「マルチメディア，分散，協調とモバイル
(DICOMO2021)シンポジウム」 令和3年6月-7月

パフォーマンス低下を抑制するオンライン3D シューティングゲーム向け遅延補償に関する一考察

赤間 俊介[1] 本生 崇人[1] 石岡 卓将[1] 藤橋 卓也[1] 猿渡 俊介[1] 渡辺 尚[1]

概要：ネットワーク遅延の増大はオンラインゲームにおけるプレイヤパフォーマンスの低下，各プレイヤに対する不公平なゲーム体験を招く．本稿ではオンライン 3D シューティングゲームを対象として，ネットワーク遅延がプレイヤのパフォーマンスにもたらす影響を軽減する遅延補償技術を提案する．より具体的には，これまでに受信した相手プレイヤの位置を入力とする 2 次関数による回帰曲線を用いることでネットワーク遅延下にある相手プレイヤの位置情報を推定する遅延補償技術とゲーム画面から得られた深度画像を入力とした深層強化学習を用いてネットワーク遅延下にある相手プレイヤの行動を推定する遅延補償技術を提案する．提案手法では，オンライン 3D シューティングゲーム ViZDoom を用いて遅延補償技術による効果を評価した．評価結果から，提案手法を用いてネットワーク遅延下にある相手プレイヤのゲーム情報を補償することでプレイヤのパフォーマンスにつながる銃弾命中率の低減を抑制できることが分かった．

1. はじめに

家庭へのブロードバンドネットワークの普及を通して遠隔のプレイヤ同士がネットワークを介して共通のゲームを協力プレイ・対戦プレイするオンラインゲームへの需要が高まっている．ネットワークを介した協力プレイ・対戦プレイは esports に挙げられる新たなエンタテイメントの創出や，教育を含む様々な分野への応用が期待されている．一般的にオンラインゲームは，サーバに対して複数プレイヤがネットワークを介して同時に接続するクライアント・サーバ型とプレイヤ同士が相互に接続するピアツーピア型とに分けられる．多くのオンラインゲームが導入しているクライアント・サーバ型のオンラインゲームでは，各プレイヤが逐次的に自身のゲーム情報をサーバに送信する．受信した各プレイヤのゲーム情報にしたがってサーバは各種処理を実施するとともに，他プレイヤのゲーム情報を各プレイヤに伝送する．例えば，シューティングゲーム Doom では，自身の操作情報と銃弾の命中結果を各プレイヤはサーバに対して送信するとともに，サーバから受信した他プレイヤの操作情報を元にして自身の画面を更新する．このとき，サーバから他プレイヤが発射した銃弾の命中結果が届いた場合，各プレイヤは自身の体力を減少させる処理を実施する．

このとき，サーバと各プレイヤ間のネットワークではパケット損失や輻輳に起因して，たびたび遅延が増大する．より具体的には，オンラインゲームでやり取りするパケットはプレイヤの位置情報や操作情報など少量のデータから構成されるため，伝送中にパケットが損失した場合，パケットを再送することで損失による影響を緩和することができる．一方で，再送に起因してゲーム情報を受信するまでの遅延が増加する．

オンラインシューティングゲームにおけるネットワーク遅延の増大は各プレイヤに対して不公平なゲーム体験をもたらすとともに，ゲームパフォーマンスに影響を及ぼす [1–3]．例えば，遅延が増大するにしたがって，遅延下にあるプレイヤの画面上には他プレイヤの過去の位置情報が表示されることとなる．このとき，過去の位置情報に対して遅延下にあるプレイヤが銃弾を発射した場合，他プレイヤの現在位置に関わらず，銃弾は命中する．

ネットワーク遅延に起因するプレイヤパフォーマンスへの影響を抑制するために，サーバ側でゲーム情報の差異を補償する手法が提案されている．本手法では，サーバが各プレイヤに対する銃弾命中を判定するとき，最新の位置情報ではなく，ネットワーク遅延量に相当する過去の位置情報を命中判定に利用する．サーバ側での遅延補償はサーバ上でゲーム情報の差異を補償することでパフォーマンス低下の抑制を実現する．一方で，各プレイヤが保持する他プレイヤのゲーム情報と他プレイヤの実際のゲーム情報との差異は含まれたままとなる．各プレイヤが命中判定をするオンラインシューティングゲームにおいては，各プレイヤが保持する過去の位置情報に基づいて命中判定を実施する

[1] 大阪大学大学院情報科学研究科

ため，ネットワーク遅延によるプレイヤパフォーマンスへの影響を防ぐことが困難となる．

本稿では，各プレイヤが過去のゲーム情報から他プレイヤのゲーム情報を予測する遅延補償技術を提案する．より具体的には，プレイヤが保持する他プレイヤの過去の位置情報および 2 次関数による回帰曲線を用いて遅延下における他プレイヤの位置情報を推定する手法，ゲーム画面のスナップショットから得られた深度画像を入力とする深層強化学習を用いて遅延下における他プレイヤの行動を推定する手法をそれぞれ提案する．提案手法を用いてネットワーク遅延下にある他プレイヤのゲーム情報をより正確に再現することができれば，プレイヤ間に生じるゲーム情報の差異を軽減できるため，ネットワーク遅延に起因するパフォーマンス低下を抑制できると考えられる．

性能評価では，Doom を元にした 3D オンラインシューティングゲーム ViZDoom を用いて提案手法による遅延補償効果を評価した．より具体的には，ネットワーク遅延下にある相手プレイヤの実際の位置情報と提案手法によって推定した相手プレイヤとの位置情報とを比較するとともに，遅延補償技術による命中率への影響を評価した．評価結果から，提案手法によってネットワーク遅延下にある相手プレイヤの位置情報をより正確に推定できるとともに，ネットワーク遅延による命中率の低減を抑制できることが分かった．一方で，高遅延下における推定精度はまだ十分でないことが明らかとなったため，提案手法のさらなる改良が必要であることも分かった．

