Vol.2014-CG-157 No.14
Vol.2014-CVIM-194 No.14
2014/11/21

密度情報を用いた人混みシミュレーションのための衝突回避アルゴリズム

龍宇涛[†1]　アランニャ・クラウス[†2]　狩野均[†2]

人混みシミュレーションための新たな衝突回避アルゴリズムを提案する。本手法では、より高速なシミュレーションと現実に近い動きを得るため、従来の Optimal Reciprocal Collision Avoidance 手法にエージェントの密度情報を加える。事前に前方の混雑具合をエージェントに感知させ、動的にエージェントの目的地を変える。本手法は数千規模のエージェントのリアルタイムのシミュレーションを実現できる。従来手法と比べ、シミュレーション時間が 60%から 28%までに短縮された.

Collision Avoidance Algorithm for Crowd Simulation using Density Information

YUTAO LONG[†1]CLAUS ARANHA[†2]
HITOSHI KANOH[†2]

We present a new collision avoidance algorithm for crowd simulation. To obtain faster simulation and more realistic movement, our approach extends Optimal Reciprocal Collision Avoidance method with agent density information. We use forward information to inform agents and change the destination of agents dynamically. This approach can handle the simulation of thousands of agents in real-time. As compared to prior collision avoidance algorithm, the simulation time is reduced by 40% to 72%.

1. はじめに

人混みのシミュレーションは、都市計画、建物の安全評価、コンピュータアニメション、ゲーム、映画、バーチャルリアリティなど様々な分野に応用されている。実世界の人混みは、個々の通行人（エージェント）の目的地だけでなく、環境の中にある障害物、周辺の人の動きなどにも左右されるため、非常に複雑な動きをすることが知られている。このため、一つのモデルで人混みのすべての動きを再現することは難しい。従来の人混みシミュレーション手法では、問題を簡単化にするため、問題を「大域的なパスプランニング」と「局所的な衝突回避」に分割して、シミュレーションを行っている。

実世界の通行人は近傍の障害物・通行人だけではなく、前方の状況も感知しながら、動的にパスプランニングと衝突回避を行っている。前方の密度情報を感知することによって、人々は混雑になる区域を事前に避けることができ、より適切なパスを選択することが可能となる。結果として、人の流れが速くなり、目的地までの到達時間が短縮されることになる。しかし、従来手法の多くはこの点を考慮していないため、シミュレーションの中で、エージェントが混雑している区域に飛び込むという不自然な現象がよく見られる。

本稿では、実用性を高めるため、既存の衝突回避手法に対応できる新しい密度の計算方法と速度の選択方法を提案する。計算した密度の情報に応じて、エージェントの速度を変えることで、人混みシミュレーションの高速化を図る。提案手法の計算コストは小さく、市販のパソコンで、数千規模のエージェントのリアルタイムでのシミュレーションが可能である。

以下、２章では、衝突回避アルゴリズムの従来研究、ならびに、提案手法のベースとなった Optimal Reciprocal Collision Avoidance アルゴリズムについて述べる。３章では、提案する密度の計算方法と速度の選択アルゴリズムを説明する。４章では、ベンチマーク問題に対するシミュレーション結果を従来手法と比較し、本手法の有効性を示す。

2. 研究分野の概要

2.1 従来研究

人混みのシミュレーションは主に大域的なパスプランニングと局所的な衝突回避によって構成されている。大域的なパスプランニングの段階では、目的地と環境の中にあ

†1 筑波大学大学院システム情報工学研究科コンピュータサイエンス専攻
Department of Computer Science, Graduate School of Systems and Information Engineering, University of Tsukuba
†2 筑波大学システム情報系情報工学域
Division of Information Engineering, Faculty of Engineering, Information and Systems, University of Tsukuba

る静的な障害物だけを考え、目的地への最適なパスを決める。A*アルゴリズム[1][2][3]がよく使われるパスプランニングの手法である。一方、局所的な衝突回避は、動的なエージェントと障害物を対象としている．大域的なパスプランニングで得られたパス（エージェントの速度の大きさと方向）は衝突回避の段階で修正される．

Reynolds の先導的な研究[4][5]が発表された以来，人混みシミュレーションのための衝突回避アルゴリズムが多く提案されている．人混みの中のエージェントに対する表現によって，衝突回避アルゴリズムをさらに「連続的」と「離散的」に分類することができる．

