F-017

深層強化学習による最適な分散衝突回避
Optimal Distributed Collision Avoidance with Deep Reinforcement Learning

郷原 一眞[1)]　平山　勝敏[1)]　沖本　天太[1)]　金　東均[2)]
Kazuma Gohara　Katsutoshi Hirayama　Tenda Okimoto　Kim Donggyun

1　はじめに

衝突回避の自動化は，ロボット [23]，船舶 [10]，自動車 [21]，自律飛行体 [3] などの移動体からなるマルチエージェントシステム（MAS）における中心的な研究テーマである．この技術を実世界で実現するためには，アルゴリズムの基本設計や実装，エージェントの運動力学や操縦性，環境ダイナミクスなど，さまざまな課題に取り組む必要がある．しかし，アルゴリズムの基本設計に焦点を当てると，衝突回避アルゴリズム開発における最重要課題は「安全性の確立」にあり，次に「効率性の追求」であると言えるだろう．

衝突回避アルゴリズムに対する一般的なアプローチとして，「一対多（one-to-many）」アルゴリズムがある．これは，単一のエージェントが近傍のエージェントの観測情報を元に自身の衝突回避行動（針路や速度）を最適化するものである．一対多アルゴリズムはさらにルールベース [1, 7, 14, 17, 20] や学習ベース [2, 4, 5] に分類できる．一対多アルゴリズムには計算コストが小さいという利点がある一方で，システム全体の最適性を保証できないという難点がある．すなわち，ある特定のエージェントの意思決定が他のエージェントに与える影響を考慮できず，結果として衝突を誘発する可能性があるため，システム全体で見れば必ずしも最適な衝突回避とはならない．

これに対し，エージェント間の通信が可能な環境下ではシステム全体の最適性を考慮でき，このようなアルゴリズムは「多対多（many-to-many）」アルゴリズムと呼ばれる．例えば，主に倉庫ロボットの衝突回避に用いられるマルチエージェント経路探索（MAPF）の分野では，中央の計算機が全エージェントの経路を最適化する集中型アルゴリズムが用いられており [6, 19]，中央に完全なアルゴリズムを適用すれば全体最適が厳密に担保できる．しかしながら，多数のエージェントが参加する大規模なシステムの場合，集中型アルゴリズムは計算量が増大し，迅速な意思決定が困難となる．また，そもそも中央の計算機を想定できない状況（自動車や船舶など）には集中型アルゴリズムは不向きである．

一方，中央の計算サーバを用いることなく，計算を各エージェントに分散させる方法論として，分散型アルゴリズムが提案されており，主に船舶衝突回避の分野で研究されている [9, 11, 12, 13, 15]．いずれも AIS と呼ばれる既存の船舶間通信システムを利用したエージェント間通信を想定しており，特に衝突回避問題を分散制約最適化問題（DCOP）として定式化している．$DSSA^+$[9] はそのうちの一つであり，ヒューリスティックな DCOP アルゴリズムである DSA[24] を適用することでシステム全体の挙動を最適化するため，通信と計算のオーバーヘッドが他の分散アルゴリズムに比べ小さいという特長を持つ．しかしながら，$DSSA^+$ をはじめとする分散アルゴリズムでは，衝突回避の一連の流れにおいて，一部のエージェントが予定した針路から過度に逸脱したり，加減速を繰り返したりといった，極めて非効率的な挙動に陥る場合がある．これは，分散アルゴリズムがシステム全体を短期的にのみ最適化するため，エージェント個人の長期的な効率性を考慮できないことが原因である．言い換えれば，分散型アルゴリズムではエージェントがシステム全体の短期的な最適解に追従することを想定するが，エージェント個人の長期的視点から見ればその解は必ずしも最適ではない．このように分散型アルゴリズムでは，最適化の反復により全体最適と個人最適の間のギャップが蓄積されることで，エージェント単位でみた長期的な移動効率性が悪化する場合がある．

本研究では，分散型アルゴリズムにおける上述の問題を解消する初の試みとして，$DSSA^+$ と深層強化学習 [16] を組み合わせた新しいアルゴリズム DSSQ（*Distributed Stochastic Search algorithm with deep Q-learning*）を提案する．具体的には，$DSSA^+$ において各エージェントが保有するコスト関数のパラメータをエージェント自らが変更することで，自身にとって長期的に効率的な経路を生成するための戦略を学習により獲得させる．DSSQ の新規性は，エージェントに全体最適と個別最適の両者を追求させる点にある．本稿では，分散型アルゴリズムが発見した短期的なシステム最適解の列が，学習によって徐々に個人最適解に近づくことを実験的に示す．

