修士論文

敵対的生成ネットワークを用いたゲーム開発支援システムの有用性

早稲田大学大学院基幹理工学研究科
情報理工・情報通信専攻

鳥羽 洸誠

学籍番号　5122F065-5
提出　2024 年 1 月 22 日
指導教授　中島 達夫
研究指導名　分散システム研究

The Usefulness of Game Development Support Systems using Adversarial Generative Networks

Kosei Toriba

A Thesis Submitted to the Department of Computer Science and Communications Engineering, the Graduate School of Fundamental Science and Engineering of Waseda University in Partial Fulfillment of the Requirements for the Degree of Master of Engineering

Student ID	5122F065-5
Submission Date	January,22nd,2024
Supervisor	Professor Tatsuo Nakajima

Department of Computer Science and Communications Engineering
Graduate School of Fundamental Science and Engineering
Waseda University

概要

今日ゲームにおける 3D モデルやグラフィックの品質が頭打ちになり始めており, ユーザーによるゲームの評価はシステム等のアイデアに比重が大きく置かれていると考える. さらに, 深層学習による自動生成技術はゲーム開発にも用いられている.

そこで本研究では, 主に画像/動画生成の分野で多く用いられている深層学習モデルのアーキテクチャである敵対的生成ネットワーク (Generative Adversarial Networks)[以降 GAN と呼ぶ] を用いて, 既存のゲーム映像から新たなゲーム映像を生成することで, システムのアイデア出しとしてゲーム開発を支援するモデルを設計し, その有用性を調査する評価実験を行った.

Abstract

As the quality of modeling and graphics in today's games is beginning to reach a ceiling, We think that the evaluation of fun by users is placing more weight on the ideas of systems or rules of game. In addition, automatic generation techniques based on deep learning are beginning to be used in game development.

In this study, We designed a model that supports game development by generating new game videos using existing game videos with Generative Adversarial Networks (GANs), an architecture of deep learning models commonly used mainly in the field of image/video generation, and conducted an evaluation experiment to investigate its usefulness.

目次

図目次

表目次

第 1 章

序論

本章では, 本研究の関連研究・背景について述べる.

1.1 背景

今日ゲームにおける 3D モデルやグラフィックの品質が頭打ちになり始めており, ユーザーによるゲームの評価は, システム等のアイデアに比重が大きく置かれていると考える. さらに近年では, ChatGPT[*1]をはじめとした生成 AI による自動生成技術が, マップの地形生成等の目的でゲーム開発にも用いられている. 深層学習技術はモデル内部に無数のパラメータを持っており, 一般的に推論結果がどのように導かれたのか人間が理解することはできない. このブラックボックス性は医療機関や異常検出システムにおいては問題点となりうるが, ゲーム開発のようなエンターテインメント業界においてはむしろこの特徴を逆手にとって, 人間が思いつかないようなものを生成できるという強みになり得ると考える. そこで本研究はゲームデザイナーに対して, 新たなゲームシステムのアイデアを出す役割を深層学習モデルが担うことができるという仮定のもと, 主に画像/動画生成の分野で用いられる深層学習モデルのアーキテクチャである GAN を用いて, 既存のゲーム映像から新たなゲーム映像を生成することでゲーム開発支援を行うシステムの設計およびその有用性を調査する.

[*1] https://chat.openai.com/

1.2 深層学習

1.2.1 ニューロン

ニューロンとは, 生物の脳を構成する神経細胞のことであり, ヒトの脳内に数百億個程度存在する. ニューロンは, 立体的に集まって回路を構成しており, 他のニューロンから受け取った出力が一定の値を超えると, 神経インパルスと呼ばれる電気パルスを出力するという特性がある.

1.2.2 ニューラルネットワーク

ニューラルネットワークとは, ニューロンの働きを模した数理モデルであり, 線形関数と非線形関数の合成関数を, 重みパラメータやバイアスを調整することで. 入力に対して望ましい出力をするようなものに訓練していく仕組みの事を指す.

始めに, ニューラルネットワークの起源であるパーセプトロンの概念について説明する. N 個の入力ノード x_i および重みパラメータ $w_i[i = 1, 2, ..., N]$ に対する出力 y について考える. パーセプトロンの概念では閾値を θ として y は以下のように計算される.

$$y = \begin{cases} 1 & \left(\sum_{i=1}^{N} w_i x_i > \theta \right) \\ 0 & \left(\sum_{i=1}^{N} w_i x_i \leq \theta \right) \end{cases} \tag{1.1}$$

次に $N = 2$ の場合において, パーセプトロンを用いて AND 演算を表現することを考える. AND 演算は, 以下の真理値表で表される.

表 1.1　AND　演算の真理値表

x_1	x_2	y
0	0	0
0	1	0
1	0	0
1	1	1

この場合は例えば, $w_1 = 0.7$, $w_2 = 0.7$, $\theta = 1.2$ とすれば, AND 演算を表現できる.
次に N = 2 の場合において, パーセプトロンを用いて XOR 演算を表現することを考える. XOR 演算は, 以下の真理値表で表される.

表 1.2　XOR　演算の真理値表

x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	0

この場合は, 要件を満たすパラメータは存在しない.
これは入力を二次元空間上に配置したとき, XOR の値によって線形分離不可能であるためである. 一方で上述のパーセプトロンは線形変換であるが, これを何層か重ねた多層パーセプトロンは, 非線形分離も可能で, 線形・非線形演算を表現することが出来る.

以上のように, 多層パーセプトロンは非線形な演算を表現することが出来るが, 各重みや閾値のパラメータを手動で適切に設定する必要がある.
ニューラルネットワークの基本構造は多層パーセプトロンから成り, その中でもパーセプトロンの層をより複雑にしていったものが深層学習モデル

である. 以下に最も基本的なニューラルネットワークの構造である 3 層の FNN(Feedforward Neural Network) の概要図と働きを示す.

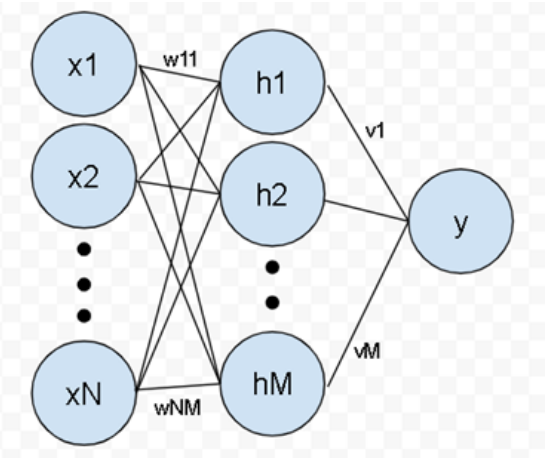

図 1.1　FNN の概要図

w_{ij} は x_i と h_j を結び, v_k は h_k と y を結ぶ重みパラメータである.
またそれぞれの値は以下のように計算できる.

$$h_j = f\left(\sum_{i=1}^{N} x_i w_{ij} + b_j \right) \quad (1.2)$$

$$y = g\left(\sum_{j=1}^{M} h_i v_j + b' \right) \quad (1.3)$$

ここで, 入力次元 N や隠れ層の次元 M はモデルの学習時に指定することが出来るパラメータであり (以下ハイパーパラメータと呼ぶ), 学習効率や得られるモデルの精度に大きな影響を及ぼす. 一般に次元を大きくする程モデルの表現力は大きくなり, より複雑な問題を解くことが出来るが, 学習時の計算量が多くなり計算資源や時間的問題が生じる. また b_j および b' はバイアスであり, 式 1. 1 の条件部 θ を移項したものと捉えることが出来る.
また f および g は活性化関数と呼ばれる非線形関数で, シグモイド関数 (式 1. 4) や Leaky ReLU 関数 (式 1. 5) などが用いられる.

6

$$y = \frac{1}{1 + e^{-x}} \quad (1.4)$$

$$y = \begin{cases} x & (\ x\ >\ 0\ ) \\ ax & (\ x \leq\ 0\ )[a \text{ は定数}] \end{cases} \quad (1.5)$$

以上の機構を複数組み合わせる事で, 入力テンソルを望ましい出力テンソルに変換する学習器である深層学習モデルを作ることが出来る. なおテンソルとは, スカラー・ベクトル・行列などの実数の多次元配列である.

1.2.3 GAN

ゲームにおける表現は, 音楽やコントローラーの振動などが挙げられるが, 最も重要であるのは画面出力である. 近年深層学習を用いた自動生成技術が発展しており, AI タレントの CM 起用[*2]やディープフェイク技術の悪用[*3]が話題になっている. これら画像や映像の生成タスクには, しばしば GAN というアーキテクチャが使われている. GAN とは, 画像の潜在表現であるベクトル空間から取り出されたノイズ入力から画像を出力する生成器 および 入力画像が学習データセットに含まれる本物か, 生成器によって生成された偽物かを判別する識別器の 2 つの深層学習モデルがお互いを騙すように学習を進めることで, 最終的に生成器が学習データセットに含まれる画像の潜在表現を理解し, 出力する画像がより本物に近くなるというものである. 以下に, 基本的な GAN のアーキテクチャを示す. 生成画像の例には MNIST[1] を用いている.

[*2] https://www.itmedia.co.jp/business/articles/2310/29/news021.html
[*3] https://www3.nhk.or.jp/news/html/20231115/k10014256291000.html

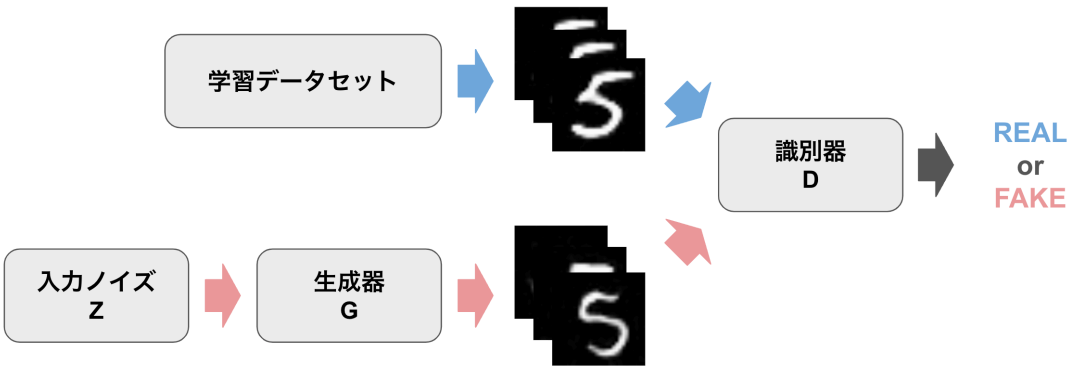

図 1.2　GAN の概要図

純粋な GAN においては, 生成器の入力は入力ノイズのみであるため, 学習データセットが複数ドメインの画像を含んでいた場合どのドメインを選択して画像を出力するかは定まらない. 具体的に, 犬と猫の画像を混ぜたものを学習データセットとした際, 出力される画像を犬か猫のどちらかに限定することはできない. 一方で, Mirza らによって発表された Conditional-GAN(CGAN) [2] では, 生成器の入力にラベル情報を加え, 識別器はラベルと生成画像が一致している, かつ本物画像であることを判別するように学習することで, 特定のドメインを選んで画像を出力できるような GAN を学習させることができる. 本研究ではゲーム映像の生成モデルとして GAN のアーキテクチャを選択した. 生成 AI には, 他にも Kingma らによって発表された VariationalAutoEncoder(VAE) [3] および Sohl-Dickstein らによって発表された DiffusionModel [4] などが挙げられるが, VAE は損失関数にピクセルごとの 2 乗誤差を使うことが原因で, GAN に比べて生成画像がぼやけやすいという特徴がある. ここで, GAN と VAE の生成画像に現れる特徴について実験を行った. 以下は, 同一環境において, およそ同数のパラメータ数を持つ GAN と VAE に MNIST を同時間学習させた際の生成画像である.

