筑波大学大学院博士課程

システム情報工学研究科修士論文

振動触覚提示のための GANによる時系列データ生成

我妻 正太郎

修士（工学）

（コンピュータサイエンス専攻）

指導教員　高橋 伸

2020年3月

概要

触覚ディスプレイへの応用のため，現実の物体を触った際の様々なデータを収集が広く行われている．しかし，物体を触るという行為には様々な条件が関係しており，これらの条件を考慮して網羅的にデータ収集を行うことは非現実的である．この問題を解決するため，我々は深層学習の一手法である Generative Adversarial Network (GAN) を用いて，収集できない条件のデータの代替となるデータを生成する手法を提案する．本稿では，提案手法実現の第一歩として，現実の物体を触った際の加速度データを元にデータ生成を行う GAN を提案した．この GAN は先行研究のものよりシンプルな構成であり，先行研究で行われていなかった 3 軸加速度データを元にしたデータ生成が可能である．提案するこの GAN を用いて 9 種類のテクスチャから収集した加速度データを用いてデータを生成した．生成したデータと訓練データをスペクトログラム化することにより比較したところ，この GAN により訓練データの特徴を捉えたデータを生成できることがわかった．生成したデータをもとに触覚提示実験を行ったところ，訓練データを用いる場合と変わらないリアリティを実現できることがわかった．また，GAN に入力するラベルデータを操作することで，２つのテクスチャの特徴を合成したデータを生成する実験およびラベル割合変化による影響の検証を行った．

目 次

第 1 章　序論 **1**
1.1　研究背景 1
1.2　研究目的とアプローチ 1
1.3　本研究の貢献 2
1.4　本論文の構成 3

第 2 章　関連研究 **4**
2.1　触覚ディスプレイについての研究 4
2.2　触察時のデータ収集に関連する研究 4
2.3　GAN について 5
2.4　GAN の応用研究 7

第 3 章　GAN によるデータ生成実験 **10**
3.1　用いる GAN のアルゴリズムについて 10
3.1.1　C-RNN-GAN について 10
3.1.2　Deep Convolutional GAN について 11
3.1.3　WaveGAN について 11
3.2　C-RNN-GAN を用いた生成モデル 13
3.2.1　モデルの構成について 13
3.2.2　データ生成実験 17
訓練データと学習の設定 17
データ生成と結果の考察 19
3.3　DCGAN を用いた生成モデルについて 20
3.3.1　モデルの構成について 20
3.3.2　単純な時系列データを用いたデータ生成実験 24
3.3.3　収集データを用いたデータ生成実験 26
3.3.4　生成データを用いた触覚提示実験 30
3.4　WaveGAN を用いた生成モデルについて 32
3.4.1　モデルの構成について 32
3.4.2　収集データを用いた 1 クラスデータ生成実験 35
3.4.3　データセットを用いた多クラスデータ生成実験 37
3.4.4　収集データを用いた多クラスデータ生成実験 38

第 4 章　触覚提示実験 **42**
4.1 実験について 42
4.2 実験手順 42
4.3 結果と考察 43

第 5 章　ラベル合成による未知データの生成 **47**

第 6 章　結論 **52**

謝辞 **53**

参考文献 **54**

著者論文リスト **59**

付 録 A　WaveGAN をベースにしたモデルによる生成データのスペクトログラム **62**

付 録 B　第 5 章での触覚提示実験時用いたアンケート用紙 **67**

図目次

1.1 データ生成手法の想定図 2

3.1 Hyperbolic tangent (tanh) 関数のグラフ 15
3.2 シグモイド関数のグラフ 16
3.3 C-RNN-GAN を用いたモデルによるデータ生成テストのための訓練データ収集の様子と用いたテクスチャ 18
3.4 訓練データ例と C-RNN-GAN を用いたモデルによる生成データ例の比較 . . . 19
3.5 ReLU 関数のグラフ 23
3.6 Leaky ReLU 関数のグラフ 25
3.7 シンプルな時系列データをもとに DCGAN により生成された時系列データ (1) 27
3.8 シンプルな時系列データをもとに DCGAN により生成された時系列データ (2) 28
3.9 触察時の 3 軸加速度データをもとに DCGAN により生成された時系列データ 29
3.10 DCGAN を用いたモデルの損失関数の値 30
3.11 触覚提示テストのために用いたテクスチャと生成データ例 31
3.12 触覚提示テストの結果 32
3.13 WaveGAN ベースの GAN による多クラス生成の概要図 33
3.14 短い収集データをもとに WaveGAN を用いて生成された時系列データ 36
3.15 WaveGAN をもとにしたモデルの学習のため LMT haptic texture database より抜粋したテクスチャ 37
3.16 LMT haptic texture database をもとにした生成データのスペクトログラム例 . . 39
3.17 収集データの先頭と末尾の連続 1000 点を切り取る処理の概要図 39
3.18 収集データを 10 回繰り返したデータを作成する処理の概要図 40
3.19 データ収集に用いた 9 種類のテクスチャ 40
3.20 9 クラスのデータを用いた生成テストのためのデータ収集の様子 41
3.21 収集データをもとにした生成データのスペクトログラム例 41

4.1 触覚提示実験の様子 43
4.2 触覚ディスプレイ上の画面と評価用インタフェース 44
4.3 触覚提示実験におけるテクスチャごとの生成データと訓練データの区別タスクの正答率 45
4.4 触覚提示実験におけるテクスチャごとのリアリティ評価値 46

5.1 ラベル合成によるデータ生成の概要図 47
5.2 Tile ラベルと Place Mat 03 ラベルを合成したラベルによるデータ生成結果のスペクトログラム 48
5.3 Tile ラベルと他のテクスチャのラベルを合成したラベルによるデータ生成結果のスペクトログラム (1) 49
5.4 Tile ラベルと他のテクスチャのラベルを合成したラベルによるデータ生成結果のスペクトログラム (2) 50
5.5 Tile ラベルと他のテクスチャのラベルを合成したラベルによるデータ生成結果のスペクトログラム (3) 51

A.1 LMT Haptic Texture database をもとにした生成データのスペクトログラム (1) 63
A.2 LMT Haptic Texture database をもとにした生成データのスペクトログラム (2) 64
A.3 収集データをもとにした生成データのスペクトログラム (1) 65
A.4 収集データをもとにした生成データのスペクトログラム (2) 66

B.1 触覚提示実験時に用いたアンケート用紙 68

表 目 次

3.1 C-RNN-GAN をベースにしたデータ生成モデルの構成 14
3.2 DCGAN をベースにしたデータ生成モデルの generator の構成 21
3.3 DCGAN をベースにしたデータ生成モデルの discriminator の構成 22
3.4 WaveGAN をベースにしたデータ生成モデルの構成 34
3.5 WaveGAN をベースにしたモデルのハイパパラメータ 35

第1章　序論

1.1　研究背景

現在，様々な触覚ディスプレイが開発され，仮想物体に触れることができる様々なコンテンツが実用化されている．ここでいう触覚ディスプレイとは，触覚刺激を人に提示する機器全般を指すものとする．これらのコンテンツにおいては，仮想物体に触れた際の触感にどれだけリアリティがあるかがユーザのコンテンツへの評価を左右する．しかし，現段階の技術では，現実の物体に迫るような総合的なリアリティある物体の触感の再現には至っていない．特に，物体表面の質感をリアルに再現することが大きな課題となっている．これの解決が難しい理由の一つとして，物体を触察する際に関連するパラメータが非常に多いことが考えられる．触察とは，手などを用いて物体を触ることで物体の質感などの情報を得ることである．物体に触る際には様々な条件が関係し，触感が変化する．そもそも触れるという行為とそこで起きる現象は双方向性のものであり，触れるものと触られるものの双方の表面形状等の物理的な性質やそれぞれの動作速度，力のかけ方等が異なれば全く異なる触感が得られる．リアリティある触覚提示を実現するためには，これら多数の条件について，振動や物体形状等，触察に影響を与えうるデータを収集し，分析した上で仮想物体の触感に反映させる必要がある．このようなデータを収集し，物体の性質に迫った研究はこれまでにいくつか行われている [1] [2]．しかし，条件の量は膨大であり，これらの研究では収集しきれない条件が存在している．例えば Strese ら [2] はペン型機器で様々な条件のデータを収集したが，ペンを物体に押し当てる角度など決められた条件下のデータのため，それ以外の条件では取れるデータが変化する可能性が高い．このように，実際の物体からデータを網羅的に収集するのは考慮するべき条件が膨大なため非現実的である．収集しきれない条件のデータが存在するということは，その条件下の物体の正確な触感の再現はできないということにもなる．

1.2　研究目的とアプローチ

本研究では，物体から直接データを収集する手法に代わる新たな手法を提案，実現することを目的とする．その成果により，仮想物体へのリアリティある触感付与の将来的な実現に貢献する. 目的実現のためのアプローチとして，網羅的に直接データ収集を行う手法の代わりに，機械学習を用いて収集済みデータから未収集データの代替となるデータを生成する手法を提案する．これが実現すれば，様々な条件下におけるデータの解析が可能になるほか，データの収集コストを抑えることができる．また，代替データを触覚ディスプレイの出力に応用し

た場合，より豊かな触感の再現が可能になる．このデータ生成手法の想定図を図 1.1 に示す．

本論文では，機械学習によるデータ生成の実現の第一歩として，触察時の 3 軸加速度データに着目した機械学習によるデータ生成モデルを開発した．触察時の加速度データは主に振動触覚ディスプレイの出力信号の生成に使用される [3] [4] [5]．また，加速度センサという既存のセンサ技術との整合性が高いためデータを容易に収集できる．これらの観点から，触覚ディスプレイへの直接の応用が可能であり，データ収集も容易である加速度データを用いるのは，提案手法実現の第一歩としては適切であると考えている．

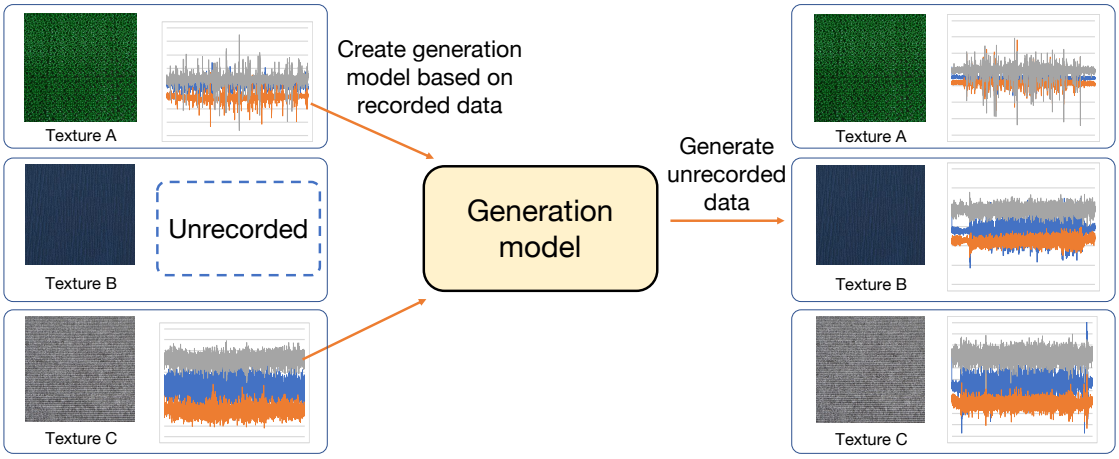

図 1.1: データ生成手法の想定図．収集済みのデータから未収集のデータの代替となるデータを生成する

データの具体的な生成手法としては，Generative Adversarial Network (GAN) [6] を用いる．GAN は主に画像生成を行うための機械学習手法であるが，画像の高解像度化や画像の性質合成さらには音声合成など様々な分野に応用が可能である．実際に音声合成の分野では先行研究により，人間の発音した場合とほぼ変わらない音声の合成に成功しており [7]，GAN を用いた質の高い時系列データ生成が可能であることが示されている．本論文では，音声合成のための GAN である WaveGAN [7] をベースに振動データ生成用機械学習モデルを構築した．このモデルでは，あらかじめ用意したテクスチャを触察した際に記録した 3 軸加速度データを訓練データとし，9 種類のテクスチャを模した 3 軸時系列データを生成できる．生成したデータは触覚ディスプレイの出力信号に応用することが可能である．また，このモデルでは，モデルへの入力データを操作することで，複数の訓練データの特徴を合成したデータを生成することができる．

1.3 本研究の貢献

本研究の貢献は，実際のテクスチャを触察した際の加速度データをベースとした 3 軸時系列データを生成するモデルを作成したことである．このモデルには音声生成用の GAN をベー

スとしたシンプルなネットワーク構成を採用し，先行研究において行われていなかった 3 軸の時系列データの生成が可能である．この生成データを用いることで，訓練データとほぼ同等のリアリティを持つ触覚提示が可能であることを検証により明らかにした．3 軸の時系列データを生成できたことで，単一の振動信号を用いる触覚ディスプレイのための信号としての利用にとどまらず，より多くの情報を利用する触覚ディスプレイや，触覚信号そのものの認識や解析に利用できるデータ生成が可能になった．

1.4 本論文の構成

次章以降の本論文の構成について以下に示す．本章では，本研究の背景および目的とそれに対するアプローチについて述べた．続く第 2 章では，本研究に関連する先行研究や事例を述べる．第 3 章ではデータ生成に用いる機械学習モデルについて詳説し，実際にデータ生成を行った結果について述べる．第 4 章では生成されたデータを用いて触覚提示実験を行い，その結果を考察する．第 5 章では，モデルへの入力データを操作することにより，複数の訓練データの特徴を合成したデータの生成した結果について考察する．最後に第 6 章では，結論を述べる．

第2章　関連研究

2.1　触覚ディスプレイについての研究

物体の質感やテクスチャ感を再現するため，様々な触覚ディスプレイが提案されている．Bau ら [8] は静電気力を用いた触覚ディスプレイを開発した．この手法は，電圧が印加された電極上に絶縁膜を貼り，絶縁膜上に指を置くと誘電分極により指に吸引力が働くことを利用している．Vezzoli ら [9] は超音波によるスクイーズ膜効果を利用した触覚ディスプレイを提案し，その設計について考察した．Iwamoto ら [10] は超音波により空気を振動させ，空中に触覚提示を行う手法を提案し，実際に触覚提示実験を行った．Ishihara ら [11] は粘性が温度によって変化する特別なポリマーシート用いて，画像に粘着質の触感を付与する触覚ディスプレイを開発した．

このように，様々な手法による触覚ディスプレイが提案されているが，特に振動子による機械振動を用いた触覚ディスプレイについての研究が盛んに研究されている．Culbertson ら [5] はペンに振動子を取り付け，アクチュエータを振動させることでペンで物体をなぞった時の触感を再現した．Cho ら [4] はペンに２つの振動子を取り付けた触覚ディスプレイを用いて，タブレット上に文字を書く際に既存のペンの書き味を再現するアプリケーションを開発した．Minamizawa ら [12] は振動子を用いて簡単に制作が可能かつ低コストな触覚ディスプレイを開発し，様々な応用例を示した．Saga ら [3] は指を置くための台座を糸で巻き取ることで振動触覚提示を行うタブレット装着型の触覚ディスプレイを開発した．これらの手法ではあらかじめ収集された加速度データを触覚提示信号生成に利用している．このことから，加速度データをもとにした時系列データ生成を行うことで，生成データを多くの振動触覚ディスプレイでの触覚提示に直接応用できると考えられるため，研究目的の実現の第一歩としては，加速度データをもとにしたデータ生成が適切であると考えている．

2.2　触察時のデータ収集に関連する研究

触覚ディスプレイ上においての物体の質感を再現するため，触察時のデータを収集する研究が広く行われている．Abdulali ら [1] は，加速度センサと圧力センサとを用いた専用の機器を PHANToM[1] に装着し，動作させることで 9 種類の素材の表面の情報を収集し，収集データを解析した結果から，異方性素材のモデリング手法を提案した．PHANToM[1] は仮想物体に触

[1] Phantom premium (https://ja.3dsystems.com/haptics-devices/3d-systems- phantom-premium (最終閲覧日 2020 年 1 月 25 日))

れて操作することを目的とした触覚ディスプレイの 1 つである．異方性素材とは，素材の縦方向，横方向，高さ方向のそれぞれで性質が異なる素材のことである．Strese ら [2] は加速度センサ，圧力センサ，マイク，ウェブカメラ，金属探知機，赤外線センサの 6 つのセンサを複合した装置により 108 種の物体表面のデータを収集した．Burka ら [13] は自律型ロボットへの応用を目的とし，物体表面の視覚情報と触察時の情報を収集するデバイスを提案した．Jamali ら [14] は，人工指に圧電フィルムを内蔵した機器を用いて物体表面のデータ収集を行う手法を提案し，実際に 7 種類の物体表面の情報を収集した．嵯峨ら [15] は，市販の小型の無線型加速度センサ付きマイコンを指などに取り付け，センサを取り付けた指やペンで物体をなぞった際の加速度情報を収集する手法を提案した．

このような様々な手法により，触察時のデータの収集が行われてきた．しかしどの研究においても限定的な収集条件下でのデータ収集を行っており，考慮されていない収集条件が存在している．物体を触察する際に関連するパラメータは非常に多く，それら全てを網羅的に収集するのは非現実的である．実際の物体からデータを収集する代わりに，機械学習によって収集済みデータを元に代替データを生成する．この手法が実現すればデータの収集コストが削減できる．また，実際に網羅的なデータ収集を行わずとも，機械学習手法を調整することで収集していない条件のデータを合成することができる．

2.3 GAN について

機械学習による未収集データの生成の第一歩として，GAN [6] を用いて 3 軸加速度データを用いたデータ生成を行う．GAN は Goodfellow らが提案したニューラルネットワークを用いたデータ生成用機械学習モデルである．元々の GAN は訓練データに一致するような生成データを得ることを目的とした機械学習手法であるが，ネットワーク構成次第では訓練データとは異なるデータを生成することも可能である [16] [17]．ここでは，GAN のアルゴリズムについて概説する．

Goodfellow らが提案した GAN は訓練データに一致するような生成データを得ることを目的とした機械学習手法である．目的実現のため，GAN では，データ生成を行う generator と生成データと訓練データを分類する discriminator の 2 種類の機械学習モデルを用いる．generator によってデータを生成した後，discriminator が生成データと訓練データを分類することで，2 つのデータを正確に分類できるよう学習が進む．discriminator の分類結果を受けて，generator は discriminator に分類されないように，訓練データと類似したデータを生成できるように学習を進める．これらの generator と discriminator の学習を交互に繰り返すことで，最終的には generator が訓練データと非常によく似たデータを生成できるようになる．

discriminator と generator を最適化する目的関数を式 2.1 に示す．V は目的関数の値を示す．D は discriminator，G は generator，z は乱数，$data$ は訓練データを表す．また，$D(x)$ は discriminator の出力，$G(x)$ は generetor の出力である．p_{data} は訓練データの分布，p_z は乱数の分布である．$\mathbb{E}$ は期待値を表す．

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim p_{data}(x)}[\log D(x)] - \mathbb{E}_{z\sim p_z(z)}[\log(1-D(G(z)))] \tag{2.1}$$

generator は目的関数の最小化を行うことで，discriminator が訓練データと区別できないようなデータを生成できるよう学習を進める．discriminator は目的関数の最大化を行うことで，訓練データと生成データを高精度で分類できるように学習を進める．GAN の学習時には，generator を学習せず固定し，discriminator を最適化するステップと，discriminator を固定し generator を最適化するステップを交互に行う．また，Goodfellow らによれば式 2.1 において，generator を固定したした場合の最適な discriminator は式 2.2 により表される．この式において，p_{data} は訓練データの分布，p_g は生成データの分布である．

$$D^*_G(x) = \frac{p_{data}(x)}{p_{data}(x) + p_g(x)} \tag{2.2}$$

この式を式 2.1 に代入することで，式 2.3 を得ることができる．C は最適化された D を用いた場合の V を示す．

$$\begin{aligned} C(G) &= \max_D V(G,D) \\ &= \mathbb{E}_{x\sim p_{data}(x)}[\log D^*_G(x)] - \mathbb{E}_{z\sim p_z}[\log(1-D^*_G(G(z)))] \\ &= \mathbb{E}_{x\sim p_{data}(x)}[\log D^*_G(x)] - \mathbb{E}_{z\sim p_g}[\log(1-D^*_G(x)] \\ &= \mathbb{E}_{x\sim p_{data}(x)}\left[\log \frac{p_{data}(x)}{p_{data}(x)+p_g(x)}\right] - \mathbb{E}_{z\sim p_g}\left[\log \frac{p_g(x)}{p_{data}(x)+p_g(x)}\right] \end{aligned} \tag{2.3}$$

ここで，discriminator が最適の場合，$p_g = p_{data}, D^*_G(x) = \frac{1}{2}$ であるため，式 2.3 に代入すると $C(G) = \log\frac{1}{2} + \log\frac{1}{2} = -\log 4$ となる．このことに加え，2 つの確率分布の差異を比較する尺度である Kullback-Leibler Divergence を用いると式 2.3 を変形して式 2.4 が得られる． 式 2.4 中の KL は Kullback-Leibler Divergence を示す．

$$C(G) = -\log(4) + KL\left(p_{data} \parallel \frac{p_{data}+p_g}{2}\right) + KL\left(p_g \parallel \frac{p_{data}+p_g}{2}\right) \tag{2.4}$$

Kullback-Leibler divergence を改良した尺度である Jensen-Shannon Divergence を用いて式 2.4 を変形し，式 2.5 が得られる．ここでの JSD は Jensen-Shannon Divergence を示す．

$$C(G) = -\log(4) + 2 \cdot JSD(p_{data} \parallel p_g) \tag{2.5}$$

この式から， generator は最適な discriminator のもとでは Jensen-Shannon divergence を用いて $p_{data}(x)$ と $p_g(x)$ を近づけるように学習しようとしていると言える．また，Goodfellow ら

は，generator と discriminator に十分な表現力があり，訓練データ量が十分多いと仮定した場合，GAN の学習によって $p_g(x)$ が $p_{data}(x)$ に収束することを証明した．このことから，GAN の学習を進めることで， generator が訓練データに近いデータを生成できるようになることがわかる．

GAN はニューラルネットワークを用いた高性能なデータ生成手法として広く用いられているが，機械学習を用いたデータ生成手法は他にも存在している．GAN の他に広く用いられている手法としては，Variational Autoencoder (VAE) [18] が挙げられる．VAE では潜在変数からデータを生成する decoder とその逆向きの推論を行う encoder から構成される機械学習モデルである．VAE の長所としてはデータ生成時のサンプリングコストが低い点，生成結果をコントロールし易い点が挙げられる．VAE の短所としては，理想的な学習が行われた場合でも $p_g(x)$ と $p_{data}(x)$ が一致するとは限らない点が挙げられる．GAN では理想的な学習が行われた場合には $p_g(x)$ が $p_{data}(x)$ に収束することが示されているため，理論上では VAE より高品質なデータ生成が可能である．さらに，GAN は VAE と同様にデータ生成時のサンプリングコストが低く，生成結果をコントロールし易い．

GAN による生成結果をコントロールするための手法としては Mirza らが提案した Conditional GAN [19] が用いられる．Conditional GAN では，訓練データに条件ベクトルを付与して学習させる．これにより，データ生成時に条件ベクトルに対応したデータを指定して生成することが可能になる．Conditional GAN の目的関数を式 2.6 に示す．式 2.6 中の y は条件ベクトルである．GAN の目的関数の discriminator と generator の入力に条件ベクトル y を加えたものが Conditional GAN の目的関数となっている．Mirza らは論文中，手書き数字画像データセットである MNIST [20] を訓練データとしたデータ生成において，Conditional GAN を用いることで任意の数字に対応した画像を生成できることを示した．Conditional GAN は GAN によってデータ生成を行う場合の一般的手法として知られており，既存の様々な研究が Conditional GAN を応用して生成データの制御を行っている [21] [22] [17].