本稿の構成は以下の通りである．2 節では本研究と関連するオンラインシューティングゲームを対象としたサーバ側での遅延補償技術，クライアント側での遅延補償技術，模倣学習について述べる．3 節では，提案手法である 2 次関数に基づく回帰曲線を用いた遅延補償技術および深層強化学習を用いた遅延補償技術について述べる．4 節では性能評価で使用した 3D ゲームである ViZDoom について述べるとともに，提案手法による効果を評価する．最後に 5 節において結論を述べる．

2. 関連研究

本研究はオンラインシューティングゲームを対象としたサーバによる遅延補償技術，クライアント端末による遅延補償技術，模倣学習に関する研究と関連する．

2.1 サーバでの遅延補償技術

サーバ側での遅延補償技術として命中判定時のゲーム情報を補償する手法 [4,5]，ゲーム情報を更新するたびに補償する手法 [6,7] が提案されている．

[5] では，サーバが各プレイヤの過去の位置を保持しておくとともに，各プレイヤに対する命中判定時にネットワーク遅延分遡った過去の位置情報を利用する．過去の位置情報を利用することで，ネットワーク遅延に起因するプレイヤパフォーマンス，すなわち，命中率の低下を抑制できる．しかしながら，各プレイヤが保持するゲーム情報を補償しないため，プレイヤ間でゲーム情報の差異が生じる．ゲーム情報の差異はプレイヤ間での不公平なゲーム体験をもたらす．[4] では，プレイヤ間に残されたゲーム情報の差異に起因する不公平なゲーム体験を抑制する．より具体的には，サーバから通知された命中判定結果をそのまま適用せず，各プレイヤが保持する位置情報にしたがって判定結果が妥当であったかを確認するとともに，プレイヤにとって判定結果が妥当でない場合は再確認を要望するパケットを位置情報を含むパケットをサーバに送信する．サーバは送られてきた位置情報を元に過去の位置情報を更新し，更新された位置情報と再確認メッセージを元にして再度命中判定を実施するとともに，命中判定に誤りがあった場合には判定結果を差し戻す．[6] では，ネットワーク遅延を最も体感しているプレイヤに合わせてサーバがゲーム情報を更新する．本手法はすべてのプレイヤ間でゲーム情報を同期することが可能となる一方，短いネットワーク遅延を体感しているプレイヤはゲーム情報の更新を待機する必要があるため，リアルタイム性が求められるシューティングゲームには適さない．

2.2 クライアントでの遅延補償技術

クライアント側での遅延補償技術として Dead Reckoning (DR) が提案されている [8]．本手法では，各プレイヤは位置情報 $\boldsymbol{r}(t) = (x(t), y(t), z(t))$，速度ベクトル $\boldsymbol{v}(t)$ から構成される DR ベクトルを他のプレイヤに対して送信する．各プレイヤは他プレイヤから受信した DR ベクトルを用いることで他プレイヤの将来の位置を推定する．各プレイヤはすでに送信した DR ベクトルと現在の状況から得られる DR ベクトルとの差分が大きくなったとき，新たな DR ベクトルを他のプレイヤに対して送信する．文献 [9,10] では，加速度ベクトルを DR ベクトルに追加する Motion-Aware Adaptive Dead Reckoning (MAADR) を提案して推定精度の向上を図っている．より具体的には，各プレイヤによる行動を曲率を用いて定常運動・等加速度運動・可変加速運動・曲線運動に分類した後，分類結果に応じて位置情報を推定する．

2.3 模倣学習

模倣学習とは教示者の行動と設定した報酬関数を元にして，報酬の期待値が最大化するように教示者の行動を模倣する技術である．模倣学習を実現する方法として大きく 2 通りの方法が提案されている．1 つ目は教師あり学習を用いて模倣対象の行動を学習する Behavioral Cloning である [11–13]．Behavioral Cloning では教示者の行動を教師データとして与えて学習したモデルにしたがって各状態か

ら教示者の行動を模倣する次の行動を推定する．2 つ目は模倣対象の行動を入力として報酬関数を推定する逆強化学習 [14–16] と強化学習 [17,18] を組み合わせて教示者の行動を推定する方法である．本手法は教師データを必要としない一方で，逆強化学習で得られた報酬関数を用いた強化学習が必要となる．逆強化学習と強化学習の組み合わせによる学習時間の増大を抑制するために Generative Adversarial Imitation Learning (GAIL) [19] が提案されている．GAIL では，Generative Adversarial Network(GAN) [20,21] を元にして模倣対象である教示者の行動から報酬関数を介せず教示者の行動と類似した行動を推定する．

2.4 本研究の貢献点

本研究ではオンラインシューティングゲームを対象として 2 次関数に基づく回帰曲線を用いた遅延補償技術および深層強化学習を用いた遅延補償技術をそれぞれ提案する．2.2 節で示したクライアントでの遅延補償技術と同様に，提案した遅延補償技術はネットワーク遅延下にある他プレイヤの位置情報あるいは行動を各プレイヤが推定する．

本研究の貢献点は以下の 3 点である．

(1) DR とは異なり，提案手法は各プレイヤが他プレイヤの位置情報を推定するため，オンラインシューティングゲームにおけるパケット通信に影響を及ぼさない点．

(2) 教師あり学習を元にした Behavioral Cloning とは異なり，ゲーム画面のスナップショットを入力とする深層強化学習を用いて相手プレイヤの行動を推定する点．

