

2022年度 修士論文

スタイル変換を用いた折り紙画像の生成

兵庫県立大学 大学院
情報科学研究科 データ計算科学専攻

IM21D043
大橋 卵香

2023年1月18日 提出
指導教員 川嶋宏彰教授

概 要

本研究では、動物や人の顔、物などの対象が撮影された実物体画像と折り紙画像を入力として、その対象物があたかも折り紙で折られたかのような画像を自動生成することを目的とする。Neural Style Transfer の研究では、実物体画像（コンテンツ）を特定の画風（スタイル）に変換する手法が多く提案されているが、造形物に特有の特徴をスタイルとした変換は、実物体画像と参照するスタイル画像とが、レイアウトを保持しながらも対応点を設定できる程度に類似させる必要がある。本研究では、折り紙特有のスタイル情報を抽出しスタイル変換に利用できる手法を開発することで、実物体画像と参照する折り紙画像の構造が大きく異なる場合でも変換可能な新たな手法の開発を目指す。

本研究における提案手法は、折り紙と非折り紙で判定可能なモデル[?]を取り入れ、判定器の結果を元にスタイル変換の損失に加えることで、生成画像に折り紙情報を持たせる。さらに本研究では手法を分割し、(1) 判定器を変換と同時に学習する、(2) 予め学習させてから利用する 2 つの異なる手法を行う。(1) では敵対的生成ネットワーク (GAN) の枠組みを利用し、生成画像と折り紙画像を学習データとしてモデルにスタイル変換と同時に学習することで、判定器が学習する折り紙画像の折り紙情報を持たせるように設計する。(2) では折り紙と判別できるネットワークをデータセットから学習し、その判別結果を損失として生成画像に加える。

実験では、スタイル変換においても折り紙で折られたかのような十分近い画像が生成可能であることを見出した。提案手法においては、(1) は折り紙でなくテクスチャとして情報が学習され、単一のデータのみでは折り紙の情報を得ることは困難であることが分かった。(2) では学習不足によりスタイル変換のみの場合と比較して改善は出来なかったが、大域的な情報だけでなく、局所的な情報の層も学習対象にすることで、より大きな変化は可能であると考える。

目次

1. 序論	1
1.1 研究背景	1
1.2 関連研究に対する本研究の位置づけ	1
1.3 本研究における貢献	3
1.4 本論文の構成	3
2. 関連研究	4
2.1 折り紙構造を持つ形状の作成	4
2.1.1 創作向けの生成に関する研究	4
2.1.2 CG 向けモデルの作成に関する研究	5
2.2 スタイル変換を用いた画像の合成	6
2.2.1 スタイル変換の原理	6
2.2.2 アルゴリズムの概要	7
2.2.3 Gatys らによる設計	8
2.2.4 Kolkin らによる設計	9
3. 提案手法	12
3.1 既存のスタイル変換に折り紙画像を用いた場合の課題	13
3.2 折り紙らしさを表現するための判定器の導入	14
3.2.1 手法 1: 判別機を生成と同時に学習する方法	15
3.2.2 手法 2: 事前学習済み判定器を用いた学習方法	18
3.3 実装方法	20
4. 評価手法と結果	21
4.1 判定器の同時学習による手法の生成結果	21
4.2 手法 2 による生成結果	22
4.3 損失関数のパラメータによる生成画像の変化	26
4.4 アブレーション	26
4.5 既存のスタイル変換との比較	28
5. 結論	30
謝辞	31

図 目 次

1	絵画を用いた変換(上段)と風景写真を用いた変換(下段). それぞれ Gatys ら [7] と Yoo ら [25] の手法を使用して作成した	2
2	TreeMaker を用いた展開図の生成例. (左: 木構造のデザイン, 中央: アルゴリズムを適用した直後, 右: 展開図と折り畳んだ図)	4
3	Origamizer を用いた生成例. ドーナツ型の形状を入力とすると, 画面右に展開図が作成される	5
4	鶴田らの手法 [27] を用いた折り紙作品の生成例. (左: 入力形状のデザイン, 右: 類似形状の検索結果)	5
5	Kato ら [14] による生成例. 折り紙画像を入力とし, 注釈によって立体モデルを生成することができる	6
6	ImageNet[4] で学習された VGG[23] の中間層の出力	7
7	Gatys ら [6] の手法を用いた, 19 層の VGG でのスタイル情報の可視化	7
8	スタイル変換の概要図. 各画像から学習済みモデルを用いて特徴マップを取り出し, コンテンツ, Style 損失を計算する. スタイル変換ではこれらの損失を最小化するように生成画像を反復的に最適化する	8
9	パレイドリア効果の例. 色や質感はそれぞれ異なるが, 星のレイアウトとしてはいずれも類似する	10
10	Gatys らの手法 [7] と Kolkin らの手法 [16] を用いて作成した画像. 各パラメータについては著者らの推奨する値を利用した	11
11	条件に該当する画像(上段)と該当しない画像(下段). 以上の画像は OrigamiSet[21] からサンプルしたものである	12
12	折り紙画像に対して, Gatys ら [7], および Kolkin らの手法 [16] を適用した場合の結果. 損失関数のパラメータについて, Gatys らの手法では $\alpha = 1.0, \beta = 1000.0$, Kolkin らの手法では $\alpha = 4.0$ としている	13
13	Kolkin らの手法での α を変化した結果	14
14	手法 1 の概要図. 1 回の学習で判定器とスタイル変換が交互に行われる	15
15	判定器のネットワーク図. Shaman らの構造 [22] とは異なり, 各レイヤにおける出力の高さと幅は入力画像と一致する	16
16	判定器の正解画像スタイル画像をそのまま利用した結果. 図 14 の判定器は画像のパッチ毎で結果を判定するため, 生成画像の前景部分にも背景の色が混入する	17

17	折り紙の部分的な前景領域の切り取り方	17
18	折り紙画像の前景から作成したテクスチャ画像	18
19	手法 2 の概要図 . 予め折り紙の 2 値分類として学習させたモデルの出力を損失に用いる	19
20	手法 1A において正解画像にテクスチャ画像を用いた結果	21
21	手法 1A を用いた生成結果 . 損失のパラメータは $(\alpha, \beta, \gamma) = (4.0, 1.0, 1.5)$ とした	23
22	手法 1B を用いた生成結果	24
23	手法 2 を用いた生成結果 . 損失のパラメータは $(\alpha, \beta) = (4.0, 1.0)$ とした	25
24	(a) の兎についての , 生成過程における画像の変化	26
25	手法 1B での損失関数のパラメータ (α, β) の変化	27
26	手法 1B における手法の変化	28
27	提案手法と既存の Neural Style Transfer との比較	29

表 目 次

1. 序論

1.1 研究背景

カメラで撮影した写真をあるアーティストの絵画の特徴に寄せた画像にする手法として、スタイル変換 (Neural Style Transfer) と呼ばれる画像変換アルゴリズムが存在する。このスタイル変換では、物体認識を問題として学習された深層ニューラルネットワークが持つ特徴表現を利用することによって、ある実物体画像の持つ「コンテンツ」を保ちつつ、別の画像の「スタイル」情報を転送することができ、2015年頃から現在に至るまで活発に研究がされている手法の1つである。

これまでのスタイル変換において対象とされてきたものは、図1のような絵画と風景写真である。絵画(図1上段)においては、ピカソやゴッホといった画家による作品について、その絵画が持つ画風(使われる色、筆触など)の情報を、風景写真(図1下段)ではその写真的持つトーンや色合いなどをスタイルと捉えて転送を行う。

一方で、スタイル変換では造形物に焦点を当てた研究はされていない。造形物におけるスタイルとはその作品に使われた材料などが挙げられるが、その材料としてのパーツをどのようにして実物体画像に当てはめていくべきか、実物体と生成画像とで対応する点を保ったまま造形物の特徴を転送させる必要がある。

この造形物の例として最も分かりやすいのが折り紙である。折り紙は日本の伝統的な遊びの1つであり、作られた形の持つその独特的な幾何学的特徴は芸術としても世界中において注目されている。折り紙作品は見立てられた物体に対してその抽象度は高く、単純な多角形を当てはめるだけでなく、どのようにして折り紙で折られたかのように見せるかが必要である。もし折り紙のような形状へ変換することができれば、造形物の一般化として、創作化へのインスピレーション付与の材料として利用できる。

1.2 関連研究に対する本研究の位置づけ

折り紙 折り紙はその独自の幾何学的性質から、数学や工学、医学など様々な領域で研究がなされている。近年では折り紙の数学的性質とコンピュータを組み合わせることによって、それまでには存在しなかった新しい折り紙の設計技法やソフトウェアなどが開発されており[26, 18]、今日における折り紙の創作技術の発展に貢献している。本研究に近い研究としては、幼児向けを対象とした簡易折り紙作品の生成や[27]や3Dモデルから1枚折り可能な展開図の生成[24]、そして平面形状から折り紙の3Dモデルの生成といった手法が存在する[14]。このような研究ではある物体の入力から特定の折り紙形状の物体を生成することがで

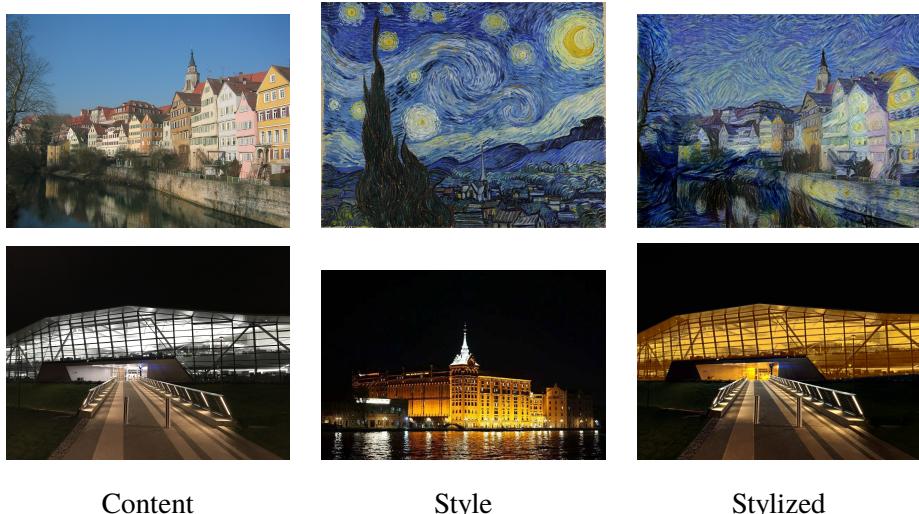


図 1: 絵画を用いた変換(上段)と風景写真を用いた変換(下段). それぞれ Gatys ら [7] と Yoo ら [25] の手法を使用して作成した.

き, 2 章でより具体的に述べることとする.

一方で, 折り紙と深層学習の組み合わせは非常に新しく, 画像生成に焦点を当てた研究は本研究が初である. Ma らは機械学習向けのデータセットとして OrigamiSet を公開しており, 実例として折り紙と非折り紙の分類や作品の難易度推定を行っている [21]. この手法は 2018 年に公開されたものであるが, 以降画像認識に関連した研究はほとんど存在しておらず, 今後発展が望まれる分野である.

スタイル変換 スタイル変換は Texture Synthesis と呼ばれる画像処理の分野から発展したものであり [5], 署み込みニューラルネットワークを用いた手法は Gatys ら [7] が 2016 年に提案した手法が初めてである. Gatys らによる手法が登場して以降は, モデルを学習することによる変換の高速化や [2, 10, 12], 写実的画像を対象とした手法 [20, 25] など, 現在多くの研究がされている.

スタイル変換は一般的には物体のスタイルのみに着目されるが, 物体の形状を含めて転送する手法も研究がされている. Kim らはコンテンツとスタイル画像の特徴点を抽出し, 特徴点同士を近づけるように変換を行う [15]. Liu らは物体の幾何的特徴を取るためのネットワークを利用して, スタイル画像の持つ幾何的形状に合わせたスタイル変換を可能としている [19].

