

研究論文

Smart Layer Splitter : pix2pix を用いた デジタルイラスト制作の色塗り工程における 自動レイヤ分けシステム

渡邊 優^{1,a)} 阿倍 博信^{1,b)}

受付日 2020年7月15日, 採録日 2020年11月26日

概要: デジタルイラスト制作の一工程である色塗り工程では, 線画を髪や肌などのパーツにレイヤ分けする作業が必要である. しかし, 既存のグラフィックソフトに付属する塗りつぶしツールでは, 手作業のため手間がかかってしまうという問題があった. そこで, 本論文では, conditional GAN の一方式である pix2pix と後処理を組み合わせることでレイヤ分け作業を自動化する方式について提案する. 提案方式の評価を行った結果, Mean Accuracy で 84.8% の精度が得られることを確認した. さらに, 提案方式に基づき, 自動レイヤ分け作業において誤りが発生した場合でも誤りを手動で修正する UI を持った自動レイヤ分けシステム: Smart Layer Splitter を開発し, システムの有効性について評価を行った. その結果, 既存のグラフィックソフトと比較して, 作業時間を 39.8% 短縮できるとともに, 操作回数を 68.6% 削減できることを確認した.

キーワード: イラスト, pix2pix, レイヤ分け, 創作活動支援, AI 応用

Smart Layer Splitter: An Automatic Layer Splitting System for Coloring Process of Creating Digital Illustration Using pix2pix

YU WATANABE^{1,a)} HIRONOBU ABE^{1,b)}

Received: July 15, 2020, Accepted: November 26, 2020

Abstract: In the coloring process of creating digital illustration, it is necessary to split line drawing into layers for each part such as hair and skin. In the past, it was necessary to do this work with the fill tool attached with existing graphics software, but there was a problem that it takes time and effort because it becomes manual. In this paper, we propose a method to automate layer splitting work by combining pix2pix, which is a method of condition GAN, and post-processing. As a result of the evaluation of the proposed method, the accuracy of 84.8% was confirmed in Mean Accuracy. Furthermore, we developed an automatic layer splitting system: Smart Layer Splitter based on the proposed method, and evaluated the effectiveness of the system. This system has a UI that manually corrects errors even if errors occur during automatic layer splitting operations. As a result, it was confirmed that the working time can be reduced by 39.8% compared with existing graphics software, and the number of operations can be reduced by 68.6%.

Keywords: illustration, pix2pix, layer splitting, supporting creative activities, AI application

1. はじめに

一般的に, デジタルイラストの制作工程は, ラフ, 線画,

色塗りの3つの工程から構成され, さらに色塗り工程はレイヤ分けと影塗りの2つの作業から構成される. レイヤ分けはイラストの線画に対して, 肌や髪などのパーツにレイヤに分ける作業であり, 後で行う影塗りの下地となる部分を制作する作業である.

従来, レイヤ分け作業は, グラフィックソフトに付属す

¹ 東京電機大学
Tokyo Denki University, Adachi, Tokyo 120-8551, Japan
^{a)} 19jkm22@ms.dendai.ac.jp
^{b)} hironobu.abe@mail.dendai.ac.jp

る塗りつぶしツールや選択範囲ツールを使用して行う場合が多い。選択したい領域の先端が細い場合や、線画の線が途切れている部分に対してこれらのツールを使用した場合、領域を認識することができず、領域から色があふれ出してしまい、ユーザの想定どおりに選択することができない。その対応として人手で線画の隙間を修正した後、再度塗りつぶしツールを使用する必要があるため、修正作業に手間がかかる問題がある。さらに、このレイヤ分け作業は領域を単色で塗りつぶす単純作業であり、イラストの品質に直接影響しない。このような背景から、レイヤ分け作業を AI 技術により自動化することで、制作者のデジタルイラスト制作作業の支援になると考えられる。

本論文では、デジタルイラストの制作現場において、AI 技術を用いたレイヤ分け作業の自動化を目的として、conditional GAN の一種である pix2pix [1] を用いてレイヤ分け作業を自動化する方式について提案し、それを用いた自動レイヤ分けシステムである Smart Layer Splitter の開発およびその評価について述べる。

以下、本論文では、2 章にて関連研究、3 章にて自動レイヤ分け方式の提案、4 章にて自動レイヤ分けシステム：Smart Layer Splitter の開発、5 章にてシステム評価、6 章にて考察について述べ、最後にまとめる。

2. 関連研究

前述のとおり、デジタルイラストの制作工程は、ラフ、線画、色塗りの 3 つの工程から構成される。このなかでも、本論文でも対象としている色塗り工程の作業に対する AI 技術の適用は最も研究がさかんな分野の 1 つである。

PaintsChainer [2], [3] は、ユーザが線画をアップロードし、領域の色や影を指定することで色塗り工程を自動で行う Web サービスである。基本構造としてセマンティックセグメンテーション技術の 1 つである U-net [4] を採用しており、線画と着色ヒントを入力することで着色を行う。しかし、U-net のみで着色を行うとセピア色の着色結果となってしまう以上は学習が進まないことから、GAN 構造とすることで着色ヒントに沿った線画の着色を実現している。

線画の着色に関しては、PaintsChainer 以外の研究も存在する。Style2PaintsV3 [5] は 2 つのモデルを組合せ線画の着色を行っている。着色ヒントをもとに、線画に対して色を散布させ色の下書きを生成するモデルと、色の下書きからはみ出しやゆがみなどを除去し洗練するモデルから構成されている。線画への着色タスクを色の着色と消しに役割を分けることで、学習が容易になり、ゆがみやにじみのない高品質な着色結果を実現している。

また、漫画の色塗り工程の自動化に関する研究も存在する。Comicolorization [6] は漫画の色塗り工程を自動化するシステムである。モデルは白黒写真に対して自動的に色塗

りを行う CNN を漫画に適応したものである。システムに対して、漫画と一緒に参照画像を入力すると、参照画像の色を使用して漫画に色を塗る。

また、セグメンテーションを使用して漫画の色塗りを行う研究も存在する [7]。画像処理を用いて漫画のトーンを除去した後、線画と参照画像を pix2pix に入力して色塗りを行い、セグメンテーションや色の補正によってグレースケールの漫画をカラー漫画に変換している。本研究では、セグメンテーションについての方式を参考にしている。

上記以外にも、AI 技術を活用して色塗り工程の作業を自動化する研究が行われているが、基本的に色塗り工程の作業の全工程を自動化する技術が中心であるため、出力画像が 1 枚のみであり、本論文で必要となるパーツへのセグメンテーションや、レイヤ分けが行われていないため、本論文の目的には適さない。

デジタルイラスト制作以外の分野では、前述した PaintsChainer にも使用されている U-net を用いてアニメ制作における色塗りの自動化に関する研究が存在する [8]。アニメは同じキャラクターが複数出現するため、キャラクターごとに U-net を用いて学習させ、セマンティックセグメンテーションに基づき着色を行い、はみだしなどが発生した場合には、後処理において、各領域内での色数の投票により、最終的な色を決定している。

色塗り工程以外の作業に対する制作支援としては、線画工程における作業への AI 技術の適用に関する研究も存在する。線画工程における制作作業は、ラフと呼ばれる大雑把な構図を描いた物から適切な線を選択し、なぞる作業である。Smart Inker [9] では、FCN を用いることで、ラフから線画を生成する。ユーザはラフの画像を入力し、消去したい線と加筆したい線を指定することで線画の自動生成を実現している。

また、カラーイラストの陰影に着目し、1 枚のイラストからベース色、ライティングと陰影などのレイヤに分ける研究も存在する [10]。レイヤの合成モードを考慮しており、陰影は乗算、ハイライトはハードライトで表現している。しかし、本研究では完成後のイラストではなく、線画からパーツに分解する点で異なる。