連続的な手法では，グリッド状のポテンシャルフィールドを用いて，エージェントの動きを決める．Hughes らは流体の研究分野の手法を人混みシミュレーションへ応用した[6]．Treuille らは Hughes らの手法を発展させ，すべてのエージェントを幾つかの連続的な流体と見なした[7]．流体の密度と速度フィールドを用いて，高密度なエージェントのシミュレーションにおいてスムーズな動きを得ることができた．しかし，連続的な手法は大規模なシミュレーションに対応できるが，エージェントの個性的な動きを表現することが難しいという問題がある．

離散的な手法では，一つのエージェントに対して，他のすべてのエージェントを動的な障害物と見なして衝突回避を行う．個々のエージェントのパラメータを個別に設定することも可能になる．Reynolds[4]は簡単な局所ルール用いて，群れの行動を再現した．Golas らは「lookahead」というコンセプトを用いて，ハイブリッドな遠距離衝突回避手法を提案した[10]．その他の研究として，Force[8][9]に基づき，セルオートマトン[11]に基づき，Vision[12]に基づき，確率に基づき[13], Velocity Obstacle[14]に基づきの手法も提案されている．

本論文では，Velocity Obstacle の代表的な手法である Optimal Reciprocal Collision Avoidance[15][16]（以下 ORCA）に基づいて，密度情報を加え，比較実験を行った．次節では，この ORCA について説明する．

2.2 比較手法

2.2.1 問題定義

ORCA が対象とする Reciprocal n-body collision avoidance 問題では，n 個の円形のエージェントが平面$\mathbb{R}^2$上を移動している．任意のエージェント A は，現在位置$\boldsymbol{P}_A$（円の中心），現在速度$\boldsymbol{V}_A$，半径r_A，最大スピードV_A^{max}，優先速度$\boldsymbol{V}_A^{pref}$といったパラメータを有している．ここで優先速度とは，エージェントが目的地に向かう途中に他のエージェントは存在しないと仮定しているときの速度である．これらのパラメータは，他のエージェントから観測可能であるとする．

すべてのエージェントの第一目標として，新しい速度$\boldsymbol{V}_A^{new}$を選択する．その速度が，すべてのエージェントがこの新しい速度で移動し続け，τ秒以内で衝突が起こらないとのことを保証する．第二目標として，エージェントが優先速度$\boldsymbol{V}_A^{pref}$に一番近い$\boldsymbol{V}_A^{new}$を選択する．

2.2.2 予備知識

1) *Velocity Obstacle*

エージェント A と B に対して，Velocity Obstacle $VO_{A|B}^{\tau}$とは，時刻τ以前のある時刻で，A と B が衝突するような，B に対する A の相対速度の集合である[14]．Velocity Obstacle は以下ように定式化されている：

$$VO_{A|B}^{\tau} = \{\boldsymbol{v} | \exists t \in D(\boldsymbol{P}_B - \boldsymbol{P}_A, r_A + r_B)\} \quad (1)$$

$$D(\boldsymbol{P}, r) = \{\boldsymbol{q} | \|\boldsymbol{q} - \boldsymbol{p}\| < r\} \quad (2)$$

$D(\boldsymbol{P}, r)$は中心が点 P にある開区間の半径 r の円を表している．

2) *Reciprocally Collision-avoiding*

エージェント A と B の速度を$\boldsymbol{v}_A$，$\boldsymbol{v}_B$とする．Velocity Obstacle の定義から，$\boldsymbol{v}_A - \boldsymbol{v}_B \notin VO_{A|B}^{\tau}$が成立すれば，A と B が時間$\tau$以内では衝突しないことがわかる．一般的に，集合$V_B$に対して，もし$\boldsymbol{v}_B \in V_B$と$\boldsymbol{v}_A \notin VO_{A|B}^{\tau} \oplus V_B$が成立すれば，A と B が時間$\tau$以内では衝突しないことが保証される．ここで，$\oplus$は集合の Minkowski sum を意味する．よって，B が集合V_Bから速度を選択する場合，A の Collision-avoiding 速度の集合は以下のように定義することができる．

$$CA_{A|B}^{\tau}(V_B) = \{\boldsymbol{v} | \boldsymbol{v} \notin VO_{A|B}^{\tau} \oplus V_B\} \quad (3)$$

