彩色済み参考画像から生成される領域情報を用いたアニメ線画の自動彩色手法

南谷大輔[†1]　　米澤弘毅[†2]

近年、アニメやマンガを代表とする日本のサブカルチャーは世界中で人気が高まっており、様々な媒体で放送されている。アニメ制作における彩色の工程は、重要であると同時に多くの労力を必要とする工程でもある。本論文では、彩色済みアニメ画像を用いて線画を自動で彩色する手法について取り扱い、彩色された画像の確認と検証を行った。線画の自動彩色手法を開発することにより、我々は彩色にかかる労力を減少させることを目標にしている。提案手法は、線画に存在する領域が参考画像のどの領域と対応するのかを推測することにより彩色を行う。

Colorization of anime line art using region information obtained from reference images

Daisuke Nanya[†1]　　Kouki Yonezawa[†2]

1. はじめに

アニメの制作には多くの時間と人的資源を必要とし、多数の工程を経由することにより制作される。工程には、脚本の制作や絵コンテの制作など作品の展開を導く工程もあれば、原画の制作やコンピュータグラフィックスを用いたキャラクターの動きの構築など実際に放送される画像を制作する工程もある。近年のコンピュータ技術の発達によって、アニメの制作にもコンピュータグラフィックスやモーションキャプチャの技術を導入するなどコンピュータを用いた工程も増えてきた。しかし、全ての工程がコンピュータに置き換わってはおらず、原画の作成など手作業の工程も多数存在する。

アニメの彩色についてアニメ制作会社である株式会社トリガー[1]へインタビューを行った。現在、彩色（専門用語では仕上げという）を担当する部署では若年層が少なくなりつつあり、担当者の高齢化という課題があることがわかった。そのため、自動彩色ツールによる仕上げ作業の効率化は非常に有用なものであると確認できた。また、現在日本のアニメに適応できるような自動彩色のツールは業界では使われていないことも確認できた。

続いて仕上げに至るまでの工程について説明する。最初に手作業で紙に絵を描き原画を制作する。原画には複数の色の線を描く。黒色は仕上げ後も残る線（専門用語では実線という）、赤色は基準となる色（専門用語ではノーマルという）よりも明るい（専門用語ではハイライトという）領域、青色はノーマルよりも暗い（専門用語では影という）領域を表している。実線は仕上げ後も残るが、ハイライトの線や影の線は仕上げ後には残らない。作品やカットによっては複数の影が存在する場合もある。この場合、明るい影から暗い影に向かって一号影、二号影というように影の前に付く数字が増えていく。

完成した原画から動画を作成する。動画の制作は原画をスキャナーでデジタルデータに変換し、原画の線を清書する（クリーンアップ）作業である。クリーンアップでは原画を参考に実線、ハイライトの線、影の線を液晶タブレット上で描き、二値画像を制作する。この際、仕上げの工程でベタ塗り機能を利用するため、線が途切れることなくあらゆる領域が閉領域になるよう注意しながら作業を行う。図 1 に原画と動画の例を示す。

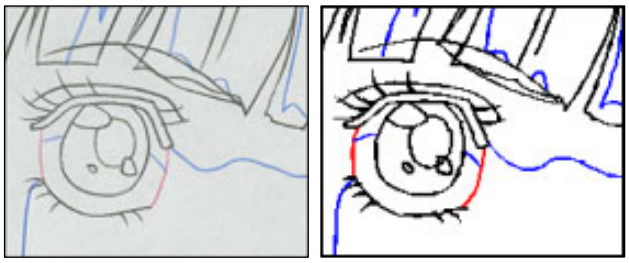

図 1　原画（左）と動画（右）の例（共に[2]より引用）

仕上げは動画の制作の後で行われ、ペイントツールを用いて行い、アニメの制作現場においては主に RETAS STUDIO[3]というペイントツールが利用されている。仕上げでは制作された動画を参考に領域ごとにベタ塗りを行う。使用する色はあらかじめ指定されており、キャラクターごとに昼のシーン、夜のシーンなどカットの状況に合わせてハイライト、ノーマル、一号影、二号影というように複数の色が指定されていることが多い。仕上げは動画に描かれている領域の数にもよるが、一枚当たり 30 分ほどかかることもある。図 2 に仕上げまでの工程の概要図を示す。