2　分散確率的探索アルゴリズム $DSSA^+$

2.1　フレームワーク

本章では，多対多の衝突回避アルゴリズムの一つである $DSSA^+$[9] について説明する．$DSSA^+$ は移動体エージェント間の継続的な意図交換をベースとしたアルゴリズムであり，$DSSA^+$ に追従する各エージェントは自身の「意図（針路変更 $\Delta\theta$ 及び速度変更 Δv）」を「検知範囲」と呼ばれるエリア内の他のエージェントと交換した情報をもとに最適化する．検知範囲内の全エージェントが自身の意図を変更しない状態に至るまで意図の更新を繰り返すため，$DSSA^+$ によって最終的に決定された衝突回避行動は，検知範囲内の局所的な系におけるシステム最適な解となる．なお，図 2 に示すように，エージェントは離散値の行動空間（*Dom*）を持ち，針路は左 45 度～右 45 度の可能舵角範囲内で 5 度刻みに，速度は加速 8kt～減速 8kt の 2kt 刻みに変更できる．また針路変更に関しては，上記の離散的な行動に加えて目標地点への針路（連続値）を選択可能である（ただし目標方向が可能舵角範囲外にある場合は選択できない）．すなわち，エージェントは合計 180 の離散行動空間から自身の意図を選択できる（$(\Delta\theta, \Delta v) \in Dom$）．

1）　神戸大学大学院海事科学研究科 Graduate School of Maritime Sciences, Kobe University
2）　木浦海洋大学 Mokpo Maritime University

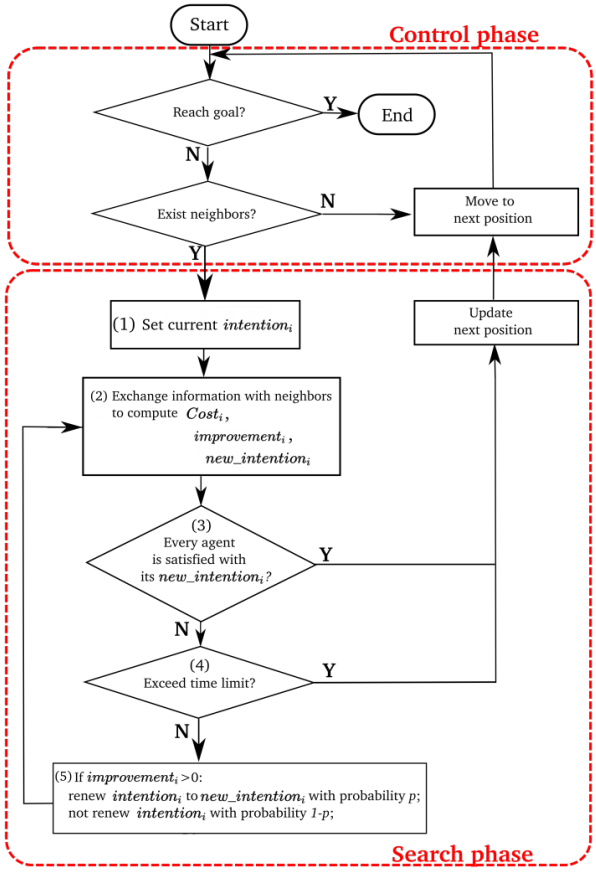


図 1: $DSSA^+$ のフローチャート

2.2 探索と制御

図 1 に $DSSA^+$ のフローチャートを示す．$DSSA^+$ は「制御フェーズ（Control Phase）」及び「探索フェーズ（Search Phase）」の 2 段階から構成される．グローバルな離散時刻 t において，エージェント i（以下 i）は制御フェーズを実行する．制御フェーズでは，i は自身の検知範囲内に他のエージェントが存在するかを確認し，存在しない場合，自身の針路・速度を変更せず時刻 $t+1$ にて次の座標へ移動する．

一方，検知範囲内にエージェントが存在する場合，i は探索フェーズを実行する．探索フェーズでは，i は検知範囲内に存在する他のエージェントの集合 $Neigh_i^{(t)}$ を観測するとともに，現在時刻における自身の針路・速度を「意図（$intention_i$）」として設定する．次に i はエージェント j （$\forall j \in Neigh_i^{(t)}$）と意図の交換を行い，自身の行動空間内の各点 $(\Delta\theta, \Delta v) \in Dom$ について「コスト $Cost_i(\Delta\theta, \Delta v)$」を計算するとともに，意図 $intention_i$ をコスト最小となる $new\_intention_i$ に確率 p で更新し，確率 $1-p$ で意図 $intention_i$ を更新せず保持する．すなわち，

$$new\_intention_i \equiv \underset{(\Delta\theta,\Delta v)\in Dom}{\arg\min}\ Cost_i(\Delta\theta, \Delta v) \quad (1)$$

$$intention_i \leftarrow \begin{cases} new\_intention_i & \text{with } p, \\ intention_i & \text{with } 1-p, \end{cases} \quad (2)$$

なお，i の意図更新は i を含む検知範囲内の全てのエージェントが自身の意図を更新しなくなるまで繰り返されるため，時刻 t において最終的に決定される行動はシステム全体で最適となる．