もし$\mathrm{V_A} \subseteq \mathrm{CA_{A|B}^{\tau}(V_B)}$と$\mathrm{V_B} \subseteq \mathrm{CA_{B|A}^{\tau}(V_A)}$が成立すれば，A と B の速度集合$\mathrm{V_A}$と$\mathrm{V_B}$のペアを reciprocally collision-avoiding と呼ぶ．さらに，$CA_{A|B}^{\tau}(V_B) = V_A$と$CA_{A|B}^{\tau}(V_B) = V_A$が成立すれば，$\mathrm{V_A}$と$\mathrm{V_B}$が reciprocally maximal になる．

2.2.3 ORCA の定義

以上の定義を用いて，A と B が時間τ以内で衝突しないために，V_AとV_Bが reciprocally maximal になるようなV_AとV_Bのペアを選択する．このようなペアは無数に存在するが，現在速度に最も近く，かつ選択可能な速度の数を最大にするV_AとV_Bのペアを選択する．そのようなV_A，V_Bのペアを$ORCA_{A|B}^{\tau}$，$ORCA_{B|A}^{\tau}$と書く．

正確な定義は以下のようになる．

$$\left|ORCA_{A|B}^{\tau} \cap D(\boldsymbol{v}_A, r)\right| = \left|ORCA_{B|A}^{\tau} \cap D(\boldsymbol{v}_B, r)\right| \geq \min(|V_A \cap D(\boldsymbol{v}_A, r)|, |V_B \cap D(\boldsymbol{v}_B, r)|) \quad (4)$$

ここで，$|V|$は集合Vが$\mathbb{R}^2$平面上での面積を意味する．

さらに，A の最大スピードも考慮すると，他のすべてのエージェントに対して，A の選択したい速度集合$ORCA_A^{\tau}$は次のように書ける．

$$ORCA_A^{\tau} = D(0, V_A^{max}) \cap \bigcap_{B \neq A} ORCA_{A|B}^{\tau} \quad (5)$$

2.2.4 ORCA アルゴリズム

ORCA の定義を用いて，Reciprocal n-body collision avoidance 問題を解決する．基本的なアプローチとして，すべてのエージェントが衝突することなく，毎タイムステップΔtでなるべく早く目的地に到着するために，$ORCA_A^{\tau}$から優先速度$\boldsymbol{V}_A^{pref}$に一番近い速度を選択する．

Vol.2014-CG-157 No.14
Vol.2014-CVIM-194 No.14
2014/11/21

```
パラメータの初期化；
while(まだ目的地に到着していないエージェントがいる){
    for(すべてのエージェント){
        ほかのエージェントの速度と位置を獲得；
        ORCA_A^τを計算する；
        新しい速度を計算する；
        エージェントの位置を更新する；
    }
}
```

図 1 ORCA アルゴリズム

```
パラメータの初期化；
while(まだ目的地に到着していないエージェントがいる){
    for(すべてのエージェント){
        ほかのエージェントの速度と位置を獲得；
        密度情報を更新する；
        優先速度を計算する；
        ORCA_A^τを計算する；
        新しい速度を選択する；
        エージェントの位置を更新する；
    }
}
```

図 2 提案アルゴリズム

$$\boldsymbol{V}_A^{new} = \underset{\boldsymbol{v} \in ORCA_A^{\tau}}{\arg\min} \left\| \boldsymbol{v} - \boldsymbol{V}_A^{pref} \right\| \tag{6}$$

次に，$\boldsymbol{V}_A^{new}$を用いて，エージェントの位置を更新する．

$$\boldsymbol{P}_A^{new} = \boldsymbol{P}_A + \boldsymbol{V}_A^{new} \Delta t \tag{7}$$

上述のアルゴリズムを図 1 で定義する．

3. 提案手法

ORCA アルゴリズムでは，優先速度$\boldsymbol{V}_A^{pref}$はエージェントの目的地と現在位置から算出している．密度情報を考慮していないため，$\boldsymbol{V}_A^{pref}$の方向上が大変混雑しても，新しい速度$\boldsymbol{V}_A^{new}$は$\boldsymbol{V}_A^{pref}$に近づく必要があるため，エージェントが混雑区域を避けられないといった問題がある．結果として，エージェントの動きの信憑性とシミュレーションの効率が低下している．本提案手法は，ORCA アルゴリズムのパイプラインを変えずに，前方の密度情報に応じて，動的に優先速度$\boldsymbol{V}_A^{pref}$を変える．これによって，実世界の通行人が事前に混雑する区域を避けるという行動の再現とシミュレーション時間の短縮を図る．