†1 名城大学　理工学研究科
Meijo University
†2 名城大学　情報工学部
Meijo University

ⓒ2024 Information Processing Society of Japan

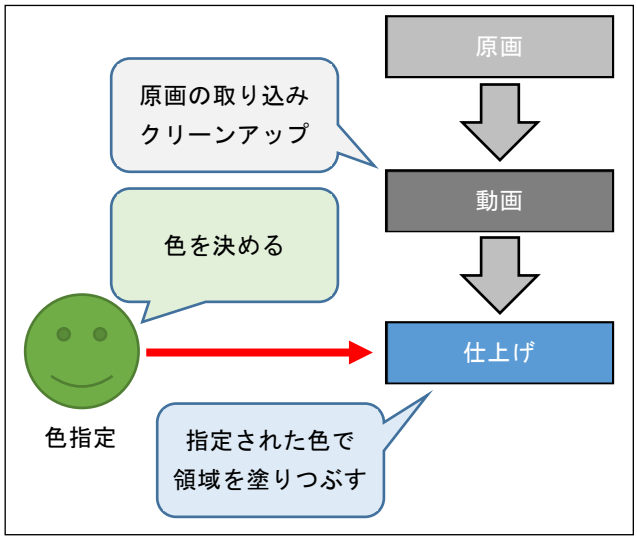

図 2　仕上げまでの工程の概要図

本研究では仕上げの工程にかかる労力の削減を目的とする。提案手法ではアニメ画像が多数の領域から構成されていることに着目した。線画に存在する領域が参考画像のどの領域と対応するのかを推測し、対応すると推測できる領域の色で線画を彩色するシステムを構築した。トリガー社とのインタビューから、正解率は 90 パーセント以上かつ動画一枚にかかる仕上げの時間は 10 秒以内に抑えることを目標にしているが、提案手法では面積ベースの計算で90%を超える正解率が得られた。

2. 関連研究

画像をカラー化する手法は実画像を対象とした研究が多数で、手描きの絵を対象とした研究は多くない。その中でも手描きの絵を対象とし、彩色済み参考画像を用いて絵をカラー化する研究はイラスト調の手法が多く、アニメを対象とした手法はほとんどない。ここではマンガとイラストのカラー化に関する手法について紹介する。

2.1 グラフマッチングを用いた手法

佐藤ら[4]はマンガ調の画像をカラー化することを目的として、マンガの絵が多数の領域で構成されている点に着目し、領域同士のノードを推測することにより領域の対応付けを行う手法を提案している。このシステムへは彩色済み参考画像と線画を入力し、参考画像に含まれる領域同士のノードを用いて線画をカラー化する。具体的には、参考画像と線画のそれぞれにおいて領域同士のノードを作成する。作成された線画のノードが参考画像のノードと対応するのかを推測することにより、領域同士の対応付けを行い、参考画像の色で彩色する。

この手法のメリットとして、線画は参考画像に含まれる色のみで彩色される点が挙げられる。また、機械学習を利用しないため、著作権に配慮した大量の学習データを用意する必要がない。

2.2 ニューラルネットワークを用いた手法

Cao ら[5]はイラスト調の画像の線画を入力として、機械学習の手法の一つである Generative Adversarial Networks（敵対的生成ネットワーク）を用いて彩色する手法を提案している。機械学習を用いた手法ではニューラルネットワークを利用し、線画を畳み込むことでその特徴量を利用したものが多くを占める。この手法では特徴量を抽出する際に近年様々な研究で利用されている Attention 機構を導入することで精度向上を図っている。

この手法のメリットとして、参考画像と線画の構図が異なっても彩色できる点、参考画像を入れ替えることで様々な色使いを基にした彩色ができる点が挙げられる。これは大量の画像を用いてネットワークを学習することにより、構図の異なる場合においても対応できるようになっている。しかしながら、生成画像がアニメに向いていると保証できないデメリットがある。また、著作権に配慮した学習データを大量に用意する必要がある点、GPU の利用が必要である点、学習に時間がかかる点などのデメリットが挙げられる。

3. 提案手法

提案手法は関連研究と異なる点がいくつかある。まず、グラフマッチングを用いた手法とは異なり領域同士のノードの推測ではなく、領域を直接推測する点である。次に、ニューラルネットワークを用いて直接の画像生成は行わない点である。そのため、参考画像に含まれない色が出力されることはなく、アニメの画像に向いた一つの領域で使用される色は一色のみとなる画像を出力できる。最後に、機械学習を行わない点である。

提案手法のメリットは機械学習を使用しないため、機械学習を利用する際に度々問題になる学習データの著作権に関する問題は発生しない。また、画像の出力に長時間かかることはない。

提案手法では動画は多数の領域で構成されており、一つの領域で使われる色は一色のみであることに着目した。画像の色を直接予測するのではなく、動画の領域が彩色済み参考画像のどの領域と対応するのかを予測する手法となっている。そのため、参考画像に含まれる色のみで彩色するため、参考画像に存在しない色で彩色されることはない。

3.1 システムの概要

システムは彩色したい動画と彩色済みの参考画像を入力として受け取り、彩色された動画を出力する。図 3 に構築したシステムの概要を示す。

ⓒ2024 Information Processing Society of Japan

システムの目標は彩色済み参考画像と同一カット内の動画の画像を彩色することである。システムでは動画と参考画像における領域の対応付けを推測することにより動画を彩色する。まずは動画と参考画像から領域情報を抽出する。その後、線画に存在する領域がどの参考画像の領域と対応しているのかを推測し、その結果を基に参考画像の領域の色で線画の彩色を行う。ただし、このままでは境界画素が残ったまま画像が出力されてしまうので、境界画素の除去を行う。

