第 24 回日本バーチャルリアリティ学会大会論文集（2019 年 9 月）

強化学習を用いた 回転量操作型リダイレクションコントローラの構築

A Rotation Gain Controller for Redirected Walking Using Reinforcement Learning

張祐禎 1)，松本啓吾 1)，鳴海拓志 1)，葛岡英明 1)，廣瀬通孝 1)
YuChen CHANG, Keigo MATSUMOTO, Takuji NARUMI, Hideaki KUZUOKA, and Michitaka HIROSE

1) 東京大学 情報理工学系研究科（〒 113-0033 東京都文京区本郷 7-3-1,
{chang,matsumoto,narumi,kuzuoka,hirose}@cyber.t.u-tokyo.ac.jp）

概要: 本研究では，強化学習を利用し汎用性のある回転量操作型リダイレクションコントローラを構築し，シミュレーションを行なって評価した。歩行体験できるバーチャル環境においてユーザが自主回転する際に，角度に補正をかけることでユーザの進行方向を変化させる。これにより壁や障害物などに対する回避操作を減少でき，ユーザ体験の向上が期待できる。進行方向角度について強化学習を適用することで，バーチャルコンテンツの内容や現実空間の形状にとらわれず，最適な操作量をリアルタイムで出力することができる。

キーワード： リダイレクテッド・ウォーキング，機械学習

1. はじめに

バーチャル空間（Virtual Environment, VE）を探索するもっとも直感的な方法はユーザの実際の歩行である。しかし，現実空間の大きさが VE より小さいことが多いため，足で探索可能な空間は現実空間の大きさと形状に制限される。この問題の解決方法として，Redirected Walking[1] が提案されている。ユーザが気づかない程度に HMD 内の視野を少しずつ移動、回転させて，現実空間の障害物からユーザを遠ざけることが出来る。

Redirected Walking の主な手法として，移動時の曲率操作と移動量操作，及び静止回転時の回転量操作の三種類がある。ユーザの現在状況に合わせて適宜な操作を出力するために，操作のコントローラが必要である。本研究では強化学習を利用して，VE の事前知識なしに回転量操作型コントローラを構築する。強化学習を利用する利点として，VE ごとに最適なコントローラを構築する必要はないこと，コントローラの特徴に合わせて VE のコンテンツを調整する必要がないことが挙げられる。

2. 関連研究

2.1 Redirected Walking

VE と現実空間の形状非対等がもたらす問題は，VE に無いはずの現実空間の障害物は VR 体験の邪魔になることである。目の前に無限大な VE が広がっていても，歩いているうちに現実世界の障害物にぶつかる可能性がある。

歩行経路上の障害物を回避する方法として，2002 年，Razzaque et al.[1] は Redirected Walking を提案した。Redirected Walking では実際のユーザの動きとわずかに異なる映像を HMD を通して提示することにより，ユーザの進行方向や移動量を一定の範囲で調整することを可能にする。基本的な Redirected Walking の手法として，歩行時の進行方向を変化させる曲率操作，移動量を変化させる移動量操作，回転量を変化させる回転量操作が提案されている [2]。こうした操作は視覚と前庭感覚の矛盾を生起させるため，一定の操作量を超えると没入感の低下や VR 酔いなどの問題が生じることが知られている [1]。Steinicke et al.[2], Hodgson et al.[3] らはユーザに気づかれない閾値について検証している。

これらの操作はどの場合でどの程度設定すべきか，多数の研究がされていて，提案された設定方針はコントローラと呼ばれる。数あるコントローラの中で，Razzaque が提案した Steer to Center (S2C) が一番よく使用、比較される。Hodgson et al.[3] は，通常状況にてこれらの操作を一番効率的に使えるコントローラは S2C と主張した。しかし，全ての環境条件において最高効率を出しているわけではなく，汎用性にはかけている。

また，ユーザの移動ルート上に障害物が存在し，上記の操作を用いても衝突する危険性が依然として存在している場合，安全のために体験を強制的に一時中止し，ユーザを移動もしくは回転を要求して，安全なルートに誘導しなければならない。この対処法はリセットと呼ばれる [4]。リセットは VR 体験の没入感を著しく損なうとされており，体験中のリセット回数をできるだけ減らすことが望ましい [5]。

