写真からの対話的イラスト生成

Interactive Illustration Generation from Pictures

岩田 大輝† 関 啓夢† 豊浦 正広† 茅 暁陽†

Taiki IWATA† Hiromu SEKI† Masahiro TOYOURA† and Xiaoyang MAO†

† 山梨大学大学院医学工学総合教育部

† Graduate School of Medicine and Engineering, University of Yamanashi

1. はじめに

ありのままの情景を記録し伝達するには写真が有用であるが，その情景のなかの一部の情報だけを伝達したい場合や，そこに制作者の意図を盛り込みたい場合には，イラストの方がむしろ優れている．なぜならば，イラストは必要な情報だけを強調し，余分な情報を省略して描くことができるからである．本研究では，写真からユーザの意図に合わせて対話的にイラストを生成する新しい手法を提案する．イラストは植物学の文献に用いられている挿絵や，医学の教科書にある人体や組織の説明図，指名手配犯の似顔絵など，情報の効果的な表現が求められる様々な場面で利用されている．

写真から効果的なイラストを生成するには，必要な情報のみを選択し，対象に合わせて描画スタイルを変化させることで，対象の特徴を際立たせる必要がある．情報の取捨選択は通常ユーザの主観に大きく依存する．また，選択した情報をどのように表現するかもユーザごとに異なる．これまで提案されたコンピュータによるイラスト生成技術は，フリーハンド描画を行なうペイントソフトに代表される対話型システムと，画像フィルタリングや 3 次元シーンを入力として利用する自動システムの 2 つに大別される．ペイントソフトの場合はユーザが手作業でイラストをゼロから描くため，意図を反映できる反面，書き終えるまでに時間がかかり負担が大きい．一方自動生成法では，情報の取捨選択におけるユーザの意図やユーザ独自の描画スタイルを反映することが難しい．Kalnins らは 3D モデルに対して，その表面や輪郭の一部分に描くだけで残りの表面や輪郭に対して同じようにレンダリングを行う直感的な手法を提案した[1]．彼らのユーザの直感性や対話性を重視したアプローチを、本研究の写真からのイラスト生成でも利用するべきであると考える．また，異なるイラスト生成のアプローチとして Kang らは写真上でユーザが手動でトレースした線を自動補正することでイラストの描画効率を上げることに成功した[2]．しかし，Kang らの方法では描きたい部分をすべてトレースする必要があるため，ユーザにとっての負担がやはり大きい．

提案手法はユーザが写真上で描きたい領域や輪郭の一部に描画例を与えることで，ユーザの描画したい領域や輪郭線を自動選択し，選択部分へユーザの描画スタイルを反映させることができる．これにより，従来のイラスト生成手法に比べ，ユーザへの負担や作業時間を大幅に軽減しながら，ユーザの意図に合った情報の取捨選択とユーザ独自の描画スタイルを反映したイラストの生成が可能である．

2. 提案手法

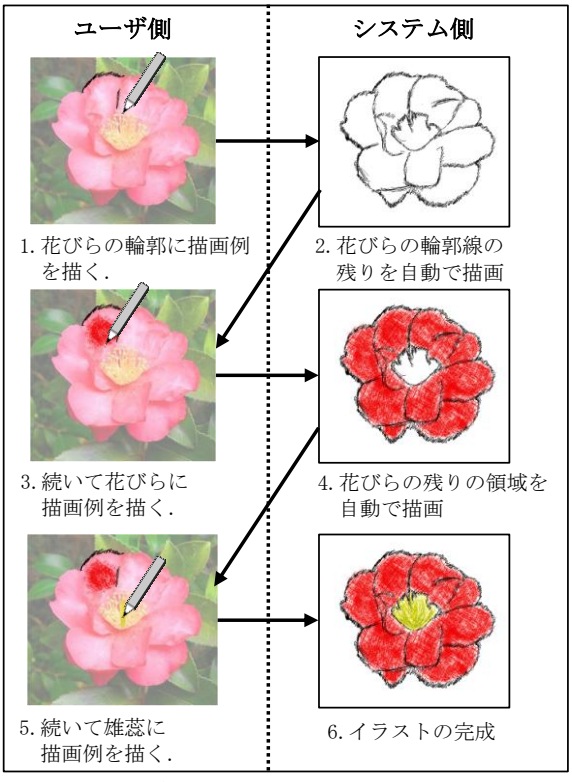


図 1 提案システムのイラスト生成

図 1 に示すように，提案手法ではユーザがシステムと対話しながらイラストを作成する．ユーザが描画例を一つ与えると，システムがユーザの描画したい輪郭線または領域を画像上から自動選択し，選択部分へユーザの描