しかし折り紙の形状となる場合, 形状の転送よりも, その画像を構成するレイアウトの保持が重要となる. 画像としての折り紙作品は, 見立てた物体がもつ顔や足のレイアウトは保持しつつも, その中に折り紙のパーツを上手く当てはめることによって折り紙で折られたかのような作品となる. したがってスタイル変換を用いる場合, 折り紙と認識できる情報を加

えること，そしてパーツの配置関係の保持をさせることによって初めて折り紙で折られたかのような画像が生み出されると考える。

そこで本研究では，実物体画像と参照となる任意の折り紙画像の構造が大きく異なる場合でもその変換できるような手法の開発を行う。折り紙がもつ「折り紙らしさ」については明確な定義は存在しないが，一般に画像認識で学習されたモデルには，学習データに使用されたあるグループの固有の情報が含まれてあり，それらはそのグループ固有の「らしさ」を有している。この考え方に基づき，本研究では折り紙と非折り紙で判定可能な判定器を利用し，そのモデルが持つ折り紙としての情報をスタイル変換における損失項として追加することで，生成した画像にあたかも折り紙で折られたかのような形状を持たせることを試みた。

1.3 本研究における貢献

本研究をまとめると，次のような貢献が存在する。

- 造形物に焦点を当てたスタイル変換にはじめて取り組んだ。
- 折り紙の持つ特徴表現として，レイアウトとしての類似を用いた変換手法が有効であることを見出した。
- 折り紙らしさを持つ判定器による損失関数を定義した

1.4 本論文の構成

本論文の構成は序論を含め全5章で構成される。2章においてはスタイル変換の技術を含めた関連研究について述べる。3章では本研究の提案手法，4章では評価手法とその結果について，5章でまとめと今後の課題を述べる。

2. 関連研究

本研究ではスタイル変換 (Neural Style Transfer) を用いた折り紙画像の生成を行う。序論でも述べたように、折り紙画像の生成についての研究は本研究が初の取り組みである。そこで本章では、まずこれまでの折り紙形状の作成に関する研究について述べ、その後スタイル変換についての原理から損失関数の設計までを述べる。

2.1 折り紙構造を持つ形状の作成

折り紙構造を持つ形状の作成は、大きく分けて創作向けと CG クリエイター向けの 2 つに分類できる。本節ではまず創作向けの研究について、その後 CG 向けとして取り組まれた研究について述べる。

2.1.1 創作向けの生成に関する研究

Lang は任意の木構造のグラフから折り畳み可能な形状を作成するアルゴリズムを開発し [17]、TreeMaker としてオンライン上で公開している [18]。このソフトウェアでは、図 2 左のような画面上でユーザが作りたい形の棒状のグラフを入力すると、その入力が持つノードやエッジの大きさを持つ折り畳み可能な基本形¹の展開図 (図 2 右) を生成することができる。

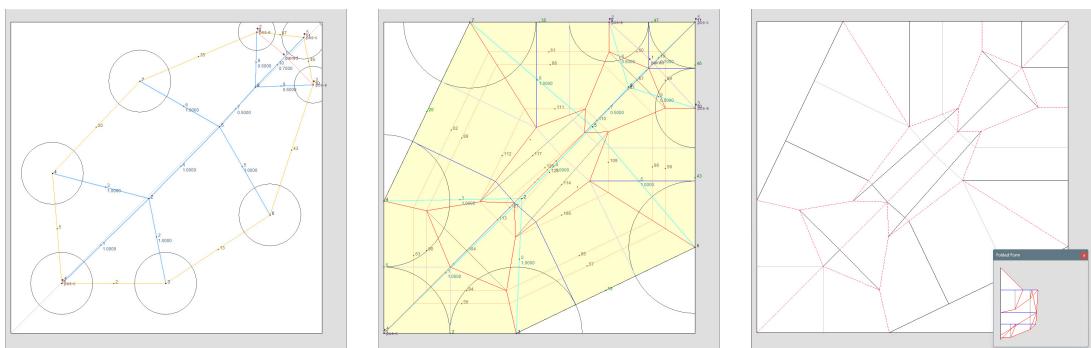


図 2: TreeMaker を用いた展開図の生成例。(左: 木構造のデザイン、中央: アルゴリズムを適用した直後、右: 展開図と折り畳んだ図)

Tachi は Origamizer と呼ばれる、任意の多面体から 1 枚の展開図を生成するソフトウェアを開発している [24]。多面体の形状を 1 枚の正方形の中に敷き詰めるために、Tachi らは Tucking 分子と呼ばれる概念を導入したアルゴリズムによって、(図 3) のように紙を切断することのない 1 枚の展開図を生成することができている。

¹ある作品を折る際に、その細部を仕上げていない段階の形状のこと。

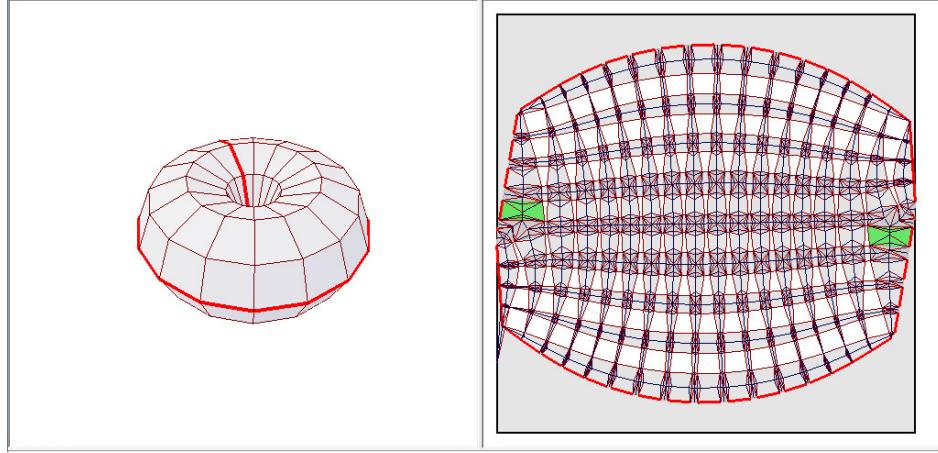


図 3: Origamizer を用いた生成例 . ドーナツ型の形状を入力とすると , 画面右に展開図が作成される .

鶴田ら [27] は 2D 形状の入力から簡単な折り紙作品を創作するためのシステムを提案している . このシステムでは 4 回以内までの折り操作を制約としており , 図 4 のようなユーザが画面上で作りたい形を入力とすると , 予め折り紙形状を格納したデータベースから最も近い形状をユーザへ提示するというシステムとなっている .

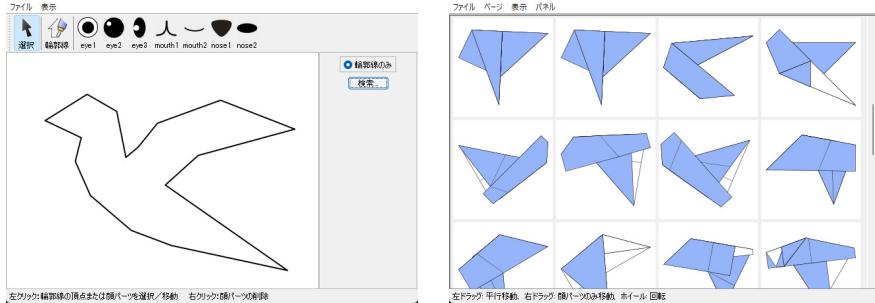


図 4: 鶴田らの手法 [27] を用いた折り紙作品の生成例 . (左: 入力形状のデザイン , 右: 類似形状の検索結果)

2.1.2 CG 向けモデルの作成に関する研究

Kato ら [14] は CG クリエイターを対象として , 平面上の折り紙画像の入力から立体的な折り紙形状をモデリングする手法を提案している . ユーザは入力画像から画面上にその形状を描き , さらに描いた物体に写るそれぞれのパーツ関係について注釈を行う . この注釈によって折り紙のもつ厚みや立体感の情報がモデルに付加され , 図 5 のような立体で折り紙で折られたかのようなモデルを生成することができる .

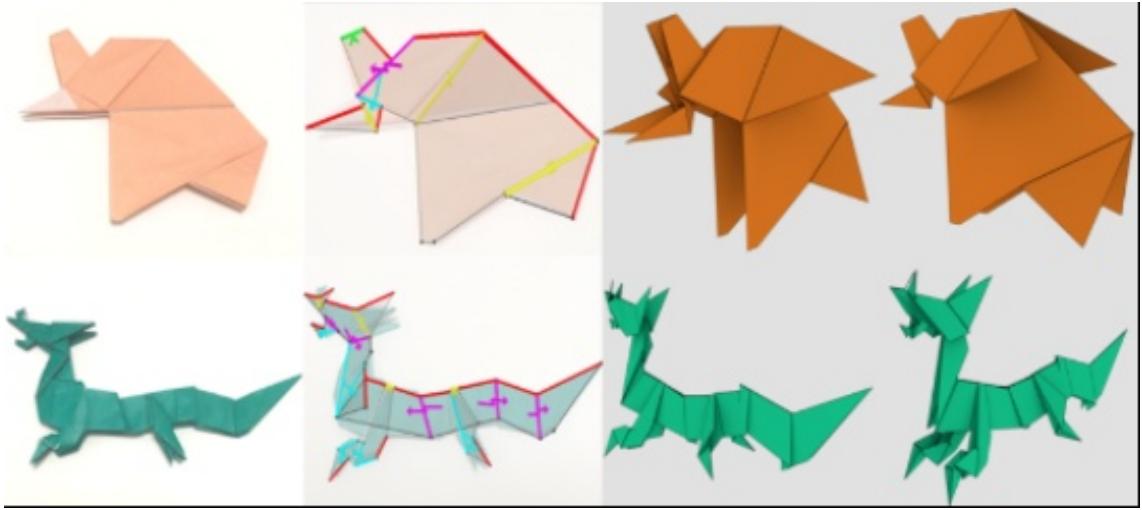


図 5: Kato ら [14] による生成例 . 折り紙画像を入力とし , 注釈によって立体モデルを生成することができる

2.2 スタイル変換を用いた画像の合成

本節では , まずスタイル変換の原理について述べる . その後アルゴリズムについて述べてから , 本研究とスタイル変換を結びづけるうえで重要となる Gatys ら , および Kolkin らによる損失関数の設計について述べる .

2.2.1 スタイル変換の原理

Gatys らによるスタイル変換の , 学習済みモデルの中間層を操作することで , 1 枚の画像から構造的情報であるコンテンツ , そして色やタッチといったスタイルの 2 つの要素に分解することができるというものである .

図 6 に入力画像に対する学習済み中間層の出力を示す . 特定の分類のために学習されたネットワークにおいて , 各中間層の畳み込みフィルタには入力に対して様々な特徴を取り出すためのフィルタが学習されている . そのフィルタは , 入力に近いほど画像の局所的な特徴が , そして出力に近くなるほど大域的な特徴を得ることができる . 言い換えれば , VGG は物体のコンテンツ情報を , 局所的な構造から大域的な構造を学習していることといえる .

スタイルとしての情報は , 図 7 のようにその特徴を統計量の観点から見ることで表現できる . 例えば入力に近い層においては局所的情報が得られることから , その統計量を取ると画像全体の色合いを情報が得られる (図 7 の block1) . 層が深くなるとより大域的情報が得られるため , その画像がどのようなタッチを持つか , 最深部では画像全体の抽象的情報といった (図 7 の block4 , block5) , 統計的に観測することで画像のスタイルとしての情報を抽出す

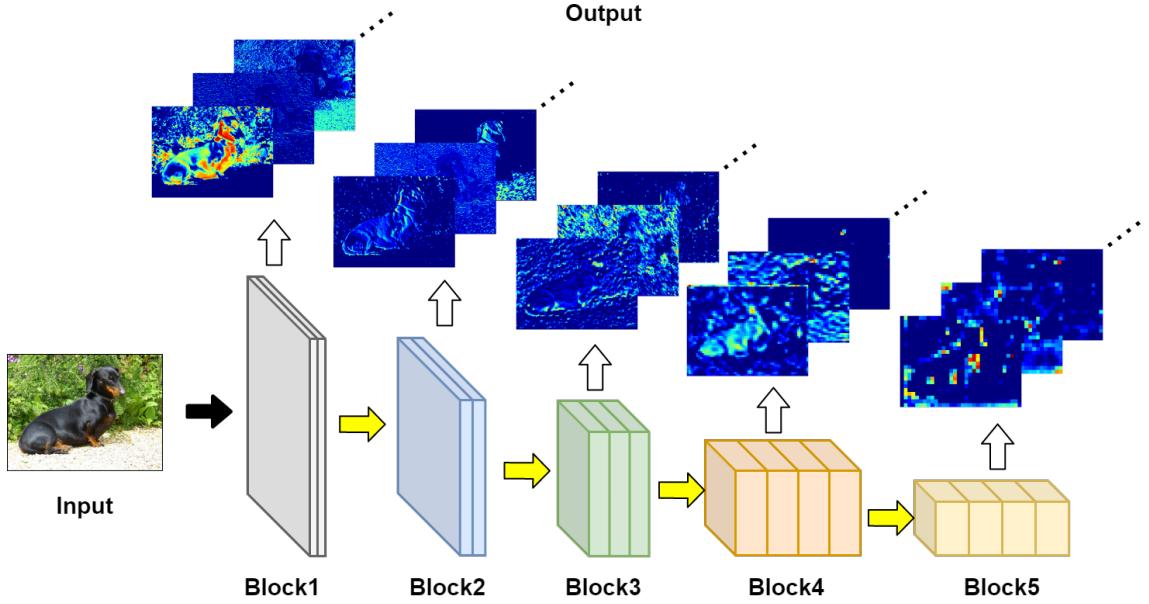


図 6: ImageNet[4] で学習された VGG[23] の中間層の出力 .