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim p_{data}(x)}[\log D(x|y)] - \mathbb{E}_{z\sim p_z(z)}[\log(1 - D(G(z|y)))] \tag{2.6}$$

VAE と比較した際の GAN の短所としては，Min-Max 最適化を行うため，学習が安定しない点が挙げられる．しかし，この短所については目的関数の改良を行う研究 [23] [24] [25] や，GAN のネットワーク構造の改良を行う研究 [26] [27] によって，様々な対策が講じられており，それらを用いることで安定した学習が可能であると考えられる．これらの観点から，我々は学習が安定すると考えられ，かつ高品質なデータ生成が可能である GAN をデータ生成モデルとして使用することにした．

2.4 GAN の応用研究

GAN は主に画像生成に用いられているが，GAN は単純な画像生成のみならず様々な分野に応用が可能である．以下に GAN の応用例を示す．

- 画像生成
  - 高品質な画像生成 ... Deep Convolutional GAN [26]
  - 画像の高解像度化 ... SRGAN [28]
  - 画像のスタイル変換 ... CycleGAN [16]
- 音声生成
  - MIDI 音声生成 ... C-RNN-GAN [29]
  - 人の音声の生成 ... WaveGAN [7]
- 触察データの生成
  - ペン型触覚ディスプレイ用 1 軸時系列データ生成 ... Ujitoko らの GAN [30]

画像生成に GAN を応用した例について示す．Radford ら [26] は Goodfellow らの GAN より高品質な画像を生成する Deep Convolutional GAN (DCGAN) を提案した．DCGAN では generator と discriminator に Convolutional Neural Network (CNN) [31] を用いることで，より正確に画像の特徴を学習できるネットワーク構成を実現している．Ledig ら [28] は画像の高解像度化を行う GAN である，SRGAN を提案した．Ledig らは SRGAN を既存の高解像度手法と比較し，その結果から GAN のアルゴリズムが画像の高解像度化に有用であることを示した．Zhu ら [16] は画像のスタイル変換を行う GAN である，CycleGAN を提案した．ここでの画像のスタイル変換とは，画像の色などの外見的特徴を変化させることである．Zhu らは GAN を用いることで，教師なし学習のもとで画像のスタイル変換を行うことに成功した．また，Zhu らは CycleGAN の応用例として，絵の画風変換や写真へのぼかしの追加などが可能であることを示した．

また，GAN は音声合成の分野でも用いられ始めており，高品質な音声合成に成功している研究がいくつか存在する．Mogren ら [29] は MIDI 音声データを生成することができる GAN である，C-RNN-GAN を提案した．C-RNN-GAN では，時系列データの処理に適するネットワークである Long Short Term Memory (LSTM) [32] を generator と discriminator に採用することで，高品質な MIDI 音声データ生成を実現している．Donahue ら [7] は人の音声と遜色ない音声データを生成できる GAN である，WaveGAN を提案した．WaveGAN は DCGAN をベースにしており，訓練データとなる音声データの時間方向にのみ畳み込み演算を行うことで音声データの時系列特徴を効果的に学習することができる．

このように GAN は様々な分野に応用が可能であるが，GAN を用いて触察に関連するデータを処理する研究については研究の数が非常に少ない．直接関連する研究としては，Ujitoko らの研究 [30] が挙げられる．Ujitoko らはテクスチャ画像からテクスチャ画像に対応した振動データを生成する GAN モデルを提案した．このモデルは，Encoder ネットワークと Generator ネットワークの２つからなるモデルである．Encoder ネットワークでテクスチャ画像をラベルデータに変換する．Generator ネットワークでは，ラベルデータとあらかじめ収集された加速

度データにより訓練されたGANを用いてデータ生成を行う．この手法により，Ujitokoらは9クラスのテクスチャについてデータ生成を行った．

UjitokoらはGANを用いて高品質なデータ生成を行なったが，生成データは1軸時系列データであるため，ペン型触覚ディスプレイ等の限られた触覚ディスプレイにしか使用することができない．また，機械学習に用いているニューラルネットワークが非常に大きいため，学習のために非常に大きな計算コストが必要と考えられる．これらの点から，まだデータ生成用モデルについては改良の余地が存在する．我々はデータ生成用モデルのこれらの問題点を解決できる生成モデルを作成することを目指す．

我々はモデルの学習のための訓練データとして，さまざまな処理に活用可能なデータとなるように，1軸に限定せず，3軸加速度データを用いる．シンプルなネットワーク構成で，かつ3軸加速度データを訓練データとした有効なデータ生成を実現するため，前述した様々なGANの応用手法の中でも，我々は音声合成のためのGANに着目した．音声データは加速度データと同じく時系列データであり，これらのGANを用いることで，効果的なデータ生成が可能であると考えられる．我々は先行研究の中でもC-RNN-GAN [29]とWaveGAN [7]に着目した．このGANでは，シンプルなGANモデルを用いて高品質な音声合成に成功しており，我々が必要としている条件と合致している．また，画像生成で成果を残しているDCGAN [26]についても着目した．DCGANは時系列データの処理も可能なCNNを採用しており，3軸加速度データを効果的に学習できる可能性がある．我々はこれらのGANをベースにデータ生成用モデルを構築することにした．

第3章　GANによるデータ生成実験

ここではGANによるデータ生成モデルの構成と，データ生成実験について述べる．我々は音声処理のためのGANに着目し，これを触覚情報処理に適用する．どのようなモデルが時系列データの生成に適しているか調査するため，音声処理のためのGANの代表的な手法である以下の2種類の手法を用いた．すなわちC-RNN-GANベース [29] のモデル，WaveGAN [7] ベースのモデルである．また，CNNを用いていることで時系列データを効果的に処理できる可能性がある，Deep Convolutional GAN (DCGAN) [26] ベースのモデルも作成した．いずれの手法もモデル構成としてはシンプルなものを用いている．これら3つのモデルを用いて，各モデルを用いて実際にデータ生成を行った．

本研究での機械学習は，プログラム言語 Python と Google 社が提供する機械学習ライブラリ Tensorflow[2]を使用し実装した．さらに，2つの GPU (NVIDIA GTX1080 Ti) と Tensorflow[1] を連携させることで，計算の高速化を図った．

3.1　用いるGANのアルゴリズムについて

シンプルなネットワーク構成で，かつ3軸加速度データを訓練データとした有効なデータ生成を実現するため，我々は，C-RNN-GAN [29]，DCGAN [26]，WaveGAN [7] に着目した．

3.1.1　C-RNN-GANについて

C-RNN-GAN [29] について概説する．C-RNN-GANはGANのgeneratorにLong Short Term Memory (LSTM) [32]，discriminatorにBidirectional LSTM (BLSTM) [33] を用いたデータ生成モデルである．LSTMはRecurrent Neural Network (RNN) [34] を改良したネットワークである．RNNは主に時系列データの処理に用いられるニューラルネットワークを用いた機械学習手法の一つである．RNNでは，前時刻の中間層のデータを現時刻での入力データとしても用いることで，訓練データの時系列的特徴を保持しながら学習を行うことができる．この性質から，RNNは時系列データの処理に有効とされている．しかし，誤差逆伝播による学習を行う際に勾配が消失しやすく，学習が安定しないという問題がある．LSTMはこの問題を解決するために提案された手法の一つである．LSTMではRNNにConstant Error Carouselと呼ばれる記憶素子を導入することで勾配の消失を防ぎ，学習を安定させている．Bidirectional LSTMはLSTMを改良した手法である．LSTMでは時刻 $t-1$ の状態を時刻 t の状態の入力として用

[2]TensorFlow (https://www.tensorflow.org/ (最終閲覧日 2020 年 1 月 25 日))

いるが，Bidirectional LSTM では時刻 $t+1$ の状態を入力として用いる逆方向の LSTM も同時に使用する．BLSTM は処理するデータの時系列情報が全て判明している状態でないと使用できないという制約があるが，タスクによっては LSTM よりも高い性能を示すことが知られている [35]．C-RNN-GAN では LSTM 層を多数組み合わせることで，時系列データの処理能力を向上させている．C-RNN-GAN の提案者である Morgen らはこのモデルを用いて，高品質な MIDI 音声データの生成に成功している．時系列データの処理に適した RNN をもとにした C-RNN-GAN を用いることで時系列データを有効に生成できる可能性がある．

3.1.2 Deep Convolutional GAN について

次に Deep Convolutional GAN [26] について概説する．DCGAN は GAN の generator と discriminator に Convolutional Neural Network (CNN) [31] を用いることで高品質な画像生成を可能にしたモデルである．CNN はニューラルネットワークを用いた機械学習手法の一つで，主に画像処理に用いられる．通常のニューラルネットワークと比べて，畳み込み層とプーリング層が追加されている．畳み込み層では，あらかじめ設定した畳み込みフィルタをもとに入力データに畳み込み演算を行うことで，入力データの特徴を抽出する．畳み込み演算を行うことでデータの空間的特徴を維持したまま，次元削減による特徴抽出をすることが可能である．これにより，データの形状や特徴を効率的に学習することが可能である．プーリング層ではデータの特徴を保持したままデータを圧縮し，扱いやすい形に変形する．これにより，データの小さな特徴に頑健となる他，計算コストを抑えることができる．CNN は主に画像処理に用いられる手法であるが，時間方向に畳み込み演算を行うことで時間方向のデータの変化の特徴を学習することが可能であるため [36] [37]，時系列データの処理にも適すると考えられる．したがって，CNN により時系列データを有効に処理できる可能性が高く，CNN を用いる DCGAN も時系列を効果的に処理できると考えられる．また，DCGAN は generator と discriminator 共に CNN のみを用いるというシンプルな構成から，画像生成の分野では広く用いられており，DCGAN から派生する GAN も数多く存在する．触覚ディスプレイ用のデータを GAN を用いて生成した Ujitoko ら [30] も DCGAN から派生した GAN を用いてデータ生成を行っている．これらの観点から，DCGAN により高品質な時系列データを生成できる可能性が高い．

3.1.3 WaveGAN について

次に WaveGAN [7] について説明する．WaveGAN は Donahue らが提案した音声生成用の GAN であり，DCGAN から派生した手法の一つである．基本的な構成は DCGAN と一致しているが，WGAN-GP [25] を導入している点が異なる．WGAN-GP について概説する．WGAN-GP は Wasserstein GAN (WGAN) [24] を改良したものである．WGAN は，通常の GAN と異なる目的関数を用いることで，学習を安定化させた GAN である．通常の GAN では，訓練データと生成データの分布の Jensen-Shannon divergence (JSD) を最小化するように目的関数を設定する．しかし，JSD を用いる場合，パラメータの最適点付近で勾配が消失する場合が多く，学習

が安定しないという問題がある [24]．この問題を解決するため，WGAN では Wasserstein 距離を目的関数に採用することで，学習を安定化させている．この Wasserstein 距離を用いた目的関数は式 3.1 のようになる．WGAN では式 3.1 中の L を最小化するように学習を進める．

$$L = \mathbb{E}_{\tilde{x}\sim\mathbb{P}_g}[D(\tilde{x})] - \mathbb{E}_{x\sim\mathbb{P}_r}[D(x)] \tag{3.1}$$

式 3.1 中の D は discriminator を示し，$\mathbb{P}_r$ は訓練データの分布，$\mathbb{P}_g$ は生成データの分布，$\mathbb{E}$ は期待値，x は訓練データ，$\tilde{x}$ は生成データを示す．この式は D が $K = 1$ のリプシッツ連続である場合にのみ成立する．式 3.2 にリプシッツ連続についての式を示す．写像 f がリプシッツ連続である場合，実定数 K が存在し，任意の x, y に対して式 3.2 を満たす．WGAN では，式 3.1 がリプシッツ連続であるようにするため，学習時に discriminator の重みの値を $[-0.01, 0.01]$ にクリッピングする．具体的には，0.01 を超える値は 0.01 に，-0,01 未満の値は-0.01 に値を修正する処理を行う．WGAN は Wasserstein 距離を用いることで通常の GAN より学習を安定化させたが，重みの値のクリッピングに起因する勾配消失が起こりやすいことが課題となっている．

$$|f(x) - f(y)| \leq K|x - y| \tag{3.2}$$

WGAN-GP はこの課題を解決するために提案された GAN である．最適化された WGAN の discriminator は $\mathbb{P}_r$，$\mathbb{P}_g$ のほぼ全ての点において単位勾配ノルムを持つという性質が証明されており，WGAN-GP ではこれを満たすように目的関数の値を調整する．具体的には，ペナルティ項を WGAN の目的関数に導入することで目的関数の値を調整する．WGAN-GP での目的関数を式 3.3 に示す．式 3.3 中の $\hat{x}$ は生成データと訓練データを結んだ直線上の任意の点である．式 3.3 中の λ はハイパパラメータである．WGAN-GP の提案者である Gulrajani ら [25] が行った実験によれば，WGAN-GP の方が基本的な WGAN より画像生成用の GAN を評価するための一般的な指標である Inception Score [38] の値が高く，学習の収束も早い．WaveGAN においても，WGAN-GP を採用することで学習の安定化を図っている．

$$L = \mathbb{E}_{\tilde{x}\sim\mathbb{P}_g}[D(\tilde{x})] - \mathbb{E}_{x\sim\mathbb{P}_r}[D(x)] + \lambda\, \mathbb{E}_{\hat{x}\sim\mathbb{P}_{\hat{x}}}[(\parallel \nabla_{\hat{x}} D(\hat{x}) \parallel_2 -1)^2] \tag{3.3}$$

Donahue らは WGAN-GP を導入した以外にも，ネットワークの各層の活性化関数の出力の位相をずらすことで時系列データの学習を安定化させた．また，ネットワークにおける畳み込み層の畳み込みフィルタサイズを大きくすることで，データの時系列特徴を学習しやすいネットワーク構造を実現した．これらの工夫により WaveGAN は，元となる DCGAN の構成よりネットワークが大型化しているものの，DCGAN と比べて時系列データの生成に特化した構成となっている．実際に Donahue らは論文中で高品質な音声合成に成功したことを報告しており，WaveGAN を用いることで，高品質な時系列データを生成できる可能性が高い．

上記の 3 種類の GAN を用いて，3 軸加速度データを用いたデータ生成を行い，有効なデータを生成できるか検証した．また，Donahue らは現在主流であるモデルの出入力をスペクト

ログラム画像化する手法について，モデルから生成されたスペクトログラムを時系列データに変換する際にデータの欠落が起こる可能性があるとした．この可能性を考慮し，我々のモデルでは，データをスペクトログラム化せず，生データのまま処理を行うことにした．

3.2 C-RNN-GAN を用いた生成モデル

3.2.1 モデルの構成について

ここでは，C-RNN-GAN を用いたデータ生成モデルについて詳説する．C-RNN-GAN は GAN の generator に Long Short Term Memory (LSTM) [32]，discriminator に Bidirectional LSTM (BLSTM) [33] を用いたデータ生成モデルである．LSTM は Recurrent Neural Network (RNN) [34] を改良したネットワークである．RNN は時刻の中間層のデータを現時刻での入力データとしても用いることで，訓練データの時系列的特徴を保持しながら学習を行うことができるため，時系列データの処理に秀でる．RNN をベースとした C-RNN-GAN を用いることで，有効な時系列データ生成ができる可能性がある．

我々は C-RNN-GAN をベースにデータ生成モデルを作成した．データ生成モデルの構成を表 3.1 に示す．表 3.1 中の “Generator”と “Discriminator”はモデルを構成するニューラルネットワーク層の構成を示す．“Input”は入力層を示す．“Output”は出力層を示す．入力層と出力層の間にある各層は隠れ層を示す．ここ示した隠れ層では，入力層側から出力層側の方向に値が伝播する．例えば，表 3.1 中の “Generator”では，入力層の次には LSTM 層に値が伝播し，その次は Tanh 層に値が伝播する．“Hidden units”は LSTM 層内の隠れ層数，“Output Shape”は各層で出力されるデータの形を示す．

はじめに generator 側について説明する．generator の入力は-1 から 1 の一様分布に基づき生成された乱数ベクトルとした．乱数ベクトルの長さについては，訓練データと同じになるようにした．出力は 1 軸の時系列データである．今回 3 軸の時系列データ生成のため，後述の生成テスト時には各軸ごとに 1 つの生成モデルを作成し，データ生成を行った．

generator は LSTM 層 2 つと全結合層 1 つによって構成されている．LSTM 層内部の隠れ層の数については，C-RNN-GAN の提案者である Morgen らと同じく 350 とした．隠れ層の活性化関数には Hyperbolic tangent (tanh) 関数を用いた．活性化関数を $h(x)$ としたとき，tanh 関数を活性化関数として用いた場合，式 3.4 のようになる．グラフにすると図 3.1 のようになる．tanh 関数はニューラルネットワークの活性化関数として広く用いられているシグモイド関数のデメリットを改善するため提案された関数である．シグモイド関数を活性化関数とした場合，式 3.5 のようになる．グラフにすると図 3.2 のようになる．tanh 関数は比較的単純な非線形関数であることから，学習時の計算負荷を減らすことができる．さらに，出力が-1 から 1 であり，出力が 0 にセンタリングされるため，シグモイド関数に比べて学習の収束が早い．これらの観点から，tanh 関数は活性化関数として広く用いられている．generator の出力層でも LSTM 内部と同じく tanh 関数を用いることで，出力データの値を-1 から 1 の間に統一

表 3.1: C-RNN-GAN をベースにしたデータ生成モデルの構成

Generator	Hidden units	Output Shape (n = batch size)
Input : Uniform(-1,1)		(n, 1000,1)
LSTM	350	(n, 1000,1)
Tanh		(n, 1000,1)
LSTM	350	(n, 1000,1)
Tanh		(n, 1000,1)
Dense		(n, 1000,1)
Output : Tanh		(n, 1000,1)

Discriminator	Hidden units	Output Shape (n = batch size)
Input : Uniform(-1,1)		(n, 1000,1)
Bidirectional LSTM	350	(n, 1000,1)
Tanh		(n, 1000,1)
Bidirectional LSTM	350	(n, 1000,1)
Tanh		(n, 1000,1)
Dense		(n,1)
Output : Sigmoid		(n,1)

した．これは後述の生成テスト時，訓練データとの比較を容易にするためである．

$$h(x) = \frac{\exp(x) - \exp(-x)}{\exp(x) + \exp(-x)} \tag{3.4}$$

$$h(x) = \frac{1}{1 + \exp(-x)} \tag{3.5}$$

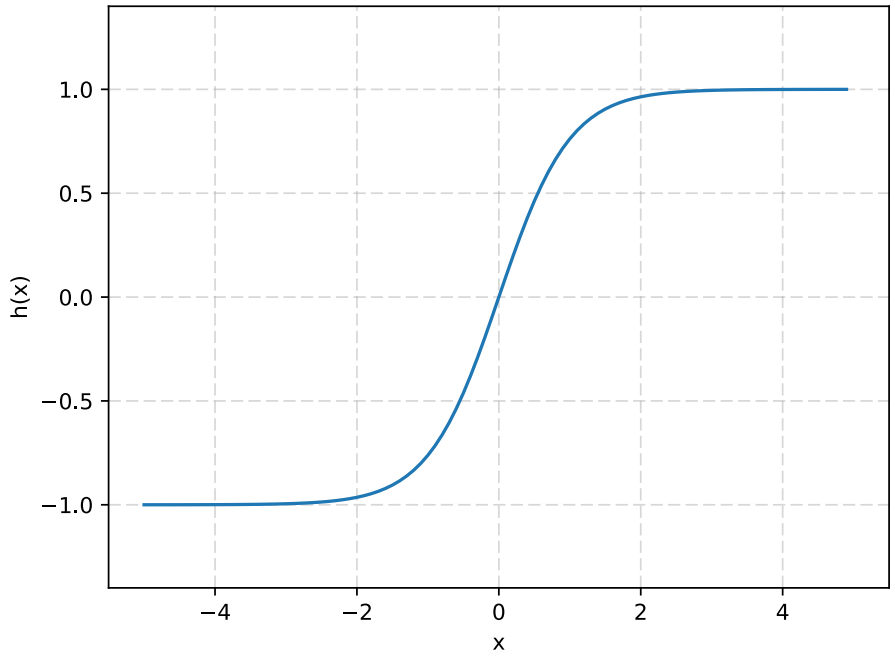

図 3.1: Hyperbolic tangent (tanh) 関数のグラフ

LSTM に限らずニューラルネットワークの学習は，出力データを基に損失関数を用いてニューラルネットワークの性能の悪さを評価し，損失関数の値をより減らすことができるように重みを更新することで行われる．この際，各層の重みパラメータに関する損失関数の勾配を微分により求めることで，どのように重みを更新するかを決定する．このモデルの generator では，損失関数には平均二乗誤差を用いる．y_i を訓練データの値，$\hat{y}_i$ を出力データの値とすると，ここで用いる平均二乗誤差 (MSE) は式 3.6 で表される．

$$MSE(y_i, \hat{y}_i) = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{3.6}$$

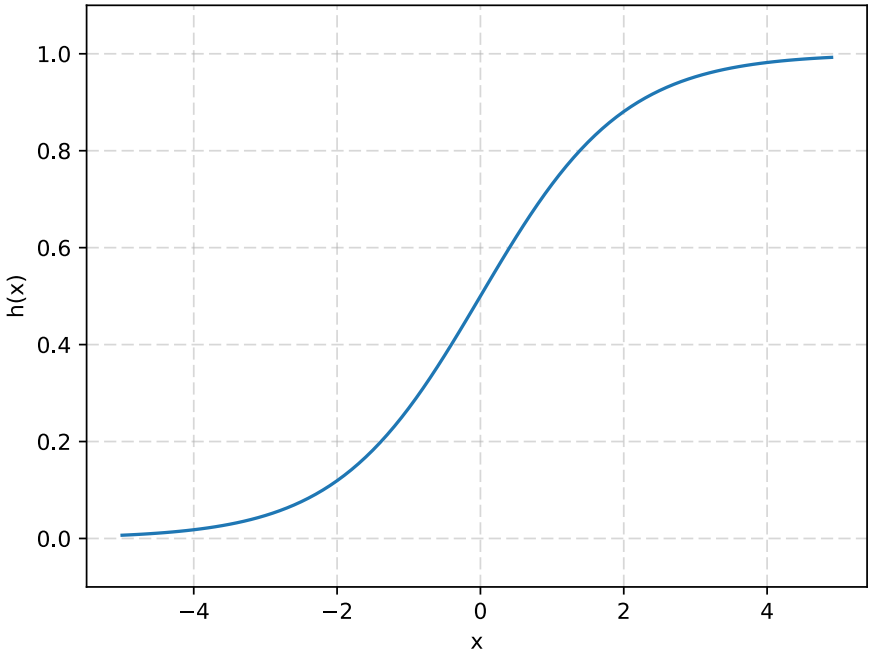

図 3.2: シグモイド関数のグラフ

このモデルの重みの更新手法については Kingma ら [39] が提案したアルゴリズム，Adam を用いる．W を更新する重みパラメータ，損失関数の値を L，$\frac{\delta L}{\delta W}$ を W に関する損失関数の勾配とすると，式は 3.7 のように表される．式中の $\alpha, \beta_1, \beta_2, \epsilon$ はハイパーパラメータである．式中 $\frac{\delta L}{\delta W} \odot \frac{\delta L}{\delta W}$ は $\frac{\delta L}{\delta W}$ と $\frac{\delta L}{\delta W}$ のアダマール積を表す．アダマール積とは行列の要素ごとの積を取ることにより定まる行列の積である． ここでは Kingma ら [39] の論文を参考に，$\alpha = 0.0001, \beta_1 = 0.9, \beta_2 = 0.999, \epsilon = 10^{-8}$ とした．

$$
\begin{aligned}
m_0 &= 0 \\
v_0 &= 0 \\
m_{t+1} &= \beta_1 m_t + (1-\beta_1)\frac{\delta L}{\delta W} \\
v_{t+1} &= \beta_2 v_t + (1-\beta_2)\frac{\delta L}{\delta W} \odot \frac{\delta L}{\delta W} \\
\hat{m} &= \frac{m_t + 1}{1-\beta_1^t} \\
\hat{v} &= \frac{v_t + 1}{1-\beta_2^t} \\
W^{t+1} &= W^t - \alpha \frac{\hat{m}}{\sqrt{\hat{v}} + \epsilon}
\end{aligned}
\tag{3.7}
$$

