エージェントの進行方向を表す単位ベクトル e_i を参照することで，パターン 1 の近似領域を視野範囲に近づける手法である．表 4–1 にパターン 2 の近似領域を選択するための条件を示す．表 4–1 中の条件 1 と条件 2 の両方を満たす進行方向から近似領域を選択する．本手法は，表 4–1 の条件に基づいて単位ベクトル

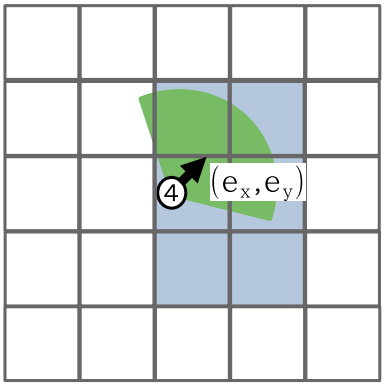


図 4–4 ： パターン 2 を用いたエージェント 4 の近似領域

e_i が示す進行方向を上下左右を $\frac{1}{2}\pi$ の均等な角度で分割し，セル分割法の近似領域から図 4–2 のように進行方向の反対側にある 3 セルを除外する．図 4–4 に図 4–1 の例におけるエージェント 4 の近似領域をパターン 2 を用いて選択した例を示す．図 4–4 の例では，エージェントの進行方向は右と判定され，図 4–2 に示す右方向の青いセルを近似領域に設定する．このため，白色のセルに存在するエージェントに対する距離計算を削減できる．パターン 2 は，進行方向を必ず上下左右のいずれかに設定するため，常に近似領域が 6 セルとなり，セル分割法の 9 セルに比べて近似領域の面積を $\frac{2}{3}$ に削減できる．一方で，図 4–4 のように視野範囲全体が近似領域内に含まれる保障がないため，本来相互作用力の計算が必要なエージェントに対する計算を削減し，誤差が発生する可能性がある．

4.2.3　パターン 3

パターン 3 は，パターン 2 で発生する誤差を防ぐために，近似領域外に視野範囲が存在しないかを判定する処理を追加し，セル分割法と同じ精度を保つ手法である [?]．本手法は，表 4–1 を用いてパターン 2 の近似領域を導出したのち，視野の座標 (L_x, L_y)，(R_x, R_y) が近似領域内のセルであればパターン 2 の近似領域を用いて相互作用力を計算し，そうでなければパターン 1 の近似領域を用いて相互作用力を計算する．図 4–5 にパターン 3 を用いて近似領域を削減できる例を示す．図 4–5 中の緑色の視野領域上に存在する黒点は，視野の座標である (L_x, L_y) と (R_x, R_y) を示す．図 4–5 の例では，表 4–1 中の進行方向が右の条件を満たし，視野の座標 (L_x, L_y) と (R_x, R_y) が青色の近似領域内であるため，図 4–2 に示す右方向の青いセルが近似領域になり，セル分割法よりも近似領域を削減できる．一方で，図 4–1 中のエージェ

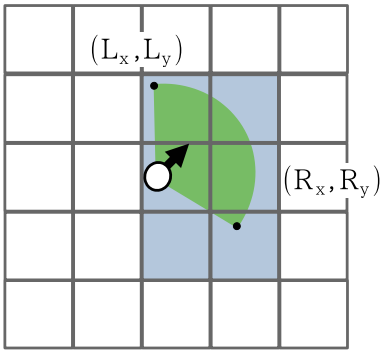


図 4–5 ： パターン 3 を用いた近似領域の削減例

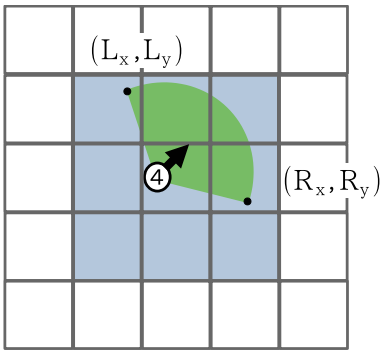


図 4–6 ： パターン 3 を用いたエージェント 4 の近似領域

ント 4 にパターン 3 の条件を適用した場合は，図 4–6 のように，視野の座標 (L_x, L_y) が図 4–2 の青いセル以外に存在するため，近似領域がパターン 1 と同様に近傍 9 セルとなる．エージェントの座標は時間ステップごとに変化するため，パターン 3 を用いると，進行方向が変化しないエージェントに対してもすべての時間ステップでパターン 1 よりもエージェント間距離の計算を削減できるとは限らない．一方で，パターン 3 は，パターン 1 のセル分割法と同じ精度のシミュレーション結果を得ることができる．

4.2.4　パターン 4

パターン 4 は，視野の左右両端の座標 (L_x, L_y)，(R_x, R_y) を用いて近似領域を設定する手法である．視野範囲は進行方向の前方に存在するため，視野の左右両端の座標はエージェント座標から見て必ず進行方向側となる．この特性を利用し，パターン 4 では，表 4–2 に示すように，エージェント座標 (A_x, A_y) と視野座標 (R_x, R_y)，

表 4–2 ： パターン 4 の進行方向判定条件

	右	左	上	下
条件 1	$R_x \geq A_x$	$R_x < A_x$	$R_y \geq A_y$	$R_y < A_y$
条件 2	$L_x \geq A_x$	$L_x < A_x$	$L_y \geq A_y$	$L_y < A_y$

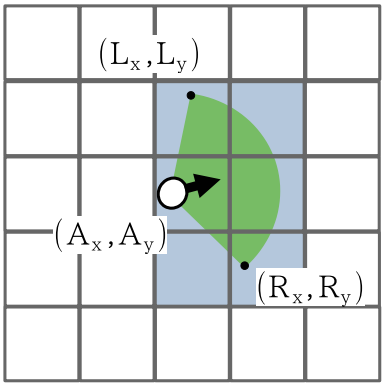


図 4–7 ： パターン 4 を用いた近似領域の削減例

(L_x, L_y) の大小関係を用いて進行方向を判定する．本手法の表 4–2 に満たない場合は，パターン 1 と同様に近傍 9 セルを近似領域にする．図 4–7 にパターン 4 を用いた近傍領域の削減例を示す．図 4–7 の例では，エージェント座標 (A_x, A_y) と視野座標 (R_x, R_y)，(L_x, L_y) が表 4–2 中の条件である $R_x \geq A_x$ および $L_x \geq A_x$ が成り立つため，進行方向が右と判定され，図 4–2 の右方向の青いセルが近似領域となる．図 4–8 に図 4–1 中のエージェント 4 にパターン 4 を用いて近似領域を決定した例を示す．図 4–8 の例は，表 4–2 中の判定条件を満たす進行方向がないため，パターン 1 と同様に近傍 9 セルを近傍領域とする．