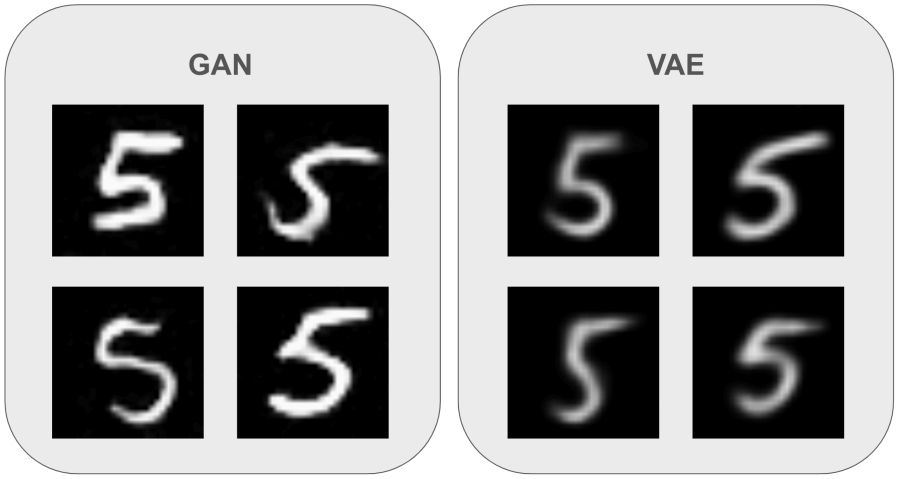

図 1.3　GAN と VAE の生成画像が持つ特徴の比較

モデルの学習方法に違いはあるものの, VAE の生成画像が GAN に比べてぼやけていることが図からわかる．また DiffusionModel では, 派生した動画生成モデルの先行研究が GAN に比べて少ない．そのため, 本研究では生成画像の精度と多様性のトレードオフをパラメータや利用する深層学習レイヤーによって比較的容易に操作できる GAN を用いた．生成映像から新たなゲームのアイデアとする際に, 生成映像の鮮明さは重要となるためである．

1.2.4　RNN

本研究では, モデルの学習対象はゲーム映像であるが, 映像は各フレームにおける画像の系列と考えることができる．また映像における各フレームの画像は基本的に前後関係があり, フレーム同士が内容の繋がりを持つ必要がある．この情報を通常の FNN で学習する場合, 学習したい映像 (画像の系列) をそれぞれテンソルにエンコードし, 連結した上でモデルの入力とする方法が考えられるが, それでは何番目の画像の埋め込み表現であるかの情報が失われてしまい, 系列の順番に関する情報をうまく学習することが出来ない．以下, 系列の学習に適した RNN(Recurrent Neural Network) について説明する．
入力のベクトル系列を $x_1,\ x_2,\ x_3,\ \ldots\ ,\ x_N$ とすると, RNN の基本構造は

以下のように表せる.

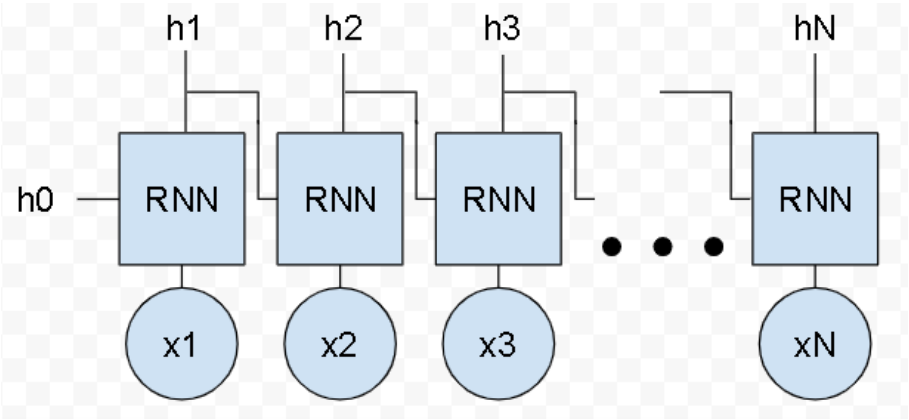

図 1.4　RNN の概要図

図の RNN ブロックはすべて共通のパラメータを持っており, 入力 x_i, h_{i-1} に対して h_i を出力する. 入力系列を順番に与えて行く事で, 順序の情報を失うことなく学習が行えるというのが RNN の大きな特徴である. 図の場合では, h_N が入力のベクトル系列をエンコードしたベクトルである. また h_1, h_2, h_3, ... , h_N を全体の出力とすれば他の RNN への入力として利用でき, RNN を重ねることで学習能力の向上も期待できる.

1.3　関連研究

1.3.1　PGGAN

GAN は他の生成 AI に比べ鮮明な画像を生成できる一方で, 二つの深層学習モデルを同時に学習させる必要があるため, 一般的に訓練が不安定になりやすい. そのため, より大きな解像度の画像を生成するタスクにおいては, 偶然識別器を騙すことができるノイズのような画像が生成され続け訓練が収束しないことや, 生成した画像の多様性が著しく低く, 学習データセットの複製となってしまうモード崩壊という問題が引き起こされやすい. Karras らによって発表された Progressive-GAN [5] では, このような問題を克服するために, 学習をいくつかのステップに分け, 生成/識別する画像の解像度を段階的に上げる方法が提唱されている. 直感的には, 訓練

の序盤では画像の大局的な構造を内部表現として学習し, 終盤になるにつれてより詳細な表現を学習していくという意味を持つ. このプロセスにおいて, 高解像度の画像を学習する際には既に低解像度の特徴を十分に捉えているため, 始めから高解像度の特徴を学習しようとするよりも, より学習の安定化が見込める. また, PGGAN にはモデルを通るテンソルの値の正則化や, 生成画像の多様性を保証するための工夫がいくつか施されている. 学習プロセスの途中で, 画像の解像度を上げる際に, Upsampling 層および DownSampling 層をそれぞれ生成器と識別器に挿入することで解像度の変更を行うが, その際にいきなり層を挿入するのではなく, これまでの入力を単純に変形したものと, 追加する層の出力をスムーズに混ぜていくことで, 学習パラメータの平滑化に対する工夫がされている.

1.3.2 MoCoGAN

GAN を用いて T フレームからなる映像を生成する方法として, 入力ノイズを T 個用意して, そこから T 枚の画像を生成し, それらを繋げるというものがある. また, 入力ノイズは画像の潜在表現のベクトル空間から取り出されるが, その空間内のある点からある点への軌跡を映像の潜在表現と考えることができる. 映像生成における課題は 3 点挙げられる. 1 点目に, 特に現実世界の映像を学習対象とする際, 映像内の物体が物理的に自然な挙動をすることを保証しなければならない. 2 点目に, 人間はそのような物理的挙動の違和感に対して知覚しやすい. 3 点目に, 時間次元を加味した映像の潜在表現は膨大な情報量を持ち, 学習が非常に難しい. 本研究で提案するモデルでは, 学習対象をゲーム映像に設定しており, ゲームの種類によっては物理的挙動が現実世界のそれに従わないことも考えられる. しかし, 各ゲーム内の空間にはそれぞれの物理法則が設定されており, その法則に従う動きを学習しなければならないため, 映像生成における課題は本研究のタスクにも同様に当てはまる. Tulyakov らによって発表された MoCoGAN [6] では, これらの課題を解決するために, 映像の潜在表現を 2 つのサブ空間に分けることを提案している. 一般に, 特に短い動画では, 背景や実際に動く物体の種類は固定であり, 映像を動きの潜在表現 (Motion)

と内容の潜在表現（Content）に分けることができると提唱されている．この際, ノイズ入力は動きの空間および内容の空間からそれぞれ取り出され, それらを合成したものを生成器へ入力する方法が提案されている．また, Motion 表現は系列の学習に適した RNN の一種である GRU を用いて取り出し, Content 表現は映像内で一貫しているという仮定から, 同じノイズをフレーム分並べて使用する．画像生成における GAN の識別器の役割は, 入力画像が学習データに含まれる本物か生成器によって生成された偽物かを判別することであるが, 映像生成タスクを対象とする MoCoGAN の識別器は, 画像の識別器に加え, 映像が本物か偽物であるかの識別器という二つの深層学習モデルが存在する．これらを同時に学習させることにより, 映像の各フレームの品質だけでなく, 映像としての一貫性の品質も向上させることに成功した．

1.3.3 GameGAN

本研究の目的は GAN を用いて新しいゲーム映像を生成し, ゲームデザイナへのアイデア出しの有用性を調査することであるが, Kim らによって発表された GameGAN [7] では, ゲームのプレイ映像および操作入力を学習することで, ノーコードで同じゲームをプレイできる状態で再生成することに成功している．これは, 異なる OS 間におけるゲームの移植を可能とする．また, 例えば迷路のゲームにおいて, プレイヤーが一度通った道に戻る際, 迷路の形状が変わってはならない．GameGAN ではこうしたフィールドの一貫性を保証するために, メモリーモジュールという外部モジュールを RNN と組み合わせることで, 長期記憶を可能とした．またその結果, ゲームプレイ中変化しないフィールドと, 動き続けるプレイヤーの表現を分割することに成功しており, 同じゲーム性でフィールドを他のものに差し替えることを可能にしている．GameGAN では MoCoGAN と比較して, Content に該当するゲームフィールドの表現を, Memory という外部モジュールに保存しておくことで一貫性を実現している．

1.4 研究の目的

本研究の目的は, GAN を用いてゲーム映像を生成することで, システムのアイデア出しとしてゲーム開発を支援するモデルを設計し, その有用性を調査および議論することである.

1.5 研究の貢献

本研究により, 生成 AI がゲーム内のオブジェクト生成だけでなく, ゲームシステムの生成に対しても有用であることを示唆するとともに, 生成 AI がゲームシステムのアイデアを考案し, ゲームデザイナと協力することで, ゲーム開発分野がより豊かになると考える.

第 2 章

提案モデル

本章では, 実装したモデルの構成について述べる.

2.1 構成

本研究で設計した深層学習モデルは Python 言語[*4]で実装し, 機械学習ライブラリである PyTorch[*5]を用いた. モデルの実装は, 個人ブログによる MoCoGAN の実装[*6]および PGGAN の実装[*7]で用いられているモジュールを参考にした. GAN を構成する生成器および識別器と利用したモジュールの構造を以下に示す. N_m, N_c はそれぞれ学習対象のゲーム映像における Motion/Content に属する特徴数を表している.
また, 画像内の注釈は以下に対応する.
※ 1 : 複数連結する際の最初の層に限り挿入しない
※ 2 : 複数連結する際の最後の層に限り挿入
※ 3 : 複数連結する際の最後の層では左, それ以外では右を挿入

[*4] https://www.python.org/
[*5] https://pytorch.org/
[*6] https://github.com/DLHacks/mocogan?tab=readme-ov-file
[*7] https://tzmi.hatenablog.com/entry/2020/05/07/230232

