入力予測誤差最小化による相互学習インタフェース

齋藤 健太郎[1,a)]　飯塚 博幸[1,b)]　山本 雅人[1,c)]

概要：ロボットの構造が複雑になるにつれて，それを操縦するインタフェースも複雑になる．本研究では簡易なインタフェースにも関わらず，複雑な構造をもったロボットを操縦するための相互学習インタフェースを提案する．ユーザである人間と機械が相互に学習を行うことで人の学習負荷を軽減してより良いインタフェースを作り上げる．本稿では，インタフェースとしてフィードフォワードニューラルネットワークとリカレントニューラルネットワークを用いる．構造の異なるニューラルネットワークをインタフェースに用いてロボット操縦を行い結果を比較すると，リカレントニューラルネットワークを用いて相互学習を行うことで，学習に時間はかかるものの，操縦性能の高いインタフェースが構築された．

Mutual Learning Interface by Minimizing Input Prediction Error

Saitoh Kentaro[1,a)]　Iizuka Hiroyuki[1,b)]　Yamamoto Masahito[1,c)]

Abstract: The user interface becomes complex as the number of degrees of freedom of a robot increase and it becomes difficult to manipulate the robot as a user intends. To this problem, we have proposed a mutual learning interface between human and machine where the machine exploits the prediction errors to read user's intention. In this paper, we compare feed forward neural network and recurrent neural network as the learning methods on a task of multi-legged robot operation. In our results, the recurrent neural network required more time for learning but showed better performance temporally.

1. はじめに

ロボットなどの機械の操縦は，一般的に，ユーザの操作による入力信号がロボットの動作に変換されることで行われる．このようなユーザの操作からロボットの動作への変換を行うインタフェースは，事前にユーザの操作とロボットの動作を対応付けることで作成される．そのため，複雑な動作を行う多自由度のロボットにおいては，いかに操作と動作の対応関係を設計するか問題となる．また，作成されるインタフェースも複雑なものとなってしまうため，必ずしもユーザにとって使いやすいインタフェースとなる保証はない．

このような問題に対しては，つもり制御 [1] や人・機械相互学習型電動車いす [2] という人間の学習を利用する方法が考えられている．つもり制御は，機械学習を用いてロボットの動作に対して人間の直観的な操作を対応付ける方法である．また，人・機械相互学習型電動車いすでは，機械学習によりロボットの動作と人間の直観的な操作を対応付けると同時に，人間側も学習を行い人間の直観的な操作の再現性を高めることで，機械学習の精度を上げている．しかし，これらの方法では，あらかじめ決められたロボットの動作に対してユーザの操作を対応付けるため，ユーザが望むロボット動作をあらかじめ用意する必要がある．

本研究では，ユーザである人間と機械が相互に学習することで，ユーザが意図するロボットの操縦が直観的に可能となる相互学習インタフェースの開発を目指す．ユーザはあらかじめ決められた操縦方法を覚える必要はなく，ロボットの操縦を行う中で自ら学習し，その入力と動作の関係から機械も学習を行う．この学習を繰り返すことでインタフェースの更新が進み，ユーザが望むロボット動作を出力できるようになる．これにより，一定時間の学習後には

[1] 北海道大学大学院情報科学研究科
Graduate School of Information Science and Technology, Hokkaido University
[a)] saitoh@complex.ist.hokudai.ac.jp
[b)] iizuka@complex.ist.hokudai.ac.jp
[c)] masahito@complex.ist.hokudai.ac.jp

それぞれのユーザに適応したインタフェースが構築されることが期待される．このような相互学習インタフェースを実現するために，フィードフォワードニューラルネットワークをインタフェースに用いる手法 [3] を提案してきた．本稿では，情報のフィードバック結合を持つことでフィードフォワードニューラルネットワークに比べ学習は困難であるものの，入力履歴に基づいてより複雑なパターンを表現可能なリカレントニューラルネットワークを用いる手法を提案する．構造の違うニューラルネットワークを用いてロボット操縦を行い，操縦性能の違いについて議論する．