次に、レイヤ分け作業と関係性が深いセマンティックセグメンテーションのアノテーション方式について説明する。セマンティックセグメンテーションの学習データを作成するアノテーション作業は非常に時間がかかる作業であるため、作業を効率化するためのツールがいくつか提供されている。代表的なアノテーションツールとしては、Sekachev らが開発した Computer Vision Annotation Tool [11] と Microsoft が提供している Visual Object Tagging Tool [12] があげられる。これらのツールではアノテーション作業を行う際、学習済みのモデルを使用して、自動でアノテーションを行う機能が提供されている。自動アノ

テーションを行ったあと、間違った部分を手動で修正することで、アノテーション作業の効率化を図っている。

3. 自動レイヤ分け方式の提案

3.1 基本方針

2章の結果をふまえ、本論文にて提案する自動レイヤ分け方式の基本方針について下記のとおり整理する。

- 類似したタスクであるイラストの線画の自動着色方式を参考に、自動レイヤ分けを彩色問題として扱う方針とし、基本方式として画像生成モデルである pix2pix を選択する。
- 彩色問題としたために生じた不安定な領域は、先行研究 [7], [8] でも採用しているセグメンテーション技術と後処理としての画像補正処理を用いて、レイヤ分けを実現する。
- 後処理としての画像補正処理で補正しきれなかった箇所については、セマンティックセグメンテーションのアノテーションツールと同様にユーザが手動でパーツ領域の修正を行う。

3.2 自動レイヤ分け方式の概要

図 1 に本論文にて提案する pix2pix を用いた自動レイヤ分け方式の概要について示す [13], [14]。最初にユーザが入力した線画を、conditional GAN の一種である pix2pix を用いて、髪、肌、目、服、背景の 5 つのパーツに分ける形で色を塗る。また、彩色問題としたために出力画像の不安定な領域にノイズやにじみが発生し、指定した色以外の色が現れる。不安定な領域が明確になるため、信頼性に欠けるピクセルを除外するといった後処理を組み合わせることで精度の向上が図れると考えた。

前述のとおり、pix2pix の実行結果は、色のはみだしやにじみが存在する可能性が高く、そのままではレイヤ分け画像として使用することはできない。レイヤ分けとは、パーツの領域を単色で塗りつぶす作業であり、領域から色がはみだすことや、色の不安定な場所があってはならない。たとえば肌の領域が不安定な場合、背景の色が透ける可能性があり、完成作品に悪影響を及ぼす可能性があるためである。そこで、pix2pix の実行結果に対する後処理として、線を塞ぐ処理とセグメンテーションを中心とする画像補正処理を行い、その結果をレイヤに分け、最終出力とする。以下、pix2pix とその後処理について、処理内容の詳細について説明する。

3.3 入力画像の仕様

本研究において対象とする入力画像は、キャラクターの線画とする。以下に入力画像の仕様について示す。

- 画像データの仕様

キャラクターの線画で、背景が透過されているものを対象

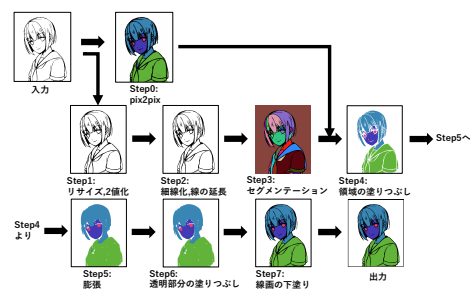


図 1 pix2pix を用いた自動レイヤ分け方式の概要

Fig. 1 Overview of an automatic layer splitting method using pix2pix.

とする。透過されていないものについては、背景と十分に明度が違うものとし、線は輝度が 200 cd/m^2 以下であり、背景は 200 cd/m^2 以上であるものを対象とする。そのため、主にデジタルで制作した線画を対象とする。アナログの線画の場合、画像内にノイズのないものとし、スマートフォンのカメラで写真を撮った際に存在する影の写りこみはないものを対象とする。画像サイズは指定しないが、幅、高さとも 100 px 以上 $2,000 \text{ px}$ 以下を目安とする。

- 構図の仕様

画像内に背景の線画は含まず、キャラクターのみが写っているものを対象とする。またキャラクターについて、画像内に上半身のみが写っているものを対象とする。顔の大きさについては制限しないが、顔全体が画面に写っているものとし、見切れていないものを対象とする。また、顔は正面に近いものを対象とし、横向きや後ろ向きの画像は対象外とする。

- キャラクターの仕様

5 頭身程度のキャラクターを対象とし、2 頭身程度のデフォルメされたキャラクターは対象外とする。髪の毛の長さや髪形については制限せず自由とする。目の形状や大きさについても制限せず、目を閉じているものや、前髪で隠れているものについても対象とする。

- 線の仕様

線画において、線の太さやテクスチャといったスタイルは自由とする。髪などの一部の領域や影をベタで塗りつぶしている画像を含んでもよいが、斜線やトーンは含まないとする。また、ラフのような線の跡切れが大きく清書されていないものや、髪の毛において毛束の先端の線を閉じていないもの、少女漫画などの線の隙間を故意に作っている絵については、線の補間が難しいため対象外とする。

3.4 出力画像の仕様

出力画像は髪、肌、目、服、背景の 5 つのパーツと線画 1 枚がセットになった PSD (PhotoShop Document) [15] 形式とする。出力画像の概要について図 2 に示す。

PSD 形式はレイヤ構造を格納することができ、一般的なグラフィックソフトで扱うことが可能である。レイヤの

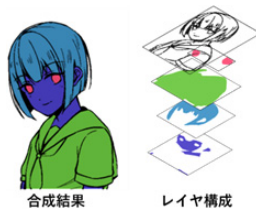


図 2 出力画像の概要

Fig. 2 Overview of output images.

合成順は上から線画、目、服、髪、肌、背景の順とし、すべてのレイヤの合成モードは通常モードとする。また、あるパーツが他のパーツと被らないようにする。たとえば肌パーツの目の領域部分は透明とする。また、イラストの線画の下については、線のスタイルや透明度によっては背景色が透けてしまう可能性がある。そのため、線画の下には周囲のパーツ色を塗る。パーツとパーツの間にある線については線の中心線を境界として扱う。

3.5 pix2pix を用いた自動レイヤ分け処理

(1) pix2pix の概要

pix2pix とは Isola らが考案した conditional GAN を用いて Image to Image のタスクを汎用的に行うアルゴリズムである [1]。GAN とは生成側 (Generator) と判断側 (Discriminator) の 2 つのモデルから構成されるモデルである。pix2pix では Generator として先行研究 [8] においても採用している U-net を使用している。U-net は医療画像のセグメンテーションにおいて高スコアを持つモデルであり、入力と出力は画像である。そのため、pix2pix は画像から画像への変換を高品質に行うことができる。図 3 に pix2pix のブロック構成について示す。

(2) 学習データの概要

イラストコミュニケーションサービス pixiv [16] から、入力画像の仕様を満たす線画を 345 枚収集した。収集した線画の画像サイズは平均で幅: 662 px, 高さ: 713 px であった。次に正解データとして、収集した線画を髪、肌、目、服、背景の 5 つのパーツをそれぞれ別の色で塗り分けして対応するレイヤ分け画像を作成した。このとき、グラフィックソフトを使用して、手動で塗り分け作業を行った。レイヤ分け画像の形式は PNG 形式とし、インデックスカラーは使用していない。このとき、塗り分けはアンチエイリアスなしで行っており、線画は 2 値化している。作成したデータを 9:1 に分け、9 割を学習データ、残りの 1 割をテストデータとした。

次に、水増し処理として左右反転、1~4 度の回転、0.5~1.5 倍にリサイズ処理を行った。また、線画とレイヤ分け画像について、ニアレストネイバー法を用いてリサイズした。ニアレストネイバー法を用いた理由は色数を増やさないためである。このとき画像を、アスペクト比を保持した

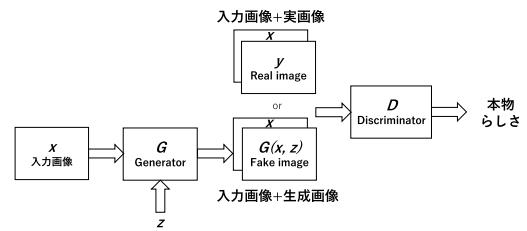


図 3 pix2pix の構成

Fig. 3 Block structure of pix2pix.