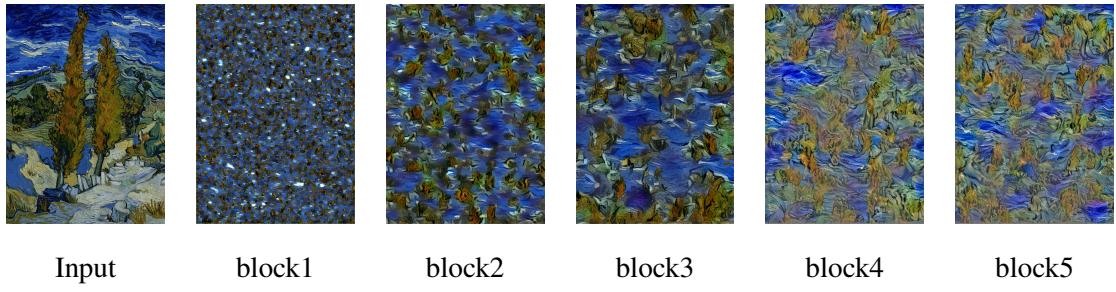


図 7: Gatys ら [6] の手法を用いた , 19 層の VGG でのスタイル情報の可視化 .

ることができる .

スタイル変換ではこのような学習済みネットワークによる画像のコンテンツとスタイル表現の分解という原理に基づいており , 学習対象をノイズとし , ベースとなるコンテンツ画像の持つ構造的な情報 , スタイル画像の持つスタイル情報の両者を持たせる画像になるように最適化することでスタイル変換を実現している .

2.2.2 アルゴリズムの概要

スタイル変換のアルゴリズムの概要を図 8 に示す . コンテンツ画像を I_c , スタイル画像を I_s , ガウシアンノイズ , あるいはコンテンツ画像等を初期値とした生成画像を I_x とする . 画像 I を入力としたとき , 学習済みモデルから取り出された第 l 層の特徴マップは $\mathcal{F}^l(I)$ で表現される . ここで $\mathcal{F}^l(I)$ は高さ H_l , 幅 W_l のピクセル数 $M_l = H_l \times W_l$, フィルタの数を

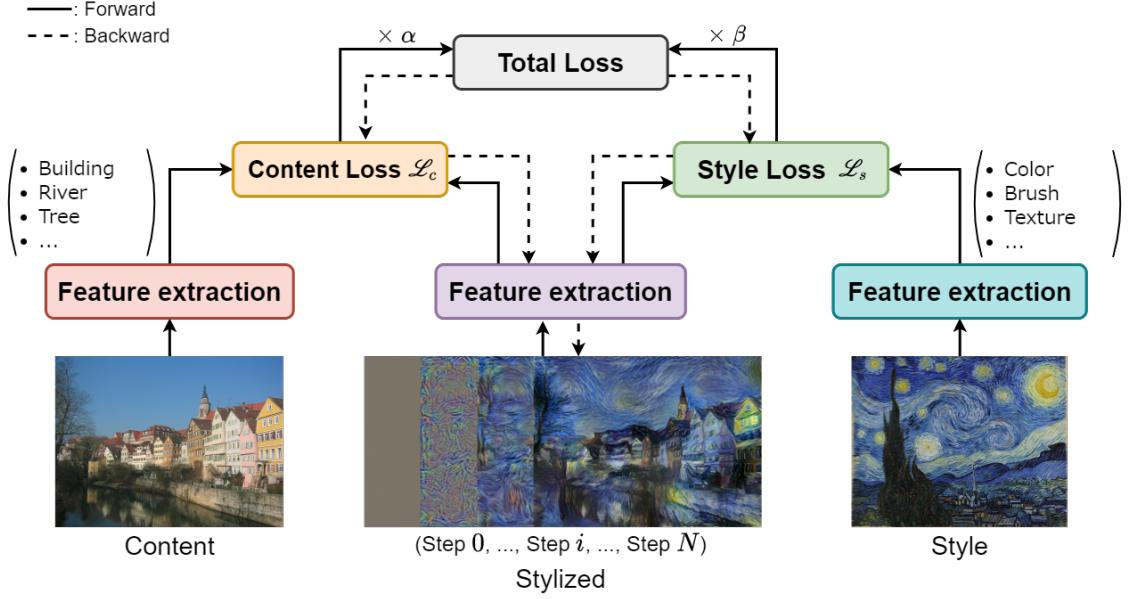


図 8: スタイル変換の概要図 . 各画像から学習済みモデルを用いて特徴マップを取り出し , コンテンツ , Style 損失を計算する . スタイル変換ではこれらの損失を最小化するように生成画像を反復的に最適化する .

N_l とした時に , $M_l \times N_l$ のベクトルを持つ .

各画像から学習済みネットワークより取り出された $\mathcal{F}^l(I)$ について , スタイル変換では次の損失関数 \mathcal{L} を最小化するように I_x を最適化する :

$$\mathcal{L} = \alpha \mathcal{L}_c(I_x, I_c) + \beta \mathcal{L}_s(I_x, I_s) \quad (1)$$

ここで \mathcal{L}_c はコンテンツ画像と生成画像の構造の類似を測る Content 損失 , \mathcal{L}_s はスタイル画像と生成画像のスタイルとしての一致性を測る Style 損失と呼ばれる . α , β はハイパーパラメータであり , これらのパラメータを調節することによって生成画像の視覚的品質を変化することができる .

スタイル変換においては , この損失関数の設計によって生成画像の視覚的品質は大きく変化する . 特に Style 損失においては , 特徴マップを統計以外にも分布や最適輸送問題として捉えるなど , 様々なアプローチからの設計がなされている [11, 13, 16] . 本研究ではその中でも , 特に Gatys ら [7] による設計 , および Koltkin ら [16] による設計について述べる .

2.2.3 Gatys らによる設計

Gatys らの Content 損失 , Style 損失の考え方は , 次のように表現される :

- Content 損失: 大域的な構造としての一致を測る

- Style 損失: スタイル情報を Gram 行列として表現し, その一致を測る

Gatys らによる Content 損失は, コンテンツ画像と出力画像のそれの中間層での出力の平均二乗誤差を取ったものとして定義される .

$$\mathcal{L}_c(I_x, I_c) = \frac{1}{M_l N_l} \sum_{i,j} \left(\mathcal{F}_{ij}^l(I_x) - \mathcal{F}_{ij}^l(I_c) \right)^2 \quad (2)$$

大域的な特徴としての一致であるため, 損失の計算には Block4 での出力が用いられている . Style 損失については, 任意の層 l における特徴ベクトルの相関を測る損失として Gram 行列 G の差を測り, それらの重み付け w による和を損失として利用している .

$$\begin{aligned} \mathcal{L}_s(I_x, I_s) &= \sum_l w_l E_l \\ E_l &= \frac{1}{4M_l^2 N_l^2} \sum_{i,j} \left(G_{ij}(\mathcal{F}^l(I_x)) - G_{ij}(\mathcal{F}^l(I_s)) \right)^2 \\ G_{ij}(\mathcal{F}^l(I)) &= \sum_k \mathcal{F}_{ik}^l(I) \mathcal{F}_{kj}^l(I) \end{aligned} \quad (3)$$

ここで Gram 行列はある中間層における特徴ベクトルの相関, すなわちそのフィルタから得られるスタイル情報の関係性を測ることを意味する . この Gram 行列の平均二乗誤差がスタイル画像と生成画像の間で小さくなるほどスタイルが類似することとなる .

2.2.4 Kolkin らによる設計

Kolkin ら手法における損失関数の考え方, 次のようにまとめられる:

- Content 損失: レイアウトとしての一致を測る
- Style 損失: 特徴の一致を最適輸送問題として捉える

Content 損失は, その物体の構成する要素が 1 つだけで決まるのではなく, それぞれの要素が相対的に関係を持つことで成立するという考え方に基づく . 例として, 図 9 に示すようなパレイドリア効果が上げられる . 人間はこれらを星の画像だと認識できるが, その理由は画像中の星のレイアウトが類似しているためである . Kolkin らはこれを Self Similarity 損失と呼んでおり, 次のような式をで損失を計算する:

$$\begin{aligned} \mathcal{L}_c(I_x, I_c) &= \frac{1}{N_l^2} \sum_{ij} \left\| \frac{D_{ij}^{I_c}}{\sum_i D_{ij}^{I_c}} - \frac{D_{ij}^{I_x}}{\sum_i D_{ij}^{I_x}} \right\| \\ D_{ij}^I &= 1 - \sum_k \frac{\mathcal{H}_{ik}(I) \mathcal{H}_{kj}(I)}{\|\mathcal{H}_{ik}(I)\| \|\mathcal{H}_{kj}(I)\|} \end{aligned} \quad (4)$$

ここで \mathcal{H} は各層における特徴マップをチャネルに沿って並べたハイパーカラム, D はコサイン距離行列である .



図 9: パレイドリア効果の例 . 色や質感はそれぞれ異なるが , 星のレイアウトとしてはいずれも類似する .

Style 損失においては , Kolkin らは Relaxed Earth Mover's Distance (REMD) , Moment Matching , Color Match の 3 つの損失関数を利用している .

Earth Mover's Distance (EMD) は統計的な分布間の距離を測る距離尺度の 1 つである . REMD は EMD の計算にかかるコストを抑えたものであり , Kolkin らはスタイルの特徴を分布として捉えることでこの損失を用いている . 特徴分布間の距離を C とした場合 , REMD の損失は次のように定義される .

$$\begin{aligned} \mathcal{L}_r(I_x, I_s) &= \max \left(\frac{1}{N_l} \sum_i \min_j C_{ij}, \frac{1}{M_l} \sum_j \min_i C_{ij}, \right) \\ C_{ij} &= 1 - \sum_k \frac{\mathcal{H}_{ik}^l(I_x) \mathcal{H}_{kj}^l(I_s)}{\|\mathcal{H}_{ik}^l(I_x)\| \|\mathcal{H}_{kj}^l(I_s)\|} \end{aligned} \quad (5)$$

Moment Matching については式 (5) で発生するアーチファクトを抑えるために , 特徴マップの統計的な一致を測る損失として次のように定義される .

$$\mathcal{L}_m(I_x, I_s) = \frac{1}{d} \left\| \mu_{\mathcal{H}^l(I_x)} - \mu_{\mathcal{H}^l(I_s)} \right\| + \frac{1}{d^2} \left\| \Sigma_{\mathcal{H}^l(I_x)} - \Sigma_{\mathcal{H}^l(I_s)} \right\| \quad (6)$$

ここで μ は平均 , Σ は分散である .

最後に , スタイル画像を構成する色のパレットを合わせるために Color Match 損失 \mathcal{L}_p が追加される . 式は式 (5) と同じだが , C にユーリッド距離を追加したものが利用されている .