2.3 コスト関数

エージェント i のコスト関数は以下の式で表される．

$$Cost_i(\Delta\theta, \Delta v) \equiv \sum_{j\in Neigh_i^{(t)}} CR_i(\Delta\theta, \Delta v, j) + EF_i(\Delta\theta, \Delta v) \quad (3)$$

where

$$CR_i(\Delta\theta, \Delta v, j) \equiv \begin{cases} \frac{TimeWindow}{TCPA(\Delta\theta,\Delta v,j)} & \text{if } i \text{ collide with } j, \\ 0 & \text{otherwise,} \end{cases} \quad (4)$$

$$EF_i(\Delta\theta, \Delta v) \equiv \alpha \cdot \frac{\left|(\theta^{(t)} + \Delta\theta) - \theta_{dest}^{(t)}\right|}{180°} + (1-\alpha) \cdot \frac{\left|(v^{(t)} + \Delta v) - v_{ref}\right|}{v_{ref}} \quad (5)$$

$$(\Delta\theta, \Delta v) \in Dom \equiv \{(-45, +8kt), \cdots, (+45, +8kt), \vdots \ (-45, -8kt), \cdots, (+45, -8kt)\} \quad (6)$$

コスト関数（式 (3)）の右辺第 1 項（式 (4)）は「衝突リスク関数」と呼ばれ，i が意図を $(\Delta\theta, \Delta v)$ に変更した場合に j と衝突する危険度を $CR(\Delta\theta, \Delta v, j)$ で測るとともに，それを全ての j について合計する関数である．$CR(\Delta\theta, \Delta v, j)$ は i が j と時間窓（$TimeWindow$）内に最接近するまでの時間（*Time to Closest Point of Approach, TCPA*）の逆数を $TimeWindow$ 倍した関数である．すなわち，$TimeWindow$ 内で i が j と最接近するまでの時間が短ければ短いほど（衝突の危険性が高いほど）$CR_i(\Delta\theta, \Delta v, j)$ の値は大きくなり，逆に長ければ長いほど小さくなる．

右辺第 2 項（式 (5)）は「非効率性関数」と呼ばれ，i が意図を $(\Delta\theta, \Delta v)$ に変更した場合の移動効率性を測る関数である．なお，$\theta_{dest}^{(t)}$ は i の目標地点への針路，v_{ref} は基準速度である．この関数は行動空間（Dom）内の各点 $(\Delta\theta, \Delta v)$ と，i にとって最も望ましい行動 $(\theta_{dest}^{(t)}, v_{ref})$ との差を針路・速度それぞれについて測る．すなわち，意図変更後の針路 $\theta^{(t)} + \Delta\theta$ が目標方向に近ければ近いほど，また速度 $v^{(t)} + \Delta v$ が基準速度に近ければ近いほど，$EF_i(\Delta\theta, \Delta v)$ の値は小さくなり，逆に遠ければ遠いほど大きくなる．また，EF_i は，「速度変更優先度パラメータ」α によってパラメタライズされている（$0 < \alpha < 1$）．α が大きければ大きいほど，速度変更の非効率性（EF の右辺第 2 項）が針路変更の非効率性（EF の右辺第 1 項）に対し相対的に小さくなるため，i は意図変更の際に速度変更を優先しやすくなる．逆に α が小さければ小さいほど，i は針路変更を優先しやすくなる．

2.4 システム最適と個人最適

前章の通り，$DSSA^+$ は各時刻 t において全てのエージェントの行動を意図交換に基づき最適化するため，スタート地点からゴール地点までエージェント i が通る軌

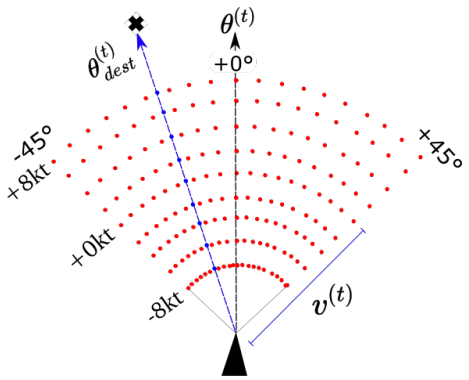


図 2: エージェントの行動空間（Dom）

跡はシステム最適な解の系列となる．しかしながら，このようなシステム最適な経路は，エージェント視点から見れば，極端に歪曲したり加減速を繰り返したりといった，長期的に非効率な経路となり得る．これは $DSSA^+$ が時刻 t におけるシステムの状態を短期的にしか最適化できないからである．言い換えると，i が時刻 t において追従する短期的なシステム最適解は，時刻 $t+1$ からゴールまでに変化するシステムの状態に対する i の経路の効率性，すなわち長期的な個人最適性を保証できないことが原因である．

