書道自動添削システムの開発

奈良工業高等専門学校　○伴藤　信一郎　◎西田茂生

要旨

現在，書道に取り組む人は世界中で増加している．しかし，海外で書道に取り組もうと思っても，書道を教える指導者がいなければ上達は困難である．既存の添削過程は書いた作品を指導者に送信し，添削を行ってもらう手法が一般的であるが，金銭的もしくは時間的余裕が足りず，添削をしてもらうことは難しい．そこで，本研究は時間・場所を選ばない，スマートフォンを利用したオフライン添削アプリの開発を行った．

1. 緒言

先行研究の結果，入力した作品をお手本と重ね合わせることで，システムの妥当性は示されている．そこで，本稿の目的は，この添削システムの実用性を向上させ，スマートフォン上で使えるように開発するものであるが，今回は実機ではなく Android エミュレータによるシミュレーション結果を本稿の結果とする．

2. 添削システム

2.1　開発環境

表 1 に開発条件について示す．この条件の下，Android 用の書道添削アプリを開発する．

表 1　開発条件

製作環境	Android Studio 3.5
対応バージョン	8.0 ~ 10.0
エミュレータ	Pixel 3 XL API 29
画像処理ソフトウェア	OpenCV 3.1

2.2　アルゴリズムと必要な処理

添削システムのアルゴリズムを図 1 に示す．入力した画像から半紙の輪郭を抽出し，それをもとに台形補正を行う．補正した入力画像はお手本画像と重ね合わせ，作品としての文字のバランスを添削する．本実験では，台形補正を行うまでに必要な各処理を Android 上にて製作し，アプリとして扱えるようにする．台形補正までに必要な処理を図 2 に示す．

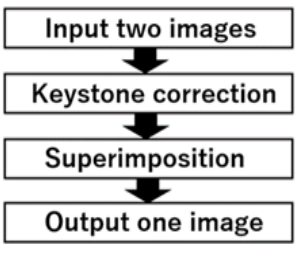


Fig.1 Schematic diagram of correction system

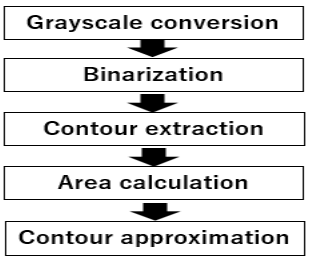


Fig.2 processing for keystone correction

台形補正とは元々四角形のものを斜めから撮影したときに，遠近により台形に見えるものを，元の四角形の形に補正することである．台形補正，及びそれに必要な輪郭抽出は，いずれも OpenCV で定義されるメソッドを使用して求められる．Cv2.findContours メソッドを使うことで、2 値化した画像から輪郭を抽出することができる．その際，輪郭は無数の点によって描写されるが，実際は直線の端点のみを保持すれば十分であり，抽出した輪郭を圧縮して凡長な点を削除すれば，四角形の頂点を検出することができる．抽出した頂点座標を画像が長方形になるように座標変換し，台形補正を行う．

3. 実験方法

3.1 システムの実用性向上実験

添削システムのプログラムの流れを図 1 に示す．先行研究では，各文字の位置と大きさを合わせ，文字の形を添削する方法と，全体を合わせ作品全体のバランスを添削する方法の二種類が示されていたが，本稿では後者を検証する．図 1 では，始めに撮影画像の中から半紙のみを切り出し，台形補正を行い，重ね合わせる．2 つの入力画像が同一作品ならば，ズレが見られないことが理想となる．

また，これまでの研究では検証を目的とし，それぞれの処理は別々のプログラムであったが，本実験では，これを一括化し，システムとしての実用性の向上を図る．また，違う角度から撮った同じ作品を重ね合わせることで，各処理の正確性が十分であることを確認する．

3.1 台形補正の精度向上実験

先行研究における図 1 の添削手法において，半紙のよれが原因と思われる台形補正のずれが見られた．そこで，本実験では，半紙のよれによる台形補正への影響を調べ，台形補正の精度向上を図る．