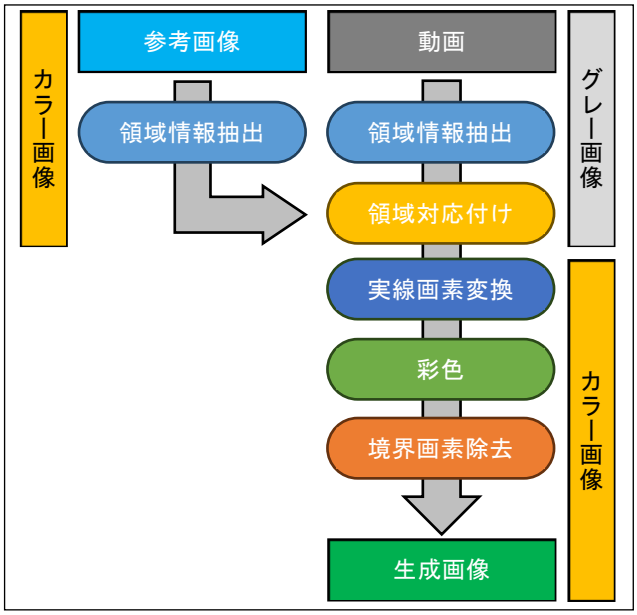

図 3　構築したシステムの概要図

3.2　領域情報抽出

領域情報抽出の工程では画像の持つ領域の情報を抽出し保存する。抽出する情報は動画と参考画像の場合で共通する情報もあれば異なる情報もある。共通する情報は領域のID、x座標、y座標、画素数、重心x座標、重心y座標、最小x座標、最小y座標、最大x座標、最大y座標である。ID は領域を識別するための番号である。参考画像の場合はこれらに加えて、領域の R, G, B の画素値を保存する。領域の画素値は彩色の工程でベタ塗りを行う際に利用する。

領域情報を抽出する手順は動画と参考画像のどちらも同じである。最初に画像中に含まれない画素値を決める。その後、対象領域を探索し、領域を抜き出す。そして領域情報を取り出しデータとして保存し、対象領域を実線画素に置き換える。この手続きを画像に含まれる全ての画素の探索が終わるまで繰り返す。

3.2.1 選択画素値の決定

領域情報を抽出した後、領域を抜き出す際に抜き出すべき領域を一時的に置き換えるための画素値を決める。この画素値を選択画素値ということにする。選択画素値は画像に含まれない画素値で設定する。以降のサンプル画像では線画の場合は 200、参考画像の場合は[128, 128, 128]の画素値で選択画素値を設定し説明を行う。

3.2.2 領域探索

画像を左上から右へ順番に画素の確認を行い、抜き出すべき領域かどうかの探索を行う。動画の場合は画素が背景画素であれば、参考画像の場合は画素が実線画素以外であれば、領域抜き出しの工程へ進む。そうでなければ確認した画素の右へ行きラスタ走査を画像全体で行う。

3.2.3 領域抜き出し

3.2.2 節で述べた抜き出すべき領域の存在する座標に選択画素値で塗りつぶし処理を行い、領域を抜き出す。塗りつぶす画素値は 3.2.1 で説明した画像中に存在しない画素値で行う。図 4 に動画における領域選択の前と後の例を、図 5 に参考画像における領域選択の前と後の例を示す。

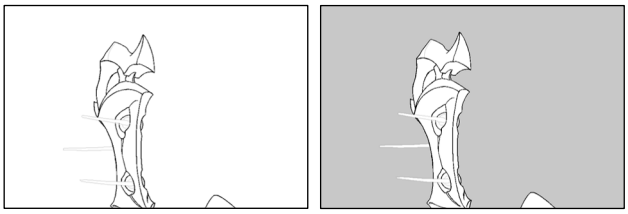

図 4　動画における領域選択の前（左）と後（右）の例

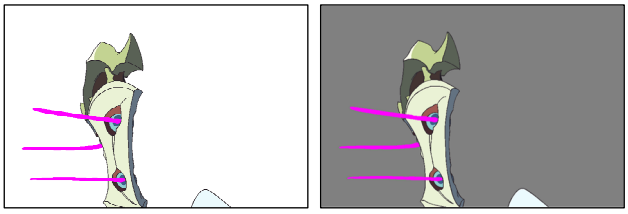

図 5　参考画像における領域選択の前（左）と後（右）の例

領域を選択した後に選択画素値となっている画素のみを抜き出す。抜き出す領域を白色、抜き出さない領域を黒色として領域を抜き出す。図 6 に動画における領域抜き出しの前と後の例を、図 7 に参考画像における領域抜き出しの前と後の例を示す。