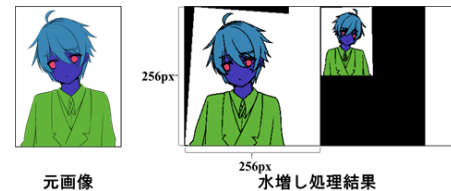


図 4 学習データの例

Fig. 4 An example of training data.

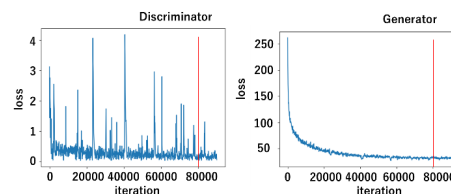


図 5 損失関数の出力の推移

Fig. 5 Transition of output of loss function.

まま長辺を 256 px に縮小した後、画像を 256 px × 256 px の白背景の画像に貼り付けを行うことで 256 px × 256 px の画像とした。水増し処理を行った結果、作成したデータは学習データ: 12,420 枚、テストデータ: 1,380 枚となった。サイズを変更する前の学習データと水増し処理の結果について図 4 に示す。

(3) pix2pix によるモデル作成

次に、pix2pix によるモデル作成について述べる。ディープラーニングのフレームワークとして Chainer [17] を選択し、chainer-pix2pix [18] の環境を構築し、Python を用いて pix2pix のモデル作成を行った。また、学習のパラメータは Batch Size を 2 とした。

Generator 側については、入力画像を 256 px × 256 px の線画とし、教師ラベルを 256 px × 256 px のレイヤ分け画像とした。また、入力チャンネル、出力チャンネルは RGB の画像であるため 3 とした。損失関数は教師ラベルと Generator の出力結果の平均絶対誤差 × 100 + Adversarial loss とした。

Discriminator 側について、入力画像は Generator の生成画像と教師ラベルとした。また、入力チャンネルを 3、出力チャンネルを 1 とした。損失関数は Adversarial loss とした。最適化方式は Adam とし、パラメータは $\alpha = 0.0002$, $\beta_1 = 0.5$, $\beta_2 = 0.999$ とした。

図 5 に、テストデータを使用した際の損失関数の出力の推移を示す。その結果、Generator 側の損失関数の出力は



図 6 pix2pix の出力

Fig. 6 Output of pix2pix.

250 から 25 まで低下した。また、Discriminator は 0.5 付近を往復していることから、正常動作していることを確認できた。80,000 iteration で損失関数が最小となったと判断し、このときのモデルを pix2pix のモデルとして採用した。

3.6 後処理としての画像補正処理

前節で作成した pix2pix のモデルに対して、線画を入力した際の出力結果を図 6 に示す。

図 6 を見ても分かるとおり、pix2pix の出力結果は同一の領域内が単色ではない。たとえば、首に着目すると、領域内の色が不安定になっていることが確認できる。レイヤ分け画像はすべての領域が単色である必要があり、レイヤ分けとはパーツの領域を単色で塗りつぶす作業で、領域からはみだすことや色の不安定な場所があってはならない。また、実際のイラスト制作では線画と色のレイヤは分けることが多い。しかし、pix2pix の出力結果は線画と色が結合されている。この状態で線画と色を分けた場合、線画の下の色は透明となってしまう。これは線画のスタイルや透明度によっては背景の色が透けてしまい、作品に悪影響を及ぼす可能性がある。以上の理由から、pix2pix の出力結果に対して後処理として画像補正処理を行うこととした。

セマンティックセグメンテーションの後処理の既存方式として、Fully connected CRF [19] がある。Fully connected CRF はピクセルの輝度を参照し、類似している輝度を持つピクセルに対して同一のラベルを付与する方式である。しかし、本研究で扱っている線画は、画像内のピクセルが白と黒のみで構成されているため、輝度情報を参考に後処理を行うことができない。また、Fully connected CRF は必ずしも 1 領域内を単色にするといったレイヤ分け画像の条件を満たすような後処理を行うとは限らない。以上の理由から、本研究の後処理では、既存方式を使用せず、新規に方式検討する方針とした。

後処理の基本方針について下記のとおり整理する。

- 線画は白と黒のピクセルのみで構成されていることから、輝度値の変化を活用した後処理による精度向上が見込めないため、Fully connected CRF などの既存方式は採用しない。
- 先行研究 [7], [8] と同様に、セグメンテーションを使用して後処理を行う。
- セグメンテーションの際、イラストの線が完全に塞が

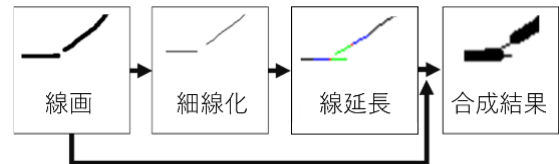


図 7 Step2 の概要

Fig. 7 Overview of Step2.

れているとは限らない。線が塞がれていない場合は、セグメンテーションに失敗し、結果として精度が低下する可能性がある。そのため、線画の跡切れを検出し、延長することで線の途切れを減らす。

- pix2pix の出力結果にセグメンテーションを適応した際に領域が小さい場合、領域内のにじみの影響を受けてしまい、色を検出できない可能性がある。小さい領域は周辺のパーツに属する可能性が高いため、周辺ピクセルを参照して領域をパーツに分類する。
- 線画の透明度によっては、線の下まで塗りつぶさなければ背景が透けてしまう可能性がある。そのため、各パーツについて領域の膨張を行うことで線画の下まで塗りつぶす。

以下、図 1 の提案方式の概要で示した Step ごとに、後処理の内容について説明する。処理速度の観点から Step2 のみ C 言語で実装し、他の Step はすべて Python で実装した。

(1) Step1: リサイズ, 2 値化

pix2pix の出力サイズを線画のサイズに合わせてリサイズする。pix2pix の出力画像は 256 px × 256 px であるため、元の線画のサイズにリサイズ処理を行う。また、入力線画を 2 値化する。理由は、Step2 にて細線化を行う必要があるためである。2 値化する際の閾値として、輝度が 200 cd/m² 以下を黒（線）、それ以上を白（背景）とし、背景は透明色に置き換える。

(2) Step2: 細線化, 線の延長

Step1 で 2 値化した入力線画を Nagendraprasad-Wang-Gupta のアルゴリズム [20] で細線化を行う。細線化する理由としては、端点を認識し、線の跡切れを減らすためである。細線化画像から端点を検出し、隣接する線を 5 px たり、コピーする。コピーした線を端点に貼り付け、線の延長を行う（図 7）。

これによって線の跡切れを軽減し、Step3 で行うセグメンテーションの精度を高める。端点とは、あるピクセルを中心としてみたとき、8 近傍のピクセルを参照したときに、1 px が黒であり、そのほかは透明である点と定義する。図 7 において、赤色のピクセルは検出された端点、青色のピクセルは端点から 5 px たどった線、緑色のピクセルが延長された線である。図 7 の合成結果より、Step2 の線の延長を行うことで線画の隙間の補正について確認できる。