画例の描画スタイルを反映してイラストを生成する．ユーザは描画例を選択的に描き足していき，必要な情報をすべて選択・描画された時点で作業を終了する．その結果，必要な情報のみが選択され，不必要な情報が省略されたイラストが生成される．本研究では画像特性の違いから，描画対象を輪郭線と内部領域の二つに分けて取り扱う．

2.1. 描画したい領域・輪郭線の自動選択

領域や輪郭線を自動選択するためには何らかの特徴を抽出する必要があるが，本研究では画像内のローカルな領域や輪郭線付近の領域のテクスチャ特徴を抽出する．テクスチャとは濃度や色の 2 次元的なパターンの集合であると定義でき，その一様性によって特徴付けることができる．領域の場合は，テクスチャが同じであれば同じ領域であるとして自動選択する．一方，輪郭線の場合には，輪郭線を挟んだ二つの領域のテクスチャが同じであれば同じ輪郭線として選択する．輪郭線の場合に 2 つの領域について特徴を抽出するのは，輪郭線は一方の領域とそれと異なる特徴を持つ領域の境界として定義できるためである．

また，ユーザの描画したい領域や輪郭線を選択するためには，あらかじめ画像内の領域の特徴や輪郭線の特徴を抽出しておき，ユーザが描いた位置の領域や輪郭線の特徴と比較して，似たような特徴をもつ領域や輪郭線を抽出すればよい．また，これらの特徴をあらかじめ同じ特徴を持つような要素同士を同じ集合にまとめておけば，ユーザが描くたびに特徴を比較する必要がなく，さらに高速に自動選択処理できる．

2.1.1. 領域の特徴抽出とクラスタリング

画像内のローカルな領域に対してテクスチャ特徴を抽出し，同じようなテクスチャ特徴を持つ領域ごとの集合にまとめるには，抽出した特徴のクラスタリングを行えばよい．本研究では Mean Shift 法[3]を用いる．Mean Shift 法は画像内の同じような特徴を持つ領域ごとにクラスタリングを行う．

領域の特徴にテクスチャ特徴を用いることで，画像内の同じようなテクスチャを持つ領域をまとめることができる．領域内からテクスチャ特徴を取得するには図 2 に示すような注目点を中心とした N×N の近傍画素を特徴とすればよい．

2.1.2. 輪郭線の特徴抽出とクラスタリング

輪郭線の特徴を抽出し，特徴空間でのクラスタリング[3]を行う手順について述べる．まず，その前に輪郭線となる画素を画像から抽出する必要がある．画像内から輪郭線となる画素を抽出するために，エッジ検出を行う．

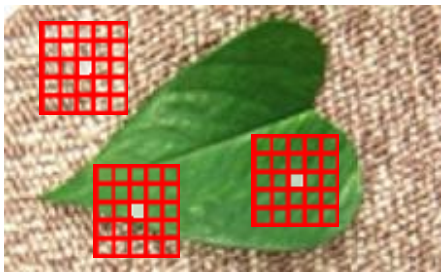


図 2 テクスチャ特徴の抽出

（1）輪郭線となる画素を画像内から抽出

エッジは隣接する画素の輝度値の変化が激しい箇所に現れる．輪郭線は領域と領域の境界であるから，2 つの領域がほとんど同じような輝度値を持つケースを除いて，エッジとして検出することができる．ただし，画像内にはノイズが含まれているため，そのままエッジ検出を行うとノイズの影響を顕著に受けてしまう．本研究では画像内のエッジを保存しつつノイズ除去を行うために，バイラテラルフィルタ[4]によるノイズ除去処理を行う．バイラテラルフィルタは画素値が急激に変化するエッジ部分を保持したまま画像の平滑化を行うフィルタであり，エッジ検出を行う前処理として有用である．

