修士論文　　2022 年度

# 水エフェクトの自動生成のための深層学習における輪郭線の推移研究

東京工科大学大学院

バイオ・情報メディア研究科

メディアサイエンス専攻

梁 兆楼

修士論文　　2022 年度

# 水エフェクトの自動生成のための深層学習における輪郭線の推移研究

指導教員　渡辺 大地 教授

東京工科大学大学院

バイオ・情報メディア研究科

メディアサイエンス専攻

梁 兆楼

# 論 文 の 要 旨

| | |
|---|---|
| 論文題目 | 水エフェクトの自動生成のための<br>深層学習における輪郭線の推移研究 |
| 執筆者氏名 | 梁 兆楼 |
| 指導教員 | 渡辺 大地 教授 |
| キーワード | 2D アニメ、水エフェクト、深層学習、LSTM、AutoEncoder、中割り |


[**要旨**]

アニメーションにおける水のエフェクトとは、波や着水のしぶきなど液体の動きの表現である。通常の 2D アニメはアニメーターが演出指示の下、手描きで完成される。最近 3D シミュレーションとレンダリングを使い、水のエフェクトを生成するアニメ作品もある。しかし、手描き作画は芸術性や意匠性が高く、数枚フレーム間の違いが大きく、線の数が多いと思われる。そのため、制作に時間がかかる。ここ数年、深層学習を用いた 2D 画像処理は多数の研究が発表されている。2D アニメキャラクターのイラストを生成する研究は多くの注目を集めているが、アニメ作画、特にエフェクト描画に関する研究は少ない。

本研究は、深層学習を用いたより効率的な 2D アニメ原画フレームの生成を目的とする。よく見られるエフェクトの一つであり、水エフェクトを研究対象とする。既存の水エフェクトのカットを収集し、AutoEncoder で水エフェクトの特徴を抽出と復元するネットワークを構築する。そして、LSTM ネットワークで水エフェクトのカットの時系列特徴を学習するモデルを構築する。二つのネットワークと共に、提供された水エフェクトのキーフレームに基づいて、自動的に中割りフレームを生成することを目指している。

今までの研究では、AutoEncoder を用いた画像特徴抽出及び画像復元ネットワークの効果が良かった。しかし、大量な実験を経ても LSTM ネットワークはまだ予想の出力結果を出られなかった。

# A b s t r a c t

| | |
|---|---|
| Title | Transition Study of Contour Lines for<br>Automatic Generation of Water Effects Using Deep Learning |
| Author | Zhaolou Liang |
| Advisor | Taichi Watanabe |
| Key Words | 2D Animation, Water Effects, Deep Learning, LSTM,<br>AutoEncoder, Inbetweening |

[**summary**]

Water effects in animation are liquid motion representations such as splash of water and sea wave. Usually, 2D animator is painting water effects by hand drawing according to the direction of the direction. Recently, more and more animations' water effects is using 3D simulation and rendering. However, hand-drawn paintings are highly artistic and designed, and the largely difference between several frames make the number of lines become larger than usual. Therefore, it takes much time for production. In the past few years, many researches on 2D image processing are using deep learning. The studies of the creating 2D animation character through deep learning have attracted much attention. There are few studies on the process of anime painting, especially the process of effects drawing.

The purpose of this study is to generate 2D animation frames more efficiently using deep learning. Water effect, which is one of the common effects, is the research object in our work. We collect the cuts of existing water effect, and construct a network to extract and restore features of water effect using AutoEncoder. We also construct a model to study the time series features of water effect cuts using LSTM network. Together with two networks, we can achieve the goal of automatically generating inbetweening frame based on key frames of the water effect.

In our study, the performance of image feature extraction and image restoration network using AutoEncoder were good. However, the LSTM network has not obtained the expected output yet.

# 目 次

# 図 目 次

# 第 1 章

# はじめに

## 1.1　背景と目的

アニメーション制作におけるエフェクトは、非生物形態の自然または超自然の現象 (表現) がある。その中で自然現象には風、火、水、雷、煙、爆発などがある。現実世界にはない超自然現象として、レーザーや魔法などがある。

2D アニメのエフェクトは、主にアニメーターによって手描きで制作されている。3D アニメやゲームでは CG シミュレーションとレンダリングで生成されることが多い。ここ数年、多くの 2D アニメ作品制作現場も、Blender などの 3D ソフトウェアを使用している。作品中のエフェクト部分に 3D シミュレーションとトゥーンレンダリングを使い、その上に手描きのスタイル処理をのせることが多い。

2D アニメのエフェクトには、原画、色付け、アフターエフェクトなどの一連のステップが含まれる。**図 1.1** は手描き原画の例である。原画には、輪郭線、陰影、ハイライト、番号とつめ指示などがある。本研究では、2D アニメのエフェクトの原画に注目する。

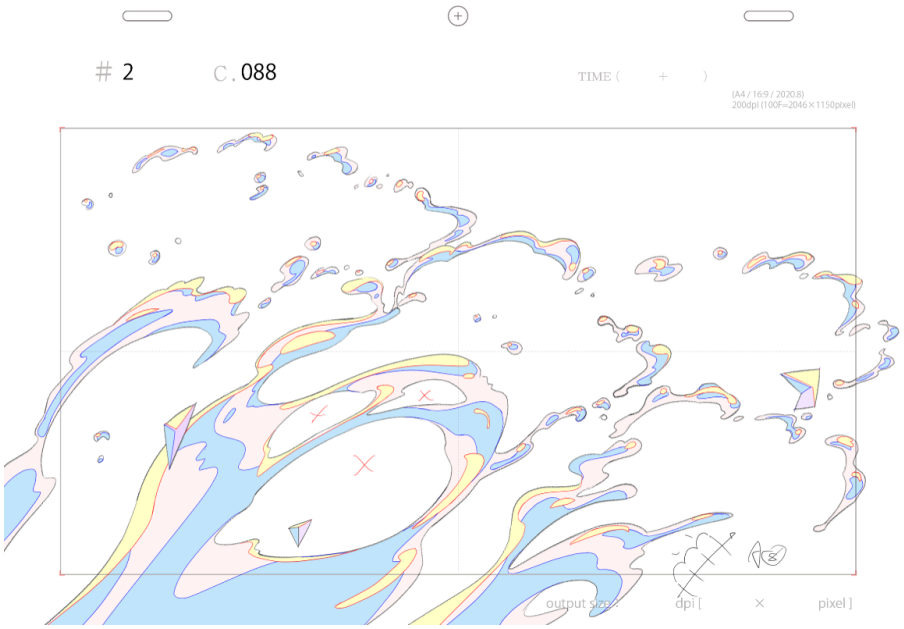


図 1.1　原画の例

エフェクトは様々な特徴がある。水は流れ、散らかすなど動きがあるので、輪郭線が普通の物

体より複雑で、ディテールが多いと思われる。陰影とハイライトの動きは、輪郭線に影響されているが、その変化がまた輪郭線から独立している。アニメーションを描く時、前後フレームの合わせに大量の時間がかかると思われる。陰影とハイライトをふくめた後、一枚一枚の線の量がかなり多くなる。

アニメーション演出家である吉田徹 [1] は「エフェクトは、人間をデフォルメしたキャラクターと違って形がはっきりしていないことも多く、そのためか学生はもとより新人アニメーターも苦手とする方が多いように見受けられます。」と述べている。さらに、吉田徹は 2D 手描きエフェクトの作画は難易度が高いことを指摘している。

2D エフェクトの手描き作画は以下の理由で難しいとおもわれている。

1. エフェクトは固定の形がないので、繰り返しの使用はできない。
2. 不自然にならないように考慮するには時間がかかる。
3. エフェクトの線の量が多いし、原画枚数も多いため、時間も労力もすごくかかる。
4. 動画スタッフはキーフレームのアニメーターのスタイルと表現力の模倣が難しいと思われる。

アニメーターの作業内容の中で、エフェクトを描くにはより多くの精力と長い工期が必要である。しかし、多くのアニメーション制作の工期は緊迫しており、工期不足で品質の低下を余儀なくされ、所望の演出効果が得られない場合がある。複雑なエフェクトは、制作に多大な労力が必要となる。

「アニメーション制作者実態調査 2019」[2] によると、アニメーター 1 日あたりの平均作業時間について、8 時間を超えて働く人の割合が 6 割を超えている。1 ヶ月あたりの平均作業時間について、160 時間を超えて働く人の割合が 8 割を超えている。多くのアニメーターは過労で働いており、労働時間の長さや休憩時間の不足に不満を持っている。

本研究の目標は、数枚水エフェクト原画によって、連続フレームを生成することである。**図 1.2**

は最後の目標の様子である。より効率的な 2D アニメのエフェクトの原画の生成手法の確立である。アニメ制作を省力化することができれば、少しでもアニメーターの負担を減らすことができる。