図 6　動画における領域抜き出しの前（左）と後（右）の例

ⓒ2024 Information Processing Society of Japan

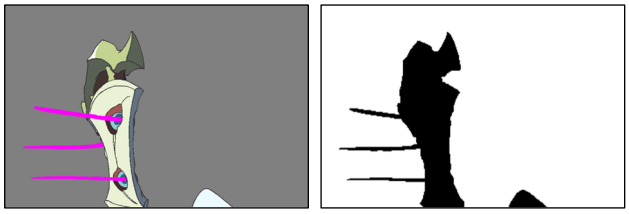

図 7　参考画像における領域抜き出しの前（左）と後（右）の例

3.2.4 領域情報の取り出し

3.2.3 節で述べた領域として抜き出された画像の情報を取り出し保存する。取り出す情報は画像の中で白色となっている画素の数、重心x座標、重心y座標、最小x座標、最小y座標、最大x座標、最大y座標を取得する。図 8 に最小と最大の座標の表し方を示す。

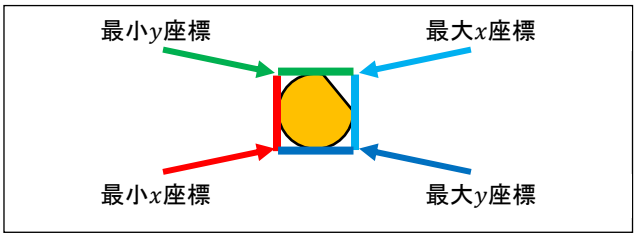

図 8　領域における最小と最大の座標

参考画像の場合はこれらに加えて選択された画素の色も保存する。これらの情報は CSV ファイルで保存される。表 1 に領域情報を含んだファイルのイメージを示す。

表 1　領域情報を含んだファイルのイメージ

ID	面積	重心x	重心y	・・・	y
1	80000	400	10	・・・	0
2	1000	50	20	・・・	0
・ ・ ・	・ ・ ・	・ ・ ・	・ ・ ・		・ ・ ・
100	4000	900	800	・・・	750

3.2.5 対象領域を実線画素に置き換え

ここでは再び抽出した領域画素を実線画素に置き換える。動画と参考画像どちらの場合でも実線画素に置き換える。この処理を行うことで次の領域情報を保存する際にこれまでに抜き出した領域情報の加味を防ぐ。

3.3 領域対応付け

領域対応付けの工程では 3.2 節の工程で得られた情報を基に動画と参考画像における領域の対応付けを行う。対応付けは動画に含まれる調べたい領域の情報を基に参考画像に含められる全ての領域情報を調査することで、調べたい領域が参考画像のどの領域に最も似ているのかを導き出す。この処理を動画に含まれる調べたい領域の数だけ繰り返し、動画に含まれる全ての領域を参考画像の領域と対応付ける。

調べる際に利用する指標は領域の面積、重心座標、バウンディングボックスの 4 隅の座標（左上座標、右上座標、左下座標、右下座標）の 3 つである。図 9 に領域におけるバウンディングボックスの 4 隅の座標について示す。これらの座標は最小x座標、最小y座標、最大x座標、最大y座標から構成される。

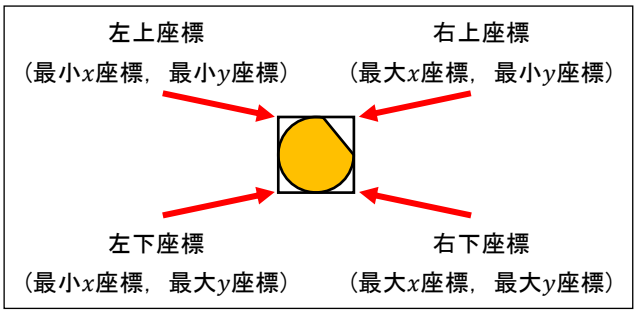

図 9　バウンディングボックスの 4 隅の座標

動画と参考画像の画素数が異なることを考慮し、領域情報を調べる前に値の調整を行う。具体的には、動画と参考画像それぞれの領域における画素数、重心x座標、重心y座標、最小x座標、最小y座標、最大x座標、最大y座標を 0 から 1 で正規化する。

ここからの領域対応付けでは表 2 を参考画像の領域情報、表 3 を線画における調査領域の情報として具体例を交えて説明する。

表 2　参考画像の領域情報の例

ID	面積	重心(x, y)	最小(x, y)	最大(x, y)
1	0.25	(0.2, 0.2)	(0.1, 0.1)	(0.6, 0.4)
2	0.05	(0.7, 0.4)	(0.5, 0.3)	(0.8, 0.6)
3	0.05	(0.6, 0.7)	(0.3, 0.4)	(0.8, 0.9)

表 3　線画における調査領域の情報の例

面積	重心(x, y)	最小(x, y)	最大(x, y)
0.2	(0.4, 0.3)	(0.1, 0.2)	(0.7, 0.6)

