WaveFontStyler: 音に基づくフォントスタイル変換

泉 幸太[1,a)]　柳井 啓司[1,b)]

概要

本研究では，入力した音に沿ったスタイルを文字画像に転送する手法を提案する．音と画像の特徴量を同一空間上に埋め込むモデル ImageBind とベジェ曲線による文字表現を組み合わせることで，音に沿ったスタイルを文字画像に転送する．実験により，音に沿ったスタイルを反映した文字画像の生成に成功した．さらに，音とテキストの両方を入れることで，より詳細なスタイルを指定できることも確認した．

1. はじめに

芸術的な文字は広告やポスター, ウェブページなど幅広く使われている．一方で，芸術的な文字のデザインは文字ごとに異なり，漢字やカタカナなど様々な種類の文字に対して別々にデザインを行う必要があり，多大な労力を要する．近年では深層学習を用いた画像生成技術の発展により，テキストで条件を指定することで新たな芸術的な文字の作成も容易になっている．しかし，テキストのみでは表現しきれない微妙なニュアンス等を表現したい場合もある．そういった細かなニュアンスの指定に音が利用できれば，より自分の求めるスタイルに近づく．本研究ではその第一歩として，音による入力でどのようなスタイルを表現できるかを検証する．近年では，音や画像・テキストをはじめとする 6 つのモダリティの特徴を同一空間上に埋め込む手法である ImageBind [4] が提案されている．この手法により学習された Encoder を使用して埋め込みを得ることで，ImageBind の空間上での音や画像・テキスト同士のコサイン距離を測ることができる．本研究では，この ImageBind を用いて，入力した音に沿ったスタイルを文字画像に転送する手法を提案する．文字の表現にはベジェ曲線を使用し，各曲線のパラメータを直接更新することで，文字を指定したスタイルに近づける．実験により，音に沿ったスタイルを反映した文字画像の生成が可能であることを示す．さらに，音とテキストの両方を条件とすることで，より細かなスタイルを指定できることを示す．

1 電気通信大学 大学院情報理工学研究科 情報学専攻
a) izumi-k@mm.inf.uec.ac.jp
b) yanai@cs.uec.ac.jp

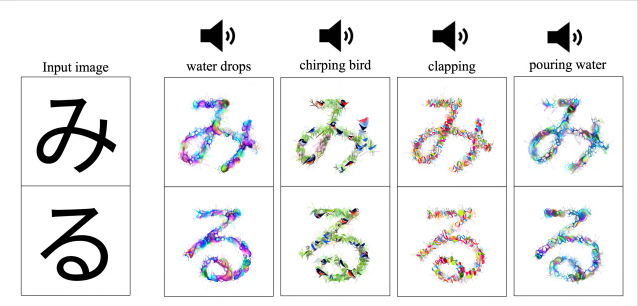

図 1: 本手法を用いて文字をスタイル変換した結果.（Adobe Reader で音声アイコンをクリックすることで実際に使用した音を聴くことが可能.）

2. 関連研究

2.1 ImageBind

近年，異なる複数のモダリティを同一空間に埋め込む ImageBind [4] という手法が提案された．この手法では，画像，テキスト，音，深度，熱，IMU データという 6 つの異なるモダリティ間で共通の埋め込みを学習する．この手法では全てのモダリティに対するペアデータではなく，画像と他のモダリティのペアデータ（画像・テキスト，画像・音など）を使用して学習を行う．ImageBind は zero-shot での音の分類・検索タスクで教師ありデータで学習したモデルと同等もしくは上回る性能を発揮している．さらに，大規模な画像生成モデルと組み合わせることで音からの画像生成も行うこともできる．

本研究ではこの ImageBind の空間における画像と音のコサイン距離を計算し，画像と音の類似度を算出する．

2.2 微分可能レンダラーを用いた生成・変換

近年，微分可能レンダラー [8] と大規模基盤モデルを組み合わせた研究が盛んに行われている．微分可能レンダラーは，ベクター画像をラスター画像にレンダリングする過程を微分可能にすることで，ラスター画像を用いて計算した損失をベクター画像を構成するプリミティブのパラメータに逆伝播することが可能となる．CLIPDraw [2] では，CLIP [10] を用いてベジェ曲線のパラメータを最適化することで，テキストのみから絵を生成している．CLIPFont [12]