上記のような Content 損失 , Style 損失で Kolkin らの損失は設計されている . また損失の式においても式 (1) のような構成と異なり , 次のような式を最小化する:

$$\mathcal{L} = \frac{\alpha \mathcal{L}_c(I_x, I_c) + \mathcal{L}_s(I_x, I_s) + \mathcal{L}_m(I_x, I_s) + \frac{1}{\alpha} \mathcal{L}_p(I_x, I_s)}{\alpha + 2 + \frac{1}{\alpha}} \quad (7)$$

Gatys らの手法と Kolkin らの手法を用いた例を図 10 に示す . Gatys らの手法は 1 行目のような部分的にしかスタイル存在しない画像に対しては失敗しているが , Kolkin らの画像ではレイアウトを残したまま , スタイル画像の情報も上手く転送できている .

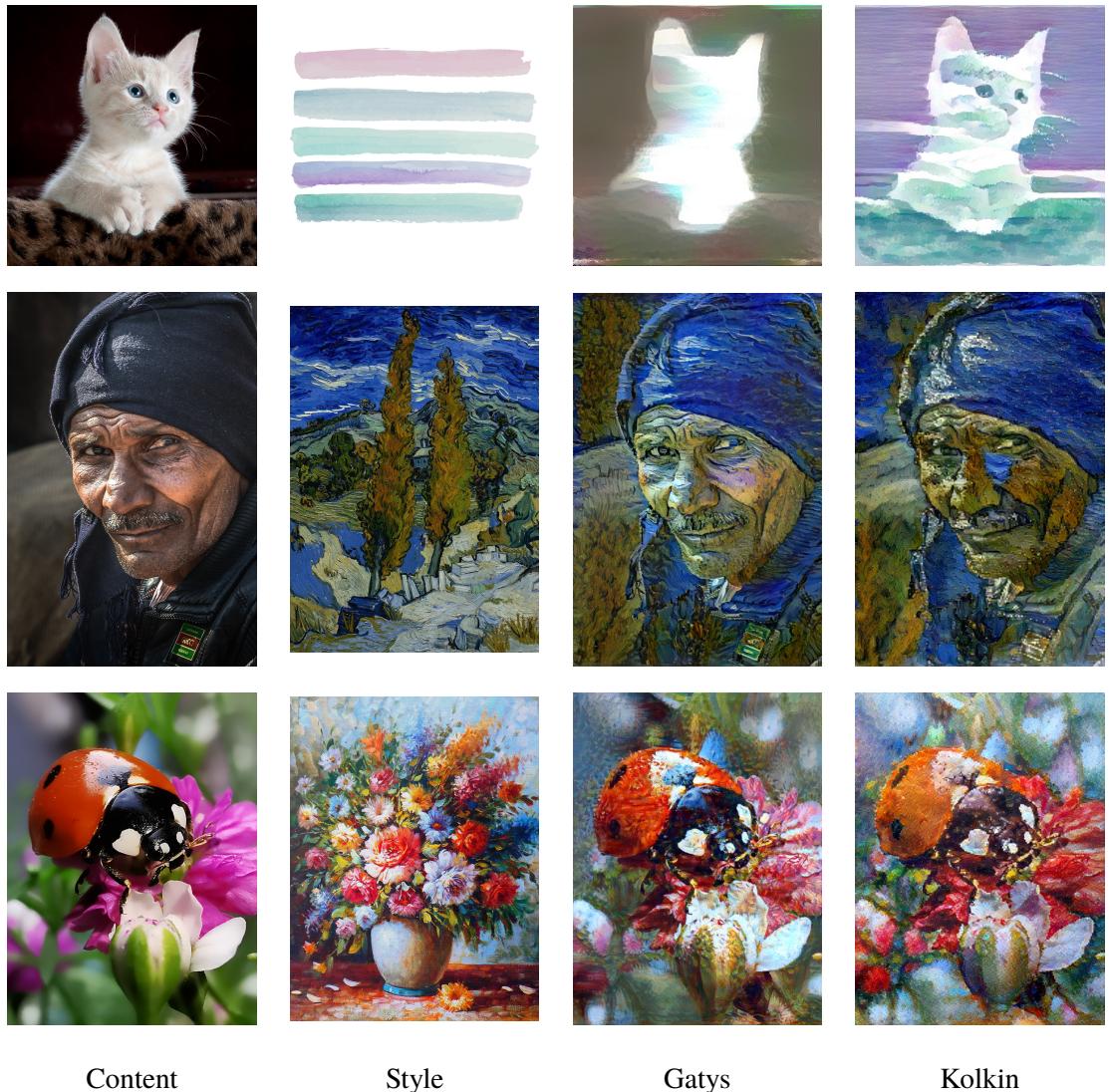


図 10: Gatys らの手法 [7] と Kolkin らの手法 [16] を用いて作成した画像 . 各パラメータについて著者らの推奨する値を利用した .

3. 提案手法

本研究では折り紙画像を用いて，動物などの実物体画像をからスタイル変換 (Neural Style Transfer) を用いて折り紙らしさを持った画像の生成を行う．

折り紙は1枚の紙で折られたもの以外にも様々な種類の作品が存在し，またその作品で用いられた紙や形状そのものによって複雑さやその特徴は大幅に変化する．そのため折り紙画像の生成として，その画像全てを包括するようなスタイルの転送は非常に困難である．そこで本研究においては，以下の条件を持つ折り紙画像を使用することを想定する．

1. 画像中に写る作品は1つである
2. 作品は画像外にはみ出でていない
3. 作品の持つ紙の色は2色以下
4. その折り紙作品が見立てたものが「生物」である
5. 形状が複雑かつ精密に折られたような作品ではない

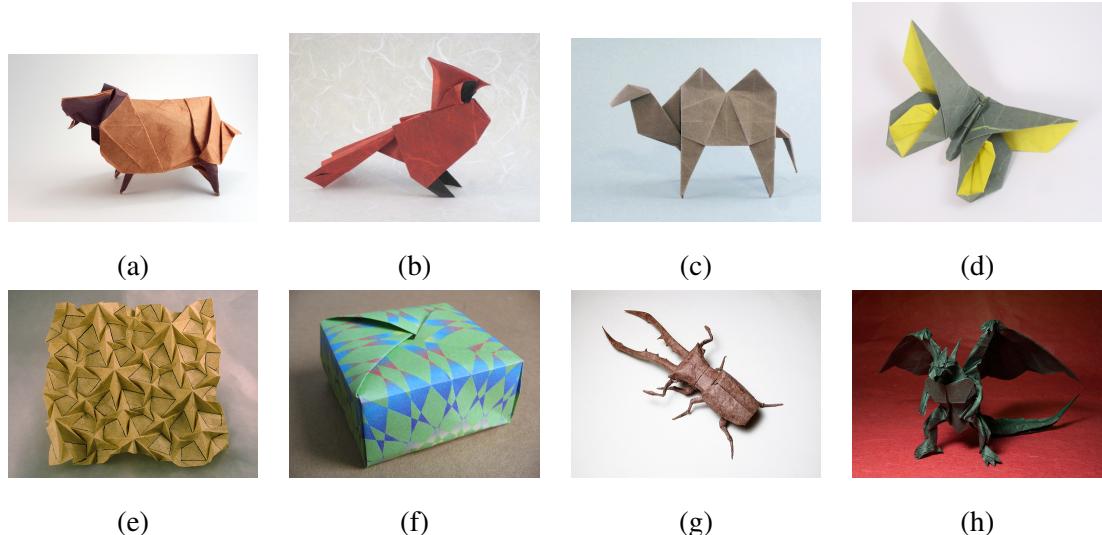


図 11: 条件に該当する画像 (上段) と該当しない画像 (下段)．以上の画像は OrigamiSet[21] からサンプルしたものである．

上記条件の対象外となる画像については図 11 の下段に示すとおりである．(e), (f) は条件 4 に当たるまらず，また (f) は条件 3 に反する．(g), (h) は実際に1枚で折られてはいるが，その形が非常に複雑であるため本研究では除外する．また本研究においては物体のみを考慮するため，コンテンツおよびスタイル画像の背景は白にしたものを利用する．

本章では提案手法に先駆け，まず2章で言及した既存のスタイル変換における折り紙画像を用いた場合の課題の整理を行う．次に提案手法として折り紙らしさの判定器を用いたアプ

ローチについて説明し，最後に実装方法について述べる．以下断りがない限り，本章以降で示す折り紙画像は全て OrigamiSet[21] における画像を使用する．

3.1 既存のスタイル変換に折り紙画像を用いた場合の課題

提案手法に先立ち，まず実際に既存のスタイル変換において折り紙画像をスタイルに用いた場合について事前実験を行う．実験手法としては，2.2 節において取り上げた Gatys ら，および Kolkin らの手法を使用し，折り紙画像をスタイルとして適用を行う．前景の物体のみを評価するため，各手法において領域変換を用いた場合²の手法を採用する．



図 12: 折り紙画像に対して，Gatys ら [7]，および Kolkin らの手法 [16] を適用した場合の結果．損失関数のパラメータについて，Gatys らの手法では $\alpha = 1.0, \beta = 1000.0$ ，Kolkin らの手法では $\alpha = 4.0$ としている．

実際に適用した結果を図 12 に示す．Gatys らの手法(図 12 中央)においては，折り紙画像の持つ質感が転送上手くできておらず，また幾何学的な形状への変化されていない．Kolkin らの手法においては，Gatys らの手法と対比して紙の質感が良く転送できており，図中のカバの画像のように折り紙で折られたかのように非常に近い結果を生むことができる．

折り紙画像に対する Kolkin の手法の利点としては，その損失の設計にある．まず Content 損失においては，2.2.4 節でも述べたように物体のレイアウト保持とした設計となっており，

²Gatys らについては後の文献 [8] で提案された手法を利用した．Kolkin らの手法は文献中にガイド付きの変換として手法が提案されている．

これは物体の形状を大幅に変化させることを許容できる。一方の Style 損失では、REMD、及び Moment Matching によって、Gram 単体では捉えられなかった折り紙画像の持つ特徴を捉えてできている。

しかし、Kolkin らの手法においても折り紙らしくない要素が存在する。例えば、図 12 下段においては画像中のバーツの境界が不揃いであること、そしてコンテンツ画像が持っていた奥行が生成画像では失われてしまっている。この要素は図 13 に示すようなハイパーパラメータの調整でも改善はされない。

この原因としては、特徴抽出として用いられるモデルのみでは折り紙の特徴が捉えきれないことが挙げられる。スタイル変換で用いられた VGG では、ImageNet で学習されたものが使われるが [23]、このデータセットには折り紙としてのラベルを持った学習がなされていない。すなわち、折り紙画像の持つ「折り紙としての情報」が用いられている特徴抽出器では捉えることができておらず、生成画像において折り紙で折られたかのように見えない画像が生成されてしまう。

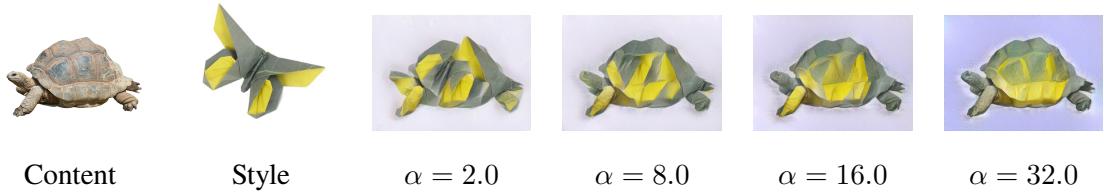


図 13: Kolkin らの手法での α を変化した結果。

3.2 折り紙らしさを表現するための判定器の導入

3.1 節でも述べたように、Gatys ら、Kolkin らの手法においては折り紙画像の折り紙としての特徴は特徴抽出器のみでは捉えることができない。提案手法においては、この特徴を生成画像で捉えるようにするために、折り紙らしさを表現するための判定器を用いてその出力を損失に加えることによる、折り紙らしさの表現の付加を行う。

スタイル変換に判定器を導入させる手法は大きく 2 通り存在する。1 つ目は敵対的生成ネットワーク (GAN) と呼ばれるアルゴリズムを応用し、スタイル変換と折り紙の有無を判定するの学習を同時にする手法、2 つ目は事前に折り紙として分類可能なモデルを学習させたモデルを用いた手法が存在する。本節では、GAN の枠組みを用いた手法を手法 1、事前学習させたモデル用いた手法を手法 2 として順に述べる。