コントローラの効能が上がれば、回避できるリセットが増え、総回数が少なくなる。そこで、本研究では強化学習を用いて，S2C より汎用性の高い回転型操作コントローラ

の作成により、リセット回数を減少させることを試みる。

2.2 強化学習（Reinforcement Learning）

強化学習は機械学習の一種で，環境の状況を観測し，それに応じて最良の行動が取る方策（policy）を学習する手法である [6]。この特徴から，強化学習は継続的出力が必要な問題の扱いによく用いられている。

機械学習によく使用されるラベルの代わりに，学習時は毎回の行動後に報酬（reward）を与える。報酬の量によって前回取った行動の評価がわかり，それを元に行動の方策が更新される。観測、行動及び方策の更新を行うものは一般的にエージェントと呼ばれる。

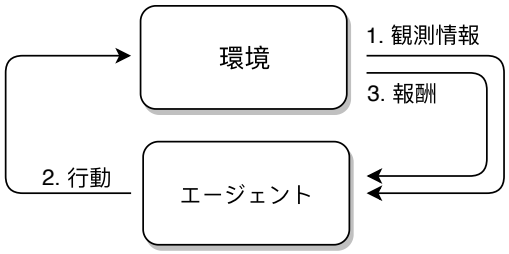

図 1: 強化学習の概略図

強化学習の基本構造はマルコフ決定過程として記述することが出来る。マルコフ決定過程は 4 つの要素 (S, A, P, R) に構成される：$S \in \mathbb{R}^n$ は観測可能な状態，$A \in \mathbb{R}^m$ は実行可能な行動，$P : S \times A \times S \to [0, 1]$ は状態 s_t で行動 a_t を取った時に環境が状態 s_{t+1} に遷移する確率，そして $R : S \times A \times S \to \mathbb{R}$ は上記行動 a_t に対する報酬 r_t である。

学習の最終目的はエージェントの行動を決定する方策 $\pi : S \times A \to [0, 1]$ を改善して，得られる報酬を最大化することである。見込み報酬を予測するために，価値関数 $V(s)$ がしばしば使われる。

2.3 深層強化学習 (Deep Reinforcement Learning)

深層強化学習は，強化学習と深層学習を組み合わせたものであり，強化学習の一部を多層ニューラルネットワークに置き換えるものである。

また，本研究で用いた Proximal Policy Optimization (PPO)[7] と呼ばれる手法では，方策及び価値関数をニューラルネットワークに置き換えられている。深層学習の対象になった方策を更新するには，方策勾配法 (Policy Gradient Method) がよく用いられている。方策勾配法は目的関数を定義し，目的関数を最大化する勾配へ方策パラメータを更新する。一般的な目的関数は (1) 式のようになる：

$$L^{PG}(\theta) = \hat{\mathbb{E}}_t \left[\log \pi_\theta(a_t|s_t) \hat{A}_t \right]. \tag{1}$$

$$\hat{A}_t = \delta_t + (\gamma\lambda)\delta_{t+1} + ... + (\gamma\lambda)^{T-t+1}\delta_{T-1}, \tag{2}$$
$$\text{where } \delta_t = r_t + \gamma V(s_{t+1}) - V(s_t)$$

π_θ は確率的な方策で，A_t は時間 t で取った行動 a_t によって生まれる利益の予測。$L(\theta)$ を θ に対して微分を取ることで，更新すべき θ の勾配が計算できる。しかし，このまま直接適用すると，勾配が大きい場合学習が不安定になる。この問題を解決するために，PPO では改悪する可能性のある更新の勾配に上下限を設けられている。

$$L^{CLIP}(\theta) = \hat{\mathbb{E}}_t \left[\min \left(r_t(\theta)\hat{A}_t, clip\left(r_t(\theta), 1-\epsilon, 1+\epsilon\right)\hat{A}_t \right) \right], \tag{3}$$
$$\text{where } r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{old}(a_t|s_t)} \tag{4}$$