(3) 奥行き方向の行動を推定するため，深度画像を入力とした深層強化学習を用いた点．

3. 提案手法

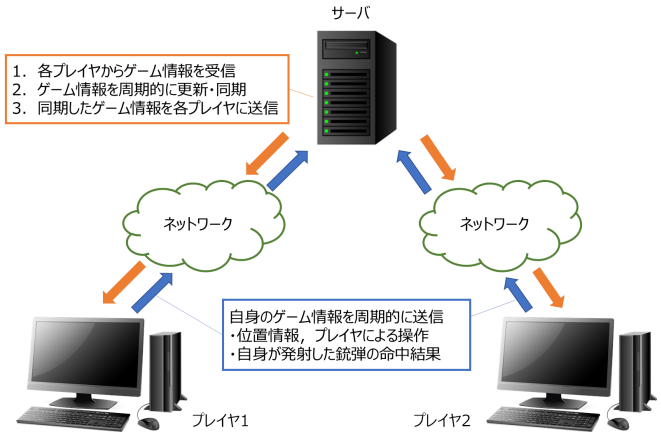
3.1 全体像

図 1 (a) に，本稿で想定するサーバと各プレイヤ間のやりとりを示す．各プレイヤは自身の位置および操作，命中判定結果をサーバに対して定期的に送信する．サーバは各プレイヤから受信したデータを元にゲーム情報をゲーム内時間 tic にしたがって周期的に更新するとともに，更新後の各プレイヤのゲーム情報を各プレイヤに対して送信する．

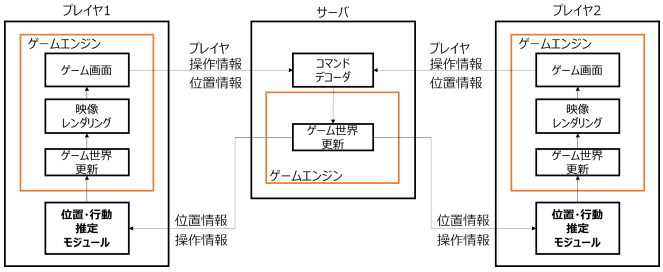
図 1 (b) に，サーバおよび各プレイヤが担当する処理の概略を示す．サーバから他プレイヤのゲーム情報を受信すると，各プレイヤは提案手法である位置・行動推定モジュールにゲーム情報を入力して他プレイヤのゲーム情報を推定する．推定した他プレイヤのゲーム情報および自身のゲーム情報を元にして各プレイヤはゲーム世界を更新するとともに，対応するゲーム映像を画面上に表示する．

3.2 2 次関数による回帰曲線を用いた遅延補償技術

本手法では，プレイヤ端末は自身が保持する相手プレイヤの過去の位置情報を入力とした 2 次関数の回帰曲線から



(a) サーバ・プレイヤ間のやりとり



(b) サーバおよびプレイヤでの処理

図 1: 提案手法が想定するクライアント・サーバ型オンライン 3D シューティングゲーム

相手プレイヤの現在位置を推定する．具体的には，直近 H 個の相手プレイヤの位置 y と時刻 t からなるパラメータの組 $(t, y) \in \mathbb{R}^H$ を用いて，最小二乗法によって 2 次関数の回帰曲線を導出する．回帰曲線に用いる 2 次関数 $f(t)$ は次式のとおり定めた．

$$f(t) = a_1 t^2 + a_2 t + a_3 \tag{1}$$

ここで，a_i はそれぞれ求めるべきフィッティング係数である．フィッティング係数は次式に示す i 番目のパラメータの組 (t_i, y_i) に対する残差のニ乗の総和を最小化する値から取得する．

$$\underset{a_1,a_2,a_3}{\arg\min} = \sum_{i=1} \{y_i - f(t_i)\}^2 \tag{2}$$

本稿では python ライブラリである polyfit を用いてフィッティング係数を算出する．その後，得られた回帰曲線，プレイヤが保持する相手プレイヤの位置情報，サーバ・プレイヤ端末間のネットワーク遅延の大きさにしたがって，遅延時間経過後に相手プレイヤの位置情報を推定する．提案手法では遅延時間の間，相手プレイヤが回帰曲線にしたがってフィールドを移動し続けているものと仮定する．このとき，相手プレイヤは得られた回帰曲線と現在の位置から遅延時間後に移動可能な位置との交点にいると予想できる．相手プレイヤが 1 (tic) あたりに移動できるピクセル

数を d (px)，ネットワーク遅延によってプレイヤ端末上に表示されている相手プレイヤの位置情報が N (tic) 前の位置情報であると仮定すると，相手プレイヤが N (tic) 後にいる位置は，現在の相手プレイヤの位置を中心とする半径 dN の円上に相当する．半径 dN の円が得られた後，プレイヤ端末は回帰直線と半径 dN の円との交点を相手プレイヤの位置と推定してフィールド上に反映する．

このとき，相手プレイヤの位置情報を推定するパラメータ N はネットワーク遅延の大きさに応じて適切に定める必要がある．ネットワーク遅延に対して設定した N が小さすぎる，あるいは，大きすぎる場合，相手プレイヤの位置を正確に推定できず，命中率低下に起因するプレイヤのパフォーマンス低下を招く．一方で，ネットワーク遅延に対して設定した N 次第では偏差撃ちと同じ現象がプレイヤの実力に関係なく発生してしまうため，プレイヤが持つ本来のパフォーマンス以上の命中率を招く可能性もある．偏差撃ちとは，オンラインシューティングゲームに長けたプレイヤが相手プレイヤに攻撃するとき，移動方向・移動速度・銃弾速度を元に，あえて相手プレイヤの位置から少しずれた位置に射撃して銃弾を命中させる技術である．提案手法では，評価に用いた 3D シューティングゲーム ViZDoom におけるゲーム情報の更新頻度が 35 (tic/s) であることを元にして $N = 35l$ と定めた．ここで，l (s) はサーバ・クライアント間の遅延量である．