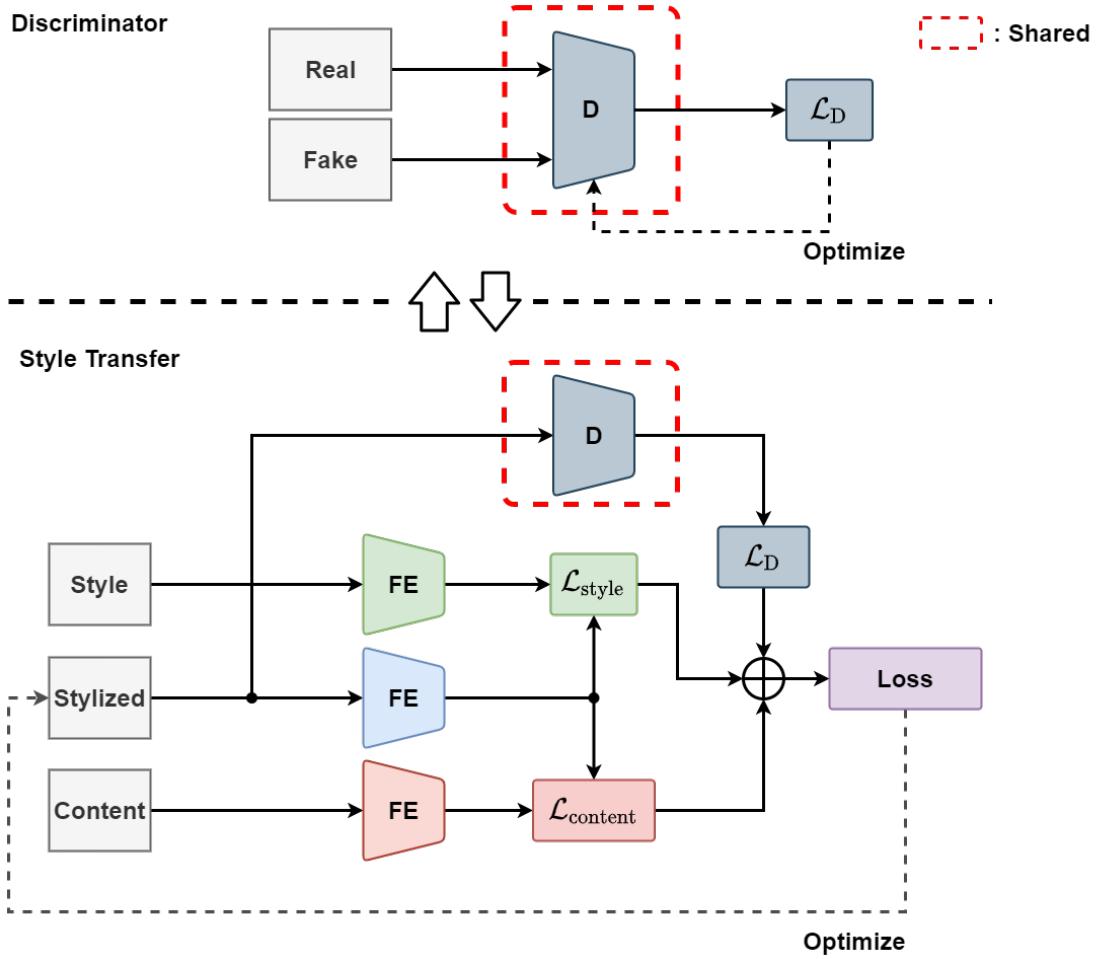


図 14: 手法 1 の概要図 . 1 回の学習で判定器とスタイル変換が交互に行われる .

3.2.1 手法 1: 判別機を生成と同時に学習する方法

提案手法の概要を図 14 に示す . この生成手法においては , 異なる画像ドメイン間の変換などのタスクで使われる GAN の枠組みを利用する .

GAN の学習においては , 画像の生成を行う生成器 , 生成画像と正解画像を識別する判定器の 2 つのモデルが存在する . この 2 つモデルでは , 正しく学習させるためには膨大なデータセットを用いて学習することが前提となっており , 通常は判定器モデルを 1 枚の画像のみで判定器を学習する事は困難である . しかし , Shaman らの手法 [22] においては , 画像をパッチに分割して判定器を判別させることで , 1 枚の画像のみで GAN の訓練をすることができる . 本手法においても , Shaman らの枠組みを利用し , 判定器のモデルを図 15 のような構造として用いる .

判定器の学習は , スタイル画像を Real , スタイル変換で得られた生成画像を Fake として入力し , 判定器の損失を計算することでモデルの重みを更新する . 判定器の損失は WGAN-GP[9]

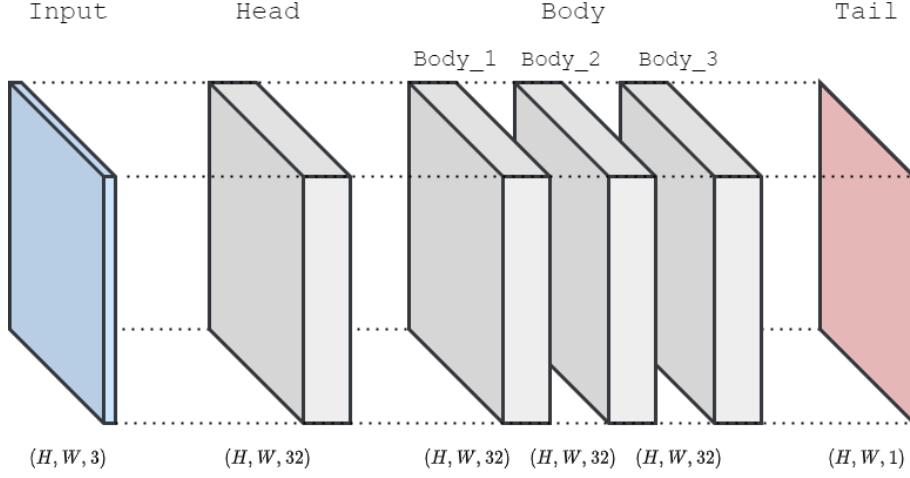


図 15: 判定器のネットワーク図 . Shaman らの構造 [22] とは異なり , 各レイヤにおける出力の高さと幅は入力画像と一致する .

の損失を使用しており , 判定器を \mathcal{D} , 期待値を \mathbb{E} としたとき , 次の式で表される:

$$\begin{aligned} \mathcal{L}_D(I_x, I_s) &= \mathbb{E}_{I_x \sim \mathbb{P}_g}[\mathcal{D}(I_x)] - \mathbb{E}_{I_s \sim \mathbb{P}_s}[\mathcal{D}(I_s)] + \lambda \mathbb{E}_{\tilde{I}_x \sim \tilde{\mathbb{P}}_g}[(\nabla_{\tilde{I}_x} \mathcal{D}(\tilde{I}_x) - 1)^2] \\ &\quad \tilde{I}_x = \epsilon I_s + (1 - \epsilon) I_x \end{aligned} \quad (8)$$

ここで λ はハイパーパラメータであり , ϵ は区間 $[0, 1]$ の一様分布から取り出されるランダムな値である .

スタイル変換では 3.1 節において折り紙に近い画像に変換ができる Kolkin らの手法を利用する . 損失については , 判定器とのバランスを取るために , 式 (7) に対して , 本研究では次のような式を利用する .

$$\mathcal{L} = \alpha \mathcal{L}_c(I_x, I_c) + \beta \mathcal{L}_s(I_x, I_s) + \gamma \mathcal{L}_D(I_x) \quad (9)$$

ここで \mathcal{L}_s , \mathcal{L}_c , α は Kolkin らと同様である . β および γ は追加のハイパーパラメータであり , \mathcal{L}_D は式 (8) においての第 1 項および第 3 項は 0 とした式である .

判定器に使用する正解画像の生成 図 14 の判定器では , 本章初めに定めた折り紙画像をそのまま判定器に用いてしまうと , 図 16 のように背景色が生成画像の前景に混ざる . この理由としては判定器が画像全体を学習することで , 背景領域と前景領域の区別なく判別するためであり , 折り紙画像中に写る折り紙作品だけを学習することができないためである .

そこで正解画像においては , 予めスタイルとして用いる折り紙画像を加工したものを使用する . この加工の種類として 2 つ存在し , 1 つ目が前景領域で Crop する方法 , 2 つ目が折り



図 16: 判定器の正解画像スタイル画像をそのまま利用した結果 . 図 14 の判定器は画像のパッチ毎で結果を判定するため , 生成画像の前景部分にも背景の色が混入する .

紙画像の前景領域からテクスチャのような画像を生成し利用する方法である .

前景領域の Crop では , 折り紙画像中から前景領域が多く含まれるように画像を切抜き , それらを正解画像のデータとして利用する . アルゴリズムの概要を図 17 に示す . なるべく Crop された領域における面積を示すために , Mask 画像に距離変換を施す . この距離変換された画像のピクセル値について n 分割に値の量子化を行い , 任意の閾値以上の領域におけるピクセルの座標を中心点の集合とする . それらの集合からランダムに選択した座を中心とし , 任意のサイズで切り取ることによって , 図中右上のように前景領域を占めるようになる . この Crop を 1 ステップごとに切り替えることで , 折り紙画像の折り紙である領域を判定器に学習させるようとする .

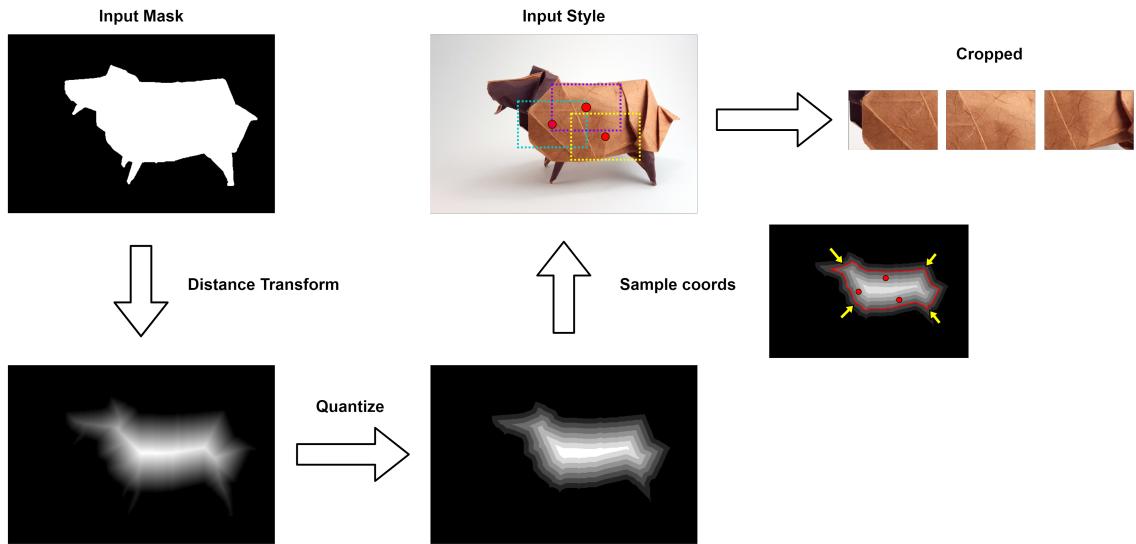


図 17: 折り紙の部分的な前景領域の切り取り方 .