2. 相互学習によるロボット操縦

提案する相互学習インタフェースでは，ユーザがロボットの操縦を行う中で，人間の学習と機械の学習が同時に進むことでインタフェースの構築が行われる．本節では，まずロボット操縦における人間の学習と機械の学習について説明し，相互学習の枠組みについて述べる．その後，本稿の提案手法であるリカレントニューラルネットワークを用いた相互学習インタフェースについて述べる．

2.1 人間の学習

インタフェースはユーザの入力（ユーザの操作）を出力（ロボットの動作）へと変換する関数である．ユーザからある入力が与えられると，インタフェースは対応した出力を生成する．仮に，このインタフェースの入力から出力への変換関数がユーザにとって未知であったとしても，ユーザはロボットを操縦しようと試行錯誤を行う中で，ある入力に対する出力を観測できるため，インタフェースにおける入力と出力の関係を推定することができる．この推定を通して，望む出力が得られる入力を理解することが人間の学習である．

2.2 機械の学習

機械の学習とは，ユーザがある意図を持って行った入力に対して，ユーザの意図に沿う出力を生成できるようにインタフェースの変換関数を調整することである．この学習を行うためには，機械側がユーザの意図を知っていることが前提となる．ここではまず，機械がユーザの意図を知っており，ユーザの意図に沿う出力も既知であるとして説明を行う．その後，機械側がユーザの意図を知ることができない場合に，ロボットの動作がユーザの意図に沿っているかを判断し，学習を行う方法を説明する．

本研究では変換関数としてニューラルネットワークを用いる．ここでは 3 層のフィードフォワード型ニューラルネットワーク（以下 FFNN）を用いて説明を行う．FFNN において，時刻 t におけるユーザの入力を $\mathbf{X_t}$，中間層の出力を $\mathbf{H_t}$，ロボット動作のための出力を $\mathbf{Y_t}$，入力層と中間層の間の結合重みを $\mathbf{W}^{HX}$，中間層と出力層の間の結合重みを $\mathbf{W}^{YH}$ とすると，インタフェースにおける入力から出力への変換関数は式 (1),(2) と表される．

$$\mathbf{H_t} = f\left(\mathbf{W}^{HX}\mathbf{X_t}\right)\ ,\ \mathbf{Y_t} = f\left(\mathbf{W}^{YH}\mathbf{H_t}\right) \quad (1)$$

$$f\left(x\right) = \frac{1}{1+e^{-x}} \quad (2)$$

この変換関数の調整はユーザの意図に沿った出力を教師信号としたバックプロパゲーション（以下 BP）により行う．ユーザの意図に沿った出力を $\mathbf{T_t}$ とすると，学習は式 (3),(4) によって行う．このとき，η は学習率である．

$$E\left(\mathbf{W}\right) = \frac{1}{2}\left|\left|\mathbf{Y_t} - \mathbf{T_t}\right|\right|^2 \quad (3)$$

$$\Delta\mathbf{W} = -\eta\frac{\delta E\left(\mathbf{W}\right)}{\delta\mathbf{W}} \quad (4)$$

以上が，ユーザの意図が既知である場合の機械の学習である．次にユーザの意図が既知でない場合の機械の学習について説明する．ユーザの意図が既知でない場合，ユーザの意図を推定する，もしくはロボットの動作がユーザの意図に沿っているかを機械側で判断する必要がある．

ここで，丹羽らの提案しているつもり制御において，ある動作に対する直観的入力は収束していくことが示されている [1]．このことから，ロボットがユーザの意図に沿った動作を繰り返し行なっているならば，ユーザはその動作に対応する特定の入力を繰り返し行うため，入力は周期的行動になると考えられる．逆に，ロボットがユーザの意図と違う動作を行っているならば，ユーザは意図する動作を出力しようと入力を様々に変化させると考えられる．