本稿では，$DSSA^+$ による短期的なシステム最適化の下で，各エージェント i が自身の経路を長期的に最適化する新しいアルゴリズムの開発を目指す．そこで，時刻 t における $DSSA^+$ の短期的なシステム最適解 $(\theta^{(t)}, v^{(t)})$ と長期的な個人最適解 $(\theta^{(t)}_{dest}, v_{ref})$ との乖離度を「損失（loss）」として以下に定義する：

$$loss_i^{(t)} \equiv \frac{\left|\theta^{(t)} - \theta^{(t)}_{dest}\right|}{C_1} + \frac{\left|v^{(t)} - v_{ref}\right|}{C_2} \tag{7}$$

上記の損失がスタート（$t=0$）からゴール（$t=T$）まで蓄積されることでエージェント i の経路の非効率性が増すと考えられる．そこでエージェント i は自身の損失の総和 L_i の最小化を目指す．

$$L_i \equiv \sum_{t=0}^{T-1} loss_i^{(t+1)} \tag{8}$$

筆者らは，この最小化問題を解くアルゴリズムとして，深層強化学習を使用したアプローチを提案する．

3 アプローチ

3.1 強化学習

強化学習（Reinforcement Learning）[18] は，教師あり学習（Supervised Learning），教師なし学習（Unsupervised Learning）と並ぶ機械学習手法の一つであり，意思決定主体である「エージェント」と「環境」との相互作用に基づく逐次意思決定過程問題である．特に，強化学習はマルコフ決定過程（Markov Decision Process, MDP）に基づき，MDP は S, A, P, R, γ の 5 つの要素から成るタプルで記述される．ここで S は環境の取りうる状態の集合（状態空間），A はエージェントの取りうる行動の集合（行動空間），P は環境の状態遷移確率，R はエージェントの報酬を与える関数（報酬関数），γ は割引率である（$0 \leq \gamma \leq 1$）．

強化学習の目的は，エージェントの利得 G の最大化であり，特に利得 G には割引報酬和（discounted sum of rewards）が用いられる（$G = \sum_{t=0}^{T-1} \gamma^t r^{(t+1)}$）．ここで γ が大きければ大きいほど，報酬の長期的な最大化が考慮され，逆に γ が小さいほど，報酬の短期的な最大化が考慮される（本研究では $\gamma = 0.99$ とした）．

ところで，本研究におけるエージェントの目的は損失の総和（式 (8)）の最小化であるが，ここで負の損失（$-loss$）」を考えると，$-L_i$ の最大化問題として扱うことができる．そこで本稿では，$-loss$ を報酬，その総和 $-L_i$ を利得とした強化学習アルゴリズムを導入することでこの最大化を図る．ただし，衝突回避は $DSSA^+$ が行うため，強化学習の目的は衝突回避戦略の獲得そのものではなく，$DSSA^+$ におけるコスト関数のパラメータの値調整であることに注意されたい（次節にて詳述）．

本稿では，特に状態空間が連続の場合に用いられる深層強化学習を導入し，その代表的手法である DQN[16] を活用する．DQN の学習原理は，以下の平均二乗誤差を最小化することで最適価値関数 $Q^*(s^{(t)}, a^{(t)})$ を推定すること（この手法は Q 学習 [22] と呼ばれる）であるが，詳しくは原論文を参照されたい [16]．

$$L(w) = \mathbb{E}\left[\left(r^{(t+1)} + \gamma \cdot \max_a Q^-(s^{(t+1)}, a) - Q(s^{(t)}, a^{(t)})\right)^2\right] \tag{9}$$

ここで w は DQN のパラメータを表し，Q^- はターゲット DQN を表す．最適価値関数 Q^* を推定すると，エージェントは原則として状態 $s^{(t)}$ にて最も価値の高い行動 $a^{(t)}$ を採用する．本稿では，$DSSA^+$ に DQN を導入したアルゴリズムを *Distributed Stochastic Search algorithm with deep Q learning* と名づけ，以下 DSSQ と称する．DSSQ の学習モデルについて，次節で説明する．

3.2 学習モデル

3.2.1 行動空間

前節で述べたように，本稿における強化学習では，衝突回避そのものではなく，$DSSA^+$ のパラメータの値を行動とする．具体的には，式 (5) の速度変更優先度パラメータ α を，各 i が検知範囲内のシステムの状態に応じて適切に設定できるように学習する（$a^{(t)} \equiv \alpha^{(t)}$）．ここで $\alpha^{(t)}$ の取りうる値は，$DSSA^+$ の原論文に準拠し，$\alpha^{(t)} \in \{0.1, 0.5, 0.9\}$ の 3 つの離散値とする．直観的には，各 i はシステムの状態に応じて「針路変更を優先するか，速度変更を優先するか」の判断を行う．

3.2.2 報酬関数

本稿ではエージェント i の負の損失（$-loss_i^{(t)}$）を報酬とする．そこで報酬関数を以下に定義する．

$$r_i^{(t)} \equiv \begin{cases} -loss_i^{(t)} & \text{if } i \text{ not collide,} \\ -10.0 & \text{otherwise,} \end{cases} \tag{10}$$