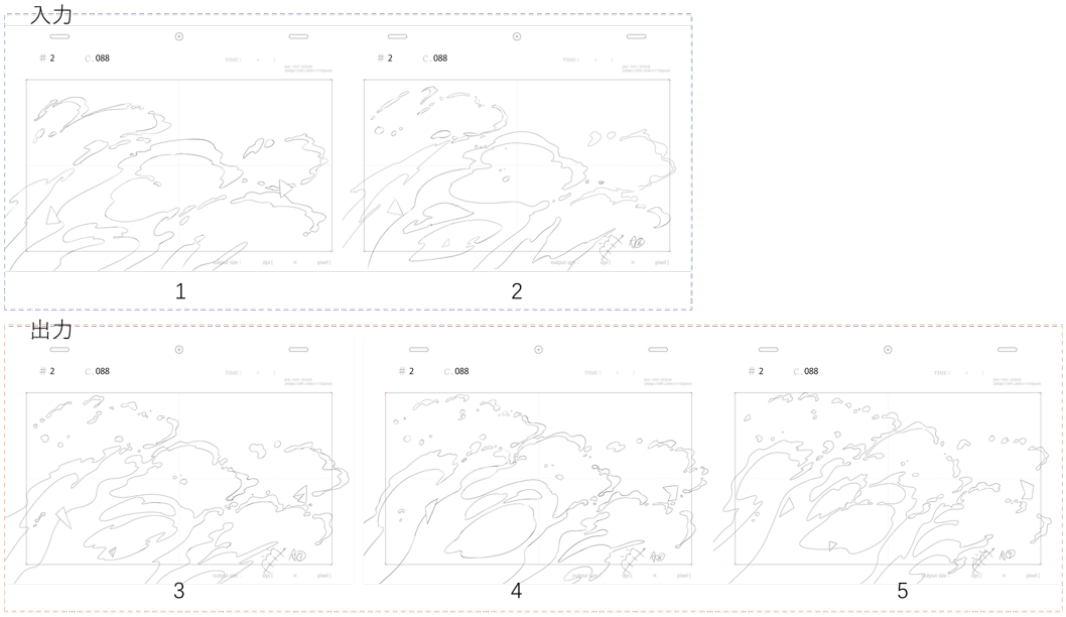


図 1.2　最後の目標の様子

本研究の手法について、以下の流れを提案した。

1. 手描きアニメから水エフェクトのカットを準備する。
2. 水エフェクトの輪郭線と Optical Flow を抽出する。
3. AutoEncoder と LSTM 二つ深層学習モデルを構築する。
4. テストキーフレームに基づいて中割りフレームを生成して評価する。

本研究では、水エフェクトを一連の手法で収集し、訓練データセットに処理した。これらのデータセットを用いて、画像特徴抽出と画像復元のための AutoEncoder モデル、及び時系列データ特徴学習のための LSTM モデルを訓練した。

実験結果について、AutoEncoder モデルの効果は良かったが、LSTM モデルがまだ予想の出力結果を出られなかった。

## 1.2 論文構成

第 1 章では、本研究の背景および意義について説明する。

第 2 章では、本研究の理論と関係がある各分野の関連研究を紹介し、本研究の目的と関係がある既存研究を紹介する。

第 3 章では、本研究の提案手法を述べる。

第 4 章では、水エフェクト生成するために、様々な前処理、AutoEncoder ネットワーク及び LSTM ネットワークをトレーニングする実験である。

第 5 章では、AutoEncoder ネットワーク及び LSTM ネットワークの検証データを入力し、実験結果の分析を示す。

第 6 章では、まとめを述べる。

# 第 2 章

# 関連研究

## 2.1 アニメソフトウエアの自動中割り

CACANi[3] という 2D アニメソフトウエアには自動中割機能がある。原画となるキーフレームを描写後、自動中割ツールを使用することで、アニメーターの描き味を活かしたまま動画を自動生成することができる。アニメーターの詰め指示、軌道線など指定もできる。この機能は Nanyang 大学の Hock Soon Seah など研究者たち [4] の成果に基づき作成されている。Chen Quan[5] の研究は両方の画像ベースとベクターベースの領域で中割りフレームを生成することを目指している。以上の研究は修正した特徴ベースマッチングアルゴリズム (MBFA) の特徴認識技術に基づいている。

ここ数年は「ジョジョ」「はたらく細胞」などのアニメーション制作における CACANi の自動中割機能が使われている。簡単な輪郭線の場合、与えられた最初と最後ベクトル線の大量の点の計算によると、良い効果の中割り線を得ることができる。しかし、数土直志 [6] は「もちろん微妙な表情や、複数のキャラクターの複雑な動きを描かせるのは難しい。既に存在している絵の線を動かすだけであり、新たな線を生み出すわけでもない。」と評価した。本研究は深層学習の手法を通じてアニメーションの中割を実現し、さらに複雑な線を生成することを目標とした。

## 2.2 コンピュータービジョン

ここ数年、深層学習はコンピュータビジョンの分野で多くの優れた研究と応用がある。

例えば CNN を用いた画像分類 [7]、RNN/LSTM を用いたターゲット識別 [8]、GAN を用いたスタイル学習 [9]、Transformer を用いた画像補完 [10] がある。

### 2.2.1 中割りの研究

坂本眞人ら [11][12][13] と石津ら [14] は、コンピュータアニメーションズの中割りアルゴリズムについて一連の考察と研究を行った。彼は Miura[15][16] のアルゴリズムに基づいて徐々に改善を行い、簡単な線画における手と足の曲げの中割り手法を改善した。しかし、動きが大きい 2 フ

レーム間では中割りを計算することが困難である。

深層学習を用いた中割りの研究もある。Yuichi Yagi の研究 [17] は、CNN を用いたトレーニングすることである。一つのカットには、1 と 3 枚目を入力画像として、2 枚目を目標画像としてトレーニングすることである。学習データが少ないので、平行移動や回転でデータを拡大する手法を使用している。しかし、この研究は長い時系列情報を考慮せずに、限られている範囲で動くキャラクターを生成することができる。

### 2.2.2 画像生成の研究

本研究はアニメ分野の研究なので、アニメについての画像生成研究に注目している。画像生成において、GAN という深層学習手法は真偽の判別が困難な画像を生成できることで有名である。StyleGAN[9] は高画質のアニメアバターを生成できる。学習したデータセットはすべてアニメスタイルの顔のため、良い顔画像が生成できるのと比べ、人物の体によって問題が大量に発生する。

人物の体を学習する GAN ネットワークもある。PSGAN[18] は CNN ネットワークを用いて、アニメーションキャラクターのキャラクターデザインによって、人物の体のひざ、手首、腕など特徴を抽出する。抽出した特徴を Unity という 3D ソフトウエアに導入し、3D キャラクターオブジェクトを生成を手伝っている。

2022 年に Stable Diffusion[19][20]、DALLE2[21] など Diffusion 手法 [22] を使用するネットワークは、GAN より正確な構造、より高画質のネットワークである。プロンプトとか呪文とか呼ばれるテキストを入力することで、高解像度のグラフィックが生成されることである。2022 年に最も人気な文字-画像生成モデルである。

AutoEncoder[23][24][25] は最初に Hinton と Sejnowski によって発明された。その中で「Learning internal representations by back propagation」[25] の研究は誤差逆伝播ニューラルネットワークを用い、現在の AutoEncoder の雛形と見なされている。深層学習による改良後の AutoEncoder[26] は徐々に多くの分岐を生み出している。

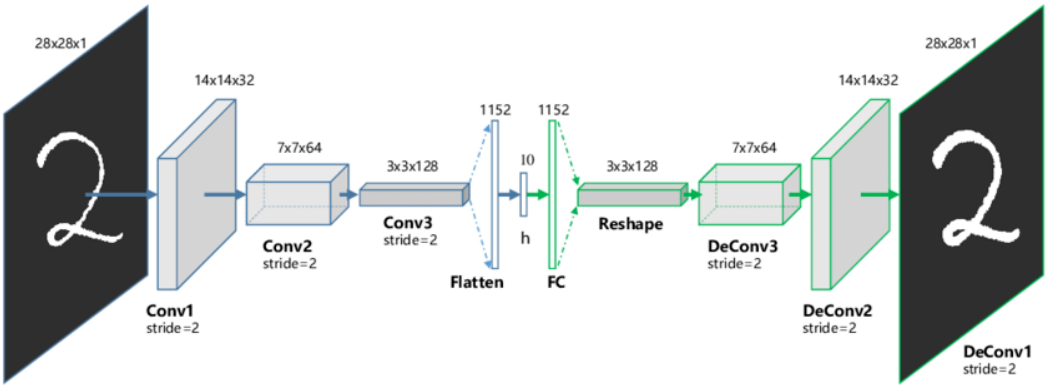


図 2.1　Convolutional Autoencoder の構造

AutoEncoder は画像の高次元特徴を抽出し、特徴により画像を復元できる方法である。これまでの研究では画像ビデオなどの圧縮にも使われてきた。Denoising AutoEncoder は他の画像圧縮手法と異なり、ノイズによって画像をノイズ除去することもできる。Vincent[27] の研究は訓練時の画像にノイズを加え、ノイズがないの元の画像と比較して学習することで、ノイズ除去能力を得る。Jonathan Masci[28] の研究では、畳み込み層を使って特徴を抽出する Encoder とデータ復元するために逆畳み込み層を使用する Decoder 二つに分ける。**図 2.1**(画像出典 [29]) はこの研究における Convolutional Autoencoder の構造である。

以上の AutoEncoder 手法は画像の正確な復元に注目している。また、画像を生成するための重要な生成モデルもある。VAE(Variational Auto-encoder)[30] は、訓練データにはないがそれと類似したデータを求めることにある。ノイズの分布及び分散を学習することにより、画像を生成する時特徴をサンプリングして訓練セットに存在しない図を生成することができる。

循環 AutoEncoder[31][32] は循環ニューラルネットワーク RNN/LSTM を用いた、Encoder と Decoder を構造する。時系列データに対応し、自然言語処理と機械翻訳でよく使われる。

### 2.2.3　動画モーション検出の研究

時系列データの深層学習において、動画モーション検出は多く研究と応用されている。例えば、自動運転の車両と歩行者の検出 [33][34][35]、ビデオ中の人の動作の検出 [36][37] などである。

主に時系列データから情報を抽出して分類や分割などのタスクを行う。一般的な手法は、Single Stream Network[38]、Two Stream Networks[39]、自己注意 [40] などである。