次に discriminator 側について示す．discriminator 側については，入力は訓練データまたは generator による生成データとした．出力は入力データが生成データである確率とした．discriminator 側は BLSTM 層 2 つと全結合層 1 つによって構成されている．LSTM 層内部の隠れ層の数と活性化関数については generator と同じ設定を用いた．出力層での活性化関数にはシグモイド関数を用いた．これにより 0 から 1 までの値に出力データを調整することで，出力を入力データが生成データである確率として扱えるようにした．重みの最適化と損失関数については generator と同じ設定を使用した．

3.2.2 データ生成実験

C-RNN-GAN をベースに構築した GAN モデルを用いて，時系列データの生成実験を行った．我々はモデルの学習のための訓練データとして，さまざまな処理に活用可能なデータとなるように，1 軸に限定せず，3 軸加速度データを用いる．加速度センサを装着した指で板状の物体 (テクスチャ) をなぞり，その際の 3 軸加速度データを収集した．収集したデータを訓練データとし，GAN によって 3 軸時系列データを生成をした．

訓練データと学習の設定

初めに訓練データの収集手法について示す．データ収集の様子を図 3.3 に示す．右手人差し指の爪側に 3 軸加速度センサ (ADXL-335) をテープで固定し，その指腹で実際のテクスチャ

をなぞった際に発生する加速度データを収集する．収集者は一人 (20 代，男性) である．テクスチャを左から右に 5 秒間なぞる試行を 90 回行い，データを収集した．なぞる速さについては約 5 cm/s で一定になるようにメトロノームを用いて調整した．テクスチャをなぞる指の角度はテクスチャ表面から約 45 度とした．データ収集の際，3 次元のデータを正確に取得するため，加速度センサを接続した Arduino からシリアル通信を利用して加速度データを PC に送信し，PC 側では Processing により送信された加速度データを収集した．加速度は通信速度の関係上 1 kHz でサンプリングされた．データ収集に用いたテクスチャは図 3.3 上部に示すような人工芝のテクスチャである．この生成テストでは試験的に 1 つのテクスチャのみでのデータ収集と生成を行った．

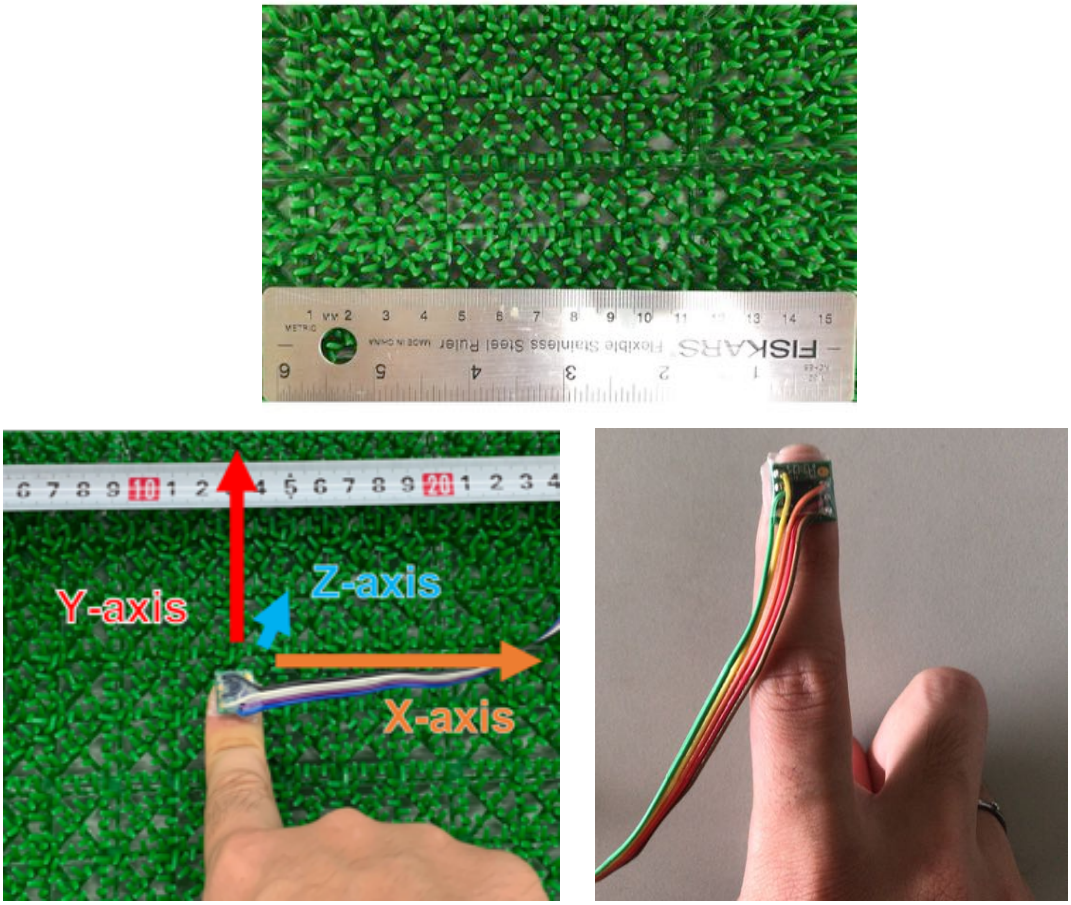

図 3.3: C-RNN-GAN を用いたモデルによるデータ生成テストのための訓練データ収集の様子と用いたテクスチャ

次に学習時の細かい設定について示す．全収集データの中から連続 1000 点のデータをランダムに 50 個抜き出して学習させる試行を 100 回繰り返すことで生成モデルを作成した．学習時間は各軸ごとに約 2 時間であり，合計で約 6 時間であった．また，学習する前に訓練デー

タを-1 から 1 の間にあらかじめ正規化した．

データ生成と結果の考察

訓練データと生成データの軸ごとの波形を図 3.4 に示す．グラフの縦軸は-1 から 1 に正規化された訓練データもしくは生成データの値である．グラフの横軸はデータの各点を時系列順に並べた際のインデックスとなっている．図 3.4 中の訓練データは，全データからランダムに抜き出した連続 1000 点の加速度データである．

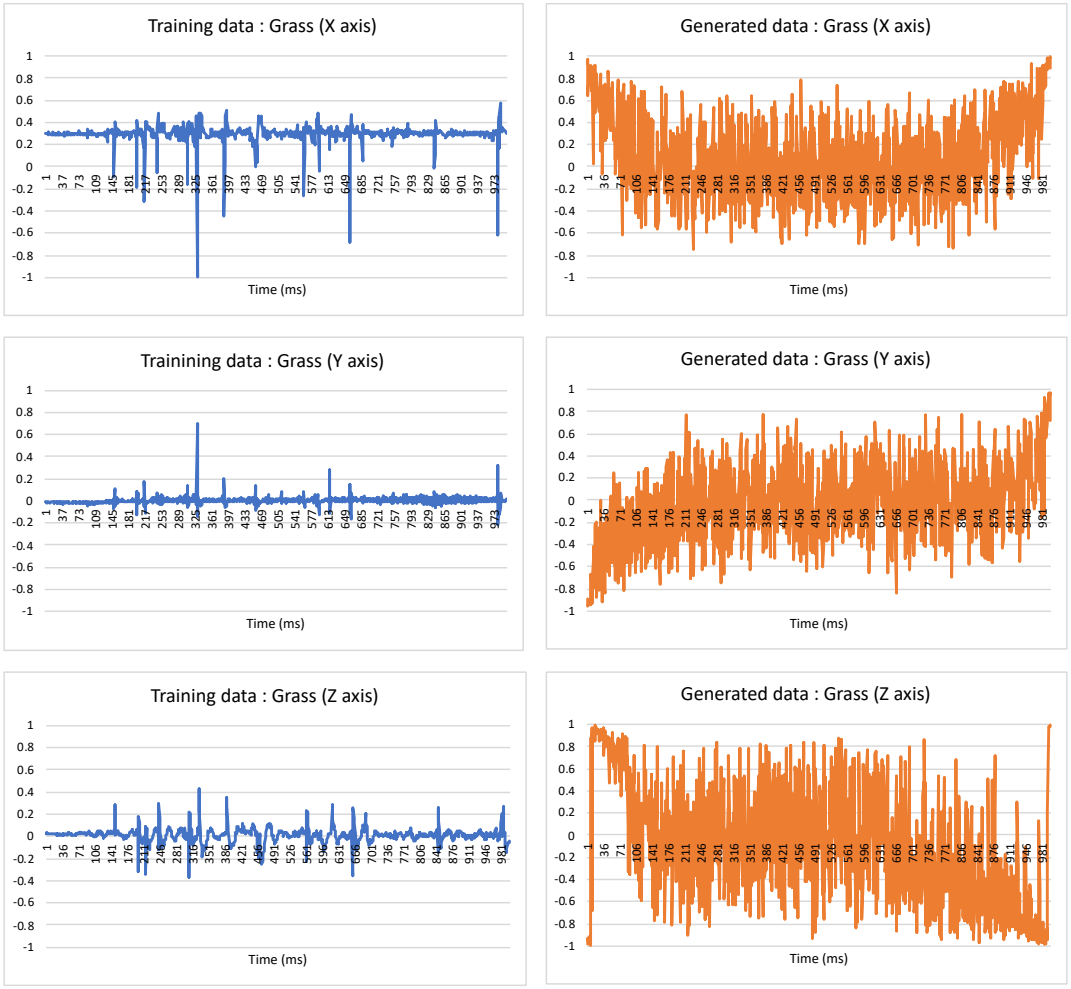

図 3.4: 訓練データ例と C-RNN-GAN を用いたモデルによる生成データ例の比較．青のグラフが訓練データ，橙色のグラフが生成データである

図 3.4 の結果を見ると，どの軸においても訓練データの変化の特徴を学習できていないことがわかる．どの軸においても，元となる訓練データよりも変化の激しいデータが生成されてしまった．この結果の原因としては，モデルの表現力が足らず，訓練データの複雑な変化

を学習できなかったことが考えられる．しかし，これ以上モデルの大型化を行う場合，用いる機材をより高性能なものに変更する必要がある．実際に LSTM 層を generator, discriminator 共に 1 層増やしたモデルを学習させようとしたところ，現在の機材では GPU のメモリが足りず学習が不可能であった．また，現在 1 種類のデータの学習に 6 時間程度かかるが，モデルの大型化を行った場合，学習の完了にさらに時間がかかることが考えられる．これらの観点から，C-RNN-GAN をベースにしたモデルを改良する場合，計算コストが非常に高くなることが考えられる．

3.3 DCGAN を用いた生成モデルについて

3.3.1 モデルの構成について

ここでは， DCGAN を用いたデータ生成モデルについて詳説する．DCGAN は GAN の generator と discriminator に Convolutional Neural Network (CNN) [31] を用いることで高品質なデータ生成を可能にしたモデルである．CNN では，入力データに畳み込み演算を行うことで，データの特徴を維持したままデータの処理が可能である．この特性により，CNN を用いることでデータの時系列特徴を逃さずに学習することが可能と考えられ，CNN を用いた GAN である DCGAN を用いることで時系列データを効果的に学習できる可能性がある．

我々は DCGAN をベースにデータ生成モデルを作成した．データ生成モデルの構成を表 3.2, 表 3.3 に示す．表中の “Generator” と “Discriminator”はモデルを構成するニューラルネットワーク層の構成を示す．“Input”は入力層を示す．“Output”は出力層を示す．入力層と出力層の間にある各層は隠れ層を示す．“Kernel Size”は畳み込みフィルタの形を示し，“Output Shape”は各層で出力されるデータの形を示す．モデルが複雑なため，generator と discriminator をそれぞれ別の表にて示す．このモデルでは 3 軸時系列データを訓練データとすることを想定して作成されており，各軸の時系列方向にのみ畳み込み演算を行うことで各軸の特徴が学習中に混ざってしまうことを防ぐ．

はじめに generator 側について示す．generator の入力は前述の C-RNN-GAN を用いたモデルと同様である．出力は 3 軸の時系列データである．このモデルでは C-RNN-GAN を用いたモデルとは異なり，一つのモデルで 3 軸の時系列データの生成が可能である．generator は 9 つの転置畳み込み層と 2 つの全結合層によって構成されている．転置畳み込み層では，入力データに空白を足してサイズを拡大したあとに畳み込み演算を行うことでデータのアップサンプリングを行う．generator 側において，畳み込みフィルタのサイズは畳み込み 1 層目から 2 層目，6 層目，8 層目と 9 層目においては 1×10，その他の層では 1×9 とした．これは後述のデータ生成テストにおいて，生成データの長さを訓練データと同じ長さである 5000 点になるように調整するためである．畳み込みフィルタ数はすべての層で 64 枚に統一した．活性化関数は出力層のみ tanh 関数，その他の層では Rectified Linear Unit (ReLU) 関数を用いた．活性化関数を $h(x)$ としたとき，ReLU 関数を活性化関数として用いる場合，式 3.8 のようになる．グラフにすると図 3.5 のようになる．ReLU 関数は 0 か入力値のどちらか大きい方を出力とする単純な関数であるが，Glorot ら [40] が活性化関数として ReLU 関数を使用することで

表 3.2: DCGAN をベースにしたデータ生成モデルの generator の構成

Generator	Kernel Size	Output Shape (n = batch size)
Input : Uniform(-1,1) + label		(n, 100 + label)
Dense		(n, 1024)
Batch normalization		(n, 1024)
ReLU		(n, 1024)
Dense		(n, 768)
Batch normalization		(n, 768)
ReLU		(n, 768)
Reshape		(n, 3, 2, 128)
Trans Conv2D (Stride = (1,2))	(1, 10, 64, 128)	(n, 3, 12, 64)
Batch normalization		(n, 3, 12, 64)
ReLU		(n, 3, 12, 64)
Trans Conv2D (Stride = (1,2))	(1, 10, 64, 64)	(n, 3, 32, 64)
Batch normalization		(n, 3, 32, 64)
ReLU		(n, 3, 32, 64)
Trans Conv2D (Stride = (1,2))	(1, 9, 64, 64)	(n, 3, 71, 64)
Batch normalization		(n, 3, 71, 64)
ReLU		(n, 3, 71, 64)
Trans Conv2D (Stride = (1,2))	(1, 9, 64, 64)	(n, 3, 149, 64)
Batch normalization		(n, 3, 149, 64)
ReLU		(n, 3, 149, 64)
Trans Conv2D (Stride = (1,2))	(1, 9, 64, 64)	(n, 3, 305, 64)
Batch normalization		(n, 3, 305, 64)
ReLU		(n, 3, 305, 64)
Trans Conv2D (Stride = (1,2))	(1, 10, 64, 64)	(n, 3, 618, 64)
Batch normalization		(n, 3, 618, 64)
ReLU		(n, 3, 618, 64)
Trans Conv2D (Stride = (1,2))	(1, 10, 64, 64)	(n, 3, 1244, 64)
Batch normalization		(n, 3, 1244, 64)
ReLU		(n, 3, 1244, 64)
Trans Conv2D (Stride = (1,2))	(1, 10, 64, 64)	(n, 3, 2496, 64)
Batch normalization		(n, 3, 2496, 64)
ReLU		(n, 3, 2496, 64)
Trans Conv2D (Stride = (1,2))	(1, 10, 1, 64)	(n, 3, 5000, 1)
Output : Tanh		(n, 3, 5000, 1)

表 3.3: DCGAN をベースにしたデータ生成モデルの discriminator の構成

Discriminator	Kernel Size	Output Shape (n = batch size)
Input : Training data or Generated data		(n, 3, 5000, 1)
Conv2D (Stride=(1, 2))	(1, 10, 1, 64)	(n, 3, 2496, 64)
LeakyReLU (α =0.2)		(n, 3, 2496, 64)
Conv2D (Stride=(1, 2))	(1, 10, 64, 64)	(n, 3, 1244, 64)
LeakyReLU (α =0.2)		(n, 3, 1244, 64)
Conv2D (Stride=(1, 2))	(1, 10, 64, 64)	(n, 3, 618, 64)
LeakyReLU (α =0.2)		(n, 3, 618, 64)
Conv2D (Stride=(1, 2))	(1, 10, 64, 64)	(n, 3, 305, 64)
LeakyReLU (α =0.2)		(n, 3, 305, 64)
Conv2D (Stride=(1, 2))	(1, 9, 64, 64)	(n, 3, 149, 64)
LeakyReLU (α =0.2)		(n, 3, 149, 64)
Conv2D (Stride=(1, 2))	(1, 9, 64, 64)	(n, 3, 71, 64)
LeakyReLU (α =0.2)		(n, 3, 71, 64)
Conv2D (Stride=(1, 2))	(1, 9, 64, 64)	(n, 3, 32, 64)
LeakyReLU (α =0.2)		(n, 3, 32, 64)
Conv2D (Stride=(1, 2))	(1, 10, 64, 64)	(n, 3, 12, 64)
LeakyReLU (α =0.2)		(n, 3, 12, 64)
Conv2D (Stride=(1, 2))	(1, 10, 64, 64)	(n, 3, 2, 64)
LeakyReLU (α =0.2)		(n, 3, 2, 64)
Reshape		(n, 768)
Dense		(n, 1024)
Dropout		(n, 1024)
Dense		(n, 1)
Output : Sigmoid		(n, 1)

他の活性化関数よりも効率的な学習ができる可能性があることを示した．また，関数が非常に単純であることから，シグモイド関数よりさらに計算負荷を減らすことが可能である．また，$x > 0$ において微分値が常に 1 であるため，各層の重み更新時に重みに関する微分の値が小さくなりすぎることによって学習が止まる危険性を減らすことが可能である．

$$h(x) = \max(0, x) \tag{3.8}$$

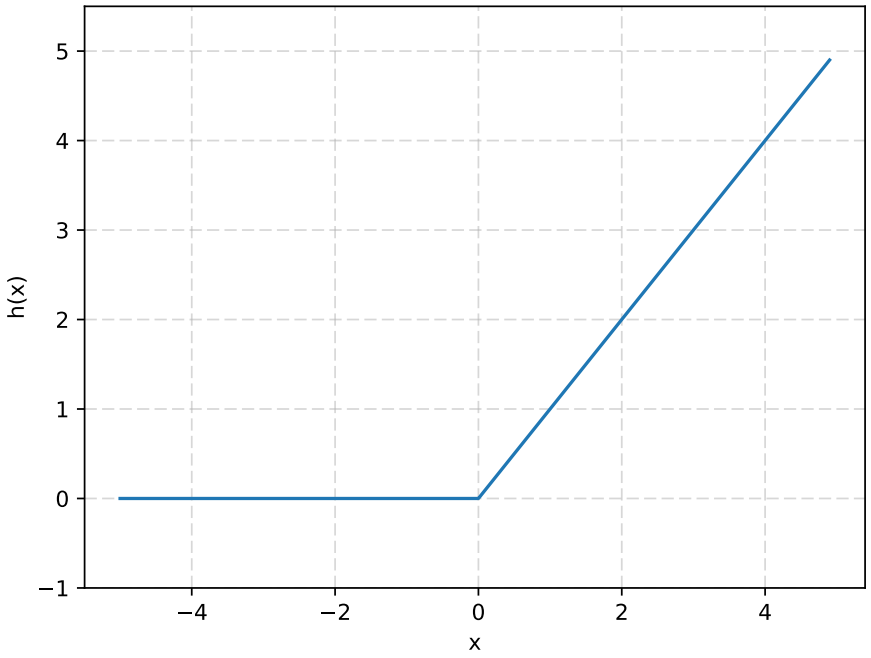

図 3.5: ReLU 関数のグラフ

このモデルの generator では，損失関数には正解の確率分布と分類器の予測分布の交差エントロピーを用いる．t_k を正解ラベル，y_k を出力データとすると，本研究で用いる交差エントロピー E の式は (3.5) で表される．ここでの出力データは discriminator の出力データを指している．GAN においての generator の学習は，generator にてデータを生成し，そのデータを discriminator にて分類した結果を損失関数にて評価することで行われる．損失関数の値を用いて重みの更新を行うが，この際 discriminator の重みは更新せず，generator の重みのみを更新することで generator の学習を行う．

$$E = -\sum_k t_k \log y_k \tag{3.9}$$

generator の重みの更新手法に関しては，C-RNN-GAN を用いたモデルと同じく，Adam を用いる．ハイパパラメータの設定は，$\alpha = 0.0001, \beta_1 = 0.9, \beta_2 = 0.999, \epsilon = 10^{-8}$ とした．また，generator 側において，勾配消失による学習の停止を防ぐため，各層の活性化関数の計算後に Batch Normalization [41] を用いた．Batch Normalization は学習を行う際の入力データの集合に対し，平均と分散を求めて，それらを基に入力データの分布について平均が 0，分散が 1 になるように正規化を行う．m 個の入力データ $\{x_1 ... x_i ... x_m\}$ とそれに対する平均 μ_B，分散 σ_B^2 とし，m 個の出力データ $\{y_1 ... y_i ... y_m\}$ とすると，このアルゴリズムは式 3.10 で表される．γ と β は学習によって最適化されるパラメータである．初期値は $\gamma = 1$，$\beta = 0$ である．ϵ は微小量 (1×10^{-5}) である．各層において活性化関数を使用した後にこのアルゴリズムを用いることで，勾配消失による学習の停止を防いだ．

$$
\begin{aligned}
\mu_B &= \frac{1}{m}\sum_{i=1}^{m} x_i \\
\sigma_B^2 &= \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_B)^2 \\
\hat{x_i} &= \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}} \\
y_i &= \gamma \hat{x_i} + \beta
\end{aligned}
\tag{3.10}
$$