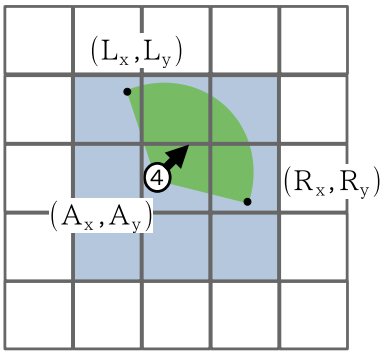


図 4–8 ： パターン 4 を用いたエージェント 4 の近似領域

表 4–3 ： パターン 5 の進行方向判定条件

	右	左	上	下
条件 1	$R_x \geq x_1$	$R_x < x_2$	$R_y \geq y_1$	$R_y < y_2$
条件 2	$L_x \geq x_1$	$L_x < x_2$	$L_y \geq y_1$	$L_y < y_2$

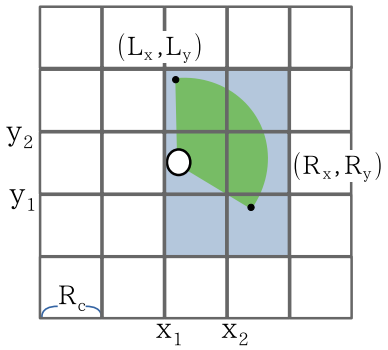


図 4–9 ： パターン 5 を用いた近似領域の削減例

4.2.5　パターン 5

パターン 5 は，セル分割法のセル座標と視野範囲の左右両端の座標 (L_x, L_y)，(R_x, R_y) を用いて近似領域を設定する手法である．本手法は，運動方程式を算出するエージェントが所属するセルの左上座標 (x_1, y_2) および右下座標 (x_2, y_1) を用いて，図 4–2 の 4 パターンの水色の領域内に視野が収まっているかを判定する．表**??**にセルの左上座標 (x_1, y_2) および右下座標 (x_2, y_1) を用いたパターン 5 の判定条件を示す．また，図 4–9 にパターン 5 を用いた場合の近似領域を削減する例を示す．図 4–9 の例では，表**??**中の $R_x \geq x_1$ および $L_x \geq x_1$ が成り立つため，進行方向は右と分類される．このため，図 4–9 の例では，近似領域は図中の青いセルとなる．図 4–10 に図 4–1 中のエージェント 4 にパターン 5 を用いて近似領域を選択した例を示す．図 4–10 の例では，$R_y \geq y_1$ および $L_y \geq y_2$ が成り立つため，進行方向は上と分類される．本手法は，エージェントのセル上の位置に応じて進行方向を分類するため，ベクトル e_i の表す進行方向が同じエージェントでもエージェント座標に応じて異なる方向に分類される可能性がある．本手法も表 4–3 のいずれの条件にも当てはまらない場合は，パターン 1（セル分割法）と同様に近傍 9 セルを近似領域とする．

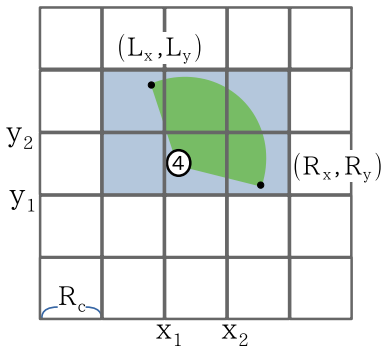


図 4–10 ： パターン 5 を用いたエージェント 4 の近似領域

表 4–4 ： パターン 6 の進行方向判定条件

右	左	上	下
$\cos(\frac{1}{2}\theta_{view}) \leq e_y$	$e_y \leq -\cos(\frac{1}{2}\theta_{view})$	$\sin(\frac{1}{2}(\pi - \theta_{view})) \leq e_x$	$e_x \leq \sin(\frac{1}{2}(\pi - \theta_{view}))$

4.2.6　パターン 6

パターン 6 は，パターン 2 で設定した上下左右の角度範囲を，シミュレーション誤差の出ない範囲で設定する手法である．パターン 6 では，図 4–2 中の水色の領域である近似領域内に視野範囲がすべて収まる進行方向を静的に計算し，エージェントの進行方向を表すベクトル e_i の閾値を定める．視野角を θ_{view} とおくと，エージェント座標がどの位置であっても上方向と判定される進行方向の角度は $\frac{1}{2}(\pi - \theta_{view})$ から $\frac{1}{2}(\pi + \theta_{view})$ の間であり，これを用いて進行方向ベクトル $e_i = (e_x, e_y)$ の範囲が $\sin(\frac{1}{2}(\pi - \theta_{view})) \leq e_x$ という条件を設定できる．表**??**にパターン 6 の進行方向判定条件を示す．パターン 6 は，パターン 3〜6 のように視野座標 (R_x, R_y) および (L_x, L_y) を算出する必要がないため，他の分類条件よりも高速に進行方向を分類することができる．一方で，進行方向が特定の方向である場合は，エージェント座標に関係なくパターン 1（セル分割法）と同じ近似領域を設定するため，エージェント間距離の計算回数は，パターン 1 に次いで多くなると考えられる．

表 4–5 ： 測定条件

A_i	2000N
B_i	0.08m
k	$1.2 \times 10^5 kgs^{-2}$
κ	$2.4 \times 10^5 kgm^{-1}s^{-2}$
v_i^0	1.4m/s
m_i	80kg
τ_i	0.5
r_i	0.25m
視野角	$\frac{2}{3}\pi$
視野距離	20m
タイムステップ数	25000

4.3　評価

視野を用いた SFM に対するエージェント間距離の計算回数削減手法の有効性を確認するために，既存手法であるセル分割法を用いたパターン 1 および提案手法であるパターン 2〜6 を用いて人流シミュレーションを行う．評価環境は，CPU が Intel Xeon CPU E5-2687w v2，メモリが 64GB，OS が Linux 4.12.9 であり，プログラムのコンパイルには gcc 7.2.0 で-O3 オプションを設定する．また，本評価では，視野を用いた SFM で期待される押し合い圧し合いを行う群衆の行動を再現するために，エージェントの初期配置および目的地を図 4–11 のように設定し，表 4–5 を用いて運動方程式を計算する．表 4–5 のパラメータは文献 [15] を参考に設定したものであり，これを踏まえてセル分割法のセルサイズは視野範囲に合わせて 20m とする．本測定では，図 4–11 の緑色の範囲内にエージェントをランダムに生成し，各エージェントの目的地を解析領域上で初期配置と点対称となる座標に設定することで，解析領域の中央でエージェントが密集した状態を作る．