Optical Flow は動画から動く物体を検出する場合によく使用している。動画のピクセル変化情報、つまり動画中の物体の動き情報を保存できる。Joe Yue-Hei Ng[41] の研究では、Optical Flow 画像を結合した車両識別精度が大きく向上することがある。この研究では、実写動画を対象として、元の画像と Optical Flow 抽出した画像をそれぞれに CNN ネットワークで特徴抽出する。次に Feature Pooling 或いは LSTM ネットワークを用いて回帰計算する。最後に特徴に基づいて動き物体を分類する。

# 第 3 章

# 深層学習手法とデータ準備の提案

## 3.1 手描きアニメからの水の動きの抽出

深層学習はデータセットに非常に依存すると思われる。深層学習では、目的に応じてデータセットを準備する必要がある。多くの研究者がデータセットの作成と補充に取り組んでいる。

画像関連の深層学習データセットはいくつかある。例えば手書き数字を認識するためによく使われる MNIST データセット [42] である。画像分類によく使用されるものとして cifar 10 と cifar 100 のデータセット [43] がある。また GPT タスクでは、1419 万枚の画像を持つ ImageNet[44] というコンピュータービジョンデータセットが非常に巨大である。以上のデータセットは、画像分類のタスクに使用されることが多い。

ビデオ動作予測に関するタスクでは、UCF 101[36] や UCF-sport[45] などのデータセットには、大量の人間の運動に関するビデオデータが含まれている。

アニメーション関連の画像データセット、例えば Animesceleb[46] などのアニメ顔データセットは、アニメ顔生成などのタスクに多く用いられ、非常に良い効果を得ている。しかし、以上の多くのデータセットは本研究に適用していない。

手描き水エフェクトにとって最適なデータは、アニメーション制作における動画検査後の原画画像である。つまり中割りした後の画像である。このステップの時系列原画画像のフレーム数が多く、フレーム間の変化が滑らかで、フレームの前後関係を学習しやすいと思われる。各アニメ作品の原画データ、特にデジタルのデータを得ることが非常に難しい。連続的な原画フレームを公開するアニメ会社は非常に少ない。そこで本研究では、既存のアニメーション作品から水エフェクトのカットを選び、3.2 節の手法を用いてカットから水エフェクトの輪郭線を抽出し、この輪郭線を原画線と見なす。本研究で必要なのは、水エフェクトを含むビデオセグメントデータセットである。私たちは最善を尽くしても、関連データセットを見つけることができなかった。

既存のアニメの水の動きデータセットは存在していないので、データセットを自分で収集する必要がある。本研究はアニメ作品やエフェクト紹介動画や sakugaboru[47] などでアニメの水エ

フェクトを収集し、FFmpeg[48] というフリーソフトウェアで編集する。そして、水の動き表現によって簡単に分類する。例えば、波が拡大と類似しているエフェクトを一緒に集める。**図 3.1** は収集したカットの例である。

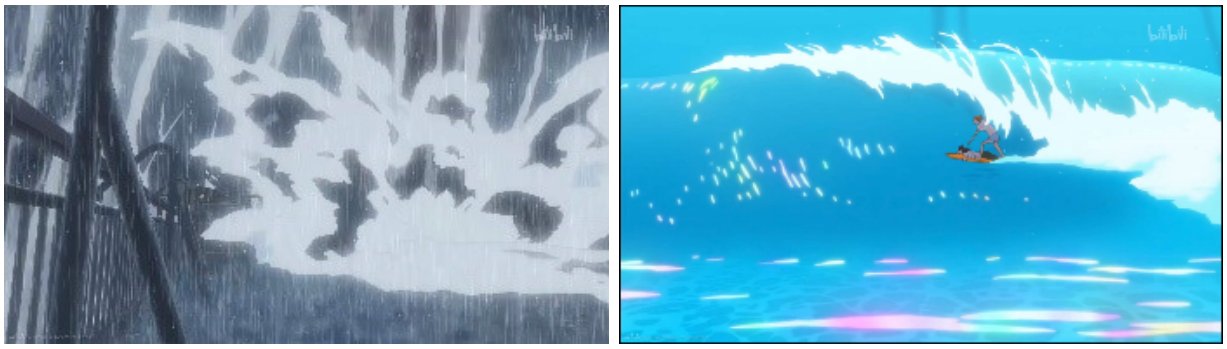

図 3.1　天気の子 (左), きみと、波にのれたら (右)

本研究で収集されたビデオ、GIF、および画像などリソースは研究用にのみ使用し、他のビジネス用途には使用しない。

## 3.2　原画線の前処理

公開されたアニメーション作品から編集と分類したビデオセグメントには、水エフェクトだけでなく、他の人物や物体も含まれている。研究対象を際立たせるために、本研究は前処理を行う。

**図 3.2** はアニメーション作品から水エフェクト部分を検出する手法の説明である。具体的には、水の色相の多くが青色であることを特徴とし、カラーマスクを使えばフレームごとに水の部分を抽出することができる。2D アニメの作画では、輪郭線、かげ線とハイライト線で水エフェクトのエッジ線を描くことがよく使われている。そのため、OPENCV という処理機能をまとめたオープンソースのライブラリの輪郭抽出機能を使い、水エフェクトの輪郭を抽出する。**図 3.3** は元の水エフェクトフレームと抽出したオリジナル画像である。今後の研究では輝度マスクに基づいて陰影とハイライトを分別抽出して学習することが可能である。

ビデオの各フレームの輪郭線を取り出した後、輪郭線情報のみを含む学習データを作成する。

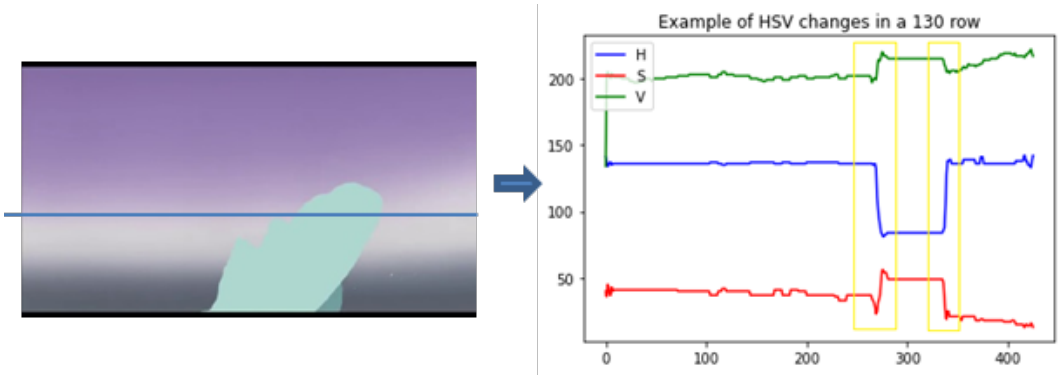


図 3.2　色相特徴に基づく水エフェクト検出

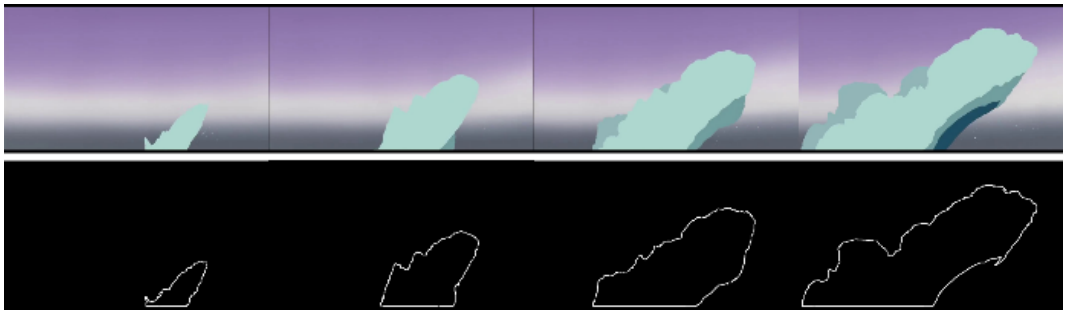

図 3.3　原画線の取り出し

同様に、異なるアニメーション作品における水エフェクトの色値に基づいて、パラメータを変更し、本研究で使用されているデータセットを作成する。

一つずつのフレームを学習すると、水エフェクトの動き情報の一部が失われる。そこで本研究では、動き検出と予測に関する研究でよく使用している手法、Optical Flow を参考にする。ビデオのピクセル変化情報を保存でき、物体の動き情報を保存できると思われる。

Joe Yue-Hei Ng[41] の研究を参考にし、本研究では、水エフェクトの変化に対しても、データセットの補足として Optical Flow 画像を抽出する。**図 3.4** はオリジナル画像と抽出した Optical Flow の可視化画像である。

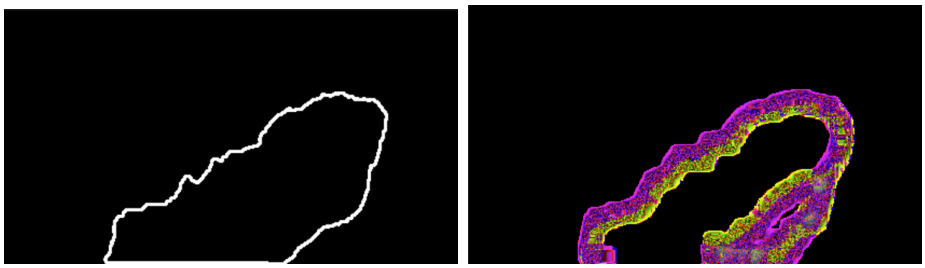

図 3.4　Optical Flow 画像の生成

説明を簡単にするため、Optical Flow を**図 3.4**（右）のように可視化する。黄色部分には輪郭線の消失傾向がある。紫色部分には輪郭線の生成傾向がある。例に示す水エフェクトの輪郭線は黄色から紫色まで移動することを示唆している。元画像と Optical Flow 画像を合わせて、水のエフェクトの特徴を含むトレーニングに適したデータセットを作成する。実際の特徴抽出計算において、Optical Flow 計算した速度場に基づく動きの特徴を抽出する。