次に discriminator 側について説明する．discriminator 側では 9 つの畳み込み層と 2 つの全結合層を持つ構成とした．discriminator の入力は訓練データまたは generator にて生成されたデータである．discriminator の出力は入力データが本物のデータである確率とした．畳み込みフィルタのサイズは畳み込み 1 層目から 2 層目，4 層目，8 層目と 9 層目においては 1×10，その他の層では 1×9 とした．ここでは試験的に generator でのアップサンプリングとは逆の処理を行うように設定した．畳み込みフィルタ数はすべての層で 64 枚に統一した．活性化関数は出力層のみシグモイド関数，その他の層では Leaky ReLU 関数を用いた．Leaky ReLU 関数を活性化関数として用いた場合，式 3.11 で表され，グラフにすると図 3.6 のようになる．Leaky ReLU 関数は ReLU 関数を改良した関数であり，$x < 0$ において小さな勾配を持つ．これにより，負の値が入力されることによる勾配消失を防ぐ．式 3.11 中の α は勾配の値を決めるハイパーパラメータであるが，このモデルでは $\alpha = 0.2$ とした．また，重みの最適化手法と損失関数については generator と同じ設定を使用した．

$$
h(x) = \begin{cases} x & (x > 0) \\ \alpha x & (x \leq 0) \end{cases}
\tag{3.11}
$$

3.3.2 単純な時系列データを用いたデータ生成実験

ここでは，触察時のデータよりもシンプルな時系列データを用いてデータ生成テストを行い，構築したモデルによって時系列データの効果的な学習が可能か調査する．データ生成には

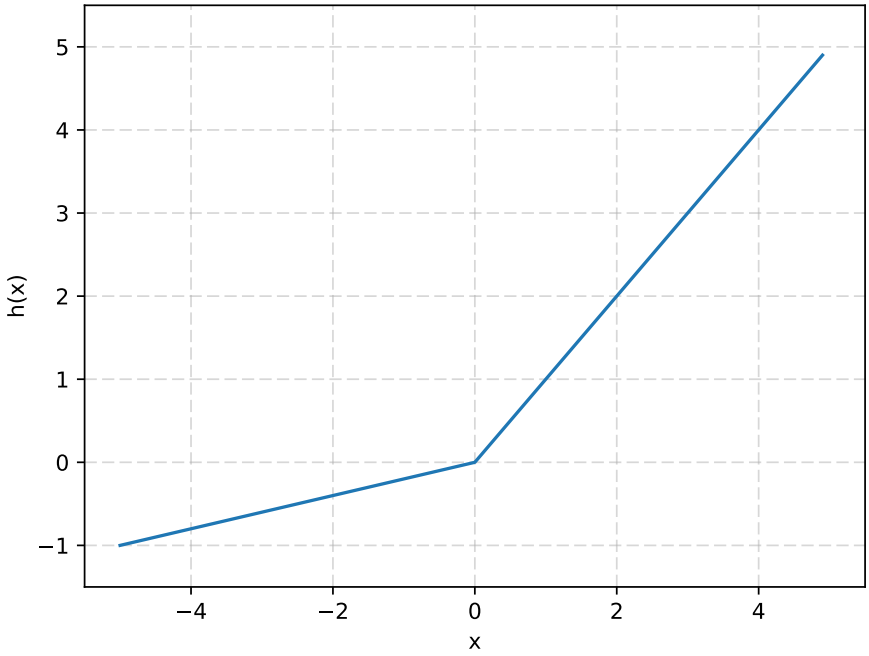

図 3.6: Leaky ReLU 関数のグラフ

DCGAN をベースとしたモデルを用いる．訓練データとして用いた時系列データは，正弦波，鋸歯状波，矩形波の 3 種類であった．これらは全て 1 軸時系列データであった．周波数はすべて 440 Hz とし，値は 0 から 1 の間になるようにした．訓練データ中の連続 5,000 点を 100 個抜き出して学習させる試行を 10,000 回行うことでモデルを作成した．この実験に限り，1 軸時系列データを学習できるようモデルの generator と discriminator の入力層を調整した．モデルから生成されるデータは $1 \times 5,000$ の 1 軸時系列データとした．学習時間は約 3 時間であった．

訓練データとモデルによって生成されたデータのグラフを図 3.7 に示す．Sin がサイン波，Sawtooth が鋸歯状波，Square が矩形波を示す．また，訓練データと生成データからそれぞれ連続 1,000 点を抜き出したグラフを図 3.8 に示す．これはデータの短い区間を可視化することで，データの細かい変化を学習できているか確認するためである．これらのグラフの縦軸は-1 から 1 に正規化された訓練データもしくは生成データの値である．グラフの横軸はデータの各点を時系列順に並べた際のインデックスとなっている．

結果を見ると，まずサイン波については，訓練データの変化の特徴を捉えたデータが生成されている．図 3.8 を見ると，データの変化の仕方が連続 1,000 点全域において似ていることがわかる．しかし，訓練データに比べ，生成データは振幅が小さくなっていることがわかる．この結果から，本実験の生成モデルではデータの大まかな特徴は捉えられるものの，データの細かい変化を捉えることが難しい可能性がある．次に鋸歯状波の結果について見ると，図 3.7 中の横軸 409–2,313 の区間については訓練データの値の変化の特徴を捉えたデータが生成されているが，サイン波と同じくデータの振幅が異なっている．また，他区間は訓練データと全く異なる波形となっている．一定の区間のみ訓練データの特徴を反映している傾向は，矩形波の生成データにおいても見られる．この結果から，本実験での GAN の構成では訓練データの特徴の学習は可能であるものの，より訓練データの特徴を捉えたデータ生成を行うためには GAN の構成を改良する必要があることがわかった．

3.3.3 収集データを用いたデータ生成実験

DCGAN を用いたモデルにより，時系列データの特徴を捉えたデータの生成が可能であることが先述の実験で判明したため，次のステップとして，実際に物体を触察した際のデータを用いたデータ生成実験を行った．生成の結果から，DCGAN を用いたモデルにより触察時のデータをもとにした効果的なデータ生成が可能か考察した．

訓練データとしては，C-RNN-GAN を用いた生成実験時と同じものを用いた．訓練データ中の連続 5,000 点を 100 個抜き出してモデルに学習させる試行を 10,000 回行うことでデータ生成モデルを作成した．学習の際，訓練データは-1 から 1 の間に正規化した．生成されるデータは $3 \times 5,000$ の 3 軸時系列データとした． 学習時間は約 3 時間であった．

先述の実験と同様に訓練データと生成データをグラフ化することで可視化した．訓練データと生成データのグラフを図 3.9 に示す．結果を見ると，まず X 軸に関しては，大まかな値の分布については図 3.9 中の縦軸 0.4 付近を中心として一致している．しかし，3.9 中の横軸 181–217 の区間のような，訓練データの値の変化の大きな特徴については生成データにあまり

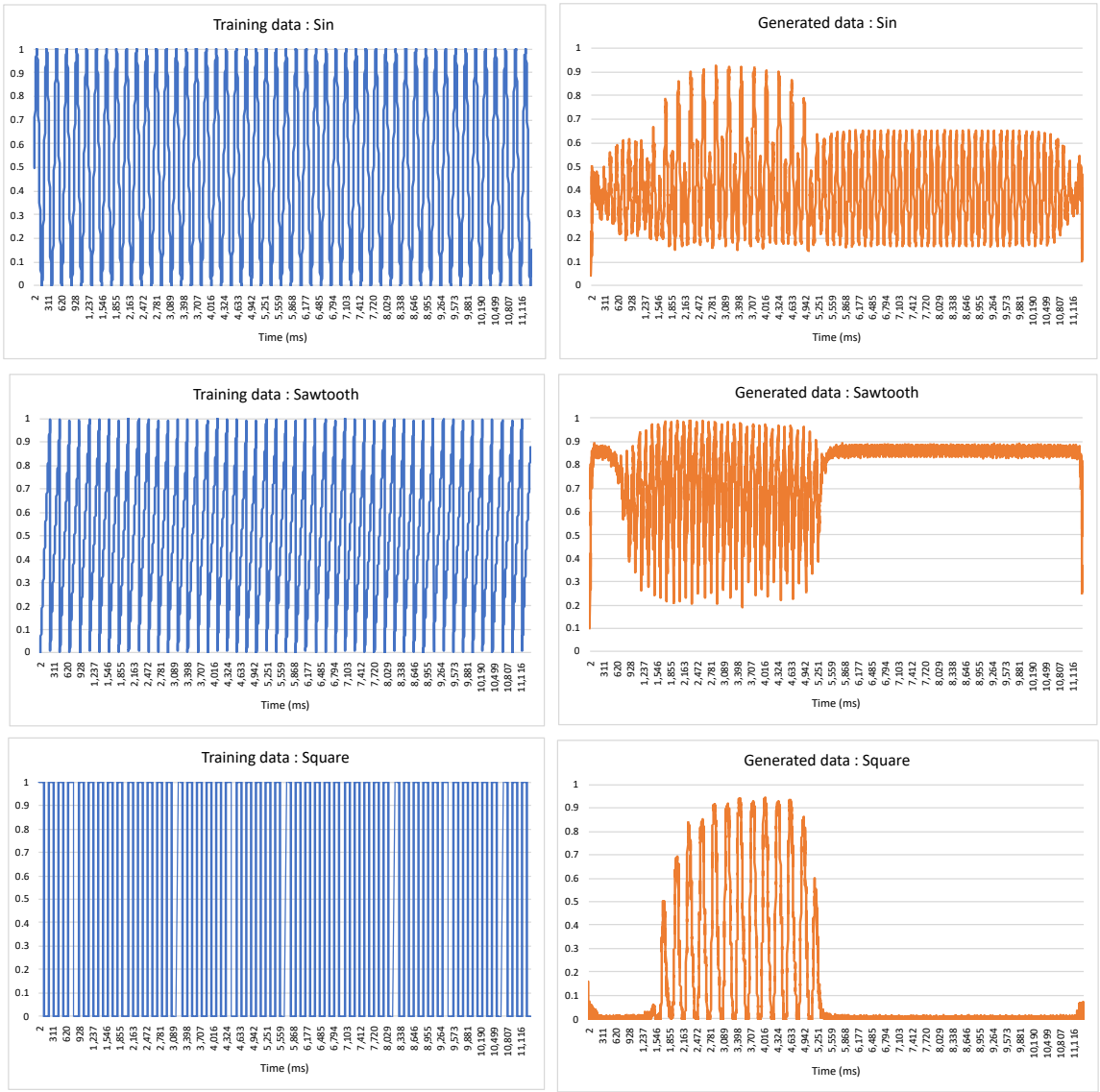

図 3.7: シンプルな時系列データをもとに DCGAN により生成された時系列データ．左列の青いグラフが訓練データ，右列の橙色のグラフが生成データである

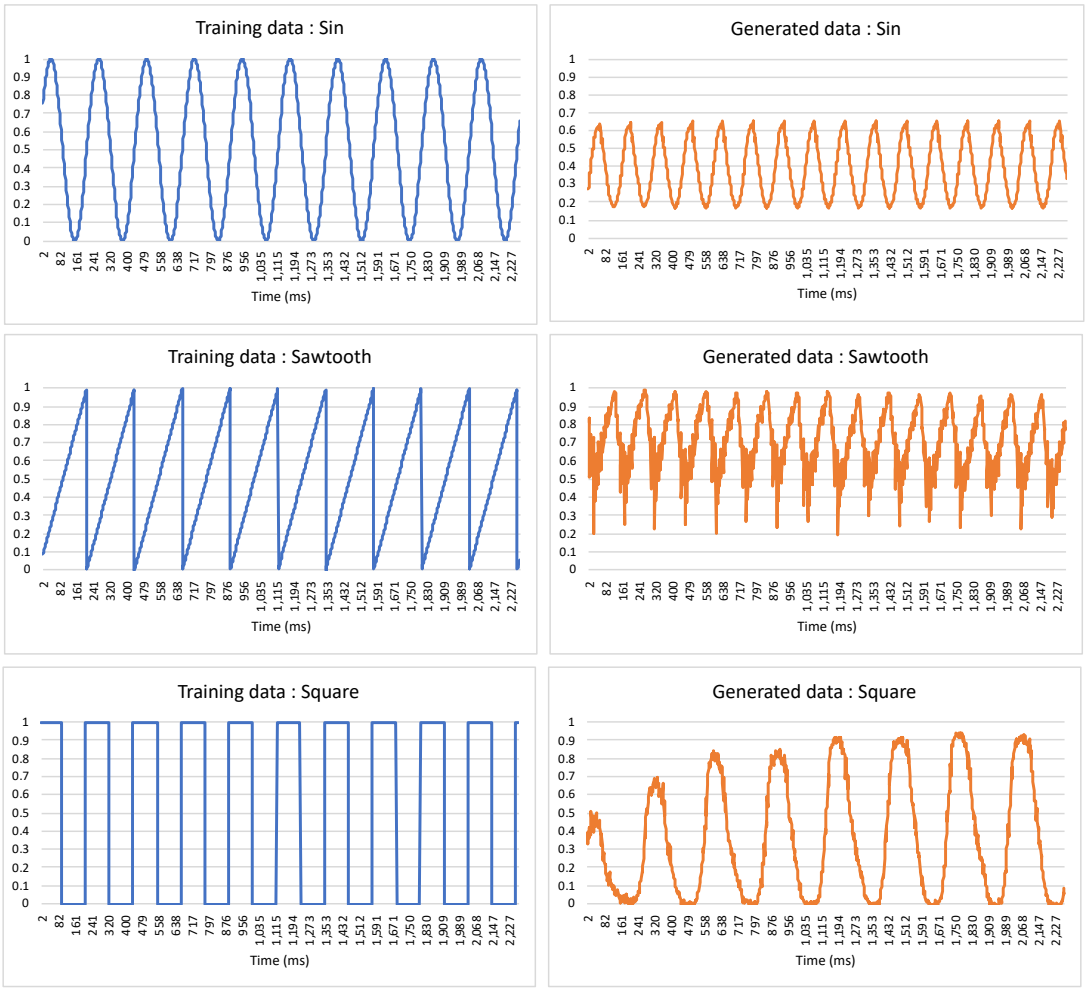

図 3.8: シンプルな時系列データをもとに DCGAN により生成された時系列データ．このグラフは図 3.7 のグラフと同じデータを用いており，連続 1,000 点分の各生成データと訓練データを示している

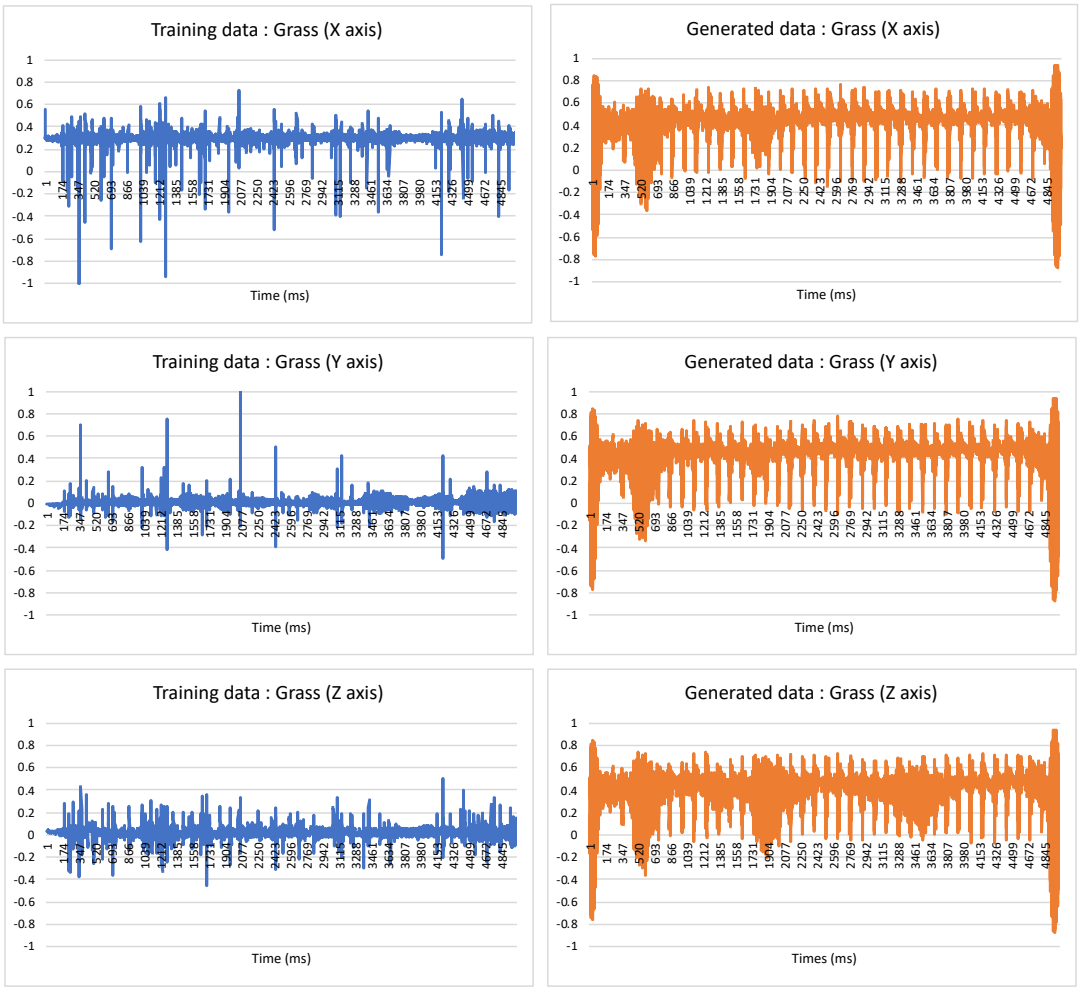

図 3.9: 触察時の 3 軸加速度データをもとに DCGAN により生成された時系列データ．左列の青いグラフが訓練データ，右列の橙色のグラフが生成データである

反映されていない．Y 軸，X 軸に関しては，訓練データと波形の特徴が異なるデータが生成されている．これらの結果の原因としては，前述のシンプルな波形を用いた生成実験の際よりも複雑に値が変化するデータを訓練データとして用いており，モデルの表現力が足らず，有効な学習ができなかったためと考えられる．しかし，C-RNN-GAN を用いた場合より訓練データに近いデータを生成することが出来たため，RNN をベースとする手法より DCGAN をベースにする手法の方が時系列データ生成に適していると考えられる．本実験で用いた DCGAN は最も基本的な構成をベースとしている．DCGAN から派生したより高性能な GAN を用いることで，より質の高いデータ生成が出来る可能性がある．また，図 3.10 に示す，学習時の損失関数の値の推移を見ると，早い段階で学習が収束し，その後一定の値の変化をし続けていることがわかる．このことから，学習時に勾配消失が発生している可能性がある．したがって，より質の良いデータ生成を行うためには，目的関数の改良やネットワーク構成の改良などの学習を安定化させる措置を導入する必要がある可能性が高い．

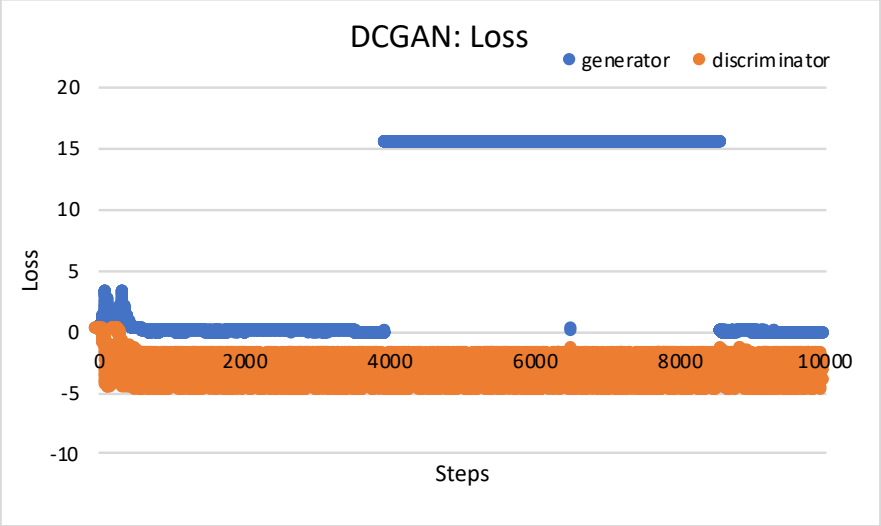

図 3.10: DCGAN を用いたモデルの損失関数の値

3.3.4　生成データを用いた触覚提示実験

ここでは，DCGAN をベースにしたモデルによる生成データを用いた触覚提示実験について示す．5 つのテクスチャについての 3 軸加速度データを用いてデータ生成を行い，生成されたデータを用いて触覚提示実験を行った．ここでは各テクスチャにつき 1 つづつモデルを作成し，データ生成を行った．訓練データは本文中で示した手法と同じ手法で収集した．モデルの構成は畳み込み層が generator，discriminator ともに 6 層のものを用いた．その他のハイパパラメータは本文中のモデルと同一である．学習時，モデルへ入力する訓練データは $3 \times 1,000$ の 3 軸加速度データとし，モデルの出力は $3 \times 1,000$ の 3 軸時系列データとした．

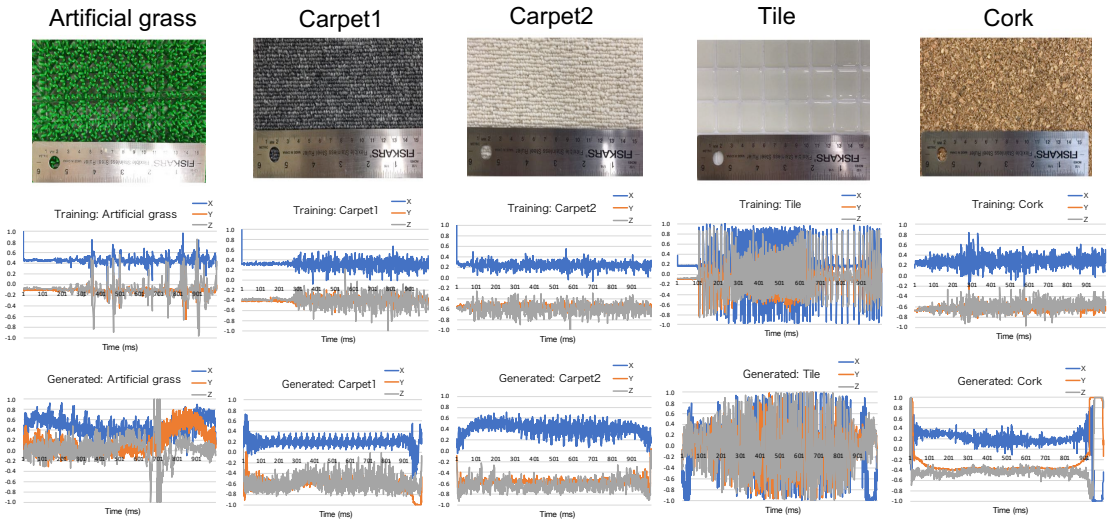

図 3.11: 触覚提示テストのために用いたテクスチャと生成データ例

データ生成に用いたテクスチャと生成例を図 3.11 に示す．図 3.11 では，上段の 5 つのグラフが訓練データ，下段の 5 つのグラフが生成データを示す．ここでの訓練データは，全データの中からランダムに 1,000 点抽出したものである．生成結果を見ると，Carpet1, Carpet2, Tile については特徴を捉えたデータ生成に成功している．しかし，Cork の Y 軸や，Artificial grass については，訓練データと大きく異なるデータが生成されているのがわかる．これについてはデータ生成実験の際と同様に，モデルの学習が安定しないことが原因として考えられる．学習の安定化を図るアルゴリズムをモデルに導入することで，結果が向上する可能性がある．