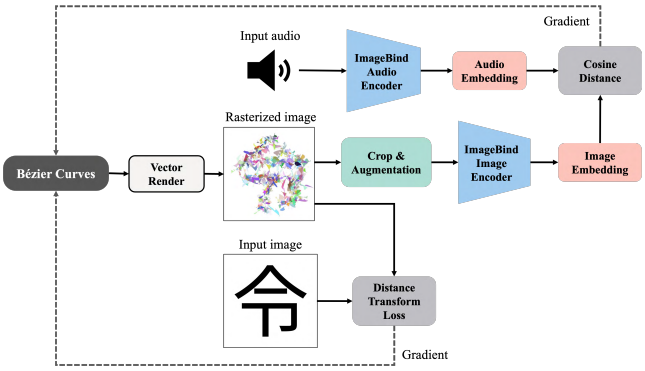

図 2: 本手法の概要図

は，ベクター形式の文字画像を入力とし，CLIP を用いてベジェ曲線で表現された文字のパラメータを最適化することで，プロンプトにマッチしたスタイルを文字に転送する．さらに Word-as-Image [5] では，学習済みの拡散モデル [11] を用いてベジェ曲線で表された文字の輪郭を最適化することで，プロンプトの意味を反映した形に変形する．また，以前我々が提案した手法 [6] では，CLIP を用いてベジェ曲線を最適化しつつ，形についての制限を加えることで，プロンプトを用いた文字のスタイル変換を実現した．

本研究ではこれらの研究を踏まえて，微分可能レンダラーと ImageBind を組み合わせ，文字の形を保ちながらベジェ曲線のパラメータを最適化することで，音によって指定されたスタイルが転送されたフォントを生成する．

3. 手法

本手法の概要図を図 2 に示す．本手法では描画をピクセルで表現されるラスター画像ではなく，パラメータとして制御点の位置, 色を持つ閉じたベジェ曲線の集合で表現し，このベジェ曲線のパラメータを勾配降下法により直接最適化する．また，ラスター形式に変換した画像から算出した損失を勾配降下法によりベジェ曲線のパラメータに伝播するために微分可能なレンダラー [8] を使用する．損失の計算では，事前学習済みの ImageBind [4] モデル用いて ImageBind 空間上におけるレンダリング画像と入力音のコサイン距離を計算する．さらに，文字の形を保つために，Atarsaikhan ら [1] がフォントスタイル変換に導入した Distrance Transform Loss も利用する．

3.1 Drawing Representation

ベクター画像は Li ら [8] の方法に基づき，閉じたベジェ曲線パスの集合で構成される．各パスは複数のセグメントで構成され，制御点の位置，塗りつぶす色，不透明度をパラメータとして持つ．これらのパラメータを勾配降下法によって最適化する．制御点の初期位置は我々が以前提案した手法 [6] と同様，図 3(b) の様に，文字画像中の黒い領域の座標からストロークの数だけランダムにサンプリングした点を各パスの始点制御点の座標とする．これは，画像中のランダムな位置を初期位置とすると，文字とは関係のない背景にまでベジェ曲線が描画されてしまうためである．制御点の初期配置は，始点制御点を決定したのち，残りの制御点を最初の制御点の周りにランダムに配置する．実験では初期配置を円形に配置する場合の結果も示す．なお，２つの初期配置による違いを図 4 に示す．

(a) (b) (c)

図 3: (a) 入力画像．(b) 各パスの始点制御点となる点．(c) 距離変換画像．文字の輪郭から離れるほど値が大きくなる．

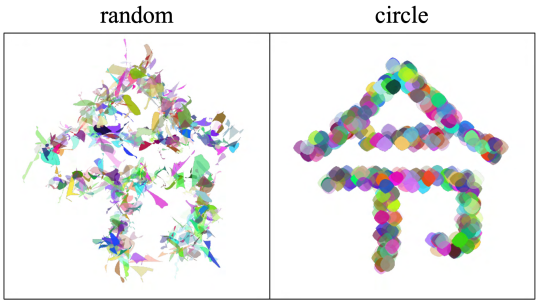

図 4: ベジェ曲線の制御点の初期配置．“random” は始点制御点の周りにランダムに配置する．“circle” では円形に配置する．

3.2 損失関数

損失関数には文字の形を保つ Distance Transform Loss と，音で指定されたスタイルを文字に転送するため Similarity Loss を使用する．