報酬関数内の C_1 と C_2 については，両項の値域が等しくなるよう適当な値を設定した．また i が衝突した場合の報酬は -10.0 と設定した．

3.2.3 状態空間

通常，深層強化学習では，エージェントが直接作用する環境の視覚情報（ゲーム画面など）を状態として扱うことが多く，本研究においてもシミュレータ画面を状態として扱うことが考えられる．強化学習に深層ニューラルネットを利用する利点の一つには，これらの生情報から状態の特徴量を抽出する表現学習と，タスクに応じた最適な方策を学習するタスク学習の両者を同時に行えることがある．しかしながら，これらを同時並行するには単一エージェントの問題例でさえ数百万フレームという多大な時間を要するほか [8]，使用可能な計算機の処理能力を考慮すると，このような大規模な学習の実行が困難な場合もある．そこで本研究では，$DSSA^+$ に従うシステムの状態の特徴量を以下のように人為的に抽出することで，表現学習の省略ないしは簡略化を試みる．

DQN は時刻 t の状態 $s^{(t)}$ を観測したのち，価値が最大となる $\alpha^{(t)}$ を出力するように学習するが，ここで価値は割引報酬和の期待値であるため，時刻 $t+1$ における報酬 $-loss_i^{(t+1)}$ を決定づける，時刻 t における変数について考える．時刻 $t+1$ における損失（式 (7)）は，i の目標針路 $\theta_{dest}^{(t+1)}$ 及び基準速度 v_{ref} と，i の現在針路 $\theta^{(t+1)}$ 及び現在速度 $v^{(t+1)}$ との差で与えられるが，$\theta^{(t+1)}$ 及び $v^{(t+1)}$ は，時刻 t にて i 及び全ての $j \in Neigh_i^{(t)}$ が意図を変更しなくなった時のコストの Dom 上の分布において値が最小となる点（式 (1)）に確率 p で等しい（式 (2)）．したがって，この時のコストの Dom 上の分布は状態の特徴量と考えることができる．

ここでコストの分布は衝突リスクの分布と非効率性の分布の和に等しいが（式 (3)），衝突リスクの分布はエージェント i 及び全ての $j \in Neigh_i^{(t)}$ の意図変更によって，時刻 t 内で変化することに注意が必要である．そこで本研究では，便宜的に，エージェントが時刻 t にて意図を変更する前の衝突リスクの分布（すなわち衝突リスクの初期分布）を状態の特徴量として利用する．一方，非効率性の分布は，速度変更優先度パラメータ α，及び i の時刻 t での現在針路と目標針路との差 $|\theta^{(t)} - \theta_{dest}^{(t)}|$，及び現在速度と基準速度との差 $|v^{(t)} - v_{ref}|$ によって一意に定まるが，前者は DQN の出力に相当するため省略し，後者を状態の特徴量として扱う．

従って本研究では，i の時刻 t における状態 $s^{(t)}$ を，i の Dom における衝突リスクの初期分布（以下 $\mathbb{CR}^{(t)}$ とする），現在針路と目標針路との差 $|\theta^{(t)} - \theta_{dest}^{(t)}|$，及び現在速度と基準速度との差 $|v^{(t)} - v_{ref}|$ からなる 182 次元のベクトルとして以下に定義する．

$$s^{(t)} \equiv \mathbb{CR}^{(t)} \cap \left\{ \left|\theta^{(t)} - \theta_{dest}^{(t)}\right|, \left|v^{(t)} - v_{ref}\right| \right\} \quad (11)$$

3.2.4 ネットワークアーキテクチャ

本稿における DQN は，状態入力に対して α の 3 つの各値（$\alpha^{(t)} \in \{0.1, 0.5, 0.9\}$）に対応する行動価値（Q 値）を出力する．ここで，状態入力のうち，衝突リスクの分布（$\mathbb{CR}^{(t)}$）からの特徴量抽出には，畳み込みニューラルネット（CNN）を適用する．画像の特徴量抽出に効果的な CNN を用いることで，図 2 に示す Dom 上に分布する衝突リスクから，検知範囲内の状態の特徴量を効果的に抽出できると期待できる．一方，i の目標針路及び速度は，CNN から出力された特徴量ベクトルと結合され，全結合層（FC）に入力される．本稿では，CNN はカーネルサイズ 2 の畳み込み層 2 層，Max プーリング層 1 層から成る単層 CNN を用いる．また，FC の層数は 5 とし，各層で 128 個の活性化ユニット（ReLU）を用いる．以上を踏まえた DQN のアーキテクチャを図 3 に示す．