バイラテラルフィルタによるノイズ除去後にエッジ検出を行う．エッジ検出には Canny オペレータ[4]によるエッジ検出法を用いる．Canny オペレータの特徴は，曲線形状のエッジ抽出において効果が大きく，エッジ画素の連結性が高いという特徴がある．

（2）輪郭線となる画素の連結情報の取得

Canny オペレータによるエッジ検出で検出されたエッジは画素群の集まりであり，連結情報を持っていない．輪郭線に対してスタイルを反映するときに，輪郭線上の画素を順にたどる必要があるので連結情報を前もって取得しておく必要がある．検出したエッジ上の画素の連結情報を取得するには，注目画素の隣接画素を探索ルールに従って順に追跡しメモリに記録していく．追跡には 8 近傍の探索窓を用いて行い，その連結情報をチェインコード[4]で符号化しメモリに記録する．追跡の手順は以下のとおりである．エッジ検出した結果に対してラスタ走査をしてエッジ上の画素を見つける．ただし，追跡済みの場合は無視し，ラスタ走査を続ける．エッジ上の画素はチェインコードの記述ルールに従って追跡していき，追跡した画素には追跡済みのフラグを立てる．追跡する画素がなくなったら終了する．これらの処理をラスタ走査が終了するまで行う．

（3）輪郭線となる画素の特徴抽出

次に画像内の輪郭線の特徴の定義と抽出方法について述べる．輪郭線となる画素の特徴は周辺 2 つの領域から

調べればよい．2 つの領域の特徴には 2.1.1 節と同様に領域に含まれているテクスチャ情報を用いることする．特徴ベクトルの次元数は，2 つの近傍領域に含まれる画素の数に等しくなる．2 つの近傍領域の大きさは，近傍領域の一辺の大きさを N としたとき，2×N×N である．従って特徴ベクトルの次元数は，2×N×N 次元である．輪郭線となる画素の上記で述べた特徴ベクトルを抽出する手順は，注目画素に連結している画素から接線を求め，注目画素を通る接線の法線を引く．法線上の 2 点 p，q を定め，その点を中心に N×N 近傍画素のテクスチャ情報を特徴ベクトルにする（図 3）．画像内のすべての輪郭線に対して上記で構成した特徴ベクトルを抽出し特徴空間に写像する．領域と同様に特徴のクラスタリングをおこなう．上記の方法であらかじめ求めたクラスタからユーザが描いた輪郭線の所属する集合を参照する事で輪郭線の自動選択を高速に行う事ができる．

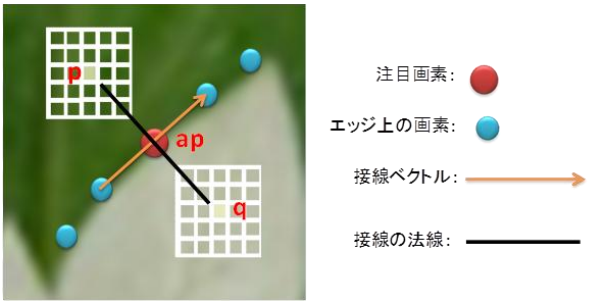


図 3　輪郭線となる画素の
テクスチャ特徴の抽出

2.2.　ユーザの描画スタイルを選択部分へ反映

得られた選択領域・輪郭線にユーザの描画スタイルを反映する方法について述べる．まず，領域に対して Image Analogies を応用した描画スタイル反映方法について述べる．

2.2.1.　選択した領域への描画スタイル反映

特定した領域に対して模様を描画する方法として Image Analogies[5]を利用する．Image Analogies は画像 A，画像 A へフィルタ処理をした画像 A'，画像 B を入力として与えると，入力画像 B に対して画像 A から画像 A'に施したフィルタ処理を行い，出力画像 B'を得る技術である．ユーザが画像の上に描いた模様はその位置での元の画像 A にフィルタ処理を行った画像 A'として扱うことができる．これにより鉛筆画，水彩画，油絵など様々な画風に対応でき，ユーザの描画スタイルを制限しないことから本研究には有用な技術である．