4.3.1　エージェント間距離の計算回数

表 4–6 に，エージェント間距離の計算回数および削減率を示す．表中の括弧内の数値は，削減率であり，パターン n のエージェント間距離の計算回数 C_n とパターン 1（セル分割法）のエージェント間距離の計算回数 C_1 より式（4–1）のように求

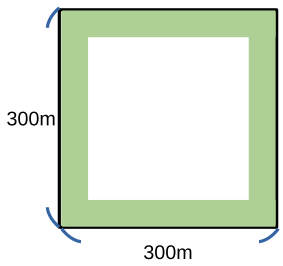


図 4–11 ： エージェントの初期配置

表 4–6 ： エージェント間距離の計算回数 [10^{10} 回]

人数	パターン 1	パターン 2	パターン 3	パターン 4	パターン 5	パターン 6
3000	5.1	**3.9** (**24.5**%)	4.0 (22.9%)	4.4 (15.3%)	4.1 (20.7%)	4.4 (15.2%)
5000	14.4	**10.9** (**23.8**%)	11.1 (22.6%)	12.2 (15.2%)	11.4 (20.5%)	12.2 (15.1%)
7500	33.1	**25.2** (**23.9**%)	25.8 (22.2%)	28.3 (14.6%)	26.7 (19.4%)	28.3 (14.6%)

める．

$$\text{削減率 [\%]} = (1 - \frac{C_n}{C_1}) \times 100 \tag{4–1}$$

表 4–6 より，パターン 2〜6 はセル分割法よりもエージェント間距離の計算回数が少なく，最も高い削減率が得られたのはパターン 2 であることが確認できる．パターン 2 は近似領域が視野範囲全体を含まない可能性のある手法であるが，セル分割法と同じ解析精度の手法に限定しても，パターン 2 と同じ手順で進行方向を判定したパターン 3 が最も高い削減率である．パターン 3 で高い削減率が得られたのは，エージェントの初期配置を図 4–11 のように設定したことにより，表??で削減可能な角度の範囲の広さが削減率に強く影響したためであると考えられる．本測定条件では，初期のエージェントの進行方向がすべて異なり，時間経過による進行方向の変化も少ない．実問題のシミュレーションでは，エージェントが同一の目的地を目指すものが多く，進行方向が一定の方向に偏ることが多いため，解析対象に応じて表 1 から適切なパターンを選択する必要があると考えられる．横方向や縦方向に進むエージェントが多い問題では，パターン 3 の近似領域を用いることで，より多くのエー

表 4–7 ： 解析時間 [s]

人数	パターン 1	パターン 2	パターン 3	パターン 4	パターン 5	パターン 6
3000	2636	**2123**	2140	2307	2184	2292
5000	7435	**5941**	6016	6463	6162	6453
7500	17198	**13730**	13985	15048	14931	15036

ジェント間距離の計算回数が削減できる．一方で，斜め方向に進むエージェントが多い問題では，パターン 3 を用いた場合に近似領域を削減できないため，進行方向の角度に応じで使用するパターンを決定することが有効であると考えられる．同様に，削減率が最も低いパターン 6 は，表??の条件を満たさない角度が最も広いため，パターン 1（セル分割法）と同じ近似領域を用いることが多く，高い削減率が得られなかったと考えられる．

また，各パターンの削減率は，エージェント数によらず一定であることが分かる．これは，パターン 2 が周囲のエージェント情報を参照せずに近似領域を設定するためである．エージェント数や，エージェントの密度が変化してもエージェントごとの近似領域の面積は変わらない．加えて，解析領域中央付近でエージェントが密集するように条件を設定しており，エージェント間距離の計算はほとんどが密集状態で実行されるものである．一方，密集領域以外では，近似領域から除外した領域のエージェント数が少なくなりやすい．これに伴い，すべての視野に対して近傍領域の面積の約 3 割を削減するパターン 2 で表 4–6 の削減率が 3 割未満となり，近傍領域の面積の削減率とエージェント間距離の演算回数の削減率に差が生じた．

4.3.2　シミュレーションの実行時間の測定

第 4.3.1 節より，提案手法であるパターン 2〜6 を用いることで，パターン 1（セル分割法）よりもエージェント間距離の計算回数が削減できることが確認できた．一方で，提案手法はパターン 1 が必要としない進行方向を求める処理を実行するため，進行方向を求める処理の分だけ実行時間は増加する．このため，進行方向を求めるために必要な時間を考慮してもエージェント間距離の計算回数削減による高速化が有効であることを確認する．表 4–7 にシミュレーションの実行時間を，図 4–12 に式

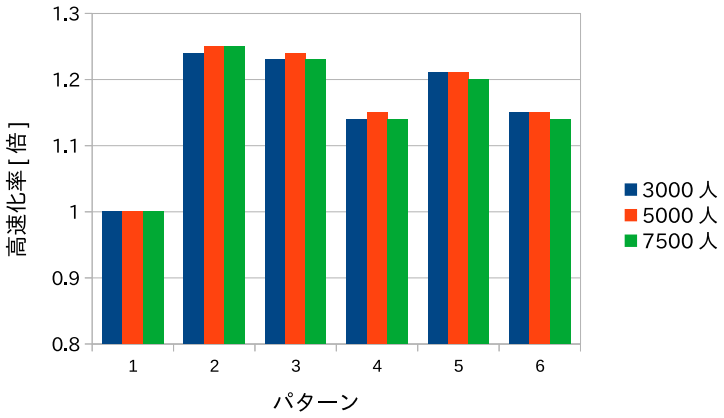


図 4–12 ： パターン 1（セル分割法）に対する高速化率

(4–2) より算出した高速化率を示す.

$$\text{高速化率 [倍]} = \frac{\text{パターン 1 の実行時間 [s]}}{\text{パターン } n \text{ の実行時間 [s]}} \tag{4–2}$$

表 4–7，図 4–12 より，提案手法であるパターン 2〜6 は，すべてのパターンにおいてパターン 1（セル分割法）よりも 1.14 倍から 1.25 倍高速であり，従来手法よりも解析時間が 1.3 割から 2.0 割削減できることが確認できる．また，すべてのパターンにおいて削減率に応じた高速化率が得られており，進行方向を求める処理を追加したことによる処理時間の時間増加は小さいといえる．これを踏まえると，パターン 2 とパターン 3 の高速化率の差は，本来計算が必要なエージェント間距離の計算の有無によるものであると考えられる．つまり，パターン 3 はパターン 2 の処理に加えて視野の座標を求め，厳密に視野範囲を予測しているが，追加の処理による実行時間の増加は小さいといえる．