3.2.1 Distance Transform Loss

フォントにスタイルを転送するためには，文字の形を保つ必要がある．この目的を達成するために，本手法では Atarsaikhan ら [1] がロゴ部分へのスタイル変換で導入した Distance Transform Loss を追加する．Distance Transform Loss の計算には，まず入力画像に対して距離変換を行い距離変換画像を作成する．その後，入力画像と出力画像それぞれに距離変換画像を掛け合わせ，その平均二乗誤差を計算する．距離変換は図 3(c) の様に，文字部分の画素値を 0 とし，文字の輪郭から離れた画素ほど値が大きくなるようにする．Distance Transform Loss は以下の式で定義できる．

$$L_{\text{distance}} = \frac{1}{2}\left(I_c \circ I_d - I_{draw} \circ I_d\right)^2 \qquad (1)$$

ここで，I_c, I_d, I_{draw} はそれぞれ入力画像，距離変換画像，レンダリング画像を表す．演算子 $\circ$ は，同じ大きさの行列の要素同士の積を取ることを意味する．

3.2.2 Similarity Loss

音によって指定されたスタイルを文字に転送するために，レンダリング画像と入力された音の ImageBind の埋め込み空間におけるコサイン距離を計算する．具体的には，まず CLIPStyler [3] に従い，微分可能なレンダラーを用いてレンダリングした画像から N 枚のパッチをランダムに切り出す．また，CLIPDraw [2] に倣い，パッチ対して射影変換を適応する．その後，ImageBind のアプローチによって学習された Image Encoder を用いることで，パッチを 1024 次元ベクトルにエンコードする．また，入力音も同様に学習されたエンコーダーを用いて 1024 次元ベクトルにエンコードする．そして，ベクトル同士のコサイン距離を計算する．最後に，パッチごとのコサイン距離の平均をとる．従って Similarity Loss は以下のように定義できる．

$$l^i_{\text{patch}} = 1 - \frac{E_I(\text{aug}(\hat{I}^i_{draw})) \cdot E_A(A)}{|E_I(\text{aug}(\hat{I}^i_{draw}))\|E_A(A)|}$$

$$L_{\text{sim}} = \frac{1}{N}\sum_i^N l^i_{\text{patch}} \tag{2}$$

ここで $\hat{I}^i_{draw}$ はレンダリング画像から切り出された i 番目のパッチを表し，aug は射影変換を表す．E_I，E_A はそれぞれ ImageBind のアプローチで学習済みの Image Encoder, Audio Encoder を意味する．

3.2.3 Total Loss

以上の 2 つの損失関数を使用し，最終的な Total Loss は以下のように定義する．

$$L_{\text{total}} = \lambda_{\text{dist}} L_{\text{dist}} + \lambda_{\text{sim}} L_{\text{sim}} \tag{3}$$

3.3 パスの再初期化・削除

最適化の途中でパスの透明度の低下や面積の減少が起き，一部のパスが描画にほとんど寄与しなくなってしまう現象が発生する．また，文字の場合は文字の一部が薄くなってしまうことも起こってしまう．VectorFusion [7] ではこの現象に対処するため，定期的に閾値以下の不透明度または面積をもつパスを再度初期化している．本手法でもこれを採用する．

また文字の場合，スタイルとは関係のない小さく薄いパスが文字から飛び出してしまう現象がみられる．これを軽減するために，本手法ではパスの再初期化に加えて最適化の最後にも閾値以下の不透明度または面積をもつパスの削除を行う．