3.3.1 対応付けた領域の面積の算出

ここでは領域の面積を指標として、調べたい領域が参考画像のどの領域の面積に近いのか調べる。具体的には、参考画像に含まれる領域の面積から調べたい領域の面積の差を取り、その差を絶対値に変換する。この処理を参考画像に含まれる領域の数繰り返し、絶対値が最も小さい領域を 1 として小さい順に順位を付けていく。同じ値の場合は同一順位を付ける。表 3 を調査領域の情報、表 2 を参考画像の領域情報とする場合の面積を指標とした順位算出の例を表 4 に示す。

表 4　面積を指標とした順位算出の例

ID	面積の差	面積の差の絶対値	順位
1	0.05=(0.25-0.2)	0.05	1
2	-0.15=(0.05-0.2)	0.15	2
3	-0.15=(0.05-0.2)	0.15	2

ⓒ2024 Information Processing Society of Japan

3.3.2 重心間の距離の計算

ここでは領域の重心間の距離を指標として、調べたい領域が参考画像のどの領域に場所が近いのか調べる。重心位置の算出には重心x座標、重心y座標を利用し、参考画像に含まれる全ての領域と調べたい領域の重心の距離を算出する。重心の距離が最も小さい領域を 1 として小さい順に順位を付けていく。順位のつけ方は 3.3.1 節と同じである。表 5 に表 3 を調査領域の情報、表 2 を参考画像の領域情報とする場合の重心距離を指標とした順位算出の例を示す。

表 5　重心距離を指標とした順位算出の例

ID	重心距離の二乗	順位
1	0.05=(0.04+0.01)=((0.2-0.4)^2+(0.2-0.3)^2)	1
2	0.10=(0.09+0.01)=((0.7-0.4)^2+(0.4-0.3)^2)	2
3	0.20=(0.04+0.16)=((0.6-0.4)^2+(0.7-0.3)^2)	3

3.3.3 バウンディングボックスの 4 隅の座標間の距離

ここでは領域のバウンディングボックスの 4 隅の座標間の距離を指標として、調べたい領域が参考画像のどの領域に場所が近いのか調べる。

4 隅の座標を利用することで、重心だけでは測ることのできない縦長や横長の領域など領域の特徴を領域の推測に用いることができる。調べたい領域と参考画像の領域において 4 隅の座標同士の距離を算出する。これら 4 隅の座標間の距離の合計が最も小さい領域を 1 として小さい順に順位を付けていく。順位のつけ方は 3.3.1 節と同じである。表 6 に表 3 を調査領域の情報、表 2 を参考画像の領域情報とする場合の重心距離を指標とした順位算出の例を示す。

表 6　4 隅の座標を指標とした順位算出の例

ID	合計距離	順位
1	$\sqrt{0.01}+\sqrt{0.02}+\sqrt{0.05}+\sqrt{0.05}$	1
2	$\sqrt{0.17}+\sqrt{0.02}+\sqrt{0.16}+\sqrt{0.01}$	2
3	$\sqrt{0.08}+\sqrt{0.05}+\sqrt{0.13}+\sqrt{0.10}$	3

3.3.4 対応する領域の決定

ここでは算出された 3 つの順位を用いて調べたい領域が参考画像のどの領域と最も似ているのかを決定する。決定には算出された 3 つの順位の積が最も小さい参考画像の領域を動画の領域に最も似ているとして対応付ける。表 7 に算出された 3 つの順位を利用した累計順位算出の例を示す。

表 7　3 つの順位を利用した累計順位算出の例

ID	順位			
	面積	重心	4 隅	累計
1	1	1	1	1
2	2	2	2	8
3	2	3	3	18

表 7 では累積順位が最も小さくのなる領域は ID が 1 である。そのため、表 3 の領域情報を持った領域は ID が 1 の領域と対応付けられる。

3.4 実線画素変換

実線変換の工程では動画に描かれている黒色の実線を色指定で決められた色に変換する処理を行う。色指定で決められる実線の色はカラーであることが多いため、この工程で線画はグレースケール画像からカラー画像に置き換える。

3.5 彩色

彩色の工程では 3.3「領域対応付け」で作成された情報を基に動画の対象領域へベタ塗りを行う。

3.6 境界画素除去

境界画素除去の工程では色のベタ塗りに使用した境界画素を除去し、その周囲の色に置き換える処理を行う。境界画素があった画素には周囲 8 画素の中から最もよく使われる画素に置き換える。この時再び境界画素に置き換えられないようにするため、境界画素を除いた画素から最もよく使われる画素に置き換える。境界画素が画像の端にある場合は周辺 8 画素は使用できないため、周囲 3 画素や周囲 5 画素を用いる。この処理は境界画素が画像中に存在しなくなるまで繰り返す。

しかし、この手法では境界画素が複数にまたがる領域の間に存在する性質上どの領域の色に置き換えるのが正解かわからない。そのため、境界画素が望んだ領域の色とならない可能性もあるが、境界画素は塗りつぶす領域に比べてそこまで大きくなく、問題にしないこととした。