提案アルゴリズムの流れを図 2 で示す．以下，密度の計算方法と優先速度の計算方法を分別に説明する．

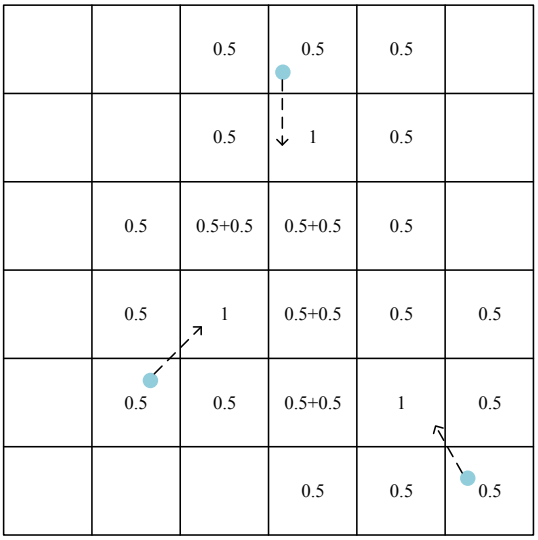

図 3 密度の計算方法

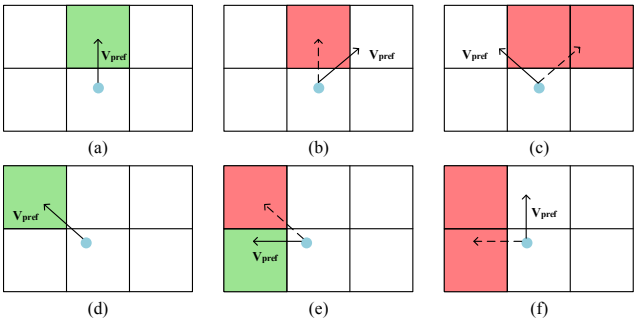

図 4 優先速度方向の選択方法

3.1 密度の計算方法

計算の手順は以下のようになる．

① 初期化の段階で，エージェントがいる区域を一定数の四角形のグリッドに分割する．

② 毎タイムステップにおいて，エージェントの現在の前進方向を基準に，エージェントを前方 8 メートルの位置までに投影する．

③ 図 3 で示したように，エージェントが投影しているグリッドを選択し，そのグリッドの密度を 1，周辺の 8 つのグリッドの密度を 0.5 増やす．

エージェントを前方へ投影する理由は，混雑するところを事前に予測するためである．

3.2 優先速度の計算方法

各グリッドの密度を計算した後に，すべてのエージェントに対して，今の前進方向の前方 8 メートルの密度を獲得する．8 メートルを取る理論基礎として，心理学分野で，パーソナルスペース[17]というよく知られている理論がある．パーソナルスペースとは，他人に近付かれると不快に感じる空間のことである．パーソナルスペースは 4 つの円

形ゾーンによって構成されている．その中で，知らない人と保ちたい距離（公共距離）は，約 7.6 メートルとなっている．これより，本研究では，前方を見る距離を 8 メートルとした．

獲得した密度の値に従って，図 4 で示した行動をとる．図 4 中の赤いグリッド（緑のグリッド）は，密度が密度閾値ρを超えている（超えていない）グリッドを示している．図 4(a)では，前方のグリッドの密度が密度閾値ρを超えてないため，前進方向を変えない．図 4(b)では，密度が閾値を超えているため，優先速度を右へ 45 度回転する．図 4(c)では，新たな前進方向の前方がまた混雑しているため，優先速度を左へ 90 度回転する．次のタイムステップで（図 4(d), 4(e), 4(f)），エージェントの位置が更新され，前タイムステップとほぼ同じ行動パータンをとる．違いは優先速度を優先的に左へ回転する．以上のアルゴリズムを図 5 に示す．

4. 評価実験

本章では，提案手法の性能を評価するために，３つのベンチマーク問題において, ORCA アルゴリズム（従来手法）との比較実験示す．提案手法と従来手法は C++で実装されている．すべての実験は Intel Core I5 4670K 3.4GHz で行った．

4.1 ベンチマーク問題

本実験では，以下の３つのベンチマーク問題を用いた．

(a) *Circle*

同じ$\mathbb{R}^2$平面にある複数のエージェントが円周上に均一に配置されている．すべてのエージェントの目的地は，初期位置の円心に対する反対側に設定されている（図 6(a)）．

(b) *4-ways*

同じ$\mathbb{R}^2$平面にある 4 組のエージェントが対角に配置されている. 各組の目的地は各組の反対側に設定されている（図 6(b)）.