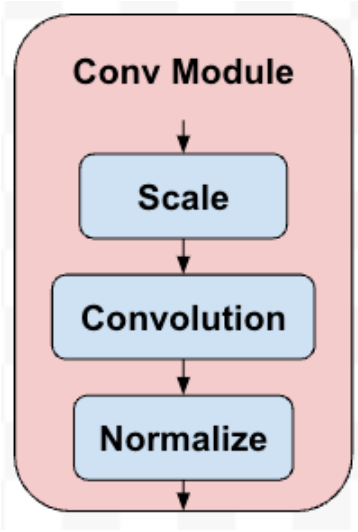

図 2.1　畳み込みモジュールの概要図

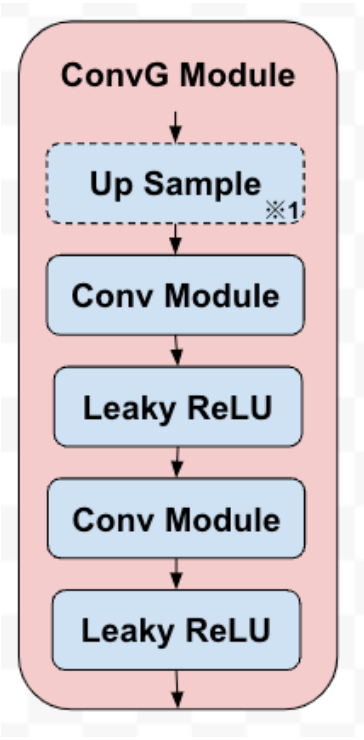

図 2.2　生成器用畳み込みモジュールの概要図

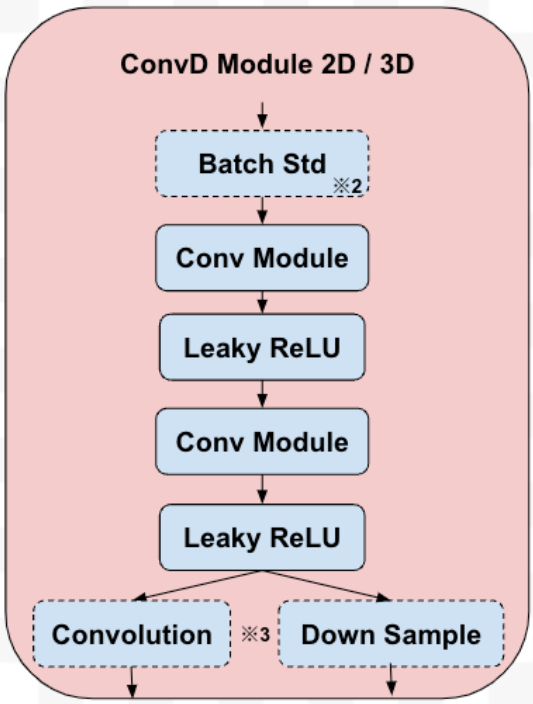

図 2.3　識別器用畳み込みモジュールの概要図

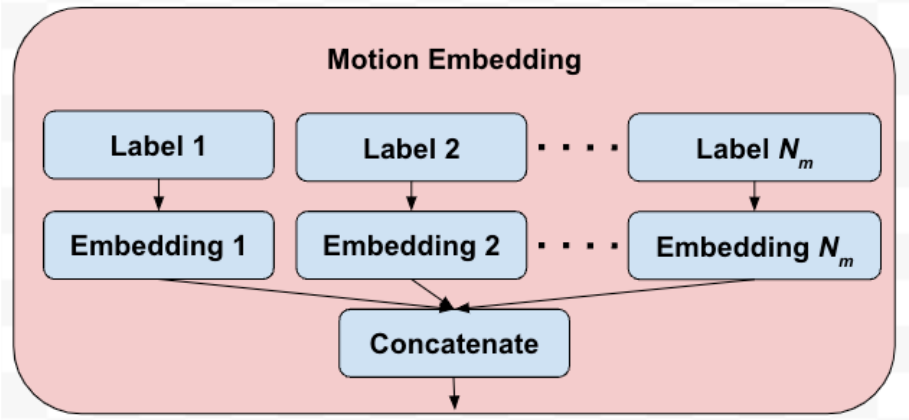

図 2.4　Motion 表現埋め込みモジュールの概要図

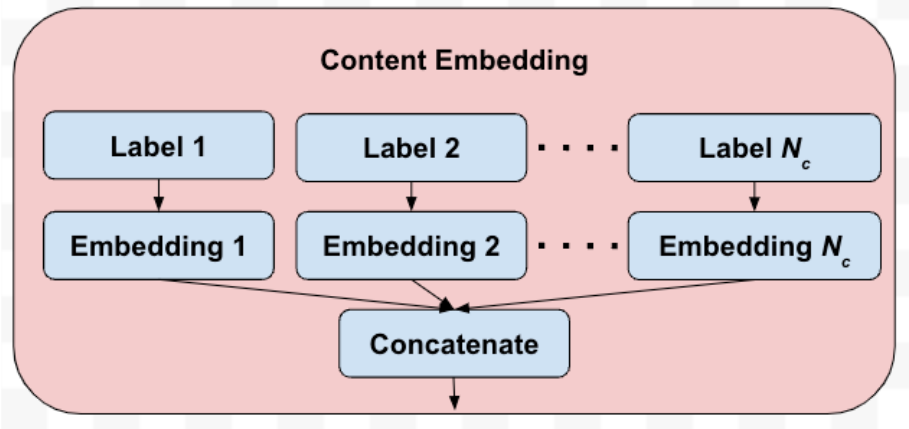

図 2.5　Content 表現埋め込みモジュールの概要図

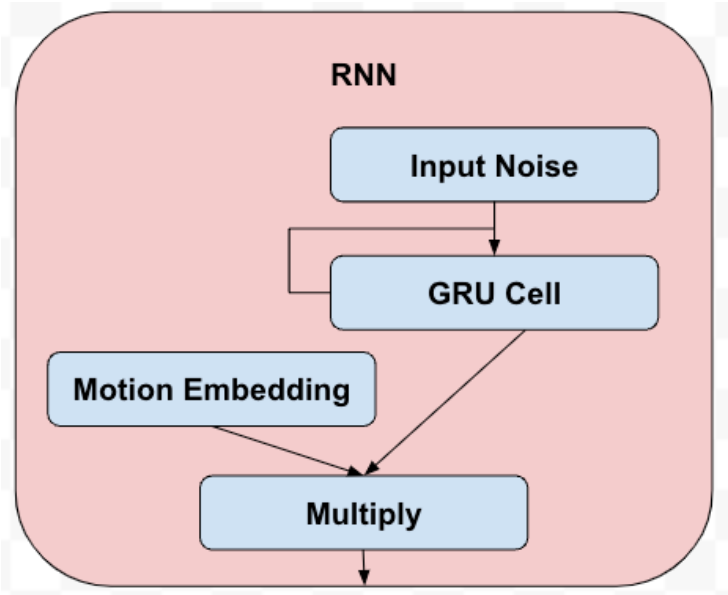

図 2.6　RNN モジュールの概要図

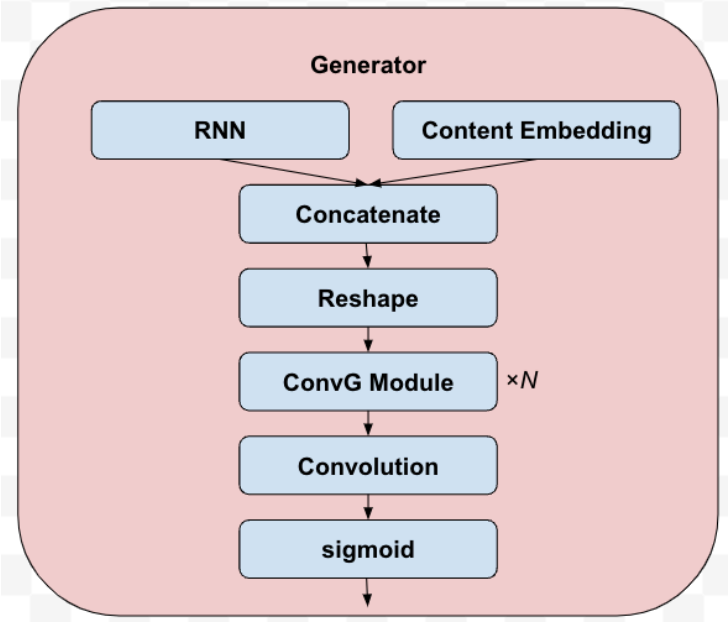

図 2.7　生成器の概要図

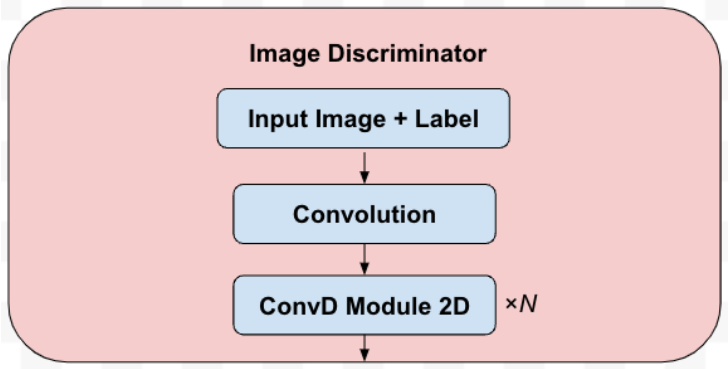

図 2.8　画像識別器の概要図

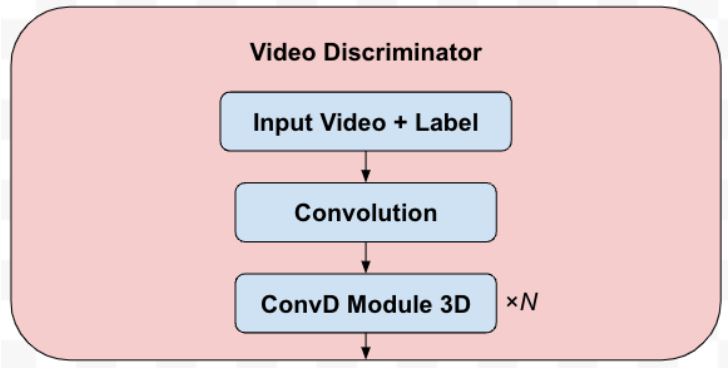

図 2.9　映像識別器の概要図

Scale 層とは, 画像テンソルのチャンネル数を c とした際に (式 2. 1) で計算される値を入力にかけることでスケーリングを行う.

$$s = \sqrt{\frac{2}{c}} \quad (2.1)$$

Normalize 層とは, 画像テンソルのチャンネル方向に 2 乗平均を取り, その逆数の平方根を入力にかけることで正規化を行う.
Batch Std 層とは, ミニバッチ方向にピクセルごとの標準偏差を取り, その平均を入力画像サイズに並べ, 新たなチャンネルとして入力データに加えることで, 生成映像の多様化を行う.
これらは PGGAN で紹介されている学習安定化の工夫であり, 本モデルにも組み込まれている.

また, 本モデルでは主要なハイパーパラメータを以下のように設定した.

表 2.1　ハイパーパラメータ 設定値

ハイパーパラメータ	設定値
ノイズ入力 次元	192
RNN 隠れ次元	192
Motion 埋め込み次元	192
Content 埋め込み次元	192

2.1.1 新規実装点

従来の MoCoGAN では CGAN の概念を用いて, 人間を対象として複数の表情変化を捉えた映像と入力ラベルの対応を学習させることで, 特定の表情変化の映像を選択して生成することを可能にしている. ここでは Motion 空間に含まれるラベルは表情の変化のみに対応し, Content 空間に含まれるラベルは表情を変化させている人物が誰であるかのみにそれぞれ対応していると考えられるが, それぞれ複数のラベルを入力とする場合, より生成映像の自由度が高まるのではないかと考えた. 具体的には, ゲーム映像における二人のプレイヤーの動きを別々の表現としてラベル付けし, それらを組み合わせたものを Motion 空間の埋め込み表現とすることで, 二人のプレイヤーの個々の動きを選択してゲーム映像を生成できると考えた. 図 2.4 および図 2.5 に示した埋め込みモジュールはそれぞれこのような仮定のもと新規に実装した. これらは複数の Embedding 層を持っており, 学習データセットに含まれる映像が持つ特徴の数だけ埋め込み表現を取得し, それらを連結させたものを全体の埋め込み表現とすることで, 複数の特徴をまとめて一つの Motion/Content 空間の表現として得ることができる. ここで得られた Motion 空間の埋め込み表現は RNN モジュールの出力との要素積を取り, Content 空間の埋め込み表現と連結させることで, 生成器の入力とした. また, 本モデルでは PGGAN の生成器に対して, MoCoGAN の RNN モジュールで出力した前後関係を汲んだテンソル列を入力することで映像生成を行なっているが, PGGAN における段階的学習の概念を映像に対して適用した点も新規性を持つ. 従来の PGGAN の学習プロセスは, 前半では画像表現における大局的な構造を理解していきながら, 後半になるにつれてより細部の表現を理解するという意味を持っていたが, 本モデルでは映像を学習対象としているため, 学習前半で各フレーム画像の大局的な構造だけでなく, その前後関係を合わせて理解する必要があり, 本モデルにおける学習の成否は, 低解像度の映像における一貫性を理解することが高解像度の映像生成において有効であるかを示唆する意味を持つ. また, PGGAN と組み合わせる映像生成モデルとして本研究では

MoCoGAN を選択したが, これはゲーム映像におけるプレイヤー等の動的オブジェクトと, 背景等の静的オブジェクトの分離が可能であることが既に GameGAN で示されており, Motion/Content 表現空間分離の観点から共通の概念を持つ MoCoGAN が, ゲーム映像における特徴を理解するのに適していると仮定したためである.

2.2 学習プロセス

GAN の学習では, 生成器と二つの識別器が敵対的に学習を進めていく必要がある. 学習は 1 プロセスにつき生成器および識別器の学習を 1 対 1 の割合で交互に行った. 生成器および識別器の学習方法を以下に示す.