4.3.3　シミュレーション精度の測定

提案手法のうち最も高速な手法はパターン 2 であるが，パターン 2 は他の手法と異なり視野範囲の一部が近似領域外となることを許容する．つまり，提案手法のうちパターン 2 のみ，パターン 1（セル分割法）のシミュレーションと比べてシミュレーション誤差が生じる手法となる．このため，パターン 1 のシミュレーション結果とパターン 2 のシミュレーション結果を比較し，シミュレーション精度を確かめる．誤差は，パターン 1 とパターン 2 が算出した時間ステップごとに各エージェン

トの座標を出力し，2 手法の同時刻・同エージェント同士によるエージェント間距離の最大値とした．本測定条件下のパターン 2 の誤差は，エージェント数によらず約 3.2cm であった．これは，解析領域全体や人の大きさを踏まえると小さな数値であり，シミュレーション結果として十分に実用的であると考えられる．誤差を小さな数値で抑えることができたのは，SFM ではエージェントが目的地に向かう特性から，時間ステップごとの位置座標に応じて進行方向が更新され，細かな誤差が発生してもそれを蓄積しにくいためであると考えられる．加えて，パターン 2 が相互作用力の計算を省略したのは視界の隅にあたる領域であり，パターン 2 を用いても人の行動を自然に再現できると考えられる．

4.4　本章のまとめ

本論文では，視野を用いた SFM を高速化するために，エージェント間距離の計算回数を削減する手法を提案し，その有効性を評価した．評価の結果，提案手法は，エージェントが交差するように移動する問題において，十分に許容可能な誤差の範囲で約 1.25 倍の高速化率が得られることを確認した．提案手法を用いた人流シミュレーションは，従来手法よりも解析時間を最大 2 割削減できるため，解析に必要な電力コストを解析時間に応じて削減できる可能性があり，これは建物の設計に必要な開発コストの削減にもつながると考えられる．また，本測定の提案手法で発生した誤差より大きい誤差を許容することができる場合，近似領域の視野範囲に対する近似制度を高めることが可能であり，エージェント間距離の計算回数の削減量を増やすことができると期待できる．このため，精度低下を許容できる問題に対しては，さらなる高速化が見込めるといえる．さらに，提案手法を用いた人流シミュレーションは，エージェントごとに独立して計算できる並列性があるため，CPU や GPU を用いて並列処理することで高速に解析できると考えられる．特に GPU を用いた人流シミュレーションは，GPU 上でエージェントごとに視野領域の内外判定を計算し，進行方向を計算することが一般的である [7]，[8]．GPU を用いた解析は，条件分岐により，実行効率が落ちることが知られている．このため，提案手法を GPU 上で実装する場合は，同じ処理をまとめて計算するなどの条件分岐を削減する工夫が必要であると考えられる．

第5章
格子分割による進行方向ベクトル計算の削減手法

5.1 本章の概要

SFM の運動方程式は，エージェントの位置が変わるたびに，目的地までのベクトルを表す e や周囲のエージェントを避ける力 f_{ij}，障害物を避ける力 f_{iW} の再計算が必要となる．周囲のエージェントを避ける力 f_{ij} の計算は，時間ステップごとにすべてのエージェントの位置が変化するため，解析中のみに計算が可能である．一方，ベクトル e や障害物を避ける力 f_{iW} は，エージェントの位置に応じて決定し，壁や机などの障害物の座標が固定であることから，解析前に計算が可能である．

そこで，本論文では，格子状に分割した領域ごとにベクトル e と障害物を避ける力 f_{iW} をあらかじめ計算することで，解析中の進行方向の再計算を削減する．

5.2 格子分割を用いた進行方向計算手法

本手法は，解析領域を格子状に分割し，格子ごとに進行方向ベクトル e や障害物を避ける力 f_{iW} をあらかじめ計算する．進行方向ベクトル e は，式 (2–1) に示す SFM の運動方程式から $v_i^0(t)$ が係数である．一方で，障害物から受ける力 f_{iW} は，式 (2–1) に示すように，式の第 3 項である．このため，進行方向ベクトル e と障害物を避ける力 f_{iW} は，別々の配列に格納する必要がある．図 5–1 に格子分割を用いた進行方向ベクトル計算の削減手法のフローチャートを示す．図 5–1 中の前処理は，提案手法に必要である格子ごとの進行方向や壁を避ける力を算出する．図 5–1 中の運動方程式を用いた計算は，エージェントの進行方向を計算する処理であり，エージェントの進行方向ベクトル e や障害物を避ける力 f_{iW} を前処理で算出した値を参照する．

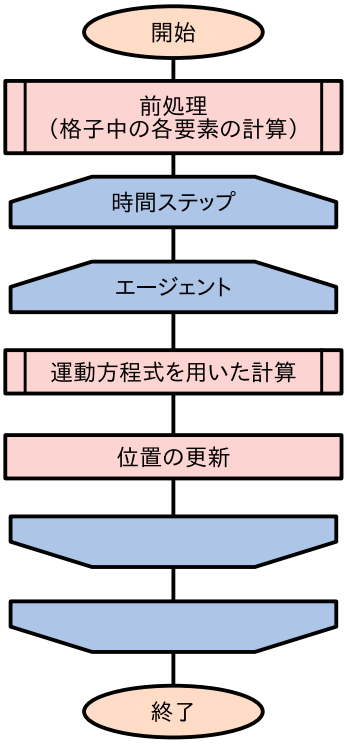


図 5–1 ： 提案手法の解析全体のフローチャート

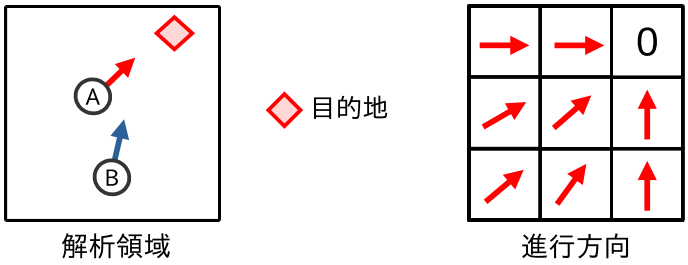


図 5–2 ： 提案する格子分割の例

5.2.1　格子ごとの進行方向ベクトル e の計算方法

進行方向ベクトル e は，目的地に進むベクトルであり，エージェントの座標から目的地の座標までのベクトルである．このため，ベクトル e は解析領域を格子状に分割した領域ごとの中心座標から目的地の座標までのベクトルを計算することで，解析前に計算が可能となる．図 5–2 に格子分割した進行方向の例を示す． 図 5–2 中の○はエージェントであり，エージェント A，B が 1 つの目的地に向かう様子である．図中のエージェント B は，エージェント A の位置に移動したとき，エージェント A