(c) *Crossing*

同じ$\mathbb{R}^2$平面にある 2 組のエージェントが十字路の対角に配置されている．各組の目的地は各組の反対側に設定されている．エージェントは十字路の外に出ることができない（図 6(c)）.

4.2 パラメータ設定

本実験で用いたパラメータを表 1 に示す．提案手法と従来手法は同じパラメータを用いた（太字のパラメータは提案手法独自のパラメータ）. ORCA と共通のパラメータは，文献[15][16]に従って設定した．

(1) 前を見る距離とエージェント半径の設定

3 章で説明した通り，パーソナルスペース理論に従い，エージェントの前を見る距離を 8 に設定した．さらに，パーソナルスペースの個体距離（手を伸ばして相手と触れ合える距離）に従い，エージェントの半径 r を 1.2 とした．

```
for(すべてのエージェント){
    回転回数 = 0 ;
    前方グリッドの密度情報を獲得 ;
    if(右へ回転){
        while(密度>=密度閾値ρ){
            回転回数++ ;
            V_A^pref を右へ 45 度回転 ;
        }
        if(回転回数 >= 2){
        V_A^pref を左へ 90 度回転 ;
        V_A^pref を正規化 ;
        次のタイムステップは優先的に左へ回転 ;
        }
    else{
            while(密度>=密度閾値ρ){
            回転回数++ ;
            V_A^pref を左へ 45 度回転 ;
        }
        if(回転回数 >= 2){
        V_A^pref を右へ 90 度回転 ;
        V_A^pref を正規化 ;
        次のタイムステップは優先的に右へ回転 ;
    }
    }
}
```

図 5 優先速度の計算方法

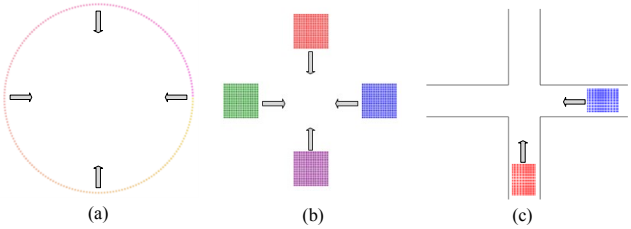

図 6 評価用ベンチマーク問題

表 1 実験のパラメータ設定

エージェントの半径 r	1.2
エージェントの数	300~2000
最大スピードV_A^{max}	2
タイムホライゾンτ	10.0
タイムステップΔt	0.25
グリッドの辺長 gridLength	5
前を見る距離 *d*	8
密度閾値ρ	5

表 2 予備実験の結果（10 回実行平均）

グリッドの辺長と密度閾値の組み合わせ	(5,5)	(5,10)	(5,15)	(5,20)	(5,30)
実行時間(s)	**100.8**	146.8	156.9	189.5	288.5
グリッドの辺長と密度閾値の組み合わせ	(10,5)	(10,10)	(10,15)	(10,20)	(10,30)
実行時間(s)	N/A	N/A	250.1	154.0	109.2

表 3 本実験の結果（10 回実行平均）

問題	エージェントの数	手法	実行時間(s)	改善倍率	毎フレームの計算時間(ms)
Circle	300	従来手法	50.0	1.7 倍	0.35
		提案手法	30.0		0.29
Crossing	600	従来手法	121.5	1.5 倍	2.4
		提案手法	81.3		1.2
4-way	2000	従来手法	349.4	3.6 倍	5.1
		提案手法	95.8		2.4

(2) グリッドの辺長と密度閾値の設定

グリッドの辺長と密度閾値の設定について，難しいベンチマーク問題 4-way-2000Agents において，予備実験を行い，シミュレーション時間を最小にするグリッドの辺長と密度閾値の組み合わせを最適値とする．なお，予備実験で，10 を超えたグリッドの辺長についても実験を行ったが，エージェントの動きが不自然になるため，グリッドの辺長の上限を 10 にした．本研究では，グリッドの辺長(5,10)と密度閾値(5,10,15,20,30)を組み合わせて，合計 10 通りの予備実験を行った．結果を表 2 に示す．グリッドの辺長と密度閾値が(5,5)のときに実行時間が最も短かったのでその値に設定した．なお，実験から，グリッドの辺長が長く，密度閾値が低い場合，シミュレーションが失敗してしまう可能性があることがわかった．考えられる理由として，グリッドの面積が大きく，密度閾値が低いので，エージェントの周辺のグリッドがすべて混雑と判定され，目的地に向かうことができなくなった．逆に，グリッドの面積が低い場合，密度閾値の増大に伴い，提案アルゴリズムが徐々に機能しづらくなるという傾向が見られる．