本研究では目的のために Image Analogies を拡張する．入力画像としてユーザが画像上に描画した位置の部分領域画像 A，描画した模様画像 A’，部分領域画像 A が所属しているクラスの領域画像 B を用意する（図 4）．これらの画像に対して Image Analogies を行う事で，ユーザの描画例と同じような描画スタイルを自動選択した領域に適用する事ができる．よって，書き加えたい領域に対して模様例と同じような模様を対話的に自動描画することができる．

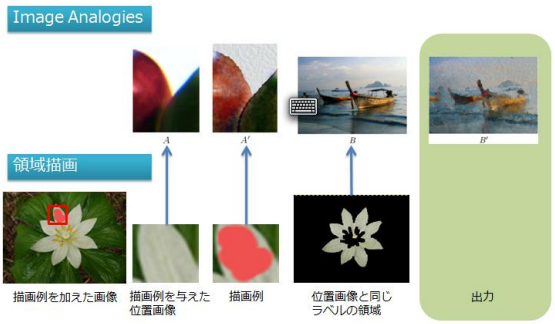


図 4　Image Analogies と領域描画との対応関係

2.2.2.　選択した輪郭線への描画スタイル反映

輪郭線にユーザの描画スタイルを反映するためには，ユーザの描いた描画例から描画スタイルの特徴を抽出して，自動選択した輪郭線の特徴に合わせて反映させなければならない．本研究の手法ではストロークの幾何形状や濃淡，線の太さを自動選択した輪郭線に反映させる方法を提案する．それはユーザの描いた描画例をテクスチャとして，選択した輪郭線の方向に合わせて貼り付けることで描画スタイルを反映する方法である．まずテクスチャの取得方法について述べた後，テクスチャを輪郭線の方向に沿って貼り付ける方法について述べる．

ユーザの描画例からテクスチャを取得するためにレイヤ構造を用いる．レイヤには，入力画像レイヤとテクスチャを保存するストロークレイヤを作成した．テクスチャを取得するにはユーザの描画筆跡座標からテクスチャの外接矩形領域をとり，その領域をスタイルレイヤから切り取る事でテクスチャを取得する事ができる．

次にテクスチャを輪郭線の方向に沿って貼り付けるには描画例を輪郭に沿って回転させる必要がある．回転角を調べるには，描画例を与えた位置の輪郭の接線ベクトルと貼り付け先の輪郭線の接線ベクトルを求めてから，それぞれのなす角度を計算すればよい．ユーザが描画例を描いた位置の輪郭線の接線ベクトルと貼り付け先の輪郭線の接線単位ベクトルは，その輪郭線の前後に分布する隣接画素の画像内の座標から求める事ができる．

輪郭線上の注目画素から前後にはなれた画素を選ぶときは，隣接画素より少し離れた画素を選んだほうが正確な接線ベクトルが得られやすい．以上の方法で，テクスチャを回転させれば輪郭線の方向に沿ったストロークを貼り付ける事ができる（図 5）．

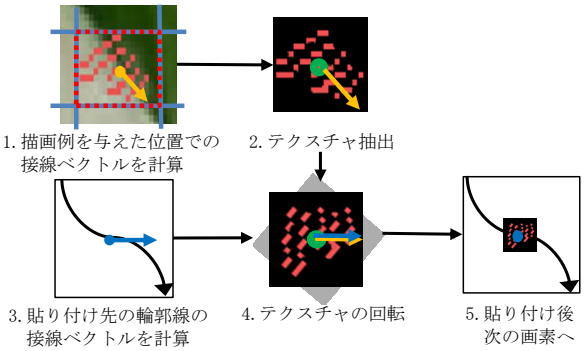


図 5　輪郭線へのストロークの反映

次の注目画素には，貼り付けたテクスチャ領域を抜けるまで輪郭線の画素を追跡していき，次の貼り付けテクスチャ領域が前のテクスチャ領域に重ならないような注目点を選択すればよい．