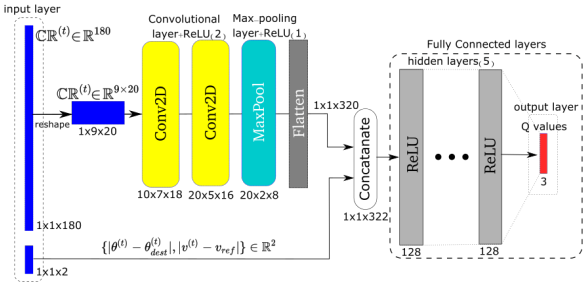


図 3: DQN のアーキテクチャ

4 実験

4.1 実験設定

実験では，エージェント数が異なる 3 つのシナリオを用意した（図 4）．これらのシナリオ及びシミュレーション環境は $DSSA^+$ の原論文 [9] に準拠しており，意図変更の確率 p（式 (2)）は 0.8 とした．各エージェントは独立した学習機構（DQN 及びリプレイバッファ D）を持つ．状態遷移タプル $(s^{(t)}, a^{(t)}, r^{(t+1)}, s^{(t+1)})$ を自身のリプレイバッファ D に保存したのち，バッチサイズ 32 のミニバッチをランダムに生成して式 (9) を適用することで DQN を最適化する．この手法は特に経験再生（Experience Replay）と呼ばれる．DQN の最適化関数には Adam を用いた．また，DQN の行動選択には探索と利用のトレードオフを解消する目的で ϵ-$greedy$ 方策を適用した．ϵ-$greedy$ 方策では，エージェント i は確率 ϵ でランダムに行動することで未経験の状態を探索し，確率 $1-\epsilon$ で Q 値が最大となる行動を選択することで，戦略の局所最適化の防止と，学習結果の活用をバランスできる．ϵ は初期値 0.9 から，500 エピソードごとに 0.1 ずつ減衰するよう設定した．実験は Intel(R) Core(TM) i9-8950HK CPU @ 2.90GHz，Python 3.8.10 で実施した．また，GPU は NVIDIA Quadro P2000，深層学習ライブラリには PyTorch 1.4.0 を利用した．

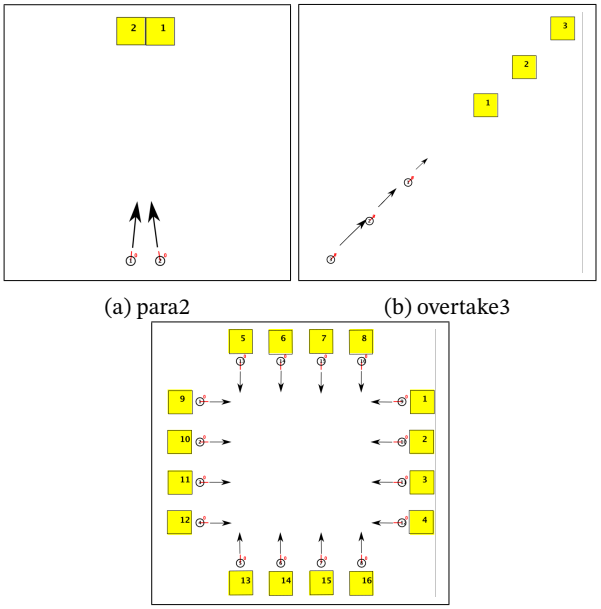
4.2 実験結果

表 1: 平均所要時間と 100 施行あたりの衝突発生数

	平均所要時間			衝突発生数		
Method	p2	ot3	c16	p2	ot3	c16
$DSSA^+(\alpha = 0.1)$	25.0	51.4	27.2	0	0	6
$DSSA^+(\alpha = 0.5)$	25.0	50.0	**26.2**	0	7	1
$DSSA^+(\alpha = 0.9)$	**22.8**	59.3	33.2	0	11	0
DSSQ	22.9	**49.6**	26.3	0	13	1

表 2: para2 と cross16 の 100 施行における所要時間の分散の最小値及び最大値

	$DSSA^+$		**DSSQ**	
シナリオ	最小値	最大値	最小値	最大値
para2	0.25	20.33	0.25	**1.00**
cross16	2.65	280.73	**1.95**	**40.06**



(a) para2　　(b) overtake3

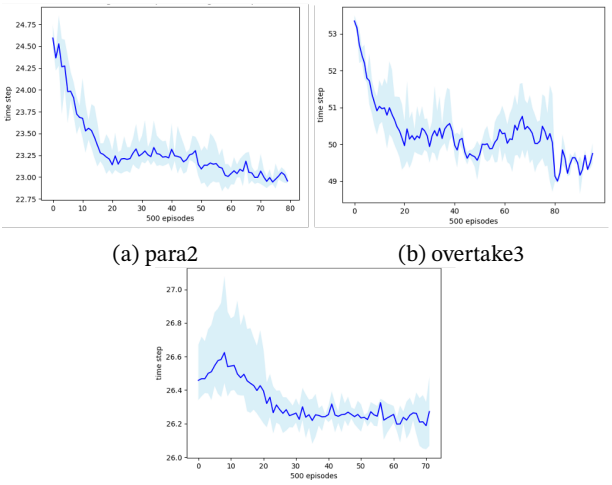
(c) cross16

図 4: 実験シナリオ．para2 （a）は交差する 2 体の系，overtake3（b）は互いに追い越す 3 体の系，cross16（c）は互いに直交する 16 体の系（para2，cross16 ではエージェントの基準速度は等しいが，overtake3 では互いに異なる）