つまり，機械の学習では，入力が周期運動である場合にはインタフェースの変換関数を変更せず，入力が周期運動から外れた場合には変換関数を変更する，という学習を行えばよい．このような学習を実現するために，変換関数であるニューラルネットワークの出力層に予測ノードを追加する．予測ノード では現在の入力値から将来の入力値への変化量を予測する．もしユーザが入力として周期運動をしているのであれば，現在の入力値から将来どのような入力値になるのかが予測可能となるため，予測誤差が小さくなる．逆にユーザが周期運動をせずに入力を様々に変化させているのであれば，予測が困難となり予測誤差が増大する．これを使ってニューラルネットワークの更新を行う．このときの入力から出力への変換関数を上記と同様に FFNN において考えると，式 (5),(6) となる．このとき，時刻 t におけるユーザの入力を $\mathbf{X_t}$，中間層の出力を $\mathbf{H_t}$，ロボット動作のための出力を $\mathbf{Y_t}$，予測のための出力を $\mathbf{P_t}$，入力層と中間層の間の結合重みを $\mathbf{W}^{HX}$，中間層と制御出力ノード $\mathbf{Y_t}$ の間の結合重みを $\mathbf{W}^{YH}$，中間層と予測ノード $\mathbf{P_t}$ の間の結合重みを $\mathbf{W}^{PH}$ とする．

$$\mathbf{H}_t\ =\ f\left(\mathbf{W}^{HX}\mathbf{X}_t\right) \quad (5)$$

$$\begin{pmatrix} \mathbf{Y_t} \\ \mathbf{P_t} \end{pmatrix} = f \begin{pmatrix} \mathbf{W}^{YH}\mathbf{H_t} \\ \mathbf{W}^{PH}\mathbf{H_t} \end{pmatrix} \quad (6)$$

この変換関数の学習は BP により行う．学習における誤差関数は式 (7) によって定義する．

$$E(\mathbf{W}) = \frac{1}{2}||\mathbf{X_t} - \mathbf{X_{t-k}} - \mathbf{P_{t-k}}||^2 \quad (7)$$

式 (7) では，現在のニューラルネットワークを用いて k ステップ前の入力から計算した予測値 $\mathbf{P_{t-k}}$ と，k ステップ前の入力値から現在の入力値への変化量との誤差を使ってニューラルネットワークの学習を行うことを意味している．この学習を行うことで更新される結合重みは，入力層と中間層の間の結合重み $\mathbf{W}^{HX}$ と中間層と予測ノード $\mathbf{P_t}$ の間の結合重みを $\mathbf{W}^{PH}$ である．入力層と中間層の間の結合重み $\mathbf{W}^{HX}$ が更新されることで，ユーザの入力に対応するロボットの動作が変更される．

以上のように予測ノードを用いることによって，もし，ロボットの動作がユーザの意図に沿っているならば，ユーザの入力は周期運動になるため，予測ノードにおける予測値の誤差は小さくなり，インタフェースの更新は行われなくなる．逆に，ロボットの動作がユーザの意図に沿っていないならば，ユーザの入力は様々に変わるため，予測ノードにおける予測誤差は大きくなり，インタフェースの更新が大きく行われることになる．

2.3 相互学習

提案する相互学習インタフェースでは上記の人間の学習と機械の学習を同時に行う．ユーザはロボットが意図する動作を行うように入力を変え，機械側はユーザの意図に沿うロボット動作を生成できるように出力を変える．これらの学習が相互に行われることで相互学習インタフェースが実現される．このとき，それぞれの学習速度が重要となる．もし，機械の学習が人間の学習に比べて早すぎると，ユーザの学習が進むことなく機械の学習だけが進んでしまうため，機械の学習が停止するとユーザはそのインタフェースを使って何もすることができない．一方，人間の学習が機械の学習に比べて早すぎると，学習によるインタフェースの更新がほとんど行われないため，人間の学習のみが行われる場合と変わらない．人間の学習と機械の学習が同じ程度の学習速度で行われることで相互学習が実現される．