次に貼り付け先の輪郭線をテクスチャのストロークが不自然に飛び出ないようにする方法について述べる．テクスチャのストロークが飛び出る場合は，選択した輪郭線が短い場合や，選択した輪郭線が急激に曲がってしまう場合である．これらの問題にはテクスチャとなるストロークを飛び出す部分でトリミングする方法を用いる．輪郭線の曲率が高い部分はコーナー検出アルゴリズムによって検出する．本研究では Harris コーナー検出法[4]という方法を用いる．以上の方法で輪郭線のコーナー検出を行い，テクスチャを貼り付ける前に輪郭線上の画素でコーナーである部分を記録しておく．

次にテクスチャのストロークのトリミング方法について述べる．テクスチャのストロークをトリミングするには，貼り付ける輪郭線の前回貼り付けた点から終点まで，あるいはコーナーまでの距離を求める．距離を求めるには，付けた点から輪郭線の画素を追跡して行き，終点あるいはコーナーになるまでの画素数を調べればよい．トリミング画像は正方形としているので幅および高さは調べた画素数に設定する．テクスチャのトリミング画像は，元のテクスチャの中心にトリミング画像サイズと同じ近傍領域とする．以上の方法でテクスチャのトリミング画像を作成する．

3.　実装

写真から対話的にイラストを描くためのペイントソフトウェアを実装した．実装には Microsoft Visual C++と OpenCV を利用した．ソフトウェアの GUI は Microsoft Visual C++で実装し，内部処理は画像処理ライブラリである OpenCV で実装を行った．入力デバイスは，書きやすさや今後の筆圧情報の利用も考え，ペンタブレットを用いる．図 6 は，実装したソフトウェアの画面イメージである．

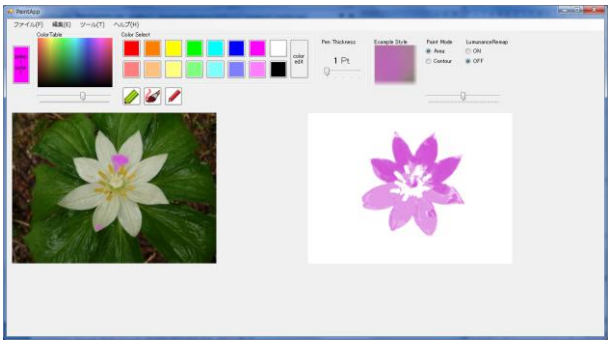


図 6　画面イメージ

3.1.　ストロークモデルの実装

ストロークのモデルとして鉛筆と筆の簡易モデルの 2 つを実装する．鉛筆簡易モデルは，筆圧によってその濃淡が変化し，ストロークが重なる部分は濃度が高くなるように設計した．筆簡易モデルは，筆圧によってその濃淡および太さが変化し，鉛筆簡易モデルと同様にストロークが重なる部分は濃度が高くなるようにした．ストロークを生成するには選択している色で塗りつぶされた円を濃淡や太さに合わせて選択座標上にプロットする事で生成していく方法を用いる．

図 7　筆ストロークモデル

4.　結果

実装したアプリケーションを用いてイラストの生成をおこなった．対象とする画像のサイズは，640×480pixel とした．まず，領域部分，輪郭部分にいくつか模様とストロークを与えた．

4.1.　領域処理の結果に対する考察

図 8 は白い花の画像の花びらの部分に手動でいくつかの模様例を書き込み，本研究の手法を用いて領域処理を行った結果である．出力された画像は花びらの部分に描画した描画例とほぼ同じ模様で描かれていることがわかる．また，描画例を書き加えてから結果はすぐにでき，対話的な描画が実現された．