(3) Step3: セグメンテーション

Step2 の結果と Step1 の線画を結合し、その結果について、閉領域の塗りつぶしを行い、すべての領域に対してセグメンテーションを行う。

(4) Step4: 領域の塗りつぶし

Step0 の pix2pix の出力と Step3 の出力を参照し、各領域に最も含まれているパーツの色を用いて領域を塗りつぶす。これをすべての領域に対して行う。このとき、領域を矩形として面積が 500 px 以下となった場合は、領域が小さいと判断し、塗りつぶさない。

イラストには髪の毛など、小さい領域が複数存在するほか、Step2 の線の延長によってさらに小さい領域が発生している。そのような小さい領域が領域内のピクセルが不安定な場合には、周囲と異なるパーツが割り当てられてしまう可能性があり、周囲から浮いてしまう。そのため、周囲のピクセルに合わせるために、Step4 では塗りつぶさず、後の段階でパーツが確定している周辺ピクセルを参照して塗りつぶしを行う。このとき、各色は異なるレイヤに描画する。

(5) Step5: 膨張

レイヤのパーツの各色について、線画の黒色領域をマスクとして 3px の膨張を行うことで、パーツの間の線の下を塗りつぶす。線画の透明度によって、線の下まで塗りつぶさなければ背景が透けてしまう可能性があるためである。

(6) Step6: 透明部分の塗りつぶし

Step4 にて塗りつぶされなかった透明ピクセルについて、8 近傍の色を参照して、最も多かった色を取得し、その領域に閉領域の塗りつぶしを行う。ただし、周辺ピクセルに透明色が一番多かった場合は塗りつぶさない。また、領域が背景色であると判断した場合は、参照するピクセルを 15 近傍に増やしてもう 1 度スキャンを行い、その結果の色を塗る。

(7) Step7: 線画の下塗り

線画と延長線を描画しているレイヤの下のレイヤは色が塗られていないピクセルが存在するため、塗られていないピクセルについて、8 近傍のピクセルの色をカウントし、最も多かった色を線画の下塗りを行う。この処理を透明色のピクセルがなくなるまで繰り返す。

3.7 自動レイヤ分け方式の精度評価

セマンティックセグメンテーションの評価方式の 1 つである Mean Accuracy [21] を用いて、提案した自動レイヤ分け方式の精度評価を行った。Mean Accuracy とは、パーツ単位に精度を求める評価方式の一方式であり、パーツの面積に精度が依存しないことが特徴である。本研究において使用した Mean Accuracy の定義について式 (1), (2) に示す。

表 1 自動レイヤ分け方式の精度評価の結果

Table 1 Evaluation results of accuracy.

	平均精度(%)				
	肌	髪	服	目	全体
pix2pix	74.5	87.5	84.4	56.0	81.6
pix2pix+後処理	78.6	90.0	86.0	64.2	84.8
U-net	80.6	83.7	85.7	79.1	82.5

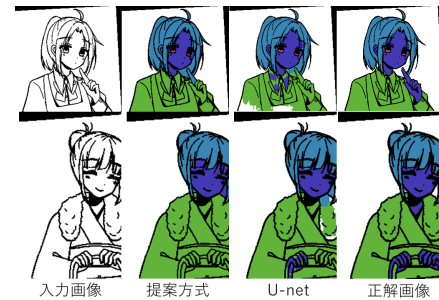


図 8 出力画像の比較

Fig. 8 Comparison of output image.

$$\text{Mean Accuracy} = \frac{1}{n_{cl}} \cdot \sum_i \frac{n_{ii}}{t_i} \quad (1)$$

$$t_i = \sum_j n_{ij} \quad (2)$$

ここで、 n_{ij} はクラス j に属すると予測されるクラス i に属するピクセル数、 n_{cl} はクラス数、 t_i はクラス i の総ピクセル数とする。ただし、背景と線画についてはカウントしない。イラストは線画と背景の領域が多く含まれていることから、精度の差が現れないためである。また、pix2pix の出力形式は RGB であり、パレットを使用していない。そのため各ピクセルを使用したパーツのパレットの色に変換する形で評価を行った。変換のルールはパレットの色で一番近い色を採用する方式を選択した。

また、提案方式に対する比較評価として、代表的なセマンティックセグメンテーション方式の 1 つである U-net を用いたレイヤ分け方式の評価を行った。U-net での評価において、入力画像を 256 px × 256 px の線画とし、教師ラベルを 256 px × 256 px のレイヤ分け画像とした。また、U-net の仕様にあわせ、提案方式と同じ学習データを学習の際にラベルに変換する形で使用した。また、入力チャンネルは 3、出力チャンネルは 4 パーツに線と背景を追加する形で 6 とした。U-net の損失関数は softmax_cross_entropy を選択し、最適化方式は Adam とした。また、パラメータは $\alpha = 0.0002$, $\beta_1 = 0.5$, $\beta_2 = 0.999$ とした。

表 1 に、自動レイヤ分け方式の精度評価の結果について示す。また、提案方式と U-net を使用してレイヤ分けを行った結果を図 8 に示す。

表 1 の結果から、pix2pix を用いた自動レイヤ分け方式の全体の精度が 81.6% となることが確認できた。その後、後処理により、すべてのパーツにおいて精度が向上していることを確認することができた。また、後処理後の

全体の精度として 84.8%となることが確認できた。また、比較対象とした U-net の精度は 82.5%となり、pix2pix の 81.6%よりも高く、pix2pix に後処理を組み合わせた提案方式の 84.8%よりも低い結果となった。

3.8 自動レイヤ分け処理結果の考察

表 1 の結果から、後処理を追加することで、すべてのパーツにおいて自動レイヤ分けの精度が向上していることを確認することができ、全体の精度として 84.8%となることが確認できた。また、パーツ単位の精度について、後処理後で最も精度が高くなったパーツは髪であり 90.0%となった。最も精度が低かったのが目であり 64.2%となった。しかし、目は後処理によって 7.8%と最も精度が向上していることが確認できた。

後処理によって精度が向上した理由として、pix2pix の結果、一部の色が不安定であった領域が後処理によって単色になったために、本来のレイヤ分け画像の仕様に近づいたためであると考えられる。特に首や前髪、耳といったパーツどうしの境界に当たるパーツは不安定になることが多く、これらのパーツを単色にしたことで精度が向上したと考えられる。また、目の領域について精度が大きく上昇した理由としては、領域が小さくパーツの数も少ないことから、後処理の影響を大きく受けたことが考えられる。

提案方式の精度について、比較対象である U-net と比較を行った。その結果、pix2pix の精度は 81.6%となり U-net の精度である 82.5%よりも低かったが、提案方式である後処理と組み合わせることで U-net の精度を超える 84.4%まで改善することが確認できた。パーツ単位に提案方式と U-net の精度を比較したところ、髪、服は提案方式の方が高く、肌、目は U-net の方が高く、総合的に提案方式の方が高くなった。また、図 8 を用いて出力画像を用いて提案方式と U-net の比較を行った。その結果、図 8 の上部のケースでは、U-net は肌、服、髪が混同しているピクセルが首付近に多く発生したことが原因で、提案方式よりも精度が低くなったと考えられる。

3.9 自動レイヤ分け処理結果の誤りに関する考察

次に、自動レイヤ分け処理に誤りについて考察を行ったところ、大きく以下の 2 種類の誤りが存在することが分かった。その結果について図 9 に示す。

(1) パーツの間違い

パーツの間違いとは、ある閉領域の色がパーツと対応していない場合である。図 9 ①では首が肌パーツであるにもかかわらず、服パーツであると認識されている。これは pix2pix において間違っ

(2) 色のあふれ

色のあふれは、入力線画において線が大幅に途切れて



図 9 自動レイヤ分け結果の誤り

Fig. 9 Error examples of output image.