3.3 深度画像および深層強化学習に基づく遅延補償技術

本手法では，各プレイヤのゲーム画面から得られた深度画像を入力とする深層強化学習を用いてネットワーク遅延下にある相手プレイヤの操作を推定する．ここで，時刻 t におけるゲーム画面の深度画像を f_t，同時刻におけるプレイヤ i による操作を a_i^t とする．ある時刻 t における相手プレイヤの行動 a_i^t を推定するために，提案手法では直近 M 枚分のゲーム画面の深度画像 $\{f_{t-M}, ..., f_{t-1}\}$ を入力とした．各深度画像は 240×180 画素のグレースケール画像とした．

図 2 に提案手法のネットワーク構造を示す．入力したゲーム画面の深度画像に対して畳み込み層と活性化関数 ReLU (Rectified Linear Unit) を 3 回繰り返し用いてゲーム画面内・ゲーム画面間の特徴を捉える．ここで，ReLU は負の値を 0 とする活性化関数 $f(x) = \max(0, x)$ である．その後，畳み込み層が出力した 3 次元テンソルを Reshape 層で 1 次元テンソルに変換する．得られた 1 次元テンソルは半分に分割してそれぞれを状態価値関数・アドバンテージ関数の学習に用いる．最後に，状態価値関数から得られた出力とアドバンテージ関数の出力を結合した状態行動価値関数から相手プレイヤによる操作 $\hat{a}_i^t$ を推定する．各プレイヤは推定した相手プレイヤの操作にしたがってゲーム画面 f_t を生成して表示する．

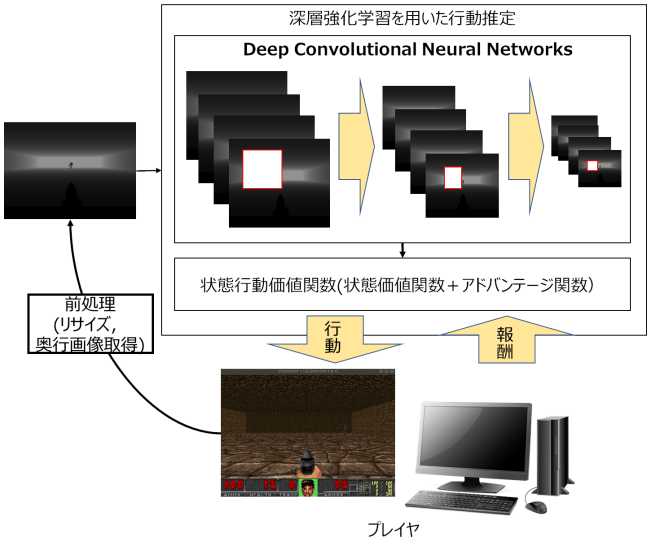


図 2: 深層強化学習を用いたプレイヤの行動推定

ここで，相手プレイヤと同じ操作を推定可能な学習済モデルを取得するために，報酬関数を次式のとおり定めた．

$$l(\hat{a}_i^t, a_i^t, a_i^{t-1}) = \begin{cases} 10, & \hat{a}_i^t = a_i^t, a_i^t \neq a_i^{t-1} \\ 0, & \hat{a}_i^t = a_i^t, a_i^t = a_i^{t-1} \\ -1, & \text{else} \end{cases}$$

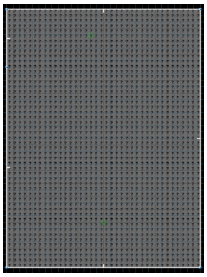
より具体的には，推定精度の低下につながる相手プレイヤによる行動変化を追従できるように報酬関数を設定している．相手プレイヤによる操作と提案手法が推定した操作が一致した場合，直近の行動と同じ行動を継続していた場合は 0，異なる行動を選択していた場合は 10 を与えるものとした．一方で，相手プレイヤによる操作と提案手法が推定した操作が一致しない場合は-1 とした．

なお，ネットワーク遅延が大きくなるにつれて，提案手法は複数手先の相手プレイヤの操作を先読みする必要がある．従来の深層強化学習では，1 手先の相手プレイヤの操作を推定することを前提としているため，ネットワーク遅延の増大に対処できない．本研究では，学習済モデルが一度推定した操作を相手プレイヤが実行し続けると仮定して相手プレイヤの操作を先読みする．例えば，t 番目および $t+1$ 番目のゲームフィールドに対する相手プレイヤの操作を推定することを考える．まず，$\{f_{t-M}, ..., f_{t-1}\}$ を入力として学習済モデルが時刻 t における相手プレイヤの操作 $\hat{a}_i^t$ を推定する．推定した相手プレイヤの操作を元にして相手プレイヤの位置を更新したゲーム画面 f_t を生成する．その後，同様に $\hat{a}_i^{t+k} = \hat{a}_i^t (k = 1, 2, ...)$ であると仮定して，相手プレイヤの位置を更新した $t+1$ 番目のゲームフィールド f_{t+1} を生成する．