2.4 リカレントニューラルネットワーク

インタフェースに FFNN を用いた場合，出力値は現在の入力値からのみ生成される．一方，リカレントニューラルネットワーク（以下 RNN）は，情報のフィードバック結合を持つため，出力値は入力値の時系列情報から生成される．そのため，RNN を用いることで FFNN に比べ，学習は困難であるものの，入力履歴に基づいてより複雑なパターンを表現可能となる．本稿では Jordan 型 RNN[4] を用いる．RNN の構造を図 1 に示す．Jordan 型 RNN は 1 ステップ前の出力層の値が文脈層にフィードバックされる RNN である．本稿では出力層におけるロボット動作のためのノードの値を文脈層にフィードバックさせる．つまり，インタフェースにおける変換関数は式 (8),(9),(10) となる．このとき，時刻 t におけるユーザの入力を $\mathbf{X_t}$，文脈層の出力を $\mathbf{C_t}$，中間層の出力を $\mathbf{H_t}$，ロボット動作のための出力を $\mathbf{Y_t}$，予測のための出力を $\mathbf{P_t}$，入力層と中間層の間の結合重みを $\mathbf{W}^{HX}$，文脈層と中間層の間の結合重みを $\mathbf{W}^{HC}$ 中間層と制御出力ノード $\mathbf{Y_t}$ の間の結合重みを $\mathbf{W}^{YH}$，中間層と予測ノード $\mathbf{P_t}$ の間の結合重みを $\mathbf{W}^{PH}$ とする．

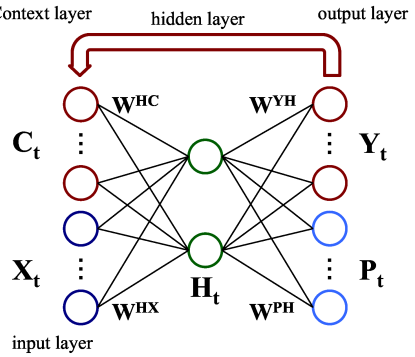


図 1 RNN の構造
Fig.1 Structure of RNN.

$$\mathbf{C_t} = \mathbf{Y_{t-1}} \quad (8)$$

$$\mathbf{H_t} = f(\mathbf{W}^{HX}\mathbf{X_t}) + f(\mathbf{W}^{HC}\mathbf{C_t}) \quad (9)$$

$$\begin{pmatrix} \mathbf{Y_t} \\ \mathbf{P_t} \end{pmatrix} = f \begin{pmatrix} \mathbf{W}^{YH}\mathbf{H_t} \\ \mathbf{W}^{PH}\mathbf{H_t} \end{pmatrix} \quad (10)$$

この RNN を用いた変換関数の学習は BackPropagation Through Time[5], [6]（以下 BPTT）により行う．BPTT とは，RNN を時間的に展開することで，時間を遡るように BP を行う学習方法である．BPTT では学習開始時まで遡って学習を行うことができるが，計算時間の問題上，一般的には遡るステップ数を決めて BPTT を行う [6]．誤差関数は FFNN と同様に式 (7) で表され，予測ノードと誤差関数を最小化するように式 (11) によって結合重みの更新が行われる．

$$\Delta\mathbf{W} = -\eta \sum \frac{\delta E_t(\mathbf{W})}{\delta \mathbf{W}} \quad (11)$$

この学習を行うことで RNN 上のすべての結合重みが更新される．学習によって入力層と中間層の間の結合重み $\mathbf{W}^{HX}$，文脈層と中間層の間の結合重み $\mathbf{W}^{HC}$，中間層と制御出力ノードの間の結合重み $\mathbf{W}^{YH}$ が更新されることで，ユーザの入力に対応するロボットの動作が変更される．

3. 提案手法の評価実験

構造の違うニューラルネットワークをインタフェースに

用いたときのロボットの操縦性能の違いについて評価するために，FFNN および RNN を用いたインタフェースを実装してロボット操縦を行う．まず，FFNN，RNN それぞれにおいて相互学習の手法を評価するために，相互学習を行った結果と人間だけが学習を行った結果を比較する．その後，インタフェースに FFNN を用いた結果と RNN を用いた結果を比較する．