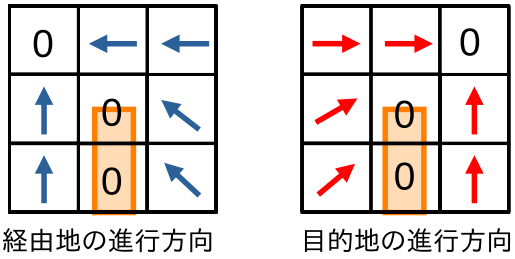


図 5–3 ： 経由地がある場合の進行方向の例

と同じ進行方向になる．このため，解析領域を分割し，格子ごとに進行方向を算出することで，解析中の進行方向の再計算が削減できる．目的地と障害物が含まれる格子は，エージェントが正しく目的地に進むようにするために，0 ベクトルを格納する．0 ベクトルが格納された格子中に存在するエージェントは，改めて個別に進行方向ベクトル e を計算する．格子ごとの進行方向は，格子の中心座標から目的地までのベクトル e を求め，配列に格納する．目的地の他に経由地が複数存在する場合，進行方向は，経由地ごとに異なるため，経由地ごとに違う配列に格納する必要がある．図 5–3 に経由地が複数存在する例を示す．図 5–3 中の右図は経由地の進行方向，左図は目的地の進行方向，青色の矢印は経由地に進む進行方向，青矢印は目的地の進行方向，0 は進行方向を個別計算するための 0 ベクトル，オレンジ色の四角は机などの障害物を示す (要工事：元となる画像がない)．図 5–3 のように，経由地と目的地の進行方向は，それぞれの座標が異なるため，ベクトル e も大きく異なることから，それぞれ別の配列で保持する必要がある．経由地ごとに別の配列に進行方向を格納するため，進行方向を格納する配列の要素数は式 (5–1) に進行方向を格納する要素数を示す．

$$\text{進行方向を格納する格子の要素数 [個]} = \left(\frac{\text{解析領域 [m]}}{\text{格子サイズ [m]}}\right)^2 \times \text{経由地数 [個]} \quad (5\text{–}1)$$

式 (5–1) のように，進行方向を格納する配列は，解析領域の大きさや格子サイズの小ささ，経由地数の大きさに応じて要素数が増加する．また，進行方向を格納する配列の前処理にかかる時間は，配列の要素数に応じて格子ごとの進行方向計算回数が増加するため，配列の大きさに応じて増加する．

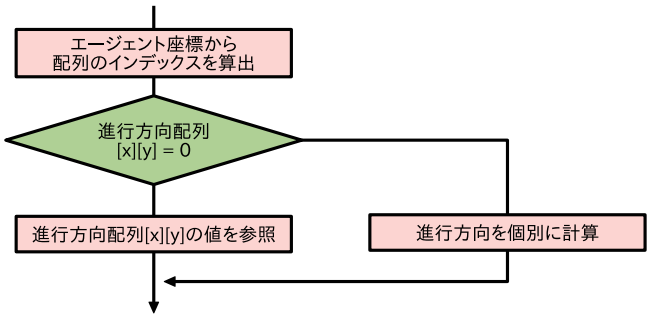


図 5–4 ： シミュレーション中の進行方向計算のフローチャート

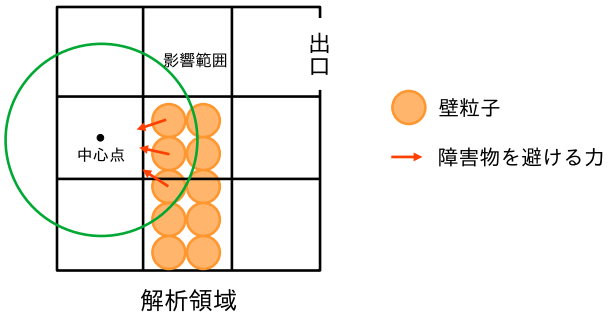


図 5–5 ： 格子分割した障害物を避ける力 F_{iW} の例

5.2.2　格子ごとの障害物を避ける力 F_{iW} の計算方法 (工事中)

障害物を避ける力 F_{iW} は，エージェントの座標と障害物の座標に応じて変化する．障害物は，机や壁などの固定された物であるため，座標が変化しない．このため，障害物を避ける力 F_{iW} は，解析領域を格子状に分割した領域ごとの中心座標と障害物の座標から解析前にあらかじめ計算が可能となる．図 5–5 に格子分割した障害物を避ける力 F_{iW} を示す．図 5–5 中の四角は解析領域を格子状に分割した格子，黒点は格子の中心点，緑色の丸は格子の中心点からの影響範囲，オレンジ色の丸は壁粒子，赤色の矢印は黒点の中心点が受ける障害物を避ける力 f_{iW} である．図 5–5 の例は，黒点が存在する格子が受ける障害物を避ける力 f_{iW} を示しており，黒点の中心点から緑色の範囲に存在する壁粒子から力を受ける．

ーーーーーーーーーーーーー下記は下書きーーーーーーーーーーーーー

格子ごとの進行方向は，格子の中心座標から目的地までのベクトル e を求め，配列に格納する．同様に，

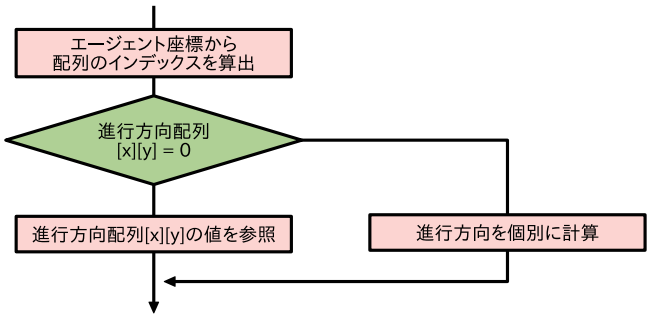


図 5–6 ： シミュレーション中の f_{iW} 計算のフローチャート

図??に提案手法の前処理のフローチャートを示す．図??に示すように，提案手法は，進行方向のベクトル e を格納する配列と壁を避ける力 F_W を格納する配列が必要となる．各配列の計算は，図??のように格子ごとにあらかじめ計算する．進行方向のベクトル e を格納する配列中の要素は，図 5–3 のように経由地ごとに違うため，それぞれの経由地で計算する必要となる．このため，進行方向ベクトル e を格納する配列の前処理は，経由地数の多さや格子サイズの細かさに応じて計算回数が増加する．図??中の壁を避ける力 f_{iw} を格納する配列は，経由地が変わっても変化しないため，解析領域全体で一つとなる．このため，壁を避ける力 f_{iw} を格納する配列の前処理は，格子サイズの細かさに応じて計算回数が増加する．

$$\text{障害物を避ける力を格納する格子の要素数 [個]} = \left(\frac{\text{解析領域 [m]}}{\text{格子サイズ [m]}}\right)^2 \qquad (5\text{–}2)$$

5.2.3　格子分割を用いた SFM(工事中)

工事中.