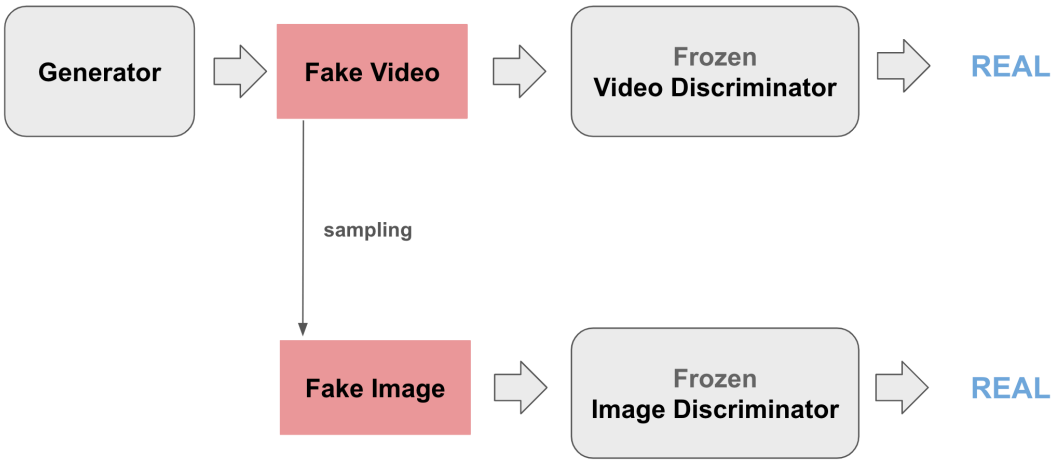

図 2.10 生成器の学習方法

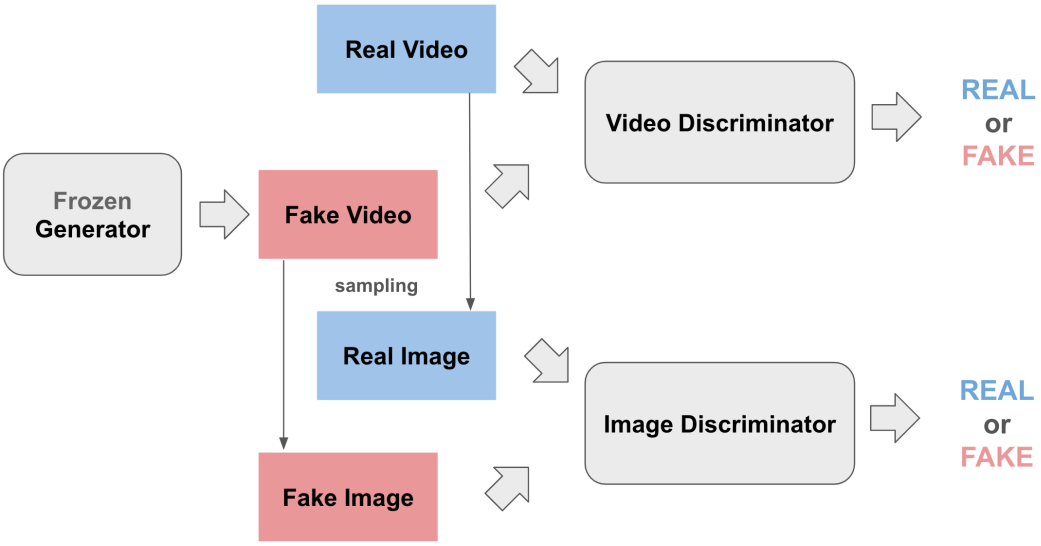

図 2.11　識別器の学習方法

それぞれ相手が学習を進めている際は内部パラメータを凍結することで, 敵対的な学習を実現している. また, 損失関数には Gulrajani らによって発表された WGAN-GP[8] の概念を用いている.

第 3 章

性能評価実験

本章では, 行った性能評価実験について述べる.

3.1 実験目的

本実験の目的は, 学習させたモデルが出力するフレーム画像の品質およびゲーム映像が持つ内容の一貫性を既存の評価手法を用いて定量評価することである.

3.2 学習環境

本実験では, ブラウザ上で Python を実行できる Google Colaboratory[*8] の Colab ノートブックを利用して深層学習モデルを学習させた. また, 学習には NVIDIA A100 GPU を使用した.

3.3 学習データセット

本研究では, 複数のゲーム映像から特徴を抽出し, 入力を工夫することで新たな特徴を持つゲーム映像を生成することを目的としているため, ゲーム映像を想定し, それぞれ異なる特徴の組み合わせを持つ 8 カテゴリの映像 (512 フレーム× 50 本, 128 × 128 ピクセル) を学習データセットとし

[*8] https://colab.research.google.com/notebooks/welcome.ipynb

て用意した．学習データセットに含まれる映像とそのカテゴリを以下に示す．なお映像は変化がわかりやすいよう 3 フレームおきに並べている．

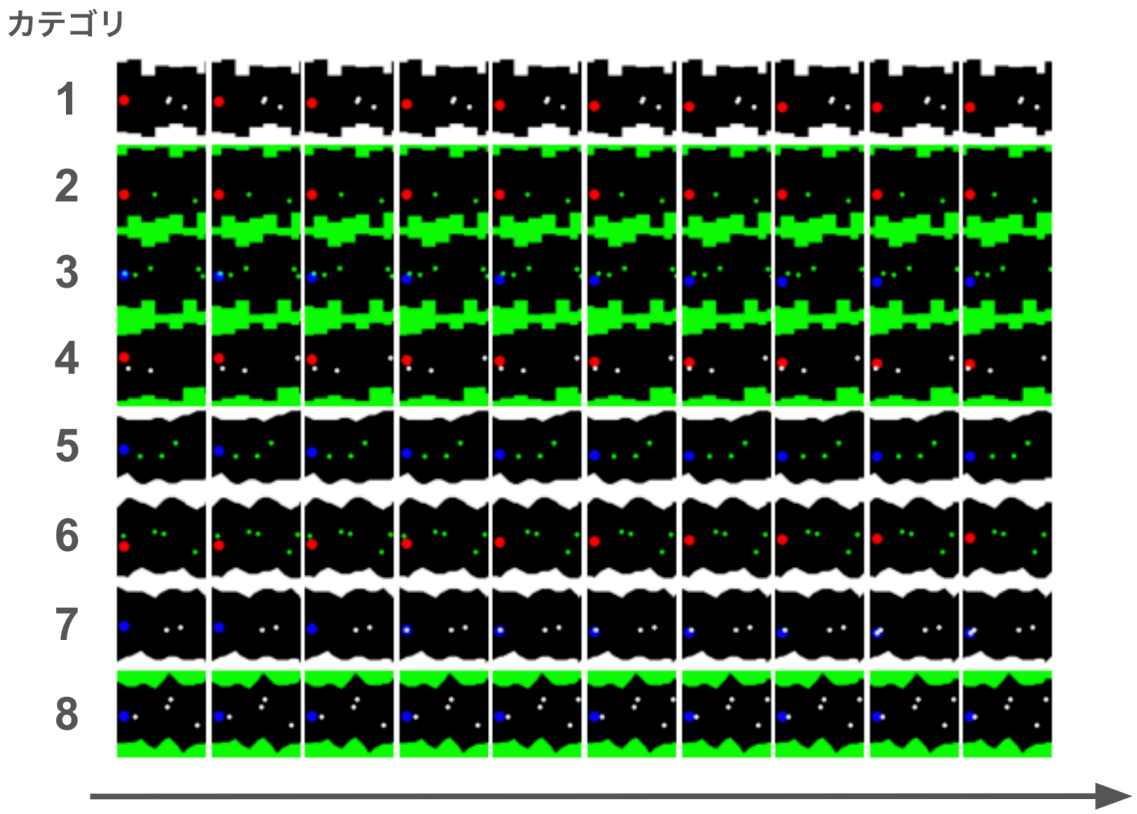

図 3.1　用意した学習データセット

学習データセットに含まれるゲーム映像は, 右スクロールという共通の特徴を持つ一方で, 障害物の形状, オブジェクトの役割, プレイヤーの色, 障害物の色およびオブジェクトの色という特徴の組み合わせがそれぞれ異なっている．また学習難度の観点から各特徴が取り得る値は全て 2 種類とした．そのためモデルが持つ Embedding 層のインデックスサイズは全て 2 となる．以下に各特徴の意味および各カテゴリが持つ特徴を示す．

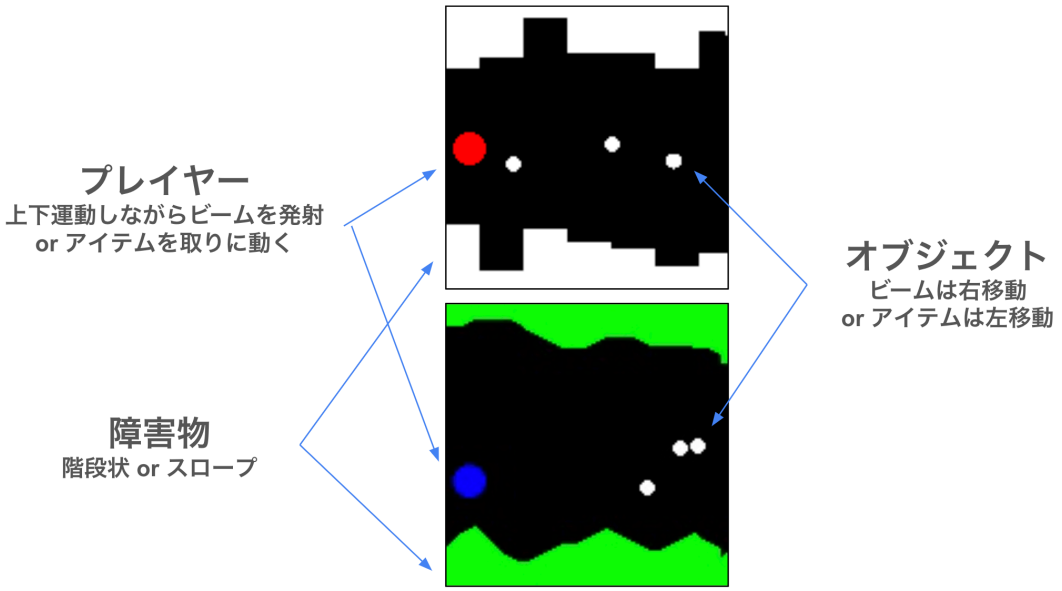

図 3.2　各特徴の意味

表 3.1　各カテゴリが持つ特徴

特徴	カテゴリ							
	1	2	3	4	5	6	7	8
障害物の形状	階段	階段	階段	階段	スロープ	スロープ	スロープ	スロープ
オブジェクトの役割	ビーム	アイテム	ビーム	アイテム	ビーム	アイテム	ビーム	アイテム
プレイヤーの色	赤	赤	青	赤	青	赤	青	青
障害物の色	白	緑	緑	緑	白	白	白	緑
オブジェクトの色	白	緑	緑	白	緑	緑	白	白

各特徴の内, 障害物の形状およびオブジェクトの役割を Motion 空間の特徴, それ以外を Content 空間の特徴に設定した. そのため本モデルにおける N_m および N_c はそれぞれ 2,3 となる. 学習データセットにおける各特徴は以下のルールに従って実装した.

3.3.1　障害物の形状

階段状では, 20 フレームおきに上限および下限をそれぞれ端から 5 から 30 ピクセルの間でランダムに指定し, 範囲外を塗りつぶしている. スロープ状では上限および下限の目標値を階段状と同様に設定し, 20 フレームおきにちょうど一致するように上限および下限を移動させて, 範囲外を塗り

つぶしている.

3.3.2 オブジェクトの役割

ビームシステムでは, プレイヤーが 32 フレームを周期とする正弦波に従って移動しており, 各フレームで 3% の確率で発射されるビームが秒速 1 フレームで右に移動していく. アイテムシステムでは, 各フレームで 3% の確率で出現するアイテムが秒速 1 フレームで左に移動しており, プレイヤーは最も近いアイテムに向かって移動する.

3.3.3 内容物の色

背景は黒で塗りつぶしており, 各内容物の色である, 黒, 赤, 青, 緑および白はそれぞれカラーコードを用いて #000000, #ff0000, #00ff00, #0000ff および #ffffff に設定した.