いる場合に発生する領域外まで色がはみだす現象である。図 9 ②では襟部分の線画が途切れていることによって、背景と襟の色が同一となってしまっている。発生原因としては、大幅な線の跡切れによって、自動レイヤ分け方式の Step2 において隙間を塞ぎきれず、Step3 において閉領域の認識に失敗したためと考えられる。

4. 自動レイヤ分けシステム：Smart Layer Splitter の開発

本章では、3 章で提案した自動レイヤ分け方式を適用した自動レイヤ分けシステムである Smart Layer Splitter の開発について述べる [22], [23], [24]。

4.1 開発方針

まず、システムの開発方針について以下に示す。

- システムの開発にあたり、3.1 節の基本方針に従い、自動レイヤ分け処理で発生した誤りは、ユーザが手動で修正する方針とする。
- 既存のグラフィックソフトよりも簡易かつ簡単に修正することができる UI とする。
- 既存のグラフィックソフトの機能を結合することで、操作回数を減らすような UI を提案する。
- 絵を描くデバイスの多様化に対応し、Web アプリケーションで構築する。ただし、PC と液晶ペンタブレットでの使用を想定する。
- レイヤ分けに対応するパーツは、提案方式に従い、髪、肌、目、服、背景の 5 種類とする。
- 本研究において対象とする自動レイヤ分け処理の誤りは、パーツの間違いとする。その理由は、パーツの間違いは領域の色を修正するのみでユーザ側で容易に対応できるためである。

4.2 システム構成

本システムは PC と液晶ペンタブレットから構成される。本システムで使用した PC のスペックについて表 2 に示す。

また、液晶ペンタブレットは Huion 社の kamvas PRO 12 を採用した。画面のタッチ機能はついておらず、ペンのサイドにボタンが付属している。開発したシステムの外観について図 10 に示す。

次に、開発したシステムのソフトウェア構成について

表 2 PC のスペック

Table 2 Specification of PC.

プロセッサ	intel Core i7-7500U
システムクロック	2.70 GHz
RAM	8,192 MB
ビデオカード	NVIDIA GeForce 940MX 4GB
OS	Windows 10 PRO 64bit

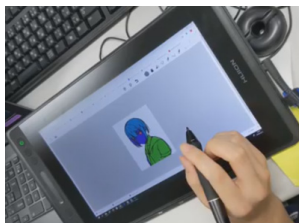


図 10 システムの外観

Fig. 10 Appearances of system.

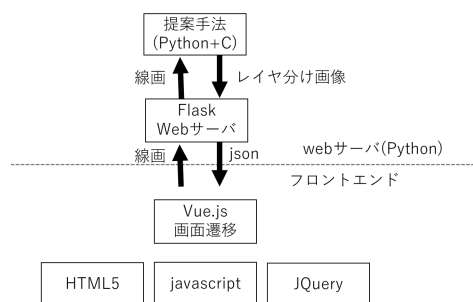


図 11 ソフトウェア構成

Fig. 11 Software configuration.

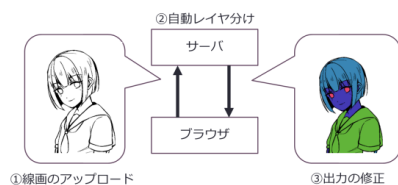


図 12 レイヤ分け処理の流れ

Fig. 12 Flow of layer splitting process.

図 11 に示す。本システムは Vue.js と HTML5, JavaScript, jQuery を用いて SPA (Single Page Application) 形式の Web アプリケーションとして構築した。画面のスタイルは Bootstrap を、Web サーバは Python のライブラリである Flask を用いて構築した。自動レイヤ分け処理は、Python と C を用いて、線画を自動的にレイヤ分け画像に変換する処理を実装した。

4.3 レイヤ分け処理の流れ

本システムを用いたレイヤ分け処理の流れを図 12 に示す。ユーザはブラウザを用いて、レイヤ分けを行う線画を選択し、ユーザ操作によりサーバにアップロードする (図 12 ①)。次に、サーバで自動レイヤ分け処理を実行す



図 13 UI の概要

Fig. 13 Overview of UI.

る (図 12 ②)。最後に、サーバは自動レイヤ分けの処理結果をブラウザ側に戻す。処理結果に、パーツの誤りなどの修正すべき誤りが存在する場合は、ユーザがブラウザ上で修正を行う (図 12 ③)。修正が終わった際、現在の状態をサーバに送信することでレイヤ分けした PSD ファイルをユーザに返す。

4.4 UI の概要

本システムの UI の概要を図 13 に示す。UI 上部のツールバーは、ブラシの変更や色の変更を行うことができる。ツールバーのアイコンは、左から①PSD 出力、②拡大/縮小/アンドゥ、③色の選択/ツールの変更/ブラシサイズの変更である。

(1) PSD 出力

PSD 出力ボタンをクリックすることで、修正結果を反映したレイヤ分け画像を PSD ファイルとして出力することができ、既存のグラフィックソフトで開くことでそのまま着色作業を継続することができる。

(2) 拡大/縮小/アンドゥ

拡大と縮小により、キャンバスの表示サイズを変更することができる。また、アンドゥは編集結果を 1 つ前に戻すことができる。

(3) 色の選択/ツールの変更/ブラシサイズの変更

現在の表示色を示している色の選択アイコンをクリックすると、色の選択ウィンドウが表示される。色の選択ウィンドウでは、パーツの色一覧を表示する。また、色の選択を行うことで、編集対象が対応した色のレイヤに自動的に切り替わる。これによって、既存のグラフィックソフトでは、色の変更と編集レイヤの変更を行う必要がある操作を、本システムでは色の変更のみで行うことができる。

ツールの変更では、左から、手のひらツール、塗りつぶしツール、投げ縄選択ツール、矩形選択ツール、ペン選択ツール、ペンツールが配置されている。また、ペン選択ツールとペンツールを選択している場合のみ、隣にブラシサイズの変更ができるスライダーが表示され、ブラシサイズの変更を行うことができる。ブラシサイズの範囲は 1px から 200px である。



図 14 ペン選択ツールの使用方法

Fig. 14 How to use the pen selection tool.

4.5 ツールの概要

次にツールの概要について説明する．領域の修正には塗りつぶしツール，投げ縄選択ツール，矩形選択ツール，ペン選択ツール，ペンツールを使用する．塗りつぶしツール，ペンツールについて，使用法は既存のグラフィックソフトと同様の機能であるが，ユーザが描画動作を行った際，現在選択している色に対応したレイヤに対しては領域の描画を行い，ほかのレイヤに対しては先ほど描画した領域の消去を行っている点異なる．これによって1ストロークで対象レイヤに対する描画とそれ以外のレイヤに対する消去を同時に行うことができる．

次に既存のグラフィックソフトと異なる機能を持つ選択ツールについて説明する．選択ツールはペン選択ツール，投げ縄選択ツール，矩形選択ツールが存在する．これらのツールは使用した際に領域の内部を自動で認識し，その内部の領域のみを塗りつぶす．この機能によって，イラストに存在する耳や髪の毛の隙間などの小さい複数の閉領域を修正する際に，はみ出してしまうことを防ぐ効果がある．また，前述した塗りつぶしツールとペンツールと同様に，選択した色に対応するレイヤのみ描画し，それ以外のレイヤに対しては削除を行う．

図 14 にペン選択ツールの使用方法について示す．このとき，肌色を描画色として選択している状態である．ペン選択ツールを用いて中央の図のように耳の領域を大まかに囲うことで，耳の内部領域を自動で認識し，内部領域に対して描画を行う．このとき，修正前に耳の領域内に塗られていた髪色は髪色レイヤから削除される．自動レイヤ分けの結果に対する修正は図 14 のように細かいパーツが多いことが想定されるため，このツールをメインに使用することを想定としている．

5. システム評価

5.1 評価方法

開発した自動レイヤ分けシステム：Smart Layer Splitter のシステム評価を行い，システムの有効性について確認した．具体的には，実験協力者として大学生 23 人に対して，本システムと既存のグラフィックソフトを用いて実際にレイヤ分け作業を行う評価実験を実施した．比較対象としたグラフィックソフトは株式会社セルシスの CLIP STUDIO PAINT PRO を選定した．その理由は，本ソフトは 800 万



図 15 評価実験での課題

Fig. 15 Work items of evaluation experiments.