5.3　評価 (工事中)

SFM を用いた人流シミュレーションに対する格子分割を用いた進行方向の計算回数削減手法の有効性を確認するために，既存手法であるセル分割法と提案手法を用いて人流シミュレーション評価環境は，表 5–1 に示すマシンである．本評価では，表 5–2 に示すパラメータを用いて SFM の運動方程式を計算する．

表 5–1 ： 評価環境

CPU	Intel Xeon CPU E5-2667w v2
メモリ	32GB
OS	Linux 6.5.8
コンパイラ	gcc 13.2.0
最適化オプション	-O3

表 5–2 ： 測定条件

A_i	2000N
B_i	0.08m
k	$1.2 \times 10^5 kgs^{-2}$
κ	$2.4 \times 10^5 kgm^{-1}s^{-2}$
v_i^0	1.4m/s
m_i	80kg
τ_i	0.5
r_i	0.25m
相互作用範囲	5m

5.3.1　通路幅を変えた時の解析時間

5.4　本章のまとめ (工事中)

工事中.

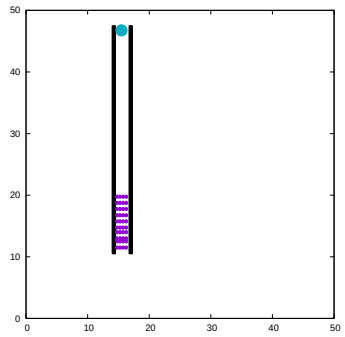


図 5–7 ： 通路幅 2m の初期配置

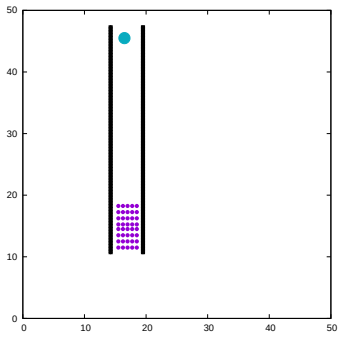


図 5–8 ： 通路幅 5m の初期配置

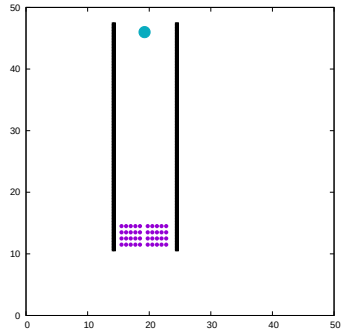


図 5–9 ： 通路幅 10m の初期配置

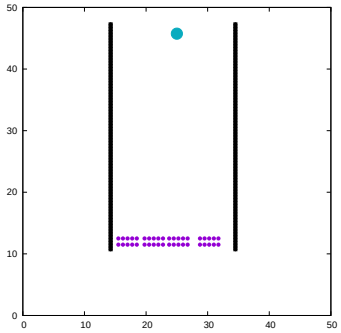


図 5–10 ： 通路幅 20m の初期配置

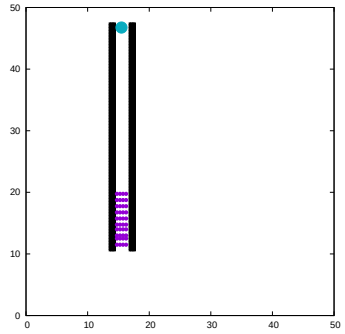


図 5–11 ： 厚さ 2 倍

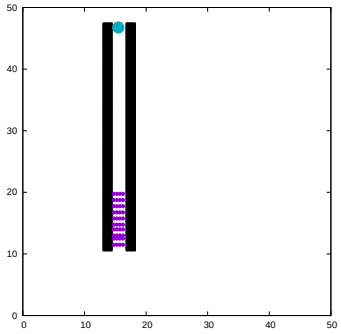


図 5–12 ： 厚さ 3 倍

第6章
おわりに

\(^o^)/

謝辞

aa aa
(16)

2023 年 12 月 24 日

参考文献

(1) 北上靖大,　森俊勝，坂平文博，志村泰知，杉浦哲平： 都市課題の改善に向けたマルチエージェント・シミュレーションの活用，電気学会論文誌 C(電子・情報・システム部門誌)， Vol. 133, No. 9, pp. 1640–1644 (2013).

(2) NTT データ理数システム人流シミュレーションによるオリンピック開催時の駅の混雑分析事例，入手先 <https://www.msiism.jp/article/taguchi-olympic-human-flow-simulation.html> (2021-04-21).

(3) 大西正輝，山下倫央，星川哲也，佐藤和人： 人の流れの計測とシミュレーションによる避難誘導方法の伝承 – 新国立劇場における避難体験オペラコンサートを例に –，人工知能学会第二種研究会資料， Vol. 2015, No. KST-26, p. 06 (2015).

(4) 株式会社ホロラボホロラボが鹿島の BIM データを活用した避難シミュレーションの Microsoft HoloLens 2 版開発を支援，入手先 <https://prtimes.jp/main/html/rd/p/000000056.
(2023-02-20).

(5) 磯崎勝吾,　中辻隆： Social force model を基にした歩行者の避難シミュレーションモデルに関する研究，土木学会北海道支部論文報告集， Vol. 66 (2009).

(6)　安藤歩： ソーシャルフォースモデルを用いた教室内における避難行動シミュレーション，東京都立大学都市教養学部都市教養学科卒業論文.

(7) Chisholm, R., Maddock, S. and Richmond, P.: Improved GPU near neighbours performance for multi-agent simulations, Journal of Parallel and Distributed Computing, Vol. 137, pp. 53–64 (2020).

(8) Li, X., Cai, W. and Turner, S. J.: Efficient Neighbor Searching for Agent-Based Simulation on GPU, 2014 IEEE/ACM 18th International Symposium on Distributed Simulation and Real Time Applications, pp. 87–96 (2014).