## 3.3　前後フレーム間の輪郭線の関係性を探す

**図 3.5** は本研究の深層学習提案手法の全体的にモデルの構築である。オレンジ色の破線で囲んだ AutoEncoder である。青色の破線で囲んだモデルは、LSTM をマスタメソッドとして Encoder と Decoder を用いたシーケンスモデルである。

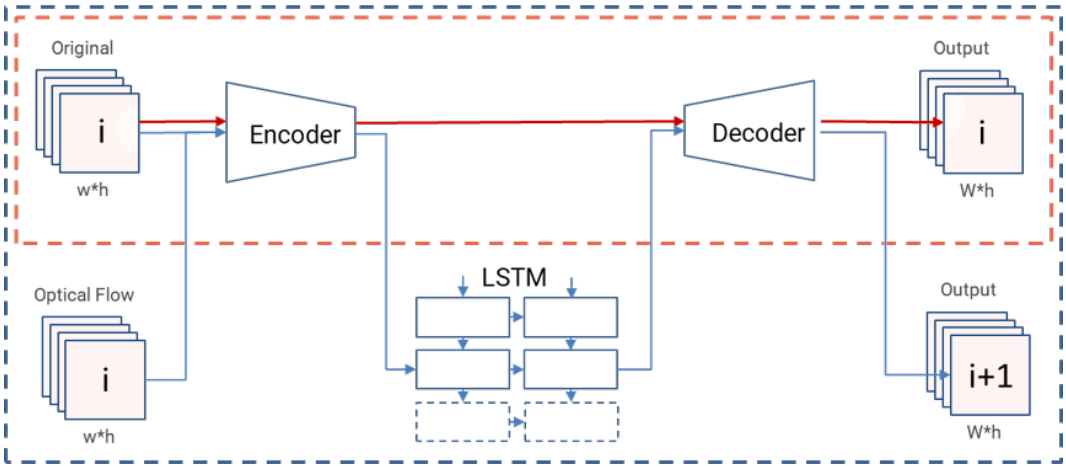


図 3.5　全体的モデル

まず、3.2 節の手法で作成した水エフェクトデータセットを用いて、AutoEncoder ネットワー

クを訓練する。AutoEncoder ネットワークは原画のタイプの画像を特徴抽出し、抽出した特徴を通じて入力画像とほぼ同じ画像を復元することができる。

本研究では、Convolutional AutoEncoder[28] という AutoEncoder の派生モデルを参考し、Maxpooling を畳み込み層の代わりに構築した AutoEncoder ネットワークを提案する。

そして、訓練した AutoEncoder ネットワークは Encoder と Decoder 二つに分け、LSTM ネットワークの前後を組み合わせる。

訓練する時、全体的なモデルの入力は元の時系列画像と Optical Flow 時系列画像である。AutoEncoder の Encoder 部分で学習データの特徴を抽出と圧縮し、特徴データを得ることができる。LSTM 部分で時系列特徴データの時系列特徴を学習し、次の特徴データを予測する。AutoEncoder の Decoder 部分を用いて、予測した特徴データから画像復元する。

### 3.3.1 Encoder 部分

Encoder 部分は FCN ネットワークでデータの特徴を抽出することができる。**図 3.6** は Encoder の構築である。FCN ネットワークは 6 層の畳み込み層を含む。各層のチャネル数は、入力データの 1 から増加した後、減少する。同時に各層の得られた特徴データのサイズは段階的に減少する。最終的には、1 つのチャネルの小さいサイズの特徴データが得られる。このようにすることで、元のピクチャのデータ量を 10% 以下に縮小することができる。

オリジナル画像と Optical Flow 画像はいずれも類似の FCN 多層ネットワークを経由して特徴抽出と圧縮を行う。抽出した特徴ベクターは元のデータより小さくなり、より学習効率が高いと思われる。トレーニングシステムの負荷も小さくなると思われる。

### 3.3.2 Decoder 部分

Decoder は Encoder の逆関数を使い、特徴データから入力画像と同じサイズのデータに復元することである。**図 3.7** は Decoder の構築である。Decoder は 3 層の逆畳み込み層である。逆畳

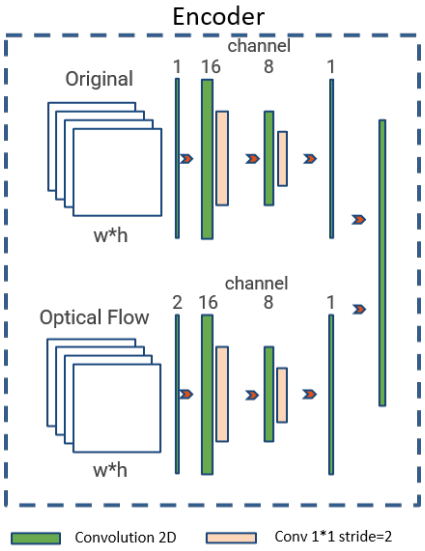


図 3.6　Encoder の構築

み込み層のチャネル数は Encoder のチャネル数と逆である。同時に各層で得られた特徴データのサイズは段階的に増加する。最後に元の画像のサイズに復元することができる。この復元には損失がある [27]。

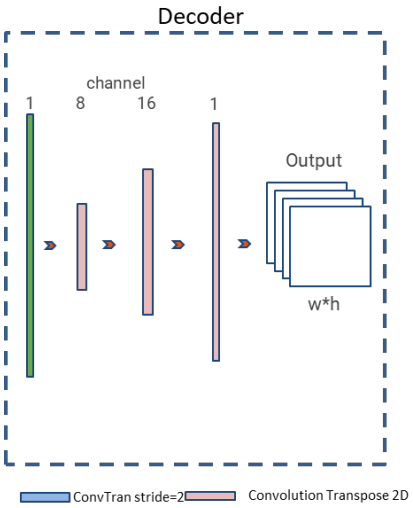


図 3.7　Decoder の構築

LSTM の出力特徴データは損失を計算しやすいが、人々が生成した画像の画質を主観的に評価することが出来ない。Decoder を用いて、LSTM の出力特徴データから復元した画像に基づいて評価しやすくなると思われる。

### 3.3.3 LSTM 部分

LSTM ネットワーク [49] はシーケンス特性を保存できる深層学習ネットワークである。自然言語処理 [50] に多く用いられ、ワードをベクトルと見なすこと [51] で、文をワードのシーケンスとして再帰学習することである [52]。意味の理解と問答などの目的を実現する [53]。LSTM はシーケンスデータに対応し、長い記憶を保存と学習することができると思われている。時系列データの予測もできる [54]。本研究の画像のシーケンスは、自然言語処理における文処理と類似している。

LSTM はゲート機構 (gated) を用いて単純な RNN をアップグレードすることで、長距離依存性を捉え、勾配消失の問題を緩和している。**図 3.8**(文献番号 [55]) に LSTM の構成図を示す。

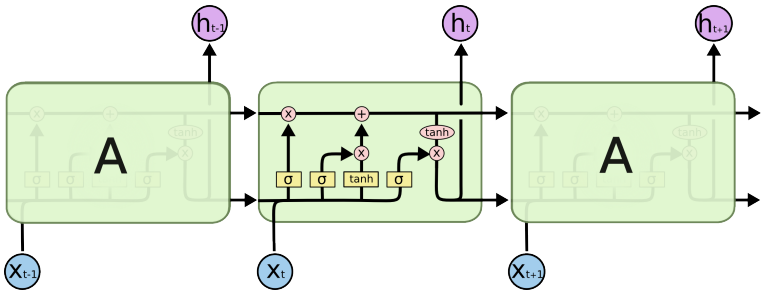


図 3.8　LSTM の中に 4 つの相互作用層

LSTM ネットワークの各層は式 (3.1) のように計算する。LSTM は長期記憶 $c_t$ と短期記憶 $h_t$ という 2 つの隠れ層状態を持っている。W は入力-隠れ層の学習可能な重みである。b は入力-隠れ層の学習可能なバイアスである。LSTM の各ユニットへの入力は、現在の入力 $x_t$ と、前の入力の長期記憶 $c_{t-1}$ および短期記憶 $h_{t-1}$ である。RNN に比べて、忘却ゲート $f_t$、入力ゲート $i_t$、出力ゲート $o_t$、セルゲート $g_t$ の 4 つのゲートを持つ。

$$
\begin{aligned}
i_t &= \sigma\left(W_{ii}x_t + b_{ii} + W_{hi}h_{t-1} + b_{hi}\right) \\
f_t &= \sigma\left(W_{if}x_t + b_{if} + W_{hf}h_{t-1} + b_{hf}\right) \\
g_t &= \tanh\left(W_{ig}x_t + b_{ig} + W_{hg}h_{t-1} + b_{hg}\right) \\
o_t &= \sigma\left(W_{io}x_t + b_{io} + W_{ho}h_{t-1} + b_{ho}\right) \\
c_t &= f_t \odot c_{t-1} + i_t \odot g_t \\
h_t &= o_t \odot \tanh\left(c_t\right)
\end{aligned}
\tag{3.1}
$$

ここで、$\sigma$ は sigmoid 関数である。符号 $\odot$ はアダマール積 (Hadamard product) である。

**図 3.9** はシーケンスフレームを LSTM に代入と処理する流れである。LSTM ネットワークを訓練する時、LSTM の出力特徴データと次の時刻の入力特徴を比べ、損失関数を計算し、バックプロパゲーション (Backpropagation) を行う。