生成データにより，どれほどのリアリティのある触覚提示ができるか調査するため，生成したデータを用いて触覚提示実験を行った．触覚提示には，Saga ら [3] が開発したタブレット装着型の振動触覚ディスプレイを用いた．このディスプレイでは，タブレット上に指を置くための台座が設置されており，その台座とタブレット 4 隅のモータが糸で接続されている．タブレット四隅のモータにより，台座につながっている糸を巻き取ることで，台座を振動させて触覚を提示する．このディスプレイでは，時系列データをモータを巻き取るための信号として利用する．また，このディスプレイでは，X 軸の振動と Y 軸の振動を独立に制御することができる．我々のモデルでは，3 軸の振動データを生成するため，このディスプレイのように複数の振動を利用したディスプレイでの触覚提示が可能である．このように，我々のモデルの生成データは，単一の振動信号を用いる触覚ディスプレイのための信号としての利用にとどまらず，多くの情報を利用する触覚ディスプレイにも利用できる．

ここでは，触覚提示の手順について説明する．被験者には実物のテクスチャと触覚ディスプレイを触り比べてもらい，触覚ディスプレイのリアリティを回答してもらう．ディスプレイには生成データか訓練データがランダムに表示される．被験者には，生成データまたは訓練データによる提示にどれほどのリアリティを感じたか 5 段階のリッカート尺度を用いて評

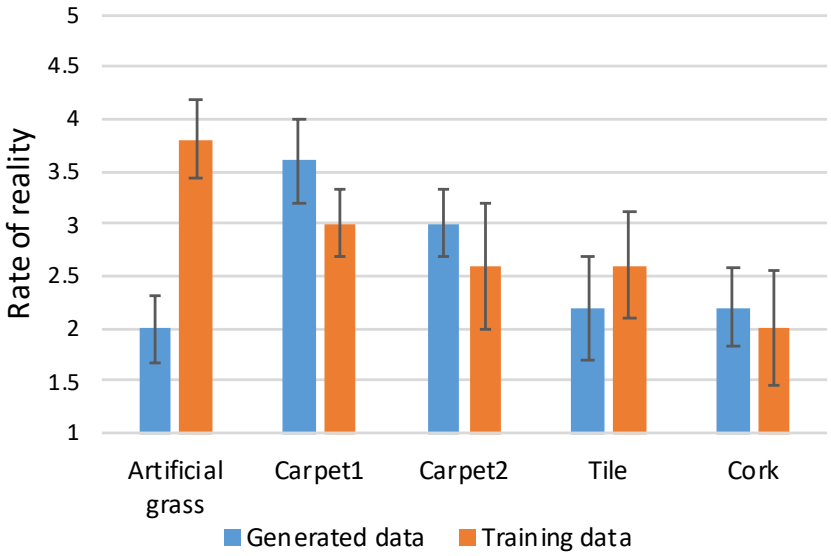

図 3.12: 触覚提示テストの結果

価してもらった．この試行を生成データと訓練データそれぞれ各テクスチャごとに一回行った．被験者は 4 人 (20 代，男性) であった．

実験結果を図 3.12 に示す．青色のグラフが生成データ，橙色のグラフが訓練データの結果を示す．結果を見ると，Carpet1, Carpet2, Tile, Cork については生成データにより訓練データと同程度のリアリティを実現できていることがわかる．Artificial grass については生成データのリアリティが訓練データのリアリティに比べ，大きく低いという結果になった．これについては，データ生成例のグラフから，訓練データの特徴を反映しきれていないデータを用いて触覚提示を行ったためだと考えられる．

3.4 WaveGAN を用いた生成モデルについて

3.4.1 モデルの構成について

ここでは，WaveGAN を用いたデータ生成モデルについて詳説する．WaveGAN は Donahue らが提案した音声生成用の GAN であり，DCGAN から派生した手法の一つである．基本的な構成は DCGAN と一致しているが，目的関数に WGAN-GP [25] のアルゴリズムを導入し，学習を安定化させた点で大きく異なる．さらに，WaveGAN では，ネットワークの各層の活性化関数の出力の位相をずらすことで時系列データの学習を安定化させている．また，ネットワークにおける畳み込み層の畳み込みフィルタサイズを大きくすることで，データの時系列特徴を学習しやすいネットワーク構造を実現した．これらの特性から WaveGAN は DCGAN

と比べて，より時系列データの学習に特化した性能を持つ．よって WaveGAN を用いることで，高品質な時系列データ生成が可能であると考えられる．

我々は WaveGAN をベースにデータ生成モデルを作成した．構築した GAN の構成の概略を表 3.4 に示す．C は訓練データのクラス数，n はバッチサイズである．表中の “Generator”と “Discriminator”はモデルを構成するニューラルネットワーク層の構成を示す．“Input”は入力層を示す．“Output”は出力層を示す．入力層と出力層の間にある各層は隠れ層を示す．“Kernel Size”は畳み込みフィルタの形を示し，“Output Shape”は各層で出力されるデータの形を示す．

前述の通り，この GAN の構成は WaveGAN をベースに構築している．しかし，WaveGAN は単クラス生成を想定したモデルであるため，様々な種類のデータを生成するには不向きである．この問題に対応するため，我々のモデルでは Conditional GAN [19] を採用した．Conditional GAN では，訓練データにラベルデータを付与して学習させることで，データ生成時にラベルデータに対応したデータを指定して生成することができる．我々の GAN に Conditional GAN を導入することで，多クラス生成を実現した．図 3.13 に多クラス生成の概要図を示す．GAN の generator の入力に用いる乱数ベクトルにクラス固有のラベルデータを付与し，学習させる．これによりデータ生成時にラベルデータを GAN に入力することで，任意のクラスのデータを生成することが可能になる．

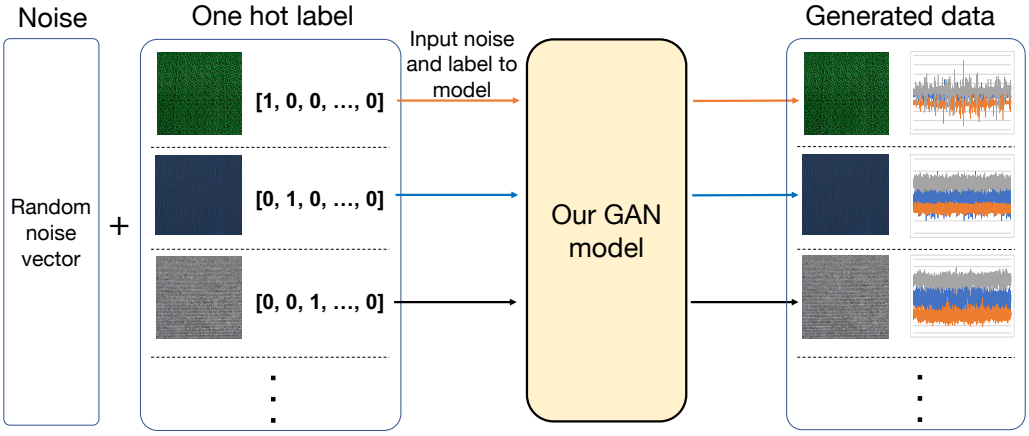

図 3.13: WaveGAN ベースの GAN による多クラス生成の概要図

ラベルデータとしては，学習させるクラス数と同じ長さを持ち，0 か 1 のみを要素として持つベクトルである one hot vector を用いた．今回は訓練データのクラスに対応したインデックスのみ 1 とした one hot vector をラベルデータとした．また，今回は WaveGAN と異なり，3 軸の加速度データを処理する．このため，学習の際，軸ごとに独立に学習できるように，時間方向にのみ畳み込みを行うように設定した．

generator と discriminator の詳細な設定は以下のようになっている．なお，このモデルでは，訓練データは 16394 点の 3 軸時系列データを想定している．これは後述の生成実験時，訓練データ点数を Ujitoko らの先行研究と合わせるためである．generator 側の入力は-1 から 1 の一

表 3.4: WaveGAN をベースにしたデータ生成モデルの構成

Generator	Kernel Size	Output Shape
Input : Uniform(-1,1)+C		(n, 100+C)
Dense	(100+C, 49152)	(n, 49152)
Reshape		(n, 3, 16, 1024)
LeakyReLU (α = 0.2)		(n, 3, 16, 1024)
Trans Conv2D (Stride = (1, 4))	(1, 25, 512, 1024)	(n, 3, 64, 512)
LeakyReLU (α = 0.2)		(n, 3, 64, 512)
Trans Conv2D (Stride = (1, 4))	(1, 25, 256, 512)	(n, 3, 256, 256)
LeakyReLU (α = 0.2)		(n, 3, 256, 256)
Trans Conv2D (Stride = (1, 4))	(1, 25, 128, 256)	(n, 3, 1024, 128)
LeakyReLU (α = 0.2)		(n, 3, 1024, 128)
Trans Conv2D (Stride = (1, 4))	(1, 25, 64, 128)	(n, 3, 4096, 64)
LeakyReLU (α = 0.2)		(n, 3, 4096, 64)
Trans Conv2D (Stride = (1, 4))	(1, 25, 1, 64)	(n, 3, 16384, 1)
Output : Tanh		(n, 3, 16384, 1)

Discriminator	Kernel Size	Output Shape
Input : Training data or Generated data		(n, 3, 16384, 1+C)
Conv2D (Stride = (1, 4))	(1, 25, 1+C, 64)	(n, 3, 4096, 64)
LeakyReLU (α = 0.2)		(n, 3, 4096, 64)
Phase Shuffle (n = 2)		(n, 3, 4096, 64)
Conv2D (Stride = (1, 4))	(1, 25, 64, 128)	(n, 3, 1024, 128)
LeakyReLU (α = 0.2)		(n, 3, 1024, 128)
Phase Shuffle (n = 2)		(n, 3, 1024, 128)
Conv2D (Stride = (1, 4))	(1, 25, 128, 256)	(n, 3, 256, 256)
Phase Shuffle (n = 2)		(n, 3, 256, 256)
LeakyReLU (α = 0.2)		(n, 3, 256, 256)
Conv2D (Stride = (1, 4))	(1, 25, 256, 512)	(n, 3, 64, 512)
LeakyReLU (α = 0.2)		(n, 3, 64, 512)
Phase Shuffle (n = 2)		(n, 3, 64, 512)
Conv2D (Stride = (1, 4))	(1, 25, 512, 1024)	(n, 3, 16, 1024)
LeakyReLU (α = 0.2)		(n, 3, 16, 1024)
Reshape		(n, 49152)
Output : Dense	(49152, 1)	(n, 1)

様分布に基づき生成された 1 × 100 の乱数ベクトルとした．出力は訓練データに基づく 3 軸時系列データである．どのようなデータが生成されるかは訓練データにより異なる．generator は WaveGAN と同じく，1 つの全結合層と 5 つの転置畳み込み層にて構成される．畳み込みフィルタのサイズは全ての転置畳み込み層で 1 × 25 とした．活性化関数は出力層のみ tanh 関数，その他の層では Leaky ReLU 関数を用いた．discriminator 側において，入力は訓練データまたは generator より生成されたデータとし，出力は，入力データが畳み込まれた数値データである．discriminator は WaveGAN と同じく，1 つの全結合層と 5 つの畳み込み層にて構成される．活性化関数は全畳み込み層において Leaky ReLU 関数を用いた．損失関数としては WGAN-GP [25] のアルゴリズムを採用し，discriminator の出力値を損失関数の計算に利用する．また，WaveGAN に用いられている，活性化関数の出力の位相をずらす手法である PhaseShuffle のアルゴリズムを採用し，学習の効率化を図った．

さらに，WaveGAN とは異なり，我々のモデルでは generator, discriminator の両者の畳み込み層の重みの初期値に He らが提案した初期値 [42] を用い，さらなる学習の安定化を図った．重みの最適化手法については Donahue らと同じく Adam を用いた．ハイパパラメータの設定を表 3.5 に示す．この表の “Generator updates per discriminator”は discriminator の学習 1 回につき generator を何回学習させるかの数値である．この表では，“Generator updates per discriminator” の値のみ WaveGAN と異なる．これについては，後述の生成実験において，表に示す値を用いた方が WaveGAN で用いられている値を用いる場合より学習が安定したためである．

表 3.5: WaveGAN をベースにしたモデルのハイパパラメータ

Name	Value
Batch size	64
Phase Shuffle	2
WGAN-GP λ	10
Generator updates per discriminator	2
Optimizer	Adam (α = 1e-4, β 1 = 0.5, β 2 = 0.9)

3.4.2 収集データを用いた 1 クラスデータ生成実験

前節で構築したモデルを用いてデータ生成実験を行い，構築したモデルによって時系列データの効果的な学習が可能か調査する．ここで使用するモデルは多クラス生成に対応しているが，まず 1 クラスのデータを学習させ，モデルによってそのデータを再現できるか調査する．

訓練データとしては，現実のテクスチャを指でなぞった場合の 3 軸加速度データを用いた．データ収集の条件については，C-RNN-GAN を用いた生成実験時と同じである (図 3.3)．学習時，全収集データの中から連続 1,024 点のデータをランダムに 70,000 個抜き出し，訓練データセットとした．このデータセットの中から 64 個のデータを抜き出して学習させる試行を 1,093 回行うことを 1 epoch とし，50 epoch 行った．学習の際，訓練データは-1 から 1 の間に

正規化した．なお，この実験に限り generator, discriminator 共に畳み込み層を 5 層から 3 層に削減してデータ生成を行った．それに伴い，入力層，出力層についても調整を行った．モデルを縮小した理由は訓練データの長さが 16,384 点に満たないためである．モデルから生成されるデータは $3 \times 1,024$ の 3 軸時系列データとした．学習時間は約 9 時間であった．

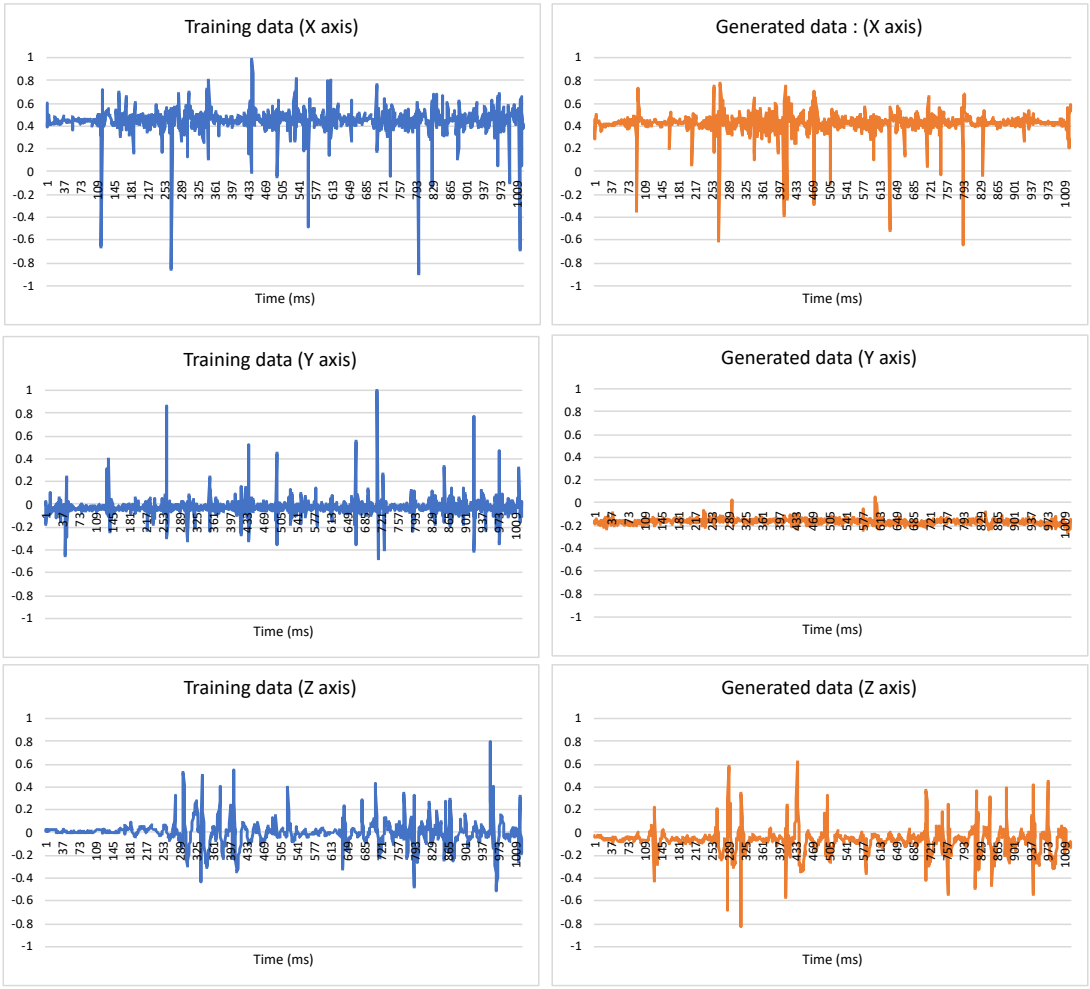

図 3.14: 短い収集データをもとに WaveGAN を用いて生成された時系列データ．左列の青いグラフが訓練データ，右列の橙色のグラフが生成データである．

訓練データとモデルによって生成されたデータのグラフを図 3.14 に示す．全訓練データ中から縦軸は-1 から 1 に正規化された訓練データもしくは生成データの値である．グラフの横軸はデータの各点を時系列順に並べた際のインデックスとなっている．まず X 軸の生成結果についてみると，訓練データの特徴をよく捉えたデータが生成されていることがわかる．訓練データの特徴としては，例えば訓練データのグラフの横軸 253–325 のような，大きな値の変化の前後に細かく値が変化している点が挙げられる．この特徴が生成データにも表れてお

り，効果的なデータ生成ができた可能性がある．値の分布も縦軸 0.4 付近で一致している．同様に，Z 軸についても訓練データの特徴を捉えたデータの生成に成功したことがわかる．Y 軸について見ると，訓練データに比べて平坦なデータが生成されていることがわかる．これについては，畳み込み層を減らしたことで，モデルの表現力が下がったため，効率的な学習ができなかったことが原因として考えられる．また，前述の C-RNN-GAN や DCGAN を用いた際の実験結果と本実験の結果を比較すると，本実験の生成データの方が訓練データの特徴をより反映したデータを生成できていると考えられる．このことから，WaveGAN を用いたモデルがより時系列データ生成に適している可能性があることがわかった．

3.4.3 データセットを用いた多クラスデータ生成実験

構築したモデルを用いて，データの生成と，生成されたデータの再現性検証実験を行う．本実験では，先行研究でも用いられているデータセットを訓練データとしてデータ生成を行い，我々の学習モデルにより先行研究同様にデータ生成が可能か調査した．また，9 クラスのデータを訓練データとし，我々が構築したモデルにより効果的な多クラスデータ生成ができるか検証した．

訓練データとしては，LMT haptic texture database [43] から 9 クラス分の 3 軸加速度データを抜粋し，使用した．抜粋したデータの元となったテクスチャ画像を Fig. 3.15 に示す．これらのテクスチャは Ujitoko ら [30] が生成に用いていたものと同様のものとなる．これらの加速度データはペン型デバイスによってテクスチャを 1 方向になぞることにより収集され，約 10 kHz で収集された．

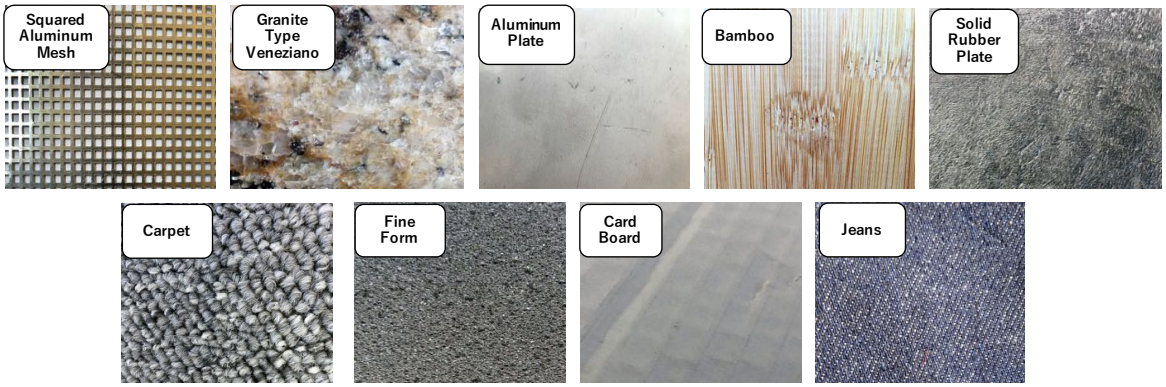

図 3.15: WaveGAN をもとにしたモデルの学習のため LMT haptic texture database より抜粋したテクスチャ

discriminator にデータを入力する際，訓練データを-1 から 1 の間で正規化した．学習の際，各テクスチャのデータの中から連続 16,384 点のデータをランダムに抜き出して訓練データとした．テクスチャごとにデータを 6,000 個抜き出し，合計 54,000 個を学習に用いた．生成データは 16384 点の 3 軸時系列データである．今回は mini batch size を 64 に設定し，36epoch 分の

学習を行った．訓練データ量については，MNIST データセット [20] のデータ量と等しくなるように調整した．MNIST は 10 クラスの手書き数字データセットで，1 クラスあたり約 6,000 個の手書き数字画像データで構成されている．MNIST は GAN をはじめとした様々な機械学習モデルの性能評価に利用されており，MNIST と同等のデータ量を用意できれば訓練データ量としては十分と考えられる．

学習には GPU (GTX1080 Ti×2) を導入した Windows PC を用いた．学習に用いた時間は約 47 時間であった．汎用的な GPU と PC を用いて約 2 日で学習が終了するため，モデルの新規作成が容易である．1 クラスのデータ生成の場合より学習時間が増加したが，これは訓練データのサイズが 1,024 点から 16,384 点に増加したためと考えられる．また，この実験では 1 クラスデータ生成の場合より畳み込み層数が generator, discriminator ともに 3 層から 5 層に増加しており，計算が煩雑化したことも理由として考えられる．今回用いた GPU 付き PC よりさらに高性能な計算機が利用可能な場合，さらなる高速化が実現できる．

抜粋した 9 クラスについてのデータ生成結果について確認するため，生成データと訓練データをスペクトログラム画像化した．スペクトログラムの例を Fig. 3.16 に示す．スペクトログラム化するため，生成データと訓練データに対し Short-Time Fourier Transform (STFT) を行った．この際，ハミング窓 (N=256) を用い，ホップサイズを 128 とした．スペクトログラムの値は 0 から 1 に正規化されている．

スペクトログラムを比較すると，訓練データの特徴が生成データによく現れていることがわかる．特に，Bamboo テクスチャは訓練データと生成データの区別が難しく，モデルがデータの特徴を効果的に学習できたことがわかる．しかし，Granite Type Veneziano テクスチャにおける特徴があらわれる周期については，生成データに現れていない．特徴が出現する周期に関連し，Granite Type Veneziano テクスチャでは，一瞬全ての周波数帯が弱く出力されている箇所があるが，生成データにはその特徴が表れていない．これについては，訓練データ自体が元のデータから一部分を抜き出したものであり，抜き出したデータごとに，特徴が現れる周期が異なってしまうため，モデルが正確には学習できなかったものと考えられる．訓練データの特徴が現れる周期を一定になるようにデータ抽出すれば，現在のモデルでも周期を学習できると考えられる．