3.1 実験設定

実験では物理シミュレータである Bullet Physics[7] によって仮想物理環境中に作成したロボットの操縦を行う．ロボットは 4 脚のヒトデ型ロボットを用いる．ロボットの 1 脚には関節が 2 つあり，それぞれ上限 $\pi/6$[rad]，下限 $\pi/4$[rad] の範囲で稼働する．ロボットの形状を図 2 に示し，脚の関節の可動範囲を図 3 に示す．

また，ロボットの操縦には Apple 社の iPhone に搭載されている画面のタッチセンサーを使用する．検出されたタッチ座標はインタフェースの変換関数であるニューラルネットワークの入力層 $\mathbf{X_t}$ に入力され，出力値 $\mathbf{Y_t}$ が生成される．出力値 $\mathbf{Y_t}$ はロボットの脚関節の上限から下限に写像され，各関節の目標角度が決定する．その目標角度になるように PID 制御により関節が動く．これらのタッチ座標検出および物理計算は 1 ステップ（1/30[s]）ごとに行われる．

実験ではインタフェースに FFNN と RNN を用いる．FFNN の入力層，中間層，出力層の制御用，出力層の予測用のノード数はそれぞれ 2，6，8，2 である．BP による学習率は 0.01 としており，学習では過去 90 ステップ分の入力列を保持しておき，各入力値を現在のニューラルネットワークに入力して BP を行う．これを全入力に対して 4 回ずつ行うことで 1 ステップにおける結合重みの更新を行う．また，RNN の入力層，文脈層，中間層，出力層の制御用，出力層の予測用のノード数はそれぞれ 2，8，6，8，2 である．BPTT による学習率は 0.01 としており，5 ステップ前まで遡って結合重みの更新を行う．さらに，FFNN と同様に，過去 20 ステップ分の入力列に対してそれぞれ 1 回ずつ BPTT を行うことで 1 ステップにおける結合重みの更新を行う．FFNN，RNN のいずれにおいても予測ノードでは現在の入力から 10 ステップ後 (333[ms] 後) の入力への変化量を予測する．

3.2 タスクと実験条件

この実験におけるタスクは，ロボットを実験開始地点から目標方向へと，目標到達距離に達するまでなるべく速く移動させることである．今回の実験では目標到達距離を 50[m] とした．このタスクを，FFNN で相互学習を行う条件（F_{HM}），FFNN で人間のみ学習を行う条件（F_H），RNN で相互学習を行う条件（R_{HM}），RNN で人間のみ学習を行う条件（R_H）の 4 つの条件の下で行う．相互学習を行う条件では，BP および BPTT で利用する入力データを蓄積するために，タスクを開始して 5 秒経ってから機械の学習を開始する，また，機械の学習だけでなく人間の学習も行われたことを確認するために，ロボットの移動距離が 40[m] を超えたところで機械の学習を停止する．実験は被験者 A，B，C，D，E，の 5 人が行う．各被験者とも，F_{HM}，R_{HM}，F_H，R_H の順番で実験を行う．このとき，被験者にはどの条件で実験が行われるのか知らされない．実験におけるニューラルネットワークの初期の結合重みは，FFNN，RNN それぞれで，各条件，各被験者ともに共通の結合重みを設定した．結合重みは center crossing[8] を行い，ニューラルネットワークの出力値が 0.5 を中心に出力されるよう調整を行った．

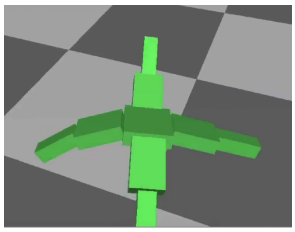


図 2 4 脚のヒトデ型ロボット
Fig.2 4-legged robot.