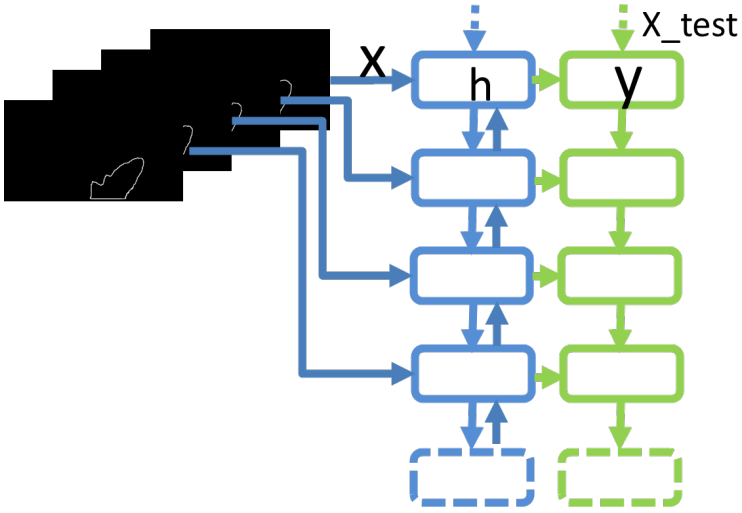


図 3.9　LSTM に対する画像シーケンス入力

## 3.4　評価基準の提案

本研究では、長さが 48 フレーム程度の水エフェクトの輪郭線シーケンスを学習データとする。すべての学習データを 8:2 にトレーニングデータ、テストデータと検証セットに分ける。検証セットは、予測された後続フレームと検証データの類似度を基準に評価する。

本研究の目標により適合するために、水エフェクトのキーフレームを使用し、LSTM ネットワークで学習完了したモデルを用いて、中割りフレームを生成する。アニメーターや視聴者などが中割りフレームの生成品質を主観的に評価する。

# 第 4 章

# 実験

## 4.1 開発環境

前章で述べた手法を実装し、検証を行った。開発および検証に用いた PC のスペックを**表 4.1** に示す。使用したソフトウエアおよびライブラリを**表 4.2** に示す。

表 4.1 開発環境

| | |
|---|---|
| OS | Windows 10 |
| CPU | Intel ® Core™ i7-10700F |
| Memory | 16GB DDR4 |
| GPU | GeForce RTX 3060 12GB |
| 開発言語 | Python |

表 4.2 使用したソフトウエアおよびライブラリ

| 名前 | 機能 |
|---|---|
| FFmpeg | 動画の編集及び解像度など調整 |
| OPENCV | フレーム前処理及び学習データ構築 |
| Numpy | 画像の行列計算 |
| Pytorch | 深層学習ネットワーク構築 |

## 4.2 学習データを収集と前処理

本研究では、20 本以上の水エフェクトあるアニメ作品、GIF、MAD 動画を収集し、FFmpeg[48] を用いた水エフェクト部分のカットを切り出した。シングルデータのサイズを統一するために、OPENCV[56] を用いた 2 秒間 48 フレームの長さにカットした。画像解像度を統一するために、640 × 480 の選択枠でカットの水エフェクトをできるだけ切り取った。

学習データ量を拡張するために、二つ手法を行った。一つ目は、AutoEncoder ネットワークに畳み込み層を構築しやすくするため、ランダム中心点の 256 × 256 選択枠で時系列フレームを切り取った。これには 4 つのメリットがある。

1. 縦横不一致が AutoEncoder の画像復元に与える影響を回避できる。

2. 画像全体の情報を AutoEncoder で学習することを回避し、学習目標を簡略化する。
3. 一連の試みを経た後、256 × 256 のサイズのフレームは、水エフェクトの一部のディテールと動きをよく保存することができる。
4. ランダム中心点切り取りは、同じ水エフェクト動画の中で多くの異なるサンプルを得ることができる。

二つ目は同じカットの異なる開始点でそれぞれ切り出した。LSTM ネットワークに対して、異なる開始点の時系列フレームは、他の時系列フレームを使用するのと同等の効果で、異なる特徴を学習することができる。

切り出した時系列フレームの例は**図 4.1** に示す。

図 4.1　切り出した時系列フレームの例

OPENCV の findcontours 手法により、3.2 節の手法でカットの各フレームを水エフェクトの輪郭線のバイナリイメージとして抽出した。**図 4.2** はバイナリイメージの例である。

OPENCV の calcOpticalFlowFarneback 手法により、各フレームの Optical Flow 画像を生成した。各フレーム Optical Flow 画像の画像サイズと位置が輪郭線のバイナリイメージに対応する。

生成したバイナリイメージと OpticalFlow 画像を合わせて、Numpy[57] を用いた学習データに構築した。

構築したデータは、8:2 の割合に基づいてランダムに訓練データとテストデータに分けられる。本研究の深層学習ネットワークは教師なしネットワークなので、トレーニング時の検証データは訓練データ自身である。各フレームの学習後の結果は、次のフレームの元画像と検証される。

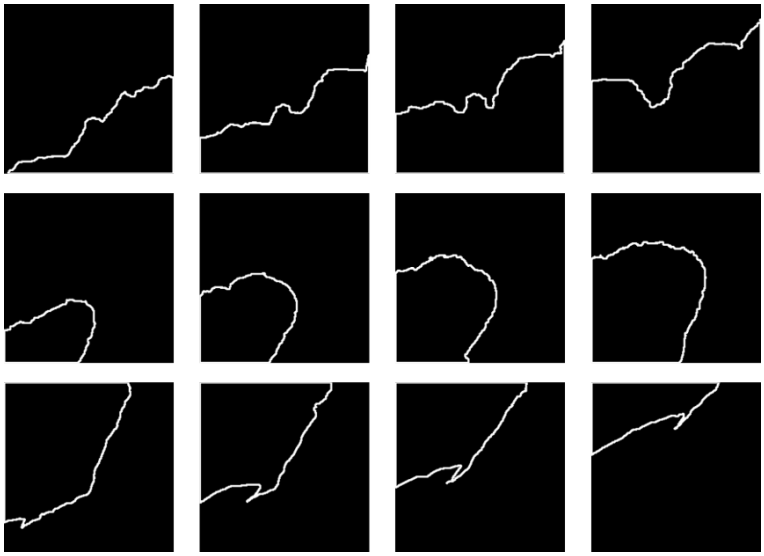

図 4.2　バイナリイメージの例

## 4.3　AutoEncoder ネットワークの構築

Pytorch という深層学習フレームワークを使用し、AutoEncoder ネットワークを構築した。AutoEncoder ネットワークの Encoder 部分は FCN ネットワーク [58] を参考し、多層畳み込み層を構築した。構造を**表 4.3** に示す。256 × 256 × 1(width × height × channel) サイズの元の画像を Encoder に入力し、32 × 32 × 1 のサイズの特徴画像を得ることができる。

Decoder 部分は Encoder 部分の逆畳み込み操作なので、構造を**表 4.4** に示す。32 × 32 × 1 のサイズの特徴画像を 256 × 256 × 1 サイズの画像に復元することができる。

AutoEncoder をトレーニングするために、画像の復元効果を評価することが必要である。本研究では、復元画像と元の画像をピクセルずつ比較するために、損失関数が MSELoss[59] を選択

表 4.3　Encoder の構造

| | in channels | out channels | kernel size | stride |
|---|---|---|---|---|
| conv2d 1 | 1 | 16 | 3 | 1 |
| conv2d 2 | 16 | 16 | 1 | 2 |
| conv2d 3 | 16 | 8 | 3 | 1 |
| conv2d 4 | 8 | 8 | 1 | 2 |
| conv2d 5 | 8 | 1 | 3 | 1 |
| conv2d 6 | 1 | 1 | 1 | 2 |

表 4.4　Decoder の構造

| | in channels | out channels | kernel size | stride |
|---|---|---|---|---|
| convtran2d 1 | 1 | 8 | 3 | 2 |
| convtran2d 2 | 8 | 16 | 3 | 2 |
| convtran2d 3 | 16 | 1 | 3 | 2 |

する。

MSELoss(Mean Square Error Loss) は平均二乗誤差の損失関数のことである。式 (4.1) によって復元画像と元の画像の平均二乗誤差を計算し、誤差逆伝播学習し、パラメータを自動的に調整することができる。

$$\ell(x, y) = mean(L) = mean(\{l_1, ..., l_N\}^T), l_n = (x_n - y_n)^2. \tag{4.1}$$

式 (4.1) の中で、$l_i$ は生成したピクセル値 $x_n$ とターゲットピクセル値 $y_n$ の二乗誤差である。一つのフレームにおけるピクセル値の二乗誤差の平均値は損失とする。

以下の**図 4.3** に AutoEncoder の試作結果を示す。上の 3 つの図は入力されたオリジナル画像であり、下の 3 つの図はそれぞれの AutoEncoder ネットワークで圧縮と復元された画像である。Encoder で抽出した特徴パラメータ数は 1024 である。

Decoder の最後の層の活性化関数は tanh を使用しており、出力データを [0,1] の範囲にマッピングしているため、ほとんどの非活性化領域 (黒の領域) をマッピングした後は 0.5 であり、出力画像ではグレーになる。