4. 実験評価

提案手法による遅延補償がプレイヤパフォーマンスにもたらす影響を評価するため，3D シューティングゲーム

(a) ゲーム画面



(b) マップ構造

図 3: ViZDoom

VizDoom を用いた実験評価を実施した．

4.1 実験環境：ViZDoom

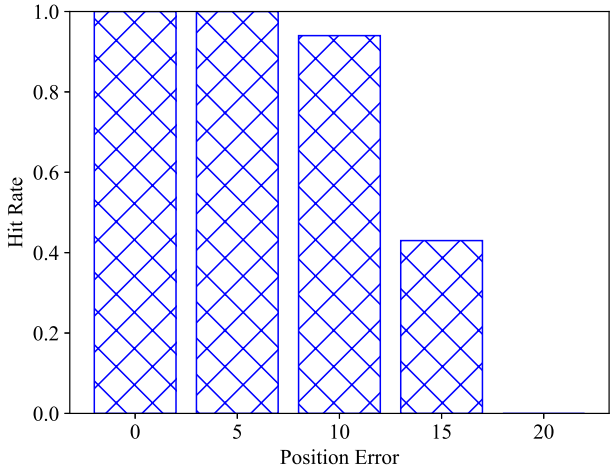
図 3 に実験評価で利用するオンライン 3D シューティングゲーム ViZDoom を示す．ゲームの種類は図 3(a) に示す一人称視点の対戦型シューティングゲーム，図 3(b) に示す障害物のない三次元のフィールド，プレイヤ数は 2 人と定めた．ViZDoom はピアツーピア型またはクライアント・サーバ型のいずれかを選択できる．クライアント・サーバ型でゲームを実行する場合，各クライアント端末で銃弾の命中判定を行う．サーバと各クライアントの端末間のネットワークには tc コマンドを使用することで任意の双方向遅延を設定するものとした．また，プレイヤの移動方法として，キーボードとマウスを用いた人間による操作が可能である．より具体的には，キーボードの↑キー，↓キー，<キー，>キーを用いて画面上のプレイヤを前後左右斜めに移動できる．

4.2 評価環境

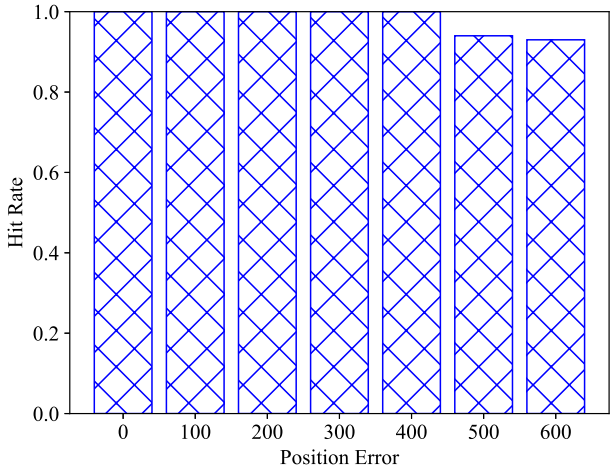
ネットワーク遅延：本評価では，tc コマンドを用いることでサーバに接続する 2 プレイヤ端末のうち，一方のプレイヤとサーバとの間に 50 (ms)，100 (ms)，200 (ms)，300 (ms) の 4 種類のネットワーク遅延を与えた．このとき，もう一方のプレイヤとサーバとの間のネットワーク遅延は 0 (ms) とした．

比較手法：比較手法として遅延補償を利用しない No Compensation (NC)，1 次関数による回帰直線を用いて遅延補償を実現する First-Order Regression (FOR) [22]，3.2 節に示した 2 次関数による回帰曲線を用いた遅延補償を実現する Second-Order Regression (SOR)，3.3 節に示した深層強化学習による遅延補償を実現する Deep Reinforcement Learning (DRL) を用意した．

評価指標：各遅延補償技術による効果を示す指標として，ネットワーク遅延を体感したプレイヤが保持する相手プレイヤの実際の位置情報と推定した位置情報との平均ユークリッド距離，ネットワーク遅延下にあるプレイヤが発射した銃弾に対してネットワーク遅延下にないプレイヤに命中した銃弾数の割合を示す命中率を利用する．



(a) x 軸の誤差量



(b) y 軸の誤差量

図 4: x 軸および y 軸における位置誤差に対する命中率への影響

4.3 位置情報の誤差が命中率にもたらす影響

遅延補償技術による位置推定精度がプレイヤパフォーマンスにもたらす影響を議論するために，x 軸および y 軸における位置情報の誤差が命中率にもたらす影響を評価した．本評価では，一方のプレイヤはゲーム画面上を左右に往復し続け，もう一方のプレイヤは直立し続けるものとした．直立したプレイヤは画面上に表示される相手プレイヤの推定位置が自身の位置と直線上に表示されるとき，自動的に銃弾を発射するものとした．このとき，相手プレイヤの推定位置は x 軸の誤差量 r_x，y 軸の誤差量 r_y に応じてずれるものとした．プレイヤによる銃撃は 100 回実施されるものとした．また，両プレイヤがネットワーク遅延下にない環境で全ての銃弾が命中するように，銃撃のタイミングを制御した．

図 4 に x 軸における誤差量，y 軸における誤差量に対する銃弾の命中率をそれぞれ示す．評価結果から，x 軸にお

ける誤差量 r_x は命中率に対して大きな影響を及ぼすことが分かった．一方で，y 軸における誤差量 r_y は命中率に対する影響が小さいことが分かった．より具体的には，x 軸における誤差量 r_x が 5 に至るまでは命中率が 100%を維持できる．一方で，誤差量が 10 を超えると命中率が急激に低下し，誤差量が 20 となったときには全銃弾が命中しなくなることが分かった．一方で，y 軸における誤差量 r_y は 600 まで増加しても，誤差がないときと比較して約 6%しか差がないことが分かった．

4.4 ネットワーク遅延下における位置推定精度

前節では，遅延補償技術による位置推定精度がプレイヤパフォーマンスにもたらす影響を評価した．本節では，各遅延補償技術による位置推定精度を比較することで各手法によるプレイヤパフォーマンスへの影響を議論する．

本節では，ネットワーク遅延を 0 (ms) と定めたプレイヤは定められた移動モデルにしたがってゲーム画面上を移動し続けるものとした．移動の種類は画面内を左右に往復するもの（以後，左右往復移動），画面内を時計回りに円を描いて移動するもの（以後，円移動），画面内をジグザグに往復するもの（以後，ジグザグ移動）をそれぞれ用意した．一方で，ネットワーク遅延を体感するプレイヤは各遅延補償技術を用いて相手プレイヤの位置を推定するものとした．