3.3.4 データフォーマット

ゲーム映像のフォーマットには gif 画像を用い, 学習時には Python の画像処理ライブラリである Pillow[*9]を用いて 16 フレームごとの映像に整形を行った.

3.4 実験結果

本実験では PGGAN における段階的学習の概念を用い, 学習を 7 つのステップに分割して合計約 4 時間学習させた. それぞれのステップと学習プロセスの回数および出力される映像の解像度の対応を以下に示す.

[*9] https://pillow.readthedocs.io/en/stable/

表 3.2　学習ステップと学習プロセスおよび出力解像度の対応

学習ステップ	プロセスの回数	出力される映像の解像度
1	2000	4 × 4
2	4000	4 × 4
3	6000	8 × 8
4	8000	16 × 16
5	10000	32 × 32
6	12000	64 × 64
7	14000	128 × 128

カテゴリ 1 について, 訓練中に出力された映像例を以下に示す.

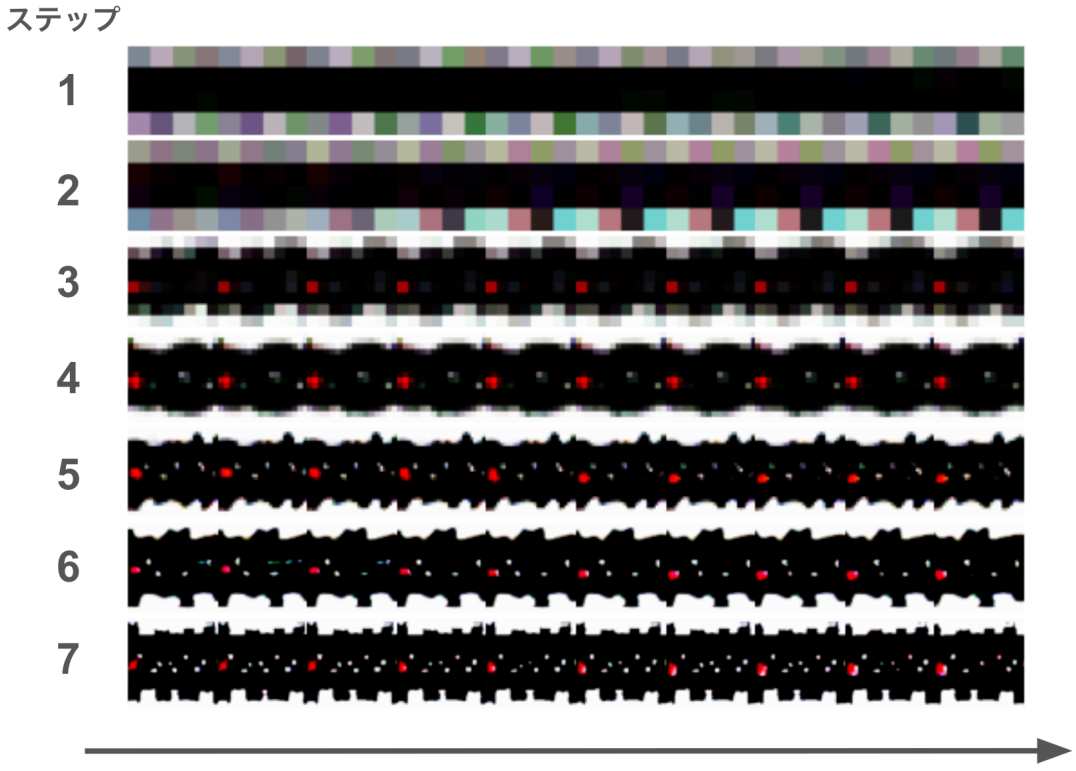

図 3.3　出力された映像例

また学習ステップ 7 において生成された映像の中で品質の良いものと悪いものを著者の主観で選択した. 以下, 選択例をカテゴリごとに示す.

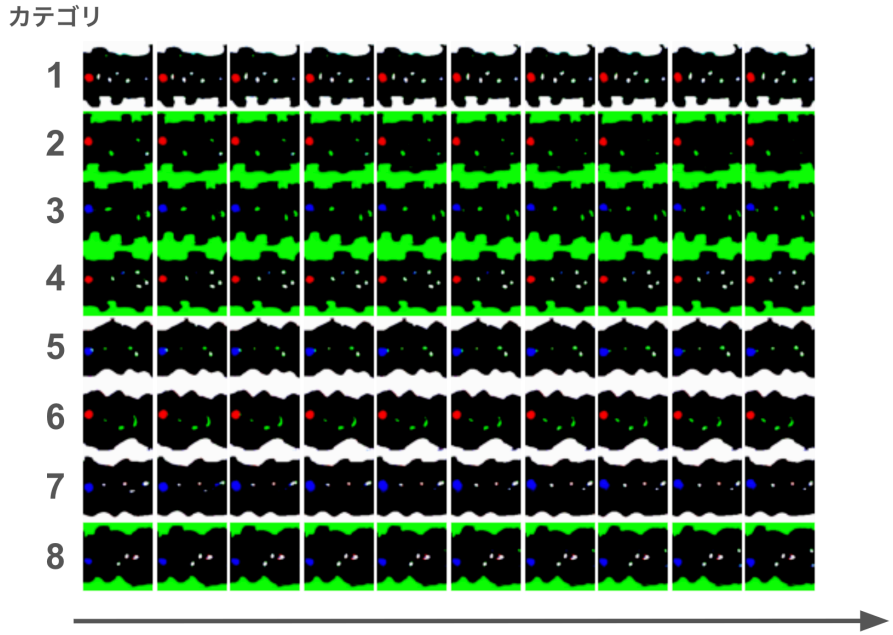

図 3.4　出力された映像例 品質の良いもの

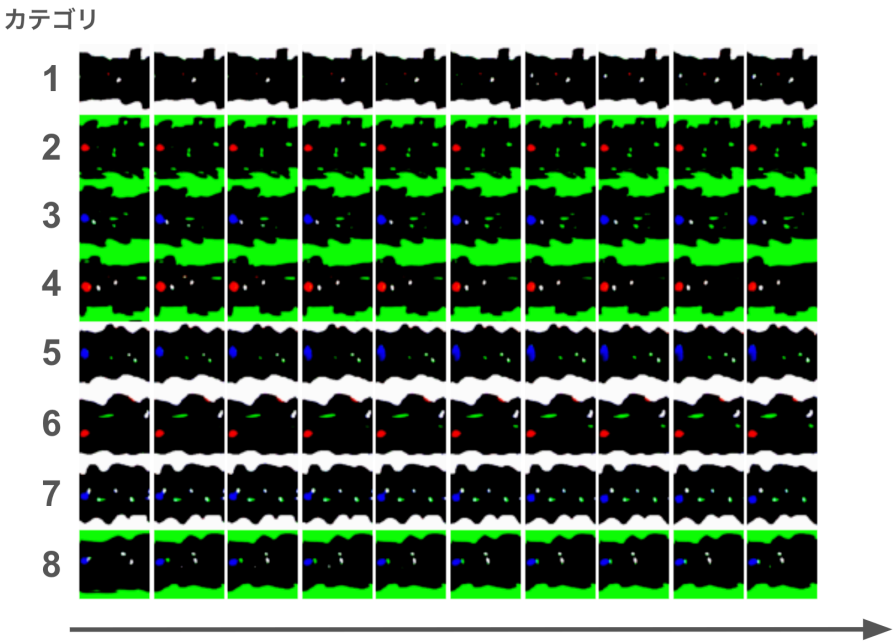

図 3.5　出力された映像例 品質の悪いもの

品質の悪い映像例では, プレイヤーが消える, 障害物の形状が波状になっ

ている, アイテムの形状が細長く伸びてしまっている, アイテムの色が混在しているなどの問題が生じていた.

3.4.1 画像の品質評価

生成映像の各フレーム画像の品質を評価する指標として, Heusel らによって発表された Fréchet Inception Distance(FID) スコアを用いた. [9] これは, Salimans らによって発表され, 従来 GAN の生成画像を評価する指標としてしばしば用いられていた Inception Score [10] のデメリットである, 学習データセットに含まれる画像との品質比較を行なっていない点を補っており, 生成画像のクオリティと多様性を同時に評価することができる. 以下に, 学習データセットおよび生成した映像からランダムに抽出したフレーム画像 2048 枚で計測した FID スコアを, カテゴリごとに計算した結果の比較を示す. なお FID スコアの計算には pytorch_fid パッケージ[*10]を用いた.

表 3.3 各カテゴリで得られた FID スコア

カテゴリ	FID Score	
	生成映像	学習データセット
1	192.98	3.0757
2	208.09	2.7383
3	155.26	2.3999
4	241.80	3.4825
5	78.854	3.3848
6	92.54	3.6973
7	133.85	4.2149
8	130.59	5.1520
平均	154.24	3.5182

3.4.2 映像の一貫性評価

生成映像における内容の一貫性を評価する指標として, MoCoGAN の論文内で提案されている Average Content Distance(ACD) を用いた. ACD

[*10] https://github.com/mseitzer/pytorch-fid

は次のように計算される.

1. あるフレームにおける画像全体の平均色をチャンネルごとに求める
2. 隣接フレーム間で差を取り, チャンネル方向に L2 ノルムを取る
3. 映像全体で平均を取り ACD とする

映像内で, 特定のオブジェクトが移動したとしても, 背景が一定であれば画像の平均色は変わらないことから, ACD が小さいほど映像の内容は一貫性を持っていると言える. 以下に, 生成した映像で計算した ACD を学習データセットと比較した結果をカテゴリごとに示す.

表 3.4　得られた ACD の学習データセットとの比較

カテゴリ	ACD			
	生成映像	学習データセット		
		通常	1フレームおき	ランダム
1	0.00216	0.00130	0.00266	0.0512
2	0.00165	0.000945	0.00166	0.0297
3	0.00123	0.000852	0.00156	0.0358
4	0.00158	0.00102	0.00185	0.0333
5	0.00155	0.000904	0.00241	0.0553
6	0.00241	0.00115	0.00184	0.0544
7	0.00186	0.00122	0.00212	0.0436
8	0.00169	0.000792	0.00142	0.0290
平均	0.00177	0.00102	0.00194	0.0415

第 4 章

ユーザー評価実験

本章では, 学習したモデルを用いて生成したゲーム映像の質的評価実験および生成 AI を用いたゲーム開発支援モデルの有用性評価実験について述べる.

4.1 生成したゲーム映像の質的評価実験

4.1.1 実験目的

本実験の目的は, 学習させたモデルの入力を工夫することで生成した新たなゲーム映像の品質を定性評価することおよびゲーム開発支援システムの有用性を確認することである.

4.1.2 実験内容

本研究で学習させたモデルの入力は, 標準正規分布に従うノイズ入力と Content/Motion 空間に含まれる各特徴のラベルである. 特徴のラベルについて, 表 3.1 に示すように全ての組み合わせが学習データセットに含まれていないため, 含まれていないラベルの組み合わせを入力とすることで, 学習データセットに存在しない特徴の組み合わせを持ったゲーム映像を生成できるのではないかと考えた. 入力を工夫して生成されたゲーム映像のうち, 著者が学習データセットに含まれていない特徴の組み合わせを持つと判断した映像例および入力ラベルの関係を以下に示す.