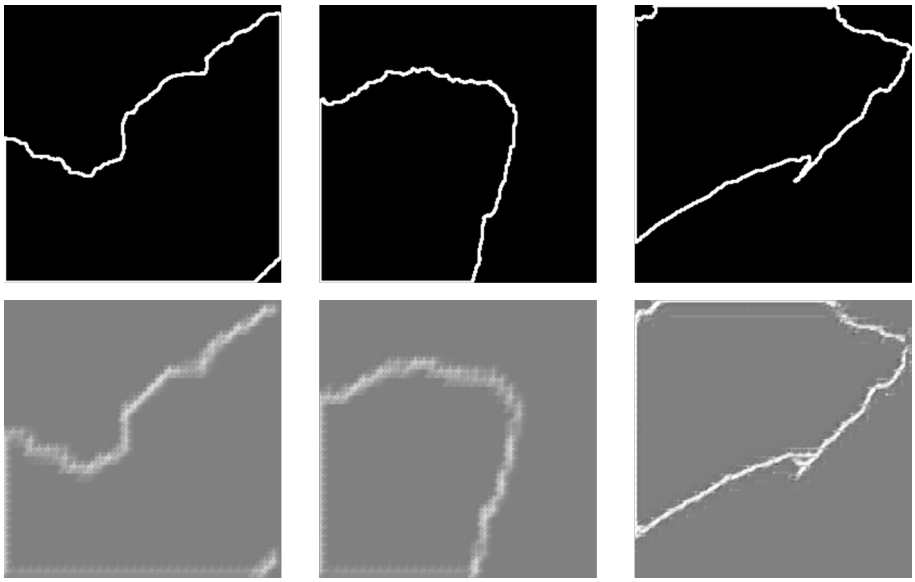

図 4.3　AutoEncoder の出力結果の例 (1024 特徴パラメータ)

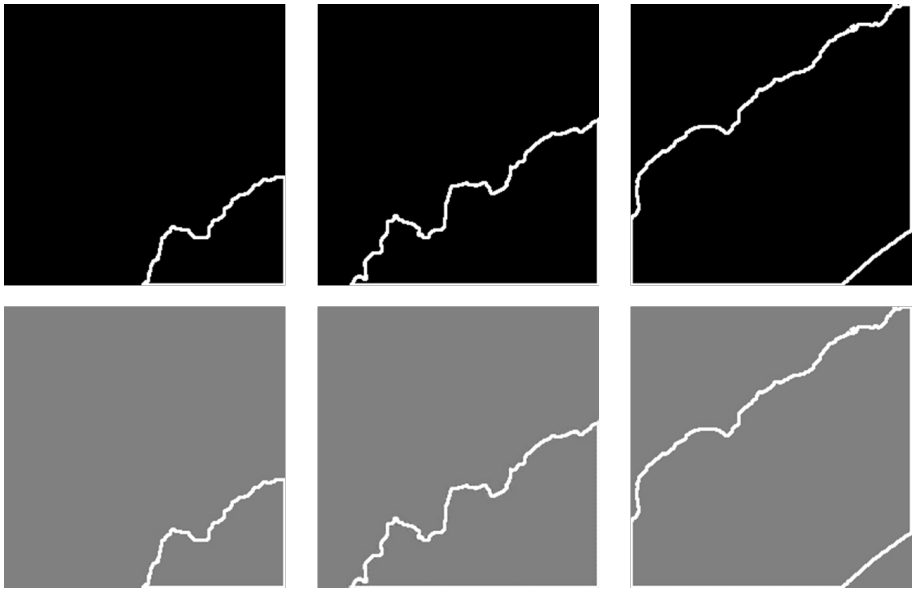

図 4.4　AutoEncoder の出力結果の例 (32768 特徴パラメータ)

本研究では、多数の AutoEncoder ネットワークの構造を試みた。AutoEncoder は損失のある画像圧縮手法なので、特徴パラメータが多くなるほど画像復元の効果が良くなる。**図 4.4** は 32768 特徴パラメータの場合の画像復元の効果である。

Encoder を通じて生成された特徴パラメータは、LSTM ネットワークの入力データとして使用される。入力データのサイズが LSTM ネットワークの総パラメータ数に大きく影響するため、特

徴パラメータが多い場合 GPU メモリを超えやすくなる。したがって、次の実験では 1024 パラメータ量の Encoder の出力を用いた。

AutoEncoder モデルを独立学習できた後、Encoder 部分と Decoder 部分のモデルパラメータを固定する。LSTM ネットワークのトレーニングが AutoEncoder モデルのパラメータに影響しないようにする。

## 4.4 2 層 LSTM ネットワークの構築

Pytorch というフレームワークを用いて 2 層の LSTM ネットワークを構築した。**表 4.5** は LSTM ネットワークの構造である。

表 4.5 LSTM の構造

| | in channels | out channels | batch size |
|---|---|---|---|
| LSTM 1 | 1024 | 4096 | 1 |
| LSTM 2 | 4096 | 1024 | 1 |

学習データは Encoder を経て $(1 \times 1 \times 1024)$ サイズの特徴パラメータに Flatten される。48 フレームの時系列フレームは一つ batch の時系列入力データ $48 \times 1 \times 1024$ になる。本研究では、多数の batch size での入力データを試みた。

元の画像フレームと Optical Flow 画像フレームデータをそれぞれ Encoder で特徴抽出する。Encoder で特徴抽出と圧縮した特徴データ (サイズ : $48 \times n \times 1024, n$ は $batchsize$) を入力し、式 (4.2) に基づいてモデルをトレーニングする。

$$y_n = LSTM\left(CAT\left(x_{original,n}, x_{opticalflow,n}\right)\right) \tag{4.2}$$

その中の $CAT(a, b)$ 関数は二つの特徴データを既存の次元 (軸) に沿って結合する関数である。

本研究では、損失関数を以下の三つを試した。損失をゼロに近づけることで予測値と正解値の差が少なくなる。

1. MSELoss(Mean Square Error Loss)(式 (4.1))。AutoEncoder ネットワークと同じ損失関数。
2. CosineEmbeddingLoss(式 (4.3) 式 (4.4))。この損失は、コサイン距離を用いて 2 つの入力が類似しているかどうかを測定するために用いられ、非線形埋め込み学習や半教師あり学習などに用いられる。
3. HingeembeddingLoss(式 (4.5))。この損失は通常、2 つの入力が類似かどうかの測定に使われる。

$$\ell(x,y) = \begin{cases} 1 - \cos(x_1, x_2) & \text{if } y = 1, \\ \max\{0, \cos(x_1, x_2) - margin\} & \text{if } y = -1. \end{cases} \tag{4.3}$$

その中に、

$$\cos(\theta) = \frac{A \cdot B}{\|A\| \|B\|} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n} (A_i)^2} \sqrt{\sum_{i=1}^{n} (B_i)^2}} \tag{4.4}$$

式 (4.3) に $x_1, x_2$ は生成したピクセル値とターゲットピクセル値である。$y$ は異なる比較基準のラベルである。

$$\ell(x,y) = mean(L) = mean\left(\begin{cases} x_n & \text{if } y_n = 1, \\ \max\{0, \Delta - x_n\} & \text{if } y_n = -1. \end{cases}\right), L = \{l_1, ..., l_n\} \tag{4.5}$$

式 (4.5) に $x_n$ は生成したピクセル値とターゲットピクセル値の差の絶対値である。$y$ は異なる比較基準のラベルである。

本研究で解決したいのは分類問題ではなく、正解のカテゴリラベルに基づく正確率を計算することはできないと思う。LSTM ネットワークの出力評価について、二つ提案がある。

1. 本研究構築した LSTM ネットワークは、出力データのサイズと入力データのサイズが同じなので、各特徴パラメータの差を最小化することを提案する。つまり MSELoss(式 (4.1)) 損失関数に基づく計算する。

2. LSTM ネットワーク出力データが Decoder を用いて画像を復元する。復元した画像と次のフレームの画像は各ピクセルの差を最小化することを提案する。これも MSELoss(式 (4.1)) 損失関数で計算する。

# 第 5 章

# 評価

本研究の深層学習モデルは AutoEncoder ネットワークと LSTM ネットワークの 2 つの部分に分けられる。ここで、AutoEncoder 部分は単独で訓練し、LSTM モデルの入出力処理層として Encoder と Decoder を得る。

## 5.1 AutoEncoder の輪郭線復元実験

AutoEncoder には多くの分岐がある。本研究では、以下のような AE の構築方式を試みた。

1. 三つ層の全結合層 (Fully-connected layer) を用いた構造した Encoder と Decoder。
2. 三つ層の畳み込み層 (Convolutional layer) と MaxPooling 層を用いた構造した Encoder と Decoder[28]。
3. 前述の Maxpooling を畳み込み層の代わりに構築した Encoder と Decoder。

3 つ目の方式は 1 つ目の方式に比べてパラメータ量が大幅に減少し、グラフィックメモリの節約に有利である。また、画像データを畳み込んで処理するのが得意で、演算速度が速い。Springenberg[60] の研究によると、3 つ目の方式は 2 つ目の方式に比べて分類タスクの精度が向上する。また、Decoder の構築が簡易化になる可能である。本研究では最終的に 3 つ目の方式も選択する。

本研究では、異なる畳み込み層の層数、異なる kernel サイズ、および異なる出力特徴パラメータ数を試みた。4.3 節では、**図 4.3** と**図 4.3** に 2 つの設定の画像復元効果を紹介した。AutoEncoder ネットワークの構造設定の一部パラメータを**表 5.1** に示す。この二つネットワークのトレーニング誤差推移とテスト誤差推移グラフは**図 5.1** に示す。

表 5.1　AutoEncoder の構造の比較

| 畳み込み層数 | epoch | kernel size | flatten | MSE(test) |
|---|---|---|---|---|
| 9 | 100 | 3 | 1024 | 1.6579 |
| 6 | 100 | 3 | 32768 | 0.0132 |