図 5 に各移動モデルにおけるネットワーク遅延に対する実際のプレイヤ位置と各遅延補償技術を用いて推定したプレイヤ位置との平均ユークリッド距離を示す．ここで，横軸はサーバとネットワーク遅延下にあるプレイヤ間の遅延の大きさ (ms)，縦軸は平均ユークリッド距離 (Mean Euclidean Distance: MED) を示す．また，遅延補償技術 FOR，SOR において遡る相手プレイヤの過去の位置情報の個数 H は移動の種類ごとに最適値を設定した．具体的には，左右往復移動では $H_{\mathrm{FOR}} = 2$，$H_{\mathrm{SOR}} = 4$，円移動では $H_{\mathrm{FOR}} = H_{\mathrm{SOR}} = 2$，ジグザグ移動では $H_{\mathrm{FOR}} = 2$，$H_{\mathrm{SOR}} = 3$ となった．なお，H の最適値については本節後半で議論する．

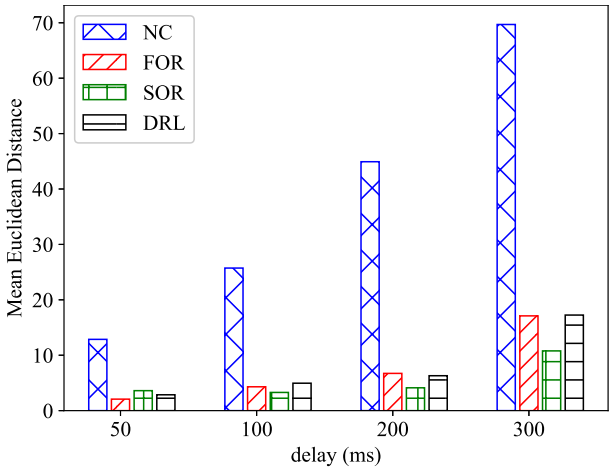
評価結果から，以下の 4 つのことが分かる．

(1) 遅延補償技術を用いない場合，自身が保持する相手プレイヤの位置情報と実際の位置情報との間の平均ユークリッド距離は著しく増加すること．

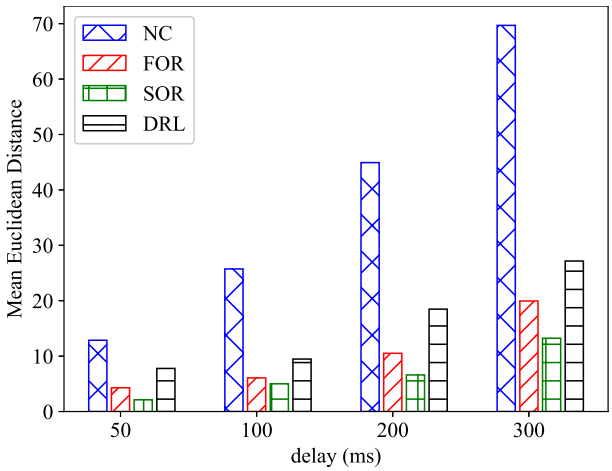
(2) ネットワーク遅延が 200 ms 以下のとき，いずれの移動モデルにおいても SOR は平均ユークリッド距離を 10 程度に低減できること．

(3) ネットワーク遅延が 300 ms まで増加すると，いずれの移動モデルにおいても SOR における平均ユークリッド距離が増大すること．

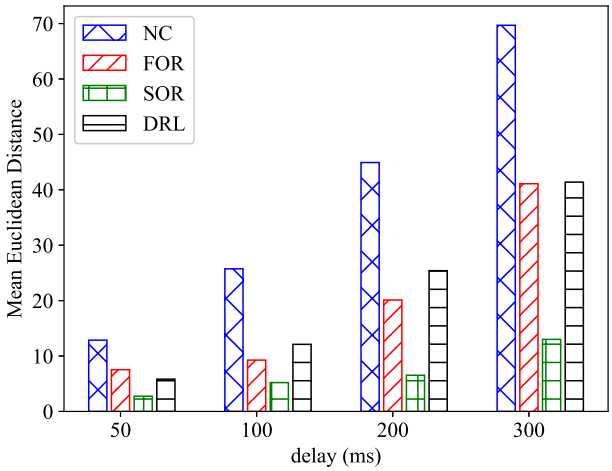
(4) DRL では，ネットワーク遅延が小さく，左右往復移動であるとき，平均ユークリッド距離を低減できる一方で，その他の移動モデルでは SOR や FOR よりも平均ユークリッド距離が増加した．