台形補正を行うにあたって，抽出した紙の輪郭をより単純な直線に近似する必要がある．OpenCV において，Cv2.apporoxPolyDP 関数を使うと，検出した形状の近似ができる．この関数における第２引数は epsilon と呼ばれ，実際の輪郭と近似輪郭の最大距離を表す，即ち，近似の精度を表すパラメータである．Epsilon による直線近似の例を図 3 に示す．(a) は切れ込みを入れた長方形であり，入力画像とする．この時，（b）は epsilon = 弧長の 10% と設定した出力結果であり，元の長方形を近似していることがわかる．(c) は epsilon = 弧長の 1% と設定した出力結果であり，より実際の輪郭に近い形で近似していることがわかる[1].

本実験では，紙によれをつける前と後の２枚の画像を用意し，epsilon を同じ値に設定し，図 1 の処理を行う．出力結果から，重ね合わせに誤差が生じていた場合，誤差を解消するまで epsilon を調節し，紙のよれによる誤差を定量化する．

(a) input image　(b) epsilon=0.1×Arc length　(c) epsilon=0.01×Arc length

Fig.3 Example of linear approximation [1]

4. 実験結果

4.1 システムの実用性向上実験結果

作成した一括添削システムの入力画像を図 4(a)，(b)，出力画像を図 4（c）に示す

(a)inputimage1 (b) input image2 (c) output image
Fig.4 Evaluation result

図 3 の出力結果から，(a)（b)は正しく重なっていることが確認できる．(a)，(b）は同一作品なので，ズレなく出力できた事は理想的な結果であると言える．これより，入力画像の位置の違いに関係なく，正しく添削できていることから，このシステムの実用性の向上を確認することができた．

4.2 台形補正の精度向上実験結果

入力画像として，よれをつける前の画像を図 5（a)，よれをつけた後の画像を図 5（b）に示す．また出力画像を図 5（c）に示す．

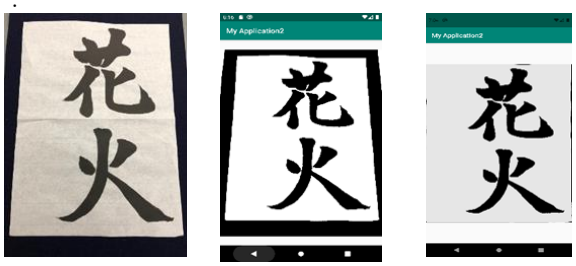
(a) input image1 (b) input image2 (c) output image
Fig.5 Evaluation result

図 5 の出力結果から，紙のよれに影響されず，正しく重なっている事がわかる．(a)，(b）は同一作品なので，ズレなく出力できた事は理想的な結果であると言える．これより，入力画像の紙のよれの違いに関係なく，正しく台形補正ができていることがわかる．よって，添削システムを開発するにあたって，半紙のよれや歪みは，考慮する必要がないことがわかった．

5. アプリ化

4 で示した実験結果を踏まえて，作成した添削アプリの各処理の出力画像を図 6(a), (b), (c)に示す

.



(a) input image (b) binaryimage (c)output image

Fig.6 Application result

開発結果から，入力した画像に対して，正しく台形補正ができていることがわかる．これより Android 上において，書道作品を添削用に台形補正するアプリを製作することができた．

6. 結言

今回のアプリ開発で，これまで製作した添削システムの一部をアプリとして動作させることができたが，重ね合わせの機能はもちろん，抽出した輪郭座標値を利用する過程の自動化など，アプリとして不十分な点が多い．

これらの機能を追加していくことで，アプリとしての完成度は高めることができると考えられる．

7. 参考文献

(1) Alexander Mordvintsev，Abid Rahman K， “OpenCV-Python チュートリアル” URL：http://labs.eecs.tottori-u.ac.jp/sd/Member/oyamada/OpenCV/html/py_tutorials/py_imgproc/py_contours/py_contour_features/py_contour_features.html

(2)小枝正直，上田悦子，中村恭之”Opencv による画像処理入門” 改訂第 2 版　講談社(2017)