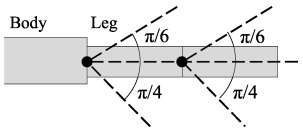


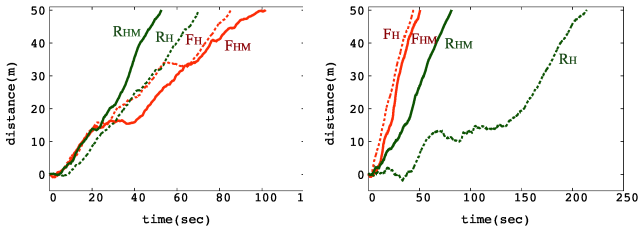
図 3 脚関節の可動範囲
Fig.3 Angle limit of each joints.

3.3 実験結果

各被験者の各条件でのロボットの目的方向への移動距離を示したのが図 4 である．各図において，FFNN で相互学習を行う条件 F_{HM} と人間のみ学習を行う条件 F_H を比較すると，被験者 A，B，C，E において条件 F_H のほうがより早く目標到達距離に到達していた．つまり，FFNN を用いたインタフェースでは，機械の学習によりインタフェースの更新を行わずに与えられたインタフェースを人間のみが学習してロボット操縦を行うほうが良い操縦性能を得ることができた．これは，FFNN を用いたインタフェースでは出力値が現在の入力値のみから決まるため操作が簡単であり，相互学習を行う必要がなかったからだと考えられる．

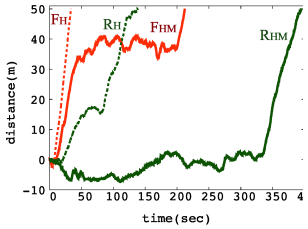
また，各図において，RNN で相互学習を行う条件 R_{HM} と人間のみ学習を行う条件 R_H を比較すると，被験者 A，B，E において，条件 R_{HM} のほうがより早く目標到達距離に到達していた．つまり，RNN を用いたインタフェースでは，相互学習を行うことで良い操縦性能を得ることができた．これは，RNN を用いたインタフェースでは出力値が入力値の時系列情報から決まるため操作が難しくなるが，相互学習を行うことでユーザが操作しやすいインタフェースが構築されたからだと考えられる．

次に，FFNN を用いたインタフェースと RNN を用いたインタフェースの比較を行う．各図において各条件の比較

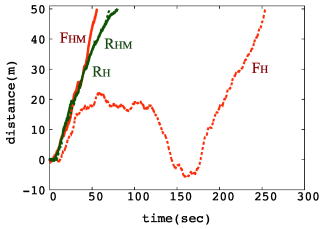


(a) Subject A

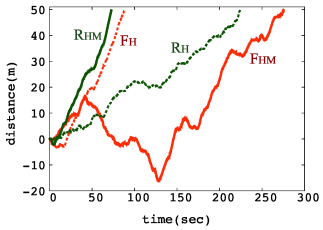
(b) Subject B



(c) Subject C



(d) Subect D



(e) Subject E

図 4 各被験者におけるロボットの移動距離
Fig.4 Moving distances of the robot operated by each subject.

表 1 ロボットの目標方向への速度の最大値 (m/s)
(色付きの値は被験者内の各条件における最大値)

Table1 Maximum speed to the target direction(m/s).

	Sub.A	Sub.B	Sub.C	Sub.D	Sub.E
R_{HM}	1.48	1.50	2.43	1.26	1.86
R_H	1.24	0.83	1.46	1.59	1.03
F_{HM}	1.19	2.45	1.61	1.99	1.54
F_H	1.40	1.69	2.31	1.52	1.18

を行うと，最も早く目標到達距離に到達した条件は被験者によりばらつきが見られた．そこで，タスク全体における操縦性能ではなく，タスク中に構築されたインタフェースの操縦性能を見る．今回の実験におけるタスクは目標到達距離になるべく早く到達することである．そこで，タスク中におけるロボットの目標方向への速度を見る．表 1 は各被験者，各条件ごとにロボットの目標方向への速度の最大値を示した表である．表 1 を見ると，被験者 A，C，E において RNN で相互学習を行う条件 R_{HM} で最大速度が最も高い値になっている．つまり，RNN を用いたインタフェースで相互学習を行うことで、一時的にではあるが、操縦性能の高いインタフェースが構築された．