各シナリオにおけるエージェントの平均所要時間の推移を図 5 に示す．なお各シナリオにおいて学習は 5 回実施し，図 5 における青の実線はそれらの平均を，水色の領域は最大値及び最小値間の範囲を示す．いずれのシナリオにおいても，学習が進捗するにつれてエージェントの平均所要時間が減少していることがわかる．すなわち，エージェントが検知範囲内の状態に応じて長期的に効率的な経路を生成するような速度変更優先度パラメータ α の設定方法を学習できており，$DSSA^+$ による短期的なシステム最適化の枠組みにおいても，深層強化学習によるエージェント個人の長期的な移動効率性の最大化が機能していると考えることができる．

各シナリオにおけるエージェントの平均所要時間の学習終了時の結果を表 1 に示す．DSSQ は $DSSA^+$ において α を各値で固定したどの場合よりも平均所要時間が小さいか，あるいはほぼ同じという結果となった．エージェントの基準速度が異なる overtake3 では，DSSQ は $DSSA^+$ を有意に上回ることがわかった．これは基準速度の異なるエージェントが各々の状況に特化した戦略を学習したためと考えられる．一方，para2 や cross16 など，エージェントの初期位置に対称性があり，かつ基準速度が等しいようなシナリオにおいては，DSSQ は $DSSA^+$ を有意に上回らないことがわかった．しかしながら，表 2 を見ると，DSSQ は $DSSA^+$ よりエージェントの所要時間の分散が小さいことがわかる．これは，各エージェントが個人最適なパラメータ設定方法を学習したことで，長期的な移動効率性が拮抗した結果と考えることができる．

また，DSSQ の学習終盤における衝突発生数を表 1 に示す．DSSQ は $DSSA^+$ とほぼ同等の性能で衝突回避できることがわかる．これは，DQN は $DSSA^+$ における



(a) para2　　(b) overtake3

(c) cross16

図 5: エージェントの平均所要時間の推移

コスト関数のパラメータを変更するのみであり，結局，システム全体での衝突回避は $DSSA^+$ によって最適化されるためである．このように，DSSQ にはシステム全体での安全性を担保しながら，エージェントごとの長期的な効率性も最大化できるという利点がある．

以上の結果より，分散最適化と深層強化学習の組み合せが，衝突回避タスクのような逐次意思決定過程においてシステム最適な解の系列と個人最適な解の系列の差異を縮小できる可能性が示された．

5　結論と今後の課題

本稿では，分散衝突回避アルゴリズムに深層強化学習を適用することにより，システム最適と個人最適の両立を試みる DSSQ を提案した．DSSQ では，各エージェントが，長期的に効率的な衝突回避経路を生成するようコスト関数のパラメータに対する最適な値を独自の DQN を通して獲得する．シミュレーション実験では，エージェント数体の簡単なシナリオだけでなく，より複雑なシナリオにおいても，エージェント個人の移動効率の最大化とシステム全体の最適性の両立を示唆する結果が確認された．本研究は，システム最適と個人最適のギャップの問題に分散協調問題解決の立場から取り組んだ新しい試みである点を最後に強調しておきたい．

今後の課題は以下の通りである．一般に，マルチエージェントシステムでは，各エージェントがシステムの完全な状態を観測できないという不完全知覚問題があり，本来的には部分観測マルコフ決定過程（POMDP）によるモデル化が妥当である．本稿では，簡単化のため衝突回避問題を近似的に MDP と見なして DQN を適用したが，実験的にはともかく理論的な収束性等は何ら保証されない．今後は，POMDP を想定したより発展的なアルゴリズムやマルチエージェント強化学習（MARL）アルゴリズム等を導入し，アルゴリズムの更なる性能向上を図りたい．

参考文献

[1] Javier Alonso-Mora, Andreas Breitenmoser, Martin Rufli, Paul Beardsley, and Roland Siegwart. Optimal reciprocal collision avoidance for multiple non-holonomic robots.

In *Distributed autonomous robotic systems*, pages 203–216. Springer, 2013.

[2] Yu Fan Chen, Miao Liu, Michael Everett, and Jonathan P. How. Decentralized non-communicating multiagent collision avoidance with deep reinforcement learning. In *2017 IEEE international conference on robotics and automation (ICRA)*, pages 285–292. IEEE, 2017.

[3] Soon-Jo Chung, Aditya Avinash Paranjape, Philip Dames, Shaojie Shen, and Vijay Kumar. A survey on aerial swarm robotics. *IEEE Transactions on Robotics*, 34(4):837–855, 2018.