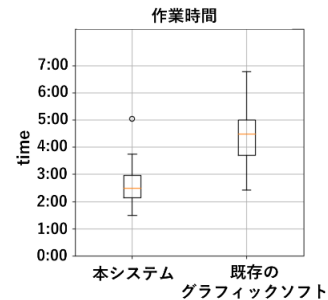


図 16 作業時間の評価結果

Fig. 16 Evaluation result of working time.

人以上のクリエイターが使用しているソフトである点と，イラストコミュニケーションサービス pixiv においても使用率が第 1 位である点を考慮し，比較対象として適切であると判断した．

ユーザに対して事前アンケートを行い，日常的にデジタルイラストを描くと回答した実験協力者 7 人を経験者，それ以外の 16 人を初心者に分類した．その後，実験協力者全員に本システムと既存のグラフィックソフトを用いたレイヤ分け作業の作業時間と操作回数の計測を行い，その後アンケートを用いたユーザビリティ評価を行った．

図 15 に評価実験での課題について示す．具体的な作業内容は，本システムの入力画像の条件を満たす線画を本システムに入力し，自動レイヤ分け処理の結果を修正してレイヤ分け画像を作成する作業を実施する．既存のグラフィックソフトでは，同じ線画からレイヤ分け画像を作成する作業を実施する．評価実験で利用した線画は，図 15 の入力線画であり，画像サイズは 416px × 512px，アンチエイリアスなしとした．アンチエイリアスなしとした理由は，アンチエイリアスありの線画は本システム，既存のグラフィックソフトともにレイヤ分け作業が煩雑になるためである．また，本システムを使用し，自動レイヤ分け後に修正の必要となる領域は 7 カ所であった．

5.2 作業時間の評価

システムと既存のグラフィックソフトを用いてレイヤ分け作業を行った際に計測した作業時間について図 16 に示す．また，経験別の作業時間について図 17 に示す．

5.3 操作回数の評価

本システムと既存のグラフィックソフトを用いてレイヤ

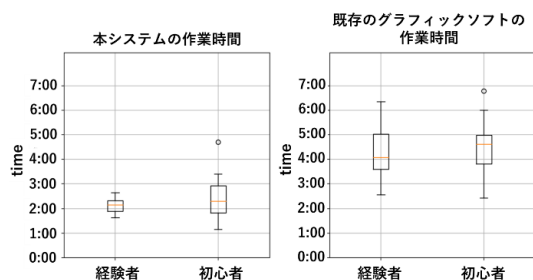


図 17 経験別の作業時間の評価結果

Fig. 17 Evaluation results of working time by experience.

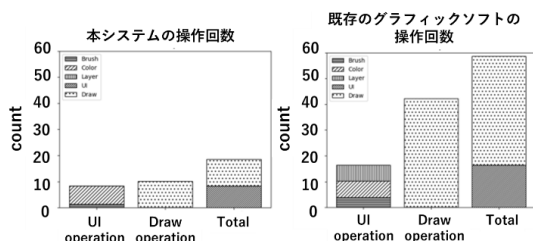


図 18 操作回数の評価結果

Fig. 18 Evaluation results of the number of operations.

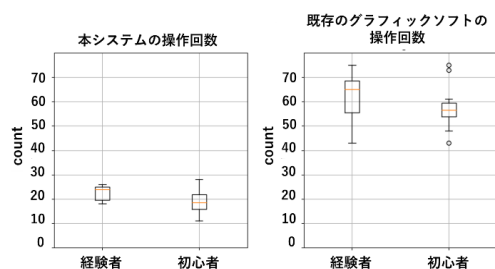


図 19 経験別の操作回数の評価結果

Fig. 19 Evaluation results of the number of operations by experience.

表 3 ユーザビリティの評価結果

Table 3 Evaluation results of questionnaire.

アンケート内容	結果
最後までレイヤ分けを行うことができましたか？	はい(23) いいえ(0)
システムは使いやすいですか？	5(6) 4(14) 3(2) 2 1
システムの操作に迷うことがありましたか？	はい(9) いいえ(14)
自動で塗り分けした結果について満足ですか？	はい(18) いいえ(5)
実際にあったらイラスト制作の一部で使いたいですか？	はい(22) いいえ(1)
ボタンの位置やツールバーの位置は適切でしたか？	適切(14) その他(9)

分け作業を行った際に計測した操作回数について図 18 に示す。また、経験別の操作回数について図 19 に示す。

5.4 ユーザビリティの評価

アンケートを用いたユーザビリティの評価結果について表 3 に示す。また、作業時間と操作回数と同様に、アンケートの結果をもとに経験者と初心者に分けた際の評価結

表 4 経験者の評価結果

Table 4 Evaluation results of questionnaire by experienced persons.

アンケート内容	結果
最後までレイヤ分けを行うことができましたか？	はい(7) いいえ(0)
システムは使いやすいですか？	5(1) 4(4) 3(2) 2 1
システムの操作に迷うことがありましたか？	はい(3) いいえ(4)
自動で塗り分けした結果について満足ですか？	はい(4) いいえ(3)
実際にあったらイラスト制作の一部で使いたいですか？	はい(6) いいえ(1)
ボタンの位置やツールバーの位置は適切でしたか？	適切(3) その他(4)

表 5 初心者の評価結果

Table 5 Evaluation results of questionnaire by beginners.

アンケート内容	結果
最後までレイヤ分けを行うことができましたか？	はい(16) いいえ(0)
システムは使いやすいですか？	5(5) 4(11) 3 2 1
システムの操作に迷うことがありましたか？	はい(6) いいえ(10)
自動で塗り分けした結果について満足ですか？	はい(14) いいえ(2)
実際にあったらイラスト制作の一部で使いたいですか？	はい(16) いいえ(0)
ボタンの位置やツールバーの位置は適切でしたか？	適切(11) その他(5)

果をそれぞれ表 4、表 5 に示す。

6. 考察

6.1 作業時間の評価に関する考察

(1) 実験協力者全体の考察

図 16 において、実験協力者全体の作業時間の平均は、本システム：2分 31 秒、既存のグラフィックソフト：4分 44 秒となり、作業時間を 46.8%短縮することができた。また、本システムでの自動レイヤ分け処理時間の 10 回計測時の平均は 20.2 秒であったため、自動レイヤ分け処理時間を加算した結果は、2分 51 秒となる。以上より、自動処理の時間を加えても既存のグラフィックソフトよりも作業時間を 39.8%短縮できることを確認した。これは本システムでは提案方式の出力に存在する誤りのみ修正すればよい点と、自動レイヤ分けシステムの UI について、レイヤと色の選択を同時に行うことができることが理由であると考えられる。

(2) 作業経験に関する考察

図 17 において、経験者の作業時間の平均は、本システム：2分 16 秒、既存のグラフィックソフト：4分 18 秒となった。初心者の作業時間の平均は本システム：2分 24 秒、既存のグラフィックソフト：4分 30 秒となった。本システムを使用した場合は、経験者のほうが初心者よりも作業時間の平均が 8 秒少なく、既存のグラフィックソフトを

使用した場合は、経験者のほうが初心者よりも作業時間の平均が12秒少なくなる結果となった。理由としては、経験者はグラフィックソフトや液晶タブレットの扱いに慣れており、操作に迷わなかったためと考えられる。また、ペイントソフトに慣れている経験者の中には、右クリックに割り当てられたスポイト機能を本システムと既存のグラフィックソフトで積極的に使用していることも見受けられたことから、作業の短縮を積極的に図っていると考えられる。