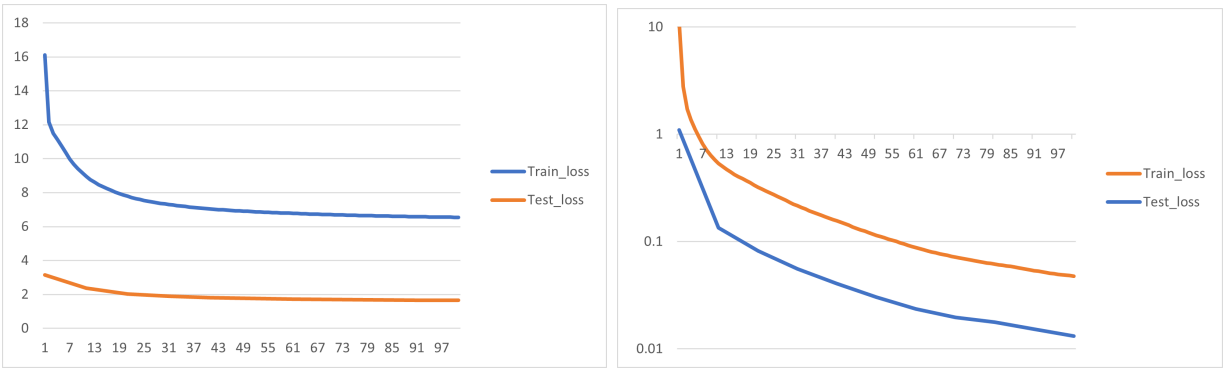


図 5.1　1024 特徴パラメータ (左)、32768 特徴パラメータ (右)

特徴パラメータ数の多い AutoEncoder モデルが画像復元効果は高いが、LSTM の入力データサイズを考慮すると、復元効果と特徴パラメータ数を取捨する必要がある。本研究では、9 つの畳み込み層の AutoEncoder ネットワークを選択する。

## 5.2　単一配列の輪郭線生成実験

学習データとして水エフェクトフレームは、AutoEncoder モデルの Encoder 部分で処理し、1024 次元の特徴データに変換される。LSTM モデルのトレーニングする時 AutoEncoder モデルのパラメータが変化しないので、同じ入力画像は常に同じ特徴データを得ると思う。逆に、近い特徴データを持つフレームは、Decoder 部分復元後のフレームも類似しているべきだと思われる。

本研究では、3 節で説明した提案手法に基づいて多くの試みを行った。しかし、最終的には理想的な出力画像が得られなかった。

例えば、2 層 LSTM ネットワークを使用し、次のフレームの特徴パラメータをターゲットとして 100 回のトレーニングを行った後も、無意味な出力結果が表示される。**図 5.2** はフレームずつの出力結果である。トレーニング誤差推移グラフは**図 5.3** に示す。

もう一つの例として、1 層 LSTM ネットワークと 1 層全結合層を使用し、LSTM ネットワークの出力特徴データ (1024 特徴パラメータ) が Decoder 部分で画像を復元する。そして、次のフレームの元の画像をターゲットとして 50 回のトレーニングを行った。最終的には**図 5.4** ような

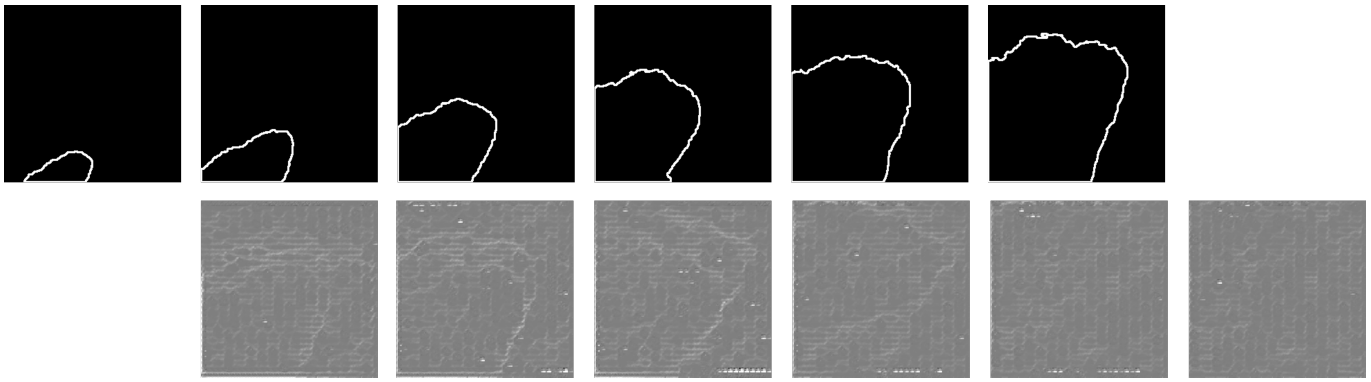

図 5.2　フレームずつ出力結果

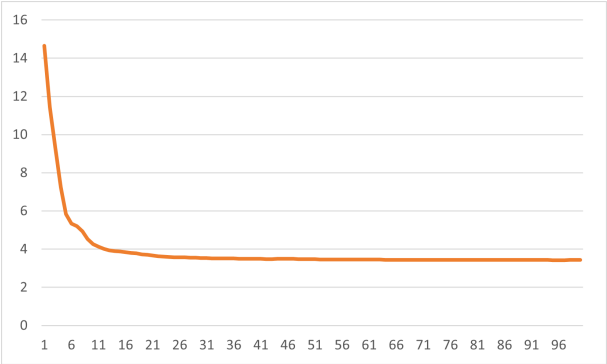


図 5.3　トレーニング誤差推移グラフ

無意味な出力になった。トレーニング誤差推移グラフは**図 5.5** に示す。

図 5.4　第 4 フレームの出力結果

一つ目の LSTM ネットワークのパターンでは、トレーニングの誤差推移は正常に見えるが、実際に Decoder を通じて出力した画像の輪郭線が認識できなかった。二つ目の L パターンでは、ト

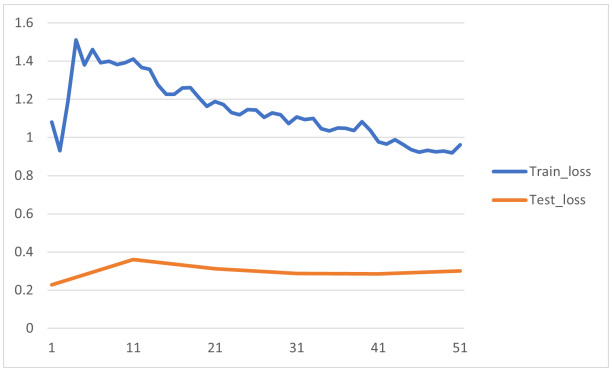


図 5.5　トレーニング誤差推移とテスト誤差推移グラフ

レーニングの誤差推移とテストの誤差推移は最小化できなかった。出力した画像の輪郭線も認識できなかった。

以上のような理想的な出力が得られない理由は以下の 3 つあると思う。

1. 「特徴値が似ていれば類似画像を復元できる」という仮定は間違っているかもしれない。今後 Decoder 画像復元に関する数学原理についてもう一回学習と検討を行う。
2. LSTM モデルは、過去に入力された時系列情報を記憶し、ある特徴にまとめることができ、ビデオタスクの識別と分類に適している。しかし、画像などの高次元情報の生成に適していない可能性がある。
3. VAE[30] 手法における「確率分布を学習し、サンプリングにより生成する」などの追加手段を加える必要があるかもしれない。

## 5.3　キーフレームによる中間フレームの生成実験

数枚の連続入力されたフレームから後続フレームを成功に生成することができれば、1 枚だけの入力されたフレームに基づく後続フレームを生成することを試したいと思う。Srivastava の研究 [32] より、後方学習と同時に前方の入力を再復元し、モデルを調整すると、より良い予測効果が得られる。後続のキーフレームに基づいて双方向に学習することで中割りフレームを生成するこ

とができると思う。

しかし、前述の単一配列の輪郭線生成実験では期待の結果が得られなかったので、残念ながらキーフレームによる中間フレームの生成実験はまだ進行していなかった。

今後の研究について、2 つ手法を検討している。1 つ目は、画像データを輪郭線の特徴ベクタ及び速度場データに置き換えて学習する手法である。ネットワークが複雑な画像情報からピクセルの変化ルールを学ぶことを期待するよりも、意味がより明確な輪郭線ベクタデータと速度場データに基づくネットワークを訓練することで、より良い効果が得られるかもしれない。

2 つ目は、画像データを用いて、VAE[30] 及び Stable Diffusion[19] のような手法で画像変化の確率分布及び分散を学習し、逆サンプリングにより予測画像を生成する手法である。この手法では、特定の意味を持つ新しい画像の生成をうまく実現することができる。水エフェクトの輪郭線の変化はある確率分布に合うはずで、ニューラルネットワークにこのような確率分布を学習させることができれば、適切な輪郭線を生成することができるかもしれない。

# 第 6 章

# まとめ

本研究では、手描き 2D アニメのエフェクトを研究対象として、ここ数年コンピュータービジョン分野で人気がある深層学習手法に基づいて、手描き原画を生成する研究である。

本研究では、2D アニメの水エフェクトに対して、水エフェクトに関する動画などを収集し、データセットを作成する。水エフェクトの動き情報を組み込むために、水エフェクトの動きの Optical Flow を計算し、その特徴も後続の深層学習ネットワークと一緒に訓練する。

本研究は水エフェクトデータセットで AutoEncoder に基づく画像特徴抽出及び画像復元ネットワークを訓練する。LSTM ネットワークを使用し、水エフェクトの時系列情報を学習し、後続の画像を予測したい。2 つのネットワークを組み合わせて、水エフェクトを生成することを目指している。

今までの研究では、AutoEncoder を用いた画像特徴抽出及び画像復元ネットワークの効果が良かった。しかし、大量な実験を経ても LSTM ネットワークはまだ予想の出力結果を得られなかった。

今後の研究では、画像データとベクタデータの 2 つの方向から検討を行う。

# 謝辞

本論では、深層学習を用いたアニメ生成について研究を進めてきたが、まだ十分だとは言えない。本稿の作成にあたって、ご指導、ご協力していただいた先生、友人の皆様方に心より感謝の意を申し上げます。まず、指導教員である渡辺先生に深く御礼を申し上げます。日頃渡辺先生から親切にご指導やご佃撻をいただき、また研究方法や論文の書き方など方面で有益なご助言をいただき、心より深く感謝の意を申し上げたいと思います。渡辺先生と阿部先生には、本論文の研究開始より完成に至るまで、手間を惜しまないで、たくさんのご意見やご指導をいただきました。また、本論文に詳細で有益なご助言をいただきました柿本先生、三上先生と菊池先生に心より感謝いたします。本論文の中間審査発表時においても、貴重なご助言をいただき、厚くお礼申し上げます。最後に、コンピューターサイエンス専攻の先輩王梓萱さんと研究室の皆さんに多大なご協力をいただき、大学院の二年間を楽しく過ごして、感謝の誠をささげる次第であります。

# 参考文献

[1] 吉田徹. 吉田流！アニメエフェクト作画. 株式会社ボーンデジタル, 2016.