もう 1 つのテクスチャ生成では , あらかじめスタイル画像の前景領域の折り紙の部分から画像を作り , それを利用する手法である . この手法の折り紙画像の持つ特徴が前景にあるの

であれば、そこから得られるテクスチャ画像からも同様の折り紙の特徴的な情報が得られるという考え方に基づく。テクスチャの生成については、Kolkin らの手法において I_x をノイズとしたときに、損失関数をスタイル画像の前景領域と生成画像全体の Style 損失のみとすることによって図 18 のように全体が折り紙画像になるように生成を行い、その画像を判定器の正解画像として用いる。



図 18: 折り紙画像の前景から作成したテクスチャ画像。

3.2.2 手法 2: 事前学習済み判定器を用いた学習方法

3.2.1 節では判定器を同時に学習させることで、スタイル画像の持つ特徴を転送しするようを行う。一方で、折り紙らしさを反映させることは、事前に折り紙のデータセットを用いて学習させたモデルを用いても可能であると考え、図 19 のように折り紙の分類モデルを事前に作成し、その結果を損失に組み込むこととした。

分類モデルについてはベースを Imagenet で学習された VGG16 として [23]、中間層の *block5* 以降を Fine-tuning することで学習した。使用したデータセットについては、折り紙画像を OrigamiSet[21] において 3 章の初めて述べた制約を満たす画像を使用し、非折り紙では動物画像を対象とした Kaggle の Animal Image Dataset³を利用する。本研究では背景は考慮しないため、学習前の前処理として Rembg⁴を用いて背景を白色化させたものをデータセットとして使用した。

上記のデータセットでは、折り紙と非折り紙の画像を使って学習される。しかし、スタイル変換で生成された画像は自然画像ではないため、誤判定を防ぐために、Kolkin らの手法で作成した幾つかの画像についても Fine-tuning の学習データに使用する。

最終的に Fine-tuning に用いたデータの総数は 4224 枚であり、このデータについて学習データと検証データ、テストデータで 6 : 2 : 2 で分割を行う。学習対象となる層は全てランダムノイズで初期化し、Optimizer として Adam、損失関数を Binary Cross Entropy、評価指

³<https://www.kaggle.com/datasets/iamsouravbanerjee/animal-image-dataset-90-different-animals>

⁴<https://github.com/danielgatis/rembg>

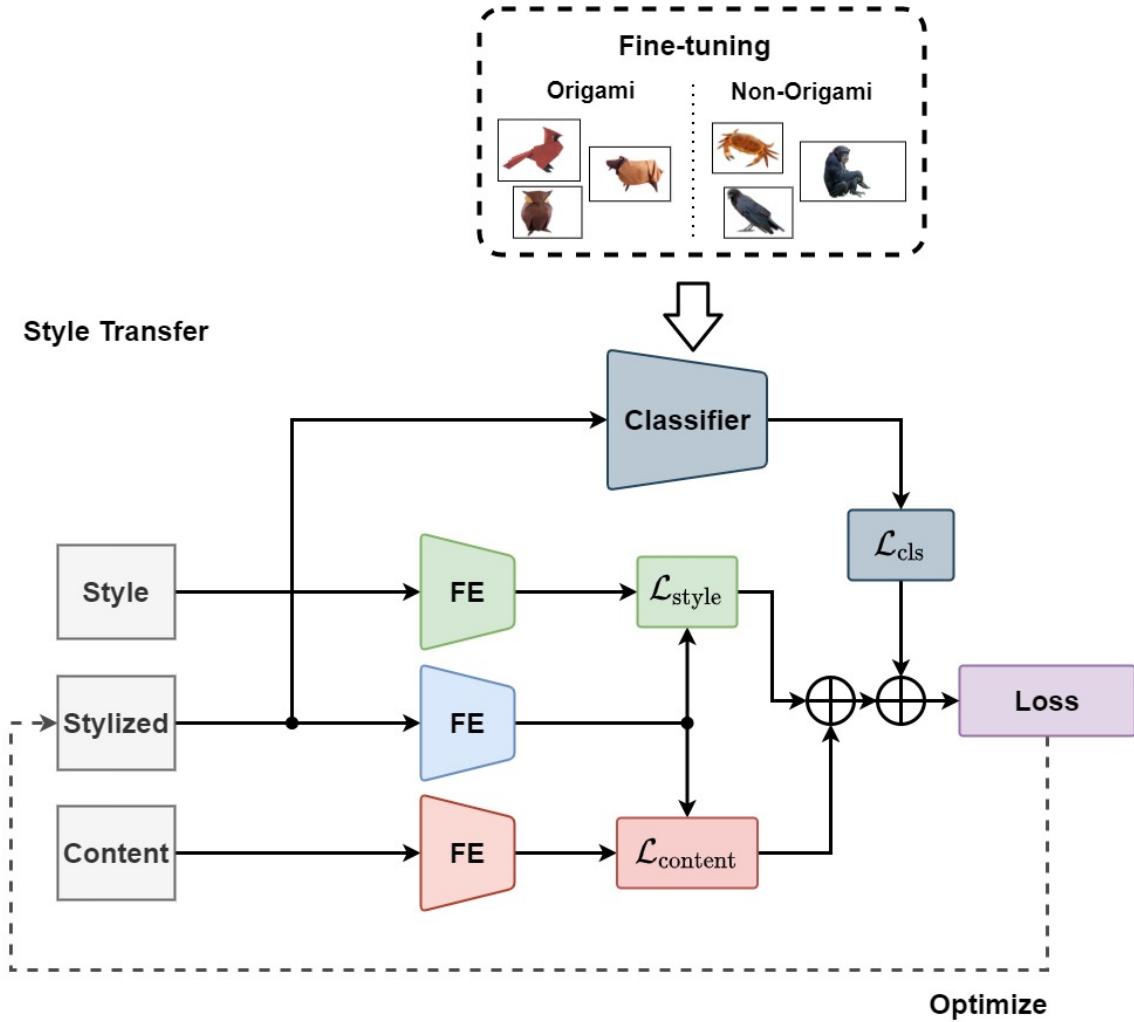


図 19: 手法 2 の概要図 . 予め折り紙の 2 値分類として学習させたモデルの出力を損失に用いる .

標として Accuracy を用いた .

この Fine-tuning されたモデルを判定器に導入する際は , 判定器のパラメータは固定とし , 学習は行わない . このときスタイル変換における損失を次のようにする :

$$\mathcal{L} = \mathcal{L}_{NST}(I_c, I_s, I_x) + \mathcal{L}_{cls}(\mathcal{D}(I_x)) \quad (10)$$

ここで \mathcal{L}_{NST} は式 (7) である . \mathcal{L}_{cls} は Binary Cross Entropy であり , 正解を 1 として判定器 \mathcal{D} の出力から計算する .

3.3 実装方法

提案手法の実装では，プログラミング言語として Python，機械学習フレームワークとして Tensorflow[1] を使用した。スタイル変換の画像生成では，3.1 節で示した折り紙画像を最も良く転送できる Kolkin らの手法をベースとした。手法 1 で学習する判定器は，最初の解像度で作成したものを全スケールで共有して学習を行った。

4. 評価手法と結果

本研究においては大きく3つの異なる手法で折り紙画像の変換を行っている。本節では、まず3.1.1説において各手法における生成結果の定性的評価を行う。3.1.2説ではパラメータの変化による影響、3.1.3説では既存のスタイル変換を用いた場合の比較を行う。

各生成においては画像は全てIntel(R) Core(TM) i9-10900K CPU @ 3.70GHzおよびNVIDIA RTX 3080を用いて作成を行った。コンテンツ画像についてはそれぞれPixabay⁵から6枚、スタイル画像はOrigamiSet[21]からサンプルした5枚を使用した。また物体のみを評価するため、生成した画像は全て背景を白にする処理を行っている。

4.1 判定器の同時学習による手法の生成結果

ここでは3.2.1節を用いた手法1Aおよび手法1Bを用いた結果について述べる。

手法1Aによる生成結果 図21に手法1Aを用いた結果を示す。手法自体の評価としては、この手法では判定器が寄与せず、生成には失敗している。全体的には前景の部分がスタイル画像の前景全体の平均色で塗られたように見られる。それ以外のスタイルの特徴も転送はできているが、(d)のカバと図12のKolkinらの結果を比べると、これらの特徴はKolkinらのStyle損失によるものであり、判定器による損失は殆ど効果が現れていないとみれる。これは図20に示すようなテクスチャ画像を正解画像に適用した場合だとより明確であり。判定器の影響がほとんど寄与できておらず、切り抜かれた画像の色のみしか反映できていない。

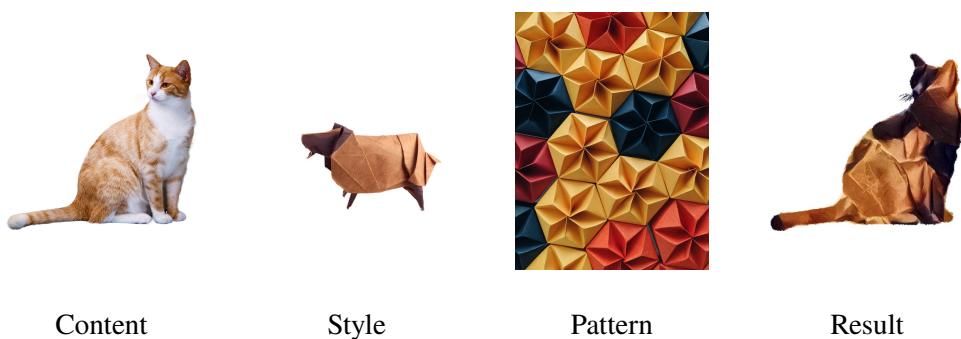


図20: 手法1Aにおいて正解画像にテクスチャ画像を用いた結果。

この手法による生成が失敗している理由としては、判定器に用いている画像の解像度が小さすぎるために、判定器が特徴を捉えることができていないことが挙げられる。図21で用いたスタイル画像の解像度は最大でも 640×480 ピクセルであり、本実験での切抜きサイズ

⁵<https://pixabay.com/>

は $1/8$ の 80×60 ピクセルが正解画像に入る事となる。すなわち、解像度が小さい画像で判定器が学習され、情報量が抽象的になるがために判定器の損失が色のみしか寄与しなくなつたものと考えられる。切り抜かれた画像が高解像度になればより詳細な特徴を捉えられる可能性はあるが、これはスタイル変換の初期解像度も上げる必要があり、本研究で使用した GPU のメモリよりも大きなサイズの確保が必要であるため本研究では追試は行わない。

手法 1B による生成結果 図 22 に手法 1B を用いた結果を示す。手法 1A に対して、判定器に入れたテクスチャ画像の特徴が上手く転送できている。しかし (a) のスタイルでは、蝶の翅の部分の折り目が全体的にバラバラに分布しており、Content 損失によるレイアウトが薄れてしまっている。(b) ではテクスチャ画像の特徴が殆ど紙の表面のような画像となっており、生成画像もほとんどそのテクスチャを張り付けたような結果となっている。(d) では羊の後ろ足が持っていた折り目が全体的に現れており、亀や猫、ペンギンなどは折り紙としては不自然な変換となっている。一方で (c) と (e) はテクスチャ画像の持つ特徴が上手く混ざつてあり、例えば (c) のカバ、(e) の兎や猫は近しい結果になっていると考える。

4.2 手法 2 による生成結果

ここでは 3.2.2 節を用いた手法 (以下手法 2 とする) での結果について述べる。

図 23 に手法 2 を用いた結果を示す。幾つかの画像においてアーチファクト発生しており、生成画像も大きく改善したとはいえない。例えばペンギンの画像においては、腹の斑点模様などの情報が残ってしまっている。また (a) の兎では全体的にノイズがかかった画像になっている。

そこでこの損失の寄与の有無を確認するため、(b) の兎の画像の学習過程での生成画像の変化を (図 24) を示す。この兎の画像においては、低解像度で Content 損失および Style 損失による最適化ができない。すなわち分類器による損失による寄与はほとんどなく、むしろ他の損失による最適化を阻害する要因となっている。