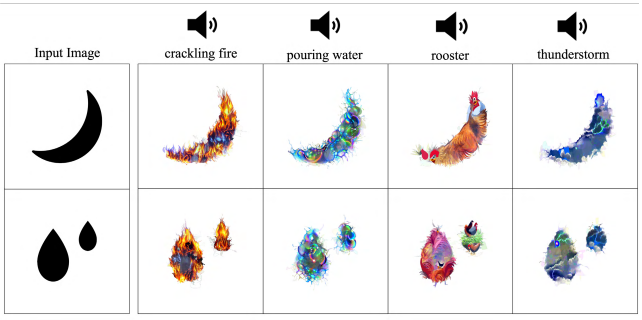

図 5: シンプルなロゴに適応した場合．

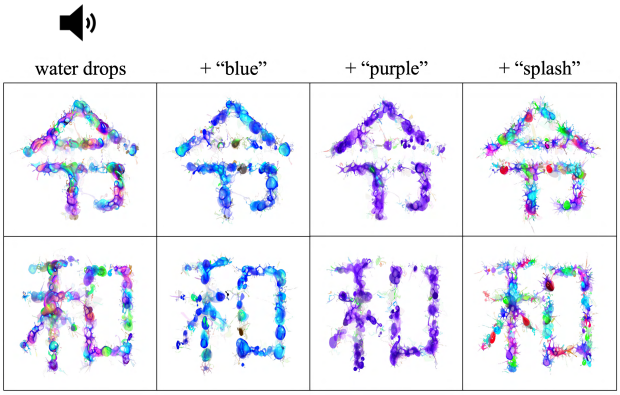

図 6: 音に対してテキスト条件も加えた結果．

4. 実験

4.1 実験設定

入力画像の解像度は 512×512 を用いた．重みについては，λ_{dist}, λ_{sim} をそれぞれ，1.0, 0.5 とした．それぞれの重みは，スタイルを反映しつつ，最も文字が認識しやすい重みに設定した．最適化関数は Adam を使用し，学習率は制御点の位置，色（不透明度を含む）に対してそれぞれ異なる値 1.0, 1.0×10^{-2} とした．イテレーション回数は 100 回，切り出す画像のサイズは 160×160 とした．また，損失関数の計算にあたってレンダリング画像から切り出すパッチの枚数は 24 枚とした．512 個のパスで 1 文字を表現する．生成時間は，NVIDIA RTX A6000 で画像 1 枚あたり 2 分程度である．なお，入力には環境音データセット ESC-50 [9] の音を使用する．

4.2 定性評価

本手法での結果を図 1 に示す．文字画像が入力した音のスタイルに変換されていることがわかる．例えば “water drop”（水の滴る音）を入力とした場合，色水を画用紙に落としたようなテクスチャを持つスタイルになっていることがわかる．“pour water”（水を注ぐ音）は光の反射のようなものが出ており，水の滴る音を入力した時とは別のスタイルとなっていることがわかる．また “chirping bird”（小

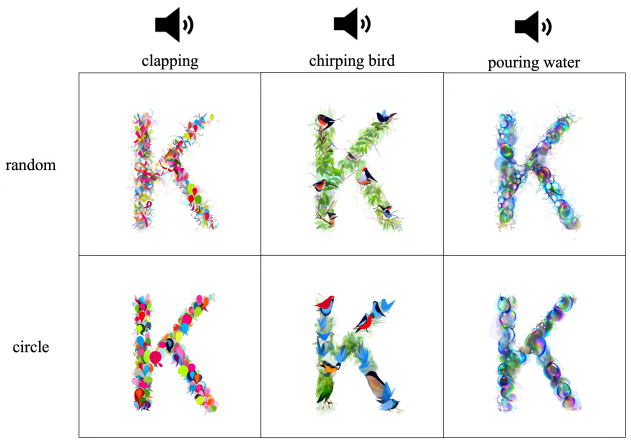

図 7: ベジェ曲線の制御点の初期配置による結果の違い．

鳥のさえずりの音）を入力した場合，鳥が木々にとまっている様子が文字に反映されている．“clapping” の場合，お祝い事のイメージを想起してか，クラッカーのようなものが現れている．また，本手法は図 5 のように白黒の画像であれば文字以外でもスタイルの転送が可能である．

4.3 アブレーション研究

4.3.1 テキスト条件を加えた場合

図 6 に音による条件に加えてテキスト条件も加えた結果も示す．テキスト条件を加える場合，従来の損失に加えてテキスト埋め込みと画像埋め込みのコサイン距離を計算する．テキスト条件を追加することで，音で指定されたスタイルを維持しつつ細かいスタイルを変更することができる．例えば，“blue” や “purple” をテキスト条件として追加した場合，“water drops”（水の滴る音）のみを入力した際の結果の色を変更したようなスタイルとなっていることがわかる．さらに，“splash” を追加すると飛沫がより多いスタイルとなっていることがわかる．

4.3.2 制御点の初期配置による結果の違い

手法で説明したように，ベジェ曲線の制御点の初期配置による結果の違いを図 7 に示す．初期配置を円形とすると，ランダムな配置とは大きく異なるスタイルになる場合がある．例えば “clapping”（拍手の音）を入力すると，ランダムな配置の場合はクラッカーのようなものが出現しているが，円形の場合は風船のようなものが現れていることがわかる．

4.3.3 クロップサイズによる結果の違い

図 8 には Similarity Loss を計算する際のクロップサイズによる違いを示す．クロップサイズを徐々に大きくしていくと，より大きなオブジェクトが出現することがわかる．例えば “sea waves”（波の音）を入力した場合，クロップサイズが 320 × 320 の時，160 × 160 では見られなかった石などが見られる．“frog”（カエルの鳴き声）の場合は，160 × 160 まではカエルの住処のようなものが現れている