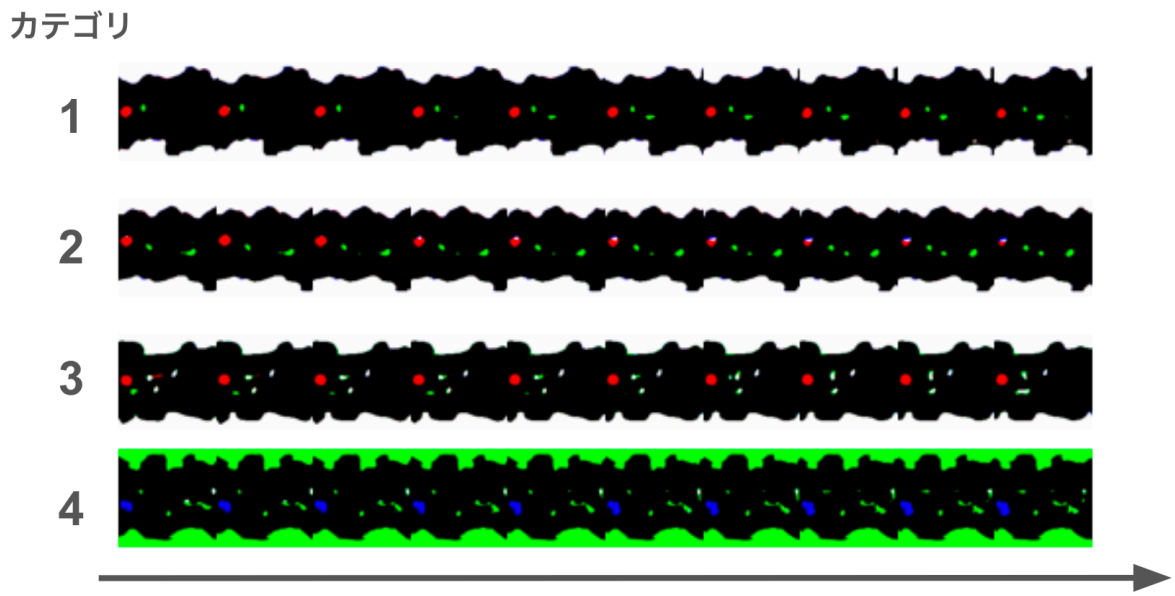

図 4.1　入力を工夫して生成したゲーム映像

表 4.1　カテゴリと入力ラベルの関係

特徴	カテゴリ			
	1	2	3	4
障害物の形状	階段	スロープ	スロープ	スロープ
オブジェクトの役割	ビーム	ビーム	アイテム	アイテム
プレイヤーの色	赤	赤	赤	青
障害物の色	白	白	白	緑
オブジェクトの色	緑	緑	白	緑

図に示した出力映像例では, データセットに含まれていないかつ入力ラベルに従った特徴の組み合わせを持っている. 一方で出力映像のおおよそは, 入力ラベルに従っておらず, データセットに含まれる特徴の組み合わせと同じ特徴を持つものとなってしまった. このことから本研究で学習させたモデルは, 学習データセットに含まれる特徴の潜在表現をある程度理解しているが, 埋め込み表現の処理方法が単純すぎたために, 各特徴と入力ラベルの関係まで学習しきれなかったと考える. また本実験では入力を工夫して生成させた映像のうち, 学習データセットに含まれていない特徴の組

み合わせを持つと判断したものを評価の対象とした. 本実験の被験者は 13 名の男性および 2 名の女性で, 平均年齢は 23.1 歳であり, いずれも情報系学科もしくは情報系専攻の大学生および大学院生である. 被験者には, 学習データセットおよび生成したゲーム映像に関するアンケートに回答してもらった. 以下にアンケートの設問および回答形式を示す.

表 4.2　生成したゲーム映像の質的評価実験 アンケートの設問

質問番号	質問内容	回答形式
1	学習データセットの映像はゲームに見えますか	択一A
2	学習データセットの映像からゲームの大まかなシステムを推測できますか	択一A
3	モデルの生成映像はゲームに見えますか	択一A
4	モデルの生成映像からゲームの大まかなシステムを推測できますか	択一A
5	入力を工夫した生成映像 特徴の選択問題	特徴選択
6	入力を工夫した生成映像は学習データセットに含まれていない特徴の組み合わせを持つと言えますか	択一A
7	アンケートを通じての意見など	自由記述

表 4.3　生成したゲーム映像の質的評価実験 アンケートの回答形式

択一形式	選択肢
A	5段階のリッカート尺度(とても~する, やや~する, どちらでもない, あまり~しない, 全く~しない)
B	5時間以上, 3~5時間, 1~3時間, 1時間未満, 全くプレイしない
C	はい, いいえ

また質問番号 5 で被験者に見せた生成映像は図 4.1 のカテゴリ 1 であり, 各特徴についてどちらに見えるか選択してもらった.

4.1.3　アンケートの結果

生成したゲーム映像の質的評価実験アンケートの結果を以下に示す.