6.2 操作回数の評価に関する考察

(1) 実験協力者全体の考察

図18において、実験協力者全体の操作回数の平均は、本システム：18.5回、既存のグラフィックソフト：58.9回となり、操作回数を68.6%と大幅に削減することができた。各項目の内訳は、ブラシや色の変更を含むUI操作について、本システム：8.3回、既存のグラフィックソフト：16.4回となり、49.4%削減することができた。また、描画操作について、本システム：10.2回、既存のグラフィックソフト：42.2回となり、75.9%削減することができた。これは、本システムでは自動レイヤ分け処理後の修正のみ行えばよい点と、レイヤの選択と色の選択の機能を統合したことで、1回の動作で切り替えることができる点が理由であると考えられる。

(2) 作業経験に関する考察

図19において本システムの操作回数の平均は、経験者：19.4回（UI操作：8.0回、描画操作：11.4回）、初心者：18.1回（UI操作：8.4回、描画操作：9.6回）となった。

また既存のグラフィックソフトの操作回数の平均は、経験者：61.6回（UI操作：16.9回、描画操作：44.7回）、初心者：57.3回（UI操作：16.2回、描画操作：41.1回）となった。

既存のグラフィックソフトを使用した場合、経験者は初心者よりも操作回数が多い傾向が見られた。特に描画操作については経験者の方が3.6回多い結果になった。この結果について、実験協力者がレイヤ分けを間違えた際に、修正する方法の違いによって、この差が現れたのではないかと考えられる。たとえば、ある経験者はレイヤ分けのミスを発見した際に、間違えたパーツのレイヤに移動した後、消しゴムツールで間違えた部分を消し、正しいパーツのレイヤを変更して再び色を塗る処理を行った。一方、初心者はレイヤ分けを間違えた際に、間違えたパーツのレイヤに塗ってはならない色を置いて修正するといった行動が実験協力者数人に見られた。このような修正方法は想定されておらず、誤った修正方法である。このような修正方法の違いから経験者は初心者よりも操作回数が増える傾向が見られたと考えられる。

本システムでは、レイヤの選択と色の選択機能は統合し

ているため、上記のような問題は起こらない。そのため、経験者と初心者の操作回数は既存のグラフィックソフトよりも差が少なくなったのではないかと考える。

6.3 ユーザビリティの評価に関する考察

(1) 実験協力者全体の考察

表3において、本システムを使用した実験協力者全員が最後までレイヤ分けを行うことができた。また、使いやすさについても5段階評価で4という結果が得られた。コメントでは、他のパーツを塗る際にレイヤを切り替えしなくて良いので便利、ツールの持ち替えが楽といったポジティブなコメントを得ることができた。一方、実験協力者がペン選択ツールを使用した際、目的の部分が塗りつぶせない問題が数回確認した。評価実験の際、目的の領域が塗れなかったことで戸惑った様子を見せており、アンケートにおいても、塗りつぶしたい領域をすべて囲うのは面倒だというコメントがあった。

またUIについては、ブラシサイズの変更バーが右端に配置されていることから、画面端にあることで使いにくい、左利きでは使いにくいといったコメントも得られた。また、9人の実験協力者が操作に迷ったと答えた。これはUI上にツールの説明や作業のフローなどを提示するなどの改善が必要だと考えられる。液晶タブレットは利き手によって使いやすいUIの配置が異なるため、操作パネルを自由に変更することができる柔軟なUIが必要であると考えられる。

UIについて、色の選択ウィンドウは、「×」ボタンか色の選択ウィンドウ外をクリックすることで閉じることができ、開いているときには描画操作を行うことができない。しかし、ウィンドウが閉じていない状態で、描画操作を行う様子が多くみられ、描画できないことに実験協力者は戸惑った様子であった。このことから、色の選択ウィンドウについてのUIも改良していく必要があると考えられる。

(2) 作業経験に関する考察

経験者について、使いやすさの5段階評価は3.86となり、実験協力者全体と比べ0.14ポイント減少した。また、自動で塗り分けした結果について満足かの設問については、4人が満足と答えたが、3人が満足ではないと答えた。ふだんからイラストを描いており、レイヤ分けを行っている経験者は、自動レイヤ分け機能に要求する精度が高いためこのような結果になったと考えられる。ボタンの位置やツールバーの位置は適切かについては、4人が不満をかかえていることが分かった。これはふだん使用しているグラフィックソフトと比較を行っていることからこのような結果になったと考えられる。コメントとして、ブラシの調節ウィンドウはペンに追従した方が使いやすい、左右に色の変更などの操作アイコンを配置して欲しいという意見を得た。これは既存のグラフィックソフトのUIと同様であり、

経験者は本システムにおいても既存のグラフィックソフトのような UI を求めていると考えられる。

初心者について、使いやすさの 5 段階評価は 4.31 となり、実験協力者全体と比べ 0.31 ポイント上昇した。また、自動で塗り分けした結果について満足かの設問については、14 人が満足と答えた。ボタンの位置やツールバーの位置は適切かについては、5 人が何かしらの不満をかかえていることが分かった。実際のコメントとして、液晶タブレットは左利きだと使いにくく、カーソルが見えないといった意見を得た。これは本システムの UI の問題に加えて、ハードウェアの液晶タブレットについての意見も含まれていると考えられる。特にカーソルが見えない問題については、実験協力者のペンの角度と頭の位置、液晶ペンタブレットの位置が問題であることから、液晶ペンタブレットに不慣れであることで発生していると考えられる。

6.4 成果物に対する考察

ある実験協力者に本システムと既存のグラフィックソフトを用いてレイヤ分けを行った際の成果物の例を図 20 に示す。成果物に対して評価した結果、本システム、既存のグラフィックソフトともに許容範囲ではあるものの間違いが存在することを確認した。確認した間違いはパーツの間違い、塗り忘れ、はみだしである。それぞれについて発生原因について考察する。

(1) パーツの間違い

パーツの間違いとは、たとえば前髪部分は肌パーツであるにもかかわらず、髪パーツを間違えて塗っていることである。これは既存のグラフィックソフト、本システムを使用したときの両方に見られたが、本システムを使用した際に頻出していたことを確認した。これは実験協力者が本システムの自動レイヤ分けの結果から、間違っている領域を発見できずに保存ボタンを押してしまったことが原因であると考えられる。また、パーツの間違いは前髪と眉の境界パーツで頻出していた。そのことから、髪と肌の色が似ている点や、線画において眉が前髪にかかっており、髪と肌の境界が認識しにくい点が原因であると考えられる。解決策として、システム上でパーツの境界線を強調し、ユーザの注意を向けることでパーツの間違いが減るのではないかと考えられる。

(2) 塗り忘れ

塗り忘れとは、髪の毛の先端や小さな領域が塗り切れていないものを指す。これは本システムと既存のグラフィックソフトどちらにも確認できた。これは実験協力者が細かな領域を認識できなかった際に発生すると考えられる。認識できない理由としては、線画付近は領域が小さくなる傾向があり、色の間違いに気づかないことがあげられる。解決方法として、レイヤ分けに使用する色を認識しやすい色にすることで、塗り忘れが少なくなると考えられる。また、



図 20 成果物に存在する誤り

Fig. 20 Errors present in deliverables.