(9) 後藤仁志，堀智恵実五十里洋行，KHAYYER, A.： GPU による粒子法半陰解法アルゴリズムの高速化，土木学会論文集 B， Vol. 66, No. 2, pp. 217–222 (2010).

(10) Hu, X. and Adams, N.: An incompressible multi-phase SPH method, Journal of Computational Physics, Vol. 227, No. 1, pp. 264–278 (2007).

(11) Oguchi, K., Shibuta, Y. and Suzuki, T.: Accelerating Molecular Dynamics Simulation Performed on GPU, Journal of the Japan Institute of Metals and Materials, Vol. 76, No. 7, pp. 462–467 (2012).

(12) 茂渡悠介，酒井幹夫，越塚誠一，山田祥徳：GPU による離散要素法シミュレーションの高速化，粉体工学会誌， Vol. 45, No. 11, pp. 758–765 (2008).

(13) Allen, M. P. and Tildesley, D. J.: Computer simulation of liquids, Oxford university press (2017).

(14) 平林久義，佐藤雅弘：線形リストを用いた粒子法の近傍粒子探索，日本計算工学会論文集， Vol. 2010, pp. 20100001–20100001 (2010).

(15) Helbing, D., Farkas, I. and Vicsek, T.: Simulating dynamical features of escape panic, Nature, Vol. 407, No. 6803, pp. 487–490 (2000).

(16) 茨木俊秀： 組合せ最適化 —分枝限定法を中心として—，産業図書 (1983).

付 録 A
プログラムの説明

本論文の実装では，SFM を用いた人流シミュレーションを C 言語を用いて実装する．本実装は，関連文献を参考に作成した．本章では，本論文で使用したプログラムのデータ構造やファイル構造，実行方法，測定条件の変更方法，入出力ファイルについて述べる．

A.1 データ構造

エージェントのデータ構造は，agent_t の構造体名で構造体配列として保持する．agent_t の構造体中のメンバを下記に示す．

エージェントの構造体

```
typedef struct{
    int num;
    double x, y;
    double gx, gy;
    int r_num;
    struct route_t *goal_p;
}agent_t;
```

agent_t 中の num のメンバはエージェントの番号，x と y のメンバはエージェントの座標，gx と gy のメンバは目的地の座標，r_num のメンバは経由地の番号，*goal_p のメンバはエージェントが向かう経由地のポインタを示す．経由地のデータ構造造は，route_t の構造体名で構造体配列として保持する．route_t の構造体中のメンバを下記に示す．

経由地の構造体

```
typedef struct{
    int num;
    double x, y;
    double rad:
    int next_num;
    struct route_t *next;
    double **ex, **ey;
}route_t;
```

route_t 中の num のメンバは経由地の番号，x と y のメンバは経由地の座標，rad のメンバは経由地のゴール判定の半径，next_num のメンバは，次の経由地番号，*next のメンバは次の経由地のポインタ，**ex と**ey のメンバは，格子分割を用いた場合の前処理で算出した経由地までの進行方向を保持する配列である．

壁粒子のデータ構造は，wall_t の構造体名で単方向リストとして保持する．本論文は，壁を複数の粒子として実装しているため，壁粒子から受ける力の計算にセル分割法を用いる．壁粒子は，解析開始から終了まで固定されているため，座標が変化しない．このため，壁粒子のセル分割法のデータ構造は，シミュレーション開始の一回のみの生成となるため，連結リスト法で実装する．下記に壁粒子の構造体を示す．

壁粒子の構造体

```
typedef struct {
    double x, y;
    struct wall_t *np;
}wall_t;
```

wall_t 中の x と y のメンバは壁粒子の座標，*np のメンバは次の壁粒子のポインタを示す．下記にエージェントのセル分割法に用いるハッシュ法の構造体 cell_t のメンバを示す．

セル分割法 (ハッシュ法) の構造体

```
typedef struct {
    int *hash;
    int *index;
    int *start;
}cell_t;
```

構造体 cell_t 中の*hash のメンバはハッシュ法のハッシュ配列，*index のメンバはハッシュ法のインデックス配列，*start のメンバはハッシュ法のスタート配列である．エージェントのセル分割法は，時間ステップごとにエージェントの座標が変わるため，時間ステップごとに再構築する必要がある．このため，再構築する速度や保守性の観点からセル分割法の中でもハッシュ法を構造体としてデータを保持する．

A.2　プログラムのファイル構造

SFM を用いた人流シミュレーションのファイル構造を下記に示す．

プログラムのファイル構造

```
sfm_program
├── sfm.c
└── inputwall
    ├── pc_3.c
    ├── kyositu.c
    ├── haba2.c
    ├── haba5.c
    ├── haba10.c
    └── haba20.c
```

sfm.c は，SFM を用いた人流シミュレーションのプログラムであり，C 言語で記述されている．inputwall 内に存在する C 言語のファイルは，sfm.c で用いるエージェントや壁，経由地の初期配置が記述されたコードである．表 A–1 に本論文中の評価で使用した初期配置のファイル名を示す．表 A–1 に示すファイルは，sfm.c 中にインクルードすることで使用することができる．

表 A–1 ： 本論文中の評価で使用した初期配置

配置名	ファイル名
演習室	pc_3.c
教室	kyositu.c
演習室 (dense)	pc_3_dense.c
演習室 (sparse)	pc_3_sparse.c
演習室 (sparse&wide)	pc_3_sparse_wide.c
通路 (幅 2m)	haba2.c
通路 (幅 5m)	haba5.c
通路 (幅 10m)	haba10.c
通路 (幅 20m)	haba20.c

A.3　実行方法

SFM を実装したプログラムのコンパイルは下記のコマンドの通りである．

コンパイル方法

```
gcc [ファイル名] -lm -O3 ‘pkg-config –cflags –libs glib-2.0‘
```

本プログラムの入力データは，GIMP を用いて C 言語に出力したものであり，プログラム上で処理するために glib.h をインクルードする必要がある．glib.h をインクルードするためには，コンパイル時に ‘pkg-config –cflags –libs glib-2.0‘を追記することが必要である．

コンパイルして生成されたファイルは，次のコマンドで実行することができる．

実行方法

```
./[実行ファイル名] [実行ステップ数] [格子サイズ]
```

第 2 引数は，実行ステップ数で解析したいステップ数を入力する必要がある．第 3 引数は，提案手法の格子サイズを入力する必要がある．既存手法で実行する場合は，第 3 引数を使うことがないが仮に入力が求められる．既存のセル分割法と提案手法は．一つのファイルで構成されており，切り替えるために，#DEFINE で定義された変数 (KIZON) が使用されている．表 A–2 に既存手法と提案手法を切り替えるための変数 (KIZON) の状態を示す．