[2] 一般社団法人日本アニメーター・演出協会 (JAniCA). アニメーション制作者実態調査報告書 2019. http://www.janica.jp/survey/survey2019_report.html. 参照: 2023.1.28.

[3] CACANi. 自動中割 cacani 日本語マニュアル web ページ. https://cacani.info/. 参照: 2022.11.8.

[4] Eiji Sugisaki, Hock Soon Seah, Fumihito Kyota, and Masayuki Nakajima. Simulation-based in-between creation for cacani system. pp. 1–1, 2009.

[5] Quan Chen. Computer-assisted in between generation for cel animation. 2008.

[6] sudotadashi. 「アニメの絵を自動で描く」ai が出現——アニメーターの仕事は奪われるのか. https://www.itmedia.co.jp/business/articles/1910/21/news029_5.html. 参照: 2023.1.29.

[7] Forrest N Iandola, Song Han, Matthew W Moskewicz, Khalid Ashraf, William J Dally, and Kurt Keutzer. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and¡ 0.5 mb model size. *arXiv preprint arXiv:1602.07360*, 2016.

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. pp. 770–778, 2016.

[9] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. pp. 4401–4410, 2019.

[10] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[11] 坂本眞人, 友添直子ほか. コンピュータアニメーションの中割りアルゴリズムに関する一考察. *Memoirs of the Faculty of Engineering, Miyazaki University*, Vol. 35, pp. 281–285, 2006.

[12] 坂本眞人, 石崎裕一朗, 飯干淳志ほか. アニメーションの中割りアルゴリズムによるキャラクターの描写. 宮崎大學工學部紀要, Vol. 46, pp. 295–298, 2017.

[13] 坂本眞人, 黒木脩平ほか. コンピュータアニメーションの中割りアルゴリズムに関する一考察 2. 宮崎大學工學部紀要, Vol. 47, pp. 339–342, 2018.

[14] 石津貴弘, 那須航, 坂本眞人, 飯干淳志. アニメーションの中割りアルゴリズムによるキャラクターの描写 II. 2019.

[15] Takeo Miura, Junzo Iwata, and Junji Tsuda. An application of hybrid curve generation: cartoon animation by electronic computers. In *Proceedings of the April 18-20, 1967, spring joint computer conference*, pp. 141–148, 1967.

[16] William T Reeves. Inbetweening for computer animation utilizing moving point constraints. *ACM SIGGRAPH Computer Graphics*, Vol. 15, No. 3, pp. 263–269, 1981.

[17] Yuichi Yagi. A filter based approach for inbetweening, 2017.

[18] Koichi Hamada, Kentaro Tachibana, Tianqi Li, Hiroto Honda, and Yusuke Uchida. Full-body high-resolution anime generation with progressive structure-conditional generative adversarial networks. pp. 0–0, 2018.

[19] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models, 2021.

[20] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10684–10695, June 2022.

[21] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 2022.

[22] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, pp. 2256–2265. PMLR, 2015.

[23] David H Ackley, Geoffrey E Hinton, and Terrence J Sejnowski. A learning algorithm for boltzmann machines. *Cognitive science*, Vol. 9, No. 1, pp. 147–169, 1985.

[24] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *nature*, Vol. 323, No. 6088, pp. 533–536, 1986.

[25] DE Rumelhart, GE Hinton, and RJ Williams. Learning internal representations by back propagation. *Parallel Distributed Processing*, Vol. 1, .

[26] Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. *science*, Vol. 313, No. 5786, pp. 504–507, 2006.

[27] Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. pp. 1096–1103, 2008.

[28] Jonathan Masci, Ueli Meier, Dan Cireşan, and Jürgen Schmidhuber. Stacked convolutional auto-encoders for hierarchical feature extraction. In *International conference on artificial neural networks*, pp. 52–59. Springer, 2011.

[29] colah's blog. Autoencoder. https://kyonhuang.top/blog/autoencoder-intro-1/. 参照: 2023.1.29.

[30] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[31] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*,

2014.

[32] Nitish Srivastava, Elman Mansimov, and Ruslan Salakhudinov. Unsupervised learning of video representations using lstms. In *International conference on machine learning*, pp. 843–852. PMLR, 2015.

[33] Chenxu Luo, Xiaodong Yang, and Alan Yuille. Exploring simple 3d multi-object tracking for autonomous driving. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 10488–10497, 2021.

[34] Rui Qian, Xin Lai, and Xirong Li. 3d object detection for autonomous driving: a survey. *Pattern Recognition*, Vol. 130, p. 108796, 2022.

[35] Yingjie Wang, Qiuyu Mao, Hanqi Zhu, Yu Zhang, Jianmin Ji, and Yanyong Zhang. Multi-modal 3d object detection in autonomous driving: a survey. *arXiv preprint arXiv:2106.12735*, 2021.

[36] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012.

[37] Will Kay, Joao Carreira, Karen Simonyan, Brian Zhang, Chloe Hillier, Sudheendra Vijayanarasimhan, Fabio Viola, Tim Green, Trevor Back, Paul Natsev, et al. The kinetics human action video dataset. *arXiv preprint arXiv:1705.06950*, 2017.

[38] Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei. Large-scale video classification with convolutional neural networks. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 1725–1732, 2014.

[39] Karen Simonyan and Andrew Zisserman. Two-stream convolutional networks for action recognition in videos. *Advances in neural information processing systems*, Vol. 27, , 2014.

[40] Li Yao, Atousa Torabi, Kyunghyun Cho, Nicolas Ballas, Christopher Pal, Hugo

Larochelle, and Aaron Courville. Describing videos by exploiting temporal structure. In *Proceedings of the IEEE international conference on computer vision*, pp. 4507–4515, 2015.

[41] Joe Yue-Hei Ng, Matthew Hausknecht, Sudheendra Vijayanarasimhan, Oriol Vinyals, Rajat Monga, and George Toderici. Beyond short snippets: Deep networks for video classification. pp. 4694–4702, 2015.

[42] Li Deng. The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, Vol. 29, No. 6, pp. 141–142, 2012.

[43] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[44] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

[45] Mikel D Rodriguez, Javed Ahmed, and Mubarak Shah. Action mach a spatio-temporal maximum average correlation height filter for action recognition. In *2008 IEEE conference on computer vision and pattern recognition*, pp. 1–8. IEEE, 2008.

[46] Kangyeol Kim, Sunghyun Park, Jaeseong Lee, Sunghyo Chung, Junsoo Lee, and Jaegul Choo. Animeceleb: Large-scale animation celebheads dataset for head reenactment. In *European Conference on Computer Vision*, pp. 414–430. Springer, 2022.

[47] SAKUGABORU. sakugabooru - a booru dedicated to sakuga videos and images. https://www.sakugabooru.com/. 参照: 2022.11.8.

[48] FFmpeg. Ffmpeg - a complete, cross-platform solution to record, convert and stream audio and video. https://ffmpeg.org/about.html. 参照: 2023.1.28.

[49] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computa-*

*tion*, Vol. 9, No. 8, pp. 1735–1780, 1997.

[50] Yoshua Bengio, Réjean Ducharme, and Pascal Vincent. A neural probabilistic language model. *Advances in neural information processing systems*, Vol. 13, , 2000.

[51] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[52] Alex Graves. Generating sequences with recurrent neural networks. *arXiv preprint arXiv:1308.0850*, 2013.

[53] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. *Advances in neural information processing systems*, Vol. 27, , 2014.

[54] Carmina Fjellström. Long short-term memory neural network for financial time series. *arXiv preprint arXiv:2201.08218*, 2022.

[55] Kyon Huang. Understanding lstm networks. http://colah.github.io/posts/2015-08-Understanding-LSTMs/. 参照: 2023.1.29.

[56] OPENCV. Opencv provides a real-time optimized computer vision library, tools, and hardware. it also supports model execution for machine learning (ml) and artificial intelligence (ai). https://opencv.org/. 参照: 2023.1.28.

[57] Numpy. The fundamental package for scientific computing with python. https://numpy.org/. 参照: 2023.1.28.

[58] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. pp. 3431–3440, 2015.

[59] MSELOSS. Mseloss. https://pytorch.org/docs/stable/generated/torch.nn.MSELoss.html. 参照: 2023.1.28.

[60] Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin Riedmiller. Striving for simplicity: The all convolutional net. *arXiv preprint arXiv:1412.6806*, 2014.

# 発表業績

## 口頭発表

- 梁兆楼, 阿部雅樹, 渡辺大地, 深層学習を用いた 2D アニメの水エフェクト自動生成, 令和 3 年度 第 2 回芸術科学会東北支部研究会発表会, 2022.

## ポスター発表

- 梁兆楼, 阿部雅樹, 渡辺大地, 深層学習を用いた 2D アニメの水エフェクト自動生成, 映像表現・芸術科学フォーラム, 2022.