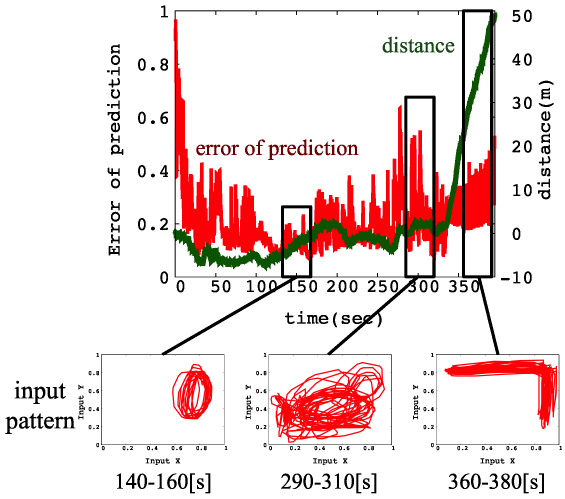


図 5 被験者 C の R_{HM} 条件下における
ロボットの移動距離，誤差の推移，入力パターン
Fig.5 Moving distance , prediction error and inputpattern operatedby subjectC under R_{HM} condition.

ここで，図 4(c) および表 1 で被験者 C の条件 R_{HM} の結果を見ると，図 4(c) では目標到達距離に到達した時間は最も遅く，操縦性能の低いインタフェースであるように見えるが，表 1 では最大速度が最も高い値になっている．このとき，図 4(c) で被験者 C の条件 R_{HM} におけるロボットの移動の様子を見ると，330 秒付近までほとんど移動できていないが，330 秒以降は目標方向に向けて最大速度に近い速度で移動できている．このことから，学習に時間がかかったため目標到達距離に到達した時間は最も遅いが、学習が完了した後は操縦性能の高いインタフェースで操縦を行うことができたため最大速度は最も高い値になったと考えられる．つまり，RNN を用いた相互学習インタフェースでは，長い時間をかけて学習を行うことで操縦性能の高いインタフェースが構築できると期待される．

また，被験者 C の条件 R_{HM} におけるロボットの移動距離と，そのときの予測誤差の推移，入力パターンを示したのが図 5 である．まず，図 5 の 140 秒から 160 秒までの結果を見る．ロボットの移動距離を見ると，速度は遅いが目標方向へと進んでいる．また，このときの入力パターンを見ると周期運動で安定しており，予測誤差を見ると低い値となっている．これは，目標方向へのロボットの操縦ができたため，被験者は現在のロボット動作を繰り返すために同じ入力を繰り返し行い，入力が周期運動となって安定したため予測誤差が下がったと考えられる．このとき，被験者は固定化されたインタフェースで意図するロボット操縦を行うことができる．

次に，290 秒から 310 秒までの結果を見る．ロボットの移動距離を見ると，目標方向へと進めずにほぼ同じ位置に留まっている．また，このときの入力パターンを見ると周期運動ではなく様々なパターンの入力が行われており，予

測誤差を見ると高い値となっている．これは，より良いロボット動作を見つけようとして様々な入力パターンを試しているところであり，そのため予測誤差が上がっていると考えられる．このとき，機械の学習によりインタフェースの更新が大きく行われる．

最後に，360 秒から 380 秒までの結果を見る．ロボットの移動距離を見ると，目標方向へと速く進んでいる．また，このときの入力パターンを見ると周期運動で安定しているが，予測誤差を見ると高い値となっている．これは，ロボットを目標方向へと操縦できたため被験者の入力が周期運動となったが，比較的複雑な入力パターンであるため，予測がうまくできずに予測誤差が上がってしまったと考えられる．このような状況では，いい操縦性能が得られるインタフェースが出来上がっているにも関わらず，予測誤差によるインタフェースの更新が行われてしまうため，徐々にインタフェースが崩れてしまう．図 5 においても，340 秒から 350 秒における平均速度は 1.14[m/s] であったが，370 秒から 380 秒における平均速度は 0.64[m/s] であった．これは，インタフェースが崩れてロボットの動作パターンが変化したためだと考えられる．このように，入力パターンによっては周期運動をしているにも関わらず誤差が高くなることがあり，インタフェースが徐々に崩れてしまう問題が起こる．この問題に対しては，わずかな予測誤差ではインタフェースの更新を行わないように，予測誤差に閾値を入れる方法が考えられる．