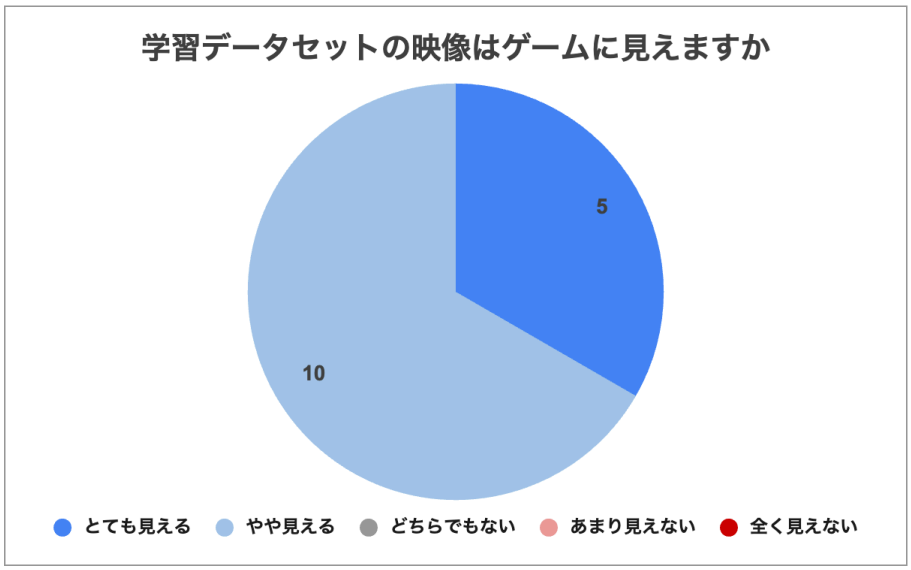

図 4.2　質問 1 回答

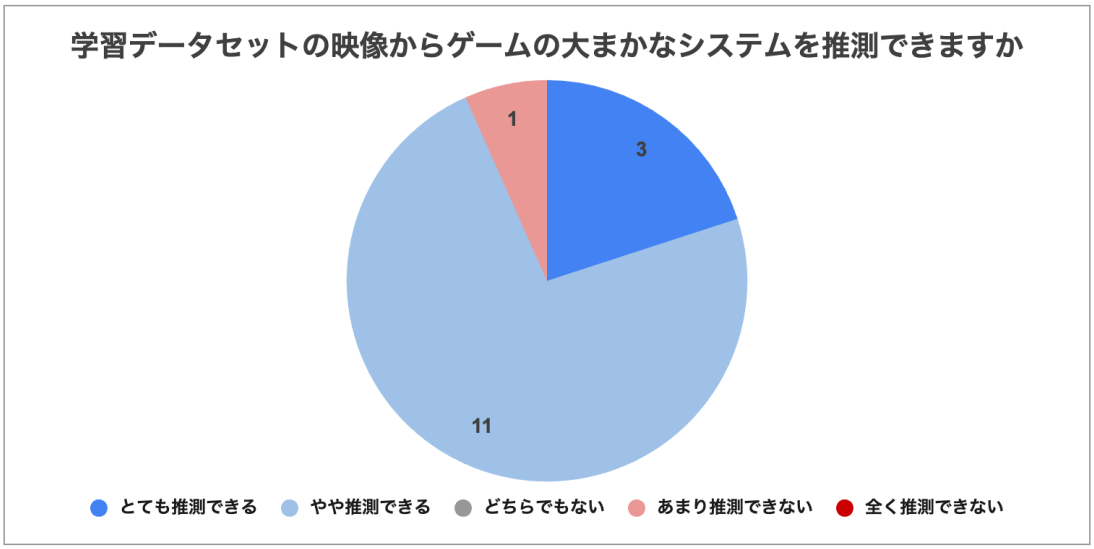

図 4.3　質問 2 回答

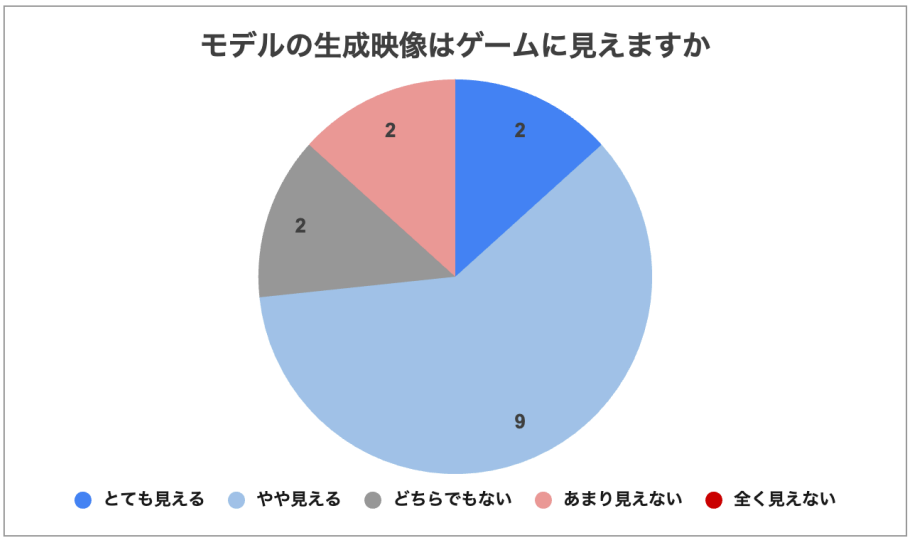

図 4.4　質問 3 回答

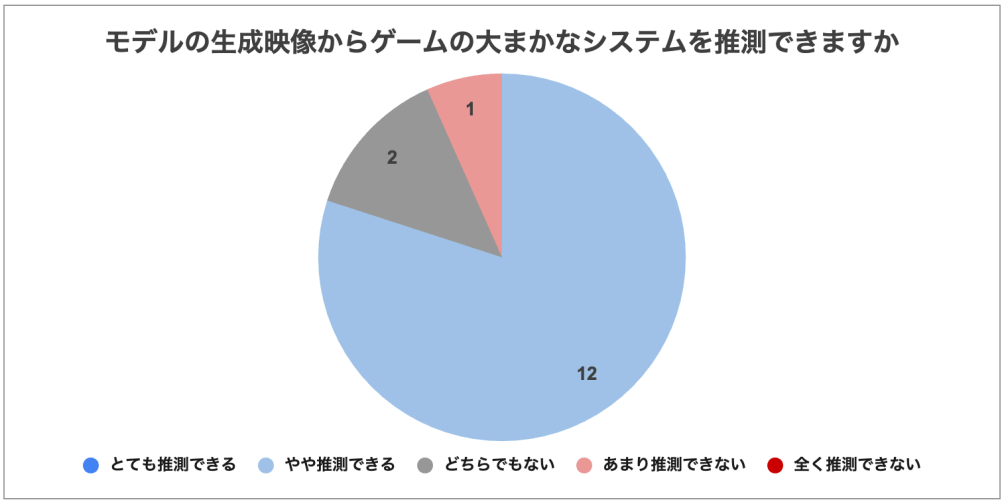

図 4.5　質問 4 回答

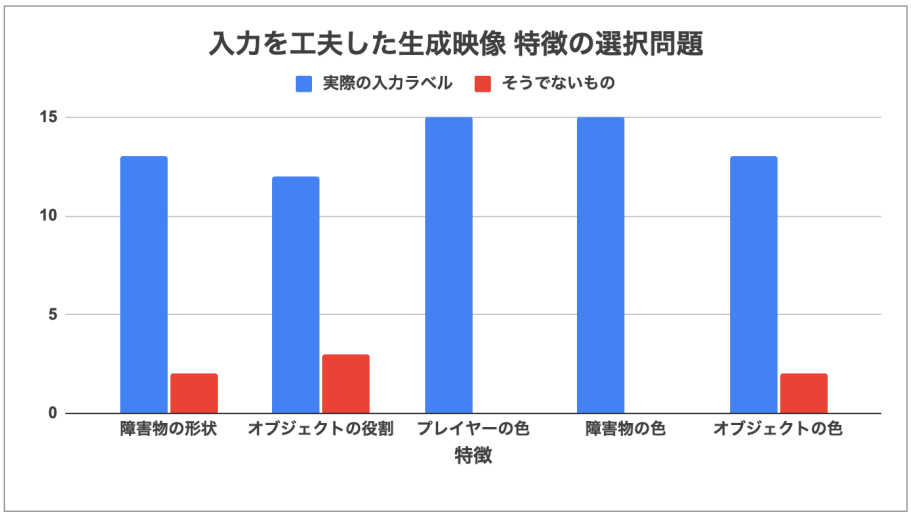

図 4.6　質問 5 回答

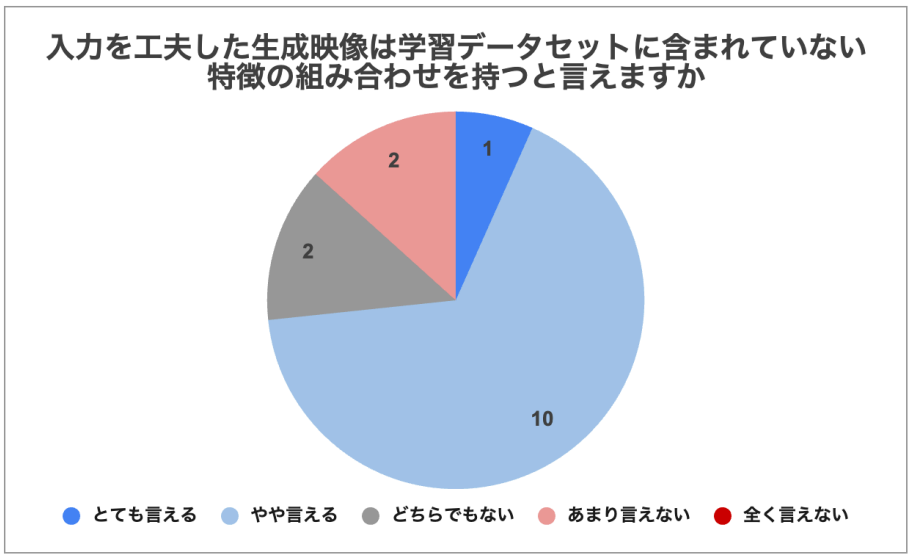

図 4.7　質問 6 回答

4.2　ゲーム開発支援モデルの有用性評価実験

4.2.1　実験目的

本実験の目的は, 生成 AI を用いたゲーム開発支援モデルの有用性を評価することである.

4.2.2　実験内容

本実験の被験者は生成したゲーム映像の質的評価実験と同じである. 被験者には, ゲーム開発に関するアンケートに回答してもらった. 以下にアンケートの設問を示す. なお, 回答形式は表 4.2 に準ずる.

表 4.4　ゲーム開発支援モデルの有用性評価実験 アンケートの設問

質問番号	質問内容	回答形式
1	デジタル・アナログに関わらず1日のうちどの程度ゲームをプレイしますか	択一B
2	ゲームの面白さを評価する際に重要視する項目並べ替え問題	並べ替え
3	デジタル・アナログや規模に関わらずゲームを自主制作したことがありますか	択一C
4	自主制作したゲームのシステムを考えるのに何かを参考にしましたか	択一C
5	ゲームのシステムを考えるのに何を参考にしましたか	自由記述
6	ゲームデザイナであることを想定して,生成AIのアイデアを利用したいと感じますか	択一A
7	その理由はなんですか	自由記述
8	生成AIがゲーム開発のアイデア出しをすることへの懸念点はありますか	自由記述
9	アンケートを通じての意見など	自由記述

また, 質問番号 2 の選択肢は [11] にまとめられているゲームを構成する 4 大要素である, ゲームシステム, ストーリー, ビジュアルおよび技術に設定した.

4.2.3　アンケートの結果

ゲーム開発支援モデルの有用性評価実験の結果を以下に示す. なお自由記述形式の設問に対しては一部抜粋したものを示す.

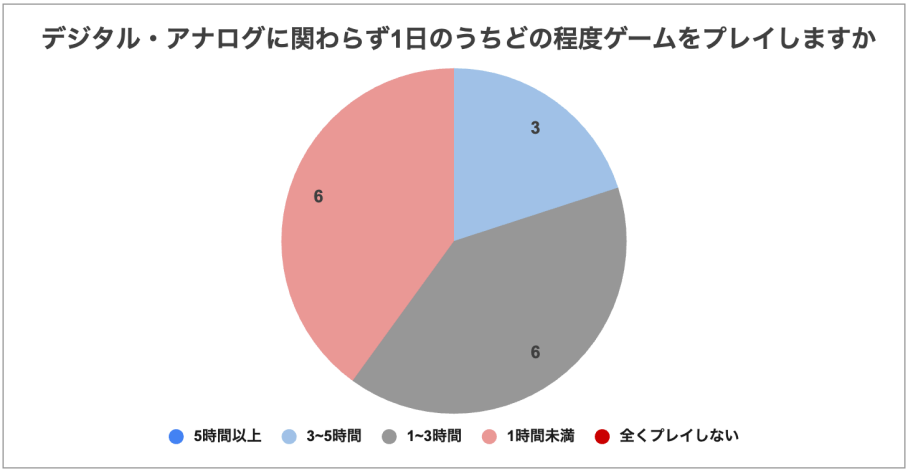

図 4.8　質問 1 回答

表 4.5　質問 2 回答

構成要素	選択順															
	被験者番号															平均
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	
ビジュアル	2	4	3	2	4	3	3	3	3	1	1	2	1	4	2	2.5
ゲームシステム	1	1	1	1	1	2	1	2	2	2	2	1	2	1	1	1.4
ストーリー	4	3	4	4	2	1	4	1	1	4	3	3	3	3	3	2.9
技術	3	2	2	3	3	4	2	4	4	3	4	4	4	2	4	3.2

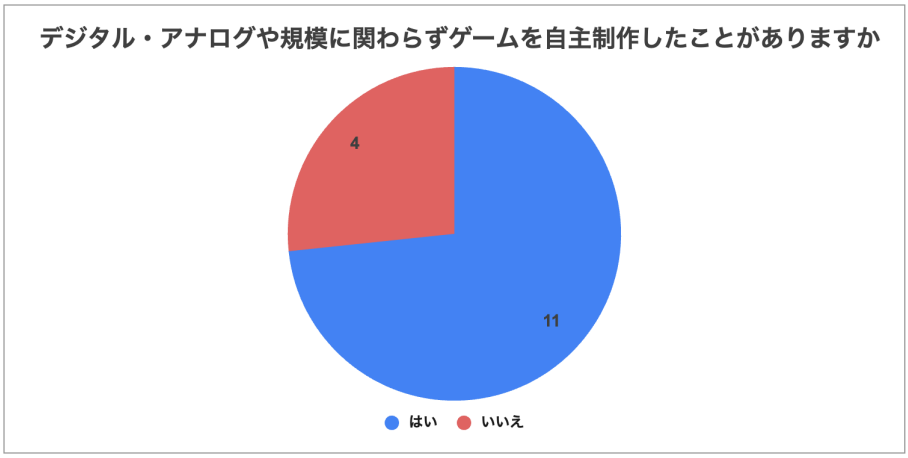

図 4.9　質問 3 回答

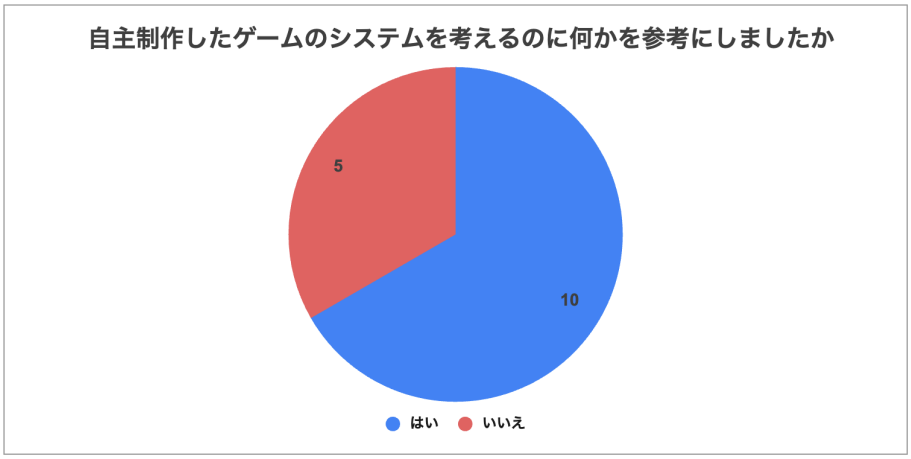

図 4.10　質問 4 回答

表 4.6　質問 5 回答

回答例
自分が普段遊んでいるゲームから連想した
既存のゲーム、ボードゲーム
ボードゲーム全般（著名な作品）、売れたインディーゲームなど
既存の人気のゲーム
Unityroomなどにアップされているインディーズゲームのアイデアやゲームシステムを参考にした

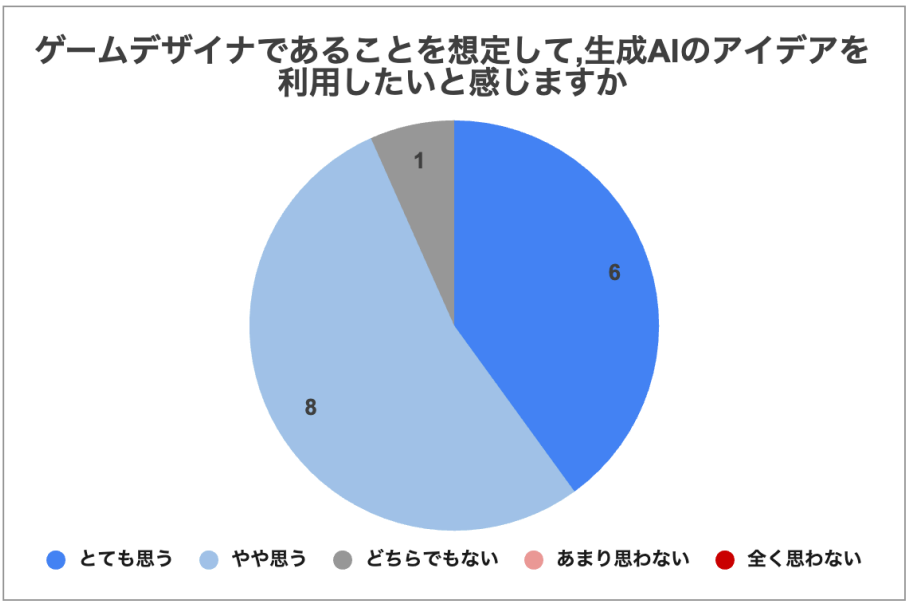

図 4.11　質問 6 回答

表 4.7　質問 7 回答

回答例
自分の思い付かないアイデアを出してくれそうだから
AIは自分が思いつかないアイデアを出したことがある
0から1を考えるのが苦手だから
短時間で大量のアイデアを示してくれるから
近頃はいろんなゲームが作られていて 他の人が考えつかないようなゲームのアイデアを出すことがとても難しくなっているので アイデア出しをAIがサポートしてくれたらとても助かるのではないかと考えたから

表 4.8　質問 8 回答

回答例
過去のゲームに似たゲームができてしまったり ゲームの新規性がなくなってしまう可能性がある
特にない よく話題にのぼる「倫理面に問題あるアイデア」などはこちらで棄却すれば良い
何の模倣をしているのかわからず 権利問題に発展するかもしれない
有用でないものだったら時間の無駄になる
パクリゲームになる

4.3　ゲーム開発経験者へのインタビュー

また更なる議論のために, 特にゲームの自主制作の経験を持つ 3 名の被験者に対してアンケートの回答に関するインタビューを行った. 以下に, 得られた意見の抜粋を示す.