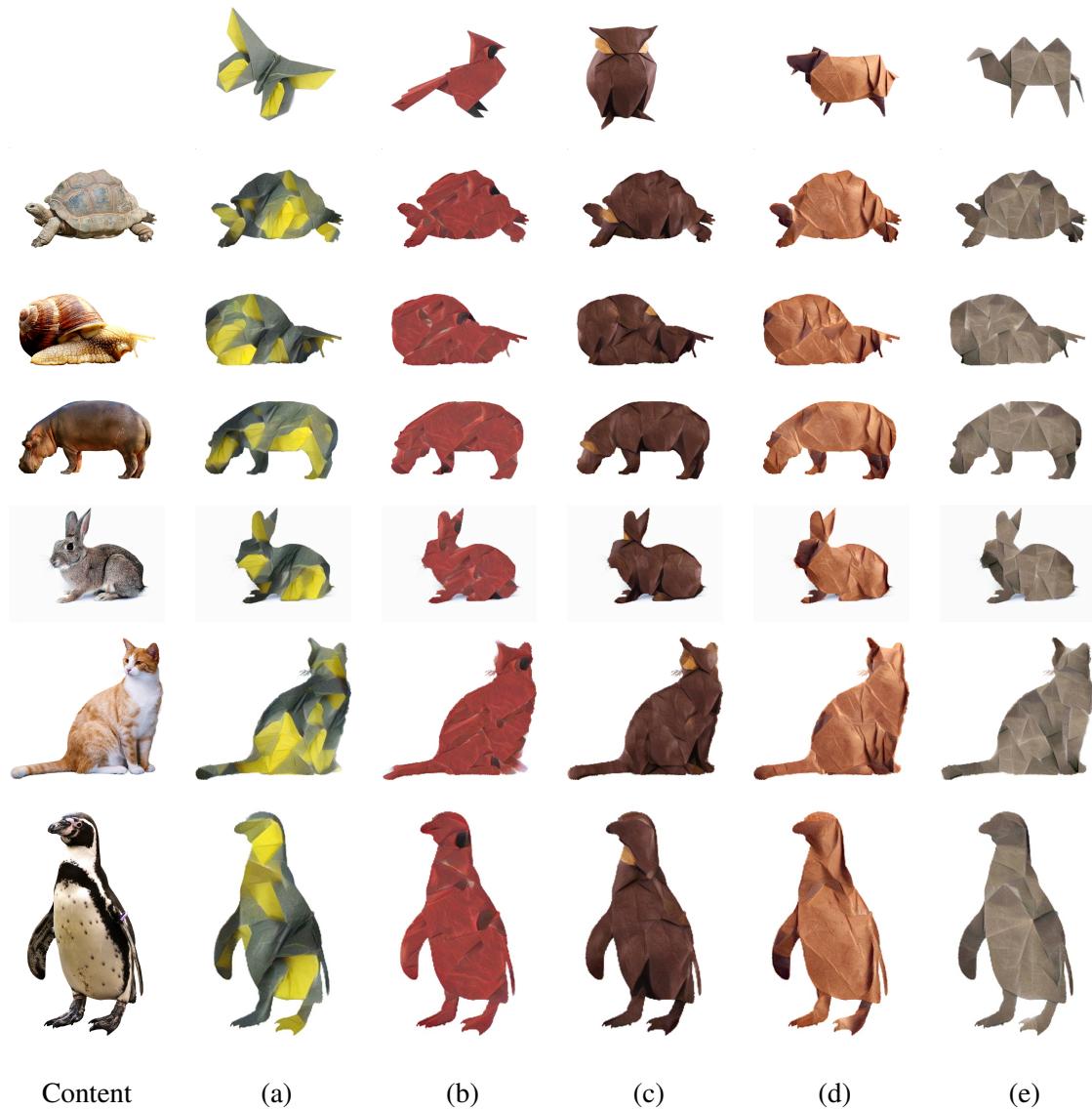


図 21: 手法 1A を用いた生成結果 . 損失のパラメータは $(\alpha, \beta, \gamma) = (4.0, 1.0, 1.5)$ とした .



図 22: 手法 1B を用いた生成結果

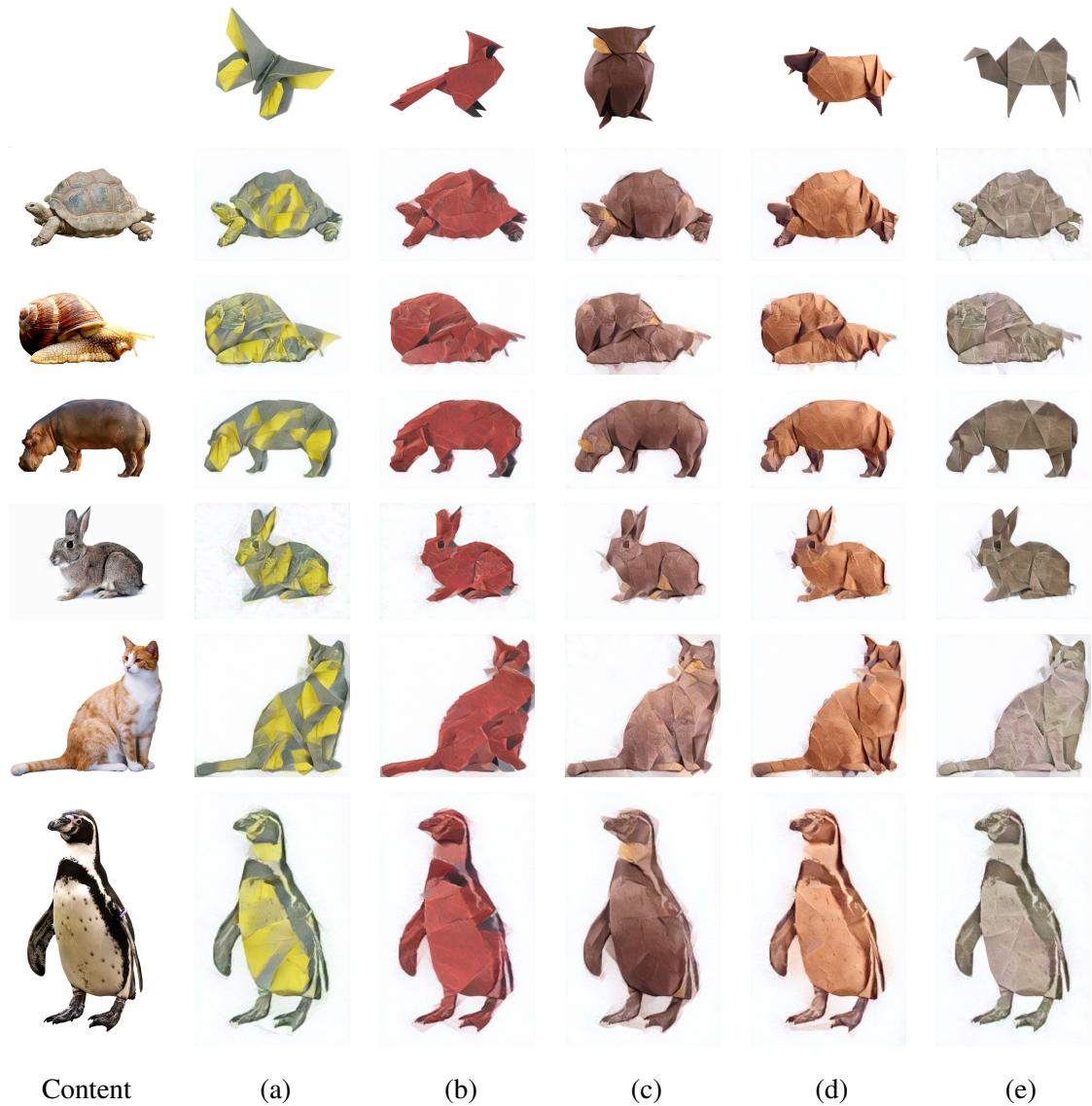


図 23: 手法 2 を用いた生成結果 . 損失のパラメータは $(\alpha, \beta) = (4.0, 1.0)$ とした .

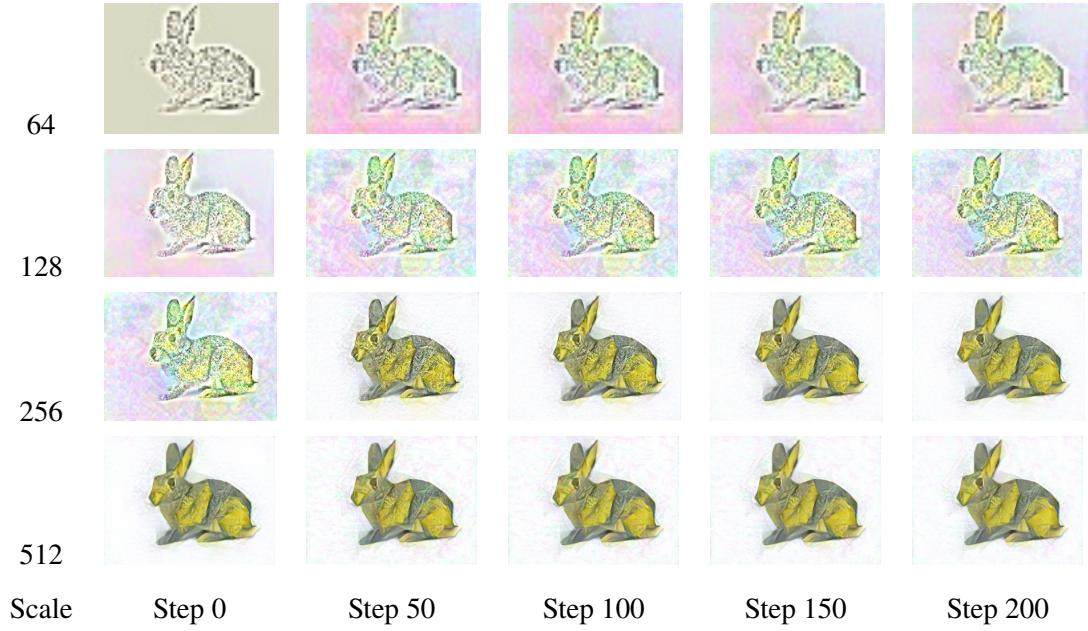


図 24: (a) の兎についての，生成過程における画像の変化

分類器の損失がほとんど寄与できていない原因としては，用いた分類器が折り紙の特徴を完全に捉えられていないことが考えられる。本手法で用いた分類器では *block5* 以降を Fine-tuning 対象として学習を行ったものを利用している。しかし VGG の *block5* は大域的特徴を見る層であるため，局所的な画像の情報などは学習することはできない。すなわち，折り紙の大まかな情報のみが学習されているために，その情報のみでスタイル変換において最適になった可能性があるとみれる。

4.3 損失関数のパラメータによる生成画像の変化

式 (3.2.1)において調整可能なパラメータ α, β, γ が存在するため，パラメータ次第ではより生成画像の改善が出来る可能性がある。そこで手法 1B で生成した (e) の猫について着目し，パラメータの変化による生成画像の品質について評価を行った。その結果を図 25 に示す。

4.4 アブレーション

手法 1B における考え方は，3 章で述べたように「スタイル画像の前景の折り紙の情報はテクスチャ情報でも保持される」というものである。この考えをさらに検証するため，手法 1B における生成をさらに次のパターンに分割して評価を行った：

Pattern 1 判定器損失にテクスチャ画像を使用する (手法 1B)