4. 評価検証

システムの評価検証には実際にアニメとして制作された「映画リトルウィッチアカデミア魔法仕掛けのパレード」[6]のデータセット[7]を用いて行う。このデータセットには脚本、絵コンテ、キャラクターデザイン、原画などアニメの制作に関する多数のデータから構成されている。この中から彩色済み画像を使用して評価検証を行った。本来であれば、彩色前の動画を用いてシステムの評価検証を行うべきであるが、データセットに動画は含まれていなかった。そのため、動画に相当する画像を彩色済み画像から生成することで評価検証を行った。

4.1 彩色済み画像から動画の生成

ここでは彩色前の動画になるべく近づけられるよう図 10 のように変換を行い、彩色済み画像から動画を生成する。画像に含まれる輪郭を目立たせることで線画を生成するツールは画像編集ソフトなどで多数存在するが、ほとんどが白色と黒色に変換するものであり、実線と領域の境界を区

ⓒ2024 Information Processing Society of Japan

別して線画を生成するようなツールは見つからなかった。そのため、アニメの動画に該当する画像を生成するツールを自分たちで作成した。

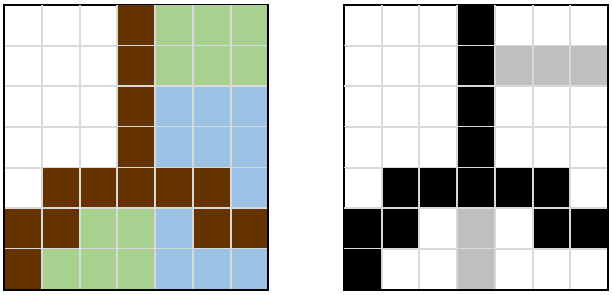

図 10　動画へ変換前（左）と変換後（右）の画像

画像中の色について説明する。変換前の画像において、白色は背景色、茶色は実線、緑色はカラー色 1、青色はカラー色 2 を表す。変換後の画像において、白色は背景またはベタ塗りさせる領域、灰色は彩色後には残らない色が変わる境界画素、黒色は実線を表す。

評価検証における動画の生成では画素値を黒線が 0、灰色が 128、白色が 255 とする。これは利用するデータセットの中に画素値が[128, 128, 128]の画素値が含まれていないためである。また、境界画素はハイライトの線や影の線を示すため、本来であれば青色や赤色の色を使用する。しかし、利用するデータセットには動画が含まれておらず、どのような色が利用されていたか分からないため、今回は実線によらない領域の境界を全て灰色にすることとした。本来とは異なる色で領域の境界を示すことになるが、提案手法では境界画素の色は推論に用いていないため、結果に影響を及ぼすことはない。

彩色済み画像から動画を生成するには実線となる画素値の設定、実線の描画、境界画素の描画という三つの工程を経由することで行う。彩色済み画像はカラー画像、変換後の動画はグレースケール画像である。

しかし、画像の解像度を変更せずに彩色済み画像をそのまま線画化すると、図 11 の右側の青色の領域のように同じ領域が分断されてしまうことがある。領域が分断されてしまうと彩色の工程でベタ塗りを行う際に不都合が生じる可能性があるため、意図しない領域の分断は回避する必要がある。

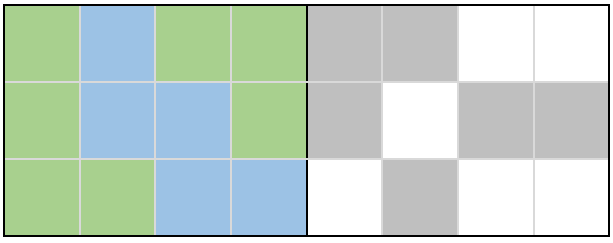

図 11　解像度変更前の彩色済み画像（左）と線画（右）

そこで、同じ領域が分断されないようにするため、画像における縦横の画素数をそれぞれ 2 倍にし、解像度を 4 倍にしてから線画化を行う。図 12 のように解像度を 4 倍に変更した後で線画を生成すると、同じ領域の画素が分断されるてしまうことはなくなった。このような領域の分断は領域の縦や横の画素数が極めて小さいときに発生する。

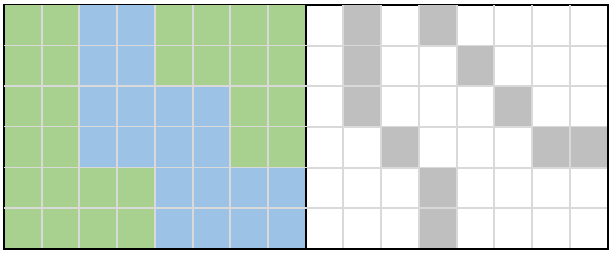

図 12　解像度変更後の彩色済み画像（左）と線画（右）

4.1.1 実線となる画素値の設定

実線となる画素値の設定では参考画像の実線の画素値を設定する。これは色指定で画素値が決められているので、その値で設定する。

4.1.2 実線の描画

実線の描画では彩色済み画像に含まれる実線のみを取り出し、新しい画像に書き込む処理を行う。

具体的には、まず彩色済み画像と同じ画素数の白色のグレースケール画像を用意する。次に、4.1.1「実線となる画素値の設定」で設定した画素値と彩色済み画像に含まれる同じ画素値の場所に黒色の画素値をグレースケール画像に書き込む。