表 A–3 ： 測定条件のパラメータ変数

変数名	内容
A_I	A_i
B_I	B_i
K	k
KAPPA	κ
V_0	v_i^0
M_I	m_i
TAU	τ_i
R_I	r_i
H	刻み値
MAX_SPEED	最大歩行速度
NEIGHBOR_RADIUS	エージェントの影響半径

表 A–2 ： 既存手法と提案手法の切り替え方法

#DEFINE KIZON	既存手法 (セル分割法) で実行
//#DEFINE KIZON	格子分割を用いた進行方向計算の削減手法で実行

表 A–2 に示すように，提案手法で実行する場合は，#DEFINE KIZON をコメントアウトする必要がある．

A.4 測定条件の変更方法

本節では，測定条件のパラメータの変更方法について述べる．測定条件のパラメータは，C 言語中の#DEFINE を用いた変数で設定できる．表 A–3 にそれぞれの変数とパラメータ内容について示す．表 A–3 中のパラメータの初期設定は，表 4–5 と同じ値である．

表 A–4 ： 動作確認済み環境

CPU	Intel Core CPU i5-8259U
メモリ	16GB
OS	Linux 6.5.9
GIMP バージョン	2.10.34

表 A–5 ： 各要素の設定方法

要素名	R	G	B
エージェント	240.0	初期経由地番号	-
壁粒子	0.0	-	-
経由地番号	24.0	経由地番号	次の経由地番号

A.5　初期配置のデータの作成方法

初期配置データの作成方法は，画像編集ソフト GIMP を用いて作成することができる．表 A–4 に本論文で作成したときに動作確認した環境を示す．また，本節で述べるデータの作成方法は，表 A–4 の環境下での作成方法である．

A.5.1　GIMP を用いた初期配置の作成方法

GIMP 画像中の 1 ピクセルの大きさは，プログラム中の変数 GRID_SIZE と同じである．変数 GRID_SIZE は，初期の設定で 0.25m であるため，4 ピクセルで 1m の大きさとなる．1 ピクセルは，RGB と A と呼ばれる 4 要素で構成され，R が赤，G が緑，B が青，A が透過度を示す．本プログラムでは，1 ピクセル中の RGB(RGB225 形式) を用いてエージェントと壁，経由地を判定する．表 A–5 に各要素の設定方法を示す．表 A–5 中のハイフンは，プログラム中で使用しない値である．エージェントの R(赤) に 240.0，G(緑) にエージェントが向かう経由地番号を設定する．エージェントの G(緑) に設定した経由地番号は，プログラム実行時に存在しない場合，エラーとして入力時に中断するため，注意が必要である．壁粒子は，R(赤) に 0.0 を設定する．壁粒子は，解析中動くこともなく，プログラム中で座標の情報のみを使用する．このため，壁粒子の G(緑) と B(青) は，任意の値を設定する．経由地は，R(赤) に 24.0，G(緑) に経由地の番号，B(青) に次進む経由地番号を設定する．

図 A–1 ： GIMP の出力ファイル形式の選択

図 A–2 ： GIMP の C 言語出力画面

A.5.2　GIMP を用いた C 言語の出力方法

前節の設定方法で初期配置した場合は，sfm.c を用いたプログラムで使用するために，C 言語で出力する必要がある．C 言語で出よくするためには，図 A–1 に示すように GIMP のエクスポート機能を用いたときに，C ソースコードを出力ファイル形式に設定する必要がある．C ソースコードをファイル形式に設定後は，図 A–2 に示すようなダイアログが表示されるため，図 A–2 と同じようにチェックボックスを付けて保存する．保存後は，sfm.c 中に保存した C 言語ファイルをインクルードすることで使用することができる．

表 A–6 ： 出力の設定方法

出力形式	出力方法
EPS 形式	プログラム中で「#DEFINE EPS」と記述する.
GIF 形式	プログラム中で「#DEFINE GIF」と記述する.

A.6　プログラム実行時の出力フォーマット

本プログラムを実行したときは，下記のような数字が出力される.

プログラム終了時の出力結果

[格子サイズ],[前処理の時間],[解析時間],[エージェント間距離の計算回数],[壁を避ける力の計算回数]

また，本プログラムは解析中の様子や初期配置を出力することが可能である．本プログラムは，画像を出力するために，gnuplot を使用しており，使用バージョンが 5.4 である．表 A–6 に解析中の様子を GIF 形式で出力する方法や初期配置を EPS 形式で出力する方法を示す．それぞれの出力方法を設定した場合は，プログラム終了時に test.gif や test.eps のファイル名で出力される.

付 録 B
タイトル案について

B.1 目的

SFM を用いた人流シミュレーションの高速化

B.2 手段

計算回数を削減するために下記の二つの手段を用いる.

- セル分割法のセル選択
- 格子分割

B.3 結果

本研究の提案手法は，下記の結果を確認した.

- エージェント間距離 d_{ij} の計算回数の削減
- 進行方向ベクトル e の計算回数の削減
- 障害物を避ける力 F_{iW} の計算回数の削減
- 解析時間の短縮（高速化）

B.4 タイトル（案）

上記の目的や手段，結果から，下記に示すタイトルを案として考えます.

- **進行方向の計算回数削減による SFM を用いた人流シミュレーションの高速化**
  Speedup of Pedestrian Simulation using SFM by Reducing the Number of Calculations for Traveling Direction
- **進行方向の計算回数削減による SFM の高速化**
  Speedup of SFM by Reducing the Number of Calculations for Traveling Direction
- **進行方向の計算回数削減による人流シミュレーションの高速化**
  Speedup of Pedestrian Simulation by Reducing the Number of Calculations for Traveling Direction

B.5 下書き

- **進行方向の計算回数削減による SFM を用いた人流シミュレーションの高速化**
  Speedup of Pedestrian Simulation using SFM by Reduction of the Number of Calculations for Traveling Direction
- **進行方向の計算回数削減による SFM の高速化**
  Speedup of SFM by Reduction of the Number of Calculations for Traveling Direction
- **進行方向の計算回数削減による人流シミュレーションの高速化**
  Speedup of Pedestrian Simulation by Reduction of the Number of Calculations for Traveling Direction

B.6 発表名，論文名

- エージェント間距離の計算回数削減による視野を用いた SFM の高速化（情報処理学会論文誌）
- 視野を考慮した探索範囲削減による人流シミュレーションの高速化（FIT）
- 人流シミュレーションにおける格子分割を用いた進行方向ベクトル計算の削減手法（全国大会）
- 格子分割を用いた進行方向計算の削減による人流シミュレーションの高速化（xsig）