システムの機能として周辺パーツと色が異なっている領域を認識し、警告を表示するなど、ユーザに対して注意を向けさせるようにすることで改善できると考えられる。

(3) はみだし

はみだしとは、領域から色がわずかにはみだしたものを指す。これは既存のグラフィックソフトのみに確認できた。原因としては、ブラシツールを用いて修正を行った際に、線画から色がはみ出してしまい、はみだした色を削除せずに保存したことが原因であると考えられる。また、はみだしについては本システムにおいては発生しなかった。本システムはブラシツールを使用せず、領域を自動で認識するペン選択ツールのみでレイヤ分けを行うことができることから、このような結果になったと考えられる。

7. おわりに

本論文では、デジタルイラスト制作におけるレイヤ分け作業を対象として、pix2pix の出力結果に対して後処理を行うことで、レイヤ分け処理を自動化する方式について提案した。提案方式について精度評価を行った結果、Mean Accuracy で平均 84.8%の精度が得られた。次に、提案方式を用いた自動レイヤ分け処理において誤りが発生した場合、発生した誤りを手動で修正する UI を持った自動レイヤ分けシステム：Smart Layer Splitter を開発し、システム評価を行った。

その結果、既存のグラフィックソフトと比較して、作業時間を 39.8%短縮できるとともに、操作回数を 68.6%削減できることが確認でき、システムの有効性について確認することができた。また、ユーザビリティの評価の結果、本システムを使用した実験協力者全員が最後までレイヤ分けを行うことができ、使いやすさについても 5 段階評価で 4 という結果が得られた。

今後の課題としては、機能面については、アルゴリズムの見直しによる自動レイヤ分け処理に対するさらなる精度の向上、対応できなかった自動レイヤ分け処理の誤りである色のあふれに対応した UI の開発、ユーザにとって視認しやすいレイヤ分けに使用する色の検討などがあげられる。また、実施したアンケートでは、システムの操作に迷った実験協力者が 9 名いたことから、システム上に操作の手順をわかりやすく提示することや、ユーザが認識できなかった

た領域を強調するなどの UI の改良についても検討していく予定である。

謝辞 本研究を推進するにあたり、有益な助言をいただいた東京電機大学システムデザイン工学部倉持卓司教授、阿部清彦准教授、および開発したシステムの評価実験に協力いただいた東京電機大学システムデザイン工学部の学生各位に感謝する。

参考文献

- [1] Isola, P., Zhu, J.-Y., Zhou, T. and Efros, A.A.: Image-to-Image Translation with Conditional Adversarial Networks, *Proc. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR2017)*, pp.5967–5976 (2017).
- [2] Preferred Networks: PaintsChainer, 入手先 https://petalica-paint.pixiv.dev/index_ja.html.
- [3] 米辻泰山：線画自動着色サービス「PaintsChainer」について、映像情報メディア学会誌, Vol.72, No.5, pp.353–357 (2018).
- [4] Ronneberger, O., Fischer, P. and Brox, T.: U-Net: Convolutional Networks for Biomedical Image Segmentation, *Proc. 18th International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI2015)*, LNCS, Vol.9351, pp.234–241, Springer (2015).
- [5] Zhang, L., Li, C., Wong, T.-T., Ji, Y. and Liu, C.P.: Two-stage Sketch Colorization, *ACM Trans. Graphics*, Vol.37, Article No.6 (2018).
- [6] Furusawa, C., Hiroshiba, K., Ogaki, K. and Odagiri, Y.: Comicolorization: Semi-Automatic Manga Colorization, *Proc. ACM SIGGRAPH Asia 2017*, Technical Briefs (2017).
- [7] Hensman, P. and Aizawa, K.: cGAN-based Manga Colorization Using a Single Training Image, *14th IAPR International Conference on Document Analysis and Recognition, Workshop MANPU2017*, pp.72–77 (2017).
- [8] Ramassamy, S., Kubo, H., Funatomi, T., Ishii, D., Maejima, A., Nakamura, S. and Mukaigawa, Y.: Pre- and Post-Processes for Automatic Colorization using a Fully Convolutional Network, *Proc. ACM SIGGRAPH Asia 2018*, Posters (2018).
- [9] Simo-Serra, E., Iizuka, S. and Ishikawa, H.: Real-Time Data-Driven Interactive Rough Sketch Inking, *ACM Trans. Graphics*, Vol.37, No.4, Article No.98 (2018).
- [10] Koyama, Y. and Goto, M.: Decomposing Images into Layers with Advanced Color Blending, *Computer Graphics Forum*, Vol.37, No.7, pp.397–407 (2018).
- [11] Sekachev, B., Zhavoronkov, A. and Manovich, N.: Computer Vision Annotation Tool: A Universal Approach to Data Annotation, available from <https://software.intel.com/content/www/us/en/develop/articles/computer-vision-annotation-tool-a-universal-approach-to-data-annotation.html>.
- [12] Microsoft: VoTT (Visual Object Tagging Tool), available from <https://github.com/microsoft/VoTT/>.
- [13] 渡邊 優, 阿部清彦, 阿倍博信: pix2pix を用いたデジタルイラスト制作におけるレイヤ分け作業の自動化, 情報処理学会第 81 回全国大会, 4Q-02 (2019).
- [14] 渡邊 優, 阿倍博信: デジタルイラスト制作の色塗り工程における自動レイヤ分け方式, *Visual Computing 2019 P40* (2019).
- [15] Adobe Systems: The Photoshop File Format, available from <https://www.adobe.com/devnet-apps/photoshop/fileformatashtml/#50577409.72092/>.
- [16] pixiv: pixiv, available from <http://www.pixiv.net/>.
- [17] Preferred Networks: Chainer, available from <https://chainer.org/>.
- [18] Preferred Networks: chainer-pix2pix, available from <https://github.com/pfnet-research/chainer-pix2pix/>.
- [19] Krähenbühl, P. and Koltun, V.: Efficient Inference in Fully Connected CRFs with Gaussian Edge Potentials, *Advances in Neural Information Processing Systems 24 (NIPS2011)* (2011).
- [20] Wang, P.S.P. and Zhang, Y.Y.: A Fast and Flexible Thinning Algorithm, *IEEE Trans. Computers*, Vol.38, No.5, pp.741–745 (1989).
- [21] Shelhamer, E., Long, J. and Darrell, T.: Fully Convolutional Networks for Semantic Segmentation, *IEEE Trans. Pattern Analysis and Machine Intelligence*, Vol.39, No.4, pp.640–651 (2017).
- [22] 渡邊 優, 阿倍博信: pix2pix を用いたデジタルイラスト制作における自動レイヤ分けシステム, 情報処理学会デジタルコンテンツクリエーション研究会第 24 回研究会, Vol.2020-DCC-24, No.27, pp.1–8 (2020).
- [23] 渡邊 優, 阿倍博信: Smart Layer Splitter: デジタルイラスト制作の色塗り工程における自動レイヤ分けシステム, 映像情報メディア学会映像表現・芸術科学フォーラム 2020, Vol.44, No.10, AIT2020-146, pp.317–320 (2020).
- [24] 渡邊 優, 阿倍博信: Smart Layer Splitter: pix2pix を用いたデジタルイラスト制作の色塗り工程における自動レイヤ分けシステム, 情報処理学会マルチメディア, 分散, 協調とモバイル DICOMO2020 シンポジウム, pp.1584–1593 (2020).



渡邊 優 (学生会員)

2019 年東京電機大学情報環境学部卒業。現在、同大学院情報環境学研究科修士課程に所属。AI を活用したデジタルコンテンツの創作活動支援に関する研究に従事。



阿倍 博信 (正会員)

1988 年慶應義塾大学理工学部計測工学科卒業, 1990 年同大学院理工学研究科修士課程修了, 同年三菱電機株式会社入社。以来, グループウェアシステム, マルチメディア応用システムの研究開発に従事。2018 年, 東京電機大学システムデザイン工学部情報システム工学科教授。2005 年慶應義塾大学理工学研究科後期博士課程修了。博士 (工学)。電子情報通信学会, 映像情報メディア学会, 画像電子学会, 教育システム情報学会各会員。