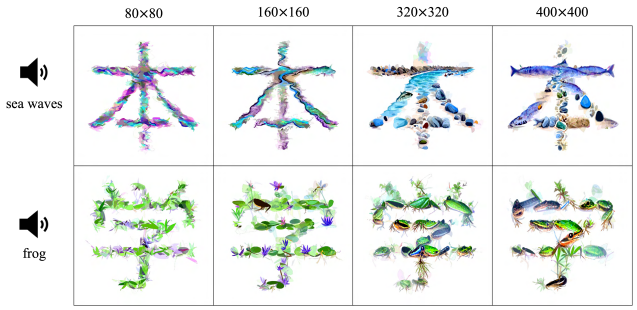

図 8: Similarity Loss を計算する際のクロップサイズによる結果の違い．なお，“sea waves” の際はベジェ曲線の制御点の配置を円形としている．

のに対して，320 × 320 以降はカエルの体の一部のようなものが出現している．

5. まとめ

本研究では，入力した音に沿ったスタイルを文字画像に転送する手法を提案した．ImageBind とベジェ曲線による文字表現を組み合わせることで，音に沿ったスタイルが文字画像に転送されることを実験により確認した．またアブレーション研究により，テキストを加えることでより詳細な条件を加えることができることや，ベジェ曲線の制御点の配置・クロップサイズの調整によって同じ音から様々なスタイルの文字を生成できることを示した．しかし，現状は同じ種類の音であっても上手くいく場合と全く音に関するスタイルにならない場合がある．今後の課題として，そういった問題に対処するために，ImageBind モデルの追加学習や損失関数の改良が挙げられる．

なお，会議当日は参加者が自由に文字やパラメータを指定し，オリジナルのフォントを作ることのできるデモを行うことを予定している．

参考文献

[1] Atarsaikhan, G., Iwana, B. K. and Uchida, S.: Contained Neural Style Transfer for Decorated Logo Generation, *In 13th IAPR International Workshop on Document Analysis Systems* (2018).

[2] Frans, K., Soros, L. and Witkowski, O.: Clipdraw: Exploring text-to-drawing synthesis through language-image encoders, *arXiv preprint arXiv:2106.14843* (2021).

[3] Gihyun, K. and Chul, Y. J.: Clipstyler: Image style transfer with a single text condition, *Proc.of IEEE Computer Vision and Pattern Recognition*, pp. 18062–18071 (2022).

[4] Girdhar, R., El-Nouby, A., Liu, Z., Singh, M., Alwala, K. V., Joulin, A. and Misra, I.: ImageBind: One Embedding Space To Bind Them All, *Proc.of IEEE Computer Vision and Pattern Recognition* (2023).

[5] Iluz, S., Vinker, Y., Hertz, A., Berio, D., Cohen-Or, D. and Shamir, A.: Word-As-Image for Semantic Typography, *ACM Trans. Graph.*, Vol. 42, No. 4 (2023).

[6] Izumi, K. and Yanai, K.: Zero-Shot Font Style Transfer

with a Differentiable Renderer, *Proc.of the 4th ACM International Conference on Multimedia in Asia*, No. 32, pp. 1–5 (2022).

[7] Jain, A., Xie, A. and Abbeel, P.: VectorFusion: Text-to-SVG by Abstracting Pixel-Based Diffusion Models, *Proc.of IEEE Computer Vision and Pattern Recognition* (2023).

[8] Li, T.-M., Lukáč, M., Gharbi, M. and Ragan-Kelley, J.: Differentiable Vector Graphics Rasterization for Editing and Learning, *ACM Transactions on Graphics (Proc. SIGGRAPH Asia)*, Vol. 39, No. 6, pp. 193:1–193:15 (2020).

[9] Piczak, K. J.: ESC: Dataset for Environmental Sound Classification, *Proceedings of the 23rd Annual ACM Conference on Multimedia*, pp. 1015–1018 (2015).

[10] Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Pamela Mishkin, J. C., Krueger, G. and Sutskever, I.: Learning transferable visual models from natural language supervision, *Proc.of the 38th International Conference on Machine Learning*, Vol. 139, pp. 8748–8763 (2021).

[11] Rombach, R., Blattmann, A., Lorenz, D., Esser, P. and Ommer, B.: High-Resolution Image Synthesis with Latent Diffusion Models, *Proc.of IEEE Computer Vision and Pattern Recognition* (2022).

[12] Song, Y. and Zhang, Y.: CLIPFont: Text Guided Vector WordArt Generation, *33rd British Machine Vision Conference* (2022).