ϵ はハイパーパラメータで，Schulman et al.[7] は ϵ を 0.2 と設定した。この制限を設けることにより，学習効率は減る可能性があるが，学習はより安定になる。

3. 環境及び提案手法

本研究はシミュレーション環境でリダイレクテッドウォーキングを行い，その中の回転量操作を深層強化学習で設定し，結果を検証する。以下各環境設定について説明する。

3.1 シミュレーション環境

本紙が行うシミュレーションは Unity で行われ，図 2 のように模擬ユーザは 15m 四方の現実空間にて，無限大の VE を自由探索する。現実空間の限界を超えた時，模擬ユーザの探索行動は中断され，現実空間で再び前へ歩けるように適度な再配置をされる。この処理はリセットと呼ばれる。

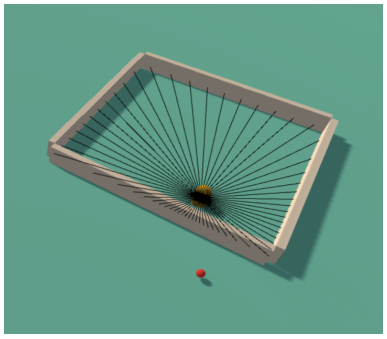

図 2: シミュレーション環境。黄色いカプセルは模擬ユーザ，赤い玉はターゲット，ベージュ色は現実空間の壁を表す。模擬ユーザから発された黒い線はエージェントの距離観測情報を表す。

3.2 模擬ユーザと歩行経路

本節ではユーザーが色々な VR 環境での歩行経路をシミュレーションする方法を説明する。シミュレーションプロセスは下記になる。

1. VE 内に目的ターゲットを一つ生成される。
2. 模擬ユーザは回転フェーズに入り，向きをターゲットの位置に合わせる。
3. 模擬ユーザは歩行フェーズに入り，ターゲットへ直進する。
4. 移動中障害物が 0.5m 以内にある場合，移動を一時中止し，リセットを行う。
5. 模擬ユーザがターゲットに着く。1 に戻る。

表 1: ターゲット生成方法

方法	距離	角度
Exploration Small	$unif(2,6)$	$unif(-\pi,\pi)$
Office	$unif(2,8)$	$\{-\frac{\pi}{2},\frac{\pi}{2}\}$
Random	$unif(2,12)$	$unif(-\pi,\pi)$
Exploration Large	$unif(8,12)$	$unif(-\pi,\pi)$

ターゲットの生成方法は Azmandian et al.[8] が提案した方法を参考にし，表 1 の 4 つにする。

LongWalk と呼ばれる歩行経路条件も提案されているが，本紙の目標である回転量操作の学習及び運用に不向きなので除外した。また，新たに上下限がそれぞれ Exploration Large (XLarge) と Exploration Small (XSmall) の上下限の値になっている，距離の偏差の大きい Random を追加した。

3.3 リセット

本シミュレーションでは，模擬ユーザと障害物の距離が 0.5m 以下になるとリセットを行う。よく用いられるリセット手法の一つは to-center[9] である。部屋の中心点を定義し，リセット時にユーザの視野に体験中止のメッセージもしくはシグナルを提示し，ユーザが部屋の中心へ向くように回転を要求する。しかし現実運用の場合，障害物は部屋の中心にある可能性もあるため，上記の手法は効果不十分である。

そのため，本研究では to-center を改変した turn-to-furthest (T2F) を用いた。T2F では部屋の中心点ではなく，リセットが行われる地点から見て一番遠い場所をターゲットとし，ユーザがターゲットを向くように回転を要求する。この方法を用いると上記の問題を解決できる。

3.4 コントローラ

3.4.1 曲率操作型コントローラ

- Steer To Center(S2C)