[4] Michael Everett, Yu Fan Chen, and Jonathan P. How. Motion planning among dynamic, decision-making agents with deep reinforcement learning. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3052–3059. IEEE, 2018.

[5] Tingxiang Fan, Pinxin Long, Wenxi Liu, and Jia Pan. Distributed multi-robot collision avoidance via deep reinforcement learning for navigation in complex scenarios. *The International Journal of Robotics Research*, 39(7):856–892, 2020.

[6] Ariel Felner, Roni Stern, Solomon Shimony, Eli Boyarski, Meir Goldenberg, Guni Sharon, Nathan Sturtevant, Glenn Wagner, and Pavel Surynek. Search-based optimal solvers for the multi-agent pathfinding problem: Summary and challenges. In *International Symposium on Combinatorial Search*, volume 8, 2017.

[7] Daniel Hennes, Daniel Claes, Wim Meeussen, and Karl Tuyls. Multi-robot collision avoidance with localization uncertainty. In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems - Volume 1*, AAMAS '12, pages 147–154, Richland, SC, 2012. International Foundation for Autonomous Agents and Multiagent Systems.

[8] Matteo Hessel, Joseph Modayil, Hado Van Hasselt, Tom Schaul, Georg Ostrovski, Will Dabney, Dan Horgan, Bilal Piot, Mohammad Azar, and David Silver. Rainbow: Combining improvements in deep reinforcement learning. In *Thirty-second AAAI conference on artificial intelligence*, 2018.

[9] Katsutoshi Hirayama, Koki Miyake, Tomohiro Shiota, and Tenda Okimoto. DSSA+: Distributed collision avoidance algorithm in an environment where both course and speed changes are allowed. *TransNav, the International Journal on Marine Navigation and Safety of Sea Transportation*, 13:117–124, 03 2019.

[10] Yamin Huang, Linying Chen, Pengfei Chen, Rudy R. Negenborn, and P.H.A.J.M. van Gelder. Ship collision avoidance methods: State-of-the-art. *Safety Science*, 121:451–473, 2020.

[11] Dong-Gyun Kim, Katsutoshi Hirayama, and Gyei-Kark Park. Collision avoidance in multiple-ship situations by distributed local search. *Journal of Advanced Computational Intelligence and Intelligent Informatics*, 18(5):839–848, 2014.

[12] Donggyun Kim, Katsutoshi Hirayama, and Tenda Okimoto. Ship collision avoidance by distributed tabu search. *TransNav, the International Journal on Marine Navigation and Safety of Sea Transportation*, 9:23–29, 03 2015.

[13] Donggyun Kim, Katsutoshi Hirayama, and Tenda Okimoto. Distributed stochastic search algorithm for multi-ship encounter situations. *Journal of Navigation*, 70(4):699–718, 2017.

[14] Markus Kuderer, Henrik Kretzschmar, Christoph Sprunk, and Wolfram Burgard. Feature-based prediction of trajectories for socially compliant navigation. In *Robotics: science and systems*, 2012.

[15] Shijie Li, Jialun Liu, and Rudy R. Negenborn. Distributed coordination for collision avoidance of multiple ships considering ship maneuverability. *Ocean Engineering*, 181:212–226, 2019.

[16] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Hassabis Demis. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

[17] Mike Phillips and Maxim Likhachev. Sipp: Safe interval path planning for dynamic environments. In *2011 IEEE International Conference on Robotics and Automation*, pages 5628–5635, 2011.

[18] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. A Bradford Book, Cambridge, MA, USA, 2018.

[19] Sarah Tang, Justin Thomas, and Vijay Kumar. Hold or take optimal plan (hoop): A quadratic programming approach to multi-robot trajectory generation. *The International Journal of Robotics Research*, 37(9):1062–1084, 2018.

[20] Jur van den Berg, Stephen J. Guy, Ming Lin, and Dinesh Manocha. Reciprocal n-body collision avoidance. In Cédric Pradalier, Roland Siegwart, and Gerhard Hirzinger, editors, *Robotics Research*, pages 3–19, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

[21] Qichen Wang and Chris Phillips. Cooperative collision avoidance for multi-vehicle systems using reinforcement learning. In *2013 18th International Conference on Methods Models in Automation Robotics (MMAR)*, pages 98–102, 2013.

[22] Christopher JCH Watkins and Peter Dayan. Q-learning. *Machine learning*, 8(3-4):279–292, 1992.

[23] Zhi Yan, Nicolas Jouandeau, and Arab Ali Cherif. A survey and analysis of multi-robot coordination. *International Journal of Advanced Robotic Systems*, 10(12):399, 2013.

[24] Weixiong Zhang, Guandong Wang, Zhao Xing, and Lars Wittenburg. Distributed stochastic search and distributed breakout: properties, comparison and applications to constraint optimization problems in sensor networks. *Artificial Intelligence*, 161(1):55–87, 2005.