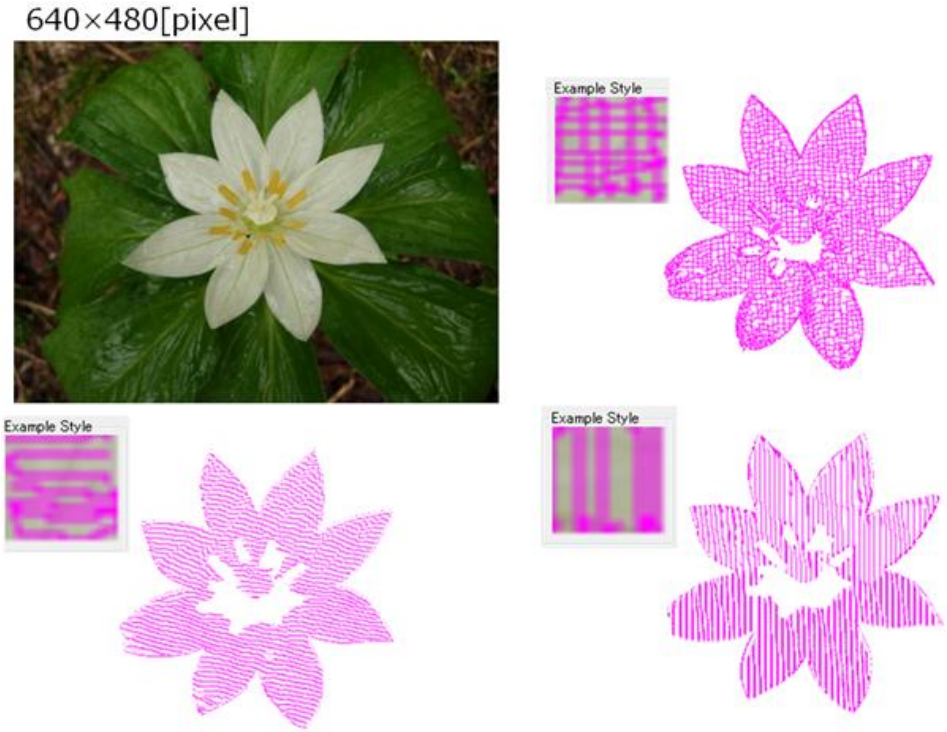


図 8　模様描画結果

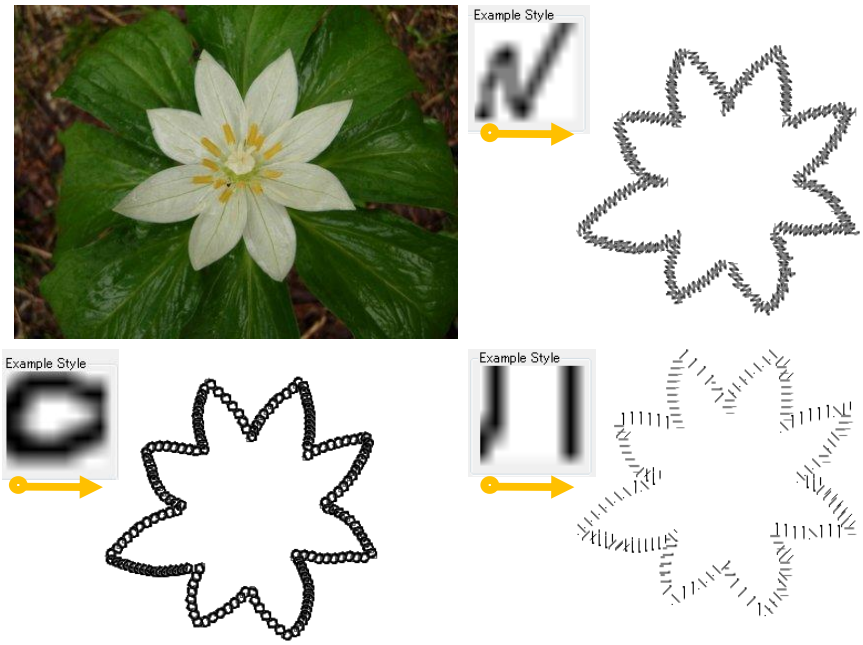
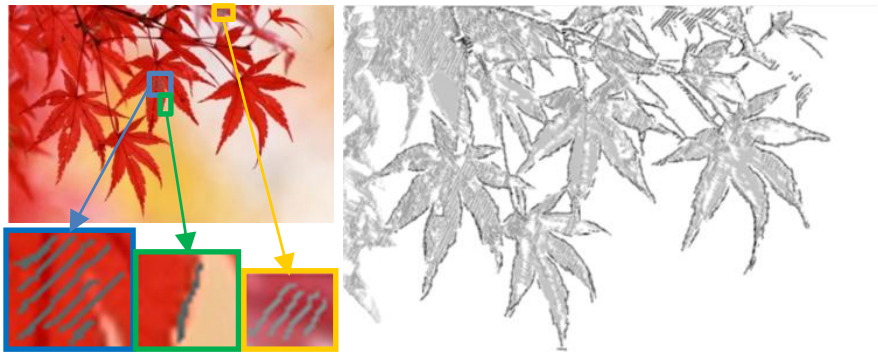