本研究は回転型操作コントローラの作成及び評価を行うが，曲率操作と回転量操作は基本的に合わせて行われる [1] ため，学習時及び評価時に模擬ユーザの移動に曲率操作を適用する。本研究で使用しているのは S2C の改変 [3] で，操作の出力は (5) 式にようになる：

$$gain_t = \begin{cases} gain_{t-1} & , if\ \mathrm{abs}(\alpha_c) > \frac{8}{9}\pi \\ T_{curv} \cdot Sign\,(\alpha_c) & , if\ \frac{1}{4}\pi < \mathrm{abs}(\alpha_c) < \frac{8}{9}\pi \\ T_{curv} \cdot \sin\,(4\alpha_c) & , else \end{cases} \tag{5}$$

T_{curv} を曲率操作の閾値とする。現実空間の中心点を定義し，上記の式は模擬ユーザの現在位置から中心点までのベクトルと向きが挟む角度 $\alpha_c \in [-\pi,\pi]$ によって方向と曲率を算出する。S2C の基本方針は模擬ユーザを部屋の中心に戻すように曲率を操作している。閾値 T_{curv} は Hodgson et al. が主張した閾値 [10] を参考に，曲率半径下限を 7.5m までと設定する。

3.4.2 回転量操作型コントローラ

- Reinforcement Learning (RL)

強化学習を用いてエージェントに周辺情報を入力し，出力を閾値にマッピングする。学習は回転時のみとし，歩行時は学習しない。強化学習の入力は計 64 個で，出力は 1 個である。各入力は観測可能な数値の絶対値の最大値で割って正規化する。強化学習の出力は −1 と 1 の間で，使える数値に変換するためにこれを閾値にマッピングする。回転量操作の閾値は Steinicke et al.[2] を参考にして，下限を 0.67，上限を 1.24 と設定する。

入力の詳細は表 2，報酬は表 3 から参照できる。

表 2: 環境観測の入力

入力説明	数
周辺 6 度ずつ，障害物までの距離	60
現在位置の平面座標	2
現在の向きから部屋の中心までの角度	1
前ステップの回転角度	1
立ち止まってからのの累計回転角度	1

表 3: 報酬

報酬説明	値
歩行フェーズ中リセット回数（初ステップのみ）	$-45 \times n$
回転量操作による時間伸び縮みの補正	$c_1(g_{rot}-1)$

特に注目すべき報酬は回転量操作による時間伸び縮みの補正である。ゲインが 1 より小さい回転量操作は，回転に費やす時間を伸ばせてしまい，一見リセット回数が減少しているように見える。しかし歩行距離を固定して比較したところ，リセット回数はかえって増加してしまう。補正報酬を設けることにより，ゲインを減らしてリセット回数が減っても全体報酬は増えないので，学習は正しい方向に進むことが出来る。

- Baseline

回転量操作を 1 に固定するコントローラで，操作していない時と同じ出力である。後述の実験で対照として使う。

- Steer To Center (S2C)

曲率操作の S2C と同じく，基本方針はユーザを部屋の中心に戻るように回転量を操作している。後述の実験で対照として使う。

3.5 フレームワーク

本研究のシミュレーションはゲームエンジン Unity で行い，一ステップは 0.02 秒と設定した。深層強化学習のフレームワークは Unity の ML-Agents toolkit[11] を使用した。

4. シミュレーション実験

4.1 実験手順

上記の環境設定で 100 万ステップ学習して，出力したモデルを同じ環境で 100 万ステップ実行した結果を他二種類のコントローラの結果と比較する。RL は 5 ステップごとに環境を観測して行動を行う。評価指標はリセット回数とし、回数が少なければ少ないほど，手法が効果的とする。

4.2 結果と考察

図 3 は各環境条件において，三手法の移動距離を Baseline の移動距離に合わせた時のリセット回数である。

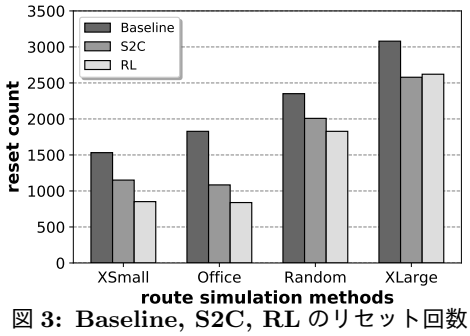

図 3: Baseline, S2C, RL のリセット回数

S2C と比べて XSmall では 26.1% ，Office では 22.6% ，Random では 8.9% 少なくなっている。一方，XLarge では 1.6% 多くなっている。XLarge 環境のみ RL が S2C よりリセット回数が多かったが，一番結果の良かった XSmall 環境のモデルを XLarge 環境に適用した結果，リセット回数が S2C より 7.2%少なくなっていることがわかった。以上のことから，XLarge では効率的に学習できなかったことが考えられる。