4.3 実験結果

本実験では，ベンチマーク問題 Circle-300Agents（円周半径 150），Crossing-600Agents（エージェント組の中心から十字路の中心までの距離は 130），4-way-2000Agents（エージェント組の中心から交差点の中心までの距離は 110）について実験を行った．実験結果を表 3 に示す．

表 3 から，すべてのベンチマーク問題において，時間と計算時間の両方が改善されたことがわかる．特に，難しい問題 4-way-2000Agents に対して，シミュレーション時間が 1/3.6 に短縮され，リアルタイムでの実行が可能となった．図 7(a)で，エージェントが分散して，混雑する区域を事前に避けるという行動が観察された．

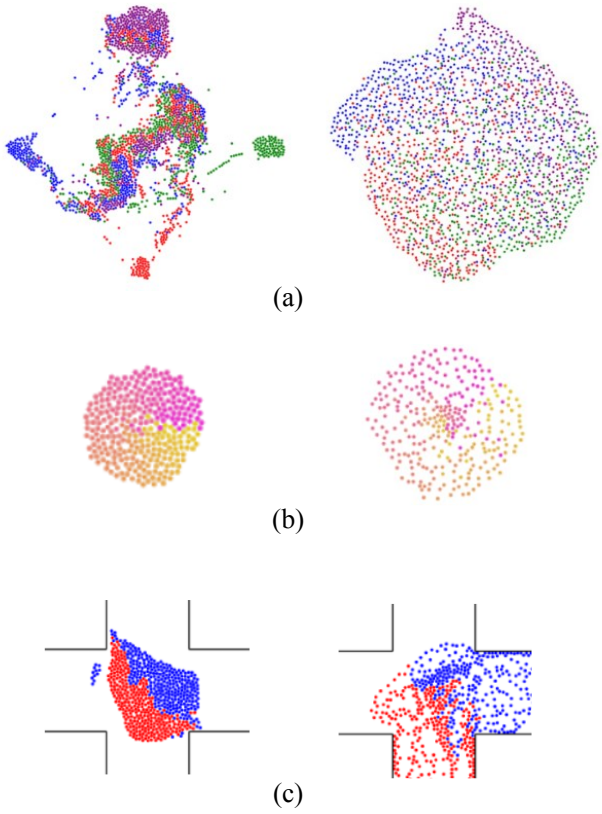

図 7 エージェント行動の比較

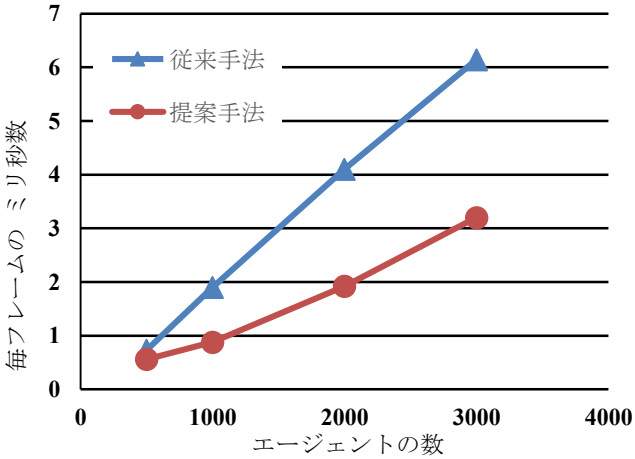

図 8 エージェント数と計算時間の関係

よく知られているベンチマーク問題 Circle に対しても，シミュレーションの実行時間が 1/1.7 に短縮された．Circle 問題に対する従来のシミュレーションによく見受けられる，すべてのエージェントが中心に集中する現象が低減された（図 7(b)）．