3.4.4 収集データを用いた多クラスデータ生成実験

訓練データについて説明する．訓練データについては，指に加速度センサを装着し，テクスチャを 1 方向になぞった際の 3 軸加速度データを用いた．データの収集には 9 種類のテクスチャを用いた．収集に用いたテクスチャを Fig. 3.19 に示す．Artificial Grass は突起の多い人工芝のテクスチャである．Artificial Leather はスベスベした布のテクスチャである．Carpet は硬めの絨毯のテクスチャである．Cork Sheet は板状のコルクである．Punched Plastic Sheet はツルツルした板に無数のパンチ穴が空いているテクスチャである．Tile は規則的に並んだタイルのテクスチャである．Place Mat 01, Place Mat 02, Place Mat 03 はそれぞれ表面の材質が異なるランチョンマットである．収集の様子を Fig. 3.20 に示す. 収集者は著者のうちの 1 人である．収集の際，収集者の指の爪に加速度センサをテープで固定し，その指腹でテクスチャを

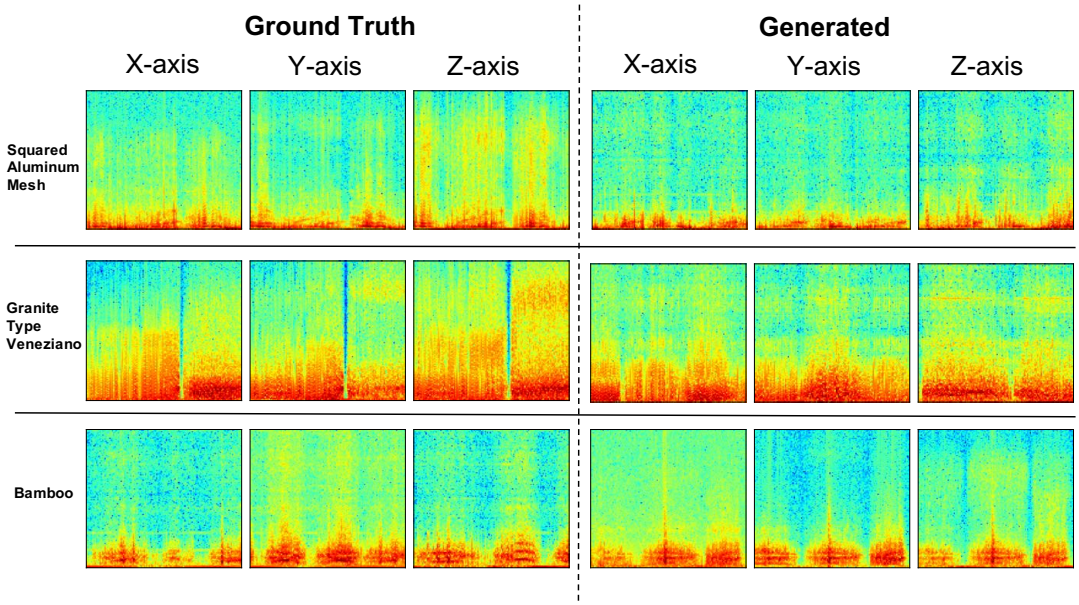

図 3.16: LMT Haptics texture database をもとにした生成データのスペクトログラム例．9 クラスのスペクトログラム画像から 3 クラスを抜粋した．左の 3 つが訓練データ (Ground Truth)，右の 3 つが生成データの各軸加速度スペクトログラムである

一方向 (左から右) に 6 秒間なぞる．センサからのデータ収集は 1 kHz で行われた．収集の際の指の速度はメトロノームを使用し，約 5 cm/s になるようにした．指の角度はテクスチャから約 45 度になるようにした．1 つのテクスチャにつきデータ収集試行を 80 回行った．また，各収集データの先頭と末尾の連続 1,000 点を切り取ることで，なぞり始めと終わりの急激な加速度の変化をモデルに学習させないようにした．この連続 1,000 点を切り取る処理の概要図を図 3.17 に示す．

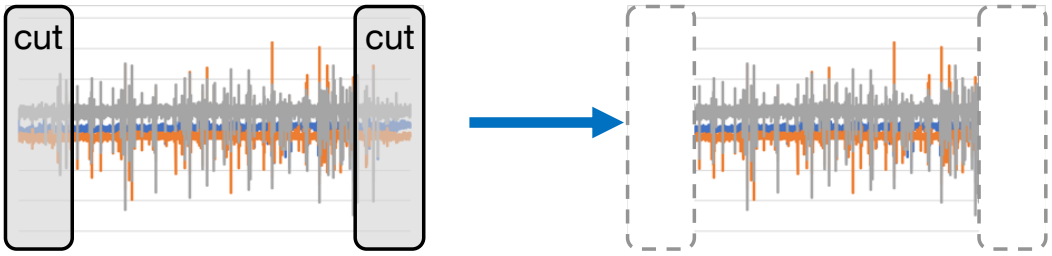

図 3.17: 収集データの先頭と末尾の連続 1000 点を切り取る処理の概要図

学習に用いたハイパパラメータは LMT データを用いた場合と同様のものを用いた．また，学習の際，収集データの長さが 16,384 点に満たないため，各収集データを 10 回繰り返したデータを作成し，その 40,000 点前後のデータから，16,384 点をランダムに抜き出して学習

させた．このデータを繰り返す処理の概要図を図 3.18 に示す．訓練データ量は LMT Haptics texture database を用いた際と同じである．生成データは 16,384 点の 3 軸時系列データである．学習量は，こちらも 36 epoch 行った．学習には前章と同じ PC を用いた．学習にかかった時間は約 46 時間であった．よって，データセットと異なるデータを用いても，高速なモデル作成が可能であることがわかった．

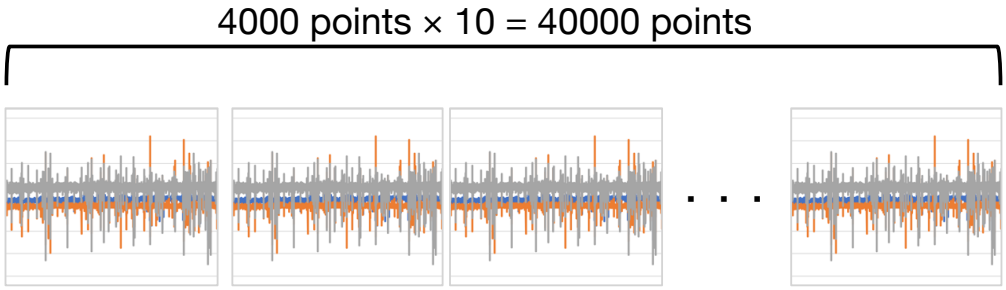

図 3.18: 収集データを 10 回繰り返したデータを作成する処理の概要図

データ生成結果のスペクトログラムの例を Fig. 3.21 に示す．スペクトログラム生成の際の設定は，LMT データセットを用いた場合と同じである．スペクトログラムを比較すると LMT データセットを用いた場合と同じく，訓練データの特徴を捉えたデータ生成に成功したことがわかる．例にあげた 3 つのテクスチャにおいては，訓練データと生成データの区別が難しく，効果的なデータ生成ができたことがわかった．よって，LMT データセットとは収集条件が異なるデータに関しても，提案モデルが対応できることがわかった．

図 3.19: データ収集に用いた 9 種類のテクスチャ

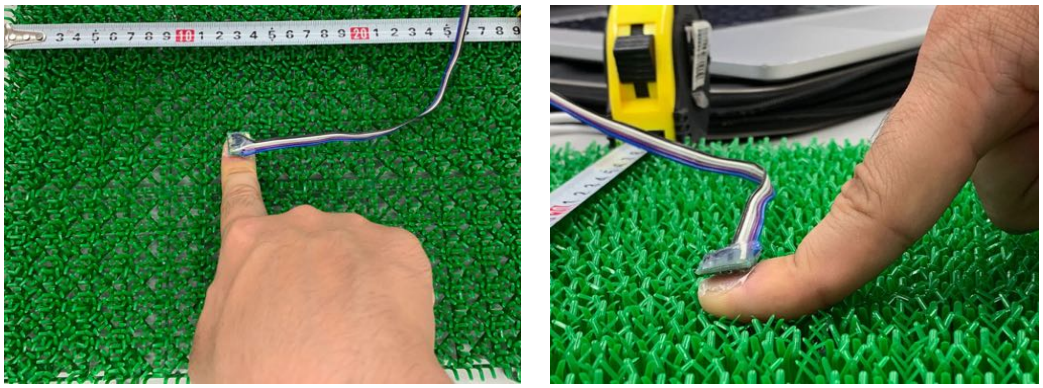

図 3.20: 9 クラスのデータを用いた生成テストのためのデータ収集の様子

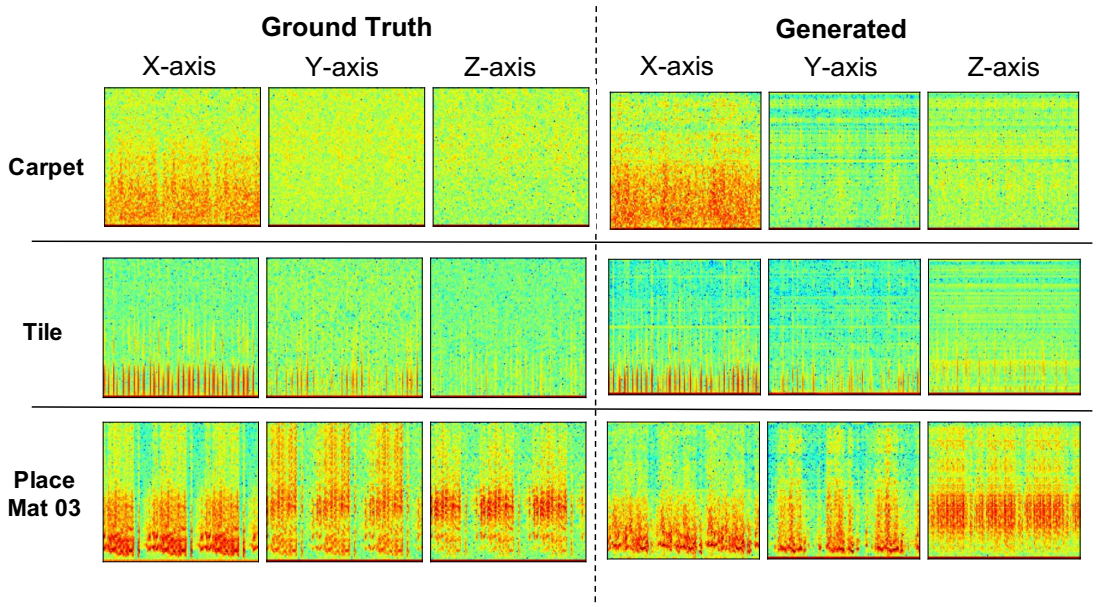

図 3.21: 収集データをもとにした生成データのスペクトログラム例．9 クラスのスペクトログラム画像から 3 クラスを抜粋した．左の 3 つが訓練データ (Ground Truth)，右の 3 つが生成データの各軸加速度スペクトログラムである

第4章　触覚提示実験

4.1　実験について

ここでは生成データを用いた触覚提示実験について述べる．我々のモデルにより生成されたデータをより詳しく評価するため，ユーザに対し生成データを用いた触覚提示をする実験を行った．この実験では，前章で述べた WaveGAN を用いたモデルを使用し，データ生成を行った．データ生成の際には前章で述べた 9 クラスの収集データを訓練データとして用いた．

この実験では，2 つの項目について調査を行う．一つは訓練データを用いた触覚提示と生成データを用いた触覚提示を区別できるかである．被験者が 2 つのデータを区別できなかった場合，有効なデータ生成ができているとする．もう一つは訓練データと生成データを用いた触覚提示にどれだけのリアリティがあるかである．実際のテクスチャと触覚ディスプレイを触り比べることで，被験者に今回の手法による触覚提示がどれだけのリアリティがあるか評価してもらう．訓練データを用いた場合と生成データを用いた場合の評価値が近いほど，有効なデータが生成できているとする．実験の結果を受けて，今回の手法で有効なデータ生成ができているか考察を行う．評価実験の具体的な手順に関しては，Ujitoko ら [30] の手法を用いることにした．今回の触覚提示には前述の Saga ら [3] が開発したタブレット装着型の振動触覚ディスプレイを用いる．

提示を行う際，訓練データと生成データともに先頭 4,000 点のみを用いた．これは，前章にて行ったデータを引き延ばす処理により発生すると思われる，収集した際にはなかったデータの周期を触覚提示に影響させないためである．また，被験者は 10 人 (男 8 人，女 2 人，20 代) であった．今回の実験は，筑波大学の倫理審査委員会に承認されている (審査承認番号，2019R299)．実験開始時，インフォームドコンセントに関する同意書を被験者に記入してもらった．

4.2　実験手順

実験の詳しい手順について説明する．実験の様子と実験に用いた触覚ディスプレイの様子を Fig. 4.1 に示す．被験者は触覚ディスプレイ上の台座に人差し指を置き，ディスプレイの表面を左から右になぞってもらう．

図 4.2 に実験中の触覚ディスプレイ上の画面と評価用インタフェースについて示す．触覚ディスプレイ上の A の領域と B の領域に触覚が提示されており，被験者には領域にそって触覚ディスプレイをなぞってもらう．なぞってもらう際，画面下部に 500 mm/s で移動するバー

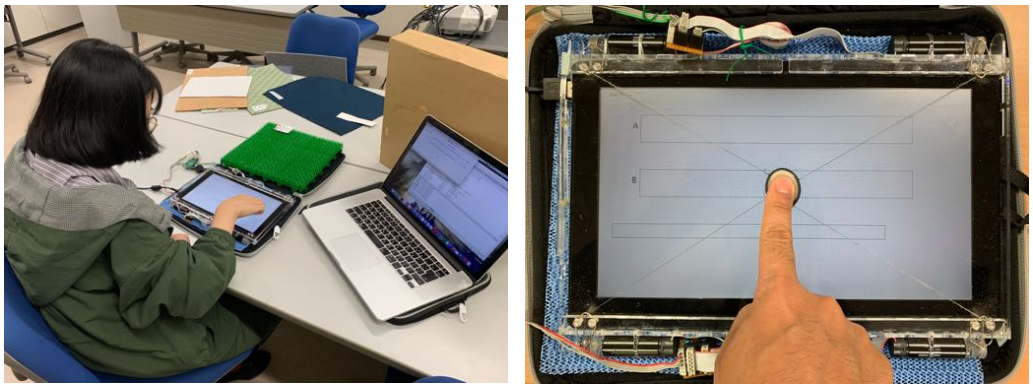

図 4.1: 触覚提示実験の様子．左図：実験の様子，右図：実験中の触覚ディスプレイの様子

を設置し，それを用いて被験者にはなぞる速度を 500 mm/s に調節してもらう．A と B にはそれぞれ訓練データ由来の触覚と生成データ由来の触覚のどちらかが提示されている．被験者には A と B のどちらが生成データ由来の触覚提示であるか，評価用インタフェースの対応したボタンにより回答してもらう．A と B のどちらが生成データ由来の提示であるかは，1 試行ごとにランダムに決定した．また，触覚ディスプレイの周りに触覚ディスプレイで提示しているデータの元となったテクスチャを設置し，被験者には A および B とテクスチャを触り比べてもらう．その後長さ 100 mm の Visual Analog Scale (VAS) [44] に基づき，A および B にどれほどのリアリティがあったか回答をしてもらう．実験終了後，VAS の値を元に評価を行う．

これら 2 つの調査を 1 試行として，1 テクスチャあたり 10 試行行った．つまり全体で被験者一人あたり 90 試行行った．全試行終了後，被験者に実験についての意見をもらうため，アンケート調査を行った．実験が終了するまでに要した時間は 1 時間ほどであった．

4.3 結果と考察

実験結果について説明する．図 4.3 に訓練データ由来の触覚提示と生成データ由来の触覚提示を区別する実験の正答率を示す．これらの値は全被験者の結果の平均である．正答率の値が 50%に近づくほど，被験者は生成データによる提示と訓練データによる提示の区別が付いていないということになる．すなわち，訓練データの特徴を反映した有効なデータ生成ができているとする．図 4.3 の結果を見ると，どのテクスチャの値も，ほぼ 50%となっている．よって，被験者は訓練データでの提示と生成データでの提示の区別がほぼできなかったものと考えられる．実験後アンケートにおいても，ほぼ全ての被験者が訓練データでの提示と生成データでの提示の区別がつかなかったと答えた．これらの結果から，我々が提案したモデルにより，実際の加速度データに近いデータを生成できる可能性が高いことがわかった．実際にテクスチャごとの結果を確認すると，大多数の被験者が正答率 40%から 60%の値を示した．特に Carpet テクスチャの結果については，10 人中 7 人が正答率 50%を示すという結果になった．この結果により，我々のモデルが特に Carpet のようなザラザラしたテクスチャのデータ

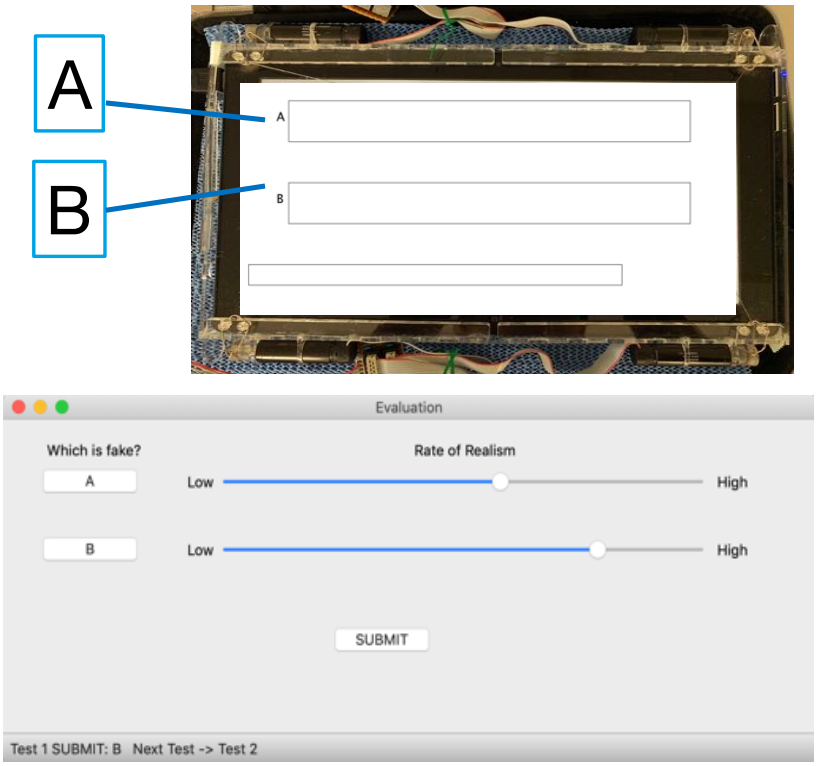

図 4.2: 触覚ディスプレイ上の画面と評価用インタフェース．上図：触覚ディスプレイの画面，下図：評価用インタフェース

をもとにしたデータ生成に対して有効である可能性があることがわかった．

次に，図 4.4 に生成データによる提示と訓練データによる提示のリアリティ値についての結果を示す．これらの値は全被験者の結果の平均である．生成データによる提示と訓練データによる提示のリアリティ値が近いほど，訓練データの特徴を反映した有効なデータ生成が出来ているとする．Fig. 4.4 に示されているリアリティ値を見ると，全テクスチャにおいて生成データによる提示と訓練データによる提示のリアリティ値はほぼ同一である．生成データによる提示と訓練データによる提示のリアリティに有意差があるかどうかを検証するため，スチューデントの t-検定をテクスチャごとのデータのペアに対して行ったところ，全てのテクスチャについて有意差は見られなかった ($p < 0.05$)．この結果から，生成データを用いて，訓練データを用いた場合と同等のリアリティを持つ触覚提示を行うことに成功したことがわかった．また，Ujitoko ら [30] の先行研究では，いくつかのテクスチャにおいて有意差が発生していることから，我々の手法により，先行研究よりも高品質なデータ生成ができた可能性がある．

リアリティ値について見ると，Artificial Leather と Tile 以外のテクスチャについては 50%から 60%の値を示している．Saga ら [3] が行った実験によれば，今回用いた振動触覚ディスプレイを用いて絨毯などの実際のテクスチャの質感を再現する際のリアリティ値は 50%から 70%で

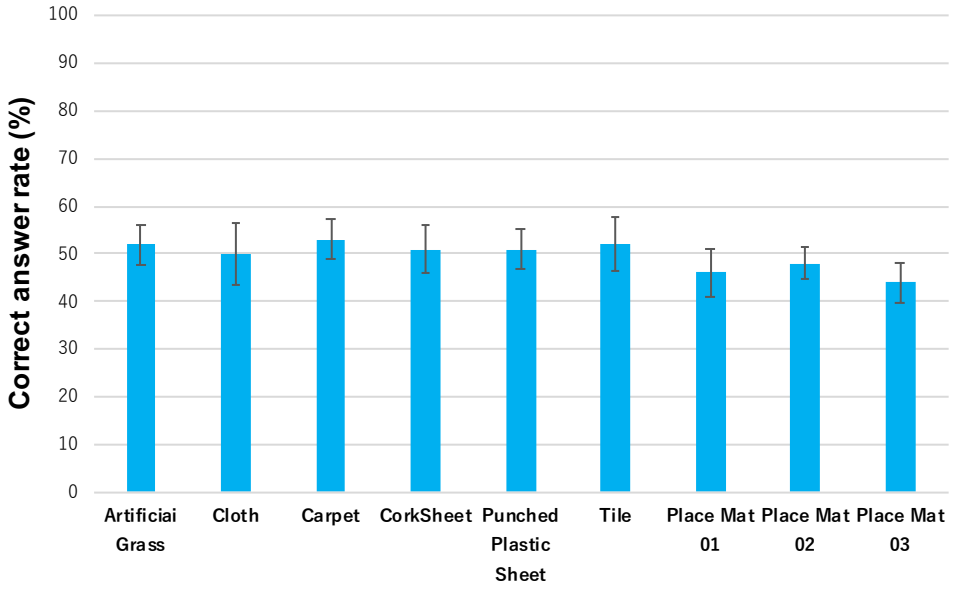

図 4.3: 触覚提示実験におけるテクスチャごとの生成データと訓練データの区別タスクの正答率

あった．今回の実験の結果は Saga らの実験結果に合致しており，触覚ディスプレイの性能を十分に用いた触覚提示を行うことができたことがわかった．リアリティ値の低い 2 つのテクスチャについて考察する．Artificial Leather については布のテクスチャであるため，今回用いた触覚ディスプレイではリアリティある提示が困難であった可能性がある．今回のディスプレイでは，指先に振動を提示するためザラザラした触感の再現には有効であるが，布のようなスベスベした触感を再現するには不向きである．今後，スベスベした触感の再現に優れた触覚ディスプレイを用いて調査を行う必要がある．また，Tile の結果については，データの特徴となる加速度の変化が小さく，被験者に非常に弱い振動が提示されてしまったためリアリティ値が低かったと考えられる．加速度の変化が小さかった理由としては Tile の凹凸が浅かったため，指先が Tile の凹凸をなぞった際の振動が少なかったことが原因であると考えられる．今回 Tile の結果ではリアリティ値は低かったものの，この値は触覚ディスプレイでの提示の仕方やデータ収集の仕方により改善すると考えられる．生成データと訓練データでのリアリティの差はほぼ無い上，Tile については訓練データに近いデータを生成できているため (図 3.21)，データ生成については成功しているものと考えられる．