(a) 左右往復移動



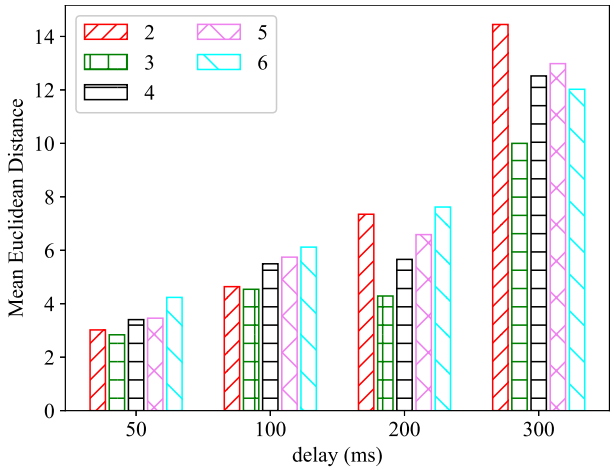
(b) 円移動



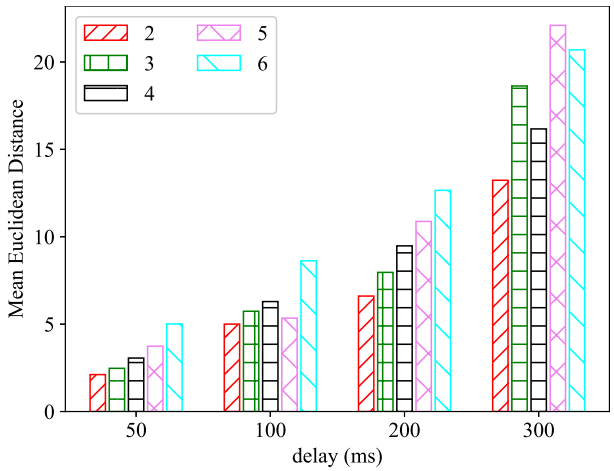
(c) ジグザグ移動

図 5: 各移動モデルにおけるネットワーク遅延に対する実際のプレイヤ位置と各遅延補償技術を用いて推定したプレイヤ位置との平均ユークリッド距離

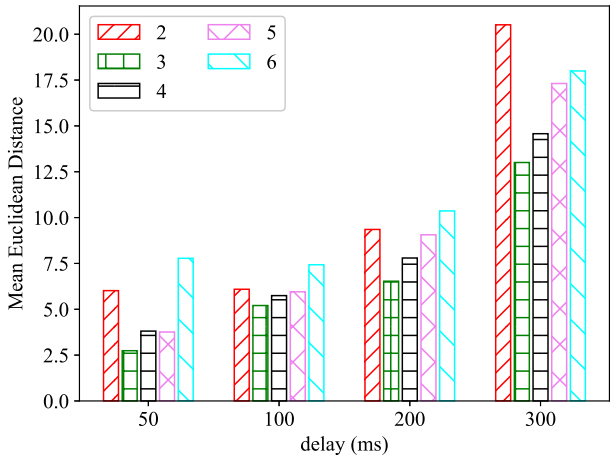
DRL における行動推定精度が低下した原因として，1) 現在のネットワーク構造は次点の行動を予測するため，ネッ



(a) 左右往復移動



(b) 円移動



(c) ジグザグ移動

図 6: 各移動モデルにおける 2 次関数による回帰曲線を用いた遅延補償技術がもたらす平均ユークリッド距離

トワーク遅延下にある移動方向の変化に対応できないこと，2) 各プレイヤの速度は一定でなく加速度に応じて変化するため，行動の予測だけでなく，行動による移動量を推定する必要があることが挙げられる．

図 6 に各移動モデルにおいて SOR に用いる過去の位置情報の個数 H_{SOR} を変更したときの平均ユークリッド距離を示す．評価結果から，SOR では推定する相手プレイヤの傾向に応じて，利用するべき過去の位置情報を変更することで推定精度の向上が見込めることが分かる．より具体的には，左右往復移動やジグザグ移動においては直前の位置情報のみを参照する場合より，さらに以前の位置情報を利用した場合は位置情報を正しく推定できることが分かった．一方で，円移動においては直前の位置情報のみを参照する方が位置情報を正しく推定できることが分かった．

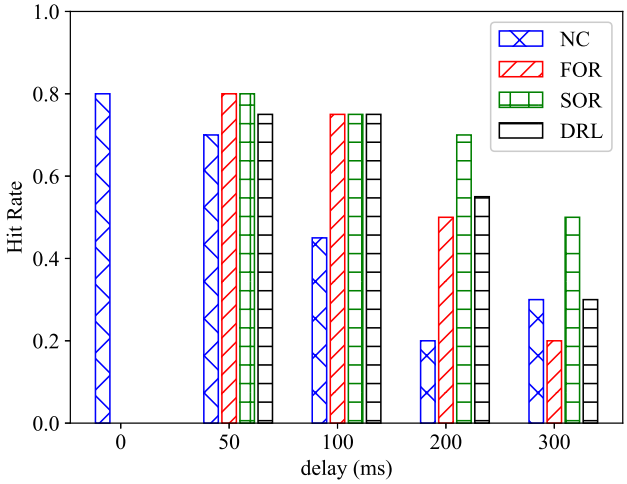


図 7: 左右往復移動におけるネットワーク遅延に対して各遅延補償技術を用いたときの命中率

4.5 左右往復移動を対象とした遅延補償技術による命中率への影響

前節の評価結果から，提案手法を用いることでプレイヤ端末上に表示される相手プレイヤの位置情報を正しく補償できることが分かった．本節では，実際にプレイヤを操作することで補償した位置情報がオンラインシューティングゲームにおけるプレイヤパフォーマンスにもたらす影響を評価する．

本評価では，ネットワーク遅延下にある ViZDoom の操作に長けたプレイヤ 1 名が画面上に表示される相手プレイヤの推定位置情報を元にして銃弾を 20 発射撃したときの命中率を取得する．このとき，相手プレイヤは左右往復移動を続けるものとした．

図 7 にネットワーク遅延下において，左右往復移動における各遅延補償技術を用いたときの命中率を示す．ここで，ネットワーク遅延が 0 (ms) であるときの命中率をネットワーク遅延下において再現するべきプレイヤパフォーマンスであると定めた．評価結果から，遅延補償技術を利用しない場合，ネットワーク遅延が 50 (ms) 時において命中率が約 7 割まで低下していることが分かる．一方で，各遅延補償技術を利用した場合，ネットワーク遅延がないときと