これに対して，Crossing 問題では 1/1.5 の短縮しか得られなかった．考えられる原因として，制限のあるフィールド

では，優先速度の選択アルゴリズムの効果が発揮しにくいと考えられる．しかし，シミュレーションの中で，現実に近い行動パータンが観察された（図 7(c)）．

エージェント数と計算時間の関係を考察するために，ベンチマーク問題 Circle に対して，エージェントの数を変えて(500, 1000, 2000, 3000)，実験を行った．実験結果を図 8 に示す．従来手法の ORCA の計算時間とエージェントの数は線形的な関係にある．提案手法もこの線形的な関係を引き継いでいると思われる．図 8 から，従来手法と提案手法の毎フレームの計算時間の差が，エージェント数の増加に伴って，増加する傾向があることがわかる．表 3 の実験結果も同じような傾向を示した．エージェント数の少ない Circle 問題に対して，従来手法と提案手法の毎フレームの計算時間に大差はないが，エージェント数が 2000 の 4-way 問題では，従来手法の毎フレームの計算時間が提案手法の２倍以上となった．

4.4 考察

実験により，提案手法が従来手法の ORCA より，多くの応用場面において，性能が優れていることを確認した．シミュレーション時間と毎フレームの計算時間が改善した理由について，以下のことが考えられる．

- 密度情報と優先速度の選択アルゴリズムを取り入れたことにより，混雑状況が改善され，ORCA 部分の計算量も少なくなった．
- 提案手法自体の計算量が少ない．

5. おわりに

密度情報を考慮した新たな人混みシミュレーションのための衝突回避アルゴリズムを提案した．従来の衝突回避アルゴリズムと比べ，シミュレーション時間とアルゴリズムの計算時間において，大きな改善が得られた．シミュレーションのなかで，エージェントのより現実に近い動きも観察された．

今後は密度と速度の関係について検討し，実世界のデータとも比較する予定である．

参考文献

1) Hart P., Nilsson N., Raphael B., A formal basis for the heuristic determination of minimum cost paths, IEEE Transactions on Systems Science and Cybernetics; 4(2): 100–107 (1968).
2) Dechter, R., Pearl, J., Generalized best-first search strategies and the optimality of a*. J. ACM 32, 3, 505–536 (1985).
3) Trovato, K. I., Dorst, L., Differential a*. IEEE Trans. on Knowl. and Data Eng, 14, 6, 1218–1229 (2002).
4) Reynolds C. W., Flocks, herds and schools: A distributed behavioral model, In SIGGRAPH ’87: Proceedings of the 14th annual conference on Computer graphics and interactive techniques (1987).
5) Reynolds C. W., Steering behaviors for autonomous characters, In GCD ’99 (1999).
6) Hughes, R.L., The flow of human crowds. Annu. Rev. Fluid Mech. 35, 169-182 (2003).
7) Treuille A., Cooper S., Popovi´c Z., Continuum crowds, ACM Trans. Graph. 25, 3, 1160–1168 (2006).
8) Helbing D., Farkas I., Viscek T., Simulating dynamical features of escape panic. Nature 407:487-490 (2000).
9) Chrabi M., Sefried A., Schadschneider A., Generalized centrifugal-force model for pedestrian dynamics, Phys Rev E 82:046111 (2010).
10) Golas A., Narain R., Lin MC., Hybrid long-range collision avoidance for crowd simulation. In: ACM Symposium on Interactive 3D Graphics, pp.29-36 (2013).
11) Schadschneider A., Cellular automaton approach to pedestrian dynamics – theory. Pedestrian and Evacuation Dynamics (2001).
12) Ondrej J., Pettre J., Olivier AH, Donikian S., A synthetic-vision based steering approach for crowd simulation, In: Proc. SIGGRAPH. pp. 123:1-123:9 (2010).
13) Metoyer, R. A., Hidgins, J. K., Reactive pedestrian path following from examples, In Proc. 16th Int. Conf. Computer Animation and Social Agents, 149 (2003).
14) Fiorini, P., Shiller Z., Motion planning in dynamic environments using Velocity Obstacles, Int. Journal of Robotics Research 17(7), pp. 760-772 (1998).
15) Van den Berg J., Patil S., Sewall J., Manocha D., Lin M., Interactive navigation of multiple agents in crowded environments, In: I3D '08. pp. 139-147 (2008).
16) Van den Berg J., Guy SJ., Lin M., Manocha D., Reciprocal n-body collision avoidance, In: Inter. Symp. on Robotics Research (2009).
17) Hall, Edward T., The Hidden Dimension. Anchor Books, ISBN 0-385-08476-5 (1966).