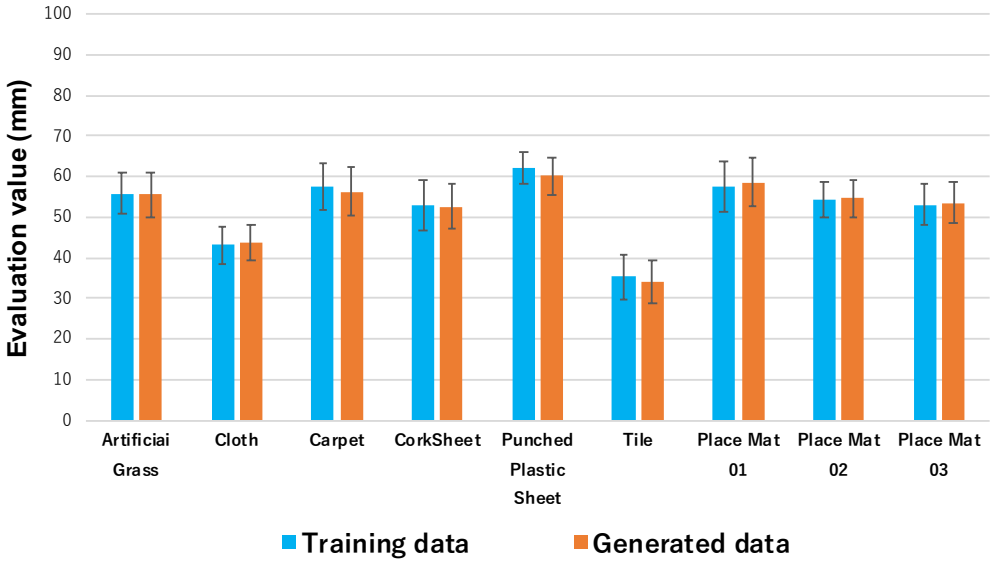

図 4.4: 触覚提示実験におけるテクスチャごとのリアリティ評価値．青色のグラフが訓練データを用いた触覚提示の結果，橙色のグラフが生成データを用いた触覚提示の結果である

第5章　ラベル合成による未知データの生成

ここではラベル合成による未知データの生成について述べる．触覚提示実験により，生成データにより訓練データの再現については成功したことが分かった．提案手法実現の次のステップとして，未知のデータ生成を行う．ここでは，GAN に入力するラベルデータを操作することで，訓練データにないデータを生成できるか調査する．ここで行うラベル合成についての概要図を図 5.1 に示す．通常，generator の入力となる one hot vector は，あるテクスチャに対応したインデックスのみ 1，その他は 0 の値を持つベクトルである．この実験では，あるテクスチャに対応したインデックスだけでなく，もう一つのテクスチャに対応したインデックスの値 0 以外に変更する．この手法により，2 つのテクスチャの特徴を合成したデータが生成できる可能性がある．

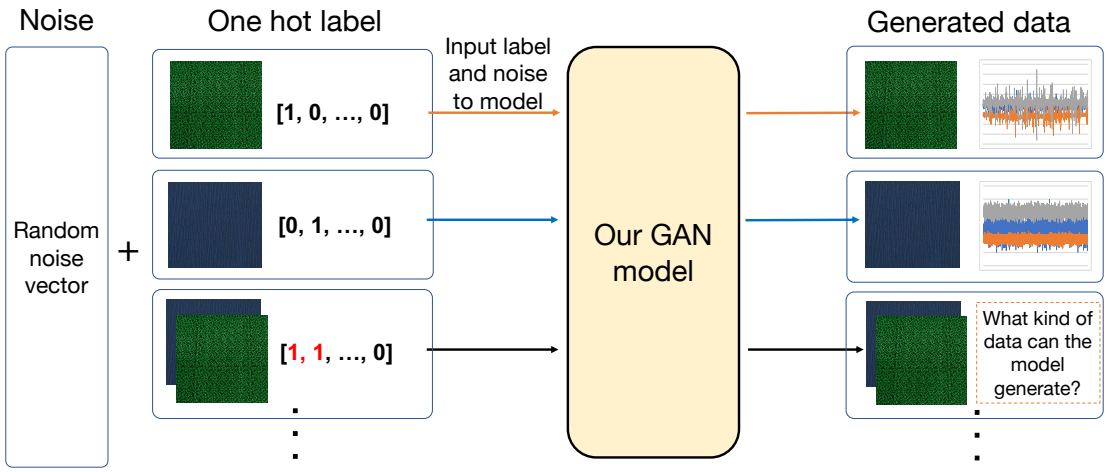

図 5.1: ラベル合成によるデータ生成の概要図

これに関連して，実際に我々が作成した GAN を用いてラベルの合成を行った．今回は，WaveGAN ベースの GAN を用い，我々が収集した加速度データについての生成を行った．合成するラベルは図 3.15 中の Tile と Place Mat 03 を用いた．データ生成の際，Place Mat 03 を元にデータを生成する際の one hot label の値を，Place Mat 03 に対応するところだけを 1 にするのではなく，Tile に対応する場所にも何らかの数値を入力し，生成を行った．

その結果を図 5.2 に示す．結果において，スペクトログラムの横に示されている数値は Tile のラベル値である．データ生成の際，入力ラベルの Place Mat 03 に対応するインデックスの

値を 1 とし，Tile に対応する値を変動させた．結果のスペクトログラムを見ると，Tile ラベルの値が増えるごとに Place Mat 03 の特徴と Tile の特徴が混ざっていくことがわかる．特に X 軸ではこの傾向が顕著である．Tile テクスチャの特徴は，スペクトルの強い場所と弱い場所が細かく繰り返される点であり，Place Mat 03 テクスチャの特徴は，スペクトルの強い場所が Tile テクスチャよりも広い点である．また，Place Mat 03 ではスペクトルの強い場所が周期的に 3 回繰り返されている．Tile ラベルの値が増えるほど，生成されるスペクトログラムに Tile の特徴が強く表れ，逆に Place Mat 03 の特徴が表れなくなっていくことがわかる．この結果から，GAN に入力するラベルデータを合成することで，触感データの合成ができる可能性が高いことがわかった．

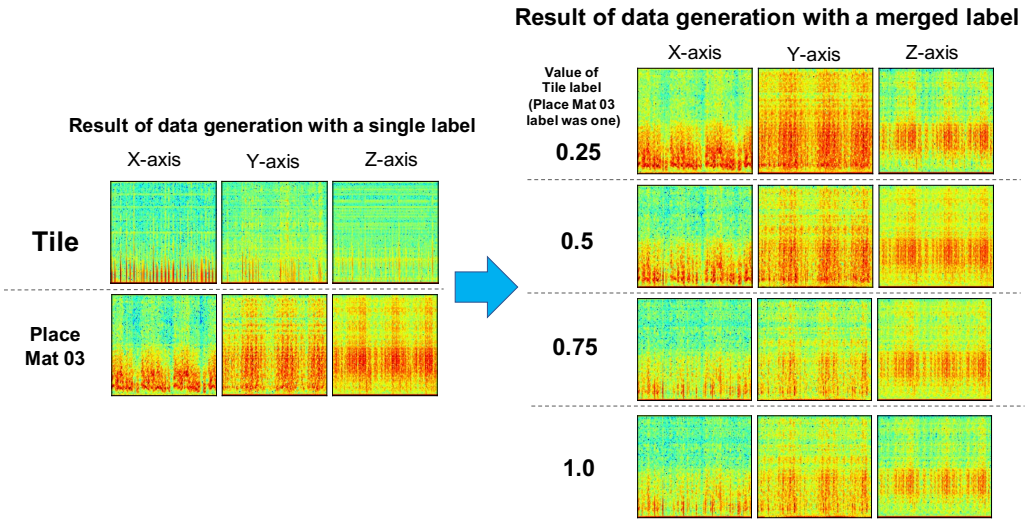

図 5.2: Tile ラベルと Place Mat 03 ラベルを合成したラベルによるデータ生成結果のスペクトログラム．左の 2 つのスペクトログラムがもともとの生成データ．右の 4 つが Place Mat 03 の one hot label の Tile に対応する場所の値を変化させた場合のデータ

また，データ生成に用いるテクスチャの違いが結果に影響するのか調査するため，図 3.15 に示す 9 クラスのテクスチャから得られた 3 軸加速度データについての生成を行った．合成するラベルは，Tile のラベルとその他 8 種類の各テクスチャのラベルとする．データ生成の際，Tile を元にデータを生成する際の one hot label の値を，Tile に対応するところだけを 0 以外に変更にするのではなく，Tile 以外のある 1 つのテクスチャに対応する場所にも何らかの数値を入力し，データ生成を行った．

データ生成の結果例を図 5.3，図 5.4，図 5.5 に示す．図 5.3 では Tile と ArtGrass, Tile と Cloth，Tile と Carpet の組み合わせについての結果を示す．図 5.4 では Tile と Cork，Tile と Punched Plastic Sheet, Tile と Place Mat 01 についての結果を示す．図 5.5 では Tile と Place Mat 02，Tile と Place Mat 03 についての結果を示す．ラベルの値については，試験的に，合成する 2 つのテクスチャの値の合計が 1 になるようにした．図 5.2 中では 2 つの数値がいくつかの組

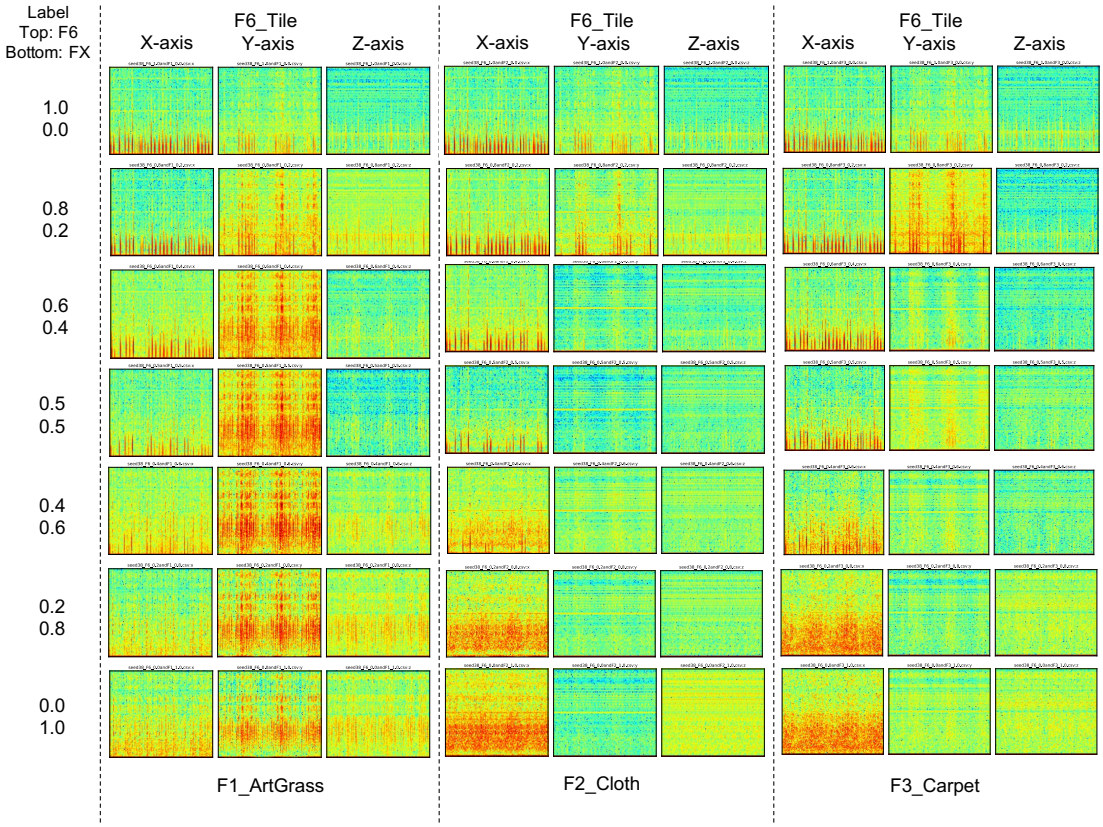

図 5.3: Tile ラベルと他のテクスチャのラベルを合成したラベルによるデータ生成結果のスペクトログラム．この図では，Tile と ArtGrass，Tile と Cloth，Tile と Carpet の組み合わせについての結果を示す

になっているが，2 つの数値の上に記載されている数値が Tile のラベル値，下に記載されている数値が Tile と合成するテクスチャのラベル値である．結果を見ると，どのテクスチャの組み合わせにおいてもラベル値の高い方のテクスチャの特徴を強く反映したデータが生成されていることがわかる．前述の生成テストと同じく，特に X 軸でこの傾向が顕著に表れている．このことから，扱うテクスチャを Place Mat 03 以外のものにした場合でもデータ合成が可能であることがわかった．また，2 つのテクスチャのラベルの値が同じ場合，Tile のテクスチャの特徴が強く生成データに表れている場合が多い．Tile テクスチャの特徴は，スペクトルの強い場所と弱い場所が細かく繰り返される点であるが，この特徴は Tile ラベルの値が小さい場合からでも生成物に現れている．結果を見ると，図 5.3 の Cloth と Carpet の結果を除く全ての結果で Tile のラベル値が 0.4 の場合から Tile の特徴が生成物に表れている．このことから，ラベルを合成した際，生成データに影響を与えやすい特徴がある可能性があることがわかった．

この章で述べた生成実験の結果により，GAN モデルに入力するラベルを操作することで，

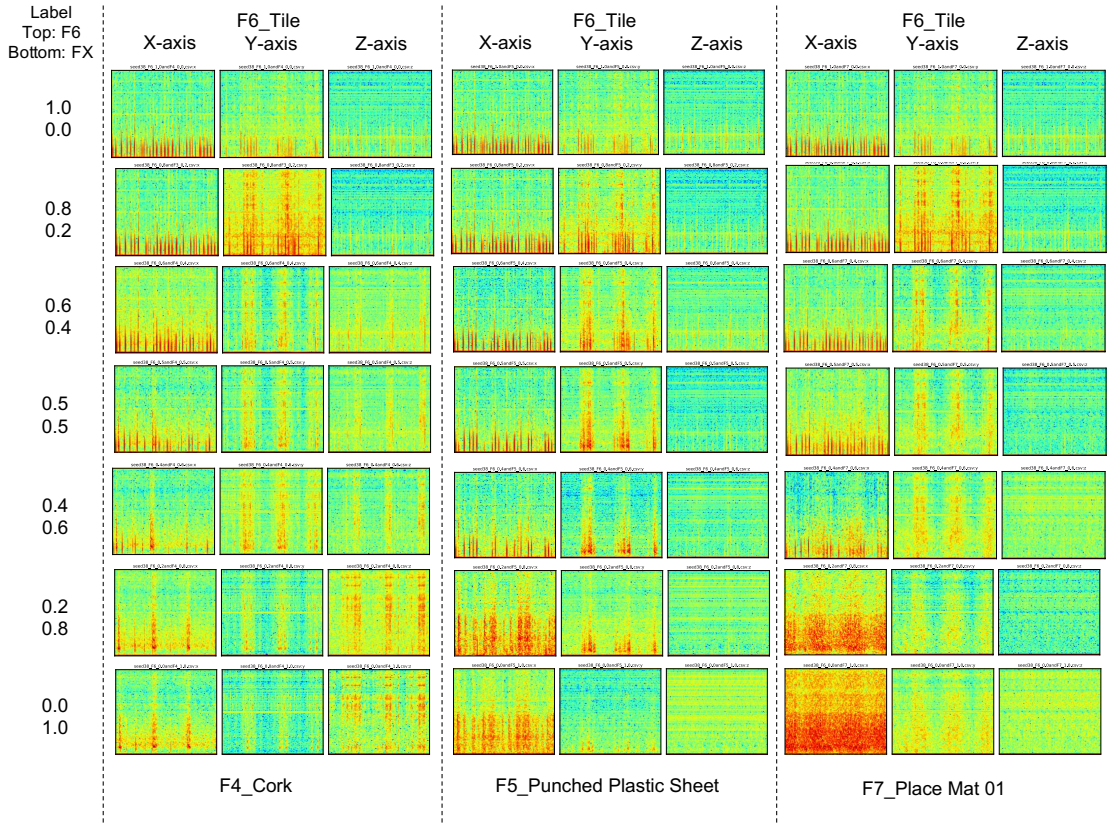

図 5.4: Tile ラベルと他のテクスチャのラベルを合成したラベルによるデータ生成結果のスペクトログラム．この図では，Tile と Cork，Tile と Punched Plastic Sheet, Tile と Place Mat 01 についての結果を示す

2 つのテクスチャの特徴を合成したデータが生成できることがわかった．また，ラベルの数値を増減させること合成の度合いで変更できることがわかった．このことを応用すれば，入力にデータ収集時の速さの数値や圧力の数値を入力し，それに応じたデータ生成を行うなど，記録のない情報の設計や生成が可能になると考えられる．今後，可変数値のラベルによりデータ生成を行った場合に生成物にどのような影響が出るのか詳しい調査を行い，その結果をもとに可変数値ラベルを前提とした新しい GAN の構築を検討する予定である．

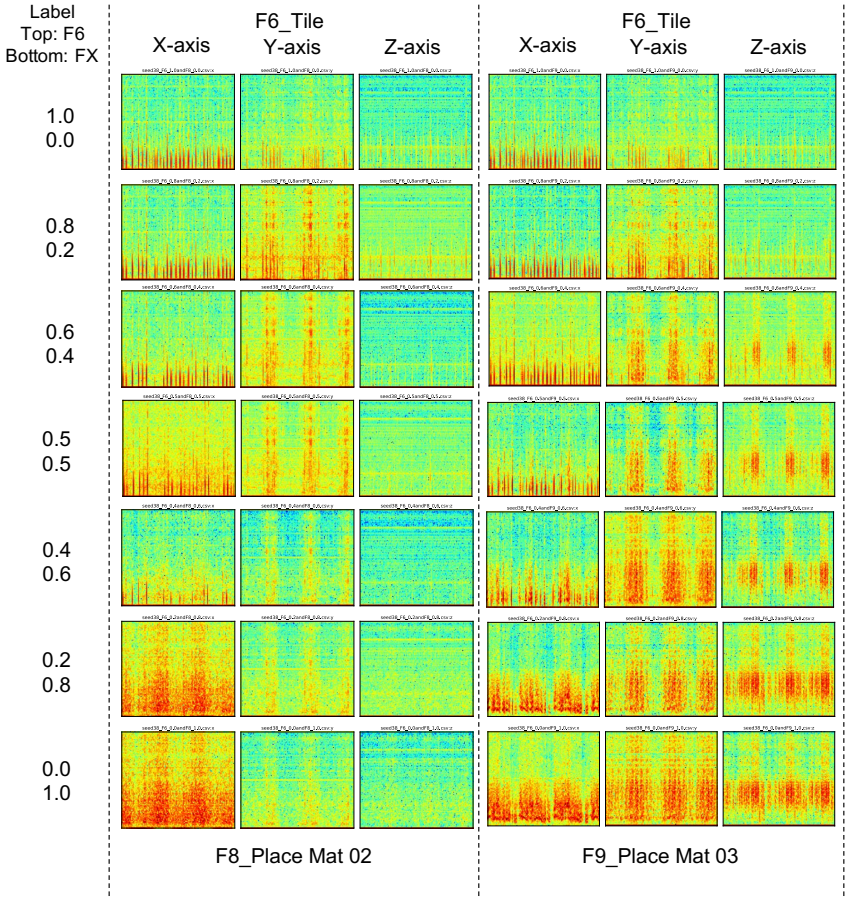

図 5.5: Tile ラベルと他のテクスチャのラベルを合成したラベルによるデータ生成結果のスペクトログラム．この図では，Tile と Place Mat 02，Tile と Place Mat 03 についての結果を示す

第6章　結論

音声生成の手法をベースとしたGANを用いることで，先行研究よりもシンプルな機械学習の構成で3軸時系列データの生成を行う機械学習モデルを作成した．GANによる振動データ生成モデル作成を汎用的なGPUを搭載したWindows PCを用いて約46時間で行うことができた．さらに，3軸のデータを生成できたことで，実際の加速度センサが取得する情報に近い情報を生成することができた．これにより，先行研究のモデルのように，単一の振動信号を用いる触覚ディスプレイのための信号としての利用にとどまらず，より多くの情報を利用する触覚ディスプレイや，触覚信号そのものの認識や解析に利用できるデータ生成が可能になった．生成データを用いた触覚提示実験により，生成データを用いることで，訓練データと同等のリアリティを持つ触覚提示が可能であることがわかった．また，データ生成時にGANモデルに入力するラベルデータを操作することで，2つのテクスチャのデータの特徴を合成した未知のデータを生成することに成功した．このデータ生成にあたり，ラベルデータの数値を操作することで，テクスチャの特徴の合成度合いを任意に変化させることにも成功した．これらの成果は，将来的に振動触覚ディスプレイでより再現性高く物体の質感をモデリングする際に必要になると考えている．

謝辞

本研究を進めるにあたり，指導教員である高橋伸准教授，志築文太郎准教授には多大なご助力をいただき，深く感謝いたします．Simona Vasilache 助教には，国際論文投稿時の英文作成の際に多くのご指摘を頂き，感謝致します．また，熊本大学の嵯峨智准教授には研究生活において数多くのご意見やご指導をいただきました．遠隔地にありながら，私の研究内容について細やかな指導をしていただきました．また，数多くの学会発表に随伴していただくなど，研究内容だけにとどまらない数多くのご支援をいただきました．深く感謝致します．先生方だけでなく，インタラクティブプログラミング研究室の皆様には研究においての様々なご支援をいただきました．特に UBIQUITOUS チームの皆様には論文投稿にあたり添削等，様々なご支援をいただきました．深く感謝申し上げます．最後に，研究生活を支えていただいた家族や友人の皆様，研究生活において出会い，お世話になった全ての方々に深く感謝致します．

参考文献

[1] Arsen Abdulali and Seokhee Jeon. Data-Driven Modeling of Anisotropic Haptic Textures: Data Segmentation and Interpolation. In *Haptics: Perception, Devices, Control, and Applications: 10th International Conference*, pp. 228–239. Springer International Publishing, 2016.

[2] Matti Strese, Yannik Boeck, and Eckehard Steinbach. Content-based Surface Material Retrieval. In *2017 IEEE World Haptics Conference (WHC)*, pp. 352–357. IEEE, 2017.

[3] Satoshi Saga and Koichiro Deguchi. Lateral-force-based 2.5-dimensional tactile display for touch screen. In *Haptics Symposium 2012*, pp. 15–22. IEEE, 2012.

[4] Youngjun Cho, Andrea Bianchi, Nicolai Marquardt, and Nadia Bianchi-Berthouze. RealPen: Providing Realism in Handwriting Tasks on Touch Surfaces using Auditory-Tactile Feedback. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*, UIST '16, pp. 195–205. ACM, 2016.