また，ターゲット間の平均距離が増えることにつれて，三手法ともリセット回数が増加し，更に RL と S2C の回数と Baseline の回数との比率が上がることが判明した。XSmall では RL と Baseline の回数比率が 55.6% だが，XLarge では 85.1%である。この現象の原因としては，ターゲット間の距離が長くなることにつれて，歩行フェーズが占める総ステップ数の比率も上がり，回転で回避出来るリセットも少なくなるためだと考えられる。

5. むすび

本研究では回転型操作のゲインの最適化を深層強化学習を用いて構築した。シミュレーションによる評価では既存手法の S2C と比べた結果，没入感の低下につながるリセット回数を最大 26.1%減少させることが出来た。一方で，ターゲット間の平均距離の増加によりリセット回数が増加する傾向がみられたことから，長距離より短距離移動のほうが回転型操作が有効にリセット回数を減らせると考えられる。

強化学習は S2C よりリセット回数を減少させるのに有効なコントローラであることが判明したが，改善の余地がある。観測情報入力の追加及びハイパーパラメータに調整を加えることによる効果を今後の課題としたい。

参考文献

[1] Sharif Razzaque, Zachariah Kohn, and Mary C Whitton. *Redirected walking*. Citeseer, 2005.

[2] Frank Steinicke, Gerd Bruder, Jason Jerald, Harald Frenz, and Markus Lappe. Estimation of detection thresholds for redirected walking techniques. *IEEE transactions on visualization and computer graphics*, 16(1):17–27, 2009.

[3] Eric Hodgson and Eric Bachmann. Comparing four approaches to generalized redirected walking: Simulation and live user data. *IEEE transactions on visualization and computer graphics*, 19(4):634–643, 2013.

[4] Betsy Williams, Gayathri Narasimham, Bjoern Rump, Timothy P McNamara, Thomas H Carr, John Rieser, and Bobby Bodenheimer. Exploring large virtual environments with an hmd when physical space is limited. In *Proceedings of the 4th symposium on Applied perception in graphics and visualization*, pages 41–48. ACM, 2007.

[5] Tabitha C Peck, Henry Fuchs, and Mary C Whitton. Evaluation of reorientation techniques and distractors for walking in large virtual environments. *IEEE Transactions on Visualization and Computer Graphics*, 15(3):383–394, 2009.

[6] Leslie Pack Kaelbling, Michael L Littman, and Andrew W Moore. Reinforcement learning: A survey. *Journal of artificial intelligence research*, 4:237–285, 1996.

[7] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[8] Mahdi Azmandian, Timofey Grechkin, Mark T Bolas, and Evan A Suma. Physical space requirements for redirected walking: How size and shape affect performance. In *ICAT-EGVE*, pages 93–100, 2015.

[9] Anh Nguyen and Andreas Kunz. Discrete scene rotation during blinks and its effect on redirected walking algorithms. In *Proceedings of the 24th ACM Symposium on Virtual Reality Software and Technology*, page 29. ACM, 2018.

[10] Eric Hodgson, Eric Bachmann, and David Waller. Redirected walking to explore virtual environments: Assessing the potential for spatial interference. *ACM Transactions on Applied Perception (TAP)*, 8(4):22, 2011.

[11] Arthur Juliani, Vincent-Pierre Berges, Esh Vckay, Yuan Gao, Hunter Henry, Marwan Mattar, and Danny Lange. Unity: A general platform for intelligent agents. *arXiv preprint arXiv:1809.02627*, 2018. https://github.com/Unity-Technologies/ml-agents/.