Pattern 2 スタイル画像をテクスチャ画像に置換する

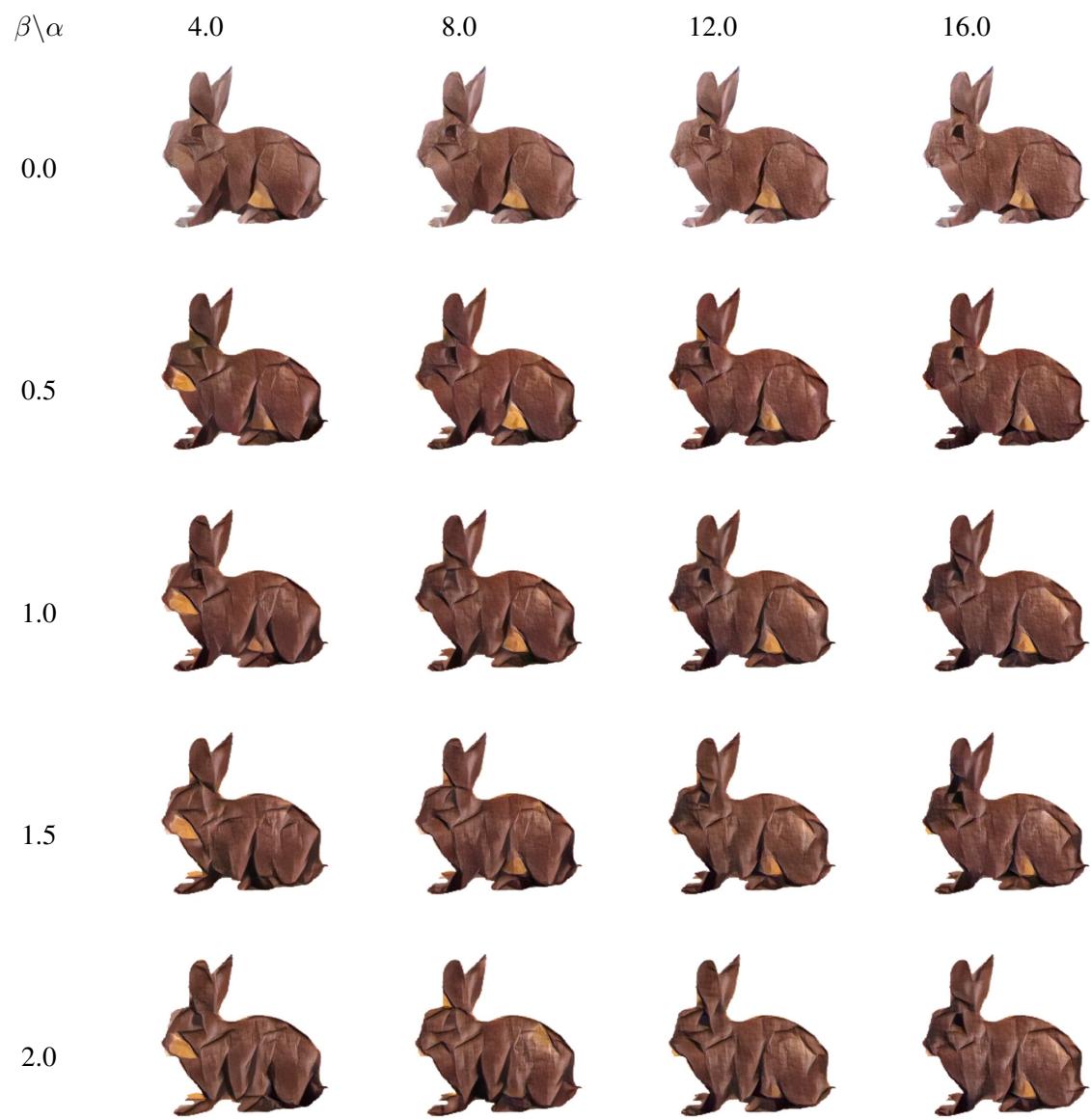
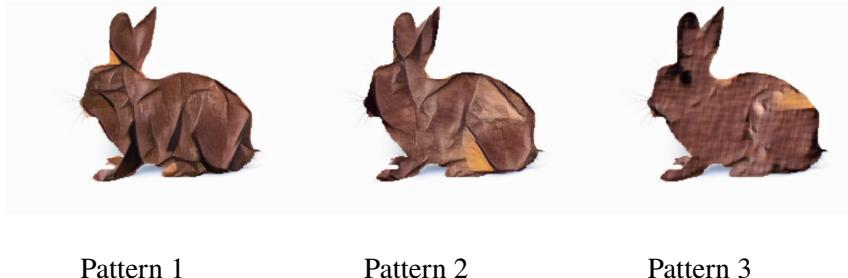


図 25: 手法 1B での損失関数のパラメータ (α, β) の変化 .

Pattern 3 コンテンツ損失と判定器損失のみで変換を行う

各パターンにおける結果を図 26 に示す . Pattern 2 の結果においては , Pattern 1 と比較してテクスチャを貼ったようになる . Pattern 3 ではテクスチャ画像の特徴がほとんど転送できておらず , 色合いのみだけで生成に失敗している .



Pattern 1

Pattern 2

Pattern 3

図 26: 手法 1B における手法の変化

これらの結果から , 手法 1B は仮説に反してテクスチャ化してしまうと折り紙の情報が失われてしまい , テクスチャ情報として学習されてしまうということが分かる . Pattern 2 の結果では Style 損失が認識する情報が折り紙ではない単純なテクスチャとして捉えることで , 同じ画像を用いている判定器においても単純な「テクスチャ画像」として認識されてしまうこと , そして Pattern 3 の結果より , 判定器において折り紙化否かの判定に必要な折り紙らしさの情報が認識できておらず , 手法 1A と同じように単純な色の要素しか学習ができていない . よって , このスタイル変換と判定器を同時に用いた手法では折り紙らしさを得ることは難しいものと考えられる .

4.5 既存のスタイル変換との比較

ここでは , 既存のスタイル変換の手法と最も変換が出来ていた手法 1B との比較を行う . その結果を図 27 に示す .

Kolkin ら [16] の手法では 3.1 節でも述べた通り , 完全に折り紙の特徴を画像から捉えることはできていないが , Kolkin らの手法では現時点において最も折り紙で折られたかのように近い結果を得られる . 手法 1B の結果と Kolkin らの手法を比較すると , 折り紙の質感としては本研究による手法がよく転送できているが , その影響が強すぎるため , レイアウトなどがほとんど消失してしまう . また判定器に用いた画像およびパラメータによって生成画像の質も変化するため , 手法 1B での有効性は図 25 のように各結果に対してかなりのチューニングによってかなり変化する . また 4.4 節で述べたように , 手法 1B はテクスチャとしての認識となるため , スタイルと判定器が与える影響が独立であるような損失 , 判定器の設計に修正する必要があると考える .



図 27: 提案手法と既存の Neural Style Transfer との比較 .

5. 結論

本論文では実物体画像から折り紙画像への変換として、スタイル変換と判定器を用いることによる折り紙画像の持つ折り紙としての特徴を反映させる手法を提案し、さらに判定器と一緒に学習させる手法、事前に学習させたものを利用する2つのアプローチからの変換を試みた。スタイル変換としては、Kolkinらの手法における損失関数の設計は折り紙に対して有効的であり、折り紙で折られたかのように近い画像も生成できることがわかった。生成画像の評価では、判定器の同時学習では折り紙としての情報ではなくテクスチャとしての情報が判定器に学習されたことで生成に失敗しており、また事前学習においては判定境界が曖昧であるがために生成画像に対して折り紙情報を十分に反映させることはできなかった。

本研究において提案した手法においては2つの問題が残っており、Style損失と判定器の損失での役割が独立関係にあること、そしてネットワークが持つ「折り紙」としての情報が十分に得られていないことの2つが挙げられる。Style損失は学習済みネットワークを用いて、一方の判定器損失はそれとは異なるネットワークから得られるデータから計算を行う。そのため判定器において「折り紙」を持った情報が得られたとしても、Style損失は「スタイル」としての情報を最適化するようにされており、互いに違う目的を持った改善となっている。また判定器に折り紙としての情報を寄り具体的に持たせるには、大域的な折り紙情報だけでなく、作品のエッジやパーツといった情報にも注意を向けさせなければならない。そのような情報を持たせるには広範囲の層を捉えるように判定器を学習する必要があり、提案手法で用いているVGGの場合 *block3*などの層も対象にしなければならない。このような層を学習させるためには、現在の使用する折り紙の条件を満たす画像をより多く収集する必要がある。

本研究での今後の課題はこの2つの改善であり、具体的には折り紙としての情報を持つ損失を考えること、そしてより厳密に折り紙として判定させるためのネットワークの改善・データの増強が課題となる。折り紙としての情報を損失に持たせるためには、コンテンツの大まかな位置関係の保持をさせること、そして1枚で折られた可能に見せるために各パーツの関係性を持たせるような定義が必要となる。パーツ関係は2次元画像のみでの推定は困難であるが、三谷ら[14]のような立体表現のための折り畳み情報の付与は利用できると考える。さらに近年Flexible Origami List Datastructure (FOLD)[3]と呼ばれる折り紙のモデル化のためのフォーマットが開発中⁶であり、これを用いたGraph Neural Network (GNN)を用いた折り畳み関係の情報付与などによって実現できるのではないかと考えられる。

⁶<https://github.com/edemaine/fold>

謝辞

初めに、指導教員である川嶋宏彰教授には研究の方向性や提案手法などを含め、様々な御指摘、御助言を頂きました。折り紙と画像生成という前例のない研究テーマではありましたが、川嶋宏彰教授によるご協力があったからこそ進めることができました。謹んで感謝の念を示します。

本年度博士前期課程2年生の金城匡氏には研究だけでなく社会人としても多くの御助言を頂きました。謹んで感謝の念を示すとともに、お子様の健やかなご成長を心からお祈り申し上げます。

本研究室の博士前期課程、および学部生の皆様におかれましても、約9ヶ月という短い期間ではありましたが、研究に対するご助言など様々な部分でお世話になりました。私の趣味である折り紙に付き合って頂いたことは忘れられません。

私が学士課程時代に在籍していた工学研究科 電子情報工学専攻 物性・デバイス研究室の中嶋誠二准教授、藤澤浩則教授におかれましては、卒業後も就職活動や修士学生としての様々な助言を頂きました。ここに感謝の念を示します。

最後に兵庫県立大学学生として6年間、私を温かく見守ってくださった家族に感謝いたします。

参考文献

- [1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, et al. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467*, 2016.
- [2] Dongdong Chen, Lu Yuan, Jing Liao, Nenghai Yu, and Gang Hua. Stylebank: An explicit representation for neural image style transfer. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1897–1906, 2017.
- [3] Erik D Demaine, Jason S Ku, and Robert J Lang. A new file standard to represent folded structures. In *Abstr. 26th Fall Workshop Computat. Geometry*, pages 27–28, 2016.
- [4] 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*, pages 248–255. Ieee, 2009.
- [5] Alexei A Efros and William T Freeman. Image quilting for texture synthesis and transfer. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 341–346, 2001.
- [6] Leon Gatys, Alexander S Ecker, and Matthias Bethge. Texture synthesis using convolutional neural networks. *Advances in neural information processing systems*, 28, 2015.
- [7] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. Image style transfer using convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2414–2423, 2016.
- [8] Leon A Gatys, Alexander S Ecker, Matthias Bethge, Aaron Hertzmann, and Eli Shechtman. Controlling perceptual factors in neural style transfer. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3985–3993, 2017.
- [9] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. *Advances in neural information processing systems*, 30, 2017.

- [10] Xun Huang and Serge Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *Proceedings of the IEEE international conference on computer vision*, pages 1501–1510, 2017.
- [11] Yongcheng Jing, Yezhou Yang, Zunlei Feng, Jingwen Ye, Yizhou Yu, and Mingli Song. Neural style transfer: A review. *IEEE transactions on visualization and computer graphics*, 26(11):3365–3385, 2019.
- [12] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *European conference on computer vision*, pages 694–711. Springer, 2016.
- [13] Nikolai Kalischek, Jan D Wegner, and Konrad Schindler. In the light of feature distributions: moment matching for neural style transfer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9382–9391, 2021.
- [14] Yuya Kato, Shinichi Tanaka, Yoshihiro Kanamori, and Jun Mitani. Single-view modeling of layered origami with plausible outer shape. In *Computer Graphics Forum*, volume 38, pages 629–640. Wiley Online Library, 2019.
- [15] Sunnie SY Kim, Nicholas Kolkin, Jason Salavon, and Gregory Shakhnarovich. Deformable style transfer. In *European Conference on Computer Vision*, pages 246–261. Springer, 2020.
- [16] Nicholas Kolkin, Jason Salavon, and Gregory Shakhnarovich. Style transfer by relaxed optimal transport and self-similarity. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10051–10060, 2019.
- [17] Robert J Lang. A computational algorithm for origami design. In *Proceedings of the twelfth annual symposium on Computational geometry*, pages 98–105, 1996.
- [18] Robert J Lang. Treemaker, 2015. <https://langorigami.com/article/treemaker/>.
- [19] Xiao-Chang Liu, Xuan-Yi Li, Ming-Ming Cheng, and Peter Hall. Geometric style transfer. *arXiv preprint arXiv:2007.05471*, 2020.
- [20] Fujun Luan, Sylvain Paris, Eli Shechtman, and Kavita Bala. Deep photo style transfer. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4990–4998, 2017.

- [21] Daniel Ma, Gerald Friedland, and Mario Michael Krell. Origamiset1.0: Two new datasets for origami classification and difficulty estimation. In *Proceedings of Origami Science Maths Education*, 2018.
- [22] Tamar Rott Shaham, Tali Dekel, and Tomer Michaeli. Singan: Learning a generative model from a single natural image. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4570–4580, 2019.
- [23] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.
- [24] Tomohiro Tachi. Origamizing polyhedral surfaces. *IEEE transactions on visualization and computer graphics*, 16(2):298–311, 2009.
- [25] Jaejun Yoo, Youngjung Uh, Sanghyuk Chun, Byeongkyu Kang, and Jung-Woo Ha. Photorealistic style transfer via wavelet transforms. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9036–9045, 2019.
- [26] 三谷 純. 折紙の展開図専用エディタ (oripa) の開発および展開図からの折りたたみ形状推定. 情報処理学会論文誌, 48(9):3309–3317, 2007.
- [27] 鶴田 直也, 三谷 純, 金森 由博, and 福井 幸男. 幼児向け折り紙作品の創作支援システム. *IPSJ symposium series*, 2011(3):49–56, 2011.