[5] Heather Culbertson, Joseph M Romano, Pablo Castillo, Max Mintz, and Katherine J Kuchenbecker. Refined Methods for Creating Realistic Haptic Virtual Textures from Tool-Mediated Contact Acceleration Data. In *2012 IEEE Haptics Symposium (HAPTICS)*, pp. 385–391. IEEE, 2012.

[6] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative Adversarial Nets. In *Advances in neural information processing systems*, pp. 2672–2680. Curran Associates, Inc., 2014.

[7] Chris Donahue, Julian McAuley, and Miller Puckette. Adversarial Audio Synthesis. *arXiv preprint arXiv:1802.04208*, 2018.

[8] Olivier Bau, Ivan Poupyrev, Ali Israr, and Chris Harrison. Teslatouch: Electrovibration for touch surfaces. In *Proceedings of the 23nd annual ACM symposium on User interface software and technology*, pp. 283–292. ACM, 2010.

[9] Eric Vezzoli, Thomas Sednaoui, Michel Amberg, Frédéric Giraud, and Betty Lemaire-Semail. Texture Rendering Strategies with a High Fidelity-Capacitive Visual-Haptic Friction Control Device. In *International Conference on Human Haptic Sensing and Touch Enabled Computer Applications*, pp. 251–260. Springer, 2016.

[10] Takayuki Iwamoto, Mari Tatezono, and Hiroyuki Shinoda. Non-contact Method for Producing Tactile Sensation Using Airborne Ultrasound. In *International Conference on Human Haptic Sensing and Touch Enabled Computer Applications*, pp. 504–513. Springer, 2008.

[11] Yoshitaka Ishihara, Ryo Shirai, Yuichi Itoh, Kazuyuki Fujita, and Takao Onoye. StickyTouch: An Adhesion Changeable Surface. In *SIGGRAPH Asia 2019 Emerging Technologies*, p. 4445. Association for Computing Machinery, 2019.

[12] Kouta Minamizawa, Yasuaki Kakehi, Masashi Nakatani, Soichiro Mihara, and Susumu Tachi. TECHTILE toolkit: A prototyping tool for designing haptic media. In *Proceedings of the 2012 Virtual Reality International Conference*, VRIC '12, p. 26. ACM, 2012.

[13] Alex Burka, Siyao Hu, Stuart Helgeson, Shweta Krishnan, Yang Gao, Lisa Anne Hendricks, Trevor Darrell, and Katherine J. Kuchenbecker. Design and Implementation of a Visuo-Haptic Data Acquisition System for Robotic Learning of Surface Properties. In *Proc. IEEE Haptics Symposium*, pp. 350–352, April 2016.

[14] Nawid Jamali and Claude Sammut. Material Classification by Tactile Sensing Using Surface Textures. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pp. 2336–2341. IEEE, 2010.

[15] 嵯峨智, 中川真史, 小野智義, 潘振雷, 張嘉袁. Zigbee を利用した日常の触覚情報収集. 電気学会研究会資料 (知覚情報研究会・力触覚提示デバイス), Vol. 2017, No. 52, pp. 11–14, 2017.

[16] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired Image-to-Image Translation Using Cycle-Consistent Adversarial Networks. In *Proceedings of the IEEE international conference on computer vision*, pp. 2223–2232, 2017.

[17] Yunjey Choi, Minje Choi, Munyoung Kim, Jung-Woo Ha, Sunghun Kim, and Jaegul Choo. Stargan: Unified Generative Adversarial Networks for Multi-Domain Image-to-Image Translation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 8789–8797. IEEE, 2018.

[18] Diederik P Kingma and Max Welling. Auto-Encoding Variational Bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[19] Mehdi Mirza and Simon Osindero. Conditional Generative Adversarial Nets. *arXiv preprint arXiv:1411.1784*, 2014.

[20] Yann LeCun, Léon Bottou, Yoshua Bengio, Patrick Haffner, et al. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, Vol. 86, No. 11, pp. 2278–2324, 1998.

[21] Takuhiro Kaneko, Hirokazu Kameoka, Nobukatsu Hojo, Yusuke Ijima, Kaoru Hiramatsu, and Kunio Kashino. Generative Adversarial Network-Based Postfilter for Statistical Parametric Speech Synthesis. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4910–4914. IEEE, 2017.

[22] Augustus Odena, Christopher Olah, and Jonathon Shlens. Conditional Image Synthesis with Auxiliary Classifier GANs. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 2642–2651. JMLR.org, 2017.

[23] Xudong Mao, Qing Li, Haoran Xie, Raymond YK Lau, Zhen Wang, and Stephen Paul Smolley. Least Squares Generative Adversarial Networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2794–2802. IEEE, 2017.

[24] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein GAN. *arXiv preprint arXiv:1701.07875*, 2017.

[25] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved Training of Wasserstein GANs. In *Advances in Neural Information Processing Systems 30*, pp. 5767–5777. Curran Associates, Inc., 2017.

[26] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks. *arXiv preprint arXiv:1511.06434*, 2015.

[27] Emily L Denton, Soumith Chintala, Rob Fergus, et al. Deep generative image models using a laplacian pyramid of adversarial networks. In *Advances in neural information processing systems*, pp. 1486–1494. Neural Information Processing Systems Foundation, 2015.

[28] Christian Ledig, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, et al. Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4681–4690. IEEE, 2017.

[29] Olof Mogren. C-RNN-GAN: Continuous Recurrent Neural Networks with Adversarial Training. *arXiv preprint arXiv:1611.09904*, 2016.

[30] Yusuke Ujitoko and Yuki Ban. Vibrotactile Signal Generation from Texture Images or Attributes using Generative Adversarial Network. In *International Conference on Human Haptic Sensing and Touch Enabled Computer Applications*, pp. 25–36. Springer, 2018.

[31] Yann LeCun and Yoshua and Bengio. Convolutional Networks for Images, Speech, and Time series. *The handbook of brain theory and neural networks*, p. 3361(10), 1995.

[32] Sepp Hochreiter and Jürgen Schmidhuber. Long Short-Term Memory. *Neural computation*, Vol. 9, No. 8, pp. 1735–1780, 1997.

[33] Mike Schuster and Kuldip K Paliwal. Bidirectional Recurrent Neural Networks. *IEEE Transactions on Signal Processing*, Vol. 45, No. 11, pp. 2673–2681, 1997.

[34] 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.

[35] Alex Graves and Jürgen Schmidhuber. Framewise Phoneme Classification with Bidirectional LSTM and Other Neural Network Architectures. *Neural networks*, Vol. 18, No. 5-6, pp. 602–610, 2005.

[36] Teng Han, David Ahlström, Xing-Dong Yang, Ahmad Byagowi, and Pourang Irani. Exploring Design Factors for Transforming Passive Vibration Signals into Smartwear Interactions. In *Proceedings of the 9th Nordic Conference on Human-Computer Interaction*, NordiCHI'16, pp. 35:1–35:10. ACM, 2016.

[37] Yang Gao, Lisa Anne Hendricks, Katherine J Kuchenbecker, and Trevor Darrell. Deep Learning for Tactile Understanding from Visual and Haptic Data. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 536–543. IEEE, 2016.

[38] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved Techniques for Training GANs. In *Advances in neural information processing systems*, pp. 2234–2242. Curran Associates, Inc., 2016.

[39] Diederik Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[40] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep Sparse Rectifier Neural Networks. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, Vol. 15 of *Proceedings of Machine Learning Research*, pp. 315–323. PMLR, 2011.

[41] Sergey Ioffe and Christian Szegedy. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. *abs/1502.03167*, 2015.

[42] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving Deep into Rectifiers: Surpassing Human-level Performance on Imagenet Classification. In *Proceedings of the IEEE international conference on computer vision*, pp. 1026–1034. IEEE, 2015.

[43] Matti Strese, Clemens Schuwerk, Albert Iepure, and Eckehard Steinbach. Multimodal Feature-Based Surface Material Classification. *IEEE transactions on haptics*, Vol. 10, No. 2, pp. 226–239, IEEE, 2016.

[44] Kathryn A Lee, Gregory Hicks, and German Nino-Murcia. Validity and Reliability of a Scale to Assess Fatigue. *Psychiatry research*, Vol. 36, No. 3, pp. 291–298, Elsevier, 1991.

著者論文リスト

本論文に関する論文

本論文の主内容は，下記の論文や発表にて公表済みである．

査読付き国際会議論文

1. Shotaro Agatsuma, Junya Kurogi, Satoshi Saga, Simona Vasilache, Shin Takahashi. Signal Generation for Vibrotactile Display by Generative Adversarial Network. In Proceedings of AsiaHaptics 2018, pp. 58-60, Springer, 2018.

査読なし国内学会論文

1. 我妻正太郎，黒木詢也，嵯峨智，高橋伸. 振動触覚ディスプレイのための DCGAN を用いた振動生成. ロボティクス・メカトロニクス講演会 2019，日本機械学会，1P2-U04，2019 年.

2. 我妻正太郎，黒木詢也，高橋伸，嵯峨智. 再帰型ニューラルネットワークを用いた GAN による触察振動生成. 第 22 回ハプティクス研究会，電気学会，pp. 39-42，2019 年.

3. 我妻正太郎，高橋伸，嵯峨智. GAN による振動触覚ディスプレイのための信号生成. 第 23 回日本バーチャルリアリティ学会大会，日本バーチャルリアリティ学会，pp.31A-6，2018 年.

国際学会招待講演

1. Shotaro Agatsuma, and Shin Takahashi, Satoshi Saga. Vibrotactile Signal Generation with GAN. In Proceedings of the 26th International Display Workshops (IDW ’19), pp. 20–23, The Society for Information Display, 2019.

その他論文

査読付き国際会議論文

1. Shotaro Agatsuma, Masafumi Nakagawa, Tomoyoshi Ono, Satoshi Saga, Shin Takahashi. Classification Method of Rubbing Haptic Information Using Convolutional Neural Network. In Proceedings of the 20th International conference on Human-Computer Interaction(HCI International 2018), pp.159–167, Springer, 2018.

2. Hirobumi Tomita, Shotaro Agatsuma, Ruiyun Wang, Shin Takahashi, Satoshi Saga, Hiroyuki Kajimoto. An Investigation of Figure Recognition with Electrostatic Tactile Display. In Proceedings of the 21th International conference on Human-Computer Interaction (HCI International 2019), pp.363–372, Springer, 2019

査読なし国内会議論文

1. 富田洋文，我妻正太郎，王瑞贇，高橋伸，嵯峨智，梶本裕之. 静電気力を用いた触覚ディスプレイによる触図認識の調査第 21 回ハプティクス研究会，日本バーチャルリアリティ学会，1A1-M06，2018 年.

2. 我妻正太郎, 中川真史, 小野智義, 嵯峨智, 高橋伸. 無線型加速度センサと畳み込みニューラルネットワークを用いた触対象の分類ロボティクス・メカトロニクス講演会 2018，日本機械学会，1P1-K14，2018 年.

3. 我妻正太郎，中川真史，小野智義，嵯峨智，高橋伸. ZigBee マイコンによる触覚情報収集と深層学習による分類手法第 18 回計測自動制御学会システムインテグレーション部門講演会，計測自動制御学会，pp. 467-471，2017 年.

4. 我妻正太郎，中川真史，小野智義，嵯峨智，高橋伸. 小型無線マイコンを用いた日常の触覚情報収集第 22 回日本バーチャルリアリティ学会大会，日本バーチャルリアリティ学会, pp.1D1-05, 2017 年.

その他発表

査読付き国際会議発表

1. Mashahiro Koga, Satoshi Saga, Shotaro Agatsuma, Junya Kurogi, Tsuyoshi Usagawa. Evaluation of Time Domain and Frequency Domain Classification Ability in Acceleration Tactile Signals. In Proceedings of the 2019 IEEE World Haptics Conference (WHC), WP1P.30, IEEE, 2019.

2. Shotaro Agatsuma, Masafumi Nakagawa, Tomoyoshi Ono, Shin Takahashi, Satoshi Saga. Daily Haptic Information Collection System Using ZigBee-based Microcomputer. In Proceedings of Eurohaptics 2018 conference: Work in Progress, 2018.

付 録A　WaveGANをベースにしたモデルによる生成データのスペクトログラム

ここでは，WaveGAN をベースにしたモデルによる生成データのスペクトログラムを示す．本文中では 9 クラスのデータを生成できるモデルを作成し，データ生成例として 3 クラス分のスペクトログラムを示した．ここでは残りの 6 クラスも含めた生成データのスペクトログラムを示す．

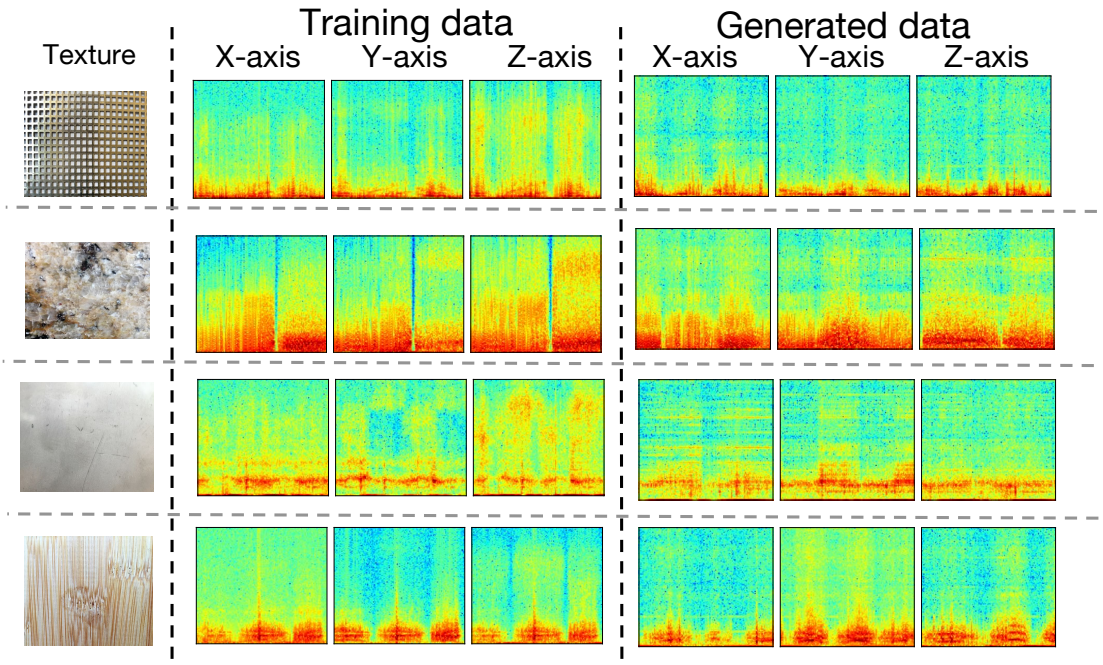

図 A.1: LMT Haptic Texture database をもとにした生成データのスペクトログラム (1)．ここでのテクスチャは本文中第 3 章の図 3.15 に示されたテクスチャと同一のものである

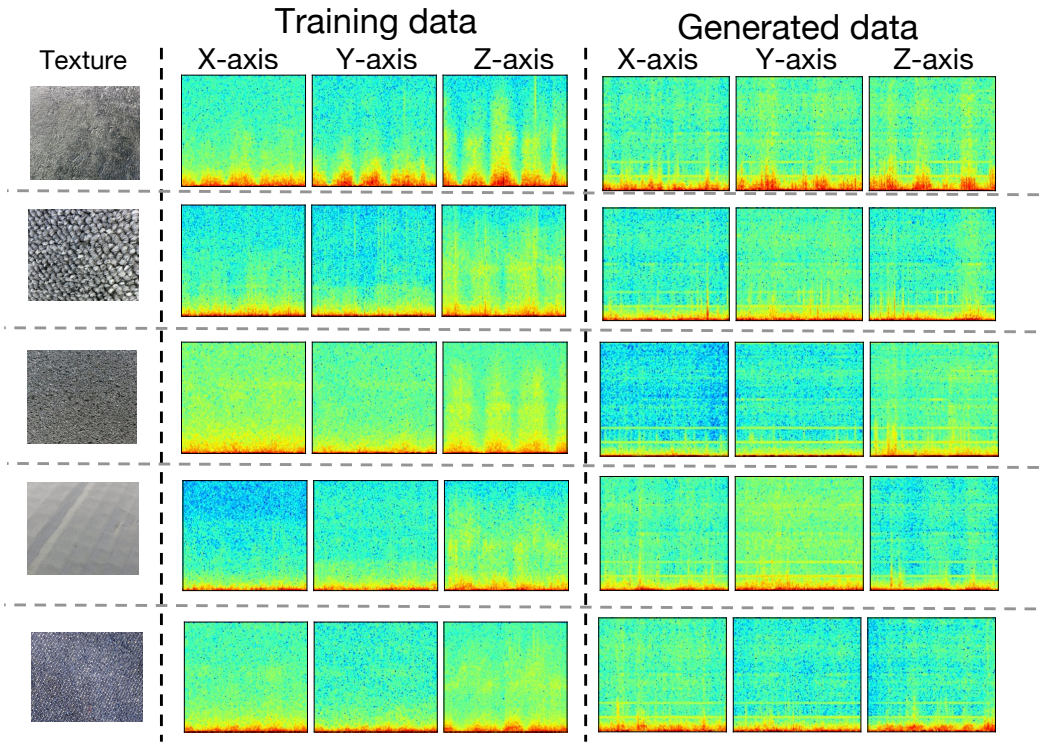

図 A.2: LMT Haptic Texture database をもとにした生成データのスペクトログラム (2)．ここでのテクスチャは本文中第 3 章の図 3.15 に示されたテクスチャと同一のものである

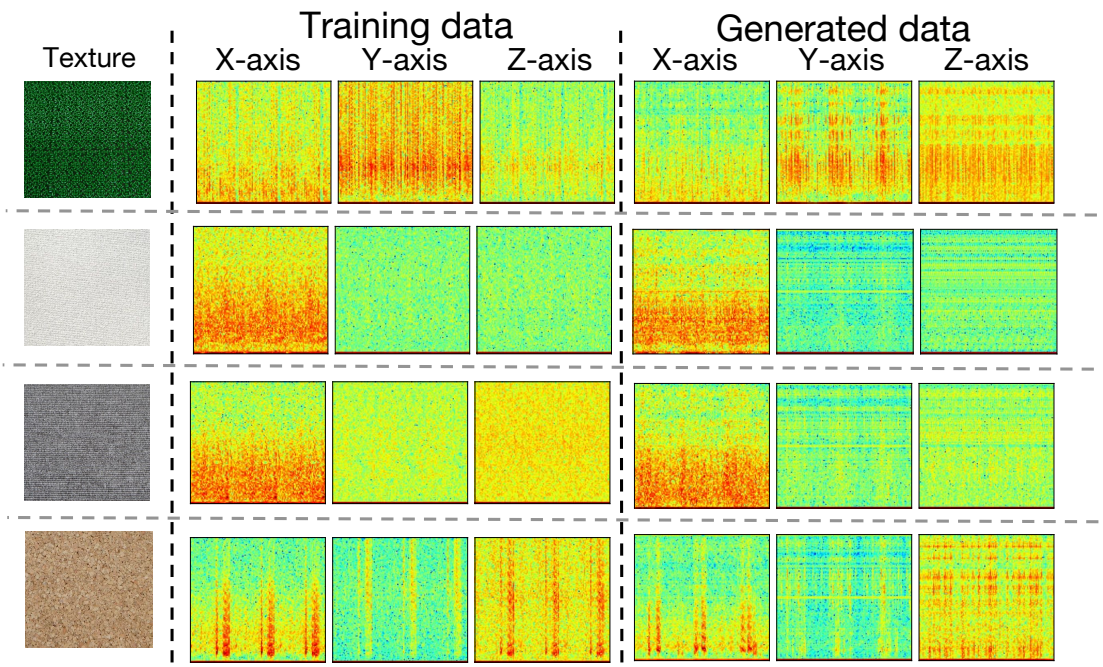

図 A.3: 収集データをもとにした生成データのスペクトログラム (1). ここでのテクスチャは本文中第 3 章の図 3.19 に示されたテクスチャと同一のものである

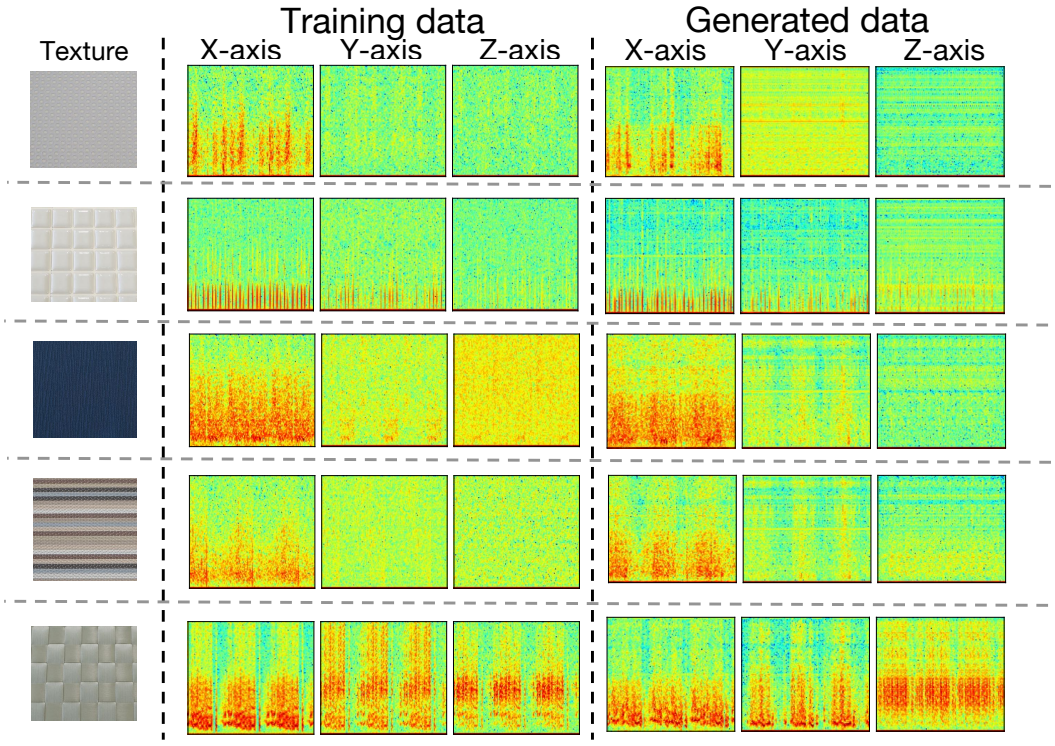

図 A.4: 収集データをもとにした生成データのスペクトログラム (2). ここでのテクスチャは本文中第 3 章の図 3.19 に示されたテクスチャと同一のものである

付 録B　第5章での触覚提示実験時用いたアンケート用紙

ここでは，第 5 章での触覚提示実験時に用いたアンケート用紙を示す.

実験時アンケート

年齢　　性別 男/女

記入日　　年　月　日

触覚提示実験について

① 実験について気になったところ，触覚提示について思ったことなど，自由に記述してください.

設問は以上です。ご協力ありがとうございました。

図 B.1: 触覚提示実験時に用いたアンケート用紙