同等の命中率が保たれていることが分かる．低ネットワーク遅延下において命中率が維持できている理由として，4.3節で示した位置情報の誤差量が5程度であれば命中率を維持できること，4.4節で示したネットワーク遅延が50 msおよび100 msのとき，遅延補償技術導入時の平均ユークリッド距離が5程度に低減できていることが挙げられる．また，ネットワーク遅延が200 (ms)まで増加したとき，遅延補償技術FORおよびDRLを導入した場合，命中率が約5割から約6割まで低下する一方で，SORは命中率を約7割まで維持することが分かった．

5. おわりに

本稿では，オンラインシューティングゲームにおいて，ネットワーク遅延に起因するプレイヤのパフォーマンス低下を軽減する遅延補償手法を提案した．具体的には，相手プレイヤの位置情報を入力とする2次関数による回帰曲線を用いた遅延補償技術，ゲーム画面から得られた深度画像を入力とする深層強化学習を用いた遅延補償技術を用いてネットワーク遅延によって生じたプレイヤ間でのゲーム情報の差異を補償する．3Dシューティングゲーム VizDoomを用いた実験評価から，遅延補償技術によってゲーム情報の差異を低減できること，ネットワーク遅延に起因するプレイヤパフォーマンスへの影響を提案手法によって一部軽減できることがわかった．

6. 謝辞

本研究はJSPS科研費JP20K19783,JSPS科研費JP19H01101及びNTTアクセスサービスシステム研究所の支援の下で行った．

参考文献

[1] M. Claypool, and K. Claypool, “Latency and player actions in online games,” Communications of the ACM, vol.49, no.11, pp.40–45, 2006.
[2] G. Armitage, “An experimental estimation of latency sensitivity in multiplayer quake 3,” The 11th IEEE International Conference on Networks, pp.137–141, 2003.
[3] L. Pantel, and L. Wolf, “On the impact of delay on real-time multiplayer games,” Proceedings of the 12th international workshop on Network and operating systems support for digital audio and video, pp.23–29, 2002.
[4] S. Lee, and R. Chang, “Enhancing the experience of multiplayer shooter games via advanced lag compensation,” Proceedings of the 9th ACM Multimedia Systems Conference, pp.284–293, 2018.
[5] Y. Bernier, “Latency compensating methods in client/server in-game protocol design and optimization,” Proceedings of Game Developers Conference, pp.1–10, 2001.
[6] P. Bettner, and M. Terrano, “1500 archers on a 28.8: Network programing in Age of Empires and beyond,” Game Developers Conference, pp.1–15, 2001.
[7] M. Mauve, J. Vogel, V. Hilt, and W. Effelsberg, “Local-lag and timewarp: providing consistency for replicated continuous applications,” IEEE Transactions on Multimedia, vol.6, no.1, pp.47-57, 2004.
[8] S. Aggarwal, H. Banavar, A. Khandelwal, S. Mukherjee, and S. Rangarajan, “Accuracy in dead-reckoning based distributed multi-player games,” Proceedings of 3rd ACM SIGCOMM workshop on Network and system support for games, pp.161–165, 2004.
[9] V. Kharitonov, “Motion-aware adaptive dead reckoning algorithm for collaborative virtual environments,” Proceedings of the 11th ACM SIGGRAPH International Conferance on Virtual-Reality Continuum and its Applications in Industory, pp.255–261, 2012.
[10] L.F.K. de Almeida, and A.S. Felinto, “Evaluation of the motion-aware adaptive dead reckoning technique under different network latencies applied in multiplayer games,” 17th Brazilian Symposium on Computer Games and Digital Entertainment (SBGames), pp.137–145, 2018.
[11] D. Pomerleau, “Efficient training of artificial neural networks for autonomous navigation,” Neural computation, vol.3, no.1, pp.88–97, 1991.
[12] T.V. Samak, C.V. Samak, and S. Kandhasamy, “Robust behavioral cloning for autonomous vehicles using end-to-end imitation learning,” arXiv e-prints, pp.1–17, 2020.
[13] C. Wen, J. Lin, T. Darrell, D. Jayaraman, and Y. Gao, “Fighting copycat agents in behavioral cloning from observation histories,” vol.33, pp.2564–2575, 2020.
[14] A. Ng, and S. Russell, “Algorithms for inverse reinforcement learning.,” Proceedings of the Seventeenth International Conference on Machine Learning, pp.663–670, 2000.
[15] P. Abbeel, and A. Ng, “Apprenticeship learning via inverse reinforcement learning,” Proceedings of the Twenty-First International Conference on Machine Learning, pp.1–8, 2004.
[16] S. Ross, G. Gordon, and D. Bagnell, “A reduction of imitation learning and structured prediction to no-regret online learning,” Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, vol.15, pp.627–635, 2011.
[17] R. Sutton, and A. Barto, Reinforcement learning: An introduction, MIT press, 2018.
[18] G. Dulac-Arnold, D. Mankowitz, and T. Hester, “Challenges of real-world reinforcement learning,” arXiv e-prints, pp.1–13, 2019.
[19] J. Ho, and S. Ermon, “Generative adversarial imitation learning,” 30th Conference on Neural Information Processing Systems (NIPS 2016), pp.1–9, 2016.
[20] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial nets,” Proceedings of the 27th International Conference on Neural Information Processing Systems, pp.2672–2680, 2014.
[21] C. Finn, P. Christiano, P. Abbeel, and S. Levine, “A connection between generative adversarial networks, inverse reinforcement learning, and energy-based models,” 30th Conference on Neural Information Processing Systems (NIPS 2016) Workshop on Adversarial Training, pp.1–10, 2016.
[22] 本生崇人，川崎慈英，藤橋卓也，猿渡俊介，渡辺尚，“プレイヤパフォーマンス低下を抑制するオンラインゲーム向け遅延補償に関する一研究，” マルチメディア、分散、協調とモバイル (DICOMO2020) シンポジウム，pp.1458-1467，2020.