4. 結論

本稿では，RNN を用いた相互学習インタフェースを提案し，RNN と FFNN を用いたインタフェースの操作性能の違いについて比較を行った．結果として，FFNN を用いたインタフェースでは，操作が簡単なため，相互学習を行わずに人間のみ学習を行ったほうが良い操縦性能を得ることができた．一方，RNN を用いたインタフェースでは，操作が難しいため，相互学習を行ったほうが良い操縦性能を得ることができた．また，FFNN を用いたインタフェースと RNN を用いたインタフェースを比較すると，タスク全体を通した操縦性能では被験者によってばらつきがあったが，タスク中に構築されたインタフェースの操縦性能では，RNN を用いたインタフェースで相互学習を行うことにより良い操縦性能を得られるインタフェースが構築されていた．このことから，長い時間をかけて学習を行うことで操縦性能の高いインタフェースが構築できると期待される．

また，今回の 4 脚ロボットの操縦においては，FFNN を用いたインタフェースでは操縦が簡単であり，RNN を用いたインタフェースでは操縦が難しいという結果であった．しかし，より多自由度のロボットの操縦について考えると，出力値が現在の入力値からのみ生成される FFNN では，ロボットの自由度が高くなり動作が複雑になるにつれて入力も複雑になる．そのため，ユーザである人間の操縦負荷が高くなると考えられる．一方，RNN の出力値は入力値の時系列情報から生成されるため，ロボットの自由度が高くなり動作が複雑になったとしても，入力履歴に基づいて複雑なロボット動作を生成できると考えられる．より多自由度のロボットの操縦では RNN を用いることでユーザの操縦不可を減らすことができると考えられる．

参考文献

[1] 丹羽真隆，飯塚博幸，安藤英由樹，前田太郎．つもり制御:人間の行動糸の検出と伝送によるロボット操縦．日本バーチャルリアリティ学会論文誌，Vol.17，No.1，2012.

[2] 安藤健，小島康史，関雅俊，川村和也，二瓶美里，佐藤春彦，辰巳友佳子，大野ゆう子，井上剛伸，藤江正克．重度脳性まひ児の残存機能を利用した人・機械相互学習型電動車いすの開発．日本ロボット学会誌，Vol.30，No.9，pp.873-880，2012.

[3] 飯塚博幸，齋藤健太郎，山本雅人．人間と機械の相互学習による共創造型インタフェースの構築．電子情報通信学会技術研究報告ー第 13 回クラウドネットワークロボット研究会，pp.17-22，2014.

[4] Jordan，M.I.．Attractor Dynamics and Parallelism in a Connectionist Sequential Machine．Eighth Annual Conference of the Cognitive Science Society，pp.513-546，1986.

[5] Rumelhart，D.E.，Hinton，G.E.，and Williams，R.J. Learning internal representations by error propagation. In Parallel Distributed Processing: Explorations in the Microstructure of Cognition. D.E. Rumelhart，J.L. McClel- land，and PDP Research Group，eds. Volume 1. MIT Press，318362，1986.

[6] Ronald J Williams and Jing Peng: An Efficient Gradient-Based Algorithm for On-Line Training of Recurrent Network Trajectories,Neural Computation, Issue.2, pp.490-501, 1990.

[7] Real-Time Physics Simulation，Home of the open source Bullet Physics Library and physics discussion forums，http://bulletphysics.org/

[8] Mathayomchan, B., Beer, R.D.: Center-crossing recurrent neural networks for the evolution of rhythmic behavior. Neural Computation 14(9), 2043-2051, 2002