図 9　輪郭描画結果

4.2.　輪郭処理の結果に対する考察

図 9 は白い花の画像の花びらの輪郭部分に手動でいくつかの模様例を書き込み，本研究の手法を用いて輪郭処理を行った結果である．図中の黄色い矢印は例示した位置での輪郭線の接線ベクトルである．花びらの輪郭に沿って描画した模様例が回転しながら張り付けられている事がわかる．また，輪郭処理は処理対象とする画素数が少ないので，領域処理以上に高速に処理することができた．



(a1)ユーザの描画例　　(a2)イラスト生成結果

(a) 鉛筆モデルによるイラスト

(b1)ユーザの描画例　　(b2)イラスト生成結果

(b) 筆モデルによるイラスト

図 10　イラスト生成結果

4.3. イラスト生成の結果に対する考察

図 10 は紅葉に対して鉛筆モデルと筆モデルを用いて描いた様子である．写真上の枠内がユーザの描画した描画例である．同じ描画対象物がたくさん含まれているものに対して，一回の描画例だけで残りの同じ構造物を同じように自動で描く事ができた．また，イラスト生成の結果として鉛筆モデルと筆モデルを用いてイラストを描き分けることができた．

5.　　おわりに

本研究では画像からユーザの意図に合わせて対話的にイラストを生成する手法を提案した．この提案手法はユーザが画像上で書きたい部分の領域やその輪郭の一部分に描画例を与えることで，ユーザの描画したい領域や輪郭線を画像上から自動選択し，選択部分へユーザの描画スタイルを反映してイラストを対話的に生成することができた．これにより，従来のイラスト生成手法に比べ，作業時間の軽減，情報の取捨選択や描画スタイルの反映を制限することなくイラスト生成を行うことができた．

本研究では画像からユーザの意図に合わせて対話的にイラストを生成する手法を提案した．今後の課題として，多解像度テクスチャ特徴による学習精度の向上や，デフォルメなどの描画スタイルの学習が挙げられる．

参考文献

[1] Robert D. Kalnins, Lee Markosian, Barbara J. Meier, Michael A. Kowalski, Joseph C. Lee, Philip L. Davidson, Matthew Webb, John F. Hughes, and Adam Finkelstein, "WYSIWYG NPR: Drawing Strokes Directly on 3D Models," ACM Transactions on Graphics (Proc. SIGGRAPH), Vol.21, No.3, pp.755- 762, 2002.

[2] H. W. Kang, W. He, C. K. Chui,and U. K. Chakraborty, "Interactive Sketch Generation," The Visual Computer, Vol.21, No.8-10, pp.821-830, 2005.

[3] 岡田和典, "ミーンシフトの原理と応用," 情報処理学会研究報告 CVIM, Vol. 2008-CVIM-27, pp.401-414, 2008.

[4] ディジタル画像処理編集委員会, "ディジタル画像処理", CG-ARTS 協会, 2004.

[5] A. Hertzmann, C. Jacobs, N. Oliver, B. Curless, D. Salesin, "Image Analogies, " ACM SIGGRAPH, pp.327-340, 2001.