4.1.3 境界画素の描画

境界画素の描画では実線ではない領域が変わる境界画素に灰色を書き込む処理を行う。

具体的には、ある画素とその右隣の画素またはその下隣の画素に差がみられる場合、4.1.2「実線の描画」で作成したグレースケール画像のある画素の場所に灰色の画素値を書き込む。ただし、その差が実線との差の場合は書き込まない。

しかし、このままでは図 13 の左側のように描画されてしまいペイントの塗りつぶし機能で必要のない画素も灰色として描画されてしまう。これは境界が斜めの場合に発生する。図 13 の右側のように灰色の画素が必要最低限となるように境界画素を描画した後に余分な境界画素の除去を行う。

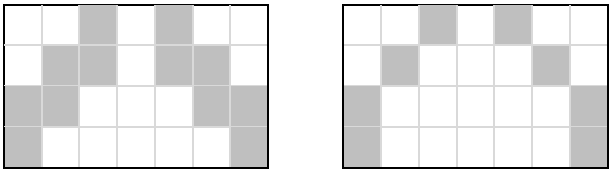

図 13　境界画素描画の改良前（左）と改良後（右）の画像

余分な境界画素の除去は、調査画素、調査画素の左隣の画素、調査画素の上隣の画素全てが灰色の場合に調査画素を白色にする。加えて、調査画素、調査画素の右隣の画素、

ⓒ2024 Information Processing Society of Japan

調査画素の上隣の画素全てが灰色の場合にある画素を白色にする。これを画像中の灰色の画素全てで行う。

4.2 出力画像の評価検証

評価検証では 10 個の異なるカットを用いて 20 枚の出力画像を生成した。いずれも動画は参考画像の 1 つ前と後ろのフレームであり、キャラクター全体が動いている画像である。動画は彩色済み画像を 4.1.1「実線となる画素値の設定」の方法で生成された画像を使用し、参考画像は彩色済み画像をそのまま使用する。生成した 20 枚において正しく領域を推測できた正解率は領域ベースで 72.66%、面積ベースで 97.00%であった。その中からいくつか例を示す。

一つ目にキャラクターの顔が映っていないカットの彩色結果を示す。図 14 にシステムへ入力する動画と参考画像を、図 15 にシステムによる出力画像とその正解画像を示す。ここでの動画は参考画像の 1 つ後ろのフレームである。

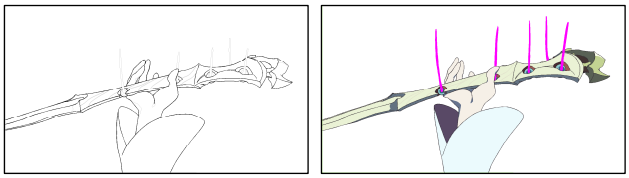

図 14　一つ目の評価で用いる線画（左）と参考画像（右）

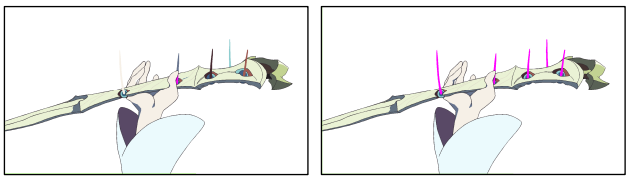

図 15　図 14 を入力としたシステムによる出力画像（左）とその正解画像（右）

図 15 の出力画像の正解率は領域ベースで 63.29%、面積ベースで 99.05%である。出力画像は背景、服、手、杖など面積の大きな領域では高い精度で領域を推定できていることが確認できた。しかし、杖の中や杖の外に伸びる細長い領域など面積の小さな領域で領域推定ミスが確認できた。

二つ目にキャラクターの全体が描かれているカットの彩色結果を示す。図 16 にシステムへ入力する動画と参考画像を、図 17 にシステムによる出力画像とその正解画像を示す。ここでの動画は参考画像の 1 つ後ろのフレームである。