表 4.9　学習データセットについて 回答

回答例
学習データセットは現代的なゲームに比べ スコアやHPのUIがないことからゲーム感が薄い
映像からは憶測程度のゲームシステムがわかる
レトロゲームなイメージでのゲームのゲームには見える どこかでみたことあるようなシステムであるため推測できた

表 4.10　生成したゲーム映像について 回答

回答例
切り取られている部分が悪く生成された映像は短さを感じた 例えばこのデータセットではプレイヤーがビームを発射する瞬間や、 アイテムを取る瞬間がないとゲームのアイデアを想像できない
組み合わせとしては学習データセットに含まれていないが、 特徴ごとは他のものを参考にしているので、まだ新たなゲームとは言い切れない
生成AIによる映像生成の課題である映像の長さについて、 十分にゲームシステムを把握できる長さであれば短くても良いと思う このデータセットでは16フレームでも十分に感じた
特徴が二つのどちらか不明確な部分もあり、スロープと階段状の両方の特徴を持っているとも言える これをうまく学習できていないと取るか、組み合わせて新たな特徴を生み出したと取るか
単純なゲームで実験したが、特徴を複数にしたりしたときに、 もっと予想外のシステムが出てきそうで、 それがアイデアの発案につながると感じた

表 4.11　ゲームの面白さについて 回答

回答例
謎解きのようなアナログゲームは一度やってしまうともう楽しめないから、 既存のものを使えない。制作者側の意見も踏まえた時に、 ゲームシステムのアイデアが一番重要であると感じる
将棋やカードゲームなど、技術がなくてもできるゲームでも、 システムが愛されているために長くプレイされているゲームは多くある
インディーゲームを含め、新たなゲームはどんどん出てきており アイデア勝負の状況で面白いゲームを少人数だけで作るのは大変

表 4.12　ゲームシステム発案作業について 回答

回答例
ゲームシステムを自分で考えるのは難しい、 ほとんどは既にあるものの組み合わせ
ゲームのシステム発案作業において 自分が面白いと思う既存のシステム同士を組み合わせたら どうなるのかを考えるのが最初のステップ
何かを作る際に過去の作品や誰かのアイデアを 参考にしない人なんていないのではないか

表 4.13　生成 AI によるゲーム開発支援システムの利点 回答

回答例
100%オリジナルのゲームはあまりない、新しく斬新な発想のゲームでも 何かと何かの組み合わせだったりする アイデアはかけ合わせて作るもの、その種がポンポンでるシステムは使い勝手が良い
生成AIと人間の協力という観点では、生成AIの弱点である倫理的な問題を人間が排除できる その一方で生成AIのみではアイデアの斬新さでも厳しい、 あくまでサポート役としての役割が望ましい

表 4.14　生成 AI によるゲーム開発支援システムの懸念点 回答

回答例
生成AIは学習データありき、新しいアイデアを出したとしても権利の問題や似ているものがあって、 絶対に新しいとは言い切れない、既存のものの組み合わせ感が出てしまう
特にインディーゲームのような少人数開発では ゲームシステムの発案作業が負担であり、AIに頼りたい気持ちもあるが、 出たアイデアを形にできる実装能力との兼ね合いが難しい
結局データセットに含まれるアイデアをもとにしているため、 著作権問題の可能性、およびそれを排除する作業の負担も生じる

第 5 章

考察/議論

5.1 性能評価実験

5.1.1 FID スコアについて

表 3.3 から, 全てのカテゴリにおいて学習データセット同士の比較に比べ数十倍程度の数値が得られた. FID スコアとは 2 つの画像群において, 各画像の持つ特徴をエンコードしたベクトルの平均と分散を比較した値であり, 小さい方が優れている. 参考として, 学習データセットのうち共通の特徴をいくつか持つカテゴリ 1 と 6 で FID スコアを計算したところ, 180 程度となった. そのため本実験で得られた結果は学習データセットの違うカテゴリ間の比較と同程度の結果であり, 共通の特徴を含むゲーム映像のフレーム画像としての品質は達成している.

5.1.2 ACD について

表 3.4 から, 通常の学習データセットの映像との比較で平均して 1.74 倍程度大きな ACD が得られた. また, 1 フレームおきに画像を並べることで映像の一貫性が悪くなることは自明であるが, 1 フレームおきの映像との比較で平均して 0.91 倍程度小さい ACD が得られた. さらに, ランダムに画像を並べたものを映像として計算すると ACD が 20 倍程度になることから, 生成映像は明らかに一貫性を持っており, その程度は学習データセット

の映像をそのまま並べたものと 1 フレームおきに並べたものの間であることが分かる.

各評価指標の値が悪くなった原因として, 本研究では計算資源の観点から十分に GAN の損失関数が安定していない状態で学習を終了してしまったこと, 各特徴から得た埋め込み表現を単純に連結したものを全体の埋め込み表現としたことおよび設計モデルでは MoCoGAN と PGGAN のアーキテクチャを単純に組み合わせただけでありアーキテクチャのチューニングを行えなかったことが原因として考えられる.

5.2 ユーザー評価実験

5.2.1 生成したゲーム映像の質的評価実験について

アンケートから, 学習データセットとして用意した映像について, ゲーム映像と認識することができ, ゲームシステムを推測しやすいものを準備できていることが確認できた. 一方で, 生成したゲーム映像について, ゲーム映像として認識できた程度およびゲームシステムを推測できる程度がそれぞれ低下したものの, ポジティブな回答がおおよそであるため, 生成した映像は学習データセットと見比べて同等の品質を持っていることがわかった. また, 入力を工夫して生成させた映像は平均して 90.6% の被験者がカテゴリ 1 の特徴を認識しており, モデルが各特徴の入力ラベルを正しく生成映像に反映させていることがわかった. しかし, あくまで既存の特徴の組み合わせを変えただけであり, 新たな特徴を生成したとは言い難いことから質問 6 ではネガティブな回答も得られた. このことから, 設計したモデルが生成したゲーム映像はゲームデザイナが十分にゲームと認識できる品質を持っている一方で, 全く新しいゲームシステムを生み出すレベルには達しておらず, 学習データセットおよびモデルのアーキテクチャについてさらなるチューニングが必要であると考える.

5.2.2 ゲーム開発支援モデルの有用性評価実験について

アンケートから, 全ての被験者が 1 日に少しでもゲームをプレイしており, ゲームに関するアンケートの被験者として適当であると考える. ゲームの面白さを評価する際, 重要視する構成要素を並べ替えた結果の比較から, 全ての被験者がゲームシステムを 1 番か 2 番目に選択しており, 平均の値も他の 3 要素に比べて順位が高いことが分かった. これは本研究の主張である, グラフィックのクオリティが頭打ちになりユーザーの評価がシステムに対してより大きな重点を置いていることに合致する. また, 被験者のうち 11 名がデジタルまたはアナログゲームを自主制作した経験があり, そのうち 90.9% にあたる 10 名がゲームシステムの発案において参考にしたものがあると答えた. このことから, 特に趣味レベルの個人ゲーム制作において参考とするシステムが存在していることが分かり, これはアイデア出しを行うゲーム開発支援モデルの重要性を示唆している. また, 被験者が参考にしたものは多岐にわたるが, 共通しているのは既に存在するゲームシステムから, 自身のアイデアを追加して新たなゲームシステムを発案しているという点である. ほぼ全ての被験者が自分のアイデアの限界や時間効率の観点から, 生成 AI が提案するゲームシステムのアイデアを利用することを望んだが, そのアイデアの新規性や権利問題に不安を感じていることがわかった.

5.2.3 ゲーム開発経験者へのインタビューについて

インタビュー結果から, 学習データセットの映像は横スクロールという既存のゲームから連想しやすい特徴に設定したことで, 被験者に対してゲームシステムをある程度推測させることのできるものを準備できたことが分かった. 一方で, 本研究において映像の学習難度の観点から省略した UI が無いことで不自然に思った被験者もおり, 将来的に UI を合わせてゲーム映像を生成/提案するシステムが必要であると考える. また生成した映像がゲームのどの場面を表現しているかによって, システムのアイデア

になり得ない場合があることが分かった. ゲームデザイナがシステムのアイデアを映像から得るためには, その挙動を映像内で確認できる必要があるためである. 本研究で設定した学習データセットはその単純さから, 短い映像でもシステムが推測できるが, より複雑なゲームを学習させる際にはさらに長い映像を高品質で生成できるようなモデルを準備する必要があり, GAN を含む映像の生成 AI 技術の発展に期待したい. また, 生成した映像は学習データセットに含まれない特徴の組み合わせを持つが, あくまで組み合わせが新しいだけであり, 特徴自体は既存のものであるという意見が多く見られた. ゲームの面白さについて, 近年インディーゲームが多数ヒットしており, そのアイデアの斬新さは重要視されていることが, 実際にゲーム製作を行なっている被験者から分かり, ゲームシステムの重要性を確認することができた. また, 全ての被験者がゲームシステムの発案において何かを参考にすることを前提としており, 既に存在するものを学習し, そこから新たなシステムのアイデアを抽出するという本モデルの方向性は実際のゲーム制作と似通っており妥当であると考える. インタビューを通じて, 生成 AI によるゲーム開発支援システムについて, 多くのアイデアを同時に大量に出力できる点が評価され, 人間がそのアイデアのブラッシュアップを行うという過程のもと, システムの有用性を示す評価を得ることができた. 一方で著作権問題や, 映像で表されるシステムを形にするクリエイターの手腕, それら問題を排除する際に発生する作業の負担などの懸念点を確認できた.

第 6 章

結論

本章では, 結論および実験結果や考察を踏まえた本研究についての議論を述べる.

6.1 結論

本研究では GAN を用いてゲーム映像を学習し, 埋め込み表現を工夫することで, 学習データセットに存在しない特徴の組み合わせを持つゲーム映像を生成することができた. また, ゲーム開発支援モデルの有用性アンケートでは, ゲーム開発におけるシステム等のアイデアの重要性および生成 AI がゲームデザイナにもたらす利点や懸念点を確認することができた.

6.2 本研究の問題点について

本研究で訓練したモデルは, 簡単なゲーム映像のデータセットを制作し, ゲームが持つ特徴を埋め込み表現として GAN に学習させた. その際計算資源や学習難度の観点から, 右スクロールという共通のシステムを持つゲーム映像のみを学習対象とした. 実験結果から, 埋め込み表現を工夫することで, 学習データセットに存在しない特徴の組み合わせを持つゲーム映像を生成することを確認できたが, 全く新たなシステムを持つゲームというよりは, 既存のゲームが持つ特徴をマージしたものを生成するという側面が強くなってしまった.

6.3 将来課題

本研究では, 計算資源の観点から十分に GAN の損失関数が安定していない状態で学習を終了してしまった. そのためより洗練された GAN の映像生成アーキテクチャを用い, 学習の収束を確認した上でゲームが持つ特徴を理解していることを確認する必要がある. また, より幅広いゲームシステムをモデルに学習させることで, システムのもつ表現空間の中からプレイヤーがまだ見ぬゲームシステムを生成 AI が見つけ出すことができると考える. さらに, 自然言語処理分野における大規模言語モデルの有用性が明らかになっている中で, テキストベースでゲームが持つ特徴を学習し, 新たなシステムのアイデアを出力させるというアプローチも有効であると考える.

参考文献

[1] LeCun Y, Cortes C "The MNIST Database of Handwritten Digit" 1998, "http://yann.lecun.com/exdb/mnist/"

[2] Mehdi Mirza, Simon Osindero "Conditional Generative Adversarial Nets", arXiv:1806.07268, 2014

[3] Diederik P Kingma, Max Welling "Auto-Encoding Variational Bayes", The International Conference on Learning Representations, 2014

[4] Jascha Sohl-Dickstein, Eric A. Weiss, Niru Maheswaranathan, Surya Ganguli, "Deep Unsupervised Learning using Nonequilibrium Thermodynamics", Proceedings of Machine Learning Research, 2015

[5] Tero Karras, Timo Aila, Samuli Laine, Jaakko Lehtinen "Progressive Growing of GANs for Improved Quality, Stability, and Variation", The International Conference on Learning Representations, 2018

[6] Sergey Tulyakov, Ming-Yu Liu, Xiaodong Yang, Jan Kautz "MoCoGAN: Decomposing Motion and Content for Video Generation", The IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2017

[7] Seung Wook Kim, Yuhao Zhou, Jonah Philion, Antonio Torralba, Sanja Fidler "Learning to Simulate Dynamic Environments with GameGAN", The IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020

[8] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, Aaron Courville "Improved Training of Wasserstein GANs",

The International Conference on Learning Representations, 2018
[9] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, Sepp Hochreiter ”GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium”, Neural Information Processing Systems, 2017
[10] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, Xi Chen ”Improved Techniques for Training GANs”, Neural Information Processing Systems, 2016
[11] Jesse Schell 著, 塩川 洋介 監訳, 佐藤 理絵子 訳, ゲームデザインバイブル, 2019, 729p

謝辞

本研究を進めるにあたり, 多くの助言をして頂いた中島達夫教授および実験に協力していただいた皆様に深く感謝申し上げます.