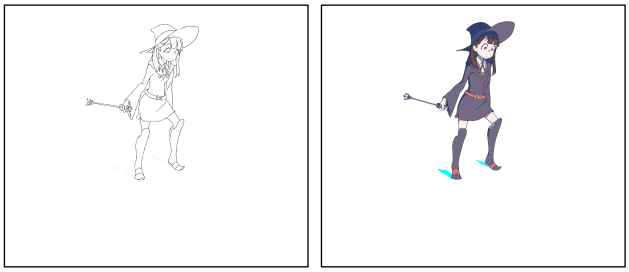

図 16　二つ目の評価で用いる線画（左）と参考画像（右）

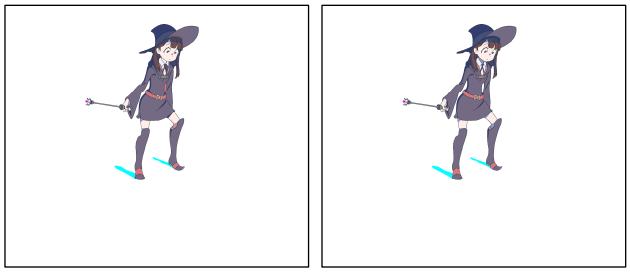

図 17　図 16 を入力としたシステムによる出力画像（左）とその正解画像（右）

図 17 の出力画像の正解率は領域ベースで 77.33%、面積ベースで 99.83%である。出力画像は背景や服など面積の大きな領域では高い精度で領域を推定できていることが確認できる。しかし、服の中や帽子の中にあるような小さな領域、髪の領域など面積の小さな領域で領域推定ミスが確認できた。

三つ目にキャラクターの全体が描かれているカットの彩色結果を示す。

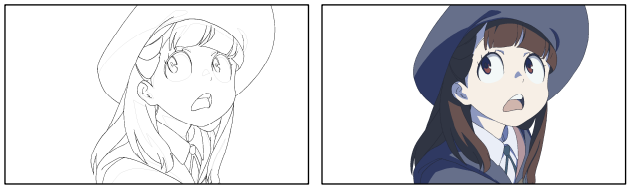

図 18　三つ目の評価で用いる線画（左）と参考画像（右）

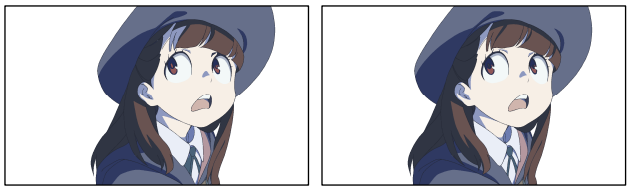

図 19　図 18 を入力としたシステムによる出力画像（左）とその正解画像（右）

図 19 の出力画像の正解率は領域ベースで 74.42%、面積ベースで 99.22%である。出力画像は背景や顔など面積の大きな領域では高い精度で領域を推定できていることが確認できる。しかし、髪や服の中の小さな領域など面積の小さな領域で領域推定ミスが確認できた。

正解率は領域ベースよりも面積ベースの方が高いことから、全体的に面積の大きな領域の方が正しく領域を推測されやすいことが読み取れる。

5. おわりに

本研究では、アニメの彩色における彩色済み参考画像を用いた線画の自動彩色手法を行うシステムの構築を行った。

また、実際にアニメとして制作されたデータセット[7]を用いて評価検証を実施した。大きい領域の予測は高い精度で彩色できるが、小さい領域の予測は精度が低く彩色の色を間違えやすいと確認できた。そのため、アニメの仕上げの工程に提案手法を用いる場合、主に小さな領域の色を手

ⓒ2024 Information Processing Society of Japan

作業で修正する作業が発生することになる。しかし、実行時間は領域の推測は数秒で完了するものの、画像の領域情報の抽出にかかるには 10 秒以上時間を要するため、実行時間をより短くする必要がある。

今後は領域における形の類似度を導入することで、推測の精度を向上させていくと同時に、彩色全体にかかる時間を短くできるように手法の効率化を図っていきたい。

謝辞 本研究は株式会社トリガー様よりデータセットを提供して頂きました。この場を借りて深く御礼申し上げます。

参考文献

[1] Trigger, Inc., TRIGGER official website, (2024). https://www.st-trigger.co.jp/.
[2] CELSYS,Inc., trace function, RETAS STUDIO.Net (2020). http://www.retasstudio.net/products/traceman/trace/.
[3] CELSYS, Inc., RETAS STUDIO.net, (2020). http://www.retasstudio.net/.
[4] K. Sato, Y. Matsui, T. Yamasaki, K. Aizawa, Reference-based manga colorization by graph correspondence using quadratic programming, in: SIGGRAPH Asia 2014 Technical Briefs, Association for Computing Machinery, 2014: pp. 1–4. https://doi.org/10.1145/2669024.2669037.
[5] Y. Cao, H. Tian, P.Y. Mok, Attention-Aware Anime Line Drawing Colorization, (2023). https://doi.org/10.48550/arXiv.2212.10988.
[6] TRIGGER, Y. Yoshinari, GOOD SMILE COMPANY, “Little Witch Academia: The Enchanted Parade” official website, (2015). http://littlewitchacademia.jp/.
[7] National Institute of Informatics, Trigger dataset, IDR Informatics Research Data Repository (2022). https://www.nii.ac.jp/dsc/idr/trigger/.

ⓒ2024 Information Processing Society of Japan